2023 IEEE International Conference on Robotics and Automation (ICRA 2023)
May 29 - June 2, 2023. London, UK

Reinforcement Learning for Laser Welding Speed Control Minimizing Bead Width Error

Toshimitsu Kaneko[1], Gaku Minamoto[1], Yusuke Hirose[2] and Tetsuo Sakai[2]

***Abstract*— In this paper, we propose a method for reinforcement learning-based laser welding control. Conventional methods apply standard reinforcement learning formulations to welding tasks, but we show that this formulation can minimize bead width or penetration depth errors only when the welding speed is constant. Therefore, conventional methods are suboptimal for training control parameters including the welding speed. The proposed method discounts future rewards with respect to the welding length instead of time steps to solve this issue. This is easily implemented by (1) modifying the discount factor used for Q-function updates in existing reinforcement learning algorithms and (2) using an appropriate reward function. Experimental results using simulators show that the proposed method achieves performance that is superior to conventional methods.**

I. INTRODUCTION

Recently, reinforcement learning (RL) has been applied to fields such as games, robotic manipulations, recommendations, and materials informatics. RL is attractive because it provides a framework for optimizing control policies without explicit formulations of complex environments.

In the field of laser welding, RL is used to develop autonomous welding systems [1]-[4]. These applications require control of welding parameters such as laser power, welding speed, and spot diameter to provide high-quality laser welding. [1] and [3] proposed learning methods for laser power control, but those methods assume a constant welding speed. Since welding quality strongly depends on both laser power and welding speed control, the constraint of a constant speed limits the ability of autonomous welding systems. On the other hand, [4] proposed an online RL architecture for learning laser welding control that includes speed control, allowing application of a standard RL formulation for learning control.

Welding quality depends on many factors, including bead width, penetration depth, pores, and spatters. However, the general quality can be judged by error from a target bead width or penetration depth. Conventional methods aim to minimize such errors by optimizing RL objectives. However, as we show below, a straightforward application of the standard RL formulation does not minimize the bead width or penetration depth error under conditions in which the welding speed varies. In this sense, the method in [4] is suboptimal.

[1]Media AI Laboratory, Corporate Research and Development Center, Toshiba Corporation, Kawasaki, Japan toshimitsu.kaneko@toshiba.co.jp
[2]Optics & Inspection Technology Research Department, Corporate Manufacturing Engineering Center, Toshiba Corporation, Yokohama, Japan

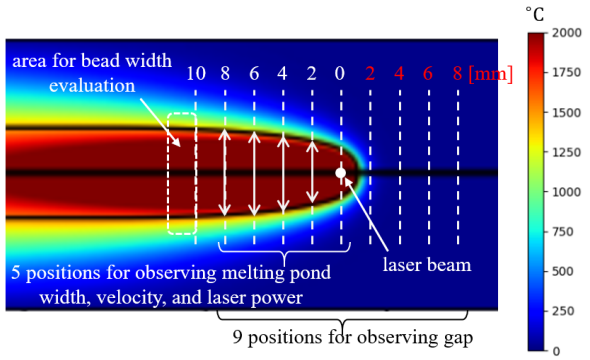

Fig. 1. Example of our laser welding simulator. Observations for welding control tasks included melting pond widths, gaps between iron plates, speeds, and laser powers. A reward is calculated from the bead width error at $d = 10$ mm behind the current laser beam position as the bead width is unstable near the laser beam.

In this paper, we show that the objective functions of conventional RL-based laser welding control methods are implicitly affected by the welding speed. Therefore, conventional methods do not minimize the root mean squared error from the target bead width. We propose a method that resolves this issue by discounting the future reward with respect to the welding length, as opposed to the conventional method of discounting with respect to time. Doing so minimizes the bead width error regardless of the welding speed. The proposed method is easily implemented by applying the proposed reward and modifying the discount factor used in the Q-function update in existing RL algorithms. We evaluate the proposed method through application to a simple toy problem and a laser welding control task using a simulator. Experimental results indicate that the proposed method is superior to conventional methods.

The main contributions are as follows: (1) we show that the objective function used in conventional RL-based welding control methods is not optimal when the welding speed is included as a control parameter, (2) we propose an RL method that minimizes the bead width error even when the welding speed varies, and (3) we provide experimental results showing that the proposed method achieves a smaller bead width error than conventional methods.

