



applied sciences

Advances in Industrial Robotics and Intelligent Systems

Edited by

António Paulo Moreira, Pedro Neto and Félix Vidal

Printed Edition of the Special Issue Published in *Applied Sciences*

Advances in Industrial Robotics and Intelligent Systems

Advances in Industrial Robotics and Intelligent Systems

Editors

António Paulo Moreira

Pedro Neto

Félix Vidal

MDPI • Basel • Beijing • Wuhan • Barcelona • Belgrade • Manchester • Tokyo • Cluj • Tianjin



Editors

António Paulo Moreira
University of Porto and INESC TEC—
Technology and Science
Portugal

Pedro Neto
University of Coimbra
Portugal

Félix Vidal
Asociación de Investigación
Metalúrgica del Noroeste
Spain

Editorial Office

MDPI
St. Alban-Anlage 66
4052 Basel, Switzerland

This is a reprint of articles from the Special Issue published online in the open access journal *Applied Sciences* (ISSN 2076-3417) (available at: <https://www.mdpi.com/journal/applsci/special-issues/Control.Robotics>).

For citation purposes, cite each article independently as indicated on the article page online and as indicated below:

LastName, A.A.; LastName, B.B.; LastName, C.C. Article Title. *Journal Name* **Year**, *Volume Number*, Page Range.

ISBN 978-3-0365-6554-5 (Hbk)

ISBN 978-3-0365-6555-2 (PDF)

Cover image courtesy of Félix Vidal

© 2023 by the authors. Articles in this book are Open Access and distributed under the Creative Commons Attribution (CC BY) license, which allows users to download, copy and build upon published articles, as long as the author and publisher are properly credited, which ensures maximum dissemination and a wider impact of our publications.

The book as a whole is distributed by MDPI under the terms and conditions of the Creative Commons license CC BY-NC-ND.

Contents

About the Editors	vii
Antônio Paulo Moreira, Pedro Neto and Félix Vidal Special Issue on Advances in Industrial Robotics and Intelligent Systems Reprinted from: <i>Appl. Sci.</i> 2023 , <i>13</i> , 1352, doi:10.3390/app13031352	1
Josuet Leoro and Tesheng Hsiao Motion Planning of Nonholonomic Mobile Manipulators with Manipulability Maximization Considering Joints Physical Constraints and Self-Collision Avoidance Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 6509, doi:10.3390/app11146509	3
Heiko Engemann, Patrick Cönen, Harshal Dawar, Shengzhi Du and Stephan Kallweit A Robot-Assisted Large-Scale Inspection of Wind Turbine Blades in Manufacturing Using an Autonomous Mobile Manipulator Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 9271, doi:10.3390/app11199271	29
Tiago Ribeiro, Fernando Gonçalves, Inês S. Garcia, Gil Lopes and António F. Ribeiro CHARMIE: A Collaborative Healthcare and Home Service and Assistant Robot for Elderly Care Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 7248, doi:10.3390/app11167248	51
Jun Wang, Mingquan Yang, Fei Liang, Kangrui Feng, Kai Zhang and Quan Wang An Algorithm for Painting Large Objects Based on a Nine-Axis UR5 Robotic Manipulator Reprinted from: <i>Appl. Sci.</i> 2022 , <i>12</i> , 7219, doi:10.3390/app12147219	81
Kui Xiao, Wentao Yu, Weirong Liu, Feng Qu and Zhenyan Ma High-Precision SLAM Based on the Tight Coupling of Dual Lidar Inertial Odometry for Multi-Scene Applications Reprinted from: <i>Appl. Sci.</i> 2022 , <i>12</i> , 939, doi:10.3390/app12030939	107
Jin-Gu Kang, Yong-Sik Choi and Jin-Woo Jung A Method of Enhancing Rapidly-Exploring Random Tree Robot Path Planning Using Midpoint Interpolation Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 8483, doi:10.3390/app11188483	125
Mateusz Fiedeń and Jacek Bałchanowski A Mobile Robot with Omnidirectional Tracks—Design and Experimental Research Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 11778, doi:10.3390/app112411778	143
Ang Zhang, Keisuke Koyama, Weiwei Wan and Kensuke Harada Manipulation Planning for Large Objects through Pivoting, Tumbling, and Regrasping Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 9103, doi:10.3390/app11199103	167
Sergio Hernandez-Mendez, Elvia R. Palacios-Hernandez, Antonio Marin-Hernandez, Ericka Janet Rechy-Ramirez and Hector Vazquez-Leal Design and Implementation of Composed Position/Force Controllers for Object Manipulation Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 9827, doi:10.3390/app11219827	187
Maurício Correa, Daniel Cárdenas, Diego Carvajal and Javier Ruiz-del-Solar Haptic Teleoperation of Impact Hammers in Underground Mining Reprinted from: <i>Appl. Sci.</i> 2022 , <i>12</i> , 1428, doi:10.3390/app12031428	205

Kailin Wen, Jie Chu, Yu Chen, Dong Liang, Chengkai Zhang and Jueping Cai A Multiorder Attentional Spatial Interactive Convolutional Neural Network (MoAS-CNN) for Low-Resolution Haptic Recognition Reprinted from: <i>Appl. Sci.</i> 2022 , <i>12</i> , 12715, doi:10.3390/app122412715	225
Tomáš Kot, Zdenko Bobovský, Aleš Vysocký, Václav Kryš, Jakub Šafařík and Roman Ružarovský Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 5398, doi:10.3390/app11125398	239
Piotr Cybulski, Zbigniew Zieliński Design and Verification of Multi- Agent Systems with the Use of Bigraphs Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 8291, doi:10.3390/app11188291	259
Riccardo Karim Khamaisi, Elisa Prati, Margherita Peruzzini, Roberto Raffaeli and Marcello Pellicciari UX in AR-Supported Industrial Human–Robot Collaborative Tasks: A Systematic Review Reprinted from: <i>Appl. Sci.</i> 2021 , <i>11</i> , 10448, doi:10.3390/app112110448	295

About the Editors

António Paulo Moreira

António Paulo Moreira graduated with a degree in Electrical Engineering from the University of Porto in 1986, M.Sc. degree in Electrical Engineering—Systems in 1991, and Ph.D. degree in Electrical Engineering in 1998. Associated Professor with Tenure at the Electrical Engineering Department of the University of Porto. Coordinator of the Centre for Robotic and Intelligent Systems Centre of INESC TEC—Institute for Systems and Computer Engineering, Technology and Science. Participation in 25 scientific projects; coordinator of 7 projects. The knowledge developed in these research projects generated 40 development contracts and technology transfer to companies, as coordinator of 18 projects. Has 308 scientific publications at present, 202 of which are in international conferences, books and thesis, and 78 are in international journals with a referee.

Pedro Neto

Pedro Neto, Ph.D., is an Associate Professor at the Mechanical Engineering Department and coordinator of the Collaborative Robotics Laboratory at the University of Coimbra. He is a two-time recipient of the Impact and International Publications Award at the Faculty of Science and Technology, received more than a dozen awards, and is in the World's Top 2% Scientists Rank from Elsevier. Pedro Neto served as vice president of the Portuguese Robotics Society, is a member of the IEEE Committee on Factory Automation, associate editor in refereed journals, and scientific committee member of flagship conferences. His current research interests include human–robot interaction and collaboration, machine learning, soft robots, and advanced robot applications in the manufacturing domain. He co-authored more than 100 papers and collaborated with more than 100 companies in projects and consulting, some of them granted by the European Commission in the prestigious HORIZON framework.

Félix Vidal

Félix Vidal has an M.Sc. in Industrial Engineering (Automation and electronics) and International Welding Engineer (IWE). At present, he is working in AIMEN as head of Smart Systems and Smart Manufacturing, and project coordinator of several European R&D projects related with digital transformation. He has more than 20 years' experience as a researcher in European and National projects dealing with the digitisation of the EU industry. He has authored papers in major conferences and journals related to smart manufacturing in recent years. He is a member of EFFRA and euRobotics and reviewer for different papers in international manufacturing and mechatronics conferences and journals.

Editorial

Special Issue on Advances in Industrial Robotics and Intelligent Systems

António Paulo Moreira ¹, Pedro Neto ^{2,*} and Félix Vidal ³

¹ Faculty of Engineering, University of Porto and INESC TEC—INESC Technology and Science, 4099-002 Porto, Portugal

² Department of Mechanical Engineering, University of Coimbra, CEMPRE, 3030-788 Coimbra, Portugal

³ Asociación de Investigación Metalúrgica del Noroeste, 36410 Porriño, Spain

* Correspondence: pedro.neto@dem.uc.pt; Tel.: +351-239790767

Robotics and intelligent systems are key technologies to promote efficient and innovative applications in the most diverse domains (industry, healthcare, agriculture, construction, mobility, etc.), performing and supporting activities that are not suitable to be performed by humans. Such activities are frequently time-consuming, repetitive tasks with low added value, physically demanding, and/or dangerous. Nevertheless, robotics and intelligent systems face several scientific and technological challenges related to their integration and interoperability with other systems, safety, flexibility, reconfigurability and autonomy. These challenges are especially relevant when robots operate in real unstructured environments and share the workspace with humans and other equipment.

This Special Issue collects research achievements, ideas, and applications of advanced intelligent robotic systems, covering diverse technologies and application domains. Generally, the contributions cover optimal path planning strategies and innovative designs for mobile manipulators, the integration of robotic and intelligent systems, grasping, manipulation, teleoperation, haptics, user experience approaches for collaborative robots, and multi-agent systems.

In the last few years, we addressed the emergence of mobile manipulators as versatile robotic machines, combining the best abilities of mobile robots and robotic manipulators. An interesting study reports a mobile manipulator unified framework for motion planning considering joint limits, joint velocity limits, self-collisions, and singularities [1]. A novel path planning strategy for the autonomous navigation of a mobile manipulator operating in inspection processes is proposed in [2]. A mobile manipulator, which operates as a healthcare and domestic assistant, demonstrated its capability to perform generic service tasks in non-standardized healthcare and domestic environments [3]. In [4], a robot-based framework is proposed to automatically plan trajectories designed for painting large objects, e.g., a car roof. A Simultaneous Localization and Mapping (SLAM) framework used to solve the problem of the poor positioning accuracy of mobile robots, by fusing horizontal and vertical lidar data with Inertial Measurement Unit (IMU) data, eliminates the motion distortion of the dual-lidar odometry [5]. A robot path planning method using midpoint interpolation increased the efficiency of optimization by minimizing the planning time [6]. An interesting study presents a design of a mobile robot with omnidirectional tracks, combining the advantages of a typical track drive with the omnidirectional Mecanum wheels [7].

A dual-arm robotic manipulator demonstrated the ability to manipulate large and heavy objects avoiding obstacles by using a hierarchical manipulation planner [8]. A position/force controller used to grasp objects through a robotic manipulator, find the position of the object to be grasped accurately, and apply the appropriate force to each finger to handle the object properly is proposed in [9]. In [10], a haptic teleoperation of impact hammers in mining operations is proposed, where the 3D model of the environment

Citation: Moreira, A.P.; Neto, P.; Vidal, F. Special Issue on Advances in Industrial Robotics and Intelligent Systems. *Appl. Sci.* **2023**, *13*, 1352. <https://doi.org/10.3390/app13031352>

Received: 13 January 2023
Accepted: 17 January 2023
Published: 19 January 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

is used to estimate repulsion forces that are transferred to the operator via haptics so that the hammer does not collide with the structures of the mine. A multi-order attentional spatial interactive convolutional neural network for haptic recognition is detailed in [11]. It was validated on the recognition of letter shape (A–Z) with complex contours from a haptic acquisition platform based on three-scale pressure arrays. In [12], the optimization of robot placement was studied to reduce the overall robot joint wear, where a proper robot base placement results in an overall reduction in the wear of the joints of a robotic arm. An algorithm for verifying the correctness of multi-agent systems modeled as tracking bigraphical reactive systems and checking whether a behavior policy for the agents meets non-functional requirements is presented in [13]. A review of the recent literature on augmented reality-supported collaborative robotics from a human-centered perspective to solve issues related to operators' needs is proposed in [14]. The study elaborates on a structured framework driven by User eXperience approaches to design augmented reality interfaces.

Funding: This research received no external funding.

Acknowledgments: Thanks to all the authors and peer reviewers for their valuable contributions to the Special Issue 'Advances in Industrial Robotics and Intelligent Systems'.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Leoro, J.; Hsiao, T. Motion Planning of Nonholonomic Mobile Manipulators with Manipulability Maximization Considering Joints Physical Constraints and Self-Collision Avoidance. *Appl. Sci.* **2021**, *11*, 6509. [[CrossRef](#)]
2. Engemann, H.; Cönen, P.; Dawar, H.; Du, S.; Kallweit, S. A Robot-Assisted Large-Scale Inspection of Wind Turbine Blades in Manufacturing Using an Autonomous Mobile Manipulator. *Appl. Sci.* **2021**, *11*, 9271. [[CrossRef](#)]
3. Ribeiro, T.; Gonçalves, F.; Garcia, I.; Lopes, G.; Ribeiro, A. CHARMIE: A Collaborative Healthcare and Home Service and Assistant Robot for Elderly Care. *Appl. Sci.* **2021**, *11*, 7248. [[CrossRef](#)]
4. Wang, J.; Yang, M.; Liang, F.; Feng, K.; Zhang, K.; Wang, Q. An Algorithm for Painting Large Objects Based on a Nine-Axis UR5 Robotic Manipulator. *Appl. Sci.* **2022**, *12*, 7219. [[CrossRef](#)]
5. Xiao, K.; Yu, W.; Liu, W.; Qu, F.; Ma, Z. High-Precision SLAM Based on the Tight Coupling of Dual Lidar Inertial Odometry for Multi-Scene Applications. *Appl. Sci.* **2022**, *12*, 939. [[CrossRef](#)]
6. Kang, J.; Choi, Y.; Jung, J. A Method of Enhancing Rapidly-Exploring Random Tree Robot Path Planning Using Midpoint Interpolation. *Appl. Sci.* **2021**, *11*, 8483. [[CrossRef](#)]
7. Fiedor, M.; Bałchanowski, J. A Mobile Robot with Omnidirectional Tracks—Design and Experimental Research. *Appl. Sci.* **2021**, *11*, 11778. [[CrossRef](#)]
8. Zhang, A.; Koyama, K.; Wan, W.; Harada, K. Manipulation Planning for Large Objects through Pivoting, Tumbling, and Regrasping. *Appl. Sci.* **2021**, *11*, 9103. [[CrossRef](#)]
9. Hernandez-Mendez, S.; Palacios-Hernandez, E.; Marin-Hernandez, A.; Rechy-Ramirez, E.; Vazquez-Leal, H. Design and Implementation of Composed Position/Force Controllers for Object Manipulation. *Appl. Sci.* **2021**, *11*, 9827. [[CrossRef](#)]
10. Correa, M.; Cárdenas, D.; Carvajal, D.; Ruiz-del-Solar, J. Haptic Teleoperation of Impact Hammers in Underground Mining. *Appl. Sci.* **2022**, *12*, 1428. [[CrossRef](#)]
11. Wen, K.; Chu, J.; Chen, Y.; Liang, D.; Zhang, C.; Cai, J. A Multiorder Attentional Spatial Interactive Convolutional Neural Network (MoAS-CNN) for Low-Resolution Haptic Recognition. *Appl. Sci.* **2022**, *12*, 12715. [[CrossRef](#)]
12. Kot, T.; Bobovský, Z.; Vysocký, A.; Kryš, V.; Šafařík, J.; Ružarovský, R. Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell. *Appl. Sci.* **2021**, *11*, 5398. [[CrossRef](#)]
13. Cybulski, P.; Zieliński, Z. Design and Verification of Multi-Agent Systems with the Use of Bigraphs. *Appl. Sci.* **2021**, *11*, 8291. [[CrossRef](#)]
14. Khamaisi, R.; Prati, E.; Peruzzini, M.; Raffaeli, R.; Pellicciari, M. UX in AR-Supported Industrial Human–Robot Collaborative Tasks: A Systematic Review. *Appl. Sci.* **2021**, *11*, 10448. [[CrossRef](#)]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.

Article

Motion Planning of Nonholonomic Mobile Manipulators with Manipulability Maximization Considering Joints Physical Constraints and Self-Collision Avoidance

Josuet Leoro [†] and Tesheng Hsiao ^{*,†}

Department of Electrical and Computer Engineering, National Yang Ming Chiao Tung University, Hsinchu 30010, Taiwan; jl2005ec.04g@g2.nctu.edu.tw

* Correspondence: tshsiao@cn.nctu.edu.tw

† These authors contributed equally to this work.

Abstract: The motion of nonholonomic mobile manipulators (NMMs) is inherently constrained by joint limits, joint velocity limits, self-collisions and singularities. Most motion planning algorithms consider some of the aforementioned constraints, however, a unified framework to deal with all of them is lacking. This paper proposes a motion planning solution for the kinematic trajectory tracking of redundant NMMs that include all the constraints needed for practical implementation, which improves the manipulability of both the entire system and the manipulator to prevent singularities. Experiments using a 10-DOF NMM demonstrate the good performance of the proposed method for executing 6-DOF trajectories while satisfying all the constraints and simultaneously maximizing manipulability.

Keywords: motion planning; trajectory tracking; mobile manipulator; joint constraints; self-collision avoidance; manipulability

Citation: Leoro, J.; Hsiao, T. Motion Planning of Nonholonomic Mobile Manipulators with Manipulability Maximization Considering Joints Physical Constraints and Self-Collision Avoidance. *Appl. Sci.* **2021**, *11*, 6509. <https://doi.org/10.3390/app11146509>

Academic Editors: Pedro Neto, António Paulo Moreira and Félix Vidal

Received: 27 June 2021
Accepted: 14 July 2021
Published: 15 July 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

A mobile manipulator is a robotic system that consists of a standard robot manipulator mounted on a mobile platform. This system integrates the dexterity provided by the manipulator with the extended workspace provided by the platform. Additionally, the combination of both subsystems usually introduces kinematic redundancy, which increases flexibility and dexterity. Therefore, mobile manipulators are suitable to perform delicate tasks over a large space, such as welding large parts or painting large, curvy surfaces.

The practical applications of robotic systems commonly define tasks by either a point-to-point movement or a continuous path of the end-effector in task space (also known as operational space). This paper aims to solve the latter, and in particular, focuses on the task space trajectory tracking problem. A trajectory is a path on which a timing law is specified, for instance in terms of velocities and/or accelerations [1]. In other words, not only is the end-effector's pose profile defined, but so is its velocity profile. To accomplish this, a motion planning algorithm that exploits the capabilities of both the manipulator and the mobile platform and that coordinates their movements is required. The redundancy of mobile manipulators can be used to perform additional subtasks or satisfy system constraints. These constraints include joint limits, joint velocity limits, joint velocity boundary constraints (i.e., the constraints on the initial and final joint velocities), and self-collision avoidance. Furthermore, for task space trajectory tracking to be achievable, it is important that the system is kept away from singularities. All these requirements make the motion planning for trajectory tracking a challenging problem.

There exist recent efforts in solving the motion planning of mobile manipulators [2,3]. Liao et al. [2] presented an optimization-based solution that not only handles constraints at the position level, but can also set a target joint configuration for the manipulator at

the end of the trajectory. A heuristic approach was proposed by Santos et al. [3] that is simple to implement and can accomplish additional constraints such as joint limits and manipulability improvement. Nonetheless, these methods do not deal with constraints at the velocity level and are only applicable to mobile manipulators with omnidirectional platforms. The present work focuses on the motion planning of mobile manipulators with nonholonomic mobile platforms. The movement of this type of platform is constrained by the rolling without slipping condition, which inhibits the platform from instantly moving in any arbitrary direction [4].

The motion planning of NMMs has also been studied in the literature using different approaches [5–10]. De Luca et al. [7] implemented the reduced gradient method for NMMs. This method finds a permutation matrix that helps reduce the velocity input to the subspace of commands that satisfy the given task. The remaining velocity inputs are used to maximize an objective function. Even though this method is computationally more efficient than the projected gradient approach, it is difficult to find such a permutation matrix for highly redundant robots since the Jacobian must be pre-analyzed by hand. Jia et al. [9] studied an adaptive motion distribution and coordinated control between the manipulator and the mobile platform to minimize the end-effector's positioning error.

In task space trajectory tracking, it is important that the motion planning algorithm moves the system away from singularities. This is because the system is in kinematic singularity, and the dexterity of the structure is reduced because the robot's end-effector cannot be moved in a certain direction. In addition, when the system is in the neighborhood of a singularity, small velocities in the task space may cause large velocities in the joint space [1], which is unacceptable because this would result in the failure of the trajectory tracking task, and even damage the mobile manipulator. For these reasons, the manipulability maximization was included in multiple motion planning methods for NMMs [5,6,8–10]. Bayle et al. [5,6] maximized the system's manipulability using the projected gradient method. Huang et al. [8] studied the coordination of the platform and the manipulator, simultaneously considering the mobile platform stability and the manipulator's manipulability. Although these techniques can successfully follow the end-effector path while considering additional criteria, none of these consider joint constraints.

Furthermore, the solution of the task space trajectory tracking problem must not only consider joint limits but also joint velocity limits. This is because if a joint reaches its velocity limit, the end-effector might not be able to comply with the desired velocity profile. To the authors' knowledge, the literature of motion planning for trajectory tracking in task space with NMMs that includes joint constraints is limited. Zhang et al. [10] proposed formulating the motion planning problem as an optimization problem where the manipulability is maximized and the joint limits and joint velocity limits are included as constraints. This optimization problem is reformulated as a quadratic programming problem and converted into a linear variational inequality problem, that can be solved by different numerical methods. This approach is effective but does not consider boundary constraints for joint velocities. These constraints are also relevant because, for a given task, zero joint velocities are expected at the start and end of the trajectory. Additionally, NMMs are not only subject to physical limits but also to self-collisions, especially between the manipulator and the mobile platform, which are not included in their work.

An important remark is that maximizing the manipulability of the whole system might result in poor manipulability for the arm alone, even though the manipulability for the whole system is preserved or improved [6,10]. Additionally, it is important that both the robot arm subsystem and the whole mobile manipulator system maintain a certain level of manipulability once a coordinated task is completed. After completing an arbitrary path, a subsequent task might only require the arm to manipulate an object. If the arm is in a configuration with low manipulability, executing such a task might not be feasible. For these reasons, in this work, we propose a new manipulability measure for mobile manipulators that, when maximized, and as demonstrated in our experiments,

intrinsically improves the manipulability of the robot arm as well as the manipulability of the whole system.

The solution to the motion planning of NMMs for trajectory tracking presented in this work includes joint constraints (range and maximum velocities), self-collision avoidance and manipulation capability preservation. Figure 1 summarizes how all these constraints are included in our solution. Both the particular and homogeneous solutions of our proposed scheme are weighted to avoid joint limits and self-collisions while the trajectory is tracked. The homogeneous solution is used to maximize the manipulability by exploiting the redundancy of the system. To satisfy joint velocity boundary constraints, the step size for searching the maximum manipulability is varied. The joint velocity limits are satisfied by restraining the maximum step size based on the allowable self-motion.

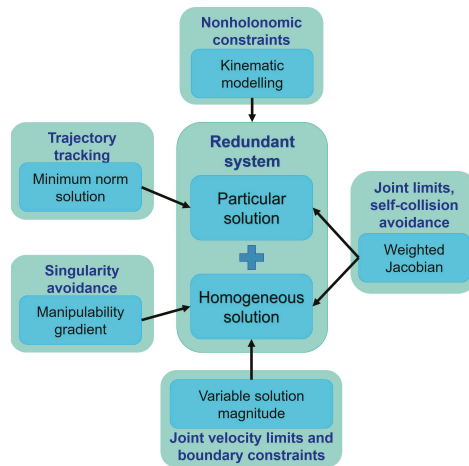


Figure 1. Summary of the proposed motion planning approach.

Experimental results demonstrate that our method can successfully solve the motion planning problem of NMMs under all the mentioned constraints. This work focuses on task space trajectory tracking at kinematic level. In other words, the outputs of the motion planning algorithm are the joint positions and velocities that will be fed to a joint space dynamic controller for motion control. Then, the motion controller is responsible for suppressing the model uncertainties and external disturbances to guarantee that the actual joint positions follow the ones output by the motion planning algorithm. In the experiments shown in this paper, we use the built-in motion controller of the commercial manipulator and leave the design of our own motion controller for future research.

The contributions of this work are detailed as follows:

- A motion planning solution for NMMs that allows to include joint physical constraints and the execution of a secondary task is presented.
- Multiple constraints required for the practical implementation of task space trajectory tracking are included in a unified solution. These constraints include joint limits, joint velocity limits, joint velocity boundary constraints and self-collision avoidance.
- A new manipulability measure for mobile manipulators is presented. It is demonstrated that the maximization of this measure simultaneously improves the manipulabilities of the whole system and the robot arm.

This paper is organized as follows. Section 2 describes the kinematic modeling of NMMs. Section 3 describes the motion planning problem for trajectory tracking and presents the proposed solution. In Section 4, the concepts that are employed to satisfy each of the mentioned constraints are described and included in the motion planning algorithm.

Experimental results using 6-DOF tasks are presented in Section 5 to validate the efficacy of our approach. Finally, Section 6 concludes the paper.

2. Kinematic Modeling

The kinematic modeling described here follows the procedure of De Luca et al. [7] and Bayle et al. [6]. For a general procedure of kinematic modeling of wheeled mobile manipulators, please refer to Bayle et al. [5].

Let $r \in \mathbb{R}^m$ be the end-effector's position and/or pose in the task space. The configuration vector q , also known as the generalized coordinates of the mobile manipulator, is given by the combination of the platform configuration vector q_p and the robot arm configuration vector q_a . Figure 2 illustrates these configuration vectors. A frame $x'y'$ is attached to the mobile platform at the center of the wheels' axle (x_p, y_p) , with respect to the world reference frame xy , with its x' axis pointing in the forward direction and the y' axis pointing in the direction parallel to the wheels' axle. The angle between the x axis of the world reference frame and x' attached to the platform is denoted as θ_p . Then, the platform configuration is given by $q_p = [x_p \ y_p \ \theta_p]^T \in \mathbb{R}^3$. The robot arm configuration is given by the position of its joints as $q_a = [q_{a1} \ q_{a2} \ \dots \ q_{an}]^T \in \mathbb{R}^{n_a}$. Finally, the configuration vector of the mobile manipulator is $q = [q_p^T \ q_a^T]^T \in \mathbb{R}^n$ with $n = 3 + n_a$. The end-effector's position/pose is a function of the configuration vector defined by the kinematic map:

$$r = f(q_p, q_a), \tag{1}$$

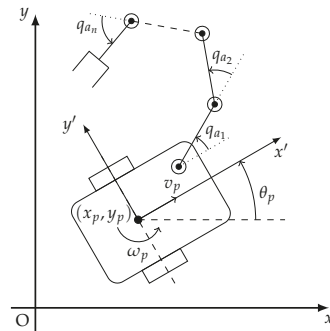


Figure 2. Nonholonomic mobile manipulator schematic.

The wheeled platform movement is constrained under the rolling without a slipping assumption on both wheels, which can be expressed as the following nonholonomic constraint:

$$\dot{q}_p = G(q_p)u_p, \tag{2}$$

where $u_p = [v_p \ \omega_p]^T \in \mathbb{R}^2$ is the velocity input of the platform, consisting of the linear and angular velocities of the platform, which are known as pseudo velocities or quasi-velocities. The columns of matrix $G(q_p)$ span the admissible velocity space for a given platform configuration. The matrix $G(q_p)$ for a differential-drive platform is defined as

$$G(q_p) = \begin{bmatrix} \cos \theta_p & 0 \\ \sin \theta_p & 0 \\ 0 & 1 \end{bmatrix},$$

On the other hand, the robotic arm kinematics at the velocity level is not constrained for any configuration, and therefore, the generalized velocities of the arm are equal to the velocity inputs of the arm:

$$\dot{q}_a = u_a. \tag{3}$$

The velocity input vector for the entire NMM can be constructed as $u = [u_p^T \quad u_a^T]^T \in \mathbb{R}^{\delta_m}$, with $\delta_m = 2 + n_a$. The dimension δ_m is called the mobility degree of the mobile manipulator and indicates the space dimension of the admissible generalized velocities [6]. Using (2) and (3), the map from the velocity input vector u to the generalized velocities $\dot{q} = [\dot{q}_p^T \quad \dot{q}_a^T]^T$ can be written as

$$\dot{q} = S(q)u, \tag{4}$$

with:

$$S(q) = \begin{bmatrix} G(q_p) & 0 \\ 0 & I_{n_a} \end{bmatrix},$$

where $G(q_p)$ maps the platform’s velocity input vector $u_p = [v_p \quad \omega_p]^T$ to the platform’s generalized velocities $\dot{q}_p = [\dot{x}_p \quad \dot{y}_p \quad \dot{\theta}_p]^T$, and matrix I_{n_a} is an identity matrix of size n_a that sets the arm’s generalized velocities equal to its input velocities $\dot{q}_a = u_a$.

The differential kinematics of the NMM is obtained by differentiating the relation (1) with respect to time:

$$\begin{aligned} \dot{r} &= [J_p(q_p) \quad J_a(q_a)] \begin{bmatrix} \dot{q}_p \\ \dot{q}_a \end{bmatrix} \\ &= [J_p(q_p) \quad J_a(q_a)] S(q)u \\ &= J(q)S(q)u = \bar{J}u, \end{aligned} \tag{5}$$

where the matrices $J_p \in \mathbb{R}^{m \times 3}$ and $J_a \in \mathbb{R}^{m \times n_a}$ are the Jacobians of the platform and the arm, respectively. The matrix $J \in \mathbb{R}^{m \times n}$ is the Jacobian of the mobile manipulator, and $\bar{J} \in \mathbb{R}^{m \times \delta_m}$ is a reduced version of the Jacobian in which the admissible velocities of the platform under the nonholonomic constraint have been included. Equation (5) follows the same form as the differential kinematics of standard manipulators; therefore, the well-known methodologies for the motion planning of redundant manipulators can be extended to NMMs in terms of \bar{J} , including concepts for joint limits avoidance, self-collisions avoidance and manipulability maximization.

3. Motion Planning Method

The motion planning problem for task space trajectory tracking consists of finding the input velocities u , given the desired end-effector’s position/pose $r_d(t) \in \mathbb{R}^m$ for $t \in [t_0, t_f]$, such that $r(t)$ follows $r_d(t)$. If the task space velocity of the end-effector is set as

$$\dot{r} = \dot{r}_d + K(r_d - r), \tag{6}$$

where $K \in \mathbb{R}^{m \times m}$ is a positive definite matrix, then the convergence of $r(t)$ to $r_d(t)$ is guaranteed. Consequently, the motion planning problem is turned into solving the input velocities u from the underdetermined linear equations (5) where \dot{r} is set according to (6).

During the execution of the trajectory, the movement of the joints that get closer to a position constraint (joint limit or self-collision) must be penalized (slowed down). This can be achieved by using a weighting matrix. In this work, we define the weighted input velocity and the reduced weighted Jacobian as follows:

$$\bar{J}_W = \bar{J}W^{\frac{1}{2}}(q) \quad \text{and} \quad u = W^{\frac{1}{2}}(q)u_W, \tag{7}$$

where $W(q) \in \mathbb{R}^{\delta_m \times \delta_m}$ is a configuration-dependent positive semidefinite matrix. It is constructed so that its elements penalize the motion of the joints based upon the system constraints. We choose $W(q)$ as a diagonal matrix in this paper; furthermore, we drop the notational dependency of W on q to simplify the expression in the subsequent derivation.

Using (7), the system (5) can be rewritten as

$$\dot{r} = \bar{J}_W u_W. \tag{8}$$

Because of the kinematic redundancy, i.e., $m < \delta_m$, infinite solutions to \mathbf{u}_W exist. The solution to (8) can be decomposed as a sum of a particular solution and a non-zero homogeneous solution. The particular solution satisfies the end-effector’s task, while the homogeneous solution changes the manipulator configuration without changing the end-effector’s position/pose.

One way to solve the redundant system (8) is to formulate the problem as a constrained optimization problem [1], where the goal is to minimize a cost function and satisfy the constraint (8). We propose the minimization of the quadratic cost function:

$$g(\mathbf{u}_W) = \frac{1}{2}(\mathbf{u}_W - W^{\frac{1}{2}}\mathbf{u}_0)^T(\mathbf{u}_W - W^{\frac{1}{2}}\mathbf{u}_0);$$

this choice is aimed to find a vector of velocities \mathbf{u}_W that is as close as possible to $W^{\frac{1}{2}}\mathbf{u}_0$, where $\mathbf{u}_0 \in \mathbb{R}^{\delta_m}$ is a velocity vector that is used to satisfy an additional task such as maximizing the manipulability. The choice of \mathbf{u}_0 will be presented shortly. Notice that the physical constraints are also imposed to \mathbf{u}_0 by weighting it similarly to how J_W is obtained in (7). By using the method of Lagrange multipliers to minimize $g(\mathbf{u}_W)$ with the equality constraint (8), the following solution is obtained:

$$\mathbf{u}_W^* = \bar{J}_W^{\dagger}\dot{\mathbf{r}} + (I - \bar{J}_W^{\dagger}\bar{J}_W)W^{\frac{1}{2}}\mathbf{u}_0, \tag{9}$$

where \bar{J}_W^{\dagger} is the Moore–Penrose pseudoinverse of the matrix J_W such that $J_W\bar{J}_W^{\dagger} = I$, $I - \bar{J}_W^{\dagger}J_W$ is the orthogonal projection into the null space of J_W . The first term of (9) is the particular solution, and the second term is the homogeneous solution. It is trivial to show that $I - \bar{J}_W^{\dagger}J_W$ projects \mathbf{u}_0 onto the null space of J_W by multiplying both sides of (9) by \bar{J}_W . The velocity input vector is then recovered using the second part of (7):

$$\mathbf{u} = W^{\frac{1}{2}}\bar{J}_W^{\dagger}\dot{\mathbf{r}} + W^{\frac{1}{2}}(I - \bar{J}_W^{\dagger}\bar{J}_W)W^{\frac{1}{2}}\mathbf{u}_0. \tag{10}$$

The degree of kinematic redundancy at the velocity level is relevant for this solution to be feasible, since it defines the number of simultaneous constraints that can be satisfied in the differential kinematics inversion without \bar{J}_W losing its rank. The degree of redundancy for NMMs is calculated as $R = \delta_m - m$. Whilst analyzing the matrix $W^{\frac{1}{2}}$ for three different cases, it is possible to understand the expected behavior of solution (10). Let z represent the number of elements that are zero in the diagonal of $W^{\frac{1}{2}}$, i.e., the number of joints that are forced to stop due to z simultaneously activated constraints. When $z < R$, both the particular and homogeneous solutions exist, and therefore, the secondary task will still be considered. When $z = R$, the system is not redundant anymore and only the particular solution exists. Finally, when $z > R$, the system (8) has no solution. Therefore, for the solution (10) to exist, the condition $z \leq R$ is necessary.

Our proposed solution (10) has an advantage over the projected gradient solution used in [5,6], because it includes the physical constraints and penalizes both the particular and homogeneous solutions. Furthermore, in contrast with the weighted least-norm method [11], this solution takes advantage of the redundant joints that have not been penalized, due to physical constraints, in attempt to satisfy the task defined by \mathbf{u}_0 .

In this paper, the matrix W is constructed to satisfy joint position constraints (joint limits and self-collision avoidance), which is discussed in Section 4.2. The vector \mathbf{u}_0 is designated to locally maximize a proposed new manipulability measure for mobile manipulators, as discussed in Section 4.1. To satisfy the joint velocity constraints, the particular and homogeneous solutions are analyzed. Because the particular solution is in charge of following the end-effector’s task, it cannot be modified. On the other hand, the homogeneous solution can be varied to satisfy the joint velocity constraints. Furthermore, the homogeneous solution must also consider the admissible velocity commands with

respect to the nonholonomic constraints [7]. Taking these two points into account, we define vector u_0 as

$$u_0 = \pm\alpha(t)\beta(t)S^T(q)\nabla_q F(q), \tag{11}$$

where $\alpha(t)$ is a scalar coefficient that is adjusted online to satisfy joint velocity limits (Section 4.3.2), $\beta(t)$ is a positive variation factor used to satisfy joint velocity boundary constraints (Section 4.3.1), $S(q)$ is defined in (4), and $\nabla_q F(q)$ is the gradient of the objective function $F(q) : \mathbb{R}^{3+n_a} \rightarrow \mathbb{R}$, which is set to a manipulability measure. The product $\alpha(t)\beta(t)$ is the step size of the gradient step ascent/descent, and the sign of u_0 defines whether the objective function $F(q)$ will be maximized (+) or minimized (-).

Replacing (11) in (10), the final form of the proposed solution for u is:

$$\begin{aligned} u &= u_p + \alpha(t)\beta(t)u_h, \\ \text{with} \\ u_p &= W^{\frac{1}{2}}\bar{J}_W^{\dagger}\dot{r}, \\ u_h &= W^{\frac{1}{2}}(I - \bar{J}_W^{\dagger}\bar{J}_W)W^{\frac{1}{2}}S^T(q)\nabla_q F(q), \end{aligned} \tag{12}$$

where the positive sign for the gradient step descent has been used because we are interested in maximizing the manipulability. Notice that when the sign of $\alpha(t)$ is positive, the objective function $F(q)$ is maximized. However, for some cases, $\alpha(t)$ could be negative for short periods of time to respect joint velocity limits, as explained in Section 4.3.

4. Manipulability Maximization and Constraints Satisfaction

In this section, the details for manipulability maximization are described, the weighting matrix W is defined so that it penalizes the joint movement to avoid both joint limits and self-collisions, and a scheme to satisfy joint velocity limits as well as joint velocity boundary constraints is presented.

4.1. Manipulability Maximization

The term manipulability, introduced by Yoshikawa [12], describes the ability of a robotic system to provide end-effector’s velocities in any direction for a given configuration. The manipulability of wheeled mobile manipulators was studied in detail in [5]. There exist different algebraic measures to characterize the manipulability of a robotic system [5,12–14]. The most widely used measure, known as the manipulability measure and noted here as Ω , is given by $\Omega = \sigma_1\sigma_2 \dots \sigma_m$, where $\sigma_1, \sigma_2, \dots, \sigma_m$ are the singular values of the system Jacobian $J(q)$. Therefore, Ω is the manipulability measure for system configuration q . The measure Ω is proportional to the volume of the manipulability ellipsoid [12]. Furthermore, it can be shown that Ω has the following property:

$$\Omega = \sqrt{\det(J(q)J^T(q))} \tag{13}$$

By definition, $\Omega \geq 0$, and $\Omega = 0$ if and only if $\text{rank}(J(q)) < m$. The elements of $\nabla_q \Omega$ are given mathematically by

$$\begin{aligned} \frac{\partial \Omega}{\partial q_i} &= -\frac{1}{2}\Omega \cdot \text{trace} \left((JJ^T)^{-1} \left(\frac{\partial J}{\partial q_i} J^T + J \left(\frac{\partial J}{\partial q_i} \right)^T \right) \right), \\ \text{with } i &= 1, 2, \dots, n. \end{aligned} \tag{14}$$

As mentioned before, we are not only concerned with improving the manipulability of the whole system but also that of the arm alone. In this work, we present a new manipulability measure that better expresses the manipulability of a mobile manipulator, because both the manipulability of the arm and the whole system are intrinsically considered.

Let us define the manipulability of the whole system, denoted as Ω_{p+a} , and the manipulability of the robot arm, denoted as Ω_a , to be the measures obtained using Equation (13) with the Jacobians \bar{J} and J_a from Equation (5), respectively. Notice that by using the reduced Jacobian \bar{J} , the platform's nonholonomic constraints are included in the manipulability measure of the whole system. Our proposed manipulability measure for mobile manipulators is defined as

$$\Omega_{MM} = \hat{\Omega}_{p+a} \hat{\Omega}_a, \tag{15}$$

where $\hat{\Omega}_{p+a}$ and $\hat{\Omega}_a$ are the normalized manipulabilities computed by dividing them by their respective maximum values over all the possible configurations of the system:

$$\hat{\Omega}_{p+a} = \frac{\Omega_{p+a}}{\Omega_{p+a_{max}}} \quad \text{and} \quad \hat{\Omega}_a = \frac{\Omega_a}{\Omega_{a_{max}}}.$$

The normalized values are used in our proposed measure to make sure that all its components are in the same scale. This normalization also makes the measures invariant to units and chosen reference frame [14,15].

Notice that $\bar{J} = [J_p \ J_a]S(q) \in \mathbb{R}^{m \times \delta_m}$ in (5) indicates the relation between \bar{J} and J_a . Hence, the singular values of \bar{J} are not necessarily the same as the singular values of J_a . There are cases where J_a is rank-deficient while \bar{J} is not. For these cases, the whole system manipulability $\Omega_{p+a} \neq 0$ even though the arm is in a singular configuration. Therefore, the measure Ω_{p+a} fails to express the ability of the arm alone to provide end-effector's velocities in any direction. As a result, maximizing the measure Ω_{p+a} might result in the poor manipulability of the arm, even though the manipulability of the whole system is preserved or improved. This is an issue that has been discussed in the literature [6,10].

On the other hand, our proposed measure Ω_{MM} is defined by the product of the singular values of \bar{J} and J_a , i.e., the product of the manipulability ellipsoids of \bar{J} and J_a . Therefore, it encapsulates the abilities of the whole system and the arm to provide end-effector's velocities in any direction. If any of the singular values of J_a or \bar{J} is zero, e.g., the arm is in a singularity, the measure $\Omega_{MM} = 0$. Hence, the measure Ω_{MM} is a better representation of the manipulability of mobile manipulators because it intrinsically considers the manipulability of the arm and the whole system.

Using the product rule, the gradient of Ω_{MM} is calculated as

$$\nabla_q \Omega_{MM} = \frac{1}{\Omega_{p+a_{max}} \Omega_{a_{max}}} (\Omega_a \nabla_q \Omega_{p+a} + \Omega_{p+a} \nabla_q \Omega_a). \tag{16}$$

Analyzing the right hand side of (16), it is clear that the manipulability of the arm subsystem affects the gradient of the whole system. Likewise, the manipulability of the whole system affects the gradient of the arm subsystem. Therefore, maximizing Ω_{MM} , i.e., setting $\nabla_q F(q) = \nabla_q \Omega_{MM}$ in our solution (12) will simultaneously improve the manipulabilities of the arm and the whole system. In the experiments section (Section 5), a comparison is performed among the different objective functions for manipulability maximization to demonstrate the advantages of the proposed manipulability measure.

The reader may have noticed that because our proposed motion planning solution (12) multiplies $W^{\frac{1}{2}}$ with the gradient $\nabla_q F(q)$, the direction of the original gradient that maximizes the manipulability is changed. However, as our goal is to push the system away from singularities by gradually improving the manipulability rather than finding the shortest path towards its maximum value, it is sufficient to prove that the weighted vector $W^{\frac{1}{2}}u_0$ also points in a direction that increases the manipulability as the unweighted gradient u_0 does. Since $W^{\frac{1}{2}}$ is positive semidefinite, we have $u_0^T W^{\frac{1}{2}} u_0 \geq 0$ for all u_0 . This implies that the weighted gradient also points in a direction that increases the manipulability.

4.2. Joint Position Constraints

Joint position constraints are common requirements in the motion planning of robotic systems. In this work, joint limits and self-collision avoidance are considered. These constraints are included in the proposed solution (12) using matrix W , which we define as the product of three terms as follows:

$$W = W_{Jlim}W_{Col}T_q^{-1},$$

where W_{Jlim} is the weighting matrix for joint limits constraints, W_{Col} is the weighting matrix for self-collision avoidance, and the matrix T_q (known as the Jacobian normalization matrix), is used to normalize the velocity commands u as follows [12]:

$$T_q = \text{diag}\left(\frac{1}{u_{1max}}, \frac{1}{u_{2max}}, \dots, \frac{1}{u_{\delta mmax}}\right),$$

where u_{imax} is the i th maximum velocity command. The weighting of this matrix handles the differences in units and scales among all the joints.

For mobile manipulators, matrices W_{Jlim} and W_{Col} should be constructed only considering the arm. This is because there is no limit to the movement of the mobile platform along each degree of freedom. Likewise, the movement of the mobile platform cannot be used to avoid self-collisions because the platform moves the base of the arm. Therefore, in this work, the structure of both weighting matrices was designed to take this into account. To simplify the presentation, let W_g represent either W_{Jlim} or W_{Col} . W_g is a diagonal $\delta_m \times \delta_m$ matrix with the following form:

$$W_g = \begin{bmatrix} I_2 & 0 \\ 0 & W_a \end{bmatrix},$$

where I_2 is an identity matrix of size 2 (for the case of the differential drive) and W_a is a diagonal $n_a \times n_a$ matrix given by

$$W_a = \begin{bmatrix} w_1 & 0 & 0 & \dots & 0 \\ 0 & w_2 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & w_{n_a} \end{bmatrix},$$

where each element of the diagonal matrix W_a is defined using a performance criterion $H(q)$.

In the next two subsections, two separate criterion functions for joint limits and self-collision avoidance are defined. These two criterion functions have common properties. The values of $H(q)$ and $|\partial H(q)/\partial q_i|$ become very large as the constraints are violated. When constructing matrix W_a using these criterion functions, the joint movement towards or away from a constraint must be contemplated [11,16]. Under this consideration, the elements of W_a are given by

$$w_i = \begin{cases} \frac{1}{1 + \left| \frac{\partial H(q)}{\partial q_i} \right|}, & \text{if } \Delta \left| \frac{\partial H(q)}{\partial q_i} \right| > 0 \\ 1, & \text{if } \Delta \left| \frac{\partial H(q)}{\partial q_i} \right| \leq 0, \end{cases} \tag{17}$$

with $i = 1, 2, \dots, n_a$,

where $\Delta|\partial H(q)/\partial q_i|$ is the change rate of $|\partial H(q)/\partial q_i|$ with respect to time, and is numerically calculated during implementation. With this choice, a value of one is assigned to w_i , indicating that no penalty is imposed on the i th joint, if the i th joint is not moving ($\Delta|\partial H(q)/\partial q_i| = 0$), or it moves away from a constraint ($\Delta|\partial H(q)/\partial q_i| < 0$). On the other hand, w_i tends towards zero if the movement of the i th joint gets closer to a constraint.

Hence, the element w_i penalizes the movement of the i th joint by means of (12) if it moves towards a constraint and stops the joint if it is too close to it.

4.2.1. Joint Limits Avoidance

To construct the weighting matrix for joint limits avoidance W_{Jlim} , a well-known criterion function [17] is used:

$$H_{Jlim}(\mathbf{q}) = \sum_{i=1}^{n_a} \frac{1}{4\gamma} \frac{(q_i^+ - q_i^-)^2}{(q_i^+ - q_i)(q_i - q_i^-)},$$

where q_i is the i th joint angle, q_i^- and q_i^+ are its lower and upper limits, respectively, and γ is a scalar constant that adjusts the rate of change of $H_{Jlim}(\mathbf{q})$. This criterion function increases as the joint gets closer to its limits and goes to infinity at the joint bounds. The elements of the gradient of this function are given by

$$\frac{\partial H_{Jlim}(\mathbf{q})}{\partial q_i} = \frac{1}{4\gamma} \frac{(q_i^+ - q_i^-)^2 (2q_i - q_i^+ - q_i^-)}{(q_i^+ - q_i)^2 (q_i - q_i^-)^2}. \tag{18}$$

Each element $\partial H_{Jlim}(\mathbf{q})/\partial q_i$ is equal to zero if q_i is at the middle of its joint range and goes to infinity at either limit.

4.2.2. Self-Collision Avoidance

To construct the weighting matrix for self-collision avoidance W_{Col} , an exponential function of the distance between two collision points is used as the criterion function [16]:

$$H_{Col}(\mathbf{q}) = \rho e^{-c_1 d(\mathbf{q})} d(\mathbf{q})^{-c_2},$$

where $\rho > 0$ controls the amplitude of $H_{Col}(\mathbf{q})$, and $c_1, c_2 > 0$ control the rate of decay. This function has a maximum value when the distance d between two links is zero, and exponentially decreases as this distance increases. The distance between two collision points is defined as $d(\mathbf{q}) = \|\mathbf{p}_{l1} - \mathbf{p}_{l2}\|_2$, where \mathbf{p}_{l1} and \mathbf{p}_{l2} represent the position vectors, referred to a common frame, of the collision points on two nonadjacent links. \mathbf{p}_{l1} and \mathbf{p}_{l2} can be calculated from the configuration vector \mathbf{q} through forward kinematics.

The elements of the gradient of $H_{Col}(\mathbf{q})$ are given by

$$\frac{\partial H_{Col}(\mathbf{q})}{\partial \mathbf{q}} = \frac{\partial H_{Col}(\mathbf{q})}{\partial d} \frac{\partial d}{\partial \mathbf{q}}, \tag{19}$$

where each of the partial derivatives is given by

$$\frac{\partial H_{Col}(\mathbf{q})}{\partial d} = -\rho e^{-c_1 d} d^{-c_2} (c_2 d^{-1} + c_1), \tag{20}$$

$$\frac{\partial d}{\partial \mathbf{q}} = \frac{1}{d} [J_1^T (\mathbf{p}_{l1} - \mathbf{p}_{l2}) + J_2^T (\mathbf{p}_{l2} - \mathbf{p}_{l1})], \tag{21}$$

where J_1 and J_2 are the associated Jacobian matrices of \mathbf{p}_{l1} and \mathbf{p}_{l2} , respectively. The collision points are chosen from the surface of the links for which the collision distance is computed. For the case of potential collisions between the arm and the mobile platform, \mathbf{p}_{l2} is picked as a point fixed on the surface of the mobile platform (\mathbf{p}_{l2} does not move with respect to the arm). By using a frame attached to the mobile platform as the common frame, and selecting \mathbf{p}_{l2} as a fixed point, (21) reduces to:

$$\frac{\partial d}{\partial \mathbf{q}} = \frac{1}{d} [J_1^T (\mathbf{p}_{l1} - \mathbf{p}_{l2})]. \tag{22}$$

When constructing matrix W_a , the partial derivative $\partial d/\partial \mathbf{q}$ in (19) was calculated using (21) or (22) depending on whether the collision is evaluated between two moving links or

a moving link and a fixed one, respectively. Each element $\partial H(q)/\partial q_i$ tends toward zero as the distance between two evaluated collision points increases, and tends towards infinity as the distance gets closer to zero. Because $d = 0$ is not admissible, i.e., the two links are in contact, the chosen points must contemplate a threshold.

Multiple pairs of potential collision points might exist in a robotic system. Let N_c be the number of potential collision point pairs that are evaluated. A self-collision matrix W_{Col_j} is constructed for each distance d_j , with $j = 1, 2, \dots, N_c$. The weighting matrix that includes the contribution of each evaluated pair is finally obtained by

$$W_{Col} = \prod_{j=1}^{N_c} W_{Col_j}. \tag{23}$$

With this combination, the i th diagonal element of the matrix W_{Col} penalizes the movement of the i th joint. In contrast with the combination of collision weighting matrices proposed in [16], the combination (23) guarantees that the joints are stopped if they attempt to decrease any of the collision distances d_j to a value of zero. Since some potential collisions are taken care of by the joint limits, the number of points considered in (23) is small, and therefore the computation cost can be relieved.

4.3. Joint Velocity Constraints

In a task space trajectory tracking problem, joint constraints at the velocity level are as important as joint limits and self-collision avoidance. These constraints include joint velocity boundary constraints and joint velocity limits. Satisfying joint velocity boundary constraints is a requirement for the end-effector to stop at the beginning and end of the task, i.e., the initial and final joint velocities must be zero. Joint velocity limits must be considered because the motion planning algorithm might generate joint velocity commands that are out of bounds in order to follow the task space velocity profile. These unfeasible velocities cannot be achieved by the real joints. Therefore, the trajectory tracking will fail if the joint velocity limits are not respected.

4.3.1. Joint Velocity Boundary Constraints

To satisfy joint velocity boundary constraints, the initial and final joint velocities must be zero. u_p in (12) is directly associated with the end-effector’s task; therefore, it implicitly satisfies the boundary constraints. However, u_h in (12) does not necessarily satisfy them, because the manipulability maximization task is dependent on the mobile manipulator configuration at the initial and final poses. In this work, we propose using a time-varying self-motion magnitude to handle these constraints. The objective is to avoid non-zero values of u_h only at the beginning and end of the trajectory. With this in mind, it is proposed to set the variation factor $\beta(t)$ to increase from zero to one at the beginning of the trajectory, maintain a value of one for most of the trajectory, and then decrease from one to zero at the end of the trajectory.

Let t_b be the blending time for $\beta(t)$ to increase from zero to one, and t_f be the trajectory execution time. To achieve a smooth transition, a 5th order polynomial $\beta_1(t) = a_0 + a_1t + a_2t^2 + a_3t^3 + a_4t^4 + a_5t^5$ is used when $t < t_b$. The decrement in $\beta(t)$ at the end of the trajectory, when $t > t_f - t_b$, is the complement of the polynomial $\beta_1(t)$ and is defined by $\beta_2(t) = 1 - \beta_1(t - t_f + t_b)$. By imposing the conditions $\beta_1(0) = 0, \beta_1(t_b) = 1, \beta_1(0) = 0, \beta_1(t_b) = 0, \beta_1(0) = 0, \beta_1(t_b) = 0$, the values of the coefficient of $\beta_1(t)$ are found.

After finding the polynomial coefficients, the self-motion magnitude variation factor is given by

$$\beta(t) = \begin{cases} a_3t^3 + a_4t^4 + a_5t^5, & \text{if } t < t_b \\ 1, & \text{if } t_b \leq t \leq t_f - t_b \\ 1 - (a_3(t - t_f + t_b)^3 + a_4(t - t_f + t_b)^4 + a_5(t - t_f + t_b)^5), & \text{if } t > t_f - t_b, \end{cases} \quad (24)$$

with:

$$\begin{aligned} a_0 &= 0, & a_1 &= 0, & a_2 &= 0, \\ a_3 &= \frac{10}{t_b^3}, & a_4 &= -\frac{15}{t_b^4}, & a_5 &= \frac{6}{t_b^5}, \end{aligned}$$

The blending time t_b is chosen by the user. Figure 3 shows the shape of $\beta(t)$ for $t_f = 15(s)$, $t_b = 2.5(s)$.

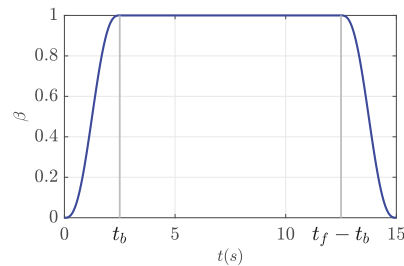


Figure 3. Example shape of self-motion variation factor β .

4.3.2. Joint Velocity Limits

To satisfy joint velocity limits, the maximum magnitude of self-motion is determined such that the velocity limit for each joint is not violated [18], namely:

$$|u_{ip}(t) + \alpha\beta u_{ih}(t)| \leq u_i^+, \quad (25)$$

where u_i^+ is the i th joint velocity limit with $i \in 1, 2, \dots, \delta_m$. Therefore, to avoid exceeding the joint velocity limits, $\alpha(t)$ must be selected such that it satisfies (25). The upper and lower limits of $\alpha(t)$ can be found using this equation. For each joint, it can be shown that the maximum and minimum values of $\alpha(t)$, denoted by $\alpha_{i_{max}}(t)$ and $\alpha_{i_{min}}(t)$, respectively, are given by

$$\alpha_{i_{max}}(t) = \max \left(\frac{u_i^+ - u_{ip}(t)}{\beta(t)u_{ih}(t)}, \frac{-u_i^+ - u_{ip}(t)}{\beta(t)u_{ih}(t)} \right), \quad (26)$$

$$\alpha_{i_{min}}(t) = \min \left(\frac{u_i^+ - u_{ip}(t)}{\beta(t)u_{ih}(t)}, \frac{-u_i^+ - u_{ip}(t)}{\beta(t)u_{ih}(t)} \right). \quad (27)$$

Then, $\alpha_{max}(t)$ and $\alpha_{min}(t)$ for all the joints are:

$$\alpha_{max}(t) = \min (\alpha_{1_{max}}(t), \alpha_{2_{max}}(t), \dots, \alpha_{\delta_{m_{max}}}(t)), \quad (28)$$

$$\alpha_{min}(t) = \max (\alpha_{1_{min}}(t), \alpha_{2_{min}}(t), \dots, \alpha_{\delta_{m_{min}}}(t)), \quad (29)$$

where $\alpha_{max}(t)$ is the self-motion magnitude limit. In [18], the maximum value of $\alpha_{max}(t)$ and minimum value of $\alpha_{min}(t)$ are calculated for the whole trajectory, and the upper bound of $\alpha_{max}(t)$ or lower bound of $\alpha_{min}(t)$ is used as the step size to take advantage of the maximum admissible velocities. In our approach, a suitable initial value α_s is selected

through experimentation and the upper and lower limits of α are calculated for each time t . The value of α at time t is then given by

$$\alpha(t) = \begin{cases} \alpha_{max}(t), & \text{if } \alpha_s > \alpha_{max}(t), \\ \alpha_{min}(t), & \text{if } \alpha_s < \alpha_{min}(t), \\ \alpha_s, & \text{otherwise.} \end{cases} \quad (30)$$

This technique, in contrast with using the maximum/minimum self-motion for the entire trajectory, prevents sudden joint accelerations due to the use of maximum velocities in every step. Note that for some cases $\alpha(t)$ may be negative if $\alpha_{max}(t)$ is negative, which means that the joint velocity limits cannot be satisfied without decreasing the manipulability of the system. The task can be executed as long as the system is still away from any singularity. The reader may have noticed that $\beta(t)$ is in the denominator of (26) and (27), and as shown in Section 4.3.1, $\beta(t)$ is zero at the beginning and the end of the trajectory. For these two cases, $\alpha_{min} = -\infty$ and $\alpha_{max} = \infty$, but this does not cause instability in the system because by following (30), the value of $\alpha(t)$ is set to α_s .

It is also possible to detect whether a task can be accomplished or not by using the limits of $\alpha(t)$. If the inequality $\alpha_{max}(t) < \alpha_{min}(t)$ is true, then a suitable value of $\alpha(t)$ which avoids the joint velocity limits does not exist, because even the particular solution will violate them. In other words, the given task space trajectory cannot be accomplished without violating at least one of the joint velocity limits. For these situations, the task space trajectory must be replanned with a longer execution time t_f or with lower end-effector's maximum velocities.

5. Experiments

Experiments were carried out to verify the efficacy of our scheme to solve the motion planning problem for trajectory tracking at the kinematic level. In this section, the results for the tracking of two trajectories, a Lissajous trajectory (see Section 5.4), and an elliptic trajectory (see Section 5.5), are analyzed. These trajectories were picked to demonstrate the ability of our approach to comply with the different constraints introduced in this manuscript while improving the manipulabilities of the whole system and the robot arm. For the Lissajous trajectory, a comparison of different objective functions for manipulability maximization is made to highlight the advantages of the proposed manipulability measure for mobile manipulators.

The experiments were performed using a 10-DOF NMM developed by the Industrial Technology Research Institute (ITRI), as shown in Figure 4. This NMM is composed of a differential-drive wheeled mobile platform, a prismatic joint mounted on top of the platform, and a Universal Robots UR5 6-DOF robotic arm attached to the prismatic joint. From this point forward, the UR5 arm's joints are denoted as q_{ai} with $i = 1, 2, \dots, 6$ and the prismatic joint as z_{pj} . The respective Denavit–Hartenberg (DH) parameters are shown in Table 1. The joint limits and joint velocity limits are shown in Table 2.

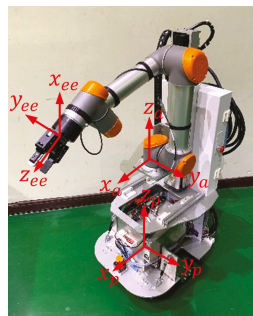


Figure 4. Nonholonomic mobile manipulator used for the experiments.

Table 1. D-H parameters.

Joint	a (m)	α (rad)	d (m)	θ (rad)
mob. plat.	0	0	0	θ_p
z_{pj}	-0.049	0	$z_{pj}+0.5562$	0
q_{a1}	0	$\pi/2$	0.08916	$q_{a1} + \pi$
q_{a2}	-0.425	0	0	q_{a2}
q_{a3}	-0.39225	0	0	q_{a3}
q_{a4}	0	$\pi/2$	0.1093	q_{a4}
q_{a5}	0	$-\pi/2$	0.09465	q_{a5}
q_{a6}	0	0	0.0823	q_{a6}

Table 2. Joints physical constraints.

Joint	q_{min}	q_{max}	u_{max}
v_p	$-\infty$	∞	0.3 (m/s)
ω_p	$-\infty$	∞	$\pi/2$ (rad/s)
z_{pj}	0 (m)	0.25 (m)	0.025 (m/s)
q_{a1}	-1.7453 (rad)	0.0175 (rad)	π (rad/s)
q_{a2}	$-\pi/2$ (rad)	0.4363 (rad)	π (rad/s)
q_{a3}	0 (rad)	π (rad)	π (rad/s)
q_{a4}	-2π (rad)	2π (rad)	π (rad/s)
q_{a5}	-2π (rad)	2π (rad)	π (rad/s)
q_{a6}	-2π (rad)	2π (rad)	π (rad/s)

The Jacobian of the arm J_a used for calculation of $\hat{\Omega}_a$ in (15) was constructed only using the UR5 arm without the prismatic joint. If the prismatic joint were included, the arm would be allowed to stretch for most tasks since it would always be able to move the end-effector vertically using the prismatic joint, i.e., the manipulability is not affected when the arm is horizontally stretched. Stretching the arm for most tasks is an undesirable behavior, and therefore, the manipulability maximization of the UR5 arm without the prismatic joint is a suitable selection. The Jacobian J_a is calculated with respect to the frame $X_a Y_a Z_a$ shown in Figure 4.

Due to the lack of a reliable positioning system, the odometry of the wheels was used in the experiments to compute the position and orientation of the mobile platform, and the forward kinematics were used to compute the end-effector's pose, which in turn was fed back to the motion planning algorithm. Therefore, the errors presented in the experiments are not the real-world errors, but the errors in the trajectory in which it is assumed that a reliable positioning system exists. Furthermore, as mentioned in the introduction section, the scope of this work is the motion planning of NMM for trajectory tracking at the kinematic level; therefore, the objective of the presented experiments is to validate the proposed algorithm at the kinematic level. Problems inherent to the dynamic behavior of the system, including vibrations of the mechanical structure, friction from the ground, etc., have an impact on the real end-effector tracking error, but they are out of scope of this paper and not considered in the algorithm. For these reasons, the simulation results of the position and orientation errors along the trajectory are also shown in this manuscript to highlight the performance of our motion planning algorithm.

An important remark is that the system vibrations due to the NMM mechanical structure greatly affected the performance of the motion planning algorithm when the system was close to its joint limits or self-collision constraints. This behavior was not observed when performing the simulations. To address this issue, a moving average filter with a window size of five was used to filter the gradients of the criterion functions for joint limits avoidance and self-collision avoidance. This filter diminished the impact of such vibrations on the motion planning algorithm.

5.1. Orientation Error for 6-DOF Tasks

In all the experiments, 6-DOF tasks were performed. For a 6-DOF task ($m = 6$), where the position and orientation of the end-effector are considered, the expression $r_d - r$ (mentioned in Section 3) has a specific definition that depends on the orientation representation, i.e., $\tilde{r} = r_d - r$ does not hold for all orientation representations [1]. In this work, unit quaternions are used to describe the end-effector's orientation because of their efficiency and nonsingular representation for all orientations [1,19–21].

A unit quaternion $Q = [w + xi + yj + zk]$ is represented in scalar-vector form by $Q = \{s, v\}$ with $s \in \mathbb{R}$ and $v \in \mathbb{R}^3$, where s and v are called the scalar and vector elements of Q , respectively. The desired and current pose are defined using unit quaternions for orientation as $r_d = [P_d \ Q_d]^T$ and $r_c = [P_c \ Q_c]^T$, where $P_d = [x_d, y_d, z_d]$ and $P_c = [x_c, y_c, z_c]$ are the desired position and current position, respectively, and $Q_d = \{s_d, v_d\}$ and $Q_c = \{s_c, v_c\}$ are the desired orientation and current orientation, respectively. The position error e_P is defined as $e_P = P_d - P_c$. The orientation error can be described in terms of the quaternion $\Delta Q = \{\Delta s, \Delta v\}$, where [1]:

$$\begin{aligned} \Delta s &= s_c s_d - v_d^T v_c, \\ \Delta v &= s_c v_d - s_d v_c - v_d \times v_c. \end{aligned} \tag{31}$$

If the desired orientation and current orientation are aligned, i.e., with zero orientation error, then $\Delta Q = \{1, 0\}$. Consequently, it would be sufficient to define the orientation error to be Δv . It is also important to follow a convention for the sign of the quaternion because $Q = \{s, v\}$ and $-Q = \{-s, -v\}$ represent the same orientation. A common convention is to keep the scalar quaternion element positive. Take this into account and the orientation error is defined as follows:

$$e_O = \begin{cases} \Delta v, & \text{if } \Delta s \geq 0 \\ -\Delta v, & \text{if } \Delta s < 0. \end{cases} \tag{32}$$

Separating the position and orientation errors, the motion planning control law (6) is rewritten as

$$\dot{r} = \dot{r}_d + \begin{bmatrix} K_P & 0 \\ 0 & K_O \end{bmatrix} \begin{bmatrix} e_P \\ e_O \end{bmatrix}, \tag{33}$$

where K_P and K_O are positive definite diagonal 3×3 matrices. Note that e_O in (32) is not an angle difference but its size represents the size of the orientation error that, as shown in [1,22], can achieve the convergence of the orientation error.

5.2. Evaluated Self-Collision Pairs

The types of self-collision can be significantly reduced by setting the joint limits properly. In addition, collisions among the first three links and the last three links can be taken care of by maximizing the arm manipulability, because the 6-DOF manipulator has poor manipulability if the arm is retracted to the point where the wrist is close to the base of the arm. Therefore, only the self-collision between the arm and the mobile platform needs consideration. Such a type of collision takes place when the elbow of the manipulator collides with the top of the platform, or when the wrist collides with the front of the platform. The distances associated with these potential self-collisions are depicted in Figure 5. Given that the platform is fixed with respect to the arm, setting frame $X_p Y_p Z_p$ (located on the center of the wheels' axle, as shown in Figure 4) as the reference frame allows to use (22), where p_{11} is a point in the arm's elbow or wrist, correspondingly, and p_{12} is a fixed point with respect to $X_p Y_p Z_p$:

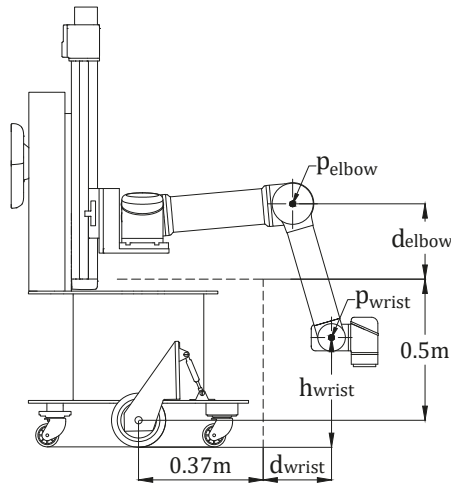


Figure 5. NMM’s self-collision distances.

As illustrated in Figure 5, to prevent the elbow collision, a collision pair between the z component of p_{elbow} with respect to frame $X_p Y_p Z_p$ and a point located at 0.5 (m) from the origin of frame $X_p Y_p Z_p$ in the Z_p direction is selected. The distance for this pair is named d_{elbow} . To prevent wrist collision, a collision pair between the x component of p_{wrist} with respect to frame $X_p Y_p Z_p$ and a point located at 0.37 (m) from the origin of frame $X_p Y_p Z_p$ in the X_p direction is selected. The distance for this pair is named d_{wrist} . The wrist collision is only evaluated if the wrist height h_{wrist} , the z component of p_{wrist} with respect to frame $X_p Y_p Z_p$, is lower than 0.5 (m), otherwise, its associated weighting collision matrix is assigned to identity. This was done to avoid restricting the movement of the joints that reduce d_{wrist} when the wrist is above the top of the mobile platform. All these parameters were chosen based on the physical dimensions of the NMM, and the second point in each of the collision pairs was selected so that when d_{elbow} and d_{wrist} are zero, a gap still exists between the potentially colliding links.

5.3. Common Parameters of the Simulations and Experiments

For all the experiments, the selected feedback gain matrices in (33) are $K_p = 10I_{3 \times 3}$ and $K_O = 20I_{3 \times 3}$, where I is an identity matrix. The initial value of $\alpha(t)$ in (30) is set to $\alpha_s = 3$, and the blending time is set to $t_b = 0.2t_f$ for the variation factor $\beta(t)$ in (24). The parameters for self-collision avoidance in (20) are $\rho = 1 \times 10^{-3}$, $c_1 = 50$ and $c_2 = 1$. The starting configuration of the robot arm is $q_a = [0 \ -80 \ 110 \ -120 \ -90 \ 0]^T$ (°). Furthermore, a sampling time $t_s = 0.02$ (s) was used for the simulations and experiments.

5.4. Lissajous Trajectory

The Lissajous trajectory was picked to demonstrate the ability of the proposed approach to track a trajectory for which the nonholonomic constraints greatly affect the movement of the system. The proposed Lissajous trajectory is defined by

$$r_d(t) = \begin{bmatrix} P_d(t) \\ Q_d(t) \end{bmatrix} = \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ Q_0 \end{bmatrix} + \begin{bmatrix} A \cos(s(t) + \pi/2) \\ B \cos(2(s(t) + \pi/2) + \pi/2) \\ C \cos(2s(t)) - C \\ 0 \end{bmatrix},$$

where $[x_0, y_0, z_0]^T$ is the end-effector’s starting position, $s(t)$ is the trajectory timing variable, and A, B and C define the size of the path. The orientation is set to be the same for the entire

trajectory, i.e., $Q(t) = Q_0$. Notice that this is a 6-DOF trajectory because the orientation is constrained for the entire trajectory.

The starting configurations of the mobile platform and prismatic joint for this trajectory were set to $x_p = -0.1$ (m), $y_p = -0.13$ (m), $\theta_p = -90^\circ$ and $z_{pj} = 0.2$ (m). With this configuration, the end-effector's initial pose is given by $P_0 = [-0.009 \quad -0.6491 \quad 0.9884]$ (m) and $Q_0 = \{0, 0i + 1j + 0k\}$. A trapezoidal velocity profile [1] was used for the derivative of the timing variable $\dot{s}(t)$. The parameters for the path size were set to $A = 1.3$ (m), $B = 1.3$ (m) and $C = 0.27$ (m), and an execution time $t_f = 64$ (s) was chosen. Figure 6a,b show snapshots of the NMM's movement along the Lissajous trajectory, in simulations and experiments, respectively.

Figure 7 compares the trajectory tracking results between the simulation and the experiment. In the simulation, the position and orientation errors are kept small during the whole trajectory, as shown in Figure 7b,c. This demonstrates the good tracking performance at the kinematic level of our proposed motion planning algorithm. The observed larger errors in the experiments, as exhibited in Figure 7e,f, are due to the vibrations of the system during the experiments. Furthermore, as mentioned before, the control of the dynamic behavior of the system is beyond the scope of this work. Nevertheless, the position errors in the experiments are kept below 2×10^{-3} (m) and the orientation errors below 1.5×10^{-3} . Likewise, we observe that the obtained trajectories in the simulation (Figure 7a) and the experiment (Figure 7d) are fairly similar.

Figure 8 illustrates the experiment results. Note that the negative joint velocity limit $-u_i^+$ is denoted as u_i^- in the pertinent figures. The manipulabilities of both the arm and the complete system are kept high during the execution of the trajectory, and their final values are higher than at the start of the trajectory, as shown in Figure 8b. It is important to remark that there are no potential self-collisions during the execution of this trajectory. Even though d_{wrist} is negative at the beginning of the trajectory, as shown in Figure 8c, the wrist is above the top of the mobile platform and therefore the joint movements were not restricted, as explained in Section 5.2.

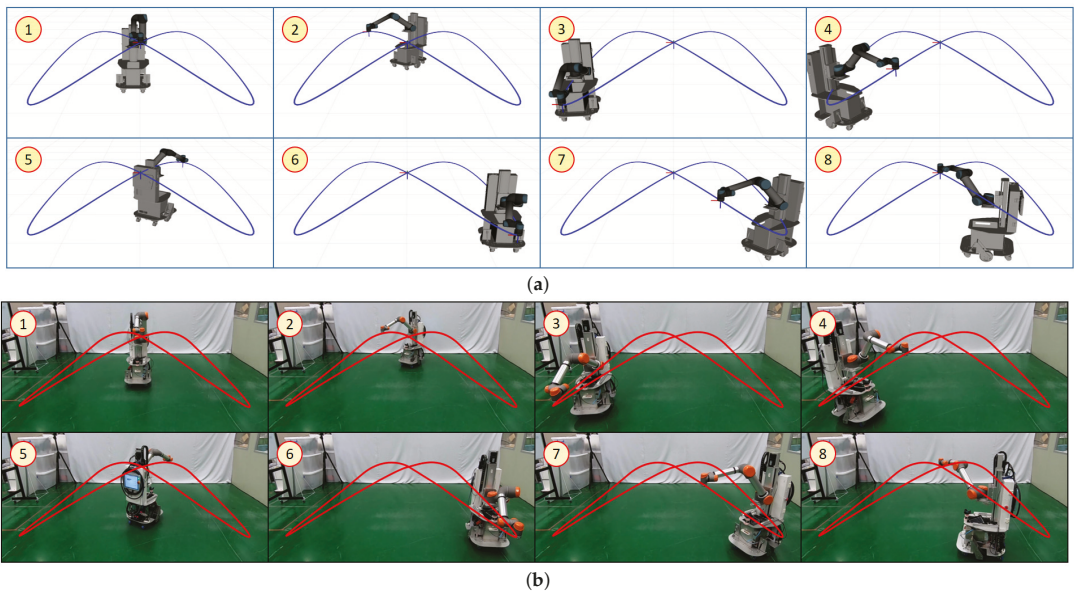


Figure 6. Snapshots of the NMM's motion for the Lissajous trajectory tracking: (a) simulations; and (b) experiments.

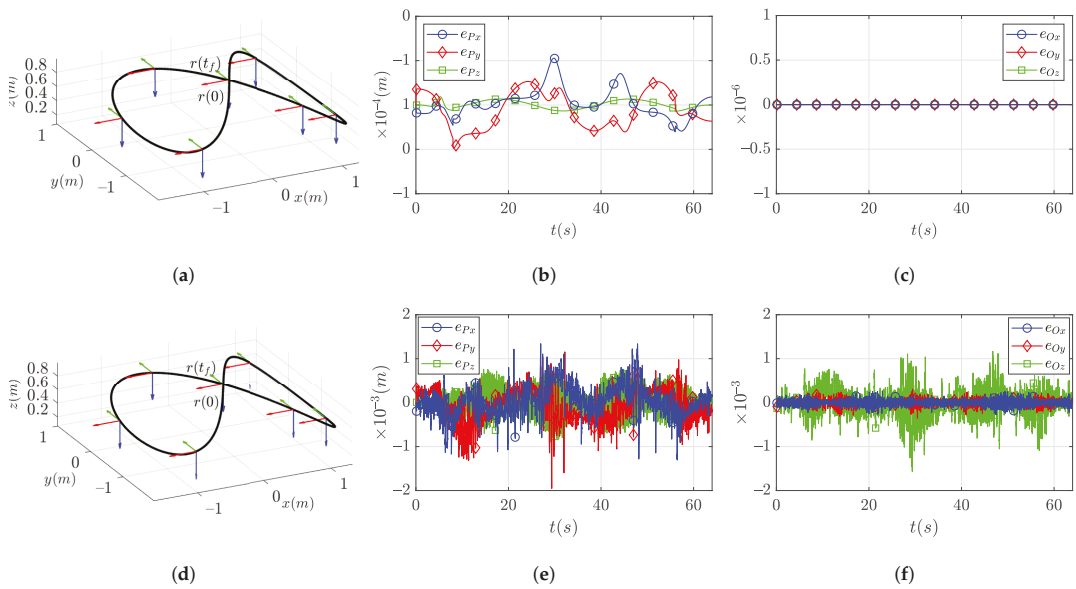


Figure 7. Tracking performance comparison between simulations and experiments for the Lissajous trajectory. End-effector’s trajectory in simulations (a) and experiments (d). Position errors in simulations (b) and experiments (e). Orientation errors in simulations (c) and experiments (f). Notice that the position and orientation errors, panels (b,c), respectively, are small in the simulations. In addition, notice that the trajectories in simulations (a) and experiments (b) are quite similar. The reason for the larger errors obtained in the experiments (panel (e,f)) are discussed in Section 5.4.

Figure 8d–i demonstrate the fulfillment of the joint limits and joint velocity boundary constraints. All the joints are kept within their corresponding limits. Notice that the movement of q_{a1} , Figure 8f, is restricted in the time interval $t = (29, 32)$ (s) to respect its lower limit. This time interval corresponds to the trajectory section between snapshots number three and four in both Figure 6a,b. For this section of the trajectory, the movement of the end-effector in the XY plane is taken care of by the platform due to the restriction imposed to q_{a1} . We observe in Figure 8e that the velocity limits of the prismatic joint are respected. The remaining joints do not reach their respective limits as shown in Figure 8f–h. Furthermore, as seen in Figure 8d,e,i, the velocity profiles for all the joints are smooth and satisfy the boundary constraints, i.e., the initial and final joint velocities are equal to zero.

To demonstrate the advantages of the proposed manipulability measure Ω_{MM} , a comparison of the manipulability maximization results using different objective functions in simulations of the Lissajous trajectory tracking is shown in Figure 9. Here, a task is considered as failed when none of the constraints are satisfied, and thus the simulation is stopped. These results were obtained by using the same parameters shown in Section 5.3 with the only change being the objective function.

The Lissajous trajectory tracking fails when the objective function is set to maximize the manipulability of the arm, i.e., $F(q) = \hat{\Omega}_a$, as shown in Figure 9a. The manipulabilities of both the arm and the whole system start a fast decay after $t = 44$ s. This decrement in the manipulabilities is the result of restricting the homogeneous solution to respect the joint velocity limits. Consequently, the arm manipulability is not maximized anymore and the system moves towards singularity, ultimately failing the task. The results of setting the objective function to the manipulability of the whole system, i.e., $F(q) = \hat{\Omega}_{p+a}$, are shown in Figure 9b. In this case, the task succeeds, however, the manipulability of the arm

deteriorates, and at the end of the trajectory has a value close to zero. This is the behavior discussed in previous studies [6,10].

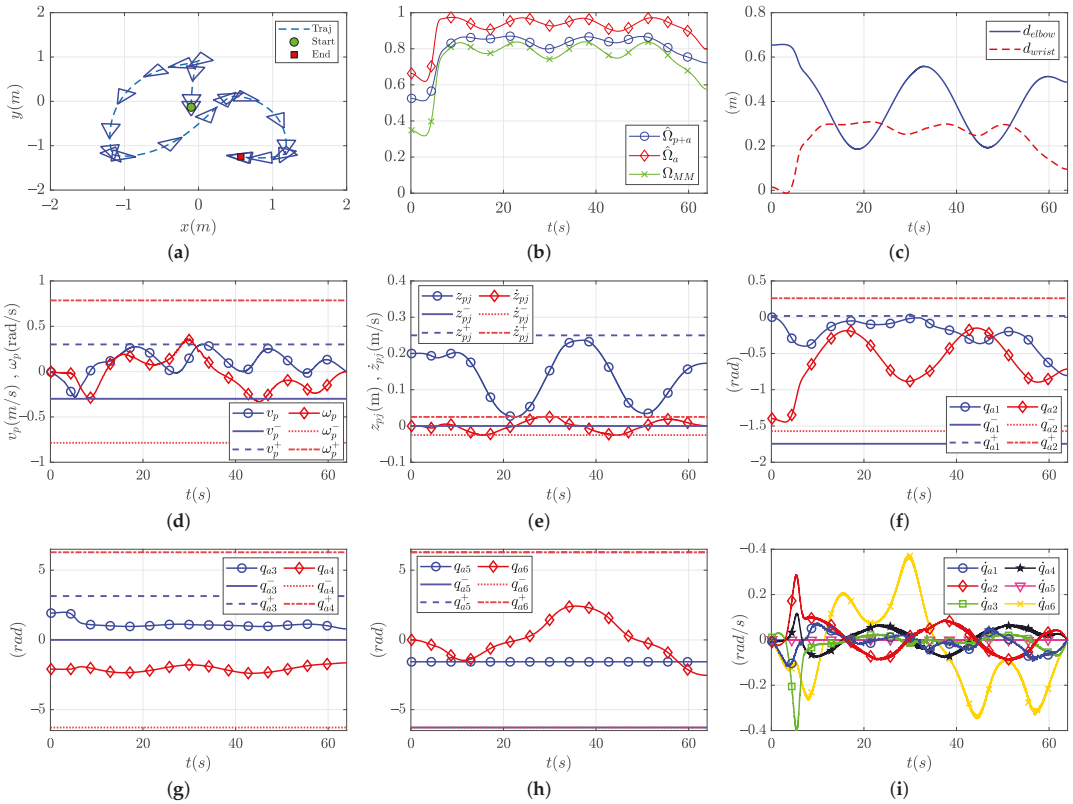


Figure 8. Experiment results for the Lissajous trajectory tracking: (a) mobile platform’s trajectory; (b) normalized manipulability measures; (c) self-collision distances; (d) mobile platform’s velocity commands; (e) prismatic joint’s position and velocity; (f–h) arm’s joint positions; and (i) arm’s joint velocities. Notice that all the joint limits and velocity limits are respected (panels (d–i)). Furthermore, the manipulabilities of both the arm and the whole system are improved (panel (b)). See Section 5.4 for a detailed discussion of this figure.

Figure 9c shows the results of maximizing a linear combination of the manipulabilities of the arm and the whole system, i.e., $F(q) = 0.5\hat{\Omega}_{p+a} + 0.5\hat{\Omega}_a$. Figure 9d shows the results of maximizing the proposed manipulability measure for mobile manipulators, i.e., $F(q) = \Omega_{MM}$ from Equation (15). Both objective functions preserve the manipulability of the arm. However, it is clear that maximizing the proposed measure has better results overall. First, the manipulability of the arm is higher along the trajectory in Figure 9d compared to the arm manipulability obtained with the linear combination in Figure 9c. Secondly, the obtained final manipulabilities have improved with respect to the initial ones in Figure 9d. On the other hand, the linear combination improves the manipulability of the whole system, but not that of the arm with respect to the values at the start of the trajectory, as shown in Figure 9c.

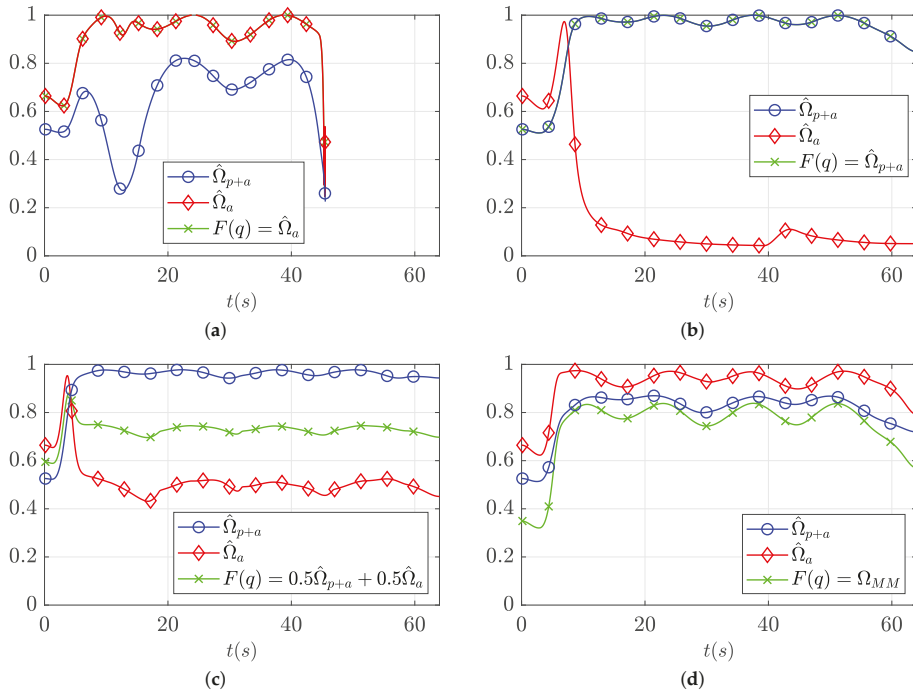


Figure 9. Lissajous trajectory’s comparison of objective functions for manipulability maximization in simulations: (a) maximization of the manipulability of the arm; (b) maximization of the manipulability of the whole system; (c) maximization of a linear combination of the manipulability of the arm and the whole system; and (d) maximization of the proposed mobile manipulator manipulability measure. This figure demonstrates the advantages of the manipulability measure for mobile manipulators (panel (d)) presented in Section 4.1. See Section 5.4 for a detailed discussion of this figure.

5.5. Elliptic Trajectory

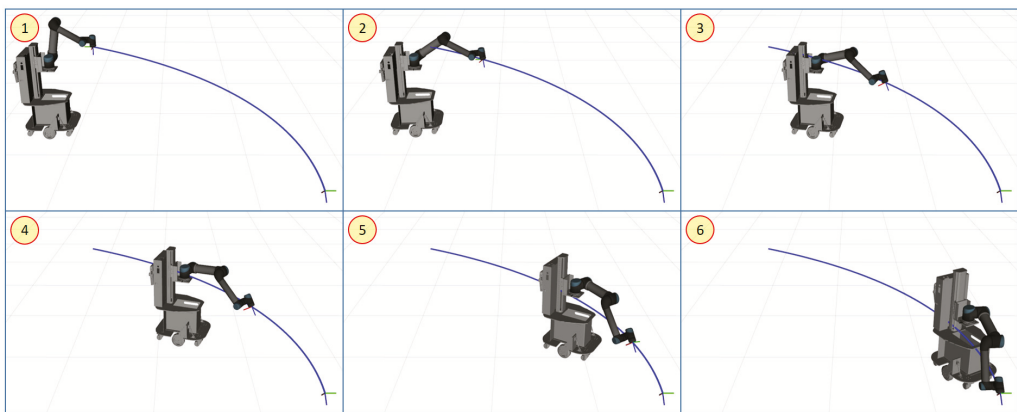
The elliptic trajectory was picked to demonstrate the ability of the proposed approach to comply with joint velocity limits and to prevent self-collisions. This trajectory consists of moving the end-effector from an initial pose $r_0 = [P_0 \ Q_0]^T$ to a desired final pose $r_d = [P_d \ Q_d]^T$ using an elliptic path. The trajectory position is defined by

$$P_d(t) = \begin{bmatrix} A \cos(s(t)) + c_x \\ B \sin(s(t)) + c_y \\ m(s(t) - s_0) + z_0 \end{bmatrix},$$

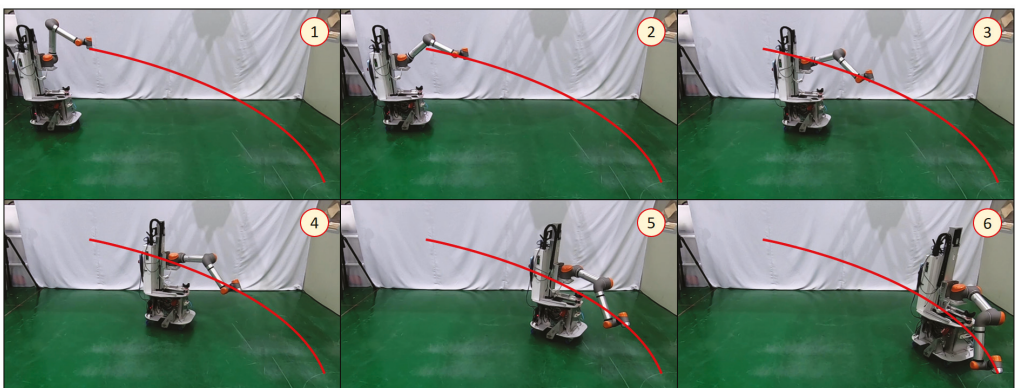
where A, B, c_x, c_y and s_0 are calculated using the XY coordinates of P_0 and P_d , such that the elliptic path is centered at the closest point to the origin, while it covers a 90° angle between the start and end points. The trajectory’s Z coordinate follows a straight line between the points (z_0, s_0) and (z_d, s_d) , where z_0 and z_d are the Z coordinates of P_0 and P_d , respectively; s_0 and s_d are the starting and ending angles of the elliptic path, respectively; and m in the equation above is the slope of this line. A fifth-order polynomial profile [1] was used for the timing variable $s(t)$. The orientations and rotational velocities along the trajectory were computed using quaternion polynomials [23]. This technique has two main advantages: a smooth velocity profile is obtained, and the rotational velocities and accelerations in the task space are included as boundary constraints.

The starting configuration of the mobile platform and prismatic joint for this trajectory was set to $x_p = -1.3$ (m), $y_p = 0.56$ (m), $\theta_p = 0(^{\circ})$, and $z_{pj} = 0.24$ (m). With this configuration, the end-effector's initial pose is given by $P_0 = [-0.781 \ 0.669 \ 1.028]$ (m) and $Q_0 = \{0, 0.7071i - 0.7071j + 0k\}$. The final pose was selected to have position $P_d = [1.55 \ -1.0 \ 0.26]$ (m) and orientation $Q_d = \{0.2706, 0.6533i + 0.6533j - 0.2706k\}$, and an execution time $t_f = 20$ (s) was chosen. Figure 10a,b show snapshots of the NMM's movement along the elliptic trajectory, in simulations and experiments, respectively.

Figure 11 compares the trajectory tracking results between the simulation and the experiment. The small position and orientation errors obtained in the simulation (Figure 11b,c) again demonstrate the good tracking performance of the proposed approach. In the experiment, the position errors are kept below 1.5×10^{-3} (m) and the orientation errors are kept below 1×10^{-3} as depicted in Figure 11e,f. Once again, the vibrations of the system played a role in the larger errors seen in the experiment.



(a)



(b)

Figure 10. Snapshots of the NMM's motion for the elliptic trajectory tracking: (a) simulations; and (b) experiments.

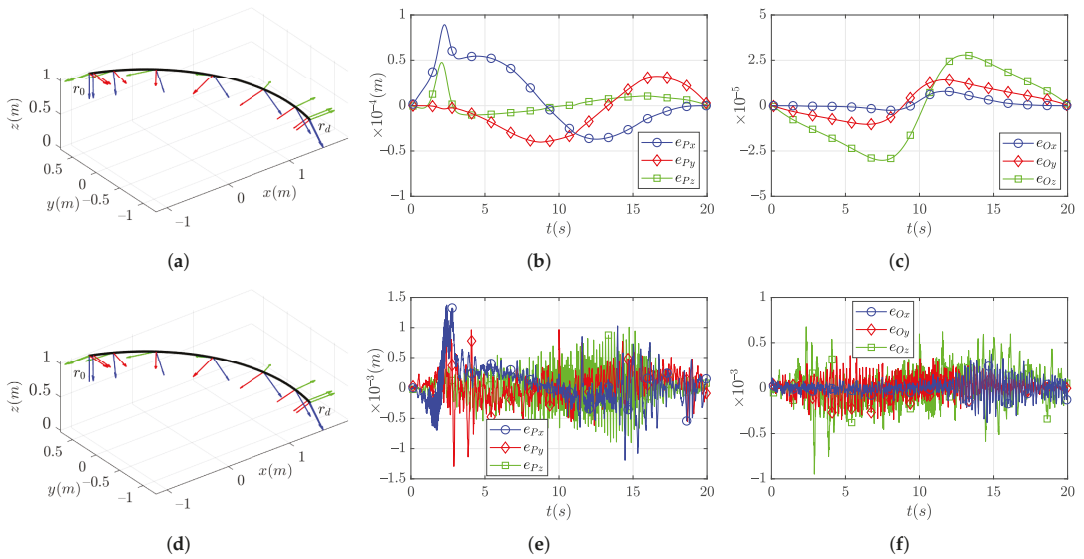


Figure 11. Tracking performance comparison between simulations and experiments for elliptic trajectory. End-effector’s trajectory in simulations (a) and experiments (d). Position errors in simulations (b) and experiments (e). Orientation errors in simulations (c) and experiments (f). Notice that the position and orientation errors, panels (b,c), respectively, are small in the simulations. In addition, notice that the trajectories in simulations (a) and experiments (b) are quite similar. The reason for the larger errors obtained in the experiments (panel (e,f)) are discussed in Section 5.5.

Figure 12 illustrates the experiment results. The manipulabilities of both the arm and the complete system are again improved at the end of the trajectory compared to the initial configuration, as shown in Figure 12b. However, we notice how both manipulabilities decreased during the time interval $t = (8, 15)(s)$. This behavior is due to the value of $\alpha(t)$ (manipulability maximization step size) being negative for this span of time in order to respect the joint velocity limits, as discussed in Section 4.3.2. Notice that v_p (Figure 12d) and \dot{z}_{pj} (Figure 12e) are kept at their maximum values during this span of time. It would not be possible to track this particular trajectory while respecting the joint velocity limits without stretching the arm; hence, the manipulabilities of the complete system and the arm decreased.

We observe in Figure 12c how the self-collision distance of the elbow gets close to zero. In this trajectory, the elbow needs to get close to the platform in order for the end-effector to track the desired trajectory. The motion planning algorithm gradually stopped the elbow to prevent the collision with the platform, as shown by the slow decay of d_{elbow} . To achieve this, the movement of the prismatic joint was restricted, as shown in Figure 12e. The wrist also moves towards the front of the platform, as shown by the decrement in d_{wrist} in Figure 12c. This potential self-collision is also prevented by limiting the movement of q_{a2} and q_{a3} , as shown in Figure 12f,g.

Figure 12d–i show that the joint limits, joint velocity limits and joint velocity boundary constraints are also satisfied. All the joints were kept within their respective limits, as shown in Figure 12e–h. As depicted in Figure 12d,e, the platform’s linear velocity and the prismatic joint’s velocity are kept within their limits. Once more, the velocity profiles are smooth and the initial and final velocities for all the joints are zero.

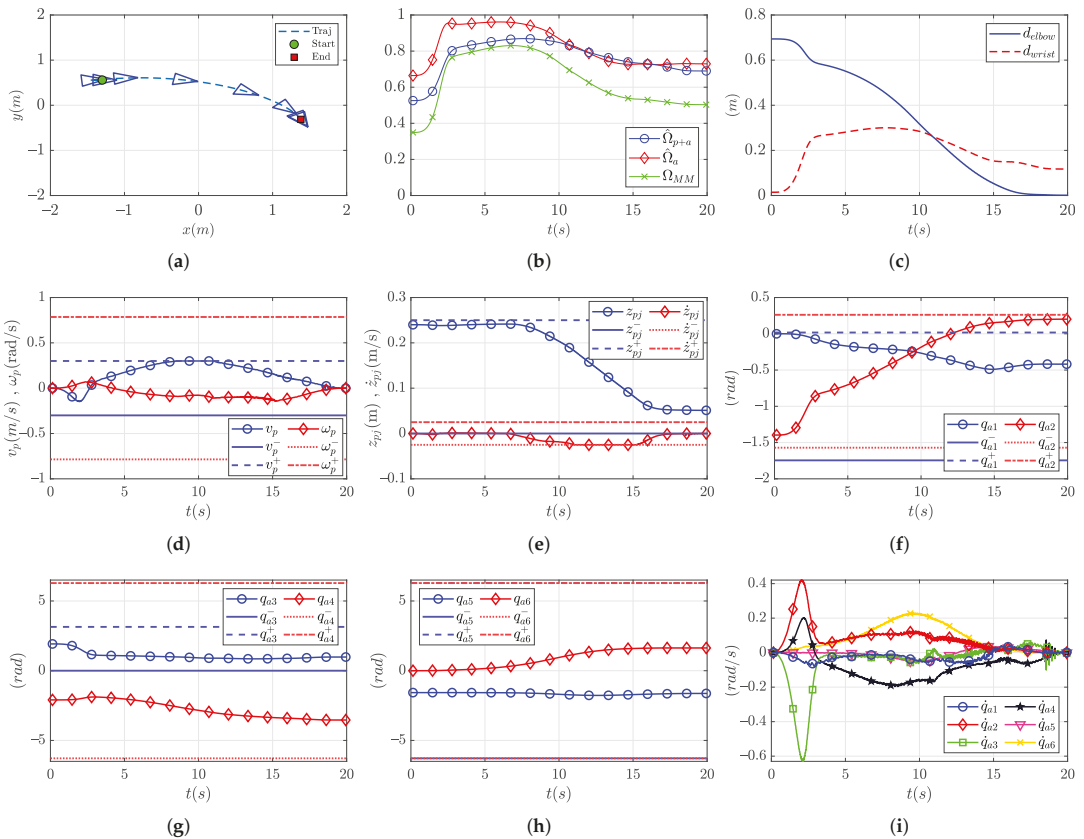


Figure 12. Experiment results for the elliptic trajectory tracking: (a) mobile platform’s trajectory; (b) normalized manipulability measures; (c) self-collision distance; (d) mobile platform’s velocity commands; (e) prismatic joint’s position and velocity; (f–h) arm’s joint positions; (i) arm’s joint velocities. Notice that all the joint limits and velocity limits are respected (panels (d–i)). The potential self-collisions described in Section 5.2 are prevented (panel (c)). Furthermore, the manipulabilities of both the arm and the whole system are improved (panel (b)). See Section 5.5 for a detailed discussion of this figure.

6. Discussion

A scheme was proposed to solve the motion planning of NMMs considering joint limits, joint velocity limits, joint velocity boundary constraints, and self-collision avoidance while maximizing the manipulabilities of both the robot arm and the whole system.

The proposed solution uses a weighted input velocity vector and a weighted Jacobian to penalize the movement of joints that get close to a position constraint. A proposed quadratic cost function is minimized when solving the motion planning problem for redundant NMMs. This cost function includes a secondary task to be satisfied that is also weighted to comply with the position constraints. The maximization of an introduced manipulability measure for mobile manipulators is used as the secondary task to push the system away from singularities. In the experiments section, it is demonstrated that maximizing this new measure simultaneously improves the manipulability of the whole system and that of the manipulator alone.

This work focuses on the motion planning for trajectory tracking at the kinematic level, which must not only comply with joint position constraints, but must also respect joint velocity constraints and joint velocity boundary constraints. Joint velocity boundary

constraints are satisfied by varying the magnitude of the homogeneous solution at the start and end of the trajectory. The manipulability maximization at these points is not necessarily zero; hence, using an adequate variation in the magnitude of self-motion is needed. Joint velocity limits are satisfied by evaluating the maximum allowable self-motion for each joint. Using this information, the step size of the gradient ascent/descent is limited when required, and consequently, the joint velocity limits are not exceeded.

The experiments for 6-DOF trajectories were conducted to verify the efficacy of the proposed scheme. The results demonstrate that the proposed approach can solve the motion planning problem for NMMs to perform trajectory tracking at the kinematic level while considering the constraints required for real implementation including manipulation capability preservation or improvement.

The experiments designed in Section 5 consider an open environment without obstacles, because this is the scope of our manuscript. However, the proposed solution can be extended to prevent collisions with obstacles by including collision pairs between the robot arm and these obstacles, using the same definitions as in Section 4.2.2. This will penalize the movement of the arm's joints that get closer to an obstacle in the environment. Nevertheless, in the case of the platform, stopping it is not an efficient approach. In this case, an additional task can be added to the solution to push the platform away from the obstacles. One way to achieve this is by using a task priority scheme [24] using the nullspace of the motion planning algorithm.

Even though our work focuses on NMMs, our approach can be effortlessly adapted for use with holonomic mobile manipulators. Future work will focus on dynamic modeling and controller design to deal with system vibrations and tire slip such that the tracking errors of the system can be reduced.

Author Contributions: Conceptualization, methodology, J.L. and T.H.; software, validation, visualization, writing—original draft preparation, J.L.; supervision, project administration, funding acquisition, resources, writing—review and editing, T.H. Both authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Ministry of Science and Technology, Taiwan (Project code MOST 107-2923-E-009-004-MY3).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors wish to thank the Mechanical and Mechatronics Systems Research Laboratories from the Industrial Technology Research Institute (ITRI), Taiwan, for the support and resources provided for this work.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

References

1. Siciliano, B.; Sciavicco, L.; Villani, L.; Oriolo, G. *Robotics: Modelling, Planning and Control*, 2009th ed.; Springer: London, UK, 2009; p. 632.
2. Liao, J.; Huang, F.; Chen, Z.; Yao, B. Optimization-based motion planning of mobile manipulator with high degree of kinematic redundancy. *Int. J. Intell. Robot. Appl.* **2019**, *3*, 115–130. [[CrossRef](#)]
3. Santos, P.C.; Freire, R.C.S.; Carvalho, E.A.N.; Molina, L.; Freire, E.O. M-FABRIK: A New Inverse Kinematics Approach to Mobile Manipulator Robots Based on FABRIK. *IEEE Access* **2020**, *8*, 208836–208849. [[CrossRef](#)]
4. Campion, G.; Bastin, G.; Dandrea-Novel, B. Structural properties and classification of kinematic and dynamic models of wheeled mobile robots. *IEEE Trans. Robot. Autom.* **1996**, *12*, 47–62. [[CrossRef](#)]
5. Bayle, B.; Fourquet, J.Y.; Renaud, M. Manipulability of Wheeled Mobile Manipulators: Application to Motion Generation. *Int. J. Robot. Res.* **2003**, *22*, 565–581. [[CrossRef](#)]
6. Bayle, B.; Renaud, M.; Fourquet, J.Y. Nonholonomic mobile manipulators: Kinematics, velocities and redundancies. *J. Intell. Robot. Syst. Theory Appl.* **2003**, *36*, 45–63. [[CrossRef](#)]

7. De Luca, A.; Oriolo, G.; Giordano, P.R. Kinematic modeling and redundancy resolution for nonholonomic mobile manipulators. *Proc. IEEE Int. Conf. Robot. Autom.* **2006**, *2006*, 1867–1873.
8. Huang, Q.; Tanie, K.; Sugano, S. Coordinated Motion Planning for a Mobile Manipulator considering Stability and Manipulation. *Int. J. Robot. Res.* **2000**, *19*, 732–742. [[CrossRef](#)]
9. Jia, Y.; Xi, N.; Cheng, Y.; Liang, S. Coordinated motion control of a nonholonomic mobile manipulator for accurate motion tracking. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 1635–1640.
10. Zhang, Y.; Yan, X.; Chen, D.; Guo, D.; Li, W. QP-based refined manipulability-maximizing scheme for coordinated motion planning and control of physically constrained wheeled mobile redundant manipulators. *Nonlinear Dyn.* **2016**, *85*, 245–261. [[CrossRef](#)]
11. Chan, T.F.; Dubey, R.V. A Weighted Least-Norm Solution Based Scheme for Avoiding Joint Limits for Redundant Joint Manipulators. *IEEE Trans. Robot. Autom.* **1995**, *11*, 286–292. [[CrossRef](#)]
12. Yoshikawa, T. *Foundations of Robotics: Analysis and Control*; MIT Press: Cambridge, MA, USA, 1990.
13. Lee, J. A study on the manipulability measures for robot manipulators. In Proceedings of the 1997 IEEE/RSJ International Conference on Intelligent Robot and Systems. Innovative Robotics for Real-World Applications. IROS '97, Grenoble, France, 11 September 1997; Volume 3, pp. 1458–1465.
14. Patel, S.; Sobh, T. Manipulator Performance Measures—A Comprehensive Literature Survey. *J. Intell. Robot. Syst.* **2015**, *77*, 547–570. [[CrossRef](#)]
15. Staffetti, E.; Bruyninckx, H.; De Schutter, J. *Advances in Robot Kinematics: Theory and Applications*; Lenarčič, J., Thomas, F., Eds.; Springer: Dordrecht, The Netherlands, 2002; p. 536.
16. Dariush, B.; Zhu, Y.; Arumbakkam, A.; Fujimura, K. Constrained closed loop inverse kinematics. In Proceedings of the 2010 IEEE International Conference on Robotics and Automation, Anchorage, AK, USA, 3–7 May 2010; pp. 2499–2506.
17. Zghal, H.; Dubey, R.V.; Euler, J.A. Efficient gradient projection optimization for manipulators with multiple degrees of redundancy. In Proceedings of the IEEE International Conference on Robotics and Automation, Cincinnati, OH, USA, 13–18 May 1990; pp. 1006–1011.
18. Euler, J.A.; Dubey, R.V.; Babcock, S.M. Self motion determination based on actuator velocity bounds for redundant manipulators. *J. Robot. Syst.* **1989**, *6*, 417–425. [[CrossRef](#)]
19. Chiaverini, S.; Siciliano, B. The unit quaternion: A useful tool for inverse kinematics of robot manipulators. *Syst. Anal. Model. Simul.* **1999**, *35*, 45–60.
20. Dam, E.B.; Koch, M.; Lillholm, M. *Quaternions, Interpolation and Animation*; Technical Report DIKU-TR-98/5; Department of Computer Science, University of Copenhagen: Copenhagen, Denmark, 1998; p. 103.
21. Funda, J.; Taylor, R.; Paul, R. On homogeneous transforms, quaternions, and computational efficiency. *IEEE Trans. Robot. Autom.* **1990**, *6*, 382–388. [[CrossRef](#)]
22. Yuan, J. Closed-loop manipulator control using quaternion feedback. *IEEE J. Robot. Autom.* **1988**, *4*, 434–440. [[CrossRef](#)]
23. Shahbazi, M.; Kashiri, N.; Caldwell, D.; Tsagarakis, N. Orientation planning in task space using quaternion polynomials. In Proceedings of the 2017 IEEE International Conference on Robotics and Biomimetics (ROBIO), Macau, Macao, 5–8 December 2017; pp. 2343–2348.
24. Flacco, F. The tasks priority matrix: A new tool for hierarchical redundancy resolution. In Proceedings of the 2016 IEEE-RAS 16th International Conference on Humanoid Robots (Humanoids), Cancun, Mexico, 15–17 November 2016; pp. 1–7.

Article

A Robot-Assisted Large-Scale Inspection of Wind Turbine Blades in Manufacturing Using an Autonomous Mobile Manipulator

Heiko Engemann ^{1,2,*}, Patrick Cönen ¹, Harshal Dawar ¹, Shengzhi Du ^{2,*} and Stephan Kallweit ¹

- ¹ IaAM Institute, Faculty of Mechanical Engineering and Mechatronics, University of Applied Sciences Aachen, 52074 Aachen, Germany; coenen@fh-aachen.de (P.C.); dawar@fh-aachen.de (H.D.); kallweit@fh-aachen.de (S.K.)
- ² Faculty of Engineering and Built Environment, Tshwane University of Technology, Pretoria 0001, South Africa
- * Correspondence: engemann@fh-aachen.de (H.E.); dus@tut.ac.za (S.D.)

Abstract: Wind energy represents the dominant share of renewable energies. The rotor blades of a wind turbine are typically made from composite material, which withstands high forces during rotation. The huge dimensions of the rotor blades complicate the inspection processes in manufacturing. The automation of inspection processes has a great potential to increase the overall productivity and to create a consistent reliable database for each individual rotor blade. The focus of this paper is set on the process of rotor blade inspection automation by utilizing an autonomous mobile manipulator. The main innovations include a novel path planning strategy for zone-based navigation, which enables an intuitive right-hand or left-hand driving behavior in a shared human–robot workspace. In addition, we introduce a new method for surface orthogonal motion planning in connection with large-scale structures. An overall execution strategy controls the navigation and manipulation processes of the long-running inspection task. The implemented concepts are evaluated in simulation and applied in a real-use case including the tip of a rotor blade form.

Keywords: mobile manipulation; large-scale inspection; wind turbine production; autonomous navigation; surface-orthogonal path planning; intelligent robot; flexible production

Citation: Engemann, H.; Cönen, P.; Dawar, H.; Du, S.; Kallweit, S. A Robot-Assisted Large-Scale Inspection of Wind Turbine Blades in Manufacturing Using an Autonomous Mobile Manipulator. *Appl. Sci.* **2021**, *11*, 9271. <https://doi.org/10.3390/app11199271>

Academic Editors: António Paulo Moreira, Pedro Neto and Félix Vidal

Received: 14 September 2021
Accepted: 29 September 2021
Published: 6 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Wind energy has gradually taken the dominant share of renewable energies. The worldwide capacity increased from 7600 MW in 1998 to 93 GW in 2020 [1]. The main components of a wind turbine are: a rotor equipped with wing-shaped blades, a nacelle that houses a drive train, and a tower [2]. The rotor blades transform the wind energy into rotary energy and must be lightweight, robust, and long-lasting. Therefore, they are made from composite materials, such as glass or carbon fiber material with a resin-like epoxy. During rotation, extremely high forces affect the blades. Therefore, it is important to avoid imperfections during the manufacturing process.

In the production of glass fiber reinforced structural components, the fiber structure is fixed by enclosing laid semi-finished glass fibers with a resin matrix [3]. Imperfections in the alignment of the structure or during the fiber reinforcement change the structural properties and thus reduce the quality of the composite material. Currently, such imperfections are detected with help of ultrasonic [4,5], thermal [6,7], or radar [8,9] techniques, whereby a differentiation has to be made in pre- and post-resin-injection inspections. If an imperfection is detected after the resin injection, no corrections are possible, and the component is, therefore, a reject. In research, radar imaging has gained a lot of attention for inspection tasks of fiber composite material. In contrast to other methods based on x-rays, thermal imaging, or ultrasonic imaging, radar imaging is non-invasive and provides a high resolution combined with a high penetration depth [10]. Millimeter wave radar scans can

be used to generate a detailed layer-by-layer visualization of a rotor blade [9]. Therefore, it is possible to locate imperfections in 3D. However, the underlying algorithms for the 3D reconstruction require a surface orthogonal sensor alignment and a low measurement uncertainty of the sensor pose during the scanning process. These requirements exclude a manual execution with handheld devices.

Figure 1 shows a schematic of the manufacturing of a rotor blade, including the two main components: aeroshells and shear webs. The typical diameter of a wind turbine rotor, including the blades, has increased from 54 m (2005) to 158 m (2017) for onshore and from 76 m (2005) to 164 m (2015) for offshore installations [11]. The huge dimensions of the rotor blades complicate the inspection processes, which is why they are mostly performed manually by human workers. The automation of the inspection processes has great potential to increase the overall productivity and to create a consistent reliable database for each individual rotor blade. Such an automation approach must be large-scaled and flexible to cover the high variety of rotor blade dimensions.

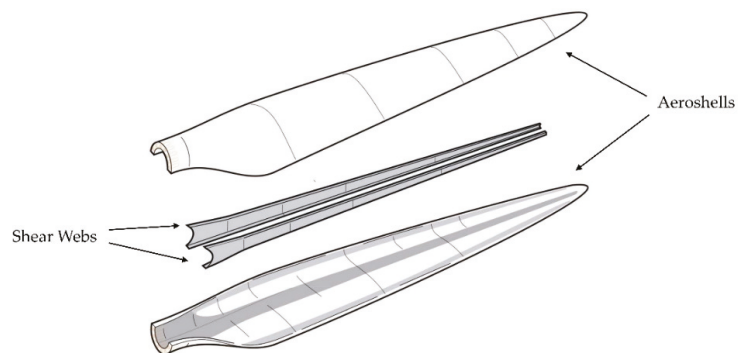


Figure 1. Schematic of the manufacturing of a wind turbine blade [12].

An *autonomous industrial mobile manipulator* (AIMM) [13] combines the flexibility of an industrial robot arm (manipulator) with the mobility of an autonomous mobile robot (AMR). Equipped with various sensors, AIMMs are capable of autonomous navigation, even in dynamic and large-scale production environments. Furthermore, the perceived data can be used to realize a shared human–robot workspace, which is of particular importance for manual labor-dominated production types. One of the first AIMMs was introduced in 1984: MORO [14]. MORO was capable of navigating on the shopfloor and executing pick and place tasks. Based on this pioneering work, many further developments were carried out, focusing on different industrial applications: part feeding [15–17], transportation and assembly [18–20], as well as large-space manufacturing [21–23]. In recent years, inspection robots have become more present in different domains, such as the oil and gas industry [24,25], the power industry [26,27], and civil infrastructure [28]. Therefore, the idea to utilize an AIMM for inspection tasks is obvious. In contrast to recent developments in the area of inspection robots, our approach focuses on the motion planning addressing the surface orthogonal sensor alignment and the special requirements of a shared human–robot workspace in an industrial context.

Therefore, we make use of the AIMM OMNIVIL [29]. OMNIVIL consists of a self-built mobile platform. The mobile platform was designed to address the needs in a dynamic production environment, such as positioning, accuracy, and maneuverability. Therefore, the platform is based on four Mecanum wheels [30], whereby a pivoting axle guarantees continuous ground contact. The collaborative manipulator UR5 is mounted on top of the mobile platform. The sensor concept includes various sensor types to perceive the environment and the internal state of the robot. The workspace monitoring concept and the localization and positioning capabilities of the mobile platform were evaluated in

various experiments ranging from static to highly dynamic scenarios [29]. The collaborative manipulator is equipped with an RGB-D camera (Intel-RealSense 435) and a radar module that works at 80 GHz with a 24 GHz bandwidth. Figure 2 shows the AIMM OMNIVIL.

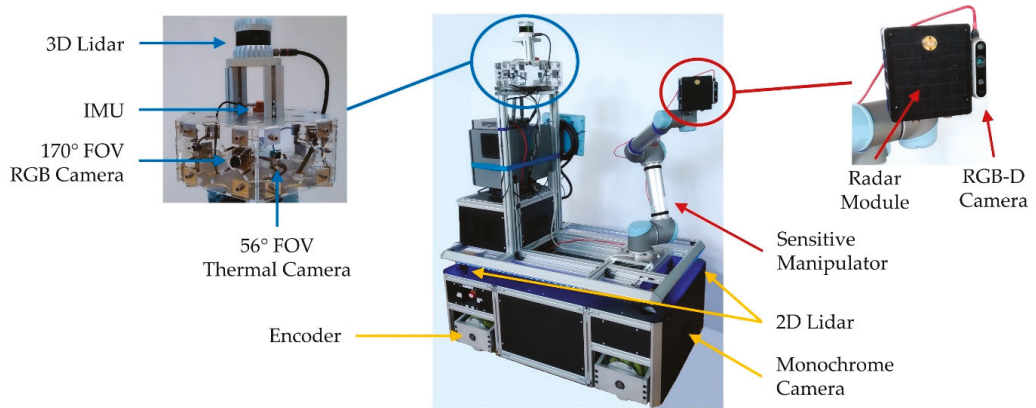


Figure 2. The AIMM OMNIVIL.

In this paper, we present an automated inspection process for wind turbine rotor blade manufacturing executed by the AIMM OMNIVIL. The focus is set on the robot control setup rather than the composite material analysis. Therefore, this work addresses motion planning, environmental perception, and the human–robot interaction. The following strategies are implemented towards a fully autonomous inspection of a rotor blade: (1) a novel path planning strategy for a zone-based navigation concept that enables an intuitive right or left driving behavior of the mobile platform without limiting the planning based navigation; (2) a novel method for surface orthogonal motion planning in connection with large scale structures; (3) therefore, the large-scale structures are divided into feasible subparts based on a workspace analysis, including the reachability and manipulability of the manipulator; (4) an overall execution strategy that controls the corresponding navigation and manipulation processes of the long-running inspection task.

The rest of the paper is structured as follows. Section 2 presents the control system, including the zone-based segmentation of the production environment, the path planning strategy, and the surface orthogonal motion planning. In Section 3, two experiments are carried out to validate the path planning strategy and the overall execution of the rotor blade inspection. Section 4 concludes the paper.

2. Robot Control System

2.1. Zone-Based Segmentation of the Production Environment

The general concept of zone-based navigation is inspired by zone management within an industrial production environment. The approach is based on virtual navigation zones [29]. Instead of using infrastructural markers, the zones can be defined in a given map by using 2D polygons. For each zone, a predefined robot behavior can be set with different parameter settings for maximum velocity, maximum acceleration, goal tolerance, warning indicators (visual and acoustic), or even different kinematic models reaching from the differential to holonomic models.

For instance, Figure 3a shows a zone-based segmentation of a 60×60 m production environment. The coloring represents the different cost levels of the zones ranging from green (minimum cost value) to red (maximum cost value). In Figure 3b, the zones are colored with an individual color for each zone. Figure 3c shows the corresponding layered costmap [31] configuration in hierarchical order. The ordering of the layers allows modulating the costs by overwriting them only when and where required.

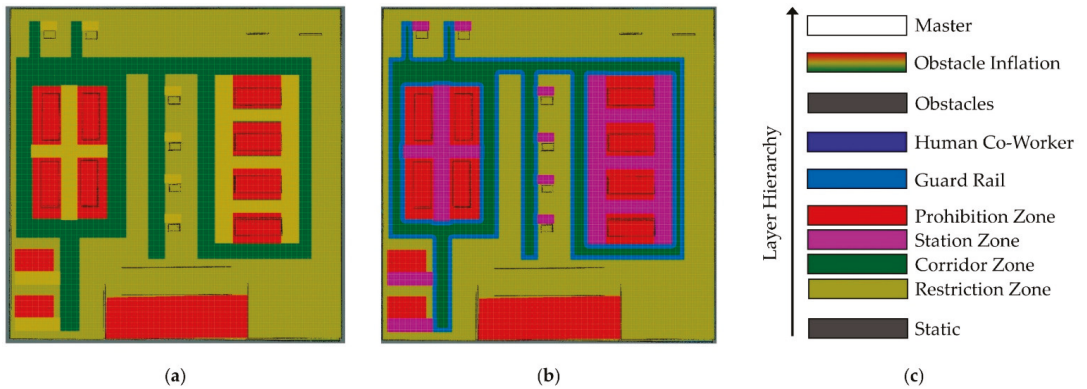


Figure 3. Zone-based segmentation of a production environment. (a) Zone-based segmentation of a 60 × 60 m environment colored from green (minimum cost value) to red (maximum cost value); (b) same zone setup, but with individual zone colors of layer hierarchy; (c) hierarchical ordered costmap layers.

The standard configuration of a layered costmap configuration includes a static, an obstacle, and an obstacle inflation layer. This default configuration does not fulfil the requirements of a dynamic production environment, including different manufacturing and transportation zones as well as a shared human–robot workspace. The implemented approach is based on the concept of layered costmaps described in [32], whereby each layer tracks the data to a specific functionality. With the cost value $\Omega_{xy}^i = \{0, 1, \dots, 254, 255\}$ for a cell c_{xy}^i at position $P_j^c = (x_j, y_j)$, the layer i contributes to the Master costmap as follows:

The Static layer contributes to the master costmap by analyzing an a priori created 2D occupancy grid map G . The layer provides information about free and occupied spaces in the production environment by Equation (1):

$$\Omega_{xy}^1 = \begin{cases} 0, & G(x, y) == Free \\ 254, & G(x, y) == Occupied \\ 255, & G(x, y) == NoInformation \end{cases} \quad (1)$$

with *Free*, *Occupied*, and *NoInformation* reflecting the related values of the occupancy grid map implementation.

The Zone layers, Corridor (2), Restricted (3), Station (4), and Prohibition (5), contribute to the zone-based navigation layout of the production environment. The dimensions of the zones are provided in form of a polygon list. The corridor zone should be the preferred zone for the implemented path planner; therefore, the cost value of the corresponding cells is set to a small value of $\Omega_{xy}^2 = 10$. The robot is allowed to enter restriction zones, but only if necessary. Therefore, the cost value for the corresponding cells is set to $\Omega_{xy}^3 = 100$. The same condition applies to the station zones. Since a motion of the robot through a restriction zone is from higher preference than a motion through a station zone, the cost value of the corresponding cells is set to $\Omega_{xy}^4 = 120$. The robot is not allowed to enter a prohibition zone; therefore, the cost value of the corresponding cells is set to $\Omega_{xy}^5 = 250$.

The Guard Rail layer is based on the inflation layer described in [32]. Originally, the inflation layer was designed to create a buffer zone around lethal obstacles to avoid the robot coming too close to the obstacles. We applied that method to define a buffer zone at the borders of the corridor zones. The robot uses these guard rail zones as a guide for navigating inside the corridor zones. The robot will navigate with a high preference alongside the edge between the guard rail zone and the corridor zone. The developed method is further explained in Section 2.2. The inflation radius is adjustable and defines the width of the guard rail zone. The corresponding cells are set to the cost value $\Omega_{xy}^6 = 12$.

The Human layer contributes the information of a multilayer and redundant workspace monitoring system (WMS) described in [29]. The WMS uses RGB and thermal images as well as Lidar data. The multilayer sensor setup is improved by the implementation of redundant algorithms for human co-worker detection based on neural networks. The fused confidence intervals between 0 and 1 are provided as a 2D heatmap in form of an occupancy grid map. The corresponding cells in the costmap M are called human cells and assigned with the cost value $\Omega_{xy}^7 = 200$. A threshold of 0.5 is used for the confidence level to neglect low confidence detections. An area with an adjustable radius around each human cell is defined as a human safety zone. The Human layer calculates the Euclidean distance between the current robot position and each human cell to determine if the robot is inside a human safety zone. Following the concept of the navigation zones, the robot will preventively change its motion behavior if it is inside a human safety zone. In addition, the human cells are inflated to create a buffer zone around the human co-workers. The radius of these buffer zones is smaller compared to the human safety zones. The cost value of the corresponding cells is set to $\Omega_{xy}^7 = 200$. Theoretically, the robot is allowed to plan close to a human, but it is highly cost inefficient. Figure 4 shows an exemplary scenario of an occupancy grid map provided by the WMS and the resulting costmap, including the buffer zones in blue and one visualized human safety zone in red.

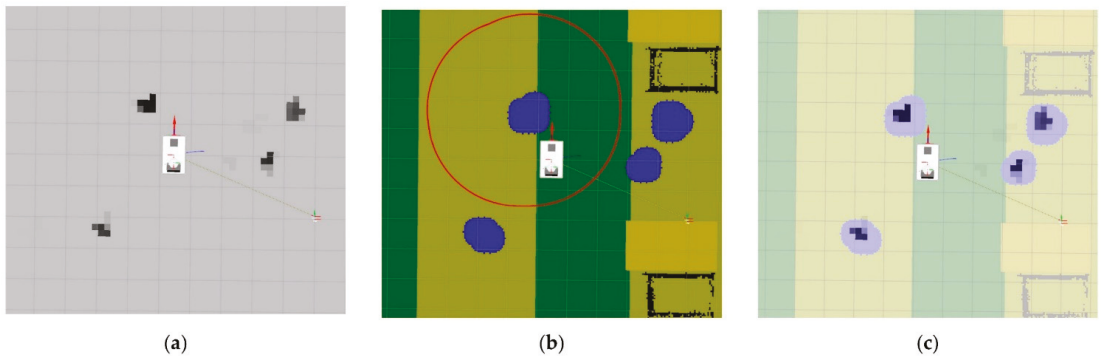


Figure 4. Human layer. (a) Occupancy grid map provided by the WMS; (b) costmap M with buffer zones around the human cells in blue and one exemplary visualized human safety zone (red circle); (c) costmap and occupancy grid map overlaid.

The Obstacle layer tracks the data from the 2D Lidar sensors. The Lidar data is provided in form of an array S , including the sensor readings. The sensor readings are converted into the costmap space to determine the corresponding cells. Analog to the Static layer, the obstacle layer contributes to the master layer by following Equation (2).

$$\Omega_{xy}^8 = \begin{cases} 0, & M_{Lidar}(x, y) == Free \\ 254, & M_{Lidar}(x, y) == Occupied \\ 255, & M_{Lidar}(x, y) == NoInformation \end{cases}, \quad (2)$$

where M_{Lidar} is the sensor readings in the costmap space. The Obstacle Inflation layer adds an adjustable buffer zone around the obstacle. Therefore, the distance between the obstacle and the planned path is increased.

2.2. Cost Adaption Based on Search Expansion Direction

Common industrial navigation concepts are based on line following strategies [33] using passive [34] or active [35] landmark detection. These methods are approved in industrial environments, but not as flexible as desired. The well-known planning algorithms A* [36] or Dijkstra [37] provide the most cost-efficient path available from a start position to a goal position, given a costmap. The presented navigation concept segments the production environment in zones of different cost levels. The resulting costmap is provided

to the path planner, resulting in a high preference for the corridor zone, without limiting the robot in the manner of a line follower. However, a line follower behavior is socially desirable for the corridor zone. This is particularly true for a shared human–robot production environment. In [38], the cell costs were slightly reduced for cells on the right side of a corridor. As a result, the pass speed of the human and the robot was significantly increased which indicates an optimized human–robot collaboration. The social behavior of right or left driving is widely used in public and industrial environments. Such a predictable and intuitive behavior of the robot is a key feature for beneficial human–robot cooperation.

Instead of manipulating the costs before the actual planning process, our approach is applied while planning and takes the expansion direction of the planning algorithm into concern. As a result, the desired left or right driving behavior can be applied to any zone and in any direction. The aim is that the global planner prefers a path right next to the previously mentioned guard rail zone. This behavior is comparable to a line follower, without limiting the robot to a particular line motion. Figure 5 shows a horizontal and a vertical example scenario of a planning procedure in a grid-based costmap.

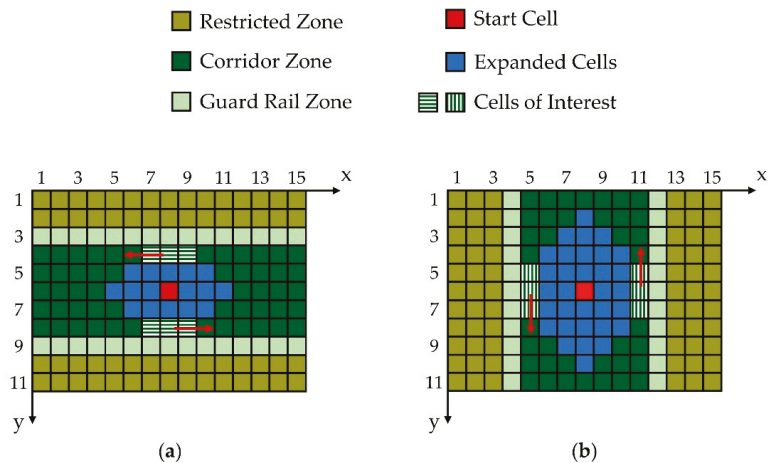


Figure 5. Exemplary planning scenario. (a) Horizontal; (b) vertical.

The start cell for the planning procedure is set to the middle of the corridor zone. During the search for a valid path, a check is performed for each expanded cell of the corridor zone; if the cell is a neighbor of the guard rail zone, the cell is called a cell of interest. Each cell of interest can be either declared as a left-hand or right-hand driving cell. The cell type is determined by taking the expansion direction of the search algorithm into concern. The expansion direction can only be determined if a neighbored cell of interest exists. In Figure 5a, the cells at positions (7,4) and (9,8) are right-hand driving cells. The expansion direction is visualized with a red arrow. The cells at positions (9,4) and (7,8) are left-hand driving cells. The cells at (8,4) and (8,8) do not feature a specific type, since they had no neighbored cell of interest when they were expanded. In Figure 5b, the cells at (5,7) and (11,5) are right-hand driving cells. The cells at (5,5) and (11,7) are left-hand driving cells, and the cells at (5,6) and (11,6) of no type.

During search expansion the path potential is stored in form of a grid-based potential map Φ . The potential map Φ holds the information for each expanded cell of how much it costs to reach that cell from the start cell. Cells that are not expanded yet, are set to a clearly identifiable default value φ . Algorithm 1 shows our approach applied to realize a right-hand driving behavior.

Algorithm 1. Calculation of Potential Map for right-hand Driving

```

1: function calculateCellPotential(cell  $c$ , costmap  $M$ , potential_map  $\Phi$ )
2: if  $c$  is not inside the corridor zone then
3:    $\Phi(c.x, c.y) = \text{calculateCostsToReachCell}(c, M, \Phi)$ 
4: end if
5: if  $M(c.x, c.y - 1)$  equal cost of GuardRailZone and  $\Phi(c.x + 1, c.y)$  not equal  $\varphi$  then
6:    $\Phi(c.x, c.y) = \Phi(c.x + 1, c.y) + \text{corridor\_cell\_cost} / 4$ 
7: else if  $M(c.x, c.y + 1)$  equal cost of GuardRailZone and  $\Phi(c.x - 1, c.y)$  not equal  $\varphi$  then
8:    $\Phi(c.x, c.y) = \Phi(c.x - 1, c.y) + \text{corridor\_cell\_cost} / 4$ 
9: else if  $M(c.x - 1, c.y)$  equal cost of GuardRailZone and  $\Phi(c.x, c.y - 1)$  not equal  $\varphi$  then
10:   $\Phi(c.x, c.y) = \Phi(c.x, c.y - 1) + \text{corridor\_cell\_cost} / 4$ 
11: else if  $M(c.x + 1, c.y)$  equal cost of GuardRailZone and  $\Phi(c.x, c.y + 1)$  not equal  $\varphi$  then
12:   $\Phi(c.x, c.y) = \Phi(c.x, c.y + 1) + \text{corridor\_cell\_cost} / 4$ 
13: else
14:   $\Phi(c.x, c.y) = \text{calculateCostsToReachCell}(c, M, \Phi)$ 
14: end if
15: end function

```

In this example, the right-hand driving behavior is applied to the corridor zone. The function *calculateCostsToReachCell* symbolizes the standard calculation of an expanded cell in the potential map, as shown in Equation (3):

$$\text{potential} = \Phi(c_{prev.x}, c_{prev.y}) + M(c_{exp.x}, c_{exp.y}) + \Omega_{mv}, \quad (3)$$

where c_{exp} is the expanded cell, c_{prev} is the previous cell, and Ω_{mv} is the cost value to move from one cell to a neighboring cell. The function is triggered in case the expanded cell is not part of the corridor zone or the expanded cell is not determined as a right-hand driving cell. Therefore, in these cases, the default behaviors of the global planners are not changed.

