IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.
Cite as T-ASE paper.

Progressive-Resolution Policy Distillation: Leveraging Coarse-Resolution Simulations for Time-Efficient Fine-Resolution Policy Learning

Yuki Kadokawa[1], Hirotaka Tahara[1,2], Takamitsu Matsubara[1], *Member, IEEE,*

***Abstract*—In earthwork and construction, excavators often encounter large rocks mixed with various soil conditions, requiring skilled operators. This paper presents a framework for achieving autonomous excavation using reinforcement learning (RL) through a rock excavation simulator. In the simulation, resolution can be defined by the particle size/number in the whole soil space. Fine-resolution simulations closely mimic real-world behavior but demand significant calculation time and challenging sample collection, while coarse-resolution simulations enable faster sample collection but deviate from real-world behavior. To combine the advantages of both resolutions, we explore using policies developed in coarse-resolution simulations for pre-training in fine-resolution simulations. To this end, we propose a novel policy learning framework called Progressive-Resolution Policy Distillation (PRPD), which progressively transfers policies through some middle-resolution simulations with conservative policy transfer to avoid domain gaps that could lead to policy transfer failure. Validation in a rock excavation simulator and nine real-world rock environments demonstrated that PRPD reduced sampling time to less than 1/7 while maintaining task success rates comparable to those achieved through policy learning in a fine-resolution simulation.**

***Note to Practitioners*—This paper is motivated by the issue of computation time in excavation simulation using soil particles. The behavior of real soil is highly complex, and approximating it at high resolution requires enormous computational costs. Therefore, existing soil simulators have focused on improving simulation accuracy while maintaining reduced computation time. This paper takes a different approach by focusing on the learning of control policies in excavation simulators and proposes a framework for reducing calculation time in such use cases. In this framework, a control policy is first learned in a low-resolution simulation, significantly reducing computation time. The learned policy is then transferred to a high-resolution simulation for retraining, thereby achieving an overall reduction in simulation time. Furthermore, to enable robust policy transfer across different resolutions, this paper discusses a stable policy distillation scheme and insights into resolution design. This approach enables the development of autonomous excavation systems without relying on expensive real-world data collection, improving the scalability and adaptability of autonomous excavation. Simulation experiments suggest that this framework significantly reduces training time compared to conventional policy learning approaches. However, real-world validation has so far been limited to simple excavation robots. Future research will explore applications to excavators and other machinery more suitable for real-world operations. Although this paper focuses on autonomous excavation, the proposed approach can also be extended to environments where increased simulation resolution critically impacts computation time, such as liquid and soft object manipulation.**

***Index Terms*—Simulation Resolution, Excavation, Policy Distillation, Reinforcement Learning**

This work was supported by JST Moonshot Research and Development, Grant Number JPMJMS2032. *(Corresponding author: Yuki Kadokawa.)* [1] Nara Institute of Science and Technology, Nara 630-0192, Japan. [2] Kobe City College of Technology, Hyogo 651-2194, Japan. kadokawa.yuki@naist.ac.jp, h-tahara@kobe-kosen.ac.jp, takam-m@is.naist.jp

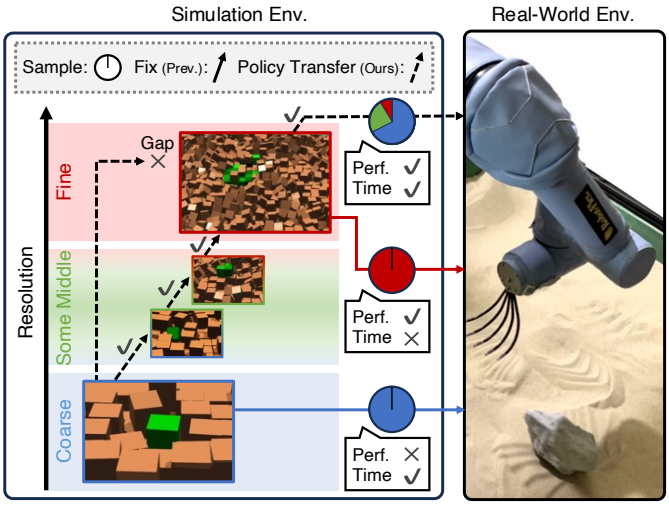

Fig. 1. Overview of proposed framework: Fine-resolution simulations yield high policy performance but require long learning times, while coarse-resolution simulations allow for quick learning but perform poorly in sim-to-real transfer. Our framework starts with coarse-resolution simulations for quick learning and progressively transfers policies to fine-resolution simulations. Progressive resolution shift with conservative policy transfer is applied to avoid large domain gaps that could lead to policy transfer failure. This approach balances learning time with real-world performance.

I. Introduction

AUTONOMOUS excavation has been studied to meet the rising demand for earthwork and construction [1]–[3]. In excavation, especially in mountainous areas, quarry sites, and construction sites, large rocks are often mixed with sand, gravel, and pebbles in the soil [4, 5]. Operating excavators must have advanced operational expertise to find, collect, and transport these rocks [6, 7]. For rock excavation, excavators must efficiently lift and move rocks using a bucket. This involves non-grasping bucket motion while considering forces from rock-soil-bucket interactions, a task only skilled operators can perform [8, 9]. Due to the varied shapes and sizes of rocks, automating rock excavation with simple autonomous control policies is challenging [10, 11].

Reinforcement learning (RL) is potentially useful to learn optimal actions from diverse state-action transitions caused

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

by various rocks [12]–[14]. RL learns control policies from interaction samples between a robot and the environment. Thus, simulators are commonly used since real-world sampling requires significant manual efforts to set up various types of soil and objects [15, 16]. In previous simulation work, computational mechanics and particle methods have been used. Computational mechanics, which discretize terrain into elements, are suitable for simple terrains but struggle with complex terrains involving rocks or nonlinear phenomena [15, 16]. Conversely, particle methods model materials as individual particles and simulate their interactions, effectively reproducing complex phenomena like friction, collisions, and soil deformation [17, 18]. This makes particle methods suitable for simulating complex terrains that include rocks.

While particle-based simulations can be accurate and realistic, they face challenges in computation time for our RL application, particularly as *spatial resolution* improves. This resolution can be defined by the particle size/number in the whole soil space, with smaller particle sizes (more amounts of particles) allowing for a more detailed particle representation of the soil space and obtaining a finer resolution. Fine resolution leads to complex particle interactions, making it impractical to collect the vast samples needed for RL in a reasonable calculation time [19, 20]. Ideally, the resolution should be coarsened until the simulator is sufficiently fast; however, simulation deviates from real-world behavior, utilizing the learned policies ineffective in real-world environments. Our focus is on addressing RL problems where there is a trade-off between resolution-dependent calculation time and sample quality, aiming to reduce the computation time required for policy learning.

As shown in Fig. 1, we explore using policies developed in coarse-resolution simulations for pre-training in fine-resolution simulations, even though these policies cannot work in real-world settings. This approach potentially reduces sampling time compared to using only fine-resolution simulations since coarse-resolution simulations can be calculated in a shorter time. However, behaviors in different resolution simulations typically deviate due to domain gaps, making it challenging to utilize coarse-resolution simulations for pre-training. Therefore, by transferring policies progressively through some middle resolutions and by transferring policies gradually while checking the stability of the policy updates instead of transferring policies all at once, we could effectively transition these policies to simulations that closely resemble real-world settings.

We propose a novel policy learning framework called Progressive-Resolution Policy Distillation (PRPD) for achieving time-efficient policy learning of fine-resolution simulations by utilizing coarse-resolution simulations effectively. PRPD progressively improves simulation resolution while learning and transferring policies at each stage. PRPD uses a conservative policy transfer scheme to regularize policies, stabilizing policy transfer across different resolution simulations having different behaviors. We demonstrated the effectiveness of PRPD by constructing a variable-resolution rock excavation simulator using Isaac Gym [21]. This paper demonstrated a 7-fold improvement in learning time efficiency (PRPD: 90 minutes, policy learning in the finest-resolution simulation: 600 minutes), resulting in an 8-hour time gap. This difference is likely to become more significant with increasing state-action space complexity or task difficulty. Furthermore, in evaluations of the learned policies within a real-world environment containing nine types of rocks, PRPD achieved rock excavation in approximately $90\,\%$ of scenarios.

As an impact of this study, achieving realistic simulations of complex environments, such as excavation tasks, remains challenging. While advances in computational resources and simulation technologies may alleviate simulation time issues, simulating real-world environments accurately remains prohibitively costly and will still face resource limits. This study offers insights into addressing these challenges by improving policy-learning efficiency.

The following are this paper's main contributions:

- It proposes a novel policy learning framework, PRPD, which enables time-efficient policy learning by progressively increasing simulation resolution and scheduling tasks from coarse to fine based on policy performance.
- While the underlying policy optimization and distillation mechanisms are based on existing methods, the novelty of the proposed framework lies in integrating them into a unified structure that enables stable and progressive policy transfer across simulation resolutions. This framework effectively achieves the core novelty of progressive policy distillation across varying simulation resolutions for time-efficient policy learning.
- The effectiveness of PRPD is demonstrated in a complex particle-based excavation task, achieving a 7-fold reduction in total learning time compared to fixed-resolution training, while maintaining comparable task performance and enabling sim-to-real policy transfer in diverse real-world environments.

