Received November 9, 2021, accepted January 2, 2022, date of publication January 6, 2022, date of current version January 13, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3140781

Deep Reinforcement Learning With Adversarial Training for Automated Excavation Using Depth Images

TAKAYUKI OSA[1,2,3] **AND MASANORI AIZAWA**[4]
[1]Department of Human Intelligence Systems, Kyushu Institute of Technology, Fukuoka 808-0135, Japan
[2]Research Center for Neuromorphic AI Hardware, Kyushu Institute of Technology, Fukuoka 808-0135, Japan
[3]RIKEN Center for Advanced Intelligence Project, Tokyo 103-0027, Japan
[4]Komatsu Ltd., Kanagawa 254-0913, Japan

Corresponding author: Takayuki Osa (osa@brain.kyutech.ac.jp)

This work was jointly supported by the Kyushu Institute of Technology and Komatsu Company Ltd. The work of Takayuki Osa was supported in part by JSPS KAKENHI under Grant JP19K20370.

ABSTRACT Excavation, which is one of the most frequently performed tasks during construction often poses danger to human operators. To reduce potential risks and address the problem of workforce shortage, automation of excavation is essential. Although previous studies have yielded promising results based on the use of reinforcement learning (RL) for automated excavation, the properties of excavation task in the context of RL have not been sufficiently investigated. In this study, we investigate Qt-Opt, which is a variant of Q-learning algorithms for continuous action space, for learning the excavation task using depth images. Inspired by virtual adversarial training in supervised learning, we propose a regularization method that uses virtual adversarial samples to reduce overestimation of Q-values in a Q-learning algorithm. Our results reveal that Qt-Opt is more sample-efficient than state-of-the-art actor-critic methods in our problem setting, and we verify that the proposed method further improves the sample efficiency of Qt-Opt. Our results demonstrate that multiple optimal actions often exist within the process of excavation and the choice of policy representation is crucial for satisfactory performance.

INDEX TERMS Construction industry, automation, machine learning, reinforcement learning.

I. INTRODUCTION

Construction often involves tasks that pose danger to human operators, and the construction industry is recently facing a shortage of the workers. To address such issues, automation of excavation has been investigated for decades [1]–[7]. Recently, machine learning and robotics have demonstrated promising results in various applications [8], [9]. Particularly, recent studies on deep reinforcement learning (RL) [10] in various applications, including robotic manipulation [11], [12] and autonomous driving [13]. In RL, the optimal policy, which maximizes the expected return, is obtained through autonomous trials and errors. Therefore, RL removes the needs for manual design of policies or an expert-demonstration dataset. Recently, Kurinov *et al.* investigated a framework for automating excavation based on deep RL [14] and demonstrated that they had successfully obtained a policy for excavation. However, in their work, the policy uses as input, a low-dimensional state vector, which is carefully designed and difficult to obtain in real-world systems. To make the policy more generalizable, training a policy that plans excavation motion based on visual information is desired.

To address this problem, we study deep RL methods for the planning of excavation motions, using depth images of the landscape. Specifically, we investigate Qt-Opt, which is a variant of Q-learning algorithms for a continuous action space, to learn the excavation task in this study. We present novel techniques to improve the sample-efficiency of Qt-Opt and demonstrate the advantages of the proposed technique in the excavation task. Inspired by virtual adversarial training (VAT) proposed in [15], we propose a regularization method using virtual adversarial samples to avoid the overestimation of the Q-values. We refer to this method as *conservative adversarial training* (CAT). Additionally, we propose a strategy for selecting actions using two critics in Qt-Opt,

The associate editor coordinating the review of this manuscript and approving it for publication was Son Lam Phung.

FIGURE 1. Excavation automation has become essential to reduce risks to humans and cope with workforce shortage.

which is less sensitive to the approximation error of the learned critic. The proposed variant of Qt-Opt is applied to autonomous excavation using the depth image of the landscape. The experimental results revealed that the proposed method significantly reduced overestimation of Q-values and improved sample efficiency of Qt-Opt. The proposed method is compared with the following state-of-the-art actor-critic methods: soft actor-critic (SAC) [16] and twin delayed deep deterministic policy gradient (TD3) [17], for our excavation task. Interestingly, our results revealed that these state-of-the-art actor-critic methods did not provide satisfactory performance and that Qt-Opt is more sample-efficient than SAC and TD3 in our problem setting. We present the multimodality of the Q-function for the excavation task and discuss why Qt-Opt outperforms SAC and TD3 for excavation tasks. We believe that this study will provide valuable insights to researchers developing deep RL algorithms and practitioners developing automated excavators using deep RL methods.

The remainder of this paper proceeds as follows. Section II describes the related work. Subsequently, in Section III, we present the background of the proposed method. The proposed method is described in Section IV, and the experimental results are presented in Section V. The characteristics of the proposed method are discussed in Section VI, and the conclusions are provided in Section VII.

II. RELATED WORK

Automation of excavation has attracted significant attention because of its expected social impact [1]–[7]. Early studies on autonomous excavation, such as [2], [5], and [6], have focused on modeling soil behavior to design an efficient scooping motion. These studies implicitly assumed that the scooping motion is analytically designed by human engineers. Moreover, it is challenging to design an optimal strategy to achieve efficient excavation. A recent study by Fukui *et al.* employed an approach based on imitation learning to automate excavation [7]. Imitation learning is an approach that obtains the optimal strategy by learning from human demonstrations [18], [19]. Fukui *et al.* proposed classifying the excavation motions demonstrated by human experts and adapting them to achieve efficient excavation [7]. However, their method requires a database of excavation motions, and it is difficult to build such a database in practice.

In RL, a policy that maximizes the expected return is obtained through trial and error [10]. This approach is especially attractive for tasks in which simulations are available, because the optimal policy can be obtained from virtual samples. Recent studies have applied deep RL to excavation tasks [14], [20]. A simulator for a bucket-leveling task was developed in a previous study [20], and the efficacy of deep RL methods was investigated. In a previous study by Kurinov *et al.* [14], a 3D simulation of the excavation task was developed, and deep RL was applied to automate the excavation task. Although they achieved promising results, the obtained policy is based on a low-dimensional state vector, which is difficult to obtain in real-world systems. To extend the applicability of the trained policy, it is necessary to investigate methods for learning a policy that uses vision-based inputs and outputs actions for the excavation task. Previous studies on autonomous excavation have often focused on learning a controller that is robust against disturbances [21], [22].

Qt-Opt was originally developed for grasping tasks that involve large-scale off-policy data collection [23]. Qt-Opt should be a reasonable choice for tasks where the Q-function is non-convex and highly complex and a simplified policy representation is not suitable. However, Qt-Opt was not directly compared with SAC and TD3 in the original study [23]. To the best of our knowledge, previous studies have not directly compared Qt-Opt with SAC and TD3 on the same task.

It is well known in the field of deep RL that there can be multiple optimal policies that elicit the optimal value function [10]. In other words, multiple optimal actions exist for a given state, although the optimal value function is unique. However, existing RL methods typically learn a policy that models the conditional distribution of actions as a unimodal distribution [16], [17]. This simplified policy model may not be sufficient for tasks in which there are several optimal actions for a specified state. Our study demonstrates that the excavation process is one such task and the performance of RL methods is significantly affected by the flexibility of the policy model.

Virtual samples that are generated by injecting noise into samples in a given dataset are often called *adversarial examples* [15], [24], [25]. Previous research revealed that neural networks are often vulnerable to small noise injected into samples [24] and that regularization using adversarial samples can improve the generalization performance [15], [25].

In the deep RL literature, previous studies [26]–[28] investigated the sensitivity of RL agent against adversarial perturbations or adversarial agents. However, these studies did not address how to improve the learning performance of RL agents. Consequently, it is not clear how to leverage adversarial examples for training RL agents. Recent studies [29], [30] proposed methods for obtaining a robust policy by jointly training an adversarial agent. In these studies [29] and [30], the term "adversary" represents an adversarial agent, and not adversarial examples. Although the methods in these studies are applicable to control problems wherein we can introduce an adversarial agent that disturbs the dynamics, it is difficult to apply them to planning problems wherein it is not clear how to define an adversarial agent.

III. BACKGROUND

First, we introduce the problem formulation of RL. Subsequently, we introduce Qt-Opt and TD3, which are the state-of-the-art RL algorithms, because it is essential to understand the similarities and differences between Qt-Opt and TD3 to interpret this study. Additionally, we briefly describe the VAT proposed in [15].