If the expanded cell is part of a corridor zone and neighbors a guard rail zone, the cell is of interest and further analyzed. Four cases are differentiated:

1. The guard rail zone is a neighbor in the negative y-direction in the costmap;
2. The guard rail zone is a neighbor in the positive y-direction in the costmap;
3. The guard rail zone is a neighbor in the negative x-direction in the costmap; or
4. The guard rail zone is a neighbor in the positive x-direction in the costmap.

Depending on the relative position of the expanded cell in relation to the guard rail zone and the already expanded cells stored in the potential P , the cell is defined as a right-hand driving cell. In the case of a right-hand driving cell, the calculation of the cell potential is changed from Equation (3). The applied formula is shown in Equation (4):

$$\text{potential} = \Phi(x_i, y_i) + M_{\text{Corridor}} / 4 \quad (4)$$

The coordinates x_i, y_i , with $i = \{1, 2, 3, 4\}$, reflect the above mentioned four cases. M_{Corridor} is the cost value of a cell inside the corridor zone.

The presented approach can be applied to any zone in any direction, as long as a particular cost gradient is present between the zone and the guard rail zone. Figure 6 shows an example scenario for a large-scale planning scenario in a simulated production environment. The red dot represents the start position and the green dot the goal position. The costmap is divided into navigation zones according to Section 2.1. Figure 6a shows the default behavior of the implemented A* planner. As desired, the resulting path is mostly located inside the corridor zone. Figure 6b shows the resulting right driving behavior by applying our approach.

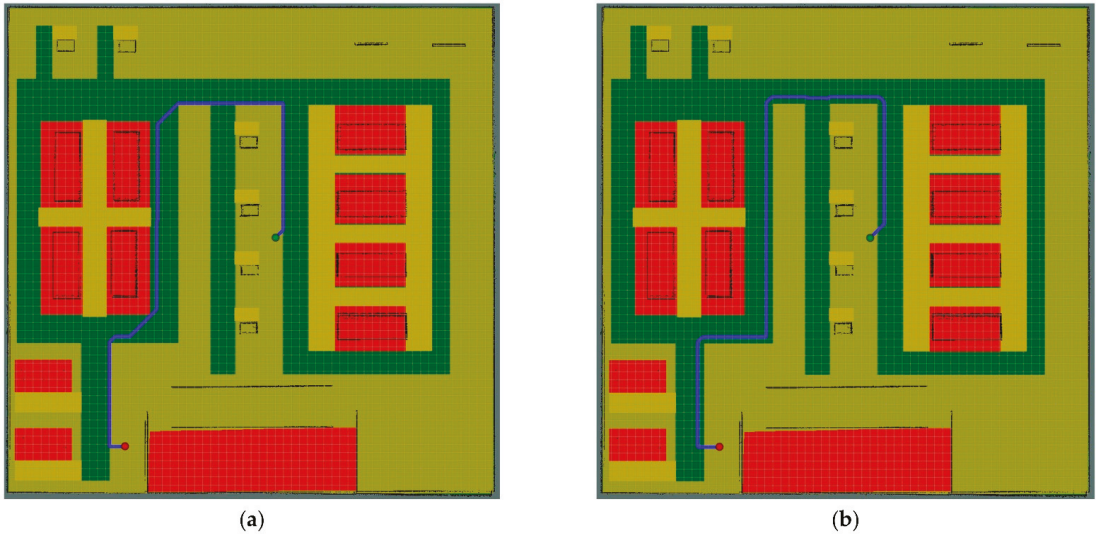


Figure 6. A* based global path planning in a 60×60 m sized grid-based costmap. (a) Default behavior; (b) right driving behavior.

2.3. Large-Scale Surface Orthogonal Motion Planning

The addressed inspection task requires a high positioning accuracy of the radar module ($<200 \mu\text{m}$) in reference to the workpiece frame for each scan process. Therefore, the large-scale inspection task is executed in asynchronous mobile manipulation mode. The workpiece is divided into smaller subsegments, which can be inspected standalone with the motion capabilities of the manipulator. At each subsegment, the following process steps are performed:

1. Surface reconstruction;
2. Sensor waypoint generation;
3. Motion planning.

The approached positions of the mobile platform are determined by analyzing the reachability of the manipulator in its workspace and segmenting the workpiece accordingly.

2.3.1. Surface Reconstruction

To plan a path on a surface, the geometrical shape of the surface must be known. Especially for inspection tasks, it cannot be assumed that the actual state of the workpiece still corresponds to its computer-aided design (CAD) data. In our implementation, the surface is reconstructed from a point cloud using the Point Cloud Library [39] (PCL). The point cloud is provided by the RGB-D camera at the end effector of the manipulator. In the first step, the point cloud is down-sampled, and then the surface is reconstructed by applying B-spline fitting [40]. Figure 7 shows the reconstructed surface of an exemplary rotor blade segment.

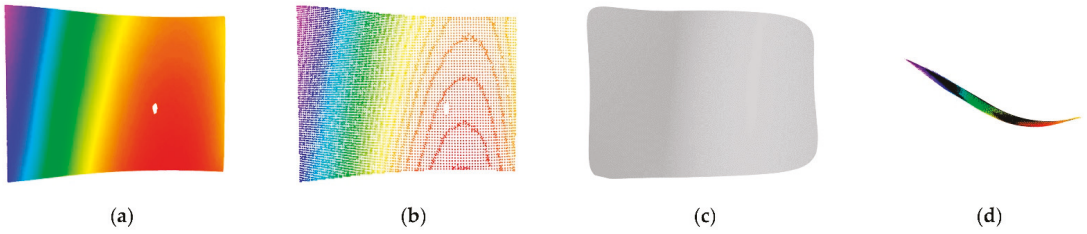


Figure 7. Surface reconstruction. (a) Dense point cloud of a rotor blade segment; (b) point cloud after down-sampling; (c) reconstructed surface (front view); (d) reconstructed surface (top view).

2.3.2. Waypoint Generation

The surface orthogonal motion planning is inspired by the approach presented in [41], which addresses an inspection task of a landscape with an unmanned aerial vehicle (UAV). For this purpose, the landscape is rasterized into a grid. The grid resolution depends on the technical parameters of the sensor, the desired image resolution, and the desired overlaps of the individual images. The centers of the grid cells are shifted along the normal of the ground surface. This approach was adapted for the creation of 6D waypoints, representing surface orthogonal 6D poses of the radar sensor. Figure 8 shows the three steps of the implemented waypoint generation method, applied to the reconstructed surface of Figure 7. Figure 8a shows the centers of the grid cells. These are projected onto the reconstructed surface (see Figure 8b). The projected centers are shifted along the surface normal to the desired distance from the surface. The z-axis of the 6D waypoints are aligned with the negative surface normals (see Figure 8c).

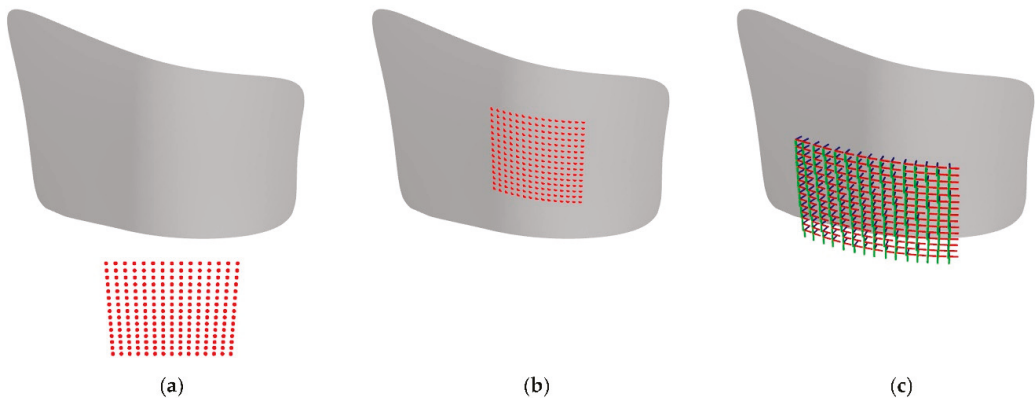


Figure 8. Grid-based waypoint generation. (a) Centers of the grid cells; (b) projection of grid onto the reconstructed surface; (c) 6D waypoints.

2.3.3. Path Planning

The 6D waypoints can be considered as nodes in a complete graph. A Hamiltonian cycle [42] in this graph visits all waypoints. The search for the shortest Hamiltonian cycle in a complete and weighted graph is called the traveling salesman problem (TSP). This problem is NP-complete, and there is no known efficient algorithm to solve it [43]. One approximation approach is the algorithm according to Christofides [44]. The implemented approach is shown in Algorithm 2.

Algorithm 2. Factor 3/2 approximation of the travelling salesman problem

- 1: **function** approximateTSP(Graph G , costfunction Ω)
 - 2: choose a root node $r \in V(G)$ as base v
 - 3: calculate the minimum spanning tree T for G with $MST - Prim(G, \Omega, r)$
 - 4: calculate the perfect matching with minimum weight m for odd $v \in V(G)$
 - 5: add $E(m)$ to T
 - 6: determine the Euler cycle Γ in $m + T$
 - 7: the Hamiltonian cycle H is the ordered list of nodes visited on Γ , multiple nodes are skipped
 - 8: return H
 - 9: **end function**
-

The path length is limited to a maximum of one and a half times the optimal solution. The algorithm first calculates a minimum spanning tree [43] (MST). A minimum perfect match is formed between the nodes of the MST with an odd degree. This is possible because there is always an even number of nodes with an odd degree. In the path planning implementation, the Blossom-V implementation from [45] is used for this purpose. The edges of the matching are added to the MST, such that the degree of all nodes is even. Thus, a Euler cycle can be formed in the graph. By truncation, i.e., deleting multiple visited nodes, the approximate solution of the TSP is formed.

2.3.4. Positioning of the Mobile Platform

The mobile platform must be positioned in such a way that the surface orthogonal planned path of the end effector lies in a workspace region with high reachability. The method used to determine the characteristics of the manipulator’s workspace is described in [29]. The size of the chosen suitable area of the manipulator workspace directly affects the segmentation of the workpiece in local subsegments.

The workpiece segmentation consists of the following steps:

1. Create the scanning grid accordingly to the workpiece size (see Figure 9a);
2. Project the scanning grid onto the surface of the workpiece (see Figure 9b);
3. Reflect the projected points alongside the surface normal (see Figure 9c); and
4. Cluster the reflected points into processable local scan areas (see Figure 9d).

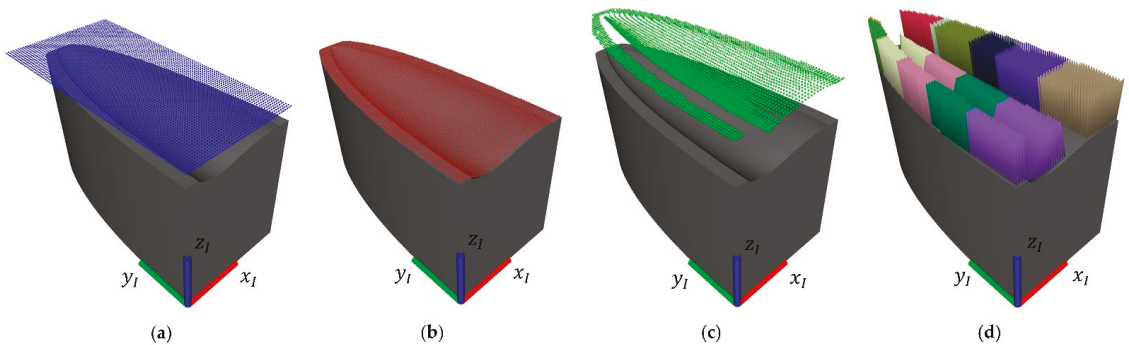


Figure 9. Process of workpiece segmentation. (a) Scanning grid; (b) projection onto surface; (c) reflection alongside surface normal; (d) clustering into local scan areas.

The reflected points represent positions of the manipulator’s end-effector. The aim of the workpiece segmentation is to find a set of neighbored projected points, whereby the corresponding reflected points are inside the defined workspace of the manipulator. Therefore, a list of position tuples Λ_{PR} is created, whereby each tuple consists of a projected point and the corresponding reflected point. By continuously expanding in the domain of the projected points and checking the reachability of the corresponding reflected points,

the tuples of list Λ_{PR} are clustered into suitable local scan areas. The implemented method is described in detail in Algorithm 3.

Algorithm 3. Segmentation of the workpiece into subsegments

```

1: function segmentWorkpiece(CADModel  $\Psi$ , Workspace  $Y$ , Distance  $d$ , ClusterList  $\Lambda_{PR}$ )
2: create sampling grid  $S$  accordingly to 3D bounding box size of  $\Psi$ 
3: project grid points  $s_i \in S$  onto surface of  $\Psi$ 
4: reflect projected points alongside the corresponding surface normal to distance  $d$  from surface of  $\Psi$ 
5: create a list  $L_{PR}$  with corresponding pairs of projected and reflected points
6: ascendingly order  $L_{PR}$  based on Euclidean distance between projected point and workpiece frame  $I$ 
7: while  $L_{PR}$  not empty do
8: initialize empty cluster  $C_{PR}$ 
9: add position pair at first position of  $L_{PR}$  to  $C_{PR}$  and remove pair from  $L_{PR}$ 
10: initialize filter dimensions
11: set current search direction to  $I_x$ -direction
12: while not all search directions are exhausted do
13: extend filter in search direction
14: if filter dimension is beyond the bounding box of  $\Psi$  then
15: mark current search direction as exhausted
16: reset filter dimensions
17: end if
18: find all position pairs  $L_i$  in  $L_{PR}$  where the reflected point is inside filter bounds
19: calculate bounding box dimension  $B$  of all reflected points  $\in L_i$ 
20: if  $B \subseteq Y$  then
21: add  $L_i$  to  $C_{PR}$  and delete  $L_i$  from  $L_{PR}$ 
22: else
23: reset filter dimension
24: mark search direction as exhausted
25: end if
26: switch search direction
27: end while
28: add  $C_{PR}$  to  $\Lambda_{PR}$ 
29: end while
30: end function

```

The mobile platform is positioned in such a way that the center of the calculated scan area $l_i = (x_I, y_I)^T$ is located at the center of the system's reachable workspace $r_i = (x_R, y_R)^T$. Therefore, the centers c_i are translated alongside the x-axes of the workpiece coordinate system I . Depending on the approach direction of the mobile platform, the translation is performed in x^+ - or x^- -direction. The shifted positions l_i^* are converted into 2D poses $L_i = (x_I, y_I, \theta_I)^T$ by adding an orientation accordingly to the translation direction. The 2D poses are transformed into the world coordinate system W , resulting in the required 2D goal poses for the navigation task.

2.4. Task Management

The inspection of a large-scale rotor blade is a long-running task. Therefore, we designed a task management system in form of a nested state machine. The state machine uses the SMACH framework [46]. The *main* state machine includes two nested state machines managing the manipulation and the navigation task. Figure 10 shows the task management system.

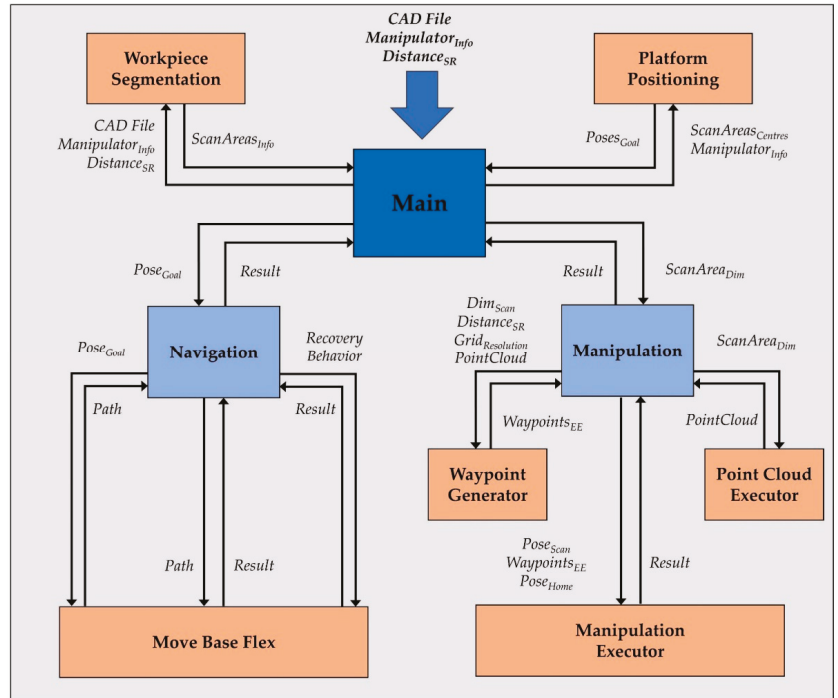


Figure 10. Task management system. State machines in blue and ROS nodes in orange.

The main state machine requires as input the model of the workpiece (CAD File), the desired distance between the radar sensor and the surface of the workpiece ($Distance_{SR}$) and the characteristics of the used manipulator ($Manipulator_{Info}$). The $Manipulator_{Info}$ consists of the suitable area of the manipulator workspace, including the corresponding maximum reach of the manipulator.

The workpiece segmentation and the position determination of the platform at the local scan subsegments is executed a priori. The workpiece segmentation provides the local scan areas ($ScanAreas_{Info}$). Each $ScanArea_{Info}$ consists of the area center ($ScanArea_{Center}$) and the dimensions ($ScanArea_{Dim}$) in reference to the workpiece coordinate system. The *platform positioning* converts the centers of the local scan areas into 2D navigation goals ($Poses_{Goal}$) in reference to the world coordinate system.

The navigation and manipulation state machines are triggered sequentially for each pair of navigation goal ($Pose_{Goal}$) and local scan area ($ScanArea_{Dim}$). The navigation state machine takes care about the path planning and path execution processes. The move base flex [47] framework is implemented to change between different motion behaviors of the mobile platform depending on the current navigation zone. The manipulation state machine is divided into three process steps. The *point cloud executor* gathers the point cloud data at the local scan pose ($Pose_{Scan}$) and crops it accordingly to the dimensions of the local scan area. The *waypoint generator* determines the 6D end-effector waypoints ($Waypoints_{EE}$), orthogonal oriented to the surface of the workpiece. The manipulation executor controls the motion of the manipulator.

3. Experiments and Discussions

3.1. Production Environment Navigation

We evaluated the presented zone-based navigation concept in the multi-robot simulator Gazebo [48]. Therefore, we created 10 different production environments of the size

60 × 60 m. Each simulated production environment features an individual navigation zone layout, comparable to the layout shown in Figure 6. In each simulated production environment, we defined 10 start-goal-position tuples, which resulted in a total of 100 tuples. The tuples were divided into five groups:

- Tuples T_{RR} from restricted-to-restricted zone;
- Tuples T_{SS} from station-to-station zone;
- Tuples T_{SC} from station-to-corridor zone;
- Tuples T_{CS} from corridor-to-station zone;
- Tuples T_{CC} from corridor-to-corridor zone.

The five groups are equally distributed across the 10 simulated environments, resulting in two tuples for each group per environment.

For each tuple, we manually annotated the desired path P_i with $i = \{1, 2, \dots, 100\}$. Each path P_i fulfills two criteria. The first one is the reduction in the overall costs of the path by preferring the corridor zone. The second one respects the desired right-hand driving behavior alongside the corner between the corridor zone and the guard rail zone.

The experiment compares the performance of the default A* and Dijkstra implementation of the robot operating system (ROS) [49] with our adapted version, later called A*_{zone} and Dijkstra_{zone}.

The deviation between a planned path A and the corresponding manually annotated path $B \in P_i$ is represented by the average mean square error $aMSE$. The calculation is given in Equation (5):

$$aMSE = \frac{\sum_{i=0}^n d_i^2}{n}, \tag{5}$$

where d_i is the Euclidean distance between each of the n positions $a_i \in A$ and its closest neighbored position $b_i \in B$. Figure 11 shows the experimentally determined results for each of the 10 simulated production environments averaged over all five start-goal-position tuple groups.

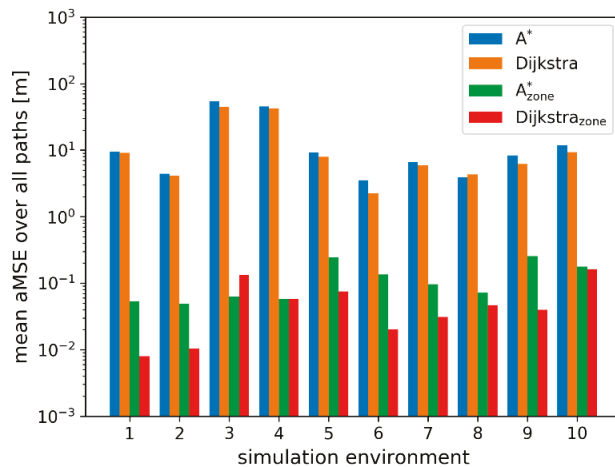


Figure 11. Average mean square error over all paths for each simulated environment.

As expected, the $aMSE$ is significantly higher for the default implementations of A* and Dijkstra. This result is caused by the right-hand driving criteria of the manually annotated reference paths. In addition to the $aMSE$, we analyzed the following performance parameters:

1. The required process time;
2. The number of expanded cells; and

3. The path length.

The required process time indicates the additional computation effort caused by the implementation of Algorithm 1. The number of expanded cells reflects the efficiency of reaching the goal cell. The overall path length shows the additional path caused by the desired right driving behavior. Table 1 shows the performance parameters averaged over all 100 start-goal-position tuples.

Table 1. Comparison of the evaluated planning approaches. Performance parameters averaged over 100 start-goal-position tuples in 10 simulated environments.

Approach	aMSE [m]	Processing Time [ms]	Expanded Cells	Path Length [m]
A*	15.72	56.2	401,497	62.43
Dijkstra	13.78	53.3	596,325	62.22
A* _{zone}	0.11	49.8	355,200	71.04
Dijkstra _{zone}	0.06	49.1	545,255	72.84

The default implementations of A* and Dijkstra provide the most cost-efficient path. The cost efficiency is mainly influenced by the cost levels of the different zones. Inside a zone, the length of the path is the factor of influence. Therefore, the average path length of the default implementations is approximately 14% smaller compared to our approach. With an average aMSE of 0.11 m for A*_{zone} and 0.06 m for Dijkstra_{zone}, our approach proved to provide consistent paths following the concept of right-hand driving.

The general difference in processing time between the approaches A*, A*_{zone} and Dijkstra, and Dijkstra_{zone} is caused by the calculation of the heuristic. Therefore, the average processing time for A*_{zone} is 2.5% greater than that for Dijkstra_{zone}, even though the number of expanded cells is approximately 35% smaller.

The additional computation effort needed to determine the right-hand driving cells at each expansion step does not increase the overall processing time. Quite the opposite, the approach A*_{zone} needs approximately 11% less average processing time. This decrease correlates with the number of expanded cells, which is also approximately 11% smaller than that of A*. The same applies to the approach Dijkstra_{zone}, which results in 9% less expanded cells and a corresponding decrease in the overall processing time. Table 2 shows a comparison of the five start-goal-position tuple groups by evaluating the number of expanded cells normalized to the resulting path length for each tuple.

Table 2. The number of expanded cells normalized to the path length.

Approach	T_{RR}	T_{SS}	T_{SC}	T_{CS}	T_{CC}
A*	7643	9801	2836	8852	3298
Dijkstra	10,366	11,294	6261	11,663	6425
A* _{zone}	6170	7745	1695	7963	1623
Dijkstra _{zone}	8219	8818	4401	10,115	4496

The goal zone has a high impact on the efficiency of the evaluated path planners. If the goal position lies within the corridor zone (T_{SC} , T_{CC}), the number of expanded cells is significantly lower compared to a goal position outside the corridor zone (T_{RR} , T_{SS} , T_{CS}), by comparing columns T_{SC} , T_{CC} with other columns of Table 2. This behavior is caused by the cost levels of the zones and the nature of the path planning algorithms. Since the cost level of the corridor zone is relatively low the planners will expand inside the corridor zone with a high preference, before starting to expand inside a restricted or station zone. The A* and A*_{zone} implementations provide a lower number of expanded cells for each paths group, as highlighted by the bold font in Table 2, which is caused by the additional heuristic.

Our right-hand driving approach results in an additional decrease in the number of expanded cells. This is caused by the effect of our method on the behavior of the path planning algorithms. Figure 12 shows a comparison of a path $p \in T_{SC}$ and the corresponding expansion potentials for each of the evaluated approaches.

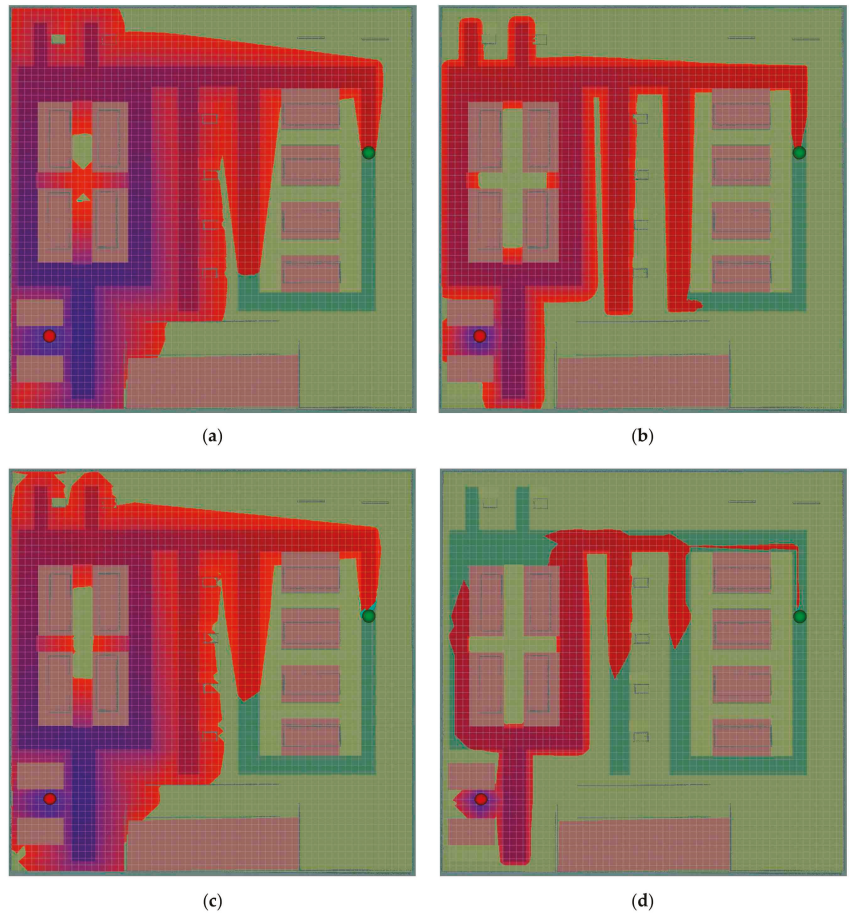


Figure 12. Path potentials of a path related to a start-goal-position tuple $t \in T_{SC}$. Start position in red and goal position in green. (a) Dijkstra; (b) DijkstraZone; (c) A*; (d) A*Zone.

The relatively low-cost cells at the edge between the corridor zone and the guard rail zone let the path planning algorithms concentrate their expansion towards a particular direction. Since the corridor zone is designed to connect typical station areas in the industrial environment, our method reduces the number of cells that are expanded in the corridor zone. Therefore, our method provides a higher expansion efficiency which results in lower computational effort compared to standard implementations of Dijkstra or A*.

3.2. Evaluation in Real-World Use-Case

The presented concept is evaluated at a typical rotor blade form, which features a length of 11 m. Due to space limitations of the experiment environment, only the tip of the form is used throughout the experiment. Figure 13a shows the 3D model of the tip. The tip has the dimensions $2.0 \times 0.8 \times 0.9 \text{ m}^3$ (LxHxW). Figure 13b,c show the real-world form equipped with glass fiber mats and a setup for vacuum generation.

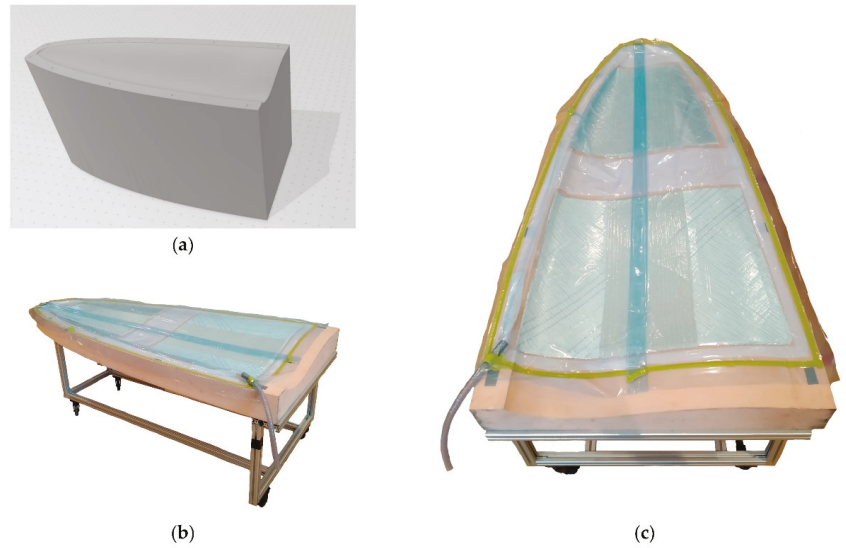


Figure 13. Rotor blade forms. (a) 3D model of the tip of the rotor blade form; (b) rotor blade form used in the real-world experiment (side view); (c) rotor blade form used in the real-world experiment (top view).

Figure 14a shows the analyzed workspace of the used Universal Robot UR5 displayed as a cut in the x, y -plane of the manipulator coordinate frame R . Since the end-effector will be downward oriented most of the time during inspection and accordingly to [29], the used geometric primitive to analyze the workspace is a downward facing hemisphere. Each analyzed voxel has the edge length of 50 mm. Figure 14b shows the same cut, with a threshold at 50% reachability applied.

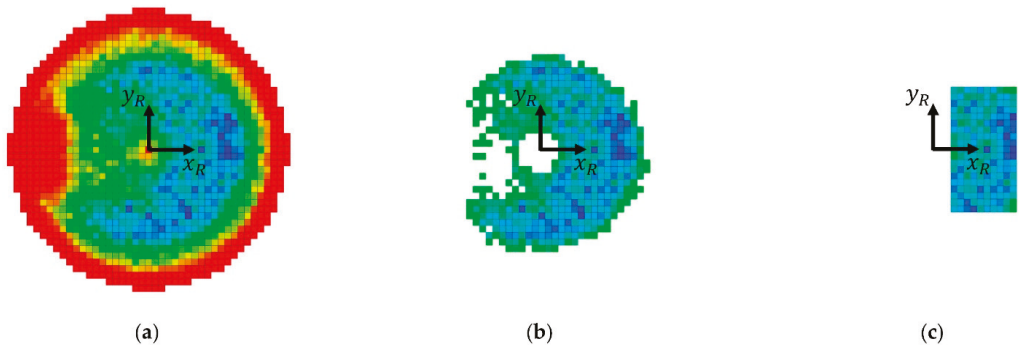


Figure 14. Workspace analysis of the manipulator for the inspection task at the corresponding height. (a) Cut through the x, y -plane; (b) Cut through the x, y -plane with reachability threshold of 50%; (c) chosen workspace area at height $z_r = 0.1$ m.

The chosen suitable workspace area consists of two layers in z -direction to cover the curved shape of the blade tip form. The area is shown in Figure 14c and of size: 0.45 m in x -direction, 0.7 m in y -direction, and 0.1 m in z -direction of the coordinate system R .

In addition, the so-called max reach value max_{reach} of the mobile manipulator is calculated. The max_{reach} symbolizes the maximum distance the manipulator is able to reach

into the workpiece alongside the robot coordinate system R . The calculation is given in Equation (6):

$$max_{reach} = D_{MW} - (d_{RPF} + d_{PFW}), \quad (6)$$

where D_{MW} is the distance between the origin of R and the reachable voxel, which is located at the maximum distance alongside the x -axis of R . The distances d_{RPF} and d_{PFW} depend on the hardware setup and the minimum safety distance between the platform and the workpiece. The distance d_{RPF} describes the distance between the origin of R and the front of the mobile platform. The distance d_{PFW} describes the safety distance between the front of the mobile platform and the workpiece.

Based on the method presented in Section 2.3.4, the workpiece is segmented in inspectable subsegments taking the maximum reach max_{reach} of the hardware setup into account. Figure 15a shows the colored subsegments. Figure 15b shows the corresponding poses of the mobile platform as black arrows for each subsegment. Figure 15c shows the mobile manipulator executing the scan process at a subsegment of the work piece.

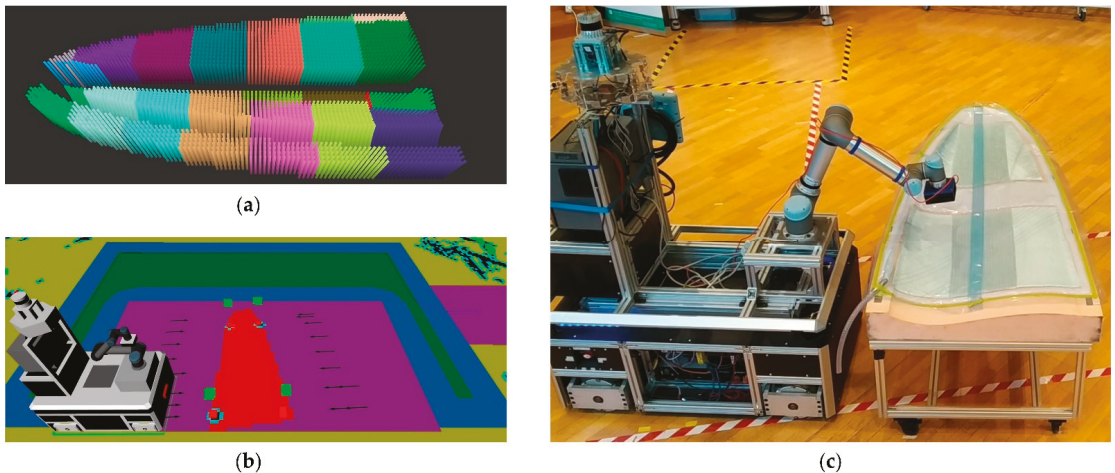


Figure 15. Evaluation in a real use-case. (a) Work piece segmentation; (b) visualization of the navigation zone setup and the platform positioning; (c) actual execution of the scanning process.

At each local subsegment, the surface reconstruction (cf. Section 2.3.1) is executed. Therefore, an RGB-D camera provides a point cloud of the work piece. A crop box filter removes all points, which are related to the ground or the robot itself. In the next step, a HSV filter is applied to the remaining points to remove points related to the yellow corner tape of the vacuum setup (cf. Figure 15c). The resulting points are clustered by their Euclidean distance to determine the points belonging to the scanning surface. A down sampled version of the resulting point cloud is used for the surface reconstruction. Figure 16 shows the complete pipeline for a point cloud captured at a local subsegment.

The waypoint generation (cf. Section 2.3.2) is based on the surface reconstruction. Figure 17a shows the approached 3D positions of the end-effector in the robot coordinate frame R during the inspection process of one local subsegment. Figure 17b shows the surface-orthogonal orientation of the end-effector at each 3D position.

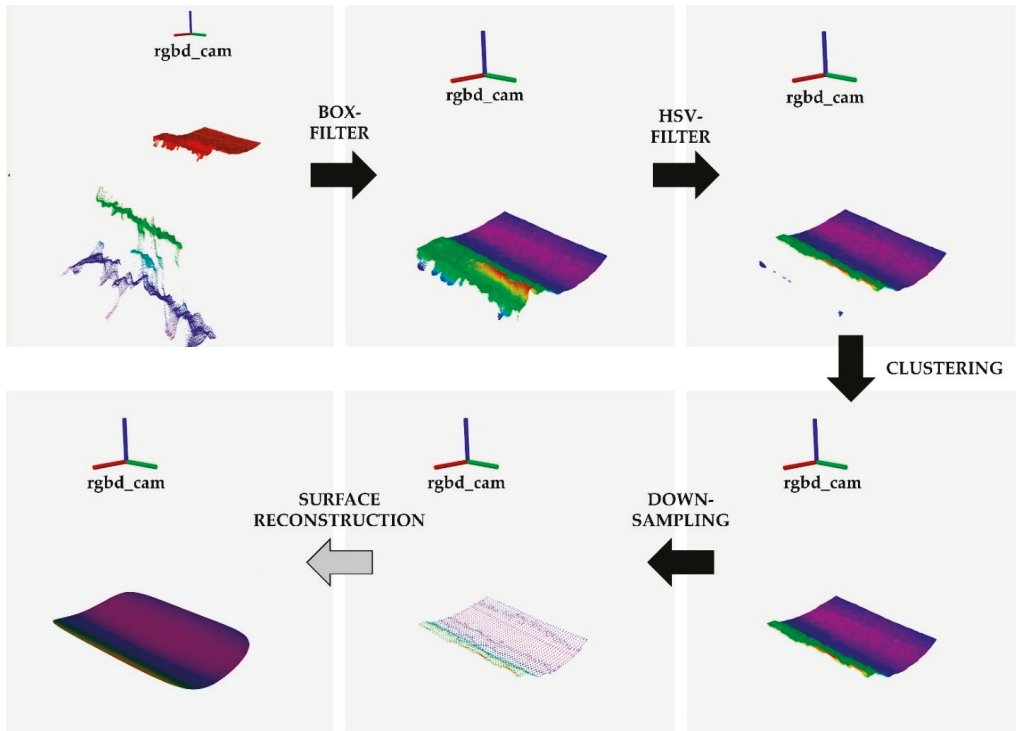


Figure 16. Processing pipeline of the surface reconstruction at a local workpiece subsegment.

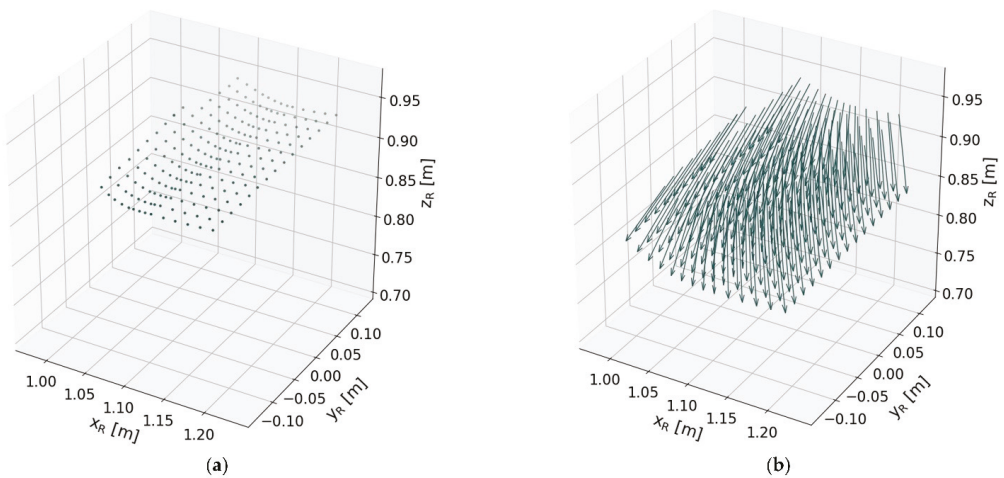


Figure 17. Execution of inspection process at one local subsegment of the work piece. (a) 3D positions of the end-effector; (b) orientation of the end-effector at each 3D position.

The 3D positions of the end-effector are shifted alongside the calculated surface normal by the amount of the chosen scanning height. The resulting 3D positions reflect the surface

of the workpiece (see Figure 18a). Figure 18b shows the path of the end-effector during the inspection process.

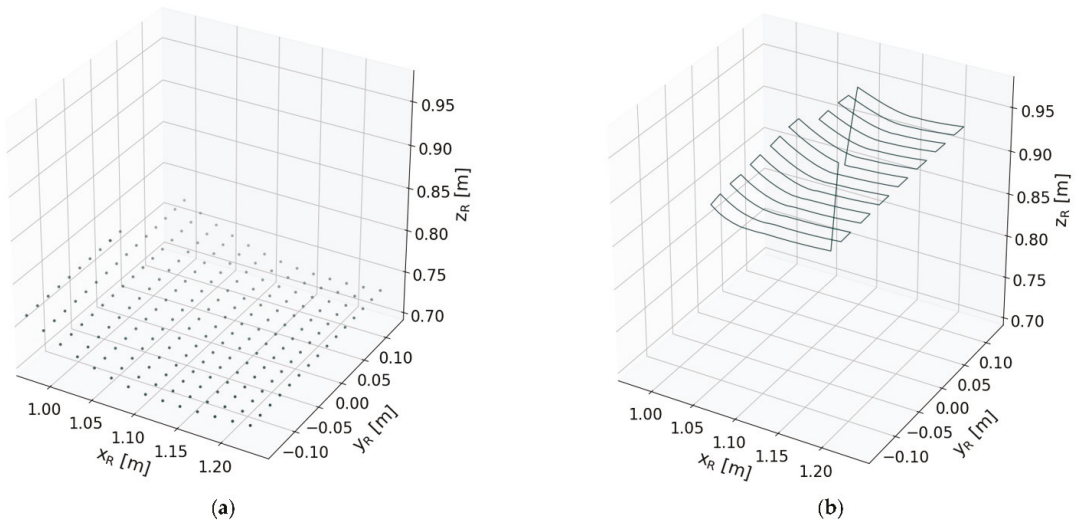


Figure 18. Execution of inspection process at one local subsegment of the work piece. (a) 3D positions of the end-effector shifted alongside the surface normal by the amount of the scanning distance; (b) path executed by the end-effector during the inspection process.

The pose information of each local subsegment is transformed into the static global frame map. Therefore, the localization capabilities of the mobile manipulator OMNIVIL are used, as described in [29]. Figure 19a shows the approached 3D positions of the end-effector in the map frame. The positions are colored accordingly to their related local subsegment. Figure 19b shows the surface-orthogonal orientation of the end-effector at each 3D position.

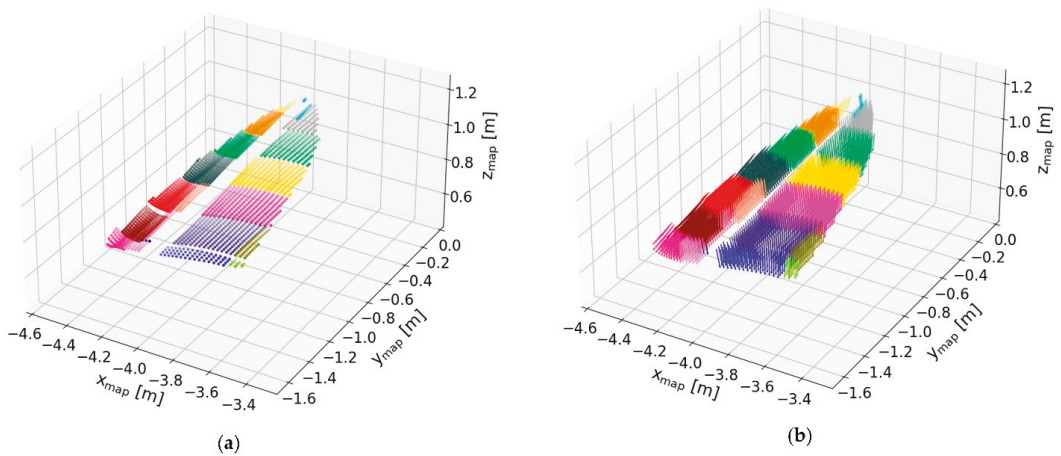


Figure 19. Execution of inspection process of the complete work piece. (a) 3D positions of the end-effector; (b) orientation of the end-effector at each 3D position.

Figure 20a shows the shifted end-effector positions, which reflect the concave and convex shape of the scanned surface. The middle part of the form is not covered due to the

limited maximum reach of the used manipulator UR5. Figure 20b shows the path of the end-effector executed at each local subsegment.

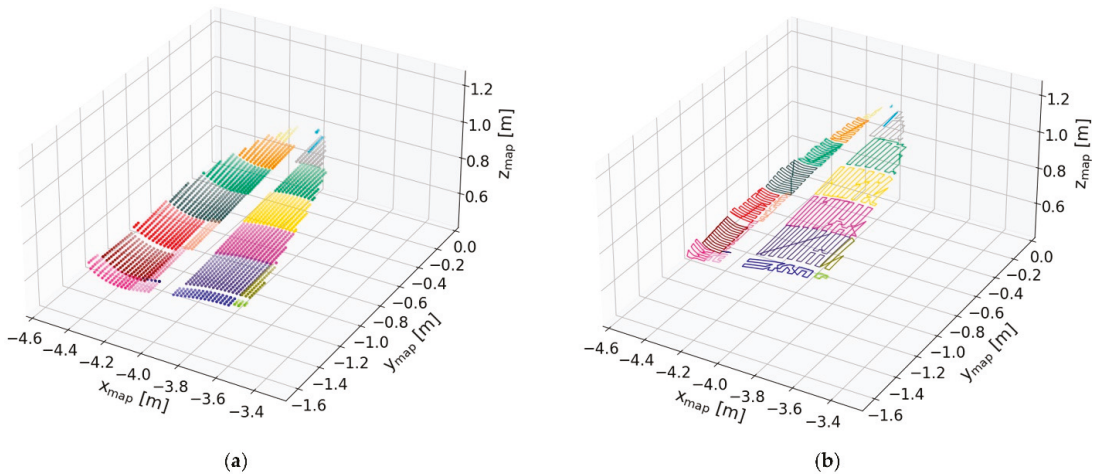


Figure 20. Execution of inspection process of the complete work piece. (a) 3D positions of the end-effector shifted alongside the surface normal by the amount of the scanning distance; (b) path executed by the end-effector during the inspection process.

4. Conclusions

This study presented a method for the automation of the large-scale inspection process of wind turbine blades in manufacturing. The focus was set on the control of the autonomous mobile manipulator. It provided insights into related research fields, including autonomous navigation and surface orthogonal motion planning. The presented methods are applicable to various tasks related to large-scale inspection.

The developed approach realized autonomous navigation including a zone-based segmentation of the production environment. The common approach of a layered costmap was extended to fulfil the needs of a collaborative human–robot industrial environments. In addition, a new method was presented, which manipulates the cost values during the search expansion of a path planner. The method was applied to the state-of-the-art algorithms A* and Dijkstra and was used to realize a right-hand driving behavior of the mobile manipulator in corridor zones. An experiment in a simulation environment showed the superior efficiency and reliability of the method. The actual inspection process was performed in an asynchronous mode by the mobile manipulator. Therefore, a method was developed to segment the workpiece into smaller subsegments, which can be inspected by the manipulator. The motion planning at the local subsegments used a surface reconstruction based on point cloud data. The resulting waypoints were considered nodes in a complete graph. The problem to find the shortest path was solved by applying algorithms related to the traveling salesman problem. The developed system was evaluated in a real-use case.

Further improvements will focus on the segmentation of the production environment. The segmentation can be performed automatically by taking documentation of the factory layout into a concern or by identifying workstations by the robot itself. Furthermore, the generation of the workpiece model should be performed by the robot itself.

Author Contributions: Conceptualization and methodology, H.E.; supervision, S.D. and S.K.; software, H.E., P.C. and H.D.; validation, H.E., P.C. and H.D.; writing—original draft preparation, H.E.; writing—review and editing, H.E. and S.D.; project administration, H.E. and P.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by European Regional Development Fund. Research project FiberRadar (EFRE-0801493).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: This project was supported by the Faculty of Engineering and Built Environment, Tshwane University of Technology and the Faculty of Mechanical Engineering and Mechatronics, University of Applied Sciences Aachen.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. GWEC. Global Wind Report 2021. Available online: https://www.windenergyhamburg.com/fileadmin/windenergy/2022/pdf/we_gwec-global-wind-report-2021.pdf (accessed on 3 May 2021).
2. Ancona, D.; McVeigh, J. *Wind Turbine-Materials and Manufacturing Fact Sheet*; Princeton Energy Resources International, LLC: Rockville, MD, USA, 2001; p. 19.
3. Murray, R.; Swan, D.; Snowberg, D.R.; Berry, D.; Beach, R.; Rooney, S. Manufacturing a 9-Meter Thermoplastic Composite Wind Turbine Blade. In Proceedings of the American Society for Composites Thirty-Second Technical Conference, West Lafayette, IN, USA, 23–25 October 2017; DEStech Publications: Lancaster, PA, USA, 2017. ISBN 978-1-60595-418-9.
4. Yang, K.; Rongong, J.A.; Worden, K. Damage detection in a laboratory wind turbine blade using techniques of ultrasonic NDT and SHM. *Strain* **2018**, *54*, e12290. [CrossRef]
5. Garcia Marquez, F.P.; Gomez Munoz, C.Q. A new approach for fault detection, location and diagnosis by ultrasonic testing. *Energies* **2020**, *13*, 1192. [CrossRef]
6. Yang, R.; He, Y.; Mandelis, A.; Wang, N.; Wu, X.; Huang, S. Induction infrared thermography and thermal-wave-radar analysis for imaging inspection and diagnosis of blade composites. *IEEE Trans. Ind. Inform.* **2018**, *14*, 5637–5647. [CrossRef]
7. Hwang, S.; An, Y.-K.; Sohn, H. Continuous line laser thermography for damage imaging of rotating wind turbine blades. *Procedia Eng.* **2017**, *188*, 225–232. [CrossRef]
8. Arnold, P.; Moll, J.; Mälzer, M.; Krozer, V.; Pozdniakov, D.; Salman, R.; Rediske, S.; Scholz, M.; Friedmann, H.; Nuber, A. Radar-based structural health monitoring of wind turbine blades: The case of damage localization. *Wind Energy* **2018**, *21*, 676–680. [CrossRef]
9. Herschel, R.; Pawliczek, S. 3D millimeter wave screening of wind turbine blade segments. In Proceedings of the 15th European Radar Conference (EuRAD), Madrid, Spain, 26–28 September 2018; pp. 115–117, ISBN 2874870536.
10. Froehly, A.; Herschel, R. Refraction Compensation in Non-Destructive Testing. In Proceedings of the 15th European Conference on Antennas and Propagation (EuCAP), Düsseldorf, Germany, 22–26 March 2021; pp. 1–5, ISBN 8831299026.
11. Enevoldsen, P.; Xydis, G. Examining the trends of 35 years growth of key wind turbine components. *Energy Sustain. Dev.* **2019**, *50*, 18–26. [CrossRef]
12. Mishnaevsky, L.; Branner, K.; Petersen, H.N.; Beauson, J.; McGugan, M.; Sørensen, B.F. Materials for wind turbine blades: An overview. *Materials* **2017**, *10*, 1285. [CrossRef] [PubMed]
13. Hvilshøj, M.; Bøgh, S.; Nielsen, O.S.; Madsen, O. Autonomous industrial mobile manipulation (AIMM): Past, present and future. *Ind. Robot Int. J.* **2012**, *39*, 120–135. [CrossRef]
14. Schuler, J. *Integration von Förder- und Handhabungseinrichtungen*; Springer: Berlin, Germany, 1987.
15. Hvilshøj, M.; Bøgh, S.; Nielsen, O.S.; Madsen, O. Multiple part feeding—Real-world application for mobile manipulators. *Assem. Autom.* **2012**, *32*, 62–71. [CrossRef]
16. Halt, L.; Meßmer, F.; Hermann, M.; Wochinger, T.; Naumann, M.; Verl, A. AMADEUS-A robotic multipurpose solution for intralogistics. In Proceedings of the 7th German Conference on Robotics, ROBOTIK 2012, Munich, Germany, 21–22 May 2012; pp. 1–6.
17. Krueger, V.; Chazoule, A.; Crosby, M.; Lasnier, A.; Pedersen, M.R.; Rovida, F.; Nalpantidis, L.; Petrick, R.; Toscano, C.; Veiga, G. A Vertical and Cyber-Physical Integration of Cognitive Robots in Manufacturing. *Proc. IEEE* **2016**, *104*, 1114–1127. [CrossRef]
18. Bøgh, S.; Schou, C.; Ruehr, T.; Kogan, Y.; Doemel, A.; Brucker, M.; Eberst, C.; Tornese, R.; Sprunk, C.; Tipaldi, G.D.; et al. Integration and Assessment of Multiple Mobile Manipulators in a Real-World Industrial Production Facility. In Proceedings of the 41st International Symposium on Robotics, Munich, Germany, 2–3 June 2014; pp. 1–8.
19. Dömel, A.; Kriegel, S.; Kaßbecker, M.; Brucker, M.; Bodenmüller, T.; Suppa, M. Toward fully autonomous mobile manipulation for industrial environments. *Int. J. Adv. Robot. Syst.* **2017**, *14*, 1–19. [CrossRef]
20. Outón, J.L.; Villaverde, L.; Herrero, H.; Esnaola, U.; Sierra, B. Innovative Mobile Manipulator Solution for Modern Flexible Manufacturing Processes. *Sensors* **2019**, *19*, 5414. [CrossRef] [PubMed]
21. Saenz, J.; Vogel, C.; Penzlin, F.; Elkmann, N. Safeguarding Collaborative Mobile Manipulators—Evaluation of the VALERI Workspace Monitoring System. *Procedia Manuf.* **2017**, *11*, 47–54. [CrossRef]

22. Fritzsche, M.; Saenz, J.; Penzlin, F. A large scale tactile sensor for safe mobile robot manipulation. In Proceedings of the 11th International Conference on Human-Robot Interaction (HRI), Christchurch, New Zealand, 7–10 March 2016; pp. 427–428.
23. Andersen, R.S.; Bøgh, S.; Moeslund, T.B.; Madsen, O. Intuitive task programming of stud welding robots for ship construction. In Proceedings of the International Conference on Industrial Technology, Seville, Spain, 17–19 March 2015; pp. 3302–3307.
24. Yu, L.; Yang, E.; Ren, P.; Luo, C.; Dobie, G.; Gu, D.; Yan, X. Inspection Robots in Oil and Gas Industry: A Review of Current Solutions and Future Trends. In Proceedings of the 25th International Conference on Automation and Computing (ICAC), Lancaster, UK, 5–7 September 2019; pp. 1–6, ISBN 978-1-8613-7665-7.
25. Hashim, A.S.; Grămescu, B.; Nițu, C. State of the Art Survey on Using Robots in Oil and Gas Industry. In Proceedings of the International Conference of Mechatronics and Cyber-MixMechatronics—2017, Bucharest, Romania, 7–8 September 2017; Springer International Publishing: Cham, Switzerland, 2018; pp. 177–185.
26. Lu, S.; Zhang, Y.; Su, J. Mobile robot for power substation inspection: A survey. *IEEE/CAA J. Autom. Sin.* **2017**, *4*, 830–847. [[CrossRef](#)]
27. Alhassan, A.B.; Zhang, X.; Shen, H.; Xu, H. Power transmission line inspection robots: A review, trends and challenges for future research. *Int. J. Electr. Power Energy Syst.* **2020**, *118*, 105862. [[CrossRef](#)]
28. Lattanzi, D.; Miller, G. Review of Robotic Infrastructure Inspection Systems. *J. Infrastruct. Syst.* **2017**, *23*, 4017004. [[CrossRef](#)]
29. Engemann, H.; Du, S.; Kallweit, S.; Cönen, P.; Dawar, H. OMNIVIL—An Autonomous Mobile Manipulator for Flexible Production. *Sensors* **2020**, *20*, 7249. [[CrossRef](#)] [[PubMed](#)]
30. Diegel, O.; Badve, A.; Bright, G.; Potgieter, J.; Tlale, S. Improved mecanum wheel design for omni-directional robots. In Proceedings of the Australasian conference on robotics and automation, Auckland, New Zealand, 27–29 November 2002; pp. 117–121.
31. Lu, D.V. Contextualized Robot Navigation. Ph.D. Thesis, Washington University, Washington, DC, USA, 2014.
32. Lu, D.V.; Hershberger, D.; Smart, W.D. Layered costmaps for context-sensitive navigation. In Proceedings of the International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 709–715.
33. Hasan, K.M.; Al Mamun, A. Implementation of autonomous line follower robot. In Proceedings of the International Conference on Informatics, Electronics & Vision, Dhaka, Bangladesh, 18–19 May 2014; pp. 865–869.
34. Herrero-Pérez, D.; Alcaraz-Jiménez, J.J.; Martínez-Barberá, H. An accurate and robust flexible guidance system for indoor industrial environments. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 292–302. [[CrossRef](#)]
35. Yoon, S.W.; Park, S.-B.; Kim, J.S. Kalman filter sensor fusion for Mecanum wheeled automated guided vehicle localization. *J. Sens.* **2015**, *2015*, 347379. [[CrossRef](#)]
36. Hart, P.; Nilsson, N.; Raphael, B. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybern.* **1968**, *4*, 100–107. [[CrossRef](#)]
37. Dijkstra, E.W. A note on two problems in connexion with graphs. *Numer. Math.* **1959**, *1*, 269–271. [[CrossRef](#)]
38. Lu, D.V.; Smart, W.D. Towards more efficient navigation for robots and humans. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Tokyo, Japan, 3–7 November 2013; pp. 1707–1713, ISBN 1467363588.
39. Rusu, R.B.; Cousins, S. 3d is here: Point cloud library (pcl). In Proceedings of the IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4, ISBN 1612843859.
40. Mörwald, T. Object Modelling for Cognitive Robotics. Ph.D. Thesis, Vienna University of Technology, Vienna, Austria, 2013.
41. Choi, Y.; Choi, Y.; Briceno, S.; Mavris, D.N. Three-dimensional UAS trajectory optimization for remote sensing in an irregular terrain environment. In Proceedings of the International Conference on Unmanned Aircraft Systems (ICUAS), Dallas, TX, USA, 13–15 June 2018; pp. 1101–1108, ISBN 1538613549.
42. Bondy, J.A.; Murty, U.S.R. *Graph Theory*; Springer: New York, NY, USA, 2008; ISBN 9781846289699.
43. Cormen, T.H.; Leiserson, C.E.; Rivest, R.L.; Stein, C. *Introduction to Algorithms*; MIT Press: Cambridge, MA, USA, 2009; ISBN 0262533057.
44. Christofides, N. *Worst-Case Analysis of a New Heuristic for the Travelling Salesman Problem*; Technical Report 388; Carnegie-Mellon University: Pittsburgh, PA, USA, 1976.
45. Kolmogorov, V. Blossom V: A new implementation of a minimum cost perfect matching algorithm. *Math. Program. Comput.* **2009**, *1*, 43–67. [[CrossRef](#)]
46. Bohren, J.; Cousins, S. The smach high-level executive [ros news]. *IEEE Robot. Autom. Mag.* **2010**, *17*, 18–20. [[CrossRef](#)]
47. Putz, S.; Santos Simon, J.; Hertzberg, J. Move Base Flex A Highly Flexible Navigation Framework for Mobile Robots. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Madrid, Spain, 1–5 October 2018; pp. 3416–3421, ISBN 978-1-5386-8094-0.
48. Koenig, N.; Howard, A. Design and use paradigms for gazebo, an open-source multi-robot simulator. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Sendai, Japan, 28 September–2 October 2004; pp. 2149–2154, ISBN 0780384636.
49. Quigley, M.; Conley, K.; Gerkey, B.; Faust, J.; Foote, T.; Leibs, J.; Wheeler, R.; Ng, A.Y. ROS: An open-source Robot Operating System. In Proceedings of the Workshops at the IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; p. 5.

Article

CHARMIE: A Collaborative Healthcare and Home Service and Assistant Robot for Elderly Care

Tiago Ribeiro ^{1,2}, Fernando Gonçalves ^{3,4}, Inês S. Garcia ^{1,2,5}, Gil Lopes ⁶ and António F. Ribeiro ^{1,2,*}

¹ Industrial Electronics Department, University of Minho, 4800-058 Guimarães, Portugal; id9402@alunos.uminho.pt (T.R.); id8672@alunos.uminho.pt (I.S.G.)

² Centro ALGORITMI, University of Minho, Campus Azurém, 4800-058 Guimarães, Portugal;

³ Mechanical Engineering Department, University of Minho, 4800-058 Guimarães, Portugal; id8699@alunos.uminho.pt

⁴ CMEMS, University of Minho, Campus Azurém, 4800-058 Guimarães, Portugal

⁵ International Iberian Nanotechnology Laboratory (INL), 4715-330 Braga, Portugal

⁶ University Institute of Maia—ISMAI, 4475-690 Maia, Portugal; alopes@ismai.pt

* Correspondence: fernando@dei.uminho.pt

Abstract: The global population is ageing at an unprecedented rate. With changes in life expectancy across the world, three major issues arise: an increasing proportion of senior citizens; cognitive and physical problems progressively affecting the elderly; and a growing number of single-person households. The available data proves the ever-increasing necessity for efficient elderly care solutions such as healthcare service and assistive robots. Additionally, such robotic solutions provide safe healthcare assistance in public health emergencies such as the SARS-CoV-2 virus (COVID-19). CHARMIE is an anthropomorphic collaborative healthcare and domestic assistant robot capable of performing generic service tasks in non-standardised healthcare and domestic environment settings. The combination of its hardware and software solutions demonstrates map building and self-localisation, safe navigation through dynamic obstacle detection and avoidance, different human-robot interaction systems, speech and hearing, pose/gesture estimation and household object manipulation. Moreover, CHARMIE performs end-to-end chores in nursing homes, domestic houses, and healthcare facilities. Some examples of these chores are to help users transport items, fall detection, tidying up rooms, user following, and set up a table. The robot can perform a wide range of chores, either independently or collaboratively. CHARMIE provides a generic robotic solution such that older people can live longer, more independent, and healthier lives.

Keywords: service robot; assistant robot; collaborative robot; healthcare; elderly care; intelligent systems; COVID-19

Citation: Ribeiro, T.; Gonçalves, F.; Garcia, I.S.; Lopes, G.; Ribeiro, A.F. CHARMIE: A Collaborative Healthcare and Home Service and Assistant Robot for Elderly Care. *Appl. Sci.* **2021**, *11*, 7248. <https://doi.org/10.3390/app11167248>

Academic Editors: António Paulo Moreira and Emanuele Carpanzano

Received: 7 May 2021

Accepted: 2 August 2021

Published: 6 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The world's population is consistently growing older, with the over-65 age group overgrowing all other age groups. The predicted worldwide growth in the elderly population is from 702 million in 2019 to over 1.5 billion by 2050 [1]. In 2014, the over-55-years old population outnumbered persons aged 15 to 24 years old, and by 2035 it is expected that the ages 0 to 14 will also be outnumbered [2]. The elder age group is estimated to continuously grow and outnumber all youth and child populations under 24 years old by 2050. It is estimated that the percentage of elderly people (the over-65 age group) in the European Union will grow from 19% to 29% over the next approximately 50 years [3]. Population ageing and public health expenses primarily dedicated to older dependent persons presents significant challenges, with implications not just on the social aspect but also economically [4]. The World Health Organization (WHO, Geneva, Switzerland) even developed a global strategy and action plan on ageing and health that focused, among other strategic objectives improving measurement, monitoring, and research on healthy ageing [5].

These statistics/data prove the growing need for efficient elderly care solutions regarding therapy, rehabilitation, companions and activity planning, but most importantly, healthcare robotics [6–9] that are capable of aiding in day-to-day tasks, collaboratively or independently. Healthcare generic service and assistive robots can provide practical help to increase the elderly population’s life quality, improving cognitive and physical health. These robots can play an essential role concerning healthcare support and independent life, especially when problems related to ageing start to appear. Moreover, service robots provide safe healthcare assistance in public health emergencies such as the SARS-CoV-2 virus (COVID-19) [10–12].

The Collaborative Healthcare/Home Assistant Robot by Minho Industrial Electronics (CHARMIE), represented in Figure 1, is an anthropomorphic healthcare and domestic service and assistive robot capable of performing tasks in non-standardised environmental settings. The social goal of the CHARMIE project is the development of a robot capable of aiding in nursing homes, healthcare facilities and domestic houses, among other settings. The scientific objective of CHARMIE is the development of a multifaceted anthropomorphic robot capable of performing a broad set of tasks based solely on machine learning solutions that allow the robot to learn how to perform and improve tasks via observation and trial-and-error direct interaction with the environment.

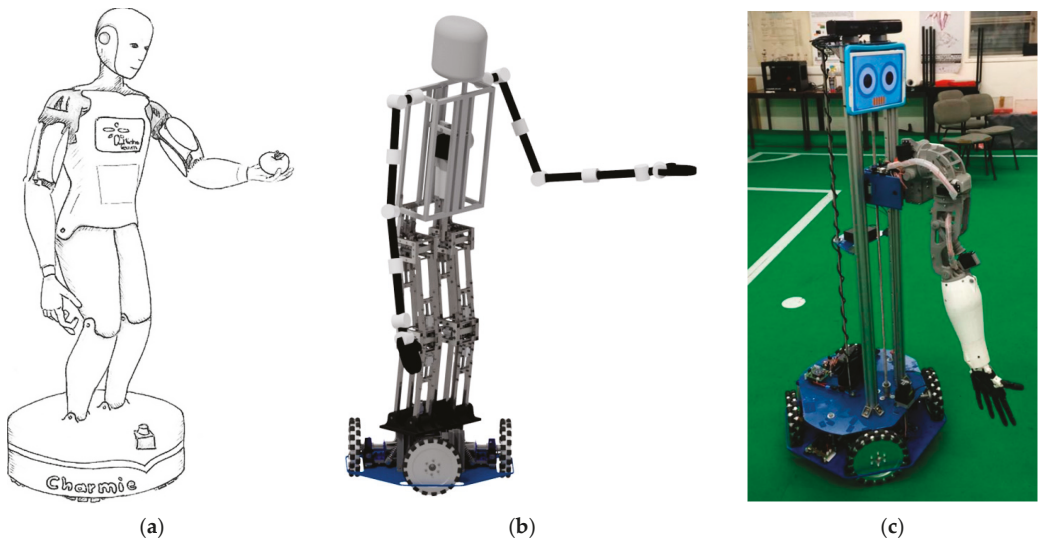


Figure 1. CHARMIE (Collaborative Healthcare/Home Assistant Robot by Minho Industrial Electronics) different variations. (a) Conceptual sketch of the anthropomorphic robot. (b) Developed anthropomorphic design. (c) Primary prototype assembled.

Some service robots are already being implemented in geriatric care [13]. The development of such robots is encouraged with funding from the European Union with projects such as Hobbit [14], ENRICHME [15], ARNA [16] and Sobi [17]. These four robots go beyond the simpler pet-like social companion robots already possessing sensors and actuators that allow them to perform more complex tasks. Focusing on enhancing older people’s well-being, some examples of tasks already performed are user entertainment, object manipulation and transportation, empathic and social human-robot interaction, monitoring persons and home-related chores. MONarCH [18] presents a multi-robot cognitive system whose operation is already being tested in hospitals. It targets the development of a novel framework to model mixed human-robot societies and its demonstration using a network of heterogeneous robots and sensors in an oncological hospital’s pediatric area. It

can handle uncertainties introduced by people and robots, generate natural interactions and engage in educational entertainment activities. However, the previously mentioned robots only solve tasks using pre-programmed methods and cannot generalise to different task variations or significant environmental changes.

Robotics competitions can be used for benchmarking specific functionalities and integrated systems [19,20]. RoboCup@Home [21,22] is no exception and is regarded as a top competition with numerous state-of-the-art contributions in the field of service and assistive robotics. The goal is to foster the development of versatile domestic service robots that operate safely in daily life non-standardised environments. Some of the robots achieving the best results in the latest competitions are Walking Machine [23], RoboFEI [24], CATIE [25], Homer [26] and AMIGO [27]. Recent years have shown robots solving a wide range of different domestic tasks, like watering plants, taking the trash out, storing groceries, serving breakfast, cleaning and setting up a table. These tasks provide broad coverage of multidisciplinary subjects such as Navigation, Mapping, Person Recognition, Person Tracking, Object Recognition, Object Manipulation, Speech Recognition, Gesture Recognition and Cognition. Even though the tasks referred fit the domestic environment standards, these can straightforwardly be adapted to healthcare purposes and environments due to their genericness. Regarding cognition, RoboCup@Home teams are already using machine learning technique primarily based on supervised learning to solve tasks such as face recognition [28], body pose estimation [29,30] and object detection [31,32].

The presented robot, CHARMIE, is an anthropomorphic general service and assistive robot that focuses on healthcare and domestic environments to aid elderly people and healthcare employees. The purpose of CHARMIE is to provide a robotic solution capable of providing healthcare support and more independent life, assisting both the elderly's cognitive and physical health. It can perform a wide range of tasks in almost all non-standardised indoor environments and some outdoor environments. The primary focus regarding environments are nursing homes, hospitals, healthcare facilities, and domestic houses. These show the highest necessity and opportunity to integrate a robotic system capable of helping elderly people and healthcare workers. The capability of a wide range of tasks allows this robot to help in various domains, such as pick and place tasks, goods transportation, following workers/patients/users, speech communication (hearing and talking), visual analysis of the environment, navigation, object recognition and manipulation. The cooperative and collaborative aspect is demonstrated by the robot's ability to perform tasks parallel to user chores and even aid in tasks that the user cannot perform alone. The generic tasks described can be easily adapted to the robot's purpose. An example of pick and place tasks adapted to the purpose and environment can be: (i) delivering food trays to patients in hospitals; (ii) collecting crutches from residents in nursing homes; (iii) transporting boxes of medicines between different areas in healthcare facilities; and (iv) setting and cleaning up a table or loading a washing machine at a house. The variety of different tasks these generic robots can perform allows the robot to help in various ways, whichever best suits the user's present needs.

2. Materials and Methods

To perform the wide variety of non-standardised chores, CHARMIE's system and hardware went through different development choices regarding hardware components and related dependencies. The complexity required by some tasks dictated that the robot's system and hardware solutions had to contemplate a significant number of degrees of movement. From the omnidirectional platform to the arms, all components were dimensioned, envisioning the complex movements the robot must make. An example is pushing wheeled trolleys that force the robot to adjust the position of both arms throughout the action and the platform movement that must fit the moving object. Additionally, the anthropomorphic shape of CHARMIE allows users to be more receptive to interact with the robot, as described in [33]. From a practical perspective, it is easier to interact with the real world if the robot has the same shape as a human since the environment is optimised

for human-shape interaction. From a social perspective, creating a small human's physical appearance gives the impression users are interacting with a small, friendly robot, which translates into a sense of friendliness and proximity to its users. Even though, at the moment, CHARMIE is not fully anthropomorphic, some parts are, such as the arm. The final objective is to reach the anthropomorphic level presented in Figure 1a,b.

2.1. System and Hardware

As described in Figure 2, CHARMIE's hardware [34] can be divided into four sections: (i) the motion platform; (ii) the robot arms; (iii) the lifting mechanism and torso; and (iv) the robot head. The goal is to provide solutions that best fit generic service tasks to improve elderly care. Thus, [34] provides a more in-depth description of hardware sections (i) and (ii). One aspect that must also be referred to regarding the robot's design is its anthropomorphic design. One study concerning robot shape indicates that the robot's visual design is directly related to the number of human-robot interactions initiated by humans [35]. The anthropomorphic shape grants the people who interact with the robot a greater sense of comfort or friendliness than differently shaped robots. Different shapes made human users more reluctant to interact with the robot due to fear of the unknown and its movement. Having the shape of a human body allowed the robot to have a higher number of interactions, resulting in more tasks performed by the robot, thus helping more significantly its users, both the elderly and healthcare workers. Another significant advantage of the anthropomorphic shape is that every day-to-day human environment is refined to the human body height-wise, weight-wise, and shape-wise. This aspect facilitates robot interaction with day-to-day environments without requiring an adaptation of the world to best fit the robot's capabilities. With this shape, the concept is precisely the opposite: adjust the developed robot shape to enhance its interaction with human environments, rather than adapting every human environment the robot must interact with.

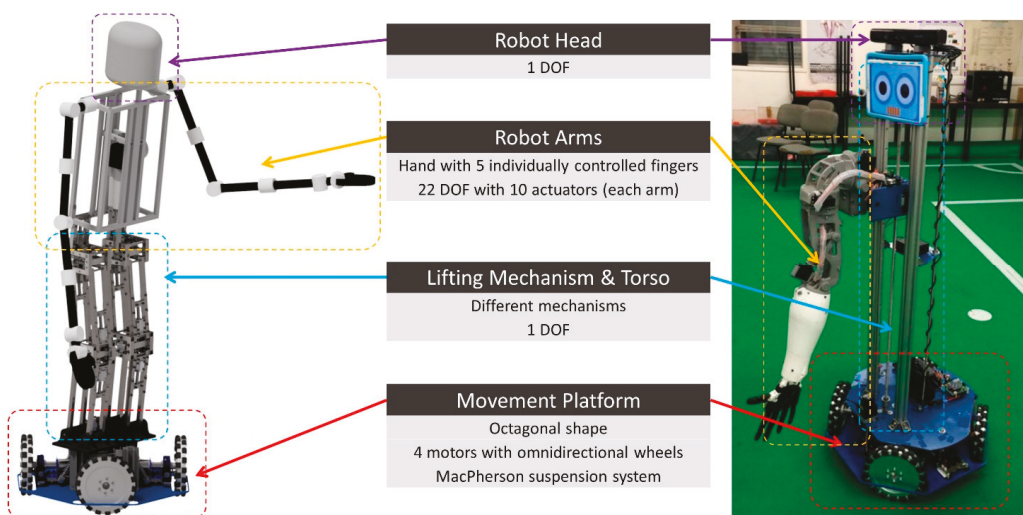


Figure 2. CHARMIE's four hardware sections, namely robot head, robot arms, the lifting mechanism and torso and the motion platform on both the physical robot and the anthropomorphic implementation.

2.1.1. Motion Platform

For service and assistive robots that operate in highly dynamic environments with humans working beside them, it is a significant advantage to have a platform that allows movement in any direction at any point in time. The locomotion system is the only

part of CHARMIE that is not intended to be anthropomorphic (bipedal) for stability and simplicity reasons. The motion platform developed uses four omnidirectional wheels with individual suspension systems. This design was conceived with the robot's predicted interaction environments in mind, mainly large/medium indoor environments such as hospitals, healthcare facilities, nursing homes and houses. The wheels are 203 mm double aluminium omnidirectional wheels with roller bearings. When motion platforms use three omnidirectional wheels, and the center of mass is considerably high, the platform may be at risk of falling under certain circumstances. If a linear momentum is applied in the 120° gap between wheels the robot may lose balance and end up falling. With the addition of the fourth wheel and consequential reduction of the degree gap between wheels the robot significantly improved in safety and stability. However, with this addition, it is possible for a wheel to occasionally lose contact with the floor due to slight surface irregularities or slopes. Locomotion wise, this translates into unpredictable and incorrect movement. Thus, a compact MacPherson [36], Figure 3a, suspension system was developed to overcome floor irregularities, small bumps, and slope variations while improving its control smoothness. The motion platform is a regular octagonal shape with a 54 cm diameter designed so CHARMIE fits through every standard door frame size. It is purposely heavy (~20 kg without batteries) to guarantee a low centre of mass for safety reasons, ensuring the robot does not fall if an external force pushes it. Additionally, this motion platform can transport a load of approximately 65 kg, as shown in Figure 3b. In [34], a more in-depth analysis of the motion platform and suspension system is presented.

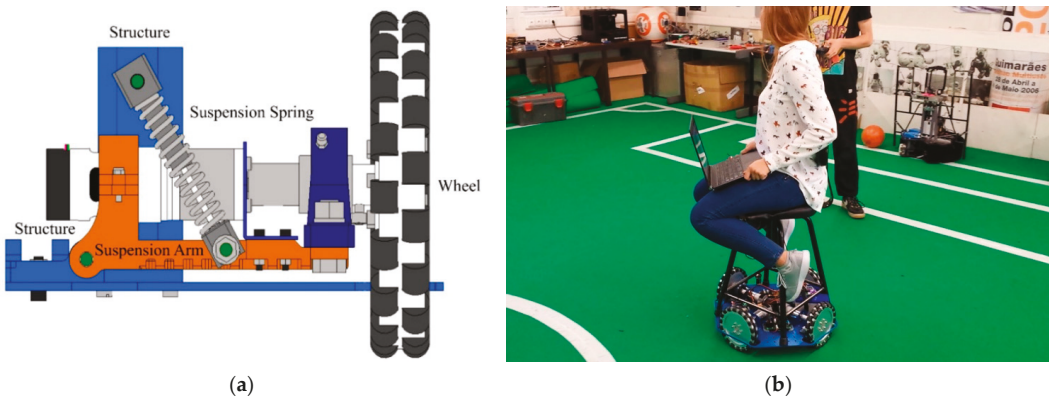


Figure 3. (a) MacPherson suspension system designed for CHARMIE to improve stability and guarantee all four wheels are in contact with the floor surface. Image from [34]. (b) CHARMIE's motion platform load transportation experimental test. The platform was tested, moving for approximately 5 min while maintaining the movement performance and stability. The load in the picture shows an average adult human weight of approximately 65 kg.

2.1.2. Lifting Mechanism and Torso

A significant number of the tasks that generic service and assistive robots must perform involve object manipulation. The motion platform allows the robot to move throughout the environment on the x -axis and y -axis. To interact at different heights with objects such as cabinets with shelves at different heights or with users, from children to adults, it is of extreme value to implement a system that allows the robot to have a z -axis DOF (Degree of Freedom). By implementing a lifting system, the workspace of the redundant manipulators attached to the torso increases. Moreover, the robot can move up and down in its z -axis, allowing it to interact with objects at different heights, from picking/placing objects from/to the floor to interacting with objects on tables, counters and shelves, among others. In the elderly care context, the robot needs to collect things from the floor. This is primarily due to older people having significant difficulties in lowering

themselves to pick up something from the floor, which sometimes leads to falls or injuries that can be avoided with the proposed robot solution.

The initial lifting system solution shown in Figure 4a,b consists of a ball screw spindle mechanism. The torso that is in the threaded shaft has two linear bearings connected to aiding beams that go from the motion platform to the head. By activating the motor, the torso moves linearly up and down, providing the lifting mechanism. In this system, only the torso moves, and the only parts assembled to the torso are the redundant manipulators.

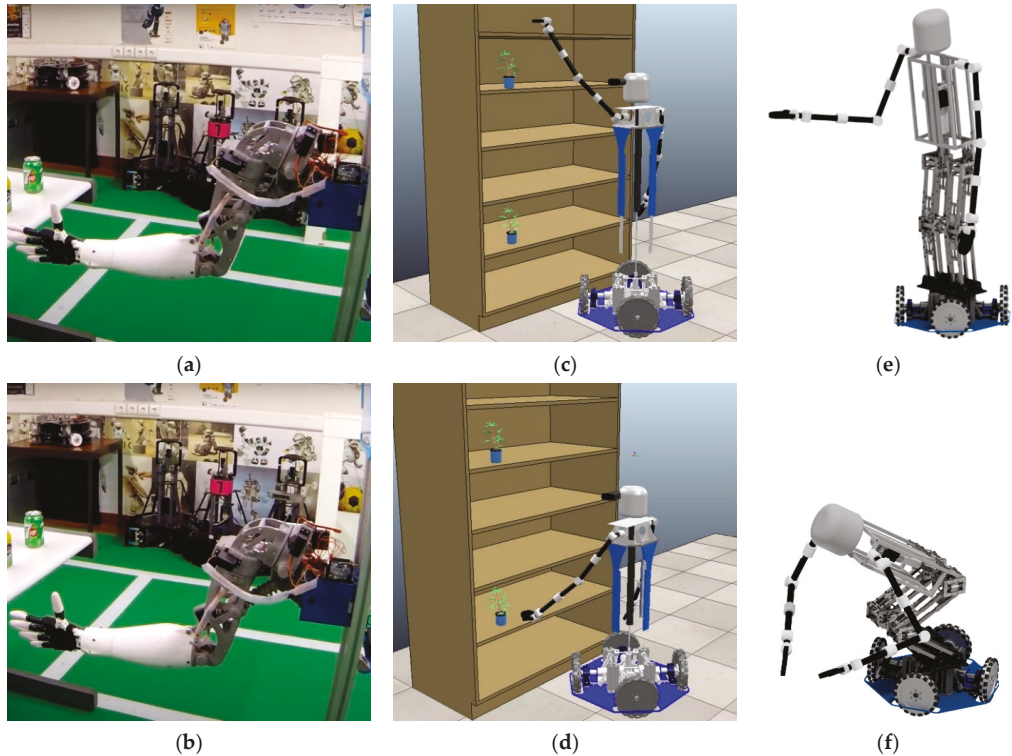


Figure 4. The different lifting mechanisms implemented in CHARMIE. The top three images show the lifting mechanisms at a higher elevation, and the bottom three images show the lifting mechanism at lower heights. (a) Initial ball screw spindle lifting mechanism on top position. (b) Initial ball screw spindle lifting mechanism on the bottom position. (c) Height changing ball screw spindle lifting mechanism on top position. (d) Height changing ball screw spindle lifting mechanism on the bottom position. (e) Anthropomorphic lifting mechanism on top position. (f) Anthropomorphic lifting mechanism on the bottom position.

Moreover, two new different systems were developed to overcome the problems the initial elevation system had. The first, shown in Figure 4c,d, resembles the lifting mechanism presented by AMIGO [37,38]. Similarly to the initial lifting system, a ball screw spindle mechanism lifts the torso. However, in this implementation, an aluminium tube is connected to the threaded shaft. This system allows the torso to be assembled in the aluminium tube, which with three slider rails can move the torso up and down in a linear movement. The two main differences from the AMIGO elevation system to the initial lifting system design are: (i) instead of moving just the torso and consequently the redundant manipulators in the z-axis, it also moves the head; and (ii) the variation of the robot's height. As previously described, this system can adjust the torso height to best fit its goals. CHARMIE can pick objects from the floor in its lower position, and in its higher position, it

can place its arms height-wise, similar to an average size adult human. Since the head is now also present on the torso in this solution, all sensors and the multimodal user interface can alter their height. This system can now move the head height-wise, positioning the camera height to better analyse the environment using the computer vision system. In the initial implementation system, the only degree of freedom was a rotation on the neck so the robot could look up and down. Often, this DOF proved insufficient to analyse the environment since, at times, the robot needed to place its head at the height of what it had to analyse for better results. Computer vision wise, object and user detection are simplified by allowing the robot to position itself at the best angle. This system grants this movement that improved all computer vision-related tasks significantly. The multimodal user interface previously connected to the head can now also move up and down to best fit the user's height. The other advantage of this system is the ability to change the robot's height. Psychologically, it is highly advantageous to alter the robot's height according to the person with which it interacts. Similarly to a human body, when the robot moves down to pick an object from the floor, it moves all of its body with that purpose. The initial solution could only move the arms, not following the anthropomorphic ideology. The field tests developed demonstrated that people were more interactive with this version of the robot due to it being more anthropomorphic. The robot altered its height to be slightly smaller than the user it was interacting with. Creating a sense of inferiority to the human reduces the users' reluctance to intervene with the robot. This way, CHARMIE can operate most features in a domestic environment while at the same time having a friendly appearance.

The second lifting mechanism under testing, shown in Figure 4e,f is based on the human's legs. For this elevation system, the following requirements were taken into consideration: (i) structural integrity to support the robot's weight; (ii) anthropomorphic look; (iii) self-locking actuation to reduce energy consumption; and (iv) allowing the robot to squat, increasing its workspace and making it able to interact with objects on the floor. This lifting mechanism has the same advantages as the first lifting mechanism, all of the upper body can move up and down, and the robot's height can change, with the addition of resembling the human body even more. Although the mechanical system's complexity increases substantially in this iteration, a design based on the same mechanical principles is being made, which results in a more straightforward and more reliable solution with a significant increase in robustness. Thus, this mechanism was developed to encompass only 1 DOF that controls the ankles, the knees and the hips of the robot, allowing a complete squatting movement. In [34], a more in-depth analysis of this anthropomorphic lifting mechanism is presented.

2.1.3. Robotic Arm

The arm's design objective was to develop an affordable, lightweight component with a human-like design. The initial design demonstrates an autonomous robotic manipulator with four degrees of freedom, named the Robotic Arm for Collaboration with Humans in Industrial Environment (RACHIE) [39]. This robot's primary goal was to perform a simplified service pick and place task with human interaction to sort cans according to their colour and shape. By simplifying some of the tasks that generic service robots must perform, it allowed the development of a robotic arm that fulfilled all of the objectives previously stated. However, this manipulator did not provide the degrees of freedom necessary for some generic service robot tasks' complexity. The desired solution, to also comply with the anthropomorphic objective, should present similarities to a shoulder, elbow and hand redundant manipulator [40].

The solution implemented in CHARMIE is based on the InMoov arm [41], initially designed by Gaël Langevin. InMoov consists of the first Open Source 3D printed life-size robot. The main differences between the original InMoov arm and CHARMIE's arm reside on the bicep and shoulder. The whole arm is printable on a $12 \times 12 \times 12$ cm 3D, and its PLA parts weigh around 1.414 kg, with the actuators weighing around 0.766 kg. The whole

weight of the arm is approximately 2.2 kg. From shoulder to the hand, the arm measures 75 cm and can lift a maximum load of around 400 g.

This arm can be divided into three main parts: (i) hand and forearm; (ii) bicep; and (iii) shoulder. Most generic service robots with robotic manipulators tend to use grippers to simplify this task. However, the anthropomorphic hand presents more DOFs that allow the robot to best use the hand according to the object, providing greater dexterity capability [42,43]. The hand is composed of five fingers, similar to a human hand, to provide different ways to pick up various objects. Each finger has a different number of joints, as can be seen in Figure 5a. The thumb has two DOF, the index and middle finger have three DOF, and the ring and little finger have four DOF. Each finger has one actuator (servo motor) that moves all DOF of the respective finger. The movement is based on a pulley mechanism, similar to tendons, where the motors are located in the robot's forearm, as shown in Figure 5b with fishing line tendons leading to the fingers. So, regarding kinematic architecture, this is an underactuated hand since, in total, it has seventeen DOF, but only five actuators. The wrist has one servo motor responsible for rotating the hand. The bicep has two servo motors, one responsible for lifting and lowering the forearm, and the second responsible for rotating the bicep and the forearm. The shoulder has two individual DOF for moving the whole arm parallel to the robot body and lifting the whole arm perpendicular to the body. In total, as shown in Figure 5c this redundant manipulator has 22 DOF with 10 actuators, which define the movement of the arm. The Denavit-Hartenberg parameters for the arm are described in the Table 1.

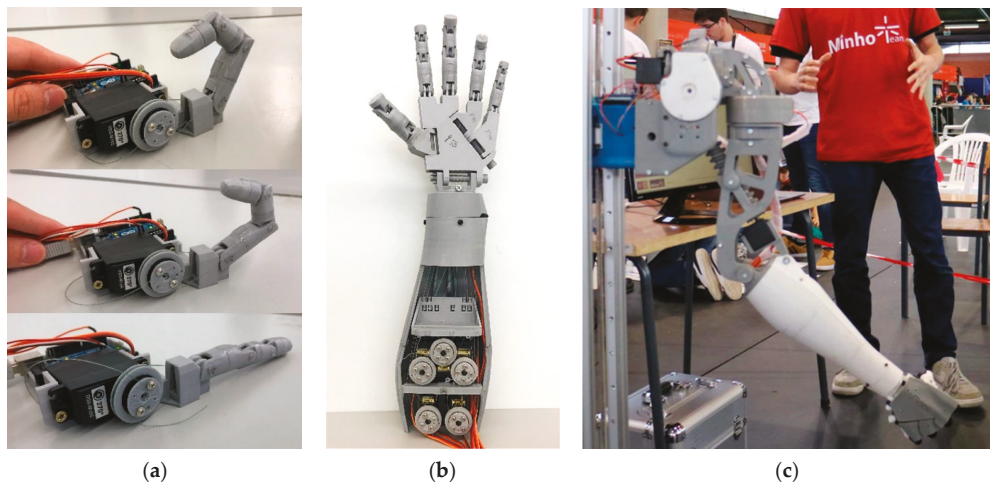


Figure 5. (a) An assembly of a finger, with three degrees of freedom but only one actuator, using fishing line as tendons. From top to bottom, the finger starts closed and opens until it is fully stretched. (b) The interior of the forearm has five servos actuators that individually control each finger. (c) The complete assembly of CHARMIE's arm.

Table 1. Denavit-Hartenberg from CHARMIE's robot arm.

θ (deg)	d (mm)	l (mm)	α (deg)
θ_1	0.00	40.02	-90.0
$\theta_2 + 90$	66.10	0.00	90.0
$\theta_3 + 90$	287.83	31.04	90.0
$\theta_4 - 26.6$	0.00	0.00	-90.0
θ_5	283.03	59.74	0.0

The four parameters are described as: θ_i the rotation around the Z_{i-1} axis by the angle between the links, d_i the translation along the Z_{i-1} axis of the distance between the links, l_i the translation along the X_i axis (rotated X_{i-1} axis) of the length of the link, and α_i , the rotation about the X_i axis of the twist angle.

For elderly care specifically, the arm and hand allow the robot to perform essential tasks. Picking up objects from the floor, low tables, counters and shelves and transporting them to the desired place ease many tasks that otherwise would either not be done or be done with many risks associated. With the addition of the anthropomorphic hand, the robot can easily interact with everyday objects specifically designed for the human hand with different types of grips and grasps. In Figure 4 solutions with both two arms and one arm are presented. The real-world robot and real-world tasks only uses one arm, as can be seen in Figure 5c, whereas the simulated robot already uses the two arms.

2.1.4. Robotic Head

The robot head holds the RGB-D camera, the multimodal user interface and the microphone. From a human interaction perspective, the head is the part of the robot the users look at when communicating with the robot, so it must be appealing and functional. Since the robots' head height could not be adjusted in the initial lifting mechanism system, a DOF was introduced to simulate a neck, so the robot could rotate its head, similar to a yes nod movement. This movement allows the robot to adjust its RGB-D camera angle to see objects at different heights. In the medium position, parallel to the motion platform, it can see objects on tables, shelves and people with an above-average size, as can be seen in Figure 6a. In the minimum position, at -60° from the horizontal field of view, it can see objects on the floor touching the front of the robots motion platform, as can be seen in Figure 6b. Lastly, at $+30^\circ$ degrees, it can see objects and people higher than the robot in a higher position, as can be seen in Figure 6c. For now, if the robot needs to see further to the sides, it rotates its motion platform, so it is always facing what it is trying to analyse. With the addition of the two new lifting mechanisms, the same system is used just for the RGB-D cameras. This camera is responsible for all visual related tasks such as: (i) user recognition; (ii) pose detection and tracking; (iii) gesture recognition; (iv) obstacle detection in navigation; (v) mapping; and (vi) object learning and recognition. Both the microphone and the multimodal user interface were part of the initial lifting system head. Being able to adapt both systems' height to the user's height allowed a cleaner interaction overall. The microphone is placed at the same height as the users head, allowing better voice recognition in environments with a significant noise level.

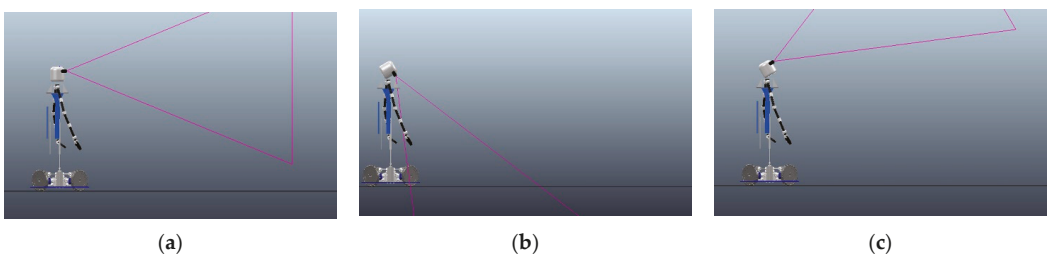


Figure 6. CHARMIE's field of view using the neck DOF. (a) Parallel to the motion platform (0°). (b) Looking down to see objects near the motion platform or on the ground (-60°). (c) Looking up to see items on top shelves ($+30^\circ$).

Moreover, the multimodal user interface also best fits the users' height to be more comfortable to interact with. Thus, from the initial lifting mechanism onwards, these two systems are now part of the lifting mechanism, since it is unnecessary to rotate them similarly to the RGB-D camera.

2.2. Components

To accomplish generic service and assistive tasks to aid in day-to-day life, for both the elderly and healthcare workers, CHARMIE must safely navigate in healthcare-related and domestic environments, perceive and track its human users, recognise gestures or poses and detect and manipulate different everyday objects. To achieve this, CHARMIE has a set of sensors and actuators that best fit both its environment and its tasks. With the sensorial data, the robot must perform some low-level cognitive functions that, when combined, allow the robot to perform more complex chores, both independently and collaboratively. The low-level functions can be classified into four groups of tasks: (i) map building and self-localisation; (ii) navigation; (iii) human-robot interaction; and (iv) object detection and manipulation. The training of all systems that require neural networks is made off-line and the main processor is a MSI Cubi 2 mini-PC with Core-i5 processor and 4 GB RAM.

2.2.1. Sensor System

To move safely according to the operating environment and with the purpose to perform the necessary tasks, CHARMIE must have an adequate perception of the following low-level functions:

- Map building and self-localisation;
- Safe navigation (obstacle detection and avoidance);
- Human-robot interaction (user and pose/gesture detection);
- Object detection and subsequent manipulation.

For map-building and self-localisation, CHARMIE uses two sensors: (i) 2D LiDAR (HOKUYO URG-04LX-UG01) mounted on the motion platform; and (ii) an RGB-D camera (Microsoft Kinect) located on the robot head. The laser range finder provided a 2D map of the environment near the floor, whereas the RGB-D camera provided a 3D map. The combination of both technologies allowed the robot to take advantage of the positive sides of both technologies. The 2D map could detect small objects on the floor or at the motion platform height, while the 3D map illustrates the complete environment map.

For safe navigation in various indoor environments such as hospitals, nursing homes and domestic houses, the robot uses the same sensors as the mapping and self-localisation tasks. Again, the combination of the 2D laser range finder and the RGB-D camera lets the robot detect different types of obstacles and react appropriately. The 2D detects all small objects on the floor that are harder to detect using the RGB-D camera, and the 3D information can detect all other objects at every height. In the motion platform, the motors have encoders embedded to close the control loop. Besides, the motion platform has current and voltage sensors for every motor. These sensors allow CHARMIE to know whether a wheel is not touching the floor, if a motor is stuck and if the robot is pushing against an object.

For human-robot interaction, CHARMIE uses the RGB-D to detect its users, their pose and some specific recognisable gestures. Recent works developed on this robot already started using a different RGB-D camera (Intel[®] RealSense[™] Depth Camera D455). One of the goals of CHARMIE is to communicate with its human users, both healthcare workers and older people, the same way humans communicate with each other by talking and hearing. Thus, the primary way to communicate with the robot is to speak some set of instructions that are interpreted and converted into tasks. This communication skill allows users who have not been familiarised with technology to easily interact and take the most advantage possible of CHARMIE. The robot has an MV5 Digital Condenser Microphone that allows the robot to hear 360° and a JBL GO Speakers. To overcome some difficulties that this communication system may present in particular situations, the robot has a multimodal user interface in its body so users can select the necessary tasks.

For object detection, the robot uses its RGB-D camera. The objects that need to be detected are usually on the floor, on counters and tables, or hand delivered by a user to the robot. Thus the focus lies primarily on objects that are either between 50 cm to 120 cm or laying on the floor. To grasp objects that are hand-delivered or on counters the

robot adjusts its lifting mechanism to best accommodate its redundant manipulators to the object's position. The same happens to the objects on the floor. That is why all lifting mechanisms can place themselves so that the robot arms can pick objects from the floor.

The robot's sensory system setup [44] is presented in Figure 7, where each sensor location is described on both implementations of CHARMIE's body. In the head, at a maximum height when the lifting mechanism is at the top position, the RGB-D camera is at 1.50 m which can rotate, tilting the head up and down. The initial RGB-D camera used is the Microsoft Kinect for all the tasks. However, with the recent acquisition of the Intel[®] RealSense™ Depth Camera D455 camera for CHARMIE, some human pose estimation tasks have already been developed using the new camera. The microphone, the speaker and the multimodal user interface lay on the torso with a maximum height of 1.30 m. The laser range finder is on top of the motion platform at approximately 25 cm of height. All the sensors related to the motors (voltage, current and encoders) are attached to the motors.

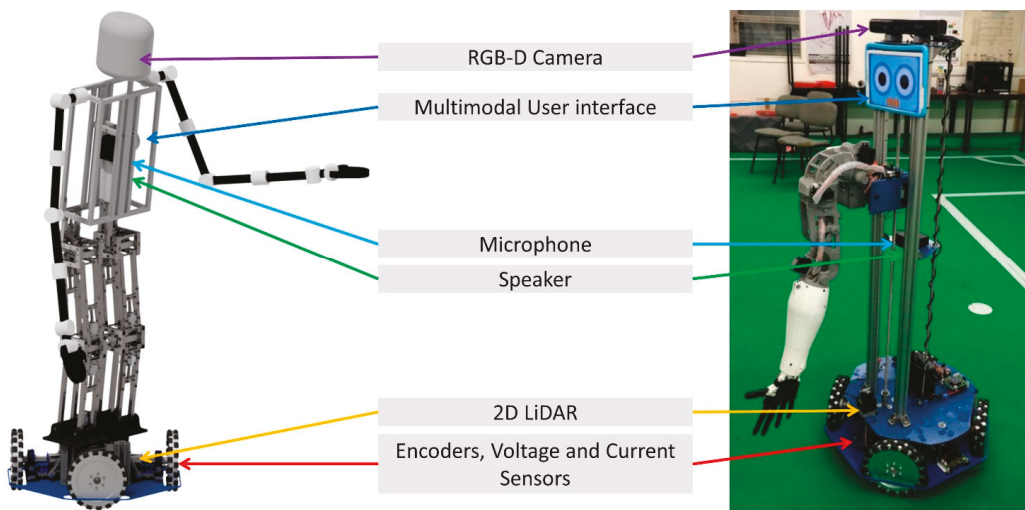


Figure 7. CHARMIE's sensorial description with all sensors' location used on both the physical robot and the anthropomorphic implementation.

2.2.2. Map Building and Self-Localisation

To perform mapping of the environment, it is necessary to detect the points of interest known as keypoints. Additionally, to calculate travelled trajectory, it is necessary to quantify the movement occurred in-between frames. Rather than using all the pixels from an image to detect keypoints, the FAST [45] algorithm presents a computationally more efficient strategy than other similar solutions. This algorithm works by creating an adjustable bounding circle on every pixel that might resemble a corner. To be a corner, three nearby pixels inside the bounding circle must have a higher intensity than the fourth with a factor μ . Also, inside the bounding circle, there must be a set of collinear points with intensity higher than μ . These two conditions must be satisfied to consider this region a corner, and thus a keypoint. Next, to estimate the visual odometry using just the camera, since Microsoft Kinect does not have an IMU (Inertial Measurement Unit), the selected algorithm uses monocular odometry. It detects the keypoints using the FAST algorithm on consecutive frames and associates these between frames using essential matrix estimate through LMeds algorithm.

The method selected to build the 3D map of the environment was using OctoMap [46], based on octrees to group all points. This method loses part of the detail, which translates into higher computational efficiency. For CHARMIE's purpose, it is more valuable to have

a time-efficient algorithm than to have 3D mapping with great detail, since the robot only needs the map to safely navigate to the desired position. Figure 8 shows a point cloud converted into an OctoMap in three different perspectives representing an object on top of a table and a guitar next to the wall. Figure 9 shows an example of a complete indoor environment (office) mapped by CHARMIE.

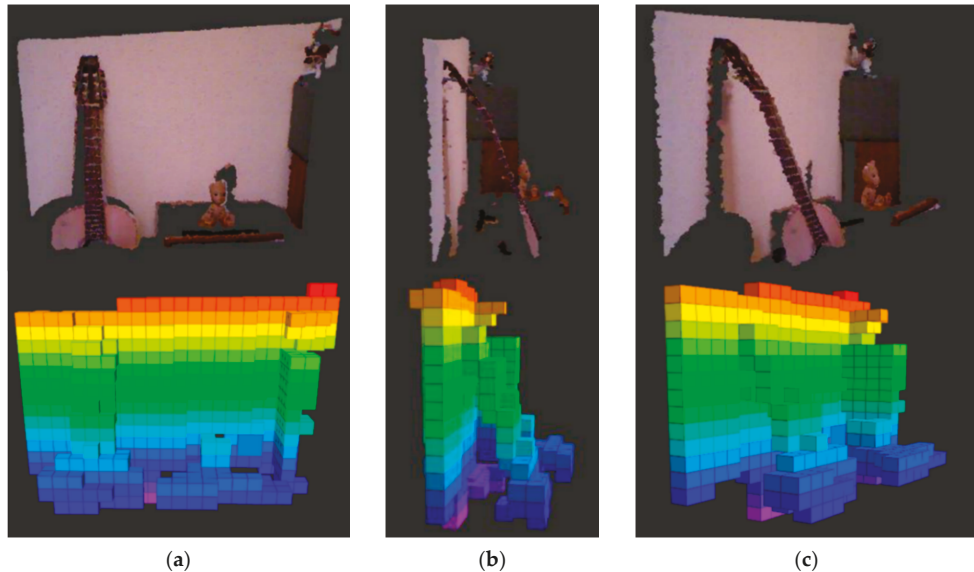


Figure 8. Three different perspectives (a–c) of the same two objects, a toy on a desk and a leaning guitar on a wall converted into an OctoMap to simplify mapping high detail objects.

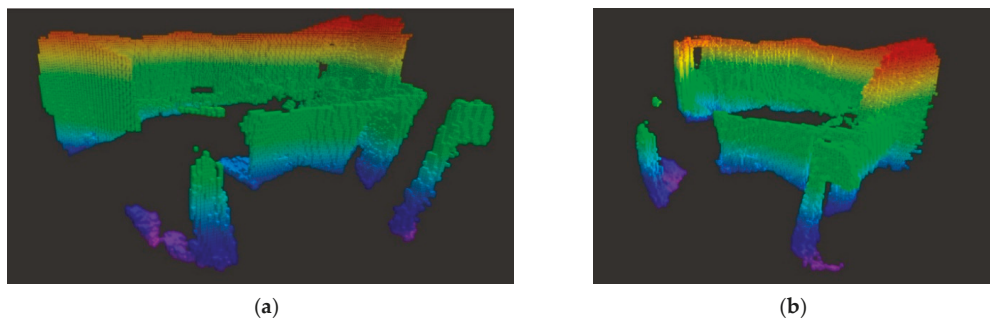


Figure 9. A complete map of an indoor environment, more precisely, an office. After some movement inside the office, CHARMIE created the OctoMap. In the middle of the image, a desk can be seen, two pillars on the bottom are speakers, and the remaining information is mainly regarding the walls. (a,b) demonstrate two different angles of the map.

The mobile platform's self-localisation is done using Adaptive Monte Carlo Localization (AMCL) [47]. In this algorithm, the robot's pose is represented as a set of multiple hypotheses concerning a prior known map. AMCL combines the information from the mobile platform's odometry and the 2D LiDAR. It starts by performing global localisation, so it is immune to the initial position and, after knowing its location, only performs local tracking using adaptive particle filters.

2.2.3. Navigation (Obstacle Detection and Avoidance)

To safely navigate a previously mapped environment, CHARMIE must detect dynamic and static obstacles that might not be in the environment map and navigate accordingly to overcome these.

Regarding obstacle detection, the robot uses the same sensors as in the mapping function, the 2D LiDAR and the RGB-D camera. The 2D LiDAR is used for small obstacles at the motion platform height. It starts by checking if there is any obstacle inside a 1.50 m radius and calculates both the obstacle position relative to the robot and its size. The RGB-D camera starts with the same principle regarding the 1.50 m radius, calculates its position relative to the robot and size, creating a virtual obstacle from the floor to the robot's height. In Figure 10, an example of the same image from three different angles is displayed, showing an example of data fusion between the RGB camera and the Depth camera. The obstacles derived from the sensors are combined in a temporary virtual obstacle map that is constantly updated. By applying this method in consecutive frames, the direction of movement of dynamic obstacles is also added to the virtual obstacle map.

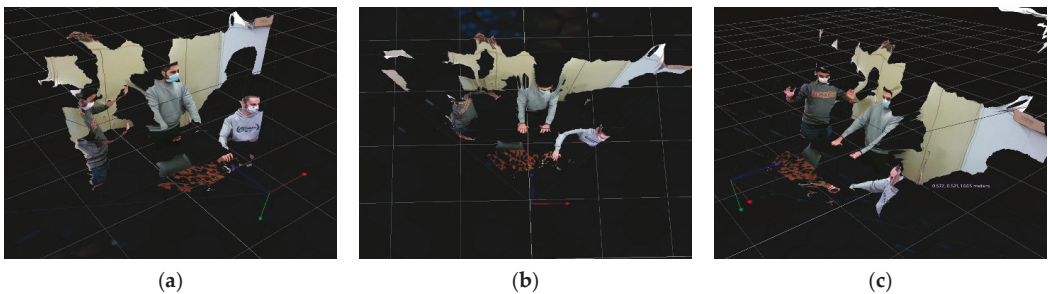


Figure 10. Three different perspectives (a–c) of three human users interacting with CHARMIE. The robot fuses the RGB and Depth cameras' information to create a 3D view of the environment.

After parameterising all the static and dynamic obstacles in a nearby radius and defining a target location, CHARMIE uses dynamic non-linear systems [48,49] as a distributed control architecture that generates navigation. Task constraints are component forces that are cast together into the vector field of this dynamical system. For example, the directions $\varphi = \psi_{obs}$ (where obstacles are from the robot's viewpoint) and the directions $\varphi = \psi_{tar}$ (where the target is) are constraints represented by repulsive and attractive forces acting on the heading direction. The attractive force attracts the system to the desired heading direction value, whereas the repulsive forces prevent the system from moving in an undesired direction. As the robot moves, the directions to the target and obstacles in the world varies, and consequently, the attractor and repellers move in the vector field.

As stated above, the virtual obstacle map foresees the direction and size of all obstacles that must influence the robot's trajectories according to the robot's reference. So, a repulsive force is applied to all obstacles:

$$f_{obs,i}(\phi) = \lambda_i(\phi - \psi_i) \exp \left[\frac{-(\phi - \psi_i)^2}{2\sigma_i^2} \right], i = 1, 2, \dots (\text{no of obstacles}), \quad (1)$$

where ϕ is the robot direction, ψ_i is the obstacle direction, thus $(\phi - \psi_i)$ is the obstacle direction relative to the robot. σ is the angular magnitude on which a repulsive force acts, defined as:

$$\sigma_N = \arctan \left[\tan \left(\frac{\Delta\theta}{2} \right) + \frac{R_{robot}}{d_i} \right] \quad (2)$$

where $\Delta\theta$ is the angle the robot occupies, R_{robot} is the robot radius and d_i is the distance from the robot's centre to the obstacle. λ_i is the maximum repulsion force, defined as:

$$\lambda_i = \beta_1 * exp\left[-\frac{d_i}{\beta_2}\right] \tag{3}$$

where β_1 controls the maximum repulsion strength, and β_2 controls the decay rate with increasing distance. The repulsors contributions from all obstacles are summed, creating the repulsor vector field. To get the target position, it starts by transforming the target coordinates into the target direction and apply an attractive force as:

$$f_{tar}(\phi) = -\lambda_{tar} * sin(\phi - \psi_{tar}) \tag{4}$$

where ψ_{tar} is the target direction and λ_{tar} is the attraction force magnitude. To finish the dynamic field vector system, all contributions are added.

$$\frac{d\phi}{dt} = \sum_{i=1}^N f_{obs,i}(\phi) + f_{tar}(\phi) \tag{5}$$

In Figure 11, two different scenarios regarding nearby objects are presented. Figure 11a shows three obstacles in red that can be seen inside the maximum security distance circle. Consequently, the repulsors are created considering their distance to the robot and their size, represented in the second graph. In the first graph, the attractor (pink) and the sum of all repulsors (green) are totalled to create the dynamic field vector system (black) that yields the angle to which the robot should rotate to avoid the obstacles. Similarly, in Figure 11b, all the graphs shown represent the same variables as in Figure 11a.

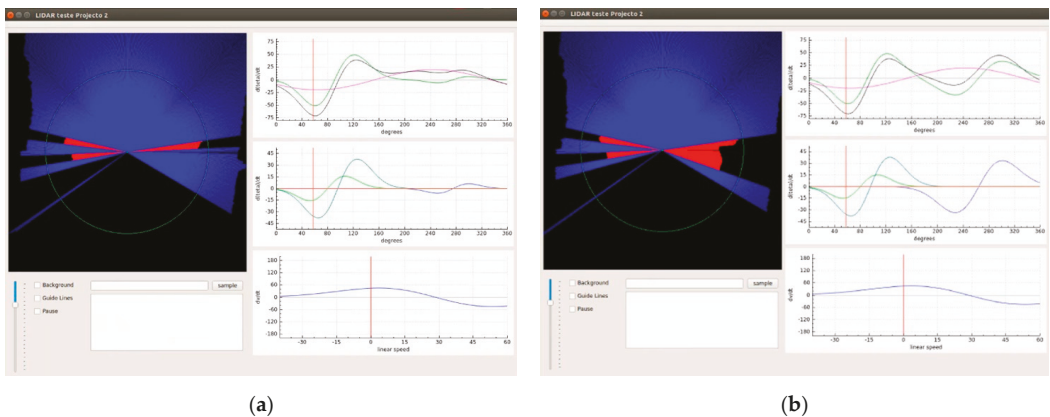


Figure 11. Two different variations (a,b) of the dynamic field vector system are displayed depending on the various obstacles that the robot faces. The blue and red 2D graph on the right of both images represents the 2D LiDAR data. The bottom graph represents the velocity graph with one attractor to the desired speed. The middle graph represents the influence of the various obstacles the robot faces and consequential repulsors. The top graph represents the angle attractor (pink), the sum of all repulsors (green) and the final dynamic field vector system (black).

The difference between the two images lies in the increase in the obstacle size at the right. This translates into a more aggressive repulsor on the right side, represented in blue in the middle graph. Consequently, both the sum of all repulsors (green) and the dynamic field vector system (black) will also be more aggressive. The third graph of each image represents the speed of the robot, also manipulated by an attractor.

To fit CHARMIE’s medium-term objective of using machine learning algorithms to solve all of the proposed tasks, a different implementation of the robot’s autonomous movement using reinforcement learning was developed. Deep reinforcement learning is primed to revolutionise the field of artificial intelligence. It represents a step towards building autonomous systems with a higher-level understanding than any other learning technique. The application of deep reinforcement learning to robotics allows robots to learn control policies directly from real-world sensorial information through trial-and-error interactions. The tasks performed by service robots such as CHARMIE can be characterised by long-term planning, high-dimensional continuous action-space, and in most cases, incomplete information. Even though this is a problem for some reinforcement learning algorithms, novel solutions such as those presented in [50–53] already solve a wide range of simulated tasks similarly characterised. The algorithm selected to implement into CHARMIE’s motion platform is based on Q-Learning [54,55]. The reinforcement learning setup consists of an agent (CHARMIE motion platform) interacting with the environment (simulated indoor environment) in discrete timesteps. At each timestep, the agent receives an observation, takes an action and receives a scalar reward. The agent uses its sensory information, the 2D LiDAR, to navigate towards the target position avoiding the obstacles that come up in its way, similarly to [56]. A mapless motion planner can be trained end-to-end with no manually designed features nor prior demonstrations through this reinforcement learning method. This trained planner can be directly applied to never before seen environments. In Figure 12, it is demonstrated the evolution of how the reinforcement learning algorithm, without any previous knowledge about the environment, solved three iterative complexity mazes. The first more straightforward maze is learnt just by trial-and-error, and the two following mazes use the knowledge gathered from the previous maze.

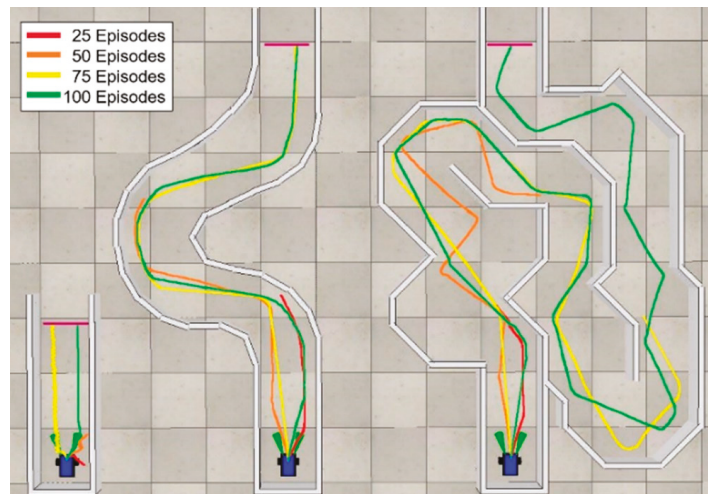


Figure 12. Trajectory evolution throughout the learning process of a differential robot autonomously solving a maze using Q-Learning. Image from [54].

2.2.4. Human-Robot Interaction (User and Gesture Detection)

One of the biggest concerns regarding the CHARMIE project is to provide communication tools to ease communication with users, both older people and healthcare workers. To initialise a communication process, the robot must first recognise its users and their pose/gestures. The functionalities regarding user detection demonstrate solutions using the RGB-D camera, both with and without the depth image. The initial 2D visual user detection algorithm focuses on detecting faces, then crop and align the detected faces and recognise which previously trained, or new users are attempting to communicate.

For this purpose, a Multi-task Cascaded Convolutional Network (MTCNN) [57] is used. It consists of three convolutional networks with different architectures with increasing complexity attached to each other, where the output of the previous net is the input of the next. The first network is known as P-Net (Proposal Network), which generates different proposals of where the faces must be, used as inputs to the following network. Next, the R-Net (Refinement Network) analyses the proposals and filters false positives. The final network, O-Net (Output Network), creates the final output, the detected face's image and its facial landmarks.

To classify the faces, the detected faces output by the MTCNN are introduced as input in an Inception-ResNet v1 neural network [58], transforming it into an image representation vector in space, also known as embedding. This network combines the Inception and ResNet architectures. The Inception architecture uses multiple inception modules, which have different convolutional-layers with different kernel sizes operating in parallel. These filter the same level layer in the architecture, concatenating in the next level, finding various features with fewer convolutional-layers. This process makes the network less deep without losing information. The Resnet network introduces residual connections. In traditional neural networks, a layer feeds data to the next one, but with this algorithm, it sends direct information to a deeper layer. These blocks improve two areas. The training time, given that skip-connections can jump layers without training and the lost information when making gradient descent since more deep networks have a hard time being accurate without overfitting or working more straightforward tasks.

The last step is implemented through an SVC-C (Support Vector Classification), with training and a test dataset. This supervised learning algorithm separates the different sample classes, known as embeddings, using a hyperplane where the margins are optimised through support vectors. Figure 13a shows an application of the MTCNN algorithm, detecting different famous people's faces. It was trained to detect 64 different personalities with the "Labeled Faces in the Wild dataset" [59]. Only using information from the 2D camera it could successfully identify all of the trained people on the test set with certainty percentages of over 60% in the worst cases. Additionally, Figure 13b shows the testing dataset's mapping with PCA (Principal Component Analysis), where the same label/output images are projected close to each other while distanced from different ones. For better visualisation purposes, the photos also have a colour differentiation filter. Even though it can detect 64 different personalities, to maintain the same levels of accuracy, CHARMIE is set to detect 20 different people.

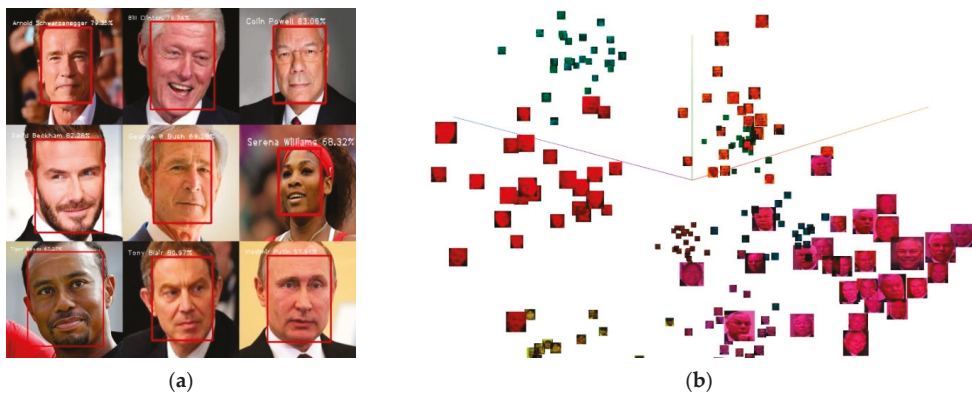


Figure 13. (a) The MTCNN algorithm's output using "Labeled Faces in the Wild dataset" [59] of nine different personalities. (b) The PCA graph from Tensorboard from the MTCNN test dataset.

When updating to 3D user recognition technology, some works that use the same camera (Microsoft Kinect) [60] show algorithms that can overcome different face poses, expressions, illumination and disguises. Other solutions, such as FaceNet [61], demonstrate a deep convolutional network trained to directly optimise the embedding itself. The depth solution CHARMIE employs to recognise a small number of users uses various deep convolutional neural networks, similar to the one with just the RGB that integrates both the RGB and the depth image. Figure 14a demonstrates a classification example of two users using the depth information in addition to the RGB. The lack of different trained users can justify the significantly higher values of certainty compared to the 2D solution. However, some problems, such as lightning and the pictures of users, can be successfully filtered with the depth camera's addition. An example of the lightning condition being filtered is shown in Figure 14a right image. Figure 14b shows the two images input to the neural networks, the RGB and the depth matrix.

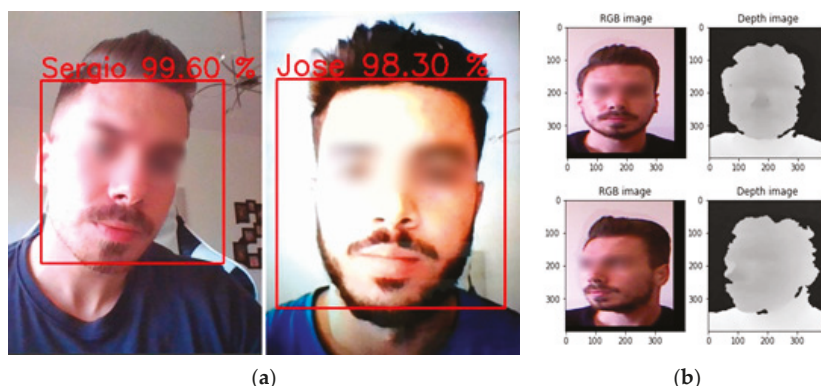


Figure 14. (a) The output of the 3D user recognition system. The image on the right demonstrates the neural networks filtering bad lighting. (b) Two examples of RGB and depth images used as input to train the 3D user recognition system.

After recognising the users, the robot is establishing an interaction with, CHARMIE analyses their poses. Essential information can be interpreted from the pose: (i) the human's position, standing, sitting and laying, among others; (ii) an estimate of their intention to perform an action; (iii) whether they are pointing at something; and (iv) an analysis of their motion when doing collaborative tasks. Skeleton tracking processes depth image data to determine multiple skeleton joints' positions on a human body.

The cameras distinguish a human from the background and identify the position of several features or joints, such as the head, knees, elbows and hands. Once identified, the software connects the joints into a humanoid skeleton and tracks their position in real-time, providing the X, Y and Z coordinates for each of the skeleton points. The addition of depth cameras [62] allows the skeleton tracking system to remove uncertainties between overlapping or occluded objects or limbs, making the method more robust to different lighting conditions than a 2D camera-based algorithm [63]. The algorithm used is based on the Skeleton Tracking SDK by cubemos. It provides fast and highly accurate 2D and 3D human pose estimation that allows tracking of 18 joints simultaneously (two ankles, two knees, two hips, two wrists, two elbows, two shoulders, one between the chest and the neck, one nose, two eyes and two ears). Due to the artificial intelligence algorithms, it can track up to five people in real-time. Figure 15 shows the skeleton-fitting pose estimation overlaid on the users. It can detect high-speed movements and estimate a joint location, even when hidden from the camera. Figure 15c shows that the robot with pose only estimation can detect when a person raises their arm in the air to ask for assistance.

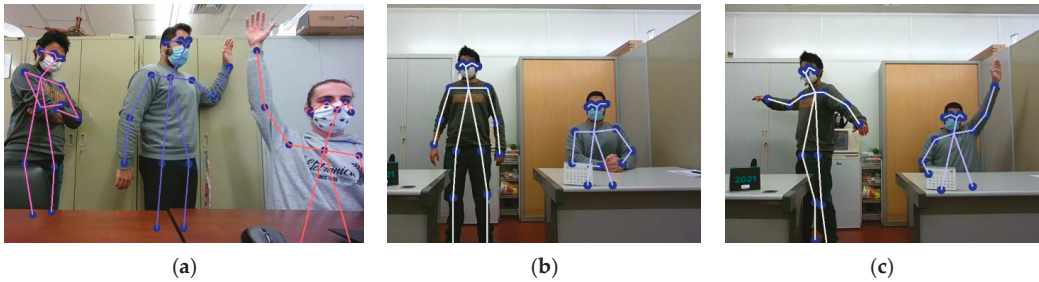


Figure 15. Three different situations (a–c) where CHARMIE estimates the humans' pose to further analyse whether the users require its help, their position, their movement and their intention to act.

The joints tracked by the algorithm also allow the agents to communicate or transmit information using different gestures. Some technologies like optical flow provide solutions to track movement between images. This demonstrates the movement from people or animals, as long as it is in a different direction from the camera's movement and allows 2D verification of the movement direction. FlowNet2 [64] presents an end-to-end solution based on convolutional neural networks to estimate optical flow. It uses various methods that allow estimating movements at both quick and slow speeds. All of the methods have different purposes, and their combination, despite providing good results, is very computationally expensive. So, LiteFlowNet [65] introduces a different end-to-end convolutional neural network architecture to estimate optical flow. This network, used by CHARMIE for gesture recognition, uses an optimised neural network structure whose goal is to have results with the precision of FlowNet, but with a lower computational expense. In Figure 16, the human agent performs a gesture, from (a) to the (c), where it shows the palm of the hand to the robot, then closes the fist and brings it to its chest. The robot can detect different sets of movements that can be configured and associated with a specific task the user intends the robot to do.

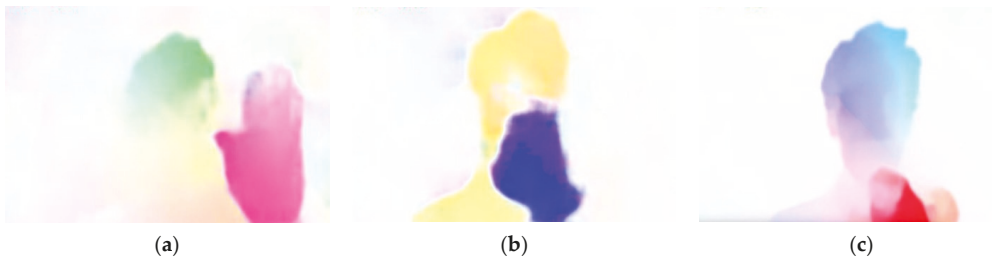


Figure 16. Optical flow image from a known gesture. From (a–c), the user shows the palm, then closes it and brings it to its chest. The different colours represent both different moving directions and speed.

Apart from the visual human-machine interfaces, the robot has two more communication systems. One is through speaking and listening, similar to a human-human conversation. CHARMIE uses CMU Sphinx tools from Carnegie Mellon University for speech recognition and Emic 2 Text-to-Speech Module to perform text to speech conversion. All conversations with the robot must be made using the English language. The most significant advantage of this method is that the users do not need any prior knowledge regarding the robot to successfully communicate with it. The robot recognises sequences of keywords from the human. Even though a robot may not understand the entire conversation from a user, it understands keywords and confirms if the information received is correct by asking the user if what is understood is the correct answer. Some examples of keywords are names of its users, names of different rooms, names of objects, and actions.

The robot's response may vary according to its perception, location, priorities, whether it is performing a task or moving somewhere. Every sentence a user says to the robot must start with the word "CHARMIE", so the robot knows whether the conversation is towards it or not. In the video, in Appendix A, it is possible to see CHARMIE introducing himself and some conversations with users.

The last human-machine interface the robot has is a multimodal user interface. In environments where there is significant noise level, the user presents difficulties speaking, or the user cannot make a predefined gesture, this system can be used. A tablet with a menu lets users select all features that the previous human-robot interactions presented at the robot's torso, with the addition of being available in languages other than English.

2.2.5. Object Detection and Subsequent Manipulation

For learning and recognising objects, both healthcare-related and household items, CHARMIE uses the supervised learning algorithm named YOLO (You Only Look Once) [66]. YOLO is a state-of-the-art, real-time object detection system known for its high-speed and accuracy. The YOLOv3 [67] algorithm starts by separating the image into a grid. Each grid cell predicts several boundary boxes around objects that score highly with predefined classes. Each boundary box has a respective confidence score of how accurate it assumes the prediction must be and detects only one object per bounding box. The boundary boxes are generated by clustering the ground truth boxes' dimensions from the original dataset to find the most common shapes and sizes. Unlike other models, YOLO looks at the entire image when testing, so its prediction reflects the image's global context. It makes predictions with a single network evaluation, unlike systems such as R-CNN which requires thousands evaluation systems for a single image. This makes it extremely fast; more than a thousand times faster than R-CNN and a hundred times faster than Fast R-CNN. However, YOLO is not ideal to use with models where large datasets may be hard to obtain. Even with its high speed, YOLO is not fast enough to run on embedded devices such as a Raspberry Pi. To help make YOLO even faster, the algorithm creators defined a YOLO architecture variation called Tiny-YOLO. This architecture is approximately 442% faster than YOLO and can achieve 244 FPS on a single GPU. Since Tiny-YOLO is a more compact version, this also means that it is less accurate. The architecture that is used in YOLOv3 is called DarkNet53 [67]. With its 53 layers of convolutions and no max-pooling, its main job is to perform feature extraction. A BatchNormalization and a leaky RELU follow each convolution operation. Darknet53 architecture is proved to be an extremely efficient network regarding object classification. CHARMIE uses Tiny-YOLOv3 architecture to detect different healthcare-related and household objects. Figure 17 shows some of the things that the robot has already learnt to detect, like bottles, cans and bags of chips. To introduce new objects into CHARMIE's database, CHARMIE records a video of the item, collecting all the frames. This information is later used to retrain Tiny-YOLOv3.

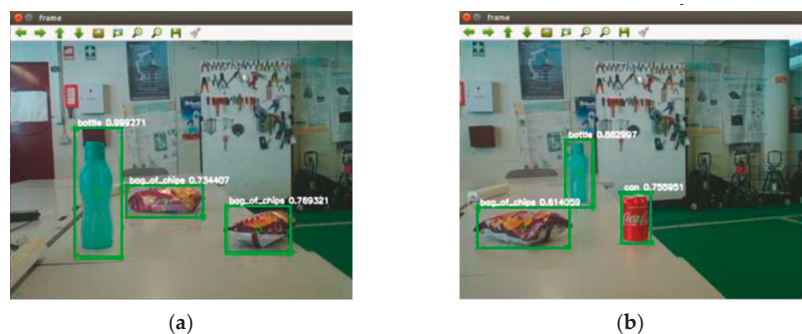


Figure 17. Two different outputs (a,b) from the YOLO detection algorithm that detect and locate various pre-trained household objects simultaneously in real-time.

To grasp the detected objects, CHARMIE uses its redundant manipulator, the robot arm. Some objects with unusual shapes and whose shape changes, such as a bag of chips, have different programmed collection algorithms. Nevertheless, for most of the items, the picking-up system is similar. After detecting the desired object, CHARMIE calculates the inverse kinematics to place the robot arm and the lifting mechanism in the best position to collect the object. Then the hand-closing is performed according to the object's physical properties. Figure 18 demonstrates an example of picking a can. As stated, the robot moves its platform and lifting mechanism to best fit the item's position, as can be seen in Figure 18a. The arm is moved right next to the object that wants to be manipulated, as can be seen in Figure 18b, and moving the hand right next to it, as can be seen in Figure 18c. In this case, it starts by using the thumb as a back wall, as can be seen in Figure 18d, and then it starts closing the fingers one by one from the index finger to the little finger, as can be seen in Figure 18e. This movement allows the robot to pick up the can, similar to how a human picks it up, as can be seen in Figure 18f.