II. PROBLEM FORMULATION

A. Laser welding control

Laser welding is a process for using a laser beam as a heat source to join metal pieces. To optimize the weld quality, process parameters such as laser power, laser feed speed, and

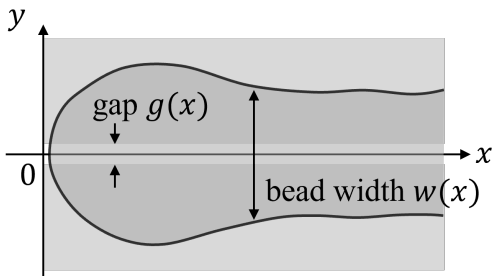

Fig. 2. An illustration of laser welding for joining two metal pieces.

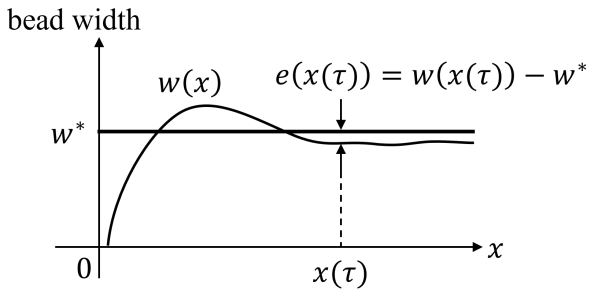

Fig. 3. Bead width and bead width error.

laser spot diameter must be controlled during the welding process. The weld quality is judged from the penetration depth, bead width, and defects such as pores and spatters. In this paper, we assume that the weld quality is measured by bead width and that the control parameters include laser feed speed.

Fig. 2 illustrates the two-dimensional geometry of laser welding. Let the x axis be the center line between two metal pieces to be joined, and assume that the laser beam is aligned with the x axis. Before welding, there is a gap between the two metal pieces, and the gap width g is a function of x. Dynamic control is thus required to optimize the quality.

After laser welding, a bead is created around the x axis. Let w^* and $w(x)$ be the target and actual bead widths at x, respectively. Then the bead width error at x is $e(x) = w(x) - w^*$ (Fig. 3). Letting $x(\tau)$ be the laser position at time τ, the bead error is also specified by the process time. The goal of the welding control problem is to obtain a control policy that minimizes the squared error

$$\int_0^L e^2(x)\,dx, \tag{1}$$

where L is the weld length.

B. Reinforcement learning

RL is an efficient learning framework for a wide range of decision-making and control tasks [5]. The RL problem is defined by a Markov decision process $\{\mathcal{S}, \mathcal{A}, p, r\}$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces. At each discrete time step t, the agent decides an action $\boldsymbol{a}_t \in \mathcal{A}$ according to the policy $\pi(\boldsymbol{a}_t|\boldsymbol{s}_t)$ given the state $\boldsymbol{s}_t \in \mathcal{S}$. The unknown transition probability density of the next state is $p(\boldsymbol{s}_{t+1}|\boldsymbol{s}_t, \boldsymbol{a}_t)$, and the agent receives a reward $r(\boldsymbol{s}_t, \boldsymbol{a}_t)$. We use ρ_π to describe the state–action marginals of the trajectory distribution induced by π and p. The RL objective is to find the optimal policy that maximizes the expected sum of rewards

$$\mathop{\mathbb{E}}_{\boldsymbol{s}_t, \boldsymbol{a}_t \sim \rho_\pi} \left[\sum_{t=0}^{T} \gamma^t r(\boldsymbol{s}_t, \boldsymbol{a}_t) \right], \tag{2}$$

where $\gamma \in (0, 1)$ is the discount factor and T is the horizon.