A. REINFORCEMENT LEARNING

We consider an RL problem under a Markov decision process (MDP) defined by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, d)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P}(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$ is the transition probability density, $r(\boldsymbol{s}, \boldsymbol{a})$ is the reward function, γ is the discount factor, and $d(\boldsymbol{s}_0)$ is the probability density of the initial state. A policy $\pi(\boldsymbol{a}|\boldsymbol{s}) : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is defined as the conditional probability density over actions given states. The cumulative discounted reward given by $R_t = \sum_{k=t}^{T} \gamma^{k-t} r(\boldsymbol{s}_k, \boldsymbol{a}_k)$ is often referred to as a return. RL aims to obtain a policy that maximizes the expected return, $\mathbb{E}[R_0|\pi]$. The expected return after taking action $\boldsymbol{a}$ under state $\boldsymbol{s}$ and then following policy π is denoted by $Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})$, and it is called the Q-function. In deep RL, the Q-function is often approximated using a neural network. We refer to the neural network that approximates the Q-function as the *critic* in this study. In the following sections, we consider off-policy RL methods, which train a policy using samples stored in a replay buffer $\mathcal{D} = \{(\boldsymbol{s}_i, \boldsymbol{a}_i, \boldsymbol{s}'_i, r_i)\}_{i=1}^{N}$.

B. QT-OPT: Q-LEARNING FOR CONTINUOUS CONTROL

In deep Q-learning (DQN) presented by Mnih *et al.* [31], the optimal Q-function is directly approximated using a neural network. Herein, $Q_{\boldsymbol{w}}$ denotes the critic parameterized with a vector $\boldsymbol{w}$, and $\boldsymbol{w}$ is updated by minimizing the following objective function:

$$\boldsymbol{w}^* = \arg\min \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}',r)\sim\mathcal{D}} \left[(Q_{\boldsymbol{w}}(\boldsymbol{s}, \boldsymbol{a}) - y_i)^2 \right], \tag{1}$$

where y_i is the target value given by

$$y_i = r + \gamma \max_{\boldsymbol{a}'} Q_{\boldsymbol{w}'}(\boldsymbol{s}', \boldsymbol{a}'), \tag{2}$$

and $\boldsymbol{w}'$ is the parameter of the target network, which maintains the old parameter of the critic. DQN is developed for tasks with discrete actions, wherein it is simple to compute $\max_{a'} Q_{\boldsymbol{w}'}(s', a')$. However, when the action space is continuous, it is not easy to compute $\max_{a'} Q_{\boldsymbol{w}'}(s', a')$. To mitigate this issue, $\max_{a'} Q_{\boldsymbol{w}'}(s', a')$ is approximated using the cross-entropy method (CEM) [32], [33] in Qt-Opt. CEM is a black-box optimization method that can be applied to arbitrary functions. In Qt-Opt, to avoid the overestimation bias in Q-learning and stabilize the learning process, the target Q value is computed using two models of the Q-function as

$$y_i = r + \gamma \min_{i=1,2} Q_{\boldsymbol{w}_i}(\boldsymbol{s}', \boldsymbol{a}'), \tag{3}$$

where $Q_{\boldsymbol{w}_1}$ and $Q_{\boldsymbol{w}_2}$ represent two separate models for approximating the Q-function, and $\boldsymbol{a}'$ is given by

$$\boldsymbol{a}' = \arg\max_{\tilde{\boldsymbol{a}}} Q_{\boldsymbol{w}_1}(\boldsymbol{s}', \tilde{\boldsymbol{a}}), \tag{4}$$

which is determined by CEM. For CEM, we need to prepare an initial sampling distribution, for example, Gaussian distribution. At each iteration, N actions are randomly generated using the sampling distribution. Then, the sampling distribution is fitted to the best M actions, which have the highest estimated Q-values. In the original Qt-Opt, two iterations are performed with $M = 5$ and $N = 64$. Although CEM is used to determine the action that maximizes the Q-function, other black-box optimization methods can also be applied.

C. ACTOR-CRITIC METHODS

When policy $\pi(\boldsymbol{a}|\boldsymbol{s}, \boldsymbol{z})$ is deterministic, $\pi(\boldsymbol{a}|\boldsymbol{s})$ is a Dirac-delta function that satisfies

$$\int Q(\boldsymbol{s}, \boldsymbol{a})\pi(\boldsymbol{a}|\boldsymbol{s})\mathrm{d}\boldsymbol{a} = Q(\boldsymbol{s}, \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s})), \tag{5}$$

where $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s}) : \mathcal{S} \mapsto \mathcal{A}$. Silver *et al.* [34] proposed using the following objective function to train a deterministic policy:

$$J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{s}\sim\beta(\boldsymbol{s})} \left[Q^{\pi}(\boldsymbol{s}, \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s})) \right], \tag{6}$$

where $\beta(\boldsymbol{s})$ is the distribution of the states induced by a behavior policy for collecting the state action pairs. A deterministic policy can be updated using the deterministic policy gradient (DPG) algorithm

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{s}\sim\beta(\boldsymbol{s})} \left[\nabla_{\boldsymbol{a}} Q^{\pi}(\boldsymbol{s}, \boldsymbol{a})|_{\boldsymbol{a}=\boldsymbol{\mu}(\boldsymbol{s})} \nabla_{\boldsymbol{\theta}} \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s}) \right]. \tag{7}$$

The performance of the DPG algorithm has been demonstrated in various studies [17], [35]. In TD3, two critic networks were introduced to mitigate the issue of the overestimation of the Q-function owing to the maximization bias in Q-learning [10], [36]. The target value of the action-value function is computed as

$$y = r + \gamma \min_{i=1,2} Q_{\boldsymbol{w}'_1} \left(\boldsymbol{s}', \boldsymbol{a}' \right) \tag{8}$$

where

$$\boldsymbol{a}' = \boldsymbol{\mu}_{\boldsymbol{\theta}}(\boldsymbol{s}) + \boldsymbol{\epsilon}, \; \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{\sigma}), \tag{9}$$

and noise $\boldsymbol{\epsilon}$ is generated from the Gaussian distribution. In SAC [16], a stochastic policy is modeled using a reparameterization trick given by

$$\boldsymbol{a} = \boldsymbol{\mu}(\boldsymbol{s}) + \boldsymbol{\epsilon} \odot \boldsymbol{\sigma}(\boldsymbol{s}), \tag{10}$$

where $\boldsymbol{\mu}(\boldsymbol{s})$ and $\boldsymbol{\sigma}(\boldsymbol{s})$ are the mean and standard deviation of the action for a given state $\boldsymbol{s}$, respectively. We express the element-wise product with $\odot$ in (10). SAC uses the entropy-regularized value function; therefore the target value for the critics is given by:

$$y = r + \gamma \min_{i=1,2} Q_{\boldsymbol{w}'_1}\left(\boldsymbol{s}', \boldsymbol{a}')\right) + \alpha \log \pi(\boldsymbol{a}|\boldsymbol{s}), \tag{11}$$

where α is a constant that balances the task reward and the entropy term.

The difference between Qt-Opt and TD3 is the policy representation; Qt-Opt determines the action without any explicit model of a policy by searching for the action that maximizes the Q-value with CEM, whereas TD3 explicitly models the policy with a neural network. As described later, this difference is crucial in the excavation task.

D. VIRTUAL ADVERSARIAL TRAINING

In this study, we adapt VAT to avoid the overestimation of the Q-values in Qt-Opt. To make this paper self-contained, we introduce the VAT proposed in [15]. VAT was developed for supervised and semi-supervised learning, and the study in [15] investigated the problems of training a conditional probability density model $p_{\boldsymbol{\theta}}(\boldsymbol{y}|\boldsymbol{x})$, parameterized with a vector $\boldsymbol{\theta}$ for given input-output pairs $\{\boldsymbol{x}_i, \boldsymbol{y}_i\}_{i=1}^{n}$. In VAT, the perturbation is generated in the direction in which the change in the distribution is the largest. When the divergence between two distributions p and p' can be quantified using a non-negative function $D[p, p']$, the perturbation is generated as follows:

$$\boldsymbol{d}_{\text{adv}} = \arg\max_{\boldsymbol{d}} D\left[p(\boldsymbol{y}|\boldsymbol{x}), p(\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{d}))\right] \tag{12}$$

subject to $\|\boldsymbol{d}\|_2^2 < \epsilon$, and ϵ is the step size of the perturbation. Although there is no closed form for the adversarial perturbation in (12), the perturbation, $\boldsymbol{d}_{\text{adv}}$, can be approximated as

$$\boldsymbol{d}_{\text{adv}} \approx \epsilon \frac{\boldsymbol{g}}{\|\boldsymbol{g}\|_2} \tag{13}$$

$$\boldsymbol{g} = \nabla_{\boldsymbol{d}} D\left[p(\boldsymbol{y}|\boldsymbol{x}), p(\boldsymbol{y}|\boldsymbol{x}+\boldsymbol{d})\right] |_{\boldsymbol{d}=\xi \boldsymbol{u}}, \tag{14}$$

where $\boldsymbol{u}$ is a randomly generated unit vector and ξ is a constant for computing a finite difference. Using the obtained adversarial perturbation $\boldsymbol{d}_{\text{adv}}$, the model is trained by minimizing the following objective function:

$$\ell(\boldsymbol{\theta}) + \mathcal{L}_{\text{adv}}(\boldsymbol{\theta}), \tag{15}$$

where $\ell(\boldsymbol{\theta})$ is the negative log-likelihood and $\mathcal{L}_{\text{adv}}(\boldsymbol{\theta})$ is the regularization term using the adversarial perturbation, $\boldsymbol{d}_{\text{adv}}$.