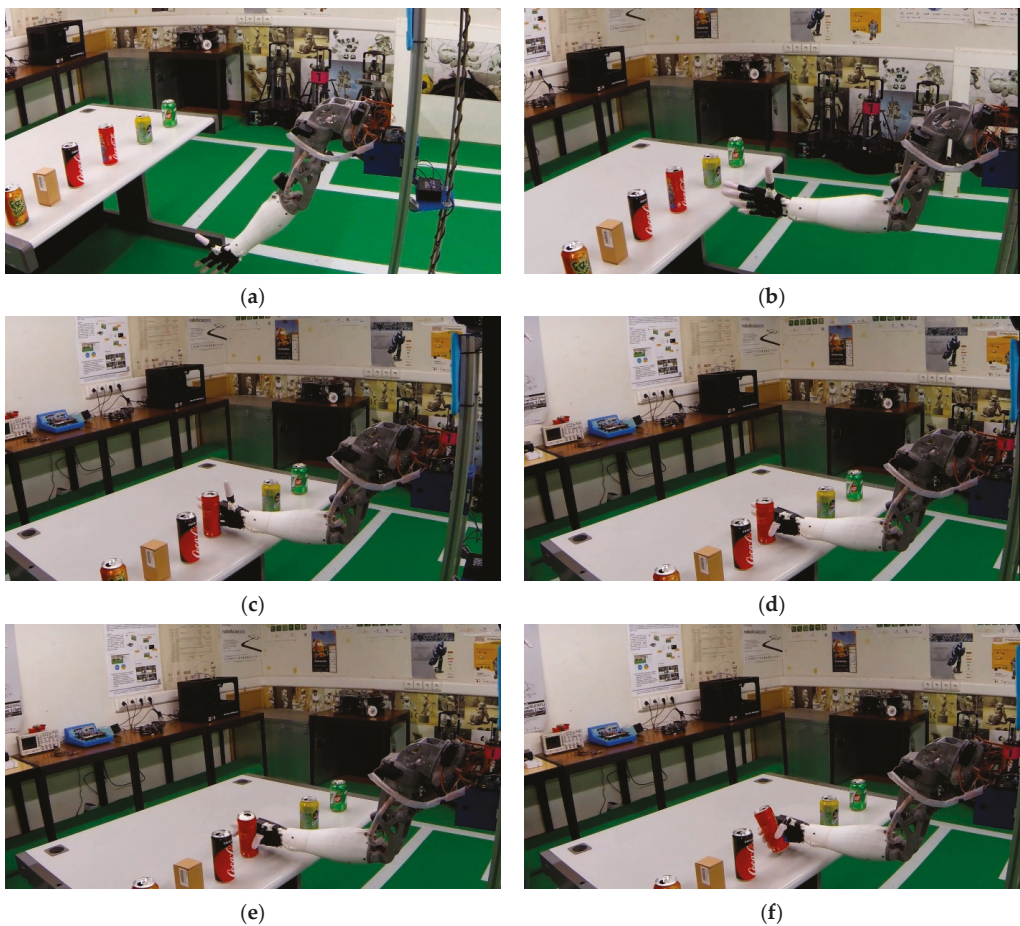


Figure 18. A step-by-step demonstration of CHARMIE's process of picking the desired item to furtherly be used in subsequent tasks.

3. Results

CHARMIE performs a wide range of tasks as a service and assistant robot combining the four low-level functions previously stated. When developing algorithms to perform new tasks, the central focus lies mostly on how helpful these are for elderly or healthcare workers. By aiding both, CHARMIE can provide a higher quality of life for older people. Some central problems the robot tries to tackle regard tasks: (i) where older people have a lack of mobility (picking objects from the floor); (ii) where they have a lack of strength (carrying heavy objects); (iii) that are safety-related (if a person falls and cannot get up); and (iv) that happen on a day-to-day basis and that may require a lot of energy and include injury risks. As stated, CHARMIE can perform a wide variety of tasks. Of those, five chores that encompass different scenarios and different interactions are thoroughly explained.

3.1. Help Me Carry This Bag

One of the most severe difficulties for older people is carrying heavy objects. A common practice among the elderly is to shop for groceries with higher regularity, which translates to less weight but more trips to the stores. For now, CHARMIE is only intended for indoor use, with some exceptions. Thus, groceries-carrying tasks are idealised to receive the bags at the environment entrance and be further transported to the desired location. Figure 19 shown all the steps CHARMIE underwent to complete this task. Additionally, this task can be seen in the video in Appendix A.



Figure 19. A step-by-step demonstration of CHARMIE’s “Help me carry this bag” task. From receiving a task from the first user, collecting the bag, navigating to the desired location and hand-delivering it to the second user.

The user starts by initialising the dialogue requesting help from the robot to collect and transport the grocery bags, as can be seen in Figure 19a. It must indicate where the robot must transport the loads, and optionally if these must be given to another known user. After receiving this information, CHARMIE expresses that it is ready to receive the bag for transportation, and asks the user to place it in its hand, as can be seen in Figure 19b. The robot detects when the user places the load in its hand through human pose estimation and confirms it via the force the arm actuators' must provide, as can be seen in Figure 19c. Furthermore, the robot starts moving to the delivery location while performing safe navigation with static and dynamic obstacle avoidance, as can be seen in Figure 19d. If the robot does not have to deliver the grocery bags to another user, it analyses the table surface and tries to find an empty spot to drop the bags. In the situation of not having an open space for the bags, the robot would wait for a user to clarify where the bags should be placed. If the robot's task is to hand in the bag to a human user, it searches for the user when arriving at the desired location. If no user is there to receive it, the robot tries to place it on the table. Otherwise, the robot moves in the user's direction and asks if he/she is available to receive the bag, as can be seen in Figure 19e. If the answer is no, the robot patiently waits until the user is available. When the user confirms its availability, the robot extends its arm in the human's direction and asks the user to collect the bag, as can be seen in Figure 19f. Afterwards, the robot returns to the initial user to verify whether there are more grocery bags to transport.

3.2. *Can You Find This Item and Bring It to Me*

With ageing, older people tend to have more difficulties getting up from a sitting position. Additionally, with mental health deterioration, there is a tendency for memory to start failing and forgetting an object's placement. One of the main chores CHARMIE performs is to collect things and return them to the user. Some examples of objects might be medicine boxes, cans, bottles, cellphones and remote controllers. A variation of this task is when a user does not know the exact location of the object it asks the robot to retrieve. In the example provided for this task, a scenario is presented where the user is laying in bed, not feeling very well and needing to drink some water.

The user starts by calling CHARMIE, and upon arrival the robot sees that the person is laying in bed and stays alert to the task it must do. Through dialogue, the user indicates that he/she is not feeling very well, and cannot get up, but needs to drink some water. In this situation, the user can say where the drink is or say the location and the robot must find it. CHARMIE starts moving towards the kitchen through safe navigation. When arriving at the kitchen, if the user stated where the drink is, the robot goes directly to that location. If the user did not define the area, the robot must look around the kitchen searching, starting with open spaces like the counters, tables and open shelves. After detecting the specified drink, the robot picks up the object with its redundant manipulators and, if possible, closes the opened kitchen cabinets. In Figure 18a scenario is displayed where CHARMIE has various drinks and objects placed on a table, and it must analyse and collect the required drink. The robot returns to the user and asks if it is available to receive the drink. When the user is available, the robot extends the arm in the human's direction, and waits for the user to collect it through pose estimation. After finishing, the robot asks whether the user is feeling better and if it needs anything else.

3.3. *Check on the Patients*

One of the major causes of serious injuries in the elderly has to do with falls. The reflexes to protect themselves start to degrade, and older people lose balance and movement capabilities, which may cause serious falls. In situations where the person lives alone, this is extremely dangerous, since the person is only analysed for potential injuries when someone goes to their house. One of CHARMIE's most safety-related tasks consists of patrolling indoor areas and, through pose estimation, understanding if a person is in danger. The robot can patrol nursing homes or hospitals at night to check if the patients are laying on

their beds, lowering the detection time for an older person who fell. CHARMIE can also check if a person is not on the bed or even sitting down needing some assistance. In cases where older people live without any health professional, the robot can quickly send an alert message to the emergency contacts.

In the nursing home patrolling task, CHARMIE uses the known environment mapping where the patient rooms are defined. By patrolling the bedrooms, the robot starts by very calmly opening or pushing the door and slowly moving inside the bedroom to not scare or wake up the patients. As displayed in Figure 20, the robot may encounter various scenarios when estimating the human's pose to evaluate patient safety.

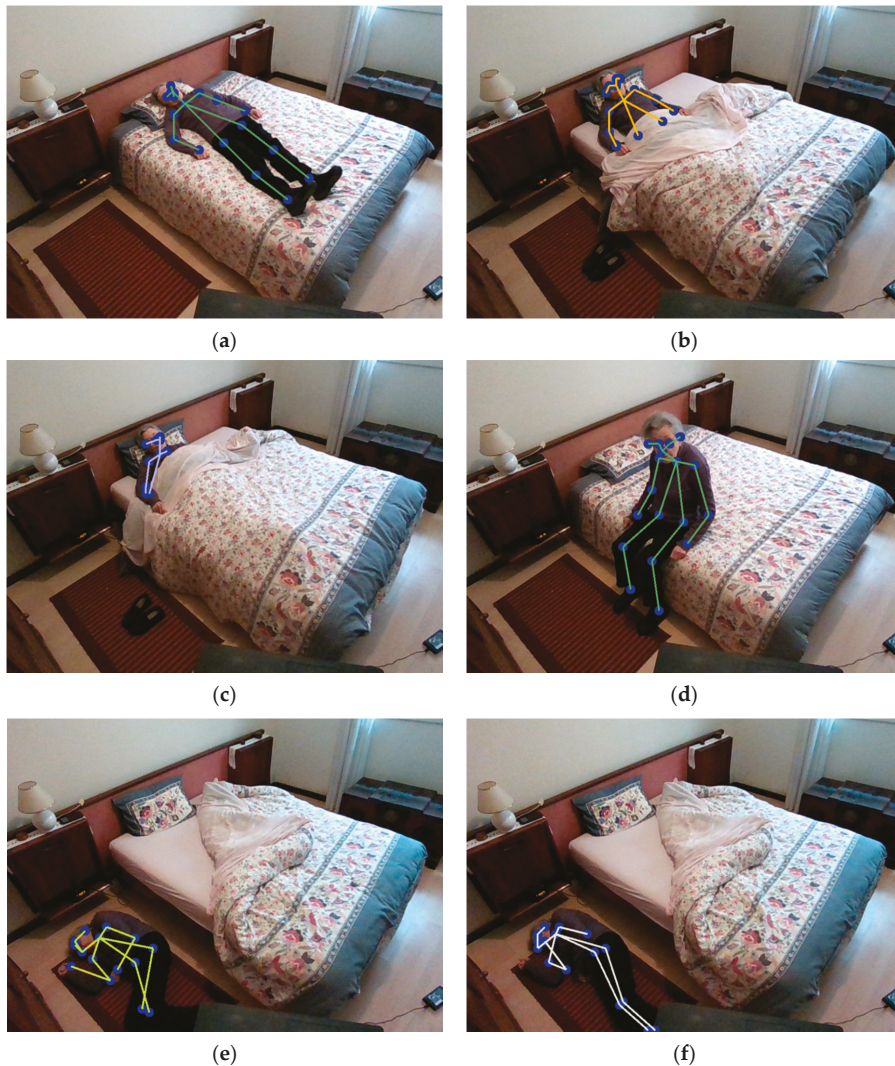


Figure 20. The different possible scenarios CHARMIE may encounter when patrolling nursing homes and the respective pose estimation output. (a) shows an elderly person laying in bed over the covers. (b) shows an elderly person laying under the covers, covered from the waist down. (c) shows an elderly person laying under the covers, covered from the neck down with one arm outside the covers. (d) shows an elderly person sitting in bed. (e,f) show an elderly person who fell from the bed and is laying on the floor.

The first possible scenario is the detection of a patient laying in bed. However, this scenario may present some different variations. The first, displayed in Figure 20a, shows the pose detection of a patient laying on the bed without any covers. The second, shown in Figure 20b, demonstrates a patient laying in bed but covered from the waist down. The third, Figure 20c, shows a patient laying in bed but covered from the neck down with an arm also showing. In the first two cases, the pose estimation algorithm can detect properly that the user is safely laying in bed, but the third example does not always detect successfully. Thus, when the robot cannot estimate the pose, it analyses the bed height variation to differentiate between the patient laying in bed and the bed being empty. If the patient is not laying in bed and no pose estimation is made inside that room, two different situations may occur, the patient is either missing or sleeping deep inside the covers. If the patient is laying on the bed, the robot calmly leaves the room and slightly closes the door. By analysing the bed height variation, should it detect there is no patient in the room, it sends an emergency warning to the healthcare workers that a patient might be missing. Another scenario results when a patient is sitting on the bed, Figure 20d, and requires non-emergency help. In this case, CHARMIE also summons a healthcare professional but in a different way to the first case. The last scenario displays the worst case: an elderly person has fallen out of bed and cannot get up, as can be seen in Figure 20e,f. If CHARMIE detects this scenario, it immediately sends an emergency message to all of the healthcare professionals. This recognition process is repeated throughout all of the rooms in the nursing home. For an older person who lives by themselves, this recognition system can be temporarily programmed.

3.4. *Store the Groceries” or “Clean up the Room*

CHARMIE can also execute some tasks regarding cleaning and tidying up. Even though these tasks do not require carrying heavy loads, they involve a different physical restriction that may end in injury. When tidying a room, older people may have to place themselves in positions, such as picking objects from the floor or stretching to reach the top or bottom shelves that may lead to accidents. With the goal of helping the elderly, and at the same time freeing healthcare workers to focus on more critical patient-related chores, CHARMIE can analyse these environments that need to be cleaned. By perceiving which objects are out of place, the robot can collect and place those in areas previously indicated. To describe this task, two different variations are presented.

To store some groceries laying on the kitchen table, the correct places for every object need to have been previously indicated to the robot. After the bags have been unloaded, the robot analyses the various things it must store. If an object is not in the robot’s database, CHARMIE asks the user the correct spot to place the item. When the robot is storing this product, it captures images from different angles to later add to its training data. The remaining known products are packed in the correct place using CHARMIE’s redundant manipulators. If the robot comes across a full shelf when attempting to place an item, the robot returns the item to the kitchen table and informs the user.

When cleaning a specific room, the robot starts by analysing the environment, extracting all the information regarding all objects on the floor or tables. With its depth image, the robot can differentiate objects from flat surfaces and save their location. Similarly to the storing groceries example, the robot has to already know the object’s place. Contrary to the groceries examples, the places where the items go are varied height-wise, which forces the robot to adapt its height to successfully manipulate the out of place objects.

3.5. *Follow Me” and “Lay the Table*

Another beneficial task of a service robot is to follow a user who needs help performing a task. At times, CHARMIE might be in a different compartment than where the user needs it to complete the next job. Therefore, to ease the whole process, the user might request CHARMIE to follow him/her to the new room. This chore is particularly useful in large indoor environments where following a user is significantly more efficient than looking for

the worker. In the following process, the robot can avoid static and dynamic obstacles or even calculate a new trajectory when it realises it cannot pass.

In Figure 21, extracted from the video in Appendix A, all of the steps regarding the “Follow Me” task can be seen. The user starts by asking the robot to follow him/her to navigate to the compartment where help is needed, as can be seen in Figure 21a. The robot uses both the 2D LiDAR and the RGB-D camera to lock the user it is following, as can be seen in Figure 21b. During the process, the robot can navigate narrow spaces and even avoid collisions with humans that pass between the followed user and CHARMIE, as can be seen in Figure 21c,d. Almost at the end of the route, a new obstacle comes up, the robot detects that the followed user has an insuperable barrier similar to a wall (represented by the black cardboard demonstrated in the video in Appendix A that the robot cannot surpass, as can be seen in Figure 21e. Thus the robot, using the environment’s map, calculates an alternative route to get back to its user, as can be seen in Figure 21f. In the return path, CHARMIE finds novel obstacles that it can successfully overcome.



Figure 21. A step-by-step demonstration of CHARMIE’s “Follow Me” task. From receiving the task to navigating behind the human user while avoiding static and dynamic obstacles that cross the path between the robot and the user. When the robot cannot continue following the user, it recalculates a new trajectory.

As an example of a collaborative duty, the robot can lay the table with a human user. The methodology used is similar to the storing/cleaning tasks but with the collaboration twist. When this task is selected, CHARMIE knows that it needs five items to lay on the table: a plate, a fork, a spoon, a knife and a cup. The robot analyses the objects already laid on the table by a worker or a human user. Furthermore, it complements the user's work by signalling which items it will distribute while analysing whether the human user forgot one item on a previously laid set.

4. Conclusions

A description of hardware and software solutions for healthcare and domestic collaborative service and assistant robot, CHARMIE, is presented in this article. Additionally, results from the development of the initial prototype and the first set of user trials in a controlled laboratory setting focusing on developing an assistive care robot for older adults are described. The focus of the chores presented displays several different scenarios where CHARMIE can directly impact the quality of life of older people, mainly regarding physical safety, but also concerning social interactions and mental health. The majority of the robot's tasks designed for geriatric care involve fall detection and prevention, such as patrolling through nursing or domestic homes and picking up objects from the floor. In addition to the tasks demonstrated, CHARMIE can also work as a social company robot, asking questions throughout the day, allowing the user to play some games in its multimodal user interface and reminding the users of their schedules. Even though most tasks mainly aid the elderly, these chores can be adapted to be more oriented to healthcare workers or people with reduced mobility.

From a different perspective, due to the difficulties brought by the COVID-19 pandemic and its associated lockdowns, robots such as CHARMIE provide a safer solution to overcome this disease's challenges. Robots provide a superlative solution, since the virus cannot replicate itself in a robot as it does in a human and drastically reduces person-to-person contacts. Elderly care facilities and all other healthcare-related environments are particularly at risk of heavy breakouts since these encompass a very vulnerable population. Without a robotic solution, residents without the virus inside such facilities face a higher chance of contamination, which may happen through asymptomatic healthcare professionals. In addition to all of the social and mental health tasks social robots can perform, service and assistant robots such as CHARMIE can provide more direct help not only to patients, but also to healthcare professionals in a wide range of healthcare facilities. The longevity of the virus has dictated that all healthcare workers undergo very long working shifts, with low sleeping schedules, and in many situations being away from their families. This can lead to healthcare workers reaching a state of high fatigue and burnout in overburdened health systems, which can be eased if some of the tasks are performed by healthcare robots such as CHARMIE. Some examples of tasks are: transporting goods, providing patients information, patrolling the facilities, and sending an alert when any unexpected patient behaviour is detected. Additionally, due to the COVID-19 pandemic, the desired tests meant to be performed on real world environments, in this case in two nursing homes, two domestic homes and one hospital, initially scheduled for 2020 had to be indefinitely postponed as mandated by the national public health committee.

As short-term and middle-term objectives, novel concepts and tasks are projected to be developed and implemented in CHARMIE. The robot can successfully detect falls and prevent some scenarios where older people might fall by picking up items from the floor. However, one of the most significant difficulties regarding movement and mobility of older people happens when trying to get up from a sitting position, which is more aggravated in single-person households. This scenario happens several times during the day: getting out of bed, after a meal, when getting up from the sofa where the elderly do not have enough strength or apply too much force and lose balance, which results in dangerous falls. The goal is to provide active help to older people when trying to stand up. Another goal is to increase the robot's working area in buildings with more than one floor. Since

these healthcare facilities are commonly prepared for people with reduced mobility or wheelchairs, it is uncommon to have rooms that can only be accessed through stairs. Since the motion platform cannot overcome stairs, the goal is to create a system that allows the robot to successfully move between floors using elevators. The map would consist of all the floors in the building, and the robot must move to the elevator to switch floors. However, this task encompasses many steps that are still being worked on, such as calling the elevator, pressing the correct buttons and entering and leaving without colliding with other users. In order to benchmark all of these different tasks, it is intended to apply for RoboCup@Home participation. The CHARMIE project had already done so in 2017, as the video in Appendix A is the qualification video, part of the necessary qualification material. The video demonstrates CHARMIE performing some of the tasks previously described.

The desirable long-term goals for the CHARMIE project can be divided into two categories. From a technological perspective, it is intended to create the necessary hardware and software solutions to transport broader wheeled objects such as hospital beds and wheelchairs that may have patients. It is extremely challenging to move these wheeled platforms with human patients on top. The robot must adapt its omnidirectional way of motion to the wheeled object, detect obstacles further away and analyse the patient pose. All this must be performed while considering all the strong safety measures that come with patient transportation. Additionally, as previously stated, it is intended to use machine learning algorithms for a wide range of tasks allowing the robot to learn how to perform and improve chores via observation and trial-and-error direct interaction with the environment. However, these household and healthcare tasks usually consist of long-term planning, high-dimensional continuous action-space, simulation to real-world transition problem and, in most cases, incomplete information. Such issues require highly complex reinforcement learning algorithms to be solved. This methodology enables adaptive learning, using reinforcement learning based service and assistive elderly care chores like those previously described. From a global perspective, it is planned to start thoroughly testing CHARMIE in real healthcare and domestic environments. All functions, such as map building, self-localisation, obstacle detection and avoidance, human-robot interaction, object detection and manipulation, will be executed in a hospital, two nursing homes, and two domestic houses. This will allow a qualitative and quantitative evaluation from the developers and users of how the robot can perform in such environments. The final results will comprise a fitting framework for a socially assistive robot for end-to-end chores whose final goal is to enhance all its users' quality of life.

Author Contributions: Conceptualization, I.S.G., G.L. and A.F.R.; Methodology, T.R. and F.G.; Software, T.R., G.L. and A.F.R.; Validation, I.S.G., G.L. and A.F.R.; Formal Analysis, T.R. and F.G.; Investigation, T.R., F.G. and I.S.G.; Resources, T.R., F.G. and I.S.G.; Data Curation, F.G. and I.S.G.; Writing—Original Draft Preparation, T.R. and I.S.G.; Writing—Review and Editing, F.G., G.L. and A.F.R.; Visualization, I.S.G.; Supervision, G.L. and A.F.R.; Project Administration, G.L. and A.F.R.; Funding Acquisition, T.R., F.G., G.L. and A.F.R. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by FCT—Fundação para a Ciência e Tecnologia within the R&D Units Project Scope: UIDB/00319/2020. The author T.R. received funding through a doctoral scholarship from the Portuguese Foundation for Science and Technology (Fundação para a Ciência e a Tecnologia) [grant number SFRH/BD/06944/2020], with funds from the Portuguese Ministry of Science, Technology and Higher Education and the European Social Fund through the Programa Operacional do Capital Humano (POCH). The author F.G. received funding through a doctoral scholarship from the Portuguese Foundation for Science and Technology (Fundação para a Ciência e a Tecnologia) [grant number SFRH/BD/145993/2019], with funds from the Portuguese Ministry of Science, Technology and Higher Education and the European Social Fund through the Programa Operacional do Capital Humano (POCH).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Acknowledgments: The authors of this work would like to thank the members of the Laboratory of Automation and Robotics from the University of Minho as well as all former CHARMIE project members.

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A. Video Link

Link to the MinhoTeam qualification video for RoboCup@Home 2017 in Nagoya, Japan. It displays some of the tasks described in this article and all of CHARMIE's functionalities: <https://youtu.be/n0ZariVsIj0> (accessed on the 2 April 2021).

References

1. United Nations Department of Economic and Social Affairs. *World Population Prospects 2019*; United Nations: New York, NY, USA, 2019; ISBN 9789211483161.
2. Eurostat, Population Structure and Ageing—Statistics Explained. 2016. Available online: https://ec.europa.eu/eurostat/statistics-explained/index.php?title=Population_structure_and_ageing (accessed on 1 August 2021).
3. European Commission. *The 2018 Ageing Report: Underlying Assumptions & Projection*; European Commission: Brussels, Belgium, 2018; ISBN 9789279353512.
4. Cristea, M.; Noja, G.G.; Stefea, P.; Sala, A.L. The Impact of Population Aging and Public Health Support on EU Labor Markets. *Int. J. Environ. Res. Public Health* **2020**, *17*, 1439. [[CrossRef](#)] [[PubMed](#)]
5. WHO. *Global Strategy and Action Plan on Ageing and Health*; WHO: Geneva, Switzerland, 2017; ISBN 9789241513500.
6. Garmann-Johnsen, N.F.; Mettler, T.; Sprenger, M. Service Robotics in Healthcare: A Perspective for Information Systems Researchers? In Proceedings of the 35th International Conference on Information Systems (ICIS 2014), Auckland, New Zealand, 14 December 2014. [[CrossRef](#)]
7. Kim, J.; Gu, G.M.; Heo, P. Robotics for healthcare. In *Biomedical Engineering: Frontier Research and Converging Technologies*; Springer International Publishing: Berlin, Germany, 2015; ISBN 9783319218137.
8. Doelling, K.; Shin, J.; Popa, D.O. Service Robotics for the Home: A State of the Art Review. In Proceedings of the ACM International Conference Proceeding Series, Rhodes, Greece, 27 May 2014.
9. Martinez-Martin, E.; Del Pobil, A.P. Personal robot assistants for elderly care: An overview. In *Intelligent Systems Reference Library*; Springer: Berlin, Germany, 2018.
10. Yang, G.Z.; Nelson, B.J.; Murphy, R.R.; Choset, H.; Christensen, H.; Collins, S.H.; Dario, P.; Goldberg, K.; Ikuta, K.; Jacobstein, N.; et al. Combating COVID-19-The Role of Robotics in Managing Public Health and Infectious Diseases. *Sci. Robot.* **2020**, *5*, eabb5589. [[CrossRef](#)] [[PubMed](#)]
11. Shen, Y.; Guo, D.; Long, F.; Mateos, L.A.; Ding, H.; Xiu, Z.; Hellman, R.B.; King, A.; Chen, S.; Zhang, C.; et al. Robots under COVID-19 Pandemic: A Comprehensive Survey. *IEEE Access* **2021**. [[CrossRef](#)]
12. Khan, Z.H.; Siddique, A.; Lee, C.W. Robotics Utilization for Healthcare Digitization in Global COVID-19 Management. *Int. J. Environ. Res. Public Health* **2020**, *17*, 3819. [[CrossRef](#)] [[PubMed](#)]
13. Holland, J.; Kingston, L.; McCarthy, C.; Armstrong, E.; O'Dwyer, P.; Merz, F.; McConnell, M. Service Robots in the Healthcare Sector. *Robotics* **2021**, *10*, 47. [[CrossRef](#)]
14. Fischinger, D.; Einramhof, P.; Papoutsakis, K.; Wohlkinger, W.; Mayer, P.; Panek, P.; Hofmann, S.; Koertner, T.; Weiss, A.; Argyros, A.; et al. Hobbit, a Care Robot Supporting Independent Living at Home: First Prototype and Lessons Learned. *Rob. Auton. Syst.* **2016**, *75*, 60–78. [[CrossRef](#)]
15. Coşar, S.; Fernandez-Carmona, M.; Agrigoroaie, R.; Pages, J.; Ferland, F.; Zhao, F.; Yue, S.; Bellotto, N.; Tapus, A. ENRICHME: Perception and Interaction of an Assistive Robot for the Elderly at Home. *Int. J. Soc. Robot.* **2020**, *12*, 779–805. [[CrossRef](#)]
16. Abubakar, S.; Das, S.K.; Robinson, C.; Saadatzi, M.N.; Cynthia Logsdon, M.; Mitchell, H.; Chlebowy, D.; Popa, D.O. ARNA, a Service Robot for Nursing Assistance: System Overview and User Acceptability. *IEEE Int. Conf. Autom. Sci. Eng.* **2020**, *2020*, 1408–1414. [[CrossRef](#)]
17. Stuede, M.; Westermann, K.; Schappler, M.; Spindeldreier, S. Sobi: An Interactive Social Service Robot for Long-Term Autonomy in Open Environments. *arXiv* **2021**, arXiv:2105.03242.
18. Alvito, P.; Marques, C.; Carriço, P.; Sequeira, J.; Gonçalves, D. Deliverable D2.2.1: MOnarCH Robots Hardware. 2014. Available online: http://users.isr.ist.utl.pt/~||jseq/MOnarCH/Deliverables/D2.2.1_update.pdf (accessed on 1 August 2021).
19. Holz, D.; Iocchi, L. Benchmarking Intelligent Service Robots through Scientific Competitions: The RoboCup @ Home Approach. In Proceedings of the AAAI Spring Symposium—Designing Intelligent Robots: Reintegrating AI II, Stanford University, Stanford, CA, USA, 25–27 March 2013.
20. Basiri, M.; Piazza, E.; Matteucci, M.; Lima, P. Benchmarking Functionalities of Domestic Service Robots through Scientific Competitions. *KI—Kunstl. Intelligenz* **2019**, *33*, 357–367. [[CrossRef](#)]

21. Iocchi, L.; Holz, D.; Ruiz-Del-Solar, J.; Sugiura, K.; Van Der Zant, T. RoboCup@Home: Analysis and Results of Evolving Competitions for Domestic and Service Robots. *Artif. Intell.* **2015**, *229*, 258–281. [\[CrossRef\]](#)
22. Matamoros, M.; Seib, V.; Paulus, D. Trends, Challenges and Adopted Strategies in RoboCup@Home. In Proceedings of the 19th IEEE International Conference on Autonomous Robot Systems and Competitions, Gondomar, Portugal, 24–26 April 2019.
23. Cousineau, J.; Le, H. *Walking Machine @ Home 2019 Team Description Paper*; RoboCup: Sydney, Australia, 2019; pp. 1–9.
24. Perez, B.F.V.; Meneghetti, D.R.; Matiuci, E.; Neves, L.C.; Pimentel, F.; Melo, G.S.; Santos, J.V.M.; Gazignato, L.I.; Gonbata, M.Y.; Carvalho, M.G.; et al. *RoboFEI @ Home Team Description Paper for RoboCup @ Home 2019*; RoboCup: Sydney, Australia, 2019.
25. Albar, B.; Joffroy, L. *CATIE Robotics @ Home 2019 Team Description Paper*; RoboCup: Sydney, Australia, 2019.
26. Memmesheimer, R.; Mykhalchyshyna, I.; Wettengel, N.Y.; Evers, T.; Buchhold, L.; Schmidt, P.; Schmidt, N.; Germann, I.; Mints, M.; Rettler, G.; et al. *RoboCup 2019—Homer@UniKoblenz (Germany)*; RoboCup: Sydney, Australia, 2019; Volume 2014, pp. 1–9.
27. van der Burgh, M.F.B.; Lunenburg, J.J.M.; Appeldoorn, R.P.W.; van Beek, L.L.A.M.; Geijsberts, J.; Janssen, L.G.L.; van Dooren, P.; van Rooy, H.W.A.M.; Aggarwal, A.; Aleksandrov, S.; et al. *Tech United Eindhoven @Home 2020 Team Description Paper*; RoboCup: Dordeaux, France, 2020; pp. 1–9.
28. Wang, M.; Deng, W. Deep Face Recognition: A Survey. *Neurocomputing* **2021**. [\[CrossRef\]](#)
29. Memmesheimer, R.; Seib, V.; Paulus, D. Homer@UniKoblenz: Winning Team of the Robocup@home Open Platform League 2017. In *Proceedings of the Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *Robot World Cup XXI, Cham, Switzerland, 2018*; Springer International Publishing: Cham, Switzerland, 2018; pp. 509–520.
30. Weber, T.; Triputen, S.; Danner, M.; Braun, S.; Schreve, K.; Rättsch, M. Follow Me: Real-Time in the Wild Person Tracking Application for Autonomous Robotics. In *Proceedings of the Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *RoboCup 2017: Robot World Cup XXI*; Springer International Publishing: Cham, Switzerland, 2018; pp. 156–167.
31. Martínez, L.; Loncomilla, P.; Ruiz-Del-solar, J. Object Recognition for Manipulation Tasks in Real Domestic Settings: A Comparative Study. In *Proceedings of the Lecture Notes in Artificial Intelligence (Subseries of Lecture Notes in Computer Science)*, *RoboCup 2014: Robot World Cup XVIII*; Springer International Publishing: Cham, Switzerland, 2015; pp. 207–219.
32. Massouh, N.; Brigato, L.; Iocchi, L. RoboCup@Home-Objects: Benchmarking Object Recognition for Home Robots. In *Robot World Cup*; Springer International Publishing: Berlin, Germany, 2019; Volume 11531. [\[CrossRef\]](#)
33. Zlotowski, J.; Proudfoot, D.; Yogeewaran, K.; Bartneck, C. Anthropomorphism: Opportunities and Challenges in Human–Robot Interaction. *Int. J. Soc. Robot.* **2015**, *7*, 347–360. [\[CrossRef\]](#)
34. Gonçalves, F.; Ribeiro, T.; Garcia, I.; Ribeiro, F.A.; Monteiro, C.; Lopes, G. Development of an Anthropomorphic Mobile Manipulator with Human, Machine and Environment Interaction. *FME Trans.* **2019**, *47*, 790–801. [\[CrossRef\]](#)
35. Portugal, D.; Alvito, P.; Christodoulou, E.; Samaras, G.; Dias, J. A Study on the Deployment of a Service Robot in an Elderly Care Center. *Int. J. Soc. Robot.* **2019**, *11*, 317–341. [\[CrossRef\]](#)
36. Reddy, K.V.; Kodati, M.; Chatra, K.; Bandyopadhyay, S. A Comprehensive Kinematic Analysis of the Double Wishbone and MacPherson Strut Suspension Systems. *Mech. Mach. Theory* **2016**, *105*, 441–470. [\[CrossRef\]](#)
37. Van der Burgh, M.F.B.; Lunenburg, J.J.M.; Appeldoorn, R.P.W.; Wijnands, R.W.J.; Clephas, T.T.G.; Baeten, M.J.J.; Van Beek, L.L.A.M.; Ottervanger, R.A.; Van Rooy, H.W.A.M.; van de Molengraft, M.J.G. Tech United Eindhoven @Home 2017 Team Description Paper. In *RoboCup@Home*; RoboCup@Home: Nagoya, Japan, 2017; Volume 10.
38. Lunenburg, J.J.M.; Coenen, S.A.M.; van den Dries, S.; Elfring, J.; Janssen, R.J.M.; Sandee, J.H.; van de Molengraft, M.J.G. Tech United Eindhoven Team Description 2013. In Proceedings of the 17th RoboCup International Symposium (May 2013), Eindhoven, The Netherlands, 24–30 June 2013; pp. 1–8.
39. Garcia, I.; Gonçalves, F.; Ribeiro, T.; Fernandes, P.; Rocha, C.; Boucinha, R.; Lopes, G.; Ribeiro, A.F. Autonomous 4DOF Robotic Manipulator Prototype for Industrial Environment and Human Cooperation. In Proceedings of the 19th IEEE International Conference on Autonomous Robot Systems and Competitions, Gondomar, Portugal, 24–26 April 2019.
40. Paik, J.K.; Shin, B.H.; Bang, Y.B.; Shim, Y.B. Development of an Anthropomorphic Robotic Arm and Hand for Interactive Humanoids. *J. Bionic Eng.* **2012**, *9*, 133–142. [\[CrossRef\]](#)
41. Langevin, G. InMoov—Open Source 3D Printed Life-Size Robot. Available online: <https://inmoov.fr/> (accessed on 1 August 2021).
42. Llop-Harillo, I.; Pérez-González, A.; Andrés-Esperanza, J. Grasping Ability and Motion Synergies in Affordable Tendon-Driven Prosthetic Hands Controlled by Able-Bodied Subjects. *Front. Neurobot.* **2020**, *14*, 57. [\[CrossRef\]](#) [\[PubMed\]](#)
43. Zhou, J.; Yi, J.; Chen, X.; Liu, Z.; Wang, Z. BCL-13: A 13-DOF Soft Robotic Hand for Dexterous Grasping and in-Hand Manipulation. *IEEE Robot. Autom. Lett.* **2018**, *3*, 3379–3386. [\[CrossRef\]](#)
44. Lopes, A.G.T.; Ribeiro, A.F.M.; Pereira, D.C.; Neves, F.M.A.; Garcia, I.S.M.; Ribeiro, J.C.L.; Ferreira, J.F.R.; Fernandes, P.N.L.; Ribeiro, T.A. *CHARMIE, Minho Team @ Home 2017 Team Description Paper*; RoboCup@Home: Nagoya, Japan, 2017.
45. Rosten, E.; Porter, R.; Drummond, T. Faster and Better: A Machine Learning Approach to Corner Detection. *IEEE Trans. Pattern Anal. Mach. Intell.* **2010**, *32*, 105–119. [\[CrossRef\]](#)
46. Hornung, A.; Wurm, K.M.; Bennewitz, M.; Stachniss, C.; Burgard, W. OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees. *Auton. Robots* **2013**, *34*, 189–206. [\[CrossRef\]](#)

47. Xiaoyu, W.; Caihong, L.; Li, S.; Ning, Z.; Hao, F. On Adaptive Monte Carlo Localization Algorithm for the Mobile Robot Based on ROS. In Proceedings of the 37th Chinese Control Conference (CCC), Wuhan, China, 25–27 July 2018; pp. 5207–5212.
48. Monteiro, S.; Bicho, E. A Dynamical Systems Approach to Behavior-Based Formation Control. In Proceedings of the Proceedings—IEEE International Conference on Robotics and Automation, Washington, DC, USA, 30 May 2002.
49. Fernando, R.; Lopes, G.; Maia, T.; Ribeiro, H.; Silva, P.; Roriz, R.; Ferreira, N. Motion Control of Mobile Autonomous Robots Using Non-Linear Dynamical Systems Approach. In Proceedings of the CONTROLO, Guimarães, Portugal, 14–16 September 2016; pp. 409–421.
50. Schulman, J.; Levine, S.; Moritz, P.; Jordan, M.; Abbeel, P. Trust Region Policy Optimization. In Proceedings of the 32nd International Conference on Machine Learning, Lille, France, 7–9 July 2015.
51. Schulman, J.; Wolski, F.; Dhariwal, P.; Radford, A.; Klimov, O. Proximal Policy Optimization Algorithms. *arXiv* **2017**, arXiv:1707.06347.
52. Lillicrap, T.P.; Hunt, J.J.; Pritzel, A.; Heess, N.; Erez, T.; Tassa, Y.; Silver, D.; Wierstra, D. Continuous Control with Deep Reinforcement Learning. *arXiv* **2016**, arXiv:1509.02971.
53. Abdolmaleki, A.; Springenberg, J.T.; Tassa, Y.; Munos, R.; Heess, N.; Riedmiller, M. Maximum a Posteriori Policy Optimisation. In Proceedings of the 6th International Conference on Learning Representations, ICLR 2018—Conference Track Proceedings, Vancouver, BC, Canada, 30 April–3 May 2018.
54. Ribeiro, T.; Gonçalves, F.; Garcia, I.; Lopes, G.; Ribeiro, A.F. Q-Learning for Autonomous Mobile Robot Obstacle Avoidance. In Proceedings of the 2019 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Gondomar, Portugal, 24–26 April 2019. [[CrossRef](#)]
55. Ribeiro, T. *Deep Reinforcement Learning for Robot Navigation Systems*; Repositorium, University of Minho: Braga, Portugal, 2019.
56. Tai, L.; Paolo, G.; Liu, M. Virtual-to-Real Deep Reinforcement Learning: Continuous Control of Mobile Robots for Mapless Navigation. In Proceedings of the IEEE International Conference on Intelligent Robots and Systems, Vancouver, BC, Canada, 24–28 September 2017.
57. Zhang, K.; Zhang, Z.; Li, Z.; Qiao, Y. Joint Face Detection and Alignment Using Multitask Cascaded Convolutional Networks. *IEEE Signal Process. Lett.* **2016**, *23*, 1499–1503. [[CrossRef](#)]
58. Szegedy, C.; Ioffe, S.; Vanhoucke, V.; Alemi, A.A. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. In Proceedings of the 31st AAAI Conference on Artificial Intelligence, San Francisco, CA, USA, 7–12 February 2017.
59. Huang, G.B.; Mattar, M.; Berg, T.; Labeled, E.L.; Images, R.; Learned-miller, E. Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments. HAL Archives-Ouvertes. In *Workshop on Faces in 'Real-Life' Images: Detection, Alignment, and Recognition*. 2008. Available online: <https://hal.inria.fr/inria-00321923/> (accessed on 1 August 2021).
60. Li, B.Y.L.; Mian, A.S.; Liu, W.; Krishna, A. Using Kinect for Face Recognition under Varying Poses, Expressions, Illumination and Disguise. In Proceedings of the IEEE Workshop on Applications of Computer Vision, Clearwater Beach, FL, USA, 15–17 January 2013.
61. Schroff, F.; Kalenichenko, D.; Philbin, J. FaceNet: A Unified Embedding for Face Recognition and Clustering. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 12 June 2015.
62. Huang, C.C.; Nguyen, M.H. Robust 3D Skeleton Tracking Based on Openpose and a Probabilistic Tracking Framework. In Proceedings of the Conference Proceedings—IEEE International Conference on Systems, Man and Cybernetics, Bari, Italy, 6–9 October 2019.
63. Cao, Z.; Hidalgo, G.; Simon, T.; Wei, S.E.; Sheikh, Y. OpenPose: Realtime Multi-Person 2D Pose Estimation Using Part Affinity Fields. *IEEE Trans. Pattern Anal. Mach. Intell.* **2021**, *43*, 172–186. [[CrossRef](#)]
64. Ilg, E.; Mayer, N.; Saikia, T.; Keuper, M.; Dosovitskiy, A.; Brox, T. FlowNet 2.0: Evolution of Optical Flow Estimation with Deep Networks. In Proceedings of the Proceedings—30th IEEE Conference on Computer Vision and Pattern Recognition, Honolulu, HI, USA, 21–26 July 2017.
65. Hui, T.W.; Tang, X.; Loy, C.C. LiteFlowNet: A Lightweight Convolutional Neural Network for Optical Flow Estimation. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Salt Lake City, UT, USA, 23 June 2018.
66. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 30 June 2016.
67. Redmon, J.; Farhadi, A. YOLOv3: An Incremental Improvement. *arXiv* **2018**, arXiv:1804.02767.

Article

An Algorithm for Painting Large Objects Based on a Nine-Axis UR5 Robotic Manipulator

Jun Wang ^{1,*}, Mingquan Yang ¹, Fei Liang ¹, Kangrui Feng ¹, Kai Zhang ¹ and Quan Wang ²

¹ Institute of Additive Manufacturing and Industrial Robotics, School of Mechanical Engineering, Hubei University of Technology, Wuhan 430068, China; mingquan_yang@outlook.com (M.Y.); 102010139@hbut.edu.cn (F.L.); 102010085@hbut.edu.cn (K.F.); 102000015@hbut.edu.cn (K.Z.)

² Institute of Marine Special Equipment and Technology, School of Mechanical Engineering, Hubei University of Technology, Wuhan 430068, China; quan_wang2003@163.com

* Correspondence: junwang@hbut.edu.cn

Featured Application: The proposed algorithm for painting large objects based on a nine-axis UR5 robotic manipulator can be applicable in many automobile repair shops where paint jobs can be performed. With the help of a nine-axis UR5 robotic manipulator with the proposed algorithm, vehicles can be automatically painted with the least amount of human manual labor. Simultaneously, the quality and efficiency of the paint jobs can be drastically improved, since the UR5 robot maintains its consistency, accuracy, and proficiency while conducting paint jobs.

Abstract: An algorithm for automatically planning trajectories designed for painting large objects is proposed in this paper to eliminate the difficulty of painting large objects and ensure their surface quality. The algorithm was divided into three phases, comprising the target point acquisition phase, the trajectory planning phase, and the UR5 robot inverse solution acquisition phase. In the target point acquisition phase, the standard triangle language (STL) file, algorithm of principal component analyses (PCA), and k-dimensional tree (k-d tree) were employed to obtain the point cloud model of the car roof to be painted. Simultaneously, the point cloud data were compressed as per the requirements of the painting process. In the trajectory planning phase, combined with the maximum operating space of the UR5 robot, the painting trajectory of the target points was converted into multiple traveling salesman problem (TSP) models, and each TSP model was created with a genetic algorithm (GA). In the last phase, in conformity with the singularities of the UR5 robot's motion space, the painting trajectory was divided into a recommended area trajectory and a non-recommended area trajectory and created by the analytical method and sequential quadratic programming (SQP). Finally, the proposed algorithm for painting large objects was deployed in a simulation experiment. Simulation results showed that the accuracy of the algorithm could meet the requirements of painting technology, and it has promising engineering practicability.

Keywords: genetic algorithm; principal component analyses; standard triangle language; traveling salesman problem; trajectory planning

Citation: Wang, J.; Yang, M.; Liang, F.; Feng, K.; Zhang, K.; Wang, Q. An Algorithm for Painting Large Objects Based on a Nine-Axis UR5 Robotic Manipulator. *Appl. Sci.* **2022**, *12*, 7219. <https://doi.org/10.3390/app12147219>

Academic Editors: Pedro Neto, António Paulo Moreira and Félix Vidal

Received: 7 June 2022

Accepted: 14 July 2022

Published: 18 July 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In recent years, UR5 robots have been widely popularized in industrial production fields such as painting, assembly, and micromanipulation [1]. In the above-mentioned fields, the painting process is the integral manufacturing procedure for automatically coating large objects, and it is one of the essential technologies for improving the surface quality of painted objects, which can then offer better performances under different working conditions [2–4]. Painting feasibility and trajectory planning are critical in the painting field.

To paint large objects, some engineers tried to mount UR5 robots on a mobile platform to enlarge the workspace of the UR5 robots [5,6], while other engineers tried to widen

the workspace by changing the structure of the UR5 robots [7]. However, the method of changing the structure of the UR5 robots entails enormous cost, and the UR5 robot with changed structure offers low adaptability. Thus, this paper proposes a UR5 robot which is mounted on an *xyz* three-axis motion platform to achieve the painting of large objects.

A well-designed trajectory can improve painting efficiency. Bureerat, S. et al. employed the MRPEIL-DE algorithm to optimize the trajectory of a six degree-of-freedom (DOF) UR5 robot [8]. Yin, S. et al. utilized mechanical learning methods to devise an energy-saving trajectory for industrial UR5 robots [9]. Serralheiro, W. et al. proposed employing a non-based trajectory planning method for time-energy optimization of a completely wheeled mobile UR5 robot [10]. Kazim, I.J. et al. compared improving the artificial potential field (APF) by the traditional particle swarm optimization (PSO) algorithm and the serendipity-based PSO (SBPSO) algorithm to control the path of a universal robot UR5 with collision avoidance [11]. Kazim, I.J. et al. utilized differential evolution (DE) optimization with the MATLAB toolbox, which has an applied robot operating system (ROS), to quantify the contour tracking performance of a collaborative universal manipulator robot (UR5) [12]. Al-Shanoon, A. et al. proposed a novel reliable framework for deep ConvNet combined with visual servoing using a single RGB camera [13]. Balanji, H.M. et al. proposed a novel calibration framework based on a single camera and computer vision techniques using ArUco markers [14]. Vivas, A. et al. designed the implementation of the control of a real UR5 robot from Matlab/Simulink using ROS [15]. Araki, R. et al. proposed a 6D pose estimation method for an object from a single RGB image for a robotic grasping task [16].

Nonetheless, most of the above-mentioned algorithms neither consider the singularities of the kinematics of the UR5 robot’s joints nor meet the requirements of the painting process. Therefore, this paper proposes a painting algorithm which can generate a painting trajectory satisfying the painting of large objects. The painting algorithm employs the standard triangle language (STL) file, algorithm of principal component analyses (PCA), and k-dimensional tree (k-d tree) to create a digital model of the object to be painted. The digital model is converted into multiple traveling salesman problem (TSP) models, and each TSP model is created with a genetic algorithm (GA). The painting trajectory offers high precision and efficiency.

2. Construction of a Point Cloud Model of the Target Object

STL files are popular in computer graphics application systems, and their file formats are simple and widely harnessed in machine vision and 3D reconstruction [17,18].

2.1. STL File Parsing

A STL file consists of multiple triangles and can be divided into the American Standard Code for Information Interchange (ASCII) format and binary format as per the storage format. The STL file obtained in this paper was suitable for ASCII. The analysis of a STL file in the ASCII format is shown in Figure 1.

```

Solid<name>
Facet normal:  $n_i, n_j, n_k$ 
Outer loop
Vertex  $v_{1x}, v_{1y}, v_{3z}$ 
Vertex  $v_{1x}, v_{1y}, v_{3z}$ 
Vertex  $v_{1x}, v_{1y}, v_{3z}$ 
End loop
End facet
.....
End solid<name>
    
```

Figure 1. Parsing of a STL file in ASCII format.

In Figure 1, η_i , η_j , and η_k represent the x , y , and z components of the triangle patch normal vector, respectively; v_{ix} , v_{iy} , and v_{iz} represent the x , y , and z coordinates of the i -th triangular patch, respectively.

2.2. Filling Triangles

The STL file only contains the coordinates of the vertices of the triangles, and the coordinates of all points in the model cannot be obtained. To obtain the full 3-dimensional information of the model, the triangles must be filled. This paper utilized the depth-first search (DFS) to fill the triangles. The triangle is represented by S . The center of the triangle is represented by o . Lengths of the 3 sides of the triangle is represented by l_1 , l_2 , and l_3 . The triangle S is later separated into 3 smaller triangles S_1 , S_2 , and S_3 . The algorithm model and the main flowchart of the filling process are shown in Figures 2 and 3.

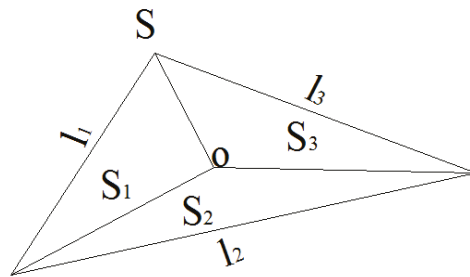


Figure 2. The model of the algorithm that fills triangles inside an STL file.

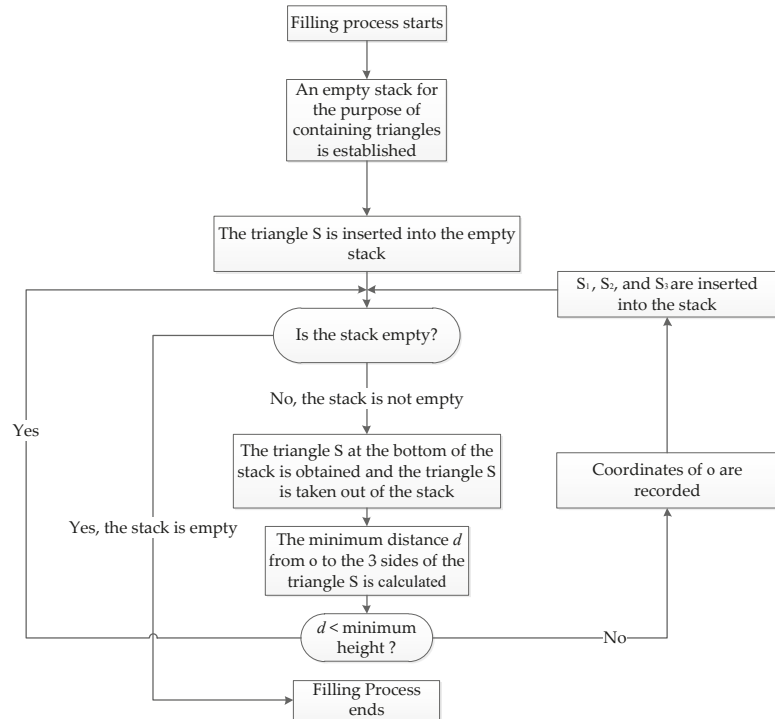


Figure 3. The filling process of the DFS algorithm.

The stopping condition of DFS recursion is that the minimum height corresponding to the three sides of the triangle is less than the specified minimum height. This requirement is shown in Equation (1), where the minimum triangle height h_{\min} is given.

$$S > \max(l_1, l_2, l_3) \cdot h_{\min} / 2 \quad (1)$$

The area of the triangle in Equation (1) is difficult to directly calculate. Therefore, Heron's formula, which is shown in Equation (2), is introduced, and the area is calculated from the lengths of the sides. The recursive stop condition is shown in Equation (3).

$$S = \sqrt{\varepsilon(\varepsilon - l_1)(\varepsilon - l_2)(\varepsilon - l_3)} \quad (2)$$

$$\varepsilon = (l_1 + l_2 + l_3) / 2$$

$$S \geq 2 \cdot \max(l_1, l_2, l_3) \cdot h_{\min} \quad (3)$$

The initial vertices are randomly selected as (10, 10, 0), (520, 100, 0), and (260, 410, 0), and h_{\min} equals 0.05. The filling results are shown in Figure 4a.

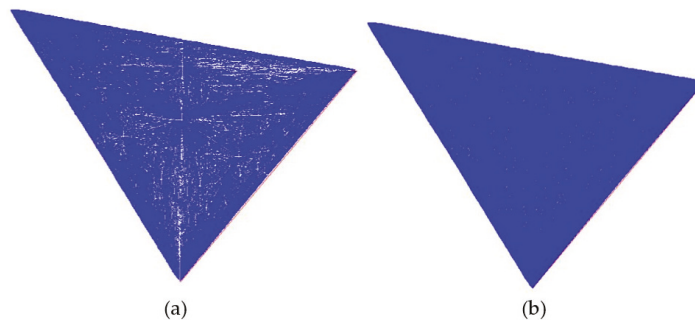


Figure 4. (a) The triangle filling results; (b) the improved triangle filling results.

However, there are a large number of unfilled lines in Figure 4a, and the filling effect is not satisfactory. After the filling process had been analyzed, it was found that the point filling method was harnessed in the filling, which could result in the points on the straight lines from the center of the triangle to the vertices of the triangle not being filled. If the number of filling points needs to be increased, the coordinates of the recorded point during the filling process can be changed into the coordinates of each point above the line connecting the recorded point and the vertex of the triangle with the same initial value condition and iteration termination condition. The final filling result is shown in Figure 4b. Compared to Figure 4a, Figure 4b is better filled, but the time taken to fill Figure 4b was much longer than to fill Figure 4a. If the accuracy is not high or the time is short, point filling is recommended. For high-precision conditions, linear filling is recommended.

2.3. Determining the Normal Vector of a Target Point on the Painting Trajectory

In order to ensure that the painting area is uniform during the painting process, it is necessary to ensure that the axis of the end of the spray gun coincides with the normal vector of the target point on the painting trajectory. Thus, the normal vectors of the target model must be obtained. As is shown in Figure 3, the target points on the painting trajectory can be divided into two types, which include the trajectory points obtained through the filling algorithm and the vertices of the original triangle.

The points obtained by filling are still coplanar with the original triangles. The normal vectors of the points obtained by filling can be determined with the normal vector of the plane. The normal vectors of the triangles can be directly extracted from the STL file. However, for the normal vectors of the vertices of the original triangle, there is a common vertex of multiple triangles. Ergo, the normal phase of the plane cannot be employed

instead of the normal phase of the points. In this paper, local plane fitting and principle component analysis (PCA) estimation methods were harnessed to create the normal vectors of the target points [19–21]. In order to acquire the target point field conveniently, the k-d tree algorithm was harnessed to store the coordinates of the target points [22]. The step-by-step algorithm is shown in Figures 5 and 6.

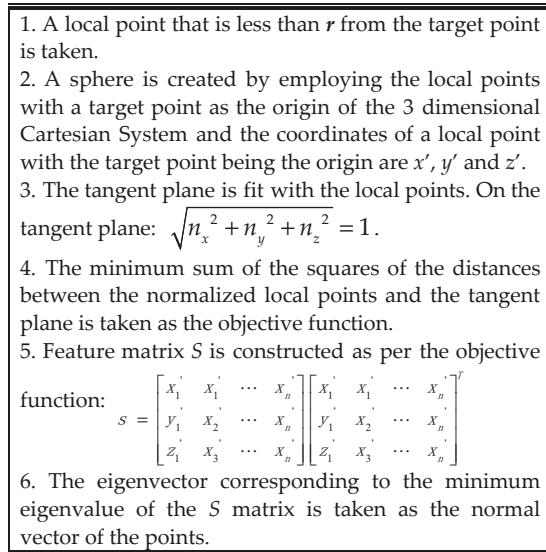


Figure 5. The PCA method to estimate the target point normal vector.

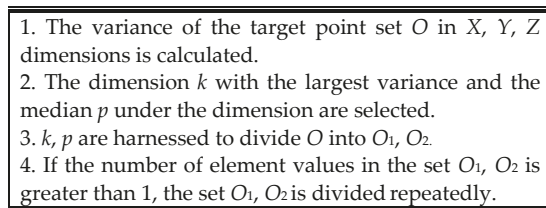


Figure 6. The k-d tree stores cloud data.

In the actual processing procedure, the normal vector of the target model should take its external normal phase, and the external normal vector of the model can be determined by Equation (4), where $\bar{\mathbf{p}}$ is the center of all target points, and $\|x\|_2$ is the second norm of the vector x .

$$\mathbf{n} = \begin{cases} \mathbf{n} \|(\mathbf{p} + \mathbf{n}) - \bar{\mathbf{p}}\|_2 \geq \|(\mathbf{p} - \mathbf{n}) - \bar{\mathbf{p}}\|_2 \\ -\mathbf{n} \|(\mathbf{p} + \mathbf{n}) - \bar{\mathbf{p}}\|_2 < \|(\mathbf{p} - \mathbf{n}) - \bar{\mathbf{p}}\|_2 \end{cases} \quad (4)$$

2.4. Determining the Pose of a Target Point on the Trajectory

In the actual painting process, the pose matrix is often deployed to describe the rotation of the target point in space. During the inverse kinematics calculation of the UR5 robot, the pose of the target point needs to be described by the matrix, and Rodrigues' rotation formula can determine the pose of the target point by the axis direction of the target point.

The diagram of Rodrigues' rotation formula is shown in Figure 7, where X_1, Y_1 , and Z_1 represent the three axes, with O being the origin of the 3-dimensional Cartesian system after rotating.

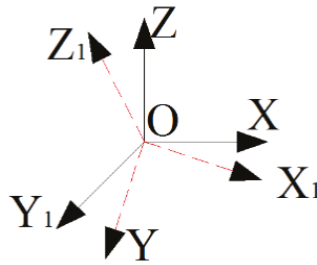


Figure 7. The schematic diagram of Rodrigues' rotation formula.

The rotation matrix R is shown in Equation (5),

$$E \cos \theta + (1 - \cos \theta)r^T \times r + \sin \theta \times \begin{bmatrix} 0 & -r_z & r_y \\ r_z & 0 & -r_x \\ -r_y & r_x & 0 \end{bmatrix} \quad (5)$$

where E represents the third-order identity matrix, and θ represents the angle created by vectors \mathbf{n} and \mathbf{n}_0 . In Equation (5), \mathbf{r} is the vector product of vectors \mathbf{n} and \mathbf{n}_0 ; $\mathbf{n}_0 = [0 \ 0 \ 1]^T$, and \mathbf{n} is the normal vector of the point.

2.5. Compression of the Data of Target Points

The design of the painting process parameters ensures that the end of the spray gun will produce a circle with a radius of r at a specified distance. During the painting process, it is necessary to ensure that the trajectory coverage during painting is 0.4. The painting model is shown in Figure 8, and the coverage of Figure 8, denoted by p , is shown in Equation (6).

$$p = \frac{4r^2 \cdot \arccos(d/2r) - d\sqrt{r^2 - d^2/4}}{2\pi r^2} \quad (6)$$

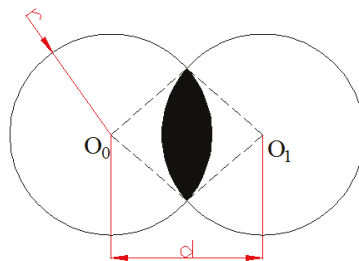


Figure 8. The painting model.

When the painting trajectory is executed, not all target points are to be painted. Only the path of key points should be painted. The distance between key points should meet the requirements of painting uniformity. In Figure 8 and Equation (6) d is employed to reduce the time and space complexity of the trajectory planning. The compression formula for the data points is shown in Equation (7).

$$\begin{aligned} k &= \text{floor}\left(d / (\sqrt{3})\right) \\ P_x &= \text{round}(P_x/k) \cdot k \\ P_y &= \text{round}(P_y/k) \cdot k \\ p_z &= \text{round}(p_z/k) \cdot k \end{aligned} \quad (7)$$

where P_x , P_y , and P_z represent the x - y - z coordinates of the target points in the point cloud, $floor$ represents the floor function, which takes as input a real number q , and gives out as output the greatest integer less than or equal to q , and $round$ represents the round function, which rounds off a numeric value to its nearest integer.

3. Nine-Axis UR5 Robot Forward Kinematics Model

The space that the UR5 robot covers is limited, and large-size objects may not be painted at once. When an area that needs to be painted is outside the working space of the UR5 robot, the D-H parameters of the UR5 robot must be changed, or auxiliary equipment must be added to help with the painting. The former is costly. Changing joints is not conducive to the popularization of UR5 robots. The proposed 6-DOF UR5 robot is installed on a 3-axis motion platform that moves horizontally, laterally, and vertically to assist painting. The 3D drawing of the UR5 robot is shown in Figure 9. The 9-axis UR5 robotic manipulator includes the 3-axis motion platform that has 3 prismatic pairs and the UR5 robot that has 6 revolute pairs. The 6-DOF UR5 robot selected in this paper is a UR5 robot, which is shown in Figure 10.



Figure 9. 3D drawings of the UR5 robot.



Figure 10. The UR5 robot.

The pose of the actual UR5 robot end **T** is shown in Equation (8).

$$\mathbf{T} = \mathbf{Trans}(t_x, t_y, t_z)\mathbf{Rot}(x, \pi)\mathbf{T}_{\text{robot}}\mathbf{T}_{\text{tool}}\mathbf{T}_{\text{dis}}$$

$$\mathbf{Trans}(t_x, t_y, t_z) = \begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{Rot}(x, \pi) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \pi & \sin \pi & 0 \\ 0 & -\sin \pi & \cos \pi & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$
(8)

in which t_x , t_y , and t_z represent the movement of the motion platform along the x , y , and z axes, respectively, and **Trans** represents the movement operator. **Rot** represents the rotation matrix of the x -axis. \mathbf{T}_{tool} , which is shown in Figure 11, represents the pose of the end-effector relative to its installation center. \mathbf{T}_{dis} , which is also shown in Figure 11, represents the end fixture of the UR5 robot's position. \mathbf{T}_{tool} is determined by the structure of the end-effector, and \mathbf{T}_{dis} is determined by the parameters of the end fixture of the UR5 robot's position. The UR5 robot system has 9 axes which are described in Figure 11. Axes 1 to 6 are the 6 joints of the UR5 robot. Axes 7 to 9 are the three-dimensional Cartesian coordinate system of the motion platform.

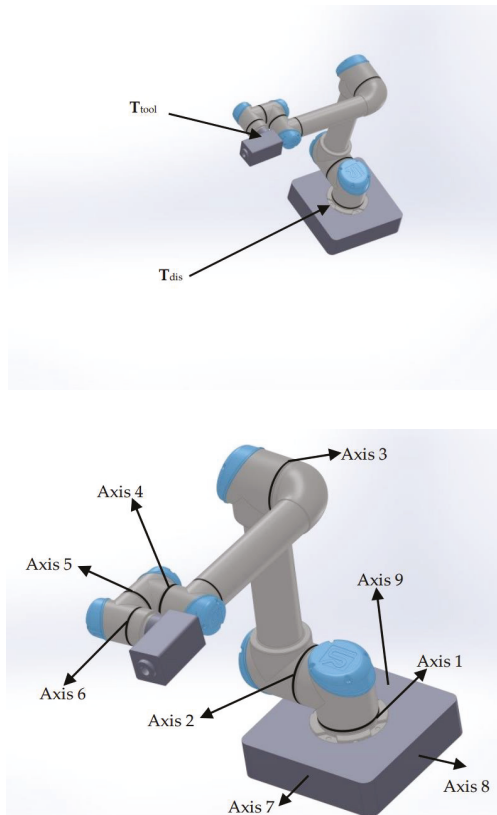


Figure 11. \mathbf{T}_{tool} , \mathbf{T}_{dis} , and the 9 axes of the UR5 robot system.

Figure 12a is the structural diagram of the 6-DOF UR5 robot composed of 6 revolute pairs, represented by 1, 2, 3, 4, 5, and 6 and 0 represents the end fixture of the UR5 robot. In order to analyze the pose change of the UR5 robot's end fixture coordinates relative to the end-effector, the D-H model is harnessed to establish the kinematics forward solution model of the UR5 robot.

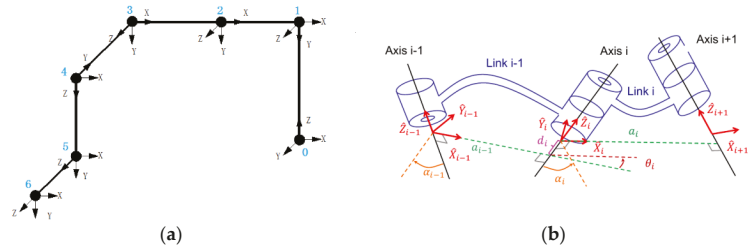


Figure 12. (a) The schematic structure of the UR5 robot (b) detailed explanation of θ_i , d_i , a_i , and α_i .

The UR5 robot's pose change matrix in the D-H model is shown in Equation (9).

$${}^i{}_{i-1}\mathbf{T} = \mathbf{Rot}(z, \theta_i) \mathbf{Trans}(0, 0, d_i) \mathbf{Trans}(a_i, 0, 0) \mathbf{Rot}(x, \alpha_i)$$

$$\mathbf{Rot}(z, \theta_i) = \begin{bmatrix} \cos \theta_i & \sin \theta_i & 0 & 0 \\ -\sin \theta_i & \cos \theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{9}$$

In Equation (9), θ_i is the angle for which x_{i-1} spins around z_i to become x_i ; d_i is the distance between x_{i-1} and x_i along z_i ; a_i is the distance between z_i and z_{i+1} along x_i , and α_i is the angle for which z_i spins around x_i to become z_{i+1} . $\mathbf{Rot}(z, \theta_i)$ represents the rotation matrix of the z -axis; θ_i , d_i , α_i , and a_i are shown in Figure 12b. ${}^i{}_{i-1}\mathbf{T}$ is shown in Equation (10), where s represents the sine function and c represents the cosine function.

$${}^i{}_{i-1}\mathbf{T} = \begin{bmatrix} c\theta_i & -s\theta_i c\alpha_i & s\theta_i s\alpha & a_i c\theta_i \\ s\theta_i & c\theta_i c\alpha_i & -c\theta_i s\alpha & a_i s\theta_i \\ 0 & s\alpha_i & c\alpha_i & d_i \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{10}$$

$\mathbf{T}_{\text{robot}}$, which represents the pose of the UR5 robot, is obtained via Equation (11). The UR5 robot's pose should satisfy Equation (12), and the D-H parameters of the UR5 robot are shown in Table 1.

$$\mathbf{T}_{\text{robot}} = \prod_{i=1}^6 {}^i{}_{i-1}\mathbf{T} \tag{11}$$

$$\mathbf{Trans}(x, y, z)^{-1} \cdot \mathbf{Rot}(x, \pi)^{-1} \cdot \mathbf{T} \cdot \mathbf{T}_{\text{tool}}^{-1} = \prod_{i=1}^6 {}^i{}_{i-1}\mathbf{T} \tag{12}$$

Table 1. The D-H parameters of the UR5 robot.

i	θ_i	d_i/mm	α_i	a_i/mm
1	-2π to 2π	89.2	$\pi/2$	0
2	-2π to 2π	0	0	425
3	-2π to 2π	0	0	392.3
4	-2π to 2π	109.2	$\pi/2$	0
5	-2π to 2π	94.7	$-\pi/2$	0
6	-2π to 2π	82.3	0	0

4. The Inverse Solution Model of Kinematics of the 9-Axis UR5 Robot

Nevertheless, since the system that controls the UR5 robot and the system that controls the motion platform are separated, communication between the systems may cause problems, such as delays and errors. In order to improve painting accuracy, the motion platform cannot be moved frequently. During the painting process, each time the motion platform is moved, the UR5 robot paints an area. After the painting is completed, the motion platform is moved again. The recommended operating space of the UR5 robot is shown in Figure 13.

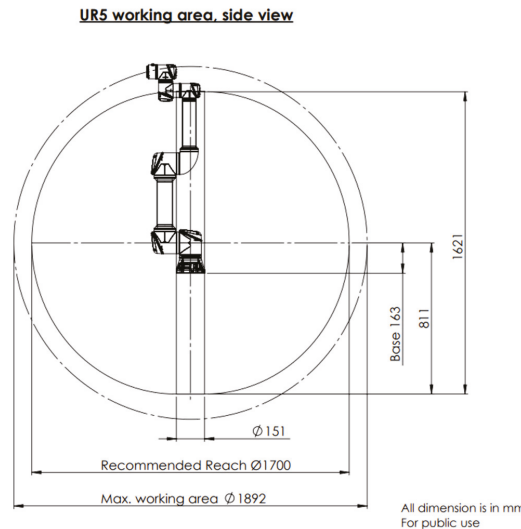


Figure 13. The working area of the UR5 robot.

The recommended working area in Figure 13 represents the feasible area recommended by the UR5 robot. The area between the maximum working area, which is represented by Max. working area in Figure 13, and the recommended feasible area represents the non-recommended feasible area that the UR5 robot may not reach. The non-recommended area represents the singularity of the UR5 robot. The non-recommended area takes the largest cylinder contained in the green area in Figure 13 as the largest area for each painting procedure.

However, the recommended area in Figure 13 is not a complete ball, and the cylindrical area with a diameter of 151 mm is not recommended. If this area is removed, the volume of the largest cylinder obtained is 0.25 times that without removal. Therefore, the platform will be moved slightly. In order to move the platform as little as possible, the green area is assumed to be a complete ball when the maximum cylinder is calculated.

In order to prevent the UR5 robot from colliding with the platform, it is necessary to keep the end of the UR5 robot below its installation center during the movement of the UR5 robot. The area under the side wall of the machine is taken as the effective area. The volume of the cylinder included in Figure 13 is shown in Equation (13).

$$V = \pi(rcos\theta)^2 \cdot (163 - rsin\theta) \tag{13}$$

where r , θ , and V represent the maximal length of the UR5 robot, the radian of the intersection of the cylinder, and the maximum machining space of the UR5 robot, respectively. When θ equals -0.54 rad, the volume of the cylinder is the largest. At this time, the cylinder has a radius of 727 mm and a height of 602 mm. Due to the existence of the non-recommended area, the UR5 robot inverse kinematics model is divided into a recommended area inverse kinematics model and a non-recommended area inverse kinematics model.

4.1. Inverse Solution Model for Recommended Regions

The three joint axes of the UR5 robot, joint axes 2, 3 and 4, which are shown in Figure 14, are parallel, and their spatial structure satisfies the Pieper criterion. The closed-form solution method can be harnessed to solve the angular sequence of the UR5 robot joint under the target pose constraints.

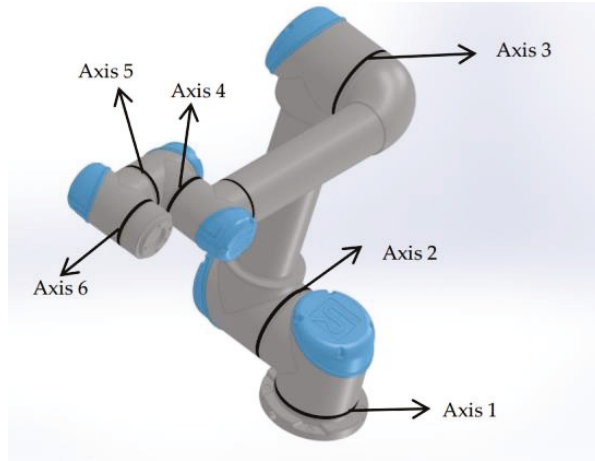


Figure 14. The axes of the UR5 robot.

Equation (14) determines the pose of the end-effector of the UR5 robot.

$$\prod_{i=1}^6 {}^{i-1}\mathbf{T} = (\mathbf{Trans}(t_x, t_y, t_z)\mathbf{Rot}(x, \pi))^{-1} \cdot \mathbf{T} \cdot (\mathbf{T}_{tool} \cdot \mathbf{T}_{dis})^{-1} = \begin{bmatrix} n_x & o_x & a_x & x' \\ n_y & o_y & a_y & y' \\ n_z & o_z & a_z & z' \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (14)$$

The main idea of the closed-form solution is creating the constraint equation established by matrix changes. The constraint matrix of the UR5 robot, which is shown in Equation (15), is established in accordance with Equation (11). The third row of the left and right sides of Equation (15) is expanded to establish Equation (16). Equation (16) can be regarded as the equation set of $\theta_1, \theta_5,$ and θ_6 .

The procedure for finding the values of $\theta_2, \theta_3,$ and θ_4 is similar.

$$\mathbf{L} = ({}^0\mathbf{T}^{-1})\mathbf{T}_{robot} = \prod_{i=2}^6 {}^{i-1}\mathbf{T} = \mathbf{R} \quad (15)$$

$$\begin{bmatrix} c\theta_6s\theta_5 & & & \\ s\theta_6s\theta_5 & & & \\ c\theta_5 & & & \\ d_2 + d_3 + d_4 + d_6c\theta_5 + a_5s\theta_5 + a_6c\theta_6s\theta_5 & & & \end{bmatrix}^T = \begin{bmatrix} n_y c\theta_1 + n_x s\theta_1 \\ o_y c\theta_1 + o_x s\theta_1 \\ a_y c\theta_1 + a_x s\theta_1 \\ y c\theta_1 + x s\theta_1 \end{bmatrix}^T \quad (16)$$

By employing Equations (15) and (17), the result can be obtained in Equation (18),

$$\mathbf{L} = ({}^0\mathbf{T}^{-1}){}^0\mathbf{T}({}^4\mathbf{T}^{-1})({}^5\mathbf{T}^{-1})({}^6\mathbf{T}^{-1}) = \prod_{i=2}^4 {}^{i-1}\mathbf{T} = \mathbf{R} \quad (17)$$

$$\mathbf{R} = \begin{bmatrix} c\theta_{234} & 0 & -s\theta_{234} & a_3 \cdot c\theta_{23} + a_3 \cdot c\theta_2 + a_4 \cdot c\theta_{234} \\ -s\theta_{234} & 0 & c\theta_{234} & -a_3 \cdot s\theta_{23} - a_3 \cdot c\theta_2 - a_4 \cdot c\theta_{234} \\ 0 & -1 & 0 & d_2 + d_3 + d_4 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{18}$$

where $\theta_{i_1 i_2 \dots i_n}$ is defined in Equation (19).

$$\theta_{i_1 i_2 \dots i_n} = \sum_{j=1}^n \theta_{ij} \tag{19}$$

In Equation (18), θ_{234} , θ_{23} , and θ_2 can be acquired by the fourth column of \mathbf{R} in Equation (17); θ_2 , θ_3 , and θ_4 can be obtained afterwards. In Equations (16) and (18), s represents the sine function and c represents the cosine function. Nonetheless, the solution involves a large number of inverse trigonometric functions, and the range of the angles within which the UR5 robot’s joints move is -2π to 2π , which results in multiple solutions. Therefore, the solution with the smallest Euler distance from the initial joint angle sequence is selected.

4.2. The Inverse Solution Model for Non-Recommended Regions

In a non-recommended area, the UR5 robot may not be able to achieve certain poses. It is necessary to move the motion platform to match the position of the UR5 robot. Then, the inverse solution model of the manipulator can be changed to obtain the minimum joint rotation radian and the number of movements of the manipulator’s motion platform under posture and pose constraints. This problem is a nonlinear optimization problem with constraints. Sequential quadratic programming (SQP) is an iterative method for constrained nonlinear optimization. When the input value is close to the real solution, the algorithm has a second-order convergence speed and can quickly solve the target solution [23,24].

4.2.1. The Objective Function and Constraints

In the process of solving the inverse kinematics, in order to avoid singularities, the platform is allowed to move slightly. The number of movements of the platform and the total arc of the UR5 robot joint give the objective function, which is shown in Equation (20).

$$f = \sqrt{\sum_{i=1}^{i=6} (\theta_i^r - \theta_i)^2 + K_d \left((x^r - t_x)^2 + (y^r - t_y)^2 + (z^r - t_z)^2 \right)} \tag{20}$$

where θ_i^r and θ_i are the actual joint curvature and the starting point arc of the UR5 robot. The target point K_d represents the number of movements of the platform; x^r , y^r , and z^r represent the actual moving distance.

The error es is allowed in the actual working situation. The constraint condition should be that the Euler distance between \mathbf{T} and the target pose \mathbf{T}' is less than es , which is shown in Equation (21).

$$\sqrt{\sum_{j=1}^{j=3} \left({}^0\mathbf{T}_{j4} - {}^0\mathbf{T}'_{j4} \right)^2} - es \leq 0 \tag{21}$$

During the painting process, an error es' between the end axis of the spray gun and the target point axis is allowed. The pose constraints are shown in Equation (22).

$$\left| \arccos \left(\frac{\mathbf{n} \cdot \mathbf{n}_0}{|\mathbf{n}| \cdot |\mathbf{n}_0|} \right) \right| < es' \tag{22}$$

where \mathbf{n} is the normal direction of \mathbf{T} to the z axis, and \mathbf{n}_0 is the normal direction of \mathbf{T}' to the z axis.

4.2.2. The Solution Process of the SQP Algorithm

The SQP algorithm decomposes the problem into the quadratic programming (QP) sub-problem, obtaining the descending direction d by solving the current angle sequence θ_k in the QP problem and updating θ_{k+1} via d until the Karush–Kuhn–Tucker (KKT) condition is satisfied or the maximum number of iterations is reached. The specific solving process of the algorithm is described below.

(1) The Lagrange function $L(\theta_k, \lambda)$ of the current angular sequence θ_k is constructed, and the expression of $L(\theta_k, \lambda)$ is shown in Equation (23).

$$L(\theta_k, \lambda) = f(\theta_k) + \lambda g(\theta_k) \tag{23}$$

where λ is the Lagrangian multiplier, and $\lambda \geq 0$.

(2) The descending direction d of the current angular sequence θ_k is obtained. Equation (23) is converted into the solution to d , which is shown in Equation (24).

$$\begin{aligned} \min_d \quad & \nabla f(\theta_k)^T d + \frac{d^T H_k d}{2} \\ \text{st:} \quad & g(\theta_k) + \nabla g(\theta_k)^T d \leq 0 \end{aligned} \tag{24}$$

In Equation (24), H_k represents the Hessian matrix of the Lagrangian function $L(\theta_k, \lambda)$.

(3) The angle sequence in which θ_{k+1} equals θ_k plus d is updated, and whether the result of the solution satisfies the KKT condition or reaches the maximum number of iterations is determined. If the KKT condition is satisfied, the iteration is stopped. Step 2 is repeated if the KKT condition is not satisfied. The KKT condition is shown in Equation (25) [25].

$$\begin{aligned} \nabla f(\theta_k) + \lambda g(\theta_k) &= 0 \\ \lambda &\geq 0 \\ g(\theta_k) &= 0 \\ \lambda g(\theta_k) &= 0 \end{aligned} \tag{25}$$

In the KKT condition, the third condition means that the result of the solution satisfies the constraint. If $g(\theta_k)$ equals 0, the KKT condition becomes $\nabla f(\theta_k)$, which equals 0. If $g(\theta_k)$ is smaller than 0 and λ equals 0, the KKT condition also becomes $\nabla f(\theta_k)$, which equals 0.

5. The Painting Trajectory

During the painting process, not only should the trajectory of the motion platform be planned, but the trajectory of the UR5 robot’s movement should also be planned. During the painting process, the posture of the UR5 robot end is shown in Equation (26).

$$\mathbf{Trans}(x, y, z) \cdot \mathbf{Rot}(x, p) \cdot \mathbf{T}_{robot} = \mathbf{T} \cdot \mathbf{T}_{dis}^{-1} \cdot \mathbf{T}_{tool}^{-1} \tag{26}$$

Since $\mathbf{Rot}(x, \pi)$ only affects the directions of the x -axis and z -axis of the 3-dimensional Cartesian system of the UR5 robot system, it does not change the range of the UR5 robot’s operating space. Equation (26) can be transformed into Equation (27).

$$\mathbf{Trans}(x, y, z) \cdot \mathbf{T}_{robot}' = \mathbf{T} \cdot \mathbf{T}_{dis}^{-1} \cdot \mathbf{T}_{tool}^{-1} \tag{27}$$

where $\mathbf{T}'_{UR5\ robot}$ equals $\mathbf{Rot}(x, \pi) \cdot \mathbf{T}_{UR5\ robot}$.

5.1. The Trajectory of the Motion Platform

When the size of the object to be painted exceeds the maximum painting area of the UR5 robot, the auxiliary movement of the motion platform is required to complete the painting. During the painting process, the motion platform is moved to bring the UR5 robot close to the target area, and the UR5 robot paints the area. After painting, the motion platform is moved again, and the UR5 robot paints another area. Then, the trajectory

planning of the motion platform is optimized by taking the least number of movements of the motion platform as the objective function.

Because the UR5 robot’s permitted painting space is a cylinder and the height of the painted object is generally less than the maximum height of the painting space, the model can be projected onto the xy plane. The model can be transformed to cover all target points with the fewest circles.

5.1.1. The Minimum Envelope Rectangle of the Target Points

Since the number of target points is large, in order to simplify the calculation, a rectangle enveloping the target points—instead of a set of target points—is harnessed. The solution model, which is shown in Figure 15, is created by employing the minimum rectangle method of the main direction target points. The dotted section in the figure represents the target points; the blue line represents the main direction of the target points, and the red rectangle represents the minimum envelope rectangle of the target points.

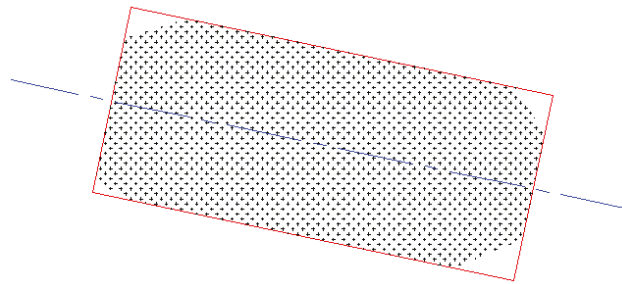


Figure 15. The minimum envelope rectangle of the target points.

The method for obtaining the principal direction of the plane is shown in Equation (28).

$$\theta = \frac{\text{atan2}(2M_{11}, M_{20} - M_{02})}{2} \tag{28}$$

in which θ is the minimum angle between the major axis direction and the direction of the positive side of the x -axis, and M_{pq} is the $p + q$ order center moment of the projection surface. M_{pq} is shown in Equation (29).

$$M_{pq} = \sum_x \sum_y (x - \bar{x})^p (y - \bar{y})^q \tag{29}$$

where \bar{x} and \bar{y} represent the center coordinates of the original target point.

5.1.2. The Minimum Number of Movements of the Motion Platform

When a circle fills a rectangle, the effective filling area of each circle is generally rectangular. Then, the model can be simplified again by filling the target rectangle with the smallest number of rectangles inside the circle. The filling model is generally shown in Figure 16. The smaller rectangle in the figure represents the target rectangle that needs to be filled. The larger rectangle represents the largest filled area, and the circular areas outside the largest rectangle represent the areas not recommended for use by the UR5 robot. Each circle represents the maximum processing range of the system each time. P_x and P_y represent the vertical distance and horizontal distance between the target rectangle and the largest filled area; l and h represent the length and width of the effective rectangle inside each circle.

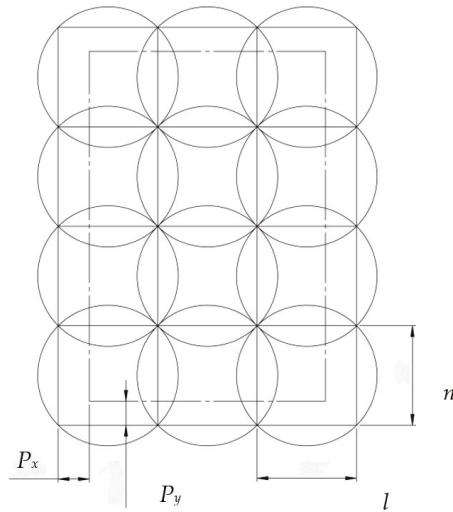


Figure 16. The schematic diagram of the filling model.

The size of the rectangle inside the circle indicates the effective area size of the circle. The length and width of the maximum inscribed rectangle in the circle are both $\sqrt{2}r$. The length and width of the inscribed rectangle of the circle are generally close to $\sqrt{2}r$ when the target rectangle is selected. The constraints of the filled model are shown in Equation (30).

$$\begin{aligned}
 n &= \left[l_{\max} / \sqrt{2}r \right] + 1 \\
 \sum_{i=1}^n l_i &\geq l_{\max} \\
 \sqrt{4r^2 - l_i^2} &\geq h_{\max}
 \end{aligned}
 \tag{30}$$

where $[l_{\max} / \sqrt{2}r]$ represents the largest integer that does not exceed $\frac{l_{\max}}{\sqrt{2}r}$.

Since the number of circles in Figure 16 is an integer, there are many kinds of $P_x, P_y, h_1,$ and h_2 satisfying Equation (29). However, the UR5 robot's non-recommended area may cause the movement of the motion platform. The minimum number of target points in the non-recommended area during processing is the optimization goal, and the optimal $P_x, P_y, h_1,$ and h_2 are obtained by employing the exhaustive method.

5.1.3. The Movement Sequence of the Motion Platform

Reasonably planning the movement sequence of the motion platform can reduce the movement of the motion platform. During processing, the initial point of the motion platform is at zero. The movement sequence planning model of the motion platform can be transformed into a TSP model starting from zero and returning to zero after accessing all target circle centers. Because the number of points is low, the problem lies in creating a small TSP model. The problem can be directly solved by employing the example method. Since each axis of the motion platform is controlled by a separate motor, the Halton distance between two points is taken as the weight when creating the TSP model.

5.2. The Kinematic Trajectory of the UR5 Robot

According to Figure 8, some painting areas overlap, and, therefore, the greedy principle is harnessed to include as many target points as possible for each time of painting. The painting process requires uniform motion during the painting process, and, therefore, the shorter the total painting path is and the shorter the painting time is, the higher the painting efficiency is.

When each area is painted, the painted model can be transformed into a TSP model that finds the shortest path which can access all of the target points [26]. When different areas are painted, in order to prevent the UR5 robot from colliding with the processed objects, the UR5 robot’s joints must be moved after returning to the origin of the 3-dimensional Cartesian system. Then, the UR5 robot kinematics trajectory planning model can be transformed into the solution of multiple TSP models.

However, due to the large number of points to be painted, this model is a large TSP model. Genetic algorithms (GA) are harnessed to create each TSP model [27,28].

5.2.1. The TSP Model Based on the Genetic Algorithm

A genetic algorithm-based computational model that simulates the evolutionary process of natural selection and genetic mechanisms of Darwin’s theory of evolution is proposed. The genetic algorithm does not need continuous function limitation and has a better global optimal solution. The core method of GA is to evaluate the fitness of all individuals in each generation of the population and select some individuals for genetic selection, crossover, and mutation to form the next generation population. The main solution steps of the genetic algorithm are shown in Figure 17.

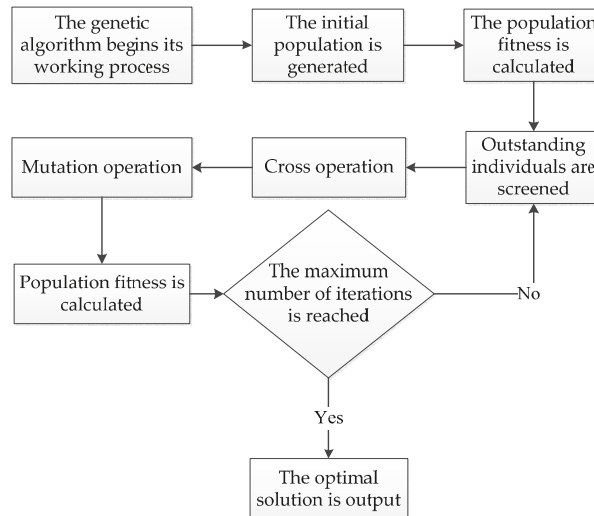


Figure 17. The solution steps of the genetic algorithm.

5.2.2. Operations Related to the Genetic Algorithm

(1) Coding method: all target points are encoded into one chromosome $G_1G_2 \dots G_wG_k$, and G_i in the chromosome represents the i -th number of the target point. In this genetic algorithm, each individual has one and only one chromosome.

(2) Weight: the Euler distance between two points is taken as the weight between the two points.

(3) Chromosome distance and individual fitness: the fitness of the individual f equals $N / \sum_{i=2}^n d_{i(i+1)}$, where N represents the gain coefficient. The value of N only affects the value of fitness and does not affect the final TSP solution result.

(4) Selection operation: the fitness of the population is calculated; the best individual is screened, and the coding method of the individual is retained. For the remaining individuals, the individual distribution function is calculated in accordance with their individual fitness. The maximum and minimum normalization algorithms are employed

to normalize the distribution function to 0-1. The selection operation process is shown in Figure 18.

- 1: An empty population vector of n is created.
- 2: The population records the best optimal individual
- 3: A random number of 0-1 is generated.
- 4: The maximum number of points is found which is less than or equal to the random number in line with the distribution function
- 5: The population records the points' numbers
- 6: If the size of the population is less than n , steps 3, 4, and 5 are repeated.
- 7: The population is output.

Figure 18. The selection operation process.

(5) Crossover operation: in order to ensure the positive optimization of the genetic algorithm, crossover operations are performed only on non-optimal individuals in the population. The crossover operator uses the Order Crossover operator. The crossover algorithm is shown in Figure 19, where P_c represents the crossover probability.

- 1: Two individual parent 1 and parent 2 are selected in the population in order. Their individual chromosomes are $G_1G_2...G_n$ and $G_nG_{n-1}...G_1$.
- 2: A random P number within 1 is generated.
- 3: If $P > P_c$, parent 1 and parent 2 are output and the cross operation of the individual is finished.
- 4: Cross positions *start* and *end* are randomly generated.
- 5: Individual chromosomal gene fragments are generated:
 offspring 1: ... $G_{start}...G_{end}$... and offspring 2: ... $G_{end}...G_{start}$
 ...
- 6: The unknown gene fragments in offspring1 are filled in conformity with the genetic order of the parent 2 and the unknown gene fragments in offspring 2 are filled as per the genetic order of the parent1. The genes in the offspring are not duplicated during the filling process. For example, if $n = 5$, $start = 2$ and $end = 3$, offspring 1's chromosomal is $G_5G_2G_3G_4G_1$ and offspring 2's chromosomal is $G_1G_4G_3G_2G_5$.
- 7: Offspring 1 and offspring 2 are output.

Figure 19. The crossover algorithm.

(6) Mutation operation: in order to prevent the GA from falling into a local optimal solution, an adaptive mutation rate is harnessed. The equation for calculating the mutation probability is shown in Equation (31),

$$P_m = \begin{cases} \frac{K_1(f_{max}-f)}{f_{max}-f_{avg}}, & f \geq f_{avg} \\ K_2, & f < f_{avg} \end{cases} \quad (31)$$

where f_{max} represents the maximum fitness of the group; f is the fitness of the individuals to be crossed. K_1 and K_2 are the adaptive parameters. The mutation operation is shown in Figure 20.

- 1: One individual in the population in order is selected. The individual's chromosome is $G_1G_2...G_n$
- 2: A random P number within 1 is generated.
- 3: If $P > P_c$, this individual is output and the cross operation of the individual is finished.
- 4: Cross positions *start* and *end* are randomly generated.
- 5: The genes are reversed from start to end in an individual and the chromosome of this individual will become $G_1...G_{start-1}G_{end} ...G_{start}G_{end+1}...G_n$
- 6: This individual is output.

Figure 20. The mutation operation.

6. Simulation Experiments

The spray gun of model WA-101 was adopted as the painting equipment, and the offset distance of the spray gun from the UR5 robot end axis was $T_{tool} = Trans(5.5, 94.5, 165.5)$.

The painting process requirements are described in this paragraph. The distance between the fixture and the target point was T_{dis} , which equals $Trans(0, 0, 113)$, and the end of the spray gun generated a circle with a radius of 50 mm at this distance. The maximum position error of the target point was 5 mm, allowing a 5° error between the gun's end axis and the target's normal vector.

6.1. The Pose Acquisition of the Target Objects

The analysis of the STL file of a car is shown in Figure 21, and the point cloud model of the car is shown in Figure 22. The shapes and sizes of Figures 21 and 22 are the same as the actual three-dimensional model. The target extraction algorithm proposed in this paper has promising practicality.

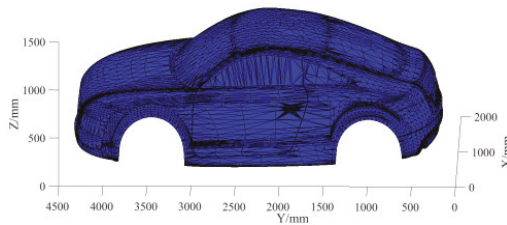


Figure 21. The STL analysis model of a car.

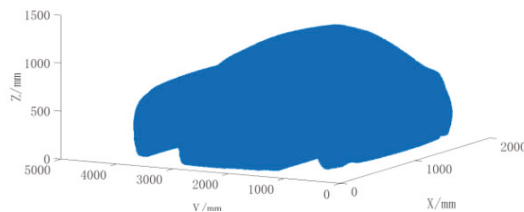


Figure 22. The point cloud model of the car.

The part of the car roof where z coordinates are larger than 1000 mm was taken for the target points for painting. The normal vector at the end of the target point was created by employing the PCA algorithm, as is shown in Figure 23 (because the performance of the computer graphics card was not satisfactory, and only the normal phase of some points is shown in the figure). As can be seen from Figure 23, the normal vectors of the target points conform to the shape of the roof, and the algorithm can better establish the normal vectors

of the target points. After the variable r in Equation (6) had been given a value of 50, the model of the compression points of the car roof, which is shown in Figure 24, was achieved as per Equation (7). After Figures 23 and 24 are compared, it is clear that the size and shape of the target points remain unchanged after compression.

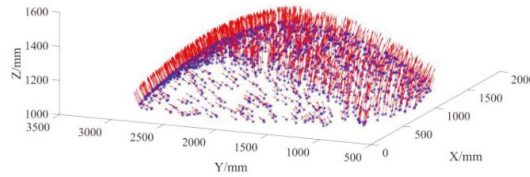


Figure 23. Normal vectors of target trajectory points.

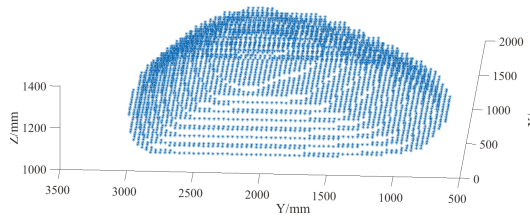


Figure 24. Compressed point cloud data.

6.2. Simulation of the Establishment of the Motion Platform Trajectory

After Equation (27) is given values T_{tool} and T_{dis} , the distribution of points at the ends of the UR5 robot joints is shown in Figure 25. By utilizing the smallest rectangle model in Figure 15, the smallest rectangle containing the target points was obtained, which is shown in Figure 26.

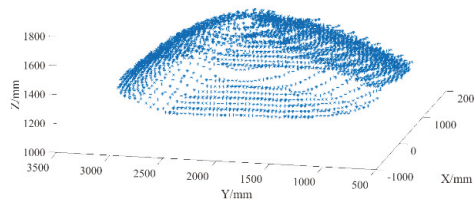


Figure 25. Distribution of points at the ends of the UR5 robot's joints.

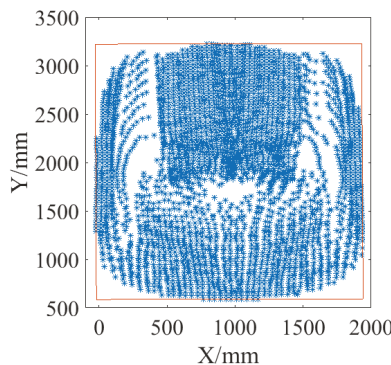


Figure 26. The minimum envelope rectangle.

The rectangle in Figure 26 has a width of 2635 mm and a length of 1959 mm. Data in Figure 26 were extracted and employed in Figure 16 and Equation (30) to obtain the preliminary partition model of the target points, which is shown in Figure 27.

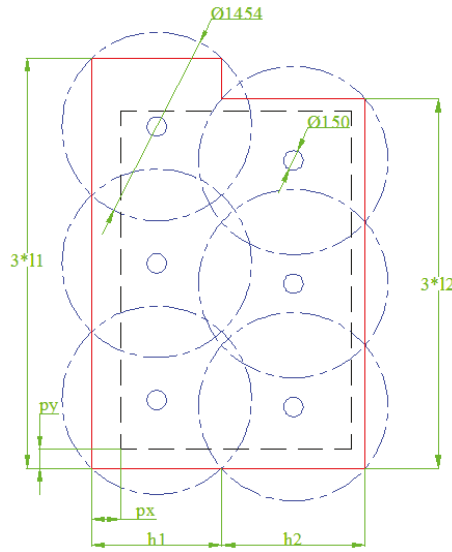


Figure 27. The preliminary partition model of the target points.

The model in Figure 27 was created by the exhaustive method. The result is shown in Figure 28, which describes the minimum number of target points in the non-recommended area when h_1 and h_2 are determined. The blue areas in the figure represent an invalid combination that does not satisfy Equation (30). In this optimal combination, h_1 equals 980 mm; P_x equals 7 mm; and P_y equals 3 mm.

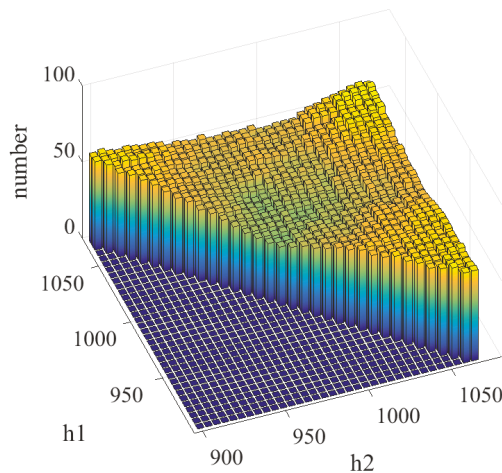


Figure 28. Statistical results of the objective function.

The model in Figure 27 was given values P_x , P_y , h_1 , and h_2 to obtain the partition model of the target points which are shown in Figure 29. The red circles in Figure 29 are the

maximum processing area and the non-recommended processing area of the UR5 robot. C_1 represents the coordinates of the i -th circle, and the coordinates of C_i are shown in Table 2.

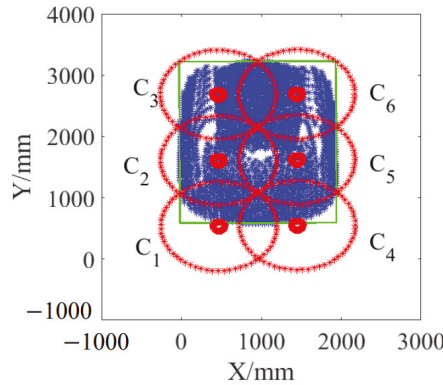


Figure 29. The partition model of the target points.

Table 2. The center point C_i .

C_i	X/mm	Y/mm
1	469	2685
2	465	1611
3	461	537
4	1451	552
5	1447	1622
6	1443	2691

All of the possibilities in Figure 29 were iterated, and the processing order with the smallest total movement of the motion platform was taken. The final painting order of the UR5 robot was $C_1 C_2 C_3 C_6 C_5 C_4$. The painting intervals of the UR5 robot are shown in Figure 30. The dots in different colors in Figure 30 represent different painting intervals.

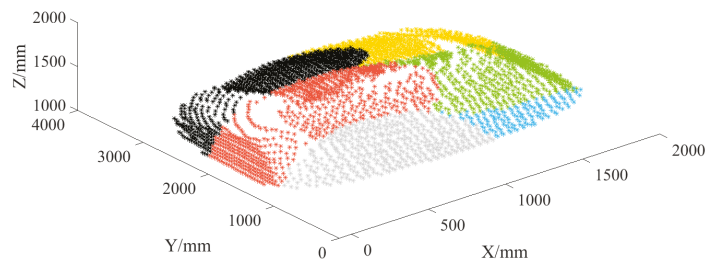


Figure 30. The painted zones of target points.

6.3. The Establishment of the Motion Platform Trajectory

The target points with the same color in Figure 30 were introduced into and utilized in GAs, and the population number was set to 5000. The maximum number of iterations is 25,000, and the cross probability was 0.9. K_1 equals 0.15, and K_2 equals 0.2 in the selection probability. The GA solution process is shown in Figure 31, and the final painting trajectory is shown in Figure 32.

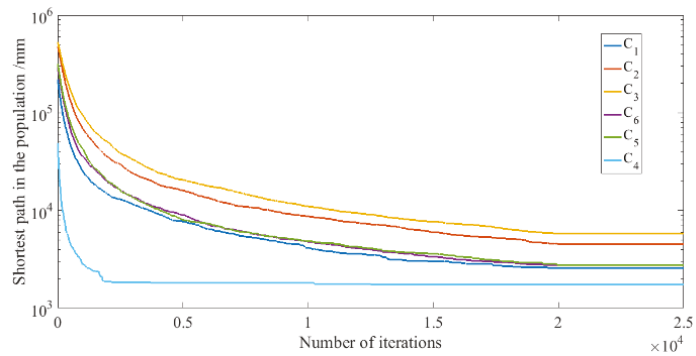


Figure 31. The iterative process of the genetic algorithm.

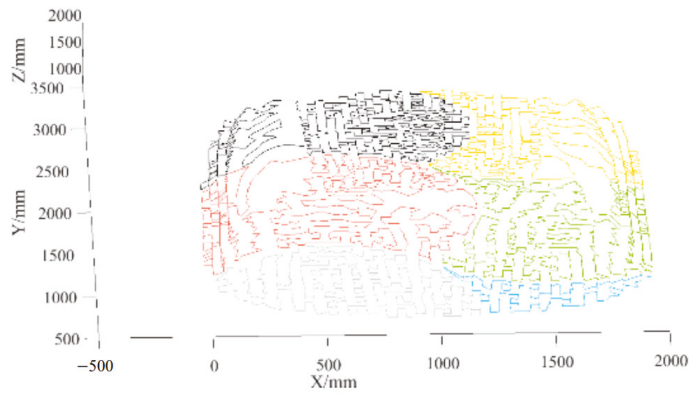


Figure 32. The final painting trajectory.

6.4. The Inverse Kinematics Simulation of the UR5 Robot

When the UR5 robot found its inverse kinematics, it generated multiple solutions. When the UR5 robot’s inverse kinematics were being solved, the angle values of the target points on the previous painting trajectory were harnessed as the initial angle values, and the target points on the final painting trajectory in each colored area with the smallest radian changes from the initial angle values were selected. The solution was the inverse kinematics of the target points. The position errors and axis errors of the statistical UR5 robot simulation are shown in Figures 33 and 34, respectively.

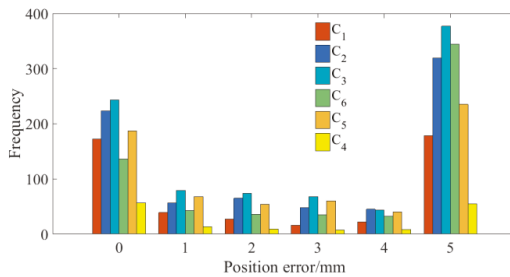


Figure 33. The UR5 robot’s position errors.

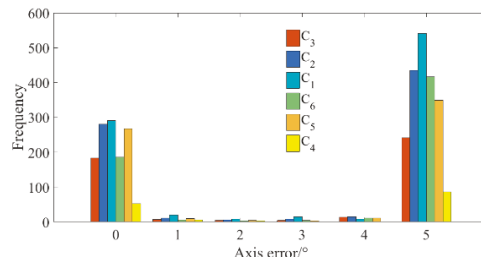


Figure 34. The UR5 robot's axis errors.

As per Figures 33 and 34, the errors are within the technical requirements of painting large objects, and the algorithm has satisfactory engineering practicability. Since the main solution in this paper was an analytical solution, when the joint radian of the UR5 robotic manipulator's motion platform in the recommended area needed to be solved, an iterative method was harnessed to solve Equations (14) to (18).

7. Discussion and Conclusions

This paper proposes an algorithm that employs a nine-axis robotic manipulator to automatically paint large objects. With the proposed algorithm, automatic processing of complex objects is achieved. The algorithm is divided into three aspects that consist of extraction of the target model, the establishment of the inverse kinematics model of the manipulator, and the planning of the target trajectory.

In the section of the proposed algorithm for extracting the target model, as per the STL file of the target trajectory, the triangles of the target trajectory are extracted. A full 3D data point cloud model is obtained with a triangle-based filling algorithm. Subsequently, the PCA algorithm is harnessed to identify the normal vectors of the target points inside the point cloud model. With the normal vectors, Rodrigues' rotation formula is harnessed to extract the pose of each point of the painting trajectory. Finally, the number of target points is compressed to reduce the time and space complexity of the algorithm so that the painting process requirements can be satisfied.

In the section of the proposed algorithm where the inverse kinematics model of the manipulator is established, in conformity with the processing range of the robot, the inverse solution algorithm is divided into the inverse solution of the recommended region and the inverse solution of the non-recommended region. The corresponding inverse kinematics model is established by employing the closed-form solution method and SQP, respectively.

During the planning of the painting trajectory, in line with the point cloud model of the target points, the minimum envelope rectangles of the target points are found. The target points are divided into different painting areas in accordance with the minimum rectangles. For each painting area, a TSP trajectory planning model based on GA is harnessed to plan the robot's painting trajectory with which the inverse kinematics model of the UR5 robot is created.

Not only does the trajectory created by the proposed algorithm consider the singularities of the kinematic joints of the UR5 robot joints, but it also helps the UR5 robot to paint large objects with precision and efficiency. Multiple reliable simulations of the painting process on the car roof surface were conducted, and the results show that the algorithm can meet the technical requirements of painting and that the algorithm has promising practicability.

The proposed algorithm has several benefits. The trajectory created by the proposed algorithm allows the motion platform of the UR5 robotic manipulator to have the least amount of movement while the UR5 robot paints a large object. Therefore, the proficiency of the painting process can be significantly improved. The standard triangle language (STL) file, algorithm of principal component analyses (PCA), and k-dimensional tree (k-d tree) were employed to obtain the point cloud model of the car roof to be painted. The

point cloud model was later converted into multiple traveling salesman problem (TSP) models, each of which was created with genetic algorithms (GAs). In this way, the UR5 robot could identify the object to be painted and generate a trajectory to paint the large object more efficiently.

However, limitations still exist in the current research. The closed-form solution method and SQP were employed to establish the inverse kinematics model of the UR5 robot. SQP is not intelligent enough and still requires a myriad of training solutions. If a more intelligent neural network model had been utilized, the preparation time for the entire painting process would have been reduced. In this research, only the painting trajectory was created, but the dynamics and velocity-planning of the UR5 robot were not involved. Therefore, future studies need to focus on a more intelligent neural network model that helps to establish the inverse kinematics model of the UR5 robot as well as the dynamics and velocity-planning of the UR5 robot.

Author Contributions: Conceptualization, F.L.; Data curation, J.W.; Formal analysis, M.Y. and K.Z.; Funding acquisition, J.W. and Q.W.; Investigation, M.Y. and F.L.; Methodology, M.Y., F.L. and K.Z.; Project administration, Q.W.; Resources, K.Z.; Software, F.L.; Supervision, J.W.; Visualization, K.Z.; Writing—original draft, K.F.; Writing—review & editing, M.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The study did not report any data.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Craig, J.J. *Introduction to UR5 Robotics: Mechanics and Control*, 3rd ed.; Prentice Hall: Upper Saddle River, NJ, USA, 2008.
2. Goetz, J.; Kiesler, S.; Powers, A. Matching UR5 Robot Appearance and Behavior to Tasks to Improve Human-UR5 Robot Cooperation. In Proceedings of the 12th IEEE International Workshop on UR5 Robot and Human Interactive Communication, Millbrae, CA, USA, 2 November 2003; pp. 55–60.
3. Brogårdh, T. Present and future UR5 robot control development—An industrial perspective. *Annu. Rev. Control* **2007**, *31*, 69–79. [\[CrossRef\]](#)
4. Wang, Y.; Liu, S.; Xu, D.; Zhao, Y.; Shao, H.; Gao, X. Development and Application of Wall-Climbing UR5 robots. In Proceedings of the 1999 IEEE International Conference on UR5 Robotics and Automation (Cat. No.99CH36288C), Detroit, MI, USA, 10–15 May 1999; Volume 2, pp. 1207–1212.
5. Form, P.J.; Gravidahl, J.T.; Pettersen, K.Y. Kinematics of vehicle-manipulator systems. In *Vehicle-Manipulator Systems. Advances in Industrial Control*; Springer: London, UK, 2014; pp. 169–189. [\[CrossRef\]](#)
6. Ren, S.; Yang, X.; Xu, J.; Wang, G.; Xie, Y.; Chen, K. Determination of the base position and working area for mobile manipulators. *Assem. Autom.* **2016**, *36*, 80–88. [\[CrossRef\]](#)
7. Li, Z.; Zhao, D.; Zhao, J. Structure synthesis and workspace analysis of a telescopic painting UR5 robot. *Mech. Mach. Theory* **2019**, *133*, 295–310. [\[CrossRef\]](#)
8. Bureerat, S.; Pholdee, N.; Radpukdee, T.; Jaroenapibal, P. Self-adaptive MRPBIL -DE for 6D UR5 robot multiobjective trajectory planning. *Expert Syst. Appl.* **2019**, *136*, 133–144. [\[CrossRef\]](#)
9. Yin, S.; Ji, W.; Wang, L. A machine learning based energy efficient trajectory planning approach for industrial UR5 robots. *Procedia CIRP* **2019**, *81*, 429–434. [\[CrossRef\]](#)
10. Serralheiro, W.; Maruyama, N.; Saggini, F. Self-Tuning Time-Energy Optimization for the Trajectory Planning of a Wheeled Mobile UR5 robot. *J. Intell. Robot. Syst.* **2019**, *95*, 987–997. [\[CrossRef\]](#)
11. Kazim, I.J.; Tan, Y.; Li, R. Comparison study of the PSO and SBPSO on universal robot trajectory planning. *Appl. Sci.* **2022**, *12*, 1518. [\[CrossRef\]](#)
12. Kazim, I.J.; Tan, Y.; Qaseer, L. Integration of DE algorithm with PDC-APF for enhancement of contour path planning of a universal robot. *Appl. Sci.* **2021**, *11*, 6532. [\[CrossRef\]](#)
13. Al-shanon, A.; Lang, H. Robotic manipulation based on 3-D visual servoing and deep neural networks. *Robot. Auton. Syst.* **2022**, *152*, 104041. [\[CrossRef\]](#)
14. Balanji, H.M.; Turgut, A.E.; Tunc, L.T. A novel vision-based calibration framework for industrial robotic manipulators. *Robot. Comput. Manuf.* **2022**, *73*, 102248. [\[CrossRef\]](#)

15. Vivas, A.; Sabater, J.M. UR5 Robot Manipulation Using Matlab/Simulink and ROS. In Proceedings of the 2021 IEEE International Conference on Mechatronics and Automation, Takamatsu, Japan, 8–11 August 2021. [[CrossRef](#)]
16. Araki, R.; Mano, K.; Hirano, T.; Hirakawa, T.; Yamashita, T.; Fujiyoshi, H. Iterative Coarse-to-Fine 6D-Pose Estimation Using Back-Propagation. In Proceedings of the 2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Prague, Czech Republic, 27 September–1 October 2021. [[CrossRef](#)]
17. Sunil, V.B.; Pande, S.S. Automatic recognition of features from freeform surface CAD models. *Comput. Aided Des.* **2008**, *40*, 502–517. [[CrossRef](#)]
18. Hallmann, M.; Goetz, S.; Schleich, B. Mapping of GD&T information and PMI between 3D product models in the STEP and STL format. *Comput. Aided Des.* **2019**, *115*, 293–306. [[CrossRef](#)]
19. Deun, K.V.; Thorrez, L.; Coccia, M.; Hasdemir, D.; Westerhuis, J.A.; Smilde, A.K.; Mechelen, I.V. Weighted sparse principal component analysis. *Chemom. Intell. Lab. Syst.* **2019**, *195*, 103875. [[CrossRef](#)]
20. Bro, R.; Kjeldahl, K.; Smilde, A.K.; Kiers, H.A.L. Cross-validation of component models: A critical look at current methods. *Anal. Bioanal. Chem.* **2008**, *390*, 1241–1251. [[CrossRef](#)]
21. Markiewicz, P.J.; Matthews, J.C.; Declerck, J.; Herholz, K. Verification of predicted robustness and accuracy of multivariate analysis. *NeuroImage* **2011**, *56*, 1382–1385. [[CrossRef](#)] [[PubMed](#)]
22. Hu, L.; Nooshabadi, S. High-dimensional image descriptor matching employing highly parallel KD-tree construction and approximate nearest neighbor search. *J. Parallel Distrib. Comput.* **2019**, *132*, 127–140. [[CrossRef](#)]
23. Liao, H.; Wu, W.; Fang, D. The reduced space Sequential Quadratic Programming (SQP) method for calculating the worst resonance response of nonlinear systems. *J. Sound Vib.* **2018**, *425*, 301–323. [[CrossRef](#)]
24. Lee, J.H.; Jung, Y.M.; Yuan, Y.-X.; Yun, S. A subspace SQP method for equality constrained optimization. *Comput. Optim. Appl.* **2019**, *74*, 177–194. [[CrossRef](#)]
25. Singh, D.; Dar, B.A.; Kim, D.S. KKT optimality conditions in interval valued multiobjective programming with generalized differentiable functions. *Eur. J. Oper. Res.* **2016**, *254*, 29–39. [[CrossRef](#)]
26. Zhou, Y.; Luo, Q.; Chen, H.; He, A.; Wu, J. A discrete invasive weed optimization algorithm for solving traveling salesman problem. *Neurocomputing* **2015**, *151*, 1227–1236. [[CrossRef](#)]
27. Ahmadi, E.; Goldengorin, B.; G. Süer, A.; Mosadegh, H. A hybrid method of 2-TSP and novel learning-based GA for job sequencing and tool switching problem. *Appl. Soft Comput.* **2018**, *65*, 214–229. [[CrossRef](#)]
28. Liu, F.; Zeng, G. Study of genetic algorithm with reinforcement learning to solve the TSP. *Expert Syst. Appl.* **2009**, *36*, 6995–7001. [[CrossRef](#)]

Article

High-Precision SLAM Based on the Tight Coupling of Dual Lidar Inertial Odometry for Multi-Scene Applications

Kui Xiao ¹, Wentao Yu ^{1,*}, Weirong Liu ², Feng Qu ¹ and Zhenyan Ma ¹

¹ College of Computer and Information Engineering, Central South University of Forestry and Technology, Changsha 410004, China; 20191100304@csuft.edu.cn (K.X.); qufeng0817@csuft.edu.cn (F.Q.); t20070552@csuft.edu.cn (Z.M.)

² School of Computer Science and Engineering, Central South University, Changsha 410083, China; frat@csu.edu.cn

* Correspondence: wentaoyu@csuft.edu.cn; Tel.: +86-0731-134-6758-9376

Featured Application: This paper fuses different sensors to form a general high-precision SLAM framework for multi-scene applications. The algorithm framework in this paper can be extended to the fields of autonomous driving, robot navigation, and 3D reconstruction.

Abstract: Simultaneous Localization and Mapping (SLAM) is an essential feature in many applications of mobile vehicles. To solve the problem of poor positioning accuracy, single use of mapping scene, and unclear structural characteristics in indoor and outdoor SLAM, a new framework of tight coupling of dual lidar inertial odometry is proposed in this paper. Firstly, through external calibration and an adaptive timestamp synchronization algorithm, the horizontal and vertical lidar data are fused, which compensates for the narrow vertical field of view (FOV) of the lidar and makes the characteristics of vertical direction more complete in the mapping process. Secondly, the dual lidar data is tightly coupled with an Inertial Measurement Unit (IMU) to eliminate the motion distortion of the dual lidar odometry. Then, the value of the lidar odometry after correcting distortion and the pre-integrated value of IMU are used as constraints to establish a non-linear least-squares objective function. Joint optimization is then performed to obtain the best value of the IMU state values, which will be used to predict the state of IMU at the next time step. Finally, experimental results are presented to verify the effectiveness of the proposed method.

Keywords: simultaneous localization and mapping; dual lidar inertial odometry; IMU; time synchronization; tight coupling

Citation: Xiao, K.; Yu, W.; Liu, W.; Qu, F.; Ma, Z. High-Precision SLAM Based on the Tight Coupling of Dual Lidar Inertial Odometry for Multi-Scene Applications. *Appl. Sci.* **2022**, *12*, 939. <https://doi.org/10.3390/app12030939>

Academic Editors: António Paulo Moreira, Pedro Neto and Félix Vidal

Received: 30 November 2021

Accepted: 13 January 2022

Published: 18 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Simultaneous localization and mapping (SLAM) require building a map of an unknown environment by a mobile vehicle and simultaneously localizing the vehicle in such a map [1–3]. SLAM is essential for vehicles to fulfill many tasks, including vehicle rescue [4] and exploration [5]. The perception of the unknown external environment by various onboard sensors provides vital information for SLAM. Thus, integrating different sensors to develop a practical SLAM framework that can be applied in multiple scenes is essential.

Generally, both vision-based and lidar-based SLAM [6–10] are used. Although the vision-based SLAM can obtain high-precision positioning [11–13], the vision sensors are vulnerable to light change in the environment and cannot work in dark or untextured scenes. In comparison, lidar is not affected by light and can usually measure the angle and distance of obstacles with higher accuracy. Therefore, this paper focuses on the design of lidar-based SLAM to adapt the multi-scene applications.

In recent years, many different lidar-based SLAM schemes have been proposed. Among them, the Lidar Odometry and Mapping in Real-time (LOAM) method, where a single-line lidar and a motor are used to form a multiline lidar to realize low-drift and

low-calculation real-time positioning and mapping, has been studied extensively [14]. In the LOAM method, SLAM is divided into two parts: lidar odometry and lidar mapping. In the lidar odometry part, to reduce the computation, the plane smoothness of the lidar point cloud, which is utilized to distinguish the edge points, is calculated according to the curvature and then invalid point clouds are discarded to improve the feature of point cloud. After the point cloud information is obtained through feature extraction, a scan-to-scan method [15] is proposed to realize feature matching between frames, including edge points and planar points matching. To eliminate the motion distortion of lidar, linear interpolation is utilized. After the distortion is eliminated, the pose transformation of the lidar point cloud data of two adjacent frames is acquired to obtain the lidar odometry. Then, after accumulating a certain number of point cloud data, lidar mapping is performed through a map-to-map matching method [16]. Although LOAM can create a high-precision point cloud map, it cannot remove moving people or objects during feature extraction. Furthermore, in an environment with fewer feature points, it is easy for the lidar odometry to fail, resulting in positioning and mapping with worse accuracy.

To solve the above problems, a Lightweight and Ground-Optimized Lidar Odometry and Mapping on Variable Terrain (LeGO-LOAM) [17] is proposed. Its core comprises four modules: point cloud segmentation and denoising, feature extraction, lidar odometry, and lidar mapping. Firstly, the point cloud segmentation technology [18] solves the defects of moving people or objects in LOAM mapping and filters out noise. Then, feature extraction is applied to obtain planar and edge features from the segmented point cloud. The lidar odometry module uses the features to find out the optimal pose transform between two consecutive scans with the help of a two-step Levenberg–Marquardt (LM) optimization method [19]. The extracted features are further processed by a scan-to-map [20] matching method to obtain a global point cloud map in lidar mapping. LeGO-LOAM optimizes LOAM to a certain extent, but in large scenes (e.g., long corridors) or environments with few feature points, the low frequency of lidar and less characteristic information can lead to significant errors in positioning and mapping.

Some recent research has used low-cost IMU [21–23] to assist lidar for SLAM. The simplest way to integrate the IMU with lidar is loose coupling [24], in which IMU is regarded as an independent module to assist the lidar. The authors of [25] loosely couple IMU and optional GPS measurement with lidar through the Extended Kalman Filter (EKF) to improve computational efficiency and accuracy. However, the IMU error will continue to accumulate in the case of long distances, so the localization error is still increasing in such cases. To solve this problem, IMU and lidar are generally coupled via a tight coupling method that usually offers improved accuracy. A tightly coupled lidar inertial odometry and mapping framework (LIO-Mapping) is introduced in the literature [26]. It comprises the state optimization for the odometry and the refinement with rotational constraints. Results showed that this method outperformed the state-of-the-art lidar-only and loosely coupled methods. Since LIO-Mapping is designed to process all sensor measurements, real-time performance is not achieved. The authors of [27] proposed a tightly coupled Lidar Inertial Odometry via Smoothing and Mapping (LIO-SAM) based on LeGO-LOAM. LIO-SAM performs highly accurate, real-time mobile vehicle trajectory estimation and map building, suitable for multi-sensor fusion and global optimization. Although the tight coupling of lidar and IMU can improve the localization accuracy, lidar sensors have certain drawbacks. Conventional lidar can only be used to scan the environment in a narrow range of vertical angles and the point cloud feature information obtained is limited. For example, in an indoor corridor or staircase environment, lidar can receive the point cloud information of the sidewall, but only a tiny part of the point cloud information is obtained from the floor and ceiling. In such cases, the matched lidar features can easily cause ill-constrained pose estimation and incomplete mapping.

Aiming to broaden the FOV of lidar, lidars can be actuated in a number of ways. The periodic nodding and continuous rotation can be adopted and the resulting configurations can potentially enlarge the FOV. While the implementation of this method depends on

a complex mechanical structure, the localization accuracy is relatively low [28,29]. More recently, a new design for a 3D sensor system was introduced [30], involving construction from a 2D range scanner coupled with a passive linkage mechanism, such as a spring. However, the system requires reasonably continual excitation to induce motion of the sensor head. Therefore, the system is not considered appropriate for electric ground vehicles operating with infrequent accelerations on smooth terrain. In order to solve the above problems, some researchers have started investigating the use of multiple 3D lidars for better coverage of the environment. In [31], a high-precision lidar odometry system was proposed to achieve robust and real-time operation under challenging perceptual conditions. In this system, the two lidars are mounted at 180 degrees to each other to make up for the self-similar areas with low lidar observability. The authors of [32] proposed a system to achieve robust and simultaneous extrinsic calibration, odometry, and mapping for multiple lidars, while [33] proposed a scheme to combine multiple lidars with complementary FOV for feature-based lidar-inertia odometry and mapping. While the above methods can work well in single-scene applications, their adaptability to different environments was not considered, especially the environments that need to perceive height information.

This paper proposes a high-precision SLAM framework based on tight coupling of dual lidar inertial to overcome the above problems. In this framework, the horizontal lidar and the vertical lidar are integrated to broaden the FOV. Then the dual lidar and IMU are tightly coupled to improve the localization accuracy. In the outdoor environment, this paper introduces GPS measurements to further improve positioning accuracy. The main contributions of this paper include:

1. A general high-precision SLAM framework is provided by fusing different sensors. It can adapt to multi-scene applications, such as a corridor with fewer features, stairs with height, and complex outdoor environments.
2. The horizontal and vertical lidars are fused by external calibration and adaptive time synchronization algorithms to solve the narrow vertical FOV of the lidar.
3. To improve the positioning accuracy of SLAM in the environment with height information (e.g., stairs), the dual lidar odometry and IMU are tightly coupled. The dual lidar odometry measurement and IMU pre-integration are jointly optimized to obtain more accurate IMU state values, which will be used to eliminate the motion distortion of the dual lidar to improve the localization accuracy.
4. In addition, several practical experiments are carried out to verify the feasibility and effectiveness of the proposed method.

2. Prerequisites

2.1. IMU State Prediction and Pre-Integration

Usually, the frequency of IMU is much higher than lidar. IMU can obtain the acceleration and angular velocity at each time step, which can be used to predict the next state of IMU through integration.

In general, the IMU states at time t can be modeled as:

$$X_t^W = [p_t, v_t, R_t, b_t] \quad (1)$$

where X_t^W represents the state in the world frame W at time t ; p_t , v_t , and R_t represent the position, velocity, and rotation matrix, respectively, at time t ; and \hat{a}_t and $\hat{\omega}_t$ are the acceleration and angular velocity of the raw IMU measurements (\hat{a}_t and $\hat{\omega}_t$ are affected by a slowly varying bias, b_t).

The next state, $X_{t+\Delta t}^W$ is predicted through the integration of IMU, where Δt is the interval between two consecutive IMU measurements. The state prediction value of the

IMU is then used to infer the motion of the vehicle. The velocity, position, and rotation of the vehicle at time $t + \Delta t$ can be computed by Equation (2) [27]:

$$\begin{aligned} v_{t+\Delta t} &= v_t + g\Delta t + R_t(\hat{a}_t - b_t^a - n_t^a)\Delta t \\ p_{t+\Delta t} &= p_t + v_t\Delta t + \frac{1}{2}g\Delta t^2 + \frac{1}{2}R_t(\hat{a}_t - b_t^a - n_t^a)\Delta t^2 \\ q_{t+\Delta t} &= q_t \otimes \begin{bmatrix} \frac{1}{2}\Delta t(\hat{\omega}_{\Delta t} - b_{\Delta t}^\omega - n_{\Delta t}^\omega) \\ 1 \end{bmatrix} \end{aligned} \tag{2}$$

where g is gravitational acceleration; b_t^a is the varying bias of the acceleration \hat{a}_t ; and n_t^a is the white noise of the acceleration \hat{a}_t . The quaternion q_t under Hamilton notation (which corresponds to R_t) is used; \otimes is used for the multiplication of two quaternions; b_t^ω is the varying bias of the angular velocity $\hat{\omega}_t$; and n_t^ω is the white noise of the angular velocity $\hat{\omega}_t$.

The motion state of IMU between the i -th and the j -th time steps can be represented by the IMU pre-integrated measurement value $\Delta p_{ij}, \Delta v_{ij}, \Delta q_{ij}$, which can be computed by:

$$\begin{aligned} \Delta v_{ij} &= R_i^T(v_j - v_i - g\Delta t_{ij}) \\ \Delta p_{ij} &= R_i^T(p_j - p_i - v_i\Delta t_{ij} - \frac{1}{2}g\Delta t_{ij}^2) \\ \Delta q_{ij} &= q_i^{-1} \otimes q_j = \prod_{k=i}^{j-1} \begin{bmatrix} \frac{1}{2}\Delta t_{ij}(\hat{\omega}_k - b_k^\omega - n_k^\omega) \\ 1 \end{bmatrix} \end{aligned} \tag{3}$$

where $\Delta p_{ij}, \Delta v_{ij}, \Delta q_{ij}$ is the position, velocity, and rotation matrix of the IMU, respectively, which has the covariance C_{ij}^T in the error-state model. Due to space limitations, we invite readers to refer to the descriptions in [26,34] to understand the detailed derivation of IMU pre-integration.

2.2. Segmentation and Feature Extraction

Before extracting features, to filter out the noise, the point cloud is segmented to reduce noise interference. $D_t = \{d_1, d_2, \dots, d_n\}$ is the point cloud information acquired at time t ; d_τ is a point in D_t ; D_t is projected onto a range image; the point d_τ represents a pixel of the image; and the pixel value r_τ represents the Euclidean distance from the point d_τ to the lidar. An image-based segmentation method [17] is applied to the range image to group points into many clusters. Points of the same category have the same label (the ground is a particular category). Features are then extracted from ground points and segmented points, with the corresponding edge points and plane points obtained by calculating the curvature E of each point. To evenly extract features from all directions, the range image is divided horizontally into several equal sub-images. Then, the points are sorted in each row of the sub-image based on their curvature values, E . The curvature E is computed by Equation (4).

$$E = \frac{1}{|S| \cdot \|r_\tau\|} \sum_{\lambda \in S, \lambda \neq \tau} (r_\lambda - r_\tau) \tag{4}$$

where S is the set of continuous points d_τ from the same row of the range image; λ is a point in the set S ; E_{th} is a threshold which is set to distinguish different types of features; the points are called with $E > E_{th}$ edge features; and the points are called with $E < E_{th}$ planar features.

3. Multi-Sensor Fusion

3.1. Hardware System Description

The configuration of our system is shown in Figure 1, with the left model obtained through Unigraphics NX and the designed equipment in our paper shown on the right. The designed equipment is a scanning arm composed of various sensors that can be installed on a backpack or a vehicle. The sensor suite used in the scanning arm includes two 10 HZ Velodyne VLP-16 lidars, a 200 HZ YIS510A IMU, and a 5 HZ HY-269 GPS. The panoramic

camera prepares for the follow-up research. The lidar has a horizontal FOV of 360° with 0.4° resolution, a vertical FOV 15° with 2° resolution, and an accuracy of ± 3 cm. The Roll/Pitch of IMU accuracy is 0.25° , HY-269 GPS is a small-volume positioning and directional receiver, and the positioning accuracy of GPS is 1.2 m.

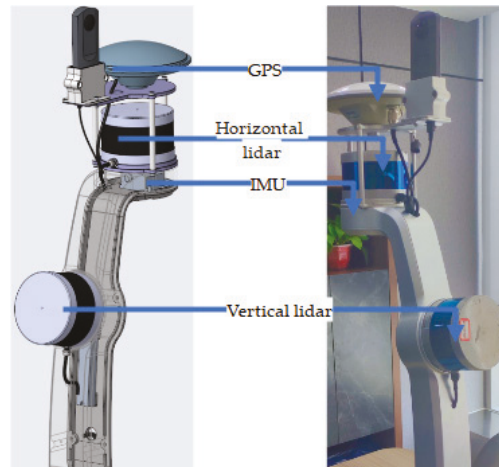


Figure 1. A CAD (Computer Aided Design) model of the scanning arm is shown on the left, with a photograph of the scanning arm shown on the right.

