IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 12, 2021, accepted January 20, 2021, date of publication February 2, 2021, date of current version February 10, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3056625

Wheel Loader Scooping Controller Using Deep Reinforcement Learning

OSHER AZULAY[1] AND AMIR SHAPIRO[2], (Member, IEEE)
[1]School of Mechanical Engineering, Tel Aviv University, Tel Aviv 69978, Israel
[2]Department of Mechanical Engineering, Ben-Gurion University of the Negev, Be'er Sheva 84105, Israel

Corresponding author: Osher Azulay (osherazulay@mail.tau.ac.il)

This work was supported by the Israeli Ministry of Defense through Project ROBIL under Grant 87726311.

ABSTRACT This article presents a deep reinforcement learning-based controller for an unmanned ground vehicle with a custom-built scooping mechanism. The robot's aim is to autonomously perform earth scooping cycles with three degrees of freedom: lift, tilt and the robot's velocity. While the majority of previous studies on automated scooping processes are based on data recorded by expert operators, we present a method to autonomously control a wheel loader to perform the scooping cycle using deep reinforcement learning methods without any user-provided demonstrations. The controller's learning approach is based on the actor-critic, Deep Deterministic Policy Gradient algorithm which we use to map online sensor data as input to continuously update the actuator commands. The training of the scooping policy network is done solely in a simplified simulation environment using a virtual physics engine, which converges to an average of a 65% fill factor from the full bucket capacity and a 5 [*sec*] average cycle time. We illustrate the performance of the trained policy in simulations and in real-world experiments with 3 different inclination angles of the earth. An additional scooping experiment compared the performance of our controller to remote manual human control. Overall, the deep reinforcement learning-based controller exhibited good performance in terms of both achieved visually bucket fill with varying scooped earth weights of $4.1-7.2[kg]$, and a $5.1-7.1[sec]$ cycle time. The experimental results confirm the ability of our planner to fill bucket as required, indicating that our controller can be used for excavation purposes.

INDEX TERMS Robotics in construction, machine learning, agricultural automation.

I. INTRODUCTION

Earthmoving systems are currently used in a variety of industries, but especially in the construction and agricultural domains. They have numerous advantages including economic efficiency, safety, and availability. Earthmoving machinery typically refers to heavy-duty vehicles designed for construction operations that involve earthworks. While earthmoving machinery continues to develop, most of the excavation cycle is still controlled by a human, either directly or via teleoperation [1]. Designing control methods for such tasks is a long-standing research goal, which has attracted considerable interest and generated a number of survey papers [2], [3]. The general approaches to autonomous excavation exploit the machine dynamics and try to follow a defined trajectory [4], [5], use compliance force control [6], [7] or employ a behavior-based approach for motion control [8], [9]. Since some tasks are more complex than others, they often require extensive engineering experience and tedious manual tuning beyond the control algorithm itself.

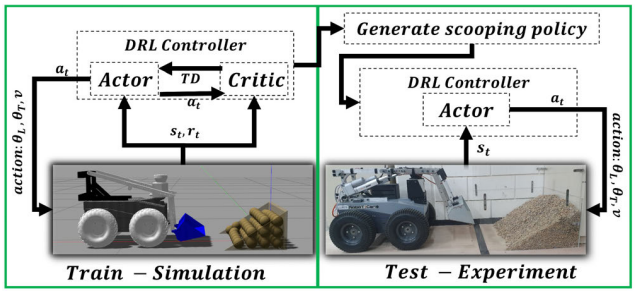

FIGURE 1. A deep reinforcement learning controller was trained in simulation to accomplish the scooping cycle using 12-dimensional sensors inputs into three actuator commands.

One of the key challenges in deploying automated earth moving machines relates to the analysis of the soil-tool interaction, due to the unpredictable nature of the soil [6].

The associate editor coordinating the review of this manuscript and approving it for publication was Yangmin Li.

Most approaches require accurate models of the machine which makes them liable to modeling errors, wear and tear, and changing conditions [10]. Another drawback is that the majority of work done to automate scooping processes is based on data recorded by expert operators, which limits the efficiency to the level of the operator's skills [1]. Since the scooping cycle is a repetitive and dynamic task involving intricate interactions with the environment, here a new strategy is put forward to solve the problem of autonomous bucket scooping automation for wheel loaders using Reinforcement Learning (RL) methods. RL is a machine learning technique that has the potential to enable robots to learn large repertoires of behavioral skills with minimal human intervention through trial and error. The main advantage of RL is that it does not require a predefined control structure and can explore the environment to find a good policy to follow, thus mitigating the shortcoming of having to explicitly derive the interconnection with the environment. Deep Reinforcement Learning (DRL) methods have been successfully applied in many complex robotic tasks from manipulation [11] to autonomous vehicles [12], and aerial applications [13]. However, practical real-world applications of RL are relatively rare since they often require unrealistic learning times, high sample complexity, and can potentially damage the robot [14], [15].

This article focuses on developing a scooping motion planner that learns transferable scooping policies solely under simulation. We show that an earthmoving machine can be fully controlled using DRL methods trained solely in simulation and then deployed on a real Unmanned Ground Vehicle (UGV). The complete system architecture is depicted in Fig. 1. To further increase the robustness of the system and narrow the reality gap, we defined a compact design of the observation space and performed detailed system identification. The controller is trained using the model-free, actor-critic Deep Deterministic Policy Gradient (DDPG) algorithm [16], which successfully completes scooping cycles without any user-provided demonstrations. We developed a ROS-Gazebo based open-source training environment (available at [17]) for the agent to learn scooping motions. To verify the learning algorithm and simulation viability, we deployed the controller in a real-world scenario. We also compared the performance of our DRL based controller against manual human control for an earthmoving cycle.

The primary contribution of this paper is its novel end-to-end Neural Network (NN) based controller for a robotic wheel loader that autonomously scoops earth from a pile, with three degrees of freedom (DoF). The controller is trained solely via simulation without any prior knowledge using DRL. To the best of our knowledge, this is the first 3-DoF DRL scooping controller to learn from simulation without previous knowledge and be deployed on a real-life excavation machine. In addition, the development of an open source [17] training environment provides the research community with a tool to collect data and observe progress performance for excavation purposes.

The remainder of this paper is structured as follows. Section II further discusses related work in this area, followed by a background review in Section III. Section IV describes the system design and Section V introduces the proposed scooping controller implementation. We demonstrate our simulation and report the experimental results in Section VI, followed by conclusions and suggested future research in Section VII.

