machines

MDPI

Communication

Continuous Control of an Underground Loader Using Deep Reinforcement Learning

Sofi Backman [1], Daniel Lindmark [1], Kenneth Bodin [1,2], Martin Servin [1,2,*], Joakim Mörk [1] and Håkan Löfgren [3]

[1] Algoryx Simulation AB, Kuratorvägen 2, 90736 Umeå, Sweden; sofi.backman@algoryx.se (S.B.); daniel.lindmark@algoryx.se (D.L.); kenneth.bodin@algoryx.se (K.B.); joakim.mork@algoryx.se (J.M.)
[2] Department of Physics, Umeå University, 90187 Umeå, Sweden
[3] Epiroc AB, Sickla Industriväg 19, 13154 Nacka, Sweden; hakan.c.lofgren@epiroc.com
* Correspondence: martin.servin@umu.se

Abstract: The reinforcement learning control of an underground loader was investigated in a simulated environment by using a multi-agent deep neural network approach. At the start of each loading cycle, one agent selects the dig position from a depth camera image of a pile of fragmented rock. A second agent is responsible for continuous control of the vehicle, with the goal of filling the bucket at the selected loading point while avoiding collisions, getting stuck, or losing ground traction. This relies on motion and force sensors, as well as on a camera and lidar. Using a soft actor–critic algorithm, the agents learn policies for efficient bucket filling over many subsequent loading cycles, with a clear ability to adapt to the changing environment. The best results—on average, 75% of the max capacity—were obtained when including a penalty for energy usage in the reward.

Keywords: autonomous excavation; bucket filling; deep reinforcement learning; mining robotics; simulation; wheel loader

Citation: Backman, S.; Lindmark, D.; Bodin, K.; Servin, M.; Mörk, J.; Löfgren, L. Continuous Control of an Underground Loader Using Deep Reinforcement Learning. *Machines* **2021**, *9*, 216. https://doi.org/10.3390/machines9100216

Academic Editor: Dan Zhang

Received: 28 August 2021
Accepted: 24 September 2021
Published: 27 September 2021

1. Introduction

Load–haul–dump (LHD) machines are used for loading blasted rock (muck) in underground mines and hauling the material to trucks or shafts, where it is dumped. Despite moving in a well-confined space with only a few degrees of freedom to control, LHDs have proved difficult to automate. Autonomous hauling along pre-recorded paths and dumping are commercially available, but the loading sequence still relies on manual control. It has not yet been possible to reach the overall loading performance of experienced human operators. The best results have been achieved with admittance control [1]—where the forward throttle is coordinated with the bucket lift and curl—by using the measured dig force in order to fill the bucket while avoiding the slipping of the wheel. The motion and applied forces must be carefully adapted to the state of the muck pile, which may vary significantly because of variations in the material and as a consequence of previous loadings. Operators use visual cues to control the loading, in addition to auditory and tactile cues, when available. Between operators, the performance varies a lot, indicating the complexity of the bucket-filling task.

In this work, we explore the possibility of autonomous control of an LHD using deep reinforcement learning (DRL), which has been proven to be useful for visuomotor control tasks, such as grasping objects of different shapes. To test whether a DRL controller can learn the task of efficient loading while adapting to muck piles of variable states, we performed the following test. A simulated environment was created, which featured a narrow mine drift, dynamic muck piles of different initial shapes, and an LHD equipped with vision, motion, and force sensors like those available on real machines. A multi-agent system was designed to control the LHD to approach a mucking position, load the bucket, and break out from the pile. Reward functions were shaped for productivity and energy efficiency, as well as the avoidance of wall collisions and the slipping of the

wheels. The agent's policy network was trained using the soft actor–critic algorithm with a curriculum that was set up for the mucking agent. Finally, the controller was evaluated over 953 loading cycles.

2. Related Work and Our Contribution

The key challenges in the automation of earthmoving machines were recently described in [2]. An overview of the research from the late 1980s to 2009 can be found in [3]. Previous research has focused on computer vision in order to characterize piles [4–6], the motion planning of bucket-filling trajectories [4,7–9], and loading control. The control strategies can be divided into trajectory control [8,10] and force-feedback control [11,12]. The former aims to move the bucket along a pre-defined path, which is limited to homogeneous media, such as dry sand and gravel. The latter regulates the bucket's motion based on the feedback from the interactions of forces, thus making it possible to respond to inhomogeneities in material and loss of ground traction. One example of this is the admittance control scheme, which was proposed in [12] and refined and tested with good results at full scale in [13]. However, the method requires careful tuning of the control parameters, and it is difficult to automatically adjust the parameters to a new situation. This is addressed in [1] by using iterative learning control [14]. The target throttle and dump cylinder velocity are iteratively optimized to minimize the difference between the tracked filling of the bucket and the filling of the target . The procedure converges over a few loading cycles, and the loading performance is robust for homogeneous materials, but fluctuates significantly for fragmented rock piles.