3.2. System Overview

In this paper, horizontal and vertical lidars are fused to make up for the shortcomings of lidar's narrow FOV. When indoors, accurate positioning information is obtained through the tight coupling of dual lidar inertial odometry. When outdoors, GPS measurements are added, providing a relatively accurate initial position for positioning, thereby further improving outdoor positioning accuracy.

Figure 2 provides a brief overview of our proposed framework. Firstly, after obtaining the IMU data, the IMU state prediction and pre-integration are updated, as detailed in Section 2.1. IMU pre-integration is used to construct a factor graph constraint, which will be used for joint optimization. Secondly, the dual lidar point clouds are obtained through external calibration and an adaptive timestamp synchronization algorithm, as detailed in Section 3.3. Since lidar has continuous measurement, the lidar measurement is obtained in continuous motion and it is almost certain that motion distortion will occur. When dual lidar point cloud is received, deskewing is applied on the dual lidar raw point cloud to obtain deskewed dual lidar point cloud [14]. To be ground-optimized and reduce the amount of calculation, point cloud segmentation is applied to filter out noise (e.g., moving people or objects) and the planar points and edge points are distinguished by calculating the curvature of the lidar point, as detailed in Section 2.2. After feature extraction, the dual lidar point cloud is registered to construct the lidar odometer factor. Then, the sliding window factor graph optimization method of local finite frame is used, with the IMU pre-integration factor and lidar odometer factor jointly optimized to realize the mutual parameter optimization of the dual lidar and IMU. Joint optimization is taken to obtain a maximum a posteriori (MAP) estimation of the IMU states, as detailed in Section 3.4, avoiding the drift from IMU propagation. To further improve outdoor positioning accuracy, GPS is introduced in this framework. When GPS is loosely coupled with IMU through an Unscented Kalman Filter (UKF), GPS measurements are used to further improve the positioning accuracy of IMU, with the optimized IMU state then used for the state estimation of IMU at the next time step. GPS measurements are added, providing a relatively accurate initial position for localization, thereby further improving outdoor localization accuracy [25]. However, the

drift of lidar inertial odometry grows very slowly; in practice, GPS measurements are used when the estimated position covariance is larger than the received GPS position covariance. Finally, the output of the figure is the global map and localization.

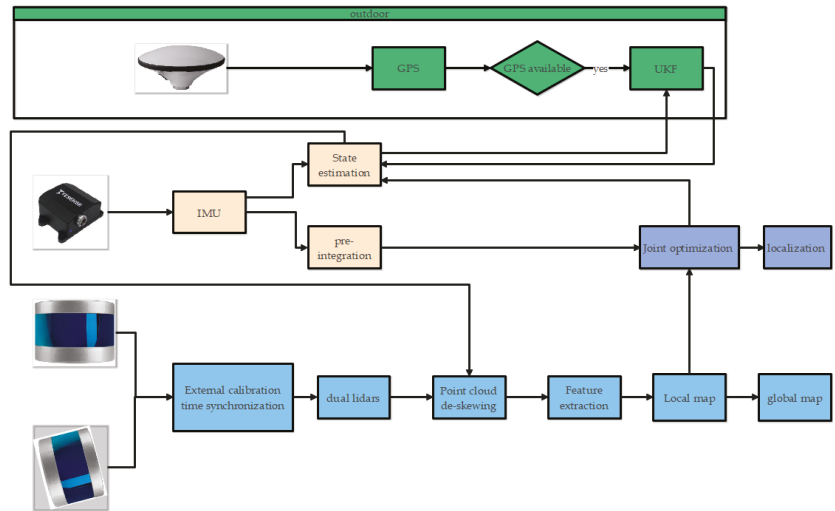


Figure 2. An overview of our proposed framework.

3.3. Fusion of Horizontal Lidar and Vertical Lidar

To accurately fuse the horizontal and vertical lidar data, their individual coordinate systems must be transformed into a unified coordinate system, which is termed “frame coordinate system”. The horizontal lidar coordinate system is used as the frame coordinate system; the vertical lidar coordinate system is registered in this frame coordinate system through external parameter calibration, with the external parameters obtained by the CAD model of the scanning arm. This paper adopts the adaptive time synchronization algorithm to ensure that the timestamps of the horizontal and vertical lidars can be simultaneously output. The fusion process of dual lidar is divided into two parts: (1) external parameter calibration of the horizontal and vertical lidars and (2) the adaptive time synchronization algorithm. Both parts are described as follows.

3.3.1. External Parameter Calibration of Horizontal Lidar and Vertical Lidar

Assuming that L_1 is the horizontal lidar coordinate system and L_2 is the vertical lidar coordinate system, the horizontal lidar coordinate system is used as the frame coordinate system and the vertical lidar coordinate system L_2 is registered in this frame coordinate system L_1 through the rotation matrix $\mathbb{R}_{L_2}^{L_1}$ and translation matrix $H_{L_2}^{L_1}$, which can be computed by Equation (5):

$$L_1 = \mathbb{R}_{L_2}^{L_1} \times L_2 + H_{L_2}^{L_1} \tag{5}$$

where $\mathbb{R}_{L_2}^{L_1}$ is the rotation matrix and $H_{L_2}^{L_1}$ is the translation matrix which can be obtained by external parameters. From the hardware structure of the scanning arm, the vertical lidar coordinate system first rotates around the y -axis and then translates to the horizontal lidar coordinate system, where

$$\mathbb{R}_{L_2}^{L_1} = \begin{bmatrix} \cos \theta & 0 & \sin \theta \\ 0 & 1 & 0 \\ -\sin \theta & 0 & \cos \theta \end{bmatrix} \tag{6}$$

where the angle of the rotation matrix $\theta = -77.94^\circ$ and the translation matrix $H_{L_2}^{L_1} = (x, 0, z)$, with $x = -0.177$ m and $z = -0.213$ m.

The point cloud of the lidar after external parameter calibration in the frame coordinate system is shown in Figure 3, with the point clouds of the vertical and horizontal lidars not affecting each other.

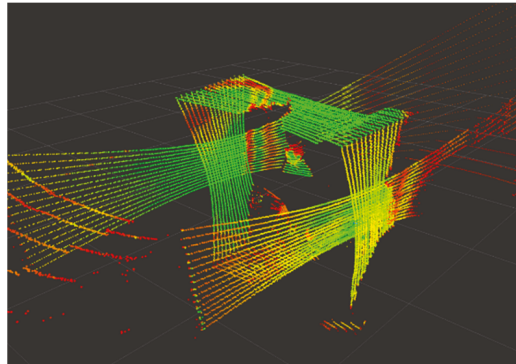


Figure 3. Dual lidar point clouds’ information after external parameter calibration.

3.3.2. The Adaptive Time Synchronization Algorithm

In this paper, the lidar time synchronization is achieved through hardware time synchronization and GNSS is used as the master clock. However, there is a time offset between the two lidars. As the time offset is constant, an adaptive timestamp synchronization algorithm is used to solve it. The adaptive time synchronization algorithm is used to match its timestamp and realize the simultaneous localization and mapping of the lidars. The timestamps of the dual lidar is shown in Figure 4, where lidar_201/scan represents the timestamp of the horizontal lidar and lidar_202/scan represents the timestamp of the vertical lidar.

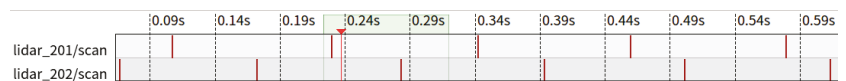


Figure 4. Timestamps of dual lidar.

The output of the adaptive time synchronization algorithm only depends on the timestamp, not on the arrival time of dual lidar point cloud messages. It means that the adaptive time synchronization algorithm can be safely used on dual lidar point cloud messages that have suffered arbitrary processing delays. As shown in Figure 5, time goes from left to right, with the first row representing the horizontal lidar timestamp and the second row representing the vertical lidar timestamp. Each dot represents the lidar point cloud with the timestamp. The blue dot represents the pivot of the lidar point clouds, with the broken line linking the dual lidar point clouds in a set. Suppose the horizontal lidar point clouds’ queue is PL_P and the vertical lidar point clouds’ queue is PV_P , with lidar point clouds inserted in a topic-specific queue (PL_P or PV_P) as they arrive. Once each topic-specific queue contains at least one message, the latest message is found at the head of the queues, known as the pivot. Assume that the queue with pivots is PL_P , which matches PV_P , being the smallest difference between the timestamps, T . The two queues are combined into a set; finally, this set is synchronous output.

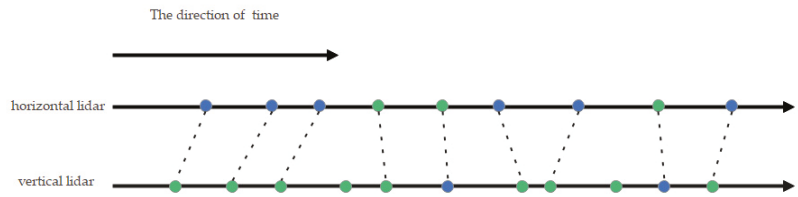


Figure 5. Timestamp matching of dual lidar.

3.4. Tight Coupling of Dual Lidar and IMU

Although the fusion of dual lidar makes up for the shortcomings of lidar’s narrow FOV, the localization accuracy is improved. However, lidars mounted on moving vehicles suffer from motion distortion, which directly affects the localization accuracy. IMU can accurately measure the three-axis acceleration and three-axis angular velocity of moving vehicles, and provide a priori information for lidar odometry. However, in the case of long or short distances, the IMU error will continue to accumulate, which directly affects positioning accuracy. In this paper, the tight coupling of dual lidar and IMU is proposed to solve the above problems. The process is divided into two parts: (1) external parameter calibration of horizontal lidar and IMU and time synchronization, and (2) joint optimization, which are described as follows:

3.4.1. External Parameter Calibration of Horizontal Lidar and IMU and Time Synchronization

Similar to Section 3.3, the external parameters of the lidar and IMU are also obtained based on the CAD model. To accurately fuse data from the horizontal lidar, the vertical lidar, and the IMU, the IMU coordinate system I is also registered in this frame coordinate system L_1 through the rotation matrix $\mathbb{R}_I^{L_1}$ and the translation matrix $H_I^{L_1}$. It can be seen from the structure of the scanning arm in Figure 1 that the IMU is located directly under the horizontal lidar. Therefore, the IMU coordinate system can be registered in the horizontal lidar coordinate system only by translation, where

$$\mathbb{R}_I^{L_1} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{7}$$

where the translation matrix $H_I^{L_1} = [-0.62.66, 0, -0.125]$.

In this paper, YIS510A IMU is used, which supports trigger correction of its internal clock by PPS signal. Lidar-IMU time synchronization is achieved through hardware time synchronization, with GNSS used as the master clock. GNSS can output PPS signals to unify the clock source of lidar-IMU and achieve time synchronization.

3.4.2. Joint Optimization

In this paper, a fixed-lag smoother and marginalization are introduced to obtain the optimal state. The fixed-lag smoother maintains a certain amount of IMU state in the sliding window, and the sliding window can effectively control the amount of calculation. When the new state enters the sliding window, the past state is marginalized. The state variable to be estimated for the whole window is $X = [X_\eta^W, \dots, X_\kappa^W, Z_I^{L_1}]$, where X_η^W is the state of IMU at the starting point η of the sliding window; X_κ^W is the state of IMU at the end of the sliding window; and $Z_I^{L_1} = [\mathbb{R}_I^{L_1}, H_I^{L_1}]$ is the external parameter between lidar and IMU. Then, the following cost function with a Mahalanobis norm is minimized to obtain the MAP estimation of the states X ,

$$X = \min_X \frac{1}{2} \left\{ \|u_o(X)\|^2 + \sum_{\chi \in Q_{L_\beta}} \|u_\phi(\chi, X)\|_{C_{L_\beta}^\chi}^2 + \sum_{\zeta \in \{\eta, \dots, \kappa-1\}} \|u_\gamma(Z_{\zeta+1}^\zeta, X)\|_{C_{L_{\zeta+1}}^\zeta}^2 \right\} \tag{8}$$

where $u_o(\mathbf{X})$ is the prior information from marginalization [26]. $u_\phi(\chi, \mathbf{X})$ is the residual of the relative lidar constraints that can be represented as point-to-plane distance [26], where $\chi \in Q_\beta$ is the residual for each relative lidar measurement with the previous correspondences. $\beta \in \{\eta + 1, \dots, \kappa\}$, $\eta + 1$ and κ are the timestamps of the lidar sweep next to the starting one and the current lidar sweep in the window, with the covariance matrix $C_{L_\beta}^\chi$ determined by the lidar accuracy [35]. $u_\gamma(Z_{\zeta+1}^\zeta, \mathbf{X})$ is the residual of the IMU constraints, where

$$u_\gamma(Z_{\zeta+1}^\zeta, \mathbf{X}) = \begin{bmatrix} R_\zeta^T \left(p_{\zeta+1} - p_\zeta - v_\zeta \Delta t - \frac{1}{2} g \Delta t^2 \right) - \Delta p_{\zeta, \zeta+1} \\ R_\zeta^T (v_{\zeta+1} - v_\zeta - g \Delta t) - \Delta v_{\zeta, \zeta+1} \\ 2 \left[\Delta q_{\zeta, \zeta+1}^{-1} \otimes q_\zeta^{-1} \otimes q_{\zeta+1} \right]_{xyz} \\ b_{\zeta+1}^a - b_\zeta^a \\ b_{\zeta+1}^\omega - b_\zeta^\omega \end{bmatrix} \quad (9)$$

$u_\gamma(Z_{\zeta+1}^\zeta, \mathbf{X})$ can be obtained by IMU state prediction (Equation (2)) and IMU pre-integration (Equation (3)), and $\left[\Delta q_{\zeta, \zeta+1}^{-1} \otimes q_\zeta^{-1} \otimes q_{\zeta+1} \right]_{xyz}$ stands for the vector part of a quaternion. With the continuous-time linearized propagation of the error states and the IMU noise parameters, the covariances $C_{I_{\zeta+1}}^{I_\zeta}$ of the pre-integration measurements and biases can be estimated. The cost function, in the form of a non-linear least-square, can be solved by the Gauss–Newton algorithm. Ceres Solver [36] is used for solving this nonlinear problem.

It can be seen from Section 3.2 that the new states \mathbf{X} are obtained by the joint optimization, which is used as the next state of the IMU, avoiding the drift from IMU propagation. The state of the IMU is applied to deskewing, thereby eliminating the motion distortion of the dual lidar.

4. Experiment

4.1. Data Acquisition Equipment

To verify the performance of the proposed high-precision SLAM framework based on the tight coupling of Dual Lidar Inertial Odometry (HSDLIO) for multi-scene applications, this paper describes a series of experiments to qualitatively and quantitatively analyze our proposed framework. The device runs on an industrial computer with a processor of i7-9700 through the port and network port, and the system configured of the industrial computer is ubuntu16.04. The scanning arm is installed on the backpack, with the industrial computer and battery installed in the backpack. The small display screen of the backpack shows the localization and mapping in real-time, with the backpack equipment shown in Figure 6. Data of three representative environments are collected, including the featureless corridor, stairs with height, and complex outdoor environments. To illustrate the effectiveness of the HSDLIO algorithm, LeGO-LOAM and LIO-SAM algorithms are compared with it.



Figure 6. Backpack collection equipment.

4.2. Indoor Experiment 1

Indoor experiment 1 was carried out in a corridor with few feature points. Since there was no GPS signal indoors, SLAM was mainly carried out through the tight coupling of dual lidar odometry and IMU.

To highlight the performance of the HSDLIO algorithm, LeGO-LOAM and LIO-SAM algorithms are compared with it. Figure 7a shows the environment map of the corridor, while Figure 7b–d shows the overall mapping of LeGO-LOAM, LIO-SAM, and HSDLIO algorithms in the corridor scene, with the white points representing backpack motion trajectories. Compared with the HSDLIO algorithm in Figure 7d, the overall mapping of LeGO-LOAM (Figure 7b) has a significant deviation and the structure is unclear. Figure 8a,b shows that the mapping of LeGO-LOAM is incomplete because the lidar has a considerable drift in a scene with fewer feature points. The mapping results of LIO-SAM (Figure 7c) is close to the real environment, with its structure complete. The main reason is that LIO-SAM tightly couples the horizontal lidar and IMU to make up for the lidar drift. However, Figure 9a,b indicates the mapping result of HSDLIO is more abundant than LIO-SAM in the top surface and ground information. The reason is that HSDLIO fuses horizontal and vertical lidars so that its angle of FOV in both horizontal and vertical directions is 360° . The dual lidars and IMU are tightly coupled to make up for the lidar drift.

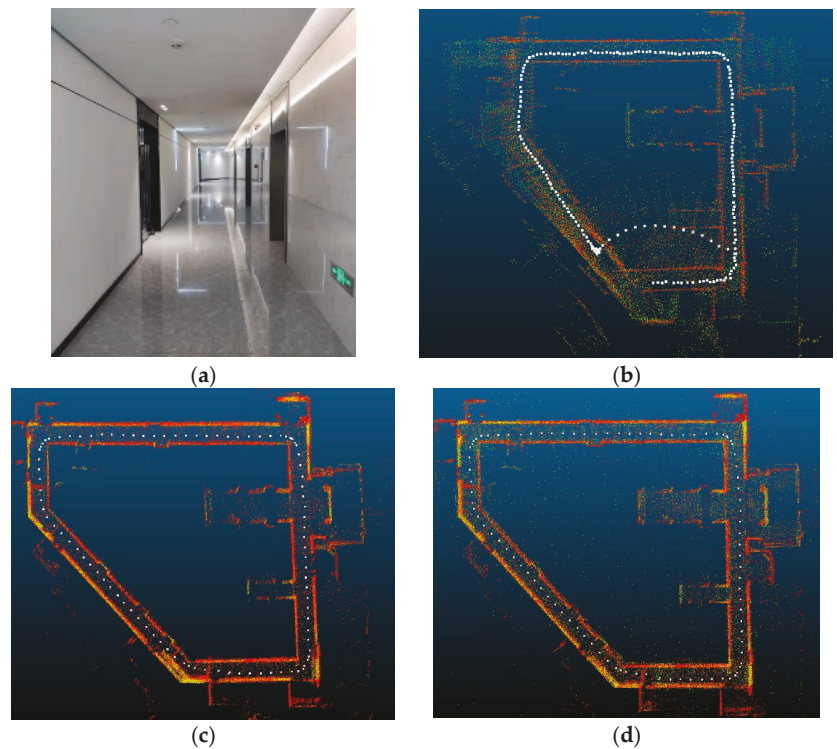


Figure 7. (a) Corridor scene for experiment 1. (b) Mapping results of LeGO-LOAM in the corridor scene, with the mapping of LeGO-LOAM having a significant deviation and the structure incomplete. (c) Mapping results of LIO-SAM in the corridor scene, with LIO-SAM having a complete mapping structure but lacking information on the top and ground. (d) Mapping results of HSDLIO in the corridor scene, with HSDLIO having a complete mapping structure.

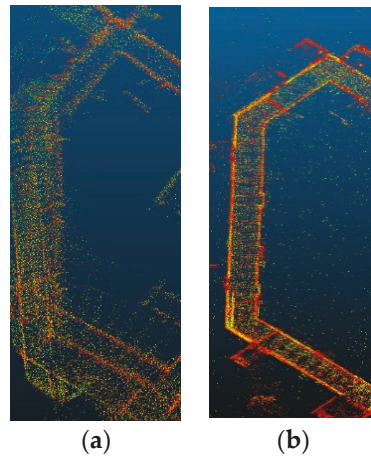


Figure 8. Comparison of mapping details at the corner of the corridor, where (a) the mapping of LeGO-LOAM has a significant deviation and (b) the mapping of the HSDLIO structure is clearer and more complete than LeGO-LOAM.

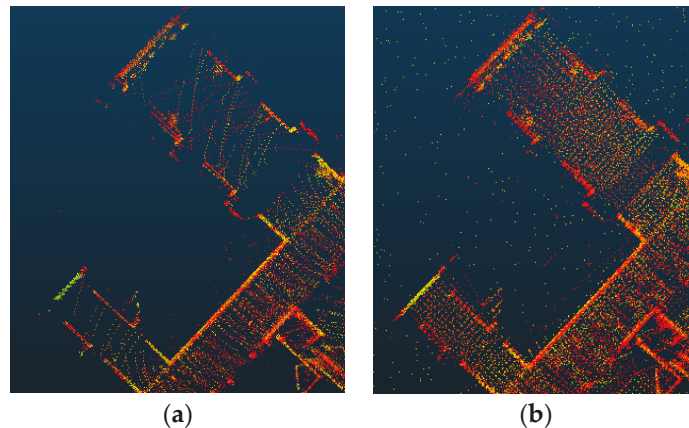


Figure 9. Comparison of mapping details at the top of the corridor, where (a) the mapping of LIO-SAM lacks top information and (b) the mapping of HSDLIO shows finer structural details of the environment.

In the long corridor scene, compared with LeGO-LOAM and LIO-SAM, the structure of HSDLIO mapping is more complete and the point cloud information on the ground and top surface is richer. To verify the positioning accuracy of the HSDLIO, the trajectories of LeGO-LOAM, LIO-SAM, and HSDLIO are compared in Figure 10. It can be observed in Figure 10 that the LeGO-LOAM trajectory has drifted. The main reason is that in the LeGO-LOAM algorithm, the lidar odometry error continues to increase in a long corridor environment with fewer feature points. Both LIO-SAM and HSDLIO use the tight coupling of IMU and lidar for positioning, which improves positioning accuracy. HSDLIO further improves the accuracy of positioning through the tight coupling of dual lidars and IMU.

To further improve the positioning accuracy of HSDLIO, the relative translational error (RTE)—the distance from the start point to the end point—is introduced. In all experiments, the data collection started and ended at the same point; when the positioning accuracy of

the device is very high, the start and end point will coincide. The RTE, when the backpack returns to the start point, is shown in Table 1. The RTE can be computed by Equation (10):

$$RTE = \sqrt{x_o^2 + y_o^2 + z_o^2} \tag{10}$$

where $x_o = x_s - x_e$, $y_o = y_s - y_e$, $z_o = z_s - z_e$, x_s , y_s , and z_s are the coordinates of the starting point and x_e , y_e , and z_e are the coordinates of the end point. The result of LeGO-LOAM is not shown because its trajectory has severely drifted. Table 1 shows that the error of HSDLIO in the x and y direction is similar to LIO-SAM, but the accuracy in the z direction is higher than LIO-SAM and the RTE of HSDLIO is smaller than LIO-SAM.

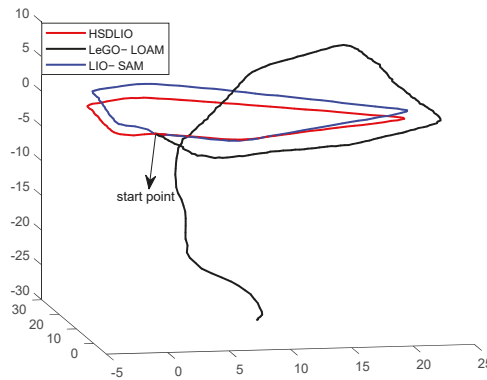


Figure 10. Comparison of LeGO-LOAM, LIO-SAM, and HSDLIO trajectories. The red line is the trajectory of HSDLIO, the blue line is the trajectory of LIO-SAM, and the black line is the trajectory of LeGO-LOAM, with the latter having drifted.

Table 1. Relative translational error when the backpack returns to the starting point (meters).

Algorithm	LeGO-LOAM	LIO-SAM	HSDLIO
x_o	Fail	0.079	0.077
y_o	Fail	0.092	0.090
z_o	Fail	0.121	0.001
RTE	Fail	0.171	0.118

4.3. Indoor Experiment 2

Indoor experiment 2 aimed to show the effectiveness of HSDLIO in the indoor environment with a certain height information. In this experiment, the backpack was carried up and down four flights of stairs. Figure 11a shows the stairs scene, while Figure 11b–d shows the overall mapping of LeGO-LOAM, LIO-SAM, and HSDLIO algorithms. The white points are backpack localization trajectories. LeGO-LOAM (Figure 11b) cannot obtain the structure of the stairs. Because the vertical FOV of the horizontal lidar is relatively narrow, the mapping result of LIO-SAM (Figure 11c) does not show the height information. With the help of the vertical lidar and the tight coupling of the IMU, the mapping of HSDLIO (Figure 11d) is the most complete and accurate, which is closer to the real stairs’ scene.

Figure 12 shows the trajectories obtained by LeGO-LOAM, LIO-SAM, and HSDLIO. The red, black, and green lines represent the trajectories of HSDLIO, LeGO-LOAM, and LIO-SAM, respectively, with the trajectory of HSDLIO approximating the real trajectory. LeGO-LOAM has no IMU correction, which causes the drift of the trajectory. Although the trajectory of LIO-SAM does not drift, due to the narrow vertical FOV of the horizontal lidar, a significant error occurs in the z direction information. Comparing the height

information of LIO-SAM and HSDLIO in the z direction in Figure 13, it can be seen that when going up and down four floors of stairs, the localization results of LIO-SAM provide no height information and also indicate a great drift, while HSDLIO provides accurate height information. Therefore, the tight coupling of dual lidars and IMU makes the advantages of HSDLIO clearly apparent. Table 2 shows that the error of HSDLIO in the x and y direction is smaller than LIO-SAM, but the accuracy in the z direction is much higher than LIO-SAM and the RTE of HSDLIO is much smaller than LIO-SAM.

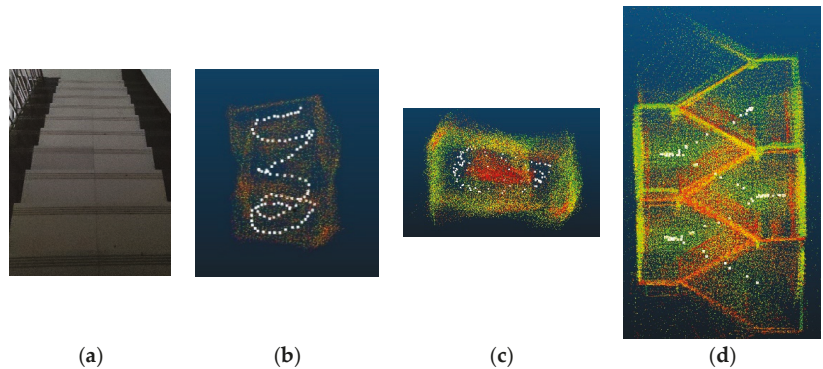


Figure 11. (a) Stairs scene for experiment 2. (b) Mapping results of LeGO-LOAM in the stairs scene, with LeGO-LOAM's mapping result having failed. (c) Mapping results of LIO-SAM in the stairs scene, with LIO-SAM's mapping results lacking height information. (d) Mapping results of HSDLIO in the stairs scene, with the mapping of HSDLIO the most complete and accurate.

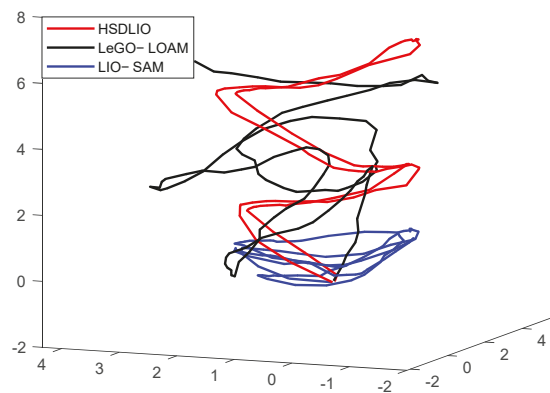


Figure 12. Comparison of LeGO-LOAM, LIO-SAM, and HSDLIO trajectories. The red line is the trajectory of HSDLIO, the blue line is the trajectory of LIO-SAM, and the black line is the trajectory of LeGO-LOAM. While LeGO-LOAM's trajectory drifted and LIO-SAM's trajectory produced cumulative errors in the z direction, the trajectory of HSDLIO approximated the real trajectory.

4.4. Outdoor Experiment

The outdoor experiment was carried out on a city street. This scene has rich feature points, so the drift of lidar odometry grows very slowly. The localization accuracy of LeGO-LOAM, LIO-SAM, and HSDLIO have very little difference, but the mapping of HSDLIO can provide finer structural details.

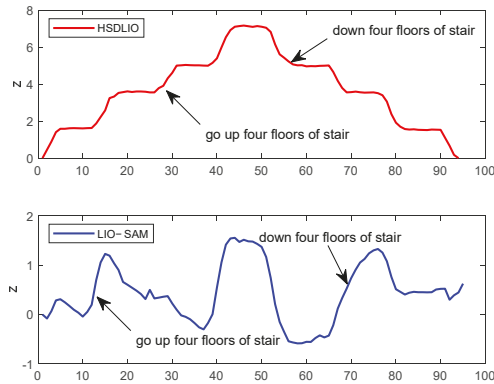


Figure 13. Comparison of the height information of LIO-SAM and HSDLIO in the z direction. The trajectory of HSDLIO provides height information from going up and down four floors of stairs, while LIO-SAM failed to position in the z direction.

Table 2. Relative translational error when the backpack returns to the starting point (meters).

Algorithm	LeGO-LOAM	LIO-SAM	HSDLIO
x_o	Fail	0.033	0.011
y_o	Fail	0.036	0.024
z_o	Fail	0.062	0.004
RTE	Fail	0.078	0.026

The trajectories obtained by LeGO-LOAM, LIO-SAM, and HSDLIO are shown in Figure 14. Because there is more feature information outdoors, the drift of lidar odometry grows very slowly. LeGO-LOAM can obtain relatively accurate localization only through horizontal lidar. In LIO-SAM and HSDLIO, GPS measurement is introduced by loosely coupling with IMU through UKF, which can provide relatively accurate initial positioning. The localization accuracy of HSDLIO and LIO-SAM is higher than LeGO-LOAM. The available data in Table 3 further supports the performance of the HSDLIO algorithm.

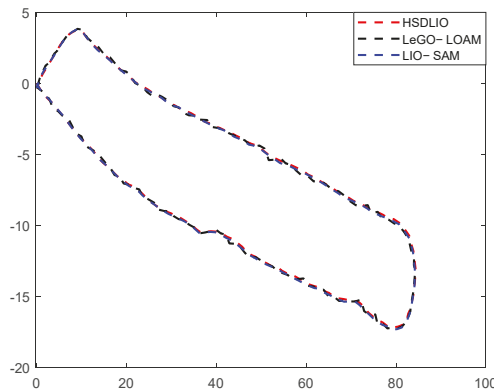


Figure 14. Comparison of LeGO-LOAM, LIO-SAM, and HSDLIO trajectories. The red dashed line is the trajectory of HSDLIO, the blue dashed line is the trajectory of LIO-SAM, and the black dashed line is the trajectory of LeGO-LOAM.

Table 3. Relative translational error when the backpack returns to the starting point (meters).

Algorithm	LeGO-LOAM	LIO-SAM	HSDLIO
RTE	0.082	0.036	0.031

The 3D city street scene is shown in Figure 15a. In Figure 15a–d, the marks in the white circle represent trees and the marks in the yellow circle represent the building structure. Figure 15b–d shows the overall mapping of LeGO-LOAM, LIO-SAM, and HSDLIO algorithms in the city street scene, with the white points being their localization trajectories. Figure 15b,c shows that the overall effect of the mapping lacks building height information, while the structure of buildings and trees is blurred. The mapping details are analyzed at the marker, compared with the mapping results of LeGO-LOAM and LIO-SAM (Figure 15b,c), while the HSDLIO mapping result is shown in Figure 15d. It can be seen in Figure 15d that the structure of the trees and buildings is complete and has finer details.

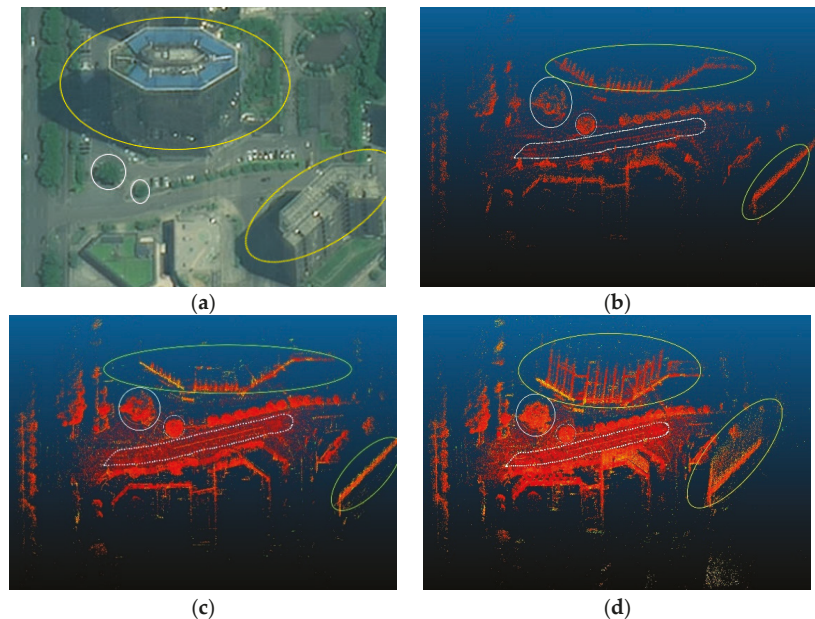


Figure 15. (a) City street scene, with the marks in the white circle representing trees and the marks in the yellow circle representing the building structure. (b) The mapping results of LeGO-LOAM in the city street scene provide a lack of building height information and the structure of buildings and trees is blurred. (c) The mapping results of LIO-SAM in the city street scene provide a lack of building height information, though the mapping of trees is relatively clear. (d) The mapping results of HSDLIO in the city street scene show that the point cloud features of the trees and the building structure are finer and more complete, including finer structural details.

To further demonstrate the advantages of HSDLIO mapping, the mapping details of the building are shown in Figure 16, with the mapping results of LeGO-LOAM and LIO-SAM lacking the structural details and height information of the building. However, the mapping result of HSDLIO show finer structural details and more height information of the building.

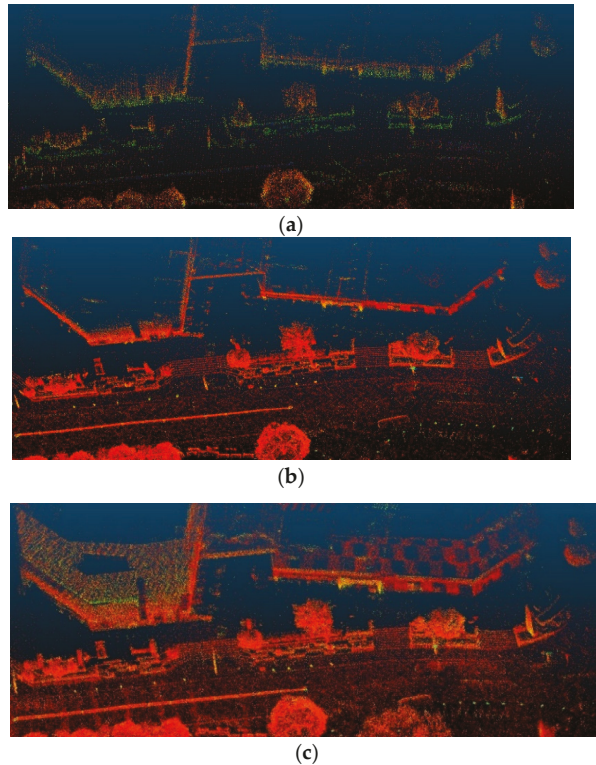


Figure 16. Comparison of mapping details in the city street scene showing that (a) the mapping of LeGO-LOAM lacks structural details of the building, (b) the mapping of LIO-SAM lacks the height of the building structure, and (c) the mapping of HSDLIO provides finer structural details of the building.

5. Conclusions

This paper proposes a high-precision SLAM framework for multi-scene applications. In this framework, dual lidars are fused to make up for the shortcomings of lidars' narrow FOV and hence improve the completeness of mapping. Meanwhile, dual lidars and IMU are tightly coupled to improve the localization accuracy. Extensive experiments were carried out and the results showed that compared with the commonly used LeGO-LOAM and LIO-SAM methods, our proposed method can produce more precise localization and more accurate mapping results with more details.

Author Contributions: Conceptualization, K.X.; methodology, K.X. and W.Y.; software, K.X., W.Y., W.L. and F.Q.; validation, W.Y. and Z.M.; formal analysis, K.X. and W.L.; investigation, W.L. and K.X.; resources, K.X. and W.Y.; data curation, K.X. and W.Y.; writing—original draft, K.X.; writing—review and editing, K.X. and W.Y.; visualization, K.X.; supervision, K.X. and Z.M.; project administration, K.X. and W.Y.; funding acquisition, K.X. and W.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation (grant nos. 61602529 and 61672539). This work was also supported by Hunan Key Laboratory of Intelligent Logistics Technology (2019TP1015) and Scientific Research Project of Hunan Education Department (No. 17C1650).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dissanayake, M.W.M.G.; Newman, P.; Clark, S.; Durrant-Whyte, H.F.; Csorba, M. A solution to the simultaneous localization and map building (SLAM) problem. *IEEE Trans. Robot. Autom.* **2001**, *17*, 229–241. [[CrossRef](#)]
2. Durrant-Whyte, H.; Bailey, T. Simultaneous localization and mapping: Part I. *IEEE Robot. Autom. Mag.* **2006**, *13*, 99–110. [[CrossRef](#)]
3. Khairuddin, A.R.; Talib, M.S.; Haron, H. Review on Simultaneous Localization and Mapping (SLAM). In Proceedings of the IEEE International Conference on Control System, Computing and Engineering (ICCSCE), George Town, Malaysia, 25–27 November 2016; pp. 85–90.
4. Belter, D.; Nowicki, M.; Skrzypczyński, P. Lightweight RGB-D SLAM System for Search and Rescue Robots. In *Progress in Automation, Robotics and Measuring Techniques*; Springer: Cham, Switzerland, 2015; Volume 2, pp. 11–21.
5. Sim, R.; Roy, N. Global A-Optimal Robot Exploration in Slam. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; pp. 661–666.
6. Cheng, Z.; Wang, G. Real-Time RGB-D SLAM with Points and Lines. In Proceedings of the 2018 2nd IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), Xi'an, China, 25–27 May 2018; pp. 119–122.
7. Liu, T.; Zhang, X.; Wei, Z.; Yuan, Z. A robust fusion method for RGB-D SLAM. In Proceedings of the 2013 Chinese Automation Congress, Changsha, China, 7–8 November 2013; pp. 474–481.
8. Hess, W.; Kohler, D.; Rapp, H.; Andor, D. Real-Time Loop Closure in 2D LIDAR SLAM. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–20 May 2016; pp. 1271–1278.
9. Yu, Y.; Gao, W.; Liu, C. A GPS-aided Omnidirectional Visual-Inertial State Estimator in Ubiquitous Environments. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 7750–7755.
10. Wang, Z.; Zhang, J.; Chen, S. Robust High Accuracy Visual-Inertial-Lidar SLAM System. In Proceedings of the 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Macau, China, 3–8 November 2019; pp. 6636–6641.
11. Mur-Artal, R.; Tardós, J.D. ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo, and RGB-D Cameras. *IEEE Trans. Robot.* **2017**, *33*, 1255–1262. [[CrossRef](#)]
12. Zhou, Y.; Li, H.; Kneip, L. Canny-VO: Visual Odometry with RGB-D Cameras Based on Geometric 3-D–2-D Edge Alignment. *IEEE Trans. Robot.* **2019**, *35*, 184–199. [[CrossRef](#)]
13. Hu, G.; Huang, S.; Zhao, L.; Alempijevic, A.; Dissanayake, G. A robust RGB-D SLAM algorithm. In Proceedings of the 2012 IEEE/RSJ International Conference on Intelligent Robots and Systems, Vilamoura-Algarve, Portugal, 7–12 October 2012; pp. 1714–1719.
14. Zhang, J.; Singh, S. Loam: Lidar odometry and mapping in real-time. In Proceedings of the Robotics: Science and Systems X, Berkeley, CA, USA, 12–16 July 2014.
15. Tsardoulas, E.; Petrou, L. Critical rays scan match SLAM. *J. Intell. Robot. Syst.* **2013**, *72*, 441–462. [[CrossRef](#)]
16. Jiang, B.; Zhu, Y.; Liu, M. A Triangle Feature Based Map-to-map Matching and Loop Closure for 2D Graph SLAM. In Proceedings of the 2019 IEEE International Conference on Robotics and Biomimetics (ROBIO), Dali, China, 6–8 December 2019; pp. 2719–2725.
17. Shan, T.; Englot, B.J. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 4758–4765.
18. Bogoslavskyi, I.; Stachniss, C. Fast range image-based segmentation of sparse 3D lidar scans for online operation. In Proceedings of the 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Daejeon, Korea, 9–14 October 2016; pp. 163–169.
19. Ranganathan, A. The levenberg-marquardt algorithm. *Tutorial LM Algorithm* **2004**, *11*, 101–110.
20. Torres-Torriti, M.; Guesalaga, A. Scan-to-map matching using the Hausdorff distance for robust mobile robot localization. In Proceedings of the 2008 IEEE International Conference on Robotics and Automation (ICRA), Pasadena, CA, USA, 19–23 May 2008; pp. 455–460.
21. Hassani, A.; Morris, N.; Spenko, M. Experimental integrity evaluation of tightly-integrated IMU/LiDAR including return-light intensity data. In Proceedings of the 32nd International Technical Meeting of The Satellite Division of the Institute of Navigation (ION), Miami, FL, USA, 16–20 September 2019; pp. 2637–2658.
22. Velas, M.; Spanel, M.; Hradis, M. CNN for IMU assisted odometry estimation using velodyne LiDAR. In Proceedings of the 2018 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Torres Vedras, Portugal, 25–27 April 2018; pp. 71–77.
23. Deilamsalehy, H.; Havens, T.C. Sensor fused three-dimensional localization using IMU, camera and LiDAR. In Proceedings of the 2016 IEEE Sensors, Orlando, FL, USA, 30 October–3 November 2016; pp. 1–3.
24. Xie, G.; Zong, Q.; Zhang, X. Loosely-coupled lidar-inertial odometry and mapping in real time. *Int. J. Intell. Robot. Appl.* **2021**, *5*, 119–129. [[CrossRef](#)]

25. Moore, T.; Stouch, D. A generalized extended kalman filter implementation for the robot operating system. In *Intelligent Autonomous Systems 13*; Springer: Cham, Switzerland, 2016; pp. 335–348.
26. Ye, H.; Chen, Y.; Liu, M. Tightly coupled 3d lidar inertial odometry and mapping. In Proceedings of the 2019 IEEE International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 3144–3150.
27. Shan, T.; Englot, B.; Meyers, D. Lio-sam: Tightly-coupled lidar inertial odometry via smoothing and mapping. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 25–29 October 2020; pp. 5135–5142.
28. Wellington, C.; Stentz, A. Learning predictions of the load-bearing surface for autonomous rough-terrain navigation in vegetation. In *Springer Tracts in Advanced Robotics (STAR)*; Springer: Berlin/Heidelberg, Germany, 2003; Volume 24, pp. 83–92.
29. Bosse, M.; Zlot, R. Continuous 3D scan-matching with a spinning 2D laser. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation (ICRA), Kobe, Japan, 12–17 May 2009; pp. 4312–4319.
30. Bosse, M.; Zlot, R.; Flick, P. Zebedee: Design of a spring-mounted 3-d range sensor with application to mobile mapping. *IEEE Trans. Robot.* **2012**, *28*, 1104–1119. [[CrossRef](#)]
31. Palieri, M.; Morrell, B. Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time. *IEEE Robot. Autom. Lett.* **2020**, *6*, 421–428. [[CrossRef](#)]
32. Jiao, J.; Ye, H.; Zhu, Y.; Liu, M. Robust odometry and mapping for multi-lidar systems with online extrinsic calibration. *IEEE Trans. Robot.* **2021**, 1–10. [[CrossRef](#)]
33. Nguyen, T.M.; Yuan, S. MILIOM: Tightly Coupled Multi-Input Lidar-Inertia Odometry and Mapping. *IEEE Robot. Autom. Lett.* **2021**, *6*, 5573–5580. [[CrossRef](#)]
34. Forster, C.; Carlone, L.; Dellaert, F. On-Manifold Preintegration for Real-Time Visual-Inertial Odometry. *IEEE Trans. Robot. Autom.* **2017**, *33*, 1–21. [[CrossRef](#)]
35. Qin, T.; Li, P.; Shen, S. Vins-mono: A robust and versatile monocular visual-inertial state estimator. *IEEE Trans. Robot.* **2018**, *34*, 1004–1020. [[CrossRef](#)]
36. Ceres Solver. Available online: <http://ceres-solver.org> (accessed on 5 October 2021).

Article

A Method of Enhancing Rapidly-Exploring Random Tree Robot Path Planning Using Midpoint Interpolation

Jin-Gu Kang¹, Yong-Sik Choi² and Jin-Woo Jung^{1,*}

¹ Department of Computer Science and Engineering, Dongguk University, Seoul 04620, Korea; kanggu12@dongguk.edu

² Department of Artificial Intelligence, Dongguk University, Seoul 04620, Korea; sik2230@dongguk.edu

* Correspondence: jwjung@dongguk.edu; Tel.: +82-2-2260-3812

Abstract: It is difficult to guarantee optimality using the sampling-based rapidly-exploring random tree (RRT) method. To solve the problem, this paper proposes the post triangular processing of the midpoint interpolation method to minimize the planning time and shorten the path length of the sampling-based algorithm. The proposed method makes a path that is closer to the optimal path and somewhat solves the sharp path problem through the interpolation process. Experiments were conducted to verify the performance of the proposed method. Applying the method proposed in this paper to the RRT algorithm increases the efficiency of optimization by minimizing the planning time.

Keywords: robot path planning; RRT; midpoint interpolation; triangular rewiring; path smoothness

Citation: Kang, J.-G.; Choi, Y.-S.; Jung, J.-W. A Method of Enhancing Rapidly-Exploring Random Tree Robot Path Planning Using Midpoint Interpolation. *Appl. Sci.* **2021**, *11*, 8483. <https://doi.org/10.3390/app11188483>

Academic Editor: António Paulo Moreira

Received: 8 July 2021

Accepted: 9 September 2021

Published: 13 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Recent path planning research for the robot has encompassed a wide range of topics [1,2]. Path planning is an important capability for autonomous mobile robots. A robot must be able to identify a path from its current position to its destination in order to move successfully. A mobile robot must be able to discover an optimal or sub-optimal collision-free path in the environment from the starting position to the destination [3].

Path planning is the formulation of a route for a mobile robot to proceed from a starting point to a destination point in Euclidean space as efficiently as possible while avoiding both static and dynamic obstacles and maintaining optimality, clearance, and completeness [4]. An optimal path is one with the ideal path length, a clear path is one without obstacles for the mobile robot to collide with, and a complete path is one in which the robot can move from the start point to the destination point without colliding with obstacles.

Furthermore, it is indeed possible for the robot to be able to optimize its path by determining the quickest and safest path to its destination point in order to save time and energy. However, an algorithm that generates the optimal path increases the computation, and an algorithm that quickly generates a path does not guarantee the optimal path [5].

It is difficult to ensure optimality with the sampling-based rapidly-exploring random tree (RRT) algorithm [6]. As shown in Figure 1a, the RRT algorithm is a path planning algorithm that involves repeatedly adding a randomly sampled position as a child node in a tree with the starting point as the root node until the destination point is reached. The tree extends out in the shape of a stochastic fractal, as shown in Figure 1b, and has an algorithm used to locate the destination point.

The RRT algorithm and other sampling-based algorithms [7,8] offer the benefit of planning a path in a shorter time with less computing than traditional path planning algorithms like the visibility graph- [9], cell decomposition- [10], and potential field-based algorithms [11]. On the other hand, it does not ensure optimality and has the drawback of having probabilistically assured completeness. The latter is also known as probabilistic completeness [12], which implies that completeness is assured when the number of random samples is infinite but not always when the number of random samples is limited. The

goal of this research is to enhance the RRT algorithm, which ensures completeness and performs better than the related algorithms.

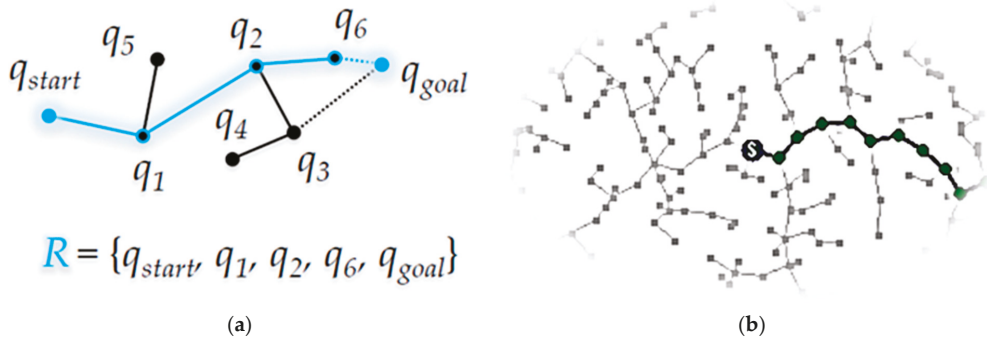


Figure 1. Overview of the rapidly-exploring random tree (RRT) algorithm: (a) planned path R from starting point q_{start} through waypoints q_1, q_2, q_6 to destination q_{goal} (q_i is a point on the path); (b) process of finding destination point by stochastic fractal shape from the root node (S) as a starting point.

To solve these problems, the main idea of the proposed post triangular processing of midpoint interpolation method is effective in path planning algorithms that do not guarantee optimality, such as the RRT algorithm that has a locally piecewise linear shape and can be used as a post-processing method after a path has been planned using one of these algorithms. It may also be used for different route planning methods since it is a post-processing technique that has no impact on calculation time.

The sampling-based path planning algorithm’s primary strength is the fast planning speed based on the small amount of computation compared to the traditional path planning algorithms [7].

Performance verification using simulation in various environments and mathematical modeling were used to validate the performance of the proposed method in this paper. The case in which the proposed algorithm is applied to the sampling-based path planning method and the case in which it is not applied are compared through simulation. Here, the planning time and path length of the first complete path to reach a destination point from a starting point are evaluated.

This paper is organized as follows: Section 2 reviews some related works. Section 3 introduces the proposed post triangular processing of the midpoint interpolation method. Various experimental environments are constructed for path planning in Section 4 to examine the effectiveness and improvements of the proposed method. Finally, the conclusions are presented in Section 5.

2. Related Works

In this section, we introduce the previous works about the RRT algorithm in Section 2.1 and the Triangular Rewiring Method for the RRT Algorithm in Section 2.2, respectively.

2.1. RRT

This section shows the pseudocode of the RRT algorithm used in the experiment of this paper that was designed based on [6] in which the RRT algorithm was proposed. In 1998, LaValle proposed the RRT algorithm, which is a representative sampling-based path planning algorithm [6]. It is designed to have many degrees of freedom and is useful for planning a path under non-holonomic constraints.

As shown in Figure 2, when a random sample is generated in the configuration space, the node nearest to the position of the random sample is identified among the nodes

constituting the tree with the starting point as the root node. A new node is generated at the random sample position and inserted into the tree if the random sample position is nearer than the step length. The process of tree extension is repeated until the destination point is reached. The RRT algorithm implemented for the proposed method and comparison experiment is Algorithm 1.

Algorithm 1 Pseudocode of RRT Algorithm

```

Input:
 $q_{start} \leftarrow$  start point
 $q_{goal} \leftarrow$  goal point
 $\lambda \leftarrow$  step length
 $C \leftarrow$  position set of all (measured) boundary points in all (known) obstacles
 $N \leftarrow$  number of random samples
Output:
 $R \leftarrow$  result of path  $R$ 
Initialize:
 $T \leftarrow$  Null tree<node, edge>
Procedure RRT
Begin
1    $T \leftarrow$  Insert root node< $q_{start}$ > to  $T$ 
2   While  $n \leftarrow 0$  To  $N$  Do
3     Generate  $n$ -th random sample
4      $q_{rand} \leftarrow$  position of  $n$ -th random sample
5      $q_{near} \leftarrow$  position of the nearest node in  $T$  from  $q_{rand}$ 
6     If not  $isInside(q_{near}, q_{rand}, \lambda)$  Then
7        $q_{new} \leftarrow$  intersection point between line segment connecting  $q_{rand}$  and  $q_{near}$ ,
       and circle with radius  $\lambda$  centered at  $q_{near}$ 
8     Else  $q_{new} \leftarrow q_{rand}$ 
9     If not  $isTrapped(q_{new}, q_{near}, C)$  Then
10     $T \leftarrow$  Insert node< $q_{new}$ > and edge< $q_{new}, q_{near}$ > to  $T$ 
11    If  $isInside(q_{new}, q_{goal}, \lambda)$  Then
12     $T \leftarrow$  Insert node< $q_{goal}$ > and edge< $q_{new}, q_{goal}$ > to  $T$ 
13     $P \leftarrow$  path from last inserted node( $q_{goal}$ ) to root node( $q_{start}$ ) in  $T$ 
14    If [length of  $R$ ] > [length of  $P$ ] Then  $R \leftarrow P$ 
15     $T \leftarrow$  Delete node< $q_{goal}$ > and edge< $q_{new}, q_{goal}$ > from  $T$ 
End

```

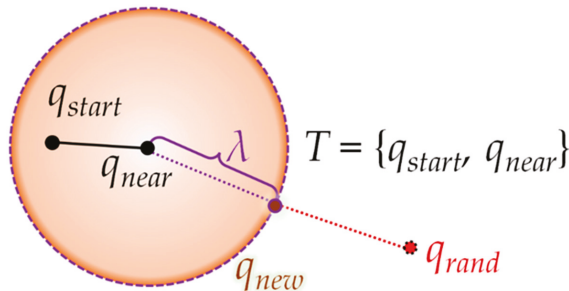


Figure 2. Process of the RRT algorithm: When creating the new node q_{new} at a position separated by step length λ in direction of random sample position (q_{rand}) based on q_{near} node (position) closest to random sample position (q_{rand}) in tree T with starting point q_{start} as the root node.

In order to overcome the limitations on optimality and convergence time [6], RRT-Connect can find a connected path more quickly by setting the start point and the destination point as the root of a separate tree, and further expanding the two trees alternately [7]. Rapidly-exploring Random Tree Star (RRT*) [13] was developed to overcome the limitation that the path generated from RRT does not guarantee convergence to the optimal path.

Informed-RRT* that can find a connected path quickly by enhancing the sampling probability inside the elliptical region with the start point and the destination point as the respective focal points [14]. The RRT*-Connect algorithm combines the advantages of RRT-Connect and RRT* [15]. RRT*-Smart [16], Quick-RRT*[17], and the proposed algorithm in [8] can show closer optimality by finding and connecting linearly connectable ancestor nodes to random samples through triangular inequality in the process of adding random samples.

2.2. Triangular Rewiring Method for the RRT Algorithm

This section shows the principle and pseudocode of the Triangular Rewiring Method for the RRT algorithm.

The triangular rewiring method is used to rewire the component based on the triangular inequality concept [8]. The triangular inequality-based RRT algorithm is a rewiring of the RRT method that is based on the concept of triangular inequality between nodes in path planning; thus, it is closer to the optimum than the RRT.

The triangular rewiring method not only can find a better initial solution but also can converge to a better solution rapidly.

The pseudocode for the triangular rewiring method is shown in Algorithm 2. This iterative process continues until no $q_{ancestor}$ exists (when no parent node exists for the previous $q_{ancestor}$, i.e., when $q_{ancestor}$ is q_{start}) or an obstacle exists between q_{child} and $q_{ancestor}$. The method for triangular rewiring is as follows: the node with the position q_{parent} and the edge connecting the q_{child} and $q_{ancestor}$ nodes are deleted. After the edge is deleted, the $q_{ancestor}$ node is updated with the q_{parent} node, and the parent node of the $q_{ancestor}$ is updated with the $q_{ancestor}$ node. The existing q_{parent} node is then deleted. Then, the Trapped method is used to check if it collides with an obstacle between the q_{child} node and the updated q_{parent} node. Then, in tree T , the last created q_{parent} is inserted as the parent node of q_{child} .

Algorithm 2 Pseudocode of Triangular Rewiring Method for RRT Algorithm

Input:

$q_{child} \leftarrow$ Position $\{q_{new} / q_{newA} / q_{newB}\}$

$q_{parent} \leftarrow$ Position q_{near}

$T \leftarrow$ Tree $\{T_{merged} / T_a / T_b\}$

$C \leftarrow$ Position Set C

Output:

$\{T_{merged} / T_a / T_b\} \leftarrow$ Result of T

Begin triangular Rewiring Procedure

1 $q_{ancestor} \leftarrow$ Position of Parent Node of q_{parent} in T

2 **If Not** *isTrapped*($q_{ancestor}$, q_{child} , C) **then**

3 $T \leftarrow$ **Delete** Node $\langle q_{parent} \rangle$, Edge $\langle q_{parent}$, $q_{child} \rangle$ and Edge $\langle q_{parent}$, $q_{ancestor} \rangle$ from T

4 $q_{parent} \leftarrow q_{ancestor}$

5 $q_{ancestor} \leftarrow$ Position of Parent Node of $q_{ancestor}$ in T

6 **While Not** $q_{ancestor} = \text{Null}$ **do**

7 **If Not** *isTrapped*($q_{ancestor}$, q_{child} , C) **then**

8 $T \leftarrow$ **Delete** Node $\langle q_{parent} \rangle$ and Edge $\langle q_{parent}$, $q_{ancestor} \rangle$ from T

9 $q_{parent} \leftarrow q_{ancestor}$

10 $q_{ancestor} \leftarrow$ Position of Parent Node of $q_{ancestor}$ in T

11 **Else**

12 **Break**

13 $T \leftarrow$ **Insert** Edge $\langle q_{parent}$, $q_{child} \rangle$ to T

14 **Else**

15 $T \leftarrow$ **Insert** Node $\langle q_{child} \rangle$ and Edge $\langle q_{child}$, $q_{parent} \rangle$ to T

End triangular Rewiring Procedure

3. Proposed Post Triangular Processing of the Midpoint Interpolation Method

The proposed post triangular processing of the midpoint interpolation method can be applied to path planning algorithms that do not guarantee optimality, such as the RRT algorithm, and rewiring and midpoint interpolation based on the triangular inequality principle.

The assumptions of the proposed method are as follows:

1. The destination point may change gradually over time, but there is only one start point and one destination point for each tree.
2. If the mobile robot cannot perform the omnidirectional motion, local planning or kinodynamic planning is performed separately on the path planning result.

The basic principle is that the node serving as a waypoint in the planned path checks whether an obstacle collides with its own grandparent node, and if it is free from obstacle collision, the grandparent node rewires to the parent node. If it is not free from obstacle collision, as shown in Figure 3, the locally piecewise linear path created between the node, its parent node, and its grandparent node is made a more optimal path through the interpolation process. In this process, a new node is interpolated into the path and deviates from the piecewise linear path, so it has the advantage of being able to somewhat solve the sharp path problem (the problem facing a mobile robot that has kinematic constraints because the slope is not smooth).

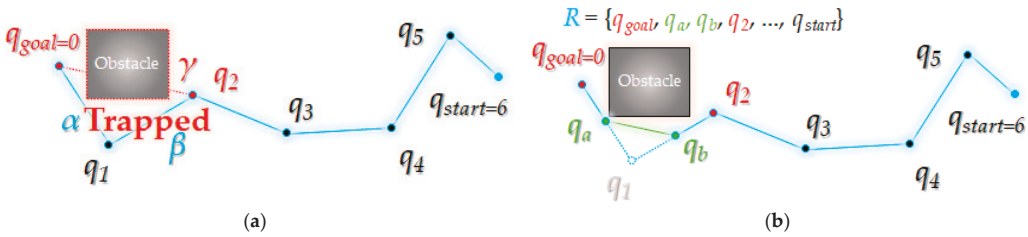


Figure 3. Summary of post triangular processing of the midpoint interpolation method: (a) line segment γ with node q_0 and its grandparent node q_2 in tree R is not free from obstacle collision; (b) Parent node q_1 of q_0 is deleted, and q_a between q_0 and q_1 , and q_b between q_1 and q_2 are inserted as waypoints of a new path between q_0 and q_2 .

This post triangular processing of the midpoint interpolation method was designed based on the polygon approximation algorithm [18,19], so, as shown in Figure 4, the path is calculated through a constant value called ϵ (the threshold of minimum clearance) ($\epsilon > 0$). It determines how closely the obstacle is approximated or, in other words, how close to the optimal path it is. Here, in Figure 4, d follows Equation (1):

$$d_n(q_i) = \begin{cases} (d_{n-1}(q_i))/2, & n > 0 \\ (2\sqrt{s(s-\alpha)(s-\beta)(s-\gamma)})/\gamma, & n = 0 \end{cases} \quad (s = (\alpha + \beta + \gamma)/2) \quad (1)$$

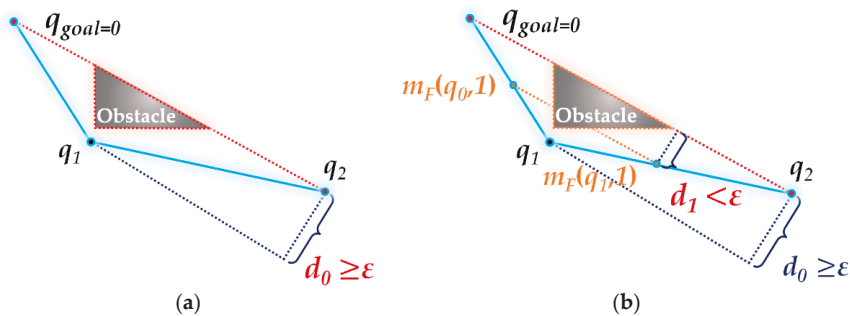


Figure 4. Interpolation function of post triangular processing of the midpoint interpolation method: (a) height d_0 of the triangle formed by waypoints q_0, q_1 , and q_2 of the path is greater than ϵ (interpolation continuation); (b) height d_1 of the triangle formed by midpoints $m_F(q_0,1)$ and q_1 of q_0 and q_1 and midpoint $m_F(q_1,1)$ is the midpoint of q_0 and q_1 , also $m_F(q_1,1)$ is the midpoint of q_1 and q_2 .

Here, $\zeta()$ is a function that receives the node as a variable and returns the parent node of that node. The n -squared ($n \geq 0$) of the $\zeta()$ function can be expressed as

$$\zeta^n(q_i) := \overbrace{(\zeta \circ \zeta \circ \dots \circ \zeta)}^n(q_i), \text{ and if } n \text{ is } 0, \zeta^0(q_i) := q_i \text{ holds.}$$

For the waypoint q_i , the value of d decreases by $1/2$ as interpolation proceeds (n). The initial value d_0 is the height of the triangle formed by the three line segments α , β , and γ ($\gamma < \alpha + \beta$), and the value of d_n becomes $(d_{n-1})/2$. Based on q_i , let α be the line segment from itself to the parent node, β be the line segment from the parent node to the grandparent node, and γ be the line segment from itself to the grandparent node. This d value serves as a measure to confirm the clearance of the obstacle compared to ϵ . This is because the smaller the d , the closer the path is to the obstacle.

Equation (1) converges to 0 when n becomes infinitely large in d_n , so epsilon receives only a value greater than 0 (positive number) from the user. ϵ is always greater than d_n when n is infinitely large. i.e., d is always smaller than epsilon at some point when n becomes infinitely large. This can be expressed as Equations (2) and (3) as follows:

$$\lim_{n \rightarrow \infty} d_n(q_i) = \lim_{n \rightarrow \infty} \frac{d_{n-1}(q_i)}{2} = \lim_{n \rightarrow \infty} \frac{d_0(q_i)}{2^n} = \lim_{n \rightarrow \infty} \frac{C}{2^n} = 0, \tag{2}$$

$$\therefore \lim_{n \rightarrow \infty} d_n(q_i) = 0, \epsilon > 0 \rightarrow \epsilon > \lim_{n \rightarrow \infty} d_n(q_i). \tag{3}$$

However, since optimality and clearance are opposite attributes, the closer ϵ is to 0 as shown in Figure 5, the more similar the path or path length is to the visibility graph, but it is not a smooth path [20]. The farther away it is from 0 (within a significant value where the path is modified), the farther it is from the optimum, but a smooth (kinetic) path is made, so ϵ should be set to a suitable value according to the environment.

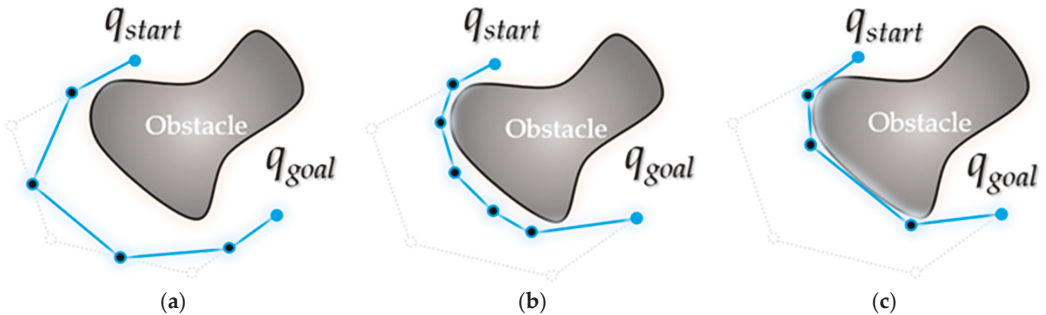


Figure 5. Variations according to ϵ values in post triangular processing of midpoint interpolation method (ϵ value of (a) $>$ ϵ value of (b) $>$ ϵ value of (c)): (a) When ϵ value is set relatively large; (b) When ϵ value is set relatively medium (smooth path); (c) When ϵ value is set relatively small (closer to the optimal path).

The following Algorithm 3 shows the pseudocode of the proposed post triangular processing of the midpoint interpolation method. It is mainly composed of the post triangular function (Algorithm 4) and interpolation function (Algorithm 5).

The input values of the post triangular processing of the midpoint interpolation method consist of the path R planned through a path planning algorithm, such as the RRT algorithm, the obstacle area information C , and the threshold value ϵ of the minimum clearance.

The f_{modify} is a variable that determines whether the input path R has been modified by this method, and if the path is modified even once, the entire process is repeated. If the path modification does not occur in the process of repeating, the algorithm is terminated. t refers to the index of the currently focused waypoint of R . That is, if t is 0, it is the starting point, which is the first point of R .

Algorithm 3 Pseudocode of Proposed Post Triangular Processing of Midpoint Interpolation Method**Input:** $R \leftarrow$ path from {RRT/ ... } $C \leftarrow$ position set of all (measured) boundary points in all (known) obstacles $\varepsilon \leftarrow$ threshold value of minimum clearance**Output:** $R \leftarrow$ modified path R **Initialize:** $f_{\text{modify}} \leftarrow \text{true}$ **Procedure** *postTriProcOfMidInterpolation***Begin**

```

1       While  $f_{\text{modify}}$  Do
2            $f_{\text{modify}} \leftarrow \text{false}$  // is the path modified
3            $t \leftarrow 0$  // index of the currently focused point
4            $q_{\text{child}} \leftarrow$  first point in  $R$ 
5            $q_{\text{parent}} \leftarrow$  next point of  $q_{\text{child}}$  in  $R$ 
6           While not [ $q_{\text{parent}}$  is the last point in  $R$ ] Do
7                $q_{\text{ancestor}} \leftarrow$  next point of  $q_{\text{parent}}$  in  $R$ 
8               If not isTrapped( $q_{\text{child}}, q_{\text{ancestor}}, C$ ) Then
9                    $R \leftarrow \text{postTriangular}(R, \varepsilon, t, f_{\text{modify}})$ 
10              Else
11                   $R \leftarrow \text{interpolation}(R, C, \varepsilon, t, f_{\text{modify}})$ 
12                   $q_{\text{child}} \leftarrow t$ -th point in  $R$ 
13                   $q_{\text{parent}} \leftarrow$  next point of  $q_{\text{child}}$  in  $R$ 
14
End

```

In R , when the first focusing point is q_{child} , the next point of that point q_{child} is q_{parent} , and the next point of that point q_{parent} is q_{ancestor} . It is determined whether the distance between q_{child} and q_{ancestor} is free from obstacle collision (*isTrapped()* function). If it is free from collision, it is called *postTriangular()*; otherwise, it is called *interpolation()*. *postTriangular()* connects q_{child} and q_{ancestor} as in the triangular rewiring method [8], and the q_{parent} between them is deleted from the path. *interpolation()* finds (interpolates) the points between q_{child} and q_{parent} and between q_{parent} and q_{ancestor} that are free from obstacle collision when connected and rewires q_{child} , q_{ancestor} , and those two points are found. If R and t are updated due to *postTriangular()* or *interpolation()*, q_{child} (the t -th waypoint of R), q_{parent} , and q_{ancestor} are updated accordingly. If q_{parent} is the last point in R , f_{modify} is checked. Otherwise, the above process is repeated for the updated q_{child} and q_{ancestor} .

Here, path modification by *postTriangular()* deletes the waypoints and makes a more optimal path but has the effect of sharpening the path shape, and path modification by *interpolation()* creates a new waypoint between the waypoints. Adding/inserting has the effect of making a more optimal path while also smoothing the path shape. Of course, due to the effect of making a more optimal path, path modification by *postTriangular()* is more efficient than that by *interpolation()*.

The input values of *postTriangular()* of the post triangular processing of the midpoint interpolation method consist of the path R , the focusing point index t , and the path modification f_{modify} from the post triangular processing of midpoint interpolation method.

Rewiring is performed on the t -th waypoint q_{child} of R , the next point q_{parent} , and the next point q_{ancestor} of that point again. First, the path between q_{child} and q_{parent} and the path between q_{parent} and q_{ancestor} are deleted. Then the path is inserted between q_{child} and q_{ancestor} . Finally, f_{modify} returns "true" because the path has been modified. Here, Path $\langle A, B \rangle$ means a partial path from Waypoint A to Waypoint B in the complete path.

Algorithm 4 Pseudocode of Proposed Post Triangular Function

```

Input:
R ← path R from postTriProcOfMidInterpolation
t ← point index t from postTriProcOfMidInterpolation
fmodify ← boolean fmodify from postTriProcOfMidInterpolation
Output:
R ← modified path R
fmodify ← result of boolean fmodify //return by reference
Procedure postTriangular From postTriProcOfMidInterpolation
Begin
1      qchild ← t-th point in R
2      qparent ← next point of qchild in R
3      qancestor ← next point of qparent in R
4      R ← Delete path<qchild, qparent> and path<qparent, qancestor> from R
5      R ← Insert path<qchild, qancestor> to R
6      fmodify ← true
End
    
```

The goal of the proposed post triangular processing of midpoint interpolation method is to find an interpolation point ($m_F(q_0)$, $m_F(q_1)$) free from obstacle collisions between waypoints ($q_0 \sim q_1$, $q_1 \sim q_2$) while descending in the direction of the midpoint (q_1), as shown in Figure 6 in the interpolation process.

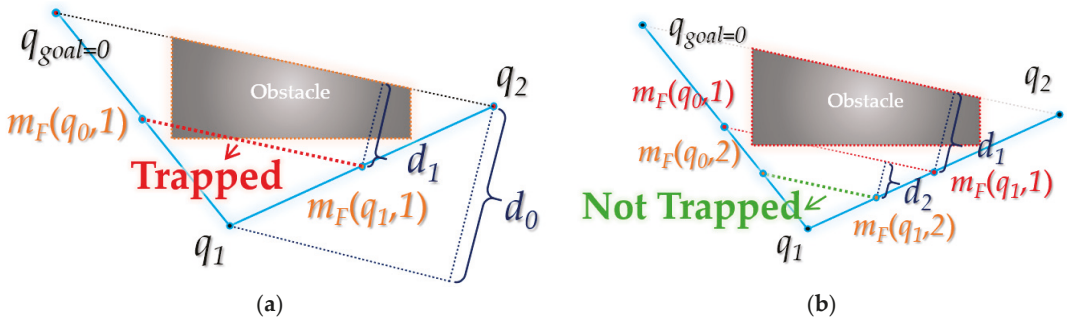


Figure 6. Details of post triangular processing of the midpoint interpolation method: (a) midpoint $m_F(q_0,1)$ of waypoint q_0 , q_1 of path and midpoint $m_F(q_1,1)$ of q_1 , q_2 are not free from obstacle collision; (b) midpoint $m_F(q_0,2)$ of interpolation point $m_F(q_0,1)$, q_1 and midpoint $m_F(q_1,2)$ between q_1 , interpolation point $m_F(q_1,1)$ is free from obstacle collision.

However, the interpolation point m_F follows Equation (4):

$$m_F(q_i, k) = \begin{cases} \left(\frac{m_F(q_i, k-1).x + \zeta(q_i).x}{2}, \frac{m_F(q_i, k-1).y + \zeta(q_i).y}{2} \right), & k > 0 \\ q_i, & k = 0 \end{cases} \quad (4)$$

When the k -th interpolation point of the interpolation point q_i is $m_F(q_i, k)$, the 0-th interpolation point becomes itself q_i , and the first interpolation point is the midpoint of q_i and the point $\zeta(q_i)$ next to q_i , and the second interpolation point becomes the midpoint of $m_F(q_i, 1)$ and $\zeta(q_i)$. That is, $m_F(q_i, k)$ ($k > 0$) becomes the midpoint between $m_F(q_i, k - 1)$ and $\zeta(q_i)$. The following Algorithm 5 shows the pseudocode of *interpolation()* of the proposed post triangular processing of the midpoint interpolation method.

Algorithm 5 Pseudocode of Proposed Interpolation Function

Input:
 $R \leftarrow$ path R from *postTriProcOfMidInterpolation*
 $C \leftarrow$ position set C from *postTriProcOfMidInterpolation*
 $\epsilon \leftarrow$ threshold value ϵ from *postTriProcOfMidInterpolation*
 $t \leftarrow$ point index t from *postTriProcOfMidInterpolation*
 $f_{modify} \leftarrow$ boolean f_{modify} from *postTriProcOfMidInterpolation*

Output:
 $R \leftarrow$ modified path R
 $t \leftarrow$ updated point index t // return by reference
 $f_{modify} \leftarrow$ result of boolean f_{modify} // return by reference

Initialize:
 $q_{child} \leftarrow t$ -th point in R
 $q_{parent} \leftarrow$ next point of q_{child} in R
 $q_{ancestor} \leftarrow$ next point of q_{parent} in R

Procedure interpolation **From** *postTriProcOfMidInterpolation*
Begin
1 $d \leftarrow$ altitude of the triangle consisting of q_{child} , q_{parent} , and $q_{ancestor}$ with base $\langle q_{child}, q_{ancestor} \rangle$
2 $m_a \leftarrow$ midpoint between q_{child} and q_{parent}
3 $m_b \leftarrow$ midpoint between q_{parent} and $q_{ancestor}$
4 **While true Do**
5 **If** $d \geq \epsilon$ **Then**
6 **If not** *isTrapped*(m_a, m_b, C) **Then**
7 $R \leftarrow$ **Delete** path $\langle q_{child}, q_{parent} \rangle$ and path $\langle q_{parent}, q_{ancestor} \rangle$ from R
8 $R \leftarrow$ **Insert** path $\langle q_{child}, m_a \rangle$, path $\langle m_a, m_b \rangle$, and path $\langle m_b, q_{ancestor} \rangle$ to R
9 $f_{modify} \leftarrow$ **true**
10 **Break**
11 **Else**
12 $d \leftarrow d / 2$
13 $m_a \leftarrow$ midpoint between m_a and q_{parent}
14 $m_b \leftarrow$ midpoint between m_b and q_{parent}
15 **Else**
16 $t \leftarrow t + 1$
17 **Break**
End

The input values of *interpolation*() of the post triangular processing of midpoint interpolation method consist of the path R , the obstacle area information C , the focusing point index t , and the path modification f_{modify} from the post triangular processing of the midpoint interpolation method.

From the t -th waypoint q_{child} of R , the next point q_{parent} , and the next point $q_{ancestor}$ of that point, the height d of the triangle is obtained, m_a is the midpoint of q_{child} and q_{parent} , and m_b is the midpoint of q_{parent} and $q_{ancestor}$. If the path between m_a and m_b is free from obstacle collision (*isTrapped*()), the path between q_{child} and q_{parent} is deleted, and the path between q_{child} and m_a , the path between m_a and m_b , and the path between m_b and $q_{ancestor}$ are inserted. Moreover, since the path is modified, f_{modify} returns "true," and the method terminates. If the line segment between m_a and m_b is not free from obstacles, the value of d decreases by $1/2$, m_a is updated to the midpoint of m_a and q_{parent} , and m_b is updated to the midpoint of m_b and q_{parent} . It is then determined whether m_a and m_b are free from obstacles again. This repeated process proceeds until a case is found in which m_a and m_b are free from obstacles or d becomes smaller than ϵ . If d becomes smaller than ϵ , the value of t is increased by 1 and the method is terminated.

Figure 7 shows the overall flowchart of the proposed post triangular processing of the midpoint interpolation method. Here, $\zeta^t(q_{goal})$ denotes the t -th next waypoint from the starting point q_{goal} of the path R , and $\zeta^{t+n}(q_{goal})$ is the n -th next waypoint in the $\zeta^t(q_{goal})$. That is, there are n waypoints between $\zeta^t(q_{goal})$ and $\zeta^{t+n}(q_{goal})$. In the flowchart shown in Figure 7, the stop condition follows the sequence:

1. Check whether the t -th ancestor node and the $(t + 2)$ -th ancestor node of q_{goal} are collision-free (Here, when t is 0, the 0-th ancestor means itself(q_{goal})).
2. If the result of Step 1 is not collision-free, compare the $d_k(\zeta^t(q_{goal}))$ value of the t -th ancestor node of q_{goal} with ϵ .
3. If $d_k(\zeta^t(q_{goal}))$ is less than ϵ , the value of t is incremented by 1, and if the parent node ($t + 1$) of the t -th ancestor node of q_{goal} after t is incremented is q_{start} , the algorithm is stopped (if f is False).

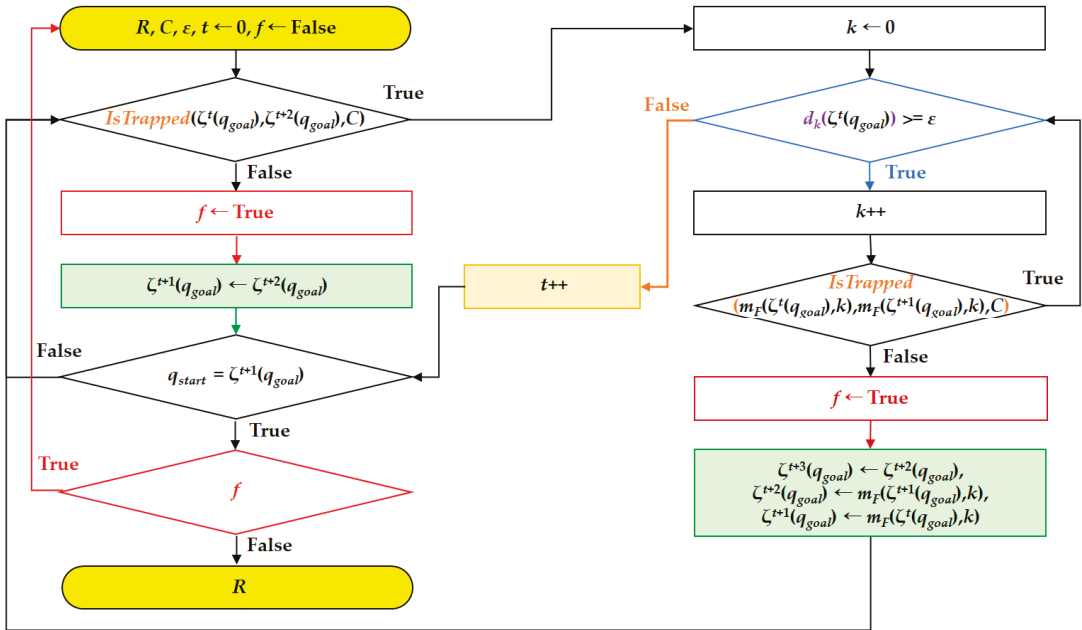


Figure 7. Flowchart of post triangular processing of midpoint interpolation method.

The stopping criterion of the proposed algorithm is based on ϵ . As shown in Figures 4 and 7, when the value of $d_k(\zeta^t(q_{goal}))$ becomes smaller than the ϵ , the algorithm stops the loop. As Equations (1) and (2) and Figure 6 show, the value $d_k(\zeta^t(q_{goal}))$ decreases deterministically.

4. Experimental Results

The path between the RRT in various environment maps through simulation and the RRT algorithm to which the proposed post triangular processing of the midpoint interpolation method is applied were used to validate the performance of the method proposed in this paper, and the path planning results were compared.

The performance measures compared were the average values after repeating the trial 100 times (sampling position was changed for each trial) of the path length (px) and the planning time (ms) of the first complete path (the first complete path to reach a destination point from a starting point).

Various environment maps have been examined and used to validate the performance of the proposed path planning algorithms in related works. Since the efficiency of the performance measures expected during the experiment varies somewhat based on the composition of obstacles (e.g., number, location, shape), it is important to choose which environment map to utilize carefully.

The four environment maps used in the experiment are shown in Figure 8. The four environment maps were created by partially referring to the experimental environment proposed by Han in 2017 [21]. All environment maps were 600×600 px in size, with a 30 px step length. The start points (S) are represented by green circles, while the destination points (G) are represented by purple circles. Obstacles are black polygons with yellow contours (blue in the experimental results).

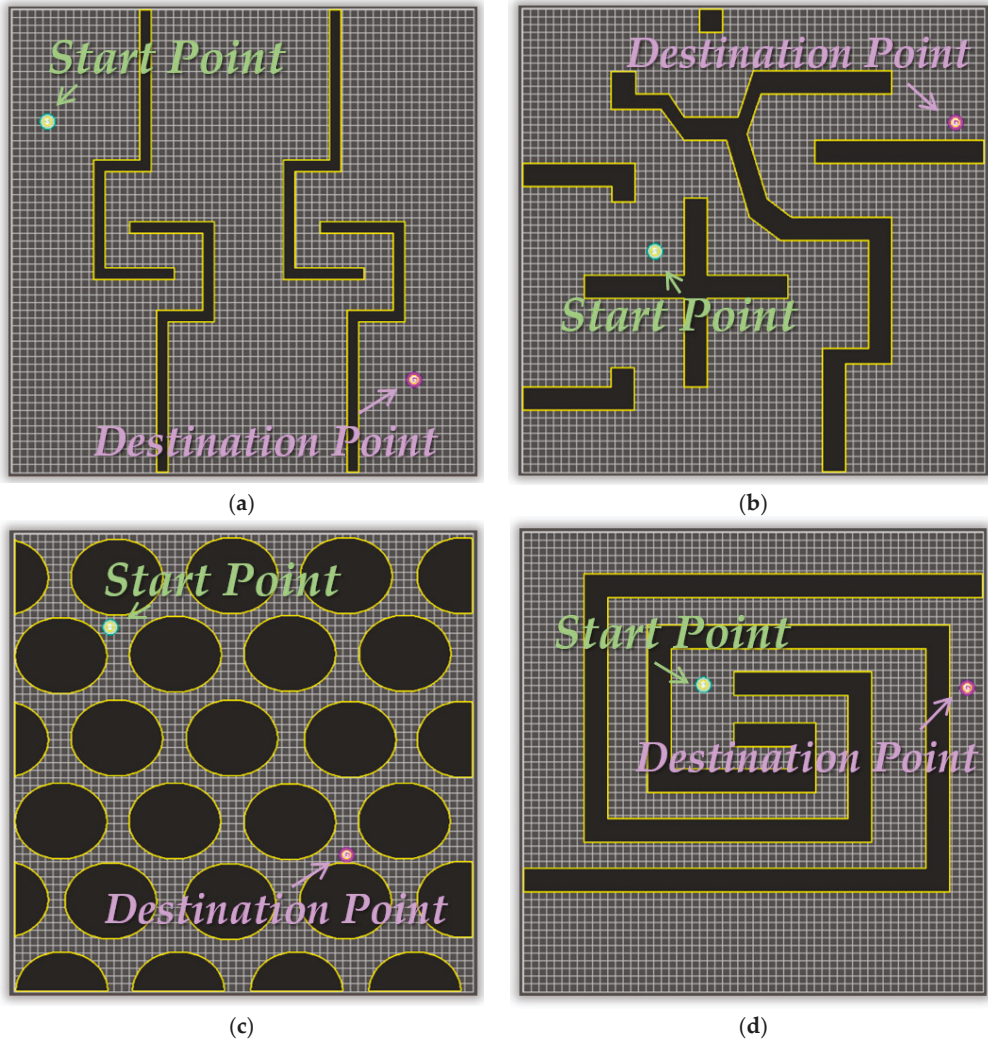


Figure 8. Environment maps for experiments: (a) Map 1; (b) Map 2; (c) Map 3; (d) Map 4.

Map 1 of Figure 8a appears to be an efficient environment for validating optimality and completeness but a weak environment for sampling-based path planning algorithms like the RRT algorithm. Many samplings are required since the probability of finding a solution is low. Map 2 of Figure 8b appears to be an efficient environment for validating the optimality and completeness of the path planning algorithm. Map 3 of Figure 8c is an environment that is efficient for validating the optimality and the planning time of the

path planning algorithm, as it consists of obstacles (50 vertices) that approximate circles. Map 4 of Figure 8d is an environment that is efficient for validating the optimality and the planning time of the path planning algorithm but a weak environment for sampling-based path planning algorithms such as the RRT algorithm.

The number of samples and planning time required increase drastically when the path to the destination point is narrow or there are few entrances since the sampling-based path planning algorithm depends on probabilistic completeness.

The specification of the computer used in the simulation is shown in Table 1. The simulator used for the simulation [8] was developed based on C# WPF (Microsoft Visual Studio Community 2019 Version 16.1.6 Microsoft .NET Framework Version 4.8.03752), and only a single thread was used for calculations except for the visual part. Generally, depending on the specification of the computer, the result of performance measurements, such as planning time, may vary during the simulation.

Table 1. Computer performance for simulation.

H/W	Specification
CPU	Intel Core i7-6700k 4.00 GHz (8 CPUs)
RAM	32,768 MB (32 GB DDR4)

We validate the experimental results (path length, planning time) in the four environment maps in which the post triangular processing of the midpoint interpolation method (ϵ : 50, 30, 10 px) is applied to the RRT algorithm and its path planning results. Since ϵ requires a higher amount of computation as it gets smaller, it was set to a value close to the 30 px step length of the experimental environment.

The experimental results for each map consist of a figure and table. The figure is a path planning result for each algorithm shown in the case of a single trial (the figure for each algorithm is not the result of repeated trials). The table shows an average value of the results of planning time and path length from the path planning repeated trials. There may be a significant difference between the performance observed visually and the numerical results in the table for one of the repeated trials. The form of the planned path for each algorithm is shown in the figure for visual reference. Furthermore, the proposed post triangular processing of the midpoint interpolation method is used to see whether there is a region where the piecewise linear path is smoothed.

The path planning results are shown in Figure 9 for Map 1 among the specified environment maps for each algorithm. In terms of appearance, the one to which the post triangular processing of the midpoint interpolation method (ϵ : 10 px) was applied seems to have a smoother slope compared to those of the other algorithms, and the path length with the method (ϵ : 10 px) seems to be the shortest.

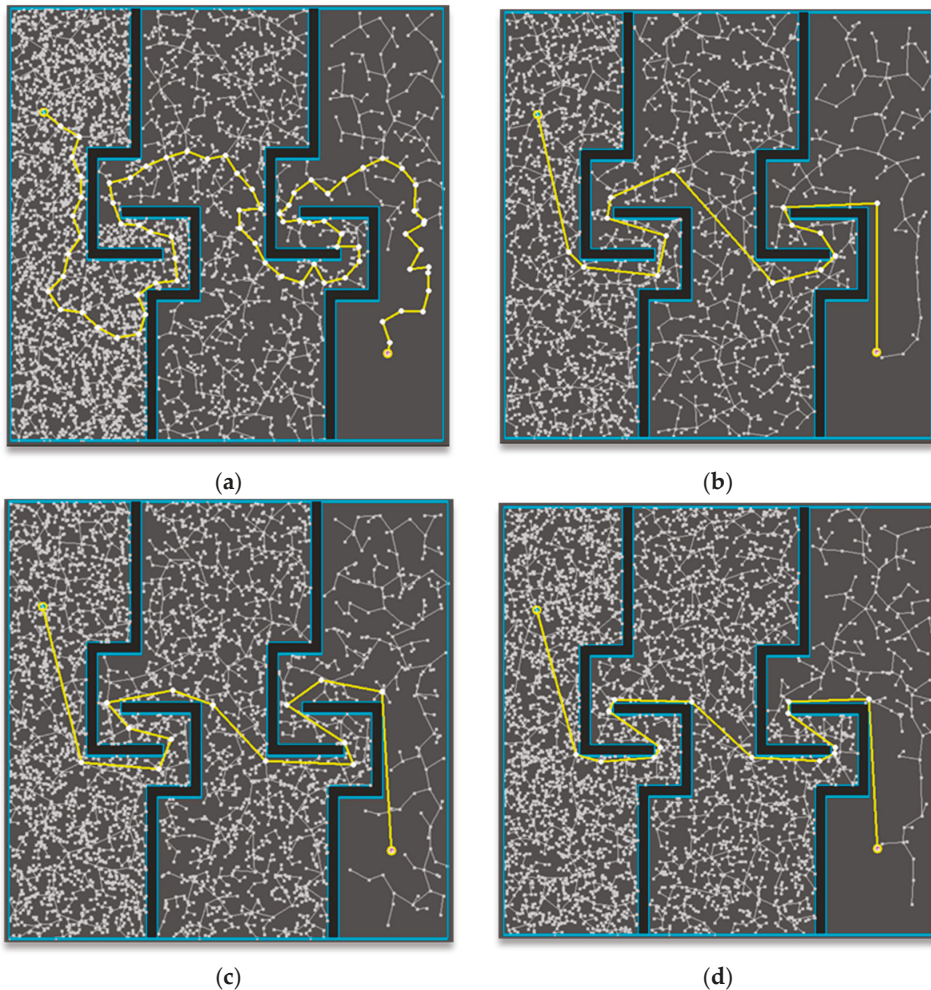


Figure 9. Experimental results of Map 1: (a) RRT; (b) proposed method (ϵ : 50 px) applied; (c) proposed method (ϵ : 30 px) applied; (d) proposed method (ϵ : 10 px) applied.

The path planning results (average values after repeating the trial 100 times) are shown in Table 2 for Map 1 among the specified environment maps for each algorithm. When applying the post triangular processing of the midpoint interpolation method (ϵ : 10 px), the path length becomes 62% (1224/1994(%)) compared to the RRT, which is the shortest of all the algorithms, and the planning times are all similar.

Table 2. Experimental results of Map 1 (numbers in parentheses (averages of repeating trial 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	RRT	Proposed Method (ϵ : 50 px)	Proposed Method (ϵ : 30 px)	Proposed Method (ϵ : 10 px)
Path length (px)	1944 (100%)	1403 (72%)	1325 (68%)	1224 (62%)
Planning time (ms)	697 (100%)	698 (100%)	698 (100%)	698 (100%)

The path planning results are shown in Figure 10 for Map 2 among the specified environment maps for each algorithm. In terms of appearance, the one to which the post triangular processing of midpoint interpolation method ($\epsilon: 10 \text{ px}$) was applied seems to have a smoother slope compared to those of the other algorithms, and the path length with the method ($\epsilon: 10 \text{ px}$) seems to be the shortest.

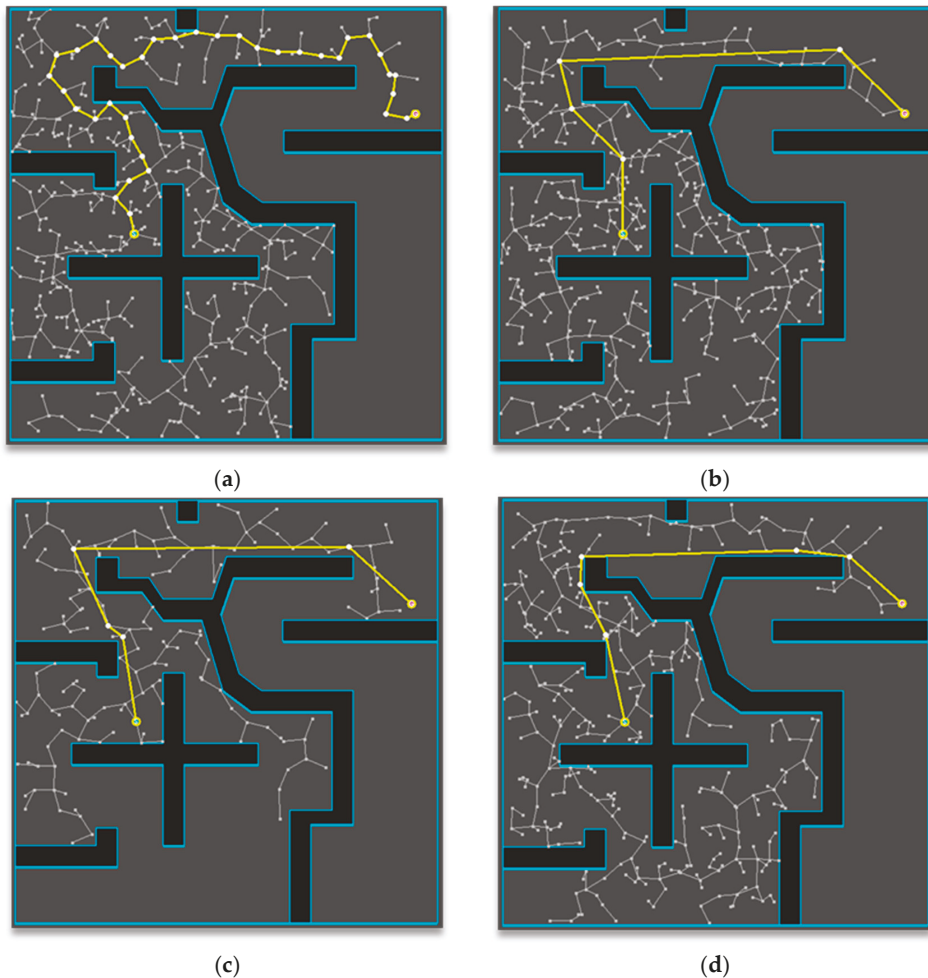


Figure 10. Experimental results of Map 2: (a) RRT; (b) Proposed method ($\epsilon: 50 \text{ px}$) applied; (c) Proposed method ($\epsilon: 30 \text{ px}$) applied; (d) Proposed method ($\epsilon: 10 \text{ px}$) applied.

The path planning results (average values after repeating the trial 100 times) are shown in Table 3 for Map 2 among the specified environment maps for each algorithm. By applying the post triangular processing of midpoint interpolation method ($\epsilon: 10 \text{ px}$), the path length becomes 74% (730/986(%)) compared to the RRT, which is the shortest of all the algorithms, and the planning time is similar to that of the RRT algorithm when the method ($\epsilon: 50 \text{ px}$) is applied. However, the absolute time difference is 1 ms, which seems to be large when the method is applied because it is an environment that requires less planning time.

Table 3. Experimental results of Map 2 (numbers in parentheses (averages of repeating trial 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	RRT	Proposed Method ($\epsilon: 50 \text{ px}$)	Proposed Method ($\epsilon: 30 \text{ px}$)	Proposed Method ($\epsilon: 10 \text{ px}$)
Path length (px)	986 (100%)	780 (79%)	752 (76%)	730 (74%)
Planning time (ms)	10 (100%)	10 (100%)	11 (110%)	11 (110%)

The path planning results are shown in Figure 11 for Map 3 among the specified environment maps for each algorithm. In terms of appearance, the one to which the post triangular processing of midpoint interpolation method ($\epsilon: 10 \text{ px}$) was applied seems to have a smoother slope compared to those of the other algorithms, and the path length with the method ($\epsilon: 10 \text{ px}$) seems to be the shortest.

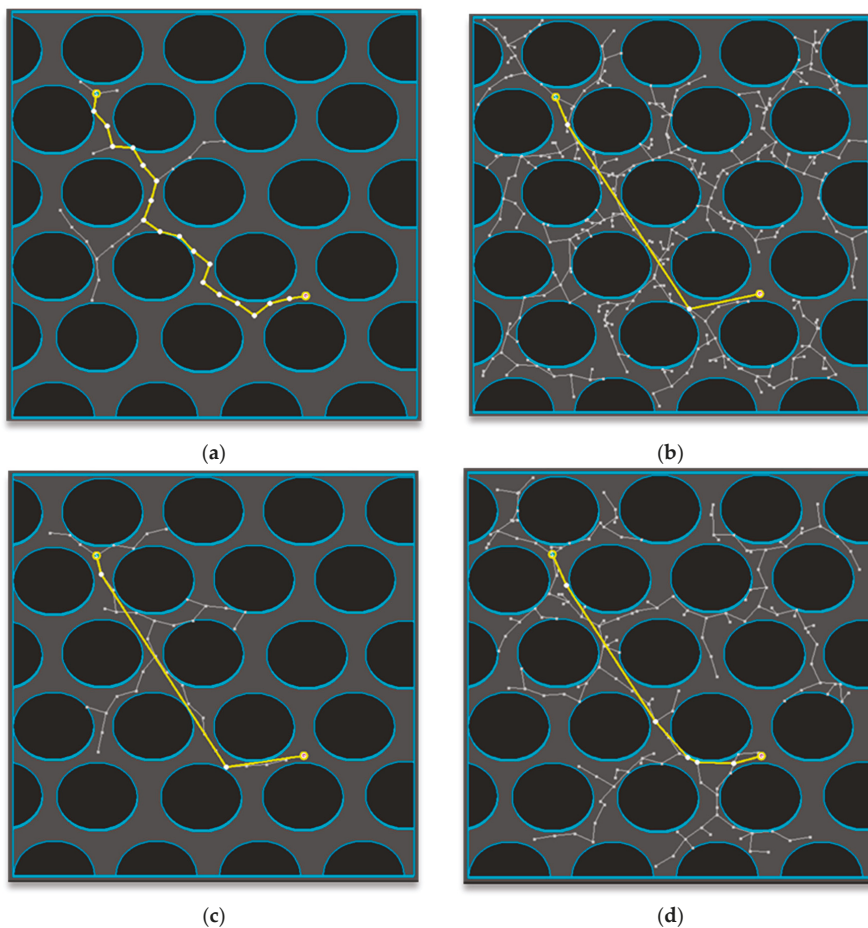


Figure 11. Experimental results of Map 3: (a) RRT; (b) Proposed method ($\epsilon: 50 \text{ px}$) applied; (c) Proposed method ($\epsilon: 30 \text{ px}$) applied; (d) Proposed method ($\epsilon: 10 \text{ px}$) applied.

The path planning results (average values after repeating the trial 100 times) are shown in Table 4 for Map 3 among the specified environment maps for each algorithm. By applying the post triangular processing of midpoint interpolation method ($\epsilon: 10 \text{ px}$), the path length becomes 82% (505/613(%)) compared to the RRT, which is the shortest of all

the algorithms, and the planning time is the most similar to that of the method ($\epsilon: 10 \text{ px}$), as it takes 16% more time than the RRT algorithm. However, the absolute time difference is 2 ms, and since it is an environment that requires less planning time, the difference appears to be large when the method is applied.

Table 4. Experimental results of Map 3 (numbers in parentheses (averages of repeating 100 times) are relative ratios to the RRT results (values less than 1 are counted as 1)).

Performance	RRT	Proposed Method ($\epsilon: 50 \text{ px}$)	Proposed Method ($\epsilon: 30 \text{ px}$)	Proposed Method ($\epsilon: 10 \text{ px}$)
Path length (px)	613 (100%)	535 (87%)	512 (83%)	505 (82%)
Planning time (ms)	6 (100%)	7 (116%)	8 (133%)	8 (133%)

The path planning results are shown in Figure 12 for Map 4 among the specified environment maps for each algorithm. In terms of appearance, the one to which the post triangular processing of the midpoint interpolation method ($\epsilon: 30 \text{ px}$) was applied seems to have a smoother slope compared to those of the other algorithms, and the path length with the method ($\epsilon: 10 \text{ px}$) seems to be the shortest.

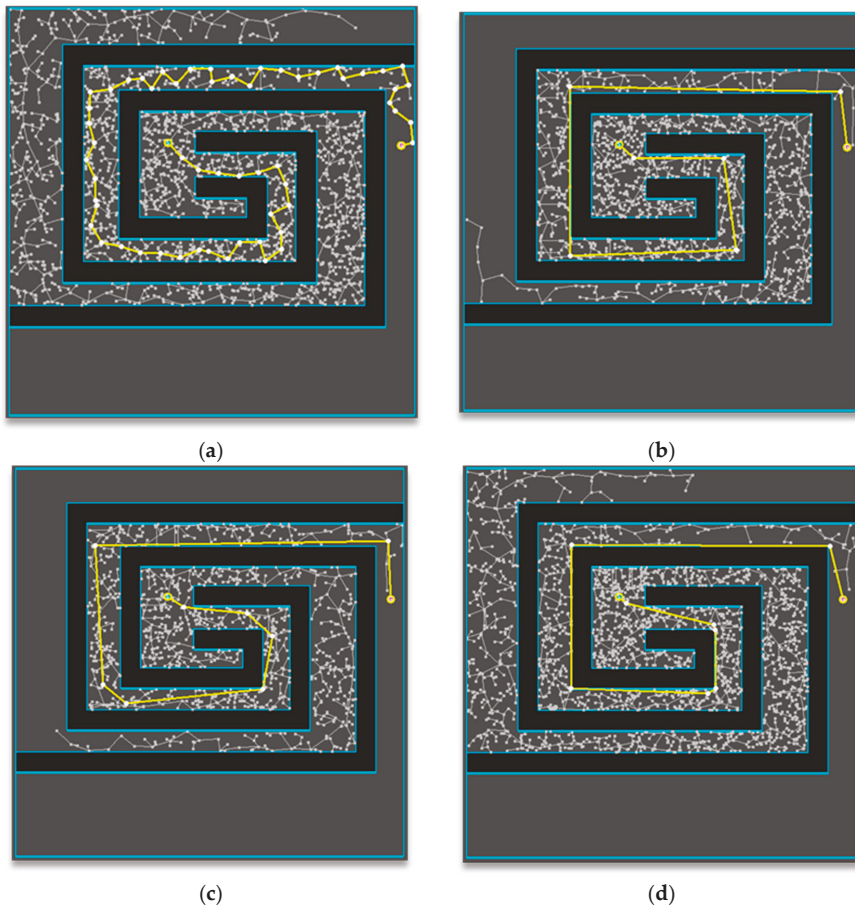


Figure 12. Experimental results of Map 4: (a) RRT; (b) Proposed method ($\epsilon: 50 \text{ px}$) applied; (c) Proposed method ($\epsilon: 30 \text{ px}$) applied; (d) Proposed method ($\epsilon: 10 \text{ px}$) applied.

The path planning results (average values after repeating the trial 100 times) are shown in Table 5 for Map 4 among the specified environment maps for each algorithm. By applying the post triangular processing of the midpoint interpolation method (ϵ : 10 px), the path length becomes 77% (1190/1536(%)) compared to the RRT, which is the shortest of all the algorithms, and all the planning times are similar.

Table 5. Experimental results of Map 4 (numbers in parentheses to (averages of repeating 100 times) are relative ratios to RRT results (values less than 1 are counted as 1)).

Performance	RRT	Proposed Method (ϵ : 50 px)	Proposed Method (ϵ : 30 px)	Proposed Method (ϵ : 10 px)
Path length (px)	1536 (100%)	1284 (83%)	1261 (82%)	1190 (77%)
Planning time (ms)	1752 (100%)	1753 (100%)	1753 (100%)	1753 (100%)

Consequently, the post triangular processing of the midpoint interpolation method (ϵ : 10 px) performed well in the path length aspect for all maps, demonstrating that the proposed method is efficient in terms of optimality.

5. Conclusions

In this research, the post triangular processing of the midpoint interpolation method minimized the planning time and shortened the path length of the sampling-based algorithm.

The proposed post triangular processing of the midpoint interpolation method was closer to the optimal path and somewhat solved the sharp path problem through the interpolation process. Furthermore, all path planning algorithms that plan a locally piecewise linear path could apply the proposed algorithm. This strength has significance in that it can be applied not only to the algorithm presented in this paper but also to various path planning algorithms. We validated the performance of the proposed method after it was applied to the RRT algorithm and its path planning results through simulation. It was verified that the path length was shortened by 18–38% (average 26%) depending on the threshold ϵ when applied to the RRT algorithm in the four different environment maps. As a result, the RRT algorithm applying the proposed post triangular processing of the midpoint interpolation method showed a more optimal path.

The proposed post triangular processing of the midpoint interpolation method is based on a general global planning case in which a robot first discovers a global path from its start point to its destination point before beginning to navigate. In dynamic environments, not only local planners but also global planners must deal with kinodynamic problems in real-time. Furthermore, the method proposed in this paper is more efficient in terms of the optimality of robot path planning, but optimality is not guaranteed. Future work should determine how to make a path that is closer to the optimal path.

Author Contributions: Idea and conceptualization: J.-G.K. and J.-W.J.; Methodology: J.-G.K. and J.-W.J.; Software: J.-G.K. and J.-W.J.; Experiment: J.-G.K. and J.-W.J.; Validation: J.-G.K., Y.-S.C. and J.-W.J.; Investigation: J.-G.K. and J.-W.J.; Resources: J.-G.K. and J.-W.J.; Writing: J.-G.K., Y.-S.C. and J.-W.J.; Visualization: J.-G.K. and J.-W.J.; Project administration: J.-W.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by a National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2020R1F1A1074974) and by the Ministry of Trade, Industry, and Energy (MOTIE) and the Korea Institute for Advancement of Technology (KIAT) through the International Cooperative R&D program (Project No. P0016096). It was also supported by the Technology development Program(S3041234) funded by the Ministry of SMEs and Startups (MSS, Korea) and by the MSIT (Ministry of Science and ICT), Korea, under the ITRC (Information Technology Research Center) support program (IITP-2021-2020-0-01789) supervised by the IITP (Institute for Information & Communications Technology Planning & Evaluation).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lindner, L.; Sergiyenko, O.; Moises, R.L.; Ivanov, M.; Julio, C.R.Q.; Daniel, H.B.; Wendy, F.F.; Vera, T.; Fabian, N.M.R.; Mercorelli, P. Machine Vision System Errors for Unmanned Aerial Vehicle Navigation. In Proceedings of the IEEE International Symposium on Industrial Electronics, Edinburgh, UK, 19–21 June 2017; pp. 1615–1620.
2. Sergiyenko, O.; Wendy, F.F.; Mercorelli, P. *Machine Vision and Navigation*, 1st ed.; Springer: Cham, Switzerland, 2020.
3. Abdi, A.; Adhikari, D.; Park, J.H. A novel hybrid path planning method based on q-learning and neural network for robot arm. *Appl. Sci.* **2021**, *11*, 6770. [[CrossRef](#)]
4. Schwab, K. *The Fourth Industrial Revolution*, 1st ed.; Crown Business: New York, NY, USA, 2017.
5. Wang, X.; Luo, X.; Han, B.; Chen, Y.; Liang, G.; Zheng, K. Collision-free path planning method for robots based on an improved rapidly-exploring random tree algorithm. *Appl. Sci.* **2020**, *10*, 1381. [[CrossRef](#)]
6. LaValle, S.M.; Kuffner, J.J., Jr. Randomized kinodynamic planning. *Int. J. Robot. Res.* **2001**, *20*, 378–400. [[CrossRef](#)]
7. Kuffner, J.J., Jr.; LaValle, S.M. RRT-Connect: An Efficient Approach to Single-Query Path Planning. In Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, CA, USA, 24–28 April 2000; Volume 2, pp. 995–1001.
8. Kang, J.-G.; Lim, D.-W.; Choi, Y.-S.; Jang, W.-J.; Jung, J.-W. Improved RRT-connect algorithm based on triangular inequality for robot path planning. *Sensors* **2021**, *21*, 333–364. [[CrossRef](#)] [[PubMed](#)]
9. Roy, D. Visibility graph based spatial path planning of robots using configuration space algorithms. *Robot. Auton. Syst.* **2009**, *24*, 1–9. [[CrossRef](#)]
10. Katevas, N.I.; Tzafestas, S.G.; Pnevmatikatos, C.G. The approximate cell decomposition with local node refinement global path planning algorithm: Path nodes refinement and curve parametric interpolation. *J. Intell. Robot. Syst.* **1998**, *22*, 289–314. [[CrossRef](#)]
11. Warren, C.W. Global Path Planning using Artificial Potential Fields. In Proceedings of the International Conference on Robotics and Automation, Scottsdale, AZ, USA, 14–19 May 1989; Volume 1, pp. 316–321.
12. LaValle, S.M. *Rapidly-Exploring Random Trees: A New Tool for Path Planning*; Springer: London, UK, 1998.
13. Karaman, S.; Frazzoli, E. Sampling based algorithms for optimal motion planning. *Int. J. Robot. Res.* **2011**, *30*, 846–894. [[CrossRef](#)]
14. Gammell, J.D.; Srinivasa, S.S.; Barfoot, T.D. Informed RRT*: Optimal Sampling based Path Planning Focused via Direct Sampling of an Admissible Ellipsoidal Heuristic. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 2997–3004.
15. Klemm, S.; Oberländer, J.; Hermann, A.; Roennau, A.; Schamm, T.; Zollner, J.M.; Dillmann, R. RRT*-Connect: Faster, Asymptotically Optimal Motion Planning. In Proceedings of the IEEE International Conference on Robotics and Biomimetics, Zhuhai, China, 6–9 December 2015; pp. 1670–1677.
16. Islam, F.; Nasir, J.; Malik, U.; Ayaz, Y.; Hasan, O. Rrt*-smart: Rapid Convergence Implementation of rrt* towards Optimal Solution. In Proceedings of the IEEE International Conference on Mechatronics and Automation, Chengdu, China, 5–8 August 2012; pp. 1651–1656.
17. Jeong, I.-B.; Lee, S.-J.; Kim, J.-H. Quick-RRT*: Triangular inequality based implementation of RRT* with improved initial solution and convergence rate. *Expert Syst. Appl.* **2019**, *123*, 82–90. [[CrossRef](#)]
18. Jung, J.-W.; So, B.-C.; Kang, J.-G.; Lim, D.-W.; Son, Y. Expanded Douglas–Peucker polygonal approximation and opposite angle based exact cell decomposition for path planning with curvilinear obstacles. *Appl. Sci.* **2019**, *9*, 638. [[CrossRef](#)]
19. Jung, J.-W.; So, B.-C.; Kang, J.-G.; Jang, W.-J. Circumscribed Douglas–Peucker Polygonal Approximation for Curvilinear Obstacle Representation. In Proceedings of the IEEE 2019 7th International Conference on Robot Intelligence Technology and Applications (RiTA), Daejeon, Korea, 1–3 November 2019; pp. 237–241.
20. Kwon, J.; Choi, K. Kinodynamic model identification: A unified geometric approach. *IEEE Trans. Robot.* **2021**, *37*, 1100–1114. [[CrossRef](#)]
21. Han, J. Mobile robot path planning with surrounding point set and path improvement. *Appl. Soft Comput.* **2017**, *57*, 35–47. [[CrossRef](#)]

Article

A Mobile Robot with Omnidirectional Tracks—Design and Experimental Research

Mateusz Fiedeń* and Jacek Bałchanowski

Faculty of Mechanical Engineering, Wrocław University of Science and Technology, Wybrzeże, Wyspiańskiego 27, 50-370 Wrocław, Poland; jacek.balchanowski@pwr.edu.pl

* Correspondence: mateusz.fieden@pwr.edu.pl

Abstract: This article deals with the design and testing of mobile robots equipped with drive systems based on omnidirectional tracks. These are new mobile systems that combine the advantages of a typical track drive with the advantages of systems equipped with omnidirectional Mecanum wheels. The omnidirectional tracks allow the robot to move in any direction without having to change the orientation of its body. The mobile robot market (automated construction machinery, mobile handle robots, mobile platforms, etc.) constantly calls for improvements in the manoeuvrability of vehicles. Omnidirectional drive technology can meet such requirements. The main aim of the work is to create a mobile robot that is capable of omnidirectional movement over different terrains, and also to conduct an experimental study of the robot's operation. The paper presents the construction and principles of operation of a small robot equipped with omnidirectional tracks. The robot's construction and control system, and also a prototype made with FDM technology, are described. The trajectory parameters of the robot's operation along the main and transverse axes were measured on a test stand equipped with a vision-based measurement system. The results of the experimental research became the basis for the development and experimental verification of a static method of correcting deviations in movement trajectory.

Citation: Fiedeń, M.; Bałchanowski, J. A Mobile Robot with Omnidirectional Tracks—Design and Experimental Research. *Appl. Sci.* **2021**, *11*, 11778. <https://doi.org/10.3390/app112411778>

Academic Editor: António Paulo Moreira

Received: 18 November 2021

Accepted: 7 December 2021

Published: 11 December 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: omnivehicle; omnitracks; Mecanum rollers; FDM

1. Introduction

In 2018, the world mobile robot market was worth USD 19 billion [1]. On the basis of the latest forecast, it is estimated that by 2023 its value will have nearly tripled. These robots have already moved from closed research centres and innovation fairs to the reality of everyday life. Autonomous transport vehicles used for supplies in storage facilities, autonomous mowers in house gardens, smart vacuum cleaners in homes, teleoperated robots, or bomb disposal robots are not only becoming a common element in company equipment, but also a commodity purchased by the average consumer [2,3]. Terrestrial mobile robots can be categorised using a number of criteria, and their constructed systems differ in terms of locomotion types. They can be wheeled, stepping, tracked, or ball balancing robots [4].

Wheeled robots are the most recognized, investigated and frequently used group of mobile robots. They achieve the highest speeds on flat surfaces. Their downside, however, is their poor ability to operate in difficult terrain, with some of them being vitiated by a significant turning radius [5]. Currently, research on improving their mobility in difficult terrain is being conducted. One new solution that has been proposed are hybrid wheeled-stepping robots [6–8].

At the opposite end of the robot spectrum are stepping robots, i.e., devices that are characterised by the best adaptation to overcome obstacles in terrain. Important disadvantages of their most common versions is low movement speed and complex problems related to the stepping control methodology (e.g., problems with maintaining balance

during dynamic stepping in unknown terrain) [9]. In recent years, research on improving the movement speed of stepping robots has been conducted [10,11].

Tracked robots can be classified as a compromise between wheeled and stepping robots. Tracks allow for the mobility of the robot in difficult terrain, while also helping them to maintain a satisfactory movement speed. This type of locomotion has been recognised, investigated and used to a significantly higher degree than stepping systems [12].

Ball-balancing robots, also called ballbots, are robots in which the wheelset system is a ball [13]. This locomotion type, apart from ensuring holonomic control, is a successful solution in robot locomotion that is used to demonstrate technology, and in toys or presentation robots. It is visually attractive. The ballbot group also encompasses robots with a spherical outer casing, or a casing resembling a spherical shape [14,15]. Due to such a construction, these robots can not only achieve a high speed, but can also overcome water obstacles in difficult terrain.

Mobile robotics is now at the stage of dynamic development, where it quickly adapts to novel technological solutions while maintaining some previously used ones. Various simple solutions are successfully used in modern applications. Paper [16] addresses the issues of the hybridization of animals and machines, as well as the use of robots for research on animals. Many of their functions, such as the movement of a mechanical fish, are performed by simple mechanical systems. Paper [17] describes the construction of a mobile robot, in which the stepping movements of 4 legs are produced by a single drive.

An example use of an existing and well-investigated solution can be seen in Mecanum wheel systems. They were patented in 1972 by Swedish engineer Bengt Erland Ilon [18]. The wheels, due to top free rotation rollers oriented at 45° on the wheel circumference, as well as appropriate steering, allow vehicles to move in any direction on a flat surface [19] (Figure 1). Another alternative is the use of transversal rollers oriented at 0° —although such a solution means that the implementation of an optimal wheel with a minimum gap may require an expensive manufacturing process [20].”

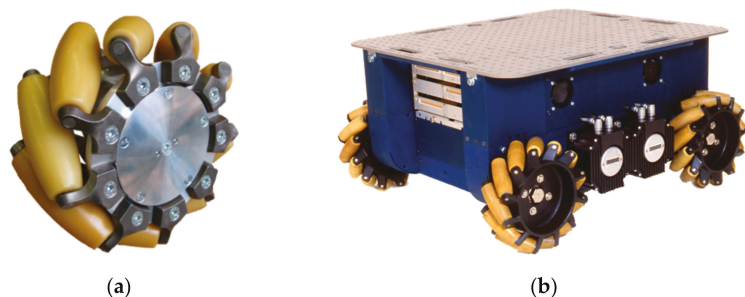


Figure 1. Mecanum wheel [21] (a), and the URANUS mobile robot with Mecanum wheels (b) [22].

A four-wheeled robot with Mecanum wheels (Figure 1b), which is driven by four independent motors, can move in any direction on a plane without the necessity to rotate [19,23,24]. A robot equipped with Mecanum wheels is holonomic, so the number of controlled degrees of freedom is equal to the number of degrees of freedom that are held. This is of particular significance in the case of robots that operate in very limited spaces. Fork-lifts and transport robots with Mecanum wheels, which are therefore able to move in any direction, can be used in storage facilities and production halls in which there is a lot of equipment and a large number of storage racks in a small space.

Apart from the whole array of advantages, Mecanum wheels also have considerable disadvantages: robots and vehicles equipped with such wheels need flat, even and clean surfaces. However, there have been many investigations aimed at improving the locomotion of such systems in difficult terrain [25]. The vibration of rotating rollers, which occurs during motion, is another drawback of Mecanum wheels. The reason for this vibration is

the cyclical shifting of the load between the subsequent rollers of a wheel [26]. In most applications, the vibration will not have a significant impact on the robot's operation, however, in some specific uses it can have a significant influence on it [27]. Another potential issue is the distribution of load on the floor surface. Due to their structure, Mecanum wheels have a small wheel-surface contact area. To improve the force distribution and to prevent defects of the surface on which the robot moves, additional, surplus wheels are used in numerous applications for the purpose of reducing the impact of load on the surface. In some applications, non-driven castor wheel types are used, with additional driven Mecanum wheels being used in some others [28].

In recent years, research has been conducted on the possibility of combining Mecanum wheels with a classical drive track. Mobile robots with such a track system have all the advantages of vehicles with Mecanum wheels, but do not take on their disadvantages. They can move on uneven, dirt-covered terrain and can carry heavier loads due to the multiple number of rollers in the tracks. Systems with such a drive can be used as mobile storage robots, automated construction machines (excavators, drilling rigs) [29], inspection or rescue robots, transport platforms and bomb disposal robots. The use of multidirectional tracks in the construction of mobile robots significantly increases their maneuverability.

In 1999, in Isod's work [30], the issue of multidirectional tracked robots was discussed. Such attempts to transfer the properties of Mecanum wheels to tracked wheels were made in 2015 and are described by Zhang [31].

In 2017, a German company, IVA Johann GmbH, in cooperation with scientists from the Bremer Institut für Produktion und Logistik, presented a commercial prototype of a mobile robot equipped with multidirectional tracks [29]. In 2018, the results of simulation research conducted by Zhang [32] on a platform equipped with multidirectional tracks were published. In 2020, Fang published research results involving the construction and tests of a fork-lift equipped with multidirectional tracks, which was designed to transport loads of a maximum of 2 tons [33]. During the research, which was both experimental and numerical, it was shown that a robot with multidirectional tracks preserves its holonomic characteristic.

The paper published by Huang discussed mobility in difficult terrain [34]. During movement along the robot's main axis, multidirectional tracks preserve the robot's ability to overcome terrain obstacles, such as embankments, stairs, thresholds or curbs. In the case of transverse movement to the robot's main axis, the ability to overcome thresholds and steps was significantly reduced. Due to the ability to overcome obstacles while travelling along the main axis, the new robot will replace systems with Mecanum wheels in warehouse facilities and production halls that may have floors with uneven surfaces, thresholds and kerbs.

Simulation research results on rectilinear motion in the transverse direction of a robot with multidirectional tracks are presented in [33]. It was demonstrated that the robot's motion trajectory becomes curved during such motion, which was also confirmed by the experimental research discussed in [32]. Deviations from the rectilinear path depend on the robot's mass, and such deviations increase with a decrease in system mass. An article written by Huang [34] also discusses research on the transverse motion of a robot on a smooth and slippery surface and includes the problems resulting therefrom.

Primarily robots with longitudinally symmetric non-overlapping tracks were investigated in the publications referred to above, and only in one case were robots with symmetric and partly overlapping tracks scrutinized. However, the publications do not discuss issues related to the applications of non-symmetric tracks, nor do they discuss other track systems. Research on the influence of load on driving properties has been conducted for robots with a mass of a maximum of 5 tons. The issue of vibration during the locomotion of robots with multidirectional tracks was not discussed. No attempt to compensate for deviations of direction during locomotion was made.

In this paper is developed a new mobile robot with a drive system that uses multidirectional tracks. The research encompasses the ability of a prototype to perform basic manoeuvres, i.e., motion along the main and transverse robot axes. A static method of im-

proving motion parameters by correcting control parameters was developed and verified. The tests were conducted on a prototype made with fused deposition modelling (FDM) 3D printing technology on a test bench equipped with a vision-based measurement system.

2. Materials and Methods

2.1. Construction of a Mobile Robot with Multidirectional Tracks

A classical vehicle with a track drive system (excavator, bulldozer, tank, armoured personnel carrier, bomb disposal robot, etc.) is made of a body and two continuous track systems located symmetrically to the body (Figure 2a). A vehicle with a drive system with multidirectional tracks is equipped with four continuous track systems, which can usually be positioned in various ways with regards to the vehicle's body. The typical schemes of continuous track positions encompass tracks that are completely overlapping (Figure 2b), tracks that are partially overlapping (Figure 2c), as well as systems with non-overlapping tracks (Figure 2d).

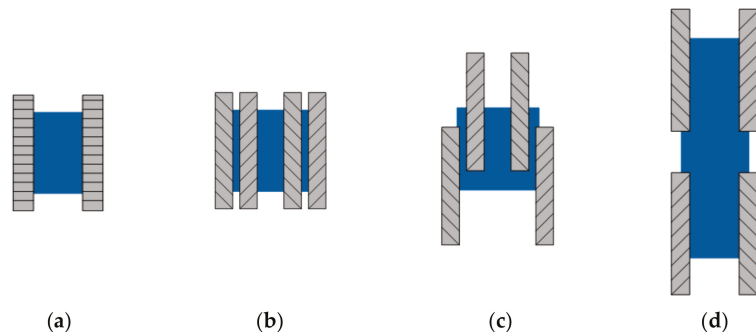


Figure 2. Schemes of track positions with two and four continuous tracks: (a) symmetrical system, (b) symmetrical system with completely overlapping tracks, (c) system with partially overlapping tracks, (d) system with non-overlapping tracks.

It was assumed that the mobile robot to be constructed would have the structure of a symmetrical system with completely overlapping tracks, i.e., it will be made of a body and four parallel track drive systems of equal length with completely overlapping tracks (Figure 2b).

During contact between the continuous track and the surface, tractive force is generated. This force is necessary for a vehicle to move. A large surface-track contact area and an appropriately adjusted track preload result in a lower unit pressure of the track when compared to the wheel. A standard, single continuous track system is made of a track located between the drive wheels, idler, road wheels and return rollers (Figure 3a). In some track vehicles (e.g., excavators), there are solutions in which the drive wheels and idlers also take the load-carrying capacity (Figure 3b). This type of powertrain was also selected for the designed mobile robot. However, due to the large radius of the drive wheels and idlers in relation to their spacing, no extra road wheels were used (Figure 3c).

Tracks in the continuous track system are composed of a closed sequence of several dozen links that are connected to one another with rotating pairs. In the case of multidirectional tracks, each link is equipped with an additional rotating roller positioned at angle γ (usually $30^\circ < \gamma < 60^\circ$) to the link axis. The view of a single link with length a , width b , and height h , and also a cylindrical roller with radius r_r , is presented in Figure 4.

The designed mobile robot is equipped with four independent continuous track systems with multidirectional tracks that have only drive wheels and idlers. It is a four-degrees-of-freedom system. The drive is transferred from the rotating drives to the drive wheels using belts. In the middle part of the robot's body, at point R , the local $x'y'$ coordinate system is located. The kinematic scheme of the robot is shown in Figure 5.

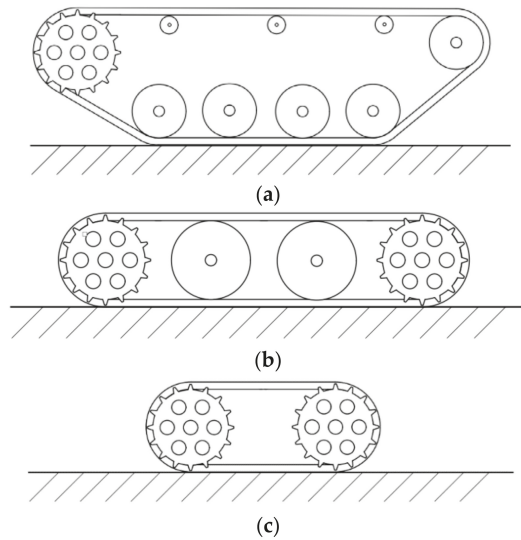


Figure 3. Continuous track system: (a) standard system, (b) system with drive road wheels, idlers and road wheels, (c) system with drive road wheels and idlers without road wheels.

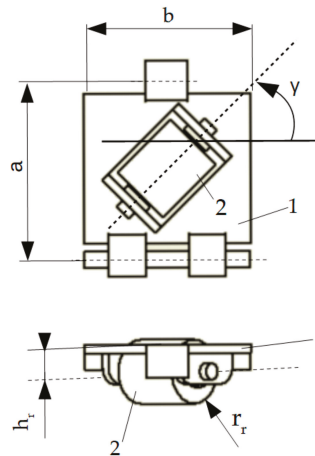


Figure 4. View of a single multidirectional track link: 1—link, 2—roller.

It was assumed that the designed robot would be used indoors (residential buildings, offices, warehouse facilities, production halls, etc.). Therefore, its dimensions must be adapted to overcome the typical obstacles seen in such places: doors, thresholds, staircases, etc. The robot should move at a maximum speed of $v_R = 0.2$ m/s, which is the speed considered acceptable for a robot moving in a ‘human environment’ [35]. This means that the maximum track speeds 1, 2, 3, 4:

$$v_R = v_1 = v_2 = v_3 = v_4 \tag{1}$$

The resulting basic dimensions are described in Table 1.

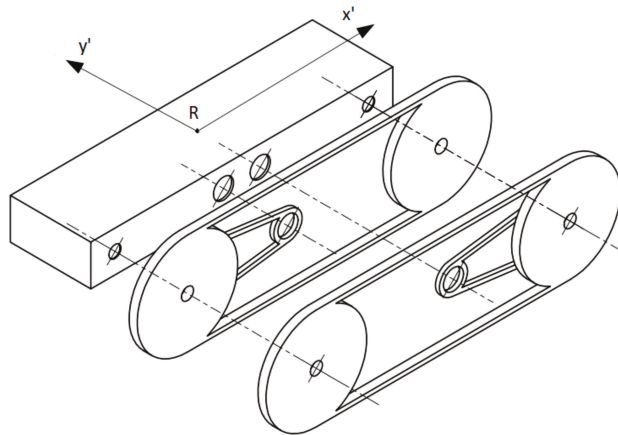


Figure 5. Kinematic scheme of the robot with a continuous track system that has multidirectional tracks.

Table 1. Dimension and construction assumptions of the mobile robot.

Dimension Name and Symbol	Value	Dimension Name and Symbol	Value
robot mass m_r	6 kg	working load u	2 kg
track link height a	45 mm	track link width b	40 mm
roller radius r_r	10 mm	distance between the axis of the roller and the plane of the track link h_r	5 mm
roller mounting angle γ	45°	distance between pairs of continuous track systems c	270 mm
distance between the continuous track systems in pair d	80 mm	distance between the axes of the driving and idler wheels e	225 mm
belt wheel radius on drive wheel r_{ks}	60 mm	belt wheel radius on motor r_s	15 mm

Figure 6 shows an overview of the developed solid model of the robot and the detailed views of the solid model of the track system.

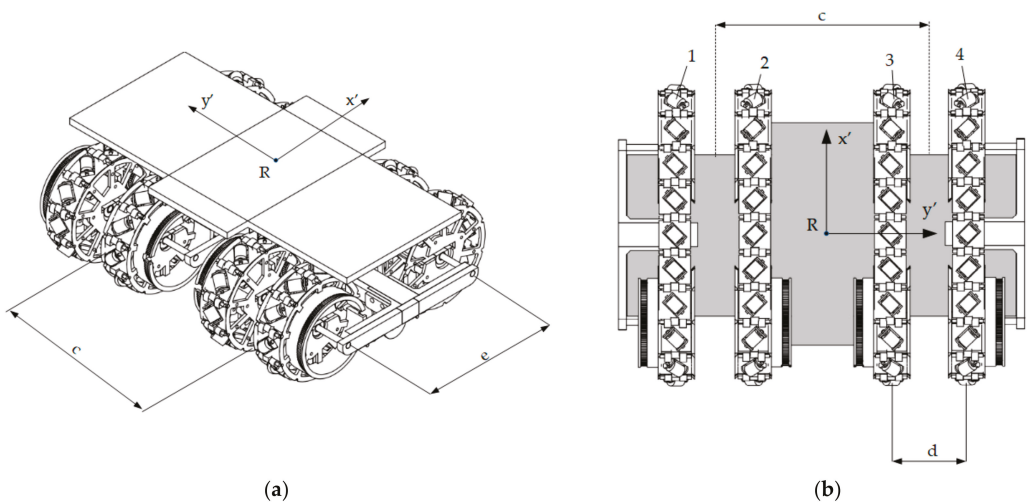


Figure 6. General view (a) and bottom view (b) of the solid model of the robot with the continuous track system with multidirectional tracks.