II. RELATED WORKS

Most previous research on automating bucket-filling processes are based on data recorded by expert operators and implement a control system that can be generally divided into three main categories. The first is made up of position control algorithms for bucket motion trajectories, which are intended to maximize the volume scooped by the bucket [4], [5], [18], [19]. These approaches have been successful, but they rely on expert trajectories and do not generalize to different machine-pile environments. The second category is composed of compliance control algorithms, such as soil estimation based methods [6], [7] and force/torque based methods [20], [21]. Soil estimation methods predict the soil-tool interaction force and apply a heuristic-based motion by modifying the soil parameters in the program. The drawbacks of soil estimation methods include the fact that they lack precision, fail to capture the rapidly changing properties in the environments or are non-real-time capable and thus not feasible/implementable [20]. Others combine these methods with a higher-level planner. For instance, a force-control trajectory controller with prioritized tasks [20], a coarse and fine planner that ensures equal performance over a large number of digs [22], or adding a disturbance observer to compensate for the difference between prediction and actual performance [21], [23]. These compliance control methods do not follow a desired trajectory but rather apply specific forces to the pile during the scooping motion. The third category is made up of control algorithms that employ a behavior-based approach for motion control, such as a rule-based algorithm that depends on the current phase and acts dynamically [8], [9]. Thus, most previous solutions to automate the scooping task (1) do not generalize to different machines or pile environments (2) rely on prior knowledge of an expert operator and (3) require accurate models of the machine and therefore are susceptible to failure in the presence of modeling errors, wear and tear, and changing conditions [10]. This underscores the need for a generic automatic scooping solution that can be adapted to different scenarios.

Few deep Learning methods have been implemented in earthmoving operations, especially in bucket-filling processes, mainly due to the task complexity. The challenges include various problems such as simulating the environment, data collection, and soil-tool dynamics. Dadhich *et al.* [10], [24] demonstrated the implementation of a NN approach to control bucket motion during real-life

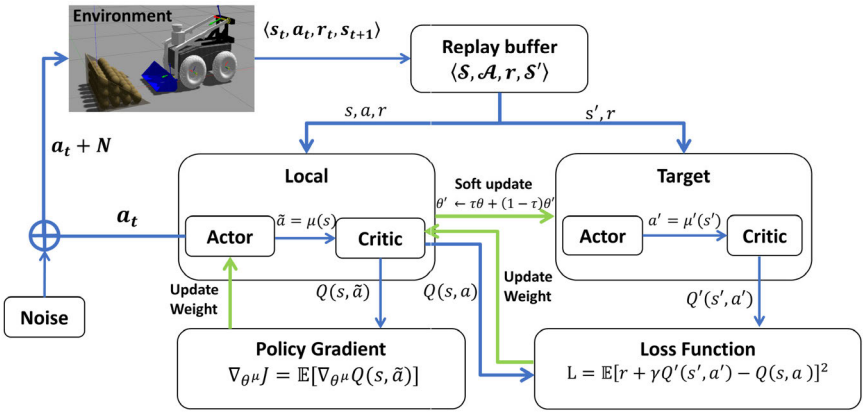

FIGURE 2. DDPG algorithm schematic.

loading processes. They used data collected from an expert operator to train a time-delayed NN to perform a bucket-filling motion with two degrees of freedom (static wheels). Halbach *et al.* [25] implemented a shallow three-DoF NN controller for automated pile loading with a robotic wheel loader that used learning from the demonstration of sensor recordings. Both of these approaches were successful in terms of the bucket filling cycle, but nevertheless rely on the prior knowledge of expert operator data, which limits the efficiency to the operator's skills which are not always available.

III. PRELIMINARIES

A. EARTHMOVING CYCLE

A wide range of earth-moving machines are currently used in industry, the most common of which are wheel loaders and excavators. Excavators are used to dig into the ground whereas wheel loaders are used to load and transport excavated material. An earthmoving cycle is comprised of several steps: 1) navigating towards the pile; 2) scooping up the earth; 3) navigating to the dump location; and 4) dumping. This paper focuses on automating the bucket-filling process, which is the most complex phase. The filling of a bucket is a complex granular flow problem that can be divided into three general phases: *approach, fill,* and *exit* [3]. Each phase operates under different conditions in terms of vehicle feedback, terrain properties, and duration. In the first phase, *approach*, the front-loader moves towards the pile of earth, and the operator chooses the height and penetration angle of the front loader's bucket. In the second phase, *fill*, there is a simultaneous change in the lift, tilt, and throttle actions to navigate the bucket tip through the earth pile while avoiding wheel slip and piston stall. The last phase, *exit*, involves tilting the bucket until the breakout, while the machine moves in reverse from the pile.

B. REINFORCEMENT LEARNING

In RL, the problem is typically characterized as a Markov Decision Processes (MDP), which defined by the tuple $\langle \mathcal{S}, \mathcal{A}, r, \mathcal{P}, \gamma \rangle$. S represents the state-space, and A is the action-space. Transitions between states are performed with transition probability $p\,(s_t|s_t, a_t) \in \mathcal{P}$, reward $r\,(s_t, a_t)$ and a discount factor $\gamma \in [0, 1]$. At each time step, the agent observes the current state $s_t \in \mathcal{S}$ and takes action $a_t \in \mathcal{A}$ according to the policy π_θ, which can be stochastic or deterministic. While interacting with the environment, the system transitions to a new state $s_{t+1} \in \mathcal{S}$ and the agent receives a reward $r_t\,(s_t, a_t, s_{t+1})$. The sequence of state-action pairs defines the trajectory $\tau = (s_0, a_0, s_1, a_1 \ldots\ldots s_H, a_H)$ of length H. The return R_t^γ is the total discounted reward for the initial state over the trajectory $R_t^\gamma = \sum_{i=t}^{H} \gamma^{i-t} r\,(s_i, a_i)$. The goal in reinforcement learning is to learn a policy which maximizes the expected return from the initial distribution $J = \mathbb{E}\left[R_1^\gamma|\pi\right]$. The state-value function is defined as the expectation value of the return over all allowed trajectories from a specific state s, $V^\pi\,(s_t) = \mathbb{E}_\pi\left[R_t^\gamma|s_t\right]$. While the action-value function describes the expected return after taking an action a_t in state s_t, $Q^\pi\,(s_t, a_t) = \mathbb{E}_\pi\left[R_t^\gamma|s_t, a_t\right]$.

