applied sciences

MDPI

Article

Reinforcement Learning for Autonomous Underwater Vehicles via Data-Informed Domain Randomization

Wenjie Lu [1,*], Kai Cheng [1] and Manman Hu [2,*]

[1] School of Mechanical Engineering and Automation, Harbin Institute of Technology (Shenzhen), Shenzhen 518055, China
[2] Department of Civil Engineering, University of Hong Kong, Hong Kong, China
* Correspondence: luwenjie@hit.edu.cn (W.L.); mmhu@hku.hk (M.H.); Tel.: +86-15168551455 (W.L.)

Featured Application: The proposed RL via data-informed Domain Randomization (DDR) is designed to stabilize autonomous underwater vehicles and the platforms of underwater vehicle-manipulator systems, which are further subject to unknown dynamics and varying payloads. The approach captures the differences in dynamics between the simulated AUVs and real AUVs, trains the controller in simulations that are suitable for real AUVs, and avoids the tedious procedures of parameter-tuning. The proposed RL-DDR requires only a few training samples, making the controller adaptation efficient for real-time applications.

Abstract: Autonomous Underwater Vehicles (AUVs) or underwater vehicle-manipulator systems often have large model uncertainties from degenerated or damaged thrusters, varying payloads, disturbances from currents, etc. Other constraints, such as input dead zones and saturations, make the feedback controllers difficult to tune online. Model-free Reinforcement Learning (RL) has been applied to control AUVs, but most results were validated through numerical simulations. The trained controllers often perform unsatisfactorily on real AUVs; this is because the distributions of the AUV dynamics in numerical simulations and those of real AUVs are mismatched. This paper presents a model-free RL via Data-informed Domain Randomization (DDR) for controlling AUVs, where the mismatches between the trajectory data from numerical simulations and the real AUV were minimized by adjusting the parameters in the simulated AUVs. The DDR strategy extends the existing adaptive domain randomization technique by aggregating an input network to learn mappings between control signals across domains, enabling the controller to adapt to sudden changes in dynamics. The proposed RL via DDR was tested on the problems of AUV pose regulation through extensive numerical simulations and experiments in a lab tank with an underwater positioning system. These results have demonstrated the effectiveness of RL-DDR for transferring trained controllers to AUVs with different dynamics.

Keywords: autonomous underwater vehicles; uncertainty attenuation; reinforcement learning; domain randomization

Citation: Lu, W.; Cheng, K.; Hu, M. Reinforcement Learning for Autonomous Underwater Vehicles via Data-Informed Domain Randomization. *Appl. Sci.* **2023**, *13*, 1723. https://doi.org/10.3390/app13031723

Academic Editors: Qiang Chen, Shubo Wang and Liang Tao

Received: 7 December 2022
Revised: 18 January 2023
Accepted: 26 January 2023
Published: 29 January 2023

1. Introduction

Autonomous Underwater Vehicles (AUVs) and underwater vehicle-manipulator systems are attracting more attention in marine survey and intervention applications, such as structure inspection, cleaning and repairing, waste searching and salvage, deepwater rescuing, sediment sampling, etc. [1–4]. However, possible thruster degeneration and damage, various payloads, and disturbances from currents inevitably bring tremendous uncertainties to AUV dynamics models [5]. In addition, other constraints, such as input dead zones and saturations, make it difficult to tune the controller parameters [6]. This paper explores the feasibility of model-free reinforcement learning approaches to these issues in underwater AUV position regulation problems.

Control problems subject to uncertainties have been investigated over the years. For example, the adaptive controller can to model the uncertainties online, and forces the controlled AUV systems to behave as some given reference models [7]. The adaptation is based on current states and errors as inputs, while long-term optimality is often neglected. The backstepping controller can guarantee system stability through its design processes but requires a sufficiently accurate dynamics model [8]. Otherwise, the control gain would be too large for real systems. The controller based on sliding mode requires sufficient time to reach the sliding surface and provide robustness [9]. These above-mentioned approaches may have provable stabilities without considering the dead zones and saturations of the control inputs.

To be applicable to dynamical systems with input saturations, many existing control approaches assume that the model mismatch is small. Then, these controllers have sufficient margins to suppress uncertainties and input constraints, offering stability to the controlled AUV systems. When applied to real systems, the controller parameters have to be re-tuned according to the altered dynamical system. More importantly, the assumption of small differences in dynamics models may not hold for shallow-water AUVs. In underwater applications, water weeds may tangle one or more thrusters, limiting their maximal thrusts, and the AUVs may be subject to various payloads. All these factors make the model uncertainties quite large.

Controllers based on deep learning have been studied in [10]. Classical model-free RL can overcome the above-mentioned issues of uncertainties, but often relies on a large number of samples to train controllers from scratch. Millions of interactions between the controller and the targeted system are time-consuming and may damage the system itself. Therefore, it is preferable to train a controller in a simulated system and then apply it to a real AUV system. However, the trained controllers often perform unsatisfactorily on real AUVs. This is because the distributions of the state and AUV dynamics in numerical simulations do not match those of real AUVs. Much effort has been devoted to transferring the controller to a different system, referred to as transfer RL [11]. In the context of transfer RL, the trained controller is obtained offline from a source domain with source dynamics. The trained controller is referred to as the source controller. Then, the source controller is transferred to a target controller and is applied to a target system in an online target domain, where the target system is often unknown.

The idea of transferring controllers to a different system has been explored in manipulator control [12]. Recently, a model-free transfer RL on AUV control was validated on numerically simulated AUVs [13]. In the transfer reinforcement learning of robot control, the source domain and source dynamics are often numerically simulated, while the target domain and the target dynamics pertain to the real systems [14]. The source controller is trained for the source dynamics in the source domain and transferred to obtain the target controller to control the target dynamical system in the target domain. In another type of transfer reinforcement learning, the source domain and target domain may share the same dynamics model and state spaces but differ in the definitions of objective functions [15], which is not discussed in this paper.

The correlation across domains is key to transferring the source controller to the target domain [16]. Domain-invariant essential features and structures were studied to build and transfer the controller across domains [17]. The correspondence between domains can be found by analyzing the unpaired trajectories between two different domains [18].

Another issue for sim-to-real transfer RL is robustness. Domain randomization algorithms vary the parameters of the numerically simulated dynamics models and obtain a controller through RL [19]. RL based on domain randomization [20] is a popular technique to reduce domain and dynamics mismatches. Instead of directly adding Gaussian noise to the outputs of the simulated dynamics models, the noise is added to the parameters of dynamics models, allowing for more diverse distributions that can cover the actual distributions of the AUV states and dynamics. The trained controller is robust to the simulated dynamics, which has a high probability of covering the actual manipulator states

and dynamics. Therefore, the trained controller has better robustness than classical RL. This approach has been successful in the control of manipulators, where an accurate dynamics model is required and a friction model is considered [20].

It is preferable to adjust the parameter noise in the domain randomization according to the actual data. The adaptive domain randomization approach adjusts the weight of parameter particles to narrow the gap between simulators and real AUV dynamics [21]. However, the effectiveness of such a strategy is based on an accurate parametric dynamics model. This assumption may not be suitable for AUVs. Existing AUV models have more unmodeled dynamics, which are difficult to describe in the simulations via domain randomization techniques. Therefore, it is desirable to quickly modify the simulator to behave as a real AUV through a data-driven module. Still, the process of obtaining a data-enabled module to close the gap between source dynamics and target dynamics may take a tremendous amount of time. Therefore, it is better if only a few samples of the real AUVs are required to quickly adapt the source controller.

State-of-the-art algorithms based on feedback control may not be able to deal with dynamics that are suffering significant changes, such as shifts in control channels or thruster failures. These changes in dynamics may occur online and jeopardize the AUV system if the same controller is used. On other hand, reinforcement learning approaches require extensive interactions between controllers and the targeted AUV and may adapt themselves to the new dynamics. This paper proposes a transfer RL approach via data-informed domain randomization (DDR) to efficiently adapt the source controller. The contributions are as follows.

(i) Data-informed domain randomization. In this paper, the numerical dynamics model (the source model) is built on a Webots simulator. According to the collected data of a real AUV, the control inputs and state outputs regarding the source model are quite different from those of the target model. A neural network is aggregated onto the Webots source model and is quickly adapted online to match the difference between the source model and the target model, reducing the gap between the source and target dynamics.

(ii) Controller adaptation mechanism. Based on the matching from the proposed DDR, the correlation between the source and target controllers is used to quickly align the source control signals to the target ones. Since the source task and domain task only differ in dynamics models, the mismatch is captured by a small neural network that can be retrained in less than a second with newly collected data.

(iii) Validation through numerical simulations and tank experiments. The proposed RL via DDR was validated by numerical simulations of AUVs with manually designed and mismatched dynamics models: these have different thruster configurations and capabilities. RL via DDR was also tested in a sim-to-real transfer setting, where the transferred controller was tested for an AUV in a tank and the parameters of the AUV were varied.