Artificial intelligence (AI)-based methods include fuzzy logic for wheel-loader action selection [15] and digging control [16] by using feed-forward neural networks to model digging resistance and machine dynamics. Automatic bucket filling by learning from demonstration was recently demonstrated in [17–19] and extended in [20] with a reinforcement learning algorithm for automatic adaptation of an already-trained model to a new pile of different soil. The imitation model in [17] is a time-delayed neural network that predicts the lift and tilt actions of joysticks during the filling of a bucket; it was trained with 100 examples from an expert operator and used no information about the material or the pile. With the adaptation algorithm in [20], the network adapts from loading medium-coarse gravel to cobble gravel, with a five- to ten-percent increase in bucket filling after 40 loadings. The first use of reinforcement learning to control a scooping mechanism was recently published [21]. Using the actor–critic algorithm and the deep deterministic policy gradient algorithm, an agent was trained to control a three-degree-of-freedom mechanism in order to fill a bucket. The authors of [21] did not consider the use of high-dimensional observation data in order to adapt to variable pile shapes or to steer the vehicle. Reinforcement learning agents are often trained in simulated environments, as they are an economical and safe way to produce large amounts of labeled data [22] before transferring to real environments. In [23], this reduced the number of real-world data samples by 99% in order to achieve the same accuracy in robotic grasping. This is especially important when working with heavy equipment that must handle potentially hazardous corner cases.

Our article has several unique contributions to the research topic. The DRL controller uses high-dimensional observation data (from a depth camera) with information about the geometric shape of a pile, which is why a deep neural network is used. A multi-agent system is trained in cooperation, where one agent selects a favorable dig position while the other agent is responsible for steering the vehicle towards the selected position and for controlling the filling of the bucket while avoiding slipping of the wheels and collisions with the surroundings. The system is trained to become energy efficient and productive. The agents are trained over a sequence of loadings and can thereby learn a behavior that is optimal for repeated loading; e.g., they can avoid actions that will make the pile more difficult to load from at a later time.

3. Simulation Model

The simulated environment consisted of an LHD and a narrow underground drift with a muck pile (see Figure 1 and the Supplementary Video Material). The environment was created in Unity [24] by using a physics plugin, AGX Dynamics for Unity [25,26]. The vehicle model consisted of a multibody system with a drivetrain and compliant tires. The model was created from CAD drawings and data sheets of the ScoopTram ST18 from Epiroc [27]. It consisted of 21 rigid bodies and 27 joints, of which 8 were actuated. The mass of each body was set to match the 50-tonne vehicle's center of mass. Revolute joints were used to model the waist articulation, wheel axles, and bucket and boom joints; prismatic joints were used for the hydraulic cylinders with ranges that were set according to the data sheets.

Figure 1. Image from a simulation of the LHD scooping muck from a pile. The right image shows a top view in which the positions of the lidar (five rays) and camera are indicated.

The drivetrain consisted of a torque curve engine, followed by a torque converter and a gearbox. The power was then distributed to the four wheels through a central-, front-, and rear-differential system. The torque curve and gear ratios were set to match the data of the real vehicle. The hoisting of the bucket was controlled by a pair of boom cylinders, and the tilting was controlled by a tilt cylinder and Z-bar linkage. The hydraulic cylinders were modeled as linear motor constraints, and the maximum force was limited by the engine's current speed. That is, a closed-loop hydraulic system was not modeled.

The drift was 9 m wide and 4.5 m tall with a rounded roof. The muck pile was modeled using a multiscale real-time model [26] that combined continuum soil mechanics, discrete elements, and rigid multibody dynamics for realistic dig force and soil displacement.

The sample piles observed by the cameras on the vehicle are shown in Figure 2. The LHD was also equipped with 1D lidar, which measured the distance to the tunnel walls and the pile in five directions, as shown in Figure 1.

The muck was modeled with a mass density of 2700 kg/m^3, internal friction angle of 50°, and cohesion of 6 kPa. Natural variations were represented by adding a uniform perturbation of $\pm$200 kg/m^3 to the mass density, with random penetration force scaling ranges that were between 5 and 8 for each pile. Variations in the sample piles' shapes were ensured by saving the pile shape generations. After loading 10 times from an initial pile, the new shape was saved into a pool of initial pile shapes and tagged as belonging to generation 1. A state inheriting from generation 1 was tagged as generation 2, and so on.

The simulations ran with a 0.02 s timestep and the default settings for the direct iterative split solver in AGX Dynamics [25]. The grid size of the terrain was set to 0.32 m.

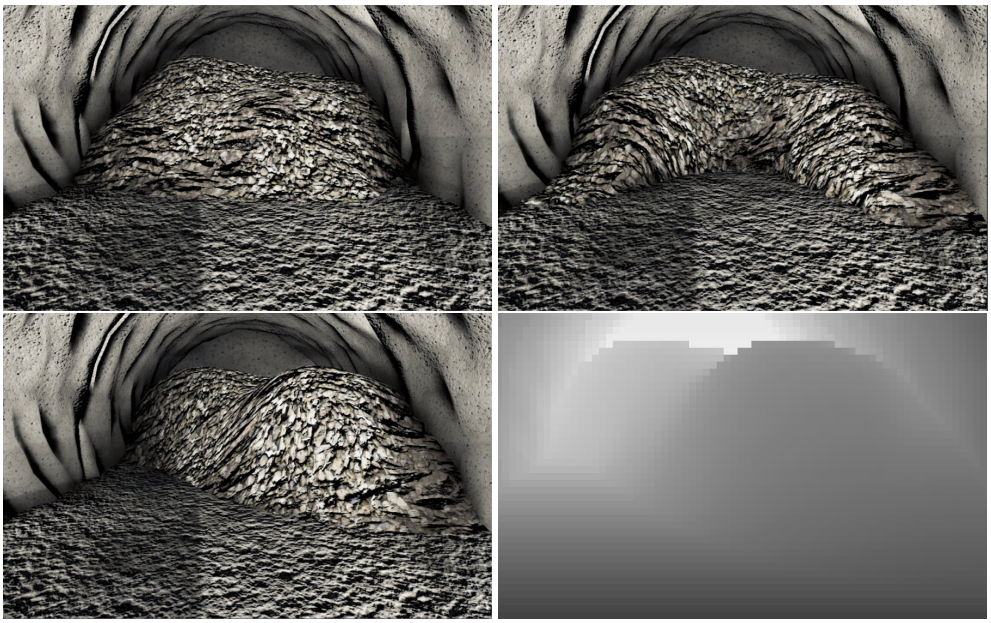

Figure 2. Sample camera images of the initial piles, referred to as convex (**upper left**), concave (**upper right**), and right-skewed (**lower left**); they are also shown in the depth camera view in the (**lower right**) image.