DDPG is an actor-critic off-policy gradient algorithm that implements a stochastic behavior policy and estimates a deterministic target policy [16]. A schematic of the procedure can be seen in Fig. 2. DDPG primarily uses two neural networks in its learning phase, one for the actor (policy network) and one for the critic (value network), with weights θ^μ and θ^Q, respectively. The actor network is used to approximate the optimal policy whereas the critic utilizes the value-based approach to estimate the value of state-action pairs. The actor function $\mu\,(s_t|\theta^\mu)$ deterministically maps states for specific actions. The critic's output is the estimated Q-value of the current state and action provided by the actor $Q\left(s_t, a_t|\theta^Q\right)$. These networks compute action predictions for the current state and generate the Temporal-Difference (TD) error at each time step while using a set of target networks $\mu'\,(s_t)$, $Q'\,(s_t, a_t)$, with weights $\theta^{\mu'}$ and $\theta^{Q'}$ respectively. The critic's loss function L, and target y_t are computed from the sum of the immediate reward and the outputs of the target actor and critic networks:

$$y_i = r_i + \gamma Q'\left[s_{i+1}, \mu'\left(s_{i+1}|\theta^{\mu'}\right)|\theta^{Q'}\right] \quad (1)$$

$$L = \frac{1}{N}\sum_i \left(y_i - Q\left(s_i, a_i|\theta^Q\right)\right)^2 \quad (2)$$

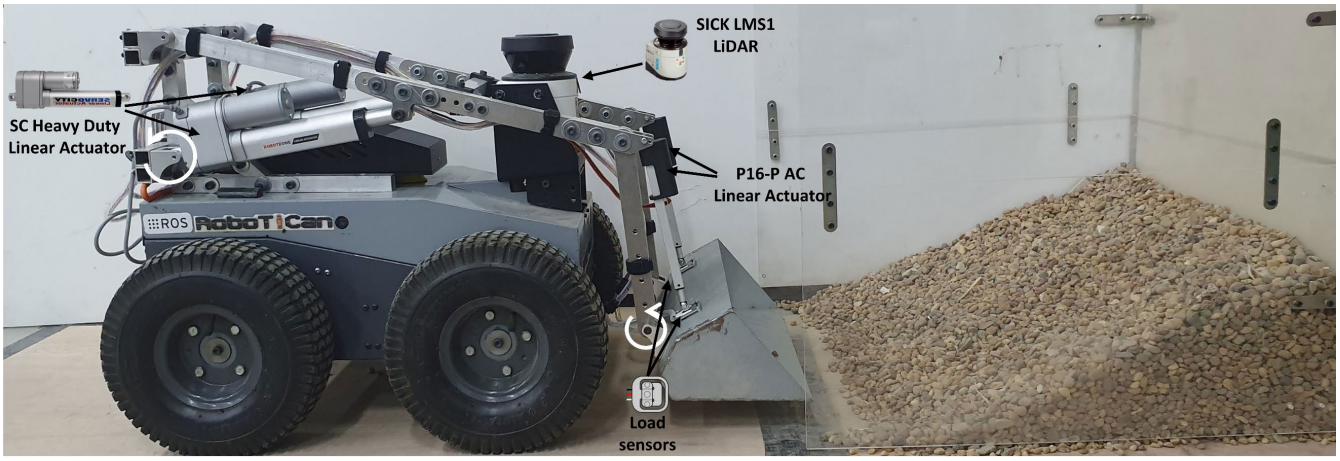

FIGURE 3. Architecture of the Komodo robot with our arm mechanism.

The critic is updated by minimizing the loss and the actor is updated by applying the chain rule using the sampled policy gradient:

$$\nabla_{\theta^\mu} J \approx \frac{1}{N} \sum_i \times \left[\nabla_a Q\left(s, a|\theta^Q\right)|_{s=s_i, a=\mu(s_i|\theta^\mu)} \nabla_{\theta^\mu} \mu\left(s|\theta^\mu\right)|_{s=s_i} \right] \quad (3)$$

IV. SYSTEM DESIGN

To investigate bucket-scooping processes, we developed an arm mechanism that we attached to a Komodo robot (see Fig. 3). The robot is a 40kg weight fully Robot Operating System (ROS) supported skid-steer robot that includes an Intel NUC i7 CPU, SICK LMS1 range laser scanner, an Asus Xtion pro depth camera, and 6-axis IMU. To control the robot motion, we used the Komodo built-in Roboteq differential driver for the motor controllers. For the loading mechanism, earth-moving mechanisms typically consist of a robotic arm controlled by hydraulic pistons and a bucket. In this work, we developed an arm mechanism that resembles to an industrial skid-steer loader that consists of three main parts: 1) a base link, 2) an arm assembly, and 3) a bucket, as shown in Fig. 3. The mechanism is constructed primarily from strong lightweight 6061 aluminum parts that can withstand a bucket load of up to 12[Kg]. The mechanism is controlled by two parallel systems of linear actuators with a built-in position feedback potentiometer. The arm's linear actuators are responsible for the lift motion, whereas the bucket's linear actuators generate its tilting motion. To control tilt and lift motion, we used a Proportional Integral Derivative controller (PID), employing feedback from the linear actuator's potentiometer. The system is operated using the Robot Operation System (ROS) interface.

The electrical system is depicted in Fig. 4, and the components are listed in Table 1. We built the electric circuit to be powered by the robot's power source, whereas the components with other power requirements utilized

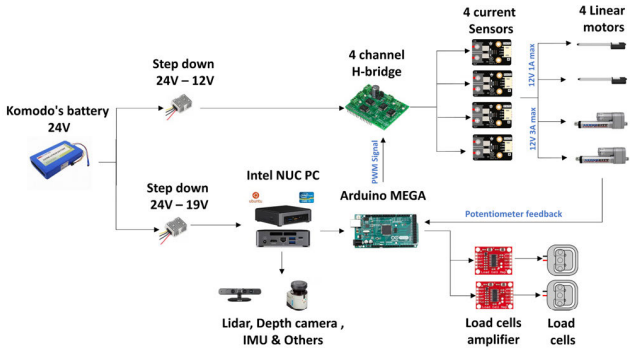

FIGURE 4. The electrical scheme of the arm mechanism.

TABLE 1. Arm mechanism components.