The remainder of this paper has the following structure. The position regulation problem of an AUV and the transfer RL problem are introduced in Section 2. Section 3 outlines the algorithm of classical RL, followed by the DDR approach, and RL via DDR in Section 4. Section 5 summarizes the simulation results of transfer RL on the AUVs in the Webots simulator and results on sim-to-real experiments. At last, the discussion and conclusions are given in Section 6.

2. Problem Formulation

This paper studies the pose regulation problem of the AUV subject to model uncertainties and other input constraints, such as dead zones and input saturations. Let $X_iY_iZ_i$ denote the earth-fixed reference frame and $X_bY_bZ_b$ denote the body-fixed reference frame [1], as shown in Figure 1. Axis O_iX_i and Axis O_iY_i are in the horizontal plane, Axis O_iZ_i is the gravity direction. The body-fixed frame $X_bY_bZ_b$ is attached to the AUV center and the O_bX_b axis coincides with the AUV heading. The O_bY_b axis is along the starboard

direction. The AUV pose in the frame $Z_iY_iZ_i$ is defined as its position $[x_i, y_t, z_i]^T$ and attitude $[\rho, \psi, \theta]^T$. Similar to many AUVs, the restoring force is often sufficiently large to maintain its roll ρ and pitch ψ close to zero under all circumstances. The restoring force can be manually designed by adjusting the distance between the mass center and the buoyancy center of the AUV.

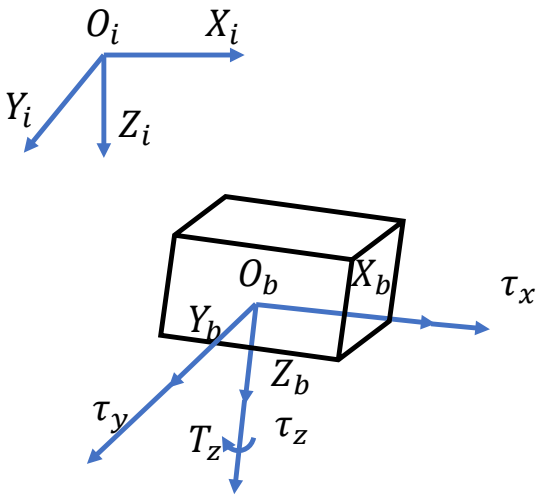


Figure 1. Coordinate frames and control inputs of the AUV.

As a result, in this paper, the studied AUV is described by motions of four Degrees Of Freedom (DOFs), including the linear motions along the O_bX_b, O_bY_b, and O_bZ_b directions and the yaw motion around the O_bZ_b axis. Then, the AUV state pose $\boldsymbol{\eta}$ with respect to the earth-fixed frame $Z_iY_iZ_i$ is redefined as

$$\boldsymbol{\eta}_i \triangleq [x_i, y_i, z_i, \theta_i]^T,$$

where x_i, y_i, and z_i are the AUV's position, and θ_i denotes the AUV's heading in the earth-fixed frame. Then, the AUV's velocity $\boldsymbol{\nu}_i$ in $X_iY_iZ_i$ is defined as

$$\boldsymbol{\nu}_i \triangleq \dot{\boldsymbol{\eta}}_i = [u_i, v_i, w_i, \omega_i]^T,$$

where u_i, v_i, and w_i are translational velocities in the x-, y-, and z-axes in the inertial frame, and ω_i is the angular velocity along the z-axis in the inertial frame. Then, the generalized velocity in $X_bY_bZ_b$ is given as

$$\boldsymbol{\nu} = [u_b, v_b, w_b, \omega_b]^T,$$

where

$$[u_b, v_b, w_b]^T = \mathbf{R}(\theta_i)[u_i, v_i, w_i]^T,$$

$$\omega_b = \omega_i,$$

and $\mathbf{R}(\theta + i)$ denotes the transform from Frame $X_iY_iZ_i$ to Frame $X_bY_bZ_b$. Let the generalized control $\boldsymbol{\tau}$ from lumped thrusts in Frame $X_bY_bZ_b$ be denoted as $\boldsymbol{\tau} \triangleq [\tau_x, \tau_y, \tau_z, T_z]^T$, which may not act through the center of mass. In fact, the mapping between thrusts to the generalized control $\boldsymbol{\tau}$ may not be accurately known, due to manufacturing issues.

The dynamics model of the AUV is often described in the body frame [22] as

$$\boldsymbol{M}\dot{\boldsymbol{\nu}} + \boldsymbol{C}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{D}(\boldsymbol{\nu})\boldsymbol{\nu} + \boldsymbol{g}(\boldsymbol{\eta}) = \boldsymbol{\tau}, \tag{1}$$

where $\boldsymbol{M}$ denotes the inertia matrix with added mass from motion in the water, $\boldsymbol{D}$ denotes the coefficient matrix of the drag forces, $\boldsymbol{C}$ denotes the Coriolis matrix, and $\boldsymbol{g}$ is the lumped vector of gravity and buoyancy. The value of $\boldsymbol{g}$ can be assumed to be constant, while the values of other matrices are partially determined by the velocities of the AUV and water

current, and are, therefore, difficult to estimate. The dynamics model in Equation (1) only captures the main aspects of the actual system under mild conditions, leaving some effects unmodeled, such as thruster dynamics. The control inputs $\boldsymbol{\tau}$ are subject to constraints from the phenomena of saturations and dead zones, which are, respectively, given by

$$\begin{aligned} \underline{\boldsymbol{\tau}} \leq \boldsymbol{\tau} \leq \bar{\boldsymbol{\tau}} \\ |\boldsymbol{\tau}| \geq \boldsymbol{\tau}_d, \end{aligned} \tag{2}$$

where $\underline{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{\tau}}$ denote the saturation bounds of the control inputs, $|\cdot|$ denotes the absolute operation, and $\boldsymbol{\tau}_d$ denotes the dead band. Let $\mathcal{U}$ denote the set of control that satisfies Equation (2). However, the feasible set $\mathcal{U}$ may not be accurately estimated due to varying power supply and current conditions.

Problem 1 (Position Regularation Problem)**.** *Design a controller that can bring the AUV starting at $\boldsymbol{\eta}_i(0)$ to the origin $\boldsymbol{O}_i$, subject unknown dynamics model (1), and the constraints of the dead zones and saturations (2).*

This paper explores the feasibility of RL techniques to train a controller from numerical simulations built on Equation (1). The continuous-time model in (1) is converted into a discrete-time model by Taylor's first-order expansion, as shown below:

$$\boldsymbol{\nu}(t+1) = \boldsymbol{\nu}(t) + \boldsymbol{M}^{-1}\boldsymbol{\tau}(t)\Delta t - \boldsymbol{M}^{-1}\zeta(\boldsymbol{\nu}(t), \boldsymbol{\eta}(t))\Delta t, \tag{3}$$

where

$$\zeta(\boldsymbol{\nu}(t), \boldsymbol{\eta}(t)) = \boldsymbol{C}(\boldsymbol{\nu}(t))\boldsymbol{\nu}(t) + \boldsymbol{D}(\boldsymbol{\nu}(t))\boldsymbol{\nu}(t) + \boldsymbol{g}(\boldsymbol{\eta}(t)) \tag{4}$$

and Δt is the sampling time. The forward dynamics model in (3) is denoted as $\mathcal{F}_s$.

This model can be simulated with estimates of unknown parameters by the Webots simulator and the controller can be trained to solve Problem 1 of the simulated system (3). The controller $\boldsymbol{u}_s$ outputs the control vector $\boldsymbol{\tau}(t)$ at time t, i.e.,

$$\boldsymbol{\tau}(t) = \boldsymbol{u}[\boldsymbol{\nu}(t), \boldsymbol{e}(t)],$$

where $\boldsymbol{e}^T(t)$ is the error of position regulation viewed in the body frame and is obtained by

$$[\boldsymbol{e}^T(t), 1]^T \triangleq \boldsymbol{T}^T(\boldsymbol{\eta}_i)[\boldsymbol{\eta}_i^T, 1]^T.$$

The transformation matrix $\boldsymbol{T}(\boldsymbol{\eta}_i)$ transforms a point in the body frame to the earth-fixed frame. The controller $\boldsymbol{u}(\cdot)$ maps from the AUV velocity and position errors to control inputs. Let $\mathbf{y}(t) \triangleq [\boldsymbol{\nu}^T(t), \boldsymbol{e}^T(t)]^T$ and refer $\mathbf{x}(t)$ as the AUV state at time t. It is assumed that the AUV state can be acquired with small noises at high frequencies.

The controller trained in the source domain $\mathcal{D}_s$ under the source dynamics models $\mathcal{F}_s$ (see Equation (3)) is denoted as $\boldsymbol{u}_s$, referred to as the source controller. The source controller would be applied to a target dynamical system $\mathcal{F}_t$ in a target domain $\mathcal{D}_t$. The forward dynamics model $\mathcal{F}_t$ presents a real AUV system or a simulated AUV with a different dynamics model. Since the direct application of $\boldsymbol{u}_s$ on $\mathcal{F}_t$ brings unsatisfactory results, $\boldsymbol{u}_s$ has to be transferred to obtain $\boldsymbol{u}_t$ for better performance.