Figure 7 shows the views of a single continuous multidirectional track with its own drive system. The continuous track system is composed of a track, an idler and a drive wheel. In each track segment there is one cylindrical rotating roller. The drives are located in the central part of the robot. The active moment is shifted to the drive wheel using transmission with a toothed belt.

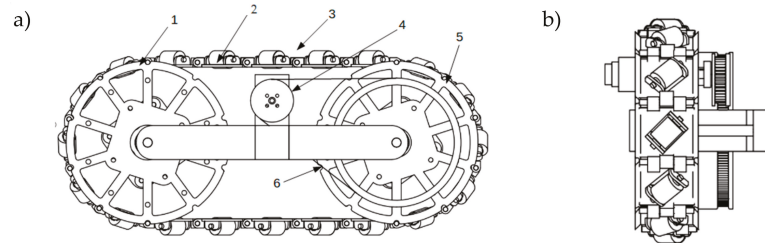


Figure 7. View of the continuous track system with multidirectional tracks: (a) side view and (b) front view (1—idler, 2—link with a rolling roller, 3—track, 4—motor pulley, 5—drive wheel, 6—toothed belt).

The track drive systems, track links, rollers and the body of the robot were made using incremental FDM technology. The material used is PET-G, which is a polymer characterised by low shrinkage and high mechanical resistance. The free rollers rotate on steel axes. There are plastic slip rings between the rollers and the mount, which is connected to the track plate. The rollers consist of a printed core, onto which the tyres are pulled. The tyres of the rollers, also made using FDM technology, are made of thermoplastic polyurethane with a Shore scale hardness of 40D. The view of the finished prototype of the mobile robot is presented in Figure 8.

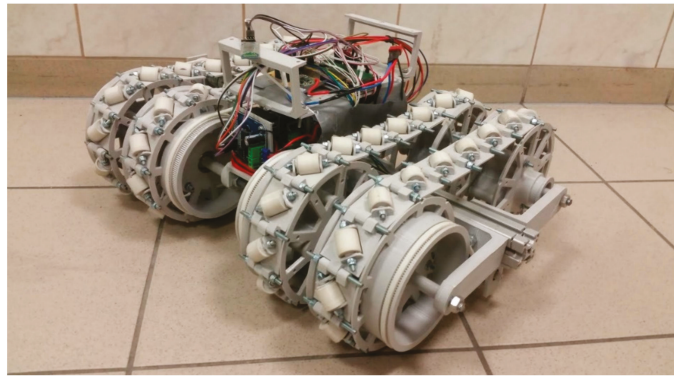


Figure 8. View of the finished robot prototype equipped with a continuous track system with multidirectional tracks.

The motor is connected to the drive wheel by a reduction pulley transmission system with a reduction ratio of $i_k = r_s / r_{ks} = 0.25$. For the assumed maximum robot motion speed v_r (Table 1), the maximum angular velocity of the motor ω_s was determined from the following equation:

$$\omega_s = \frac{v_r}{r_{ks} i_k} = 10.67 \text{ rad/s} \quad (2)$$

The rated active moment of motor M_s was selected using the simplified method of the experimental measurements of the robot's motion resistance in the longitudinal and transverse direction using a tensometric force sensor. The highest motion resistance was measured experimentally during motion in the direction perpendicular to the main axis,

and is expressed as the maximum force that excites motion in a set direction, which was $F_T = 13$ N. It was assumed that the motor moment M_s should be greater than:

$$M_s > \frac{r_k F_T}{4} \tag{3}$$

On the basis of the calculations made to design the prototype construction, POLOLU-2274 motors were selected with a torque of $M_s = 0.57$ Nm, and a nominal speed of $n_s = 21.99$ rad/s. Each motor is equipped with a magnetic encoder that provides 48 counts per revolution, and therefore one complete revolution of the wheel will provide 2249 pulses, with a drive wheel accuracy of $\Delta\phi_s = 0.0007$ rad.

2.2. Control System of the Mobile Robot with Multidirectional Tracks

For the purpose of exciting the set robot motion, it is necessary to have an appropriately designed system and control algorithms. The mobile robot has four drives that independently excite the motion of each track (Figures 5 and 8). Thus, a control system is required to control the robot. The control system should have four control regulators of speed ω_{si} of motor i ($i = 1, 2, 3, 4$), which start the drive wheels of the continuous tracks. In the robot prototype, the speed control of the used DC motors will have the form of a pulse width modulation (PWM) signal. The general scheme of the developed robot’s control system is presented in Figure 9.

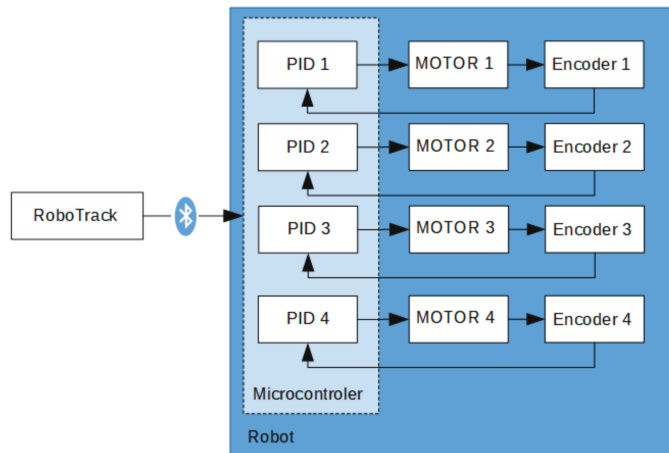


Figure 9. General block diagram of the mobile robot’s control system.

Motor i , which excites the motion of the continuous track system i , is individually controlled using a PID $_i$ regulator operating in the feedback loop with a frequency of 50 Hz. The value of $v_R(t)$ is the set speed of the entire robot, the value of $\omega_{Ti}(t)$ is the value of the set speed of track motor i ($i = 1, 2, 3, 4$), $E_i(t)$ is the control error, $U_i(t)$ is the steering signal, $P_i(t)$ is the PWM value responsible for motor control DC, $\omega_{Mi}(t)$ is the actual motor rotational speed, $\Theta_i(t)$ is the motor rotation angle measured with an encoder, while $\omega_i(t)$ is the motor angular speed. The scheme of the speed regulator of continuous track system i is presented in Figure 10.

A RoboTrack control program was developed to control the movement of the robot. The program is run on an external PC.

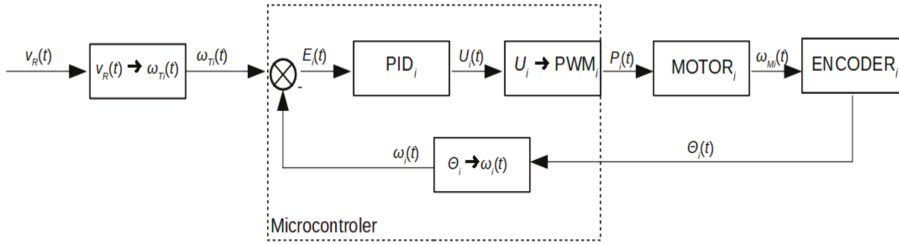


Figure 10. Scheme of the speed regulator controlling continuous track system i ($i = 1, 2, 3, 4$).

The programme is responsible for planning the robot’s motion trajectory and for sending the set speed values $\omega_{Ti}(t)$ for particular motors i to the robot’s control system based on speed values set by the user. RoboTrack also ensures the on-line registration of ride parameters. Communication between the external PC and the mobile robot is performed with a Bluetooth communication link

The excitation of the robot’s motion along the set trajectory requires a suitable control of speeds v_1, v_2, v_3, v_4 of tracks 1, 2, 3, 4. This paper discusses robot motion along two trajectories: a straight line μ_w at constant speed v_{Rx} along the longitudinal axis, and a straight line μ_p at constant speed v_{Ry} along the robot’s transverse axis (Figure 11). In the case of motion along the longitudinal axis at speed v_{Rx} , the tracks should move at the same speeds (Figure 11):

$$v_1 = v_2 = v_3 = v_4 = v_{Rx} \tag{4}$$

while during motion along the transverse axis at speed v_{Ry} , the tracks’ speed takes the following values:

$$v_1 = v_3 = v_{Ry} \quad v_2 = v_4 = -v_{Ry}. \tag{5}$$

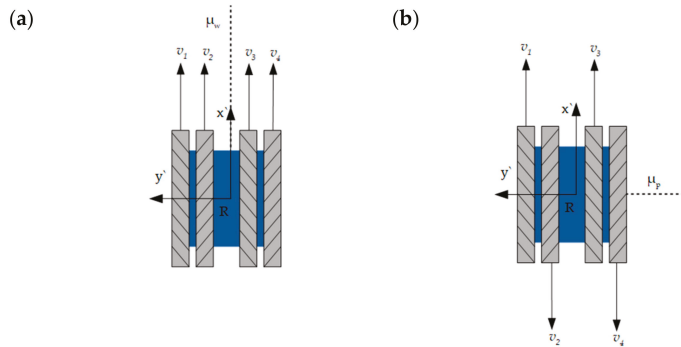


Figure 11. Scheme of the control system of continuous track system i ($i = 1, 2, 3, 4$) for motion along trajectory μ_w (a) and trajectory μ_p (b).

2.3. External Measurement Test Bench

The mechanical prototype proposed in this paper is not able to develop self-localization [36]. The localization of the mobile robot is determined with an external measurement test bench, the general scheme and view of which are presented in Figure 12. The vision-based measurement system was made of a camera located perpendicularly to the measured area. The ELP-USBFHD04H-MFV camera with a resolution of 1920×1080 px was located at the height $h = 1.95$ m with regard to the main coordinate system xy (Figure 12).

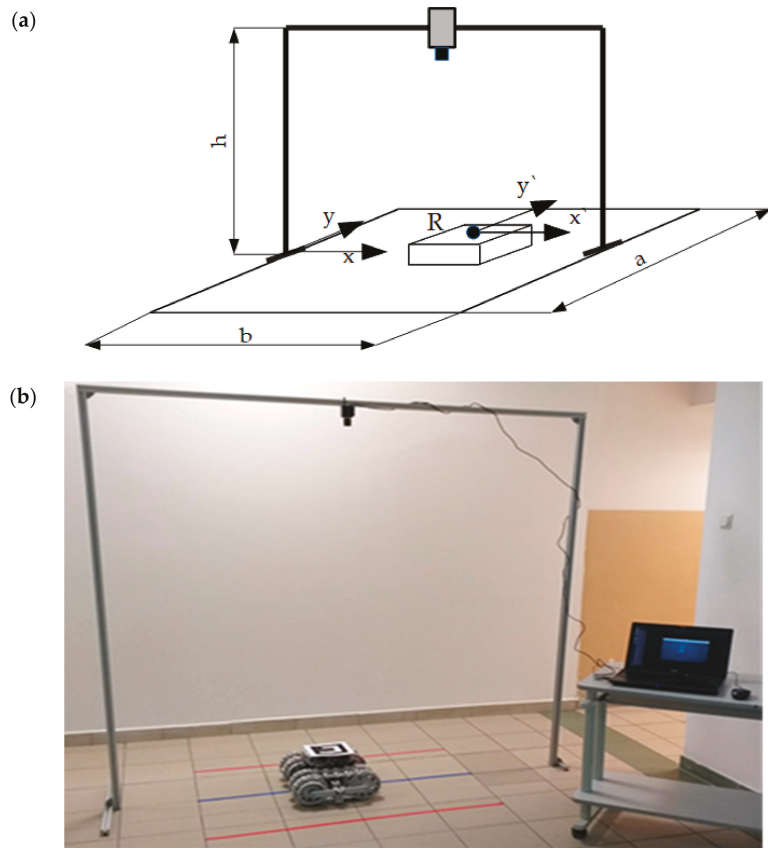


Figure 12. Robot motion test bench: (a) general scheme, (b) test bench view.

The measuring accuracy of the bench was checked using a printed calibration sheet with ArUco markers (Figure 13). The position reading error δl of the marker was less than 0.008 m and the orientation angle error $\delta\alpha$ was no more than 0.5° .

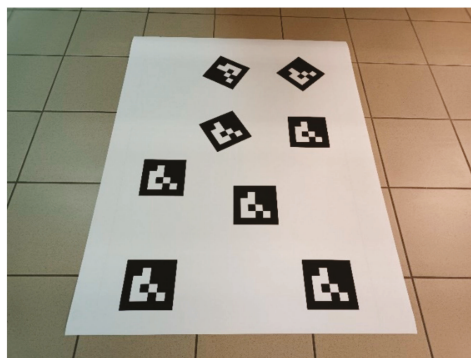


Figure 13. Calibration sheet with ArUco markers.

The vision-based system records and analyses images with a frequency of 25 Hz (frames/s). In order to allow for the detection of the robot's position in a camera image, the vision marker ArUco was placed on its top surface (Figure 14). The ArUco marker is a print on a flat tile with white and black rectangles. Owing to the precisely defined distribution of these areas, the marker can transfer its ID number, and it is therefore possible to recognize its presence in a ready image, as well as to determine its linear and angular position. For known dimensions, it is also possible to conduct a programmable estimation of its location in three-dimensional space. The middle of the marker is located in point R on the robot's body (Figure 14). During the measurement, the whole marker must be located in the camera vision area—this efficiently reduces the observation area to the area of the following dimensions: $a = 0.75$ m, $b = 1.30$ m.

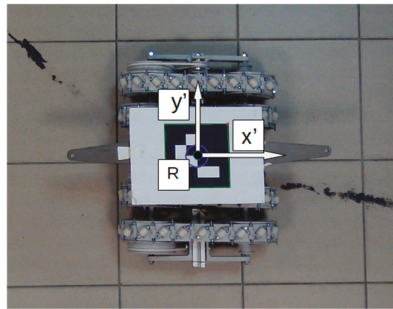


Figure 14. ArUco marker view on the robot's body.

The robot is controlled by RoboTrack software (Figure 15), which allows robot motion to be controlled and track and motor motion to be registered. The software was integrated with a VisionM module that allows the covered distance to be measured and recorded. This module is based on OpenCV libraries that are extended with OpenCV Contrib [37] modules. Its task is the online analysis of individual image frames recorded by a camera placed on the test stand. The module provides data on the angular orientation of the tracked robot, its current location and the distance covered, which are recorded in real time.

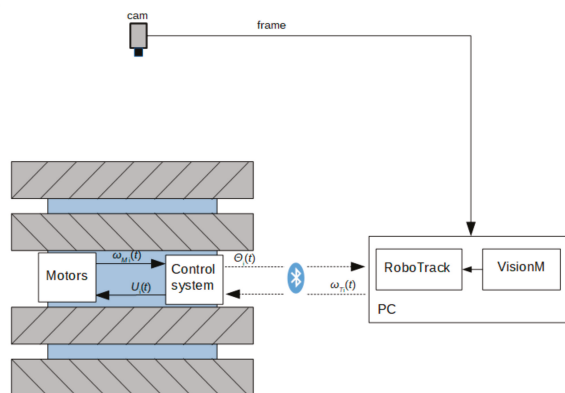


Figure 15. Data flow at the measurement test bench.

2.4. Static Correction Method of the Robot's Locomotion Direction

During the motion of the mobile robot with the continuous track system with multi-directional tracks travelling at a constant speed, the phenomenon of curved trajectories may take place [24,27]—the middle R of the robot's body deviates from the trajectory,

and the orientation angle α_R of the body changes. The deviations from the straight-line trajectory depend on the robot's mass, and they increase with the system's mass [27]. During movement along the transverse axis, the problems with maintaining straight line movement become even more significant when the robot moves on a slippery surface. This disadvantageous phenomenon can be prevented by the introduction of suitable corrections to the set speeds of particular continuous tracks.

For the purpose of maintaining the robot's motion along the set straight line trajectory, the fixed orientation angle α_R of the robot's body should be ensured by introducing suitable corrections of track speeds.

A difference in the orientation α_R of the body, i.e., between the real and set robot position, results from the robot's body rotation at constant angular speed ω_r . The rotation can be reduced by forcing the body to rotate in the opposite direction by making appropriate changes in track set speeds.

A simplified model of the change in the robot's body orientation and motion direction (Figure 16) was proposed, which entails the introduction of virtual tracks located between tracks 1 and 2,—marked as L_{12} , with 3 and 4 being marked as R_{34} . The speed v_k of the virtual tracks excites the corrective rotation of the body, which eliminates the change of orientation, can be determined from Equations (6) and (7). These equations take into consideration the change in the angle $\Delta\alpha_p$ of the orientation at time Δt , structural dimension c describing the distance between the pairs of continuous tracks, and an empirical adjustment factor c_f resulting from the surface type and the adhesion of the rotating rollers:

$$\omega_R = \frac{\Delta \alpha_R}{\Delta t} \tag{6}$$

$$v_k = -\omega_R c_f \frac{1}{2} c \tag{7}$$

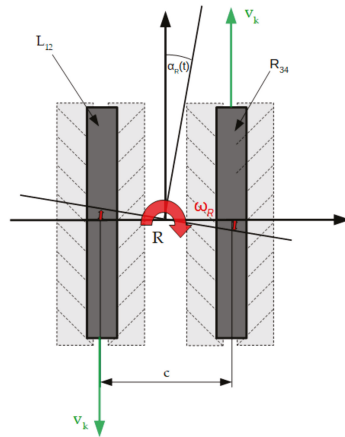


Figure 16. Illustration of the change in angular orientation α_R of the robot's body under the influence of angular speed ω_r , and also the description of this change using a method of eliminating the change in orientation by introducing corrective speeds v_k in virtual tracks R and L .

In the case of the robot's motion along the longitudinal axis, the correction of track speed involves its suitable increasing or decreasing by value v_k according to the following formula (Figure 17a):

$$v_1 = v_2 = v_{Rx} + v_k \quad v_3 = v_4 = v_{Rx} - v_k \tag{8}$$

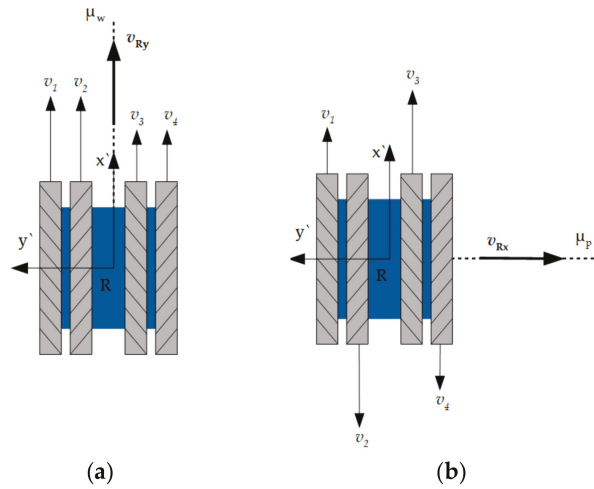


Figure 17. Illustration of track speed correction for the purpose of maintaining constant robot orientation during motion: (a) along the longitudinal axis along trajectory μ_w , (b) along the transverse axis along trajectory μ_p .

Speed correction v_k for motion along the robot’s transverse axis consists of differentiating the speed on each pair of tracks according to formula (Figure 17b):

$$v_1 = -v_4 = v_{Ry} - v_k \quad v_2 = -v_3 = v_{Ry} + v_k \quad (9)$$

2.5. Experiment Plan

A number of experimental tests of the mobile robot were carried out on the test stand in order to verify the operation of the proposed drive systems and to determine the basic kinematic properties of the movement.

The tests encompassed a number of rides on straight line trajectory μ_z located along the longitudinal and transverse axes of the robot. Various measurements of the location, the linear and angular speeds of the robot’s body, the rotation angle and angular speeds of the drive wheels, and also the speed of the tracks were conducted during the rides. The research was conducted on a specially designed test bench equipped with a vision-based location measurement system. Figure 18 presents the scheme of the experiment carried out on the measurement test bench.

Ride tests were conducted at two stages. In the first part of the experiment, the set values of the linear speeds of all the tracks did not take into account the correction parameters. In the second part of the experiment, the speeds of particular drives were differentiated by the correction value, which was calculated on the basis of the proposed model of ride trajectory correction.

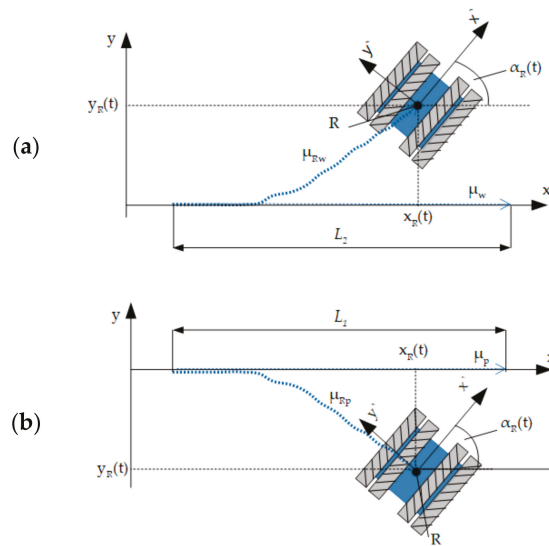


Figure 18. General scheme of the conducted tests of the robot's ride tests on a set straight line trajectory along the longitudinal axis (a) and along the transverse axis (b) of the robot (l_1, l_2 —lengths of set trajectory, μ_T —set trajectory, μ_R —real trajectory, $x_R(t), y_R(t)$ —coordinates of point R on the robot's body, $\alpha_r(t)$ —body coordination angle).

3. Results

The results of the measurements obtained during the investigated robot rides along the longitudinal and transverse robot axes are presented and discussed below. The tests and measurements were carried out on the designed measurement test bench (Figure 12). The length of the tested trajectory was limited by the area of the test bench. The trajectory waveforms $\mu_R(x_R, y_R)$, the dependence of change $\Delta\alpha$ on time, and the change of the coordinates x_R, y_R over time were plotted on the basis of the stationary measuring system, while the speeds of the individual tracks were determined on the basis of the angles indicated by the encoders in relation to time.

3.1. Robot Ride Tests

The tests of the robot's rides along the longitudinal axis were conducted in accordance with the scheme shown in Figure 18a. The robot moved along the planned straight-line trajectory μ_T by length $L_1 = 1.2$ m at the constant speed $v_{Rx} = 0.12$ m/s.

The scheme presented in Figure 18b shows the plan of the research on the robot's rides along the transverse axis. The robot moved along the planned straight-line trajectory μ_T by length $L_2 = 1.2$ m at constant speed $v_{Ry} = 0.12$ m/s.

During the research, the waveforms of real ride trajectories $\mu_R(x_R, y_R)$, the changes of body orientation angle $\Delta\alpha$, coordinates x_R, y_R of point R on the robot's body, and the real linear speeds v_1, v_2, v_3, v_4 of the tracks calculated on the basis of encoder indications were measured and registered. Figures 19–22 show the measurement results for rides along the longitudinal axis, while Figures 23–26 depict the measurement results of the robot's rides along the transverse axis.

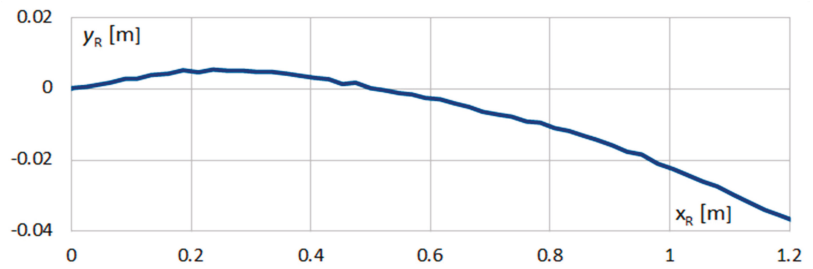


Figure 19. Trajectory $\mu_R (x_R, y_R)$ of point R movement on the robot's body during the ride along the longitudinal axis.

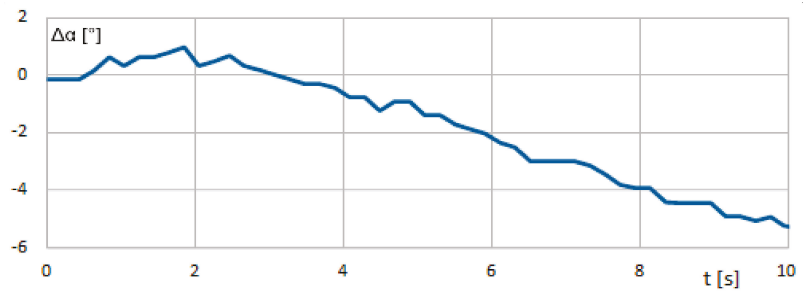


Figure 20. Waveforms of changes in angle $\Delta\alpha$ of the robot's body orientation at time t during the ride along the longitudinal axis.

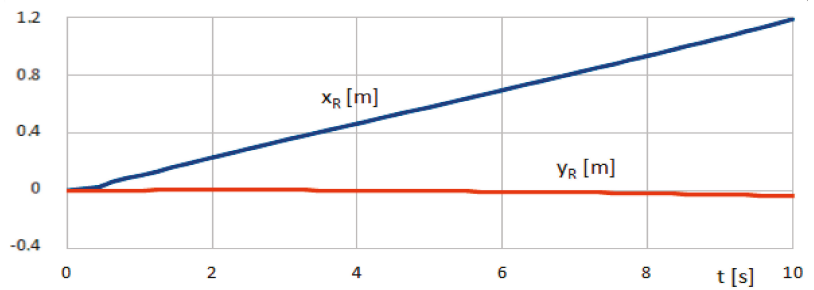


Figure 21. Plots of coordinates x_R, y_R of point R on the robot's body during the ride along the longitudinal axis.

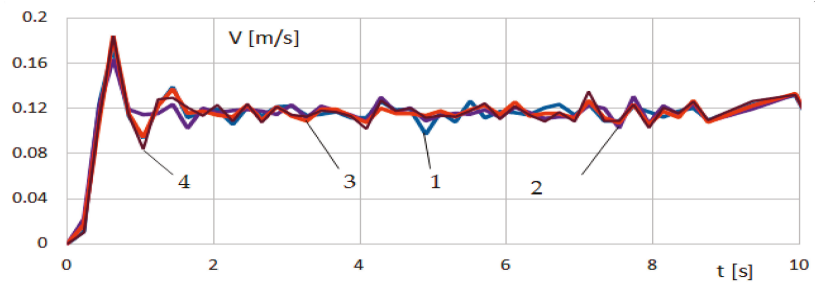


Figure 22. Waveforms of real linear speeds v_1, v_2, v_3, v_4 of tracks 1, 2, 3, 4 of the robot during the ride along the longitudinal axis.

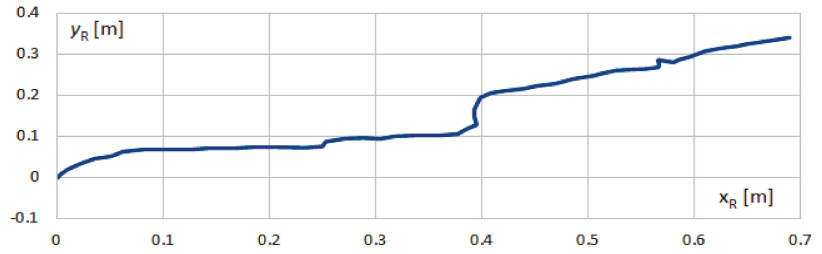


Figure 23. Trajectory $\mu_R(x_R, y_R)$ of point R movement on the robot's body during the ride along the transverse axis.

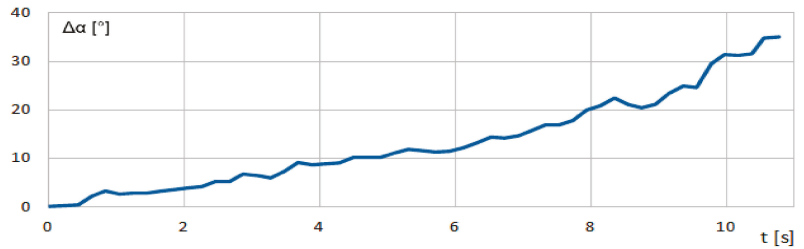


Figure 24. Waveforms of changes in angle $\Delta\alpha$ of the robot's body orientation to time t during the ride along the transverse axis.

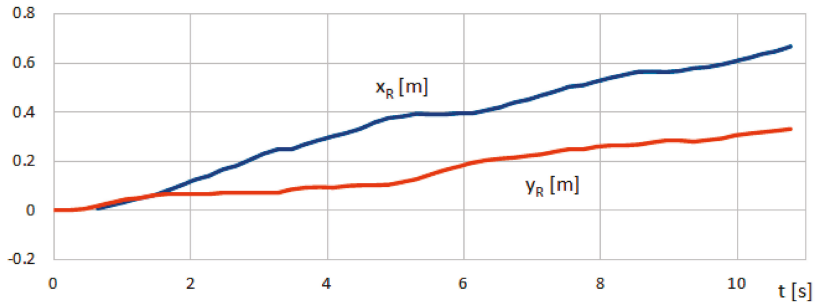


Figure 25. Plots of coordinates x_R, y_R of point R on the robot's body during the ride along the transverse axis.

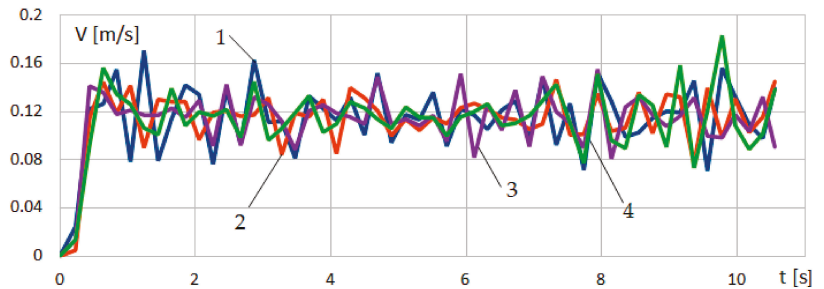


Figure 26. Waveforms of real linear speeds v_1, v_2, v_3, v_4 of tracks 1, 2, 3, 4 of the robot during the ride along the transverse axis.

The experiment showed that during the rides the real, linear speeds of tracks v_i adopted the values of the set speeds, v_{Rx} and v_{Ry} . In the case of longitudinal motion, the largest speed errors occurred at the start-up of the robot's motion. After the stabilisation of speed, the errors took significantly smaller values (Figure 22). In the case of transverse motion, the values of the track speed errors occurring at the start-up were comparable to the deviations observed during the whole ride (Figure 26). It should, therefore, be noted that no such error jump between the set and real speed in transverse motion was observed. Despite the speed jump at start-up, the maximum deviation between the real and set speed Δv_i was $\Delta v_i = 0.02$ m/s for the longitudinal motion, and $\Delta v_i = 0.03$ m/s for the transverse motion (Figures 22 and 26).

In both longitudinal and transverse motion, deviations from the planned trajectory were observed in the form of a curved ride and a change in the orientation of the robot's body (Figures 20 and 24). The location error of point R was determined to be $\Delta y_R = y_{RT} - y_R$. In both the longitudinal and transverse motion, the set trajectory is a straight line overlapping with the X axis of the measurement test bench coordinate system. Due to this, for both rides the theoretical location of point R in relation to the Y axis is $y_{RT} = 0$. Location deviation $\Delta y_R = y_R$.

In the case of the longitudinal motion, the final location error of point R was $\Delta y_R = 0.037$ m (Figure 21), while the orientation change was $\Delta\alpha = 5^\circ$ (Figure 20) at time $t = 10$ s.

During the transverse motion, the robot did not cover the whole planned distance ($L_2 = 1.2$ m), and the measurement was interrupted after it had covered the distance of 0.7 m (Figure 25) because the robot overstepped the measurement area boundary in the direction of the y axis. In the case of the transverse motion, the final orientation change was $\Delta\alpha = 35^\circ$ (Figure 24), and the location error of point R was $\Delta y_R = 0.36$ m (Figure 25) at time $t = 10.8$ s.

The analysis of the waveforms of changes in the orientation angle $\Delta\alpha$ of the robot's body at time t indicated that they had a linear nature. This was confirmed by the conducted statistical analyses. The value of the Pearson linear correlation starts to be higher at 0.97 onwards for both the longitudinal and transverse motions.

3.2. Tests of Robot Rides with Consideration for the Static Correction Method of Ride Direction

The phenomenon of the change in the angle $\Delta\alpha$ of body orientation had a linear nature, and therefore the application of the static correction method set out in Section 2.5 was justified. On the basis of the planned changes of orientation angle $\Delta\alpha$ during a ride (Figures 20 and 24), the values of speed v_{kw} , corrections for the longitudinal motion, and v_{kw} for the transverse motion, were calculated using Formulas (6) and (7).

In the calculations, the adaptation factor c_f was taken into account. For the given robot's dimensions, the value of the c parameter was 0.16 m. With consideration for the type of substrate (track link rollers made of TPU filament with a hardness of 40D moving on ceramic tiles), the value of the factor was determined, empirically, to be $c_f = 4.4$.

The corrective speed values, obtained using Formula (7), were $v_{kw} = 0.04$ m/s for the longitudinal motion and $v_{kp} = 0.35$ m/s for the transverse motion. The corrective speed value $v_{kp} = 0.35$ m/s were restricted to 0.2 m/s due to the limitations of the drive and control systems of the used equipment.

For the obtained values of the corrective speed, two more tests of the robot's rides along the longitudinal and transverse axes were conducted. Figures 27–30 present the results of the measurements for the rides along the longitudinal axis, while Figures 31–34 show the results for the measurements of the robot's rides along the transverse axis.

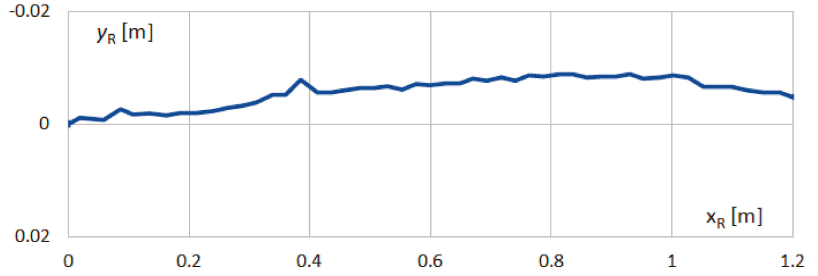


Figure 27. Trajectory $\mu_R(x_R, y_R)$ of point R movement on the robot's body during the ride along the longitudinal axis with ride direction correction.

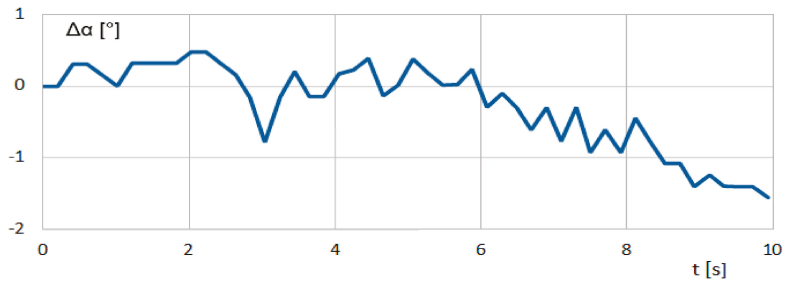


Figure 28. Waveforms of changes in angle $\Delta\alpha$ of the robot's body orientation at time t during the ride along the longitudinal axis with ride direction correction.

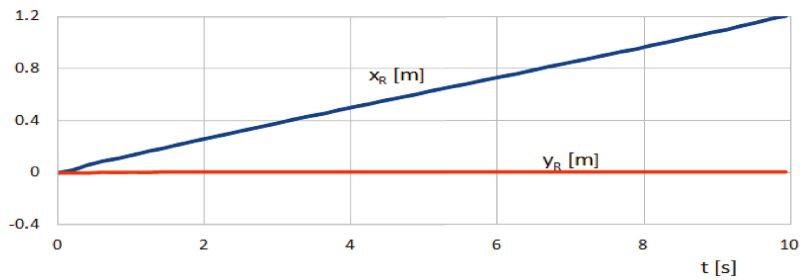


Figure 29. Plots of coordinates x_R, y_R of point R on the robot's body during the ride along the longitudinal axis with ride direction correction.

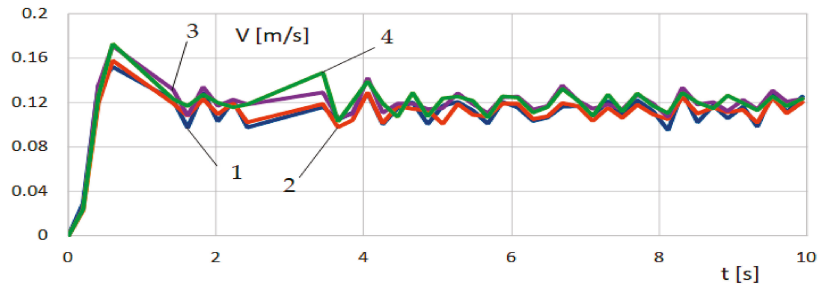


Figure 30. Waveforms of real linear speeds v_1, v_2, v_3, v_4 of tracks 1, 2, 3, 4 of the robot during the ride along the longitudinal axis with ride direction correction.

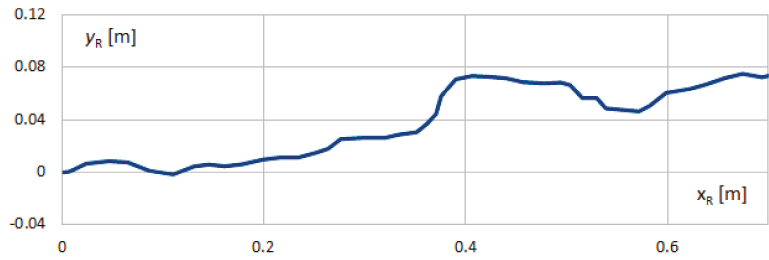


Figure 31. Trajectory $\mu_R (x_R, y_R)$ of point R movement on the robot's body during the ride along the transverse axis with ride direction correction.

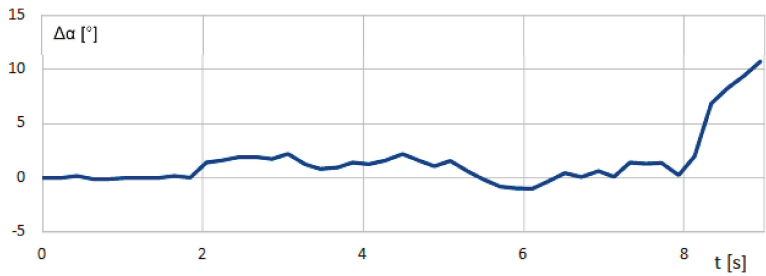


Figure 32. Waveforms of changes in angle $\Delta\alpha$ of the robot's body orientation to time t during the ride along the transverse axis with ride direction correction.

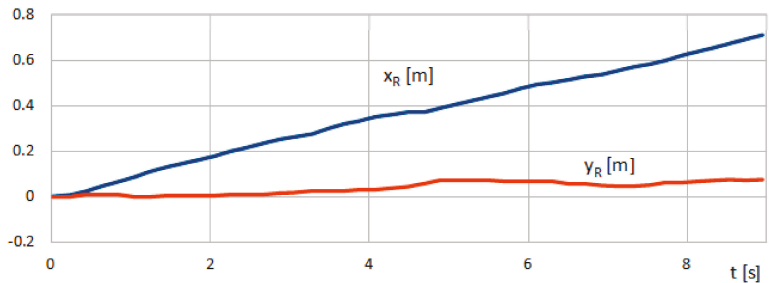


Figure 33. Plots of coordinates x_R, y_R of point R on the robot's body during the ride along the transverse axis with correction.

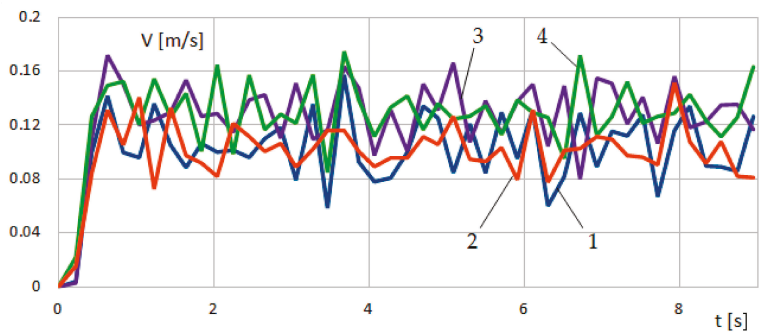


Figure 34. Waveforms of real linear speeds v_1, v_2, v_3, v_4 of tracks 1, 2, 3, 4 of the robot during the ride along the transverse axis with correction.

The measurements showed that the obtained linear speeds of the tracks v_i assume values close to the set speeds v_{Rx} and v_{Ry} , which were corrected by values v_{kw} and v_{kp} . The differences between real value v_i and set value v_{Rx} for the longitudinal motion were $\Delta v_i < 0.02$ m/s (Figure 30), disregarding the temporary error jump at the start-up. For the transverse motion, the same value was $\Delta v_i < 0.04$ m/s (Figure 34). In this case, analogically as in the transverse motion without correction, no temporary error jump of speed at start-up was recorded. Therefore, the worsening ability to maintain the set speed v_i of the tracks in the transverse motion was observed, which was probably caused by the occurrence of low friction between the surface and the track.

Both in the case of the transverse and longitudinal motion, there was a significant flattening of the trajectory curve (Figures 27 and 31). In the presented case, the final change of orientation angle $\Delta\alpha$ of the body during longitudinal movement after the introduction of corrective speeds v_{kw} was 1.4° (Figure 28), while the final point R location error was $\Delta y_R = 0.01$ m (Figure 29). A fourfold improvement of the drive parameter $\Delta\alpha$ was obtained (Table 2).

Table 2. Comparison of the conducted measurement results based on the distance of 0.7 m for the transverse motion, and 1.2 m for the longitudinal motion.

Longitudinal Motion		Transverse Motion	
Ride without Direction Correction	Ride with Direction Correction	Ride without Direction Correction	Ride with Direction Correction
$\Delta\alpha = 0.085$ rad	$\Delta\alpha = 0.025$ rad	$\Delta\alpha = 0.549$ rad	$\Delta\alpha = 0.207$ rad
$\Delta v_i < 0.02$ m/s	$\Delta v_i < 0.02$ m/s	$\Delta v_i < 0.03$ m/s	$\Delta v_i < 0.04$ m/s
$\omega_R = 0.009$ rad/s	$\omega_R = 0.003$ rad/s	$\omega_R = 0.052$ rad/s	$\omega_R = 0.024$ rad/s

In the case of transverse motion without speed correction, the robot did not cover the whole planned distance (Figure 25). Moreover, the measurement was stopped after covering 0.7 m at time $t = 9$ s (Figure 33), because the robot crossed the boundary of the measuring field in the direction of the Y axis. The same part of the trajectory was taken into account when analysing lateral movement while both considering and not considering, the correction of the driving (Figure 33).

The change of body orientation $\Delta\alpha$ after $t = 9$ s, i.e., at the time equal to the longitudinal drive without speed correction was 11° (Figure 32), while the location error was $\Delta y_R = 0.07$ m (Figure 33).

In the transverse motion, the observed improvement trajectory curve was smaller than in the case for longitudinal motion. This resulted from, among other things, a 50% decrease in the required corrective speed v_{kp} , which took place for equipment-related reasons. The so-reduced correction is not perfectly efficient and therefore it did not allow for the complete improvement of the robot's trajectory. The obtained improvement (over a threefold reduction of error $\Delta\alpha$ (Table 2)), however, proved the correctness of the used method in the transverse motion of the robot.

The obtained measurement results of drive parameters were statistically analysed and are presented in Table 2.

4. Discussion

The paper presents a solution for a robot equipped with completely overlapping multidirectional tracks with a symmetrical roller position. A light robot prototype was designed and constructed using additive manufacturing technology to be later used in experimental research on a specially constructed test measurement bench. The robot's parameters were examined along the longitudinal and transverse axes.

The experimental research without ride direction correction confirmed the occurrence of the effect of motion trajectory deviation during transverse motion. Similar effects of the trajectory curve were observed during the simulation tests of the robot with non-overlapping tracks [27] and with partly overlapping tracks [24].

The data quoted in the references indicate a 3.8% deviation between the theoretical speed and the one obtained in the dynamic simulation during transverse motion, and less than a 1% deviation in the main motion. In these references, no quantitative data was presented regarding the change in the angular orientation of the robot.

The data obtained during the conducted experiment showed a deviation increment of 0.016 rad/s in the longitudinal motion, and 0.133 rad/s in the transverse motion.

Such significant values of the trajectory curve probably result from the mechanical imperfections of the prototype, which were probably due to the adopted drive system's structure without idling rolls and the used manufacturing technology. The fact that the track links and rollers were made of TPU filament, were of 40D hardness in the Shore scale, and had a low friction coefficient resulted in low adhesion of the robot to the surface. It was shown that this effect could be corrected using a suitable control system. The proposed static method of ride direction correction performed this role. The application of the continuous track speed correction allowed the angular deviation during longitudinal motion to be reduced to 0.007 rad/s, and to 0.019 rad/s during transverse motion. Despite the used correction by a value smaller than would result from the transverse motion calculations, a significant improvement in trajectory parameters was observed. This confirms the correctness of the used method and creates opportunities for further research and experiments.

The downside of the used correction method is the fact that it is a static method, which means that it requires earlier experimental tests of a robot on a particular type of surface, with the results needing to be used as the basis for the determination of efficient correction parameters. The application of a dynamic method of ride direction correction would improve the performance even more, and, in effect, eliminate the consequences of deviations in trajectory during robot motion. In further research stages, the prototype will be equipped with additional body orientation angle measurement sensors, which will allow the use of dynamic motion direction correction.

5. Conclusions

The key considerations of this paper were the design and testing of a mobile robot with an omnidirectional track. The research confirmed the ability of the robot to drive omnidirectionally while maintaining a fixed body orientation. The use of multidirectional tracks in the construction of mobile robots significantly increases their manoeuvrability.

The obtained results clearly show that vehicles equipped with continuous track systems with multidirectional tracks can still be improved. The application of the approach, involving the compensation of mechanical imperfections with the use of additional sensors and appropriate control, will allow for an even more precise maintenance of the set motion direction. The mechanical design of the track system also had a great influence on the accuracy of the robot's movement. Introducing additional road wheels and idlers or changing the material and geometry of the rollers in the track links will improve the contact between the track and the ground.

The development of systems equipped with continuous track systems could be used, inter alia, in the construction of equipment operating on narrow construction sites with numerous obstacles. Tracked machines, which are able to move in the transverse direction, would facilitate manoeuvring and contribute to reducing the number of accidents. Another potential area of application is the transport of small, yet heavy loads, in very limited spaces—in this case a uniform load distribution on the whole track could be of key significance for the usability of such vehicles.

Author Contributions: Conceptualization, M.F. and J.B.; methodology, M.F. and J.B.; software, M.F.; validation, M.F. and J.B.; formal analysis, M.F. and J.B.; investigation, M.F. and J.B.; resources, M.F.; data curation, M.F. and J.B.; writing—original draft preparation, M.F.; writing—review and editing, M.F. and J.B.; visualization, M.F.; supervision, J.B.; project administration, M.F. and J.B.; funding acquisition, J.B. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Grants-in-Aid at Wrocław University of Science and Technology, Faculty of Mechanical Engineering, Poland, (No. 8211104160).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Contact via email.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Mobile Robots Market by Operating Environment (Aerial, Ground, and Marine), Component (Control System, Sensors), Type (Professional and Personal & Domestic Robots), Application (Domestic, Military, Logistics, Field), and Geography—Global Forecast 2023. Markets and Markets SE 3610. Available online: https://www.marketsandmarkets.com/Market-Reports/mobile-robots-market-43703276.html?gclid=CjwKCAiAksyNBhAPEiwAIDBeLNkIJXjxNs2XgzEZLhSghCanirhMDnTZn9YKn3O3mNjX-upD3xwBhoCD6wQAvD_BwE (accessed on 11 December 2021).
2. Irobot. Available online: <https://irobot.pl/pl/outlet/312-irobot-roomba-980-5060359281043.html> (accessed on 27 October 2020).
3. Husqvarna. Available online: <https://www.husqvarna.com/pl/produkty/kosiarki-automatyczne> (accessed on 27 October 2020).
4. Moreno, J.; Clotet, E.; Lupiañez, R.; Tresanchez, M.; Martínez, D.; Pallejà, T.; Casanovas, J.; Palacín, J. Design, Implementation and Validation of the Three-Wheel Holonomic Motion System of the Assistant Personal Robot (APR). *Sensors* **2016**, *16*, 1658. [[CrossRef](#)]
5. Wong, J.Y.; Huang, W. “Wheels vs. tracks”—A fundamental evaluation from the traction perspective. *J. Terramech.* **2006**, *43*, 27–42. [[CrossRef](#)]
6. Bałchanowski, J. Modelling and simulation studies on the mobile robot with self-leveling chassis. *J. Theor. Appl. Mech.* **2016**, *54*, 149–161. [[CrossRef](#)]
7. Sperzyński, P.; Bałchanowski, J.; Gronowicz, A. Simulation of motion of a mobile robot on uneven terrain. *J. Theor. Appl. Mech.* **2020**, *58*, 541–552. [[CrossRef](#)]
8. Gronowicz, A.; Szrek, J. Design of LegVan Wheel-Legged Robot’s Mechanical and Control System. *SYROM* **2009**, 145–158.
9. Machado, J.A.T.; Silva, M.F. An Overview of Legged Robots. In Proceedings of the MME 2006—International Symposium on Mathematical Methods in Engineering, Ankara, Turkey, 27–29 April 2006.
10. Raibert, M.; Blankespoor, K.; Nelson, G.; Playter, R. BigDog, the Rough-Terrain Quadruped Robot. *IFAC Proc. Vol.* **2008**, *41*, 10822–10825. [[CrossRef](#)]
11. Bledt, G.; Powell, M.J.; Katz, B.; Di Carlo, J.; Wensing, P.M.; Kim, S. MIT Cheetah 3: Design and Control of a Robust, Dynamic Quadruped Robot. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2245–2252.
12. Hornback, P. The Wheel Versus Track Dilemma. *ARMOR* **1998**, *26*, 33–34.
13. Hertig, L.; Schindler, D.; Bloesch, M.; Remy, C.D.; Siegwart, R. Unified state estimation for a ballbot. In Proceedings of the Robotics and Automation (ICRA), 2013 IEEE International Conference on Robotics and Automation, Karlsruhe, Germany, 6–10 May 2013.
14. Seeman, M.; Broxvall, M.; Saffiotti, A.; Wide, P. An Autonomous Spherical Robot for Security Tasks. In Proceedings of the Computational Intelligence for Homeland Security and Personal Safety, Alexandria, VA, USA, 16–17 October 2006.
15. Guardbot. Available online: <https://guardbot.org/> (accessed on 27 October 2020).
16. Lee, Y.; Yoon, D.; Oh, J.; Kim, H.S.; Seo, T. Novel Angled Spoke-Based Mobile Robot Design for Agile Locomotion with Obstacle-Overcoming Capability. *IEEE/ASME Trans. Mechatron.* **2020**, *25*, 1980–1989. [[CrossRef](#)]
17. Romano, D.; Donati, E.; Benelli, G.; Stefanini, C. A review on animal–robot interaction: From bio-hybrid organisms to mixed societies. *Biol. Cybern.* **2019**, *113*, 201–225. [[CrossRef](#)] [[PubMed](#)]
18. Ilon, B.E. Wheels for a Course Stable Selfpropelling Vehicle Movable in Any Desired Direction on the Ground or Some Other Base. U.S. Patent US3876255A, 8 April 1975.
19. Li, Y.; Ge, S.; Dai, S.; Zhao, L.; Yan, X.; Zheng, Y.; Shi, Y. Kinematic Modeling of a Combined System of Multiple Mecanum-Wheeled Robots with Velocity Compensation. *Sensors* **2019**, *20*, 75. [[CrossRef](#)] [[PubMed](#)]
20. Palacín, J.; Martínez, D.; Rubies, E.; Clotet, E. Suboptimal Omnidirectional Wheel Design and Implementation. *Sensors* **2021**, *21*, 865. [[CrossRef](#)]
21. Imetron GmbH. Mecanum Wheel. Attribution-ShareAlike 3.0 Unported (CC BY-SA 3.0). Available online: <https://de.wikipedia.org/wiki/Mecanum-Rad#/media/Datei:Meacnum-Rad.png> (accessed on 27 October 2020).
22. Gwpcmu. Mecanum Wheel. Attribution 3.0 Unported (CC BY 3.0). Available online: https://en.wikipedia.org/wiki/Mecanum_wheel#/media/File:UranusOmnidirectionalRobotPodnar.png (accessed on 27 October 2020).
23. Wu, M.; Dai, S.-L.; Yang, C. Mixed Reality Enhanced User Interactive Path Planning for Omnidirectional Mobile Robot. *Appl. Sci.* **2020**, *10*, 1135. [[CrossRef](#)]
24. Palacín, J.; Rubies, E.; Clotet, E.; Martínez, D. Trajectory of a human-sized mobile robot using three omnidirectional wheels: From simulation to implementation. In press. *Sensors* **2021**, *21*, 7216.

25. Yamada, N.; Komura, H.; Endo, G.; Nabae, H.; Suzumori, K. Spiral Mecanum Wheel Achieving Omnidirectional Locomotion in Step-Climbing. In Proceedings of the 2017 IEEE International Conference on Advanced Intelligent Mechatronics (AIM), Munich, Germany, 3–7 July 2017; pp. 1285–1290.
26. Bae, J.-J.; Kang, N. Design Optimization of a Mecanum Wheel to Reduce Vertical Vibrations by the Consideration of Equivalent Stiffness. *Shock. Vib.* **2016**, *2016*, 5892784. [[CrossRef](#)]
27. Moreno, J.; Clotet, E.; Tresanchez, M.; Martínez, D.; Casanovas, J.; Palacín, J. Measurement of Vibrations in Two Tower-Typed Assistant Personal Robot Implementations with and without a Passive Suspension System. *Sensors* **2017**, *17*, 1122. [[CrossRef](#)]
28. Rizzo, C.; Lagraña, A.; Serrano, D. GEOMOVE: Detached AGVs for Cooperative Transportation of Large and Heavy Loads in the Aeronautic Industry. In Proceedings of the 2020 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC), Ponta Delgada, Portugal, 15–17 April 2020; pp. 126–133.
29. Mortensen Ernits, R.; Hoppe, N.; Kuznetsov, I.; Uriarte, C.; Freitag, M. A New Omnidirectional Track Drive System for Off-Road Vehicles. In Proceedings of the XXII International Conference on “Material Handling, Constructions and Logistics”, Belgrade, Serbia, 4–6 October 2017.
30. Isoda, T. Roller-Crawler Type of Omni-Directional Mobile Robor for Off-Road Running. *IEEE Trans. Robot. Autom.* **2002**, *18*, 251–256.
31. Zhang, Y.; Huang, T. Research on a tracked omnidirectional and cross-country vehicle. *Mech. Mach. Theory* **2015**, *87*, 18–44. [[CrossRef](#)]
32. Zhang, Y.; Yang, H.; Fang, Y. Design and motion analysis of a novel track platform. *J. Phys. Conf. Ser.* **2018**, *1074*, 012014. [[CrossRef](#)]
33. Fang, Y.; Zhang, Y.; Li, N.; Shang, Y. Research on a medium-tracked omni-vehicle. *Mech. Sci.* **2020**, *11*, 137–152. [[CrossRef](#)]
34. Tao, H.; Yunan, Z.; Peng, T.; Nanming, Y.; Jian, Z. Design & Kinematics Analysis of a Tracked Omnidirectional Mobile Platform. *J. Mech. Eng.* **2014**, *50*, 206–212.
35. Guillén Ruiz, S.; Calderita, L.V.; Hidalgo-Paniagua, A.; Bandera Rubio, J.P. Measuring Smoothness as a Factor for Efficient and Socially Accepted Robot Motion. *Sensors* **2020**, *20*, 6822. [[CrossRef](#)] [[PubMed](#)]
36. Lin, H.-Y.; He, C.-H. Mobile Robot Self-Localization Using Omnidirectional Vision with Feature Matching from Real and Virtual Spaces. *Appl. Sci.* **2021**, *11*, 3360. [[CrossRef](#)]
37. OpenCV. Available online: <https://opencv.org/> (accessed on 13 May 2021).

Article

Manipulation Planning for Large Objects through Pivoting, Tumbling, and Regrasping

Ang Zhang *, Keisuke Koyama, Weiwei Wan and Kensuke Harada

Graduate School of Engineering Science, Osaka University, 1 Machikaneyamacho, Osaka 560-8531, Japan; koyama@sys.es.osaka-u.ac.jp (K.K.); wan@sys.es.osaka-u.ac.jp (W.W.); harada@sys.es.osaka-u.ac.jp (K.H.)

* Correspondence: zhang@hlab.sys.es.osaka-u.ac.jp

Abstract: Robotic manipulation of a bulky object is challenging due to the limited kinematics and payload of the manipulator. In this study, a robot realizes the manipulation of general-shaped bulky objects utilizing the contact with the environment. We propose a hierarchical manipulation planner that effectively combined three manipulation styles, namely, pivoting, tumbling, and regrasping. In our proposed method, we first generate a set of superimposed planar segments on the object surface to obtain an object pose in stable contact with the table, and a set of points on the object surface for the end-effectors (EEFs) of a dual-arm manipulator to stably grasp the object. Object manipulation can be realized by solving a graph, considering the kinematic constraints of pivoting and tumbling. For pivoting, we consider two supporting styles: stable support (SP) and unstable support (USP). Our proposed method manipulates large and heavy objects by selectively using the two different support styles of pivoting and tumbling according to the conditions on the table area. In addition, it can effectively avoid the limitation arising due to the arm kinematics by regrasping the object. We experimentally demonstrate that a dual-arm manipulator can move an object from the initial to goal position within a limited area on the table, avoiding obstacles placed on the table.

Keywords: non-prehensile manipulation; manipulation planning; pivoting; robotics

Citation: Zhang, A.; Koyama, K.; Wan W.; Harada, K. Manipulation Planning for Large Objects through Pivoting, Tumbling, and Regrasping. *Appl. Sci.* **2021**, *11*, 9103. <https://doi.org/10.3390/app11199103>

Academic Editor: Alessandro Gasparetto

Received: 3 September 2021

Accepted: 27 September 2021

Published: 30 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Humans often manipulate large and heavy objects utilizing the contact of the object with environment. Although robotic manipulation of general-shaped large and heavy objects is challenging, we aim to realize a robot manipulating such objects by effectively utilizing the contact of the object with a table (Figure 1). Among the manipulation styles using contact with a table, pivoting refers to the style of inclining and rotating an object on its vertex. Pivoting enables a robot to manipulate a bulky object with a relatively small manipulating force because the robot does not need to lift the object [1]. Moreover, the pivoting gait, which is a manipulation style, enables a robot to move an object by pivoting it multiple times by changing its rotational vertex. In addition, to start the pivoting gait from the designated initial pose of the object, a robot may once tumble the object by rotating it on its edge.

Pivoting gait has been explored for moving a simple box-shaped object [2,3]; however, this study aims to realize the robotic manipulation of a general-shaped object by combining pivoting, tumbling, and regrasping (Figure 2). Let us consider the example shown in Figure 3, where a robot moves a blue-colored object to the designated location on a table, with avoiding an obstacle (red colored). In this case, the robot first tumbles the object to an upright posture and then passes through the narrow passage using the pivoting gait. For realizing such combined manipulation, we need to determine the object face required to contact with table and the grasping pose of the object. In addition, a robot may change the grasping pose multiple times, and such changes of the grasping pose are referred as regrasping.



Figure 1. A dual-arm robot manipulates a piano by pivoting on the object’s vertex (marked by a yellow dot).

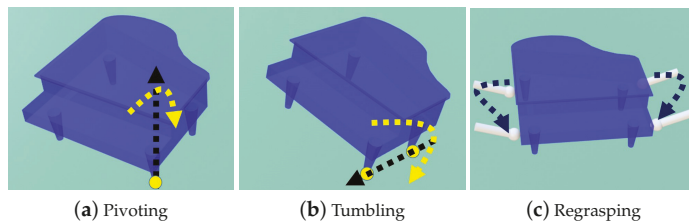


Figure 2. Manipulation of a piano by (a) pivoting, (b) tumbling, and (c) regrasping.

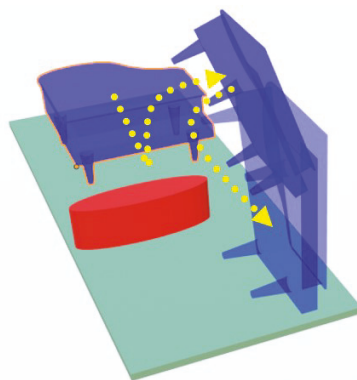


Figure 3. Motion sequence for moving a piano forward and avoiding collision with an obstacle (red colored). Both the initial and goal object poses are in SP, whereas the intermediate object pose is in USP. The yellow trajectory indicates the motion of the object’s CoM.

Our proposed planner comprises offline and online phases. In the offline phase, we first preprocess the mesh model of a general-shaped object clustered into superimposed segments [4] in order to determine the object’s contact surface with table and the contact surfaces with the end-effectors (EEFs). In the online phase, we propose a hierarchical motion planning method to determine the motion of both the object and the robot. Hierarchical planning includes task level planning and motion level planning. In the task level, we sample the object configurations on the table considering the kinematic constraints of pivoting and tumbling. At this level, the planner constructs a graph, where each node in the graph represents an object pose and a grasp configuration, and each edge in the

graph indicates a primitive motion to transform the object poses or grasp configurations. The purpose of primitive motion is to realize either pivoting, tumbling, or regrasping. On the other hand, in the motion level, the robotic manipulation motions are generated. Here, the pivoting and tumbling motions are generated by predicting the object's future dynamics using model predictive control (MPC), considering the kinematic constraints for maintaining contact with the environment.

Furthermore, this study expands the feasible object poses used in motion planning to both stable placements (SP) and unstable placements (USP). Here, SP indicates that the vertical projection of the object's center of mass (CoM) is included in the object's supporting area. The robot can easily regrasp the object in SP. On the other hand, in USP, a robot can manipulate an object within a limited support area, which offers more choice for avoiding collision in a narrow space. Despite this advantage, it is difficult to regrasp the object in USP because USP not only requires contact with the environment but also contact with the EEFs. If contact between the object and the EEFs is lost during USP, the object will fall. To solve this problem, the two EEFs sequentially change the contact point position, i.e., the left EEF moves to the desired position while the right EEF maintains contact with the object and vice versa.

The contributions of this work are as follows:

- A manipulation plan for pivoting a general-shaped object is proposed;
- Multiple motions, including tumbling, pivoting, and regrasping, are combined to better manipulate the object;
- Pivoting gait is planned in both SP and USP.

This remainder of this paper is organized as follows. Section 2 reviews the related work. Section 3 provides an overview of the proposed method. The required preprocessing of the object model to manipulate a general-shaped object is presented in Section 4. Section 5 introduces the manipulation planning design. It describes the sampling of the object configurations and the design of primitive motions, such as pivoting, regrasping, tumbling, which are combined into a graph. The performed simulation and experiments are detailed in Section 6. Finally, Section 7 summarizes the results and discusses the future work.

2. Related Works

2.1. Non-Prehensile Manipulation

The non-prehensile manipulation allows a robot to manipulate an object without firmly grasping it by taking advantage of contact with environment [5]. Examples of non-prehensile manipulation include pushing [6], tumbling [7], scooping [8], tilting [9], throwing [10], catching [11], batting [12], sliding [13] and pivoting [14].

Pivoting is efficient for manipulating heavy objects because the weight of the object is mostly supported by the table. Aiyama et al. [15] first proposed the manipulation of a heavy object by pivoting. Doshi et al. [16] and Raessa et al. [17] proposed motion planners to reorient an object by pivoting. Yoshida et al. [18,19] proposed a motion planner planning an object path during the pivoting gait. Shi et al. [20] proposed a tumbling motion planner. In addition, there have been some research on motion planning combining the pivoting gait with other manipulation styles. Fakhari et al. [21] proposed the manipulation planner combining the pivoting gait with tumbling. Murooka et al. [22,23] combined pivoting, pushing and tumbling. In all the above mentioned works, the pivoting gait was planned for a simple box shaped objects. On the other hand, this is a first trial on planning the motion of a general shaped object by combining pivoting, tumbling and regrasping. In addition, our planner considers both SP and USP to efficiently manipulate an object.

2.2. Regrasp Planning

There are several studies on the manipulation planner with regrasping an object [24–29]. Berenson et al. [30], and Bouyamane et al. [31] presented regrasp planners with a change in contact state between the object and the environment. Harada et al. [32] proposed a regrasp planning for dual-arm robots. Hayashi et al. [33] implemented manipulation planning to

wrap-up fabric by incorporating two robot arms with regrasping. Wan et al. [34] proposed manipulation planning, which determines a sequence of dual-arm robot motion, to reorient an object with regrasping.

2.3. Grasp Planning

The grasp planning searches for a grasping pose of an object satisfying the force/form closure [35–38] or grasp stability condition [39–41]. In many regrasp planner, multiple grasping poses are calculated by using a grasp planner [4,42,43], before executing the motion planner. Then, a robot regrasps an object by switching among multiple grasping poses. To provide steady grasps, many elements must be considered, such as contact zones, object surface curvatures, mechanics of robot hands, and so on [44–48]. For two-fingered robot gripper, Jones et al. [49] and Wolter et al. [50] proposed grasp planners. Surface segmentation was used to determine the grasping points for a parallel-jaw gripper [42,51]. This study uses grasp planners for a parallel-jaw gripper [4] to determine the contact points on a general-shaped object manipulated by a dual-arm manipulator with ball-shaped EEFs. Cooperating two manipulators with the EEFs enables a robot to grasp a large object which may be too large for a gripper to grasp.

3. Steps of the Proposed Motion Planner

In this study, we assume that the 3D shape of the object is given. In addition, we assume that a dual-arm robot with ball-shaped EEFs manipulates the object at two contact points. To manipulate the object to the target pose, a hierarchical manipulation planner is built at the task and motion levels, which outputs the planned motions. The proposed motion planner is designed according to the following steps:

- Before executing the motion planner, the 3D model of the target object is analyzed to obtain the information necessary for the manipulation planner. This information includes the potential rotational vertices used for pivoting, edges used for tumbling, base surfaces for stably contacting the table, and the contact points between the object and EEFs;
- Task level planning then is performed. We discretize the object poses on a table. The object configurations along with their grasp configurations are saved in the graph nodes. In this phase, we consider object poses in both SP and USP;
- Finally, in motion level planning, primitive motions, such as pivoting, tumbling, and regrasping, are planned for moving the object to the target location. In the motion level, the designed motions include pivoting, tumbling, and regrasping. MPC is implemented to generate the motions because it can find the motions required for maintaining contact between the object and environment. If we cannot find any feasible motion in this level, we go back to the task level planning.

4. Object Model Analysis

To manipulate a general-shaped object, we preprocess its 3D-shape model and obtain the necessary information for manipulation planning. We aim to determine the following:

- A set of object vertices, which are the potential rotational vertices during pivoting;
- A set of object edges, which are the potential rotational edges during tumbling;
- A set of stable object placements;
- A set of grasp configurations for the dual-arm manipulator.

Superimposed segments are implemented when preprocessing the 3D-shape model of the object [4]. These segments often generate numerous grasp configurations. However, as numerous grasp configurations increase the calculation load of the manipulation planner, we reduce the number of grasp configurations by checking the collisions with the EEFs and evaluating the grasp stability, as described in the next subsection.

4.1. 3D Surface Model Processing for Grasp Planning

By calculating the convex hull of the model, its vertices, which are the potential rotational vertices used for pivoting, can be obtained (see Figure 4a). The rotational vertices are the supporting feet of objects during pivoting gait. In addition, the edges, which are the potential rotational edges used for tumbling, can be obtained from the convex hull.

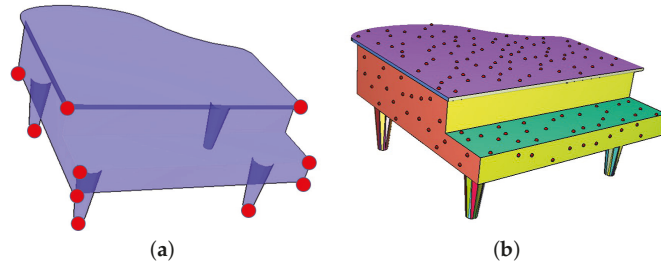


Figure 4. Object model analysis. (a) Vertices of the object (red points). (b) Contact points sampled on the object surface. The facets segmented by the superimposed segments are denoted by different colors. The contact points on the green surface are removed due to collision between the object and the EEF.

Superimposed segments are implemented to process a mesh model for contact and grasp planning. It analyzes the object model by peeling it into facets where the facets are allowed to be overlapped by each other. In addition to superimposed segments, the ray-shooting method [52] and simple segments [42] can also be used for analyzing mesh models. The ray-shooting method samples a contact point on the mesh surface and finds the candidate counter contact point by shooting a ray from the sampled point along the reversed normal direction. The simple segments method is similar to the superimposed segments while the difference lies in if the overlapping of facets is considered. The comparison among the three methods has been researched in [4] which showed that the superimposed segments can find more grasps while the time cost of it is faster than that of the ray-shooting method and is not significantly worse than that of the simple segments.

In superimposed segments, given a triangulated CAD model [53], a seed triangle is initialized and is compared with the surrounding triangles by evaluating the differences between surface normals of triangles. If the difference is smaller than a threshold, the neighboring triangle is clustered into the same facet. Otherwise, a new facet based on the neighboring triangle is built. After dealing with the initial facet, the algorithm selects a new seed and repeats the clustering until all the triangles are scanned. As a result, a set of facets are generated by the superimposed segments, where each face can be a candidate contact surface with the table in SP. When placing an object on the table, we define SP if the vertical projection of the object's CoM on the table is included in the support area. During SP, the robot is allowed to regrasp the object without influencing the stability of the object.

In addition, a set of contact points with the EEFs are computed by sampling the surface of the object model. To evenly distribute the contact points on the surface, sampling is initially performed over the entire surface. The sampled points are then repeatedly distributed as contact points on the superimposed facets. Further, some of the undesired sampling points are removed based on the following: (1) they are too close to the boundary of the facet, (2) they are close to each other, and (3) there is undesired collision between the object and the EEF. Examples of the sampled contact points are shown in Figure 4b.

4.2. Grasp Planning

In this study, a dual-arm robot with ball-shaped EEFs grasps the object at two contact points on nearly parallel facets with opposite contact normal vectors. By inspecting the

candidate contact points on these facets, the planner computes the potential contact pairs. Among several grasping point candidates, we further analyze and select a feasible one for a given object pose. We check whether each grasp point candidate leads to collision among the links of the robot or between a link and the environment. In addition, we check whether the inverse kinematics (IK) is solvable and whether the grasp stability can be satisfied by checking the wrench cone generated by the contacts with the EEFs and the environment.

Figure 5 shows examples of the selected contact points. Given an object configuration, the surface normals of the object are compared with those of the table. The contact pairs on surfaces with normals r_1 and r_2 are removed because collision occurs between the EEF and table when the EEF contacts the bottom of the object. On the other hand, the surface normals of the object are also compared with the EEF's rotational vector (e_1 and e_2). The contact pairs on surfaces with normals r_3 and r_4 are removed because they are too far to be reached by the EEF. As a result, only the contact pairs on the facets with surface normals g_1 and g_2 (green) remain.

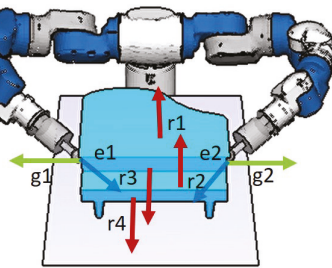


Figure 5. Selection of grasps with respect to the surface normals of the object and the rotational vector of the EEF (e_1 and e_2). The contact pairs on surfaces with normal vectors r_1 and r_2 are removed because collision occurs between the table and EEF if the EEF contacts the bottom of the piano. The contact pairs on surfaces with normal vectors r_3 and r_4 are removed because they are too far to be reached by the EEFs. The contact points on the surface with normal vectors g_1 and g_2 remain.

4.3. Grasp Stability

When pivoting, the object contacts two EEFs of the robot (see Figure 6). In addition, the object contacts the environment, such as the table. Considering these contacts, we can evaluate the grasp stability, which is the ability of an object to balance the external wrenches W_e by the force vector f applied at each contact point:

$$Gf = -W_e, \tag{1}$$

where G denotes the grasp map that maps the contact forces to the total object wrench. If an object contacts the environment with a vertex, we define $f = [f_0^T \ f_1^T \ f_2^T]^T$, where f_0 , f_1 , and f_2 are the force vectors applied by the environment and the two EEFs, respectively. On the other hand, if an object contacts the environment with two vertices during USP, one EEF will be sufficient to balance the external wrench. In this case, f_0 and f_1 are the force vectors applied by the environment, and f_2 is the force vector exerted by the EEF. This property allows us to regrasp the object in USP. Contact force $f_i = [f_{oi} \ f_{ii} \ f_{ni}]^T$, ($i = 0, 1, 2$) must lie within friction cone FC_i :

$$FC_i = \left\{ f_i : \sqrt{f_{oi}^2 + f_{ii}^2} \leq \mu_i f_{ni} \right\}, \tag{2}$$

where μ_i is the friction coefficient at the i -th contact point. In this study, we approximate the friction cone using a six-sided polyhedral cone. By calculating the Minkowski sum [54] of the force at each contact point, we obtain the grasp wrench space (GWS, W_s) [55].

$$W_s = \{w_s : w_s = \sum_{i=1}^3 Gf_i + w_g, f_i \in FC_i\}, \tag{3}$$

where w_g indicates the wrench generated by the gravitational force. The stability s can be evaluated by calculating the minimum distance between the origin in the wrench space and the convex hull of the set W_s . In this work, if the origin is inside of the GWS, the grasp is considered stable. In the physical world, the s indicates the ability to resist external wrenches W_e and it can be calculated by:

$$s = \min_{d \in \text{convexhull}(W_s)} \|d\|. \tag{4}$$

Based on the grasp stability, we select the stable grasp configurations.

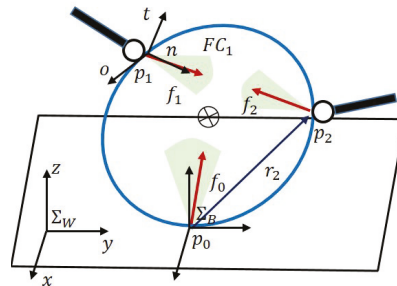


Figure 6. Object, EEFs, and environment.

5. Hierarchical Manipulation Planning

We plan the motions to manipulate an object on which information is available. We propose hierarchical planning, which includes task and motion level planning. The manipulation planner constructs a graph. In task level planning, we design the discrete object poses on a table, which along the grasp poses are saved in the nodes. In motion level planning, we plan the motions for moving the object and changing the grasp configurations. Primitive motions, such as pivoting, tumbling, and regrasping, are represented by the edges of the graph. Further, given the initial and goal configuration of the object, we search the graph and find a motion sequence to manipulate the object to the goal position.

Different from previous planners [2,3,19], we consider tumbling and regrasping in addition to the pivoting gait. Moreover, we consider the SP and USP of the object. Object motion in USP allows object movement within a limited table area. However, the object may easily fall if the robot regrasps it in USP. To avoid this problem, we design the regrasping motion in USP, such that the robot sequentially changes the contact position of each EEF.

5.1. Task Level Planning

In task level planning, we sample the object poses on a table, and save information on the object configurations and the related grasp configurations in the nodes. The object poses are not sampled randomly on the table because pivoting includes a kinematic constraint that the object must be rotated around a fixed point on the table, which is also the object’s rotational vertex. In the graph, we sample new object poses with respect to the pivoting and tumbling motions (see Figure 7). In pivoting motion, we discretize the object pose when rotating about the vertical line including the object vertex. The rotation is discretized in θ_t intervals. We repeat such sampling for all the vertices in the current base surface. By sequentially changing the rotational vertices, the object can be moved through pivoting gait along a certain direction (see Figure 8).

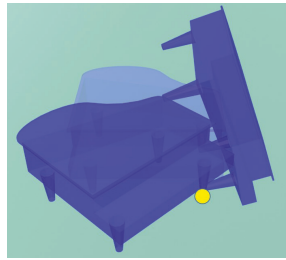


Figure 7. Sampling new object configurations through pivoting and tumbling at one object vertex.

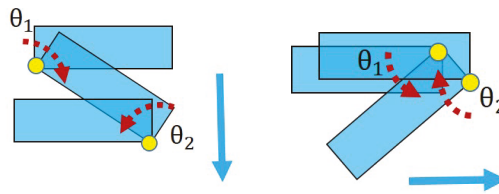


Figure 8. Examples of moving an object to certain direction through pivoting gait. The red dashed arrows indicate rotating directions. θ_i indicates the amount of yaw rotation in the i -th sequence.

To sample new object configurations by tumbling, the object is rotated around an object edge until the object moves from one SP to another (see Figures 2b and 7). Tumbling enables a change in the support surface, which provides more choice in the path planning of the object, in case an obstacle is present (see Figure 3).

In this study, most of the object poses are designed in SP. However, within a limited table area, such as the boundary of the table, it is difficult to place the object in SP. For such an area, we sample the object pose in USP where an EEF of the robot assists in holding the object. To retain an object in USP, we check the grasp stability, discussed in Section 4.3, to select the contact points of the EEFs. Here, it is to be noted that sampling the object poses in USP corresponds to pivoting the object assuming the double support (DS) gait mode, as discussed in [3]. On the other hand, sampling the object poses in SP corresponds to pivoting the object assuming the quadruple support (QS) gait mode. Figure 9 shows the difference between the DS and QS gait modes. Refer [3] for the details on the DS and QS gait modes.

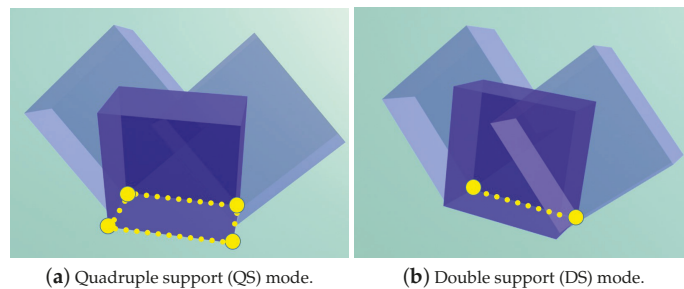


Figure 9. Changes in the object’s rotational vertices (a) in the QS mode when four vertices are in contact with the table and (b) in the DS mode when two vertices are in contact with the table.

Figure 10a shows the sampled object poses on the table. Each node indicates an object pose; the red ones denote SP, whereas the blue ones denote USP. The red and blue edges indicate pivoting and tumbling motion, respectively. For better manipulation performance in the real physical world, we also evaluate the change in the angle between the EEF and the contact surface normal during pivoting motion.

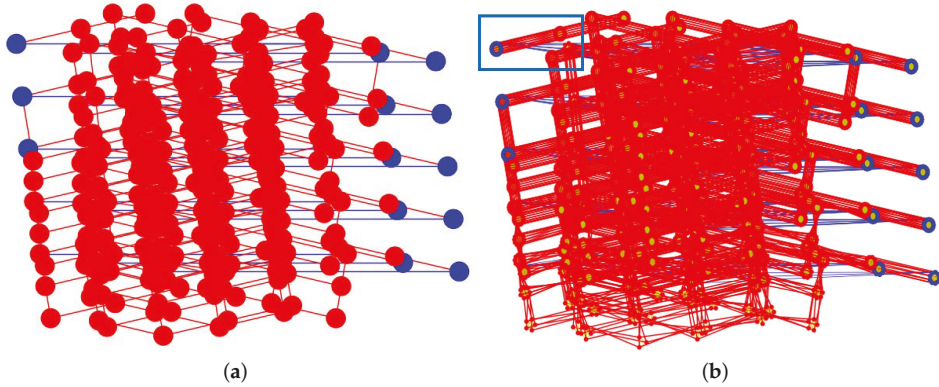


Figure 10. The graphs generated at the task level. The object poses are first sampled and shown in the left graph, then the right graph is built by combining grasp configurations. (a) Graph of the sampled object poses. The red nodes indicate the object poses in SP, whereas the blue nodes indicate USP. The red and blue edges indicate pivoting and tumbling motion, respectively. (b) Grasps combined with the graph shown in (a). Each node includes information on both the object pose and grasp configuration. Yellow edges are added, which indicate the regrasping to change the grasp poses.

Combining the corresponding grasp configurations with the object configuration, we can generate the graph shown in Figure 10b. In this graph, each node includes information on the grasp configurations and the object pose related to Figure 10a.

The detailed explanations of the graph are shown in Figure 11, where the blue rectangular part in Figure 10b is amplified. A node contains information on both an object and a grasp configuration. Nodes inside a light blue circle (which is drawn only for explanations in Figure 11 and is not included in the graph) indicate they share the same object pose but different grasp configurations. Three primitive motions are represented by edges where yellow, red, and blue edges imply regrasping, pivoting, and tumbling, respectively. The configurations saved in nodes can be transferred by the edges connecting them. For example, in Figure 11b, the grasp configurations of two nodes are changed by a yellow edge which implies regrasping. The design of SP and USP is also reflected in the graph where blue nodes indicate USP, see Figure 11b and red nodes indicate SP, see Figure 11c,d. What is more, there are fewer nodes within the right blue circle in (a) because there are fewer grasps as the contact surface in the current object pose is small, see Figure 11d.

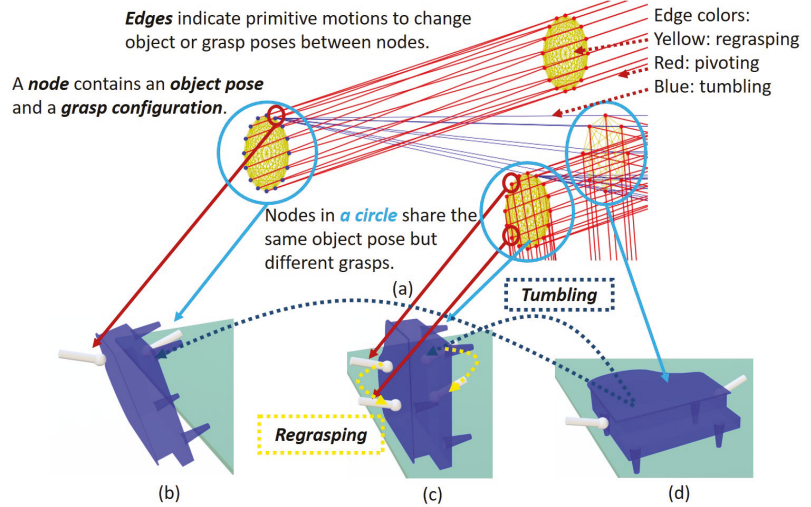


Figure 11. (a) Portion of the graph in the rectangle depicted in Figure 10b. The nodes within the light blue circle share the same object pose and each node indicates the grasp configuration for the object pose. Nodes are connected by edges which correspond to primitive motions. (b) An object pose and a corresponding grasp pose. (c) Two grasp poses can be changed by the yellow edge connecting them. (d) The object pose in (d) can be changed to (b) or (c) by tumbling.

5.2. Motion Level Planning

After sampling the object poses on the table and generating the corresponding grasps, the motions to move the object and change the grasp poses are generated in motion level planning. The primitive motions include the following: (i) pivoting motion, which is a transfer motion to change the object configuration without changing the grasps, (ii) tumbling motion, which includes both transfer and transit motions to rotate the object around one edge and change its base surface, and (iii) regrasping of an object in SP involving transit motion, which changes the grasp configurations without affecting the object configuration, and regrasping of an object in USP, which involves both transit and transfer motions.

5.2.1. Pivoting

Pivoting motion is designed by raising the object up on a vertex, rotating it around the vertex, and placing it down. A property of pivoting motion is that two object poses must share the same rotational vertex. Figure 8 displays 2D examples in which pivoting gait is used to move an object in the direction indicated by arrows.

In Figure 6, the contact point between the object and table is denoted by p_0 . Here, we assume point contact with friction at each contact point. Due to contact with table at a fixed vertex (p_0), the object motion is constrained as follows:

$$SD_{B0} \begin{bmatrix} \dot{p}_B \\ \omega_B \end{bmatrix} = S \begin{bmatrix} \dot{p}_0 \\ \omega_0 \end{bmatrix} = o. \tag{5}$$

Contact between the object and i -th EEF ($i = 1, 2$) is constrained as follows:

$$SD_{Bi} \begin{bmatrix} \dot{p}_B \\ \omega_B \end{bmatrix} = SD_{Hi} \begin{bmatrix} \dot{p}_{Hi} \\ \omega_{Hi} \end{bmatrix} = \dot{p}_i, \tag{6}$$

where S is a selection matrix that selects the linear velocity and D_{Bi} transforms the linear and angular velocity from Σ_B to Σ_W . Details on the S and D matrices can be found in our

previous work [3]. The object is accelerated by the force (f_1, f_2) applied by the two EEFs. The dynamic of the object’s rotational motion can be obtained as

$$r_1 \times f_1 + r_2 \times f_2 + r_{com} \times m_o g = \mathcal{I}_o \dot{\omega}_B + \omega_B \times \mathcal{I}_o \omega_B, \tag{7}$$

where $r_i = p_i - p_0$, $(i = 1, 2)$, and $r_{com} = p_{com} - p_0$ where p_{com} denotes the position vector of the object’s CoM. m_o and \mathcal{I}_o indicate the mass and moment of inertia of the object, respectively. ω_B and $\dot{\omega}_B$ indicate the angular velocity and angular acceleration of the object in frame Σ_B , respectively.

In the motion level, the robot motions to move the object between the object poses designed in the nodes are generated by MPC with respect to the kinematics (5), (6), and the dynamic (7). Figure 12b depicts an example of the generated motions for pivoting a piano. In MPC, we define the state vector as $x_k = [\Psi_B \ \omega_B]_k^T$ where Ψ_B indicates the Euler angles of the object in Σ_B and select the input vector as $u_k = [f_1 \ f_2]_k^T$. During the object’s rotation on the vertex p_0 , the prediction of the state is obtained by,

$$x_{k+1} = Ax_k + Bu_k + D, \tag{8}$$

where A, B , and D are coefficient matrices defined as

$$A = \begin{bmatrix} I_3 & W \\ O_3 & I_3 \end{bmatrix}, \quad B = \begin{bmatrix} \mathcal{I}_o^{-1}WT^2/2[r_1 \times] & \mathcal{I}_o^{-1}WT^2/2[r_2 \times] \\ \mathcal{I}_o^{-1}T[r_2 \times] & \mathcal{I}_o^{-1}T[r_2 \times] \end{bmatrix}, \tag{9}$$

$$D = \begin{bmatrix} \mathcal{I}_o^{-1}WT^2/2[r_{com} \times]mg \\ \mathcal{I}_o^{-1}T[r_{com} \times]mg \end{bmatrix}, \tag{10}$$

where the matrix W transforms the angular velocity to the velocity of the Euler angles.

The future states and free variables appear in the prediction horizon (N_p) are defined as $X_k = [x_k, x_{k+1}, \dots, x_{k+n_p-1}]^T$ and $U_k = [u_k, u_{k+1}, \dots, u_{k+n_p-1}]^T$, respectively. The MPC tracks a reference by solving the following optimization problem,

$$J_{mpc} = \frac{\alpha}{2} \|X_{k+1} - X^{ref}\|^2 + \frac{\beta}{2} \|U_k\|^2, \tag{11}$$

where α and β are the weights. X^{ref} is the reference trajectory of the object and it is provided by the object configuration saved in nodes of the graph. The first and the second terms in (11) evaluate the state error and the amount of the manipulating force of the EEF, respectively.

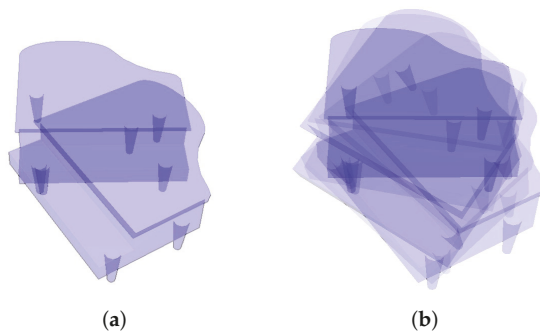


Figure 12. Example of pivoting motion to move a piano. (a) Two poses of the piano. (b) Transformation of the object poses shown in (a) through pivoting motion. The object is tilted, rotated around one vertex, and then placed down.

5.2.2. Tumbling and Regrasping

Changing the base surface of the object improves the obstacle avoidance performance. The object's base surface is the support surface between the object and the table. The proposed method has better collision-avoidance capability since pivoting gait is performed with the combination of tumbling and regrasping. By tumbling, we can control the area of support polygon and it contributes to the case when a manipulated object passes through a narrow passage. In addition, by regrasping, we may avoid the collision between the robotic arms and the environment. An example of the object motion is shown in Figure 13, where the gaits 1 cannot avoid the collision with the red obstacle since the object is not tumbled. If the object is tumbled to the right as shown in the right of Figure 13, the supporting polygon can be changed. Such a change allows the object to be moved without colliding the obstacle by performing the gaits 2. Tumbling motion contributes to change the type of gaits adaptively according to the environment.

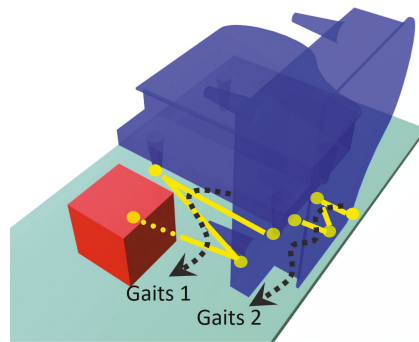


Figure 13. A change of the object's supporting base improves the ability of collision avoidance. The object motion following the gaits 1 cannot avoid a collision with the red obstacle. If the object is tumbled, the supporting base will be changed. Such a change allows the object to be moved without colliding the obstacle by performing the gaits 2. To pass a narrow passage, gaits 2 is more area-efficient than gaits 1.

For changing the object's base surface, we design tumbling motion, which is a special case of pivoting when the rotation passes through the object's edge. Rotating an object up to SP is simple, where one EEF rotates the object around one edge (the line connecting the two supporting vertices) of the object until it moves to SP. The kinematics and dynamics of tumbling motion can be derived through minor modifications in (5)–(11). However, rotating an object up to USP is more complex. This is because to maintain an object in USP, the EEF must contact and hold the object, which leads to difficulty in changing the grasp poses. To solve this problem, we incorporate multiple arms to achieve regrasping by sequentially using different arms to contact the object. Figure 14 depicts a tumbling motion where the object is rotated to USP in the boundary of the table. If the right EEF does not contact the object, it will fall from the table. The tumbling motion is designed as follows: First, the left EEF contacts and rotates the piano (see motion l1). Meanwhile, the right EEF moves to the desired position, which is also the right EEF's grasp pose for contacting the object in USP (see motion r1). The left EEF rotates the object until it contacts the right EEF. Next, the right EEF alone holds the object. The stability is evaluated by calculating the contact wrench of the contact between the right EEF and the object, as well as that between the object and table, as mentioned in Section 4.3. The left EEF then leaves the object and moves to the desired grasp configuration (see motion l2). Compared to regrasping with a gripper, which can only regrasp an object in SP, a multi-arm robot can regrasp an object in both SP and USP.

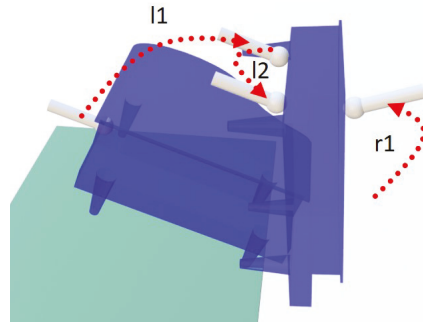


Figure 14. Tumbling motion by incorporating two manipulators.

In addition to regrasping an object in USP, regrasping motion in SP is also designed. When an object is in SP, we can easily plan the motions for regrasping without affecting the object pose. The regrasp motions are denoted by the yellow edges in Figures 10b and 11a.

5.3. Graph Searching

Given the initial and goal poses of the object, grasps of these object poses can be determined by data provided by object analysis introduced in Section 4. Then, we look for the same object and grasp configurations that are embedded in nodes of the graph. If the new object poses and their grasp configurations can not be found in the graph, new nodes will be created and be connected with the graph. The reachability of new nodes can be guaranteed by the controllability of pivoting motion [18], which proved that a sequence of three pivoting motions can move the object to an arbitrary object configuration with the same base surface in the neighborhood.

Weights are manually assigned to edges in the graph. As the object can be efficiently moved by pivoting and we expect the robot finishes the placement of the object quickly, we set the weight of edges representing pivoting to a small value 0.1. The weight of edges indicating regrasping is set to 0.5 and the weight of edges indicating tumbling is set to 1. A relative large weight is designed to edges related to tumbling because the tumbling is mainly designed for changing the object's supporting base instead of moving the object. Knowing the initial and goal configurations, we search the weighted graph to find a path. A solution path includes sequences of discrete object and grasp configurations. Intermediate configurations are added to ensure a smooth motion during pivoting and tumbling by the MPC introduced in Section 5.2. In a lower-level motion planning, we use rapidly exploring random tree (RRT) to sample the space considering grasp configurations. At this level, if configurations with feasible inverse kinematics and obstacle clearance cannot be found, we go back to the graph, remove certain nodes and edges which caused the problem and re-search the graph for a new path.

6. Simulation and Experiments

In this study, the target object is a toy piano sized $0.425 \times 0.45 \times 0.205$ m with a weight of 3.1 kg. The piano is placed on a table, which is in front of the robot, and the size of the table surface is 0.9×0.6 m. We use a dual-arm Yaskawa Motoman SDA5F robot to manipulate the object. The two EEFs of the robot are the same and ball-shaped. They are produced by a 3D printer where the diameter of the ball is 4 cm. In addition, to increase the friction between the EEFs and the object, anti-slip stickers are added to the tip of EEFs.

6.1. Object Analysis

A mesh piano model is processed by the superimposed segments offline. The computational costs and the results of the analysis of the mesh model are shown in Table 1. The results are obtained by executing the process on a desktop PC where the CPU is Intel Core i7-9700K @ 3.60 GHZ and a memory in size of 8 GB. The program is performed using Python 3.6.

Table 1. Performance of superimposed segments.

Process	Computational Costs	Results	Number
Superimposed segments	0.354 s	Faces of the object	262
Sampling	0.075 s	Contact points	388
Refine samples and plan contact pairs	2.115 s	Contact pairs	107

The number of faces of the object generated by the superimposed segments is 262 which is large. This is because curvatures of the object, for example, the legs of the piano, can be separated to many faces, see Figure 4b. It is a time-consuming process to refine samples and plan contact pairs where bad samples are removed due to the rules mentioned in Section 4.1 and collision checking between the EEFs and the object. The time cost is reasonable because a mesh-to-mesh collision detection is implemented. After the refinement, contact pairs are created based on the remaining samples.

Contact pairs under a given object configuration can be refined by grasp stability and collision checking between the EEFs and the environment by the rules discussed in Sections 4.2 and 4.3. In Figure 10b, the total number of sampled object configurations is 310 and the number of feasible grasp configurations after the refinement is 4207 where the process of the refinement costed 9.362 s. The time cost of the refinement is relatively high but acceptable since we only need to do it once and offline. If the refinement is not implemented, the evaluation of the grasp stability will be performed 33,170 (107×310) times and cost over one minute.

6.2. Simulation

In the simulation, the target is to move the piano to the goal pose, which is far from the robot (see Figure 15). The robot first grasps the piano placed on the table (see Figure 15a). Further, it pivots the piano around one vertex under the front right leg (see Figure 15b). Here, we define the left and right sides of the object and those of the robot according to the view from the robot. In Figure 15c, the piano is placed on the table and the rotational vertex is changed from right to left. Then, through pivoting motion, the piano is placed in the pose shown in Figure 15d. Note that both arms of the robot are close to their kinematic limits and the robot cannot further move the object forward. However, the robot changes its grasp configuration and grasps the rear of the object (see Figure 15e,f). Thereby, the robot can pivot the piano to move it forward and successfully place it in the goal position (see Figure 15g,h). By changing the grasp configuration, the piano is moved forward by as much as 12.5 cm (see Figure 15d,h).

This simulation demonstrates that regrasping motion extends the scope of feasible placements of the object manipulated by the robot.

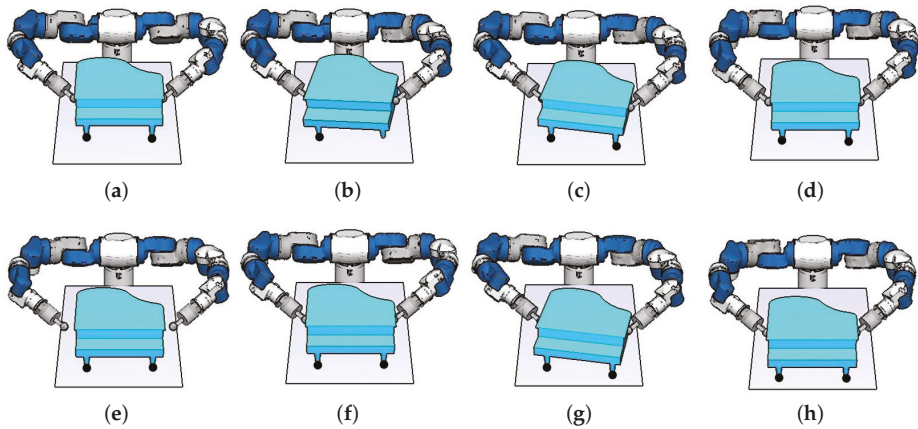


Figure 15. Manipulation of the piano by the robot for moving it forward by pivoting. The support vertices are marked as black dots. The initial configurations are shown in (a). The robot manipulates the object by pivoting in (b–d). When the joint configurations are close to the limits (d), regrasping is performed in (d–f). After regrasping, the robot moves the object to the goal pose, see (f–h).

6.3. Experiment 1: Pivoting Gait

In the first experiment, we perform the simulated pivoting gait in a real scenario. The inputs are the initial and target poses of the piano. The system automatically plans the grasp, pivoting motion, and regrasping motion. Figure 16 depicts the experimental process. The robot first grasps the piano (see Figure 16a) and pivots it around the right (from the robot view) rotational vertex (see Figure 16b). Further, the robot changes the rotational vertex to the left (see Figure 16c) and pivots the piano (see Figure 16d,e). In Figure 16e, both the arms are close to the limit of the robot workspace; hence, the robot cannot further move the piano forward. Pivoting gaits generated by the graph MPC [3] faces such limitations and the object cannot be placed in the target location. To solve this problem, regrasping is planned in this work and the robot changes the grasp poses to grasp the rear part of the object (see Figure 16f,g) and successfully pivots the piano to the target pose (see Figure 16h–j). Table 2 compares the performances of two methods where only the proposed planner can generate motions to move the object to the target location.

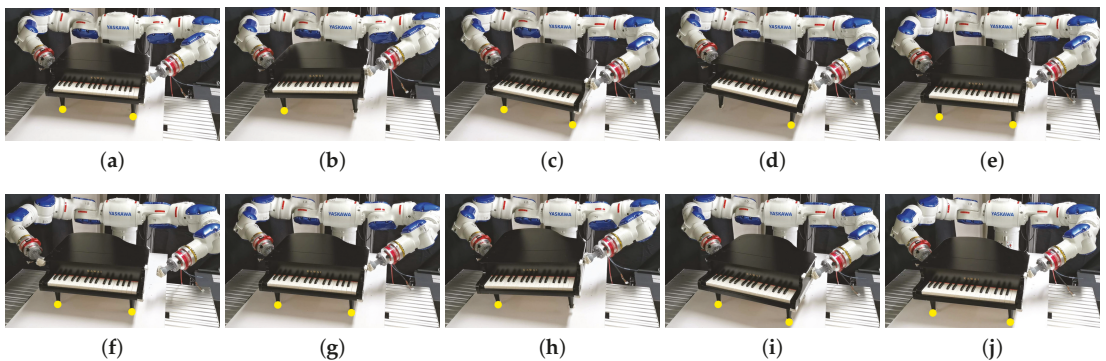


Figure 16. Experiment 1: The robot manipulates the object by pivoting in (a–d) and regrasping the object in (e–g). In (e), without regrasping, the robot cannot further move the object forward because the robot arms are close to the kinematic limit. However, after regrasping, the robot can move the object forward for two pivoting steps, see (h–j). The yellow dots indicate the object vertices that contact the table.

Table 2. Performances of the graph MPC and the proposed planner.

Methods for Generating Motions	Ability to Regrasp	Error (x-axis)
The graph MPC	No	−12.5 cm
The proposed planner	Yes	0

In the experiment, after moving the piano a few steps, the robot cannot further move the object with the initial grasp poses because of the robot workspace. To continue moving the object, the robot finds a new grasp configuration, regrasps the piano, and manipulates it to the target location. Compared with the pivoting gaits generated by the graph MPC [3], the regrasping motion enables further motion of the object by as much as 12.5 cm along the x -axis.

6.4. Experiment 2: Object Orientation

Experiment 2 tests a combination of regrasping and pivoting during the rotation of the piano around the z -axis in the world frame (see Figure 17). The initial pose of the piano is shown in Figure 17a and the support vertices are marked as yellow dots. The robot grasps the piano, pivots it around one of its support vertices, and places it down in a pose where a rotation of -45 degrees in the z -axis is performed (see Figure 17a–c). Further, the robot changes the grasp poses to different contact surfaces as depicted in Figure 17d,e to avoid self-collision of the robot and maintain the EEF in a reachable pose. After regrasping, the robot rotates the piano by another -45 degrees around the z -axis and places it down (see Figure 17f). In Figure 17g,h, a new regrasping motion is performed after which the robot pivots the piano to the target pose (see Figure 17i,j).

Experiment 2 shows that to perform a relatively large change in the object orientation, the robot needs to contact different surfaces of the object; with a combination of regrasping and pivoting, the robot successfully manipulates the object to the target pose.

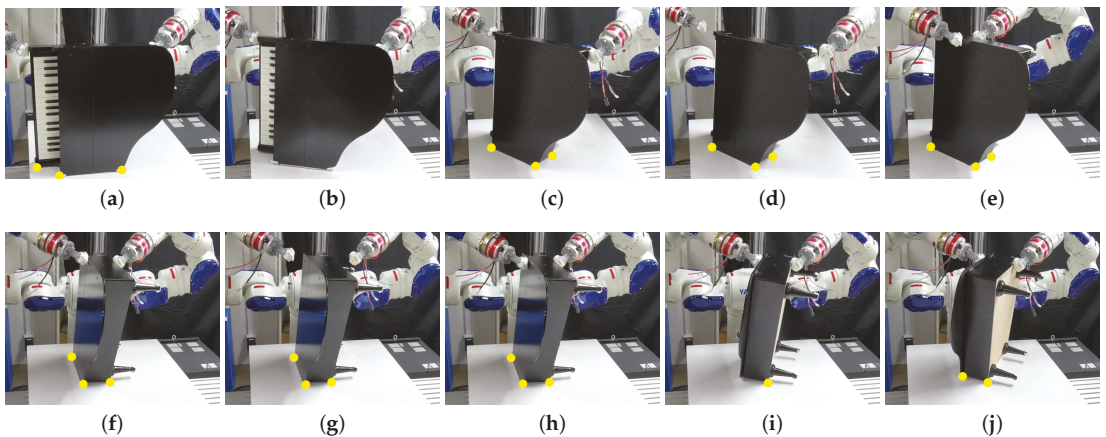


Figure 17. Experiment 2: the robot changes the orientation of the object by pivoting and regrasping. The robot first rotates the object by pivoting in (a–c). Then the robot regrasps the object at its new contact surfaces, see (c–e) and pivots the object, see (e,f). In (f–h), the robot changes the grasp configurations and finally pivots the object to the goal pose, see (h–j).

6.5. Experiment 3: Obstacle Avoidance

In the third experiment, the robot moves the piano to the target pose considering the environment where an obstacle exists and the limited support surface (see Figure 18). Different from the sheering method [2], which cannot find a feasible path to avoid collision, changing the object's base surface by tumbling can realize better collision avoidance be-

tween the object and the obstacle. Moreover, the USP design allows the robot to manipulate the object within a narrow support surface without falling.

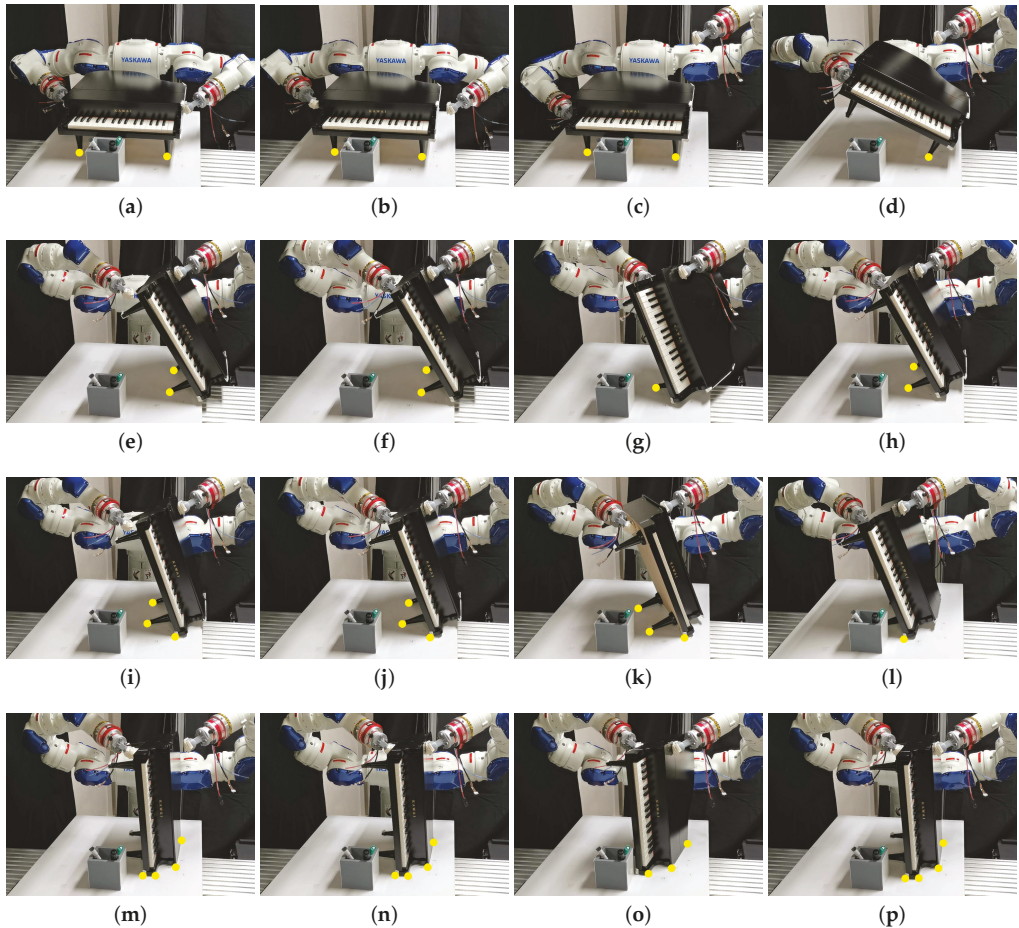


Figure 18. Experiment 3: An obstacle is placed in front of the object, see (a,b). The robot tumbles the object using the robot’s right EEF, whereas the left EEF moves to the desired position and waits for contacting the object (c,d). After the left EEF holds the object, the right EEF moves to the desired grasp pose (e,f). Note that the object is close to the boundary of the table and it is in a USP pose where the object will fall if there is no support from the EEF. After regrasping, the robot moves the object by pivoting it in the DS mode (g,h) and then places it in SP (i). The robot moves the object in the QS mode (j,k) and tumbles the object to change its support surface (l,m). Finally, the robot regrasps (m,n) and moves the object to the goal position (o,p). There is no collision between the object and obstacle during these motions.

A grey pen holder is placed in front of the piano as an obstacle (see Figure 18a). The poses of the obstacle and the piano are input to the system. As the obstacle is very close to the piano, the piano cannot avoid collision by performing the pivoting gait using the sheering method. Therefore, the robot tries to tumble the piano. The right EEF moves to a new grasp configuration and prepares for the tumbling motion, whereas the left EEF moves to the planned grasp configuration preparing to catch and hold the object after rotation (see Figure 18b,c). The robot utilizes the right EEF to rotate the piano (see Figure 18d). Once the piano is in contact with the left EEF, the right EEF leaves the piano and moves to a new

grasp pose (see Figure 18e,f). As a result, after the rotation, regrasping is performed by sequentially incorporating the right and left EEFs to hold the piano, which is an advantage of the dual-arm robot. Note that as the supporting table is narrow and the piano is near the table boundary, sufficient space is not available for the piano to be placed in SP and the left EEF is used to keep the piano in USP. For example, if the left EEF does not hold the piano, the piano will fall from the table (see Figure 18e). The USP design allows the robot to manipulate the object within a limited support space. Subsequently, the robot performs the pivoting gait designed for the USP, which is the DS gait mode (see Figure 18g,h). The robot places the piano in SP when there is sufficient space on the table, and then changes the grasp configurations (see Figure 18i,j). After regrasping, the robot pivots the piano for two steps along the y-direction (see Figure 18k,l) and then tumbles the piano to change the support surface through the right EEF (see Figure 18m). Finally, the robot regrasps the object and pivots it for two steps along the x-direction, and places it at the target location (see Figure 18n–p). During the entire process, there is no collision between the object and the obstacle.

Experiment 3 demonstrates that changing the object's base surface improves the collision avoidance performance in pivoting gait. In addition, the USP design enables the robot to move the object within a limited support space. Moreover, the multiple arms incorporated to sequentially contact the object achieve regrasping in USP.

7. Conclusions

In this study, we proposed a manipulation plan for moving a general-shaped large object. With superimposed segments, the object model was first analyzed, and the contact between the object and EEFs, as well as that between the object and environment were determined. Utilizing the object's contact with the environment, we realized the manipulation of a bulky object through pivoting, tumbling, and regrasping. In the plan, the object configurations in both SP and USP were considered, enabling the robot to manipulate the object within a limited support area. The proposed approach takes advantage of the kinematic, dynamic, and gait modes of the pivoting gait published in our previous work [3], which is related to motion planning for moving the object in both SP and USP. In this study, this pivoting gait is further improved by combining regrasping and tumbling. The efficiency of the proposed method was demonstrated through experiments in which the robot moved an object to the goal configuration, avoiding the kinematic limits of the robot arms and exhibiting better collision avoidance performance compared to those reported in prior works. The proposed approach can be employed to automate the manipulation of large objects by dual-arm manipulators. In future, we intend to manipulate more complex-shaped objects, using one EEF to perform pivoting, and planning motions in more complex environment for testing the planner's performance. Quantitative analysis based on computational costs, stability, number of gaits, tumbles and regrasps to finish a task will be researched.

Author Contributions: The authors' contributions are reported as follows: Original concept, A.Z. and K.H.; Methodology, A.Z., K.K., W.W. and K.H.; Software implementation, A.Z. and W.W.; Technical implementation, A.Z.; Writing—Original Draft Preparation, A.Z.; Writing—Review and Editing, A.Z. and K.H.; Supervision, K.K., W.W. and K.H.; Project Administration, K.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Yoshida, E.; Blazevic, P.; Hugel, V.; Yokoi, K.; Harada, K. Pivoting a large object: Whole-body manipulation by a humanoid robot. *Appl. Bionics Biomech.* **2006**, *3*, 227–235. [\[CrossRef\]](#)
2. Yoshida, E.; Poirier, M.; Laumond, J.P.; Kanoun, O.; Lamiroux, F.; Alami, R.; Yokoi, K. Pivoting based manipulation by a humanoid robot. *Auton. Robot.* **2010**, *28*, 77–88. [\[CrossRef\]](#)
3. Zhang, A.; Koyama, K.; Wan, W.; Harada, K. Controlling Pivoting Gait Using Graph Model Predictive Control. *IEEE Access* **2021**, *9*, 73757–73770. [\[CrossRef\]](#)
4. Wan, W.; Harada, K.; Kanehiro, F. Planning Grasps With Suction Cups and Parallel Grippers Using Superimposed Segmentation of Object Meshes. *IEEE Trans. Robot.* **2020**, *37*, 166–184.
5. Lynch, K.M.; Mason, M.T. Dynamic nonprehensile manipulation: Controllability, planning, and experiments. *Int. J. Robot. Res.* **1999**, *18*, 64–92. [\[CrossRef\]](#)
6. Chavan-Dafle, N.; Holladay, R.; Rodriguez, A. Planar in-hand manipulation via motion cones. *Int. J. Robot. Res.* **2020**, *39*, 163–182. [\[CrossRef\]](#)
7. Sawasaki, N.; INOUE, H. Tumbling objects using a multi-fingered robot. *J. Robot. Soc. Jpn.* **1991**, *9*, 560–571. [\[CrossRef\]](#)
8. Trinkle, J.C. On the stability and instantaneous velocity of grasped frictionless objects. *IEEE Trans. Robot. Autom.* **1992**, *8*, 560–572. [\[CrossRef\]](#)
9. Erdmann, M.A.; Mason, M.T. An exploration of sensorless manipulation. *IEEE J. Robot. Autom.* **1988**, *4*, 369–379. [\[CrossRef\]](#)
10. Satici, A.C.; Ruggiero, F.; Lippiello, V.; Siciliano, B. A coordinate-free framework for robotic pizza tossing and catching. In Proceedings of the 2016 IEEE International Conference on Robotics and Automation (ICRA), Stockholm, Sweden, 16–21 May 2016; pp. 3932–3939.
11. Cigiano, P.; Lippiello, V.; Ruggiero, F.; Siciliano, B. Robotic ball catching with an eye-in-hand single-camera system. *IEEE Trans. Control Syst. Technol.* **2015**, *23*, 1657–1671. [\[CrossRef\]](#)
12. Sun, Y.; Xiong, R.; Zhu, Q.; Wu, J.; Chu, J. Balance motion generation for a humanoid robot playing table tennis. In Proceedings of the 2011 11th IEEE-RAS International Conference on Humanoid Robots, Bled, Slovenia, 26–28 October 2011; pp. 19–25.
13. Ramirez-Alpizar, I.G.; Higashimori, M.; Kaneko, M.; Tsai, C.H.; Kao, I. Nonprehensile dynamic manipulation of a sheet-like viscoelastic object. In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 5103–5108.
14. Hou, Y.; Jia, Z.; Mason, M.T. Fast planning for 3d any-pose-reorienting using pivoting. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 1631–1638.
15. Aiyama, Y.; Inaba, M.; Inoue, H. Pivoting: A new method of grasps manipulation of object by robot fingers. In Proceedings of the 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS'93), Yokohama, Japan, 26–30 July 1993; Volume 1, pp. 136–143.
16. Doshi, N.; Hogan, F.R.; Rodriguez, A. Hybrid differential dynamic programming for planar manipulation primitives. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May 31–31 August 2020; pp. 6759–6765.
17. Raessa, M.; Wan, W.; Harada, K. Planning to repose long and heavy objects considering a combination of regrasp and constrained drooping. *Assem. Autom.* **2021**. [\[CrossRef\]](#)
18. Yoshida, E.; Poirier, M.; Laumond, J.P.; Alami, R.; Yokoi, K. Pivoting based manipulation by humanoids: A controllability analysis. In Proceedings of the 2007 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Diego, CA, USA, 29 October–2 November 2007; pp. 1130–1135.
19. Yoshida, E.; Poirier, M.; Laumond, J.P.; Kanoun, O.; Lamiroux, F.; Alami, R.; Yokoi, K. Regrasp planning for pivoting manipulation by a humanoid robot. In Proceedings of the 2009 IEEE International Conference on Robotics and Automation, Kobe, Japan, 12–17 May 2009; pp. 2467–2472.
20. Shi, F.; Zhao, M.; Murooka, M.; Okada, K.; Inaba, M. Aerial Regrasping: Pivoting with Transformable Multilink Aerial Robot. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 200–207.
21. Fakhari, A.; Patankar, A.; Chakraborty, N. Motion and Force Planning for Manipulating Heavy Objects by Pivoting. *arXiv* **2020**, arXiv:2012.06022.
22. Murooka, M.; Nozawa, S.; Bando, M.; Yanokura, I.; Okada, K.; Inaba, M. Simultaneous planning and estimation based on physics reasoning in robot manipulation. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, Australia, 21–25 May 2018; pp. 3137–3144.
23. Murooka, M.; Ueda, R.; Nozawa, S.; Kakiuchi, Y.; Okada, K.; Inaba, M. Global planning of whole-body manipulation by humanoid robot based on transition graph of object motion and contact switching. *Adv. Robot.* **2017**, *31*, 322–340. [\[CrossRef\]](#)
24. Siméon, T.; Laumond, J.P.; Cortés, J.; Sahbani, A. Manipulation planning with probabilistic roadmaps. *Int. J. Robot. Res.* **2004**, *23*, 729–746. [\[CrossRef\]](#)
25. Lozano-Pérez, T.; Kaelbling, L.P. A constraint-based method for solving sequential manipulation planning problems. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 3684–3691.

26. Srivastava, S.; Fang, E.; Riano, L.; Chitnis, R.; Russell, S.; Abbeel, P. Combined task and motion planning through an extensible planner-independent interface layer. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 639–646.
27. Lee, G.; Lozano-Pérez, T.; Kaelbling, L.P. Hierarchical planning for multi-contact non-prehensile manipulation. In Proceedings of the 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Hamburg, Germany, 28 September–3 October 2015; pp. 264–271.
28. Suárez-Ruiz, F.; Zhou, X.; Pham, Q.C. Can robots assemble an IKEA chair? *Sci. Robot.* **2018**, *3*, eaat6385. [\[CrossRef\]](#)
29. Tournassoud, P.; Lozano-Pérez, T.; Mazer, E. Regrasping. In Proceedings of the 1987 IEEE International Conference on Robotics and Automation, Nice, France, 12–14 May 1987; Volume 4, pp. 1924–1928.
30. Berenson, D.; Srinivasa, S.; Kuffner, J. Task space regions: A framework for pose-constrained manipulation planning. *Int. J. Robot. Res.* **2011**, *30*, 1435–1460. [\[CrossRef\]](#)
31. Bouyarmane, K.; Kheddar, A. Humanoid robot locomotion and manipulation step planning. *Adv. Robot.* **2012**, *26*, 1099–1126. [\[CrossRef\]](#)
32. Harada, K.; Tsuji, T.; Laumond, J.P. A manipulation motion planner for dual-arm industrial manipulators. In Proceedings of the 2014 IEEE International Conference on Robotics and Automation (ICRA), Hong Kong, China, 31 May–7 June 2014; pp. 928–934.
33. Hayashi, N.; Suehiro, T.; Kudoh, S. Planning method for a wrapping-with-fabric task using regrasping. In Proceedings of the 2017 IEEE International Conference on Robotics and Automation (ICRA), Singapore, 29 May–3 June 2017; pp. 1285–1290.
34. Wan, W.; Harada, K.; Kanehiro, F. Preparatory manipulation planning using automatically determined single and dual arm. *IEEE Trans. Ind. Inform.* **2019**, *16*, 442–453. [\[CrossRef\]](#)
35. Mason, M.; Salisbury, K. *Robot Hands and the Mechanics of Manipulation (Artificial Intelligence)*; The MIT Press: Cambridge, MA, USA, 1985.
36. Nguyen, V.D. Constructing force-closure grasps. *Int. J. Robot. Res.* **1988**, *7*, 3–16. [\[CrossRef\]](#)
37. Liu, Y.H. Computing n-finger form-closure grasps on polygonal objects. *Int. J. Robot. Res.* **2000**, *19*, 149–158. [\[CrossRef\]](#)
38. Ponce, J.; Sullivan, S.; Sudsang, A.; Boissonnat, J.D.; Merlet, J.P. On computing four-finger equilibrium and force-closure grasps of polyhedral objects. *Int. J. Robot. Res.* **1997**, *16*, 11–35. [\[CrossRef\]](#)
39. Montana, D.J. *The Condition for Contact Grasp Stability*; IEEE International Conference on Robotics and Automation (ICRA): Sacramento, CA, USA, 9 April 1991; pp. 412–417.
40. Bicchi, A. On the closure properties of robotic grasping. *Int. J. Robot. Res.* **1995**, *14*, 319–334. [\[CrossRef\]](#)
41. Howard, W.S.; Kumar, V. On the stability of grasped objects. *IEEE Trans. Robot. Autom.* **1996**, *12*, 904–917. [\[CrossRef\]](#)
42. Harada, K.; Tsuji, T.; Nagata, K.; Yamanobe, N.; Maruyama, K.; Nakamura, A.; Kawai, Y. Grasp planning for parallel grippers with flexibility on its grasping surface. In Proceedings of the 2011 IEEE International Conference on Robotics and Biomimetics, Phuket, Thailand, 7–11 December 2011; pp. 1540–1546.
43. Hang, K.; Stork, J.A.; Kragic, D. Hierarchical fingertip space for multi-fingered precision grasping. In Proceedings of the 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, Chicago, IL, USA, 14–18 September 2014; pp. 1641–1648.
44. Romano, J.M.; Hsiao, K.; Niemeyer, G.; Chitta, S.; Kuchenbecker, K.J. Human-inspired robotic grasp control with tactile sensing. *IEEE Trans. Robot.* **2011**, *27*, 1067–1079. [\[CrossRef\]](#)
45. Deng, Z.; Jonetzko, Y.; Zhang, L.; Zhang, J. Grasping force control of multi-fingered robotic hands through tactile sensing for object stabilization. *Sensors* **2020**, *20*, 1050. [\[CrossRef\]](#) [\[PubMed\]](#)
46. Ozawa, R.; Tahara, K. Grasp and dexterous manipulation of multi-fingered robotic hands: A review from a control view point. *Adv. Robot.* **2017**, *31*, 1030–1050. [\[CrossRef\]](#)
47. Mikolajczyk, T.; Bednarczyk, K.; Mikolajczyk, A. Model of human hand controlled using pneumatic muscles. *Applied Mechanics and Materials. Trans. Tech. Publ.* **2014**, *555*, 155–162.
48. Szkopek, J.; Redlarski, G. Artificial-Hand Technology—Current State of Knowledge in Designing and Forecasting Changes. *Appl. Sci.* **2019**, *9*, 4090. [\[CrossRef\]](#)
49. Jones, J.L.; Lozano-Perez, T. Planning two-fingered grasps for pick-and-place operations on polyhedra. In Proceedings of the IEEE International Conference on Robotics and Automation, Pasadena, LA, USA, 13–18 May 1990; pp. 683–688.
50. Wolter, J.D.; Volz, R.A.; Woo, A.C. Automatic generation of gripping positions. *IEEE Trans. Syst. Man, Cybern.* **1985**, *15*, 204–213. [\[CrossRef\]](#)
51. Hang, K.; Li, M.; Stork, J.A.; Bekiroglu, Y.; Pokorný, F.T.; Billard, A.; Kragic, D. Hierarchical fingertip space: A unified framework for grasp planning and in-hand grasp adaptation. *IEEE Trans. Robot.* **2016**, *32*, 960–972. [\[CrossRef\]](#)
52. Matoušek, J.; Schwarzkopf, O. On ray shooting in convex polytopes. *Discret. Comput. Geom.* **1993**, *10*, 215–232. [\[CrossRef\]](#)
53. Bern, M.; Eppstein, D. Mesh generation and optimal triangulation. *Comput. Euclidean Geom.* **1992**, *1*, 23–90.
54. Halperin, D. Robust geometric computing in motion. *Int. J. Robot. Res.* **2002**, *21*, 219–232. [\[CrossRef\]](#)
55. Strandberg, M.; Wahlberg, B. A method for grasp evaluation based on disturbance force rejection. *IEEE Trans. Robot.* **2006**, *22*, 461–469. [\[CrossRef\]](#)

Article

Design and Implementation of Composed Position/Force Controllers for Object Manipulation

Sergio Hernandez-Mendez ^{1,*}, Elvia Ruth Palacios-Hernandez ², Antonio Marin-Hernandez ^{1,*},
Ericka Janet Rechy-Ramirez ¹ and Hector Vazquez-Leal ³

¹ Artificial Intelligence Research Institute, Universidad Veracruzana, Calle Paseo No. 112, Colonia Nueva Xalapa, Xalapa 91097, Mexico; erechy@uv.mx

² Faculty of Sciences, Universidad Autónoma de San Luis Potosí, Av. Chapultepec No. 1570, Priv. del Pedregal, San Luis Potosí 78295, Mexico; epalacios@fciencias.uaslp.mx

³ Electronic Instrumentation and Atmospheric Sciences School, Universidad Veracruzana, Cto. Aguirre Beltrán S/N, Zona Universitaria, Xalapa 91090, Mexico; hvazquez@uv.mx

* Correspondence: sergihernandez@uv.mx (S.H.-M.); anmarin@uv.mx (A.M.-H.)

Abstract: In the design of a controller for grasping objects through a robotic manipulator, there are two key problems: to find the position of the object to be grasped accurately, and to apply the appropriate force to each finger to handle the object properly without causing undesirable movement of it during its manipulation. A proportional-integral-derivative (PID) controller is widely used to grasp objects in robotics; however, its main shortcomings are its sensitivity to controller gains, sluggish response, and high starting overshooting. This research presents three coupled (position/force) controllers for object manipulation using an assembled robotic manipulator (i.e., a gripper attached to a robotic arm mounted on a mobile robot). Specifically, an angular gripper was employed in this study, which was composed of two independent fingers with a piezoelectric force sensor attached to each fingertip. The main contributions of this study are the designs and implementations of three controllers: a classic PID controller, a type-I controller, and a type-II fuzzy controller. These three controllers were used to find an object to be grasped properly (position) and apply an equivalent force to each finger (force).

Keywords: robotics; manipulation; intelligent control

Citation: Hernandez-Mendez, S.; Palacios-Hernandez, E.R.; Marin-Hernandez, A.; Rechy-Ramirez, E.J.; Vazquez-Leal, H. Design and Implementation of Composed Position/Force Controllers for Object Manipulation. *Appl. Sci.* **2021**, *11*, 9827. <https://doi.org/10.3390/app11219827>

Academic Editor: António Paulo Moreira

Received: 30 September 2021

Accepted: 16 October 2021

Published: 21 October 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Due to uncertainty on localization, mobile robot manipulation faces two key problems when trying to manipulate objects: firstly, computing the correct position at which the mobile robot needs to be for grasping an object, and secondly, calculating the specific position at which the object to be grasped is.

Specifically, object manipulation through a robotic arm involves three actions: (i) moving the robot and positioning it at a place where its arm's configuration space intercepts the object's position (Figure 1a); (ii) planning and execution of the arm's trajectories—that is, to move the robotic arm towards a position where the object is inside the gripper's configuration space, i.e., where the gripper (end effector) may grasp the object [1,2] (Figure 1b); and (iii) tuning the gripping action—i.e., closing the gripper in order to take and manipulate the object properly [3–5] (Figure 1c). The first two actions are called the *pre-grasping* stage, and the last action is named the *grasping* stage.

The gripper and their controller are key components at the grasping stage in order to manipulate objects accurately. Parallel motion grippers and angular motion grippers are the most commonly used in robotics for object manipulation [6]. Parallel grippers usually have only one motor; therefore, their fingers move simultaneously in order to handle objects. Recently, the use of grippers with angular motions has increased. Thanks to their configuration, these grippers extend the gripping range of objects, and different

configurations may be calculated to handle an object; therefore, these grippers might provide dexterous object manipulation similarly to human fingers. Detailed reviews regarding several types of grippers can be found in [7–9].

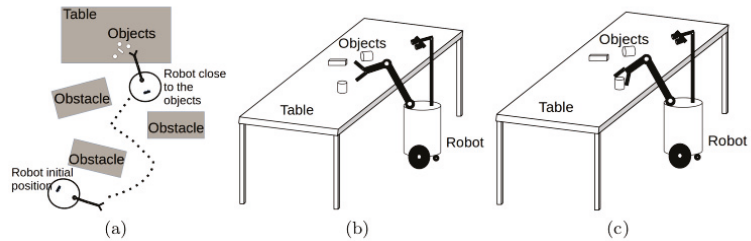


Figure 1. Actions prior to the object's manipulation: (a) moving the robot so the object's position is within the arm's configuration space, (b) object detection and position estimation, and (c) planning and execution of trajectories to drive the arm close to the object.

Focusing on controllers, the PID controller has been widely used in control systems [10–13] due to its simplicity and robustness. In the last few years, there has been increasing interest in using fuzzy logic in different control systems [14–20]. Moreover, fuzzy logic has been used for implementing several applications. For instance: in a multicriteria decision-making process [21], energy consumption for bipedal walking robots [22], and search engine systems [23]. Fuzzy logic uses fuzzy values and rules to cope with uncertainty, just as humans do. These fuzzy values and rules may be implemented based on the user's experience instead of using complex mathematical models.