Product type	Quantity	Function
Arduino Mega	1	-
Arduino MultiMoto shield	1	-
P16-P Linear actuator with feedback	2	Tilt motion
Heavy-duty linear Actuator	2	Lift motion
Gravity analog current sensor	4	Current measurement
Load cell amplifier HX711	1	Weight measurement
30Kg Load cell	2	Weight measurement

voltage converters. We divided the control system into two levels. The higher level was composed of the PID application on the tilt and lift motion, communication initializing, and syncing between the various channels using ROS. The lower-level controller was composed of two slave nodes that subscribe to the data published by the main controller channels. The first involves writing Pulse Width Modulation (PWM) and direction commands to the actuators, whereas the other reads and publishes the current voltage of the linear actuator's feedback potentiometer.

V. SCOOPING CONTROLLER ARCHITECTURE

In this section, we present our methodology for the DRL scooping motion controller. We describe the implementation of our complete learning system in a virtual environment and its training progression.

A. PROBLEM FORMULATION

We consider the combined bucket-scooping process of approaching and penetrating the pile, lifting, and then tilting the bucket until breakout using UGV with a loading mechanism. Our goal is to find a closed-loop policy using RL methods that maps sensor observations to robot action. We thus need a policy function $a_t = f(\mathrm{s_t})$, where s_t is a parameterized vector of the current observed robot state using onboard sensor readings and a_t is the action composed of the lift, tilt, and throttle commands. Given the dynamics of the robot, we limit the linear velocity to $0.1m/s$ for moving towards and away from the pile, whereas lift and tilt actuators were not limited and could extract at any of their maximum velocities.

B. SIMULATION ENVIRONMENT

The training procedure of our model was implemented in a virtual 3D environment simulated by a Gazebo. We used a particle-based discrete representation of the pile. This simplification of the real-world surfaces enabled the use of common physics simulators, which can maintain a close resemblance to reality. The number of particles impacts the real-time factor within the simulation considerably, which in turn represents the total run-time. In our environment, each pile of particles contained 72 particles with radii of 3.4 cm and a mass of 0.3kg. Only spherical particles were used in this study. The simulation properties were carefully chosen to mimic the physics of the real robot as well as possible. The robot scooped gravel with a bulk density of $1.3\,\mathrm{ton/m^3}$ (bucket capacity is approximately 5[L], 7[kg]). Although the simulation was based on a simplified scenario which we found to be sufficient to achieve our goal, other users could vary the environmental properties, such as the pile geometry, number of particles, and the maximum actuator load. In addition, users could choose to define the state according to other features (e.g., object orientation or angular velocities) as well as the authorized actions (e.g., discrete or continuous) for learning purposes. Also, users could build new models to independently collect data using the open-source code provided here [17].

C. STATE REPRESENTATION

The observation is the raw information provided by the robot sensors and the state is a compact depiction of this observation that includes the information necessary for the robot to choose its actions. Following the robotic priors in [26], [27], we defined the state using fused sensor measurements as a 12-dimensional vector consisting of the arm link positions ($\boldsymbol{x_t}$), the normalized mass of the excavated earth (w_t), the relative distance to the pile ($\boldsymbol{d_t}$), the state of the robot's DoF (v, θ_T, θ_L) and their relative changes ($\boldsymbol{\Delta_t}$). A visual depiction of these parameters is provided in Fig. 5. Deriving the amount of the excavated earth within the simulation environment was done directly using the simulation's built-in functions. Measuring it in a real-world experiment could be done indirectly by approximation using the torque applied to the bucket joint as:

$$\tau = V_S \gamma g r_{cm} \tag{4}$$

where V_S is the excavated earth volume, γ is the material density, g is the gravitational acceleration and r_{cm} is the approximate relative distance between the bucket CoM to the joint. The total force acting on the bucket can be decomposed into three main forces [6]: the shear force, the gravity force, and the remolding force. By deriving the amount of excavated earth from the applied torque, we included torque-to-mass extraction of the shearing force as the bucket penetrates the earth. Therefore, we limit our proposed controller to scooping earth with low adhesion, which is applicable to numerous earth-moving scenarios, since the main purpose of the wheel loader is the transportation of already excavated material.

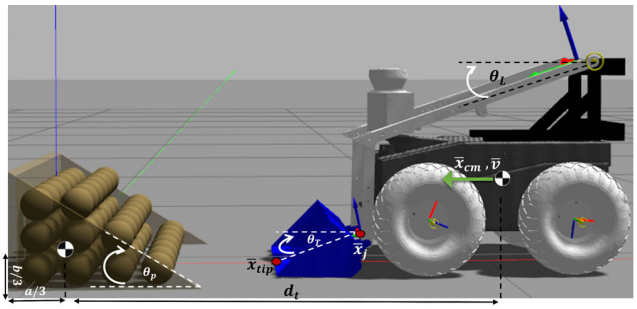

FIGURE 5. Komodo robot with an arm mechanism and a pile of particles, simulated in the Gazebo physics engine.

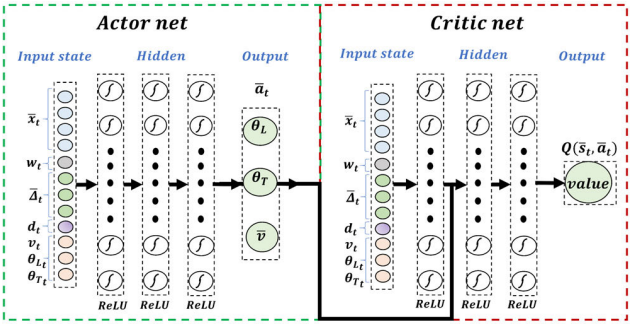

FIGURE 6. The DDPG network structure.

D. DDPG ARCHITECTURE

Our implementation of DDPG includes two neural networks (actor and critic), as shown in Fig. 6. Both networks have the current state as input; i.e., 12-dimensional sensor data.