4. DRL Controller

The task of performing several consecutive mucking trajectories requires long-term planning of how the state of a muck pile develops. Training a single agent to handle both high-level planning and continuous motion control was deemed to be too difficult. Therefore, two cooperating agents were trained—a mucking position agent (MPA) and a mucking agent (MA). The former chose the best mucking position—once for each loading cycle—and the latter controlled the vehicle at 16 Hz to efficiently muck at the selected position. To train the agents, we used the implementation of the soft actor–critic (SAC) algorithm [28] in ML-Agents [29,30], which followed the implementation in Stable Baselines [31], and we used the Adam optimization algorithm. SAC is an off-policy DRL algorithm that trains the policy, π, to maximize a trade-off between the expected return and the policy entropy, as seen in the following augmented objective function:

$$J(\pi) = \sum_t \mathbb{E}_\pi \Big[r(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \log(\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)) \Big], \tag{1}$$

where r is the reward for a given state–action pair, $\boldsymbol{s}_t$ and $\boldsymbol{a}_t$, at a discrete timestep t. The inclusion of the policy entropy in the objective function with weight factor α (a hyperparameter) contributes to the exploration during training and results in more robust policies that generalize better to unseen states. Combined with off-policy learning, this makes the SAC a sample-efficient DRL algorithm, which will be important for the MPA, as explained below.

Figure 3 illustrates the relation between the two agents. The MPA's observations are from a depth camera at a fixed position in the tunnel with a resolution 84×44. The action is to decide on the lateral position in the tunnel at which to dig into the pile. This decision is made once per loading.

The MA's task is to complete a mucking cycle. This involves the following: approaching the pile to reach the mucking position while the vehicle is straight for maximum force and avoiding collision with the walls; filling the bucket by driving forward and gradually raising and tilting the bucket while avoiding slipping of the wheels; deciding to finish and break out by tilting the bucket to the maximum and holding it still. The MA observes the pile by using depth cameras, which were simulated by using a rendered depth buffer, and lidar data, which were simulated by using line-geometry intersection, in addition to a set of scalar observations consisting of the positions, velocities, and forces of the actuated

joints, the speed of the center shaft, and the lateral distance between the bucket's tip and the target mucking position. The scalar observations were stacked four times, meaning that we repeated observations from previous timesteps, thus giving the agent a short-term "memory" without using a recurrent neural network. The actions included the engine throttle, and the target velocities of each hydraulic cylinder controlled the lift, tilt, and the articulated steering of the vehicle. Note that the target velocity is not necessarily the actual velocity achieved by the actuator due to limited power available and resistance from the muck.

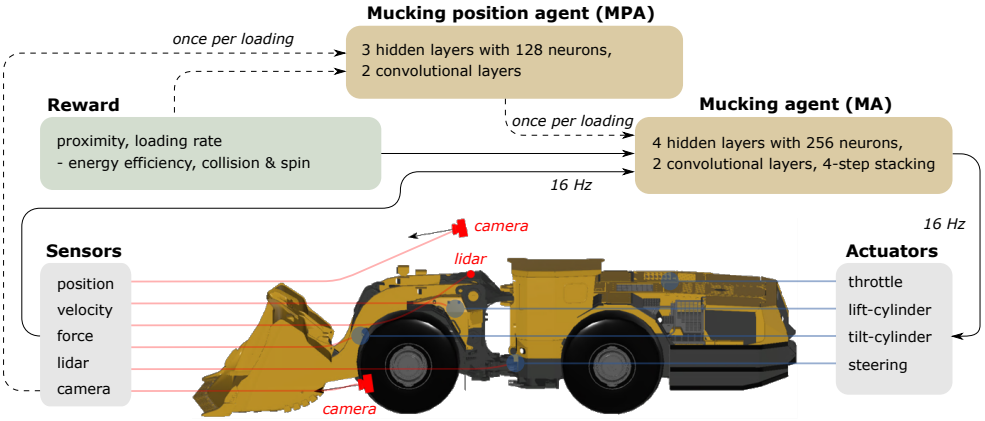

Figure 3. Illustration of the DRL controller and how the two agents—the MPA and MA—cooperate.

The MA's neural network consisted of four hidden fully connected layers of 256 neurons each. The depth camera data were first processed by two convolutional layers, followed by four fully connected layers, where the final layer was concatenated together with the final layer for the scalar observations. The MPA's neural network differed only in the fully connected layers, where we instead used three layers with 128 neurons each. Both the MA and the MPA were trained with a batch size of 256, a constant learning rate of 10^{-5}, a discount factor of $\gamma = 0.995$, an update of the model at each step, and the initial value of the entropy coefficient α in Equation (1) set to $\alpha = 0.2$. The buffer size was reduced from 3×10^5 for the MA and to 10^5 for the MPA. These network designs and hyper-parameters were found by starting from the default values in the ML agents and testing for changes that led to better training. There was probably room for further optimization.