Soft Actor-Critic (SAC) [6][7] is an efficient off-policy RL algorithm for optimizing stochastic policies. Instead of maximizing the standard RL objective (2), SAC maximizes the entropy regularized RL objective

$$\mathop{\mathbb{E}}_{\boldsymbol{s}_t, \boldsymbol{a}_t \sim \rho_\pi} \left[\sum_{t=0}^{T} \gamma^t \left(r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \alpha H(\pi(\cdot|\boldsymbol{s}_t)) \right) \right] \tag{3}$$

to control the trade-off between the expected sum of rewards and the randomness of the policy. Here, $\alpha > 0$ is a temperature parameter for controlling the policy stochasticity, and H is entropy. We use neural networks Q_θ and π_ϕ as function approximators for the soft Q-function and the policy, respectively. Using states, actions, and rewards sampled from a replay buffer $\mathcal{D}$, the soft Q-function parameters θ can be trained by minimizing the soft Bellman residual

$$L_Q(\theta) = \mathop{\mathbb{E}}_{(\boldsymbol{s}_t, \boldsymbol{a}_t) \sim \mathcal{D}} \left[\frac{1}{2} \left(Q_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t) - \hat{Q}_{\bar{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t) \right)^2 \right] \tag{4}$$

with

$$\hat{Q}_{\bar{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma \mathop{\mathbb{E}}_{\boldsymbol{s}_{t+1} \sim p} \left[V_{\bar{\theta}}(\boldsymbol{s}_{t+1}) \right], \tag{5}$$

$$V_{\bar{\theta}}(\boldsymbol{s}_t) = \mathop{\mathbb{E}}_{\boldsymbol{a}_t \sim \pi_\phi} \left[Q_{\bar{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t) - \alpha \log \pi_\phi(\boldsymbol{a}_t|\boldsymbol{s}_t) \right], \tag{6}$$

where $\bar{\theta}$ are parameters of a target soft Q-function, obtained as the exponential moving average of θ. Policy parameters ϕ are optimized by minimizing the KL-divergence between the policy and the optimal policy corresponding to the soft Q-function, which is equivalent to minimizing the loss function

$$L_\pi(\phi) = \mathop{\mathbb{E}}_{\boldsymbol{s}_t \sim \mathcal{D}} \left[\mathop{\mathbb{E}}_{\boldsymbol{a}_t \sim \pi_\phi} \left[\alpha \log \pi_\phi(\boldsymbol{a}_t|\boldsymbol{s}_t) - Q_\theta(\boldsymbol{s}_t, \boldsymbol{a}_t) \right] \right]. \tag{7}$$

C. RL-based weld control

RL has been applied to weld control problems [1]-[4]. In these works, the reward is calculated based on the penetration depth or bead width error[1]

$$r_{\mathrm{c}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = -e^2(x(\tau)). \tag{8}$$

Here, the physical continuous time τ and the algorithmic discrete time t are related by using the control time period Δt as $\tau = t\Delta t$. Similar to the limit of a near-continuous MDP [8][9], the limit of the discounted sum of rewards becomes

$$\lim_{\Delta t \to 0} \left[\sum_{t=0}^{\lfloor T/\Delta t \rfloor} \gamma^{t\Delta t} r_{\mathrm{c}}(\boldsymbol{s}_t, \boldsymbol{a}_t) \Delta t \right] = -\int_0^T \gamma^\tau e^2(x(\tau)) d\tau. \tag{9}$$

[1]Although the error is often transformed by the Gaussian or some other function to limit the maximum and minimum values, we omit that step here for simplicity.

Therefore, the RL objective (2) with the reward $r_{\mathrm{c}}(\boldsymbol{s}_t, \boldsymbol{a}_t)$ can be considered as an approximation of the expectation of the RHS of (9) by setting $\Delta t = 1$.

III. METHOD

A. Issues in previous works

As explained by (9), previous works implicitly minimize the integral of the squared bead width error with respect to time. If the welding speed is constant, (9) is equivalent to our objective (1), except for the discount introduced to apply RL. However, when the welding speed varies, the two objectives are not the same. In the particular case of training a policy for controlling the welding speed, the two objectives often yield different policies.

For example, suppose the simple case where we want to choose a constant welding speed v under the condition that the bead width error depends only on v. In this case, denoting the bead width error by $c(v)$, the RHS of (9) becomes $c(v)^2(1-\gamma^{L/v})/\log\gamma$ by using $T \approx L/v$. This shows that the sum of discounted rewards depends on both the bead width error and the welding speed. If $c(v)$ can be zero by optimizing the speed, (9) is also minimized by the optimal speed. However, in the case of $|c(v)| > 0$ for all v, the learned policy tends to choose a higher speed than the optimal because $\gamma^{L/v}$ increases with the speed. This is confirmed by a toy problem experiment in V-A.