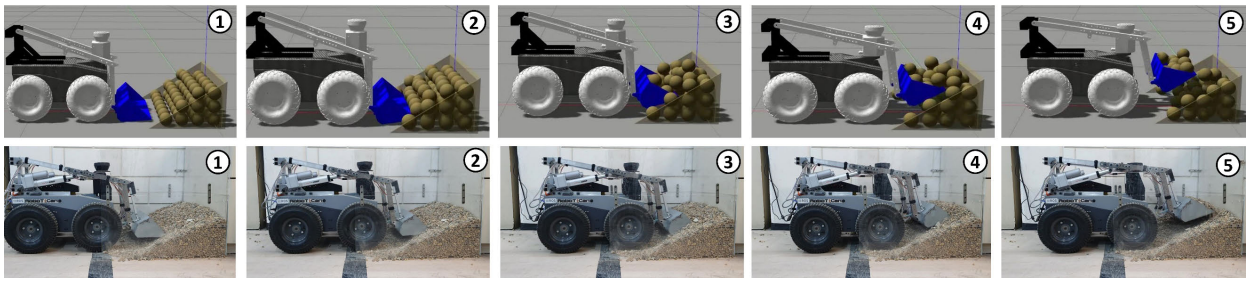

FIGURE 7. Sequence of scooping motions of the Komodo robot with an arm mechanism in simulated and real-life experiments.

The actor network output is composed of 3-dimensional deterministic actuator commands that drive the tilt, lift, and the robot's velocity. The output of the actor network is merged with the first layer of output from the critic network, which in turn is used to output the action-value function $Q^{\pi}(s, a)$. The action-value function is used to produce temporal-difference errors deriving the learning in both the actor and critic. We used three hidden layers in both the actor and critic networks. Each layer is fully connected that includes layer normalization whose dimensions decrease from 100, 80, and 60 respectively. A hyperbolic tangent function was used as the activation function to constrain the range of the actions.

E. REWARD SHAPING

To guide the robot to excavate soil during the task, while not obligating the robot to follow a specific trajectory, we defined a shaped reward:

$$P_t = \begin{Bmatrix} 1 - \tanh^2(\|\boldsymbol{d}_{end}\|) & penetrate \\ 1 - \tanh^2(\|\boldsymbol{d}_t\|) & \neg penetrate \end{Bmatrix} \tag{5}$$

Up to the penetration into the earth; i.e., $\neg penetrate$, the reward increases while the agent minimizes the relative distance $\boldsymbol{d}_t$, thus encouraging the robot to engage with the pile. During penetration; i.e., *penetrate*, the reward increases while the robot minimizes the distance to the terminal state, $\boldsymbol{d}_{end}$. To encourage the robot to scoop more soil during the process, we added a positive reward consisting of multiplying the excavated earth indicator, w_t, by a constant c_w. Hence, we defined the immediate reward as:

$$r_t = \begin{Bmatrix} P_t + c_w w_t & \|\boldsymbol{d}_{end}\| \leq \epsilon \\ P_t & \|\boldsymbol{d}_{end}\| > \epsilon \end{Bmatrix} \tag{6}$$

where ϵ is the distance tolerance to the terminal state. The precondition for penetration into the soil; i.e., *penetrate*, is defined as:

$$penetrate = min_{p_i} \left\| \boldsymbol{d}^{\boldsymbol{B}} - \boldsymbol{d}^{\boldsymbol{p_i}} \right\| \leq \Delta \tag{7}$$

The values $\boldsymbol{d}^{\boldsymbol{B}}, \boldsymbol{d}^{\boldsymbol{p_i}}$ are the robot's bucket tip and particle i positions, respectively.

F. POLICY TRAINING

In each episode during the simulation training session, the pile was initialized in a specific location and particle arrangement. Throughout the trial, the robot attempted to complete the bucket-filling process within 8 seconds using combination of the three DoF. We considered the outcome of an episode a success and terminated if, within the maximum episode time, the tip of the loader bucket reached the terminal located outside the pile. We trained the networks with algorithmic parameters using conservative values to achieve stable and reliable convergence. The model was trained from random initial weight values using a single Nvidia Quadro P2000D GPU for several trials of 800 episodes which took approximately three hours each. We employed a simple exploration strategy as described in [16] that includes noise inputted to the actor's action.

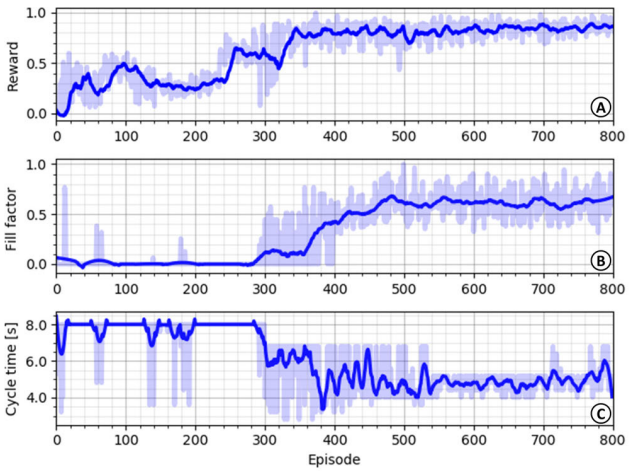

FIGURE 8. Normalized learning curve for the DDPG agent over 800 episodes (A). The fill factor is represented by the excavated soil in the bucket normalized by full bucket capacity (B). Cycle time represents the operation cycle time from the beginning of the episode until it reaches the terminal state (C).

VI. SIMULATIONS AND EXPERIMENTAL RESULTS

A. SIMULATION EVALUATION

We assessed the learned scooping cycles in the simulation using both the average episode reward, and standard earth-moving parameters; namely, the fill factor and operation cycle

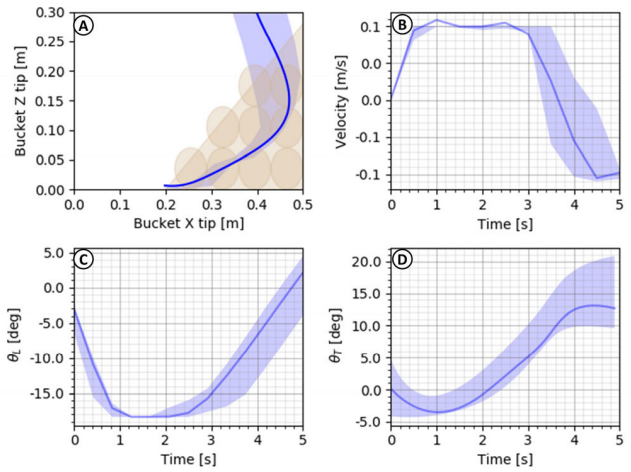

FIGURE 9. Evaluation of learned scooping cycles in the simulations. Bucket tip trajectory (A), robot velocity (B), lift angle (C) and tilt angle (D).

time, as shown in Fig. 8. The scooping policy converged at the end of the training phase with an average loading cycle time of 5[*sec*] and a fill factor of 65% from full bucket capacity with an overfill (approximately 23 particles). The learned scooping policy can be classified into 3 phases, as indicated in Fig. 9. In the first phase, the robot moves toward the pile of particles while choosing the height and penetration angle to approximately zero tilt and minimum lift angles. In the next phase, the robot changes the lift, tilt, and velocity actions simultaneously to navigate the bucket tip through the earth pile. Lift and tilt angles are increased and velocity decreases while the robot excavates deeper. In the last phase, the robot drives backward away from the pile, while lifting the arm to its maximum height to maintain the soil inside the bucket without loss of the material.