II. Related Works

A. Learning Excavation Policy

This section describes several methods for acquiring control policies for a robotic excavation task with soil and rocks. Previous works have proposed the following three approaches.

Learning in Real-world: Learning rock-excavation tasks in real-world environments requires extensive sampling time. These works have modeled soil and rock behavior from limited samples to learn control policies [1, 10]. However, accurately modeling rock behavior requires interaction samples for various rock shapes. The vast number of combinations makes it difficult to apply this method to multiple shapes due to sampling costs, so previous research has only obtained policies for a single rock shape.

Learning in Simulation by Computational Mechanics: These works simulate soil behavior in response to bucket movements using computational mechanics and collect learning samples from model interactions [15, 16]. The finite element method divides soil into small elements and solves for displacement, stress, and strain. Deformation analysis uses soil mechanics equations to analyze deformation and failure. These works are suitable for simulating simple terrain because

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

they numerically discretize and simulate terrain, but they have difficulty accurately reproducing complex terrain involving rocks and nonlinear phenomena.

Learning in Simulation by Particle Method: These works adopt the particle method that models materials as individual particles and directly simulates the interactions between them [22, 23]. This enables detailed simulation of rock and soil behavior, and approximates complex physical phenomena such as inter-particle friction, collision, and soil deformation separately. However, calculating interactions between multiple particles is time-consuming, making it difficult to collect many learning samples. While imitation learning applications exist with few samples, these methods have not been trained in comprehensive environments and can only be applied in limited situations.

This paper aims to establish learning policies applicable to various rock shapes by utilizing a particle-based simulator capable of achieving this aim. To address the sampling time challenge, we propose a novel policy learning framework that enables the short-time collection of learning samples from various rock shapes, thereby achieving rock excavation for multiple shapes.

B. Excavation Simulator by Particle Method

Excavation simulators with a large number of particle interactions require extensive calculation time [18, 22]. The following paragraphs describe previous works that employed three approaches to accelerate calculations and comparisons with our developed simulator.

Static Adjustment of Resolution: To reduce the calculation time for particle interactions, which make up the majority of simulation calculations, efforts have focused on generating environments limited to specific work areas and approximating many soil particles with fewer macro particles [20, 24]. These strategies have decreased the necessary memory resources and sampling time. However, there are limits to how much the resolution can be reduced without diverging from real-world behavior.

Dynamic Adjustment of Resolution: Some works accelerate simulations by dynamically changing the resolution [17, 19]. They dynamically estimate the work area, splitting particles in fine-resolution regions and merging particles in coarse-resolution areas, thereby reducing calculation time. While this approach achieves acceleration, focusing fine resolution only in the work area still requires significant resources and calculation time, making it unsuitable for the extensive sample collection needed in RL.

Parallel Calculation of Particles: Recently, frameworks like Isaac Gym and Isaac Sim have been developed to rapidly simulate robot environments using GPUs, significantly accelerating sampling time [21, 25]. In excavation environments, these achieve faster calculations by parallelizing the calculation of multiple particle interactions. However, generating a vast number of particles within the simulation requires substantial GPU memory, making it difficult to create parallel environments that can further reduce sampling time [24].

Our Developed Simulator: The simulator developed in this paper incorporates equivalents of the three acceleration techniques of the previous works: (1) approximating soil with micro particles, (2) generating only the work area for fine-resolution simulation, and (3) using Isaac Gym for GPU acceleration. The details of our simulator are described in Section V-D.

Despite these advancements in simulation acceleration, they remain insufficient for the vast sample collection required in RL [17, 19]. RL needs millions of samples, making traditional simulators challenging for policy learning [12, 13]. The goal of this paper is not to develop realistic simulations but to learn policies of realistic simulations. To shorten sampling time, we use both fine-resolution simulations that take longer to calculate and extremely coarse-resolution simulations that require less calculation time.

C. Sample Efficient Reinforcement Learning Methods

Several methods have been proposed to improve sample efficiency in reinforcement learning, each targeting different problem settings with distinct assumptions and trade-offs.

Model-based RL: This approach utilizes a learned model of the environment's dynamics to learn policies without direct environment interaction [26]–[28]. While highly sample-efficient in principle, it is susceptible to performance degradation caused by inaccuracies in the learned model, which makes it less reliable in high-dimensional or partially observable domains. Consequently, applying this approach to environments with complex or hard-to-model dynamics remains challenging.

Representation Learning: Representation learning aims to extract informative features from raw observations to improve policy learning efficiency [29]–[31]. It enhances sample efficiency by enabling more effective use of each observation, particularly in high-dimensional or partially observable environments. However, it often requires additional training objectives and network modules, which increase computational complexity and may demand task-specific tuning. In a related direction, data augmentation improves sample diversity by transforming observations and has shown notable success in visual RL settings [32]–[34]. Nevertheless, its effectiveness depends heavily on the choice of transformation and can be limited in tasks where observations are closely coupled with the underlying dynamics.

Imitation Learning: This approach improves sample efficiency by leveraging expert demonstrations to guide policy learning, thereby reducing the need for extensive exploration and environment interaction [35]–[37]. In particular, learning from small demonstration datasets or inferring reward functions from limited supervision can significantly accelerate early-stage training. However, the effectiveness of this approach heavily depends on the quality and diversity of the demonstration data, and it often struggles to generalize beyond the behaviors observed in expert trajectories.

Sample-Efficient RL without External Supervision: Off-policy RL methods, such as Soft Actor-Critic (SAC), improve sample reuse by storing and replaying past interactions from a buffer [38, 39]. Also, ensemble learning improves sample efficiency by leveraging multiple models to enhance model generalization [31, 40, 41]. Although sample-efficient, they

typically require large numbers of gradient updates and substantial memory for storing transitions, which may reduce time efficiency. Unlike methods that rely on external supervision, these approaches operate solely on state-action interaction data and require no additional annotation, reward shaping, or auxiliary training objectives.

Various methods have improved RL's "sample efficiency" by extracting more information from each sample or increasing update times to maximize sample utilization. However, these approaches often come with computational overhead, such as increased calculation time and resource demands by adding learning components, which can compromise "time efficiency," the overall training time for policy learning.

D. Comparison with Curriculum Reinforcement Learning

Curriculum Reinforcement Learning (CRL) is an RL method that starts with simple tasks and gradually increases task difficulty [42]. This approach builds on the initial learning experiences from simple tasks and gradually adapts the policies for more complex tasks. Eventually, this progressive increase in difficulty results in a reduction in the number of samples required compared to learning solely from complex tasks [43].

In previous CRL approaches, the learning process is typically organized using a curriculum based on task difficulty, while the simulator resolution remains fixed. For example, some methods restrict the range of goals or initial states during early training and gradually expand this range as the policy stabilizes, thereby guiding the agent toward a policy with better generalization capabilities [44, 45]. Other approaches begin training in simplified environments where external disturbances or noise are intentionally removed, and incrementally introduce real-world complexity to adjust the difficulty of learning [46, 47]. In addition, reward shaping has proven effective as a curriculum design strategy, where the reward function is gradually modified. It typically starts with easily obtainable rewards to promote exploration during early training and then transitions to the original reward structure [48, 49]. Thus, previous CRL approaches primarily rely on task difficulty as the basis for designing the learning curriculum.

Our approach can be interpreted as a type of CRL, as it also involves gradually changing the tasks from simple to complex. However, unlike previous CRL approaches, our method aims to reduce the overall learning time required for policy training in simulation, rather than minimizing the number of samples used for training, which is the typical objective in common CRL. To the best of our knowledge, this is the first approach to treat differences in the spatial resolution of complex simulations as distinct tasks within a curriculum framework.

III. Preliminaries

A. Reinforcement Learning

Reinforcement Learning (RL) is a framework where an agent learns an optimal policy through interaction with the environment [50]. Typically, RL is formulated as a Markov Decision Process (MDP) $\mathcal{M}$. An MDP is defined by the state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probability $P(s'|s,a)$, and reward function $R(s,a)$, represented as $\mathcal{M} = (\mathcal{S}, \mathcal{A}, P, R)$. The agent selects an action $a \in \mathcal{A}$ in the current state $s \in \mathcal{S}$ and, as a result, observes the next state $s' \in \mathcal{S}$ and reward $r \in \mathbb{R}$. The goal in an MDP is to find a policy $\pi(a|s)$ that maximizes the expected cumulative reward $J(\pi)$, given by:

$$J(\pi) = \mathbb{E}_\pi \left[\sum\nolimits_{t=0}^{\infty} \gamma^t r_t \right], \tag{1}$$

where $\gamma \in [0,1]$ is the discount factor.