The MA reward function depends on how well the selected mucking position was achieved, the rate of bucket filling, and the work exerted by the vehicle. The complete reward function is

$$r_t^{\mathrm{MA}} = w_1 C W r_t^{\mathrm{p}} r_t^{\mathrm{l}} - w_2 p_t^{\mathrm{w}}, \tag{2}$$

where $r_t^{\mathrm{p}} = 1 - \min\left(\Delta x / 4\mathrm{m}, 1\right)^{0.4}$ decays with the lateral deviation of Δx from the target mucking position, and $r_t^{\mathrm{l}} = l_t - l_{t-1}$ is the increment in the bucket's filling fraction l over the last timestep. If the vehicle came into contact with the wall or if a wheel was spinning, this was registered with two indicators that were set to $C = 0$ and $W = 0$, respectively. Otherwise, these had a value of 1. The work p done by the engine and the hydraulic cylinders for tilting, lifting, and steering since the previous action was summed to $p_t^{\mathrm{w}} = (p_{\mathrm{tilt}} + p_{\mathrm{lift}} + p_{\mathrm{steer}} + p_{\mathrm{engine}}/5)$. The weight factors $w_1 = 100$ and $w_2 = 10^{-6}\,\mathrm{J}^{-1}$ were set to make the two terms in Equation (2) dimensionless and similar in size. The reward favored mucking at the target position and the increase in the volume of material in the bucket compared to the previous step while heavily penalizing collision with walls and spinning of the wheels. Finally, subtracting the current energy consumption encouraged energy-saving strategies and mitigated unnecessary movements that did not increase the possibility of current or future material in the bucket.

Additionally, a bonus reward was given if the agent reached the end state. The bonus reward was the final position reward multiplied by the final bucket filling fraction:

$$r_T = 10 r_T^{\mathrm{p}} l_T. \tag{3}$$

This encouraged the agent to finish the episode when it was not possible to fill the bucket any further, except by reversing and re-entering the pile.

The MPA received a reward for each loading completed by the MA. The reward was the final filling fraction achieved by the mucking agent multiplied by a term that promoted keeping a planar pile surface, $s_t = e^{-d^2/d_0^2}$, where d is the longitudinal distance between the innermost and outermost points on the edge of the pile, as shown in Figure 1, and $d_0 = \sqrt{2}$ m. The reward became

$$r_t^{\mathrm{MPA}} = l_t s_t. \tag{4}$$

Due to the comparatively slow data collection, it is important to use a sample-efficient DRL algorithm. The two agents were trained in two steps—first separately and then together. The mucking agent was trained with a curriculum setup, using three different lessons. Each lesson increased the number of consecutive loading trajectories by five—from 10 to 20—and the number of pile shape generations by 1—from 0 to 2. During pre-training, each trajectory targeted a random mucking position on the pile. The MPA was first trained with the pre-trained MA without updating the MA model. When the MPA showed some performance, we started updating the MA model together with the MPA model. The model update rate for both models was then limited by the MPA model.

With the SAC, the model was updated at each loading action (16 Hz). The computational time for doing this was similar to that of running the simulation, which was roughly in real time. Hence, training over 30 million steps corresponded to roughly 500 CPU hours. We did not explore the possibility of speeding up the training process by running multiple environments in parallel.

5. Results

We measured the performance of the DRL controller in three different configurations, which are denoted as A, B, and C. Controller A combined the MPA and MA with the reward functions (2) and (4), while controllers B and C mucked at random positions instead of using the MPA. For comparison, controller C did not include the penalty for energy usage in Equation (2). The training progress is summarized in Figure 4. Controllers B and C were trained for the same number of steps and using the same hyper-parameters, while A continued to train from the state at which B ended.

For each controller, we collected data from 953 individual loadings. Each sample was one out of twenty consecutive loadings starting from one of the four initial piles. The pile was reset after twenty loadings. During testing, the muck pile parameters are set to the default values, except for the mass density, which was set to the upper limit so that one full bucket was about 17.5 t. The loaded mass was determined from the material inside the convex hull of the bucket. This ensured that we did not overestimate the mass in the bucket by including material that might fall off when backing away from the pile.

The mucking agents learned the following behavior, which is also illustrated in the Supplementary Video Material. The mucking agent started by lining up the vehicle towards the target mucking position, with the bucket slightly raised from the ground and tilted upwards. When the vehicle approached the mucking pile, the bucket was brought flat to the ground and driven into the pile at a speed of approximately 1.6 m/s. Soon after entering the pile, the MA started lifting and tilting the bucket. This increased the normal force, which provided traction in order to continue deeper into the pile without the wheels slipping. After some time, the vehicle started breaking out of the pile with the bucket fully tilted.