The aim of the VAT proposed in [15] is to smooth the output distribution; therefore, the adversarial perturbation is generated in the direction in which the change in the distribution is the largest. In this study, virtual adversarial samples were employed to avoid overestimation of the Q-values in Q-learning algorithms. Consequently, we propose a method that generates the adversarial perturbations in the direction in which the Q-value is likely to be overestimated.

IV. PROPOSED METHOD

In RL, the optimal action, $\boldsymbol{a}^*$, is given by

$$\boldsymbol{a}^* = \arg\max_{\boldsymbol{a}} Q^*(\boldsymbol{s}, \boldsymbol{a}), \tag{16}$$

where $Q^*(\boldsymbol{s}, \boldsymbol{a})$ is the optimal Q-function [10]. In other words, the optimal action is the extremum of the optimal Q-function. Therefore, when we model a policy explicitly, the policy should be trained so as to approximate the location of the extremum of the Q-function.

The architecture of the critics for Qt-Opt and TD3 is identical in our implementation, and the difference between Qt-Opt and TD3 is in the policy representation. In TD3, a deterministic function with a single output is used to approximate the action that maximizes the Q-function. Therefore, if the Q-function has multiple separate extrema, the policy may not have sufficient flexibility to represent the extrema of the Q-function. In contrast, in Qt-Opt, the action that maximizes the Q-function is approximately determined using the CEM. As shown in previous studies [32], [33], the CEM can deal with an objective function with multiple extrema. Therefore, the action that maximizes the Q-function can be approximated even if the Q-function has multiple extrema. Consequently, the policy representation in Qt-Opt is more flexible than that in TD3. For this reason, we employed Qt-Opt as the base algorithm. The difference in the performance of Qt-Opt and TD3 is discussed in Section V. To improve the sample efficiency, we introduce two techniques: 1) the regularization method for avoiding overestimation of Q-values, and 2) the strategy for selecting the action using two critics in Qt-Opt.

A. CONSERVATIVE ADVERSARIAL TRAINING FOR AVOIDING OVERESTIMATION OF Q-VALUES

Previous studies on RL pointed out that overestimation of Q-values often occurs in the variants of Q-learning algorithm [10], [36], [37]. Although the double-clipped Q-learning incorporated in Qt-Opt is a method for mitigating overestimation, we observed overestimation of Q-values in the training process of Qt-Opt in our preliminary experiment. When overestimation occurs, the Q-values estimated by the critic rapidly surge and then decrease at the beginning of the training process [10]. To alleviate the issue of overestimation of Q-values, we leverage a regularization method using virtual adversarial samples.

It is well-known that statistical models often suffer from overfitting. To mitigate this issue, previous studies have proposed regularization methods [38]–[40]. Given the input-output pairs, previous studies have proposed the use of virtual adversarial samples by injecting noise into the given input samples and encouraging the model to generate outputs similar to the original outputs [15], [25], [41]. These regularization methods can be used to smooth the output of neural networks and improve their generalization performance. Inspired by these previous studies, we employed regularization using virtual adversarial samples. Although existing methods using virtual adversarial samples are designed to enhance the smoothness of the output from the trained model [15], [25], we propose a strategy to avoid overestimation of Q-values.

While employing VAT, there are two important design choices: 1) how to generate adversarial perturbations and 2) how to set target values when adversarial samples are used as inputs to a model. In supervised learning, pairs of input and target values are given as a training dataset. However, the adversarial samples are not actually observed data; therefore, the target value is not known when these samples are used as inputs to the model.

In our proposed method, we compute the adversarial perturbation to the state variable that leads to the highest estimated Q-value in the neighbor of the actual sample as

$$\boldsymbol{d}^{i}_{\text{adv}} = \arg\max_{\boldsymbol{d}} Q_{\boldsymbol{w}_i}(\boldsymbol{s} + \boldsymbol{d}, \boldsymbol{a}), \tag{17}$$

subject to $\|\boldsymbol{d}\|_2^2 < \epsilon$ for $i = 1, 2$. Based on the discussion in [15], $\boldsymbol{d}^{i}_{\text{adv}}$ can be approximated by

$$\boldsymbol{d}^{i}_{\text{adv}} \approx \epsilon \frac{\boldsymbol{g}^i}{\|\boldsymbol{g}^i\|_2}, \tag{18}$$

where $\boldsymbol{g}^i$ is the derivative of the Q-function with respect to the state, and it is given by

$$\boldsymbol{g}^i = \nabla_{\boldsymbol{d}} Q_{\boldsymbol{w}_i}(\boldsymbol{s} + \boldsymbol{d}, \boldsymbol{a})|_{\boldsymbol{d}=\xi\boldsymbol{u}}. \tag{19}$$

Here, $\boldsymbol{u}$ is a randomly generated unit vector, and ξ is a constant, and $\xi = 1 \cdot 10^{-6}$ in our implementation. The motivation for choosing the perturbation in (18) is to identify the perturbation that is likely to induce the overestimation of the Q-values. The adversarial examples based on this perturbation are then used to encourage the critic to make a conservative estimation.

For computing the virtual target value, we use the following equation:

$$y_{\text{reg}} = \min_{i=1,2} Q_{\boldsymbol{w}'_i}(\boldsymbol{s} + \boldsymbol{d}^{i}_{\text{adv}}, \boldsymbol{a}). \tag{20}$$

The critics are then trained by minimizing the following regularization term using the approximated perturbation direction $\boldsymbol{d}^{i}_{\text{adv}}$:

$$\mathcal{L}_{\text{reg}}(\boldsymbol{w}_i) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}}\left[\left(y_{\text{reg}} - Q_{\boldsymbol{w}_i}(\boldsymbol{s} + \boldsymbol{d}^{i}_{\text{adv}}, \boldsymbol{a})\right)^2\right]. \tag{21}$$

Previous studies used the current estimate to generate a virtual target label in semi-supervised learning [15]. In our framework, it is essential to generate a virtual target value that does not intensify overestimation of Q-values. In [17], Fujimoto *et al.* showed that overestimation of Q-values can be mitigated using target values that are computed based on the minimum of two critics in Q-learning algorithms. Inspired by this strategy in [17], we generate a virtual target for adversarial examples using the minimum of the two target critics. When the two critics generate different Q-values for the same state, the higher estimated Q-value may be a result of overestimation. We can mitigate overestimation by using the lower estimated Q-value as the target value for adversarial examples. In our implementation, we have two critics to perform double-clipped Q-learning [17], and the adversarial perturbation is computed for each critic. Although perturbation was generated equally in all directions in the early work on training with virtual samples in [41], recent studies on VAT [15], [25] have revealed that anisotropic perturbation should be used to further improve the performance. In our method, we generate perturbation in the direction in which the Q-function is disturbed most significantly, and the virtual target value is computed to avoid overestimation of Qvalues. We refer to the proposed regularization technique as *conservative adversarial training* (CAT). The experimental results confirm that our strategy significantly reduces overestimation in Qt-Opt.

B. ACTION SELECTION BY MAXMIN OF DOUBLE CRITICS

In the original Qt-Opt, the action is determined by identifying the extrema for one of the approximated Q-functions, similar to that in (4). However, we propose using the action given by

$$\boldsymbol{a} = \arg\max_{\tilde{\boldsymbol{a}}} \min_{i=1,2} Q_{\boldsymbol{w}_i}(\boldsymbol{s}, \tilde{\boldsymbol{a}}), \tag{22}$$

which is the extreme of the minimum of the two approximated Q-functions. Action selection with (4) does not leverage double critic architecture and is continues to be sensitive to the function approximation error of the critic used for the action selection. Our strategy for action selection in (22) should be less sensitive to the function approximation error of either of the critics. We refer to this action selection strategy as the *maxmin action selection* strategy in the remainder of this paper.