To maximize the expected cumulative reward $J(\pi)$, policy gradient methods are used, where the objective function is defined as:

$$J(\pi) = \mathbb{E}_\pi \left[\sum\nolimits_{t=0}^{\infty} A(s_t, a_t) \right], \tag{2}$$

where $A(s,a) = Q(s,a) - V(s)$ is the advantage function. The value function $Q(s,a;\psi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s, a_0 = a \right]$ represents the expected cumulative reward when action a is taken in state s. The value function $V(s;\phi) = \mathbb{E}_\pi \left[\sum_{t=0}^{\infty} \gamma^t r_t \middle| s_0 = s \right]$ represents the expected cumulative reward starting from state s and following policy $\pi(s,a;\theta)$. These functions are typically approximated using function approximators such as neural networks. Here, θ, ϕ, and ψ denote the parameters of the policy π, the value functions V, Q, respectively.

A representative method based on policy gradient methods is proximal policy optimization (PPO) [51]. PPO learns an optimal policy by iteratively collecting samples (s,a,r,s') with the policy π into a rollout buffer $\mathcal{D}$, and updating the policy using these samples. The number of such iterations is denoted as i, and the number of parameter updates is denoted as k. PPO updates the policy by minimizing the following loss function:

$$\begin{aligned} \mathcal{L}^{\pi,V}_{\theta,\phi} = \mathbb{E}_{\mathcal{D}_i} \big[& \min\{\rho A_k(s,a), \text{clip}(\rho, 1-\epsilon, 1+\epsilon) A_k(s,a)\} \\ & - c_1 (V_k(s) - \hat{R})^2 + c_2 \pi_k(a|s) \log \pi_k(a|s) \Big], \end{aligned} \tag{3}$$

where ρ represents the importance sampling ratio, and ϵ is a hyperparameter controlling the clipping range. The parameters c_1 and c_2 control the weights of the value function loss and the entropy regularization, respectively. $\hat{R}$ is the cumulative reward calculated from the samples in $\mathcal{D}$ [51].

B. Conservative Policy Update in Reinforcement Learning

In RL, there are various errors such as function approximation errors and observation noise, and attempts have been made to learn policies that are robust to these errors. Conservative Policy Iteration (CPI) is one example, which is designed to achieve more stable policy updates by relaxing the policy updates scheme [52, 53].

In CPI, the policy update is modified to make it more conservative by introducing a mixing coefficient α. The new policy π_{k+1} is a mixture of the current policy π_k and the greedy policy $\mathcal{G}(Q_k)(s) = \arg\max_a Q_k(s,a)$:

$$\pi_{k+1}(s,a) \leftarrow (1-\alpha_{k+1})\pi_k(s,a) + \alpha_{k+1} \mathcal{G}(Q_k)(s), \tag{4}$$

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

where $0 \leq \alpha_{k+1} \leq 1$. This approach stabilizes policy learning by mitigating abrupt policy changes.

CPI comes with strong theoretical guarantees that ensure the policy value improves monotonically under certain conditions; if the function approximation error is bounded and the mixing coefficient α_k is chosen appropriately, the expected value of the policy is guaranteed to improve. Specifically, the mixing rate can be selected as:

$$\alpha_{k+1} = \frac{(1-\gamma)}{4R} \sum_{s \in \mathcal{S}} d_{\mu}^{\pi_k}(s) \sum_{a \in \mathcal{A}} \pi_k(a|s) A_k(s,a), \quad (5)$$

where $d_{\mu}^{\pi_{k-1}}(s)$ is the discounted state visitation distribution under policy π_{k-1} starting from initial state distribution μ, R is the maximum reward [52].

IV. Progressive-Resolution Policy Distillation

We propose a novel policy learning framework called Progressive-Resolution Policy Distillation (PRPD) to achieve time-efficient policy learning of fine-resolution simulations by utilizing coarse-resolution simulations effectively. An overview of the proposed PRPD framework is shown in Fig. 1. PRPD progressively improves simulation resolution and transfers policies at each stage. By transferring policies progressively through some middle resolutions and by transferring policies gradually while checking the stability of the policy updates instead of transferring policies all at once, we effectively transition these policies to simulations that closely resemble real-world settings. PRPD incorporates iterations of the following two steps: 1) Resolution Scheduling and 2) Policy Learning. The following sections outline the details of the learning steps. The pseudo-code is provided in Algorithm 1.

Algorithm 1: Progressive-Resolution Policy Distillation with PPO

```
# Set parameters described in Table I
# Schedule resolution progressively (Δ_1, Δ_2, ..., Δ_N)
for n = 1, 2, ..., N do
    # Initialize iteration num i=0, parameter update num k=0
    # Set values V_k^(n), Q_k^(n), policy π_k^(n), rollout buffer D_i^(n)
    # Copy network parameters (Δ_{n-1} to Δ_n )
    # Set resolution Δ ← Δ_n and generate simulator M^(n)
    while until τ > τ̂ do
        for e = 1, 2, ..., E do
            for t = 1, 2, ..., T do
                # Take action a_t from π_k^(n) in M^(n)
                # Get observation s_t, reward r_t
                # Push (s_t, a_t, r_t) to D_i^(n)
        for k' = k, k+1, ..., k+K do
            # Calculate loss: Eq. (10), update V_k^(n), Q_k^(n), π_k^(n)
        # update numbers i = i+1, k = k+K
        # Check success rate τ
```

TABLE I
Learning parameters of PRPD in experiments.

Para.	Meaning	Value
α_0	Scaling coefficient of distillation	2
γ	Discount factor of RL	0.99
Δ_1	Scale of initial resolution [mm]	70
Δ_N	Scale of final resolution [mm]	10
$\Delta_{\mathcal{R}}$	Scale of resolution interval [mm]	10
$\hat{\tau}$	Target success rate	0.95
T	Number of steps per episode	128
E	Number of episodes per iteration	128
K	Number of parameter update per iteration	64×8
c_3	Weight of distillation loss	0.5
c_4	Weight of Q-value loss	1

Simulator Assumption: To enable policy transfer, our framework follows some simulator assumptions. Policies are designed consistently across different resolutions, even though different simulators typically vary in observations and actions. We assume that resolution-affected elements do not impact observations or actions. The policies' objectives remain the same, with only the resolution differing, so the reward function is kept identical. This framework supports policy learning in high-resolution simulations, assuming that behavior is less accurate at coarser resolutions and becomes finer as resolution improves. Each resolution level corresponds to a distinct simulation environment configured with different resolution parameters, such as soil particle size. Learning samples are collected independently through interactions between policies and their corresponding environments. We assume that resolution-specific environments can be prepared in advance.

RL Scheme Assumption: PRPD assumes an actor-critic structure for the RL scheme, it requires policy π and value function Q for estimating α. Applicable learning methods include the latest DRLs PPO [51] and SAC [38].

A. Resolution Scheduling

1) Execution Flow: The simulation resolution $\Delta \in \mathbb{R}$ is scheduled based on the progress of policy learning by the resolution scheduler. Specifically, the resolution remains constant until the policy achieves the task, at which point it is progressively improved by $\Delta_{\mathcal{R}}$. The success rate is evaluated across all results of each iteration, with the task achievement defined by meeting the threshold τ. The simulation generator then creates the simulator $\mathcal{M}^{(n)}$ according to the scheduled resolution Δ.

2) Scheduling Setups: PRPD creates a simulation environment represented as MDPs $\mathcal{M}^{(1)}, \mathcal{M}^{(2)}, \ldots, \mathcal{M}^{(N)}$. We design the resolution interval $\Delta_{\mathcal{R}}$ between $\mathcal{M}^{(n-1)}$ and $\mathcal{M}^{(n)}$ to be sufficiently small so that the policy $\pi^{(n-1)}$ at the previous resolution can be stably transferred to the next policy $\pi^{(n)}$. The resolution scheduler orders multiple simulation environments, allowing the agent to learn tasks progressively.

B. Policy Learning

1) Execution Flow: An agent collects samples s, a, r from interactions with the simulation and adds them to the rollout buffer $\mathcal{D}^{(n)}$ corresponding to the current resolution. The agent then updates $Q^{(n)}$ and $\pi^{(n)}$ using the RL update scheme. At the end of each episode, the success rate τ of the policy is evaluated and passed to the resolution scheduler. When the resolution scheduler updates the resolution Δ, the simulation environments are updated by the simulation generator. Then, the previous value function $Q^{(n-1)}$ and policy $\pi^{(n-1)}$ are copied to that of the current resolution $Q^{(n)}$, $\pi^{(n)}$.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

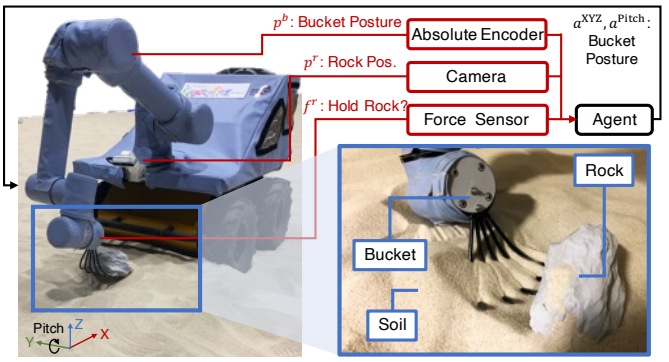

Fig. 2. Our experimental rock excavation setup: The excavator operates a bucket attached to its arm to remove rocks from the soil. Inputs to the control policy include the bucket's posture p^b (position and rotation) from the excavator's absolute encoder, rock coordinates p^r estimated by the camera, and the presence of rocks in the bucket f^r estimated by the force sensor. The output of the control policy is the position a^{XYZ} and rotation of the bucket a^{Pitch}. The fork-shaped bucket is designed to imitate the features of skeleton buckets.