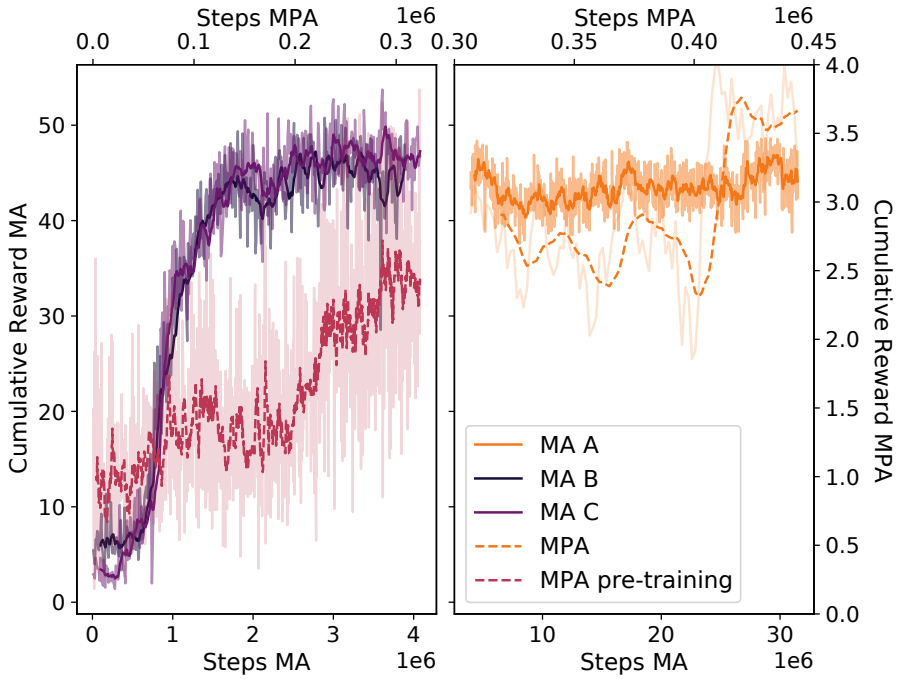

Figure 4. The cumulative reward for the different agents during training. The left panel shows the reward when the MA and MPA were trained separately. In the right panel, the training is continuous with the agents trained together. Three variants (A, B, and C) of the MA were tested, and their differences are explained in Section 5.

The mass loaded into the bucket at the end of each episode is plotted in Figure 5 with the corresponding episode duration and energy use. The majority of the trajectories are clustered in the upper-left corner of high productivity (t/s). Each of the three controllers had some outliers, loadings with low mass, and/or long loading times. If the mucking agent did not reach the final state of breaking out with the bucket fully tilted within 48 s, the loading was classified as failed. The common reason for timing out was the vehicle being stuck in the pile with a load that was too large to break out. Controller C, which was not rewarded for being energy efficient, had a longer loading duration and was more prone to getting stuck.

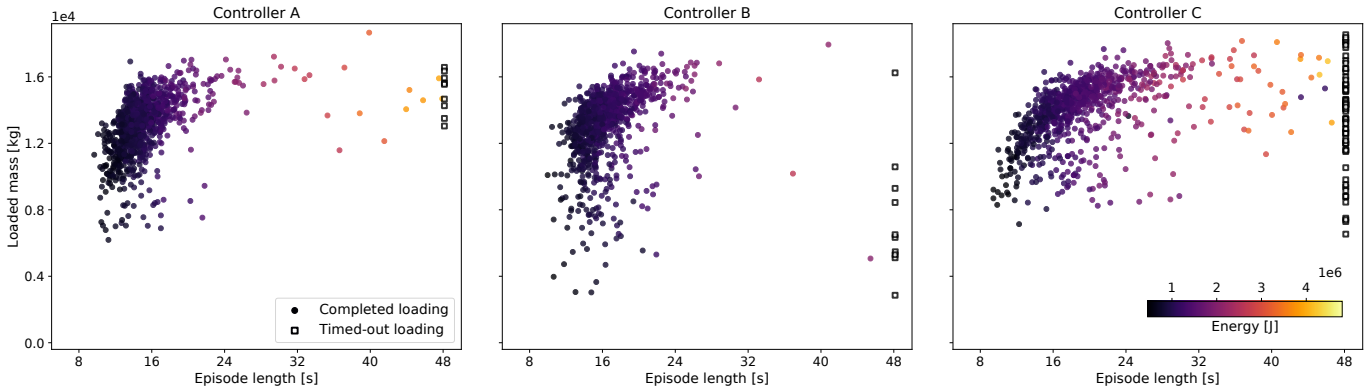

Figure 5. The performance of the three tested controllers over 953 loading cycles.

The mean productivity and energy usage for controllers A, B, and C are presented in Table 1. For reference, we have also added the performance of an experienced driver manually operating an ST14 loader [13], with the loaded mass re-scaled to the relative bucket volume of the ST18 loader, $V_{ST18}/V_{ST14} = 1.23$, while keeping the loading time and energy per unit mass the same. The failure ratio and mucking position error are

also included. Comparing controllers B and C, we observe that the energy penalty in the reward functions leads to 21% lower energy consumption, a 7% increase in productivity, and a reduction of the failure ratio from 7% to 1%. We conclude that controllers A and B learned a mucking strategy that actively avoided entering states that would lead to high energy consumption and delayed breaking out. On average, the loaded mass was between 75 and 80% of the vehicle's capacity of 17.5 t. The energy usage was 1.4–1.9 times larger than the energy required for raising the mass by 4 m. This is reasonable considering the soil's internal friction and cohesion and that the internal losses in the LHD were not accounted for.

Controller A had 9% higher productivity than controller B, which mucked at random positions, and used 4% less energy. This suggests that the mucking position agent learned to recognize favorable mucking positions from the depth images. Confirming this, Figure 6 shows the distribution of the mucking positions for the four different initial piles that were shown in Figure 2. For the left-skewed pile, the MPA chose to muck more times on the left side of the tunnel, presumably because that action allowed the filling of the bucket with a smaller digging resistance, and it left the pile in a more even shape. The same pattern was true for the pile that was skewed to the right. For the concave pile, the majority of the targets were at the left or right edge, effectively working away the concave shape. For the convex shape, the mucking positions were more evenly distributed. Figure 7 shows the evolution of the loaded mass when using controller A over a sequence of 20 repeated muckings from the initial piles. There was no trend of degradation of the mucking performance. This shows that the MPA acted to preserve or improve the state of the pile and that the DRL controller was generalized to the spaces of different mucking piles.