B. REAL ENVIRONMENT EVALUATION

To investigate the performance and adaptability of our scooping controller in real-life machinery, we conducted several experiments with three different mean inclination angles of the pile. We explored the learned policies trained in simulation in the real world without tuning the network weights. To derive the amount of excavated soil in the bucket, we attached a load sensor to the bucket joint that measured the load on the linear actuator. Under the assumption that acceleration and velocity are negligible during excavation, the scooping dynamics in joint space can be reduced to a simple static equilibrium:

$$\tau = G + J^T f \tag{8}$$

where f is a vector representing the contact forces, G is a vector containing the gravity terms, τ is a vector used to denote the torque at each joint and J is the Jacobian matrix of the arm. The masses of the links of the arm mechanism are known, as well as the Jacobian matrix, so J and G are obtained through direct computation using the joint positions. f is obtained through the load cell measurements since we can derive a simple quasi-static analysis that describes the moment's equilibrium at the bucket joint. Although we approximated the accumulated mass using the load cell, as described in Section III, most of the inconsistency between the simulation and real-life occurred during the *fill* phase, as can be seen in Fig. 10, because the robot basically translated this to additional scooped earth, resulting in an earlier *exit* phase.

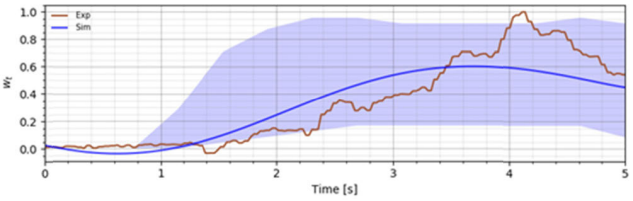

FIGURE 10. Example of comparisons between the normalized extracted mass inputs in the simulated (100 episodes) and real-life experiments. The data points are normalized by the maximum measured values of each set.

In our experiments, we defined two environment setups to be constant: (1) the position of the pile and (2) the starting position of the robot (1[m] in front of the pile). We set these conditions to keep the problem's complexity manageable and because these setups are common to various types of localization methods and feature detection methods. The bucket scooping trajectories shown in this section were obtained using direct kinematics of the arm mechanism feedback and odometry from the wheel encoders. Computation of the total excavated earth in the experiments was carried out based on the applied torque, where the bucket exits the pile of gravel.

To compare the planner's performance to human operators, the first step in the evaluation consisted of collecting data from manually driven scooping cycles using manual control. The first author collected the data while attempting to manually imitate the three phase scoop strategy. Although the goal was to achieve continuous actions, the motions were less skilled than that of an expert operator. In total, 20 trials were conducted, which were less smooth during the first attempts than in the later ones. We evaluated the learned scooping policy over 5 consecutive scooping cycles, where the earth's inclination angle set to 32 degrees. Fig. 11 and Fig. 12 show the results, where each phase is labelled. In most of the *approach* phases, the robot prepared to interact with the soil by adjusting the bucket to a horizontal angle and decreasing the lift angle. Each contact with the pile triggered the *fill* phase, where the robot penetrates the pile with a "slicing motion".

During that motion, the applied torque increased due to higher pile resistance. The controller continued the filling phase until a combination of depth and torque thresholds were reached, leading to a low variation *exit* phase initialization. Although we initialized each trial with the same setup, the controller executed different trajectories during the *approach* phase, due to changes in the soil-tool interaction,

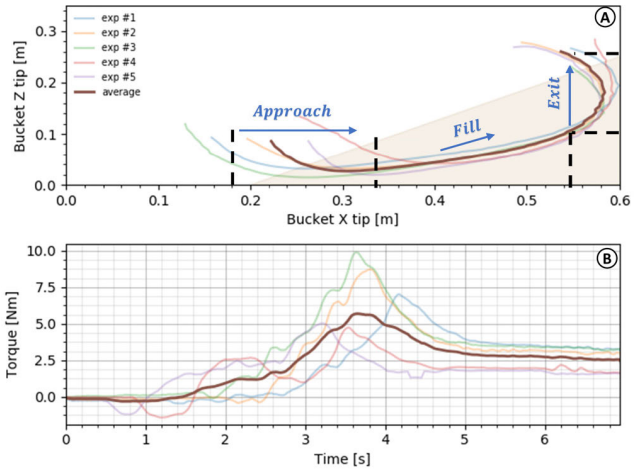

FIGURE 11. Comparisons of 5 consecutive scooping attempts from the pile with an average of 32 degrees of inclination angle. Bucket trajectory (A) applied torque at the bucket joint (B).

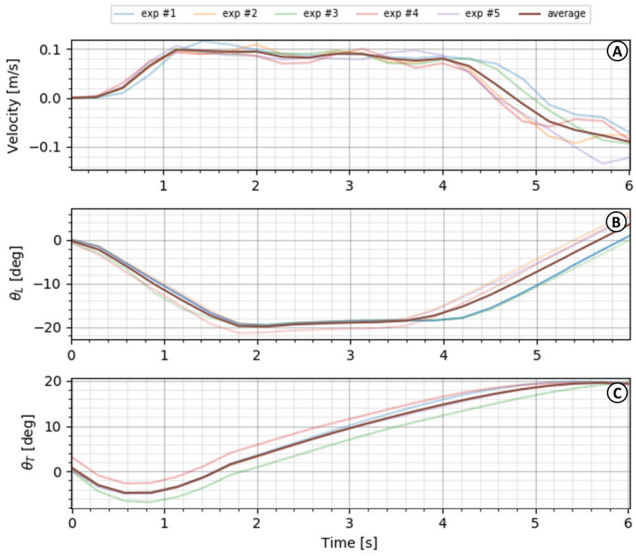

FIGURE 12. Velocity, lift and tilt motion recorded in 5 consecutive scooping cycles.