In the present work, an angular gripper with two independent fingers was used. Prior to the manipulation stage, the actions described in Figure 1 were done. The object's position, initially unknown, was obtained by a perception system and was used to bring the arm close to the object. However, at this stage of the work, no visual surveying has been used. Instead, a tactile feedback control is performed. The object's dimensions were used to estimate the gripper's initial opening. Once the object was between gripper fingers, angular position and (when object do contact) pressure forces were used to control the motion of each finger independently. The pressure forces were obtained by piezoelectric sensors on the fingertips.

The contributions of this work can be summarized as follows: The implementation of three switching force and position controllers (a PID, a fuzzy type-I and a fuzzy type-II) for grasping tasks with a two-fingered gripper (independent fingers). The evaluation and comparison of those controllers, via simulation and a real arm manipulator. A communication protocol to control and switch between controllers over an ROS. Finally, a control process over a two-fingered angular finger gripper that centers the object in order to apply equivalent forces with both fingers.

This paper begins by analyzing some relevant related work (Section 2). Next it presents the details of the system (Section 3). The communication of the gripper with an ROS is described in Section 4. The grasping stage and the design of the proposed controllers are described in Section 5. Experiments and results are presented in Section 6. Finally, conclusions are given in Section 7.

2. Related Work

Controller design for grasping objects in robotics has been an important research topic in the last few years. In this context, proportional-integral (PI) controllers and proportional-integral-derivative (PID) controllers have been designed and implemented to be used in the grippers of robotic arms. For instance, a two-finger gripper for the manipulation of deformable objects was implemented [24]. This gripper used a PI parallel force controller

with prediction models in order to regulate the force exerted by the robot on the object; thus, the risk of damaging the objects during the grip was reduced.

In order to manipulate objects, a force sensor and a flex sensor can be placed in a gripper [25]. This gripper should be controlled only by a motor that opens and closes the gripper. An algorithm and a control strategy are used to apply force to the object without damaging it.

In [26], a gripper with two parallel fingers with two phalanges each and a fixed base was used to manipulate objects. The gripper was controlled only by a motor that provided different torques, and a touch sensor was placed on the fixed surface to measure the pressure applied to the object. In order to manipulate objects, a force sensor can be placed on each finger of a parallel gripper controlled by a single motor [27].

Other studies have tested the designs of controllers using either simulated grippers or simulated robotic hands. For instance, the authors of [28] simulated and manufactured a prototype of a robotic hand with five fingers. To control the kinematic and dynamic movements of each finger, a PID controller was used. They generated a path for each finger in the configuration space, achieving fast motion towards the target angle with a small error signal and little overshooting. However, as described by the authors, when they tested it in a real prototype, the overshooting signal was larger than in the simulation, and was mainly produced by disturbances in some inputs. Moreover, when the reference value was set to a long period of time, small oscillations occurred, again only in the real prototype. Furthermore, the authors of [29] made a mathematical analysis of the contact forces for a robotic finger of three degrees of freedom. The authors proposed two simulated PID controllers; one was used to achieve position and the other to regulate strength. Both controllers were tuned using fuzzy logic.

Regarding using fuzzy logic in control design, the effects of membership functions (triangular, trapezoidal, and Gaussian) on a fuzzy control scheme for a three-finger system have been analyzed [30]. Moreover, another study [31] used a fuzzy control scheme in order to manipulate objects. This control scheme was composed of two fuzzy controllers. The first controller was employed to ensure a stable grip and the second to prevent slipping. In [32], a three-finger gripper to manipulate strawberries without damaging them was developed. A fuzzy controller was designed to compute the force to be exerted on the strawberry. The inputs for this controller were the readings of a capacitive sensor placed on each finger of the gripper. Due to a single motor on the mechanical gripper, the three fingers might be operated simultaneously. Likewise, Reference [33] proposed a fuzzy controller with a gripper using only a single servo motor in order to grasp unknown objects. The authors showed the effectiveness of the controller with three different types of object hardness: soft, moderate, and hard.

Similarly to PI and PID controllers, simulations have been used to test fuzzy controllers. For example, in [34] simulated a fuzzy controller with tactile information to manipulate objects with a two-finger gripper. This simulation was effective; nevertheless, the controller needs to be implemented in a gripper for real applicability. Another study [35] implemented a fuzzy controller to reduce the impacts of forces during the object manipulation. Moreover, the authors developed a fuzzy controller of conformity, in which a mass-spring-damper system determines a desired level of conformity. A simulated robotic hand was used to test the performance of the fuzzy conformance controller.

As can be seen, previous works have had at least one of the following shortcomings: most of them used parallel grippers, not requiring specifically centering the object but limiting the dexterity possible while grasping, and most of the works were tested only in simulated environments or had high variations while testing with real grippers. Consequently, in this work is presented an approach to control the position and force of each individual motor over a two-independent finger gripper in order to increase dexterity; in other words, a robot can adjust grasping by only adjusting the forces and positions of its fingers. Three controllers have been implemented and tested, a PID controller, a fuzzy

type-I controller, and a fuzzy type-II controller. In the following sections, implementations and tests are described.

3. System Description: The Robotic Manipulator

The robotic manipulator assembled in this research is shown in Figure 2. As can be noticed, this robotic manipulator is composed mainly of the following components:

- A two-independent-finger gripper. The gripper is composed of two Dynamixel AX-12 servo motors manufactured by CrustCrawler Robotics. Regarding the fingers, each finger is 10.16 cm long and is made of aluminum. Furthermore, they can be open up to 22.86 cm. An FFS-MT piezoelectric force sensor was placed at each fingertip (Figure 2b) The sensing force range for this sensor goes from 0 to 10 N, with an output resolution of 0.1 mV, providing a stable output over the range of force exerted. In addition, the output of this sensor exhibits linear behavior, as described in [36].
- A three-degrees-of-freedom arm. The arm is operated through Dynamixel servo motors. These servo motors each include a micro-controller, which obtains the different states of the servo motor (e.g., speed, position, temperature, and voltage). The maximum lifting capability is 900 g. A data acquisition board (Arduino) and a 12 V battery were placed on the back of the robotic arm (Figure 2a). The data acquisition board processes the signals from each force sensor and sends them to the computer.
- An iRobot Create. The robotic arm and gripper are mounted on an iRobot Create (mobile base). The iRobot allows moving the robotic arm from one place to another; therefore, objects may be repositioned.

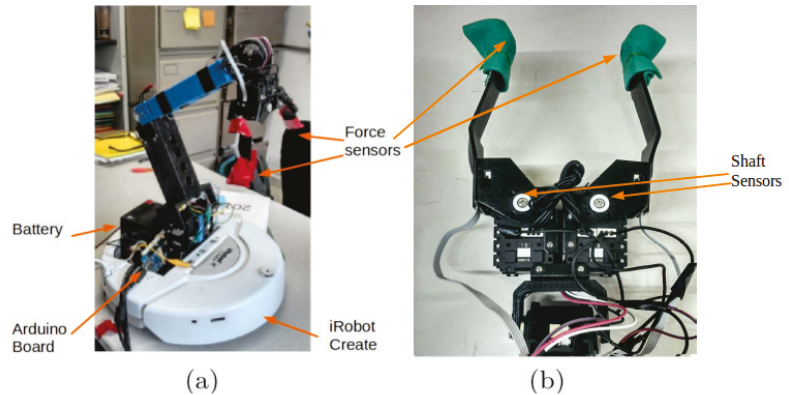


Figure 2. (a) An arm with three degrees of freedom, placed on a mobile robot (iRobot Create). On the back of the robot is a data acquisition board, as well as a 12 V battery to power the system. (b) Force sensors placed on each fingertip of the gripper.

4. Communication with an ROS

In this work, an ROS was used to communicate with all different systems of the robot. Software packages called *nodes* run independently from each other. In order to exchange information, the *nodes* send messages, enclosed in the form of *topics*. Messages are received from publishers and sent to subscribers directly from the master server of the ROS. Therefore, nodes do not directly communicate with each other.

Figure 3 shows a reduced scheme of the nodes and their communication messages on the robotic platform. Two nodes are used to communicate with the hardware: (a) the *serial_node* receives signals from piezoelectric-force sensors on each finger and encapsulates the signal in a message, which is sent to the ROS; and (b) the *dynamixel_manager* reads from the ROS the commands for each motor and returns messages about the state of these. A node called *Force_Position_controller* is in charge of reading “sensor signals” topics and

writing commands to “motor-speed” topics. It is at this node where all control computation is performed.

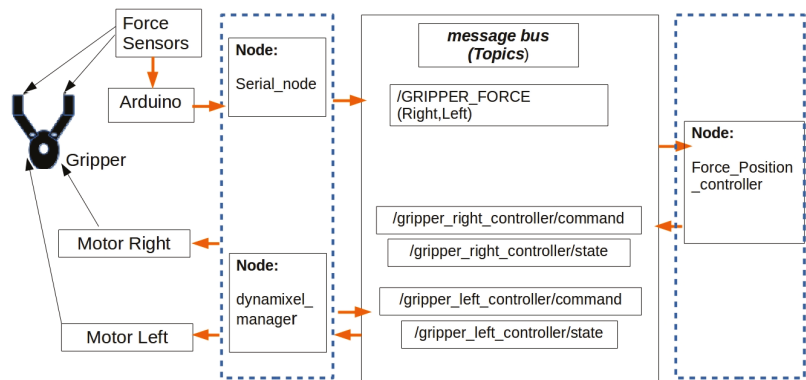


Figure 3. Communication structure of the gripper in the ROS. Nodes at the left are connected directly to hardware, whereas the node in the right (the controller) only communicates with hardware through messages sent to the public memory of ROS (at the center of the figure).

5. Grasping Stage

As has been stated in Section 1, the grasping stage refers to the action of closing the gripper to take and manipulate the object properly. In order to arrive at this stage, some considerations have to be taken into account.

Generally, mobile robotic manipulators dispose of a perception system that provides the object’s information (position, shape, color, etc.) to the arm control system. The perception system can be composed of cameras, laser range finders, depth cameras, or a combination of these sensors. Once the perception system recognizes an object, it sends its information to the position and arm controllers to do the pre-grasping stage. In this work, those stages were realized. In other words, an object’s size and position were calculated and the arm was driven to reach the object. However, due to uncertainty in: (a) the relative position of the mobile base, (b) associated with onboard sensors, and (c) due to the control and motion of the arm’s joints, the position of end-effector can be slightly different than planned. Therefore, the object’s position relative to the gripper can be one of those shown in Figure 4. Namely, the object could be placed in the center, shifted to the right, or shifted to the left between the two fingers. To avoid perception obstruction, as commonly happens with mobile robots, it was decided to not use visual surveying, and instead we used gripper sensor grasping control. The robotic arm used in this work has two-independent fingers with angular motion. Each finger was provided with a piezoelectric force sensor, imposing then, the use of one controller for each finger.

5.1. Problem Statement

Since the initial position of the object is practically known, the first thing that the gripper must do is to find and center the object. Three cases could be present: (i) the initial position of the object is in the center between the fingers; the (ii) the initial position of the object is near to the left finger; (iii) the initial position of the object is near to the right finger (Figure 4).

Then, before the gripper fingers can apply force to the object, it must be centered. If not, the differences in torques while exerting force with unbalanced fingers position can: damage the object or the motors, or make the object slide. Since, at the beginning, none of the force sensors are in contact, it was decided to divide the control system into two different objectives. The former, solving angular positions of the fingers about the object (angular position), and the latter, taking into account the force sensors when they are active.

To deal with these two objectives, it was decided to design a PID position controller and a PID force controller, commuting between them the issues regarding position errors. Once we designed and applied those controllers, and considering the global task of firm and safely handling, the performances of these controllers were not very satisfying. Thus, to compare results, but above all, to increase performance, it was decided to design two fuzzy controllers (one Mamdani type and another type-II). Their advantage over PID controllers is that it is possible to have a different output for each fuzzy-controller, and to avoid sensors' noise, which affects system performance.

Control objectives. Considering that the robotic arm has its gripper near to the object's position, and taking into account the structure of the gripper, the objective of the gripper control system is to achieve firm and delicate object manipulation with the measurement of the angular position and force exerted by each finger. Then, (see Figure 4), it was necessary to design two-feedback laws, position control and force control, such that:

$$\lim_{t \rightarrow \infty} e_p(t) \leq h = \lim_{t \rightarrow \infty} (Ref_{pos} - Pos_{\omega(t)}) \leq h, h = \pm 2Ref_{pos} \tag{1}$$

$$\lim_{t \rightarrow \infty} e_f(t) = 0 = \lim_{t \rightarrow \infty} (Ref_{force} - F(t)) = 0 \tag{2}$$

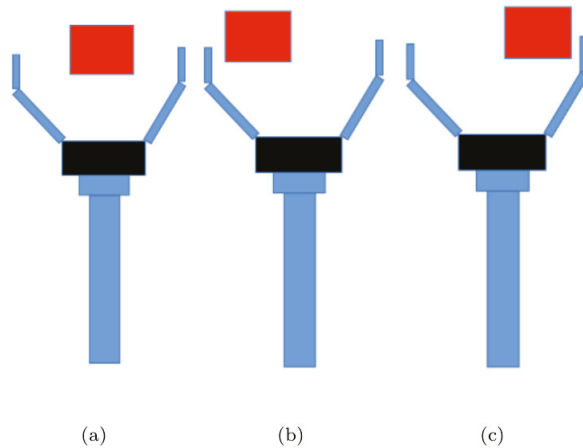


Figure 4. Initial conditions for the object's position being: (a) at the center with respect the fingers, (b) near the left finger, and (c) near to the right finger.

5.2. Design and Implementation of the Controllers

To manipulate an object, the forces applied by the fingers generally are sufficient to counteract gravitational and inertial components of the load force acting on the fingertips [37]. Due to the architecture of the two-independent-finger gripper used on the robotic manipulator, the controllers should perform two main tasks: (i) to place and keep the object at the desired position (i.e., position control), and (ii) to regulate the force applied to the object (i.e., a force control). To accomplish these tasks, three controllers are proposed: a hybrid PID controller (position–force), a type-I fuzzy controller (Mamdani), and a type-II fuzzy controller.

These controllers might use the following inputs provided by the robotic manipulator: two readings from angular positions (right $Pos_{R\omega}(t)$ and left $Pos_{L\omega}(t)$) and two readings, one each from the force sensor of each finger (right $F_R(t)$ and left $F_L(t)$).

On the other hand, the controllers produce a speed command as an output, i.e., two speeds, $M_R(t)$ and $M_L(t)$, which will be applied to the right and left servo motors, respectively.

5.3. Position–Force Hybrid PID Controller

As mentioned previously, PID controllers are widely used in the industry for implementing control systems. Due to its operational ease and low cost, the first implemented controller is based on a hybrid PID scheme; i.e., it consists of two PID controllers: a position PID and a force PID operating in a switching mode as shown in Figure 5. It is important to remark that either of the gripper fingers uses this control scheme.

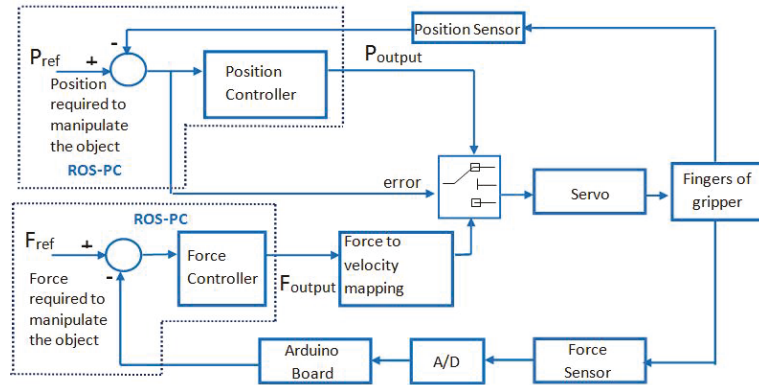


Figure 5. Control scheme of the hybrid PID controller.

The first controller is responsible for placing and keeping the object at desired position. This controller works as follows: it switches on when the object is not centered, and it switches off when the object is centered. On the other hand, the force controller ensures that the fingers will apply enough force to manipulate an object firmly. This controller works when the position controller is switched off; i.e., the position error is zero.

5.3.1. Position Control

A discrete PID algorithm is used to implement the position PID in the ROS. This algorithm replaces the derivative term using a backward difference method and the integral term using a rectangular integration method. The discrete form of position algorithm for PID is given as:

$$u_p(t) = u_p(t - 1) + [K_{p_p} + \frac{K_{p_p} T_{s_p}}{T_{i_p}} + \frac{K_{p_p} T_{d_p}}{T_{s_p}}]e_p(t) + [-K_{p_p} + \frac{K_{p_p} T_{s_p}}{T_{i_p}} + \frac{2K_{p_p} T_{d_p}}{T_{s_p}}]e_p(t - 1) + \frac{K_{p_p} T_{d_p}}{T_{s_p}}e_p(t - 2) \quad (3)$$

where $u_p(t)$ is the control input, K_{p_p} is the proportional gain, T_{i_p} is integral time, T_{d_p} is derivative time or rate time, and $e(t)$ stands for the error between the reference and process output. The values of $K_{p_p} = 0.000001$, $K_{i_p} = \frac{K_{p_p} T_{s_p}}{T_{i_p}} = 0.0001$, and $K_{d_p} = \frac{K_{p_p} T_{d_p}}{T_{s_p}} = 0.001$ were chosen using the Ziegler-Nichol tuning rule. T_{s_p} is the sampling period and $e_p(t) = Ref_{pos} - Pos_{\omega(t)}$ is the position error.

In this case, the position reference is the center of the gripper. When the position error is sufficiently small $e_p(t) \leq h$ with $h = \pm 2\% Ref_{pos}$, a change is sent to the force PID controller.

5.3.2. Force Control

The force PID controller has a similar structure to the position PID controller.

$$u_f(t) = u_f(t - 1) + [K_{pf} + K_{if} + K_{df}]e_f(t) + [-K_{pf} + K_{if} - 2K_{df}]e_f(t - 1) + K_{df}e_f(t - 2)$$

where $K_{pf} = 0.5$, $K_{if} = \frac{K_{pf}T_{sf}}{T_{if}} = 0.005$, and $K_{df} = \frac{K_{pf}T_{df}}{T_{sf}} = 0.001$ are the proportional, integral, and derivative gains, respectively. $u_f(t)$ is the control input, T_{if} is integral time, T_{df} is derivative time or rate time, and T_{sf} is the sampling period. $e_f(t) = Ref_{force} - F(t)$ is the force error. The Ref_{force} is computed according to the object’s weight and material; thus, damage on the object might be reduced.

5.4. Type-I and Type-II Fuzzy Controllers

Fuzzy control provides a formal methodology for representing, manipulating, and implementing human heuristic knowledge about how to control a system. There are type-I fuzzy systems and type-II fuzzy systems.

Generally, in a type-I fuzzy controller, crisp input values are translated to fuzzy input values (fuzzification). These fuzzy input values are computed using rules to produce fuzzy output values (inference process). Finally, these fuzzy output values are mapped to crisp output values (defuzzification). It can be noticed that there are two conversions from crisp values to fuzzy values in a fuzzy controller. To achieve these conversions, it is necessary to define fuzzy sets. A type-I fuzzy set is an extension of a classical set (see Figure 6a). In a classical set, an element can (membership value = 1) or cannot (membership value = 0) be a member of a set (see blue lines in Figure 6a). Conversely, an element can be a member of a type-I fuzzy set at different membership values; i.e., the membership value can range from 0 to 1 (see red lines in Figure 6a). There are four basic membership functions: singleton, triangular, trapezoidal, and Gaussian distribution curve.

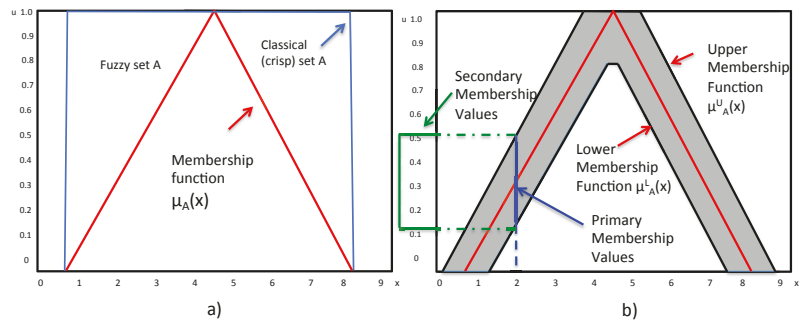


Figure 6. Membership functions of (a) type-I FLS and (b) interval type-II FLS.

Frequently, these membership functions can be designed using knowledge from experts; nevertheless, other studies have used genetic algorithms [38,39] when the defining parameters is challenging. Consequently, type-I fuzzy sets might deal with more uncertainties than classical sets due to the definitions of membership functions.

Regarding the inference process, rules are implemented in order to map the fuzzy inputs to fuzzy outputs. These rules are expressed as IF–THEN statements. The “IF” part is composed of antecedents, which connect the fuzzy inputs, whereas the “THEN” part is composed of consequents, which connect the fuzzy outputs. Generally, the antecedents are defined using two basic operators: and; or. During the inference process, it is frequently the case that more than two rules are fired according to their strengths; therefore, an aggregation operation is performed to join the rules that were fired. These fired rules will

define the fuzzy outputs. Finally, the fuzzy outputs are translated to crisp output values using a defuzzification method.

According to [40], type-I fuzzy controllers might face uncertainties in: (i) their control inputs due to noise affecting the sensors; (ii) their control outputs because of a change in the characteristics of the actuators; (iii) linguistic uncertainties because experts might have different meanings for the linguistic labels. Therefore, they implemented rules using the same antecedents and by changing the consequents. Interval type-II fuzzy controllers might cope with these uncertainties. Fuzzy inputs and fuzzy outputs of an interval type-II fuzzy controller are implemented using interval type-II fuzzy sets. These sets add a second level of membership functions; therefore, these interval type-II fuzzy sets have upper and lower membership functions (see Figure 6b). Additionally, a footprint of uncertainty can be seen in these sets, which is the area between the upper and lower membership functions.

Interval type-II fuzzy controllers perform fuzzification, inference, and defuzzification processes. Furthermore, the interval type-II fuzzy controllers execute the fuzzification and evaluation of rules, as the type-I fuzzy controllers do. The key differences between interval type-II and type-I fuzzy controllers are: (i) interval type-II fuzzy controllers use interval type-II fuzzy sets in the antecedents and consequents of the rules; and (ii) after evaluating the rules, interval type-II fuzzy controllers perform a reduction from type-II fuzzy sets to type-I fuzzy sets using a type-reducer. Once this reduction is carried out, a defuzzifier is applied to obtain the crisp output.

5.4.1. Type-I Fuzzy Controller for Position and Force Control

Figure 7 presents the design of the type-I fuzzy controller. As can be seen, the controller works as follows:

1. Firstly, crisp values are obtained from each finger of the gripper. Specifically, angular positions ($Pos_{R\omega}(t)$, $Pos_{L\omega}(t)$) and the two readings of force sensors ($F_R(t)$, $F_L(t)$) of each finger are inputs to the fuzzy controller.
2. Secondly, these crisp values are translated into input-linguistic values using trapezoidal membership functions. This stage is called fuzzification.
3. Thirdly, rules are evaluated to compute the output-linguistic values. This process is called fuzzy inference.
4. Finally, the output-linguistic values are translated into two crisp values— $M_R(t)$ and $M_L(t)$, which are the speed values for the right and left servo motors using a defuzzification method.

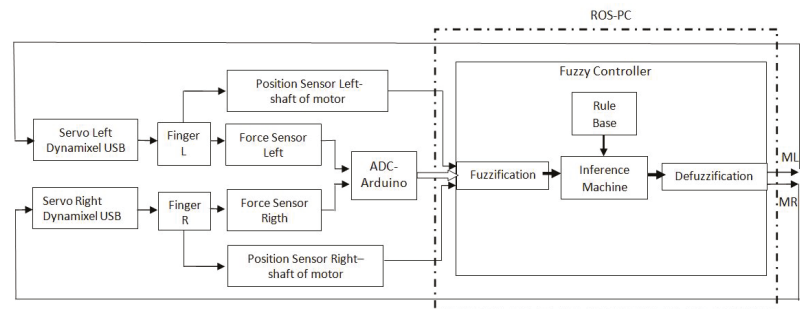


Figure 7. Block diagram of the type-I fuzzy controller with the gripper.

Eight fuzzy input sets were designed and implemented using trapezoidal membership functions for transforming the angular positions and force readings into linguistic values. Focusing on force readings, four trapezoidal membership functions were implemented to transform $F_R(t)$ and $F_L(t)$ into the following four possible linguistic values: *Sensor X Off (SXO)*, *Sensor X Soft Touch (SXTS)*, *Sensor X Touch Hard (SXTH)*, and *Sensor X Squeeze (SXSQ)*, where X is R for the right sensor and L for the left sensor.

Regarding the angular positions of the motor, four trapezoidal membership functions were implemented in order to transform $Pos_{R\omega}(t)$, $Pos_{L\omega}(t)$ into the following linguistic values: *Very Open X Motor* (VOXM), *Open X Motor* (OXM), *Center X Motor* (CXM), and *Closed X Motor* (CLXM), where X is R for the right motor and L for the left motor.

On the other hand, in terms of fuzzy output sets, sixteen singleton membership functions were used for the two outputs of the fuzzy inference, eight for each finger. Moreover, it can be seen from Figure 8 that the linguistic labels are named R1, R2, R3, R4, R5, R6, R7, and R8 for the right finger and L1, L2, L3, L4, L5, L6, L7, and L8 for the left finger. These linguistic values are ordered from the lowest speed value to the highest possible speed value of the command motor.

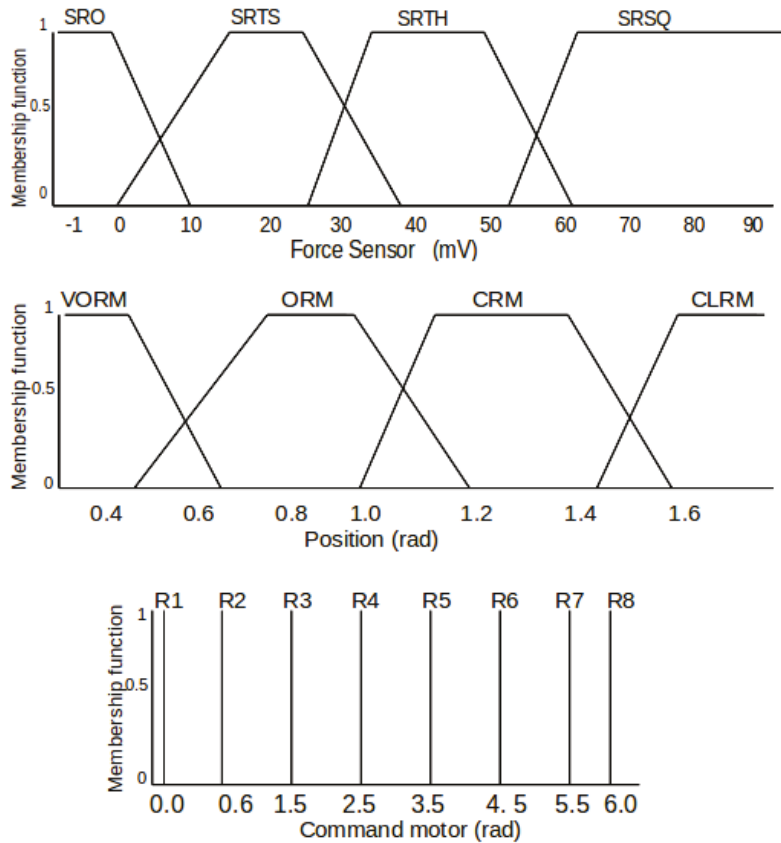


Figure 8. Type I membership functions for the inputs $F_R(t)$ and $Pos_{R\omega}(t)$ and for the output $M_R(t)$.

Table 1 presents the fifty fuzzy rules that were created using the fuzzy sets. The rules have the following structure: IF F_L AND F_R AND $Pos_{L\omega}(t)$ AND $Pos_{R\omega}(t)$, THEN $M_L(t)$, $M_R(t)$. As can be seen, these rules correspond to the type of Mamdani fuzzifier using minimum implication, i.e., the “AND” operator for connecting the antecedents.

In order to design the set of fuzzy rules, two basic scenarios were considered: the gripper has an object between its fingers, and there is no object between the gripper fingers (see Figure 9). In the first scenario, the gripper might be located at different positions with respect to the center of the object to be grasped. Specifically, there are three basic cases that the gripper might face in terms of the position of the object to be grasped:

- Case 1. The object is located on the left side concerning the center of the gripper (Table 1: rules 1–3).
- Case 2. The object is at the center of the gripper (Table 1: rules 4–9).
- Case 3. The object is located on the right side with respect to the center of the gripper (Table 1: rules 6, 10–11).

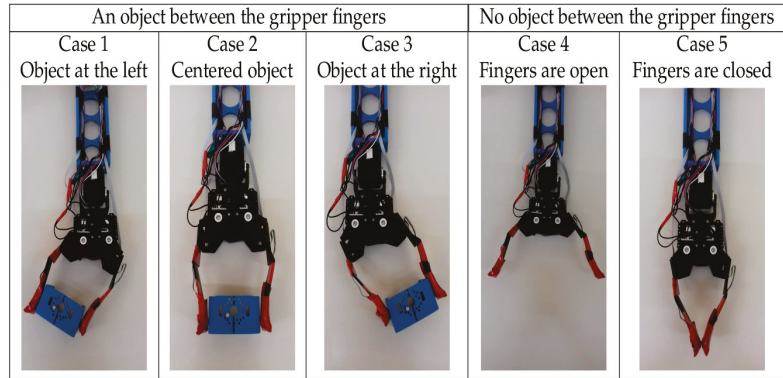


Figure 9. Cases for the design of the fuzzy rules.

Table 1. Fuzzy rules for the type-I fuzzy controller.

IF	F_L	Operator	F_R	Operator	$Pos_{L\omega}(t)$	Operator	$Pos_{R\omega}(t)$	THEN	$M_L(t), M_R(t)$
1	SLTS	AND	SRO	AND	OLM	AND	CRM	THEN	L_4, R_2
2	SLTS	AND	SRTS	AND	OLM	AND	CRM	THEN	L_4, R_3
3	SLTS	AND	SRTS	AND	OLM	AND	CLRM	THEN	L_2, R_2
4	SLTS	AND	SRTS	AND	OLM	AND	ORM	THEN	L_4, R_4
...
25	SLTS	AND	SRO	AND	OLM	AND	ORM	THEN	L_4, R_2
26	SLTS	AND	SRO	AND	CLM	AND	CRM	THEN	L_3, R_3
27	SLTS	AND	SRO	AND	CLLM	AND	CLRM	THEN	L_3, R_2
28	SLTS	AND	SRO	AND	OLM	AND	CRM	THEN	L_4, R_2
...
47	SLTH	AND	SRO	AND	CLLM	AND	CLRM	THEN	L_4, R_4
48	SLO	AND	SRTH	AND	CLM	AND	CRM	THEN	L_4, R_2
49	SLO	AND	SRTH	AND	CLLM	AND	CLRM	THEN	L_4, R_2
50	SLO	AND	SRTH	AND	CLM	AND	CLRM	THEN	L_4, R_2

On the other hand, in the second scenario, there are two cases:

- Case 4. The gripper fingers are open (Table 1: rule 12).
- Case 5. The gripper fingers are closed (Table 1: rule 13).

Additionally, there are other cases in which the object might be located at a position which is between two cases (Table 1: rules 14–50); i.e., the object might be located at a position between the left region (case 1) and the center point (case 2).

It is important to note that during the evaluation of rules, several rules might be fired; therefore, “maximum aggregation” is applied in this case. Finally, to obtain the two speed values, $M_R(t)$, $M_L(t)$, “center of gravity” was used as the defuzzification technique.

5.4.2. Type-II Fuzzy Controller for Position and Force Control

Figure 10 shows the interval type-II fuzzy input sets for the four inputs $\langle F_R(t), F_L(t), Pos_{R\omega}(t), \text{ and } Pos_{L\omega}(t) \rangle$; and interval type-II fuzzy output sets for the two outputs $\langle M_R(t) \text{ and } M_L(t) \rangle$. Both types of fuzzy sets are for the right finger. It can be seen that upper

membership functions were added to both types of fuzzy sets (fuzzy input sets and fuzzy output sets) defined in the type-I fuzzy controller.

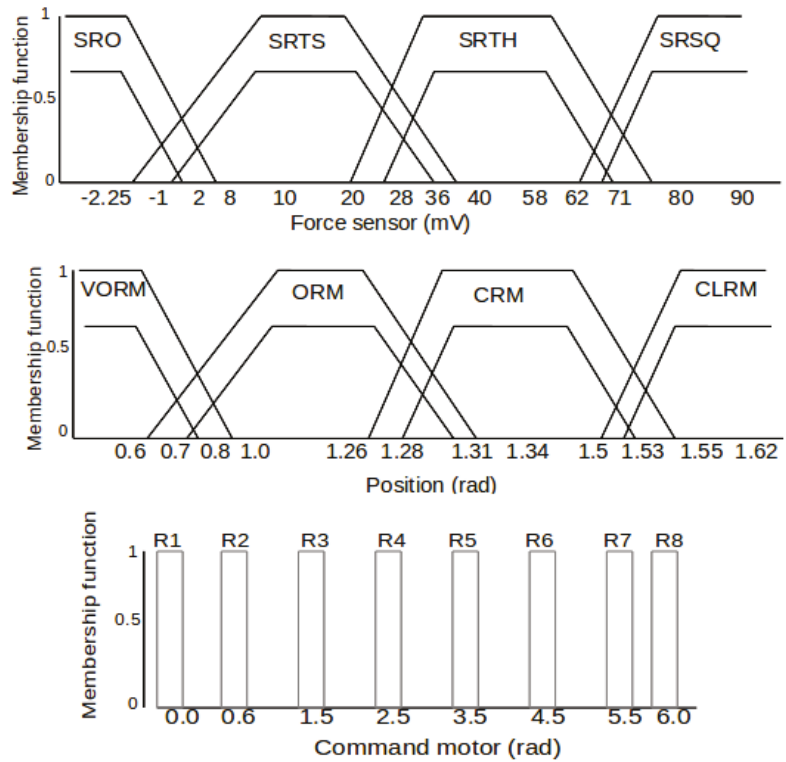


Figure 10. Type II (upper and lower) membership functions for the inputs $F_R(t)$ and $Pos_{R\omega}(t)$, and for the output $M_R(t)$.

The fuzzy rules for the type-II fuzzy controller are the same as those implemented for the type-I controller. These rules correspond to the Mamdani fuzzifier and employ minimum implications. However, the type-II fuzzy controller uses 50 fuzzy rules for the upper membership functions and 50 rules for the lower membership functions. Table 2 shows a summary of these rules. As mentioned earlier, a key difference between type-I and type-II controllers is that type-II uses a type-reducer to transform type-II fuzzy sets to type-I fuzzy sets.

The type-II fuzzy controller employs the Karnik–Mendel (KM) algorithm as the type-reducer. The KM algorithm [41,42] computes the lower and upper intervals of the type-II fuzzy outputs for the left and right motors. For instance, the command for the left motor Y^i (output) is composed of two intervals $[\underline{y}^i, \bar{y}^i]$, where \underline{y}^i is the lower left command motor, and \bar{y}^i is the upper left command motor. Therefore, the KM algorithm corresponding to the left motor works as follows:

1. Firstly, the lower left motor (y_l) is computed in the following manner:
 - (a) y_l^i is sorted in ascending order, where y_l^i is the lower left motor.
 - (b) y_l is computed as

$$y_l = \frac{\sum_{i=1}^M f_l^i y_l^i}{\sum_{i=1}^M f_l^i} \tag{4}$$

- where $f_i^j = \frac{\bar{f}^i + f^i}{2}$ and \bar{f}^i, f^i are the firing intervals. Let $y_l^j = y_l$.
- (c) Find S , such that $y_l^S \leq y_l^j \leq y_l^{S+1}$.
 - (d) Find $y_l^j = \frac{\sum_{i=1}^M f_i^j y_l^i}{\sum_{i=1}^M f_i^j}$ with $f_i^j = \underline{f}^i$ for $i \leq S$ and $f_i^j = \bar{f}^i$ for $i > S$. Let $y_l'' = y_l$.
 - (e) If $y_l^j \neq y_l''$, go to step 6. If $y_l^j = y_l''$, set $y_l = y_l''$, and stop.
 - (f) Let $y_l^j = y_l''$, and go to step 3.
2. Secondly, the upper left motor (y_u) is calculated using the previous steps; but $y_u, y_u', y_u'', y_u^i, y_u^L, y_u^{L+1}, f_u^i$ and U instances are used instead of $y_l, y_l', y_l'', y_l^i, y_l^L, y_l^{L+1}, f_l^i$ and L instances.
 3. Finally, the defuzzification process is performed; i.e., the average of y_l and y_u is computed to be used as the crisp value for the left motor (Y_{final}):

$$Y_{final} = \left(\frac{y_u + y_l}{2}\right). \tag{5}$$

These steps are repeated using the lower and upper intervals for the right motor to compute the crisp value for the right motor.

Table 2. Fuzzy rules for interval type-II fuzzy controller. \underline{F}^i stands for lower membership, \bar{F}^i stands for upper membership, $M_L(t)$ is $[y_L^i, \bar{y}_L^i]$, and $M_R(t)$ is $[y_R^i, \bar{y}_R^i]$.

IF	F_L	Operator	F_R	Operator	$Pos_{L\omega}(t)$	Operator	$Pos_{R\omega}(t)$	THEN	$M_L(t), (M_R(t))$
$[\underline{F}^1, \bar{F}^1] =$	$\frac{SLO}{\bar{SLO}}$	AND	$\frac{SRO}{\bar{SRO}}$	AND	$\frac{VOLM}{\bar{VOLM}}$	AND	$\frac{VORM}{\bar{VORM}}$	THEN	$[\underline{y}_L^1, \bar{y}_L^1] = [0.4, 0.8]$ $[\underline{y}_R^1, \bar{y}_R^1] = [0.5, 0.9]$
...					...				
$[\underline{F}^{34}, \bar{F}^{34}] =$	$\frac{SLTH}{\bar{SLTH}}$	AND	$\frac{SRTH}{\bar{SRTH}}$	AND	$\frac{CLM}{\bar{CLM}}$	AND	$\frac{CRM}{\bar{CRM}}$	THEN	$[\underline{y}_L^{34}, \bar{y}_L^{34}] = [2.0, 2.8]$ $[\underline{y}_R^{34}, \bar{y}_R^{34}] = [1.1, 1.9]$
...					...				
$[\underline{F}^{50}, \bar{F}^{50}] =$	$\frac{SLO}{\bar{SLO}}$	AND	$\frac{SRTH}{\bar{SRTH}}$	AND	$\frac{CLM}{\bar{CLM}}$	AND	$\frac{CLRM}{\bar{CLRM}}$	THEN	$[\underline{y}_L^{50}, \bar{y}_L^{50}] = [2.0, 2.8]$ $[\underline{y}_R^{50}, \bar{y}_R^{50}] = [0.5, 0.9]$

6. Results and Discussion

The robotic manipulator performs a set of actions in order to grasp an object firmly. The configuration of sensors placed at the fingertips, i.e., a sensor surrounded by a foamy surface, permits the grasping of different objects with diverse shapes. For instance, Figure 11 shows the process of grasping a solid plastic box. The set of actions can be divided into three stages:

- **Stage 1.** The main goal of this stage is to center the object with respect to the center of the gripper. Assuming that the position of the object is known, the gripper starts approaching the object. Once the gripper has touched the object, it proceeds to position the object according to its center (the center of the gripper). This stage is known as “positioning” (Figure 11a–e).
- **Stage 2.** Once the object has been centered according to the gripper, the fingers apply specific forces grasp firmly the object; consequently, the object is less vulnerable to falling. As soon as the object has been grabbed, the base of the robot begins rotating for 4.6 s, which is sufficient time for the base to rotate approximately 180 degrees. At this stage, the force PID, type-I, and type-II have to regulate the force to be applied. This stage is known as “force” (Figure 11f–j).
- **Stage 3.** Finally, once the base of the robot has stopped, the arm moves down so that the gripper can release and position the object on the table (Figure 11k,l).

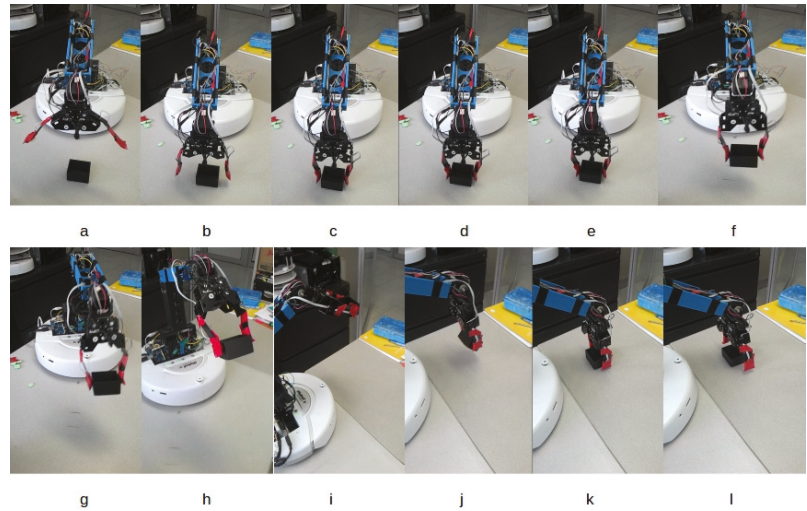


Figure 11. Sequence of actions to grasp an object during the experiments: (a–e) approaching and centering the object; (f–j) grasping and moving the object; (k,l) object’s release.

Figure 12 presents the results from grasping a solid plastic box with a width of 6 cm. This action involved the three stages explained earlier. It is important to recall that the hybrid PID is composed of a position PID and a force PID; consequently, these two PIDs were switched on and off according to the action being performed. These changes are indicated at the bottom of Figure 12. Data were continuously collected for 18 s at a sampling rate of 100 Hz.

It can be seen that the controllers coped with two key control tasks to grasp an object firmly: positioning and force. Regarding the positioning, the three controllers computed position references for each finger of the gripper at stage 1. It can be noticed from Figure 12 that to locate the object at the middle of the gripper, type-I and type-II fuzzy controllers decreased their initial position reference values for the left finger and increased their initial position reference values for the right finger. The three controllers obtained 1.8 rads as the reference position for the left finger, whereas the three controllers obtained a position reference of 3.4 rads for the right finger. In terms of force, the force regulation reference was 60 mV for both fingers. This reference value avoided damage to the object and excessive forces being exerted by the gripper motors. It can be seen that the type-I controller achieved the reference value for the left finger faster than type-II and PID controllers. Moreover, type-II and PID controllers did not achieve the force reference for the right finger. It is important to remark that from 3.62 s to 6.073 s, the hybrid PID switched from a positioning control task to a force control task, whereas the fuzzy controllers performed both control tasks (positioning and force regulation) simultaneously.

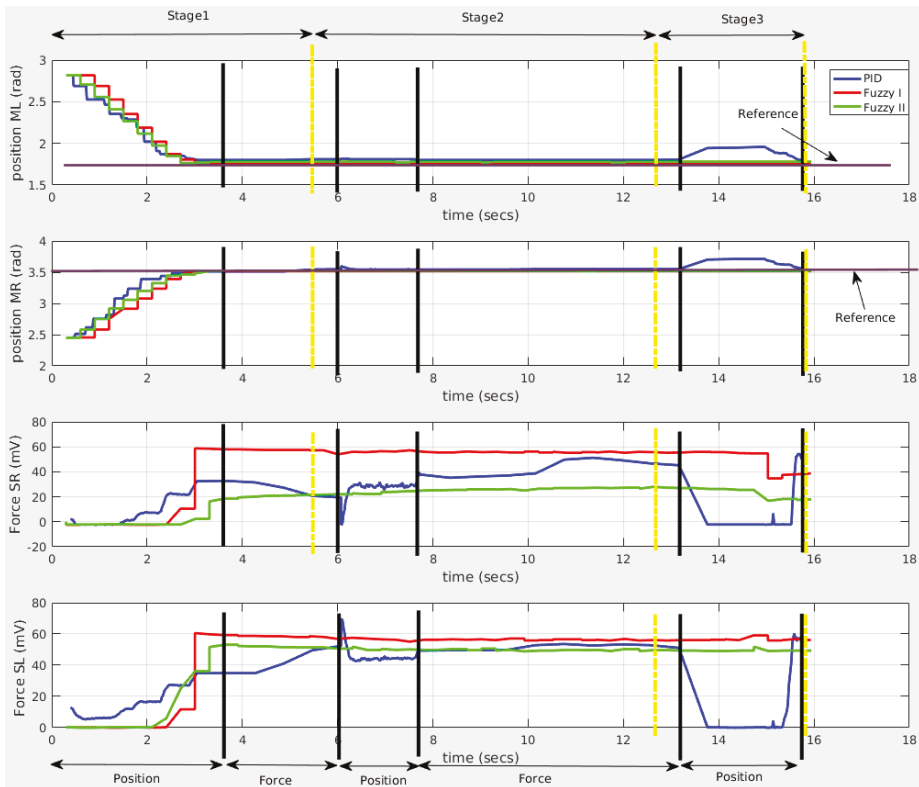


Figure 12. Performance of each controller at each stage. In the above chart are the responses in positioning of motor left and motor right, ML and MR, respectively. In the below chart are the responses of sensor left and sensor right, SL and SR, respectively.

Focusing on stage 2, it can be seen that the three controllers achieved the reference values in terms of position for both fingers. However, regarding force, the three controllers achieved values for the left finger close to the reference, but only the type-I fuzzy controller obtained values for the right finger close to the reference. Additionally, the hybrid PID controller generated various oscillations during this stage. This fact might be related to the switching between the PIDs.

Particularly, as is shown in Figure 12, there are some spikes in the PID controller’s force graph. These spikes correspond to a commutation between PID controllers from force to position. In other words, while the force controller was handling the object, it moved out of the centered position, making commutation necessary to center the object again. Therefore, force sensors measure the inverse force response in relation to the finger motion; i.e., when a spike in force is downward for the right finger, this spike is upward for the left finger, corresponding to an action to center the object.

Finally, the fuzzy controller achieved the reference values in terms of position for both fingers. Regarding the force, the fuzzy controllers obtained values for the left finger close to the reference. It is important to note that the hybrid PID controller generated oscillations at this stage.

For comparison purposes between the PID, fuzzy type-I and fuzzy type-II controllers, 20 experiments were carried out, obtaining the average values of: rise time t_r and settling time t_s . Table 3 shows the values of the rise time, the settling time, the setpoint, and the

type of response. In the same table, it can be observed that all responses were overdamping and the PID had the higher values of t_r and t_s , indicating poor performance.

Table 3. Performances t_s and t_r in a hybrid PID controller, and in type-I and type-II fuzzy controllers.

	Controller	Position MR	Position ML	Force SR	Force SL
t_r	PID	2.93	6.53	11	3.31
	Fuzzy I	3.009	3.000	3.00	3.00
	Fuzzy II	2.705	3.31	3.91	3.31
t_s	PID	3.67	10.76	7.5	4.5
	Fuzzy I	3.31	3.5	3.3	4.2
	Fuzzy II	3.61	3.9	6	4.5
Setpoint		3.4 rad	1.8 rad	60 mV	60 mV
Response		Overdamping	Overdamping	Overdamping	Overdamping

In order to perform a preliminary comparison of these controllers, the integral absolute error (IAE) and integral time of absolute error (ITAE) were computed. It can be seen from Table 4 that the type-I fuzzy controller obtained the lowest IAE and ITAE values, whereas the hybrid PID controller resulted in the highest IAE and ITAE values.

Table 4. Comparison among the hybrid PID controller, and the type-I and type-II fuzzy controllers.

Controller	IAE	ITAE
Hybrid PID	10,278.5	47,901.75
Type-1FLC	684	6366.08
Type-2FLC	4864.25	31,878.23

7. Conclusions

In this paper, a hybrid position-force PID controller, a type-I fuzzy controller, and a type-II fuzzy controller were designed and implemented to grasp an object through a three-fold arm with an independent two-finger gripper. From the results, it can be concluded that the hybrid PID controller generated oscillations in stages 2 and 3 of manipulating the object due to switching between the position PID and the force PID. On the other hand, few oscillations were generated over the three stages of object manipulation with the type-II fuzzy controller. Moreover, it was able to calculate values close to the references in terms of position for both fingers; but in terms of force, it was able to calculate the appropriate value only for the left finger. As for the type-I fuzzy controller, it obtained the best performance, because few oscillations were generated in the three stages of object manipulation, and its t_r and t_s values are practically the same. Despite the type-II fuzzy controller having a lower t_r in the position controller, it had a higher t_s in the force controller and it had more oscillations. The type-I fuzzy controller was able to calculate values close to the reference values in terms of position and force for both fingers.

Author Contributions: Conceptualization, S.H.-M., A.M.-H. and E.R.P.-H. Formal analysis, E.R.P.-H., E.J.R.-R. and H.V.-L. Investigation, S.H.-M. Supervision, A.M.-H. and E.R.P.-H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare that they have no conflict of interest.

References

1. Miller, A.T.; Knoop, S.; Christensen, H.I.; Allen, P.K. Automatic grasp planning using shape primitives. In Proceedings of the IEEE International Conference on Robotics and Automation—ICRA'03, Taipei, Taiwan, 14–19 September 2003; Volume 2, pp. 1824–1829.
2. Berenson, D.; Diankov, R.; Nishiwaki, K.; Kagami, S.; Kuffner, J. Grasp planning in complex scenes. In Proceedings of the 2007 7th IEEE-RAS International Conference on Humanoid Robots, Pittsburgh, PA, USA, 29 November–1 December 2007; pp. 42–48.
3. Hasegawa, T.; Murakami, K.; Matsuoka, T. Grasp planning for precision manipulation by multifingered robotic hand. In Proceedings of the 1999 IEEE International Conference on Systems, Man, and Cybernetics—SMC'99, Tokyo, Japan, 12–15 October 1999; Volume 6, pp. 762–767.
4. Bley, F.; Schmirgel, V.; Kraiss, K.F. Mobile manipulation based on generic object knowledge. In Proceedings of the 15th IEEE International Symposium on Robot and Human Interactive Communication—ROMAN 2006, Hatfield, UK, 6–8 September 2006; pp. 411–416.
5. Kappler, D.; Chang, L.; Przybylski, M.; Pollard, N.; Asfour, T.; Dillmann, R. Representation of pre-grasp strategies for object manipulation. In Proceedings of the 2010 10th IEEE-RAS International Conference on Humanoid Robots (Humanoids), Nashville, TN, USA, 6–8 December 2010; pp. 617–624.
6. Sanchez-Lopez, J.R.; Marin-Hernandez, A.; Palacios-Hernandez, E.R.; Rios-Figueroa, H.V.; Marin-Urias, L.F. A Real-time 3D Pose Based Visual Servoing Implementation for an Autonomous Mobile Robot Manipulator. *Procedia Technol.* **2013**, *7*, 416–423. [[CrossRef](#)]
7. Tai, K.; El-Sayed, A.R.; Shahriari, M.; Biglarbegan, M.; Mahmud, S. State of the Art Robotic Grippers and Applications. *Robotics* **2016**, *5*, 11. [[CrossRef](#)]
8. Khurshid, A.; Ghafoor, A.; Malik, M.A. *Robotic Grasping and Fine Manipulation Using Soft Fingertip*; IntechOpen: London, UK, 2011.
9. Nee, A.Y. *Handbook of Manufacturing Engineering and Technology*; Springer: London, UK, 2015.
10. Ang, K.H.; Chong, G.; Li, Y. PID control system analysis, design, and technology. *IEEE Trans. Control Syst. Technol.* **2005**, *13*, 559–576.
11. Ang, K.H.; Li, Y. Survey on PID Control System Design and Tuning Software Tools. In Proceedings of the 5th Asia-Pacific Conference On Control and Measurement, Dali, China, 8–12 July 2002; pp. 12–17.
12. Bhagwan, S.; Anil Kumar, J.S.S. A Review on: PID Controller. *Int. J. Recent Technol. Mech. Electr. Eng. (IJRMEE)* **2016**, *3*, 17–22.
13. O'Dwyer, A. *Handbook of PI and PID Controller Tuning Rules*; World Scientific: Singapore, 2009.
14. Das, S.; Guha, D.; Dutta, B. Medical diagnosis with the aid of using fuzzy logic and intuitionistic fuzzy logic. *Appl. Intell.* **2016**, *45*, 850–867. [[CrossRef](#)]
15. Lee, Z.J. A robust learning algorithm based on support vector regression and robust fuzzy cerebellar model articulation controller. *Appl. Intell.* **2008**, *29*, 47–55. [[CrossRef](#)]
16. Zadeh, L.A. Fuzzy sets. *Inf. Control* **1965**, *8*, 338–353. [[CrossRef](#)]
17. Hayward, G.; Davidson, V. Fuzzy logic applications. *Analyst* **2003**, *128*, 1304–1306. [[CrossRef](#)]
18. Yager, R.R.; Zadeh, L.A. *An Introduction to Fuzzy Logic Applications in Intelligent Systems*; Springer Science & Business Media: New York, NY, USA, 2012; Volume 165.
19. Lee, C.S.; Wang, M.H.; Hsu, C.Y.; Chen, Z.W. Type-2 fuzzy set and fuzzy ontology for diet application. In *Advances in Type-2 Fuzzy Sets and Systems*; Springer: New York, NY, USA, 2013; pp. 237–256.
20. Son, C. Similarity measuring strategy of image patterns based on fuzzy entropy and energy variations in intelligent robot's manipulative task. *Appl. Intell.* **2013**, *38*, 131–145. [[CrossRef](#)]
21. Singh, S.; Garg, H. Distance measures between type-2 intuitionistic fuzzy sets and their application to multicriteria decision-making process. *Appl. Intell.* **2017**, *46*, 788–799. [[CrossRef](#)]
22. Wang, L.; Liu, Z.; Chen, C.P.; Zhang, Y. Interval type-2 fuzzy weighted support vector machine learning for energy efficient biped walking. *Appl. Intell.* **2014**, *40*, 453–463. [[CrossRef](#)]
23. Ali, F.; Kim, E.K.; Kim, Y.G. Type-2 fuzzy ontology-based opinion mining and information extraction: A proposal to automate the hotel reservation system. *Appl. Intell.* **2015**, *42*, 481–500. [[CrossRef](#)]
24. Wang, X.; Fan, X.; Zhao, Y.; Ma, S. Parallel force control for a robot gripper based on grey prediction models. In Proceedings of the 2012 Power Engineering and Automation Conference, Wuhan, China, 18–20 September 2012; pp. 1–5. [[CrossRef](#)]
25. Xie, Y.; Zhang, B.; Zhou, J.; Bai, Y.; Zhang, M. An Integrated Multi-Sensor Network for Adaptive Grasping of Fragile Fruits: Design and Feasibility Tests. *Sensors* **2020**, *20*, 4973. [[CrossRef](#)] [[PubMed](#)]
26. Pastor, F.; Gandarias, J.M.; García-Cerezo, A.J.; Gómez-de Gabriel, J.M. Using 3D Convolutional Neural Networks for Tactile Object Recognition with Robotic Palpation. *Sensors* **2019**, *19*, 5356. [[CrossRef](#)] [[PubMed](#)]
27. Costanzo, M.; De Maria, G.; Natale, C.; Pirozzi, S. Design and Calibration of a Force/Tactile Sensor for Dexterous Manipulation. *Sensors* **2019**, *19*, 966. [[CrossRef](#)]
28. Widhiada, W.; Nindhia, T.; Budiarsa, N. Robust Control for the Motion Five Fingered Robot Gripper. *Int. J. Mech. Eng. Robot. Res.* **2015**, *4*, 226. [[CrossRef](#)]
29. Ha, X.V.; Ha, C.; Nguyen, D.K. A general contact force analysis of an under-actuated finger in robot hand grasping. *Int. J. Adv. Robot. Syst.* **2016**, *13*, 14. [[CrossRef](#)]

30. Ahmad, H.; Razali, S.; Mohamed, M.R. Fuzzy Logic Controller Design for A Robot Grasping System with Different Membership Functions. *IOP Conf. Ser. Mater. Sci. Eng.* **2013**, *53*, 012051. [[CrossRef](#)]
31. Boughdiri, R.; Bezine, H.; Alimi, A.M. Fuzzy logic control for grasping 3d objects with sliding contacts. In Proceedings of the International Conference on Control, Engineering and Information Technology, Sousse, Tunisia, 4–7 June 2013; Volume 2, pp. 116–119.
32. Dimeas, F.; Sako, D.V.; Moulianitis, V.C.; Aspragathos, N.A. Design and fuzzy control of a robotic gripper for efficient strawberry harvesting. *Robotica* **2015**, *33*, 1085–1098. [[CrossRef](#)]
33. Su, K.H.; Huang, S.J.; Yang, C.Y. Development of robotic grasping gripper based on smart fuzzy controller. *Int. J. Fuzzy Syst.* **2015**, *17*, 595–608. [[CrossRef](#)]
34. Ciobanu, V.; Popescu, N. Tactile controller using fuzzy logic for robot inhand manipulation. In Proceedings of the 2015 19th International Conference on System Theory, Control and Computing (ICSTCC), Cheile Gradistei, Romania, 14–16 October 2015; pp. 435–440.
35. Jamil, M.F.A.; Jalani, J.; Ahmad, A. A new approach of active compliance control via fuzzy logic control for multifingered robot hand. *Proc. SPIE* **2016**, *10011*, 1001111. [[CrossRef](#)]
36. Hernandez-Mendez, S.; Marin-Hernandez, A.; Palacios-Hernandez, E.R.; Vazquez-Leal, H. Characterization of two force sensors to be used in a robotic hand. In Proceedings of the 2015 International Conference on Electronics, Communications and Computers (CONIELECOMP), Cholula, Mexico, 25–27 February 2015; pp. 155–160.
37. Johansson, R.; Westling, G. Roles of glabrous skin receptors and sensorimotor memory in automatic control of precision grip when lifting rougher or more slippery objects. *Exp. Brain Res.* **1984**, *56*, 550–564. [[CrossRef](#)] [[PubMed](#)]
38. Horikawa, S.I.; Furuhashi, T.; Uchikawa, Y. On fuzzy modeling using fuzzy neural networks with the back-propagation algorithm. *IEEE Trans. Neural Netw.* **1992**, *3*, 801–806. [[CrossRef](#)] [[PubMed](#)]
39. Wang, L.X.; Mendel, J.M. Fuzzy basis functions, universal approximation, and orthogonal least-squares learning. *IEEE Trans. Neural Netw.* **1992**, *3*, 807–814. [[CrossRef](#)]
40. Hagrais, H. Type-2 FLCs: A new generation of fuzzy controllers. *IEEE Comput. Intell. Mag.* **2007**, *2*, 30–43. [[CrossRef](#)]
41. Karnik, N.N.; Mendel, J.M. Centroid of a type-2 fuzzy set. *Inf. Sci.* **2001**, *132*, 195–220. [[CrossRef](#)]
42. Mendel, J.M. *Uncertain Rule-Based Fuzzy Logic Systems: Introduction and New Directions*; Prentice Hall PTR: Upper Saddle River, NJ, USA, 2001.

Article

Haptic Teleoperation of Impact Hammers in Underground Mining

Mauricio Correa, Daniel Cárdenas, Diego Carvajal and Javier Ruiz-del-Solar *

Advanced Mining Technology Center, Department of Electrical Engineering, Universidad de Chile, Santiago 8370451, Chile; mauricio.correa@amtc.cl (M.C.); daniel.cardenas@amtc.cl (D.C.); diego.carvajal@amtc.cl (D.C.)

* Correspondence: jruizd@ing.uchile.cl

Abstract: Impact hammers are used to reduce the size of blasted ore in mining operations. In underground mines, tele-operated impact hammers are used to reduce the size of boulders placed on the orepass's grizzlies. An impact hammer consists of a hydraulic arm with 4 degrees of freedom, powered with a hydraulic impact hammer as an end-effector. The tele-operation of impact hammers is difficult due to the latency of communications, the poor visibility of the environment, and the used 2D interfaces. This may result in a collision with the hammer and the infrastructure, idle strokes, and non-optimal operation. To address these issues, this paper proposes the haptic tele-operation of impact hammers. The proposed haptic tele-operation system is based on a 3D model of the environment, which is used to estimate repulsion forces that are transferred to the operator via a haptic device, so that the hammer does not collide with the structures of the mine. The system also allows identifying the oversized boulders deposited on the grizzly and notifying the operator every time the orepass is blocked, as well as providing different 3D views of the environment. A proof of concept is presented using a scaled setup, where it is validated that the use of the proposed system allows for providing a better and more efficient tele-operation experience.

Keywords: impact hammers; industrial robotics; haptic teleoperation; underground mining

Citation: Correa, M.; Cárdenas, D.; Carvajal, D.; Ruiz-del-Solar, J. Haptic Teleoperation of Impact Hammers in Underground Mining. *Appl. Sci.* **2022**, *12*, 1428. <https://doi.org/10.3390/app12031428>

Academic Editors: António Paulo Moreira, Pedro Neto and Félix Vidal

Received: 31 December 2021

Accepted: 24 January 2022

Published: 28 January 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Underground mines commonly use orepass systems as a safe and economic method to transport blasted ore, i.e., broken rocks, between production levels. In each orepass, a steel grate, called a grizzly, is used to control the size of the material entering the orepass. Material that is too large cannot pass through the grizzly and has to be broken using impact hammers. Impact hammers, also known as rock breakers, rock-breaking manipulators, rock-breaking hammers, or pedestal-mounted breaker booms, consist of a hydraulic arm with 4 degrees of freedom (rotation, lift, tilt, and breaker joints) powered with a hydraulic impact hammer as an end-effector. Figure 1 shows a diagram of the operating environment of an impact hammer. It shows a possible arrangement of sensors to obtain data from the material over the grizzly.

In modern mining facilities, impact hammers are usually tele-operated from a control room located in a safe place, away from the operation area. Under this schema, the operator actuates the electro-valves that control the hydraulic system using joysticks and buttons via a data network. To compensate for this lack of direct vision of the environment, cameras and video transmission equipment are often installed in the operation area. However, the data network and video stream encoding and decoding introduce a delay, or latency, between what the operator sees, and what is actually happening, which adds difficulties to tele-operation [1,2]. Due to the mentioned latency, the poor visibility of the environment, and the used 2D visualization interfaces, hammer tele-operation requires highly skilled operators that are trained to deal with these two phenomena. A typical operating environment can be observed in Figure 2. This figure shows a frontal loader dumping material onto the grizzly, and a tele-operated impact hammer fragmenting rocks.

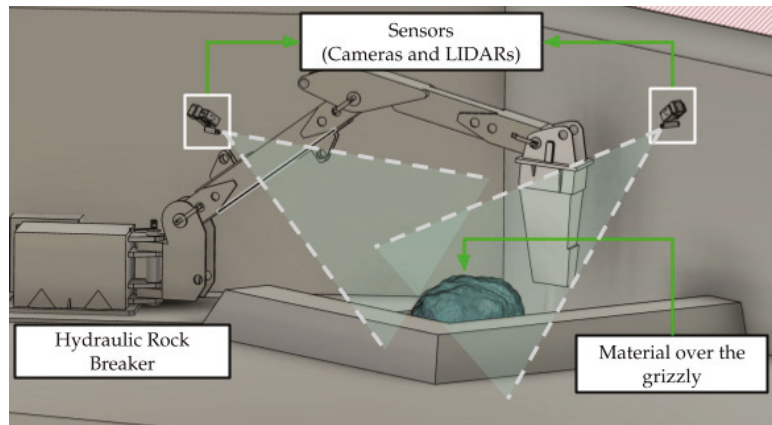


Figure 1. A rock breaking hammer environment with sensors, which includes: A hydraulic rock breaker, a grizzly with material, and sensors pointing to the working area.



Figure 2. Real impact hammer in operation, while material is dumped onto the grizzly.

Normally an operator handles more than one impact hammer, and due to the intermittent nature of this operation, sometimes the operator does not notice immediately that an orepass is being obstructed. This often causes the fragmentation process to take longer than it should because it is more difficult to clear a stoked orepass than a few rocks over the ore pass's grizzly, generating a delay in the mine's production chain. In addition, the described, standard tele-operation of impact hammers does not allow the operator to properly perceive the actions of the impact hammer in the environment because its sensors are limited to visual cameras that provide a 2D representation of the environment. This issue may cause damage to the impact hammer, for example, when the tool is activated without being in contact with the rock, which causes idle strokes in the air, or when the hammer collides with the environment (e.g., with the grizzly). In order to address these drawbacks, assistive tele-operation technology can be used for: (i) Enhancing the operator's perception by adding other sensing modalities (e.g., range sensors), (ii) notifying the operator every time that an orepass is blocked to have a shorter reaction time, and (iii) giving haptic feedback to the operator to avoid collisions of the hammer's end-effector

with the surrounding environment. Using these improvements, the operator could make better decisions that result in a more precise operation, extended lifetime of the equipment, and higher throughput of the mineral transport system.

There is very little literature related to the automation or tele-operation of impact hammers. According to [3], previous works concerning the automation or modernization of impact hammers are few. First attempts to automate impact hammers date back to 1998 [4], and the first tele-operated impact hammer was reported in 2000 [5].

Most of the reported work addressing tele-operation of impact hammers uses 2D cameras, and in some cases time-of-flight cameras (TOF) or stereo cameras. According to [3], TOF's low resolution is insufficient for this task, and stereo cameras obtain a single view of the scene. To the best of our knowledge, there are no previous works using 3D LIDAR (Laser Imaging Detection and Ranging) in this application, nor works reporting the use of haptic devices for the tele-operation of impact hammers.

Regarding the autonomous operation of impact hammers, the reported results are very poor. In [3], an average success rate of 34% is obtained in the task of breaking rocks, which does not make it possible for the application of this technology in a real mining environment. For this reason, we believe that a good intermediate step to full-automation, which improves the current technology used for the tele-operation of impact hammers, is haptic tele-operation.

In this context, this paper proposes the haptic tele-operation of impact hammers for improving the efficiency of the fragmentation process. The proposed haptic tele-operation system is based on a 3D model of the environment, which is used to estimate repulsion forces that are transferred to the operator via a haptic control device, so that the hammer does not collide with the structures of the mine. The repulsion forces are a function of the distances between the hammer's end-effector and the environment. The 3D model of the environment is built using point clouds acquired using range sensors, and updated continuously.

The proposed system also allows identifying the oversized boulders deposited on the grizzly and notifying the operator every time the ore pass is blocked. Moreover, thanks to the use of 3D sensors and cameras placed on opposite sides of the hammer (see Figure 1), it is possible to show the operator the environment from different viewpoints and a 3D model of the rocks to be crushed. With this information, the operator can make better decisions.

We do believe that the use of point clouds for improving tele-operation in mining applications will increase in the following years, thanks to the popularization of 3D scanning sensors, and the availability of libraries for processing this data (e.g., [6,7]). Field applications, as the one described here, will use this haptic technology.

In other industries, the use of point clouds (obtained using range sensors) for obtaining haptic feedback in tele-operation applications is not new, although most reported papers describe systems that operate in controlled or semi-controlled environments, which is not the case for underground mining. In ref. [8], a method of haptic feedback in real time, using streaming point clouds from RGB-D cameras, without using contact sensors and without preprocessing the data, is proposed. The concept of virtual world is used, in which the simulated robot can move freely. In this virtual world, the tip of the virtual effector or HIP (haptic interface point) is related to the movement desired by the user. This effector, in turn, is wrapped in a zone of influence called proxy, which is forbidden to interact with the point cloud. Therefore, if the effector tries to enter the point cloud (or be in a position very close to it), a difference between the position of the virtual effector or HIP with the center of the proxy will be produced. This difference in positions will result in a feedback force vector, which will cause the effector's motion to change direction. The work described in ref. [9] is based on the described feedback system, and introduces the concept of delay in tele-operation, where the motion is first performed in virtual space and then executed in the real world. This implementation and testing of the system were performed by teleoperation of a KUKA industrial robot, where the virtual world was created using point clouds obtained with an RGB-D camera. This work confirms that haptic feedbacks

increase the operator's accuracy by reducing the errors in the desired trajectory of the end effector, decreasing the collisions with the environment, and reducing the operation time. In ref. [10], a haptic system based on distances measured using a stereo camera, is evaluated on a robotic arm used to transport an object without colliding with obstacles. In ref. [11], a 3D virtual environment is recreated by a Kinect sensor used to calculate haptic forces based on potential fields. The paper evaluates the path generated when operating a robot arm with 7 degrees of freedom.

This paper is organized as follows. In Section 2, the proposed haptic tele-operation of impact hammers is described. Section 3 presents results, both in simulation and in reality, which are discussed in Section 4. Finally, in Section 5, conclusions of this work are drawn.

2. Haptic Teleoperation of Impact Hammers

2.1. Overall System Architecture

The proposed haptic tele-operation system is composed of eight sub-modules (see Figure 3). The *Point Cloud Server* acquires and merges the data from the two LIDARs, and segments it in three different point clouds, each one corresponding to measurements associated to reflections on the infrastructure, hammer, and material. These point clouds, and the acquired images, are sent to the *User Interface* where the different visualization are generated for the user. The *Collision Point Cloud Generation* module receives the point clouds and generates a so-called *collision point cloud*, which represents the points in the space to be avoided during tele-operation. The *Haptic Feedback Calculation* module estimates possible future collisions using the collision point cloud, the current commands on the end-effector (u_{eff}), and the state of the hammer's joints (x), and it generates haptic feedback information (f) for the haptic controller and haptic control interface. The *Haptic Controller* receives this feedback and generates the haptic force to be applied on the *Haptic Device* to alert the user of the limitation of the commands. At the same time, the *Haptic Control Interface* receives the control command on the end effector, which is the result of the operator force on the *Haptic Device* and the haptic force on the joystick, and the haptic feedback to limit the commands in case of any risky situation, for example, when the operator in spite of the haptic feedback decides to perform a risky task. Finally, these commands are converted from the Cartesian space (u_{eff}) to the joint space (u_{joints}) using the *Inverse Kinematics* of the machine to handle the articulation actuators directly.

2.2. Sensors

Sensors are located above the grizzly, on opposite sides, to generate complementary points of view of the working space (see Figure 1), and to provide a better representation of the material over the grizzly when both views are combined. In each side, a 3D LIDAR and camera are used. These sensors provide a streaming of point clouds and images. Point clouds are used to build a 3D model of the environment, and images are used for visualization purposes.

2.3. Point Cloud Server

This module first merges the point clouds acquired by the different 3D LIDARs into a single cloud. This point cloud is down sampled to generate a voxel grid representation of the space. Then, all the voxel grid points are separated in three different categories (see Figure 4):

1. Environment points that include all points that do not change in the operation of the system (e.g., point of the floor, points of the grizzly, and point over other parts of the structure of the mine). These points are used to build the 3D model of the environment;
2. Hammer points that correspond to the points generated by the LIDAR's reflections on the impact hammer, which are generated when certain parts of the hammer are positioned in the scanning area of the sensors;

3. Material points that correspond to any other point detected, which should correspond to the material over the grizzly.

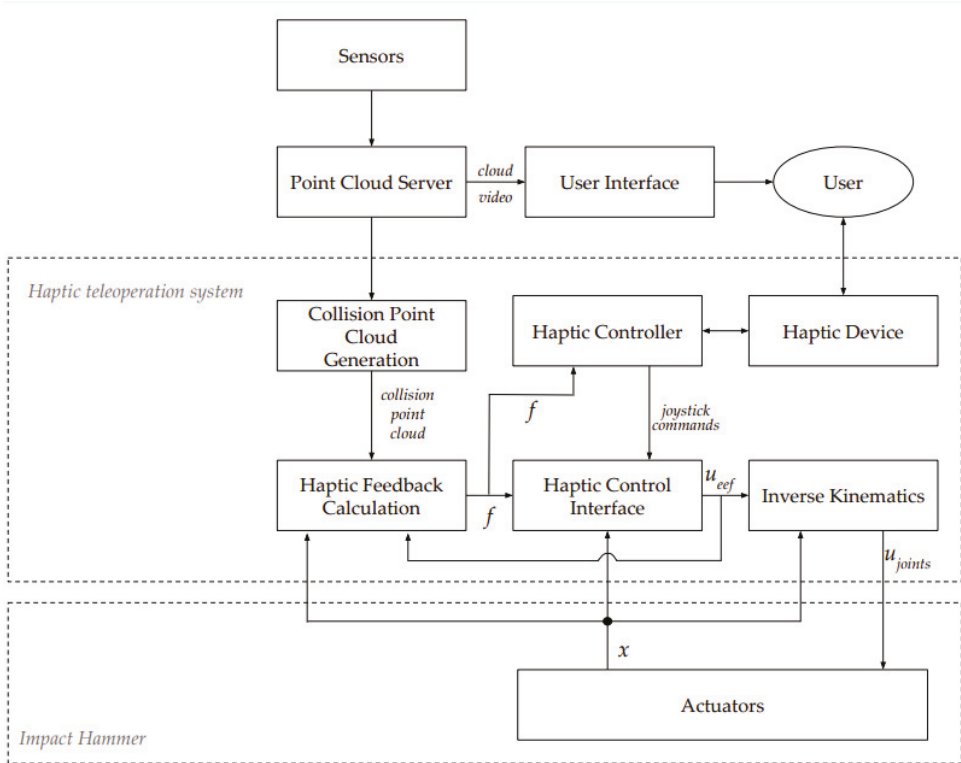


Figure 3. Overall view of the haptic tele-operation framework.

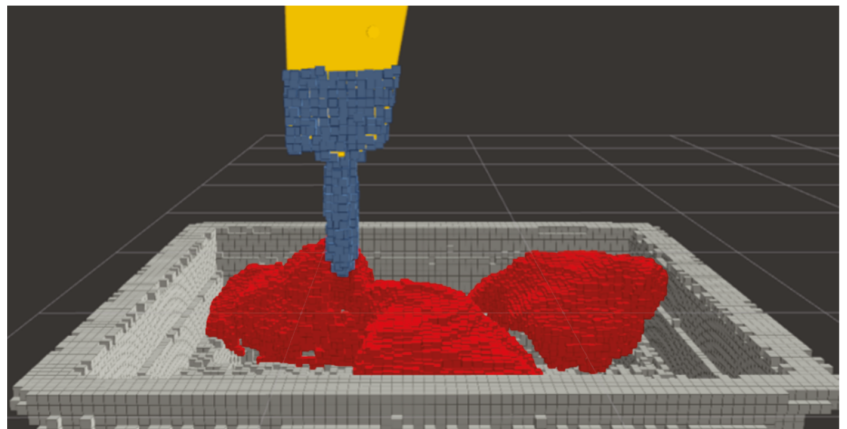


Figure 4. Point cloud categorization. Environment points (in gray) and material points (in red). Hammer points in blue, showed over the end-effector model (in yellow).

During the system's initialization, the environment model is generated by successively integrating point clouds from the environment, without considering the ones corresponding to material or the impact hammer. This results in a very dense point cloud that is first filtered for deleting isolated points that could be generated by noise on the sensors. Once the noise has been eliminated, the cloud is voxelized in 3D square voxels, reducing the number of point and maintaining the occupancy information of the environment in the space. Finally, this cloud is saved as the environment model. Figure 5 shows the main modules of the described process. Sensors 1 and 2 correspond to the 3D LIDARs used to capture the model of the environment through point clouds. The *Point Cloud Integration* module corresponds to the process of joining the point clouds of both sensors and performing the task of integrating these successive point clouds over time. The *Noise Filtering* module is responsible for removing noise. The *Voxelization & Occupancy Grid* module is in charge of voxelizing the point cloud in order to eliminate repeated points and to show the occupancy of the space. And finally, the *Save Model in Memory* module stores the resulting point cloud in memory for later use.

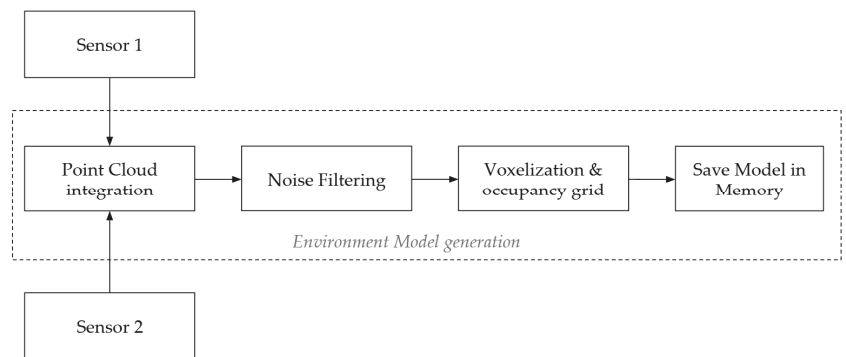


Figure 5. Processes to obtain the point cloud of the environment.

During operation, the environment model is used for determining the points that belong to the material model and hammer model. In each iteration, each new point cloud is voxelized and contrasted with the point cloud of the environment model. This comparison allows to subtract the points belonging to the environment. Once subtracted from the environment points, the resulting point cloud contain the hammer-points and material-points. Hence, to obtain the material-points, a new filtering process should be performed to subtract the hammer-points. This is done by first estimating the spatial region where the hammer is located, which is obtained using the current state of the hammer's joints and the kinematic model of the hammer. This way, all those points within this space will be classified as belonging to the hammer and therefore eliminated. Figure 6 shows the process used to obtain the point cloud of the material. Sensors 1 and 2 correspond to the 3D LIDARs used to capture the working environment. The *Point Cloud Concatenation* module is in charge of joining the point clouds coming from both sensors. The *Voxelization* module is in charge of voxelizing the point cloud resulting from the union of both sensors, thus eliminating repeated points. The *Environment Model Points* corresponds to the point cloud stored in memory, which only contains points corresponding to the grizzly. The *Subtraction of Environment Points* module compares the current point cloud with the point cloud stored in memory, and removes the points that are found in both models. Finally, the *Material Points* module is in charge of publishing the points resulting from the previous module, which correspond to the material on the grizzly.

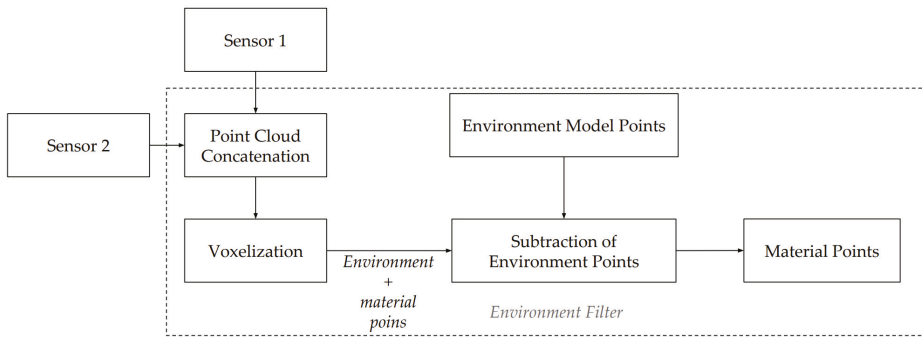


Figure 6. Processes to obtain the Point Cloud of the material.

2.4. Collision Point Cloud Generation

The generation of a collision point cloud has the goal of using it for avoiding collisions between the impact hammer and surrounding environment.

The collision point cloud always includes the environment point cloud, and the material cloud, depending on if the user wants to avoid the material or not. For instance, when the hammer’s end-effector needs to be placed in the rock to be broken, the user chooses not to include the material cloud. This is done by just pressing a button on the haptic device.

The generation of collision points starts with a dilation operation over the environment point cloud, and applies over the material point cloud, which is implemented in each case using as structuring element cube of $3 \times 3 \times 3$ points. The dilation of the input point cloud by the cube of $3 \times 3 \times 3$ (which corresponds to a cube of 27 points), can be understood as the locus of the points covered by the cube of $3 \times 3 \times 3$ point when the center of it, moves within the input point cloud. The dilation operation is defined as follows:

$$A \oplus B = \cup_{b \in B} A_b \tag{1}$$

where A is the initial point cloud, and B is the structuring element (cube $3 \times 3 \times 3$).

This means that for each initial point, 26 extra points are created, which will surround the initial point (27 points in total). Figure 7 shows the dilation operation, and the result of this operation on a single voxelized point.

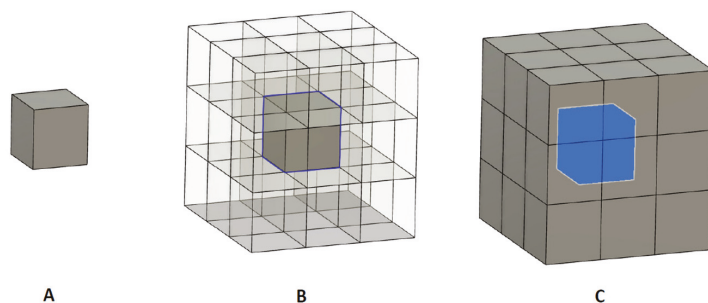


Figure 7. (A): Single voxelized point; (B): $3 \times 3 \times 3$ structuring element with origin in central point; and (C): result of dilation operation.

At the end of the morphological dilation, a new point cloud is obtained, with multiple repeated points because the implemented dilation algorithm does not check if there is already a point in the space where the new points are generated. To eliminate this redundant

data and to increase the algorithm’s performance, the resulting cloud is voxelized again. Figure 8 shows the process to obtain the collision points.

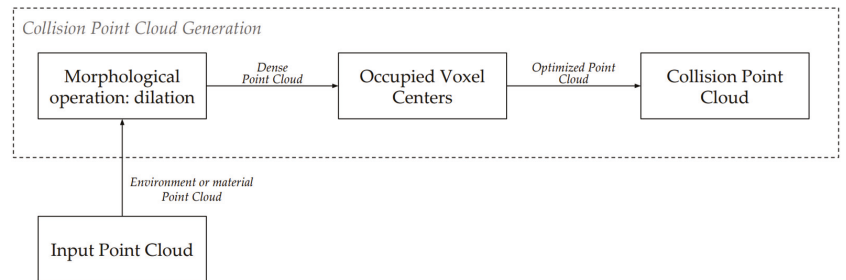


Figure 8. Processes to obtain the collision point cloud.

As mentioned, the collision point cloud is composed of two parts. The first part will come from the processing of the point cloud belonging to the environment model. As this model is static and was previously calculated in a previous module, it is only processed once to generate the cloud of collision points belonging to the environment. The second part will come from the processing of the point cloud belonging to the material. As this point cloud is dynamic (it is expected to change frequently), the process of obtaining the collision point cloud is performed in each execution cycle. In this case, for performance reasons, the dilatation operation is performed once per cycle.

In the case of requiring to generate a thicker collision point cloud, several dilation iterations should be performed in the same cycle. This would result in a larger safety distance between the real object and end effector. Figure 9 shows a comparison between the point cloud of the generated grizzly model and the point cloud resulting from the dilation process.

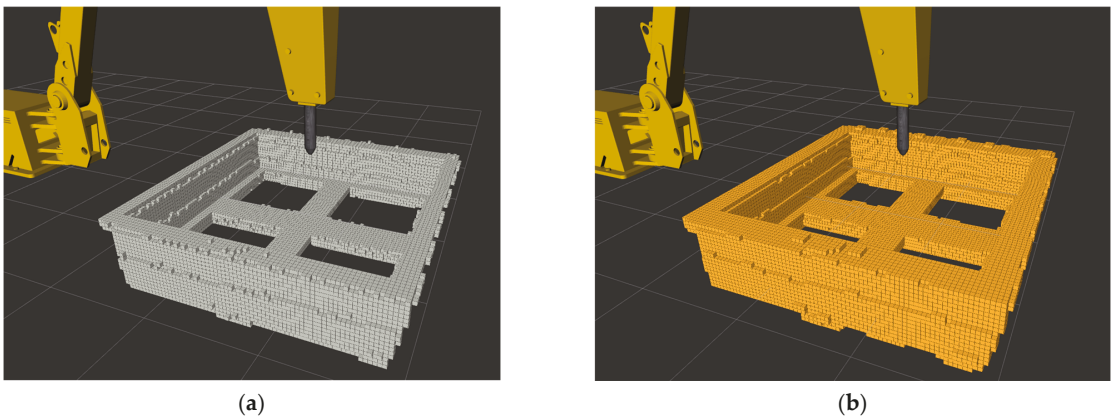


Figure 9. (a) The original collision cloud generated with the environment points. (b) The resulting collision point cloud after applying the dilation process.

Therefore, the resulting collision cloud will be the concatenation of points from the static processing (performed only once at the beginning of the program) of the environment model, with that of the dynamic processing of the material (performed in each cycle). This point cloud will result in a more homogeneous cloud, filling the empty spaces due to the imperfection of the sensors.

It must be emphasized that at any moment, the user can choose using just the environment model for the generation of the collision point cloud, not including the point clouds associated to the material. This is normally made just before placing the hammer on the rock to be broken. The user makes the selection of this operation mode by just pressing a button in the haptic device.

2.5. Haptic Feedback Calculation

This module generates the haptic feedback that will restrict the movement of the joystick axes and the movement of the impact hammer. For this purpose, a so-called *virtual collision region* is generated around the end effector, which represents the space of the possible future positions of the end-effector, considering the direction in which it moves. The intersection of the virtual collision region with the collision point cloud will indicate possible collisions of the end-effector with the environment.

For simplicity, in this work two different geometries are used for representing the virtual collision region: A rectangular cuboid and a semi-sphere, which are positioned at the end-effector point that is located on the tip of the chisel. Figure 10 shows these geometries, which are oriented with respect to the direction of movement of the end-effector. In this simulation, the yellow arrow corresponds to the direction of motion of the end-effector.

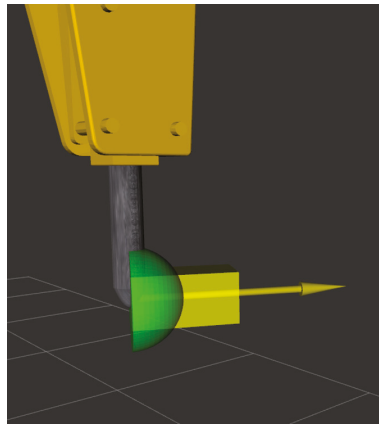


Figure 10. The yellow arrow shows the direction that the end-effector is moving. The yellow cuboid and green half-sphere are oriented according to the direction of motion of the effector.

As mentioned, the intersection of the virtual collision region with the collision point cloud will indicate how close is the effector to collide if the current direction and velocity of movement are maintained. The orientation of the virtual collision region has a direct relation to the direction of displacement of the end-effector, and the region it represents indicates the possible future positions in which the effector will find itself in future time steps.

The length of the cuboid is selected as the maximum distance that the effector will move before stopping (inertia of the system). The height and width of the cuboid are slightly larger than the height and width of the end-effector. These values make it possible to distinguish when the end-effector is in an apparent collision state with the collision point clouds, and therefore perform subsequent actions on the effector to avoid the collision. However, it could happen that a sudden movement in the actuation of the controls would produce an abrupt change on the movement direction of the end-effector, resulting in a collision that cannot be detected by the cuboid. To avoid this situation, the semi-sphere region will act as a safety region, detecting these particular cases, where the effector is not in a collision direction, but very close to the objects.

Thus, depending on the positions of the virtual collision region with respect to the collision point cloud, three different cases can be identified:

1. Rectangular cuboid and semi-sphere regions without intersection with the collision point cloud (see Figure 11a). In this case the impact hammer can continue its trajectory;
2. Rectangular cuboid intersects the collision point cloud (see Figure 11b). In this case, the velocity will be reduced according to the distance to the obstacle (see explanation in next sub section);
3. Semi-sphere region intersects the collision point cloud and the cuboid does not (see Figure 11c). In this case the velocity will be reduced to 50% (see explanation in next sub section).

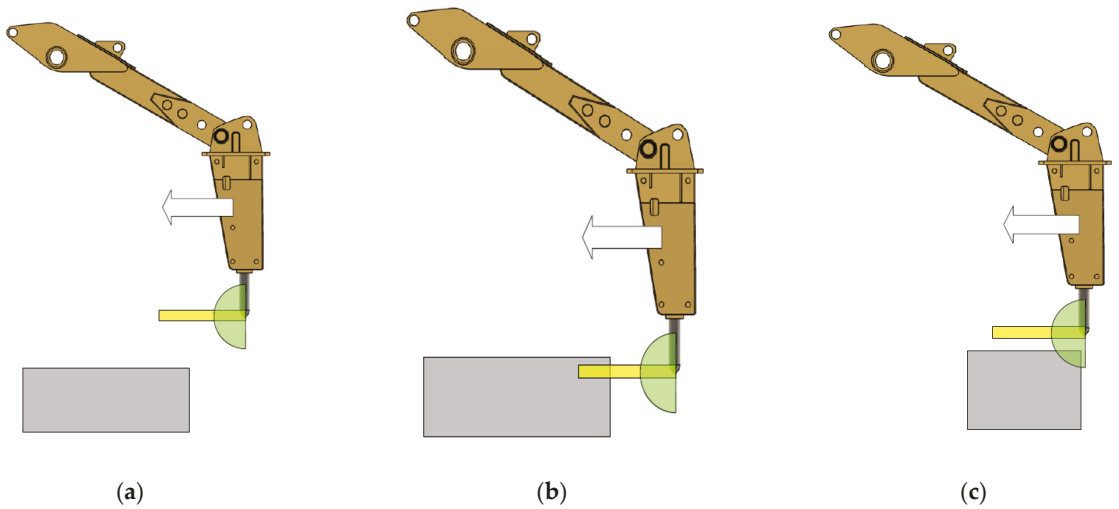


Figure 11. (a) Rectangular cuboid and semi-sphere regions without intersection with a collision point cloud. (b) Rectangular cuboid intersects the collision point cloud. (c) Semi-sphere region intersects the collision point cloud and the cuboid does not. The yellow rectangle represents the cuboid. The green semi-circle represents the semi-sphere. The white arrow corresponds to the hammer direction.

The output of haptic feedback (f) has three values: A Boolean indicating if the cuboid intersects the collision point cloud or not, a Boolean value indicating if the semi-sphere intersects the collision point cloud or not, and a normalized value that indicates a level of safety of executing a command.

The level of safety parameter (S) is calculated using the distance from the end-effector to the collision point cloud ($Dist_{EP}$), as the percentage of the cuboid region lying outside the intersection with the collision point cloud. To calculate this percentage, a max distance ($Dist_{MAX}$) is predefined. This distance is considered as a deceleration distance by the end effector when it moves with the maximal allowable speed and it stops. Thus, S is calculated as:

$$S = \min \left\{ \frac{Dist_{EP}}{Dist_{MAX}}, 1 \right\}. \quad (2)$$

2.6. Haptic Control Interface

This module limits the end-effector's actuation orders. Depending on the distances between the virtual collision region and collision point cloud, the maximum allowed speed will change, so that the effector does not collide and is not damaged. According to the cases defined in the former sub section:

1. Rectangular cuboid and semi-sphere regions without intersection with the collision point cloud. In this case no constraint action is performed, the end effector can reach maximum velocity;
2. Rectangular cuboid intersects the collision point cloud. In this case, the resulting end-effector velocity ($u_{eef\ limited}$) will be reduced depending on the level of safety parameter as:

$$u_{eef\ limited} = u_{eef} \cdot S^2. \tag{3}$$

The use of the square of S is inspired by the fact that the kinetic energy depends on the square of the speed.

Thus, the magnitude of the commands on the end-effector will decrease its maximum allowable speed as it gets closer to the collision point-cloud, arriving with velocity tending to zero as it touches the collision point cloud.

3. Semi-sphere region intersects the collision point cloud and the cuboid does not. In this case the end effector speed is limited to 50% of the maximum speed. In other words, if a higher effector speed is attempted under this condition, the system will automatically limit the speed. In case the effector movement is attempted at a lower speed, the system will not make any adjustment to the speed.

2.7. Inverse Kinematics

This module adapts desired velocities on the end-effector to the corresponding joint velocities. To do this task, the Jacobian pseudo inverse of the kinematic model is calculated using the current state of the machine to perform this conversion as:

$$u_{joints} = J_{pseudo\ inv} \cdot u_{eef} \tag{4}$$

where u_{eef} corresponds to a vector with the velocities on some axis of the Cartesian space, $\{x, y, z, pitch\}$ in this case; u_{joints} corresponds to the speed on each articulation on the machine $\{q_0, q_1, q_2, q_3\}$ related to its respective joint, and $J_{pseudo\ inv}$ corresponds to the pseudo-inverse of the Jacobian of the kinematic model of the arm, which is shown in Figure 12.

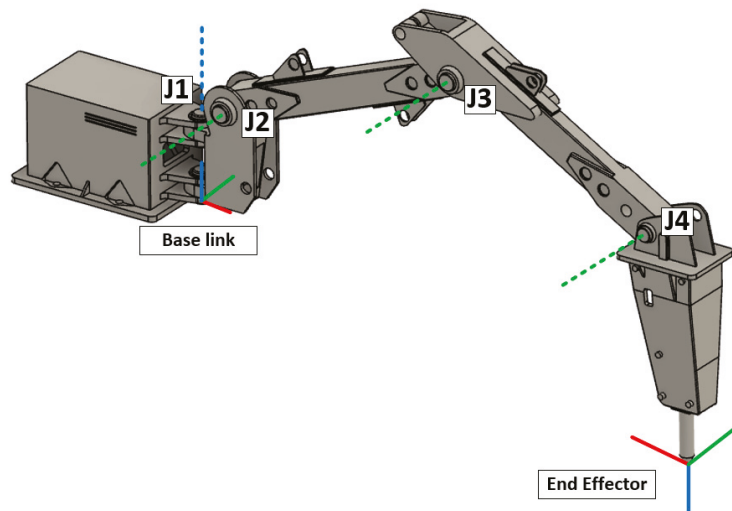


Figure 12. Simulated kinematic model of the impact hammer. J1, J2, J3, and J4 correspond to the hammer joints. The X-axis is shown in red, the Y-axis in green, and the Z-axis in blue.

The pseudo inverse method is implemented to reduce the dimension of the Jacobian, which is originally declared in Cartesian space {x, y, z, roll, pitch, and yaw}. The kinematic model of impact hammer is generated by a kinematic tool (*moveit*) and allows for simplifying the number of the states on the Cartesian space to {x, y, z, pith}.

2.8. Haptic Controller

The haptic control has the task of alerting the user of possible collisions using the haptic device. The form of warning that has been implemented is to restrict the range of movement of haptic device (control axes). In this way, as the effector gets closer to collision, the restriction to the movement of the axes will be greater, exerting a counter force to the user if they want to pass the calculated limits.

Therefore, the restriction of movements to the axes of the haptic device will allow on the one hand to alert the user of possible collisions (and how close the hammer is to collision), and on the other hand, it will be able to slow down the hammer by limiting the possible movement actions.

In the same way as the haptic control interface, the limitation sent to the haptic controller depends on three states of haptic operation:

Rectangular cuboid and semi-sphere regions without intersection with the collision point cloud: The haptic control does not exert any counteracting force to the operator's actuation. In this case, the user will not feel any restriction on the movement of any of the control axes;

4. Rectangular cuboid intersects the collision point cloud: In this case the movement of each axis is limited, depending on the remaining distance to the object to collide. The axel movement limitation is:

$$Axis_{\max possible\ pose} = Dist_{object} \cdot (Axis_{\max pose} - Axis_{init pose}) + Axis_{init pose}. \quad (5)$$

Thus, as the effector approaches the object to collide (distance in range {0,1}), the control will limit the movement of its axes, opposing the user's action. In case the user operates under the calculated limit, he/she will not feel the movement limitations of the control.

5. Semi-sphere region intersects the collision point cloud and the cuboid does not. When this condition is activated, the movement is limited to 50%, which means that the operator will feel that the control only allows them to move the axes, normally the first half in each axis. When trying to exceed the second half, the haptic control will exert a repulsive force opposite to the operator's direction.

2.9. User Interface

The user interface is used to provide the operator with multiple perspectives of the grid material. Here the operator can move the point cloud presented to him, and adjust the view to their liking, in order to improve the perception of the objects within the hammer workspace. The visual output of this interface is in the form of a visualization of the point clouds, previously processed by the modules described above. Figure 13 shows the interface implemented in this project. The interface shows the model of the UR5 robot, which can be rotated as desired by the user of the interface, and the cloud of collision points (in yellow). This interface includes visual signals to inform the operator of the current operation conditions.

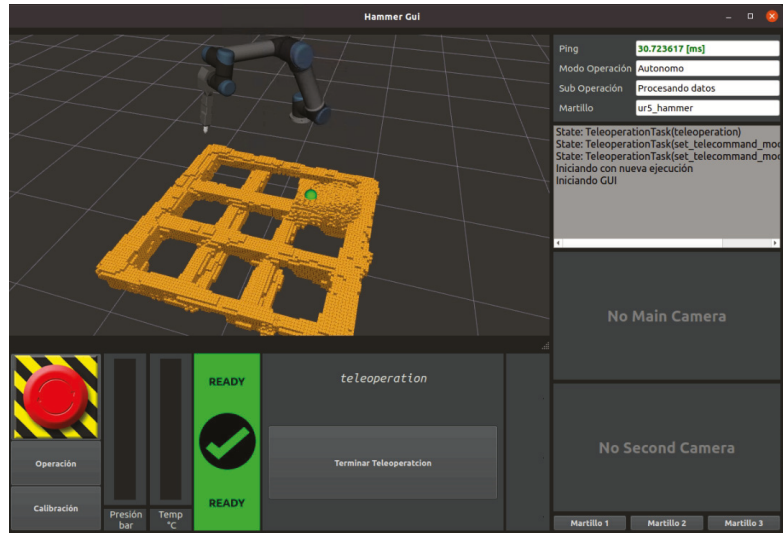


Figure 13. User interface. The yellow shows the visual output of the processing of the point clouds obtained by the sensors.

3. Results

3.1. Experimental Setup

As a proof of concept, we built a realistic laboratory setup. The testing environment used for validation consists of a real, scaled grizzly with real rocks. The scales of both elements are in relation of 1:3. For the impact hammer, an UR5 robotic arm was used with a scale of 1:5, just as a real impact hammer. For having a similar kinematic model rather than a real rock breaker, two articulations of the UR5 were fixed, and the other four were tele-operated (see Figure 14). The UR5 robot was mounted, related to the grizzly, in the configuration used in a real mining operation. Figure 15 shows how the UR5 robot was installed for the experimental tests. Next to it, there is a metallic test grizzly to scale, with material (rocks) on it.

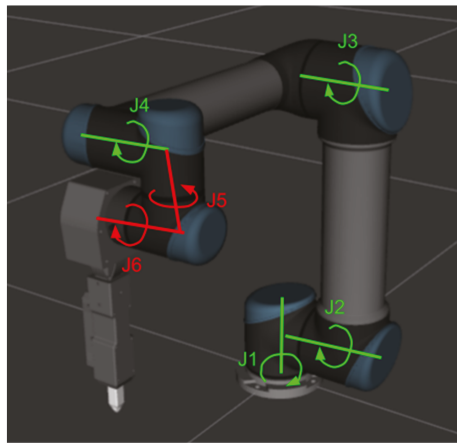


Figure 14. Articulated model of the adapted UR5. The green joints (J1, J2, J3, and J4) correspond to the articulated joints. The red (J5 and J6) correspond to the fixed joint of the kinematic model.



Figure 15. Experimental operation of the rock breaker set-up.

Two mounting-bases, each one containing a camera and a 3D LIDAR, are placed around the operation environment. This allows for the sensors to be located above the operational set-up, pointing to the grizzly and covering it completely (see Figure 16). In this work, Livox model MID-40 3D LIDARS and Arecont model AV5225 PMIR visible spectrum cameras are used.

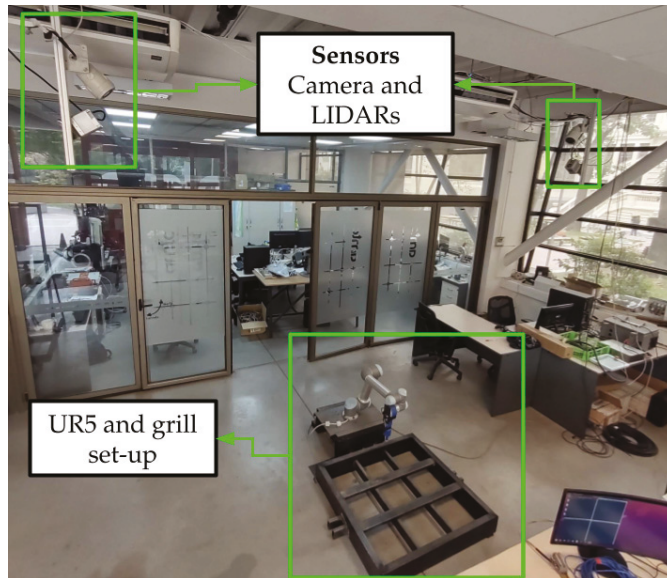


Figure 16. Experimental set-up.

To test the haptic part, 2 Brunner CLS-E controls were used, which allowed up to four joints to be controlled simultaneously (each joystick has 2 degrees of freedom). For convenience, these were installed on a chair used for the tele-operation. Figure 17 shows an operator using the UR5 robot, the Brunner haptic controls (installed on a chair), and the user interface.



Figure 17. User operating the UR5 arm, using the haptic controls (installed on a chair), and the user interface.

3.2. System Configurations to Be Tested

The main purpose of the reported experiments is to validate that the haptic system facilitates the control of the impact hammer to perform the task of positioning the tip of the end-effector above the rock, without colliding. That means that it improves the tele-operation experience.

Several tests are recorded doing this task with different configurations of the haptic system in order to study the impact of this feature. In particular, three different configurations of the system are tested. The configuration combines a different feature of the haptic system, which are:

1. Type of control: Determine how the machine will control the arm (impact hammer). In normal tele-operation, the user directly controls each joint of the arm. In the proposed system, the user directly controls the pose of the end-effector;
2. Use of the 3D reconstruction of the environment: This feature considers the use of the 3D reconstruction to get a multi-view of the operational environment. In normal tele-operation of the impact hammer, just one or two views are used, which are obtained directly by the cameras;
3. Use of haptic feedback: This feature will give the operator the feedback of the joystick to avoid colliding with the grill or other rocks

Finally, these features are combined in different configurations, A to D, which are defined in Table 1.

Table 1. Types of system’s configurations used in tests.

Configuration Id	Type of Control	Use 3D Reconstruction	Use Haptic Feedback
Configuration A	Joints	No	No
Configuration B	End effector	No	No
Configuration C	End effector	Yes	No
Configuration D	End effector	Yes	Yes

3.3. Tasks

Predefined tasks are elaborated to generate a set of tests that can be replicated by different operators. This way, a predefined task contains the following information:

1. The initial state of the arm;
2. The target position;
3. A collision point cloud: The same collision space will be used in the same task. This way the operator has the same level of difficulty for avoiding the grizzly and rocks.

The experiment consists of generating the predefined environment test and giving the operator a target position. The operator has to move the arm using the different configurations of the system. An example of a pre-defined task is shown in Figure 18.

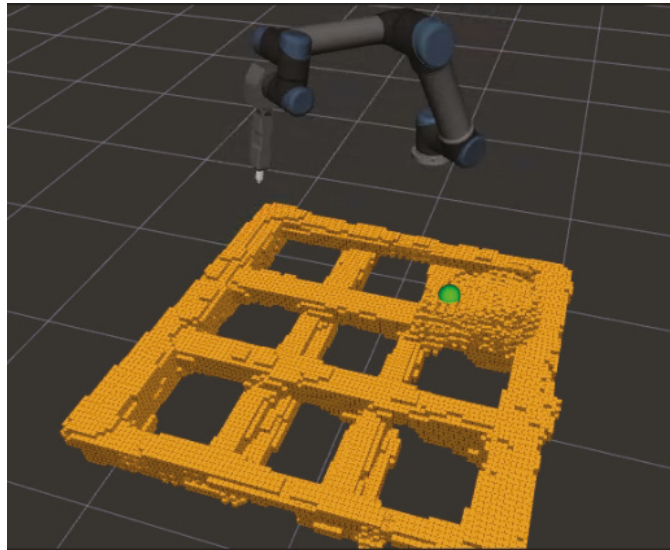


Figure 18. A pre-defined experimental test (Task 1): The arm tip is located out of the grill. The avoiding environment (the point cloud in boxes) consists of a single rock over the grill. The green circle corresponds to the target goal.

3.4. Experimental Results

Each configuration was evaluated using two different tasks, executed by two different non-expert operators. In total, 20 trials were carried out. The obtained results are shown in Table 2.

Table 2. Table with metrics of performance of different tasks with different configurations.

Task and Configuration Id	Average Success Rate	Average Reaching Time [s]	Average Tip Path Length [m]
Task 1, conf. A	0%	—	—
Task 1, conf. B	25%	28.66	0.947
Task 1, conf. C	100%	20.07	1.005
Task 1, conf. D	100%	10.70	0.633
Task 2, conf. A	25%	47.16	1.920
Task 2, conf. B	50%	37.80	1.077
Task 2, conf. C	80%	34.45	1.250
Task 2, conf. D	100%	24.71	1.080

A task is considered successful if the operator was able to move the end effector to the target position with a precision of at least 0.015 [m], without colliding with the grizzly or rocks. If the operator cannot reach the target position, the task is considered a failure.

Table 2 shows the average success rate, representing the percentage of successful tasks. The average reaching time shows how many seconds it takes for the operator to reach the target. The average tip path length represents the distance in meters that the tip of the end-effector has to move to reach the desired target position. It can be observed that the success rate increases as more features are integrated into the system. In Task 1 conf. A, the operators were not able to achieve the target point without colliding with the rocks. The integration of more features allows the operator to complete the task and reduce collisions. Configuration D allows complete completion of the task without collisions because the haptic system does not allow commands that generate collisions to be executed.

In order to illustrate how the system operates, Figure 19 shows the trajectories of the tip for a different configuration when executing task 2. It can be seen that the trajectory becomes simpler as more features of the system are added.

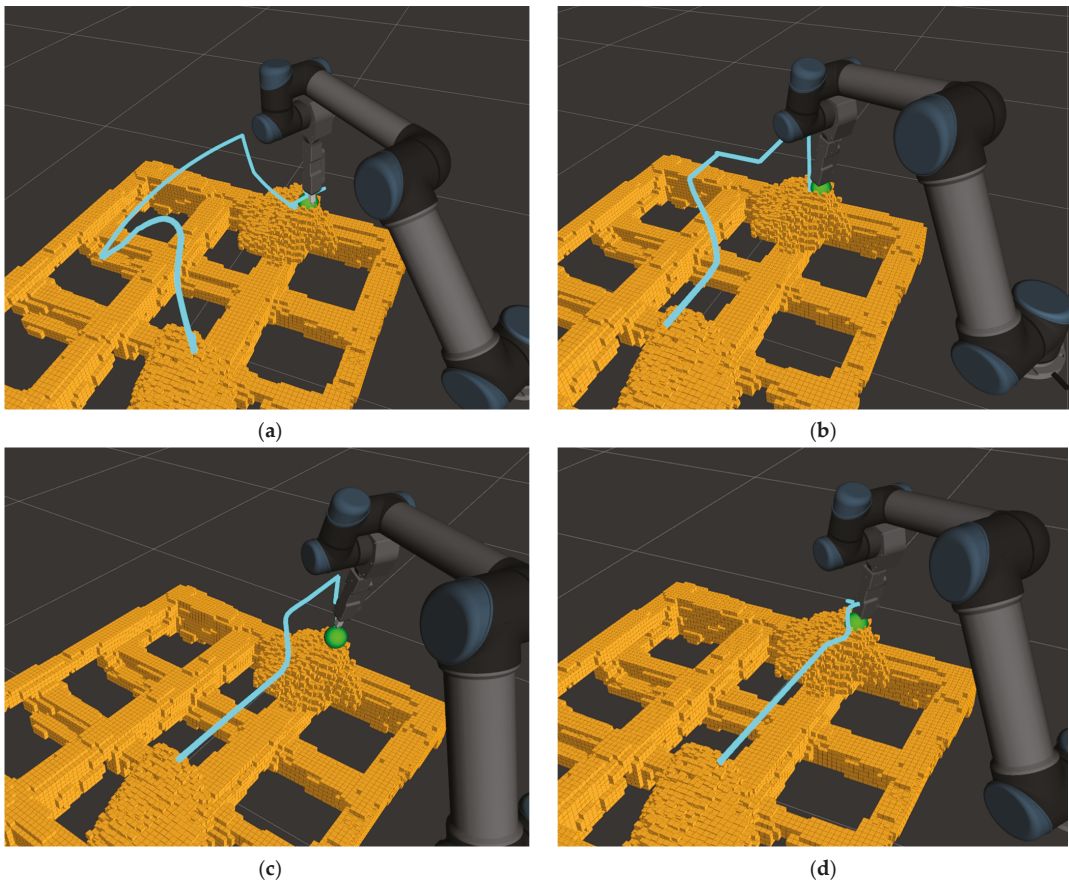


Figure 19. Different hammer tip trajectory paths of task 2 (same operator). The path is on a cyan line. (a) Conf. A (user controls each joint of the arm). (b) Conf. B (user controls the pose of the end-effector). (c) Conf. C (user controls the pose of the end-effector and 3D reconstruction of the environment) (d) Conf. D (user controls the pose of the end-effector, 3D reconstruction of the environment, and haptic feedback).

4. Discussion

The reported experiments show that the developed haptic tele-operation system allows for improving the tele-operation experience of the user. The processing of the different point clouds acquired using 3D LIDARs manages to correctly separate the rocks from the grizzly. The haptic controller and control interface manage to provide enough force feedback to the operator to get their attention, so that they can perceive elements close to the end-effector, thus increasing the perception of the environment in the operating area. In addition, the user interface improves the visual perception of the elements, improving the control over the robot, even if the haptic feedback part is not used.

The results depicted in Table 2 allow for comparing the different configurations, and in this way the different features that the proposed haptic tele-operation system includes.

When comparing configurations, A and B, a significant improvement of the performance of the task is observed when the end-effector controller is used (configuration B). The main reason for this, is that controlling an arm by actuating directly the joints is a difficult task that requires much preparation and practice. This way, controlling the end-effector makes the operation more affordable for new users.

When comparing configurations B and C, it can be observed a slight improvement in the case of using a 3D model of the environment (configuration C). The reason for this is that in case B, the operator cannot change the perspective, and therefore, in some cases it is not possible to have a proper visualization to avoid colliding. When the operator uses configuration C, they have the possibility of changing the viewpoint and in this way avoid collisions.

Finally, when comparing configurations C and D, a more notorious performance improvement is detected in the case of using haptic feedback. The reason for that is that the operator does not have to worry about colliding with the environment or rocks. This produces that the operator is able to increase the speed, and can move the arm without worrying about collisions.

By comparing the different configurations using the hammer tip trajectories (see Figure 19), it is possible to visualize the effect of each configuration on the end effector path. In this aspect, configuration A and B generate a larger deviation when approaching the target, due to the poor perspective that make it more comfortable to first raise the end-effector and then approaching the target from a higher position in order to get a better perspective. In contrast, configurations C and D generate a path that approach the end-effector directly to the target, as they are able to manage the perspective, but then finish with different maneuvers of positioning. In particular, configuration D presents a slight rise over the rock when approaching as a result of haptic feedback that allow the operator to perform fine maneuvers more easily, unlike configuration C which tends to perform wider maneuvers at the final stage of the positioning process.

Similar to [9–11], the results show that using a haptic system improves the performance and safety of performing a certain task using a robot manipulator. In addition to analyzing the number of collisions, our study evaluates the level of precision of the task being executed, and this type of consideration is not studied in other works and it is relevant for the context of this type of task in particular, which is positioning the chisel for breaking rock. In addition, our system uses the virtual environment to generate multi-perceptive views to aid in performing a task. This feature is not analyzed in the other studies describing haptic tele-operation based on point-clouds [9–11].

It should also be noted that although the development is intended for industrial environments with a real impact hammer, a small-scale robotic arm was used for the laboratory tests, which implies different behaviors due to the different construction of these elements (electric vs. hydraulic). This may imply that for final implementation, major adjustments may be needed for the control and haptic feedback modules of the system.

5. Conclusions

This paper addressed the haptic tele-operation of impact hammers. The proposed haptic tele-operation system is based on a 3D model of the environment, which is used to estimate repulsion forces that are transferred to the operator via a haptic device, so that the hammer does not collide with the structures of the mine. The system also allows for identifying the oversized boulders deposited on the grizzly, notifying the operator every time the orepass is blocked, as well as providing them with different 3D views of the environment.

A proof of concept was presented using a scaled setup. The reported experiments show that the use of a 3D model of the environment and the use of haptic feedback improves the tele-operation experience. When using the proposed system, operators were able to increase the success rate of the tele-operation task, and at the same time to reduce the completion time and the length of the path that the end-effector must follow.

As part of our future work, we will validate the tele-operation system using a real impact hammer operating in a real production environment of an underground mine. In that environment, it is expected that several aspects of the operation will emerge which are not reproducible or were not considered in our laboratory environment. The analysis of these new experiments allows for improving and adapting the haptic feedback algorithm to be applied in production systems. In addition, the system will be validated with expert rock breaker operators in order to be able to quantify the real operational impact of its use. Given that a professional operator has a higher level of expertise in controlling the hammer with the current limitations of traditional systems, the performance improvement could be different in relation to a non-expert operator.

Further improvement will focus on the use of haptic feedback to develop assistive tele-operation to guide the operator to follow optimal trajectories. Assistive tele-operation requires defining hammer tip trajectories for successful positioning and providing haptic feedback when the operator uses commands that move the hammer tip away from these trajectories.

Author Contributions: Conceptualization, J.R.-d.-S. and M.C.; methodology, J.R.-d.-S., D.C. (Daniel Cárdenas), D.C. (Diego Carvajal) and M.C.; software, D.C. (Daniel Cárdenas) and D.C. (Diego Carvajal); validation, D.C. (Daniel Cárdenas) and D.C. (Diego Carvajal); formal analysis, J.R.-d.-S., D.C. (Daniel Cárdenas), D.C. (Diego Carvajal) and M.C.; investigation, J.R.-d.-S., D.C. (Daniel Cárdenas), D.C. (Diego Carvajal) and M.C.; resources, J.R.-d.-S. and M.C.; data curation, D.C. (Daniel Cárdenas) and D.C. (Diego Carvajal); writing—review and editing, J.R.-d.-S., D.C. (Daniel Cárdenas), D.C. (Diego Carvajal) and M.C.; supervision, J.R.-d.-S.; project administration, M.C.; funding acquisition, J.R.-d.-S. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the Chilean National Research Agency ANID under project grant Basal AFB180004 and FONDECYT 1201170.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study is contained in the article itself.

Acknowledgments: We thank Isao Parra, Mauricio Mascaro, Francisco Leiva, and Patricio Loncomilla for their valuable discussions and support.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Cermak, G. Multimedia Quality as a Function of Bandwidth, Packet Loss, and Latency. *Int. J. Speech Technol.* **2005**, *8*, 259–270. [[CrossRef](#)]
2. Ohm, J.; Sullivan, G.J.; Schwarz, H.; Tan, T.K.; Wiegand, T. Comparison of the Coding Efficiency of Video Coding Standards—Including High Efficiency Video Coding (HEVC). In *IEEE Transactions on Circuits and Systems for Video Technology*; IEEE: New York, NY, USA, 2012; Volume 22, pp. 1669–1684. [[CrossRef](#)]

3. Lampinen, S.; Niu, L.; Hulttinen, L.; Niemi, J.; Mattila, J. Autonomous robotic rock breaking using a real-time 3D visual perception system. *J. Field Robot.* **2021**, *38*, 980–1006. [[CrossRef](#)]
4. Takahashi, H.; Sano, K. Automatic detection and breaking system for boulders by use of ccd camera and laser pointer. *Fragblast* **1998**, *2*, 397–414. [[CrossRef](#)]
5. Hubert, G.; Dirdjosuwondo, S.; Plaisance, R.; Thomas, L. Teleoperation at freeport to reduce wet muck hazards. *MassMin* **2000**, *2000*, 173–179.
6. Rusu, R.B.; Cousins, S. 3D is here: Point Cloud Library (PCL). In Proceedings of the 2011 IEEE International Conference on Robotics and Automation, Shanghai, China, 9–13 May 2011; pp. 1–4. [[CrossRef](#)]
7. Han, X.-F.; Jin, J.; Wang, M.-J.; Jiang, W. Guided 3D point cloud filtering. *Multimed. Tools Appl.* **2018**, *77*, 17397–17411. [[CrossRef](#)]
8. Ryden, F.; Chizeck, H. A Proxy Method for Real-Time 3-DOF Haptic Rendering of Streaming Point Cloud Data. *IEEE Trans. Haptics* **2013**, *6*, 257–267. [[CrossRef](#)] [[PubMed](#)]
9. Valenzuela-Urrutia, D.; Muñoz-Riffo, R.; Ruiz-del-Solar, J. Virtual Reality-Based Time-Delayed Haptic Teleoperation Using Point Cloud Data. *J. Intell. Robot. Syst.* **2019**, *96*, 387–400. [[CrossRef](#)]
10. Tang, X.; Zhao, D.; Yamada, H.; Ni, T. Haptic interaction in tele-operation control system of construction robot based on virtual reality. In Proceedings of the 2009 International Conference on Mechatronics and Automation, Changchun, China, 9–12 August 2009; IEEE: New York, NY, USA, 2009; pp. 78–83.
11. Xu, X.; Song, A.; Ni, D.; Li, H.; Xiong, P.; Zhu, C. Visual-haptic aid teleoperation based on 3-D environment modeling and updating. *IEEE Trans. Ind. Electron.* **2016**, *63*, 6419–6428. [[CrossRef](#)]

Article

A Multiorder Attentional Spatial Interactive Convolutional Neural Network (MoAS-CNN) for Low-Resolution Haptic Recognition

Kailin Wen ^{1,2}, Jie Chu ^{1,*}, Yu Chen ^{1,3}, Dong Liang ^{1,3}, Chengkai Zhang ² and Jueping Cai ^{1,*}¹ School of Microelectronics, Xidian University, Xi'an 710071, China² Suzhou Honghu Qiji Electronic Technology Co., Ltd., Suzhou 215008, China³ The 54th Research Institute of China Electronics Technology Group Corporation, Shijiazhuang 050081, China

* Correspondence: chujie@xidian.edu.cn (J.C.); jpcai@mail.xidian.edu.cn (J.C.)

Abstract: In haptic recognition, pressure information is usually represented as an image, and then used for feature extraction and classification. Deep learning that processes haptic information in end-to-end manner has attracted attention. This study proposes a multiorder attentional spatial interactive convolutional neural network (MoAS-CNN) for haptic recognition. The asymmetric dual-stream all convolutional neural network with integrated channel attention module is applied for automatic first-order feature extraction. Later on, the spatial interactive features based on the overall feature map are computed to improve the second-order description capability. Finally, the multiorder features are summed to improve the feature utilization efficiency. To validate the MoAS-CNN, we construct a haptic acquisition platform based on three-scale pressure arrays and collect haptic letter-shape (A–Z) datasets with complex contours. The recognition accuracies are 95.73% for 16×16 , 98.37% for 20×20 and 98.65% for 32×32 , which significantly exceeds the traditional first- and second-order CNNs and local SIFT feature.

Keywords: haptic recognition; convolutional neural network; channel attention; spatial interactive second-order feature; multiorder feature

Citation: Wen, K.; Chu, J.; Chen, Y.; Liang, D.; Zhang, C.; Cai, J. A Multiorder Attentional Spatial Interactive Convolutional Neural Network (MoAS-CNN) for Low-Resolution Haptic Recognition. *Appl. Sci.* **2022**, *12*, 12715. <https://doi.org/10.3390/app122412715>

Academic Editor: Rocco Furfieri

Received: 30 September 2022

Accepted: 29 November 2022

Published: 12 December 2022

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Haptic interpretation is an important component of perception, providing real-time feedback on changes in the external environment, and has been widely used in practical applications, such as smart machines, wearable devices, and human–computer interaction [1,2]. The essence of haptic recognition is the recharacterization and feature extraction of pressure information to obtain multiple properties of the contact object, which is the basis for subsequent actions such as grasping, manipulating, and moving. The generic method is to consider the haptic information as a two-dimensional image. Therefore, visual data-based approaches are introduced for understanding haptic information. SIFT, SURF, chain code and other descriptors combined with clustering are applied for haptic recognition [3,4]. Pohtongkam et al. applied the BoW technique using SIFT for feature extraction and k-nearest neighbors (KNN) for evaluation to achieve object recognition by a tactile glove [5]. These local feature methods require manual feature design for specific task and are labor-intensive. Recent research has proved that convolutional neural networks (CNNs) are suitable for two-dimensional information and can automatically learn features and recognize objects [6,7]. Therefore, the haptic information can be ported to the CNN for automatic recognition. Gandarias et al. proposed classical transfer CNN models combined with transfer learning to identify large items [8]. Polic et al. designed a CNN encoder structure to reduce the dimensionality of the optical-based tactile sensor image output [9]. Cao et al. trained an end-to-end CNN for tactile recognition using residual orthogonal tiling and pyramid convolution ensemble [10].

Although the existing CNN methods can achieve better recognition performance than traditional artificial features, there are still the following challenges. Firstly, the size and pixels'

amount of haptic image depend on the density and area of the pressure sensor, and thus they are at least two orders of magnitude lower than visual RGB images [11,12]. Current mainstream CNNs are constructed for high-resolution visual images. The nonlinear fitting capability is enhanced by deepening the network [13]. However, the low-resolution haptic image limits the configured network depth, so the features extracted by shallow CNN are insufficient. Secondly, the inevitable nonideal effects of the sensing element itself reduce the accuracy of the original information mapping. Due to the flexible requirements and complex manufacturing process of the sensing elements, the pseudo-outputs caused by elastic coupling and restricted response range are inevitable, resulting in blurred pressure images [14]. In addition, the fineness of the haptic image relies on the sensitivity and response range of the sensor, which is much lower than that of the visual image. Different objects show similar shapes, so more distinguishable features are needed. The traditional shallow CNNs are not adequate in feature extraction capability and feature utilization efficiency.

To address the low-resolution and blur problem in haptic perception, a multiorder attentional spatial interactive convolutional neural network (MoAS-CNN) is proposed and a pressure information acquisition platform is built (flowchart of the overall framework is shown in Figure 1). The former improves the nonlinear fitting capability of the network by deepening the network and adding channel responses to enhance the representation of first-order high-level features; introducing spatial interactive second-order features to enhance the representation of edges and fusing multiorder features to enhance the efficiency of feature utilization. The latter validates the proposed method by constructing haptic letter shapes with complex edges.

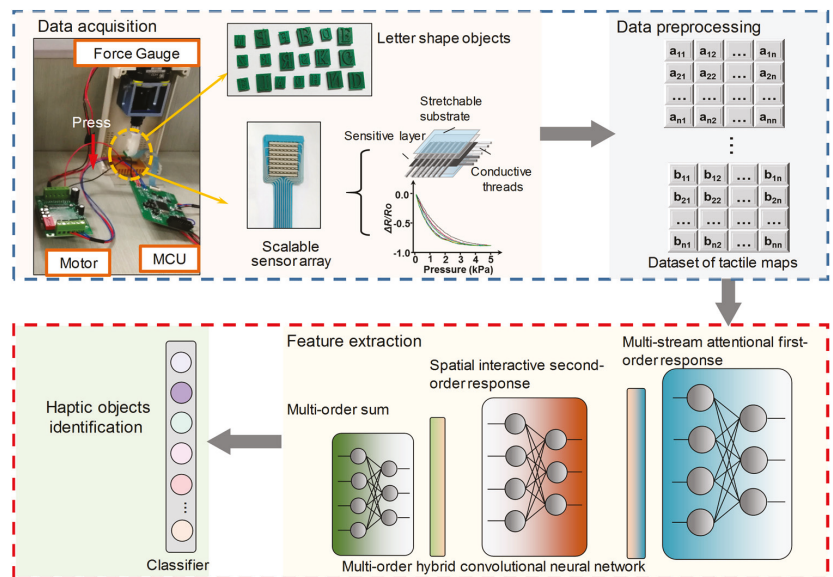


Figure 1. Flowchart of the overall framework.

The remainder of the paper is organized as follows. Section 2 discusses the nonideal characteristics about haptic images and the inadequacy of shallow CNN. In Section 3, the MoAS-CNN is proposed and each part is described. The Section 4 is dedicated to validate the proposed method using a three-scale sensor array and comparison experiments with other first-order and second-order CNNs. The discussion and conclusion are in Sections 5 and 6.

2. Problem of Insufficient Shallow CNN for Haptic Images

The original haptic information mapping of the sensing elements plays a crucial role in recognition. However, the following challenges remain. The density and area limitations as well as elastic coupling cannot be avoided due to the complex process and adhesion requirements of the sensor. In addition, the depth of field of an image depends on the sensitivity and response range of the sensing element. The resulting pixel and mapping quality of haptic images are both lower than those of visual images. The letter shape has complicated edges, which can sufficiently illustrate the issue. As shown in Figure 2a, the haptic images of 26 letter shapes are acquired by a 32×32 array sensor. It can be observed that the haptic images are low-resolution accompanied by blurring. Different categories show similar shapes, such as “V” and “Y,” “O” and “D.” The haptic image has only one channel and is grayscale, so the histogram is used as a quantitative measurement. As shown in Figure 2b, the grayscale distribution of different categories is in statistics, and the histograms of different letter shapes are easily confused. To further quantify the similarity, the Bhattacharyya coefficient is used as the proxy between each of the 26 letter shapes and is calculated as:

$$Cor(I_A, I_B) = \sum_{i=1}^n \sqrt{I_B(x, y) I_A(x, y)} \tag{1}$$

where $I(x, y)$ denotes the gray value of the haptic image at (x, y) . A large positive number indicates a strong correlation between different pressure images and potential confusion. As shown in Figure 2c, different haptic shapes show strong positive interactions. Most of the correlation coefficients are in the range of 0.7–0.85, with certain categories reaching above 0.9, including K and X, G and Q, and O and U.

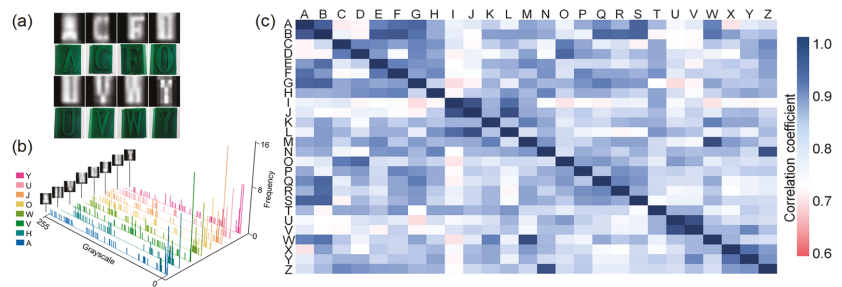


Figure 2. Nonideal effects of haptic images. (a) The letter-shape samples obtained by a 32×32 sensor array, (b) the grayscale distribution histograms of the samples, (c) the Bhattacharyya coefficient of the 26 letter shapes.

The current mainstream CNNs are designed for visual images that improve feature fitting by deepening the network. However, these nonideal effects of haptic images lead to a reduction in the differences between categories, so more distinguishable features are desired. Nevertheless, constrained by the sensor density and integrated area, the pixels of a haptic image are usually at 10^2 (RGB is above 10^4), making it impossible to improve nonlinear fitting by deepening the network, and only shallow networks can be configured [10,11]. Therefore, shallow CNNs with more powerful fitting ability need to be explored.

3. Haptic Recognition Method

3.1. MoAS-CNN Framework

Aiming to improve the feature description and utilization efficiency of the network, a shallow MoAS-CNN is constructed. Figure 3a illustrates the proposed structure, where spatial interactive-based second-order features enhance the nonlinear response of the network and a hybrid strategy of summing up the multiorder features makes the extracted

features fully utilized. It consists of three parts: first-order feature extractor, second-order feature generation, and multiorder feature hybridization.

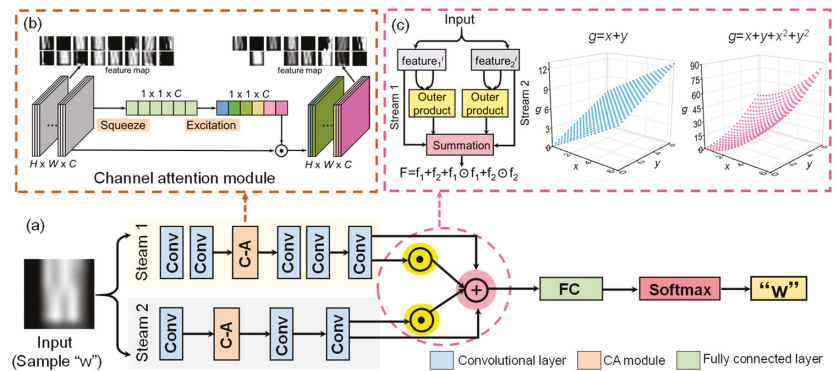


Figure 3. The MoAS-CNN framework. (a) The overall framework, (b) the squeeze–excitation channel attention module, (c) the spatial interactive feature generation and multiorder summation.