The target mucking position chosen by the MPA was not guaranteed to be the position at which the LHD actually mucked, as we expected an error in MA's precision, and it was able to weigh the importance of mucking at the target position against the importance of filling the bucket with a low energy cost. The mean difference between the target and actual mucking positions for controller A was $0.014 \pm (0.15)$ m, which was 0.4% of the bucket's width.

Compared to the manual reference included in Table 1, the DRL controllers reached a similar or slightly larger average loading mass, but with higher consistency (less variation) and shorter loading times. The productivity of controller A was 70% higher than that of the manual reference, with an energy consumption that was three times smaller. On the other hand, the DRL controller underperformed relative to the automatic loading control (ALC) in ST14 field tests [13]. The ALC reached its maximal loading mass with a productivity that was roughly two times greater than that of the DRL controller, but with four times larger energy consumption per loaded tonne. There are several uncertainties in these comparisons, however. The performance of the DRL controller was measured over a sequence of twenty consecutive loadings from the same pile in a narrow drift, thus capturing the effect of a potentially deteriorating pile state. It is not clear whether the ALC field test in [13] was conducted in a similar way, or if the loadings were conducted on piles prepared in a particular state. There are also uncertainties regarding the precise definition of the loading time and whether the assumed scaling from ST14 to ST18 holds. In future field tests, the amount of spillage on the ground and the slipping of the wheels should also be measured and taken into account when comparing the overall performance. Comparison with [21] and [18] is interesting but difficult because of the large differences in the vehicles' size and strength and the materials' properties. The reinforcement learning controller in [21] achieved a fill factor of 65%, and the energy consumption was not measured. The neural network controller in [18], which was trained by learning from demonstration, reached 81% of the filling of the bucket relative to manual loading, but neither loading time nor work was reported.

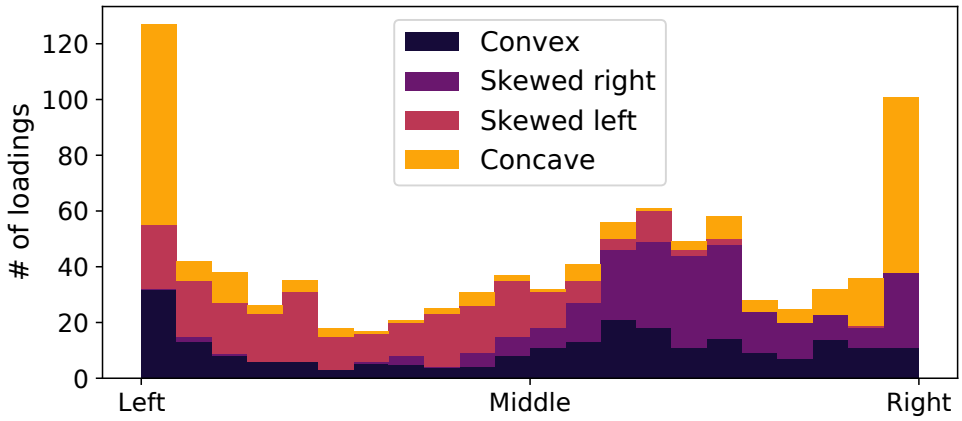

Figure 6. The distribution of mucking positions in the tunnel selected by the MPA for each of the four initial piles.

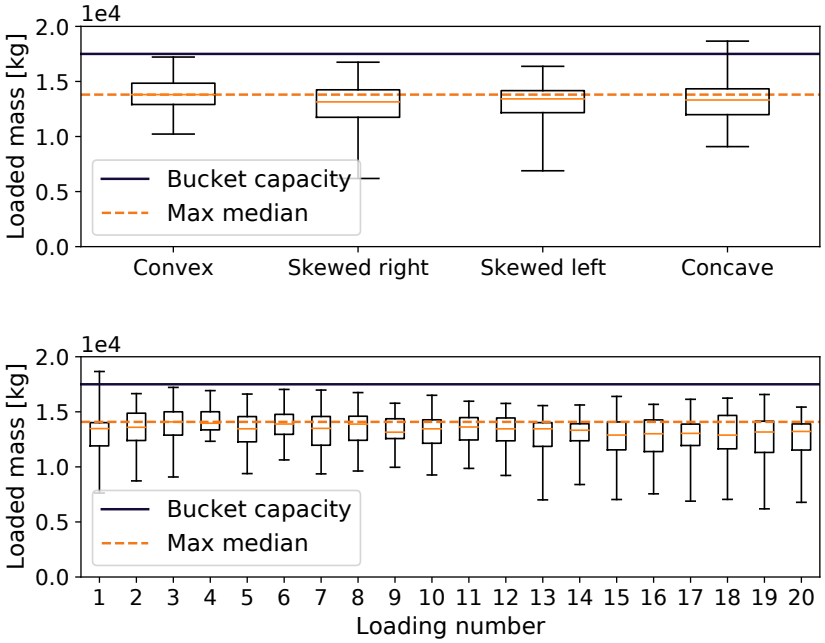

Figure 7. Boxplot of the loaded mass during repeated mucking when using controller A. The whiskers show the maximum and minimum values.

Table 1. Mean performance for the three DRL controllers and manual loading performance [13] (re-scaled from ST14 to ST18 for reference).