B. RL minimizing the squared bead width error

We propose to discount the squared bead width error along the x-axis. In this case, we can rewrite objective (1) as

$$\begin{aligned}\int_0^L \gamma^x e^2(x)dx &= \int_0^T \gamma^{x(\tau)} e^2(x(\tau))\frac{dx}{d\tau}d\tau \\ &= \sum_{t=0}^{T} \gamma^{x(t)} \left(\int_t^{t+1} \gamma^{x(\tau)-x(t)} e^2(x(\tau))v(\tau)d\tau \right).\end{aligned} \tag{10}$$

This objective can be minimized by defining the reward function as

$$r_{\mathrm{d}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = -\int_t^{t+1} \gamma^{x(\tau)-x(t)} e^2(x(\tau))v(\tau)d\tau, \tag{11}$$

and the entropy regularized RL objective as

$$\mathop{\mathbb{E}}_{\boldsymbol{s}_t, \boldsymbol{a}_t \sim \rho_\pi} \left[\sum_{t=0}^{T} \gamma^{x(t)} \left(r_{\mathrm{d}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \alpha H(\pi(\cdot|\boldsymbol{s}_t)) \right) \right]. \tag{12}$$

Since the reward is discounted with respect to x instead of t, the gradient descent update of the soft Q-function can be calculated by simply replacing (5) with

$$\hat{Q}_{\bar{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t) = r_{\mathrm{d}}(\boldsymbol{s}_t, \boldsymbol{a}_t) + \gamma^{x(t+1)-x(t)} \mathop{\mathbb{E}}_{\boldsymbol{s}_{t+1} \sim p} \left[V_{\bar{\theta}}(\boldsymbol{s}_{t+1}) \right]. \tag{13}$$

Algorithm 1 shows pseudocode for the proposed training method. λ_Q and λ_π in the pseudocode are learning rates for Q_θ and π_ϕ, respectively. μ is an interpolation factor for Polyak averaging. This algorithm requires the agent to know $x(t)$, which is easily obtained in general. We assume that $x(t+1) - x(t)$ is observed and provided in state $\boldsymbol{s}_t$.

Algorithm 1 Training method based on SAC

```
Initialize θ, φ
Set θ̄ ← θ, D ← ∅
for each iteration do
    for each environment step do
        Sample a_t ~ π_φ(a_t|s_t) and s_{t+1} ~ p(s_{t+1}|s_t, a_t)
        D ← D ∪ {(s_t, a_t, r_d(s_t, a_t), s_{t+1}, x(t+1)−x(t))}
    for each gradient step do
        Sample a batch from D
        Calculate Q̂_θ̄(s_t, a_t) using (13)
        θ ← θ − λ_Q ∇_θ L_Q(θ)
        φ ← φ − λ_π ∇_φ L_π(φ)
        θ̄ ← μθ + (1 − μ)θ̄
```

Algorithm 1 is a slight modification to SAC. However, it is straightforward to use other off-policy RL algorithms such as DQN [10] or DDPG [11] as base algorithms.

In the simple example in the previous subsection, the LHS of (10) becomes $-c(v)^2(1-\gamma^L)/\log\gamma$. Since it depends only on the bead width error $c(v)$, the policy is trained to choose the speed minimizing $c(v)$, as desired.

IV. RELATED WORK

Welding with reinforcement learning: There have been already some works applying RL for welding control [1]-[4]. [1] and [3] assume constant welding speed and propose learning methods for laser power control. [4] proposes an online RL architecture for learning laser welding control, including speed control. This architecture focuses on automatic reward calculations and assumes application of conventional RL algorithms. Therefore, none of these works can train a speed control policy that minimizes bead width error.

RL-based path following: Since a laser welding task controls the bead width to follow a given target path $w(x) = w^*$, it can be considered as a path-following problem. RL-based path-following methods have been proposed for ground vehicles [12][13], marine vehicles [14][15] and aerial vehicles [16][17]. However, these methods minimize the objective function (9) and cannot solve the issue described in Section III-A.

State-dependent discount factor: The proposed method requires the laser beam position $x(t)$ to update the soft Q-function, which is assumed to be given in state $\boldsymbol{s}_t$. Therefore, our method is one of the RL methods with a state-dependent discount factor [18]-[20]. This paper aims to minimize (1), and clarifies that both the state-dependent discount factor $\gamma^{x(t+1)-x(t)}$ and the specific reward function (11) are required to do so.