In particular, for low-resolution pressure inputs, the samples contain limited information and additional dimensionality reduction of the features is not expected. In our case, the pooling layers are removed and only the convolutional layers are retained to reduce the feature dimensionality loss [15,16]. An all-convolutional dual-stream neural network with the channel attention module inserted is constructed as a first-order feature extractor. The channel attention module based on squeeze and excitation operations is added for improving the first-order feature response. For second-order response, a cross-stream spatial interactive feature is generated to improve the feature nonlinear description. High-order features have been proved to be more sensitive to texture and edges [17,18]. Six $\{3 \times 3\}$ convolutional kernels (steam1) and three $\{5 \times 5\}$ convolutional kernels (steam2) are applied. Multiorder features are fused to enhance the utilization of different orders of features by summation. Features of different orders have different emphases, and complementary utilization promotes the overall nonlinearity of the network without wasting information [19]. This is beneficial for applications where the original information mapping is not sufficient or the network depth is limited.

3.2. All-Convolutional Neural Network-Based First-Order Feature Extractor with Channel Attentional Module Inserted

An asymmetric all-convolutional dual-stream CNN is configured as a feature extractor to automatically extract first-order features. CNN streams of different structures focus on respective priorities, which can fully mine image features. Mathematically, the feature maps at i th layer are calculated as:

$$f^i = \varphi(w^i \otimes f^{i-1} + b^i) \tag{2}$$

where φ denotes the activation function *ReLU*, and w and b represent the convolutional kernels and bias [20].

To highlight the emphasized parts of the features, the channel attention module including squeeze and excitation operations is inserted after the convolution layer [21]. As shown in Figure 3b, the feature maps are assigned scaling according to the channel importance to improve the feature representation. For a set of features $f \in \mathbb{R}^{h \times w \times c}$, the

squeeze operation is performed to obtain the global distribution $Z(1 \times 1 \times c)$, reflecting the features response over the channels. The specific mathematical expression is as follows:

$$Z = \frac{GP(f^c)}{h \times w},$$

$$GP(f^c) = \sum_{i=1}^h \sum_{j=1}^w f^c(i, j) \tag{3}$$

Here, the global average pooling is chosen to compress the feature into real numbers by spatial dimension. For excitation operation, Z is fed into two fully connected layers to further learn factor S :

$$S = \varphi(fc(Z)) \tag{4}$$

where φ denotes the activation function *ReLU* and fc represents the fully connected layer. These attention scalings are assigned to the initial feature map to obtain the rescaled feature map \tilde{f}^i :

$$\tilde{f}^i = S^i \times f^i \tag{5}$$

The attentional module is capable of suppressing the 2D features with lower response in the channel domain and instead increase the 2D features with higher response. After the squeeze and excitation, the feature maps are visualized in Figure 3b, and the “light and dark” changes of some feature maps can be clearly observed.

3.3. Spatial Interactive Second-Order Feature

To increase the nonlinear expression of the network, a cross-flow spatial interactive feature is proposed as a second-order response. In contrast to traditional second-order features captured by different channels at the same location, spatial interactive features in this work are generated by convolving single-channel feature maps at different streams with each other. Different stream branches focus on different extraction priorities, so the proposed method concerns more on the intrinsic relationship between the overall features in different streams. For tiny and low-resolution pressure inputs, further exploration of the interactions between different streams is necessary in the presence of network depth limitation.

The cross-flow spatial interactive feature generation is shown in Figure 4a. The extracted first-order features of stream1 and stream2 are represented as f^{stream1} and f^{stream2} . To make the interstream interaction more adequate, the original features of stream1 f^{stream1} are reconstructed without loss as $f^{1\text{-reconstruction}}$. Specifically, the f^{stream1} are split by interval sampling and stitched together, transforming the information on width and length to the channel dimension with no information loss (shown in Figure 5). In the case of this work, the dimension of the f^{stream1} $\{20 \times 20 \times 64\}$ is adjusted to $f^{1\text{-reconstruction}}$ $\{5 \times 5 \times 1024\}$. The reconstructed feature of stream 1 $f^{1\text{-reconstruction}}$ and the features of stream2 f^{stream2} are subjected to an interactive operation, achieving a second-order feature.

The obtained second-order feature is performed in a sum pooling step to finally obtain the cross-flow spatial interactive feature $f^{2\text{-order}}$:

$$f^{2\text{nd-order}} = \sum_{i=1}^{nc} (f^{1\text{-reconstruction}} \otimes f^{\text{stream2}}) \tag{6}$$

To further improve the cross-flow spatial interactive feature representation, $f^{2\text{-order}}$ is normalized by element-wise signed square root followed by L_2 regularization as $f^{2\text{-order-norm}}$ [16]:

$$f^{2\text{nd-order-norm}} = \frac{\text{sign}(f^{2\text{-order}}) \times \sqrt{f^{2\text{-order}}}}{\left\| \text{sign}(f^{2\text{-order}}) \times \sqrt{f^{2\text{-order}}} \right\|_2} \tag{7}$$

where *sign* represent the symbolic function.

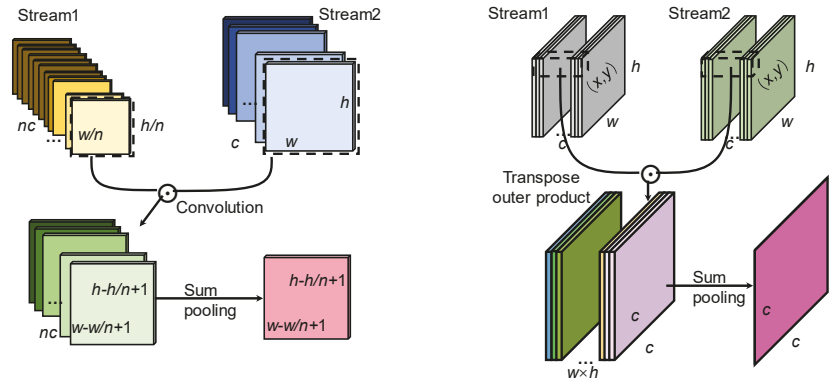


Figure 4. The generation of cross-flow spatial interactive feature. (a) The generation of proposed spatial interactive feature; (b) traditional second-order feature generation.

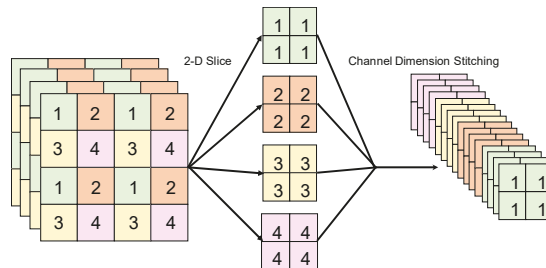


Figure 5. Schematic diagram of first-order feature reconstruction.

3.4. Multiorder Feature Hybrid

Different orders of features have different emphases and can be utilized in a more complementary way. Without adding additional parameters, the multiorder feature hybrid strategy is proposed to enhance the efficiency of feature utilization. The features of different orders are summed to obtain the fused features, as shown in:

$$y_{fusion} = f^{steam1} + f^{steam2} + g(f^{steam1} \odot f^{steam1}) + g(f^{steam2} \odot f^{steam2}) \quad (8)$$

where y_{fusion} stands for the fused feature and the $g(\bullet)$ is regularization (batch normalization here [22]). Intuitively, as shown in the upper right part of Figure 3c, only segmented linear functions are obtained when no second-order terms are added, which means that the nonlinearity appears only in a few 1D subspaces of the 2D plane \mathbb{R}^2 . However, if the second-order terms are added, nonlinearity exists $\mathbb{R}^2 \doteq [0, \infty)^2$. The multiorder features allow the efficiency of feature utilization to be enhanced while keeping the network depth constant, facilitating the nonlinear fitting of the network.

4. Results

4.1. Experiment Setup

As shown in the upper part of Figure 1, a data-acquisition system containing three-scale pressure arrays, a force gauge, a motor driver, and a microcontroller is set up. To demonstrate the effectiveness of the proposed method, we choose letter shapes with complex contours as task targets. The stamp ($0.8 \text{ cm} \times 1.0 \text{ cm}$, $1.0 \text{ cm} \times 1.5 \text{ cm}$) with raised letter shape is fixed on the force gauge to press the sensor array. The output matrix is normalized to the range 0–255 to form haptic grayscale images. The reliability of our

prototype allowed us to collect 500 samples for each letter-shape category using three scales of 16×16 ; 20×20 ; and 32×32 arrays, in a random pressure range of 0–5 kPa, angle and position. The data samples of each letter shape are shown in Figure 6. To avoid overfitting, data augmentation is applied to expand the sample quantity to 1500 per letter category. All the CNNs are constructed with MatConvNet framework of Matlab 2017 on an Intel Core i5-6500@3.2GHz CPU. The specific parameters of the network are set as shown in Table 1. The weights are initialized by the Xavier initialization scheme and optimized by Adam algorithm with hyperparameters $\epsilon = 10^{-8}$, 0.9, 0.999 [23,24]. The three-scale datasets are put into the MoAS-CNN with fivefold crossover to verify the performance of MoAS-CNN. The learning rate is 0.001, batch size 50, and training epoch 120.

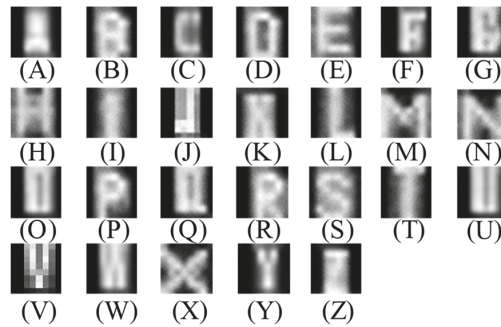


Figure 6. The dataset of 26 letter-shape samples (A–Z).

Table 1. Detailed parameters of CNNs.

Method	MoAS-CNN		Bilinear CNN		Traditional CNN
Layer	Stream1	Stream2	Stream1	Stream2	
Input	$32 \times 32 \times 1$		$32 \times 32 \times 1$		$32 \times 32 \times 1$
Conv1	$3 \times 3 \times 8$ $3 \times 3 \times 16$	$5 \times 5 \times 16$	$3 \times 3 \times 8$ $3 \times 3 \times 16$	$5 \times 5 \times 16$	$3 \times 3 \times 8$ $3 \times 3 \times 16$
CA module	Global Pooling FC: $1 \times 1 \times 16$ FC: $1 \times 1 \times 16$		Global Pooling FC: $1 \times 1 \times 16$ FC: $1 \times 1 \times 16$		/
Conv2	$3 \times 3 \times 32$ $3 \times 3 \times 32$	$5 \times 5 \times 32$	$3 \times 3 \times 32$ $3 \times 3 \times 32$	$5 \times 5 \times 32$	$3 \times 3 \times 32$ $3 \times 3 \times 32$
Conv3	$3 \times 3 \times 64$ $3 \times 3 \times 64$	$5 \times 5 \times 64$	$3 \times 3 \times 64$ $3 \times 3 \times 64$	$5 \times 5 \times 64$	$5 \times 5 \times 64$
second-order(output)	$14 \times 14 \times 1$		$64 \times 64 \times 1$		/
FC1	$1 \times 1 \times 120$		$1 \times 1 \times 120$		$1 \times 1 \times 120$
FC2	$1 \times 1 \times 26$		$1 \times 1 \times 26$		$1 \times 1 \times 26$

4.2. Performance of MoAS-CNN

The training and test process in each epoch are recorded. The training losses and test losses are shown in Figure 7a. It can be seen that the losses of all datasets decrease significantly and stabilize after 80 epochs. Accordingly, as shown in Figure 7b, the recognition accuracies achieved are 95.73% for 16×16 , 98.37% for 20×20 and 98.65% for 32×32 . This can be attributed to the fact that more information is captured through higher density and larger area, and thus the extracted features are more separable.

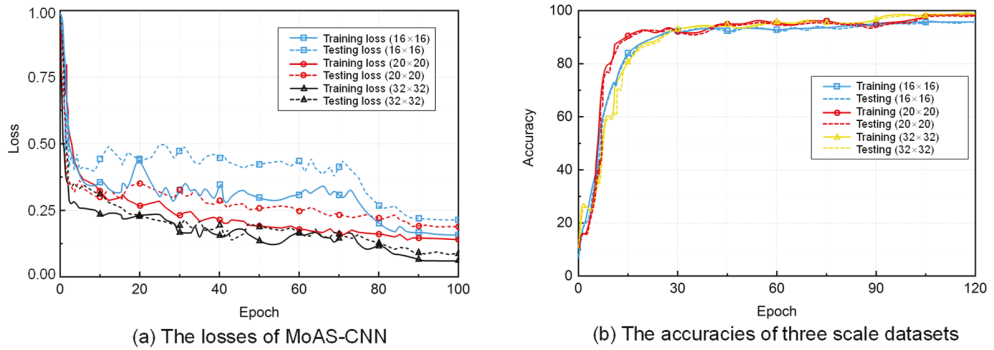


Figure 7. The performance of MoAS-CNN on three-scale datasets. (a)The losses in training and testing process, (b) the recognition accuracies in training and testing process.

We also record the gradients for all three datasets simultaneously, as shown in Figure 8a–c. The gradient values become larger in the backpropagation, and there is no gradient disappearance. Among them, the gradient value of Conv1 is the largest in the 32 × 32 dataset, making the loss converge quickly. There is no overfitting or underfitting, which indicates that our model and datasets are reasonable.

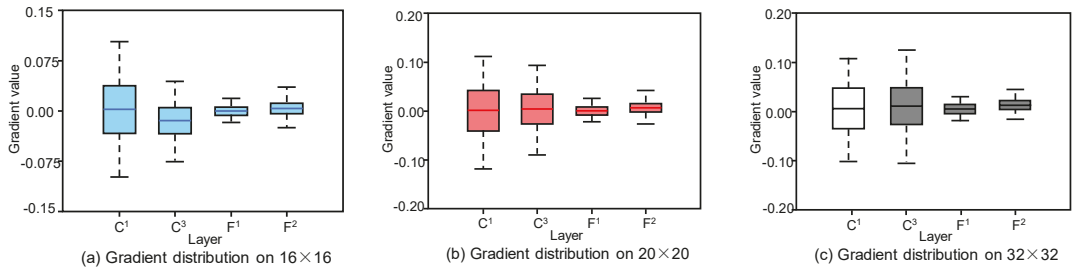


Figure 8. Gradient distribution on 16 × 16, 20 × 20, 32 × 32 datasets during training.

Figure 9 shows the test confusion matrix of the 32 × 32. Obviously, shapes with simple contours are more distinguishable, e.g., I. Conversely, complex contours are more prone to confusion, especially with similar categories, e.g., Q and G (mean Bhattacharyya correlation coefficient of 0.89).

4.3. Contribution of Spatial Interactive Second-Order Feature and Multiorder Hybrid

To further explore the contribution of the proposed spatial interactive second-order feature and hybrid order strategy, individual streams of MoAS-CNN as well as different structures are compared, and the results are shown in Figure 10. To validate the spatial interactive second-order features, we compared the recognition accuracies for $f^{\text{stream}1}$ and $f^{\text{stream}1} + f^{\text{stream}1} \otimes f^{\text{stream}1}$ with $f^{\text{stream}2}$ and $f^{\text{stream}2} + f^{\text{stream}2} \otimes f^{\text{stream}2}$, respectively. Due to the introduction of the spatial interactive features, the accuracy of stream1 increased by 3.36%, 4.11%, and 1.96% on the three scales, and stream2 was 1.98%, 2.54%, and 1.75%. This fully demonstrates the effectiveness of the spatial interactive-based second-order feature.

4.4. Performance Comparison

For a more comprehensive and fair comparison, a traditional first-order CNN, a bilinear based second-CNN, and a local feature SIFT method are chosen [16,25]. The bilinear CNN based on the traditional bilinear method with same parameters as the proposed method is constructed and the same for the first-order CNN (detailed in Table 1). The results are shown in Figure 11. It can be seen that the accuracies of MoAS-CNN are significantly higher than those of other methods on all sensor scales, especially for smaller-scale pressure arrays. This result illustrates the effectiveness of the proposed MoAS-CNN for low-resolution haptic image. For 16×16 , the MoAS-CNN obtained 95.73%, while the traditional bilinear CNN was 92.46%, traditional CNN 89.79%, and local feature method 67.24%. Since the inputs of both 16×16 and 20×20 scales are too small to be applied in SIFT, which can only extract 3–5 key points, the recognition effect is very limited.

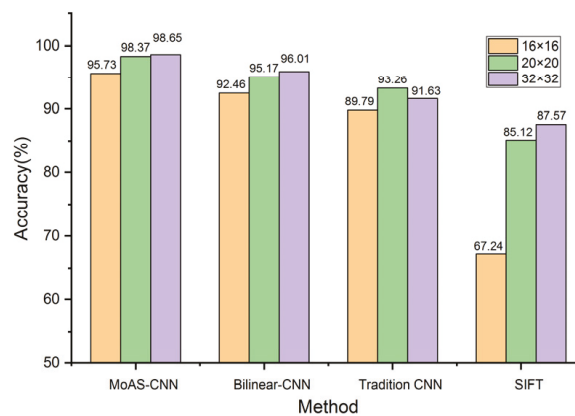


Figure 11. Performance comparison with other first-order, second-order and traditional local feature methods.

For the samples that the model failed to identify, the true labels and predicted probabilities are shown in Figure 12, marked in orange. These haptic images are collected in a random pressure range of 0.5–3 kPa, angle and position. The inference value of some letter categories with similar shapes are very close, indicating that the differences between features are not obvious. As a comparison, 10 volunteers (7 males and 3 females, age [20,30]) are invited to identify the misclassified samples and the results are marked in green. It can be observed that some samples were not recognized by the neural network or humans, as shown in Figure 12a–f, and others were recognized by humans, but not by neural networks, as in Figure 12g–i. No volunteer can completely reclassify all failed samples. The misclassification can be mainly attributed to the following: firstly, the restricted sensor density leads to fine edges not being captured; secondly, the elastic coupling of the flexible sensors causes the unpressed pixels around the pressed pixels to be deformed, producing pseudo-outputs. Thirdly, features are not sufficiently mined through the MoAS-CNN and the features of similarly shape categories are not learned separately.

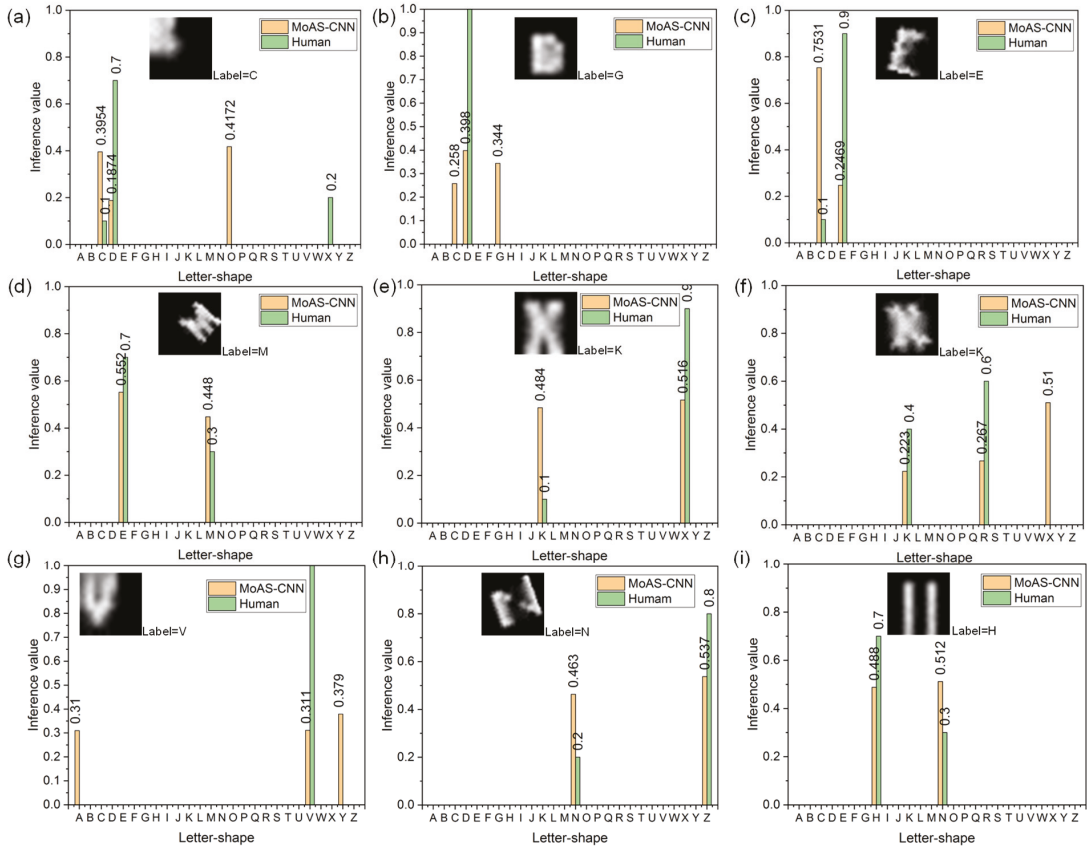


Figure 12. Comparison of recognition results between MoAS-CNN and human for some difficult samples. (a) Inference results of sample with label “C”, (b) inference results of sample with label “G”, (c) inference results of sample with label “E”, (d) inference results of sample with label “M”, (e) inference results of sample with label “K”, (f) inference results of sample with label “K”, (g) inference results of sample with label “V”, (h) inference results of sample with label “N”, (i) inference results of sample with label “H”.

5. Discussion

Haptic technology provides real-time feedback on external force changes through flexible sensors mounted on or inside mechanical surfaces, providing an aid to scene understanding beyond vision. Therefore, recognition based on haptic information has become important for smart devices and has wide application prospects in human–computer interaction, intelligent machinery, and biomedicine. The intelligence level of human–computer interaction is improved by adding haptic perception. In addition, distribute pressure data feedback and analysis is important in many industrial sensing fields. Generally, the haptic information is converted into a grayscale image and then transferred to image-processing methods for subsequent recognition. However, compared with visual images, haptic images are still challenging because of their small dimensions, low resolution, and blurred edges. The experimental results show that MoAS-CNN can realize accurate haptic perception, and the highest accuracy of haptic letters with complex shapes was 98.65%. The cross-stream spatial interactive feature as a second-order response and multidior feature fusion can significantly improve the extraction of haptic shapes by the network.

Although the proposed model has proven to be accurate and effective, it does have limitations. Through the analysis of the misclassified samples, some different classes with similar shapes cannot be clearly classified by the network. This may be attributed to two factors. Firstly, the feature extraction of the proposed method for haptic images is insufficient, and some information is lost in the extraction of high-level semantic information, especially for images with lower resolution, such as 16×16 . Secondly, the characteristics of the haptic task itself, including the low resolution of the sensor array, the small number of pixels, and the blurred edge, lead to inaccurate mapping of the original shape, and the multiple meanings of individual images, resulting in lower-than-average recognition rates for some categories.

More importantly, smallness and low-resolution inputs are also present for practical vision tasks due to factors such as suboptimal raw data and environmental interference. Therefore, we tried to validate our method on another small general-purpose dataset CIFAR-10 [26], and the recognition accuracy was 98.81%. Compared with the current state-of-the-art results, it is lower than the largest-scale visual transformer methods [27,28] and essentially on par with the deeper CNN architecture [29]. This study is based on touch recognition based on a single image after touch completion. In the future, we plan to study recognition methods based on multiple dynamic touches and further improve the recognition performance of touch perception through reasonable optimization algorithms.

6. Conclusions

A framework called MoAS-CNN is proposed to address the challenge of low-resolution haptic recognition based on a pressure sensor array. The three contributions of our recognition model are to firstly apply a dual-stream CNN integrated with the channel attention module to automatically extract first-order features and increase the response and number of features. Secondly, a spatial interactive second-order feature is introduced to depict the second-order information to improve the feature extraction capability without additional feature downscaling. Thirdly, by exploring the complementarity of features of different orders, a multiorder hybrid strategy is developed to enhance the efficiency of feature extraction. To validate the model, a self-built acquisition platform based on a three-scale pressure array was built and haptic images of letter shapes (A–Z) with complex edges were collected. The results showed that 95.73%, 98.37%, and 98.65% accuracy was achieved at the scales of 16×16 , 20×20 , and 32×32 , respectively. The accuracy of the proposed method has a significant advantage over traditional second- and first-order CNN-based methods, as well as manual feature methods. Furthermore, in addition to haptics, low-resolution inputs are common in practical applications because of the inherent limitations of various types of sensing elements and nonideal factors. Our approach provides a new general framework that can be easily extended to different systems.

Author Contributions: Conceptualization, K.W. and J.C. (Jie Chu); methodology, K.W., J.C. (Jie Chu) and J.C. (Jueping Cai); software, Y.C. and D.L.; validation, C.Z.; writing—original draft preparation, K.W. and J.C. (Jie Chu); writing—review and editing, J.C. (Jie Chu) and J.C. (Jueping Cai). All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the National Natural Science Foundation of China (62274123), Natural Science Basic Research Plan in Shaanxi Province of China (2021ZDLGY02-01), and Wuhu-Xidian University Industry-University-Research Cooperation Special Fund (XWYCYX-012021003).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All data generated or analyzed during this study are included in this paper or are available from the corresponding authors on reasonable request.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Li, Q.; Kroemer, O.; Su, Z.; Veiga, F.F.; Kaboli, M.; Ritter, J. A Review of Tactile Information: Perception and Action through Touch. *IEEE Trans. Robot.* **2020**, *36*, 1619–1634. [CrossRef]
- Uddin, R.; Jamshaid, A.; Arfeen, A. Smart Design of Surgical Suture Attachment Force Measurement Setup Using Tactile Sensor. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 4001512. [CrossRef]
- Luo, S.; Bimbo, J.; Dahiya, R.; Liu, H. Robotic Tactile Perception of Object Properties: A Review. *Mechatronics* **2017**, *48*, 54–67. [CrossRef]
- Luo, S.; Mou, W.; Althoefer, K.; Liu, H. Novel Tactile-SIFT Descriptor for Object Shape Recognition. *IEEE Sens. J.* **2015**, *15*, 5001–5009. [CrossRef]
- Pohtongkam, S.; Srinonchat, J. Object Recognition Using Glove Tactile Sensor. In Proceedings of the 2022 International Electrical Engineering Congress (iEECON), Khon Kaen, Thailand, 9–11 March 2022.
- Voulodimos, A.; Doulamis, N.; Doulamis, A.; Protopapadakis, E. Deep Learning for Computer Vision: A Brief Review. *Comput. Intell. Neurosci.* **2018**, *2018*, 7068349. [CrossRef] [PubMed]
- Li, P.; Wang, D.; Wang, L.; Lu, H. Deep visual tracking: Review and experimental comparison. *Pattern Recognit.* **2018**, *76*, 323–338. [CrossRef]
- Gandarias, J.M.; Garcia-Cerezo, A.J.; Gomez-De-Gabriel, J.M. CNN-Based Methods for Object Recognition with High-Resolution Tactile Sensors. *IEEE Sens. J.* **2019**, *19*, 6872–6882. [CrossRef]
- Polic, M.; Krajacic, I.; Lepora, N.; Orsag, M. Convolutional Autoencoder for feature extraction in tactile sensing. *IEEE Robot. Autom. Lett.* **2019**, *4*, 3671–3678. [CrossRef]
- Cao, L.; Sun, F.; Liu, X.; Huang, W.; Li, H. End-to-End ConvNet for Tactile Recognition Using Residual Orthogonal Tiling and Pyramid Convolution Ensemble. *Cognit. Comput.* **2018**, *10*, 718–736. [CrossRef]
- Wang, S.; Xu, J.; Wang, W.; Wang, G.; Rastak, R.; Molina-Lopez, F.; Chung, J.; Niu, S.; Feig, V.R.; Lopez, J.; et al. Skin electronics from scalable fabrication of an intrinsically stretchable transistor array. *Nature* **2018**, *555*, 83–88. [CrossRef] [PubMed]
- Song, L.; Zhu, H.; Zheng, Y.; Zhao, M.; Tee CA, T.; Fang, F. Bionic Compound Eye-Inspired High Spatial and Sensitive Tactile Sensor. *IEEE Trans. Instrum. Meas.* **2021**, *70*, 7501708. [CrossRef]
- Brahimi, S.; Aoun, N.B.; Amar, C.B. Improved Very Deep Recurrent Convolutional Neural Network for Object Recognition. In Proceedings of the 2018 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Miyazaki, Japan, 7–10 October 2018.
- Chen, M.; Kun, G.; Cheng, W.; Zhang, D.; Feng, W. Touchpoint-tailored ultra-sensitive piezoresistive pressure sensors with a broad dynamic response range and low detection limit. *ACS Appl. Mater. Inter.* **2019**, *11*, 2551–2558. [CrossRef] [PubMed]
- Ruderman, A.; Rabinowitz, N.C.; Morcos, A.S.; Zoran, D. Pooling is neither necessary nor sufficient for appropriate deformation stability in CNNs. *arXiv* **2018**, arXiv:1804.044338.
- Springenber, J.T.; Dosovitskiy, A.; Brox, T.; Riedmiller, M. Striving for simplicity: The all convolutional net. *arXiv* **2014**, arXiv:1412.6806v3.
- Lin, T.Y.; Roychowdhury, A.; Maji, S. Bilinear Convolutional Neural Networks for Fine-grained Visual Recognition. *IEEE Trans. Pattern Anal. Mach. Intell.* **2018**, *40*, 1309–1322. [CrossRef] [PubMed]
- Carreira, J.; Caseiro, R.; Batista, J.; Sminchisescu, C. Semantic Segmentation with Second-Order Pooling. In Proceedings of the 12th European conference on Computer Vision (ECCV), Florence, Italy, 7–13 October 2012.
- Akilan, T.; Wu, Q.; Safaei, A.; Jiang, W. A late fusion approach for harnessing multi-cnn model high-level features. In Proceedings of the 2017 IEEE International Conference on Systems, Man and Cybernetics (SMC), Banff, AB, Canada, 5–8 October 2017.
- Glorot, X.; Bordes, A.; Bengio, Y. Deep Sparse Rectifier Neural Networks. *J. Mach. Learn. Res.* **2011**, *15*, 315–323.
- Hu, J.; Shen, L.; Sun, G. Squeeze-and-excitation networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**, *42*, 2011–2023. [CrossRef] [PubMed]
- Ioffe, S.; Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In Proceedings of the 32nd International Conference on Machine Learning (ICML), Lille, France, 6–11 July 2015.
- Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. *J. Mach. Learn. Res.* **2010**, *9*, 249–256.
- Kingma, D.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2014**, arXiv:1412.6980.
- Lowe, D.G. Distinctive image features from scale-invariant keypoints. *Int. J. Comput. Vis.* **2004**, *60*, 91–110. [CrossRef]
- The CIFAR-10 Dataset. Available online: <https://www.cs.toronto.edu/~kriz/cifar.html> (accessed on 11 November 2022).
- Dosovitskiy, A.; Beyer, L.; Kolesnikov, A.; Weissenborn, D.; Zhai, X.; Unterthiner, T. An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale. In Proceedings of the 2021 International Conference on Learning Representations (ICLP), Colombo, Sri Lanka, 20–27 September 2021.
- Touvron, H.; Cord, M.; Sablayrolles, A.; Synnaeve, G.; Jégou, H. Going deeper with Image Transformers. In Proceedings of the 2021 IEEE/CVF International Conference on Computer Vision (ICCV), Montreal, QC, Canada, 10–17 October 2021.
- Tan, M.; Le, Q.V. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In Proceedings of the 2019 International Conference on Machine Learning (ICML), Long Beach, CA, USA, 9–15 June 2019.

Article

Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell

Tomáš Kot ^{1,*}, Zdenko Bobovský ¹, Aleš Vysocký ¹, Václav Krys ¹, Jakub Šafařík ² and Roman Ružarovský ³

¹ Department of Robotics, Faculty of Mechanical Engineering, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic; zdenko.bobovsky@vsb.cz (Z.B.); ales.vysocky@vsb.cz (A.V.); vaclav.krys@vsb.cz (V.K.)

² Laboratory of Big Data Analysis, IT4Innovations, VSB-Technical University of Ostrava, 17. listopadu 2172/15, 708 00 Ostrava-Poruba, Czech Republic; jakub.safarik@vsb.cz

³ Department of Production Devices and Systems, Slovak University of Technology in Bratislava, Ulica Jána Bottu č. 2781/25, 917 24 Trnava, Slovakia; roman.ruzarovsky@stuba.sk

* Correspondence: tomas.kot@vsb.cz

Abstract: We describe a method for robotic cell optimization by changing the placement of the robot manipulator within the cell in applications with a fixed end-point trajectory. The goal is to reduce the overall robot joint wear and to prevent uneven joint wear when one or several joints are stressed more than the other joints. Joint wear is approximated by calculating the integral of the mechanical work of each joint during the whole trajectory, which depends on the joint angular velocity and torque. The method relies on using a dynamic simulation for the evaluation of the torques and velocities in robot joints for individual robot positions. Verification of the method was performed using CoppeliaSim and a laboratory robotic cell with the collaborative robot UR3. The results confirmed that, with proper robot base placement, the overall wear of the joints of a robotic arm could be reduced from 22% to 53% depending on the trajectory.

Keywords: robot; manipulator; robotized workplace; robotic cell; optimization; wear

Citation: Kot, T.; Bobovský, Z.; Vysocký, A.; Krys, V.; Šafařík, J.; Ružarovský, R. Method for Robot Manipulator Joint Wear Reduction by Finding the Optimal Robot Placement in a Robotic Cell. *Appl. Sci.* **2021**, *11*, 5398. <https://doi.org/10.3390/app11125398>

Academic Editor: António Paulo Moreira

Received: 17 May 2021

Accepted: 7 June 2021

Published: 10 June 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The design of robotized work cells or lines is a complex multidisciplinary task that is influenced by many external factors and conditions. During the designing process, it is necessary to make many decisions that greatly affect the resulting performance of the workplace and its properties, including the cycle time, velocity of individual parts, dynamic effects, vibrations, lifetime, ground area, and energy consumption.

Crucial for the design of a robotic cell is the selection of an appropriate industrial or collaborative robot and its placement in relation to other subsystems. The length of the designing stage of robotic cells is still shortening, which leads to the copying of existing layouts of workplaces with similar parameters and their adaptation to the actual requirements.

Any type of advanced optimization is almost impossible in this approach. However, this is the phase where it is possible to achieve some interesting savings of energy consumption, ground area, or the lifetime of some systems. The recent progress in complex simulation tools, such as Tecnomatix, Matlab—Robotics System Toolbox, CoppeliaSim (V-Rep), Gazebo, and Webots; methods for the creation of digital twins of the designed robotized workplace; and whole manufacturing lines [1] open up new room for the realization of complex multicriteria optimizations in the early stages of design.

Optimizations of robotized workplaces to achieve the reduction of running costs are currently mostly realized in systems that are already in operation. These typically include modifications of the end-point trajectory according to a chosen criterion, such as the manipulation time [2,3] or the overall productivity and running costs of the robot [4].

The optimization of the kinematic properties of the manipulator movement can also lead to the reduction of torques in individual joints and, thus, also the reduction of the

overall energy consumption. For example, the authors in [5] used interval analysis to find the global minimum jerk trajectory of a robot manipulator in a joint space using cubic splines, resulting in a smoother and vibration-free robot movement.

A similar approach was proposed in [6], where the objective function included not only the integral of the squared jerk values but also the total execution time of the trajectory. The authors in [7] used the 12-phase sine profile for trajectories of a fast-moving robot, which leads to a more stable and accurate movement without sudden changes of torques, albeit at the cost of a slightly higher overall energy consumption. Trajectory optimization with the single goal of cycle time reduction using the chicken swarm optimization (CSO) method was described in [8].

Another optimization goal is the reduction of the workplace layout size [9] because the usable area of the factory building is extremely valuable. There are also some methods for the multicriteria optimization of robotic work cells or lines for energy consumption—even for lines with up to 12 robots, based on the Gurobi simplex method [10]. Another method of multi-robot cell optimization for time and energy reduction using a custom mechatronic model of the robots in Modelica/Dymola was described in [11]. The authors in [12] proposed a methodology for assembly line energy consumption optimization based on the implementation of energy-optimal trajectories, and, in [13], this method was improved by modification of the actuator brake release time for additional energy consumption reduction.

A systematic methodology for the on-site identification and energy-optimal path planning of an industrial robot is presented in [14] with a focus on a specific type of ABB industrial robot. Improved robot programming reduced the energy consumption compared to the built-in controller routines by up to 4%.

Further improvements and torque reduction for a robotized work cell can also be achieved by modification of the position and orientation of the robot base in relation to the optimized trajectory [15]. This can be interesting, especially for applications where it is not possible to modify the end-point trajectory and velocity for technological reasons (the application of adhesives, edging, welding, etc.). In these cases, it is necessary to create a complex simulation model—a digital twin of the workplace [16,17] and perform a multicriteria optimization.

The Concept of Robot Wear

Gearboxes are often mentioned as the component that is frequently responsible for a failure of rotary machines [18], including industrial robots [19]. The most critical components of a gearbox are the gears and bearings. The deterioration of a gear appears at the teeth [20,21], harmonic drive gears are damaged by fatigue fractures [22], and bearings are typically subjected to failures at the rolling elements or the inner or outer race [21]. The damage accumulates over time and is caused mainly by load (forces and velocity) [23,24], high temperature [25], bad lubrication [26], or manufacturing defects. The performance, accuracy, and lifetime of a robot relies on the good condition of these critical components [27].

The authors in [28] proposed a method for estimating the wear of a robot by monitoring the temperatures in the joints, while [29] suggested a more complex solution, where other parameters are also monitored in order to predict the lifetime of a multi-component system. Other common properties monitored to predict the lifetime of a machine include vibrations and noise [21]. A generic framework for predictive maintenance based on simulation models with degradation curves discovered from real data was proposed in [27]. Our approach, on the contrary, attempts to minimize wear by the use of a quite simple simulation in the design stage of a robotized workplace instead of monitoring an already existing one.

As mentioned above, the gearbox in the robot manipulator joint can be considered the most important source of the wear of the joint. The electric motor in the joint is subject to deterioration as well [30]. As far as the above-mentioned sources of wear are concerned,

the most important is the load combined with movement velocity, because a higher velocity creates higher temperatures. This applies to both the gearbox and the motor. The influence of bad lubrication or manufacturing defects are almost impossible to anticipate or even simulate and, thus, are not considered in this work.

A typical robot joint contains structural bearings that provide the mutual rotational movement of the joints. However, obtaining the values of the reacting forces required to calculate the wear of these bearings is not an easy task, even when using a dynamic simulation. This requires an exact simulation model of the robot with a detailed representation of the inner parts and mechanisms, accurate values of the mass properties, acceptably realistic values of the friction coefficients, and a good dynamic simulation engine. Although it is usually possible to obtain 3D models of commercially available industrial and collaborative robots, and sometimes even simulation models prepared for common simulation systems, these models typically are simplified and do not contain the inner mechanisms of the arms.

On the other hand, wear of the gearbox and motor can be expressed as torque that the motor must generate (and that the gearbox must transfer to the joint) over some trajectory of motion. These values depend on physical quantities that are much easier to obtain from a simulation—the overall path of motion (kinematics) and the required driving torque required to achieve the given acceleration (dynamics).

The hypothesis of our research is that the lifetime of a robot in a robotized workplace can be improved in the design stage by designing the workplace (namely the location of the robot) in such a way to ensure lower wear of the robot joints.

2. Materials and Methods

To determine the optimal placement of a robot manipulator within a robotic cell with the goal of reducing and balancing joint wear, it is necessary to propose a suitable optimization criterion, the whole optimization process, and an experiment to verify the results.

2.1. Optimization Criterion

In this work, we consider only the wear of the robot drive chain (see the previous section). We will also restrict the following notation to rotational joints (which are much more common than translational joints in robotics) and to six degrees of freedom (the most common number in robotics); however, the principles can be applied in general.

An industrial or collaborative robot in a robotized workplace usually performs a limited set of movements. Typically, there is a given trajectory that the robot end-point should follow during the work cycle, and the robot joints are controlled using inverse kinematics to adhere to this trajectory. The idea of wear being caused by a torque acting over a trajectory corresponds with the concept of mechanical work, which can be expressed as

$$W = \int_{\phi_1}^{\phi_2} \tau d\phi, \quad (1)$$

where τ is the magnitude of the torque vector, ϕ is the angle of rotation about the vector representing the joint axis, and ϕ_1 and ϕ_2 are the starting and ending angles of rotation, respectively.

The integral (1) is path-dependent and the mechanical work is defined as the change of energy. Thus, if we assume $\phi_1 = \phi_2$, which is true for a closed-loop trajectory of the robot end-point (all individual joints have to start and end in the same angle of rotation), the resulting value of W would always be equal to zero. It is, thus, more convenient to express mechanical work as the integral of mechanical power over time

$$W = \int_{\phi_1}^{\phi_2} \tau d\phi = \int_{t_1}^{t_2} \tau \omega dt, \quad (2)$$

where ω is the angular velocity of motion and t_1 and t_2 are the starting and ending time, respectively. However, this integral (2) is still path-dependent, and the calculation would result in $W = 0$. Thus, it is necessary to introduce the absolute value of both the torque and the angular velocity

$$W = \int_{t_1}^{t_2} |\tau\omega| dt, \tag{3}$$

which allows us to calculate the work over the whole trajectory, while in fact, considering the trajectory divided into segments separated by the change of direction of movement or the change of the sign of τ . This modification moves away from the concept of the conservation of energy toward the concept of wear caused by mechanical work. The idea is that the drive chain of a robot joint is worn down even when the torque is negative (the motor is actively braking) or when the angle of rotation is decreasing or moving back.

The simulation is numerical with a definitive value of Δt (simulation step size) instead of dt . Therefore, the integral (3) is replaced by a sum

$$W = \sum_{i=0}^n |\tau_i \omega_i| \Delta t, \tag{4}$$

where $i = 0, 1, \dots, n$ is the simulation step, τ_i is the instant torque, and ω_i is the instant angular velocity in the i -th simulation step. The mathematical meaning of the value W is shown in an example in Figure 1.

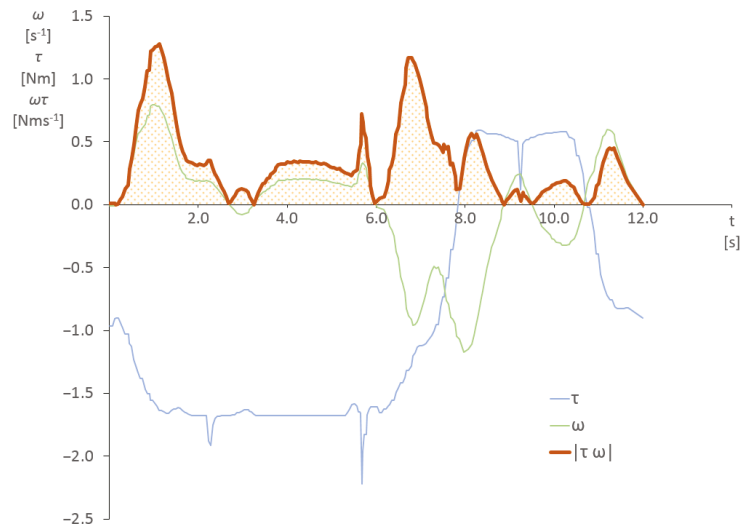


Figure 1. Example of the time progression of joint torque τ and angular velocity ω ; the dotted area represents the meaning of the value W calculated as the integral of absolute value $|\tau\omega|$.

The value of W was calculated individually for each joint over the whole robot trajectory, yielding W_1, W_2, \dots, W_6 for the most common number of 6 degrees of freedom.

We attempted to optimize two factors:

- to minimize the overall wear of all joints, and
- to balance the wear of all joints.

Therefore, it is necessary to consider the values W_j together. However, the joints of a robot manipulator are typically not equal from the mechanical point of view, and their capabilities are different—the lower joints are larger and stronger, and the joints near the end-effector are lighter. The same magnitude of W could, in reality, mean much

higher stress and wear for a smaller joint than for a larger one. We, therefore, introduce a variable called the relative wear factor w_j , which is calculated by normalizing the individual W_j values

$$w_j = \frac{W_j}{\tau_{m_j} \omega_{m_j}}, \quad w_j \geq 0, \quad (5)$$

where $j = 1, 2, \dots, 6$ is the joint number, τ_{m_j} is the maximal permissible torque, and ω_{m_j} is the maximal angular velocity of the j -th joint (both values are specified by the manufacturer of the robot).

To minimize the overall wear of the robot, we chose to use the arithmetic mean of the relative wear factors of all joints and to find the minimal value of this mean,

$$A = \frac{1}{6} \sum_{j=1}^6 w_j, \quad (6)$$

and balance is achieved by finding the minimal value of the standard deviation

$$\sigma = \sqrt{\frac{1}{6} \sum_{j=1}^6 (w_j - A)^2}. \quad (7)$$

The chosen fitness function (optimization criterion) f_f places the same weight on both these factors; therefore,

$$f_f = \frac{A}{2} + \frac{\sigma}{2}. \quad (8)$$

2.2. Optimization Process

The proposed method requires a simulation model capable of evaluating the movement of a robot manipulator through a given end-point trajectory while computing the values of joint angles and torques in individual time steps. This is possible, for example, in the popular robotic simulation system CoppeliaSim (formerly known as V-Rep), which was also chosen for our work.

The scripting capability of CoppeliaSim allows programming a robot's end-point movement through a given trajectory and time, and the built-in physics engine (i.e., Bullet 2.78) is able to check for collisions and evaluate joint torque values. The inverse kinematics (IK) are calculated using the integrated pseudoinverse IK solver. The CoppeliaSim API (Application Programming Interface) framework (*RemoteAPI*) can be used to remotely configure the simulation, which, in our case, includes particularly the process of changing the robot placement relative to the trajectory, as we are trying to determine the optimal placement of the robot. A custom application was written in Visual C++ for this purpose.

Due to the long simulation times in CoppeliaSim, especially in the cases when the IK calculation fails (the robot cannot reach some parts of the given trajectory), all valid positions of the robot relative to the given trajectory were pre-calculated in the C++ application and CoppeliaSim was used only to calculate the fitness function value in those positions. The valid locations were found using a simple kinematic simulation inside the C++ application, which can quickly verify the ability of the robot to fulfill a given task from the specific location, including collision checking.

The search space is represented as a discrete 3-dimensional grid with the spacing in all three dimensions equal to $s = 0.03$ m. The system returns the valid robot positions as a list of points, where each point represents the location of the center of the robot base (see the coordinate system in Figure 2a) in the workspace. Verification of the whole robot trajectory in this simulation system took approximately 1800-times less time than the same task in CoppeliaSim, and invalid robot locations were discarded even faster.

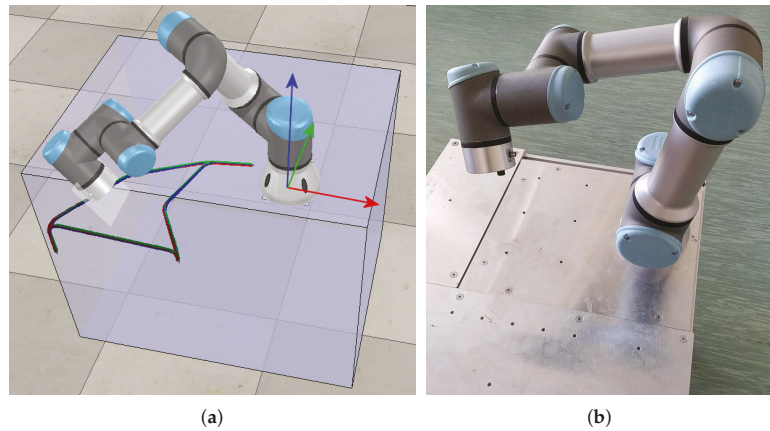


Figure 2. The UR3 robot. (a) The simulation model in the Coppeliasim environment with a path representing the end-point trajectory and a coordinate system in the center of the robot base. (b) The real robot used in the experiments.

For the reproducibility of the research, this supplementary custom simulation system is not necessary, as its main purpose is simply to shorten the overall simulation time. However, it is necessary to have an external application or a script in Coppeliasim that successively places the robot in the locations from the above-mentioned search space (grid), executes the Coppeliasim simulation, and calculates and stores the results.

The reason for choosing a 3-dimensional grid for the potential locations of the robot base instead of a 2-dimensional plane (representing the factory floor) is that the proposed method is intended to be as general as possible, and limiting the search to a 2-D grid could likely miss some interesting solutions. In reality, robots are commonly mounted on a stand, table or a console; therefore, the height of the robot can be chosen. However, the results can be easily limited to, for example, a 2-dimensional plane, if the actual application requires such a limit.

2.3. Experiment Setup

The proposed method was demonstrated and verified on a Universal Robots UR3 collaborative robot with 6 degrees of freedom (see Figure 2), first in a Coppeliasim simulation configured according to the previous chapter, and finally also on a real physical robot. The values of the maximal allowed joint torque τ_{m_j} and joint angular velocity ω_{m_j} (5) for the UR3 robot are listed in Table 1.

Five testing movement paths of the robot end-point were demonstrated, each in two variants with different velocities of the end-point (0.1 and 0.2 m/s), giving a total number of ten trajectories labeled A1, A2, B1, B2, . . . , E2; where A–E is the path type, 1 stands for 0.1 m/s, and 2 stands for 0.2 m/s.

Table 1. The joint parameters for the UR3 robot; τ_{m_j} is the maximal permissible torque, and ω_{m_j} is the maximal permissible angular velocity of the j -th joint.

j	τ_{m_j} [Nm]	ω_{m_j} [s ⁻¹]
1	56	π
2	56	π
3	28	π
4	12	2π
5	12	2π
6	12	2π

The trajectories are a combination of real use-cases and artificially created simple trajectories that contain various elements (linear segments of different lengths, circle arcs, sections with frequent speed, direction changes, etc.), see Figure 3. Various velocities of the end-point cause a difference in the solution of possible robot base locations, because every robot has some velocity limits for individual joints (see Table 1), and they also represent different cases regarding the proposed wear factor calculation.

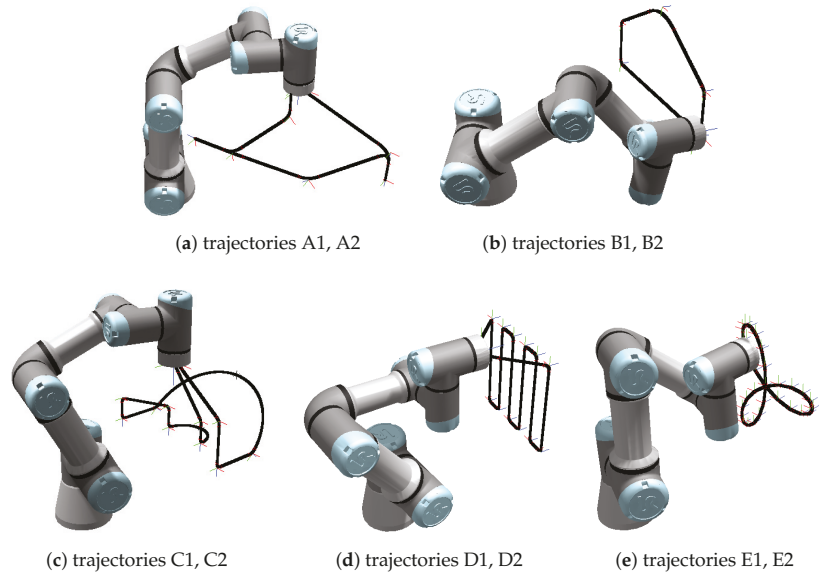


Figure 3. Visualization of the five testing robot end-point paths; the UR3 robot is shown as a scale reference.

The selected trajectories were of a smaller size, and there were no obstacles in the environment (except for the table that the robot was mounted onto), which leads to a larger number of different valid robot locations relative to the trajectory and, thus, also a larger data set of results.

The chosen robot UR3 is a small robot with a reach of only 500 mm and although using larger trajectories combined with several obstacles in the environment would be possible, the valid locations of the robot would be very limited and it would be difficult to make a statistical evaluation. In general, the proposed optimization method does not depend on the size of the trajectory. A longer trajectory, requiring a longer time to traverse, would produce larger values of the discrete integral (4); however, this is true for all robot locations relative to the particular trajectory, and thus the optimization remains valid.

In a real case, the simulation model would have to contain also all collision objects in the workplace, which could limit the valid robot locations considerably—the principle of the method is, however, still applicable. The concept of choosing the robot location is demonstrated on Figure 4 for the trajectory A1. The figure shows the grid of valid robot locations—each blue point represents a possible position of the robot base; the robot is able to reach the whole trajectory from each of these valid locations. For better clarity, the robot is shown in several selected locations.

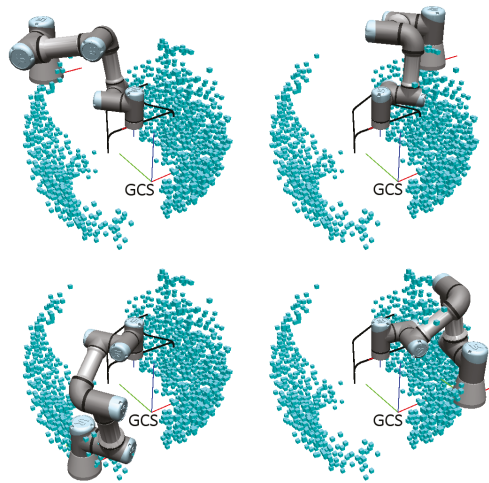


Figure 4. Example of a grid of valid robot locations for a given trajectory (A1). Each blue point represents a possible position of the robot base center point; the robot is displayed in four distinct sample locations. GCS represents the global coordinate system.

3. Results

For each of the ten trajectories, the values of w_0, w_1, \dots, w_6 (5), and then f_f (8) were calculated based on the CoppeliaSim simulation for each particular possible robot location in the grid. The important values are summarized in Table 2, particularly the lowest fitness function value f_f^B in the best robot location, which should, thus, be the selected location for the robot, provided it is physically possible. The table also shows the percentage of improvement that the best location offers compared to the worst location,

$$imp_W^B = 100 \times \left(1 - \frac{f_f^B}{f_f^W} \right), \tag{9}$$

which theoretically represents the greatest possible improvement in robot wear reduction. If we consider a random robot location versus the best one, the average improvement can be calculated compared to the average value

$$imp_A^B = 100 \times \left(1 - \frac{f_f^B}{f_f^A} \right). \tag{10}$$

Figure 5 shows the distribution of f_f values in all possible robot locations for each trajectory in the form of a standard box-plot diagram. The height of each shaded rectangle represents the interquartile range (third quartile minus first quartile), which indicates that 50% of all f_f values lie inside the rectangle. The small circles in each column represent the outliers—the topmost one corresponds to the worst location with f_f^W .

Arrangements of the robot locations in the 3D space around the corresponding trajectories are displayed in Figure 6—the color coding indicates the f_f values using the gradient red–yellow–green, where green is the best location (f_f^B), and red is the worst (f_f^W). This image also shows the positions of the best (“B”), worst (“W”), and three other robot locations (“1”, “2”, and “3”) that will be used later in the experiments on a real robot. Note that the small cubes rendered in Figure 6 as a visual representation of the possible robot locations are shifted away from the ideal grid positions by a small random offset (less than half the grid spacing). This is to achieve better visual clarity by preventing moiré patterns and reducing the concealment of more distant cubes.

Table 2. Results for the 10 testing trajectories showing the number of valid robot locations n , the fitness function value in the best location f_f^B , in the worst location f_f^W , the average value f_f^A ; and improvement of the best location against the worst imp_W^B (9) and the average imp_A^B location (10).

Traj.	n	f_f^B	f_f^W	f_f^A	imp_W^B	imp_A^B
A1	800	0.0730	0.2053	0.1239	64.4%	41.1%
A2	564	0.0773	0.1577	0.1046	51.0%	26.1%
B1	2265	0.0217	0.1090	0.0461	80.1%	52.9%
B2	2236	0.0210	0.1067	0.0448	80.3%	53.1%
C1	2365	0.0769	0.1998	0.1239	61.5%	37.9%
C2	2214	0.0757	0.1828	0.1170	58.6%	35.2%
D1	866	0.0907	0.1994	0.1163	54.5%	22.0%
D2	828	0.0841	0.1829	0.1097	54.0%	23.3%
E1	2224	0.0279	0.1028	0.0532	72.8%	47.4%
E2	2151	0.0247	0.0984	0.0502	74.9%	50.7%

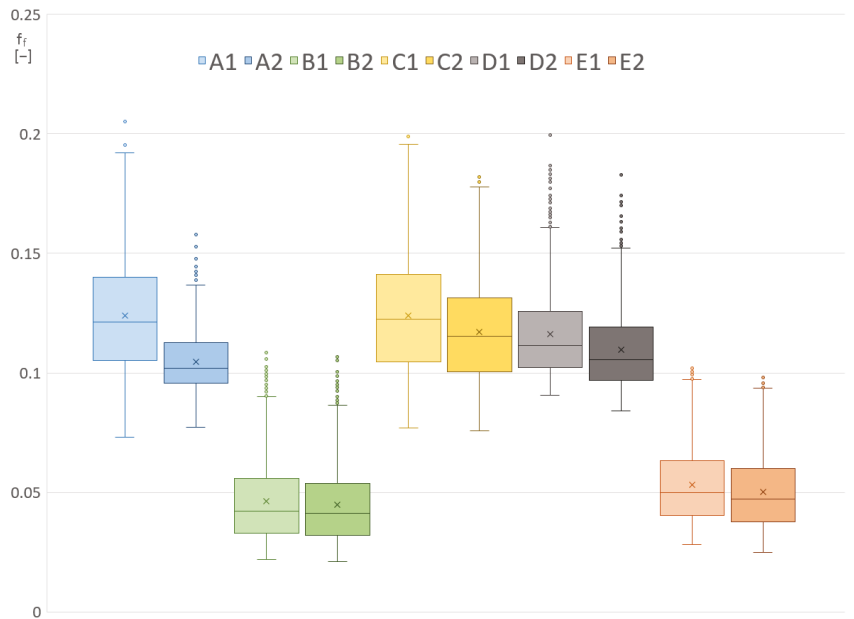


Figure 5. Statistical distribution of the f_f values in all possible robot locations for the ten testing trajectories (standard box-plot diagram—each box is bounded by the first and third quartile, the horizontal line represents the median, the \times represents the arithmetic mean, and circles represent outliers).

To better demonstrate the meaning of the proposed optimization criterion, Figure 7 shows, for the two trajectories A1 and B1, the individual values of the relative joint wear factors w_1, w_2, \dots, w_6 (5) for the best (w_i^B) and worst (w_i^W) robot location, together with the corresponding total fitness function value f_f (8). In the first example (trajectory A1), it is clear that, in the worst location, the third joint was extremely stressed and the six w_i^W values differ quite considerably. In the best location, joints 2, 3, and 6 have almost the same w_i^B values and, although the relative wear factor of the 6th joint increased, the overall f_f^B value was much lower.

A similar situation can be seen in the second image (trajectory B1). Here, the relative wear factor values in the best location are not so well balanced—the overall fitness function value was reduced considerably nonetheless because the average value A (Equation (6)) lowered.

The reason for the big improvement in the third joint relative wear factor w_3 for both these trajectories can be seen in Figure 8—the green and red solid lines show the immediate power values for the best and worst robot location, respectively. The difference in the magnitude is evident.

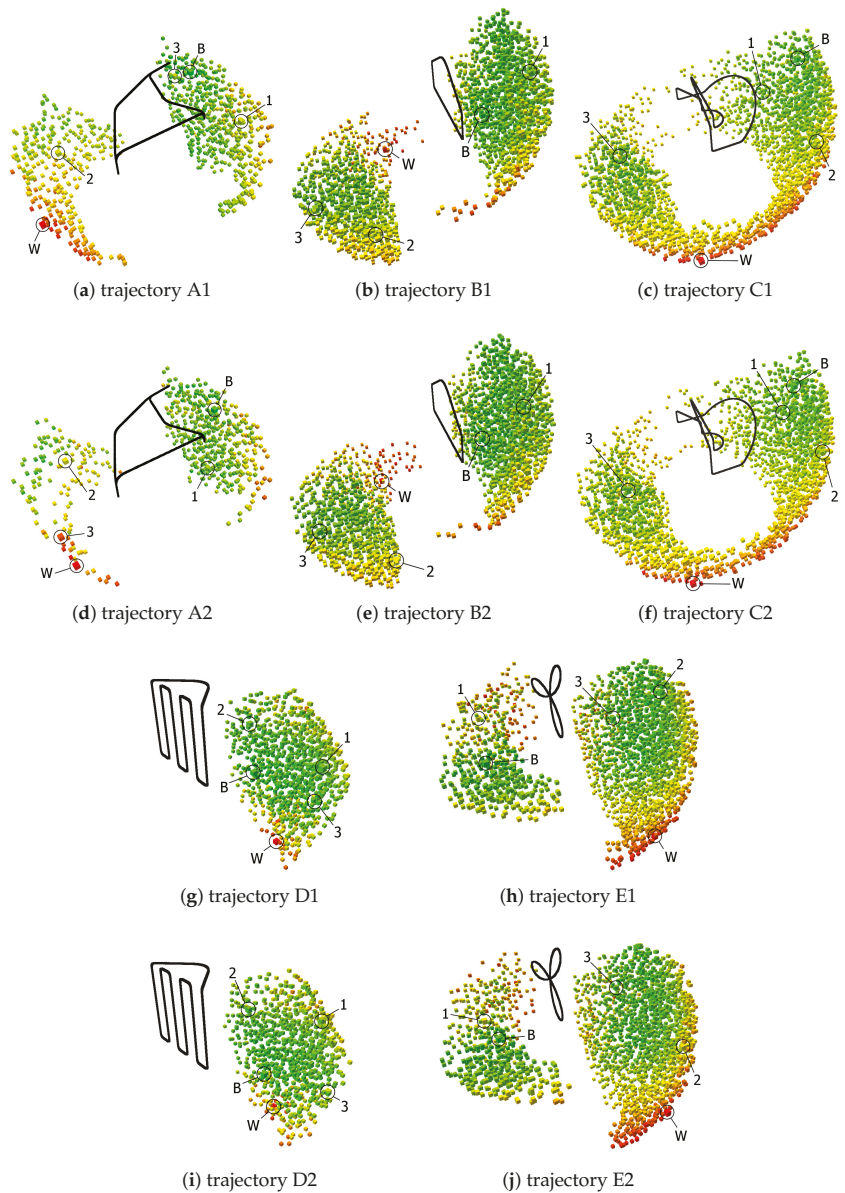


Figure 6. All valid robot locations in the grid; the color-coding indicates the f_f values using the gradient red–yellow–green, where green is the best location (B) and red is the worst (W). Locations numbered 1, 2, and 3 are the three other locations used in the experiment with a real robot.

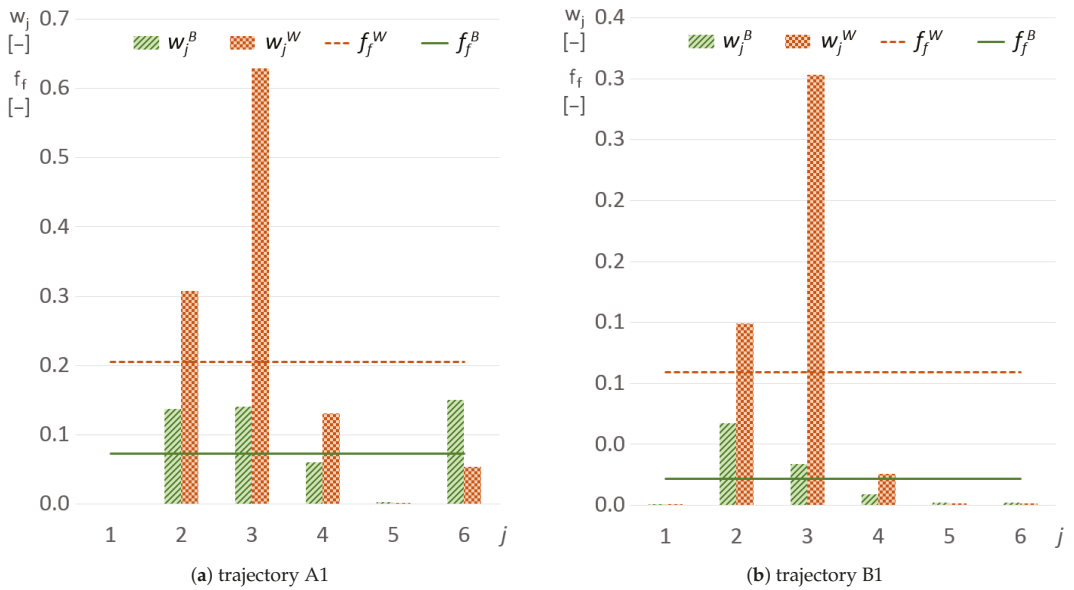


Figure 7. Detailed comparison of the components forming the fitness function value in the best (B, green) and worst (W, red) robot locations for two selected trajectories; displayed are the relative wear factors of each joint (w_j) and the total fitness function value (f_f).

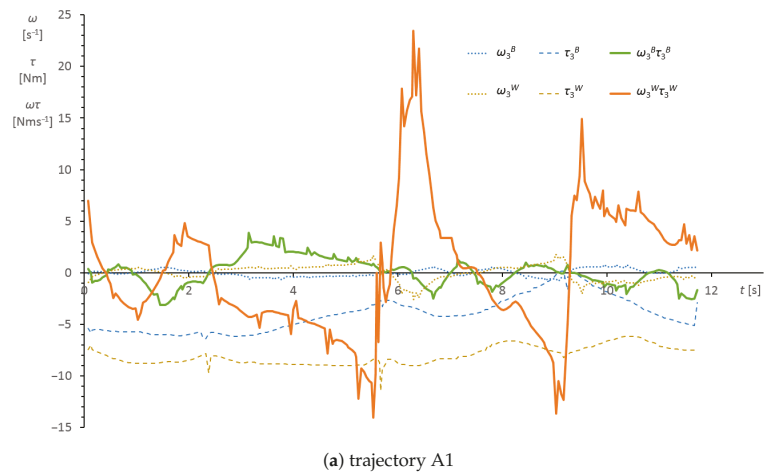
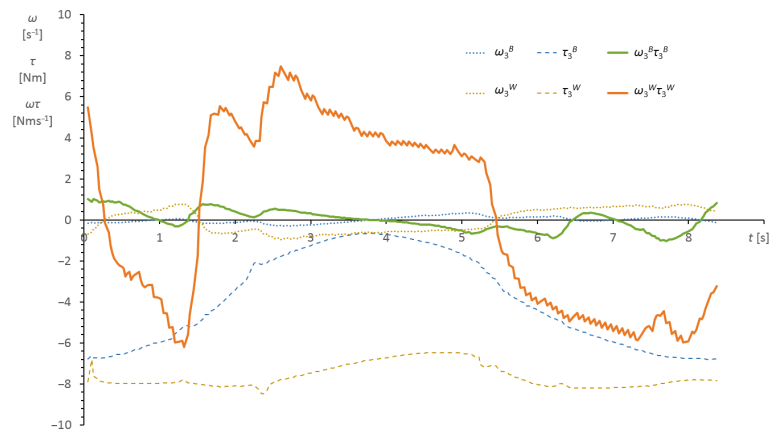


Figure 8. Cont.



(b) trajectory B1

Figure 8. Comparison of the angular velocity ω_3 , torque τ_3 , and immediate power $\omega_3\tau_3$ of the third joint during the whole trajectories A1 and B1 for the best (^B) and worst (^W) robot locations.

4. Verification on a Real Robot

To experimentally verify the impact of the chosen fitness function on the real wear of the robot joints, it would be necessary to perform simultaneous long-term testing on multiple robots and then analyze the mechanical degradation of important components of the joint construction. This type of experiment was not feasible for us at this moment. Instead, we used experiments on a real robot to verify the accuracy of dynamical simulation in CoppeliaSim.

The real experiments were performed on the same robot as was previously used for the simulations—the Universal Robots UR3 collaborative robot. The robotic arm was mounted on a table (Figure 2), and, instead of changing the robot base location relative to the trajectory, the trajectory was appropriately shifted in relation to the robot. There were no tools nor other equipment mounted to the output interface flange, which corresponds to the simulation model described above.

The robot controller (CB3 controller, firmware version 3.10.0) provides state messages via the RTDE (Real-Time Data Exchange) protocol based on TCP/IP communication. This protocol allows reading the actual state of the robot with a frequency of 125 Hz. The RTDE messages can be configured and include the actual and target values of kinematic parameters, such as the position, velocity, and acceleration, as well as the currents of individual joints, the overall current of the whole robot, and the target torque values—which are needed for our experiment.

The experiment was executed for the optimal (best) robot location found by the simulation, for the worst robot location, and for three other locations that were manually selected from the set of possible locations to cover various sections of the whole space. The locations are hereafter referred to as “B” (best), “W” (worst), and “1”, “2”, and “3” (the other locations). The three numbered locations are sorted from the lowest to the highest f_f values according to the simulation results (a lower f_f value indicates a better robot location). For each trajectory and robot location, the robot was programmed to go through the whole trajectory five times in a row. During this time, the integral (sum) was calculated according to (4), and the final value was then divided by five.

All tested real robot locations are depicted in Figure 6. Numerical values of the f_f values in all five locations for all ten trajectories are listed in Tables 3 and 4. Table 3 shows the results from the simulation, the values here are sorted from lowest to highest (“B”, “1”, “2”, “3”, and “W”). Table 4 shows the values from real experiments—as can be seen, the

best locations were identical; the worst locations were also mostly identical, except for trajectory D2.

Table 3. Fitness function values in the five robot locations for individual trajectories—results from the simulation. Color gradient red–yellow–green is used for better visual representation of the value (green is the best, red is the worst).

	B	1	2	3	W
A1	0.0730	0.1256	0.1281	0.1241	0.2054
A2	0.0774	0.1025	0.1156	0.1434	0.1578
B1	0.0217	0.0357	0.0389	0.0484	0.1090
B2	0.021	0.0289	0.0310	0.0588	0.1068
C1	0.0770	0.1074	0.1195	0.1421	0.1998
C2	0.0758	0.0939	0.1040	0.1160	0.1828
D1	0.0907	0.1045	0.1077	0.1187	0.1995
D2	0.0841	0.0967	0.1157	0.1203	0.1830
E1	0.028	0.0358	0.0388	0.0642	0.1028
E2	0.0247	0.0371	0.0585	0.0606	0.0985

Table 4. Fitness function values in the five robot locations for individual trajectories—results from the real robot. Color gradient red–yellow–green is used for better visual representation of the value (green is the best, red is the worst).

	B	1	2	3	W
A1	0.0506	0.0799	0.0815	0.0848	0.1343
A2	0.054	0.0752	0.0799	0.0900	0.1155
B1	0.019	0.0312	0.0317	0.0278	0.0681
B2	0.0191	0.0263	0.0280	0.0348	0.0679
C1	0.0635	0.0760	0.0828	0.0874	0.1176
C2	0.0621	0.0625	0.0810	0.0808	0.1124
D1	0.078	0.0877	0.0940	0.0783	0.1015
D2	0.0777	0.0869	0.1038	0.0857	0.0958
E1	0.0239	0.0310	0.0348	0.0457	0.0599
E2	0.025	0.0355	0.0481	0.0446	0.0621

Table 5 shows the ratios between the real and simulated values; ideally, these values should all be equal to 1. In general, it can be stated that the f_f values acquired by the real experiments were lower than the values from the simulation, and the average ratio was $r = 0.745$ with the standard deviation equal to $\sigma_r = 0.126$. Figure 9 displays a comparison of the simulated and real values in the form of a bar graph, together with the ratio r . In this image, it can be observed that, for the same trajectory, the ratio r typically lowers (deteriorates) with increasing the f_f values. This will be further discussed in the Discussion section.

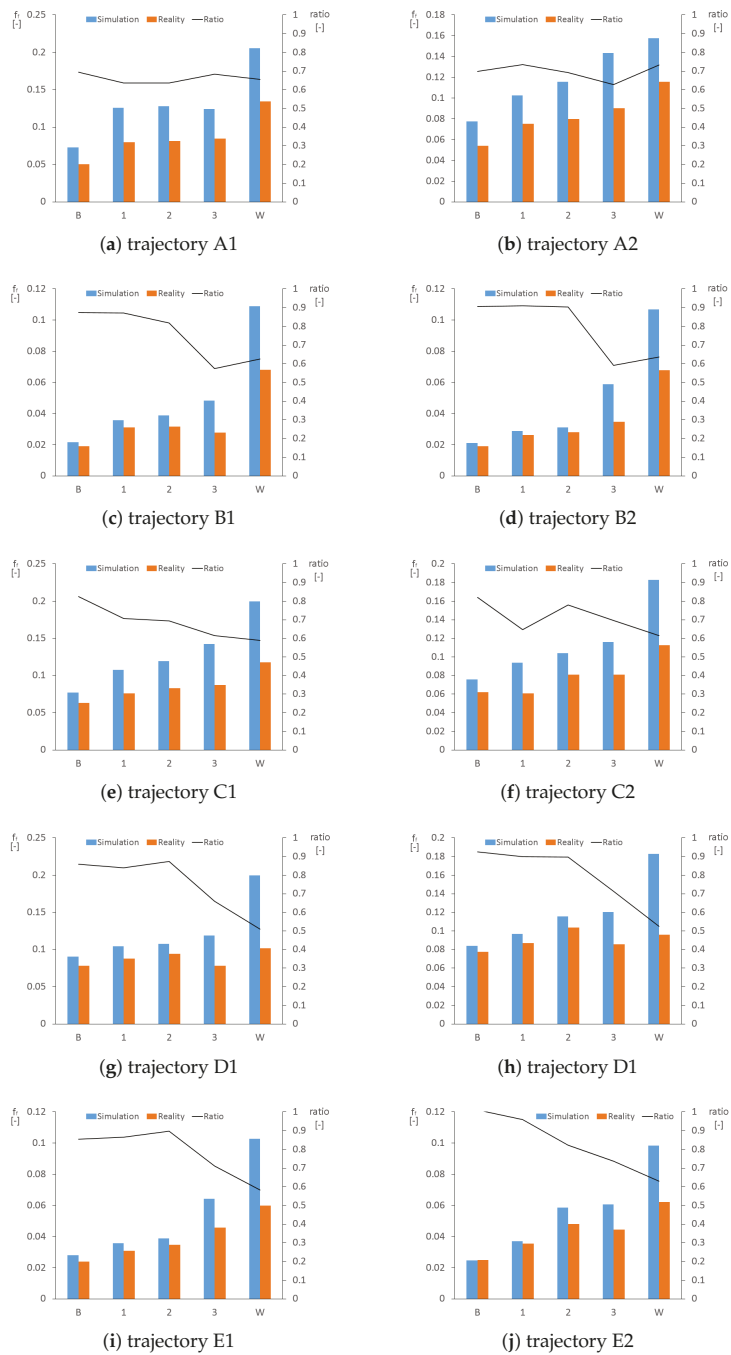


Figure 9. Comparison of the fitness function values acquired from simulation and experiments on the real robot; the black lines represent the ratio between the values.

Table 5. Ratios between the real and simulated values of fitness function in the five robot locations for individual trajectories. Color gradient red–white is used for better visual representation of the value (white is the ideal ratio of 1.0, red is the ratio 0.5).

	B	1	2	3	W
A1	0.6932	0.6359	0.6362	0.6833	0.6541
A2	0.6985	0.7334	0.6912	0.6278	0.732
B1	0.8743	0.8722	0.8168	0.5735	0.6244
B2	0.9056	0.9096	0.9025	0.5915	0.6359
C1	0.8246	0.7075	0.6933	0.6151	0.5887
C2	0.8199	0.6657	0.7792	0.6965	0.6149
D1	0.8594	0.8393	0.8728	0.6591	0.5087
D2	0.9243	0.8984	0.8968	0.7126	0.5238
E1	0.8541	0.8658	0.8960	0.7118	0.5825
E2	1.0100	0.9590	0.8224	0.7360	0.6303

5. Discussion

The paper describes the grid approach to finding the optimal robot location, where all possible robot locations are evaluated using a simulation in CoppeliaSim, and then the location with the best (lowest) fitness function f_f value is selected. In our case, there was an additional preprocessing step that found all valid robot locations using a simple and fast custom-made simulation system to increase the speed of the process. This step is not required, and therefore the research can be reproduced without this custom simulation system.

The grid approach can alternatively be replaced by using some optimization algorithms, for example, the Particle Swarm Optimization (PSO) [31], which could further reduce the time needed to find the optimal solution. However, PSO would not provide the same type of comprehensive analysis of the whole space around the trajectory, and could also possibly return a local minimum instead of the global one.

The method was demonstrated and tested on ten sample trajectories and the collaborative robot UR3. As can be seen from the results in Table 2, the percentage improvement in the fitness function between the worst and the best possible robot location ranged from 54% to 80.3%. This is mostly a theoretical improvement, as the worst-rated locations would likely be not chosen by the system integrator or workplace project architect for other reasons (they are typically on the edge of the working area of the robot or close to a singular configuration). If we instead compare the best robot location to the average f_f value of all possible locations, the improvement still ranges from 22% to 53.1%; therefore, it is clear that some interesting reduction in the robot joints wear can be achieved by selecting the optimal robot location instead of a “random” one.

This method can be used in practice, provided there is a dynamic model of the selected robot available for some suitable simulation system (for example, CoppeliaSim). It is necessary to properly define the whole simulation model of the workplace, including any potential obstacles, otherwise, the returned optimal robot location could be invalid due to a collision. Nonetheless, it is still important to verify that the optimal location is feasible from the point of view of energy connections and other similar restrictions that cannot be included in the simulation.

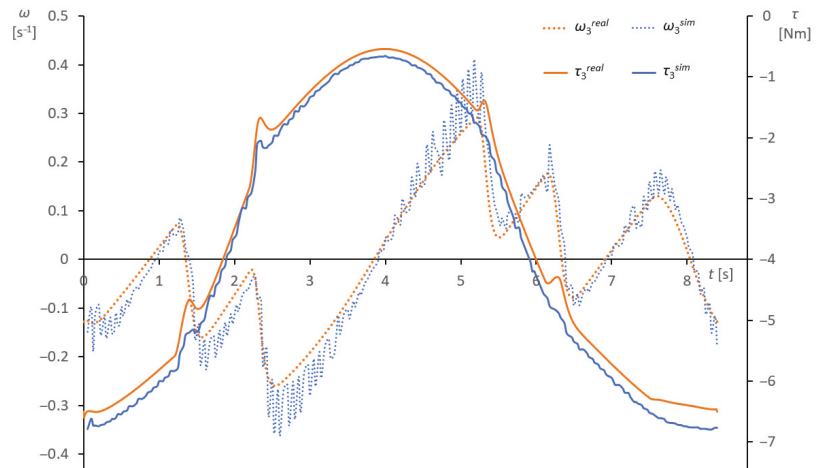
The dynamic model should include all properties and phenomena that noticeably affect the torque values in joints of the robot, as the torques are one of the main inputs for the proposed method. It is, thus, necessary to properly define and simulate also the payload in the end-effector, including any technological forces caused by the effector during, for example, cutting, milling, and spraying. The simulation system CoppeliaSim chosen in our demonstration is capable of simulating all these effects.

Experiments on a real robot UR3 verified that the simulation model in CoppeliaSim matched the reality acceptably well. The absolute values of f_f acquired by the simulation and from the real robot differed approximately by the scale factor of 0.745 ± 0.126 .

The difference between simulation and reality was caused especially by the simulation model of the UR3 robot, which is likely not absolutely perfect as far as mass properties are concerned, and also by the dynamic simulation engine, which performed some simplifications. More important is that, in the relative evaluation of individual robot locations, the real experiments led to very similar results as the simulation—as can be seen in Tables 3–5. There were some distinct deviations only in one path of robot end-point movement (trajectories D1, D2). It can be, thus, stated that, to improve the robot joints wear, potentially expensive and time-consuming real-world experiments are not necessary; a simulation suffices.

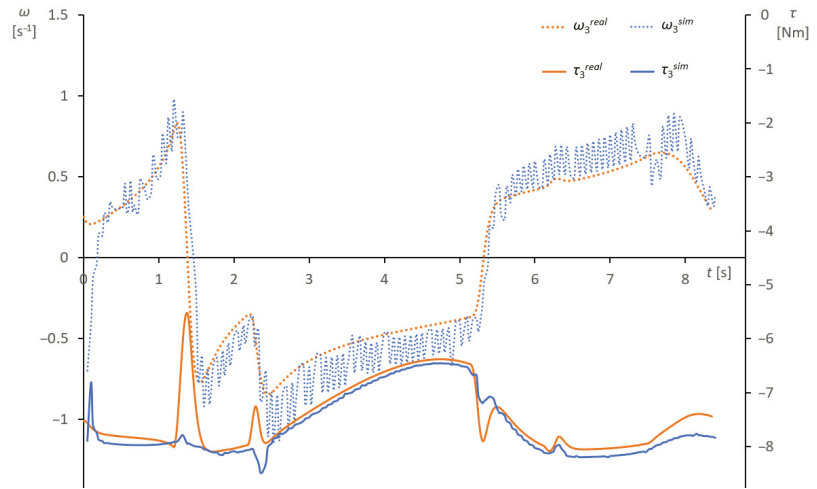
To demonstrate and analyze the source of the difference between the values from the simulation and from the real robot, Figure 10 shows an example of the simulated torque τ_3^{sim} and velocity ω_3^{sim} values directly compared with the values from the real robot (τ_3^{real} , ω_3^{real}), for the third joint of the robot during the B1 trajectory. The absolute value of τ_3^{real} was generally lower than the absolute value of τ_3^{sim} , which was likely caused by inaccurate mass parameters of the robot links in the simulation model. This is the reason why the f_f values from the real robot were mostly lower than the values from the simulation (see Table 5). The noise in the ω_3^{sim} values was caused by the discrete character of the simulation.

Figure 10b shows that, for the worst robot location, the τ_3^{real} values differed significantly from τ_3^{sim} in several peaks that correspond to moments with noticeable acceleration in the movement. This is probably caused by some simplifications in the dynamics engine in the simulation system or by some advanced algorithms in the control system of the real robot. Generally, in worse robot locations (with higher f_f values), the robot joints must perform movements with higher acceleration, which causes this additional source of difference between the simulation and reality.



(a) trajectory B1, the best robot location

Figure 10. Cont.



(b) trajectory B1, the worst robot location

Figure 10. Comparison of the real and simulated values of angular velocity ω_3 and torque τ_3 of the third joint during the whole trajectory B1. (a) Robot placed in the best location. (b) Robot placed in the worst location.

The proposed optimization process of a robotized workplace can be applied in the design phase when it is easy to make modifications to the workplace layout. Another option is to use the optimization for an existing workplace when other types of optimization are not possible (the trajectory is fixed, etc.).

6. Conclusions

The goal of this paper was to present a methodology for the optimization of a robot manipulator base position relative to the given trajectory of movement of the manipulator end-point. The optimization algorithm was based on the principle of minimization of the chosen fitness function, which comprises the arithmetic mean and standard deviation of the relative wear factors of all robot joints (8). The goal was to balance and minimize the wear of the drive chain in the robot joints, where the wear is approximated as the amount of mechanical work done by the joint, while also taking into account the abilities of the particular joint (the maximal permissible torque and velocity).

Future work will include verification on different types of robots and with various payloads. To verify the real impact on the wear of robot joints and, thus, the reduction of the robot lifetime, a long-term experiment would have to be performed simultaneously on at least two identical robots. One robot would be placed in a location with a good (low) f_f value and the other—reference—robots would perform the same task from locations with higher f_f values. Afterward, the robots would have to be partially disassembled to analyze and compare the mechanical degradation of the joints.

The cases when the workplace is used alternately for several different tasks and, thus, the robot performs more than one movement trajectory could also be addressed. The trajectories could either be combined into one for the simulation, or the method could be applied to each trajectory separately and then the results would be combined, for example by interpolation using weight coefficients given by the frequency of use of the trajectories.

Although this method does not require extremely precise dynamic simulation, this aspect could also be improved by creating a more accurate dynamic model of the robot, for example using some of the dynamic parameter identification method [32]. However, the simulation of friction and other complex phenomena will always be simplified in commonly available simulation systems.

Author Contributions: Conceptualization, T.K., Z.B. and V.K.; methodology, T.K., Z.B. and A.V.; software, T.K., A.V. and J.Š.; validation, T.K., A.V. and Z.B.; investigation, T.K., Z.B., V.K. and R.R.; writing—original draft preparation, T.K., Z.B. and V.K.; writing—review and editing, Z.B., A.V., V.K., J.Š. and R.R.; visualization, T.K.; supervision, Z.B.; project administration, Z.B. and V.K. All authors have read and agreed to the published version of the manuscript.

Funding: This article has been elaborated under support of the project Research Centre of Advanced Mechatronic Systems, reg. No. CZ.02.1.01/0.0/0.0/16_019/0000867 in the frame of the Operational Program Research, Development and Education.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are available on request from the corresponding author. The data are not publicly available due to funding project restrictions.

Conflicts of Interest: The authors declare no conflict of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript, or in the decision to publish the results.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application programming interface
CSO	Chicken swarm optimization
GCS	Global coordinate system
IK	Inverse kinematics
PSO	Particle swarm optimization
RTDE	Real-time data exchange
TCP/IP	Transmission control protocol/Internet protocol
UR3	Universal Robots 3

References

- Diaz, J.R.; Mukherjee, P.; Verl, A. Automatic Close-optimal Workpiece Positioning for Robotic Manufacturing. *Procedia CIRP* **2018**, *72*, 277–284. [[CrossRef](#)]
- Tangpattanakul, P.; Artrit, P. Minimum-time trajectory of robot manipulator using Harmony Search algorithm. In Proceedings of the 2009 6th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology, Chonburi, Thailand, 6–9 May 2009; Volume 1, pp. 354–357. [[CrossRef](#)]
- Valente, A.; Baraldo, S.; Carpanzano, E. Smooth trajectory generation for industrial robots performing high precision assembly processes. *CIRP Ann.* **2017**, *66*, 17–20. [[CrossRef](#)]
- Llopis-Albert, C.; Rubio, F.; Valero, F. Modelling an Industrial Robot and Its Impact on Productivity. *Mathematics* **2021**, *9*, 769. [[CrossRef](#)]
- Piazzzi, A.; Visioli, A. Global minimum-jerk trajectory planning of robot manipulators. *IEEE Trans. Ind. Electron.* **2000**, *47*, 140–149. [[CrossRef](#)]
- Gasparetto, A.; Zanotto, V. A technique for time-jerk optimal planning of robot trajectories. *Robot. Comput. Integr. Manuf.* **2008**, *24*, 415–426. [[CrossRef](#)]
- Hu, S.; Kang, H.; Tang, H.; Cui, Z.; Liu, Z.; Ouyang, P. Trajectory Optimization Algorithm for a 4-DOF Redundant Parallel Robot Based on 12-Phase Sine Jerk Motion Profile. *Actuators* **2021**, *10*, 80. [[CrossRef](#)]
- Mu, Y.; Zhang, L.; Chen, X.; Gao, X. Optimal Trajectory Planning for Robotic Manipulators Using Chicken Swarm Optimization. In Proceedings of the 2016 8th International Conference on Intelligent Human-Machine Systems and Cybernetics (IHMSC), Hangzhou, China, 27–28 August 2016; Volume 2, pp. 369–373. [[CrossRef](#)]
- Lim, Z.Y.; Ponnambalam, S.G.; Izui, K. Nature inspired algorithms to optimize robot workcell layouts. *Appl. Soft Comput.* **2016**, *49*, 570–589. [[CrossRef](#)]
- Bukata, L.; Šúcha, P.; Hanzálek, Z.; Burget, P. Energy Optimization of Robotic Cells. *IEEE Trans. Ind. Inform.* **2017**, *13*, 92–102. [[CrossRef](#)]
- Gadaleta, M.; Berselli, G.; Pellicciari, M. Energy-optimal layout design of robotic work cells: Potential assessment on an industrial case study. *Robot. Comput. Integr. Manuf.* **2017**, *47*, 102–111. [[CrossRef](#)]
- Pellicciari, M.; Berselli, G.; Leali, F.; Vergnano, A. A method for reducing the energy consumption of pick-and-place industrial robots. *Mechatronics* **2013**, *23*, 326–334. [[CrossRef](#)]

13. Meike, D.; Pellicciari, M.; Berselli, G. Energy Efficient Use of Multirobot Production Lines in the Automotive Industry: Detailed System Modeling and Optimization. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 798–809. [[CrossRef](#)]
14. Paes, K.; Dewulf, W.; Elst, K.V.; Kellens, K.; Slaets, P. Energy Efficient Trajectories for an Industrial ABB Robot. *Procedia CIRP* **2014**, *15*, 105–110. [[CrossRef](#)]
15. Spensieri, D.; Carlson, J.S.; Bohlin, R.; Kressin, J.; Shi, J. Optimal Robot Placement for Tasks Execution. *Procedia CIRP* **2016**, *44*, 395–400. [[CrossRef](#)]
16. Biesinger, F.; Meike, D.; Krass, B.; Weyrich, M. A Case Study for a Digital Twin of Body-in-White Production Systems General Concept for Automated Updating of Planning Projects in the Digital Factory. In Proceedings of the 2018 IEEE 23rd International Conference on Emerging Technologies and Factory Automation (ETFA), Turin, Italy, 4–7 September 2018; Volume 1, pp. 19–26. [[CrossRef](#)]
17. Miková, L.; Kelemen, M.; Virgala, I.; Michna, M. Simulation Model of Manipulator for Model Based Design. *Appl. Mech. Mater.* **2014**, *611*, 175–182. [[CrossRef](#)]
18. Georgoulas, G.; Loutas, T.; Stylios, C.D.; Kostopoulos, V. Bearing fault detection based on hybrid ensemble detector and empirical mode decomposition. *Mech. Syst. Signal Process.* **2013**, *41*, 510–525. [[CrossRef](#)]
19. Qian, H.M.; Li, Y.F.; Huang, H.Z. Time-variant reliability analysis for industrial robot RV reducer under multiple failure modes using Kriging model. *Reliab. Eng. Syst. Saf.* **2020**, *199*, 106936. [[CrossRef](#)]
20. Guangjian, W.; Lin, C.; Li, Y.; Shuaidong, Z. Research on the dynamic transmission error of a spur gear pair with eccentricities by finite element method. *Mech. Mach. Theory* **2017**, *109*, 1–13. [[CrossRef](#)]
21. Cubillo, A.; Perinpanayagam, S.; Esperon-Miguez, M. A review of physics-based models in prognostics: Application to gears and bearings of rotating machinery. *Adv. Mech. Eng.* **2016**, *8*, 1687814016664660. [[CrossRef](#)]
22. Zhang, X.; Jiang, G.; Zhang, H.; Yun, X.; Mei, X. Time-dependent reliability analysis of harmonic drive based on transient FEA and accelerated life test. *Eng. Comput.* **2020**. [[CrossRef](#)]
23. Bian, L.; Gebraeel, N.; Kharoufeh, J.P. Degradation modeling for real-time estimation of residual lifetimes in dynamic environments. *IIE Trans.* **2015**, *47*, 471–486. [[CrossRef](#)]
24. Koike, H.; Itakura, K.; Okazaki, S.; Takamiya, M.; Kanemasu, K.; Kida, K. Measurement of Backlash and Fatigue Wear of PEEK Bush in Robot Joint under Middle Load. In *Applied Mechanics and Materials*; Trans Tech Publications Ltd.: Stafa-Zurich, Switzerland, 2013; Volume 418, pp. 38–43. [[CrossRef](#)]
25. Bittencourt, A.C.; Axelsson, P.; Jung, Y.; Brogårdh, T. Modeling and Identification of Wear in a Robot Joint under Temperature Uncertainties. *IFAC Proc. Vol.* **2011**, *44*, 10293–10299. [[CrossRef](#)]
26. Lugt, P.M. A Review on Grease Lubrication in Rolling Bearings. *Tribol. Trans.* **2009**, *52*, 470–480. [[CrossRef](#)]
27. Aivaliotis, P.; Arkouli, Z.; Georgoulas, K.; Makris, S. Degradation curves integration in physics-based models: Towards the predictive maintenance of industrial robots. *Robot. Comput. Integr. Manuf.* **2021**, *71*, 102177. [[CrossRef](#)]
28. Peternel, L.; Tsagarakis, N.; Ajoudani, A. A Method for Robot Motor Fatigue Management in Physical Interaction and Human-Robot Collaboration Tasks. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 2850–2856. [[CrossRef](#)]
29. Rodrigues, L.R. Remaining Useful Life Prediction for Multiple-Component Systems Based on a System-Level Performance Indicator. *IEEE/ASME Trans. Mechatron.* **2018**, *23*, 141–150. [[CrossRef](#)]
30. Jacobs, S.; Rios-Gutierrez, F. Self organizing maps for monitoring parameter deterioration of DC and AC motors. In Proceedings of the 2013 Proceedings of IEEE Southeastcon, Jacksonville, FL, USA, 4–7 April 2013; pp. 1–6. [[CrossRef](#)]
31. Poli, R.; Kennedy, J.; Blackwell, T. Particle swarm optimization. *Swarm Intell.* **2007**, *1*, 33–57. [[CrossRef](#)]
32. Kovincic, N.; Mueller, A.; Gattringer, H.; Weyrer, M.; Schlotzhauer, A.; Brandstötter, M. Dynamic parameter identification of the Universal Robots UR5. In Proceedings of the Austrian Robotics Workshop 2019, Steyr, Austria, 9–10 May 2019. [[CrossRef](#)]

Article

Design and Verification of Multi-Agent Systems with the Use of Bigraphs

Piotr Cybulski * and Zbigniew Zieliński

Faculty of Cybernetics, Military University of Technology, ul. gen. S. Kaliskiego 2, 00-908 Warsaw, Poland; zbigniew.zielinski@wat.edu.pl

* Correspondence: piotr.cybulski@wat.edu.pl

Featured Application: Rapid development of behavior policies for agents in a controlled environment.

Abstract: Widespread access to low-cost, high computing power allows for increased computerization of everyday life. However, high-performance computers alone cannot meet the demands of systems such as the Internet of Things or multi-agent robotic systems. For this reason, modern design methods are needed to develop new and extend existing projects. Because of high interest in this subject, many methodologies for designing the aforementioned systems have been developed. None of them, however, can be considered the default one to which others are compared to. Any useful methodology must provide some tools, versatility, and capability to verify its results. This paper presents an algorithm for verifying the correctness of multi-agent systems modeled as tracking bigraphical reactive systems and checking whether a behavior policy for the agents meets non-functional requirements. Memory complexity of methods used to construct behavior policies is also discussed, and a few ways to reduce it are proposed. Detailed examples of algorithm usage have been presented involving non-functional requirements regarding time and safety of behavior policy execution.

Keywords: multi-agent systems; bigraphs; design; verification; modeling; non-functional requirements

Citation: Cybulski, P.; Zieliński, Z. Design and Verification of Multi-Agent Systems with the Use of Bigraphs. *Appl. Sci.* **2021**, *11*, 8291. <https://doi.org/10.3390/app11188291>

Academic Editors: Paola Pellegrini, António Paulo Moreira, Pedro Neto and Félix Vidal

Received: 2 August 2021

Accepted: 2 September 2021

Published: 7 September 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the increase of computational power and its availability comes the desire to incorporate it more into our daily life. Current ideas on how to do this include the Internet of Things, multi-agent systems (in which particular cases are swarms of robots), or smart objects and places (e.g., cities, homes, cars). All of them require new ways to design large-scale (i.e., consisting of a significant number of elements) software and physical systems that consider both how individual components interact and how a system as a whole works. There are various unresolved problems related to this. There is no consensus on what elements of the real world should be modeled and which of their capabilities should be taken into account in general. What is worse, among different design methods elements of the real world are used differently. Finally, the results of these methods are often incomparable, or at least, there is no common way to evaluate multi-agent system design methods. Regardless, any method for designing complex systems must offer a specific range of capabilities to be considered useful.

The concept of agent is applied to entities that have autonomy and are placed in a changing environment. Multi-agent systems [1,2] are structures within which agents can be identified. One of the advantages of designs using agents is that they can be represented at different levels of detail, from abstract entities (like mathematical structures) to actual robots. For this reason, among others, the concept of multi-agent system is used in various contexts. This term may be used to characterize a group of machine learning methods [3,4]. It can also be used to highlight attributes of certain models and simulation approaches [5–7].

The term also refers to a subgroup of robotics solutions [8–11] that make use of widely understood autonomous robots to perform assigned tasks. In this work, we will focus on multi-agent robotic systems (MARS). The literature [12–16] is replete with examples of various applications of multi-agent robotic systems. There are also methodologies and tools [10,17] to design such systems. There is no consensus on how to design such systems in general and current solutions come from different areas of science. The most common paradigms used to design MARS include software design patterns [16], control theory [12,13], optimization theory or combinations of the above [15]. Some examples are utilizing mathematical logic in MARS design [18], but they are much less common. Due to the lack of agreement on how to design MARS and the fact that results produced by different methodologies are difficult to compare, we will try to evaluate them based on their capabilities. In this paper, we will be interested not so much in how to design MARS but rather how the following questions can be answered about an existing project:

- Is the project correctly designed? We want to assure the syntactic correctness, i.e., the correct use of formal tools such as mathematical logic, differential equations, or pi-calculus. We also care about semantic correctness, i.e., the ability to transform a formal model into a real solution (implementable on robots).
- How does one perform a simulation illustrating MAS operation?
- Have non-functional requirements been met? Those regarding safety and speed of task execution in particular.

Verifying the correctness of a model is the simplest and most solutions can be verified using the tools they were made with. Verifying whether a designed system accomplishes a given task is much more difficult. The vast majority of methodologies in the literature use simulation for this purpose. Exceptions can be found among models that highly formalize the internals of agents, how they operate, and the course of a task itself. Verification by simulation also gets complicated as the model becomes more abstract. The simplest designs in this regard are those based on methods commonly used in other areas of science (such as differential equations or graph theory) or made using tools integrated with a simulator. Verification of non-functional requirements is a difficult part of the design. Methodologies commonly found in the literature such as RE4Gaia [19], TROPOS [20], DIAMOND [21], or Adelfe [22] take into account non-functional requirements during design process. They usually aim to enable design of multi-agent systems in general (not just multi-agent robotic systems). Successive stages in most of these methodologies are not closely coupled together. By loosely coupled process, we understand a design process where a designer’s interpretation of how the system works plays a significant role the whole time. In other words, one cannot treat the results of one stage as an input that the next stage will automatically transform into a form acceptable by yet another stage. When it comes to verification of system requirements, it should be noted that none of the above methodologies offer formal guarantees regarding the system’s functionality as the methods dedicated to specific tasks do. An example of a such method can be found in [13] where a formal guarantee is given for robots to move keeping at least a specified distance from each other (an example of a non-functional requirement). In [12] a guarantee of fulfillment of functional requirements is presented where a task is guaranteed to be carried out if certain conditions are satisfied.

Using bigraphs [23] to design multi-agent systems is a relatively new approach to modeling this kind of system. The bigraph theory was published by Robin Milner in 2008 but has already been extended with a notion of overlapping locations [24] and probability [25]. Bigraphs are currently found useful in areas such as system of systems design [26], IoT [27], and wireless network modeling [28]. Currently, there are a few tools that support modeling systems with bigraphs, the most notable of them are Bigraphical Model Checker [29] (discontinued), Bigraph Framework for Java [30], and BigraphER [31]. The first two of them focus on checking the reachability of certain states of a system [29,31]. At the same time, the last one provides means to analyze various aspects of a modeled system (especially useful in this regard is underlying OCaml library *bigraph*). We believe that BigraphER [31] provides the most advanced

set of utilities to model systems with bigraphs available at the moment. Multi-agent systems design methodologies [32,33] involving bigraphs are scarce, and most of them do not consider generating behavior policies based on a constructed model. As an exception to this, one may point out BigActor methodology described in [34] that uses bigraphs mixed with the notion of actors [35] or our methodology [36] based on bigraphs with tracking.

In [36] we have proposed a methodology based on bigraphs with tracking [23] that enables design of multi-agent systems. We have chosen tracking bigraphs primarily because they allow for analysis of objects' activities over time without introducing another layer of abstraction (as it was done, for example, in [34]). Our methodology is devoid of some of the drawbacks we mentioned earlier, such as loose coupling between design stages or the designer's interpretation of systems internals on all stages of the design process. Moreover, successive stages of the methodology are module-like which means their implementations can be adjusted to project needs. The methodology's main disadvantages are high computational complexity, limitation of system's agents to entities that can be fully controlled, and the fact that the operation of a designed system is determined before it is started. It also does not offer universal guarantees of task successful completion as presented in [12,13,18]. Putting our work in a broader context, we can place our methodology in a group of bottom-up [37] methods of MAS design with a note that it focuses on global goals rather than individual ones. In fact, agents in our approach do not have preferences that can affect their actions. A distinguishing feature of our proposition is the lack of abstractions outside the bigraphs framework, typically agents' internal mechanics are modeled with BDI (Belief, Desire, and Intention) [32,38] or actors [34].

This work is an extension of the methodology proposed in [36]. This paper aims to demonstrate how to verify the correctness of a design, check the fulfillment of non-functional requirements, and visualize behavior policies. We have developed an algorithm to automatically verify the correctness of a model and construct successive simulation states. We also described how to verify whether non-functional requirements are satisfied by a behavior policy for agents in the system. An example implementation [39] of the algorithm has been prepared. We also addressed the memory complexity of operations performed during behavior policy generation. We discussed how it influences the feasibility of projects and suggested a few ways to reduce the memory complexity. Finally, a tool [40] has been implemented that incorporates all of the mentioned memory complexity reduction strategies and a tool [41] to illustrate constructed behavior policies.

2. Methods and Materials

In this section, we will introduce all terms and definitions that are necessary to understand examples presented in Section 3. Section 2.1 is devoted to basic informal definitions that will be used throughout the rest of this article. Sections 2.2–2.4 aim to quickly acquaint the reader with the methodology described in detail in [36] and for that reason micro-examples are included at the end of each of these subsections. Section 2.5 is dedicated to an algorithm for verification and visualization of behavior policies. Since the algorithm is the key of this article, examples of its usage are presented in Section 3.

2.1. Basic Concepts

Before formal definitions, we will introduce the following concepts:

- **Task**—A collection of objects from the real world along with the actions they can perform, the initial state, and the target-desired (final) state(s). An example of a task might be:
 “In an area that is a 3×3 grid, there are two robots in opposite (diagonally) cells. Each robot can move to vertically and horizontally adjacent cells and connect to a second robot if both are in the same cell. The goal of the task is for both robots to connect with each other.”
- **Mission**—a realization of a task.

- Task element—a real-world entity that is relevant to the subject matter being modeled. Elements can be people, robots, areas, data sources, and receivers, etc.
- Passive object—a task element that can participate in activities without initializing them. It may contain other passive objects. We are not interested in their behavior, but we take into account the passage of time for them. The number of passive objects is constant during a mission.
- Active object (agent)—a task element that can participate in activities by initializing them. It can contain other active and passive objects. We are interested in their behavior, and we take into account the passage of time for them. We can control them. It is assumed that the number of agents during a mission is constant.
- Environment—a task element that can participate in activities without initializing them. It can contain passive and active objects and be owned by at most one other object. We are not interested in its behavior, and do not consider the passage of time for it.
- Behavior Policy—A set of planned actions for all agents that meets the following requirements:
 - Implementing a behavioral policy solves a given task;
 - All agents start the mission at the same time;
 - Agents can complete a mission at different points in time;
 - All agent activities must be performed continuously (without time gaps);
 - All agents that participate in a cooperative activity must start performing it at the same moment.
- Scenario—Mission using a specific behavioral policy.

2.2. Bigraphs

Through this article we will extensively use bigraphs, *concrete bigraphs* to be precise. Concrete bigraphs allow identifying its nodes and edges with *support* (more about that later). In contrast, *abstract bigraphs* lack the mentioned identifiers. In the rest of this article, whenever we refer to a bigraph, we will have a concrete bigraph in mind. A bigraph consists of two graphs: a place graph and a link graph. Place graph is intended to model spatial relations between system elements. A link graph is a hypergraph that can be used to model interlinking between the elements.

Formally a bigraph is defined as:

$$B = (V_B, E_B, ctrl_B, G_B^P, G_B^L) : I \rightarrow O$$

- V_B —a set of vertices identifiers;
- E_B —a set of hyperedges identifiers. A union of both of these sets makes the *bigraph support*;
- $ctrl_B : V_B \rightarrow K$ —a function assigning a control type to vertices. K denotes a set of control types and is called a signature of the bigraph;
- $G_B^P = \langle V_B, ctrl_B, prnt_B \rangle : m \rightarrow n$ and $G_B^L = \langle V_B, E_B, ctrl_B, link_B \rangle : X \rightarrow Y$ denote a place and a link graph respectively. A $prnt_B$ function defines hierarchical relations between vertices, roots, and sites. A $link_B$ function defines linking between vertices and hyperedges in the link graph;
- $I = \langle m, X \rangle$ and $O = \langle n, Y \rangle$ denotes the inner face and outer face of the bigraph B . By m, n we will denote sets of preceding ordinals of the form: $m = \{0, \dots, m - 1\}$. Sets X and Y represent inner and outer names respectively. When any of the elements of an interface is omitted it means it is either equal to 0 (when interface lacks an ordinal) or it is empty (when there is no set of names). For example, interface $I = m$ means it has no inner names.

An example of graphical representation of a bigraph is presented in Figure 1.

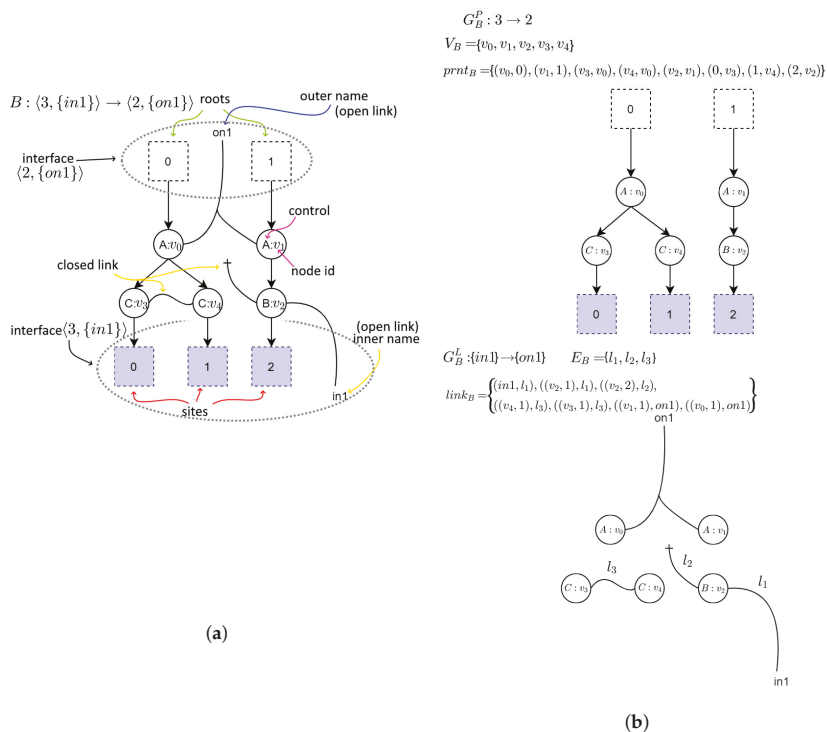


Figure 1. An example of a bigraph and its constituents. The right part represents a place graph (the upper part of the figure) and a link graph (the lower part of the figure). They share a signature which defines control types (letters in nodes) and arity of each control (number of unique links that can be connected to a node with specified control). Ports and inner names can be attached to either edges or outer names, that is why there are only three edge identifiers in the link graph. On the left there is the bigraph made from the superposition of them both. (a) A bigraph. (b) A place graph and a link graph.

Reaction rules are used to model dynamics in bigraphical systems. In this paper, we will use (simplified) tracking reaction rules. Reaction rule consists of a pattern (redex) to be found in an input bigraph that shall be replaced with another bigraph (reactum).

Formally, a tracking reaction rule is a quadruple:

$$(B_{redex} : m \rightarrow O, B_{reactum} : m' \rightarrow O, \eta, \tau)$$

where:

- B_{redex} —a bigraph called redex;
- $B_{reactum}$ —a bigraph called reactum;
- $\eta : m' \rightarrow m$ —a map between sites from reactum to sites in redex;
- $\tau : V_{reactum} \rightarrow V_{redex}$ —a map of reactum’s node identifiers onto redex’s node identifiers. It allows one to indicate which elements of an input bigraph are “residues” in an output bigraph.

Bigraphical Reactive System (BRS) is a tuple $(\mathcal{B}, \mathcal{R})$ where \mathcal{B} denotes a set of bigraphs with empty inner face and \mathcal{R} is a set of reaction rules defined over \mathcal{B} . If \mathcal{R} consists of rules with tracking then a pair $(\mathcal{B}, \mathcal{R})$ makes a *Tracking Bigraphical Reactive System* (TBRS).

Having a TBRS we can generate a *Tracking Transition System* (TTS). A *Tracking Transition System* is a 7-tuple: $\mathcal{L}_{\mathcal{T}} = (Agt, Red, Lab, Apl, Par, Res, Tra)$ where:

- Agt —a set of bigraphs;

- *Red*—a set of redexes used to construct the TTS;
- *Lab*—a set of labels;
- $\text{Apl} \subseteq \text{Agt} \times \text{Lab}$ —an applicability relation;
- $\text{Par} : V_r^{V_b} \quad r \in \text{Red}, b \in \text{Agt}$ —a participation function. It indicates which vertices in an input bigraph correspond to elements in the redex of a transition;
- $\text{Res} : V_{b_1}^{V_{b_2}} \quad b_1, b_2 \in \text{Agt}$ —a residue function. It maps vertices in an output bigraph that are residue of an input bigraph to the vertices in the input bigraph;
- $\text{Tra} \subseteq \text{Apl} \times \text{Agt} \times \text{Par} \times \text{Res}$ —a transition relation.

As we said at the beginning of this section, we will use a simple example to illustrate how the formal definitions can be used in practice. The system for our example consists of two areas and two agents (we do not care whether they are humans, robots, or other autonomous entities). Areas will be denoted by controls *A* and *B* while agents will be represented with controls *U*. We assume that agents can move from an area of type *A* to an area of type *B* in two ways, which differ in execution speed. Thus Tracking Bigraphical Reactive System of the system above consists of three bigraphs and two reaction rules. The elements of *B* set are described in Table 1 and the reaction rules are defined in Table 2. The Tracking Transition System of this TBRS is defined in Table 3.

Table 1. Elements of the *B* set for the introductory example.

Graphical Representation	Name	Description
	s0	The initial state of the system.
	s1	The state where only one of the agents has moved to the <i>B</i> area.
	s2	The state where both agents has moved to the <i>B</i> area.

Table 2. Elements of the *R* set for the introductory example. The η function for the first rule and both τ functions are identities. The first rule represents an action that allows a single agent to move between areas. The second rule is for an action where two agents move both at once. The second rule is only reasonable if underlying mechanism differs to that of the first rule.

Graphical Representation	Name
	r_1
	r_2

Table 3. The Tracking Transition System for the introductory example. Each row defines a single transition in the system.

Apl	Agt	Par	Res
$\langle s0, r_1 \rangle$	s1	$\{(0, 0), (1, 1), (3, 2)\}$	$\{(0, 0), (1, 2), (2, 3), (3, 1)\}$
$\langle s0, r_1 \rangle$	s1	$\{(0, 0), (2, 1), (3, 2)\}$	$\{(0, 0), (1, 1), (2, 3), (3, 2)\}$
$\langle s0, r_2 \rangle$	s2	$\{(0, 0), (1, 1), (2, 2), (3, 3)\}$	$\{(0, 0), (1, 3), (2, 1), (3, 2)\}$
$\langle s1, r_1 \rangle$	s2	$\{(0, 0), (1, 1), (2, 2)\}$	$\{(0, 0), (1, 2), (2, 3), (3, 1)\}$

2.3. State Space

Having a Tracking Transition System we can transform it into a state space of the modeled system. A state space can be later used to generate a behavior policy for agents (as defined in Section 2.1) in the system.

We assume the following about modeled systems:

1. A number of passive and active objects is constant during whole mission;
2. A system cannot change its state without an explicit action of an agent (alone or in cooperation with other agents);
3. No actions performed by agents are subject to uncertainty;
4. A mission can end for each agent separately in different moments. In other words, agents do not have to finish their part of the mission all at the same time;
5. In case of actions involving multiple objects (whether these are active or passive), it is required of all participants to start cooperation at the same moment.

A state space SS of a system consisting of n_o objects and n_s states is defined as:

$$SS = (S, E, L, I, C, T, M_f)$$

where:

- $S \subset \mathbb{N}$ —a set of states in the state space. It corresponds to bigraphs in the Tracking Transition System;
- $E \subseteq S \times S$ —a multiset of ordered pairs of states. Elements in this set are directed edges representing transition relations between states;
- L —a set of labels of changes in the system. It will usually consist of reaction rule names from the Tracking Transition System the state space originated from. To determine what changes, in what order, have led to a specific state we will additionally introduce set $H = \{l_t | l \in L, t \in \mathbb{N}\}$. Elements of the H set indicate what action (label) took place in what order (index value).
- $I = \{\mathbb{N}_1^2 \times \dots \times \mathbb{N}_{n_o}^2\}$ —a set of possible state-at-time (SAT) configurations. The interpretation of elements in such a set is as follows. The first element in each of inner tuples denotes *id* of an object (either passive or active) in the system. The second element in each inner tuple is meant to represent time at which the object specified by the *id* is at. For example, for $n_o = 2$ the element $i_x = \langle (1, 777), (2, 123) \rangle$ denotes a situation where the object with id 1 is at the moment 777 while the object with id 2 is at the moment 123.
- $C = (I \times 2^H) \cup \{0\}$ —a set of possible mission courses. 0 denotes the neutral element, i.e., $\forall_{x \in C} x + 0 = 0 + x = x$. For the rest of the elements of C set the + symbol serves only as an associative conjunction operator and does not denote any meaningful operation. In other words for the rest of the elements the following formula is true: $\forall_{x, y \in C \setminus \{0\}} x + y = y + x$.
- $T = \{f_i : C \times \mathbb{N} \rightarrow C | i \in \mathbb{N}\} \cup \{f_{null}\}$ —a set of functions defining progress of a mission. The f_{null} function returns 0 regardless of input. Additionally, we will denote by $T_{i,j} \subset T$ a set of all mission progress functions from the i state to the j state.
- $M_f : E \rightarrow T$ —a bijective mapping of edges to mission progress functions.

Going back to our introductory example, we will now convert the Tracking Transition System from Table 3 into a state space of the system. We will not define all of the formal elements and rather focus on the key ones. The S consists of three elements $S = \{0, 1, 2\}$ that correspond to bigraphs s_0, s_1 and s_2 respectively. The L consists of two elements that correspond to reaction rules in TBRS i.e., $L = \{r_1, r_2\}$. Knowing that there are only two agents in the system (so there are two objects in total) elements of the set I will be of the form $\langle\langle i_1, x \rangle, \langle i_2, y \rangle\rangle$. The elements i_1, i_2 of a tuple correspond to identifiers of objects (in this case $i_1, i_2 \in \{1, 2\}$) and x and y elements indicate a moment of time at which each object is at. We will clarify how to utilize the C set in the next subsection. As it was mentioned earlier, the action represented by the r_1 reaction rule takes 2 units of time while the r_2 reaction takes only 1 unit of time. How these values are obtained depends on a project and may be subject to many factors such as resolution of time need to be considered (whether these are minutes, seconds or hours) or variability (or lack thereof) of time needed to execute actions represented by reaction rules. Knowing this, the elements of the T set are listed in Table 4. Subsequent elements of this set correspond to transitions in TTS. The permutation being a result of application of a transition function corresponds to permutation of vertices corresponding to objects in res function. It is also worth noting that f_3 function requires both agents to be at the same time (variable z) in order to return something other than 0.

Table 4. Mission progress function definitions for the state space presented in Figure 2. The action represented by r_1 reaction rule is assumed to take 2 units of time while the action r_2 takes only 1 unit of time.

Function	Function Definition
f_1	$f_1(c, t) = \begin{cases} [\langle\langle (b, y), (a, x + 2) \rangle\rangle, H' \cup \{r_{1t+1}\}] & : c = [\langle\langle (a, x), (b, y) \rangle\rangle, H'] \\ 0 & : c = 0 \end{cases}$
f_2	$f_2(c, t) = \begin{cases} [\langle\langle (a, x), (b, y + 2) \rangle\rangle, H' \cup \{r_{1t+1}\}] & : c = [\langle\langle (a, x), (b, y) \rangle\rangle, H'] \\ 0 & : c = 0 \end{cases}$
f_3	$f_3(c, t) = \begin{cases} [\langle\langle (a, z + 1), (b, z + 1) \rangle\rangle, H' \cup \{r_{2t+1}\}] & : c = [\langle\langle (a, z), (b, z) \rangle\rangle, H'] \\ 0 & : c \neq [\langle\langle (a, z), (b, z) \rangle\rangle, H'] \end{cases}$
f_4	$f_4(c, t) = \begin{cases} [\langle\langle (b, y), (a, x + 2) \rangle\rangle, H' \cup \{r_{1t+1}\}] & : c = [\langle\langle (a, x), (b, y) \rangle\rangle, H'] \\ 0 & : c = 0 \end{cases}$

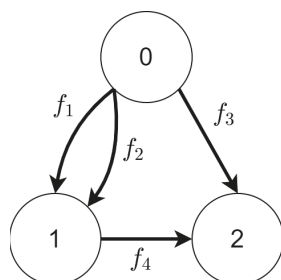


Figure 2. The state space generated from Tracking Transition System defined in Table 3. Mission progress functions definitions are defined in Table 4.

2.4. Behavior Policy

We define a behavior policy as a schedule of actions for each object from the beginning of a mission to its end that meets all the requirements listed in Section 2.1.

Having a state space, we can view a behavior policy as a walk (in graph theory sense) indicating what changes (and who did them) are required in order to reach a desired state.

To construct a proper policy behavior based on a state space, we need to define the following elements. Please note that by series we will understand a finite sum of elements.

- $K_s^t = c_1 + \dots + c_m = \sum_{i=1 \dots m} c_i \quad c_i \in C, s \in \{0, \dots, n_s - 1\}, t \in \mathbb{N}$ —a series, where summands are mission courses leading to the state s ;
- $N_K(K_s^t) \in \mathbb{N}$ —a function returning a number of elements in a given series. According to the earlier definition, for any series K_s^t this function returns a value of m (the greatest index of c_i);
- $F_{i,j}(x, t) = \sum_{k \in T_{i,j}} f_k(x, t) \quad i, j \in \{0, \dots, n_s - 1\}, t \in \mathbb{N}$ —a series, whose summands are mission progress functions from the i to the j state;
- $M_K^t = [K_0^t \quad \dots \quad K_{n_s-1}^t], t \in \mathbb{N}$ —a matrix whose elements are series indicating possible walks leading to each state. Index t denotes a number of steps made in a state space. By a step we understand a transition between vertices (including the situation where traversal does not change the vertex);
- $M_F^t = \begin{bmatrix} F_{0,0}(x, t) & \dots & F_{0,n_s-1}(x, t) \\ \dots & \dots & \dots \\ F_{n_s,0}(x, t) & \dots & F_{n_s-1,n_s-1}(x, t) \end{bmatrix}$ —a matrix of transitions between states.

Furthermore, we define two operations:

- $K_s^t \circ F_{i,j}(x, t) = \sum_{k \in T_{i,j}} \sum_{l=1 \dots N_K(K_s^t)} f_k(c_l, t)$ —a convolution of the series defined above;
- $M_K^{t+1} = M_K^t \cdot M_F^t$ —a multiplication of the matrices defined above. Elements of the new matrix are defined by the formula:

$$K_s^{t+1} = \sum_{k=0}^{n_s-1} K_k^t \circ F_{k,s}(x, t)$$

In order to generate all walks consisting of a specified number of steps from an initial state to a final state one must define the initial state, as a M_K^0 matrix and multiply subsequent results by M_F^t the specified number of times. The result will be a M_K^x matrix, whose summands in the i th column will indicate all possible walks with x steps that end in the i th state of the state space. If the element in the specified column is equal to 0, it means there is no such walk.

Summarizing our introductory example, we will demonstrate how to use the state space from Figure 2 with transition functions definitions listed in Table 4 to determine all sequences of actions that lead to the state denoted as s2. Each sequence is equivalent to behavior policy that, when applied, results in moving both agents to the area of type B.

To determine such sequences, we create two matrices, a matrix of transitions M_F^t and matrix of initial state M_K^0 . Having both of them, we can multiply subsequent M_K^t matrices by corresponding M_F^t matrices and check whether the third state (recall that numbering starts from 0) is reachable. By reachable, we understand having a value other than 0 in the specified column of the M_K^t matrix.

Definitions of both matrices are listed below:

$$M_F^t = \begin{bmatrix} f_{null} & f_1 + f_2 & f_3 \\ f_{null} & f_{null} & f_4 \\ f_{null} & f_{null} & f_{null} \end{bmatrix}$$

$$M_K^0 = [[(1,0), (2,0)], \emptyset \quad 0 \quad 0]$$

The $\langle(1,0), (2,0)\rangle$ tuple in the first column of M_K^0 matrix denotes that we have two objects. The zeros in both tuples indicate that each object starts the mission at the same moment.

Subsequent M_K^t matrices allow us to determine how a system changes when a specified number of actions occur. For example, M_K^1 gives us information about how the system evolves when one action occurs (analogously M_K^2 for two actions etc.).

In our example M_K^1 and M_K^2 are of the form:

$$M_K^1 = M_K^0 \cdot M_F^0 = [[\langle(1,0), (2,0)\rangle, \emptyset] \quad 0 \quad 0] \cdot \begin{bmatrix} f_{null}(c,0) & f_1(c,0) + f_2(c,0) & f_3(c,0) \\ f_{null}(c,0) & f_{null}(c,0) & f_4(c,0) \\ f_{null}(c,0) & f_{null}(c,0) & f_{null}(c,0) \end{bmatrix}$$

$$M_K^1 = [0 \quad [\langle(2,0), (1,2)\rangle, \{r1_1\}] + [\langle(1,0), (2,2)\rangle, \{r1_1\}] \quad [\langle(1,1), (2,1)\rangle, \{r2_1\}]]$$

$$M_K^2 = M_K^1 \cdot M_F^1 = M_K^1 \cdot \begin{bmatrix} f_{null}(c,1) & f_1(c,1) + f_2(c,1) & f_3(c,1) \\ f_{null}(c,1) & f_{null}(c,1) & f_4(c,1) \\ f_{null}(c,1) & f_{null}(c,1) & f_{null}(c,1) \end{bmatrix}$$

$$M_K^2 = [0 \quad 0 \quad [\langle(1,2), (2,2)\rangle, \{r1_1, r1_2\}] + [\langle(2,2), (1,2)\rangle, \{r1_1, r1_2\}]]$$

The interpretation of each of the above M_K^i matrices is as follows. The M_K^1 matrix indicates that with just one action there are two ways for the system to be in the state where one of the agents move to the area of type *B* and the other one will not take any action (as it is pointed out by the fact that its time is equal to 0). Both ways require specified agent to carry out the action represented by the *r1* rule. The same matrix also gives us information that with one action there is a possibility to reach *s2* state if both agents engage in cooperative execution of *r2* rule. Finally, the M_K^2 points out two walks in the state space that lead to the *s2* state. Both involve performing the action associated with *r1* rule two times (each time by a different agent).

It is worth pointing out that in a software implementation of the above algorithm labels should denote specific transition functions rather than reaction rules. While for this particular example it was sufficient to indicate what “kind” of changes (i.e., reaction rules) need to occur in the system for automated generation of behavior policies it is necessary to distinguish exactly what transformation (including who participated in a specific transformation) is required.

For more detailed examples we refer to [36].

2.5. Verification and Visualization of Behavior Policies

Below we will describe the algorithm to verify and illustrate the behavior policy. It consists of 4 phases. At the beginning of the discussion about each phase formal elements not introduced so far will be defined. Subsequent phases will be discussed so that newly introduced definitions will be directly used in the discussed phase. A diagram of relationships between phases is presented in Figure 3, from which it can be seen that the implementation of all the other phases is necessary for the execution of Phase 1. In contrast, Phases 4 and 2 are independent of the others.

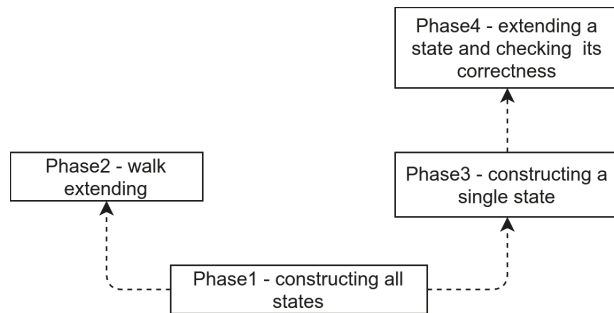


Figure 3. Diagram of relationships between phases of the algorithm. The direction of an arrow indicates the phase required by the phase from which the arrow emerges.

2.5.1. Phase 4—Applying a Single Transformation to Constructed State and Checking Correctness Beforehand

Phase 4 of the algorithm is responsible for verifying the correctness of the model and for expanding the scenario’s state at a particular point in time.

Input:

- A currently constructed state—a bigraph;
- A map of unique identifiers to vertices of the currently constructed state (a bijection);
- The reaction rule to be applied to the constructed state;
- A map of unique identifiers to rule’s redex vertices (bijection);
- State at the previous moment in time—a bigraph;
- A mapping of unique identifiers to state vertices at a previous point in time;
- First new unique identifier—used when a new task element appears after a transformation.

Output:

1. Option 1—the model is correct:
 - Newly constructed state—bigraph;
 - Mapping of unique identifiers to vertices of the newly constructed state;
 - First new unique identifier.
2. Option 2—the model is incorrect:
 - Information about the failed transformation. Whether the given reaction rule could not be applied to the state at the previous point in time or to the currently constructed state (given the mappings of unique identifiers to vertices).

Formal definitions:

- $X \subseteq \mathbb{N}$ —a set of unique identifiers (UIs) of task elements; It is used to track the environment and objects involved between system transformations. The idea behind this set is to assign to each task element a unique identifier, which makes it possible to check whether the task elements marked as taking part in a reaction rule are present in a given scenario state. The reaction rules themselves allow only to check whether alike (rather than the same) elements exist in both a reaction rule and a bigraph.
- $Corr_{Red} : \mathcal{R} \rightarrow Red$ —a function that assigns reaction rules to their corresponding redexes;
- $M_x \subset X^{V_b}$ $b \in Agt \cup Red$ —a set of functions assigning unique identifiers to elements of the support of a bigraph, which is either a scenario state or a redex of a reaction rule;
- $IsUpdatePossible : Agt \times M_x \times Red \times M_x \rightarrow \{true, false\}$ —a function that determines whether it is possible to apply a reaction rule to a given state, taking into account the mapping of the UIs to the state’s vertices and the mapping of the UIs to the redex vertices of that rule;
- $Update : Agt \times M_x \times \mathcal{R} \times M_x \times X \rightarrow Agt \times M_x \times X$ —a function that transforms the current state.

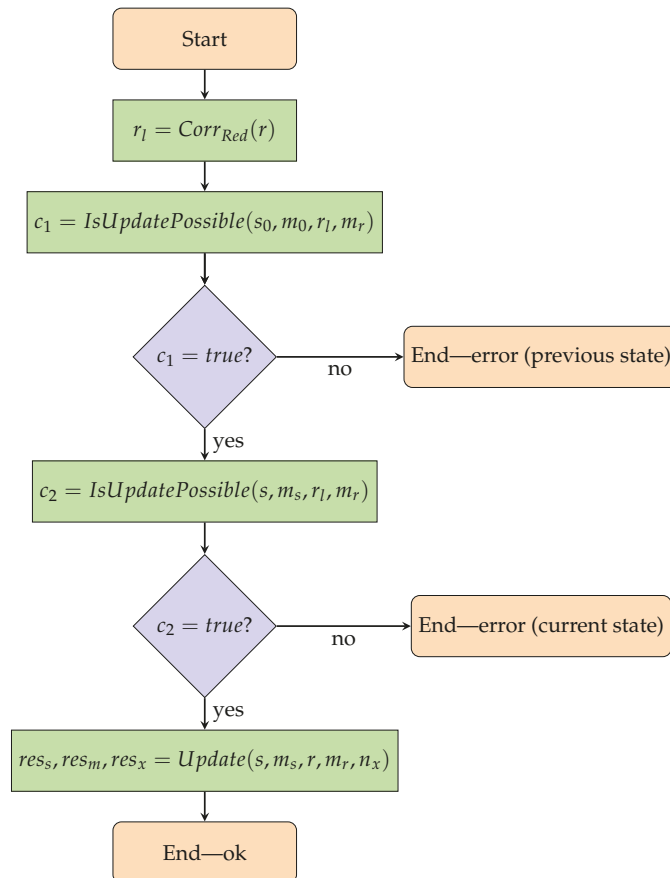
The flowchart of the Phase 4 algorithm is shown in Scheme 1. The input arguments of this algorithm and its results are described in Tables 5 and 6 respectively.

Table 5. Input data for the Phase 4 algorithm.

Variable	Description
$s \in Agt$	Currently constructed scenario state
$m_s \in M$	Mapping of UIs to vertices of currently constructed state s
$r \in \mathcal{R}$	Reaction rule
$m_r \in M_x$	Mapping of UIs to redex r vertices
$s_0 \in Agt$	State at the previous moment in time
$m_0 \in M_x$	Mapping of UIs to vertices of s_0
$n_x \in X$	The first new UI

Table 6. Output data of the Phase 4 algorithm.