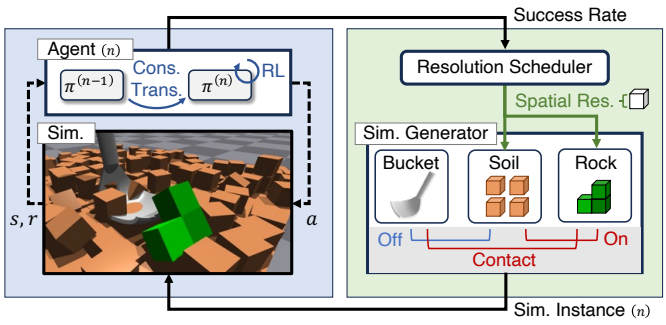

Fig. 3. Applying PRPD to our rock excavation simulator: The resolution scheduler progressively changes the simulation resolution. The simulation generator creates the environment (soil, rocks, bucket) at this resolution. At each resolution, agents collect samples and update policies.

2) Policy Update with Conservative Policy Transfer: PRPD approximately utilizes the conservative policy update scheme to stabilize policy transfer between different resolutions. This scheme was originally developed for a single environment $\mathcal{M}$ in a previous work [52]. This paper approximately utilizes this approach by assuming that a small resolution interval between environments makes these environments similar to a single environment ($\mathcal{M}^{(n)} \approx \mathcal{M}^{(n-1)}$). Therefore, it can be inferred that the lower the similarity (the larger the change in resolution), the lower the effectiveness, which will be verified in detail in Section VI-C. For this purpose, we extend the policy linear combination in Eq. (4). Specifically, whereas Eq. (4) conservatively transfers the greedy policy $\mathcal{G}(Q^\pi)$, PRPD conservatively transfers the learned previous resolution policy $\pi^{(n-1)}$ as follows:

$$\pi^{(n)}_{k+1}(s,a) \leftarrow (1-\alpha_{k+1})\pi^{(n)}_k(s,a) + \alpha_{k+1}\pi^{(n-1)}(s,a). \quad (6)$$

As shown in Eq. (5), the maximum reward and stationary distribution $d_{\pi,\mu}$ are theoretically necessary for estimating α, but obtaining these parameters in a realistic task is seldom possible. Therefore, we modify Eq. (6) to be able to infer by mini-batch of deep reinforcement learning (DRL) by following previous works [52, 53] as:

$$\begin{aligned}\alpha_{k+1} = \alpha_0\ \mathbb{E}_{s\sim\mathcal{D}^{(n)}_i}\big[&\mathbb{E}_{a'\sim\pi^{(n-1)}(s)}[Q^{(n)}_k(s,a')] \\ &- \mathbb{E}_{a\sim\pi^{(n)}_k(s)}[Q^{(n)}_k(s,a)]\big],\end{aligned} \quad (7)$$

where α_0 is a scaling coefficient, $\mathcal{D}^{(n)}$ is the rollout buffer of the current resolution, and k is an update number. Estimated α_{k+1} is utilized by clipping to $[0,1]$. This weighting scheme increases α when the current policy outperforms the previous policy, thereby making the framework robust to the poor teacher effect [54, 55] by applying distillation only when it is expected to improve the student policy, as suggested in previous work [56]. Finally, the loss function of conservative policy update is defined:

$$\begin{aligned}\mathcal{L}^{\pi^{(n)}}_{\theta^{(n)}} = \mathbb{E}_{(s,a)\sim\mathcal{D}^{(n)}_i}\big[&\mathrm{KL}\{\pi^{(n)}_k(s,a)\|| \\ &(1-\alpha_{k+1})\pi^{(n)}_k(s,a) + \alpha_{k+1}\pi^{(n-1)}(s,a)\}\big],\end{aligned} \quad (8)$$

where $\theta^{(n)}$ represents the policy network parameters for the current resolution, and KL denotes the Kullback–Leibler divergence. To estimate the policy mixture rate α, PRPD learns an auxiliary value function $Q^{(n)}$, parameterized by $\psi^{(n)}$, which is not used for PPO updates but exclusively for estimating α across resolutions. The value function $Q^{(n)}(s,a)$ is trained independently from the actor-critic structure via a temporal-difference-based loss defined as:

$$\begin{aligned}\mathcal{L}^{Q^{(n)}}_{\psi^{(n)}} = \mathbb{E}_{(s,a,r,s')\sim\mathcal{D}^{(n)}_i}\big[\{&Q^{(n)}_k(s,a) - \\ &r + \gamma\mathbb{E}_{a'\sim\pi^{(n)}(s')}[Q^{(n)}_k(s',a')]\}^2\big].\end{aligned} \quad (9)$$

Finally, the policy network is updated by combining the PPO loss (Eq. (3)) with the KL-based loss and the Q-function loss into the following overall loss function:

$$\mathcal{L}_{\theta^{(n)},\phi^{(n)},\psi^{(n)}} = \mathcal{L}^{\pi^{(n)},V^{(n)}}_{\theta^{(n)},\phi^{(n)}} + c_3\mathcal{L}^{\pi^{(n)}}_{\theta^{(n)}} + c_4\mathcal{L}^{Q^{(n)}}_{\psi^{(n)}}, \quad (10)$$

where c_3 and c_4 are hyperparameters.

V. Rock Excavation

This section outlines the task definition and the simulator developed for learning rock excavation using RL. The objective is to remove various rocks from the ground under different conditions using the bucket. To achieve this, we created both a real-world environment (Fig. 2) and a simulation environment (Fig. 3). We describe the design of excavation motions and sensors optimized with RL and formalize rock excavation as an RL problem.

A. Task Motions

The rock excavation task is defined in three steps: 1) find the rock and move the excavator to it, 2) pick up the rock from the ground, and 3) dump the scooped rock at a designated place. We focus on learning the pickup operation due to the complexity of soil and rock mechanics. To make the learning

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

TABLE II
CALCULATION TIME PER ONE-STEP SAMPLE IN EACH RESOLUTION.

Res. Δ [mm]	70	60	50	40	30	20	10
Time [ms]	0.24	0.26	0.34	0.41	0.94	2.97	5.03

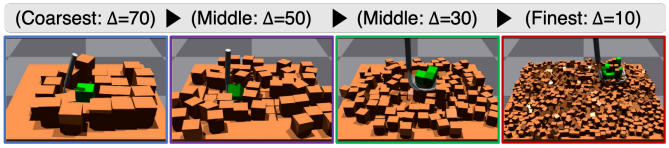

Fig. 4. Snapshots of simulation environments with each resolution

process more manageable, we observed human operators and designed parameterized motions for inclining and moving the bucket straight. The incline action allows the bucket to insert into the ground at an angle and tilt to hold the rock, while linear bucket movement helps lift the rock and adjust the relative position between the bucket and the rock. Each motion is optimized with RL.

B. Sensors and Observations

We designed three key observations for the excavation task: rock position estimated by the camera, bucket force measured by a force sensor, and bucket position and incline degree recorded by the absolute encoder. Rock position is derived from RGB images and the robot's relative position by estimating the image-based center of gravity. The bucket force observation is represented as a binary value: only the vertical force measured by the sensor is used, with a value of 1 if it exceeds a threshold (indicating contact with a rock or the ground) and 0 otherwise (indicating the bucket is in the air). The bucket's incline is limited to the scooping direction to reduce action dimensions and, if the torque limit is exceeded, the arm's movement in that direction is restricted. Each observation is recorded once at the end of the agent's action and is used as next agent's observation.

C. Learning Formulation

We formulate this rock excavation as an RL problem. The observation s is defined from the estimated rock posture (XY position and pitch rotation) p^{r}, the estimated rock force f^{r}, and bucket position p^{b}, $s = [p^{\mathrm{r}}, f^{\mathrm{r}}, p^{\mathrm{b}}]$. As supplementary information, soil details (such as appearance and particle count) differ between resolutions and exceed the assumptions of the proposed framework; to enable policy transfer across different resolutions, these details are excluded from observations. The action a is defined as $a = [a^{\mathrm{XYZ}}, a^{\mathrm{Pitch}}]$: a^{XYZ} is the move straight bucket action that represents the relative XYZ position of the bucket and a^{Pitch} is the inclining bucket action that represents the relative pitch angle of the bucket. Both a^{XYZ} and a^{Pitch} are executed simultaneously at every step t. The reward r is defined from the current rock height h^{rock}, $r = h^{\mathrm{rock}}$. The task success is evaluated by determining whether the rocks are above ground height h^{ground} using the following indicator function: $\mathbb{I}(h^{\mathrm{rock}} > h^{\mathrm{ground}})$.