V. EXPERIMENTS

We conduct experiments to compare the proposed method (mSAC(r_{d})) with two ablation baselines: the original SAC algorithm with the conventional reward function (8) and the original SAC algorithm with the new reward function (11). These two baselines are denoted as SAC(r_{c}) and SAC(r_{d}),

TABLE I

DIFFERENCE AMONG THE EVALUATED ALGORITHMS

	mSAC(r_d) (ours)	SAC(r_c)	SAC(r_d)
reward function	equation (11)	(8)	(11)
Q-function update	equation (13)	(5)	(5)

respectively (see Table I). In these methods, two soft Q-functions are trained independently and the lesser of the two is used to update parameters. As reported in [7], this approach stabilizes and speeds up training.

A. Toy problem

To show the difference between minimizing (9) and (10), we evaluate performance on a simple toy problem. In this task, the agent is required to adjust the welding speed to an optimal value provided in the states. The bead width error follows a normal distribution $N(|v_t - v_\mathrm{opt}|, \sigma^2)$, and the optimal speed v_opt is randomly selected in the range $[0.1, 1.0]$ at the beginning of each episode.

Specifically, the state is a vector of the optimal speed v_opt, the current welding speed v_t, and the current acceleration a_t. The action is the acceleration of the next time step a_{t+1} ($-0.3 \leq a_{t+1} \leq 0.3$). The speed is updated as $v_{t+1} = v_t + a_{t+1}$, then clipped by the minimum speed 0.05 and the maximum speed 1.5. We apply a hyperbolic tangent function before calculating the reward from the squared bead width error to limit the reward range. The standard deviation is $\sigma \in \{0.0, 0.1, 0.2\}$. The discount factor is set to $\gamma = 0.99$ for SAC(r_c). The discount factor for the proposed algorithm is $0.99^{2/(0.05+1.5)}$. This value is selected so that both discount factors are equivalent when the welding speed is in the middle of the speed range. All experiments are repeated ten times with different random seeds. In each trial, policies are trained over 150,000 time steps.

Fig. 4 shows the results of this experiment. The upper row in the figure shows the root mean squared error (RMSE) for each optimal speed. The left, middle, and right figures are the results where $\sigma = 0.0$, 0.1, and 0.2, respectively. When $\sigma = 0$, SAC(r_c) performance is the same as that of the proposed algorithm. In Section III-A, it is discussed that the conventional method can minimize (9) if the error can be zero by adjusting the speed. $\sigma = 0$ corresponds to this case and the experimental result is consistent to the discussion in Section III-A. On the other hand, in the case of $\sigma > 0$, the performance of SAC(r_c) becomes worse than mSAC(r_d). The lower row in the figure shows the average speeds determined by the learned policy for each optimal speed. Dashed lines indicate the optimal policy $\bar{v} = v_\mathrm{opt}$. mSAC(r_d) can select the optimal speed for all σ, but SAC(r_dc) tends to select a higher speed than the optimal when $\sigma > 0$. This is also predicted in Section III-A as the case $|c(v)| > 0$.

SAC(r_d) performance is not good when $\sigma > 0$. In contrast to SAC(r_c), SAC(r_d) chooses lower speed than the optimal. The new reward definition alone does not improve the performance of conventional methods, and it is important to use (13) instead of (5) for the soft Q-function estimation.

TABLE II

STATE DEFINITION USED IN THE LASER WELDING CONROL TASK

element	dimension
target bead width	1
melting pond widths at the observation points	5
gap widths at the observation points	9
speeds at the observation points	5
laser powers at the observation points	5

B. Laser welding control task

We developed a laser welding simulator that calculates temperature distributions in iron by solving a discretized version of the heat equation every 20 milliseconds. In this simulation, locations where the temperature exceeds the melting point of iron are regarded as part of the melting pond, and locations where melting pond temperatures fall below the melting point are considered part of the bead. As Fig. 2 shows, we assume two iron plates on either side of the x-axis, with the laser beam path fixed on the x-axis.