We employ maxmin action selection to compute the target value of the Q-function and determine the action while collecting the data during the training process. Therefore, the objective function for training the critics is given by:

$$\mathcal{L}_Q(\boldsymbol{w}_i) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a},\boldsymbol{s}',r)\sim\mathcal{D}}\left[\left(y - Q_{\boldsymbol{w}_i}(\boldsymbol{s}, \boldsymbol{a})\right)^2\right] \tag{23}$$

for $i = 1, 2$, where the target value for the critic is computed as

$$y = r + \gamma \max_{\tilde{\boldsymbol{a}}} \min_{i=1,2} Q_{\boldsymbol{w}'_i}(\boldsymbol{s}, \tilde{\boldsymbol{a}}). \tag{24}$$

Although the maxmin action strategy can select the action in a more stable manner than the strategy in the original Qt-Opt, it may intensify overestimation of Q-values because the following equation holds in general:

$$\max_{\tilde{\boldsymbol{a}}} \min_{i=1,2} Q_{\boldsymbol{w}'_i}(\boldsymbol{s}', \tilde{\boldsymbol{a}}) \geq \min_{i=1,2} Q_{\boldsymbol{w}_i}(\boldsymbol{s}', \arg\max_{\tilde{\boldsymbol{a}}} Q_{\boldsymbol{w}_1}(\boldsymbol{s}', \tilde{\boldsymbol{a}})). \tag{25}$$

In (25), the right-hand side is the second term in (3), which is used to compute the target value in the original Qt-Opt, whereas the left-hand side is the second term in (24), which is used to compute the target value in the proposed maxmin action strategy. This relationship shows that the target value of the Q-value computed using the maxmin action selection strategy is always greater than or equal to the target value computed using the strategy in the original Qt-Opt. Therefore, the estimated Q-value will be greater when we use the proposed maxmin action selection strategy compared with the original Qt-Opt. However, as shown in the experiment, overestimation of Q-values can be significantly reduced when combined with the adversarial training proposed in the previous section.

C. USING THE SAMPLE WITH THE MAXIMUM SCORE IN CEM

In the standard CEM, N samples are generated randomly using the sampling distribution at each iteration, and the sampling distribution is fitted to the best M samples. The sampling distribution is typically a unimodal Gaussian distribution, and the output of the CEM is the mean of the best M samples at the last iteration. However, when the objective function has multiple extrema, taking the mean of the best M samples is not always appropriate because the distribution of the best M samples may not be Gaussian. In previous studies, multimodal sampling distributions, such as Gaussian mixtures, were used to identify multiple extrema of the objective function [33]. However, setting an appropriate number of Gaussian components is often challenging. In our framework, it is not necessary to identify all extrema, and we need to identify only one of the extrema. Therefore, we simply used the sample with the maximum value obtained in the CEM as the output of the CEM. In this approach, if we generate sufficiently dense samples around the extrema, we can obtain an approximate solution regardless of the number of extrema. The experimental results in Section V reveal that the use of our implementation of CEM outperforms that of the standard CEM in our framework.

D. ALGORITHM

Based on the abovementioned discussion above, we train the critics by minimizing the following objective function:

$$\mathcal{L}(\boldsymbol{w}_i) = \mathcal{L}_Q(\boldsymbol{w}_i) + \mathcal{L}_{\mathrm{reg}}(\boldsymbol{w}_i) \tag{26}$$

for $i = 1, 2$, where $\mathcal{L}_Q(\boldsymbol{w}_i)$ and $\mathcal{L}_{\mathrm{reg}}(\boldsymbol{w}_i)$ are obtained by (23) and (21), respectively.

The proposed algorithm is summarized in Algorithm 1. We used the ϵ-greedy strategy for exploration [10] and linearly decreased the value of ϵ during the training process. The critics are updated once after every step if the replay buffer contains a sufficient number of samples. In our implementation, the target critic is updated after every step using the soft update, similar to that in [35]. The two terms in the objective function for the critics are minimized separately and alternatively, as described in Algorithm 1.

Algorithm 1 Qt-Opt With Conservative Adversarial Training & Maxmin Action Selection Strategy

```
Input: Schedule of $\epsilon$ for the $\epsilon$-greedy exploration
Initialize the experience replay buffer, $\mathcal{D}$, and the parameters for critics and target critics, $\boldsymbol{w}_i$, $\boldsymbol{w}'_i$ for $i = 1, 2$.
for each episode do
    for $t = 0$ to $T$ do
        generate random value $x \in [0, 1]$
        if $x < \epsilon$ then
            Select action randomly
        else
            Select action using the strategy in (22)
        end if
        Observe reward $r$ and new state $\boldsymbol{s}'$
        Store tuple $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}', r)$ in $\mathcal{D}$
        Sample mini-batch from $\mathcal{D}$
        Generate adversarial state samples $\boldsymbol{s}_{\mathrm{adv}}$
        Update the critics by minimizing $\mathcal{L}_Q(\boldsymbol{w}_i)$ in (23)
        Update the target critics by $\boldsymbol{w}'_i \leftarrow (1-\tau)\boldsymbol{w}'_i + \tau \boldsymbol{w}_i$
        Update the critics by minimizing $\mathcal{L}_{\mathrm{reg}}$ in (21)
    end for
end for
```

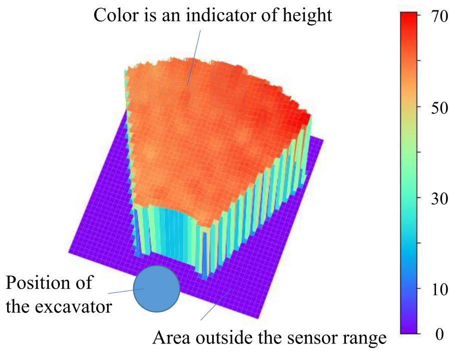


FIGURE 2. Example of the initial state of the excavation task. The state is represented as a depth map of the landscape.

V. EXPERIMENTS

A. SETUP OF SIMULATION

In our experiments, we used a 3D excavation simulator developed by Komatsu Ltd. In the simulation, we can obtain a depth image of the landscape, and the goal of the excavation task is to remove soil from the targeted area while retaining the soil near the excavator. In our implementation, the policy was trained to plan the target trajectory for the excavator bucket. The state is given by a depth image that captures the landscape in front of the excavator. A 3D plot of the state is presented in Figure 2. The dimensions of the state were 65 × 84 in the implementation. The shape of the range of the depth sensor is a sector of a circle, as shown in Figure 2, and the area outside the sensor range is set to 0 so that the shape of the state a rectangle. The state of the landscape was randomly initialized at the beginning of each episode.

The action is given by the parameter of the target trajectory of the bucket. The trajectory of the bucket is approximated as an arc, and the action space is continuous and three-dimensional. The trajectory of the bucket was constrained by the soil hardness in our simulation. The arm of the excavator is controlled by a controller that tracks the planned trajectory,

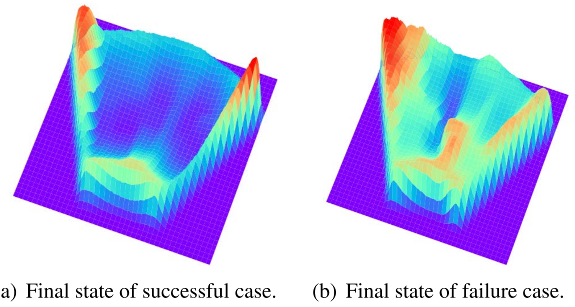

(a) Final state of successful case. (b) Final state of failure case.

FIGURE 3. **Successful and failure cases of the excavation task. The excavation should be performed considering the form of the remaining soil.**

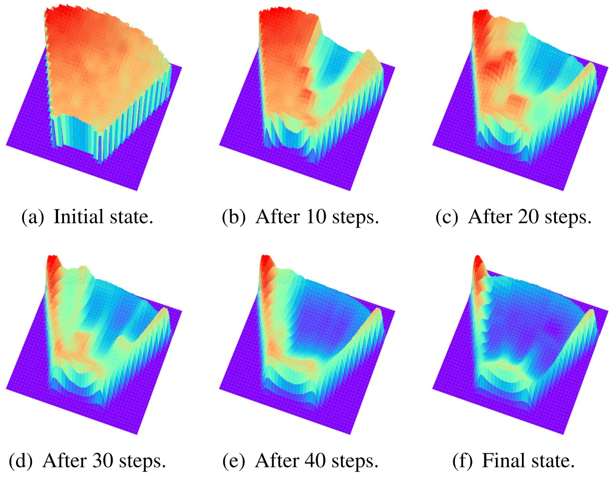
and we assume that the controller for the arm is predefined. Therefore, when an action is determined by a policy, a target trajectory for the arm is generated, and the excavator arm is controlled to achieve the planned target trajectory. The motion of the excavator arm is dependent on soil parameters, such as hardness, and these parameters were fixed in this study.

In the simulation, the reward is the amount of soil excavated by the machine. An episode is considered finished when the amount of soil is lesser than a threshold, which means that the excavator bucket is more than half empty. We regard the episode as successfully finished when 90% of the soil is removed by the end of the episode, as shown in Figure 3(a). For efficient soil excavation, the bucket must be nearly full. However, if the excavation is not performed in an appropriate order, the bucket does not fill fully because of the form of the remaining soil. An example of a failure case is depicted in Figure 3(b). In Figure 3(b), the form of the remaining soil is misshaped, and it is necessary to refine the form by a small amount of excavation. Such an action is inefficient because the bucket will not get full, and the excavation time will increase. Therefore, it is necessary to avoid states such as the one shown in Figure 3(b) and excavate the soil in an appropriate order to efficiently remove all soil from the target area. An example of successful excavation is shown in Figure 4. In the successful episode, the excavated area was gradually extended and the form of the remaining soil was nearly flat at the end of the episode.