D. Variable Resolution Simulator

We designed and implemented a variable-resolution simulator for rock excavation tasks using Isaac Gym [21]. The simulator is presented in Fig. 3 and enables us to remove the various rocks on various ground types by inclining and moving the bucket. The excavator's whole body model and the process of moving the excavator to the rocks and putting the rocks in other places are excluded. The rock position is obtained directly by the simulation property. 3D CAD software was utilized to design the bucket. The shape of the ground soil particle was assumed to be a box. The bucket was designed to make contact only with the rock, eliminating resistance to particles representing soil; this is for the purpose of expressing that the fork-shaped bucket can move and dig out the soil.

TABLE III
RANGE OF RANDOMIZED PARAMETERS IN THE ROCK EXCAVATION SIMULATOR: The parameters were used for learning policies with DR and sampled from the uniform distribution. "XY" and "XYZ" denote whether only the horizontal direction or the vertical direction is included, respectively.

Parameter	w/o DR	min	max
Obs. of Rock pos. noise (XY) [mm]	0	-25	25
Obs. of Rock pos. bias (XY) [mm]	0	-25	25
Error rate of rock in bucket [·]	0	0.2	0.2
Ground height bias [mm]	0	-25	25
Init. bucket pos. bias (XYZ) [mm]	0	-300	300
Init. rock pos. bias (XYZ) [mm]	0	-30	30
Bucket torque weight (XYZ) [·]	1	0.8	1.2
External force to rock [N]	0	0	1
Friction coefficient [·]	1	0.8	1.2
Total soil mass [kg]	3	2.7	3.3
Total rock mass [kg]	1	0.8	1.2
Total soil volume [mm^3]	125^3	120^3	130^3
Total rock volume [mm^3]	50^3	45^3	55^3

Our rock-excavation simulator generates environments containing soil and rocks based on the specified simulation resolution Δ. In this paper, we focus on spatial resolution, dynamically adjusting parameters such as soil particle size and count, and rock shape precision. For soil, particles are generated as boxes with side lengths represented by resolution Δ to match a predefined total soil volume. Rocks are represented as a collection of connected boxes. Specifically, boxes with side lengths represented by resolution Δ are connected face-to-face until the defined rock volume is reached. To create diverse rock shapes, the faces of the connected boxes are randomly selected.

The calculation times and environment snapshots at different resolutions for our developed simulator are shown in Table II and Fig. 4, respectively. The calculation time of this simulator improves as resolution Δ becomes finer due to more soil particles and complex rock shapes, which result in more contact points and higher calculation times, as well as increased GPU memory usage. Consequently, finer-resolution simulations require significant calculation time, making the collection of the vast number of learning samples needed for RL very challenging.

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

TABLE IV
TASK SUCCESS RATE COMPARISON AT TRAINED RESOLUTION AND AT THE FINEST RESOLUTION ($\Delta = 10$)

Trained Resolution Δ [mm]	70	60	50	40	30	20	10
Trained Resolution [%]	99.4 ± 2.6	99.1 ± 3.1	98.9 ± 2.3	98.3 ± 1.9	97.5 ± 5.6	96.9 ± 7.2	95.8 ± 6.4
$\Delta = 10$ [%]	61.1 ± 8.7	68.0 ± 8.3	76.0 ± 9.7	81.3 ± 5.8	85.0 ± 7.5	91.1 ± 8.9	95.8 ± 6.4

VI. EXPERIMENTS

We conducted experiments to validate the following objectives. PRPD can learn policies in a shorter learning time compared to previous works (Section VI-B). The conservative policy transfer scheme stabilizes learning in PRPD (Section VI-C). Differences in resolution scheduling affect policy learning (Section VI-D). Sensitivity of PRPD to loss weight scaling (Section VI-E). Trade-off between time efficiency and sample efficiency compared to other RL methods (Section VI-F). The policy trained in the simulator can be applied to various real-world environments (Section VI-G).

A. Common Settings

The experiments used a Universal Robots UR5e manipulator to control rocks with a bucket. Rock position was estimated using an Intel RealSense Depth Camera D435 and bucket force was measured with a Robotiq FT 300-S Force Torque Sensor. The robot controlled the fingertip pose at approximately 100 Hz. Policy learning performance was evaluated in simulation using an Intel Core i9-9900X CPU and GeForce RTX 3090 GPU. All experiments used RL parameters as shown in Table I and the network architecture from [57]. For real-world evaluation, nine types of rocks with different shapes and sizes were used as shown in Table V. Policy learning employed PPO with parallel environments; E environments are simulated in parallel at all resolutions, which is the maximum number of parallel environments achievable on the current PC setup for the highest resolution environment ($\Delta = 10$). Advantage A was estimated using the generalized advantage estimation scheme [58], and used in the PPO update described in Eq. (3).

This paper applies a domain randomization (DR) technique [56, 57] to learn robust policies for reality gaps between the simulation and the real world. The learned policy is then utilized in real-world environments without additional learning costs. We randomized simulation parameters during training as described in Table III. These parameters were uniformly randomized at every episode.

This experiment evaluates three key aspects. **Sample number** is defined as the total number of samples collected by the policy. **Success rate** is defined as the percentage of episodes per iteration that achieve the task, with success defined in Section V-C. **Learning time** is defined as the total training time, including both environment interactions and policy optimization.

B. Effect of Resolution Scheduling for Short-time Learning

1) Settings: PRPD schedules simulation resolution for short-time policy learning in fine-resolution simulations. To verify its effectiveness, we compare the learning time and performance between PRPD and the policy learning conducted in a fine-resolution simulation. Additionally, policy learning in only coarse-resolution simulations may achieve high performance in a shorter time. Thus, we evaluate policy learning with fixed coarse-resolutions in various patterns of resolutions. In addition, policy learning in various resolutions simultaneously may be more efficient than scheduling resolutions. This framework is also compared.

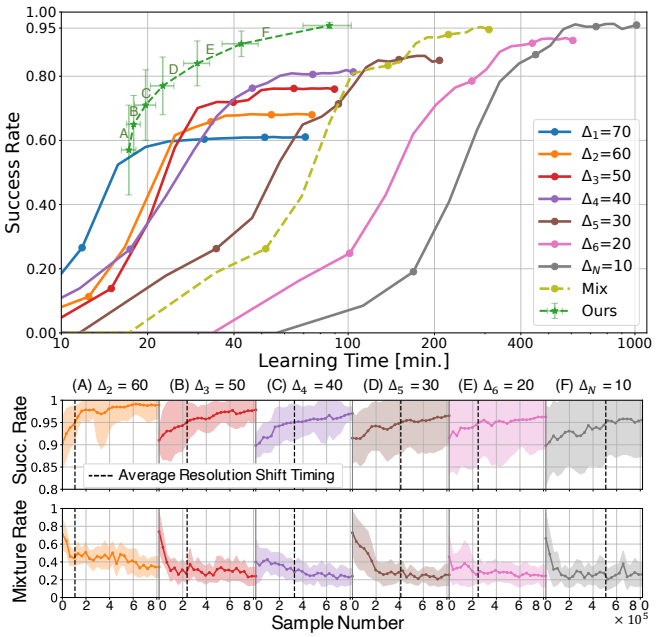

Fig. 5. Comparison of learning time (top) and policy mixture rate (bottom): **(Top)** This compares learning time between fixed resolution learning and PRPD. $\Delta = 70, \cdots, \Delta = 10$ refers to resolutions as Table II, while "Mix" refers to mixed resolutions (simultaneous policy learning), both up to $400 \times 128 \times 128$ samples (circle points are plotted every $100 \times 128 \times 128$ samples). The success rate is evaluated 100 times in $\Delta = 10$ at each iteration. **(Bottom)** This shows the task success rate and mixture rate transitions of PRPD in each resolution. The dashed line indicates when the scheduler changes resolution by achieving the target success rate $\hat{\tau}$. $\Delta = 70$ is the initial environment and lacks a previous policy, so the mixture rate is omitted. Each curve plots the mean (and variance) over five experiments.