In the simulator-based laser welding task, it is required to control welding parameters dynamically depending on changes of welding conditions such as gap width and unknown temperature distribution. The state, action, and reward in this task are defined as follows. The state is a vector consisting of a target bead width w^* and features at the observation positions. The state definition is summarized in Table II and the observation positions are depicted in Fig. 1. The target bead width w^* is randomly chosen from 0.1 to 0.6 mm at the beginning of each episode. We evaluated two action definitions, one where the action is an acceleration along the x-axis, and one where the action is a three-dimensional vector consisting of acceleration along the x-axis, the laser power, and the spot-size (diameter). The laser power was limited to 2.0 (high max power) or 0.5 (low max power). Lower power limit forces more dynamic speed control. In all cases, the control time period is 100 ms and welding speeds were clipped in a range from 10 to 50 mm/s.

Since the bead width is unstable near the laser beam, the reward is calculated from the bead width error at $d = 10$ mm behind the current laser beam position. The bead width error is evaluated by a hyperbolic tangent of the relative bead width error $e(x(\tau)) = \tanh(|w(x(\tau) - d) - w^*|/w^*)$, where $x(\tau)$ and $w(x)$ are the laser beam position and the bead width, respectively. Seven positions for the gap width change and the gap width were also randomly selected in each episode. We repeated experiments five times with different random seeds. We trained policies over 200,000 time steps for training speed control and 300,000 time steps for training speed, power, and spot-size control. We used $\gamma = 0.99$ for SAC(r_c), which was selected from the results of preliminary experiments with $\gamma \in \{0.8, 0.9, 0.99\}$. The discount factor for the proposed algorithm is $0.99^{2/(0.01+0.05)}$. The training curves are presented in Fig. 5.

Fig. 6 shows RMSEs of the learned policies. The left figures are the results of speed only control. The performance of the proposed method mSAC(r_d) is superior to the conven-

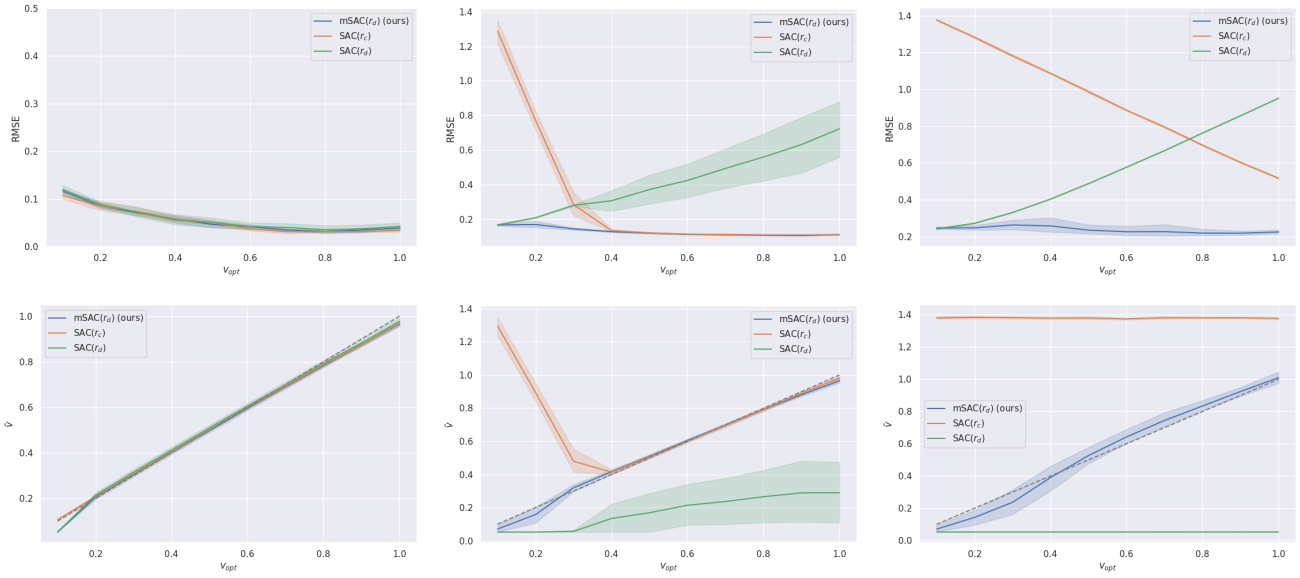

Fig. 4. Results for the toy problem. The upper and lower row are the RMSE and the mean speed, respectively. The left, middle, and right columns are the results for $\sigma = 0.0, 0.1$, and 0.2, respectively.