B. BASELINE METHODS

We evaluated methods TD3 [17], SAC [16], and the original Qt-Opt [23] as baseline methods. The implementation of TD3 and SAC was adapted from SpinningUp [42], and the structure of the neural networks was modified to deal with depth-image inputs. In our implementation, the critic structure is the same for Qt-Opt, TD3 and SAC. We used convolution layers to process a depth image as an input. The structure of the critic is shown in Figure 5.

The training process was performed five times with different random seeds for each method, and the averaged test return was reported, where the test return was computed once every 5,000 time steps by executing 10 episodes without exploration. We also evaluated a metric, task progress, which indicates the amount of soil removed from the target area when the episode is terminated. This metric indicates how the trained policy can perform a task without failure. An episode for which task progress is more than 90% is regarded as successful. All experiments were run with a single GeForce GTX 3090 GPU and an Intel Core i9-10900K CPU at 3.7GHz.


(a) Initial state. (b) After 10 steps. (c) After 20 steps.

(d) After 30 steps. (e) After 40 steps. (f) Final state.

FIGURE 4. **Sequence of states in a successful case in the excavation task. The excavated area was gradually extended, and the form of the remaining soil was nearly flat at the end of the episode.**

To compare different strategies for generating adversarial examples and their virtual target values, we considered the adversarial perturbation as

$$\boldsymbol{d}^i_{\mathrm{adv}} = \arg\max_{\boldsymbol{d}} D_{\mathrm{MSE}}\big(Q_{\boldsymbol{w}_i}(\boldsymbol{s}+\boldsymbol{d}, \boldsymbol{a}), Q_{\boldsymbol{w}_i}(\boldsymbol{s}, \boldsymbol{a})\big). \quad (27)$$

Here, $D_{\mathrm{MSE}}(x, y)$ represents the mean-squared error between x and y. The motivation of this approach is to identify the perturbation that is likely to induce the largest disturbance to the Q-value. This approach is a straightforward adaptation of the existing method for generating adversarial perturbations proposed in [27]. We refer to this variant of our method as *smooth adversary* in the following section. Furthermore, we refer to the method for generating adversarial samples in (17) as *max-Q adversary*.

Additionally, we consider an alternative method for setting the virtual target values for adversarial samples as follows:

$$y_{\mathrm{reg}} = \min_{i=1,2} Q_{\boldsymbol{w}_i}(\boldsymbol{s}, \boldsymbol{a}). \quad (28)$$

In this approach, the approximated Q-function is encouraged to be smooth and insensitive to adversarial perturbations. This approach is a straightforward adaptation of the existing method of VAT in [15]. We refer to this variant of our method as *smooth virtual target* in the following section. Similarly, the method for computing the virtual target value in (20) is referred to as *conservative virtual target*. Although there are several ways to impose the norm constraint on adversarial

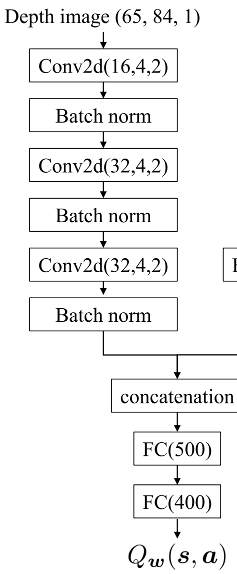


FIGURE 5. Architecture of the critic.

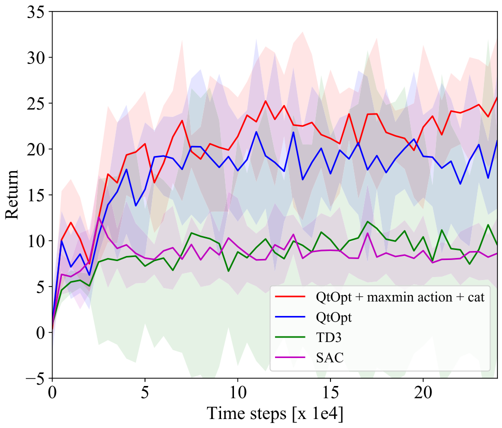


FIGURE 6. Returns during training with the proposed and baseline methods. The proposed method outperformed baseline methods such as TD3, SAC, and Qt-Opt.

perturbation [27], identifying the best norm constraint is beyond the scope of this study. We employ the method in [15] because it provides state-of-the-art performance in general machine learning tasks.

C. LEARNING CURVE

We evaluated the learning curves of the proposed and baseline methods. In this experiment, we evaluated the effect of the proposed regularization method given in Section IV-A and the maxmin action selection strategy proposed in Section IV-B. We refer to the variant of Qt-Opt with the maxmin action strategy as *Qt-Opt+maxmin_action*. Similarly, we refer to the variant of Qt-Opt combined with both the proposed regularization method and maxmin action selection strategy as *Qt-Opt+maxmin_action+cat*.

The learning curves of the proposed and the baseline methods for the excavation task are shown in Figure 6.

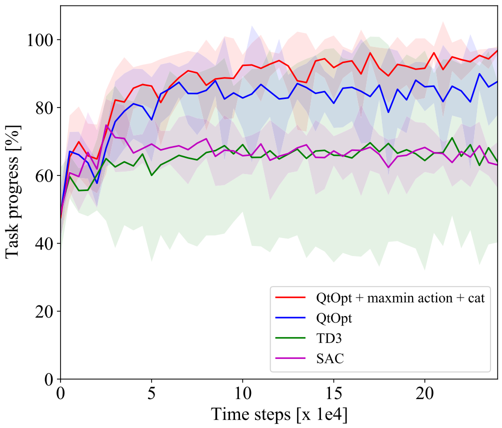


FIGURE 7. Task progress during training with the proposed and baseline methods.

As seen from the figure, the proposed method, Qt-Opt+maxmin_action+cat, outperformed the baseline methods, including TD3, SAC, and Qt-Opt. The results reveal that the proposed technique significantly improves the performance of Qt-Opt. It is remarkable that Qt-Opt outperformed TD3 and SAC in the excavation task. Neither TD3 nor SAC achieved a performance comparable to Qt-Opt; moreover, they did not improve the performance appropriately during the training. In contrast, Qt-Opt improved the performance steadily during the learning process. The only difference between Qt-Opt and TD3 is the policy representation, and this result demonstrates the importance of the flexibility of the policy model in the excavation task.

A comparison between the proposed and the baseline methods is summarized in Table 1. Although the proposed method is computationally expensive than the baseline methods, the final performance of the policy trained with the proposed method clearly outperforms that of the policies trained with the baseline methods. The success rate indicates the ratio of successful episodes during the test of the trained policy. When policies were trained by TD3 or SAC, the task progress did not reach 90% at the end of the episode. In contrast, policies trained with the proposed method steadily achieved a task progress greater than 90%. The proposed method clearly outperformed the baseline methods in terms of return, task progress, and success rate.

The learning curves of the variants of Qt-Opt are shown in Figure 8. The results indicate that both the maxmin action selection strategy and regularization using adversarial samples improve the performance of Qt-Opt. From Figure 8, it is evident that the variant of Qt-Opt with the two proposed techniques achieved the best performance in this experiment. Additionally, the proposed techniques demonstrated their effectiveness even when they were employed separately. The difference between Qt-Opt and Qt-Opt + cat indicates the advantage of the regularization of the Q-function using adversarial samples.

We compared different implementations of CEM to determine the action that maximizes the Q-value, and the results

TABLE 1. Summary of the experimental results.

Method	Task progress [%]	Return	Training time [h]	Success rate [%]
Qt-Opt + maxmin action + cat	**94.8 ± 2.7**	**21.8 ± 3.0**	21	**72.0 ± 11.7**
Qt-Opt	85.0 ± 5.9	18.0 ± 3.8	15	62.0 ± 16.0
TD3	63.4 ± 12.9	9.0 ± 6.6	11.5	0 ± 0.0
SAC	67.8 ± 2.9	8.1 ± 1.4	12	0 ± 0.0

Success rates and returns were evaluated based on the performance after the training with 250,000 steps. The training time represents the time required for training with 250,000 steps.