2) Results: As shown in Fig. 5, PRPD learns control policies in less than one-seventh of the time needed to reach the highest success rate compared to fixed resolution learning. Coarser resolutions result in faster convergence, with a 20-fold difference in learning time between $\Delta = 70$ and $\Delta = 10$, but they also reduce performance by about $35\,\%$. Also, Table IV shows that each policy achieves over $95\,\%$ success rate when evaluated at its own training resolution, while the success rate at $\Delta = 10$ gradually decreases as the training resolution becomes coarser. PRPD converges in less than one-

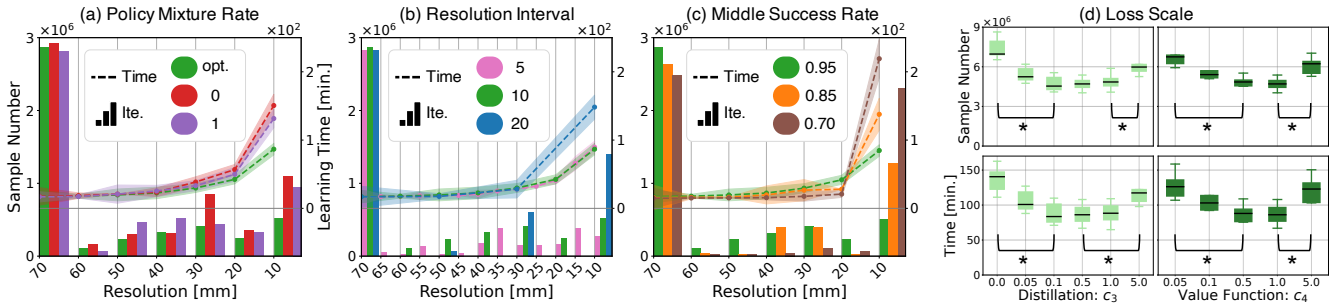

Fig. 6. Summaries of analysis: **(Left)** Performance comparison of PRPD components for (a) different patterns of policy mixture rate, (b) different numbers of grid interval, and (c) different numbers of target middle success rate. The box plots represent the sample numbers at which the policies achieve the target success rate $\hat{\tau}$ for each resolution. The dashed lines denote the total learning time from the initial resolution to the plotted resolution. α denotes the policy mixture rate, which is dynamically updated only in $\alpha = \text{opt.}$. **(Right)** (d) different scaling parameters of loss function in policy learning required to reach $\hat{\tau}$. Each curve and point of (a), (b), (c), and (d) plots the mean and variance per sample over five experiments (each box plot plots the mean). The learning sample is the value until the target success rate $\hat{\tau}$ is reached at the corresponding resolution (for Ours in (d), it is the summation value until $\hat{\tau}$ is reached at the final resolution $\Delta = 10$). $*$ mean $p < 0.05$.

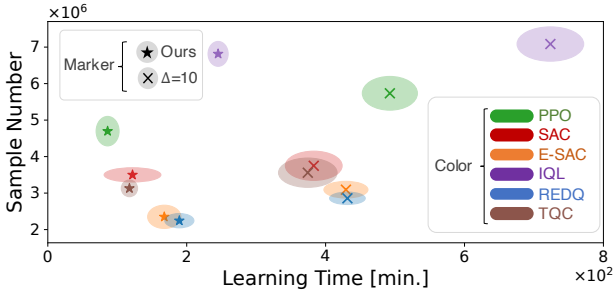

Fig. 7. Relationship between learning time and number of samples in policy learning: Each point plots the mean and variance per sample over five experiments. Sample number is the summation value until $\hat{\tau}$ is reached at the final resolution $\Delta = 10$.

fourth of the time compared to simultaneous learning across all resolutions. These results demonstrate that the proposed resolution scheduling framework achieves the fastest policy learning.

C. Effect of Conservative Policy Transfer

1) Settings: PRPD applies a conservative policy transfer scheme to stabilize policy transfers caused by progressive resolution shifts. Specifically, the learning is stabilized by dynamically optimizing the mixture rate α (Eq. (7)). To evaluate the effectiveness of optimizing α, we compared its performance with other baselines with constant α.

2) Results: As shown in Fig. 6, $\alpha = \text{opt.}$ achieves the shortest learning time compared to fixed mixture rates. Additionally, $\alpha = 1$, where previous policies $\pi^{(n-1)}$ are used without optimization, outperforms $\alpha = 0$, which does not use previous policies. These results confirm that the conservative policy transfer scheme enables learning with fewer samples.

From the transition of the policy mixture rate α in PRPD shown in Fig. 5, α tends to be high in the early learning iterations of each resolution. This denotes that, during the initial phase of progressive resolution shifts, the update of the current policy $\pi^{(n)}$ is unstable and stronger regularization is added to prevent extensive updates from the previous policy $\pi^{(n-1)}$. As learning progresses, the update of $\pi^{(n)}$ stabilizes and the regularization decreases.

D. Influence of Different Resolution Scheduling

1) Settings: Since PRPD defines the resolution scheduler deciding learning resolution in a certain resolution interval $\Delta_{\mathcal{R}}$, the interval must be small enough to ensure a successful policy transfer. Also, as the resolution scheduler decides target success rates $\hat{\tau}$ for shifting resolution, the target middle success rates, the ones used except for the final resolution, may influence sample efficiency since middle tasks are not needed to solve perfectly. Thus, we investigate the performance of PRPD with various $\Delta_{\mathcal{R}}$ and τ.

2) Results: As shown in Fig. 6, when the interval is coarse, the sample efficiency is poor, requiring about 1.5 times more samples. However, making the interval finer does not always improve efficiency, as there is almost no difference in sample efficiency when $\Delta_{\mathcal{R}} = 10$ or less. This indicates that there is an optimal interval range where policy transfer works effectively. As shown in Fig. 6, lowering the target middle success rate τ requires more samples. This suggests that policy transfer is ineffective before the policy acquires sufficient skills.

E. Impact of Loss Term Balancing in PRPD

1) Settings: To evaluate the robustness of the proposed PRPD framework with respect to loss weighting, we conduct a sensitivity analysis on the scale of each loss term in the overall objective as shown in Eq. (10). In this equation, c_3 and c_4 control the relative importance of the distillation loss and Q-value loss, respectively. This experiment systematically varies c_3 and c_4 to examine how loss balance affects training stability and performance.

2) Results: As shown in Fig. 6, the proposed framework achieves both sample- and time-efficient learning across a broad range of loss weight settings. Sample efficiency is highest when c_3 lies between 0.1 and 1, suggesting that moderate incorporation of prior policy guidance is beneficial,

IEEE Transactions on Automation Science and Engineering (T-ASE) paper, presented at ICRA 2026, Vienna, Austria.

TABLE V

SIM-TO-REAL EXPERIMENT OF ROCK EXCAVATION: This experiment evaluated nine types of rocks (five differently shaped wooden blocks and three different-sized 3D-printed artificial rocks as shown in rock images) on three types of ground (for each element, three numbers correspond to each environment, silica sand, kinetic sand, and plastic pellets in order). "Scale" indicates approximate rock size. The first-row block shows "Ours" (PRPD) and "w/o DR" (PRPD without DR). The second-row block shows PPO at a fixed resolution. Each value indicates the success number per 20 trials, with the last column showing the average rate. The 20 trials test five policies at four positions (0, 60, 120, and 180 degrees) on a 100-mm radius circumference relative to the bucket's initial position. Task success is defined as rocks being held in the bucket and the bucket being off the ground.

ID	No. 1	No. 2	No. 3	No. 4	No. 5	No. 6	No. 7	No. 8	No. 9	
Rock Image										Ave.
Scale [mm^3]	50×50×50	75×75×75	100×100×50	100×50×50	100×100×50	100×100×50	140×70×70	160×80×80	180×90×90	[%]
Ours	19-19-20	17-14-16	18-18-15	20-19-20	19-17-19	18-17-14	19-20-18	18-16-16	18-15-17	88.1
w/o DR	16-14-18	11-11-9	15-13-14	16-16-17	14-13-12	5-4-7	12-10-6	9-6-3	3-4-1	51.7
$\Delta = 10$	20-19-19	19-15-17	18-17-18	20-19-19	18-16-17	17-15-16	18-19-18	19-18-16	18-16-17	88.5
$\Delta = 20$	19-16-20	20-13-20	17-14-16	19-17-18	18-14-20	16-15-12	19-16-18	18-14-17	15-14-14	83.1
$\Delta = 30$	19-17-18	19-12-19	16-16-15	19-20-19	17-15-16	15-15-12	17-16-14	18-15-16	13-12-10	79.6
$\Delta = 40$	19-19-19	16-10-17	17-18-16	18-19-16	18-15-16	13-14-13	18-15-15	20-12-18	11-11-9	78.1
$\Delta = 50$	20-18-20	17-11-15	14-11-12	18-19-19	14-11-15	14-15-9	17-14-14	16-13-13	12-11-11	72.8
$\Delta = 60$	17-15-17	15-12-15	13-11-10	20-18-18	16-13-17	10-9-7	19-17-17	15-9-11	6-5-6	66.3
$\Delta = 70$	18-18-20	15-10-16	14-9-13	17-16-19	16-12-15	9-10-4	18-16-16	14-9-13	3-3-4	64.3

whereas excessive emphasis may hinder learning. For c_4, optimal performance is observed when the value ranges from 0.5 to 1, indicating that Q-value estimation supports conservative transfer, though overly large weights may destabilize training. While the framework remains robust to moderate variations, setting c_3 or c_4 far outside these ranges can increase the required number of samples by approximately 1.5 times.

F. Tradeoff between Time-Efficiency and Sample-Efficiency

1) Settings: As mentioned in Section II-C, various methods have improved RL's sample efficiency, but less focus has been placed on "time efficiency." This section shows that the proposed method outperforms traditional sample-efficient methods in time efficiency. We evaluate Soft Actor-Critic (SAC) [38] and its ensemble-based extension E-SAC [31], along with three recent off-policy algorithms recognized for their sample efficiency and strong empirical performance: Implicit Q-Learning (IQL) [59], Randomized Ensembled Double Q-learning (REDQ) [60], and Truncated Quantile Critics (TQC) [61]. These methods are tested within both the proposed PRPD framework and a fixed-resolution setting with $\Delta = 10$.