Controller		Mass [t]	Productivity [t/s]	Energy Usage [MJ]	Position Error [m]	Failure
A	all loadings	13.2 (±1.8)	0.87 (±0.15)	1.10 (±0.5)	0.01 (±0.15)	1%
	completed loadings	13.2 (±1.8)	0.88 (±0.14)	1.08 (±0.39)	0.01 (±0.15)	
B	all loadings	13.1 (±2.3)	0.80 (±0.16)	1.12 (±0.26)	0.11 (±0.20)	1%
	completed loadings	13.1 (±2.2)	0.81 (±0.14)	1.12 (±0.24)	0.11 (±0.20)	
C	all loadings	14.0 (±2.0)	0.73 (±0.20)	1.55 (±0.75)	0.08 (±0.23)	7%
	completed loadings	14.0 (±1.9)	0.76 (±0.16)	1.42 (±0.53)	0.08 (±0.23)	
Manual reference		13.0 (±5.9)	0.52 (±0.30)	3.3 (±1.5)		

6. Discussion

The method explored here has several limitations. The episodes ended when the bucket broke out of the pile. Consequently, the mucking agent was not given the opportunity to learn the technique of reversing and re-entering the pile if the bucket is under-filled. This is often practiced by operators. The soil's strength and mass density were homogeneous in the environment, but real muck may be inhomogeneous. Finally, it is not possible to predict how the controller will react to situations that significantly deviate from those included in the training—e.g., with piles of very different sizes or shapes, the presence of boulders, or obstacles of various kinds.

7. Conclusions

We found that it is possible to train a DRL controller to use high-dimensional sensor data from different domains without any pre-processing as input for neural network policies in order to solve the complex control task of repeated mucking with high performance. Even though the bucket is seldom filled to the maximum, it is consistently filled with a good amount. It is common for human operators to use more than one attempt to fill the bucket when the pile's shape is not optimal, which is a tactic that was not considered here.

The inclusion of an energy consumption penalty in the reward function can distinctly reduce the agent's overall energy usage. This agent learns to avoid certain states that inevitably lead to high energy consumption and the risk of failing to complete the task. The two objectives of loading more material and using less energy were competing, as seen in Table 1 with controller C's higher mass and controller A's lower energy usage. This is essentially a multi-objective optimization problem that will have a Pareto front of possible solutions with different prioritizations between the objectives. In this work, we prioritized by choosing the coefficients $w_1 = 100$ and $w_2 = 10^{-6}$ in Equation (2), but future work could analyze this trade-off in more detail.

A low-resolution depth camera observation of a pile's surface is enough to train the MPA to recognize good 1D positions for the MA to target. By choosing the target position, the overall production increases and the energy consumption decreases compared to when random positions are targeted. Continuing to clear an entire pile by using this method—essentially cleaning up and reaching a flat floor—could be an interesting task for future work.

Supplementary Materials: The following is available online at https://www.mdpi.com/article/10.3390/machines9100216/s1, Video S1: Supplementary Video.

Author Contributions: Conceptualization, K.B. and M.S.; methodology, D.L. and K.B.; software, D.L., J.M. and S.B; formal analysis, D.L., K.B., M.S. and S.B.; resources, H.L.; writing—original draft preparation, D.L., K.B. and M.S.; writing—review and editing, S.B.; visualization, D.L., S.B. and J.M. All authors have read and agreed to the published version of the manuscript.

Funding: This project was funded in part by by VINNOVA (grant id 2019-04832) in the Integrated Test Environment for the Mining Industry (SMIG) project.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: S.B., K.B., D.L., J.M. and M.S. are affiliated with the company developing the simulation software aplied in the paper. H.L. is affiliated with the company developing the mining equipment which automation is researched here.