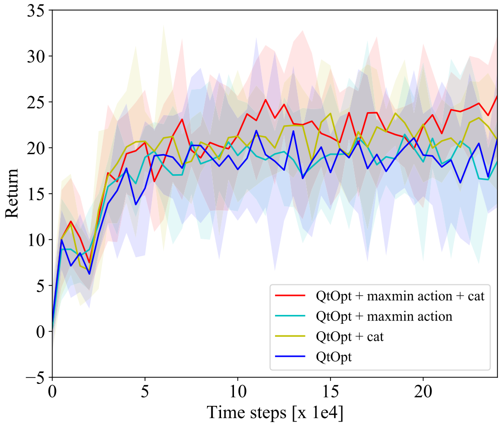


FIGURE 8. Learning curves of the variants of Qt-Opt. The proposed method, Qt-Opt + maxmin action + cat, achieved the best performance.

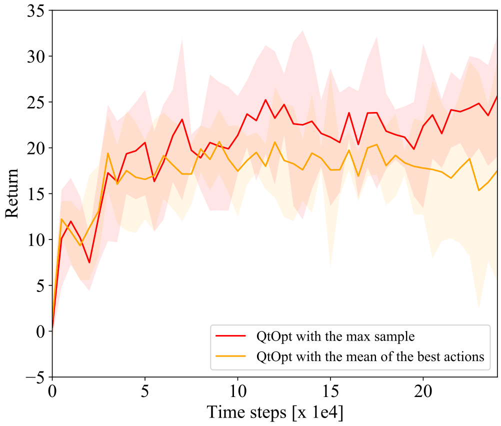


FIGURE 9. Learning curves with different implementations of CEM. The results indicate that using the sample with the highest score outperforms using the mean of the best five samples in our setting.

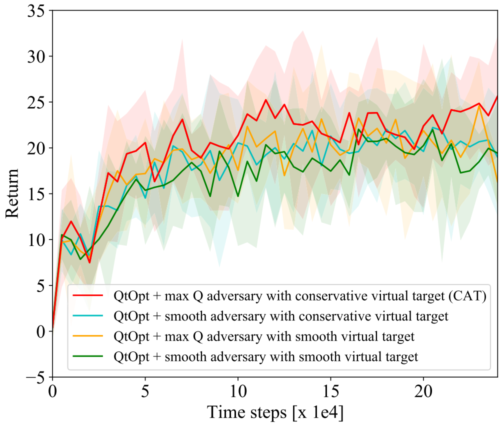


FIGURE 10. Learning curves with different strategies for generating adversarial perturbation and virtual labels. The max Q adversary with the conservative virtual target demonstrated the best performance.

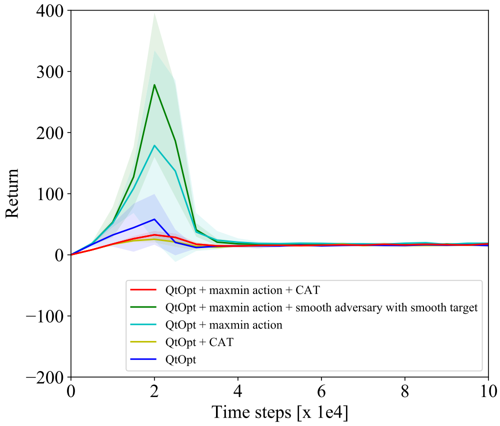


FIGURE 11. Average Q-values estimated for initial states during the training process. The standard Qt-Opt demonstrated overestimation in the beginning of the training process. In contrast, the proposed regularization technique significantly reduced overestimation of Q-values.

are shown in Figure 9. In both the variants in Figure 9, the maxmin action selection strategy and CAT were employed. As discussed in Section V-D, the Qfunction for the excavation task is highly complex and has multiple extrema. In CEM, taking the mean of the best samples is equivalent to fit ting the Gaussian distribution to the best samples. However, this may not be appropriate when the objective function has multiple extrema and the distribution of the best samples is multimodal. The results in Figure 9 imply that our excavation task is a task in which the objective function has multiple extrema.

A comparison of the different strategies for generating adversarial perturbations and virtual target values is shown in Figure 10. The maxmin action selection strategy was employed for all variants in Figure 10. Among the variants of adversarial training, the method with the max Q adversary with the conservative virtual target demonstrated the best performance. The results indicate that the regularization with adversarial samples that avoids overestimation of Q-values is more effective than the regularization, which encourages the smoothness of the approximated Q-function.

To evaluate overestimation during the training of the Q-function, we plotted the Q-values estimated for the initial states using the trained critic during the training process, as shown in Figure. 11. We report the average Q-values estimated for the initial state by randomly resetting the simulation 20 times, where the Q-value was computed using the

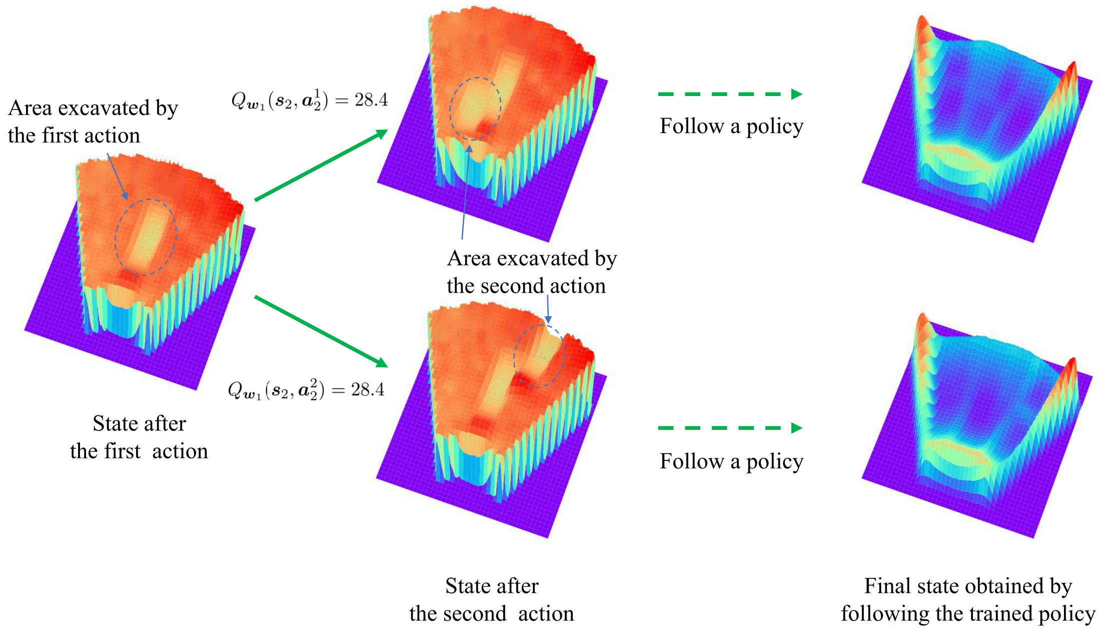

Area excavated by the first action
$Q_{w_1}(s_2, a_2^1) = 28.4$
Follow a policy
Area excavated by the second action
$Q_{w_1}(s_2, a_2^2) = 28.4$
Follow a policy
State after the first action
State after the second action
Final state obtained by following the trained policy

FIGURE 12. Multimodality of the Q-function learned for the excavation task. The leftmost figure shows the state after the first action, and the results of two candidates of the second action are visualized in the other figures. The learned Q-function indicates that both actions have comparable Q-values, and we obtained similar results after following the trained policy. This result indicates that there can be infinitely many actions that lead to optimal behaviors.