2) Results: As shown in Fig. 7, the proposed PRPD framework outperforms fixed resolution $\Delta = 10$ in time efficiency across all RL methods, despite being less sample-efficient. PPO, with the highest time efficiency among RL algorithms, achieved the fastest calculation time with PRPD. Under $\Delta = 10$, PPO had the longest calculation time due to its low sample efficiency. Other sample-efficient methods required less time and fewer samples than PPO, as the slow simulation time at $\Delta = 10$ increased sample collection time, making sample-efficient methods more time-efficient.

G. Sim-to-Real Policy Transfer

This section validates the sim-to-real policy transfer across diverse real-world rock-excavation environments, including silica sand (low viscosity), kinetic sand (high viscosity), and plastic pellets (blocky shape and large grain size). Table V shows the evaluation of the transfer policies. The experiment shows that the success rate increases as the resolution improves in the real-world environment for fixed resolution policies, as was demonstrated with the simulation results. PRPD achieves performance comparable to fine-resolution fixed policy learning, indicating no adverse effects from using coarse-resolution simulations and successful sim-to-real transfer. We also evaluated the differences in policy performance during sim-to-real transfer depending on the presence of DR parameters. Table V shows that policy performance dropped to approximately $50\,\%$ without DR. This indicates a reality gap between the simulation and real-world environments, mitigated by applying DR. These results demonstrate that the variable-resolution rock excavation simulator and the proposed policy learning framework achieve approximately $90\,\%$ success rates for nine types of real rock environments.

VII. DISCUSSIONS

Finer resolution assumption: In this paper, we assumed that finer resolution in excavation simulators leads to better transfer policy performance in real-world environments, supported by the experimental results in Section VI-G. However, this assumption may not always hold. The relationship between resolution and the reality gap could reverse or remain unchanged beyond a certain point. While finer resolutions may reveal physical differences, these might not significantly impact policy learning. The influence of policy generalization through DR on this relationship is also considered. These discussions are left for future work.

Automatic determination of resolution interval: As an extension of the proposed PRPD, automatic determination of resolution intervals $\Delta_{\mathcal{R}}$ could be considered. In Section VI-D, we showed that finer $\Delta_{\mathcal{R}}$ allows for learning with fewer samples. Since these parameters depend on the task and policy learning algorithm, determining a single optimal value is difficult. However, developing a framework to automatically

adjust simulation intervals based on their relationship would be beneficial.

Applicability of proposed framework: This paper focuses on rock excavation within earthwork tasks in the development of a simulator and an effective learning algorithm. The proposed RL framework of progressively changing simulation resolution for faster policy learning could be extended to other excavation tasks or the installation of non-rock objects. Additionally, this approach could broadly apply to tasks where simulation resolution affects simulation time beyond just earthwork tasks. This paper demonstrated a 7-fold improvement in learning time efficiency (Ours: 90 minutes, fixed highest resolution: 600 minutes), resulting in an 8-hour time gap. This difference is likely to become more significant with increasing state-action space complexity or task difficulty. Furthermore, while the finest resolution in this study was set to $\Delta = 10$, targeting even higher resolutions would likely widen the gap in learning time.

Simulation Enginnering Cost: In this paper, multi-resolution simulations are constructed as independent environments with different particle sizes, corresponding to varying spatial resolutions. Each resolution requires a separate simulation run with appropriately adjusted parameters to generate simulations at the corresponding resolution. We assume that such environments can be prepared in advance, and our framework is designed to efficiently learn high-resolution policies under this assumption. In practice, the engineering cost of creating variable-resolution simulators is relatively low in modern earthwork simulation platforms. For example, in particle-based simulators such as Vortex Studio [17] and OPERA [62], modifying a single configuration parameter is sufficient to control particle size and generate simulations at different resolutions.

VIII. Conclusion

In this paper, we propose time-efficient RL framework PRPD to address the time inefficiency of a fine-resolution rock-excavation simulator. We evaluated PRPD by developing a variable-resolution rock-excavation simulator using Isaac Gym. PRPD significantly reduced policy learning time in the simulator. Additionally, the learned policy was successfully transferred to the real-world environment, robustly removing previously unseen rocks.

References

[1] D. Lee, I. Jang, J. Byun, H. Seo, and H. J. Kim, "Real-Time Motion Planning of a Hydraulic Excavator Using Trajectory Optimization and Model Predictive Control," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2021, pp. 2135–2142.

[2] Y. Zhang, Z. Sun, Q. Sun, Y. Wang, X. Li, and J. Yang, "Time-Jerk Optimal Trajectory Planning of Hydraulic Robotic Excavator," *Advances in Mechanical Engineering*, vol. 13, no. 7, pp. 1–13, 2021.

[3] F. A. Bender, S. Göltz, T. Bräunl, and O. Sawodny, "Modeling and Offset-free Model Predictive Control of a Hydraulic Mini Excavator," *IEEE Transactions on Automation Science and Engineering*, vol. 14, no. 4, pp. 1682–1694, 2017.

[4] L. Zhang, J. Zhao, P. Long, L. Wang, L. Qian, F. Lu, X. Song, and D. Manocha, "An Autonomous Excavator System for Material Loading Tasks," *Science Robotics*, vol. 6, no. 55, 2021, eabc3164.

[5] A. A. Dobson, J. A. Marshall, and J. Larsson, "Admittance Control for Robotic Loading: Design and Experiments with a 1-Tonne Loader and a 14-Tonne Load-Haul-Dump Machine," *Journal of Field Robotics*, vol. 34, no. 1, pp. 123–150, 2017.

[6] S. Dadhich, U. Bodin, and U. Andersson, "Key Challenges in Automation of Earth-Moving Machines," *Automation in Construction*, vol. 68, pp. 212–222, 2016.

[7] D. A. Bradley and D. W. Seward, "The Development, Control and Pperation of an Autonomous Robotic Excavator," *Journal of Intelligent and Robotic Systems*, vol. 21, pp. 73–97, 1998.

[8] C. Tampier, M. Mascaro, and J. Ruiz-del Solar, "Autonomous Loading System for Load-Haul-Dump (LHD) Machines Used in Underground Mining," *Applied Sciences*, vol. 11, no. 18, 2021, 8718.

[9] O. M. U. Eraliev, K.-H. Lee, D.-Y. Shin, and C.-H. Lee, "Sensing, Perception, Decision, Planning and Action of Autonomous Excavators," *Automation in Construction*, vol. 141, p. 104428, 2022.

[10] F. E. Sotiropoulos and H. H. Asada, "Autonomous Excavation of Rocks Using a Gaussian Process Model and Unscented Kalman Filter," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2491–2497, 2020.

[11] H. A. Nguyen and Q. P. Ha, "Robotic Autonomous Systems for Earthmoving Equipment Operating in Volatile Conditions and Teaming Capacity: a Survey," *Robotica*, vol. 41, no. 2, pp. 486–510, 2023.

[12] S. E. Li, *Reinforcement Learning for Sequential Decision and Optimal Control*. Springer Nature, 2023.

[13] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep Reinforcement Learning for Robotic Manipulation with Asynchronous Off-Policy Updates," in *IEEE International Conference on Robotics and Automation*, 2017, pp. 3389–3396.

[14] A. Zhu, T. Dai, G. Xu, P. Pauwels, B. De Vries, and M. Fang, "Deep Reinforcement Learning for Real-time Assembly Planning in Robot-based Prefabricated Construction," *IEEE Transactions on Automation Science and Engineering*, vol. 20, no. 3, pp. 1515–1526, 2023.

[15] D. Jud, P. Leemann, S. Kerscher, and M. Hutter, "Autonomous Free-Form Trenching Using a Walking Excavator," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3208–3215, 2019.

[16] Q. Lu, Y. Zhu, and L. Zhang, "Excavation Reinforcement Learning Using Geometric Representation," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 4472–4479, 2022.

[17] K. Matsumoto, A. Yamaguchi, T. Oka, M. Yasumoto, S. Hara, M. Iida, and M. Teichmann, "Simulation-Based Reinforcement Learning Approach Towards Construction Machine Automation," in *Proceedings of the International Symposium on Automation and Robotics in Construction*, vol. 37, 2020, pp. 457–464.

[18] T. Ni, H. Zhang, C. Yu, D. Zhao, and S. Liu, "Design of Highly Realistic Virtual Environment for Excavator Simulator," *Computers & Electrical Engineering*, vol. 39, no. 7, pp. 2112–2123, 2013.

[19] H. Tahara, H. Sasaki, H. Oh, B. Michael, and T. Matsubara, "Disturbance-Injected Robust Imitation Learning with Task Achievement," in *International Conference on Robotics and Automation*, 2022, pp. 2466–2472.

[20] Y. Kadokawa, M. Hamaya, and K. Tanaka, "Learning Robotic Powder Weighing from Simulation for Laboratory Automation," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2023, pp. 2932–2939.

[21] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, *et al.*, "Isaac Gym: High Performance GPU-based Physics Simulation for Robot Learning," *arXiv preprint arXiv:2108.10470*, 2021.