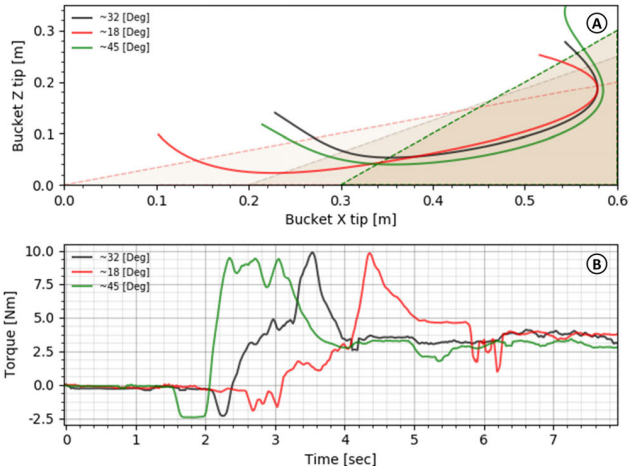

FIGURE 13. Bucket tip trajectories over 3 different mean earth inclination angles (A). Applied torque at the bucket joint (B).

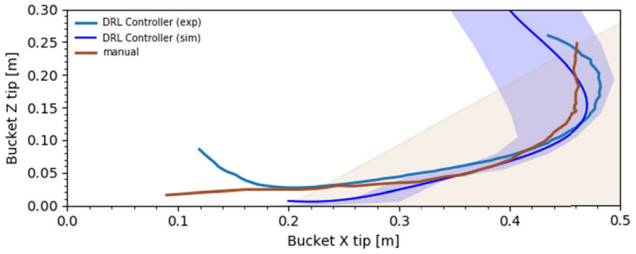

FIGURE 14. Comparisons between the scooping process in the simulations and real-life experiments.

TABLE 2. Comparison of scooping experiments.

Method	Mass [Kg]
DRL Controller - simulation	4.4 ± 0.9 (14 ± 3 particles)
DRL Controller - experiment	5.5 ± 1.0
Manual control	6.1 ± 0.7

which thus illustrates the sensitivity of the resistive model to contacts.

We also conducted another experiment to test the controller's adaptability to different environments, but with typical features such as different inclination angles of the terrain. A detailed comparison of the bucket tip trajectories (3 trials for each inclination angle) over a scooping cycle can be seen in Fig. 13 with three different pile inclination angles.

Steeper pile inclination angles of the earth caused higher resistance force from the pile, leading to an earlier *exit* phase initialization, as can be seen in Fig 13. Overcoming different inclination angles supports the claim that our DRL-based controller can perform successfully in arbitrarily designed piles. The results of both experiments suggest that the transition between different phases of the scooping process does not depend on a single feature of the state, but rather on a combination of input features.

The controller continued the filling phase until reaching the torque and depth thresholds, leading to an exit phase initialization, which appeared to be similar across all the attempts, as can be seen in Fig. 13. Comparing the DRL controller performance to the manually driven scooping attempts revealed differences between various processes of the scooping cycle, as can be seen in Fig. 14. Due to the fact that we are not expert operators, it was extremely challenging to maintain a successful smooth, fast continuous actions. The DRL controller achieved smoother action transitions and faster scooping cycles (average manual operation time was 10[*sec*], compared to 6[*sec*] for the DRL-based controller). However, our manual attempts resulted in slightly better bucket weights, where the manual control resulted in an

average of 6.1[kg] and the DRL controller resulted in an average of 5.5[kg] (9% difference), as shown in Table 2.

Comparing the experimental results of our DRL controller to the simulation polices also pointed to differences between the phases of the scooping cycle. These included a difference between the penetration angle, that reached a maximum of $7.5 \pm 1.0\,[deg]$, higher loading cycles in the experiment, and a different trajectory within the second phase of the cycle, as illustrated in Fig. 14. These can be explained by the discrepancies between the simulation environment and the real-life experiments, such as particles vs. earth, action execution time, and mass approximation.

VII. CONCLUSION

In this article, a machine learning-based deep reinforcement learning scooping motion controller was applied for UGV with a custom-built scooping mechanism to perform scooping cycles with three DoF (lift, tilt, velocity) by fusing 12-dimensional sensor data as inputs to output actuator commands. The controller was trained using the actor-critic, off-policy Deep Deterministic Policy Gradient (DDPG) algorithm without any user-provided demonstrations. The learning of the policy network is accomplished in a simplified simulation scenario, where the soil was represented as particles. The scooping policy converged at the end of the training phase with an average loading cycle time of 5 seconds and an average fill factor of 65% from full bucket capacity with an overfill. To investigate the performance and adaptability of our scooping controller, we tested our controller in several experiments with three different mean inclination angles of the earth. Overall, the DRL-based control exhibited good performance in terms of both achieved visual bucket fill with varying scooped earth weights of $4.1 - 7.2[kg]$, and $5.1 - 7.1[sec]$ cycle time. Comparisons between the manual and our DRL controller indicated that human control led to the highest average bucket load. Although the DRL controller may not yet be on a par with human control for the highest volume, the DRL controller here did achieve smoother trajectories and lower cycle times than the manual control. This evaluation underscores the importance of the expert knowledge required for excavation. By contrast, most previous solutions to the automation of the scooping task (1) do not generalize to different machines and pile environments (2) rely on prior knowledge of an expert operator and (3) require accurate models of the machine [10]. We presented a method that outperforms the above using the power of RL.

Our learned controller acts based on measured sensors and implements the transitioning between phases with fixed predetermined criteria, which were developed solely through the training in the simulation. Future work should consider testing the proposed controller on a full-size skid-steer loader, as well as tackling the full problem of earthmoving including navigating to the pile, attempting a scooping cycle, navigating to another site and dumping.

ACKNOWLEDGMENT

The authors would like to thank RoboTiCan for lending them the robot used in the study presented in this work.

REFERENCES

[1] S. Dadhich, U. Bodin, and U. Andersson, "Key challenges in automation of earth-moving machines," *Autom. Construct.*, vol. 68, pp. 212–222, Aug. 2016.

[2] S. Dadhich, "A survey in automation of earth-moving machines," Lulea Univ. Technol., Luleå, Sweden, Res. Rep., 2015.

[3] S. Singh, "State of the art in automation of earthmoving," *J. Aerosp. Eng.*, vol. 10, no. 4, pp. 179–188, Oct. 1997.