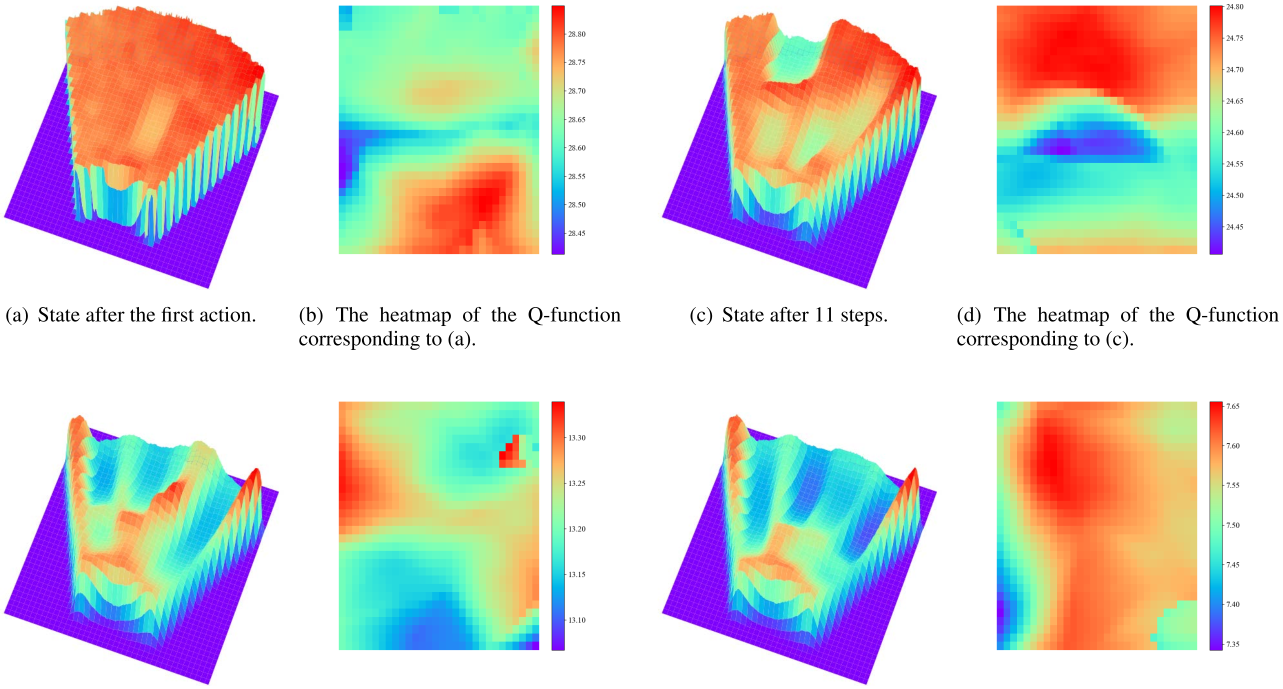
trained critic, determining the action using the action selection strategy of each method. As the maximum average return is approximately 25, the true values of the average Q-values for the initial states should be less than 25. The results in Figure. 11 indicate that overestimation of Q-values occurs in the original Qt-Opt. As expected, overestimation of the Q-values was compounded by the maxmin action selection, although the performance of the policy was improved by the maxmin action selection, as shown in Figure 6. Additionally, the results reveal that the overestimation in Qt-Opt + the smooth adversary with smooth virtual target is worse than that in the original Qt-Opt. This result indicates that the adversarial regularization that enhances the smoothness of the output of the critic works negatively for overestimation of Q-values. However, regularization with CAT significantly reduced overestimation, even when it was combined with the maxmin action selection strategy. These results demonstrate that the proposed regularization using virtual adversarial samples improves learning performance by avoiding overestimation of Q-values.

D. MULTIMODALITY OF Q-FUNCTION

In Figure 12,we demonstrate the multimodality of the Q-function for the excavation task using the policy obtained by the proposed method. The leftmost figure in Figure 12 shows the state after the first action, and the other figures show the results of taking two different actions in the second step and following the trained policy. The upper and lower figures in the middle of Figure 12 show the state after taking two different actions in the second state. The trained critic indicates that both actions have comparable Q-values, and the rightmost figures demonstrate that the soil from the target area can be successfully removed at the end of the episode, regardless of the action taken in the second step. These results demonstrate that there can be separate and multiple optimal actions in the excavation task and the optimal Q-function for the excavation task is multimodal.

Additionally, the mutimodality of the Q-function can be observed in various states. To visualize the multimodality of the Q-function, the states and heatmaps of the corresponding Q-functions are provided in Figure 13. The action is three-dimensional and given by $\boldsymbol{a} = [a_1, a_2, a_3]$; therefore,$\max_{a_3} Q_{\boldsymbol{w}}(\boldsymbol{s}, \boldsymbol{a})$ is shown in the heatmaps. From Figure 13, it is evident that there are multiple extrema of the Q-function, and the near-optimal actions are widely spread in the action space.

The Q-function for the excavation task is multimodal; therefore, a unimodal or deterministic policy is not sufficiently expressive, and the policy may not be appropriately trained. We think that this is the reason why TD3 and SAC did not learn the optimal policy in the excavation task. However, the training of critics in Q-learning algorithms does not suffer from multimodality of the Q-function as long as the model of the Q-function is sufficiently expressive. Qt-Opt determines the action that maximizes the Q-value using a black-box optimization method and the approximated Q-function; therefore, the distribution of the optimal action is not explicitly modeled in Qt-Opt. Consequently, as long as the black-box optimization method can determine one of the optimal actions, Qt-Opt does not suffer from multimodality of the Q-function.



(a) State after the first action.

(b) The heatmap of the Q-function corresponding to (a).

(c) State after 11 steps.

(d) The heatmap of the Q-function corresponding to (c).

(e) State after 31 steps.

(f) The heatmap of the Q-function corresponding to (e).

(g) State after 41 steps.

(h) The heatmap of the Q-function corresponding to (g).

FIGURE 13. States during the excavation task and the heatmap of the corresponding Q-function. To visualize the heatmap of the Q-function, we show $\max_{a_3} Q_w(s, a)$, where $a = [a_1, a_2, a_3]$. In the heatmaps, the vertical and horizontal axes represent a_1 and a_2, respectively.

VI. DISCUSSION

Our results demonstrated that the proposed variant of Qt-Opt outperformed SAC and TD3 in the excavation task. The limitation of Qt-Opt is its higher computational cost than that of SAC and TD3. This is because Qt-Opt involves hundreds of forward passes to determine the action that maximizes the Q-function using CEM. When running the simulation with a single GeForce GTX 3090 GPU and an Intel Core i9-10900K CPU at 3.7 GHz, the training period for Qt-Opt was approximately 20 h for 250,000 steps, whereas that for TD3 and SAC was approximately 12 h for 250,000 steps. However, the policy obtained by Qt-Opt clearly outperforms those obtained by SAC and TD3, and we believe that the advantages of Qt-Opt outweigh its limitations in the excavation task.

Previous studies have often discussed the properties of deep RL methods based on the results of locomotion tasks in OpenAI Gym or PyBullet. Although our findings regarding multimodality of Q-function may not be clear in such tasks, we believe that our results provide important insights for deep RL in general, and not limited to automation of excavation.

Recent studies on robotics revealed that the objective function for motion planning is often multimodal [43]–[45]. Additionally, recent studies on deep RL show that it is often beneficial to obtain multiple solutions in deep RL, which also indicates the multimodality of the Q-function [46], [47]. Although we are not aware of previous studies that directly compare Qt-Opt with TD3 and SAC, there may be other tasks where the Q-function is highly complex and Qt-Opt outperforms TD3 and SAC. Our findings demonstrate that policy representation plays an important role in deep RL. We plan to investigate flexible and explicit policy representations for actor-critic methods in future work.

Our study revealed that regularization using virtual adversarial samples can significantly improve Q-learning by avoiding overestimation of Q-values. In this study, we did not investigate regularization techniques for actor-critic methods using adversarial samples because Qt-Opt is more suitable for excavation tasks. In future work, we will investigate regularization techniques for actor-critic methods.

We demonstrated the efficacy of the proposed method using simulations. However, in reality, the soil behavior in simulation is usually different from that at the actual site. Therefore, to transfer the policy trained in simulation to a real-world system, it is necessary to train a policy that is robust against changes in soil parameters. To address this issue, recent studies employ domain-randomization techniques [48]–[50]. In future work, we will investigate the domain randomization techniques to transfer the policy trained in simulation to real-world systems.

VII. CONCLUSION

In this study, we investigated deep RL methods for learning a policy that plans the trajectory of the excavator bucket using depth images. Furthermore, we proposed novel techniques to improve the sample efficiency of Qt-Opt for excavation tasks. We proposed CAT to avoid overestimation of Q-values and verified that CAT significantly reduces overestimation and improves learning performance. Additionally, a novel strategy for selecting an action in Qt-Opt was proposed to improve sample efficiency. In our experiments, the proposed method outperformed the original Qt-Opt, TD3, and SAC, which are state-of-the-art deep RL algorithms. Moreover, our results revealed that multiple optimal actions often exist in excavation tasks and the choice of policy representation is crucial for satisfactory performance. In future work, we will investigate the domain randomization techniques to transfer the policy trained in simulation to real-world systems.

REFERENCES

[1] X. Shi, P. J. A. Lever, and F.-Y. Wang, "Experimental robotic excavation with fuzzy logic and neural networks," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 1, Apr. 1996, pp. 957–962.

[2] H. Takahashi, M. Hasegawa, and E. Nakano, "Analysis on the resistive forces acting on the bucket of a load-haul-dump machine and a wheel loader in the scooping task," *Adv. Robot.*, vol. 13, no. 2, pp. 97–114, Jan. 1998.

[3] A. Stentz, J. Bares, S. Singh, and P. Rowe, "Robotic excavator forautonomous truck loading," *Autonomous Robots*, vol. 7, no. 2, pp. 175–186, 1999.

[4] H. Shao, H. Yamamoto, Y. Sakaida, T. Yamaguchi, Y. Yanagisawa, and A. Nozue, "Automatic excavation planning of hydraulic excavator," in *Intelligent Robotics and Applications* (Lecture Notes in Computer Science), vol. 5315. Berlin, Germany: Springer, 2008, pp. 1201–1211.