[22] A. Haeri and K. Skonieczny, "Three-Dimensionsal Granular Flow Continuum Modeling via Material Point Method with Hyperelastic Nonlocal Granular Fluidity," *Computer Methods in Applied Mechanics and Engineering*, vol. 394, p. 114904, 2022.

[23] B. Son, C. Kim, C. Kim, and D. Lee, "Expert-Emulating Excavation Trajectory Planning for Autonomous Robotic Industrial Excavator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2020, pp. 2656–2662.

[24] D. Millard, D. Pastor, J. Bowkett, P. Backes, and G. S. Sukhatme, "GranularGym: High Performance Simulation for Robotic Tasks with Granular Materials," *arXiv preprint arXiv:2306.01369*, 2023.

[25] Z. Zhou, J. Song, X. Xie, Z. Shu, L. Ma, D. Liu, J. Yin, and S. See, "Towards Building AI-CPS with NVIDIA Isaac Sim: An Industrial Benchmark and Case Study for Robotics Manipulation," in *International Conference on Software Engineering: Software Engineering in Practice*, 2024, pp. 263–274.

[26] Ł. Kaiser, M. Babaeizadeh, P. Miłos, B. Osiński, R. H. Campbell, K. Czechowski, D. Erhan, C. Finn, P. Kozakowski, S. Levine, *et al.*,

"Model Based Reinforcement Learning for Atari," in *International Conference on Learning Representations*, 2020.

[27] J. Wu, Z. Huang, and C. Lv, "Uncertainty-aware Model-based Reinforcement Learning: Methodology and Application in Autonomous Driving," *IEEE Transactions on Intelligent Vehicles*, vol. 8, no. 1, pp. 194–203, 2022.

[28] X. Huang, X. Wang, Y. Zhao, J. Hu, H. Li, and Z. Jiang, "Guided Model-Based Policy Search Method for Fast Motor Learning of Robots With Learned Dynamics," *IEEE Transactions on Automation Science and Engineering*, 2024.

[29] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving Sample Efficiency in Model-Free Reinforcement Learning from Images," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 35, no. 12, 2021, pp. 10 674–10 681.

[30] M. Laskin, A. Srinivas, and P. Abbeel, "CURL: Contrastive Unsupervised Representations for Reinforcement Learning," in *International Conference on Machine Learning*, 2020, pp. 5639–5650.

[31] M. R. Maulana and W. S. Lee, "Ensemble and Auxiliary Tasks for Data-Efficient Deep Reinforcement Learning," in *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, 2021, pp. 122–138.

[32] D. Yarats, I. Kostrikov, and R. Fergus, "Image Augmentation is All You Need: Regularizing Deep Reinforcement Learning from Pixels," in *International Conference on Learning Representations*, 2021.

[33] R. Raileanu, M. Goldstein, D. Yarats, I. Kostrikov, and R. Fergus, "Automatic Data Augmentation for Generalization in Reinforcement Learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5402–5415, 2021.

[34] X. Jiang, J. Zheng, Z. Chen, Z. Ge, Z. Song, and X. Ma, "Leveraging Transfer Learning for Data Augmentation in Fault Diagnosis of Imbalanced Time-frequency Images," *IEEE Transactions on Automation Science and Engineering*, 2024.

[35] J. Huh, J. Bae, D. Lee, J. Kwak, C. Moon, C. Im, Y. Ko, T. K. Kang, and D. Hong, "Deep Learning-based Autonomous Excavation: a Bucket-trajectory Planning Algorithm," *IEEE Access*, vol. 11, pp. 38 047–38 060, 2023.

[36] W. Wang, C. Zeng, H. Zhan, and C. Yang, "A Novel Robust Imitation Learning Framework for Complex Skills with Limited Demonstrations," *IEEE Transactions on Automation Science and Engineering*, 2024.

[37] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation Learning: A Survey of Learning Methods," *ACM Computing Surveys*, vol. 50, no. 2, pp. 1–35, 2017.

[38] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," in *International Conference on Machine Learning*, 2018, pp. 1861–1870.

[39] J. Duan, Y. Guan, S. E. Li, Y. Ren, Q. Sun, and B. Cheng, "Distributional Soft Actor-Critic: Off-Policy Reinforcement Learning for Addressing Value Estimation Errors," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 11, pp. 6584–6598, 2021.

[40] Y. Song, P. N. Suganthan, W. Pedrycz, J. Ou, Y. He, Y. Chen, and Y. Wu, "Ensemble Reinforcement Learning: A Survey," *Applied Soft Computing*, p. 110975, 2023.

[41] H. P. Du, A. D. Nguyen, D. T. Nguyen, H. N. Nguyen, and D. H. Nguyen, "A Novel Deep Ensemble Learning to Enhance User Authentication in Autonomous Vehicles," *IEEE Transactions on Automation Science and Engineering*, 2023.

[42] S. Narvekar, B. Peng, M. Leonetti, J. Sinapov, M. E. Taylor, and P. Stone, "Curriculum Learning for Reinforcement Learning Domains: A Framework and Survey," *Journal of Machine Learning Research*, vol. 21, no. 181, pp. 1–50, 2020.

[43] Z. Liu, Q. Liu, L. Tang, K. Jin, H. Wang, M. Liu, and H. Wang, "Visuomotor Reinforcement Learning for Multirobot Cooperative Navigation," *IEEE Transactions on Automation Science and Engineering*, vol. 19, no. 4, pp. 3234–3245, 2021.

[44] C. Florensa, D. Held, X. Geng, and P. Abbeel, "Automatic Goal Generation for Reinforcement Learning Agents," in *International Conference on Machine Learning*, 2018, pp. 1515–1528.

[45] V. H. Pong, M. Dalal, S. Lin, A. Nair, S. Bahl, and S. Levine, "Skew-Fit: State-Covering Self-Supervised Reinforcement Learning," in *International Conference on Machine Learning*, 2020, pp. 7783–7792.

[46] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust Adversarial Reinforcement Learning," in *International Conference on Machine Learning*, 2017, pp. 2817–2826.

[47] A. Mandlekar, Y. Zhu, A. Garg, L. Fei-Fei, and S. Savarese, "Adversarially Robust Policy Learning: Active Construction of Physically-Plausible Perturbations," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2017, pp. 3932–3939.

[48] T. Matiisen, A. Oliver, T. Cohen, and J. Schulman, "Teacher-Student Curriculum Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3732–3740, 2020.

[49] S. Narvekar, J. Sinapov, M. Leonetti, and P. Stone, "Source Task Creation for Curriculum Learning," in *International Conference on Autonomous Agents and Multiagent Systems*, 2016, pp. 566–574.

[50] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement Learning in Robotics: A Survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[51] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal Policy Optimization Algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[52] N. Vieillard, O. Pietquin, and M. Geist, "Deep Conservative Policy Iteration," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 04, 2020, pp. 6070–6077.

[53] L. Zhu, T. Kitamura, and T. Matsubara, "Cautious Actor-Critic," in *Asian Conference on Machine Learning*, 2021, pp. 220–235.

[54] K.-H. Lai, D. Zha, Y. Li, and X. Hu, "Dual policy distillation," in *IJCAI*, 2020, pp. 3146–3152.

[55] W. M. Czarnecki, R. Pascanu, S. Osindero, S. Jayakumar, G. Swirszcz, and M. Jaderberg, "Distilling policy distillation," in *AISTATS*, 2019, pp. 1331–1340.

[56] Y. Kadokawa, L. Zhu, Y. Tsurumine, and T. Matsubara, "Cyclic Policy Distillation: Sample-Efficient Sim-to-Real Reinforcement Learning with Domain Randomization," *Robotics and Autonomous Systems*, vol. 165, 2023, 104425.

[57] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Sim-to-Real Transfer of Robotic Control with Dynamics Randomization," in *IEEE International Conference on Robotics and Automation*, 2018, pp. 3803–3810.

[58] J. Schulman, P. Moritz, S. Levine, M. Jordan, and P. Abbeel, "High-Dimensional Continuous Control Using Generalized Advantage Estimation," *arXiv preprint arXiv:1506.02438*, 2015.

[59] I. Kostrikov, A. Nair, and S. Levine, "Offline Reinforcement Learning with Implicit Q-Learning," *arXiv preprint arXiv:2110.06169*, 2021.

[60] X. Chen, C. Wang, Z. Zhou, and K. Ross, "Randomized ensembled double q-learning: Learning fast without a model," in *International Conference on Learning Representations*, 2021.

[61] A. Kuznetsov, P. Shvechikov, A. Grishin, and D. Vetrov, "Controlling Overestimation Bias with Truncated Mixture of Continuous Distributional Quantile Critics," in *International Conference on Machine Learning*, 2020, pp. 5556–5566.

[62] D. Endo, Y. Matsusaka, G. Yamauchi, and T. Hashimoto, "Research on an Open Source Physical Simulator for Autonomous Construction Machinery Development," in *International Symposium on Automation and Robotics in Construction*, 2024, pp. 1303–1306.