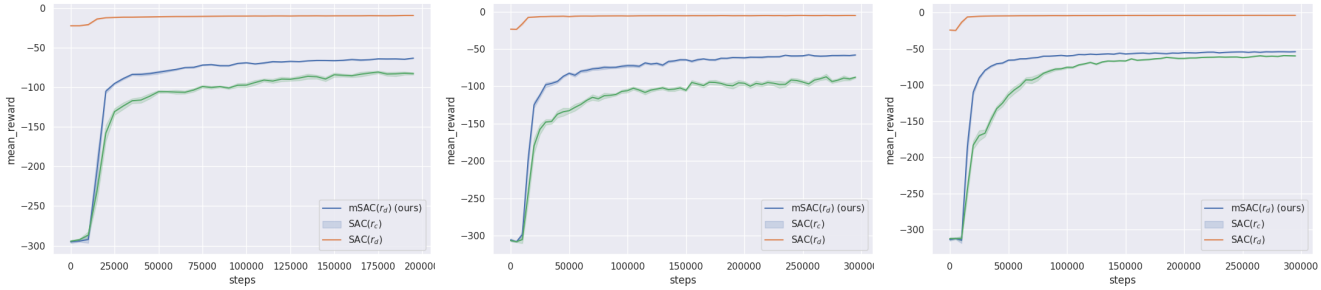

Fig. 5. Training curves for the laser welding control task. Left: speed control only. Middle: speed, power, and spot-size control (low max power), Right: speed, power, and spot-size control (high max power). Note that two reward definitions (r_d and r_c) are used, and that they cannot be directly compared to each other. The above figures show convergence speeds for each method.

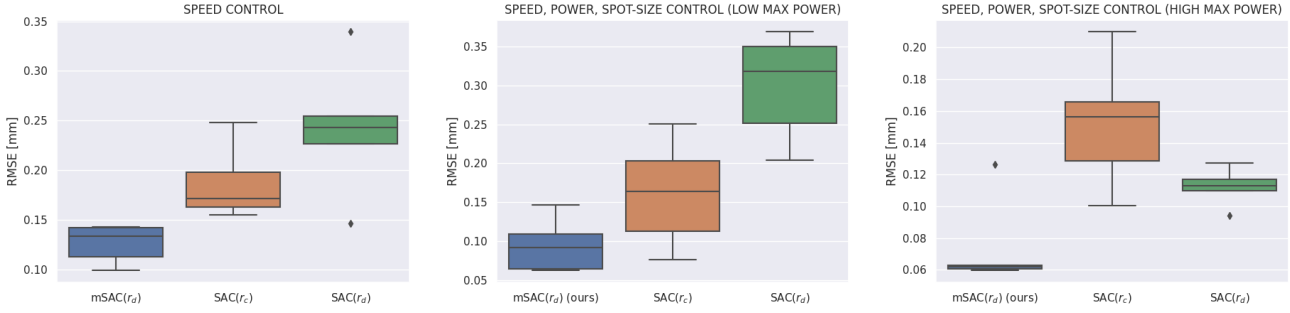

Fig. 6. RMSE comparison for the laser welding control task. Left: speed control. Middle: speed, power, and spot-size control (low max power). Right: speed, power, and spot-size control (high max power).