[5] 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.

[6] T. Yoshida, T. Koizumi, N. Tsujiuchi, Z. Jiang, and Y. Nakamoto, "Digging trajectory optimization by soil models and dynamics models of excavator," *SAE Int. J. Commercial Veh.*, vol. 6, no. 2, pp. 429–440, Sep. 2013.

[7] R. Fukui, T. Niho, M. Nakao, and M. Uetake, "Imitation-based control of automated ore excavator: Improvement of autonomous excavation database quality using clustering and association analysis processes," *Adv. Robot.*, vol. 31, no. 11, pp. 595–606, Jun. 2017.

[8] T. Osa, N. Sugita, and M. Mitsuishi, "Online trajectory planning and force control for automation of surgical tasks," *IEEE Trans. Autom. Sci. Eng. (from July 2004)*, vol. 15, no. 2, pp. 675–691, Apr. 2018.

[9] J. Schrittwieser, I. Antonoglou, T. Hubert, K. Simonyan, L. Sifre, S. Schmitt, A. Guez, E. Lockhart, D Hassabis, T Graepel, and T. Lillicrap, "Mastering Atari, Go, chess and shogi by planning with a learned model," *Nature*, vol. 588, no. 7839, pp. 604–609, 2020.

[10] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[11] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 39, pp. 1–40, Apr. 2016. [Online]. Available: http://jmlr.org/papers/v17/15-522.html

[12] C. Bodnar, A. Li, K. Hausman, P. Pastor, and M. Kalakrishnan, "Quantile QT-opt for risk-awarevision-based robotic grasping," in *Proc. Robot. Sci. Syst.*, 2020, pp. 1–8.

[13] A. Kendall, J. Hawke, D. Janz, P. Mazur, D. Reda, J.-M. Allen, V.-D. Lam, A. Bewley, and A. Shah, "Learning to drive in a day," in *Proc. IEEE Int. Conf. Robot. Autom. (ICRA)*, May 2019, pp. 8248–8254.

[14] I. Kurinov, G. Orzechowski, P. Hämäläinen, and A. Mikkola, "Automated excavator based on reinforcement learning and multibody system dynamics," *IEEE Access*, vol. 8, pp. 213998–214006, 2020.

[15] T. Miyato, S.-I. Maeda, M. Koyama, and S. Ishii, "Virtual adversarial training: A regularization method for supervised and semi-supervised learning," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 8, pp. 1979–1993, Aug. 2019.

[16] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, 2018, pp. 1861–1870.

[17] S. Fujimoto, H. V. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *Proc. 35th Int. Conf. Mach. Learn.*, vol. 80, Jul. 2018, pp. 1587–1596.

[18] S. Schaal, A. Ijspeert, and A. Billard, "Computational approaches to motor learning by imitation," *Phil. Trans. Roy. Soc. London. Ser. B, Biol. Sci.*, vol. 358, no. 1431, pp. 537–547, Mar. 2003.

[19] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Found. Trends Robot.*, vol. 7, nos. 1–2, pp. 1–179, 2018.

[20] B. J. Hodel, "Learning to operate an excavator via policy optimization," *Proc. Comput. Sci.*, vol. 140, pp. 376–382, Jan. 2018.

[21] G. J. Maeda and D. C. Rye, "Learning disturbances in autonomous excavation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Oct. 2012, pp. 2599–2605.

[22] G. J. Maeda, D. C. Rye, and S. P. N. Singh, "Iterative autonomous excavation," in *Field and Service Robotics*. Berlin, Germany: Springer, 2014, pp. 369–382.

[23] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, and S. Levine, "Scalable deep reinforcement learning for vision-based robotic manipulation," in *Proc. Conf. Robot Learn.*, vol. 87, 2018, pp. 651–673.

[24] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus, "Intriguing properties of neural networks," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–10.

[25] I. J. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–11.

[26] A. Gleave, M. Dennis, C. Wild, N. Kant, S. Levine, and S. Russell, "Adversarial policies: Attacking deep reinforcement learning," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2020, pp. 1–16.

[27] S. Huang, N. Papernot, I. Goodfellow, Y. Duan, and P. Abbeel, "Adversarial attacks on neural network policies," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017, pp. 1–10.

[28] Y.-C. Lin, Z.-W. Hong, Y.-H. Liao, M.-L. Shih, M.-Y. Liu, and M. Sun, "Tactics of adversarial attack on deep reinforcement learning agents," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Aug. 2017, pp. 3756–3762.

[29] P. Kamalaruban, Y.-T. Huang, Y.-P. Hsieh, P. Rolland, C. Shi, and V. Cevher, "Robust reinforcement learning via adversarial training with Langevin dynamics," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 33, 2020, pp. 8127–8138.

[30] L. Pinto, J. Davidson, R. Sukthankar, and A. Gupta, "Robust adversarial reinforcement learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 70, 2017, pp. 2817–2826.

[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, and A. K. Fidjeland, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[32] P.-T. de Boer, D. P. Kroese, S. Mannor, and R. Y. Rubinstein, "A tutorial on the cross-entropy method," *Ann. Oper. Res.*, vol. 134, no. 1, pp. 19–67, 2005.

[33] D. P. Kroese, S. Porotsky, and R. Y. Rubinstein, "The cross-entropy method for continuous multi-extremal optimization," *Methodol. Comput. Appl. Probab.*, vol. 8, no. 3, pp. 383–407, Sep. 2006.

[34] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Proc. 31st Int. Conf. Mach. Learn.*, Jun. 2014, vol. 32, no. 1, pp. 387–395.

[35] 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," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016, pp. 1–14.

[36] H. van Hasselt, "Double Q-learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 2613–2621.

[37] S. Thrun and A. Schwartz, "Issues in using function approximation for reinforcement learning," in *Proc. Connectionist Models Summer School*, 1993, pp. 1–9.

[38] H. Akaike, "Information theory and an extension of the maximum likelihood principle," in *Selected Papers of Hirotugu Akaike*. New York, NY, USA: Springer, 1998.

[39] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.

[40] S. Watanabe, *Algebraic Geometry and Statistical Learning Theory*. Cambridge, U.K.: Cambridge Univ. Press, 2009.

[41] C. M. Bishop, "Training with noise is equivalent to Tikhonov regularization," *Neural Comput.*, vol. 7, no. 1, pp. 108–116, Jan. 1995.

[42] J. Achiam, "Spinning up in deep reinforcement learning," 2018. [Online]. Available: https://github.com/openai/spinningup

[43] T. Osa, "Multimodal trajectory optimization for motion planning," *Int. J. Robot. Res.*, vol. 39, no. 8, pp. 983–1001, Jul. 2020.

[44] A. Orthey, B. Frész, and M. Toussaint, "Motion planning explorer: Visualizing local minima using a local-minima tree," *IEEE Robot. Autom. Lett.*, vol. 5, no. 2, pp. 346–353, Apr. 2020.

[45] T. Osa, "Motion planning by learning the solution manifold in trajectory optimization," 2021, *arXiv:2107.05842*.

[46] S. Kumar, A. Kumar, S. Levine, and C. Finn, "One solution is not all you need: Few-shot extrapolation via structured MaxEnt RL," in *Proc. Adv. Neural Inf. Process. Syst. (NIPS)*, vol. 33, 2020, pp. 8198–8210.

[47] T. Osa, V. Tangkaratt, and M. Sugiyama, "Discovering diverse solutions in deep reinforcement learning," 2021, *arXiv:2103.07084*.

[48] J. Tobin, R. Fong, A. Ray, J. Schneider, W. Zaremba, and P. Abbeel, "Domain randomization for transferring deep neural networks from simulation to the real world," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Sep. 2017, pp. 23–30.

[49] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, and L. Zhang, "Solving Rubik's cube with a robot hand," 2019, *arXiv:1910.07113*.

[50] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, J. Schneider, S. Sidor, J. Tobin, P. Welinder, L. Weng, and W. Zaremba, "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, no. 1, pp. 3–20, 2020.

TAKAYUKI OSA received the Ph.D. degree in mechanical engineering from The University of Tokyo, Japan, in 2015. From 2015 to 2017, he was a Postdoctoral Researcher at TU Darmstadt, Germany, under the supervision of Jan Peters. From 2017 to 2019, he was a Project Assistant Professor with the Department of Mechanical Engineering, The University of Tokyo. Since 2019, he has been working as an Associate Professor with the Department of Human Intelligent Systems and the Research Center for Neuromorphic AI Hardware, Kyushu Institute of Technology, Fukuoka, Japan. He is a Visiting Researcher at the RIKEN Center for Advanced Intelligence Project and the Honda Research Institute.

MASANORI AIZAWA received the master's degree in environmental science from The University of Tokyo. He started his career at Komatsu Company Ltd., in 2015, and has been engaged in the research field of hydraulic excavator automation.