Problem 2 (Controller Transfer Problem)**.** *Design an approach to learn a source controller $\boldsymbol{u}_s$ for the source dynamics $\mathcal{F}_s$ and to transfer $\boldsymbol{u}_s$ to the target dynamical system $\mathcal{F}_t$ and obtain a target controller $\boldsymbol{u}_t$, such that the Problem 1 of the target system $\mathcal{F}_t$ can be solved by $\boldsymbol{u}_t$.*

Model-free RL can deal with uncertainties and train controllers on deterministic dynamical systems with various parameters to improve the stability of the trained controller $\boldsymbol{u}_s$ the. However, dynamics models from through-parameter randomization may not cover those in the target domain. To this end, in this paper, a data-informed domain

randomization approach is developed to solve Problem 2. In the meantime, the model mismatches between the source and target dynamics models have to be quickly estimated online to efficiently adjust the controller $\boldsymbol{u}_t$.

3. Reinforcement Learning

This section briefly describes the RL approach to solve Problem 1 of the simulated AUV. As shown in Figure 2, the learning procedure interacts with the simulator and collects the trajectories and rewards. The task in Problem 1 is converted to an optimization problem as

$$\max J = \sum_{t=1}^{\infty} \gamma^t r(t), \tag{5}$$

where γ is a discount factor that penalizes long-term rewards and the reward function R is defined as follows.

$$r(\boldsymbol{e}, \boldsymbol{v}, \boldsymbol{\tau}) \triangleq -{\alpha_1}^T \|\boldsymbol{e}\| - {\alpha_2}^T \|\boldsymbol{v}\| - {\alpha_3}^T \|\boldsymbol{\tau}\|, \tag{6}$$

where $\|\cdot\|$ denotes $L2$-norm. The term $\boldsymbol{e} \in \mathbb{R}^3$ is the regulation error on positions, $\boldsymbol{v} \in \mathbb{R}^4$ is the AUV's generalized velocity and it has to be zero when the AUV is at the origin in Frame $X_i Y_i Z_i$, and $\boldsymbol{\tau} \in R^4$ denotes the control inputs. The weights $\boldsymbol{\alpha}_1$, $\boldsymbol{\alpha}_2$, and $\boldsymbol{\alpha}_3$ are positive parameters and are defined by users. In this paper, they were chosen such that $\boldsymbol{\alpha}_1 \gg \boldsymbol{\alpha}_2 \gg \boldsymbol{\alpha}_3 > 0$. The reward function has to be designed or learned to reflect the control perpurse, which is a hot topic as it affects the convergence process in the learning and stability performance of the AUV [23].

The optimization of the objective (5) is solved against the equality constraints from the AUV dynamics model (3) and the inequality constraints from the control dead zones and saturations (2), outputting the source controller $\boldsymbol{u}_s$. These two types of constraints are simulated in the Webots simulator and are rediscovered by the interaction between RL and the simulator by the data $\{\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t), \boldsymbol{x}_s(t+1), r_s(t)\}_{t \in I_t}$, where I_t is the index set.

RL is essentially a exploration-and-exploitation algorithm that updates its controller through interactions between the learning agent and the simulator (i.e., $\{\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t), \boldsymbol{x}_s(t+1), r_s(t)\}_{t \in I_t}$). After being fitted in Markovian decision processes, the objective function is the cumulated rewards and should be maximized. The reward function is defined in Equation (5) [24]. In the tth iteration, the control inputs $\boldsymbol{\tau}(t) \in \mathcal{U}$ are chosen by the controller $\boldsymbol{u}_s$ based on the state $\boldsymbol{x}_s(t)$ (i.e., the current regulation error $\boldsymbol{e}_t$ and velocities $\boldsymbol{nu}_t$). The simulator receives the control inputs $\boldsymbol{\tau}(t)$ and outputs the AUV state $\boldsymbol{x}_s(t)$ after a one-step simulation and a reward value defined in Equation (6). The obtained data $\{\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t), \boldsymbol{x}_s(t+1), r_s(t)\}_{t \in I_t}$ are used to update the controller network $\boldsymbol{u}_s$ and the critic network V_s, as shown in Figure 2. The critic V_s is essentially the objective function evaluated at the state $\boldsymbol{x}_s$, conditioned on the controller $\boldsymbol{u}_s$, which is not optimal before the convergence of the learning process. The controller $\boldsymbol{u}_s$ is then updated by maximizing the critic V_s at given $\boldsymbol{x}_s$. The procedures are briefly outlined in Algorithm 1.

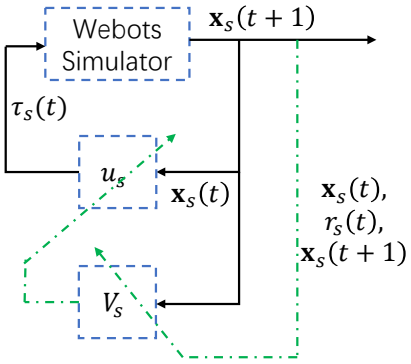


Figure 2. Learning mechanism of reinforcement learning.

While there are many methods to train the controller network, this paper adopts the Soft Actor-Critic (SAC) [25], which is an open-source off-policy learning algorithm. As reported by many researchers, SAC can achieve sufficient results in benchmark problems.

Algorithm 1: Learning Source Controller Network

Randomly initialize controller network π
Initialize memory buffer M
Initialize time step $t = 0$
Sample state $\boldsymbol{x}_s(t) \sim p(\boldsymbol{x}_s(0))$
Insert $(\boldsymbol{x}_s(t+), \boldsymbol{x}_s(t+1), \boldsymbol{\tau}_s(t), r_s(t))$ into R
while *step* $\leq$ *MaxStep* **do**
  while *M is not full* **do**
    Perform action $\boldsymbol{\tau}_s = \boldsymbol{u}_s(\boldsymbol{x}_s(t))$
    Receive next state $\boldsymbol{x}_s(t+1) = \mathcal{F}_s(\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t))$
    Receive reward $r_s(t) = R(\boldsymbol{x}_t, \boldsymbol{u}_t)$
    Insert $(\boldsymbol{x}_s(t+1), \boldsymbol{\tau}_s(t), r_s(t))$ into M
    Update $t \leftarrow t+1$
    if *episode is terminated* **then**
      Reset $t = 0$
      Sample $\boldsymbol{x}_s(t) \sim p(\boldsymbol{x}_s(0))$
      Sample $\boldsymbol{\psi} \sim p(\boldsymbol{\psi})$
    end
  end
  Update $\boldsymbol{u}_s$ using data in M
end

4. Transfer RL via Data-Informed Domain Randomization

In the studied transfer learning of Problem 2, the state space and the control space between the source domain and the target domain share the same dimensions. This section explores a method to train a mapping $H(\boldsymbol{x}_s, \boldsymbol{\tau}_s)$ that transfers $\boldsymbol{u}_s$ to obtain $\boldsymbol{u}_t$ to match the difference between the source dynamics and target dynamics. In addition, it is required that the training of $H(\boldsymbol{x}_s, \boldsymbol{\tau}_s)$ relies on a limited number of data, which can be collected within a few minutes of collection on the real AUV. This section introduces a data-informed domain randomization approach.

4.1. Data-Informed Domain Randomization

Instead of directly adding Gaussian noise to the outputs of the simulated dynamics models, the noise is added to the parameters of dynamics models, allowing for more diverse distributions that can cover the actual distributions of the AUV states and dynamics. In each training episode, the parameters in the AUV dynamics model (3) include the inertia matrix $\boldsymbol{M} \in \mathcal{M}$, the drag matrix $\boldsymbol{D} \in \mathcal{D}$, the Coriolis matrix $\boldsymbol{C} \in \mathcal{C}$, the vector $\boldsymbol{g} \in \mathcal{G}$ of lumped gravity and buoyancy. The sets $\mathcal{M}$, $\mathcal{D}$, $\mathcal{C}$, and $\mathcal{G}$ of these parameters can be estimated [1]. The parameters are randomly sampled from $\mathcal{M}$, $\mathcal{D}$, $\mathcal{C}$, and $\mathcal{G}$. Besides, there are uncertainties regarding the transform matrix from thruster forces to the generalized force that acts through the AUV's center of mass. These uncertainties are also added to the Webots simulator. The control inputs $\boldsymbol{tau}$ are subject to constraints from the phenomena of saturations and dead zones, which are determined by $\underline{\boldsymbol{\tau}}$ and $\bar{\boldsymbol{\tau}}$ and $\boldsymbol{\tau}_d$, respectively. These constraints are also randomly chosen according to an estimated set.