Variable	Description
$res_s \in Agt$	Constructed state extended by application of the provided reaction rule
$res_m \in M_x$	Mapping of UIs to the vertices of res_s
$res_x \in X$	The first new UI



Scheme 1. Flowchart of the Phase 4 algorithm. The purpose of this phase is to check if a reaction rule extended by a map of unique identifiers to its vertices can be applied to the scenario state for the previous moment in time and the currently constructed one. If it is impossible to perform either of the mentioned operations it means that the model is incorrectly constructed. If both operations are feasible, the currently constructed state is modified based on the given reaction rule and the map of unique identifiers to its vertices.

2.5.2. Phase 3—Constructing Scenario State at a Given Moment of Time

Phase 3 of the algorithm is responsible for constructing the state of the scenario at a given point in time.

Input:

- State at the previous moment in time—bigraph;
- A map of unique identifiers to state elements at the previous moment in time;

- A set of walk elements combined with a UIs mapping to the vertices of the redex of the reaction rule associated with this walk element, a UIs mapping to the vertices of the input state and the smallest new UI from which new task elements will be numbered.
- A linear order relation defined on the above set;
- State-At-Time configuration of the system at the previous moment in time;
- A moment of time for which the system state is constructed;
- Number of objects.

Output:

1. Option 1—the model is correct:
 - A subset of the walk elements (given as input) that have not been used to construct the state at the given point in time;
 - State at the given moment in time;
 - Mapping of unique identifiers to state elements at the given point in time;
 - State-at-time configuration at the set point in time.
2. Option 2—the model is incorrect:
 - A currently constructed state with its UIs mapping that could not be transformed (if it is the cause of the Phase 4 error);
 - The state from the previous moment in time with its UIs mapping that could not be transformed (if it is the cause of the Phase 4 error);
 - Reaction rule with UIs mapping to its redex vertices, which was not successfully applied.

Formal definitions:

- $A \subset 2^{\mathbb{N}}$ —A collection of sets of mission object identifiers. The same identifiers are used in SAT configurations
- $W_M \subset \mathbb{N} \times T \times M_x \times M_x \times X \times (A \times \mathbb{N})$ —an extended walk consisting of:
 1. A positional number;
 2. A transition function;
 3. A map of UIs to redex vertices. The redex is associated with the reaction rule corresponding to the above transition function;
 4. A map of UIs to vertices of the output state of the extended walk element;
 5. First new UI assigned to a new task element created by applying the reaction rule (useful only if the reaction rule corresponding to the transition function creates new environment elements);
 6. A set of object identifiers involved in the walk element along with the duration of that transformation. In other words, it is information about which objects are involved in the transformation represented by the walk element and how long it will take.
- $<_{W_M}$ —linear order relation on the elements of the extended walk.
We will assume the following rule for ordering the elements of a walk:

$$\forall e_1 = (l_1, f_1, m_{1,r}, m_{1,in}, n_1, (A_1, d_1)), e_2 = (l_2, f_2, m_{2,r}, m_{2,in}, n_2, (A_2, d_2)) \in W_M$$

$$e_1 <_{W_M} e_2 \leftrightarrow l_1 < l_2$$

- $First_M : 2^{W_M} \rightarrow W_M \times 2^{W_M}$ —a function that returns the “smallest” element of the walk and the “truncated” walk;
- $Corr_{Tra} : T \rightarrow Tra$ —a function that assigns transition functions to transitions from TTS;
- $Objects_U : I \times (A \times \mathbb{N}) \rightarrow I$ —SAT configuration update function. Takes a current configuration and a set of objects for which the time will be changed along with the value by how much. The result is the new SAT configuration;
- $Objects_F : I \times \mathbb{N} \rightarrow A$ —a function that determines for which objects activities are scheduled later than the moment of time for which the scenario state is constructed. Takes a SAT configuration and the moment of time for which the state is generated;

- $Corr_{\mathcal{R}} : Tra \rightarrow \mathcal{R}$ —a function assigning reaction rules to transitions from TTS.

The flowchart of Phase 3 of the algorithm is shown in Scheme 2. The input arguments for the algorithm are described in Table 7. The auxiliary variables, some of which are also outcomes of Phase 3, are described in Table 8. The outcome of Phase 3 is described in Table 9.

Table 7. Input data for the Phase 3 algorithm.

Variable	Description
$s_0 \in Agt$	State at the previous point in time.
$m_0 \in M_x$	Mapping of UIs to vertices of s_0 .
$W \subseteq W_{M'}, <_{W_M}$	A walk and the linear order relation on its elements.
$i_0 \in I$	The SAT configuration at the previous moment of time.
$d \in \mathbb{N}$	The moment of time for which the scenario state is constructed.
$n_o \in \mathbb{N}$	Number of objects.

Table 8. Auxiliary variables of Phase 3 algorithm.

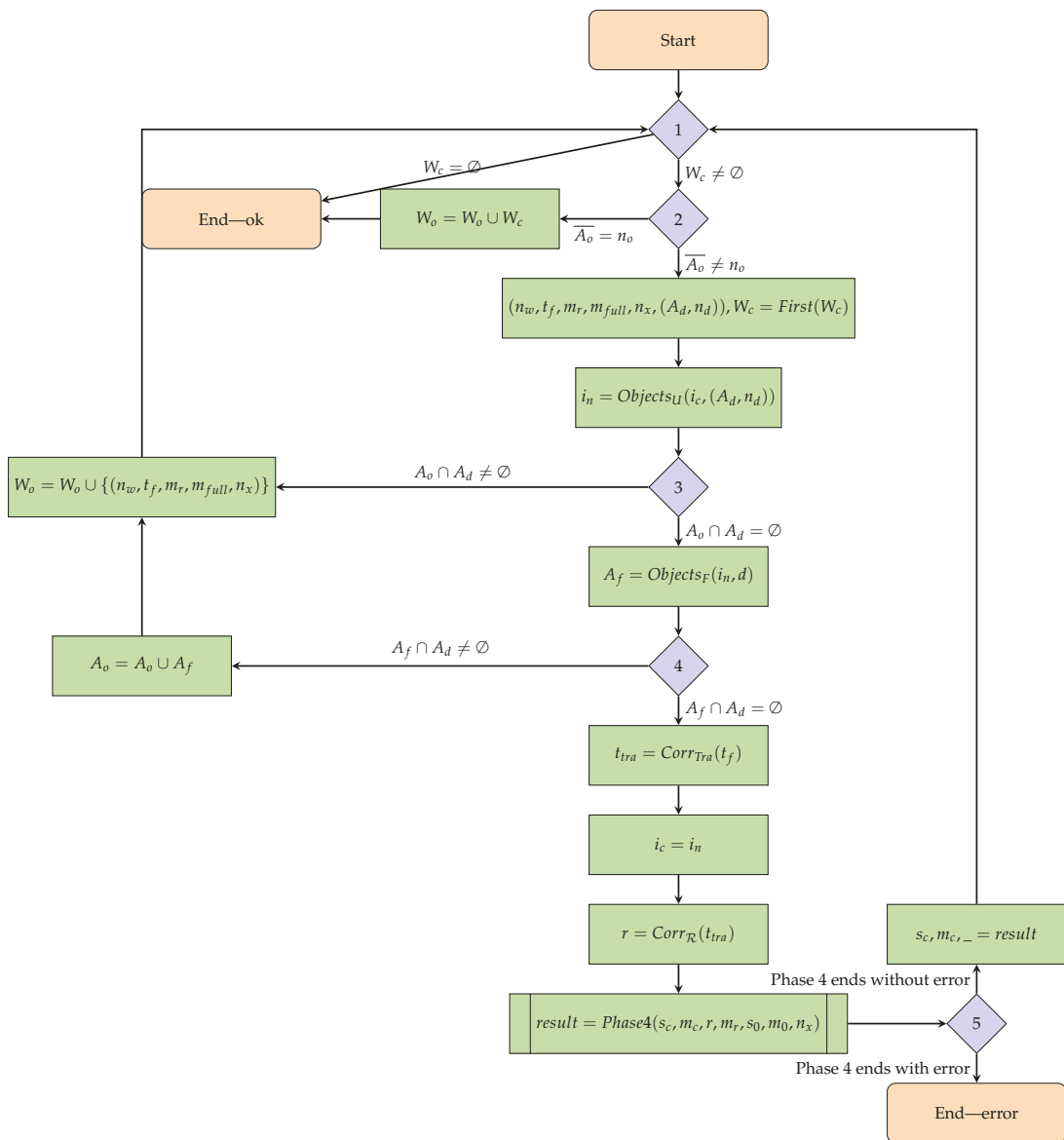
Variable	Description
$s_c \in Agt$	Current constructed state. The initial value is s_0 .
$m_c \in M_x$	Mapping of UIs to vertices of s_c .
$i_c \in I$	SAT configuration of the currently constructed state. The initial value is i_0 .
$A_o \in A$	A set of object identifiers, skipped in the constructed state. The initial value is the empty set.
$W_c \subseteq W$	A collection of usable walk elements. The initial value is W .
$W_o \subseteq W$	A collection of unused walk elements. The initial value is the empty set.

Table 9. Output data of Phase 3 algorithm.

Variable	Description
$W_o \subset W$	Unused walk elements that will be used to construct subsequent scenario states.
$s_c \in Agt$	System state.
$m_c \in M_x$	Mapping of UIs to vertices of s_c .
$i_c \in I$	SAT configuration at time d .

Noteworthy are the conditions checked in the subsequent steps of Phase 3 of the algorithm. Comments for each of them are given below.

1. The first condition checked is if we have reached the end of a walk. If so, then surely the state currently constructed is the state for the given moment of time.
2. Do we omit actions of all mission objects? If so, the state constructed so far is the state for the given moment of time.
3. Do any objects involved in the current action belong to the set of skipped objects? If so, we omit this walk element.
4. Will all objects involved in the current action have finished before the moment d ? If not, we disregard that activity in the currently constructed state and add those objects to the set of skipped objects.
5. If Phase 4 is not completed correctly, it means that the model is incorrect.



Scheme 2. Flowchart of the Phase 3 algorithm. The goal of this phase is to construct the state of a scenario at a given point in time. This phase runs in a loop until there are no available walk elements or when an execution of Phase 4 ends with an error. It takes subsequent elements of the input walk and updates both the current SAT configuration and a scenario state. If the mission objects will not have finished the activity represented by the currently processed walk element before or at the specified moment of time then the SAT configuration and state updates are not performed. The same thing happens if an activity involves objects participating in other activities that would end in a future and that have already been skipped.

2.5.3. Phase 2—Extending a Previously Constructed Walk

Phase 2 of the algorithm is responsible for extending a walk to the form acceptable by Phase 3.

Input:

- A walk resulting from the algorithm presented in Section 2.4;
- Number of objects.

Output:

- Extended walk.

Formal definitions:

- $W \subset \mathbb{N} \times T$ —a walk. The first element denotes the positional number of the transition function that is the second element of the tuple;
- $<_W$ —linear order relation on the elements of the set W .
As in the case of the set W_M , we define the order relation by the following rule:

$$\forall e_1 = (n_1, f_1), e_2 = (n_2, f_2) \in W \quad e_1 <_W e_2 \leftrightarrow n_1 < n_2$$

- $First : 2^W \rightarrow W \times 2^W$ —a function that returns the “smallest” walk element and a truncated walk;
- $Trans : Tra \times M_x \times X \rightarrow M_x \times M_x \times X$ —a function that transforms a mapping of unique identifiers based on the given transition and the first new identifier (in case new environment elements appear in the output state of the transition and need to be tagged). The results are: a new UIs map to the redex of the reaction rule corresponding to the provided transition, a UIs mapping to the output state of the transition, and a new smallest UI;
- $U \subset I^I$ —a set of functions that update SAT configurations;
- $Corr_U : T \rightarrow U$ —a function that assigns transition functions to their corresponding SAT configuration update functions;
- $Time_U : I \times U \rightarrow I$ —a SAT configuration update function;
- $Time_D : I \times I \rightarrow A \times \mathbb{N}$ —a time difference function for individual objects between SAT configurations. Returns information about which objects are involved in the transformation and how long it takes.

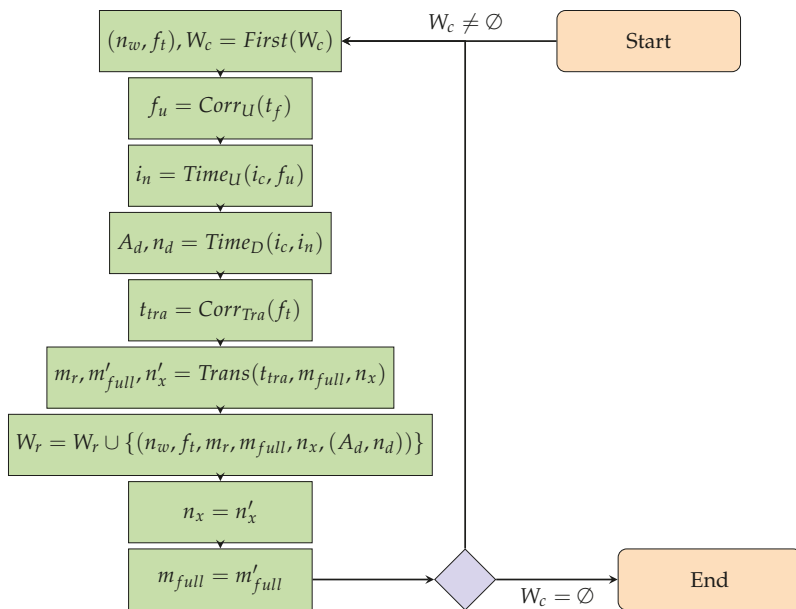
The flowchart of the Phase 2 algorithm is shown in Scheme 3. Input arguments are described in Table 10; auxiliary variables and the result of this phase are discussed in Table 11.

Table 10. Input data for the Phase 2 algorithm.

Variable	Description
$W, <_W$	A walk with a linear order relation on its elements.
n_o	Number of objects.

Table 11. Auxiliary variables of the Phase 2 algorithm.

Variable	Description
$n_x \in X$	The value of a first new UI. The initial value is the number of vertices of the input state of the first walk element.
$m_{full} \in M_x$	The current UIs mapping to the vertices of the last processed output state. The initial value is a function that assigns consecutive natural numbers to the vertices of the input state of the first element of the walk.
$W_r \subseteq W_M$	Elements of the extended walk. The initial value is the empty set. This is the result of this phase.
$W_c \subseteq W$	A subset of walk elements that have not been processed yet. The initial value is W .
$i_c \in I$	Current SAT configuration. The initial value is $((1,0), \dots, (n_o,0))$.



Scheme 3. Flowchart of the Phase 2 algorithm. The goal of Phase 2 is to expand each element of a provided walk to the form acceptable by Phase 3. Each element of the walk is coupled with the duration of its corresponding activity along with the identifiers of the objects (not unique identifiers of task elements) that participate in the activity and two bijections. The first function maps unique identifiers to vertices of the redex of the reaction rule associated with the currently processed walk element. With this function, we know exactly who is participating in the activity. The second function maps unique identifiers to the output state of a processed TTS transition (derived from the walk element). With this function, we know exactly which task element corresponds to which vertex after applying the reaction rule. The second function is used in the next iteration of Phase 2.

2.5.4. Phase 1—Constructing All Scenario States and Checking the Correctness of a Given Walk

Phase 1 of the algorithm is its entry point. It is responsible for verifying a model and constructing scenario states at successive moments in time.

Input:

- Number of objects;
- A walk with a linear order relation on its elements.

Output:

1. The model is correct:
 - A set of scenario states at consecutive moments in time with corresponding mappings of unique identifiers to the vertices of these states and SAT configurations;
2. The model is incorrect:
 - The moment of time for which the scenario state could not be generated;
 - The element that could not be transformed (constructed state or state at some point in time);
 - The reaction rule corresponding to the unsuccessful transformation;
 - The UIs mapping of the element that could not be transformed and the redex of the above reaction rule.

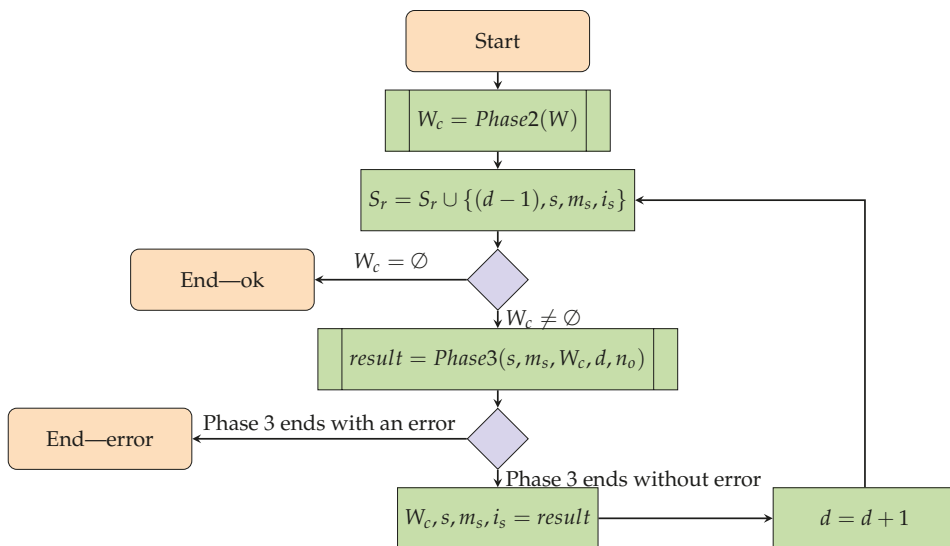
Phase 1 input parameters are described in Table 12. The auxiliary variables along with the outcome are discussed in Table 13. The flowchart of the Phase 1 algorithm is shown in the Scheme 4.

Table 12. Input data for the Phase 1 algorithm.

Variable	Description
$W, <_W$	A walk with a linear order relation on its elements.
n_o	Number of objects.

Table 13. Auxiliary variables for the Phase 1 algorithm.

Variable	Description
$W_c \subseteq W_M$	A set of extended walk elements that have not been used yet. The initial value is the empty set but it is properly initialized with the result of Phase 2.
d	The current moment of time for which a scenario state is constructed. The initial value is 1.
$s \in Agt$	The scenario state at the time $d - 1$. The initial value is the input state of the first walk element W .
$m_s \in M_x$	Mapping of UIs to vertices of the state s . The initial value is a bijection of consecutive natural numbers on the vertices of s .
$i_s \in I$	SAT configuration for the scenario state at the time $d - 1$. The initial value is $((1, 0), \dots, (n_o, 0))$.
$S_r \subset \mathbb{N} \times Agt \times M_x \times I$	A collection of states at successive moments in time with their corresponding UIs mapping and SAT configurations. The initial value is the empty set. This is the result of this phase.



Scheme 4. A flowchart of the Phase 1 algorithm. The goal of Phase 1 is to verify a model and construct the subsequent states of a scenario using a provided walk. In the first step the walk is extended to the form acceptable by Phase 3. Then the model verification and construction of successive scenario states is performed in a loop. The loop is executed until Phase 3 ends with either an error or when there are no more elements of the walk to further construct states of the scenario from.

3. Results

This section will provide example use cases of the algorithm discussed in the previous section. The first two examples show in detail how the algorithm detects errors in a model and how it constructs successive scenario states. The next examples present how to check the fulfillment of non-functional requirements for systems designed with our methodology. Finally, the problem of memory complexity of convolution operation performed during a construction of walks in a state space is discussed. We also provide a few propositions how to address this issue.

3.1. Model Verification Example

3.1.1. Introduction

The first example will demonstrate how the algorithm can detect that a system is incorrectly designed.

A task (as defined in Section 2.1) for this example consists of six elements, two actions that can be performed, and one goal. The task elements comprise three areas with two robots and an object to be carried between the areas. The goal of the task is for the robots (denoted by vertices with the control *B*) to move the object (denoted by a vertex with the control *O*) from the area *AT1* to the area *AT3*. The initial state of this system is shown in Figure 4. We will use two reaction rules to generate a tracking bigraphical reactive system: *mov1* and *mov2* depicted in Figure 5a,b, respectively.

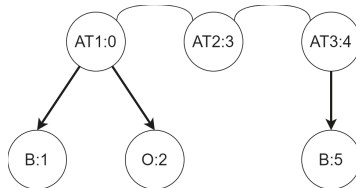


Figure 4. The initial state of a system in the example of verifying model correctness.

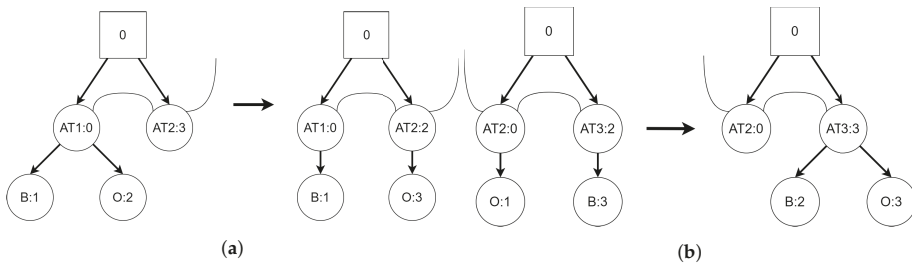


Figure 5. Reaction rules for the example of verifying a model. All residue functions are identities. (a) Reaction rule *mov1*. (b) Reaction rule *mov2*.

The elements of a tracking transition system for this example are shown in Table 14.

If we categorize the task elements as presented in Table 15 then we can transform the TTS from Table 14 into the state space as in Figure 6. However, this will not be a valid state space because no time is taken into account for the object being moved (i.e., it is not treated as a passive or active object as defined in Section 2.1).

3.1.2. Using the Algorithm for Model Verification

Walk $S_0 \xrightarrow{f_1} S_1 \xrightarrow{f_2} S_2$ can be represented as $W = \{(0, f_1), (1, f_2)\}$. Assuming that both actions associated to the reaction rules take 1 unit of time to complete, in Phase 2 both elements of set W will be transformed to form:

1. $(0, f_1, \{(0,0), (1,1), (2,2), (3,3)\}, \{(0,0), (1,1), (2,2), (3,3), (4,4), (5,5)\}, 6, (\{1\}, 1))$
2. $(1, f_2, \{(2,0), (3,1), (4,2), (5,3)\}, \{(0,0), (1,1), (2,3), (3,2), (4,4), (5,5)\}, 6, (\{2\}, 1))$

The method of constructing m_r and m'_{full} functions that result from *Trans* function in Phase 2 is shown below.

The rule of constructing m_r function:

$$\forall x \in \{0, \dots, n_x - 1\} \quad m_r(x) = par^{-1}(m_{full}(x))$$

where par^{-1} is the inverse function to par being an element of t_{tra} . In this case, the functions f_1, f_2, \dots, f_8 correspond to the subsequent rows in Table 14.

The rule of constructing m'_{full} function:

$$\forall x \in \{0, \dots, n_x - 1\} \quad m'_{full}(x) = res^{-1}(m_{full}(x))$$

res^{-1} is the inverse function of res which is an element of t_{tra} .

Table 16 lists the successive steps of the algorithm that will lead to a detection of an error in the model. The reason why this model is incorrect is not because the redex of the rule *mov2* is not in the 0 state but because the moved object is categorized as an element of the environment, thus we do not take into account the passage of time for it. As a result, the reaction rules create the appearance of being independent of each other when in fact the execution of *mov2* rule is dependent on the execution of the rule *mov1*. To fix the model, the relocated object needs to be categorized as a passive object and one need to add a reaction rule allowing a robot that is in *AT3* area to wait until the object being moved is in *AT2* area.

Table 14. Tracking transition system for the first example.





Input State	Label	Output State	Par & Res
	<i>mov1</i>		$\{(0,0), (1,1), (2,2), (3,3)\}$ $\{(0,0), (1,1), (2,3), (3,2), (4,4), (5,5)\}$
	<i>mov2</i>		$\{(0,2), (1,3), (2,4), (3,5)\}$ $\{(0,2), (1,4), (2,5), (3,3), (4,0), (5,1)\}$

Table 15. Categorization of task elements for the first example. Note that this produces an incorrect model because the moved object is considered an environment element.

Category of Task Elements	Elements Belonging to the Category
Environment	$\{AT1, AT2, AT3, O\}$
Passive objects	\emptyset
Active objects (agents)	$\{B\}$

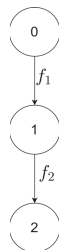
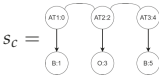
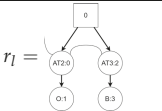
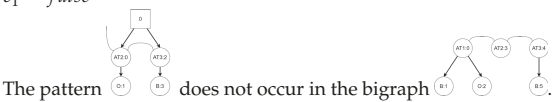


Figure 6. Incorrect state space for the task from the first example.

Table 16. Subsequent steps of the algorithm in the model validation example.

Phase	Step	Result/Comment
1	$S_r = S_r \cup \{((d - 1, s, m_s, i_s))\}$	$S_r = \left\{ \left(\begin{array}{c} 0, \begin{array}{c} \text{AT10} \quad \text{AT22} \quad \text{AT34} \\ \text{B1} \quad \text{O2} \quad \text{B3} \end{array} \\ \{(0,0), (1,1), (2,2), (3,3), (4,4), (5,5)\}, \\ (1,0), (2,0) \} \end{array} \right) \right\}$
1	Phase3(...)	
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 0$ $t_f = f_1$ $m_r = \{(0,0), (1,1), (2,2), (3,3)\}$ $m_{full} = \{(0,0), (1,1), (2,2), (3,3), (4,4), (5,5)\}$ $n_x = 6$ $A_d = \{1\}$ $n_d = 1$ $W_c = W_c \setminus \{e_1\} = \{e_2\}$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1,1), (2,0))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} =$ first row of Table 14
3	$i_c = i_n$	$i_c = (1,1), (2,0)$
3	$r = Corr_{\mathcal{R}}(t_{tra})$	$r =$ reaction rule <i>mov1</i>
3	$result = Phase4(s_c, m_c, r, s_0, m_0, n_x)$	Phase 4 completed without error
3	$s_c, m_c, n_x = result$	 $m_c = \{(0,0), (1,1), (2,3), (3,2), (4,4), (5,5)\}$ $(m_c$ is calculated in the same way as m'_{full} in Phase 2) $n_x = 6$
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 1$ $t_f = f_2$ $m_r = \{(2,0), (3,1), (4,2), (5,3)\}$ $m_{full} = \{(0,0), (1,1), (2,3), (3,2), (4,4), (5,5)\}$ $n_x = 6$ $A_d = \{2\}$ $n_d = 1$ $W_c = W_c \setminus \{e_2\} = \emptyset$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1,1), (2,1))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} =$ second row of Table 14
3	$i_c = i_n$	$i_c = ((1,1), (2,1))$
3	$r = Corr_{\mathcal{R}}(t_{tra})$	$r =$ reaction rule <i>mov2</i>
4	$r_l = Corr_{Red}(r)$	
4	$c_1 = IsUpdatePossible(s_0, m_0, r_l, m_r)$	$c_1 = false$
		
1	End—error	

3.2. Example of Scenario States Visualization

3.2.1. Introduction

The second example will demonstrate the problem of visualizing a scenario and how our algorithm can help in solving it. A task for this example is composed of three areas and two robots of the same type. The initial state of the system is presented in Figure 7. The tracking bigraphical reactive system for the purpose of this example consists of two reaction rules, $r1$ and $r2$, shown in Figure 8a,b, respectively. The goal of the task is to move the two robots from the area $AT1$ to the area $AT3$.

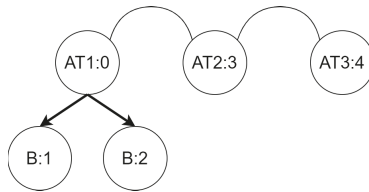


Figure 7. The initial state of a system for the scenario visualization example.

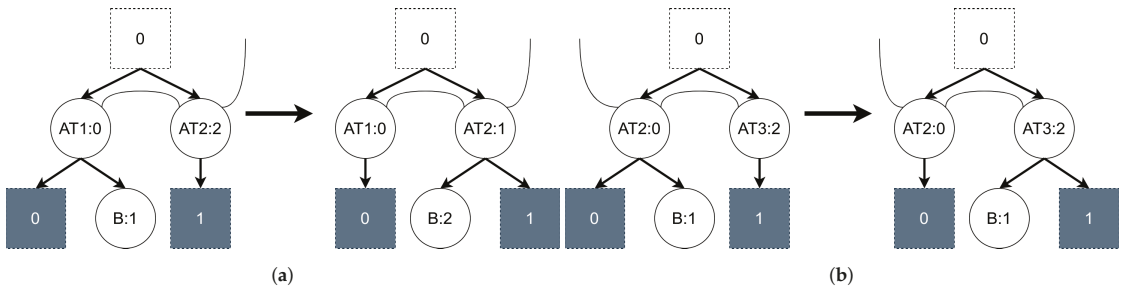


Figure 8. Reaction rules for the example of scenario visualization. (a) Reaction rule $r1$. $\tau = \{(0,0), (1,2), (2,1)\}$. (b) Reaction rule $r2$. $\tau = \{(0,0), (1,1), (2,2)\}$.

Tracking Transition System generated from this TBRS is defined in Table 17.

The tracking Transition System from Table 17 can be transformed into a state space as in Figure 9. Now, suppose that a walk chosen for the behavior policy is of the form:

$$S_0 \xrightarrow{f_1} S_1 \xrightarrow{f_3} S_2 \xrightarrow{f_5} S_4 \xrightarrow{f_8} S_5$$

The above walk never “passes” through a state where both robots are in $AT2$ area. (that is, through the state S_3). Such a situation must occur for the following reasons. For a walk representing four activities (because it consists of four arcs), that can correspond only to reaction rules from Figure 8 a course of a mission for each robot must take the form of moving from an $AT1$ area to an $AT2$ area and then from $AT2$ to $AT3$ area. Since the activities represented by the reaction rules are not cooperative (each of the reaction rules involve only one agent) the movements will be performed in parallel. We also know that the time required to perform both activities will be the same for both agents (because agents are of the same type and perform the same type of activity) so the successive movements will end at the same moment. Because of all that, during a mission there must occur a situation where both robots are at an $AT2$ area at the same time. Therefore, the algorithm for constructing subsequent scenario states must be able to construct states that are not “on” a provided walk.

Table 17. Tracking Transition System for the second example.

Input State	Label	Output State	Par & Res
	r1		$par = \{(0,0), (1,2), (2,3)\}$ $res = \{(0,0), (1,3), (2,2), (3,1), (4,4)\}$
	r1		$par = \{(0,0), (1,1), (2,3)\}$ $res = \{(0,0), (1,3), (2,1), (3,2), (4,4)\}$
	r2		$par = \{(0,1), (1,2), (2,4)\}$ $res = \{(0,1), (1,4), (2,2), (3,0), (4,3)\}$
	r1		$par = \{(0,1), (1,3), (2,1)\}$ $res = \{(0,0), (1,1), (2,2), (3,3), (4,4)\}$
	r1		$par = \{(0,3), (1,4), (2,0)\}$ $res = \{(0,3), (1,0), (2,4), (3,1), (4,2)\}$
	r2		$par = \{(0,1), (1,2), (2,4)\}$ $res = \{(0,0), (1,1), (2,3), (3,4), (4,2)\}$
	r2		$par = \{(0,1), (1,3), (2,4)\}$ $res = \{(0,0), (1,1), (2,2), (3,4), (4,3)\}$
	r2		$par = \{(0,1), (1,2), (2,3)\}$ $res = \{(0,0), (1,3), (2,4), (3,0), (4,2)\}$

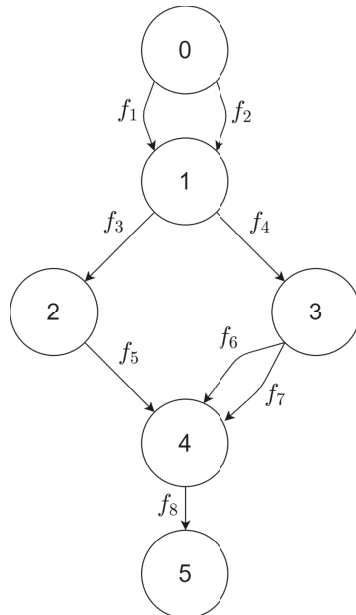


Figure 9. The state space generated from Tracking Transition System from Table 17.

3.2.2. Using the Algorithm to Construct Scenario States

The walk $S_0 \xrightarrow{f_1} S_1 \xrightarrow{f_3} S_2 \xrightarrow{f_5} S_4 \xrightarrow{f_8} S_5$ can be presented as:

$$W = \{(0, f_1), (1, f_3), (2, f_5), (3, f_8)\}$$

A linear order relation on the set W has the form:

$$\prec_W = \left\{ \begin{array}{l} ((0, f_1), (1, f_3)), ((0, f_1), (2, f_5)), ((0, f_1), (3, f_8)), \\ ((1, f_3), (2, f_5)), ((1, f_3), (3, f_8)), \\ ((2, f_5), (3, f_8)) \end{array} \right\}$$

Assuming that execution of each reaction rules takes one unit of time, in Phase 2 the consecutive elements of set W will be transformed to the following form:

- $e_1 = (0, f_1, \{(0, 0), (2, 1), (3, 2)\}, \{(0, 0), (1, 1), (2, 2), (3, 3), (4, 4)\}, 5, (\{2\}, 1))$
- $e_2 = (1, f_3, \{(3, 0), (2, 1), (4, 2)\}, \{(0, 0), (3, 1), (2, 2), (1, 3), (4, 4)\}, 5, (\{2\}, 1))$
- $e_3 = (2, f_5, \{(0, 0), (1, 1), (3, 2)\}, \{(0, 3), (1, 4), (2, 2), (3, 0), (4, 1)\}, 5, (\{1\}, 1))$
- $e_4 = (3, f_8, \{(3, 0), (1, 1), (4, 2)\}, \{(0, 0), (1, 2), (2, 4), (3, 1), (4, 3)\}, 5, (\{1\}, 1))$

Knowing the above, we can define an extended walk.

$$W_M = \{e_1, e_2, e_3, e_4\}$$

The linear order relation remains unchanged between elements, i.e.:

$$\prec_{W_M} = \{(e_1, e_2), (e_1, e_3), (e_1, e_4), (e_2, e_3), (e_2, e_4), (e_3, e_4)\}$$

Steps of the algorithm to construct the subsequent scenario states are presented in Table 18.

Table 18. Successive steps of the algorithm in the example of visualizing a scenario.

Phase	Step	Result/Comment
1	$S_r = S_r \cup \{(d - 1, s, m_s, i_s)\}$	$S_r = \left\{ \left(\begin{array}{c} \text{Diagram: } 0 \text{ with nodes } RT0, RT1, RT2, RT3, RT4 \text{ and edges } R1, R2 \\ \{(0, 0), (1, 1), (2, 2), (3, 3), (4, 4)\}, \\ (1, 0), (2, 0) \end{array} \right) \right\}$
1	Phase3(...)	
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 0$ $t_f = f_1$ $m_r = \{(0, 0), (2, 1), (3, 2)\}$ $m_{full} = \{(0, 0), (3, 1), (2, 2), (1, 3), (4, 4)\}$ $n_x = 5$ $A_d = \{2\}$ $n_d = 1$ $W_c =$ $W_c \setminus \{e_1\} =$ $\{e_2, e_3, e_4\}$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1, 0), (2, 1))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} =$ first row of Table 17
3	$i_c = i_n$	$i_c = ((1, 0), (2, 1))$
3	$r = Corr_{\mathcal{R}}(t_{tra})$	$r =$ reaction rule $r1$
3	$result = Phase4(s_c, m_c, r, s_0, m_0, n_x)$	Phase 4 completed without error

Table 18. Cont.

Phase	Step	Result/Comment
3	$s_c, m_c, n_x = result$	$s_c = \begin{matrix} (m) & & (m) \\ \diagdown & & / \\ (s) & & (s) \\ \diagup & & \diagdown \\ (a) & & (a) \end{matrix}$ $m_c = \{(0,0), (3,1), (2,2), (1,3), (4,4)\}$ $n_x = 5$
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 1$ $t_f = f_3$ $m_r = \{(3,0), (2,1), (4,2)\}$ $m_{full} = \{(0,0), (3,1), (2,2), (1,3), (4,4)\}$ $n_x = 5$ $A_d = \{2\}$ $n_d = 1$ $W_c = W_c \setminus \{e_2\} = \{e_3, e_4\}$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1,0), (2,2))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \{2\}$
3	$A_o = A_o \cup A_f$	$A_o = \emptyset \cup \{2\} = \{2\}$
3	$W_o = W_o \cup \{(n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d))\}$	$W_o = \emptyset \cup \{e_2\}$
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 2$ $t_f = f_5$ $m_r = \{(0,0), (1,1), (3,2)\}$ $m_{full} = \{(0,3), (1,4), (2,2), (3,0), (4,1)\}$ $n_x = 5$ $A_d = \{1\}$ $n_d = 1$ $W_c = W_c \setminus \{e_3\} = \{e_4\}$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1,1), (2,1))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} = \text{fifth row of Table 17}$
3	$i_c = i_n$	$i_c = ((1,1), (2,1))$
3	$r = Corr_{\mathcal{R}}(t_{tra})$	$r = \text{reaction rule } r1$
3	$result = Phase4(s_c, m_c, r, s_o, m_o, n_x)$	Phase 4 completed without error
3	$s_c, m_c, n_x = result$	$s_c = \begin{matrix} (m) & & (m) & & (m) \\ \diagdown & & / & & \diagdown \\ (s) & & (s) & & (s) \\ \diagup & & \diagdown & & / \\ (a) & & (a) & & (a) \end{matrix}$ $m_c = \{(0,0), (3,1), (2,2), (1,3), (4,4)\}$ $n_x = 5$
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 3$ $t_f = f_8$ $m_r = \{(3,0), (1,1), (4,2)\}$ $m_{full} = \{(0,0), (1,2), (2,4), (3,1), (4,3)\}$ $n_x = 5$ $A_d = \{1\}$ $n_d = 1$ $W_c = W_c \setminus \{e_4\} = \emptyset$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1,2), (2,1))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \{1\}$

Table 18. Cont.






Phase	Step	Result/Comment
3	$A_o = A_o \cup A_f$	$A_o = \{2\} \cup \{1\} = \{1, 2\}$
3	$W_o = W_o \cup \{(n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d))\}$	$W_o = \{e_2\} \cup \{e_4\}$
3	End—ok	
1	$W_c, s, m_s, i_s = result$	$W_c = \{e_2, e_4\}$  $s =$ $m_s = \{(0, 0), (3, 1), (2, 2), (1, 3), (4, 4)\}$ $i_s = ((1, 1), (2, 1))$
1	$d = d + 1$	$d = 2$
1	$S_r = S_r \cup \{(d - 1, s, m_s, i_s)\}$	$S_r = S_r \cup \left\{ \left(\begin{array}{c} 1, \text{ (AT10, AT21, AT24)} \\ \text{ (S1, S2)} \\ \{(0, 0), (3, 1), (2, 2), (1, 3), (4, 4)\}, \\ ((1, 1), (2, 1)) \end{array} \right) \right\}$ 
1	Phase3(...)	
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 1$ $t_f = f_3$ $m_r = \{(3, 0), (2, 1), (4, 2)\}$ $m_{full} = \{(0, 0), (3, 1), (2, 2), (1, 3), (4, 4)\}$ $n_x = 5$ $A_d = \{2\}$ $n_d = 1$ $W_c = W_c \setminus \{e_2\} = \{e_4\}$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1, 1), (2, 2))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} =$ third row of Table 17
3	$i_c = i_n$	$i_c = ((1, 1), (2, 2))$
3	$r = Corr_{\mathcal{R}}(t_{tra})$	$r =$ reaction rule r_2
3	$result = Phase4(s_c, m_c, r, s_0, m_0, n_x)$	Phase 4 completed without error
3	$s_c, m_c, n_x = result$	 $m_c = \{(0, 0), (3, 1), (4, 3), (1, 2), (2, 4)\}$ $n_x = 5$
3	$n_w, t_f, m_r, m_{full}, n_x, (A_d, n_d), W_c = First_M(W_c)$	$n_w = 3$ $t_f = f_8$ $m_r = \{(3, 0), (1, 1), (4, 2)\}$ $m_{full} = \{(0, 0), (1, 2), (2, 4), (3, 1), (4, 3)\}$ $n_x = 5$ $A_d = \{1\}$ $n_d = 1$ $W_c =$ $W_c \setminus \{e_4\} = \emptyset$
3	$i_n = Objects_U(i_c, (A_d, n_d))$	$i_n = ((1, 2), (2, 2))$
3	$A_f = Objects_F(i_n, d)$	$A_f = \emptyset$
3	$t_{tra} = Corr_{Tra}(t_f)$	$t_{tra} =$ eighth row of Table 17

Table 18. Cont.

Phase	Step	Result/Comment
3	$i_c = i_n$	$i_c = ((1, 2), (2, 2))$
3	$r = \text{Corr}_{\mathcal{R}}(t_{tra})$	$r = \text{reaction rule } r2$
3	$\text{result} = \text{Phase4}(s_c, m_c, r, s_0, m_0, n_x)$	Phase 4 completed without error
3	$s_c, m_c, n_x = \text{result}$	$s_c = $  $m_c = \{(0, 3), (3, 0), (4, 1), (1, 4), (2, 2)\}$ $n_x = 5$
3	End—ok	
1	$W_c, s, m_s, i_s = \text{result}$	$W_c = \emptyset$ $s = $  $m_s = \{(0, 3), (3, 0), (4, 1), (1, 4), (2, 2)\}$ $i_s = ((1, 2), (2, 2))$
1	$d = d + 1$	$d = 3$
1	$S_r = S_r \cup \{((d - 1, s, m_s, i_s))\}$	$S_r = S_r \cup \left\{ \left(\left(\begin{array}{c} \text{2, } \begin{array}{c} \text{AT1} \quad \text{AT2} \quad \text{AT3} \\ \text{AT4} \quad \text{AT5} \end{array} \end{array} \right), \right. \right. \\ \left. \left. \left\{ (0, 3), (3, 0), (4, 1), (1, 4), (2, 2) \right\}, \right. \right. \\ \left. \left. \left. (1, 2), (2, 2) \right) \right\} \right.$
1	End—ok	

The result of the algorithm contains a state where both robots are in AT2 area despite the fact that the walk has not passed through such state. A Gantt diagram for this scenario is shown in Figure 10. Both robots are performing actions r1 and r2 in parallel.

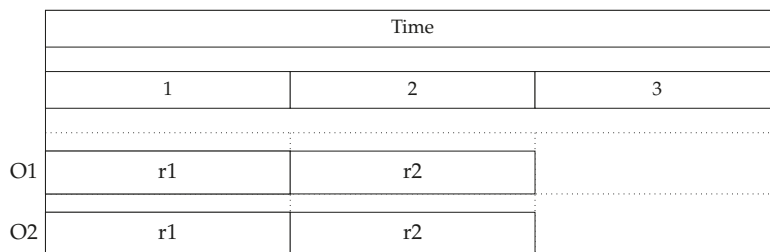


Figure 10. A Gantt diagram for the scenario from the second example. Activities marked as t in the row preceded by Ox denote involvement of the element x (x is the unique identifier of a task element given at its first appearance or at the beginning of a scenario) during the activity t . Only elements that are active objects are included in the diagram.

Functions tupled with each state allow to “track” task elements between states. For example, the function $m_s = (0, 0), (1, 1), (2, 2), (3, 3), (4, 4)$ for the state at time 0 indicates that the object tagged with the unique identifier 2 (the argument of m_s function) is represented by the vertex with identifier 2 (the value of m_s function for argument 2). The support of a bigraph itself does not track its elements between transitions, as can be seen by comparing the state of the system at time 0 and time 1. For example, knowing that there is one area of each type, we have no doubt that a vertex with the control AT1 represents the same object in both states even though the support element assigned to each vertex is different between states. However, we do not have such certainty for vertices with controls

of the type *B*. Unique identifiers point to unique objects between states, even if those objects have changed the controls representing them.

Here is an example based on the elements of set S_r from Table 18 how to use a unique identifier mapping. For the state at time 1, the UI with the value of 3 points to the vertex with identifier 1. This means that it is the same task element that in the state at time 0 is represented by the vertex with identifier value of 3 and the same element that at time 2 is represented by the vertex with support 0.

3.3. Example of Verifying the Fulfillment of Non-Functional Requirements

The last example is intended to demonstrate how non-functional requirements can be defined for systems designed using our methodology and determine whether these requirements have been satisfied.

For this example, we will define a task of relocating items in a warehouse. The goal of this task is for two robots to deploy items of different types from the warehouse to unloading areas. The initial state of the task is depicted in Figure 11. The interpretation of each control is shown in Table 19. Six reaction rules are defined for this system; all of them are listed and described in Table 20. For this example, the graphical representation of reaction rules is omitted because it will not be relevant.

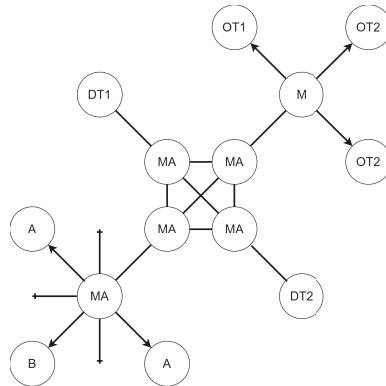


Figure 11. The initial state of a system in the example of checking whether non-functional requirements are met.

Table 19. Interpretation of controls in the example of checking whether non-functional requirements are satisfied.

Control	Real World Object
<i>A</i>	Robot
<i>MA</i>	Warehouse area—robots can move between them.
<i>B</i>	Beacon—indicates the warehouse area where robots should return after relocating objects.
<i>M</i>	Warehouse—it stores objects to be moved.
<i>OT1</i>	Object of type 1
<i>OT2</i>	Object of type 2
<i>DT1</i>	Type 1 unloading area—the location where objects of type 1 are to be relocated.
<i>DT2</i>	Type 2 unloading area—the location where objects of type 2 are to be relocated.

Table 20. System reaction rules for the example of checking whether non-functional requirements are satisfied. A value in the third column is the amount of time required to execute a rule.

Label	Description	ΔT
mov	Moving a robot between warehouse areas.	1
stay	A robot remains in the warehouse area where it is located.	1
get1	A robot retrieves a type 1 object from the warehouse.	2
get2	A robot retrieves a type 2 object from the warehouse.	2
set1	A robot deposits a type 1 object into an unloading area.	2
set2	A robot deposits a type 2 object into an unloading area.	2

The state space for the system consists of 666 states (vertices) and 5325 transitions (arcs). Due to the size of this example, the graphical representation of the state space and elements of the tracking transition system will not be presented. It is worth discussing here the increase in the size of a state space as the number of system elements increases. If one were to expand the current system to three robots, two type 1 objects, and three type 2 objects, the number of states increases to 5765 and the number of transitions to 70,701. Such a significant increase in the size of a system suggests that it is reasonable to consider ways of limited construction of a state space that will remain useful in later stages of the development of behavior policies.

Moving on to behavior policies for the agents in the task above. First walks solving the task are 15 steps long. However, these solutions are using only one robot, as can be observed in the action schedule presented in Figure 12. A mission performed using behavior policy based on such a walk takes 21 units of time.

		Time																					
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
O1		mov	mov	get1	mov	set1	mov	get2	mov	set2	mov	get2	mov	set2	mov	mov							

Figure 12. Schedule of actions for a scenario based on a walk of the length of 15 arcs.

3.3.1. Non-Functional Requirement—Length of a Mission

Now let us assume that one of the non-functional requirements imposed on the task is to limit the length of a mission to the maximum of 20 units of time. There is no walk of the length 15 that satisfies this requirement. Knowing that the current solutions use only one robot we can try to improve them by extending the walks to 18 steps. This way the second robot can move one of the objects to an unloading area. A schedule of actions constructed with a walk of 18 steps is presented in Figure 13.

		Time															
		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
O1		mov	mov	get1	mov	set1	mov	get2	mov	set2	mov	mov					
O2		mov	mov	get2	mov	set2	mov	mov									

Figure 13. A schedule of actions for a scenario created with a walk of the length of 18 arcs.

It is important to note that simply lengthening a walk does not guarantee an improved result. For example, if the walk underlying the schedule in Figure 12 is lengthened by three transition functions, all corresponding to the reaction rule *stay*, it will not yield any improvement in the quality of a solution.

Checking whether non-functional requirements are fulfilled should be done in Phase 1 after Phase 3 has been successfully completed. This step is not shown on Scheme 4 but this is how it was implemented in [39].

3.3.2. Non-Functional Requirement—Collision Avoidance

Another example of a non-functional requirement will be related to safety of mission execution. This time we impose a requirement that there should be no collisions between robots that are in the process of moving objects.

One of the advantages of using bigraphs is that they allow one to define patterns to be found in other bigraphs. These patterns are of “minimal satisfying phenomenon” type. One cannot define an “all but” type pattern in bigraph notation. In other words, you can define a pattern like “minimum three people in a room” but you cannot define a (single) pattern that detects “less than three people in a room”.

Let us assume that a collision-free mission will be guaranteed if the robots moving the objects are not in the same area. Such a requirement can be defined as “if two robots, at least one of which is moving an object, are in the same area then the scenario is unacceptable”. Bigraph patterns able to detect such a situation are shown in Figure 14.

Identical to the previous non-functional requirement, this requirement can be verified in Phase 1 after a successful completion of the Phase 3.

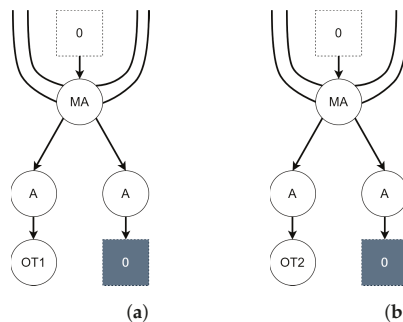


Figure 14. Bigraph patterns to detect whether a collision between robots may occur during a scenario. The two patterns differ only in type of the relocated object. (a) The first pattern. (b) The second pattern.

3.4. Memory Complexity

As we have already mentioned, the size of a system grows much faster than the number of task elements. The same is true for the memory complexity of matrix multiplication operations described in Section 2.4. We have tested how limiting the number of results of convolution operation affects memory usage of the tool [40]. All measurements were done using multi-threaded F# implementation on a PC with 64 bit Ubuntu 20.04 operating system installed and the previous example regarding non-functional requirements was used for testing. We carried out three different tests reducing the number of results to 500, 10,000, and leaving the number of results unlimited. In the first case, the peak memory usage was about 700 MB before walks of the length of 15 arcs were found. The second case resulted in memory consumption around 15 GB before similar walks were found. The last case did not succeed on a machine with 64GB of RAM.

To deal with the memory complexity, we propose three methods to reduce the number of results:

- *First N*—a result of the convolution operation performed during a matrix multiplication is limited to the first N results. This way of searching for behavior policies is suitable when the first results found satisfy non-functional requirements;
- *Best N*—a result of the convolution operation is constrained to the N best results evaluated using an evaluation function (discussed below). This method of searching for walks is useful when a desired walk should have a certain length;
- *All*—the result of a convolution operation is not constrained in any way. Useful only for small systems to verify model correctness.

In the case of *best N* method, there is a need is to define an evaluation function for partial solutions. We propose a SAT configuration evaluation function based on the involvement of task objects. The evaluation function returns a higher score the more objects are involved equally. The formula for calculating the evaluation function value can be expressed as below:

$$E(i) = m = 0, \forall_{(o_{id}, t) \in i} m = m + \frac{t}{t_{max}} \quad i \in I$$

t_{max} — — The largest engagement of any object.

Table 21 shows the values of the proposed evaluation function for a few example SAT configurations.

Table 21. Partial solution evaluation function values for random SAT configurations.

i	t_{max}	$E(i)$
$((1, 2), (2, 2), (3, 2))$	2	3
$((1, 6), (2, 0), (3, 0))$	6	1
$((1, 2), (2, 4), (3, 0))$	4	1.5
$((1, 1), (2, 1), (3, 4))$	4	1.5
$((1, 3), (2, 2), (3, 1))$	3	2
$((1, 1), (2, 1), (3, 0))$	1	2

The prepared tool [40] for walk construction offers six strategies for finding solutions:

- *All first found*—Returns all walks leading to the goal state with the shortest length;
- *First N found*—returns all walks leading to the target state. The matrix multiplication operation is constrained by *first N* method;
- *First N best found*—returns all walks leading to the target state. The matrix multiplication operation is constrained by *best N* method;
- *All up to a certain length*—returns all walks leading to the target state of a length no greater than a given value;
- *First N up to a certain length*—returns all walks leading to the target state with a length no greater than a given value. The matrix multiplication operation to find walks in a state space is constrained by *first N* method which results in each set of walks of the same length being allowed to have a count of at most N elements;
- *Best N up to a certain length*—returns all walks leading to the target state with a length no greater than a given value. The matrix multiplication operation to find walks in a state space is constrained by *best N* method which results in each set of walks of the same length being allowed to have a count of at most N elements.

We summarized all of the above strategies in Table 22.

Table 22. Summary of the proposed strategies for finding walks in a state space. The second column denoted as MNoR stands for Maximum Number of Results. The value of N is equal to a number of results of the same length. The value of L is equal to the maximum length of a result.

Strategy	MNoR	Pros	Cons
All first found	Unlimited	Perfect for assuring correctness of the model as this strategy gives all existing walks to the desired destination state.	Unfeasible for anything but small systems due to large memory consumption.
First N found	N	The fastest of all strategies since it does not sort results and can shrink an output of convolution operation. Perfect when the quality of a result is not important or when all results are expected to have similar quality.	Does not care about quality of returned results at all.
First N best found	N	With a good evaluation function this strategy can return the best results. Perfect when model has already been validated and the developer is looking for a behavior policy of a certain quality.	Slower than <i>first N found</i> since results are sorted with an evaluation function.
All up to a certain length	Unlimited	Gives a glimpse of how the length of a walk impacts the way a mission is executed. Since it is an extension of <i>all first found</i> it allows for throughout correctness testing.	Only for tiny systems. This is the most memory consuming strategy because it not only returns all found results but the search is continued until results have specified length.
First N up to a certain length	$N \times L$	Allow for insight into how the length of a result impacts the way a mission is executed. Very fast as it is an extension of <i>first N found</i> .	Does not care about the quality of returned results at all.
Best N up to a certain length	$N \times L$	It gives good insight how the quality of results varies with the length of a walk. Perfect when the developer is looking for a behavior policy that he or she has no expectations about.	It is slower than <i>first N found up to a certain length</i> strategy due to sorting of results.

4. Discussion

In this paper, we presented an algorithm to verify multi-agent system models based on tracking bigraphical reactive systems. Our algorithm can detect incorrectness of a model and unfulfillment of non-functional requirements. The algorithm considers a model to be incorrect if activities planned to be executed in parallel are not independent of each other. In this article, we presented two examples of utilizing the algorithm to check if a behavior policy meets non-functional requirements regarding time and safety of task execution. We also demonstrated how to generate successive states of a scenario, which is a task realization using a selected behavior policy, based on the the behavior policy. Finally, we discussed memory complexity of operations essential to behavior policies generation and proposed a few ways to reduce it. One of the suggested methods is to limit results to a certain number of the best ones. We gave an example of an evaluation function that allows ranking partial results (in our case, these are behavior policies that when executed do not meet functional requirements). The evaluation function is applicable to tasks of any kind and size.

This work complements our previous publication, which focused solely on designing multi-agent systems with tracking bigraphs. The methodology enables the design of a broad range of systems from warehouse robots to drone swarms performing a task without human intervention. One can also consider designing software systems where programs act as agents and operations performed by these programs could represent transition functions. The functional programming paradigm intuitively fits this kind of design.

The main drawback of our methodology is the lack of adaptability of behavior policies. This means there can be no deviation from scheduled actions when executing a behavior policy. It also means that agents in a modeled system have to be fully controllable in the real world. The biggest drawback of the algorithm presented in this article is that it verifies the correctness of a model looking for errors in a single behavior policy. Thus, the more behavior policies that are checked, the more confident we are that the model is correct.

As for directions of further development, the primary goal should be to improve the generation speed of tracking reactive systems as it is the main limitation of the methodology right now. One way to achieve it is to develop a method of partial construction of a tracking bigraphical reactive system that consists of bigraphs necessary to manufacture a good quality walk in state space. If the method of reducing the number of states is automatic, i.e., it will not require the designer to specify bigraphical patterns, it is going to significantly speed up the development of behavior policies. Right now our method can only be applied to relatively small systems because the explosion of states makes it impossible to efficiently search for walks in the state space of a modeled system.

Author Contributions: Conceptualization, P.C. and Z.Z.; methodology, P.C. and Z.Z.; software, P.C.; validation, P.C.; formal analysis, P.C.; investigation, P.C.; resources, P.C.; data curation, P.C.; writing—original draft preparation, P.C.; writing—review and editing, Z.Z.; visualization, P.C.; supervision, Z.Z.; project administration, Z.Z. Both authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Dorri, A.; Kanhere, S.S.; Jurdak, R. Multi-Agent Systems: A Survey. *IEEE Access* **2018**, *6*, 28573–28593. [[CrossRef](#)]
2. Falco, M.; Robiolo, G. A Systematic Literature Review in Multi-Agent Systems: Patterns and Trends. In Proceedings of the 2019 XLV Latin American Computing Conference (CLEI), Panama City, Panama, 30 September–4 October 2019; pp. 1–10. [[CrossRef](#)]
3. Canese, L.; Cardarilli, G.C.; Di Nunzio, L.; Fazzolari, R.; Giardino, D.; Re, M.; Spanò, S. Multi-Agent Reinforcement Learning: A Review of Challenges and Applications. *Appl. Sci.* **2021**, *11*, 4948. [[CrossRef](#)]
4. Busoni, L.; Babuska, R.; De Schutter, B. A Comprehensive Survey of Multiagent Reinforcement Learning. *IEEE Trans. Syst. Man Cybern. Part C (Appl. Rev.)* **2008**, *38*, 156–172. [[CrossRef](#)]
5. Macal, C.M.; North, M.J. Tutorial on Agent-Based Modeling and Simulation. In Proceedings of the 37th Conference on Winter Simulation. Winter Simulation Conference, Orlando, FL, USA, 4–7 December 2005; pp. 2–15.
6. Weyns, D.; Holvoet, T. A Formal Model for Situated Multi-Agent Systems. *Fundam. Inf.* **2004**, *63*, 125–158.
7. Herrera, M.; Pérez-Hernández, M.; Kumar Parlikad, A.; Izquierdo, J. Multi-Agent Systems and Complex Networks: Review and Applications in Systems Engineering. *Processes* **2020**, *8*, 312. [[CrossRef](#)]
8. Ota, J. Multi-agent robot systems as distributed autonomous systems. *Adv. Eng. Inform.* **2006**, *20*, 59–70. [[CrossRef](#)]
9. Yan, Z.; Jouandeau, N.; Cherif, A.A. A Survey and Analysis of Multi-Robot Coordination. *Int. J. Adv. Robot. Syst.* **2013**, *10*, 399. [[CrossRef](#)]
10. Iñigo-Blasco, P.; del Rio, F.D.; Romero-Ternero, M.C.; Cagigas-Muñoz, D.; Vicente-Díaz, S. Robotics software frameworks for multi-agent robotic systems development. *Robot. Auton. Syst.* **2012**, *60*, 803–821. [[CrossRef](#)]
11. Geihs, K. Engineering Challenges Ahead for Robot Teamwork in Dynamic Environments. *Appl. Sci.* **2020**, *10*, 1368. [[CrossRef](#)]
12. Bullo, F.; Cortés, J.; Martínez, S. *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*; Princeton University Press: Princeton, NJ, USA, 2009.

13. Yang, Z.; Zhang, Q.; Chen, Z. A novel adaptive flocking algorithm for multi-agents system with time delay and nonlinear dynamics. In Proceedings of the 32nd Chinese Control Conference, Xi'an, China, 26–28 July 2013; pp. 998–1001.
14. Sadik, A.R.; Urban, B. An Ontology-Based Approach to Enable Knowledge Representation and Reasoning in Worker-Cobot Agile Manufacturing. *Future Internet* **2017**, *9*, 90. [[CrossRef](#)]
15. Viseras, A.; Xu, Z.; Merino, L. Distributed Multi-Robot Information Gathering under Spatio-Temporal Inter-Robot Constraints. *Sensors* **2020**, *20*, 484. [[CrossRef](#)]
16. Siefke, L.; Sommer, V.; Wudka, B.; Thomas, C. Robotic Systems of Systems Based on a Decentralized Service-Oriented Architecture. *Robotics* **2020**, *9*, 78. [[CrossRef](#)]
17. Pal, C.V.; Leon, F.; Paprzycki, M.; Ganzha, M. A Review of Platforms for the Development of Agent Systems. *arXiv* **2020**, arXiv:2007.08961.
18. Jamroga, W.; Penczek, W. Specification and Verification of Multi-Agent Systems. In *Lectures on Logic and Computation: ESSLLI 2010 Copenhagen, Denmark, August 2010, ESSLLI 2011, Ljubljana, Slovenia, August 2011, Selected Lecture Notes*; Springer: Berlin/Heidelberg, Germany, 2012; pp. 210–263. [[CrossRef](#)]
19. Blanes, D.; Insfran, E.; Abrahão, S. RE4Gaia: A Requirements Modeling Approach for the Development of Multi-Agent Systems. In *Advances in Software Engineering*; Ślęzak, D., Kim, T., Kiumi, A., Jiang, T., Verner, J., Abrahão, S., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 245–252.
20. Bresciani, P.; Giorgini, P.; Giunchiglia, F.; Mylopoulos, J.; Perini, A. Tropos: An Agent-Oriented Software Development Methodology. *Auton. Agents-Multi-Agent Syst.* **2004**, *8*, 203–236. [[CrossRef](#)]
21. Jamont, J.P.; Raievsky, C.; Ocelllo, M. Handling Safety-Related Non-Functional Requirements in Embedded Multi-Agent System Design. In *Advances in Practical Applications of Heterogeneous Multi-Agent Systems. The PAAMS Collection*; Demazeau, Y., Zambonelli, F., Corchado, J.M., Bajo, J., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 159–170.
22. Picard, G.; Gleizes, M.P. The ADELFE Methodology Designing Adaptive Cooperative Multi-Agent Systems. In *Methodologies and Software Engineering for Agent Systems*; Kluwer Publishing: Alphen aan den Rijn, The Netherlands, 2004; Chapter 8, pp. 157–176.
23. Milner, R. *The Space and Motion of Communicating Agents*; Cambridge University Press: Cambridge, UK, 2009; Volume 20. [[CrossRef](#)]
24. Sevegnani, M.; Calder, M. Bigraphs with sharing. *Theor. Comput. Sci.* **2015**, *577*, 43–73. [[CrossRef](#)]
25. Krivine, J.; Milner, R.; Troina, A. Stochastic Bigraphs. *Electron. Notes Theor. Comput. Sci.* **2008**, *218*, 73–96.
26. Gassara, A.; Bouassida, I.; Jmaiel, M. A Tool for Modeling SoS Architectures Using Bigraphs. In *Proceedings of the Symposium on Applied Computing*; Association for Computing Machinery: New York, NY, USA, 2017; pp. 1787–1792. [[CrossRef](#)]
27. Archibald, B.; Shieh, M.Z.; Hu, Y.H.; Sevegnani, M.; Lin, Y.B. BigraphTalk: Verified Design of IoT Applications. *IEEE Internet Things J.* **2020**, *7*, 2955–2967. [[CrossRef](#)]
28. Calder, M.; Kolioussis, A.; Sevegnani, M.; Sventek, J. Real-time verification of wireless home networks using bigraphs with sharing. *Sci. Comput. Program.* **2014**, *80*, 288–310. [[CrossRef](#)]
29. Perrone, G.; Debois, S.; Hildebrandt, T. A model checker for Bigraphs. In Proceedings of the ACM Symposium on Applied Computing, New York, NY, USA, 26–30 March 2012. [[CrossRef](#)]
30. Grzelak, D. Bigraph Framework for Java. 2021. Available online: <https://bigraphs.org/products/bigraph-framework/> (accessed on 16 August 2021).
31. Sevegnani, M.; Calder, M. BigraphER: Rewriting and Analysis Engine for Bigraphs. In *Proceedings of the International Conference on Computer Aided Verification, Los Angeles, CA, USA, 21–24 July 2020*; Springer: Cham, Switzerland, 2016; Volume 9780, pp. 494–501. [[CrossRef](#)]
32. Mansutti, A.; Miculan, M.; Peressotti, M. Multi-agent Systems Design and Prototyping with Bigraphical Reactive Systems. *Distributed Applications and Interoperable Systems*; Magoutis, K., Pietzuch, P., Eds.; Springer: Berlin/Heidelberg, Germany, 2014; pp. 201–208.
33. Taki, A.; Dib, E.; Sahnoun, Z. Formal Specification of Multi-Agent System Architecture. In Proceedings of the ICAASE 2014 International Conference on Advanced Aspects of Software Engineering, Constantine, Algeria, 2–4 November 2014.
34. Pereira, E.; Potiron, C.; Kirsch, C.M.; Sengupta, R. Modeling and controlling the structure of heterogeneous mobile robotic systems: A bigactor approach. In Proceedings of the 2013 IEEE International Systems Conference (SysCon), Orlando, FL, USA, 15–18 April 2013; pp. 442–447. [[CrossRef](#)]
35. Agha, G. *Actors: A Model of Concurrent Computation in Distributed Systems*; MIT Press: Cambridge, MA, USA, 1986.
36. Cybulski, P.; Zieliński, Z. UAV Swarms Behavior Modeling Using Tracking Bigraphical Reactive Systems. *Sensors* **2021**, *21*, 622. [[CrossRef](#)] [[PubMed](#)]
37. Mermoud, G.; Upadhyay, U.; Evans, W.C.; Martinoli, A. Top-Down vs. Bottom-Up Model-Based Methodologies for Distributed Control: A Comparative Experimental Study. In *Experimental Robotics: The 12th International Symposium on Experimental Robotics*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 615–629. [[CrossRef](#)]
38. Amato, F.; Moscato, F.; Moscato, V.; Pascale, F.; Picariello, A. An agent-based approach for recommending cultural tours. *Pattern Recognit. Lett.* **2020**, *131*, 341–347. [[CrossRef](#)]
39. Cybulski, P. Verification Tool for TRS-SSP Toolchain. (trs-ssp-verify). Available online: <https://github.com/zajer/trs-ssp-verify> (accessed on 16 August 2021).

40. Cybulski, P. A Tool for Generating Walks in State Space of a TRS-Based Systems. 2021. Available online: <https://github.com/zajer/trs-ssp> (accessed on 16 August 2021).
41. Cybulski, P. Visualization Tool for TRS-SSP Toolchain. (trs-ssp-frontend). Available online: <https://github.com/zajer/trs-ssp-frontend> (accessed on 16 August 2021).

Review

UX in AR-Supported Industrial Human–Robot Collaborative Tasks: A Systematic Review

Riccardo Karim Khamaisi ¹, Elisa Prati ¹, Margherita Peruzzini ^{1,*}, Roberto Raffaelli ² and Marcello Pellicciari ²

¹ Department of Engineering “Enzo Ferrari”, University of Modena and Reggio Emilia, 41125 Modena, Italy; rcrkhamaisi@unimore.it (R.K.K.); elisa.prati@unimore.it (E.P.)

² Department of Sciences and Methods for Engineering, University of Modena and Reggio Emilia, 42122 Reggio Emilia, Italy; roberto.raffaelli@unimore.it (R.R.); marcello.pellicciari@unimore.it (M.P.)

* Correspondence: margherita.peruzzini@unimore.it

Abstract: The fourth industrial revolution is promoting the Operator 4.0 paradigm, originating from a renovated attention towards human factors, growingly involved in the design of modern, human-centered processes. New technologies, such as augmented reality or collaborative robotics are thus increasingly studied and progressively applied to solve the modern operators’ needs. Human-centered design approaches can help to identify user’s needs and functional requirements, solving usability issues, or reducing cognitive or physical stress. The paper reviews the recent literature on augmented reality-supported collaborative robotics from a human-centered perspective. To this end, the study analyzed 21 papers selected after a quality assessment procedure and remarks the poor adoption of user-centered approaches and methodologies to drive the development of human-centered augmented reality applications to promote an efficient collaboration between humans and robots. To remedy this deficiency, the paper ultimately proposes a structured framework driven by User eXperience approaches to design augmented reality interfaces by encompassing previous research works. Future developments are discussed, stimulating fruitful reflections and a decisive standardization process.

Keywords: User eXperience; human–robot interaction; human–robot collaboration; human-centered design; augmented reality; human factors

Citation: Khamaisi, R.K.; Prati, E.; Peruzzini, M.; Raffaelli, R.; Pellicciari, M. UX in AR-Supported Industrial Human–Robot Collaborative Tasks: A Systematic Review. *Appl. Sci.* **2021**, *11*, 10448. <https://doi.org/10.3390/app112110448>

Academic Editors: António Paulo Moreira, Pedro Neto and Félix Vidal

Received: 12 October 2021
Accepted: 5 November 2021
Published: 7 November 2021

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The creation of intelligent, assisted, and automated machines is characterizing the modern factory aiming at two main aspects: a more conscious distribution of roles between machines and humans, and a more flexible process control to achieve an efficient and optimized production. In this context, high standards of quality, production flexibility, and innovation push towards human-centered design (HCD) approaches, focused on the centrality of the human factors (HF). HF refers to environmental, organizational, and job-related aspects, as well as human individual characteristics, which can highly affect health and safety during the interaction with current technologies. Introducing HF in the design process is the scope of HCD, which is defined as “an approach to systems design and development that aims to make interactive systems more usable by focusing on the use of the system and applying human factors/ergonomics and usability knowledge and techniques” [1]. Today, HCD can be generically used for any type of applications to guarantee the satisfaction of user needs and the coherence with the ergonomics principles while designing any type of human–system interaction. HCD enables new ways to define requirements and recommendations to properly design complex systems according to a user-oriented approach. The final goal is to guarantee a valuable User eXperience (UX), which involves “the user’s perceptions and responses that result from the use and/or anticipated use of a system product or service” [1], including usability in terms of “the achievement of specified goals with effectiveness,

efficiency and satisfaction in a specified context of use", but also considering users' emotions and affections [2].

The current frameworks related to the application of HF and HCD in system design need to be further developed with the advent of Operator 4.0 (O4.0) concept, framing a smart and skilled operator performing highly specialized tasks aided by emerging technologies as and if needed [3], in order to reshape the industrial tasks based on the human-machine partnership and to renovate the industrial systems according to Industry 4.0 paradigm. Indeed, the O4.0 idea is introducing new assistive technologies, such as augmented reality (AR), virtual reality (VR), or mixed reality (MR) in modern industries, making them enabling technologies for the design and development of an effective human-machine cooperation. However, to achieve such challenging objectives, technologies must be centered on the figure of the modern Operator 4.0 according to new framework, able to focus on the interface design for collaborative tasks, involving humans and robots. Primarily, a precise distinction among such technologies can be summarized as follows:

- Augmented reality, as defined by Azuma et al., "supplements the real world with virtual (computer-generated) objects that appear to coexist in the same space as the real world" [4];
- Virtual reality implies a full immersion into a fictitious and digitally generated world which shuts out completely the physical world [5];
- Mixed reality combines both the previous technologies while enabling a strict interaction between the digital and physical world. Thus, the user interaction with the computer-generated environment provides feedbacks and vice versa [6].

Secondly, attention has to be paid to the technological development of modern companies, where novel forms of support and training can be introduced to enrich the operator's knowledge and encouraging the proper use of new, emerging tools, such as robots [7]. Considering all that, the proper design of AR and VR interfaces becomes crucial to promote the new-born paradigm of the Operator 4.0. In order to achieve higher task precision and market responsiveness, industrial collaborative robots and AR devices are gradually entering the shop floor level to assist operators [8]. Contextually, designing a collaborative working environment for O4.0 requires the adoption of the HCD approach in order to consider the O4.0 know-how and know-to-cooperate: the former refers to human capability to run the process, whilst the latter deals with their attitude to cooperate with other agents [9]. Hence, agents' intentions, action's adaptability, and safety concerns are steadily part of human-robot interaction (HRI). The latter is a field of study concentrated on the design of robotic systems for use by or with humans which seeks to improve the human-machine collaboration while developing innovative and usable user interfaces. Finally, the analysis of the state of the art highlights the need for defining a new HCD framework tackling the new O4.0 requirements to improve the design of AR interfaces, according to UX interface design, but applied them specifically for HRI scopes. Therefore, this review also proposes a framework to design AR interfaces as a natural outcome of this review work, due to the lack in the existing literature.

The review moves from the analysis of the different levels of HRI, namely coexistence, cooperation, and collaboration [10]. Coexistence refers to humans and robots sharing common workspace and time, but using different resources. Cooperation is characterized by a common workspace, time, and shared aim, with sequential or simultaneous tasks, on the same resources, but does not involve a direct contact between humans and robots. Finally, collaboration is the highest level of interaction that envisages common workspace, time, and shared aim, with sequential or simultaneous tasks on the same resources, involving a direct physical contact between humans and robots.

This distinction demonstrates how specific UX issues can be identified for each level of HRI. Thus, as shown in Figure 1 and according to the distinction just made, technologies such as AR could be selectively used to support specific targeted tasks of the human-robot interaction. Moreover, regardless of HRI nature, in [10] the authors suggest to integrate a human-centered view to the robot-centered and robot cognition-centered views, meaning

to harmonize the HF and human–machine interaction principles with technological and decisional capability aspects. Based on [11], it could be stated that:

- The human-centered view is primarily concerned “with how a robot can fulfil its task specification in a manner that is acceptable and comfortable to humans”;
- The robot-centered view “emphasizes the view of a robot as a creature, i.e., an autonomous entity that is pursuing its own goals based on its motivations, drives and emotions, whereby interaction with people serves to fulfil some of its ‘needs’”;
- The robot-cognition view considers “the robot as an intelligent system (in a traditional AI sense), i.e., a machine that makes decisions on its own and solves problems it faces as part of the tasks it needs to perform in a particular application domain.”.

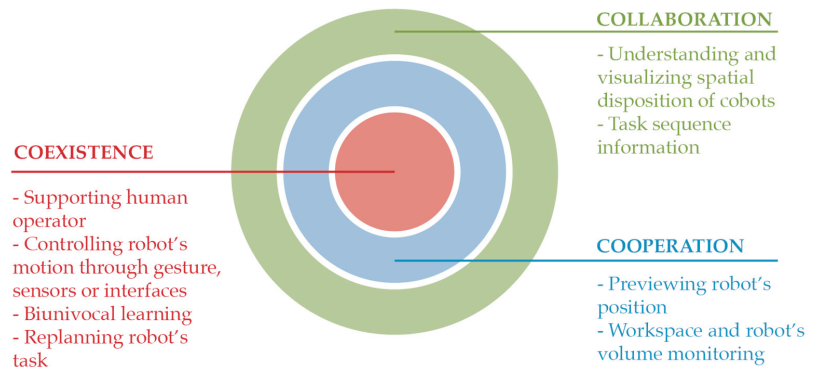


Figure 1. Main AR applications areas according to the three levels of HRI.

Such considerations are valid even if multiple humans and robots are involved in the interaction. All these terminological and conceptual distinctions demonstrate the intrinsic complexity of a HRI task and the need of a structured approach to appropriately encompass all its peculiarities.

However, in practice, collaborative robots (i.e., “cobots”) are currently regulated by the ISO 10218 technical specification document [12], providing a precise interpretation of their roles and natures, and defining the safety requirements for industrial collaborative systems. The main focus is actually for safety as addressed by ISO 15066 [13], neglecting other human implications. We could say that the focus is on the robot-centered view, and only marginally on the robot cognition-centered view where the human-centered aspects are only considered for safety purposes.

In this context, the use of AR technologies as task support tools force us to pay attention to the human-centered view by supporting HRI at different levels (i.e., coexistence, cooperation, and collaboration), thanks to the creation of specific, contextual human knowledge on the ongoing process, and the promotion of know-how development and know-to-cooperate abilities [9]. Indeed, an effective AR interface should define their contents according to the level of interaction realized between the operator and the robots, since each level is characterized by a different form of interaction that requires specific features in the AR interface to properly support the tasks. Indeed, AR introduces digitized information into the real working environment to augment the UX, promoting system usability and object visibility, while reducing the operator’s physical and cognitive workload. Current AR applications are provided mainly through tablets since head-mounted display are still far from being industrially reliable, especially as for ergonomic aspects. The state of art in literature regarding AR-supported HRI highlighted different application areas: visualizing robot actions or faults to support troubleshooting [14]; controlling robots combining head and eye gaze; visual simultaneous localization and mapping algorithms [15]; understanding the impact of AR cues on human attention [16]; supporting human–robot

collaborative assembly [17]; providing workspace and robot's volume monitoring [8,18]; improving interaction efficiency by reducing the physical strength (especially in heavy-duty industries) or letting older people to continue working in production facilities [19]; and by helping operators to have an immediate comprehension of the robot intentions in a quick and intuitive way (e.g., making visible the robot's planned motion and task state) [20,21] or adapting the AR contents to the specific environmental or task conditions [22,23].

However, the majority of existing AR solutions looks at technology and robots, while neglecting the human aspects [10]. The main problem in AR-supported HRI is the lack of user friendly and intuitive interfaces implemented in accordance with the interaction design principles to guide users. While there are some attempts to design user-centered AR interfaces for different applications [24–28], for robotics applications, AR interfaces are usually developed by technology experts and not by UX designers. As a result, interfaces are technology-driven and not user-driven and they usually appear not fully centered on the users' perspective [10].

Only recently, a limited number of papers focused on the need to apply structured HCD methods to the design of HRI, focusing on the understanding and satisfaction of human needs. For instance, a UX-oriented methodology has been recently defined to investigate the human–robot dialogue and map the interaction with robots in performing shared tasks, eliciting the requirements for a valuable HRI design [10]. Similarly, another study has considered the role and relevance of UX in HRI and defined the actual trends concerning the inclusion of UX related to socially interactive robots [29]. Another work also proposes an innovative user-centered design tool to design AR platforms for maintenance operations [30]. Despite this, they did not specifically focus on the design of AR interfaces to support human–robot collaborative tasks, where a limited attention is paid to user perception, ergonomics, and usability issues. Contrarily, the nature of AR and the role that such applications can assume in the context of O4.0 requires great attention to human aspects. Interdisciplinary research is also advisable to achieve high-quality HRI.

In this context, the paper provides two main contributions:

1. A systematic review on AR-supported applications for human–robot collaborative tasks in industry, focusing on human aspects. As a result, the reader can understand whether and how UX approaches are currently adopted in the design of AR-supported collaborative solutions, as well as the main benefits and challenges of the application of UX methods in this field;
2. A UX-driven framework to design user-centric AR interfaces for industrial HRI, discussing also the main potential future developments, after having revealed the lack of such structured framework in literature.

2. Methodology

2.1. Systematic Literature Review

A systematic literature review (SLR) approach has been adopted to investigate the literature relevance of HCD and UX-based methodologies applied to HRI in collaborative tasks. Replicability and objectivity have been considered as basic principles in carrying out the research: the review follows the PICOC framework proposed by [18], thanks to its systematicity and completeness. PICOC [31] stands for a list of items to consider in the analysis, respectively *population*, *intervention*, *comparison*, *outcomes* and *context*. It has been chosen to outline the key concepts of the research. For this review, hereafter the considered items:

- *Population* consists of AR-supported industrial collaborative tasks;
- *Intervention* involves the HCD and UX approaches to design AR application for industrial collaborative tasks;
- *Comparison* can be done considering current design approaches and similar set-ups;
- *Outcomes* can be measured in terms of common Key Performance Indicators (KPI) like time to complete the operation, task's cognitive demand or physical workload;
- *Context* includes industrial human–robot applications.

2.2. Research Questions

The goal of the study is to provide a comprehensive overview of how UX has been used in the field of AR-supported collaborative applications for industry. Bearing this in mind, the authors formulated three research questions (Qi) according to the PICOC results:

- Q1: What are the state of the art UX approaches in AR-supported collaborative solutions?
- Q2: What are the main benefits of adopting UX approaches in designing AR-supported collaborative solutions?
- Q3: What are the main challenges in designing AR-supported collaborative solutions?

2.3. Search and Selection Process

The search was conducted on the Scopus database, since it encompasses different digital libraries, such as IEEE or ACM, and provides high quality, indexed papers. The inclusion criteria are:

- Typology: the study considers articles on international journals and papers on conference proceedings, or books;
- Topics: the study contains the keywords “augmented reality” + “human robot interaction” or “human robot collaboration” + “user experience” or “user interface”. The search has been applied to “Title”, “Abstract”, and “Keywords” (TAK) fields. No reference to the “Mixed Reality” term was included since it subsumes both AR and VR;
- Year: the study has not been limited in terms of the publication year.

According to [18], the following exclusion criteria have been defined:

- Language: the paper is not written in English;
- Scope: the paper is out of scope and focuses on different research domain;
- Accessibility: the paper is not available.

Seeing the high specificity of the “user experience” and “user interface” keywords, the initial search returned 27 papers. No secondary documents nor patents were found. No further papers’ selections in terms of field of application of AR-supported collaborative tasks were provided, since the main interest is analyzing the current general sensibility towards the UX approaches.

After the above-mentioned selection process, only 21 papers were admitted. The results from inclusion and exclusion criteria, and the number of papers found at each step, are shown in Table 1.

Table 1. Search and selection results.

Search String	Database	Date	Found
TITLE-ABS-KEY ((augmented AND reality) AND (human AND robot AND interaction OR human AND robot AND collaboration) AND (user AND experience OR user AND interface))	Scopus	30/04/2021	27
Exclusion Criteria			Found
Language			27
Scope			23
Accessibility			21

Afterwards, the selected papers have been evaluated by a structured quality assessment procedure using three quality criteria (QC) similarly to [18]:

- QC1: It reflects the quality of the journal on which the paper is published, where Qi refers to the quartile score, according to Scimago Journal Ranking [32]. A score of 1 was assigned to Q1 journals, 0.5 to Q2, and 0.25 to Q3. If the journal belongs to Q4, or if it does not belong to a specific quartile yet or it is part of a conference proceeding, a “/” is assigned counting as 0;

- QC2: It reflects the relevance of the specific paper. A value of 1 is assigned if the paper specifically has “User Experience”, “User Interface”, or “Human-centered Design” as one or more paper keywords. This choice was made to further understand if the paper was intended to be searchable for UX, HCD, or UI-related topics;
- QC3: It reflects the citation impact. It considers the number of total citations of the paper (c) compared to the maximum number of citations of the most cited paper (mc) among those included in the review. Certainly, this criterion will not be quantitatively relevant for the most recent works, but it helps to understand the most significant works as recognized from the scientific community. As a consequence, a final score ranging from 0 to 1 has been determined for each paper (i) included in the review:

$$QC3(i) = c(i)/mc$$

The final aim is to focus the review attention on papers which match at their best the intentions of the authors and to allow the reader to select the best referenced articles according to such criteria.

3. Review Results

The results of the quality assessment are shown in Table 2, listed from the highest-quality paper to the lowest-quality paper: the final quality score is the direct sum of the results obtained according to the three considered criteria. The table highlights how there is still little attention given to the current topic, which has been tackled starting from the last few years. As for publications in the last year, it must be considered that still-ongoing studies have to be published and papers’ impact in terms of citations need more time to be evaluated. The majority of collected studies are quiet below the half of the maximum allowed quality score, indicating that, according to quality criteria, there is still need of further research in this field. Considering QC2 scores, nearly half of the selected research papers do not include any reference to any of the HCD keywords, remarking the abovementioned lack of a UX sensibility.

As shown in Figure 2, in the last five years the attention towards AR solutions applied to robotics has noticeably increased. It can be related to the introduction on the market of several proprietary software development kits (e.g., ARKit, ARCore) from major vendors in 2017, which stimulated a general enthusiasm in AR applications development, and the contemporary commercialization of the Microsoft Hololens, from late 2016. These two facts pushed the academic and industrial interest on AR topics and potentials. In fact, previous tools did not provide effective augmented–cognitive interaction and lack in proactively supporting operators on receiving only the relevant information at their smart devices from nearby machines [33].

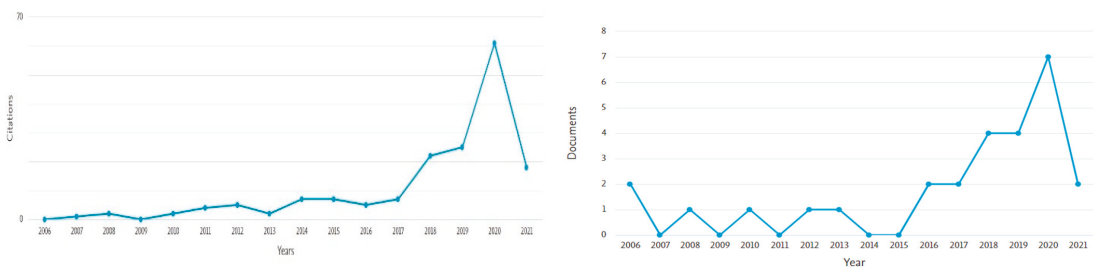


Figure 2. Papers by publication year (on the left) and total citations per year (on the right).

Table 2. Quality assessment on selected papers.

Paper	Year of Publication	Publication Destination	QC1	QC2	QC3	Quality
Hietanen, A., Pieters, R., Lanz, M., Latokartano, J., Kämäräinen, J.-K. [8]	2020	Journal	1	1	0.53	2.53
Papanastasiou, S., Kousi, N., Karagiannis, P., Gkournelos, C., Papavasileiou, A., Dimoulas, K., Baris, K., Koukas, S., Michalos, G., Makris, S. [34]	2019	Journal	1	1	0.53	2.53
De Pace, F., Manuri, F., Sanna, A., Fornaro, C. [18]	2020	Journal	1	1	0	2
Huy, D.Q., Vietcheslav, I., Gerald, S.G.L. [35]	2017	Int. Conference	/	1	0.33	1.33
Materna, Z., Kapinus, M., Beran, V., Smrž, P., Zemčík, P. [36]	2018	Int. Conference	/	1	0.26	1.26
Aschenbrenner, D., Li, M., Dukalski, R., Verlinden, J., Lukosch, S. [37]	2018	Int. Conference	/	1	0.26	1.26
de Tommaso, D., Calinon, S., Caldwell, D.G. [38]	2012	Journal	1	0	0.13	1.13
Bazzano, F., Gentilini, F., Lamberti, F., Sanna, A., Paravati, G., Gatteschi, V., Gaspardone, M. [39]	2016	Journal	/	1	0.13	1.13
Cao, Y., Wang, T., Qian, X., Rao, P.S., Wadhawan, M., Huo, K., Ramani, K. [40]	2019	Int. Conference	/	1	0.1	1.1
Materna, Z., Kapinus, M., Beran, V., Smrž, P., Giuliani, M., Mirmig, N., Stadler, S., Stollnberger, G., Tscheligi, M. [41]	2017	Int. Conference	/	1	0.06	1.06
Kyjánek, O., Al Bahar, B., Vasey, L., Wannemacher, B., Menges, A. [42]	2019	Int. Conference	/	1	0.03	1.03
Leutert, F., Herrmann, C., Schilling, K. [43]	2013	Int. Conference	/	0	1	1
Ji, Z., Liu, Q., Xu, W., Yao, B., Hu, Y., Feng, H., Zhou, Z. [44]	2019	Int. Conference	/	1	0	1
Frank, J.A., Moorhead, M., Kapila, V. [45]	2016	Int. Conference	/	0	0.83	0.83
Green, S.A., Chase, J.G., Chen, X.Q., Billinghamurst, M. [46]	2010	Journal	/	0	0.56	0.56
Jones, B., Zhang, Y., Wong, P.N.Y., Rintel, S. [47]	2020	Int. Conference	/	0	0.03	0.03
Xin, M., Sharlin, E. [48]	2006	Int. Conference	/	0	0.2	0.2
Fuste, A., Reynolds, B., Hobin, J., Heun, V. [49]	2020	Int. Conference	/	0	0	0
Chan, W.P., Hanks, G., Sakr, M., Zuo, T., Machiel Van Der Loos, H.F., Croft, E. [50]	2020	Int. Conference	/	0	0	0
Krauß, M., Leutert, F., Scholz, M.R., Fritscher, M., Heß, R., Lilge, C., Schilling, K. [6]	2021	Journal	/	0	0	0
Diehl, M., Plopski, A., Kato, H., Ramirez-Amaro, K. [51]	2020	Int. Conference	/	0	0	0

Figure 3 depicts the main subject areas dealt by the selected papers. One can infer that the design of AR applications supporting collaborative tasks do not merely involve engineering considerations on technologies or infrastructure's deployment, but also other field of study, such as psychology, neurosciences, and social sciences (i.e., tackling interaction issues from the human point of view or determining the most useful physiological parameters to consider in the evaluation of a specific interface).

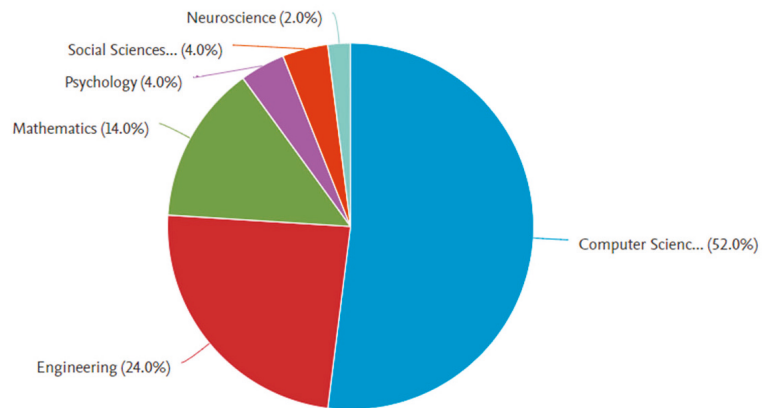


Figure 3. Papers by subject's area.

In conclusion, the relevant works being selected have been carefully considered against the research questions (Q1, Q2, and Q3) as presented in the following sections.

3.1. What Are the State of the Art UX Approaches in AR-Supported Collaborative Solutions?

The current trends in the design of collaborative tasks supported by AR technologies do not systematically show a great attention to UX topics. Hietanen et al. [8] proposed an interactive user interface to assist O4.0 in performing robot-assisted tasks comparing two separate implementations of the same system: a projection-mirror setup, and a wearable device (i.e., Microsoft HoloLens). No prior UX assessment was proposed; as part of the subjective evaluation, a final questionnaire including 13 questions divided into six categories (respectively: safety, information processing, ergonomics, autonomy, competence, and relatedness) was submitted to roughly understand mental and physical stress. Comments from users were collected to deepen the subjective impression, without using any structured method to collect the perceived workload, as used for instance in different contexts. A more structured approach is presented by Papanastasiou et al. [34]: the paper emphasizes the need of a seamless integration between the human operator and his robotic counterpart by monitoring both working entities through sensors and wearable devices. This led to the re-design of the workplace from the human point of view to promote both the robot's operability and operator's mobility, without any barrier to separate them; a multi-stage iterative process has been followed, starting from technical and functional specifications as well as safety requirements. A digital simulation is included for supporting cell setup and risk assessment.

De Pace et al. [18] placed attention on AR devices' usability as enabling tools. The authors reported how usability, workload, and likability can be investigated thanks to the standardized questionnaire (e.g., NASA-TLX [52], System Usability Scale (SUS) [53], AttrakDiff [54]). The same intention is expressed by Huy et al. [35], who introduced a novel AR handheld device inspired by the abovementioned multimodality perceptive interface, incorporating hand gesture mapping, haptic buttons, and laser pointers. The system can suggest available options to the operator and wait for a response instead of traditional inputs by keyboard or mouse; a usability investigation is eventually foreseen to improve the interface effectiveness with the help of user's feedbacks.

Materna et al. [36] evaluated the idea of spatial augmented reality (SAR) through a UX study. The outlined approach works towards avoiding continuous switching of attention during demanding tasks, thanks to a correct distribution of information along the operations and a shared workplace, to be usable also for non-expert users. Process simplification was also addressed by Aschenbrenner et al. [37] to reduce the installation time of hybrid robot-human production lines, and by De Tommaso et al. [38] that defined a

new process of skill transfer between human workers and robots. Similarly, Fuste et al. [49] presented a holistic UX framework (called “Kinetic AR”) for visual programming of robotic motions using AR: the goal was to guarantee a low entry barrier to intricate spatial hardware programming. The UX approach was achieved through interviews to robotic system integrators, manufacturers, and end-users with different expertise, to finally identify the goals and requirements to be accomplished. Communications and interactions were also investigated by Bazzano et al. [39], using 3D immersive simulation to support the design and validation of natural HRI in generic usage contexts, comparing an AR interface and a non-AR one. Among others, subjective observations were gathered through the SASSI methodology [55] to evaluate speech interaction in both interfaces. Information on completion times, overall satisfaction, ease of use, perceived time requested, and support information were collected, and their statistical relevance was given by running an independent sample *t*-test.

As a result of the review, one can state that there are few preliminary attempts to include UX in the design of AR applications for HRI purposes, as summarized in this paragraph, but a reference, ready to use model that is able to integrate the users’ subjective evaluation and the analysis of the quantitative human–robot performance is still missing.

The main weaknesses of the current attempts are:

- User testing is usually based on the collection of deconstructed data regarding device or interface usability, system likability, cognitive and physical workload, or the overall subjective sense of safety in performing the selected operation, without a robust reference model;
- Even if a good attention in using multimodal interfaces to optimize HRI is arising, this trend is not mature enough to enhance human sensorial capabilities by integrating different sensors (e.g., force/torque sensors, microphones, cameras, smartwatches, and AR glasses);
- AR application design does not consider the user perspective and does not help in the improvement of the ease of use of industrial workplaces, avoiding uncomfortable conditions (e.g., extra lightning and noise).

These results highlight the need of a structured framework to design AR interfaces for HRI and pushes towards its definition.

3.2. What Are the Main Benefits of Adopting UX Approaches in Designing AR-Supported Collaborative Solutions?

After the first analysis, the review focused on the analysis of the benefits related to the adoption of UX-based approaches in the design of AR applications for HRI: these approaches generally turn into a detailed evaluative UX phase, where subjective questionnaires represent the main source of information. Table 3 summarizes the most significant papers dealing with such an aspect, also reporting the main areas of applications.

Within the context of laboratory object manipulation tests, Frank et al. [45] focused on the user interaction effectiveness of a mobile augmented interface and on virtual graphics appearing as task’s visual cues to reduce cognitive burden on end-users. The proposed system can automatically intercept an operator’s intention on virtual objects (i.e., drag and drop of models in the space), thus reducing the human involvement while operating with the collaborative companion. No defined UX approach was adopted: a revision of the overall interface was conducted through a final questionnaire after the user-testing phase to identify possible criticalities. A concurrent interface simplification without losing its functionalities in the human–robot collaboration is indeed of extreme importance, in opposition to what has been defined by [42], where high cognitive functionalities are purposely omitted from the proposed interface.

A further critical point in AR-supported collaborative tasks is the choice of the correct interface to use, which is usually conducted without a precise validation tool or methodology. In De Pace et al. [18], a series of interesting UI studies resembling HCD approaches were collected concerning whether exocentric or egocentric interfaces are the best in limit-

ing the level of mental and physical involvement in controlling the manipulator. Another study by Chan et al. [50] reconsidered AR-based interfaces for human–robot collaboration on large-scale labor-intensive manufacturing tasks (carbon-fiber-reinforced-polymer production) where the accent is on the perceived workload and efficiency. Indeed, as stated in other studies [18], the lowest physical and temporal demand is registered with appropriately designed AR solutions, reducing user’s effort and sense of frustration while cutting down operational time. Such an approach does not explicitly make reference to a structured and systematic HCD methodology, but it relies on NASA-TLX questionnaire results. Similar conclusions were reported by Diehl et al. [51], where application circumstances for the choice of best device are examined, starting from users’ feedback on robot’s time and area of manipulation up to user sense of safety.

In Xin et al. [48], a collaborative task concerning playing board games was explored and evaluated by examining various interaction opportunities arising when humans and robots collaborate. This interesting analysis was related to two contrasting robotic behavioral conditions which have been tested: a human-centric condition where robot behavior is more accustomed to human obedience, and a robot-centric one where suggestions coming from the operator are neglected. Statistical results on the final user testing phase related to a custom questionnaire allows for a reinforced idea of the centrality of a human-centric condition to increase the sense of collaboration of O4.0.

Moreover, Palmarini et al. [56] stressed that safety is deemed as one of the most relevant aspects in human–robot collaborative systems and context-awareness information is unavoidably important to enhance user perception. Analogously, Quintero et al. [57] proposed two separate approaches to draw AR paths, respectively, a free space and a surface trajectory one. Such proposals could be effectively integrated to optimize robot’s programming phases with a UX sensibility, reducing programming time, and allowing the worker to selectively inspect different robot trajectories and to work on them in a user-friendly interface. For an optimal collaboration, robot intention is another source of essential information within a HCD approach: a general indifference on the topic emerges from actual selection, although Liu et al. [58] described a temporal and-or graph (T-AOG) to allow the human understanding of the robot’s internal decision-making process, to supervise its action planner, or to monitor its latent states (i.e., forces and moments exerted while interacting).

Table 3. Papers focusing on added value related to adoption of UX approaches in HRI.

Paper	Benefits	Adopted UX Tools	Area of Application
J. A. Frank, M. Moorhead, and V. Kapila [45]	End-user’s intentions understanding to reduce operator cognitive burden	Custom questionnaire	Object manipulation
W. P. Chan, G. Hanks, M. Sakr, T. Zuo, H. F. Machiel Van Der Loos, and E. Croft [50]	The system’s final application must be considered to prevent wrong choices in terms of interfaces and to avoid physical and cognitive repercussion on the user	NASA-TLX	Large-scale, labor-intensive manufacturing tasks
C. P. Quintero, S. Li, M. K. Pan, W. P. Chan, H. F. Machiel Van Der Loos, and E. Croft [57]	Reducing robots’ programming operation time and cognitive demand	Custom questionnaire	Robot programming

As emerged from the review findings, several benefits derived from a UX-based approach when implementing AR-supported collaborative tasks, both objective and subjective: a systematic cognitive and physical relief on the operator, an increased working efficiency, a reduction in operational time and sense of frustration when interacting with shop floor interfaces, and an improved sense of safety and inclusiveness while collaborating with the robotic counterpart. Such conclusions were mainly reported after a user testing campaign in which standard or customized questionnaires were designated to collect final tester impressions to be subsequently reanalyzed by the papers’ authors.

3.3. What Are the Main Challenges in Designing AR-Supported Collaborative Solutions?

The literary review highlighted that the design, development, and use of AR technologies to improve HRI in industrial contexts is a hot topic from a technological point of view; however, there is a lack of models to deepen the UX and only a limited number of papers have proposed the adoption of UX methods to support the design of AR application in this field, according to user-centered principles. As reported by recent market forecasts, the mixed reality market size (including both augmented and virtual reality technologies) is expected to grow by USD 125.19 billion during 2020–2024 [59], up to USD 1.45 trillion to by 2030 [60]. This rapid growth entails big challenges from both a technical and technologies viewpoint and a human viewpoint. Some issues, just considered so far, need to be investigated and faced: from privacy problems to safety requirements. This means that the design of AR-supported applications in the context of HRI will consider how to manage the robots' and operators' data collected and how to assure the proper privacy and safety levels. Considering current applications [61], one can reflect on both critical success factors and challenges related to future robust industrialization. If compared to industrial software systems, current AR hardware readiness seems to still be far from a mature adoption in industry. Thus, human-centered design methods are required to balance industrial system requirements with human needs and social concerns; in this sense AR is so close to human abilities, also affecting and empowering them. Another challenge is the integration of AR devices within modern manufacturing systems: data exchange to and from the AR application should be compliant with robotics and automation standards to assure a full adoption in industry. In regard to this topic, only few research attempts have been made (e.g., AutomationML [62]) which are still far from the inclusion of AR data.

Moreover, a proper UX evaluation framework for AR-supported collaborative tasks needs to be defined. A first attempt has been made considering UX analysis in the design of HRI applications using a structured approach [10], but not including AR tools. On this topic, the main challenge is to define a systematic and coherent way to interpret data coming from different equipment and returning AR digital contents to the O4.0, in an adaptive and intelligent way, considering the UX, and further enhancing the human physical, sensorial, and cognitive capabilities by means of human cyber–physical system integration [63]. In this direction, a further challenge is promoting socially embedded human–robot collaborations where human communications can be used to adapt service robots to the user needs accordingly: it consists of giving the robots the concept of emotional tuning and to emphatically communicate with machines [64].

Moreover, the estimation of those variables affecting trust in HRI is necessary to design new, effective AR interfaces providing situational awareness and spatial dialog, and to determine functional elements to improve human confidence in robots. This evaluation should be included in a comprehensive approach considering validated metrics for an overall UX assessment [65]. Contextually, the assessment of human cognitive and physical efforts in developing collaborative tasks has an absolute relevance.

Finally, in the context of AR-supported human–robot collaborative operations, user testing needs a more statistically reliable base, including both academic and industrial studies and increasing the number and typology of people involved to assess the effectiveness of AR in HRI tasks [18]. The results mainly imply the definition of new UX-based methods to design AR interfaces from a multiple users' point of view, involving novice and expert users, and the benchmark of the most suitable wearable interfaces to be used together with industrial robots.

4. Discussion on Review Results

From the current analysis, a substantial lack of structured methods to design user-centered AR applications for HRI have emerged. Despite several attempts in other various contexts (tourism, mobile application games, etc. [24–28]), UX-driven methodology are poorly adopted in HRI, and this led to the design of interfaces which are far from real users' needs.

On this base, the authors believe that it is crucial to promote UX-driven design processes to develop successful AR interfaces, especially for collaborative tasks involving humans and robots, to fully support the O4.0. Only the adoption of a proper HCD framework allows considering the real needs of the operators in a specific context of use, focusing not only on safety and ergonomics issues, but on the overall UX aspects (e.g., usability, intuitiveness, satisfaction, cognitive load, and emotional response) with the final aim to have a renovated human–robot relationship where both subjects are actively participating and transmitting knowledge to the equivalent counterpart, according to a win–win approach.

For these purposes, the review results suggest defining a structured UX-driven process to design AR interfaces for HRI tasks. The review represents a nonlinear and iterative process aiming at assisting the AR interface designer in implementing a valuable communication with robots, during the execution of HRI tasks. As depicted in Figure 4, the process must be made up of three main steps:

1. Requirements Gathering;
2. AR Interface Design and Prototyping;
3. UX Assessment.

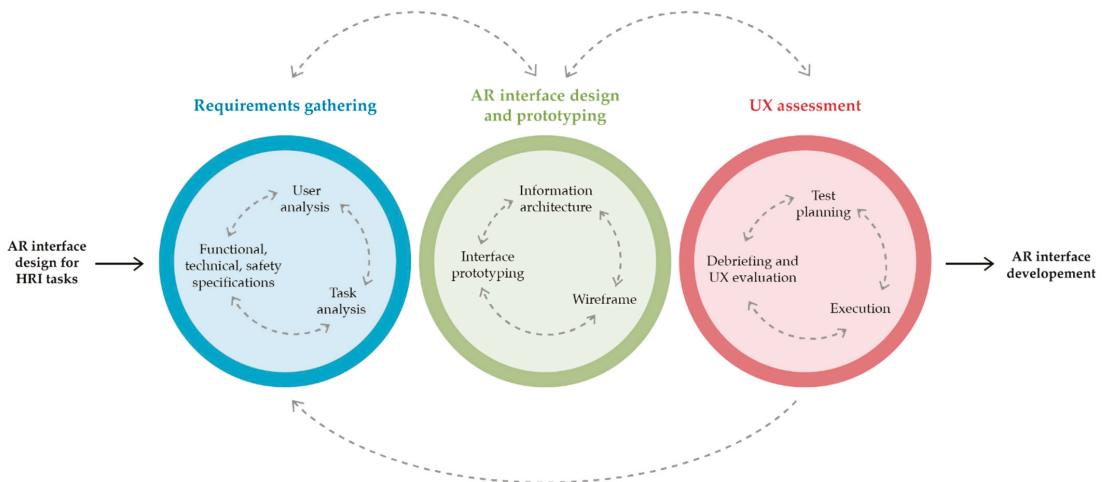


Figure 4. The need of a UX-driven process for AR interface design to support HRI tasks.

The process starts with the need to define a new AR interface for collaborative tasks, and brings to the interface definition, before the AR interface development. Such a process goes a step further with respect to a few recent studies [10,34].

The first step to be investigated consists of requirement gathering from user research. It is a vital part of any UX-driven processes because it is the act of understanding the users and their needs, making sense of who the user is, what he/she wants, and how he/she will perform a certain task. It mainly consists of a deep, accurate user analysis based on the context research to be carried out in a not invasive way using a set of UX design techniques. The most suitable techniques for user understanding are user observation, focus groups, and interviews [66]. User observation consists of observing users in their natural environment without affecting their normal behaviors and performance, with the aim to understand the users’ needs and widely used when users belong to specific categories. It can also be done using video analysis, but in some industrial contexts video recording is not always possible. Such analysis is frequently combined with focus groups or interviews, which can provide a deeper insight about the users’ habits and behaviors since they actively involve users to collect, in different ways, qualitative data about their

needs, expectations, or fears. The mixed approach combining different techniques is very useful since it allows combining quantitative and qualitative data. After that, the list of executed tasks can be easily defined, reporting also the actors involved for each task (e.g., humans, robots) and the time span. These actions finally led to the definition of design specifications considering functional, technical, and safety aspects. A different set of UX design techniques can be validly adopted, from user scenarios and personas to experience maps, as proposed by [10]. User scenarios are stories to show how users might act to achieve a goal in a system or environment. They are valuable aids for designers to visualize aspects of their solutions which users might appreciate most in their contexts of use and with their unique needs and motivations. Personas represent the target users by a set of probable users and flesh out their experiences to reflect realistic situations. Finally, experience maps represent a synthetic visualization of an entire end-to-end experience that generic users (i.e., personas) go through to accomplish a certain goal, and they allow a better understanding of human behaviors.

The second aspect to be investigated is the AR interface design and prototyping. Design consists of the definition of the interface functions as well as the items, while prototyping presents the design in a concrete way by representing the interface in action with the simulation of the final interaction between the user and the system. In this context, wireframes are very powerful as a visual representation of the interface pages; clickable wireframes are the simplest form of interactive prototype, created by linking static wireframes together. Moreover, using digital tools, wireframes can be updated and easily reused and layouts can be easily changed based on user feedback to repeat the testing process. Low-fidelity prototypes allow one to easily define the following information architecture, also comparing possible alternative solutions, and to optimize the design itself in an iterative way.

Finally, the third aspect to include consists of the UX assessment. Different evaluation techniques exist to investigate more closely the user's behavior and perception on a final prototype or products. One of the most spread is user testing, that can lead to both quantitative and qualitative data [67]. Quantitative tests carry out measurements (e.g., execution time, number of errors, and number of tasks completed) while performing a specific task on the user interface. User testing sessions often include post-test evaluation questionnaires (e.g., System Usability Scale (SUS) [53], UEQ Questionnaire [68], or mCUE questionnaire [69]) and allow the gathering of many opinions in a short time, as well as being adaptable to multiple application areas. In addition, physiological measurements allow us to investigate in real time the level of physical and cognitive workload as well as stress of the operator during the interaction. Examples of adoption of these tools for the design of modern systems is provided by [70].

Based on the review results, it is also possible to identify some trends of future implementation of a successful UX-driven design process, as depicted above. First of all, the UX assessment needs to be included in the analysis of the real human activity during collaborative task execution, to understand the level of attention and physical and cognitive responsiveness through wearable devices, in order to understand the real UX. Secondly, non-intrusive sensors and smart interfaces are required to carry out a bi-directional communication flow from-to machines and robots and create a synergic collaboration, in order to overcome the current vision based on separate entities with incompatible characteristics. Further, flexible data management is necessary to manage the crescent complexity and the larger amount of data coming from the shop floor and the operator, in order to successfully integrate AR-supported collaborative tasks in the overall productive chain. Finally, it is worth understanding the impact of AR on user perceptions, ergonomics, and human-robot interactions.

5. Conclusions

This paper reviews the overall condition of industrial collaborative tasks supported by AR technologies, pushed by the growing interest of industry in the AR market registered

in the last 5 years and the need to define new ways to make the Operator 4.0 successfully work in the factories of the future, interacting with robots. In this context, AR interfaces can help to improve human–robot communication and interaction at different levels, thanks to the possibility to show contextual and digital information and data when and where needed. However, there is a lack of a proper framework to design AR interface, including the operators’ UX, specifically designed for HRI. The paper starts with a review of the state of the art, focusing on the inclusion of HF and UX design principles in the design of AR interfaces to support HRI. In the paper, a SLR approach was used to collect the most interesting papers in this field, considering the recent scientific literature. After identifying the focus of the current study, 27 papers were gathered and assessed according to proper quality control parameters previously defined. At the end of the selection process, 21 papers were deemed suitable to answer the three research questions: Q1: *What is the state of art related to UX approaches in AR-supported collaborative solutions?*; Q2: *What are the main benefits of adopting UX approaches in designing AR-supported collaborative solutions?*; Q3: *What are the main challenges in designing AR-supported collaborative solutions?*

As a result, the research highlighted the lack of reliable and systematic user-centered methodologies to design AR applications for human–robot collaborative tasks. This fact is limiting the acceptance of such solutions and slowing down the technological integration of smart devices within the Operator 4.0 paradigm. Several added values of AR application to Operator 4.0 scenario are then presented, starting from the reduction of the worker’s cognitive workload thanks to the interface simplification and adequate usability tests, up to the realization of a shared workplace where a synergic collaboration could take place, in which both actors can reciprocally be understood and learn from the corresponding counterpart. From the discussion of the review results, the paper finally highlights the need of structured UX-driven processes to design successful AR interfaces for human–robot collaborative tasks, made up of different phases organized in an iterative cycle, including typical UX design tools and techniques for interface design, that are not currently used in AR interface design for industrial purposes. The research also defined the main trends of development for future applications, considering the need of non-intrusive human monitoring devices and smart tools to enable fruitful communication between operators and the on-going process at the shop floor.

Author Contributions: Conceptualization, M.P. (Margherita Peruzzini) and R.K.K.; methodology, M.P. (Margherita Peruzzini) and E.P.; formal analysis, R.R.; data curation, R.K.K.; writing—original draft preparation, R.K.K. and M.P. (Margherita Peruzzini); writing—review and editing, M.P. (Marcello Pellicciari) and R.R.; supervision, M.P. (Marcello Pellicciari). All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: All available data is contained within the paper.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. UNI EN ISO 9241-210:2019. *Ergonomics of Human-System Interaction—Part 210: Human-Centred Design for Interactive Systems*; ISO: Geneva, Switzerland, 2019.
2. UNI EN ISO 9241-11:2018. *Ergonomics of Human-System Interaction—Part 11: Usability: Definitions and Concepts*; ISO: Geneva, Switzerland, 2018.
3. Romero, D.; Stahre, J.; Wuest, T.; Noran, O.; Bernus, P.; Fasth, A.; Gorecky, D. Towards an operator 4.0 typology: A human-centric perspective on the fourth industrial revolution technologies. In *Proceedings of the International Conference on Computers & Industrial Engineering (CIE46)*, Tianjin, China, 29–31 October 2017; pp. 1–11, ISSN 2164-8670 CD-ROM, ISSN 2164-8689 ON-LINE.
4. Azuma, R.; Baillot, Y.; Behringer, R.; Feiner, S.; Julier, S.; MacIntyre, B. Recent advances in augmented reality. *IEEE Comput. Graph. Appl.* **2001**, *21*, 34–47. [[CrossRef](#)]

5. Peruzzini, M.; Grandi, F.; Cavallaro, S.; Pellicciari, M. Using virtual manufacturing to design human-centric factories: An industrial case. *Int. J. Adv. Manuf. Technol.* **2020**, *115*, 873–887. [[CrossRef](#)]
6. Milgram, P. A Taxonomy of Mixed Reality Visual Displays. *IEICE Trans. Inf. Syst.* **1994**, *77*, 1321–1329.
7. Krauß, M.; Leutert, F.; Scholz, M.R.; Fritscher, M.; Heß, R.; Lilge, C.; Schilling, K. Digital Manufacturing for Smart Small Satellites Systems. *Procedia Comput. Sci.* **2021**, *180*, 150–161. [[CrossRef](#)]
8. Hietanen, A.; Pieters, R.; Lanz, M.; Latokartano, J.; Kämäräinen, J.-K. AR-based interaction for human-robot collaborative manufacturing. *Robot. Comput. Manuf.* **2020**, *63*, 101891. [[CrossRef](#)]
9. Pacaux-Lemoine, M.-P.; Flemisch, F. Layers of Shared and Cooperative Control, assistance and automation. *IFAC-PapersOnLine* **2016**, *49*, 159–164. [[CrossRef](#)]
10. Prati, E.; Peruzzini, M.; Pellicciari, M.; Raffaelli, R. How to include User eXperience in the design of Human-Robot Interaction. *Robot. Comput. Manuf.* **2021**, *68*, 102072. [[CrossRef](#)]
11. Dautenhahn, K. Socially intelligent robots: Dimensions of human–robot interaction. *Philos. Trans. R. Soc. B Biol. Sci.* **2007**, *362*, 679–704. [[CrossRef](#)] [[PubMed](#)]
12. UNI EN ISO 10218-1:2012. *Robot e Attrezzature per Robot—Requisiti di Sicurezza per Robot Industriali—Parte 1: Robot*; ISO: Geneva, Switzerland, 2012.
13. ISO/TS 15066:2016. *Robots and Robotic Devices—Collaborative Robots*; ISO: Geneva, Switzerland, 2016.
14. Avale, G.; De Pace, F.; Fornaro, C.; Manuri, F.; Sanna, A. An Augmented Reality System to Support Fault Visualization in Industrial Robotic Tasks. *IEEE Access* **2019**, *7*, 132343–132359. [[CrossRef](#)]
15. Wöhle, L.; Gebhard, M. Towards Robust Robot Control in Cartesian Space Using an Infrastructureless Head- and Eye-Gaze Interface. *Sensors* **2021**, *21*, 1798. [[CrossRef](#)]
16. Rosen, E.; Whitney, D.; Phillips, E.; Chien, G.; Tompkin, J.; Konidar, G.; Tellex, S. Communicating and controlling robot arm motion intent through mixed-reality head-mounted displays. *Int. J. Robot. Res.* **2019**, *38*, 1513–1526. [[CrossRef](#)]
17. Liu, H.; Wang, L. An AR-based Worker Support System for Human-Robot Collaboration. *Procedia Manuf.* **2017**, *11*, 22–30. [[CrossRef](#)]
18. De Pace, F.; Manuri, F.; Sanna, A.; Fornaro, C. A systematic review of Augmented Reality interfaces for collaborative industrial robots. *Comput. Ind. Eng.* **2020**, *149*, 106806. [[CrossRef](#)]
19. Michalos, G.; Karagiannis, P.; Makris, S.; Tokçalar, Ö.; Chryssolouris, G. Augmented Reality (AR) Applications for Supporting Human-robot Interactive Cooperation. *Procedia CIRP* **2016**, *41*, 370–375. [[CrossRef](#)]
20. Bolano, G.; Juelg, C.; Roennau, A.; Dillmann, R. Transparent Robot Behavior Using Augmented Reality in Close Human-Robot Interaction. In Proceedings of the 2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), New Delhi, India, 14–18 October 2019. [[CrossRef](#)]
21. Andersen, R.S.; Madsen, O.; Moeslund, T.B.; Ben Amor, H. Projecting robot intentions into human environments. In Proceedings of the 2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), New York, NY, USA, 26–31 August 2016; pp. 294–301. [[CrossRef](#)]
22. Geng, J.; Song, X.; Pan, Y.; Tang, J.; Liu, Y.; Zhao, D.; Ma, Y. A systematic design method of adaptive augmented reality work instruction for complex industrial operations. *Comput. Ind.* **2020**, *119*, 103229. [[CrossRef](#)]
23. Grandi, F.; Khamaisi, R.; Peruzzini, M.; Raffaelli, R.; Pellicciari, M. A Reference Framework to Combine Model-Based Design and AR to Improve Social Sustainability. *Sustainability* **2021**, *13*, 2031. [[CrossRef](#)]
24. Chakravorty, A.; Rowe, A. UX design principles for mobile augmented reality applications. In *MCCSIS 2018-Multi Conference on Computer Science and Information Systems, Proceedings of the International Conferences on Interfaces and Human Computer Interaction 2018, Game and Entertainment Technologies 2018 and Computer Graphics, Visualization, Madrid, Spain, 17–20 July 2018*; IADIS Publications: Lisbon, Portugal, 2018; pp. 319–323.
25. Cauchi, M.; Scerri, D. Enriching Tourist UX via a Location Based AR Treasure Hunt Game. In Proceedings of the 2019 IEEE 9th International Conference on Consumer Electronics (ICCE-Berlin), Berlin, Germany, 8–11 September 2019; pp. 199–204.
26. Lankes, M.; Stiglbauer, B. GazeAR: Mobile Gaze-Based Interaction in the Context of Augmented Reality Games. *Adv. Auton. Robot.* **2016**, *9768*, 397–406. [[CrossRef](#)]
27. Law, E.L.-C.; Heintz, M. Augmented reality applications for K-12 education: A systematic review from the usability and user experience perspective. *Int. J. Child-Comput. Interact.* **2021**, *30*, 100321. [[CrossRef](#)]
28. Rusu, C.; Rusu, V.; Roncagliolo, S.; González, C.S. Usability and User Experience. *Int. J. Inf. Technol. Syst. Approach* **2015**, *8*, 1–12. [[CrossRef](#)]
29. Alenljung, B.; Lindblom, J.; Andreasson, R.; Ziemke, T. User Experience in Social Human-Robot Interaction. *Int. J. Ambient. Comput. Intell.* **2017**, *8*, 12–31. [[CrossRef](#)]
30. del Amo, I.F.; Galeotti, E.; Palmarini, R.; Dini, G.; Erkoyuncu, J.A.; Roy, R. An innovative user-centred support tool for Augmented Reality maintenance systems design: A preliminary study. *Procedia CIRP* **2018**, *70*, 362–367. [[CrossRef](#)]
31. Booth, A.; Sutton, A.; Papaioannou, D. *Systematic Approaches to a Successful Literature Review*, 1st ed.; Sage Publications Ltd.: London, UK, 2016; pp. 245–267.
32. Scimago.com. Available online: <https://www.scimagojr.com/> (accessed on 26 May 2021).
33. Romero, D.; Stahre, J.; Taisch, M. The Operator 4.0: Towards socially sustainable factories of the future. *Comput. Ind. Eng.* **2020**, *139*, 106128. [[CrossRef](#)]

34. Papanastasiou, S.; Kousi, N.; Karagiannis, P.; Gkournelos, C.; Papavasileiou, A.; Dimoulas, K.; Baris, K.; Koukas, S.; Michalos, G.; Makris, S. Towards seamless human robot collaboration: Integrating multimodal interaction. *Int. J. Adv. Manuf. Technol.* **2019**, *105*, 3881–3897. [\[CrossRef\]](#)
35. Huy, D.Q.; Vietcheslav, I.; Lee, G.S.G. See-through and spatial augmented reality—A novel framework for human-robot interaction. In Proceedings of the 2017 3rd International Conference on Control, Automation and Robotics (ICCAR), Nagoya, Japan, 24–26 April 2017; pp. 719–726.
36. Materna, Z.; Kapinus, M.; Beran, V.; Smrz, P.; Zemcik, P. Interactive Spatial Augmented Reality in Collaborative Robot Programming: User Experience Evaluation. In Proceedings of the 2018 27th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), Nanjing, China, 27–31 August 2018; pp. 80–87. [\[CrossRef\]](#)
37. Aschenbrenner, D.; Li, M.; Dukalski, R.; Verlinden, J.; Lukosch, S. Collaborative Production Line Planning with Augmented Fabrication. In Proceedings of the 2018 IEEE Conference on Virtual Reality and 3D User Interfaces (VR), Tuebingen/Reutlingen, Germany, 18–22 March 2018; pp. 509–510. [\[CrossRef\]](#)
38. De Tommaso, D.; Calinon, S.; Caldwell, D.G. A Tangible Interface for Transferring Skills. *Int. J. Soc. Robot.* **2012**, *4*, 397–408. [\[CrossRef\]](#)
39. Bazzano, F.; Gentilini, F.; Lamberti, F.; Sanna, A.; Paravati, G.; Gatteschi, V.; Gaspardone, M. Immersive Virtual Reality-Based Simulation to Support the Design of Natural Human-Robot Interfaces for Service Robotic Applications. *Lect. Notes Comput. Sci.* **2016**, *9768*, 33–51. [\[CrossRef\]](#)
40. Cao, Y.; Wang, T.; Qian, X.; Rao, P.S.; Wadhawan, M.; Huo, K.; Ramani, K. GhostAR: A Time-space Editor for Embodied Authoring of Human-Robot Collaborative Task with Augmented Reality. In Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology, New Orleans, LA, USA, 20–23 October 2019; pp. 521–534. [\[CrossRef\]](#)
41. Materna, Z.; Kapinus, M.; Beran, V.; SmrĚ, P.; Giuliani, M.; MirniĚ, N.; Stadler, S.; Stollnberger, G.; Tscheligi, M. Using Persona, Scenario, and Use Case to Develop a Human-Robot Augmented Reality Collaborative Workspace. In Proceedings of the Companion of the 2017 ACM/IEEE International Conference on Human-Robot Interaction, Vienna, Austria, 6–9 March 2017; pp. 201–202. [\[CrossRef\]](#)
42. Kyjanek, O.; Al Bahar, B.; Vasey, L.; Wannemacher, B.; Menges, A. Implementation of an Augmented Reality AR Workflow for Human Robot Collaboration in Timber Prefabrication. In Proceedings of the 36th International Symposium on Automation and Robotics in Construction (ISARC), Banff, AB, Canada, 21–24 May 2019; pp. 1223–1230.
43. Leutert, F.; Herrmann, C.; Schilling, K. A Spatial Augmented Reality system for intuitive display of robotic data. In Proceedings of the 2013 8th ACM/IEEE International Conference on Human-Robot Interaction (HRI), Tokyo, Japan, 3–6 March 2013; pp. 179–180. [\[CrossRef\]](#)
44. Ji, Z.; Liu, Q.; Xu, W.; Yao, B.; Hu, Y.; Feng, H.; Zhou, Z. Augmented reality-enabled intuitive interaction for industrial human-robot collaboration. In *Advanced Human-Robot Collaboration in Manufacturing*; Springer: Cham, Switzerland, 2021; pp. 395–411.
45. Frank, J.A.; Moorhead, M.; Kapila, V. Realizing mixed-reality environments with tablets for intuitive human-robot collaboration for object manipulation tasks. In Proceedings of the 2016 25th IEEE International Symposium on Robot and Human Interactive Communication (RO-MAN), New York, NY, USA, 26–31 August 2016; pp. 302–307. [\[CrossRef\]](#)
46. Green, S.A.; Chase, J.G.; Chen, X.; Billinghamurst, M. Evaluating the augmented reality human-robot collaboration system. *Int. J. Intell. Syst. Technol. Appl.* **2010**, *8*, 130. [\[CrossRef\]](#)
47. Jones, B.; Zhang, Y.; Wong, P.N.Y.; Rintel, S. VROOM: Virtual Robot Overlay for Online Meetings. In Proceedings of the Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, 25–30 April 2020; pp. 1–10. [\[CrossRef\]](#)
48. Xin, M.; Sharlin, E. Sheep and wolves: Test bed for human-robot interaction. In Proceedings of the CHI' 06 Extended Abstracts on Human Factors in Computing Systems, Montréal, QC, Canada, 22–27 April 2006; pp. 1553–1558.
49. Fuste, A.; Reynolds, B.; Hobin, J.; Heun, V.; Ptc, B.A.F. Kinetic AR: A Framework for Robotic Motion Systems in Spatial Computing. In Proceedings of the Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems, Honolulu, HI, USA, 25–30 April 2020; pp. 1–8.
50. Chan, W.P.; Hanks, G.; Sakr, M.; Zuo, T.; Van der Loos, H.M.; Croft, E. An Augmented Reality Human-Robot Physical Collaboration Interface Design for Shared, Large-Scale, Labour-Intensive Manufacturing Tasks. In Proceedings of the 2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Las Vegas, NV, USA, 24 October 2020; pp. 11308–11313. [\[CrossRef\]](#)
51. Diehl, M.; Plopski, A.; Kato, H.; Ramirez-Amaro, K. Augmented Reality interface to verify Robot Learning. In Proceedings of the 2020 29th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN), Naples, Italy, 31 August–4 September 2020; pp. 378–383. [\[CrossRef\]](#)
52. Scafà, M.; Serrani, E.B.; Papetti, A.; Brunzini, A.; Germani, M. Assessment of Students' Cognitive Conditions in Medical Simulation Training: A Review Study. In *Advances in Intelligent Systems and Computing*; Springer International Publishing: Cham, Switzerland, 2017; pp. 224–233.
53. Brooke, J. SUS: A 'Quick and Dirty' Usability Scale. *Usability Eval. Ind.* **1996**, *189*, 4–7.
54. AttrakDiff. Available online: <http://attrakdiff.de/index-en.html> (accessed on 5 July 2021).
55. Hone, K.S.; Graham, R. Towards a tool for the Subjective Assessment of Speech System Interfaces (SASSI). *Nat. Lang. Eng.* **2000**, *6*, 287–303. [\[CrossRef\]](#)

56. Palmarini, R.; Erkoyuncu, J.A.; Roy, R.; Torabmostaedi, H. A systematic review of augmented reality applications in maintenance. *Robot. Comput. Manuf.* **2018**, *49*, 215–228. [[CrossRef](#)]
57. Quintero, C.P.; Li, S.; Pan, M.K.; Chan, W.P.; Van der Loos, H.M.; Croft, E. Robot Programming Through Augmented Trajectories in Augmented Reality. In Proceedings of the 2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Madrid, Spain, 1–5 October 2018; pp. 1838–1844. [[CrossRef](#)]
58. Liu, H.; Zhang, Y.; Si, W.; Xie, X.; Zhu, Y.; Zhu, S.-C. Interactive Robot Knowledge Patching Using Augmented Reality. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 13 September 2018; pp. 1947–1954. [[CrossRef](#)]
59. Bloomberg. Available online: <https://www.bloomberg.com/press-releases/2021--02-02/-125-billion-growth-in-global-augmented-reality-ar-and-virtual-reality-vr-market-2020--2024-apac-to-emerge-as-major-market> (accessed on 2 July 2021).
60. Why We Believe VR/AR will Boost Global GDP by \$1.5 Trillion. Available online: <https://www.pwc.co.uk/services/economics/insights/vr-ar-to-boost-global-gdp.html> (accessed on 2 July 2021).
61. Masood, T.; Egger, J. Augmented reality in support of Industry 4.0—Implementation challenges and success factors. *Robot. Comput. Manuf.* **2019**, *58*, 181–195. [[CrossRef](#)]
62. Gonçalves, E.M.N.; Freitas, A.; Botelho, S. An AutomationML Based Ontology for Sensor Fusion in Industrial Plants. *Sensors* **2019**, *19*, 1311. [[CrossRef](#)]
63. Romero, D.; Bernus, P.; Noran, O.; Stahre, J.; Berglund, Å.F. The operator 4.0: Human cyber-physical systems & adaptive automation towards human-automation symbiosis work systems. In *IFIP Advances in Information and Communication Technology*; Springer: New York, NY, USA, 2016; Volume 488, pp. 677–686.
64. Woo, J.; Ohyama, Y.; Kubota, N. Robot Partner Development Platform for Human-Robot Interaction Based on a User-Centered Design Approach. *Appl. Sci.* **2020**, *10*, 7992. [[CrossRef](#)]
65. Palmarini, R.; del Amo, I.F.; Bertolino, G.; Dini, G.; Erkoyuncu, J.A.; Roy, R.; Farnsworth, M. Designing an AR interface to improve trust in Human-Robots collaboration. *Procedia CIRP* **2018**, *70*, 350–355. [[CrossRef](#)]
66. Still, B.; Crane, K. *Fundamentals of user-centered design: A practical approach, 1st ed*; CRC Press: Boca Raton, FL, USA, 2017.
67. Prati, E.; Grandi, F.; Peruzzini, M. Usability Testing on Tractor’s HMI: A Study Protocol. In *Lecture Notes in Computer Science*; Springer Science & Business Media: Berlin/Heidelberg, Germany, 2021. [[CrossRef](#)]
68. William, A.; Tullis, T. *Measuring the User Experience: Collecting, Analyzing, and Presenting Usability Metrics*, 2nd ed.; Elsevier Inc.: Amsterdam, The Netherlands, 2013.
69. Minge, M.; Thüring, M.; Wagner, I.; Kuhr, C.V. The meCUE Questionnaire: A Modular Tool for Measuring User Experience. *Adv. Intell. Syst. Comput.* **2017**, *486*, 115–128. [[CrossRef](#)]
70. Peruzzini, M.; Grandi, F.; Pellicciari, M. Benchmarking of Tools for User Experience Analysis in Industry 4.0. *Procedia Manuf.* **2017**, *11*, 806–813. [[CrossRef](#)]

MDPI
St. Alban-Anlage 66
4052 Basel
Switzerland
Tel. +41 61 683 77 34
Fax +41 61 302 89 18
www.mdpi.com

Applied Sciences Editorial Office
E-mail: applsci@mdpi.com
www.mdpi.com/journal/applsci



MDPI
St. Alban-Anlage 66
4052 Basel
Switzerland

Tel: +41 61 683 77 34

www.mdpi.com



ISBN 978-3-0365-6555-2