References

1. Fernando, H.; Marshall, J.; Larsson, J. Iterative Learning-Based Admittance Control for Autonomous Excavation. *J. Intell. Robot. Syst.* **2019**, *96*, 493–500. [CrossRef]
2. Dadhich, S.; Bodin, U.; Andersson, U. Key challenges in automation of earth-moving machines. *Autom. Constr.* **2016**, *68*, 212–222. [CrossRef]
3. Hemami, A.; Hassani, F. An overview of autonomous loading of bulk material. In Proceedings of the 26th ISARC, Austin, TX, USA, 24–27 June 2009; pp. 405–411.
4. Sarata, S.; Weeramhaeng, Y.; Tsubouchi, T. Approach Path Generation to Scooping Position for Wheel Loader. In Proceedings of the 2005 IEEE International Conference on Robotics and Automation, Barcelona, Spain, 18–22 April 2005; pp. 1809–1814. [CrossRef]
5. Magnusson, M.; Almqvist, H. Consistent Pile-shape quantification for autonomous wheel loaders. In Proceedings of the 2011 IEEE/RSJ International Conference on Intelligent Robots and Systems, San Francisco, CA, USA, 25–30 September 2011; pp. 4078–4083. [CrossRef]
6. McKinnon, C.; Marshall, J. Automatic identification of large fragments in a pile of broken rock using a time-of-flight camera. *IEEE Trans. Autom. Sci. Eng.* **2014**, *11*, 935–942. [CrossRef]
7. Singh, S.; Cannon, H. Multi-resolution planning for earthmoving. In Proceedings of the 1998 IEEE International Conference on Robotics and Automation (Cat. No. 98CH36146), Leuven, Belgium, 20 May 1998; Volume 1, pp. 121–126.
8. Filla, R.; Frank, B. Towards Finding the Optimal Bucket Filling Strategy through Simulation. In Proceedings of 15th Scandinavian International Conference on Fluid Power, Linköping, Sweden, 7–9 June 2017. [CrossRef]
9. Lindmark, D.; Servin, M. Computational exploration of robotic rock loading. *Robot. Auton. Syst.* **2018**, *106*, 117–129. [CrossRef]
10. Koyachi, N.; Sarata, S. Unmanned loading operation by autonomous wheel loader. In Proceedings of the 2009 ICCAS-SICE, Fukuoka City, Japan, 18–21 August 2009; pp. 2221–2225.
11. Richardson-Little, W.; Damaren, C.J. Position accommodation and compliance control for robotic excavation. *J. Aerosp. Eng.* **2008**, *21*, 27–34. [CrossRef]
12. Marshall, J.A.; Murphy, P.F.; Daneshmend, L.K. Toward Autonomous Excavation of Fragmented Rock: Full-Scale Experiments. *IEEE Trans. Autom. Sci. Eng.* **2008**, *5*, 562–566. [CrossRef]
13. Dobson, A.; Marshall, J.; Larsson, J. Admittance Control for Robotic Loading: Design and Experiments with a 1-Tonne Loader and a 14-Tonne Load-Haul-Dump Machine. *J. Field Robot.* **2017**, *34*, 123–150. [CrossRef]
14. Bristow, D.A.; Tharayil, M.; Alleyne, A.G. A survey of iterative learning control. *IEEE Control Syst. Mag.* **2006**, *26*, 96–114. [CrossRef]
15. Lever, P. An automated digging control for a wheel loader. *Robotica* **2001**, *19*, 497. [CrossRef]
16. Wu, L. A Study on Automatic Control of Wheel Loaders in Rock/Soil Loading. Ph.D. Thesis, The University of Arizona, Tucson, AZ, USA, 2003.
17. Dadhich, S.; Sandin, F.; Bodin, U.; Andersson, U.; Martinsson, T. Field test of neural-network based automatic bucket-filling algorithm for wheel-loaders. *Autom. Constr.* **2019**, *97*, 1–12. [CrossRef]
18. Halbach, E.; Kämäräinen, J.; Ghabcheloo, R. Neural Network Pile Loading Controller Trained by Demonstration. In Proceedings of the 2019 International Conference on Robotics and Automation (ICRA), Montreal, QC, Canada, 20–24 May 2019; pp. 980–986. [CrossRef]
19. Yang, W.; Strokina, N.; Serbenyuk, N.; Ghabcheloo, R.; Kämäräinen, J. Learning a Pile Loading Controller from Demonstrations. In Proceedings of the 2020 IEEE International Conference on Robotics and Automation (ICRA), Paris, France, 31 May–31 August 2020; pp. 4427–4433. [CrossRef]
20. Dadhich, S.; Sandin, F.; Bodin, U.; Andersson, U.; Martinsson, T. Adaptation of a wheel loader automatic bucket filling neural network using reinforcement learning. In Proceedings of the 2020 International Joint Conference on Neural Networks (IJCNN), Glasgow, UK, 19–24 July 2020; pp. 1–9. [CrossRef]
21. Azulay, O.; Shapiro, A. Wheel Loader Scooping Controller Using Deep Reinforcement Learning. *IEEE Access* **2021**, *9*, 24145–24154. [CrossRef]
22. Kiran, B.R.; Sobh, I.; Talpaert, V.; Mannion, P.; Sallab, A.A.A.; Yogamani, S.; Pérez, P. Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Trans. Intell. Transp. Syst.* **2021**, 1–18. [CrossRef]
23. James, S.; Wohlhart, P.; Kalakrishnan, M.; Kalashnikov, D.; Irpan, A.; Ibarz, J.; Levine, S.; Hadsell, R.; Bousmalis, K. Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 15–20 June 2019; pp. 12627–12637.
24. Unity Technologies. Unity. Available online: https://unity.com (accessed on 1 March 2021).
25. Algoryx. AGX Dynamics. Available online: https://www.algoryx.se/agx-dynamics/ (accessed on 19 October 2020).
26. Servin, M.; Berglund, T.; Nystedt, S. A multiscale model of terrain dynamics for real-time earthmoving simulation. *Adv. Model. Simul. Eng. Sci.* **2020**, *8*, 1–35. [CrossRef]
27. Epiroc. ScoopTram ST18—Specification. Available online: https://www.epiroc.com/content/dam/epiroc/underground-mining-and-tunneling/lhd/diesel-scooptram/st18/9869%200065%2001_scooptram%20st18_technical%20specification_english_High%20Res.pdf (accessed on 10 October 2020).

28. 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 35th International Conference on Machine Learning; Stockholm, Sweden, 10–15 July 2018; pp. 1861–1870.
29. Juliani, A.; Berges, V.P.; Teng, E.; Cohen, A.; Harper, J.; Elion, C.; Goy, C.; Gao, Y.; Henry, H.; Mattar, M.; et al. Unity: A General Platform for Intelligent Agents. *arXiv* **2018**, arXiv:1809.02627.
30. Juliani, A.; Berges, V.P.; Teng, E.; Cohen, A.; Harper, J.; Elion, C.; Goy, C.; Gao, Y.; Henry, H.; Mattar, M.; et al. ML-Agents Github Repository. Available online: https://github.com/Unity-Technologies/ml-agents (accessed on 15 September 2021).
31. Hill, A.; Raffin, A.; Ernestus, M.; Gleave, A.; Kanervisto, A.; Traore, R.; Dhariwal, P.; Hesse, C.; Klimov, O.; Nichol, A.; et al. Stable Baselines. Available online: https://github.com/hill-a/stable-baselines (accessed on 15 September 2021).