As illustrated in Figure 3, following the above-mentioned domain randomization approach, the dashed ellipse denotes the set $\mathcal{F}_s$ of the AUV dynamics models that can be simulated in Webots. The solid ellipse presents the set $\mathcal{F}_t$ of the real AUV under various conditions, such as thruster configurations and payloads. It is highly possible that the dashed ellipse and the solid ellipse only share a few common dynamics. Therefore, it is

difficult to guarantee the stability of the AUV if the trained source controller $\boldsymbol{u}_s$ is directly applied to the real AUV. $\boldsymbol{u}_s$ might only be robust to the variations in the AUV dynamics in $\mathcal{F}_s \cap \mathcal{F}_t$.

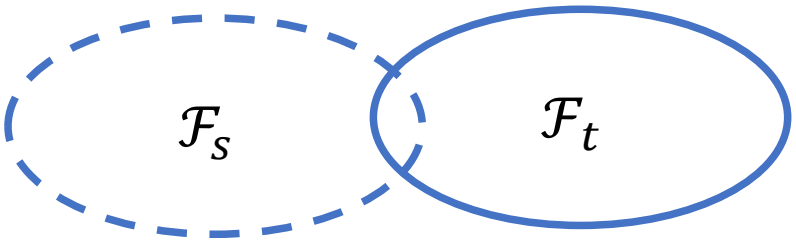
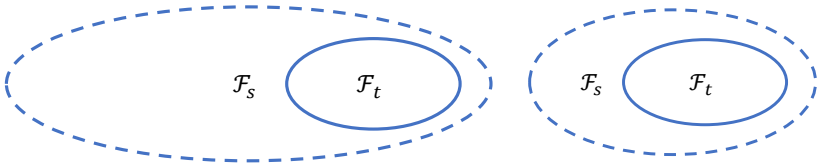


Figure 3. The overlap between the two domains regarding the states and actions.

However, this method is only applicable to the case where the set $\mathcal{F}_t$ of target dynamics (the solid ellipse) is contained by the set $\mathcal{F}_t$ of source dynamics (the dashed ellipse), as shown in Figure 4a. In this case, the trained controller $\boldsymbol{u}_s$ is robust to the dynamics in the solid ellipse. The obtained controller $\boldsymbol{u}_s$ maximizes the expectation of the objective function (5) over all possible dynamics in $\mathcal{F}_s$. In other words, the expectation of the objective function (5) is not optimized over all possible dynamics in $\mathcal{F}_t$. One way to reduce this gap is to adjust the parameter distribution so that $\mathcal{F}_s$ is close to $\mathcal{F}_t$.

Many adaptive domain adaptation techniques have been studied, and they bias the distribution of parameters based on the trajectory data collected from the target domain $\mathcal{D}_s$. The likelihood of the trajectory obtained from various simulated dynamics compared to an obtained trajectory from the real AUV can be computed and used to update the weights of the parameter particles. In each episode, particles with different weights are used to sample parameter values following the particle filter strategy, and to train the source controller $\boldsymbol{u}_s$. This procedure identifies the system parameters and many other similar approaches can be explored in the future.



(a) **(b)**

Figure 4. Adaptive domain randomization adjusts the probability distributions of sampling model parameters and converts the source domain from (**a**) to (**b**).

However, the performance of the above strategy heavily relies on the assumption that $\mathcal{F}_t \subset \mathcal{F}_s$. While treating unmodelled dynamics as noise, the nominal behavior of the actual AUV dynamics is contained by the set of source dynamics models. The actual system may exhibit dynamics that are not covered by $\mathcal{F}_s$. Therefore, this paper proposes the following data-informed domain randomization (DDR) approach, as shown in Figure 5. It is based on the assumption that the variation in a given dynamics model can be covered by the mapping between the control inputs. The mapping G is illustrated as the "G" block of the diagram in Figure 5.

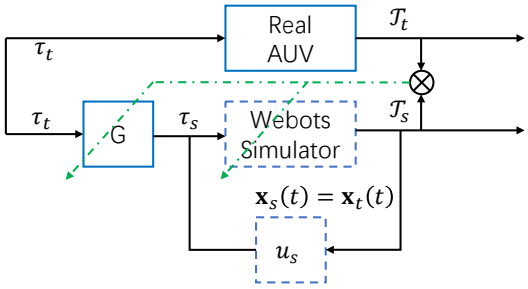


Figure 5. The Data-informed Domain Randomization (DDR) adaptation mechanism.

The mapping has a limited number of neurons, only requires a few training samples and can deal with many changes regarding the real AUV—for example, when one truster is tangled by seaweeds and its thruster force is degenerated. In other cases, when the external force from payloads changes, the mapping of the forces can also implicitly estimate the payloads. The scale between inputs forces and mass and other terms can also be captured by the mapping G. The mapping between controllers $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ is then defined as

$$\boldsymbol{\tau}_s = G(\boldsymbol{\tau}_t). \tag{7}$$

The goal of the DDR is to reduce the gap between the outputs from the simulated dynamics model and the trajectories collected from the actual AUV. The loss function in training is given as $\|\mathcal{T}_s - \mathcal{T}_t\|$, where the trajectory $\mathcal{T}_t$ is collected from the actual AUV, while $\mathcal{T}_s$ is obtained from the numerical simulation. Since it might be difficult to find a close enough trajectory, it is better to create an inverse dynamics model of the simulated model (3). Therefore, in this paper, an inverse model of (3) was obtained via supervised training. The obtained inverse dynamics model is denoted as $R(\boldsymbol{\nu}(t), \boldsymbol{\nu}(t+1), \boldsymbol{\eta})$.

In online applications, the adaptive domain randomization approach aims to reduce the variance in parameters. When trajectories are obtained from the real AUV, control signals are also obtained. From the inverse model, the control inputs can be calculated from Equation (7). The loss function to train G is given as

$$\mathcal{L}(G) \triangleq \|R(\boldsymbol{\nu}(t), \boldsymbol{\nu}(t+1), \boldsymbol{\eta}) - G(\boldsymbol{\tau}_t)\| \tag{8}$$

Through learning G, this approach is able to quickly capture the changes in the dynamics model of the real AUV. The obtained mapping $G(\boldsymbol{\tau}_t)$ is used in the controller transfer. The desired outputs $\boldsymbol{\tau}_s$ of G can be obtained from $R(\boldsymbol{\nu}(t), \boldsymbol{\nu}(t+1), \boldsymbol{\eta})$ and the input to G is $\boldsymbol{\tau}_t$. During the runtime, a bag l of pairs $(\boldsymbol{\tau}_t, \boldsymbol{\tau}_s)$ is maintained to keep the latest pairs and has a fixed size of $n_b = n_p^2/2$, where n_p is the number of total unknown weights in the neural network G. When the value of Equation (8) is within a certain threshold, the AUV dynamics do not change much and the adaptation based on the gradient is given as

$$\boldsymbol{\theta}_G = \boldsymbol{\theta}_G + \beta \frac{\partial G}{\partial \boldsymbol{\theta}_G}. \tag{9}$$

When the value of Equation (8) is larger than a certain threshold, the bag l is emptied and quickly collects new data; then, the network G is retrained from scratch. Since the network G is quite small, this often takes less than a second.

4.2. Controller Transfer across Domains

Finding correspondences between the dynamics and controller across domains is key to improving the learning efficiency in the target domain [13]. In Problem 2, the position regulation tasks of the source and target domains are the same, while the robot dynamics models of the two domains are different. As shown in Figure 6, the successful transfer of the controller $\boldsymbol{u}_s$ to the target domain $\mathcal{D}_t$ highly depends on the control alignment. The

proposed transfer RL via DDR has been illustrated in Figure 6. During training, the trajectories $\mathcal{T}_s \triangleq (\mathbf{x}_s(t), \mathbf{u}_s(t), \mathbf{x}_s(t+1))$ and $\mathcal{T}_t \triangleq (\mathbf{x}_t(t), \mathbf{u}_t(t), \mathbf{x}_t(t+1))$ of the AUV were collected through interactions with the source and target dynamics, respectively. Based on the collected $\mathcal{T}_s$ and $\mathcal{T}_t$, the nerual network H, G, and R are trained in the following order. The network H is trained and then fixed in training G. Once H and G are obtained, R was trained.

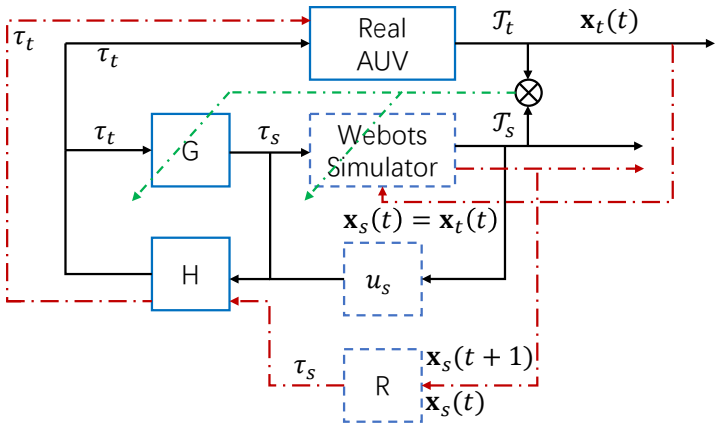


Figure 6. Transfer process between the source controller and target controller.