[4] R. Filla, M. Obermayr, and B. Frank, "A study to compare trajectory generation algorithms for automatic bucket filling in wheel loaders," in *Proc. 3rd Commercial Vehicle Technol. Symp. Commercial Vehicle Technol. (CVT)*, 2014, pp. 588–605.

[5] T. Tomatsu, K. Nonaka, K. Sekiguchi, and K. Suzuki, "Model predictive trajectory tracking control for hydraulic excavator on digging operation," in *Proc. IEEE Conf. Control Appl. (CCA)*, Sep. 2015, pp. 1136–1141.

[6] O. Luengo, S. Singh, and H. Cannon, "Modeling and identification of soil-tool interaction in automated excavation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Systems. Innov. Theory, Pract. Appl.*, Oct. 1998, pp. 1900–1906.

[7] K. Althoefer, C. P. Tan, Y. H. Zweiri, and L. D. Seneviratne, "Hybrid soil parameter measurement and estimation scheme for excavation automation," *IEEE Trans. Instrum. Meas.*, vol. 58, no. 10, pp. 3633–3641, Oct. 2009.

[8] D. Schmidt, M. Proetzsch, and K. Berns, "Simulation and control of an autonomous bucket excavator for landscaping tasks," in *Proc. IEEE Int. Conf. Robot. Autom.*, May 2010, pp. 5108–5113.

[9] S. G. Olsen and G. M. Bone, "Modelling of robotic bulldozing operations for autonomous control," in *Proc. 24th Can. Conf. Elect. Comput. Eng. (CCECE)*, 2011, pp. 001188–001193.

[10] S. Dadhich, F. Sandin, U. Bodin, U. Andersson, and T. Martinsson, "Field test of neural-network based automatic bucket-filling algorithm for wheel-loaders," *Autom. Construct.*, vol. 97, pp. 1–12, Jan. 2019.

[11] I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, and M. Riedmiller, "Data-efficient deep reinforcement learning for dexterous manipulation," 2017, *arXiv:1704.03073*. [Online]. Available: http://arxiv.org/abs/1704.03073

[12] L. Tai, G. Paolo, and M. Liu, "Virtual-to-real deep reinforcement learning: Continuous control of mobile robots for mapless navigation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 31–36.

[13] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2096–2103, Oct. 2017.

[14] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2017, pp. 3389–3396.

[15] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke, "Sim-to-real: Learning agile locomotion for quadruped robots," 2018, *arXiv:1804.10332*. [Online]. Available: http://arxiv.org/abs/1804.10332

[16] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," 2015, *arXiv:1509.02971*. [Online]. Available: http://arxiv.org/abs/1509.02971

[17] O. Azulay. *Package for Simulating Wheel Loader in Gazebo Engine for Reinforcement Learning Purposes*. Accessed: Jan. 12, 2021. [Online]. Available: https://github.com/osheraz/komodo

[18] P. J. A. Lever and F.-Y. Wang, "Intelligent excavator control system for lunar mining system," *J. Aerosp. Eng.*, vol. 8, no. 1, pp. 16–24, Jan. 1995.

[19] A. A. Dobson, J. A. Marshall, and J. Larsson, "Admittance control for robotic loading: Underground field trials with an LHD," in *Field and Service Robotics*. Toronto, ON, Canada, 2015.

[20] D. Jud, G. Hottiger, P. Leemann, and M. Hutter, "Planning and control for autonomous excavation," *IEEE Robot. Autom. Lett.*, vol. 2, no. 4, pp. 2151–2158, Oct. 2017.

[21] G. J. Maeda, I. R. Manchester, and D. C. Rye, "Combined ILC and disturbance observer for the rejection of near-repetitive disturbances, with application to excavation," *IEEE Trans. Control Syst. Technol.*, vol. 23, no. 5, pp. 1754–1769, Sep. 2015.

[22] S. Singh and H. Cannon, "Multi-resolution planning for earthmoving," in *Proc. IEEE Int. Conf. Robot. Automat.*, May 1998, pp. 121–126.
[23] O. Kanai, H. Osumi, S. Sarata, and M. Kurisu, "Autonomous scooping of a rock pile by a wheel loader using disturbance observer," in *Proc. 23rd Int. Symp. Autom. Robot. Construct.*, Oct. 2006, pp. 3–5.
[24] S. Dadhich, U. Bodin, F. Sandin, and U. Andersson, "Machine learning approach to automatic bucket loading," in *Proc. 24th Medit. Conf. Control Autom. (MED)*, Jun. 2016, pp. 1260–1265.
[25] E. Halbach, J. Kamarainen, and R. Ghabcheloo, "Neural network pile loading controller trained by demonstration," in *Proc. Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 980–986.
[26] R. Jonschkowski and O. Brock, "Learning state representations with robotic priors," *Auto. Robots*, vol. 39, no. 3, pp. 407–428, Oct. 2015.
[27] T. Lesort, N. Díaz-Rodríguez, J.-F. Goudou, and D. Filliat, "State representation learning for control: An overview," *Neural Netw.*, vol. 108, pp. 379–392, Dec. 2018.

OSHER AZULAY received the B.Sc. and M.Sc. degrees (Hons.) from the Department of Mechanical Engineering, Ben-Gurion University of the Negev, Israel, in 2019 and 2020, respectively. He is currently pursuing the Ph.D. degree with the School of Mechanical Engineering, Tel Aviv University. His current research interests include robotic manipulators and reinforcement learning. He had been awarded numerous prizes for academic and research excellence, including certificates of achievement 2017–2018 and 2018–2019 and an Excellence Scholarship from the Mechanical Engineering Department, Ben-Gurion University of the Negev.

AMIR SHAPIRO (Member, IEEE) received the B.Sc., M.Sc., and Ph.D. degrees in mechanical engineering from the Technion, Israel Institute of Technology, Haifa, in 1997, 2000, and 2004, respectively. From 2005 to 2006, he was a Postdoctoral Fellow with the Robotics Institute, Carnegie Mellon University, Pittsburgh, PA, USA. He is currently an Associate Professor and the Director of the Robotics Laboratory, Department of Mechanical Engineering, Ben-Gurion University of the Negev, Be'er Sheva, Israel. His research interests include locomotion of multi-limbed mechanisms in unstructured complex environments, motion planning algorithms for multi-limbed robots, robot grasping-design, control, and stability analysis, climbing robots, snake-like robots, multi-robot on-line motion planning, and agricultural robotics.