The control alignment depends on $H : \mathcal{X} \times \mathcal{U}_s \mapsto \mathcal{U}_t$, which is learned together with G. Different from existing research, these two mappings are learned in the DDR process. In the forward source dynamics model $\mathcal{F}_s$ is trained to build $H(\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t)) = \boldsymbol{x}_s(t+1)$. The forward model takes in the state $\boldsymbol{x}_s(t)$ and the control $\boldsymbol{\tau}_s(t)$ at time t as inputs and outputs the state at time $t+1$. The learning of F_s is conducted offline via supervised learning, based on the trajectories collected through numerical simulations. The loss function regarding forward model F_s is given as

$$\min_F \mathcal{L}(F) = \sum_{(\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t), \boldsymbol{x}_s(t+1)) \in \mathcal{T}_s} \|\boldsymbol{x}_s(t+1) - F(\boldsymbol{x}_s(t), \boldsymbol{\tau}_s(t))\|. \tag{10}$$

Given forward model F_s, the loss function to train H is defined as

$$\min_H \mathcal{L}(H) = \sum_{(\boldsymbol{x}_t(t), \boldsymbol{\tau}_t(t), \boldsymbol{x}_t(t+1)) \in \mathcal{T}_t} \|\mathbf{x}_t(t+1) - F(\mathbf{x}_t(t), H(\mathbf{x}_t(t), \boldsymbol{\tau}_t(t)))\|. \tag{11}$$

Then, keeping the F_s model fixed, the model R is trained. In addition, the transformation from $\mathbf{u}_t$ to $\mathbf{u}_s$ should be reversible. The translated control can be mapped back to the original domain. This requirement adds a regulation in training R, so the loss function is given as

$$\min_R \mathcal{L}(R) = \mathbb{E}_{\boldsymbol{\tau}_t \sim p(\mathbf{u}_t)} \|H(\boldsymbol{\tau}_s(t), R(\mathbf{x}_s(t), \mathbf{x}_s(t+1))) - \boldsymbol{\tau}_t\|. \tag{12}$$

Once these networks are trained, the target controller can be obtained as follows

$$\boldsymbol{u}_t = H\{R[\boldsymbol{x}, F(\boldsymbol{x}, \boldsymbol{u}_s(\boldsymbol{x}))]\},$$

and can be applied to the target dynamics $\mathcal{F}_t$. When the real AUV system is subject to sudden changes, the mappings G and H are retrained. As shown by the results in Section 5, only a few data from various episodes are required.

5. Simulation and Experimental Results

The proposed transfer reinforcement-learning-based data-informed domain randomization was tested in numerical simulations and experiments on a real AUV in a lab tank. In both cases, the source domain was the numerical simulations conducted in Webots, where the mass of the AUV was set as 12 kg. The dynamics model of the simulated AUV is given

in Equation (3), with neutral buoyancy. As shown in Figure 7, the AUV was equipped with six thrusters, each of which can output thrust within $[-50, 50]$ N. The dead band was chosen as $[-5, 5]$ N. As in Section 2, the AUV is able to move in the x, y, and z directions of the body frame and rotate along the z axis. In addition, the control input is mapped to PWM signals of four dimensions. An example of the numerical simulation is depicted in Figure 8, where the origin O_i in the earth-fixed frame is illustrated as a small red ball.

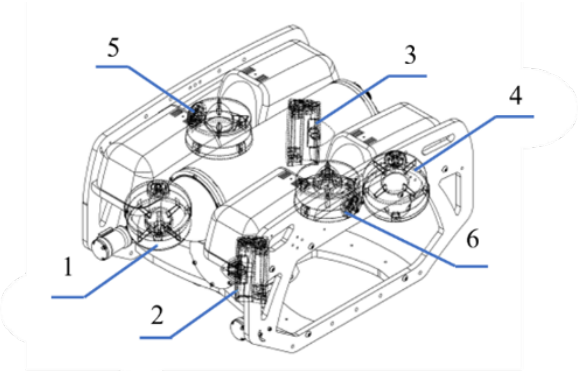


Figure 7. A schematic of the AUV used in numerical simulations and experiments, where 6 thrusters are indexed from 1 to 6.

A source policy $\boldsymbol{u}_s$ was trained by RL through interactions with $\mathcal{F}_t$ in the source domain. In each episode, the initial AUV pose and velocities were randomly sampled from a set, where the positions were within a box of $[-10, 10]$ $[m]$ $\times$ $[-10, 10]$ $[m]$ $\times$ $[-10, 10]$ $[m]$ and the velocities were within a box of $[-2, 2]$ $[m/s]$ $\times$ $[-2, 2]$ $[m/s]$ $\times$ $[-2, 2]$ $[m/s]$. The controller network $\boldsymbol{u}_s$ output the four-dimensional PWM signal vector, which was converted to a generalized force in the body frame $X_b Y_b Z_b$.

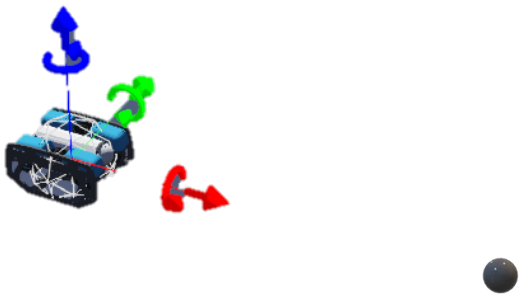


Figure 8. AUV simulated in the Webots simulator with the ball represents the targeted origin O_s.

In the numerical simulations used to train the source controller, the generalized mass matrix is given as

$$\boldsymbol{M} = \begin{bmatrix} m + X_{\dot{u}} & 0 & 0 & 0 \\ 0 & m + Y_{\dot{v}} & 0 & 0 \\ 0 & 0 & m + Z_{\dot{w}} & 0 \\ 0 & 0 & 0 & I_z + N_{\dot{r}} \end{bmatrix}, \tag{13}$$

The coefficient matrix of the drag term is

$$\boldsymbol{D} = \begin{bmatrix} -X_u - X_{u|u|}|u| & 0 & 0 & 0 \\ 0 & -Y_v - Y_{v|v|}|v| & 0 & 0 \\ 0 & 0 & -Z_w - Z_{w|w|}|w| & 0 \\ 0 & 0 & 0 & -N_r - N_{r|r|}|r| \end{bmatrix} \tag{14}$$

and the Coriolis matrix $\boldsymbol{C}$ was assumed to be zero. The parameters can be found in Table 1.

Table 1. Parameters of AUV models in numerical simualtions.

Index	Parameter	Value
1	$m + X_{\dot{u}}$	15.9 kg
2	$m + Y_{\dot{v}}$	32.2 kg
3	$m + Z_{\dot{w}}$	14.5 kg
4	$I_z + N_{\dot{r}}$	0.3 kg·m
5	X_u	−36.8 kg
6	$X_{\|u\|u}$	374.7 kg·s/m
7	Y_v	−60.6
8	$Y_{\|v\|v}$	197.2 kg·s/m
9	Z_w	−112.2 kg
10	$Z_{\|w\|w}$	153.2 kg·s/m
11	N_r	−0.32 kg·m
12	$N_{\|r\|r}$	0.95 kg·m·s

The target domains differ in the numerical simulation tests and the experimental tests. In the numerical simulation tests, the target domain and target dynamics were again simulated in the Webots simulator, subject to parameters and configurations quite different from the simulated source domain and the source dynamics. The tested scenarios in the case are referred to as sim-to-sim transfer tests. In the experimental tests, the target domain and the target dynamics were of the real AUV in the lab tank, the details of which are introduced later.

5.1. Sim-to-Sim Tests

To test the proposed transfer RL via DDR, three scenarios with manually designed model mismatches were simulated. These scenarios were manually designed to reflect possible changes in the real AUV dynamics due to some sudden events. The first scenario involves changing the pose configurations of two thrusters to create a model mismatch between the source dynamics model $\mathcal{F}_s$ and the target dynamics model $\mathcal{F}_t$. As shown in Figures 9 and 10, the source controller $\boldsymbol{u}_s$ was unable to stabilize the AUV with the target dynamics. This is because the changes in the pose configurations of thrusters with respect to the center of mass introduce a shift in the mapping, from the thrusts to the generalized. The mapping from $\boldsymbol{\tau}_t$ to $\boldsymbol{\tau}_s$ was quickly learned by a few episodes of data collection regarding the target dynamics. During these episodes, the performance of the position regulation in these episodes was poor and was not analyzed. After that, the mapping H was updated and the resultant $\boldsymbol{u}_t$ was able to stabilize the AUV of the target dynamics, as shown in Figure 9. To illustrate the effectiveness of transfer RL, the trajectories of the AUV in the target domain under $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ are shown in Figure 10. The dashed line represents the trajectory obtained by $\boldsymbol{u}_s$, and the solid line represents the one obtained by $\boldsymbol{u}_t$. Let the error of position regulation be defined as the L2 norm of the AUV state. The errors of regulating AUV positions from $\boldsymbol{u}_s$ and $\boldsymbol{u}_s$ are shown in Figure 9. The disturbance effects and control forces are shown in Figure 11.

A common issue is that the characteristics of a thruster may gradually change during its lifetime or vary suddenly due to certain events. In the second scenario, the gain and the maximum thrust of some thrusters were manually designed to simulate the case in which some thrusters were entangled by some seaweeds. As shown in Figure 12, when the source controller $\boldsymbol{u}_s$ was tested on the target dynamics $\mathcal{F}_t$, the reduced gain of some thrusters introduced additional unwanted torque along z axis, making the AUV oscillate its heading. The mapping from $\boldsymbol{\tau}_t$ to $\boldsymbol{\tau}_s$ was quickly learned over a few episodes. Again, the performance of the position regulation in these episodes was not analyzed. The trajectories of the AUV in the target domain under $\boldsymbol{u}_t$ and $\boldsymbol{u}_t$ are shown in Figure 12. The errors when regulating AUV positions from $\boldsymbol{u}_s$ and $\boldsymbol{u}_s$ are shown in Figure 13.

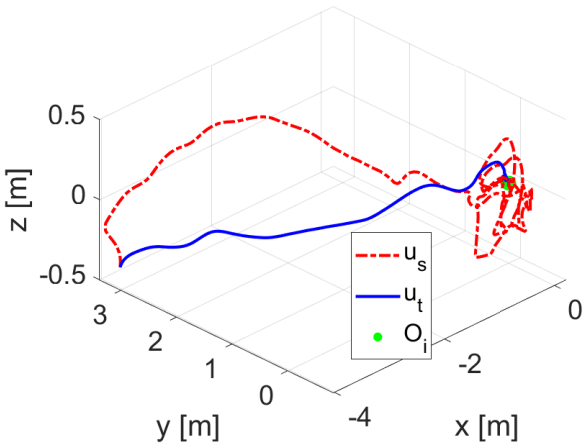


Figure 9. Trajectories obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$.

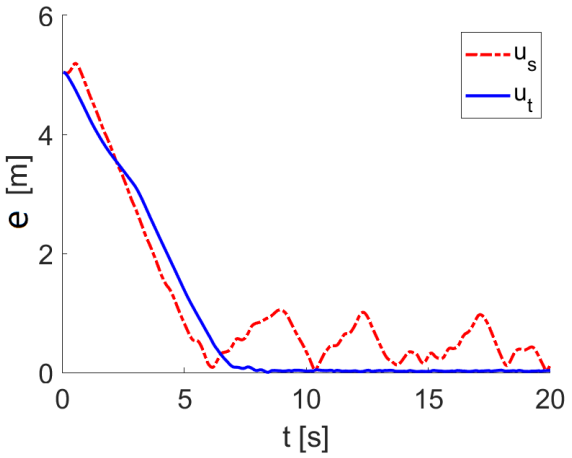


Figure 10. Errors of the position regulation obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$.

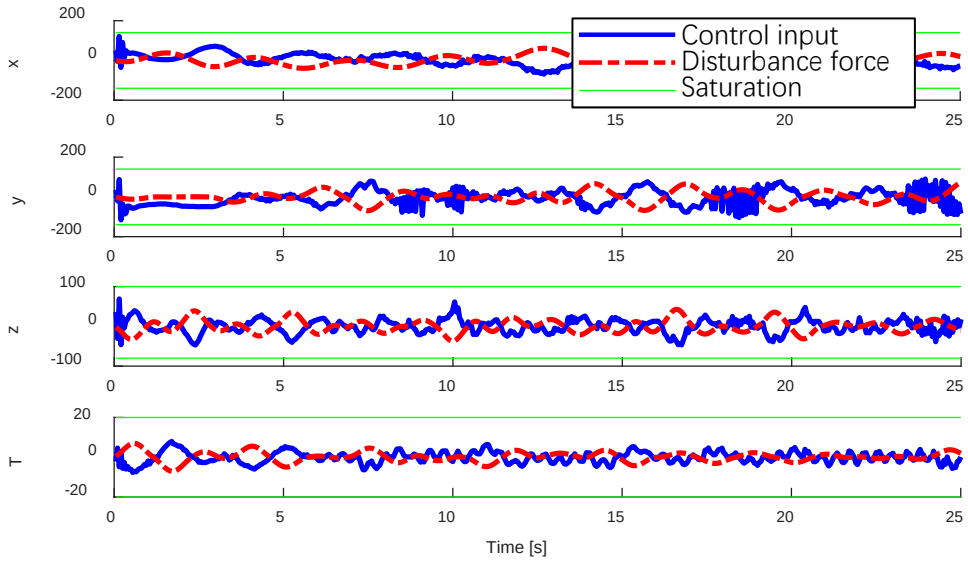


Figure 11. Disturbance effects and control forces.

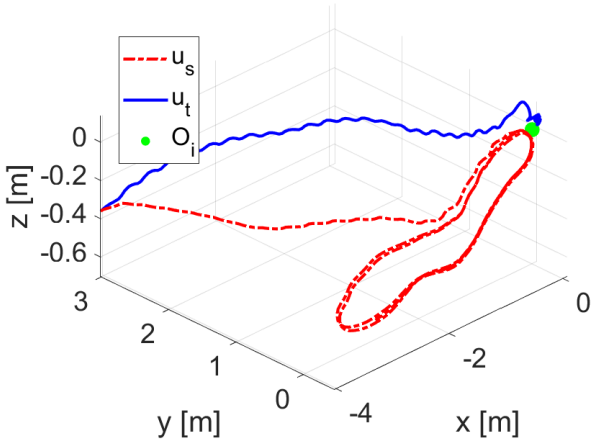


Figure 12. Trajectories obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$, he maximal thrusts of the four horizontal thrusters were modified to 50 N, 80 N, 10 N, and 50 N.

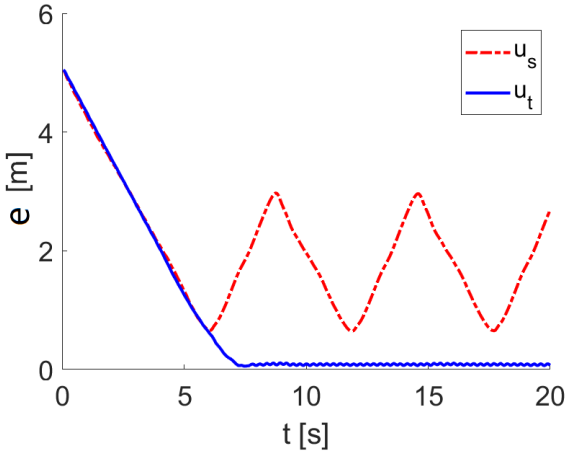


Figure 13. Errors of position regulation obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$, where, in the target dynamics, the maximal thrust of the four horizontal thrusters were modified to 50 N, 80 N, 10 N, and 50 N.

The third scenario simulated an extreme case, where one of the four horizontal thrusters was damaged and a second thruster suddenly changed its phase in the PWM driver, causing it to rotate in the opposite direction. In this scenario, classical controllers such as PID or adaptive controllers may fail, since these methods often assume that the system reserves the positiveness of the gain matrix. When one of the thrust outputs forces opposite to the desired direction, the AUV system is easily destabilized. After the mapping H was updated, the transferred controller $\boldsymbol{u}_t$ was able to stabilize the AUV in the target domain, as shown in Figure 14. The trajectories of the AUV in the target domain under $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ are shown in Figure 14. When one of the horizontal thrusters is damaged, the horizontal motion is still fully actuated in a horizontal plane, allowing for G and H to be mapped. The position error is shown in Figure 15. After a quick test, the proposed approach is unstable the AUV if two horizontal thrusters are damaged.

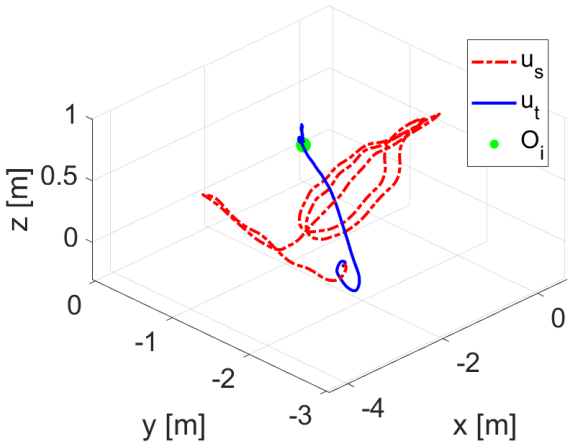


Figure 14. Trajectories obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$, where, in the target dynamics, one thruster is damaged and another reverses its rotating direction.

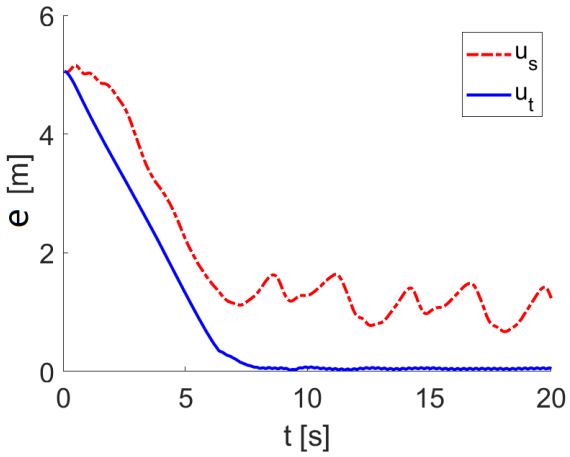


Figure 15. Errors of position regulation obtained by $\boldsymbol{u}_s$ and $\boldsymbol{u}_t$ for the AUV of $\mathcal{F}_t$, where, in the target dynamics, one thruster is damaged and another reverses its rotating direction.

It has been observed that the mapping G can be learned within a few episodes, with each episode lasting for 20 s. The results from three scenarios show that DDR is able to transfer the source controller $\boldsymbol{u}_s$ to the target controller $\boldsymbol{u}_t$, by creating mappings G and H across the source dynamics and target dynamics.

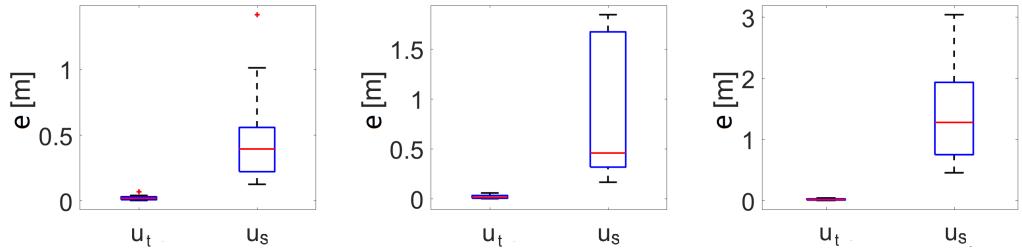
Then, for each scenario, approximately $N = 100$ initial AUV states were randomly sampled and the resultant trajectories of AUV were recorded. The stable performance of the j trajectory is given as

$$\eta_j = \mathbb{E}\|\boldsymbol{e}_j(t)\|, t \geq 10[\text{s}]. \tag{15}$$

Then, the average performance is given as

$$\eta = 1/N \sum_{j=1,\ldots,N} \eta_j, \tag{16}$$

and the variance σ can also be obtained. The mean and variance of η with respect to all above-mentioned scenarios are illustrated in Figure 16a–c and Table 2.



(a) Scenario one **(b)** Scenario Two **(c)** Scenario Three

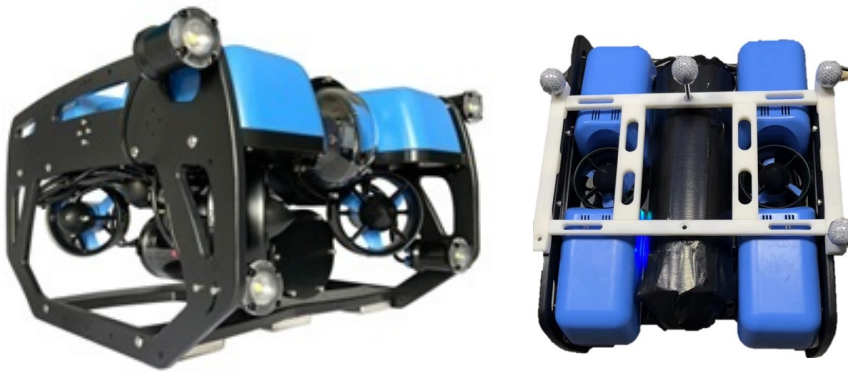
Figure 16. Mean square error of 100 trajectories of AUV regarding three scenarios.

Table 2. Mean errors from three scenarios.

Scenario	Mean Error: u_t	Mean Error: u_t
"Swapping commands to Thrusters 1 and 2"	0.019 [m]	0.41 [m]
"Manually reducing gain of the thrusters"	0.028 [m]	0.39 [m]
" Setting negative gain to Thruster 1"	0.021 [m]	1.16 [m]

5.2. Sim-to-Real Tests

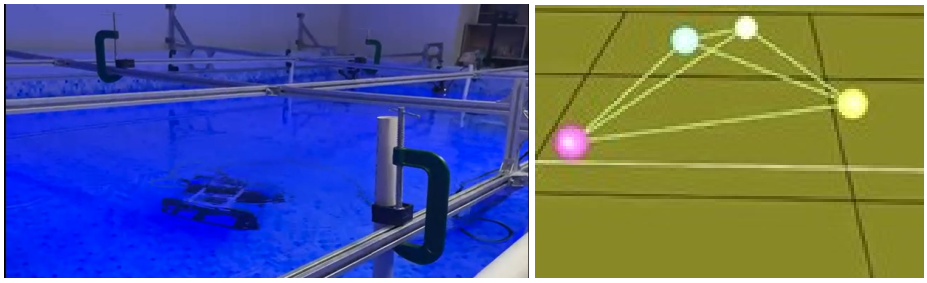
The proposed data-informed domain randomization approach was applied to an underwater robot built on BlueRov2 from Blue Robotics, Inc. (Torrance, CA, USA), as shown in Figure 17a. It is a tethered ROV with six thrusters. With the given thruster configuration, the ROV is able to translate in three directions, roll, and yaw. The ROV is designed to have a sufficient restoring force, to keep itself horizontal. The ROV communicates with a laptop through the Robot Operating System (ROS) and receives thruster commands in the form of PWM signals. In order to provide real-time state estimation of the ROV, an underwater optical motion capture system from Nokov was implemented, which also communicates via ROS and publishes the pose topic in the form of three-dimensional vectors and quaternions. Since the field of view of cameras is often small in underwater applications, the motion capture system relies on 12 underwater cameras mounted on the walls of the tank. In addition, due to the short visibility distance, the reflective markers are of 30 mm, as shown in Figure 17b.



(a) **(b)**

Figure 17. (**a**) BlueROV2 from Blue Robotivs, Inc.; (**b**) ROV with four reflective markers of 30 mm.

The cameras emit blue lights (as shown in Figure 18a) and capture the pose of a rigid frame (as shown in Figure 18b) built on markers at 60 Hz. The motion capture system was calibrated with an L-shaped calibrator, which consists of four markers placed in the middle of the pool. The positioning system can reach a sub-centimeter resolution.



(a) **(b)**

Figure 18. (**a**) The motion capture system in operations; (**b**) The rigid frame built on reflective markers.

The laptop receives the pose estimation from the motion capture system, implements the proposed RL approach, and sends the control signals to the ROV. The whole system is referred to as the testbed of the "AUV". In the future, a sonar-based localization approach and the proposed transfer RL algorithm will be implemented in the updated hardware of the ROV, making it an AUV.

The dynamics model of the real AUV (i.e., the target dynamics model) is jointly determined by the body morphology and AUV velocities. The mass and other parameters are difficult to estimate. Note that only the dead zones and the saturation of the thrusters were simulated; however, the complex dynamics of the thrusters were not simulated in the source dynamics [26]. Therefore, there mismatches between the source dynamics model and the target model are inevitable. In addition, external disturbances were generated by a propeller fixed to the tank and an example of external disturbances is shown in Figure 19. Note that, when testing the algorithm, the AUV was detached from the force-torque sensor and the disturbance values are unknown.

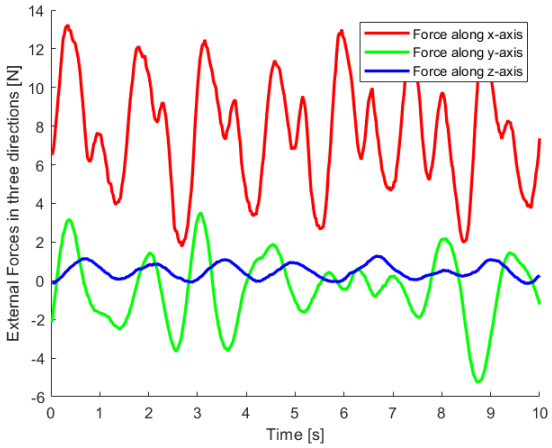
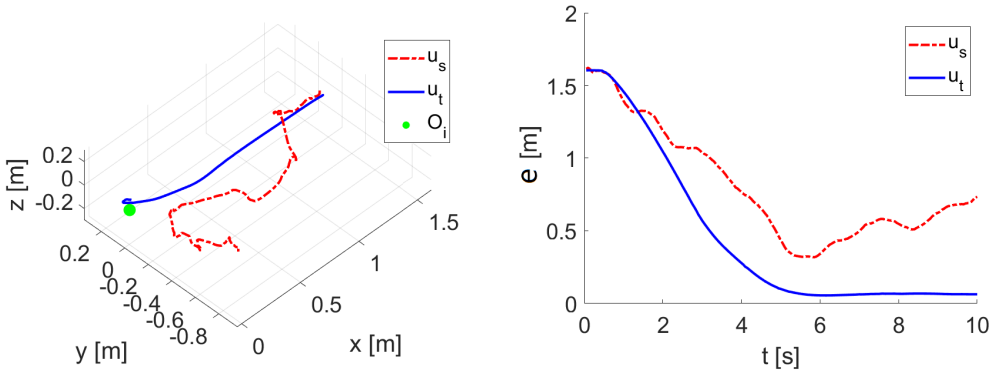


Figure 19. An example of external disturbances generated by a propeller fixed to the tank.

In one of the experimental tests, the AUV started from the position $\boldsymbol{x}(0) = [0.18, -0.46, -0.4]^T$. Two trajectories were obtained: one $\mathcal{T}_s$ obtained by $\boldsymbol{u}_s$ and the second $\mathcal{T}_t$ obtained by $\boldsymbol{u}_t$. Both trajectories are illustrated in Figure 20a, where $\mathcal{T}_s$ is shown by the red dashed line and $\mathcal{T}_t$ by the blue solid line. The trajectories demonstrated that the transferred controller $\boldsymbol{u}_t$ successfully regulated the AUV position, while the controller $\boldsymbol{u}_s$ was unable to stabilize the AUV around the origin O_t. The position regulation errors from both trajectories are shown in Figure 20b. The transferred controller $\boldsymbol{u}_t$ stabilized the AUV in a sphere of radius 0.15 mm at the origin O_t.



(a) Trajectories **(b)** Error

Figure 20. An example of trajectory and position regulation error from tank tests.

6. Conclusions

This paper studies the controller transferring problem from a source AUV system to a (simualted or real) target AUV system and proposes data-informed domain randomization to reduce the gap between the domains and to improve the efficiency in online adaptation. A small neural nework builds mapping between source and taget control signals, and enables the effective adaptation of controllers optimized from the source dynamics to the target dynamics. The proposed method was extensively tested via numerical simulations of the position regulation problems. The method was also validated by experiments in a tank with a positioning system. The proposed transfer RL via DDR relies on the assumption that the state space and the objective function across domains are the same, making the alignment easy on the trajectories. In the future, a new module to align states of different dimensions should be explored. In addition, the positioning system provides high-resolution state estimation, which is not available for outdoor applications. The effects of noise on the state estimations should be considered in future studies.

Author Contributions: Methodology, W.L.; Validation, K.C.; Writing—original draft, M.H. All authors have read and agreed to the published version of the manuscript.

Funding: This work is supported by the National Natural Science Foundation of China #62003110 and the Shenzhen Science and Technology Innovation Foundation #JCYJ20210324132607018 and #JSGG20210420091804012.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data is available at https://wenjielu.cn.

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

References

1. Antonelli, G.; Antonelli, G. *Underwater Robots*; Springer: Berlin/Heidelberg, Germany, 2014; Volume 3.
2. Griffiths, G. *Technology and Applications of Autonomous Underwater Vehicles*; CRC Press: Boca Raton, FL, USA, 2002; Volume 2.
3. Low, H.E.; Randolph, M.F.; Rutherford, C.; Bernard, B.B.; Brooks, J.M. Characterization of near seabed surface sediment. In Proceedings of the Offshore Technology Conference, Houston, TX, USA, 5–8 May 2008.
4. Li, S.; Dutta, B.; Cannon, S.; Daymude, J.J.; Avinery, R.; Aydin, E.; Richa, A.W.; Goldman, D.I.; Randall, D. Programming active cohesive granular matter with mechanically induced phase changes. *Sci. Adv.* **2021**, *7*, eabe8494. [CrossRef] [PubMed]
5. Zhu, D.; Wang, L.; Hu, Z.; Yang, S.X. A Grasshopper Optimization-based fault-tolerant control algorithm for a human occupied submarine with the multi-thruster system. *Ocean. Eng.* **2021**, *242*, 110101. [CrossRef]
6. Yu, C.; Zhong, Y.; Lian, L.; Xiang, X. An experimental study of adaptive bounded depth control for underwater vehicles subject to thruster's dead-zone and saturation. *Appl. Ocean. Res.* **2021**, *117*, 102947. [CrossRef]
7. Ghamari, S.M.; Mollaee, H.; Khavari, F. Robust self-tuning regressive adaptive controller design for a DC–DC BUCK converter. *Measurement* **2021**, *174*, 109071. [CrossRef]

8. Azinheira, J.R.; Moutinho, A.; de Paiva, E.C. A backstepping controller for path-tracking of an underactuated autonomous airship. *Int. J. Robust Nonlinear Control IFAC Affil. J.* **2009**, *19*, 418–441. [CrossRef]
9. Raptis, I.A.; Valavanis, K.P.; Moreno, W.A. A novel nonlinear backstepping controller design for helicopters using the rotation matrix. *IEEE Trans. Control Syst. Technol.* **2010**, *19*, 465–473. [CrossRef]
10. Lewis, F.; Jagannathan, S.; Yesildirak, A. *Neural Network Control of Robot Manipulators and Non-Linear Systems*; CRC Press: Boca Raton, FL, USA, 2020.
11. Parisotto, E.; Ba, J.L.; Salakhutdinov, R. Actor-mimic: Deep multitask and transfer reinforcement learning. *arXiv* **2015**, arXiv:1511.06342.
12. Lowrey, K.; Kolev, S.; Dao, J.; Rajeswaran, A.; Todorov, E. Reinforcement learning for non-prehensile manipulation: Transfer from simulation to physical system. In Proceedings of the 2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR), Brisbane, QLD, Australia, 16–19 May 2018; pp. 35–42.
13. Wang, T.; Lu, W.; Yu, H.; Liu, D. Modular Transfer Learning with Transition Mismatch Compensation for Excessive Disturbance Rejection. *arXiv* **2020**, arXiv:2007.14646.
14. Zhao, W.; Queralta, J.P.; Westerlund, T. Sim-to-real transfer in deep reinforcement learning for robotics: A survey. In Proceedings of the 2020 IEEE Symposium Series on Computational Intelligence (SSCI), Canberra, ACT, Australia, 1–4 December 2020; pp. 737–744.
15. Barreto, A.; Dabney, W.; Munos, R.; Hunt, J.J.; Schaul, T.; van Hasselt, H.P.; Silver, D. Successor features for transfer in reinforcement learning. *arXiv* **2017**, arXiv:1606.05312.
16. Zhang, Q.; Xiao, T.; Efros, A.A.; Pinto, L.; Wang, X. Learning cross-domain correspondence for control with dynamics cycle-consistency. *arXiv* **2020**, arXiv:2012.09811.
17. Gupta, A.; Devin, C.; Liu, Y.; Abbeel, P.; Levine, S. Learning invariant feature spaces to transfer skills with reinforcement learning. *arXiv* **2017**, arXiv:1703.02949.
18. Bansal, A.; Ma, S.; Ramanan, D.; Sheikh, Y. Recycle-gan: Unsupervised video retargeting. In Proceedings of the European Conference on COMPUTER Vision (ECCV), Munich, Germany, 8–14 September 2018; pp. 119–135.
19. Peng, X.B.; Andrychowicz, M.; Zaremba, W.; Abbeel, P. Sim-to-real transfer of robotic control with dynamics randomization. In Proceedings of the 2018 IEEE International Conference on Robotics and Automation (ICRA), Brisbane, QLD, Australia, 21–25 May 2018; pp. 3803–3810.
20. Tobin, J.; Fong, R.; Ray, A.; Schneider, J.; Zaremba, W.; Abbeel, P. Domain randomization for transferring deep neural networks from simulation to the real world. In Proceedings of the 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Vancouver, BC, Canada, 24–28 September 2017; pp. 23–30.
21. Ramos, F.; Possas, R.C.; Fox, D. Bayessim: Adaptive domain randomization via probabilistic inference for robotics simulators. *arXiv* **2019**, arXiv:1906.01728.
22. Newman, J.N. *Marine Hydrodynamics*; The MIT Press: Cambridge, MA, USA, 2018.
23. Devlin, S.M.; Kudenko, D. Dynamic potential-based reward shaping. In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems, IFAAMAS, Valencia, Spain, 4–8 June 2012; pp. 433–440.
24. Yao, L.; Dong, Q.; Jiang, J.; Ni, F. Deep reinforcement learning for long-term pavement maintenance planning. *Comput.-Aided Civ. Infrastruct. Eng.* **2020**, *35*, 1230–1245. [CrossRef]
25. Haarnoja, T.; Zhou, A.; Abbeel, P.; Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Proceedings of the International Conference on Machine Learning, PMLR, Stockholm, Sweden, 10–15 July 2018; pp. 1861–1870.
26. Healey, A.J.; Rock, S.; Cody, S.; Miles, D.; Brown, J. Toward an improved understanding of thruster dynamics for underwater vehicles. *IEEE J. Ocean. Eng.* **1995**, *20*, 354–361. [CrossRef]