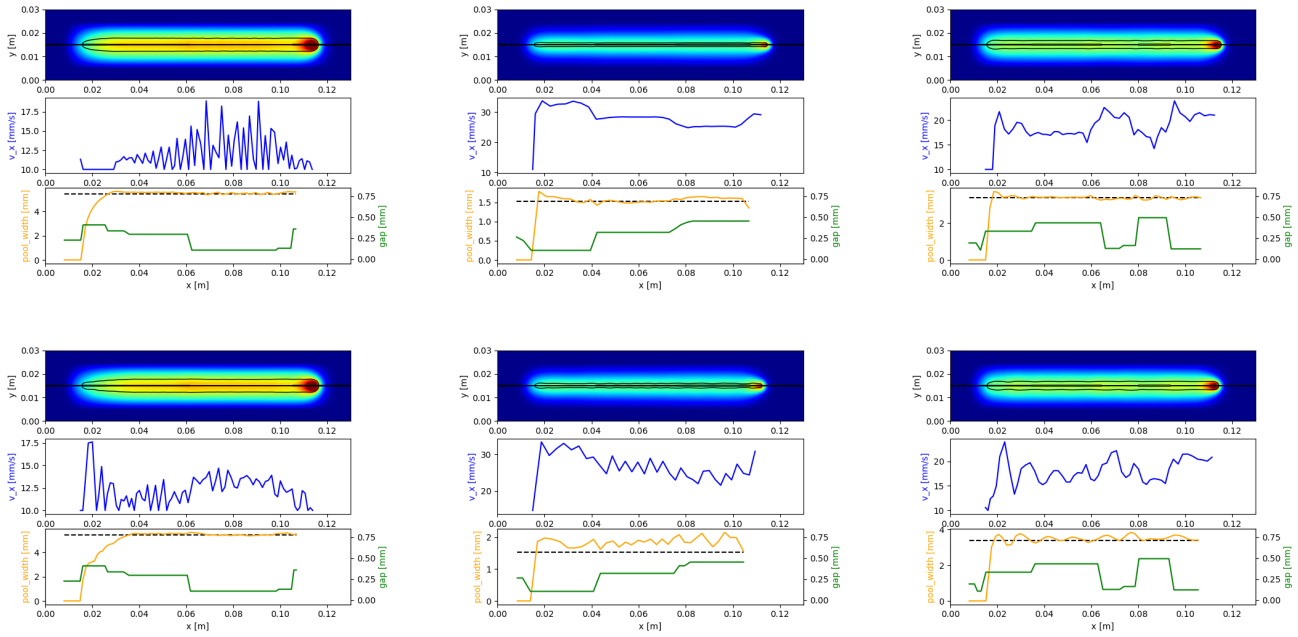

Fig. 7. Examples of welding speed control using the learned policies (upper: proposed method mSAC(r_d), lower: conventional method SAC(r_c)). Blue, orange, and green lines are the welding speeds, bead widths, and gap widths, respectively. The dashed lines are target bead widths.

TABLE III
AVERAGE SPEED OF THE LASER WELDING TASK

task	mSAC(r_d)	SAC(r_c)
Speed control	17.02	17.57
Speed, power and spot-size control		
(Low max power)	30.73	32.97
(High max power)	43.56	47.00

tional method SAC(r_c). As presented in Table III, the average speed of SAC(r_c) is faster than that of mSAC(r_d). As in the case of the toy problem, it can be considered that the policy learned by the conventional method tend to choose higher speed than the optimal and increases RMSE.

The middle and right figures in Fig. 6 are the results of speed, power and spot-size control. In both cases, RMSEs of mSAC(r_d) are smaller than those of SAC(r_c). These results show that mSAC(r_d) is still effective in the case of training some parameters in addition to speed. SAC(r_d) is always inferior to mSAC(r_d) and often worse than SAC(r_c).

Fig. 7 shows examples of speed control using the trained policies. The upper figures show the results under the proposed method mSAC(r_d) and the lower figures show those under the conventional method SAC(r_c). Dashed black lines indicate the target bead width w^*, and the orange lines are the simulated bead widths. In the left column, bead width errors under the two methods are nearly the same when $x > 0.04$. However, the welding speed under the conventional method (blue line) at around $x = 0.02$ is faster than that under the proposed method, which increases the bead width error.

Narrow bead widths are more difficult to accurately generate than wide bead widths, because small speed changes cause relatively large bead-width errors. As shown in the middle column of Fig. 7, bead widths under the proposed method are stabler and closer to the narrow target width than those under the conventional method.

Another example is shown in the right column. The proposed method can retain small bead width error under the abrupt gap width changes. If the welding condition including the gap width is constant, the bead width error can be zero by controlling the speed. As explained, the conventional method performs well in this case. However, at the beginning of the welding or gap width changes, it is impossible to remove bead width error completely. The proposed method is effective for such cases. The thickness of the base metal is one of the important parameters affecting the welding quality. It was assumed to be constant in this task, but it can be a dynamic parameter like the gap width. The proposed method is expected to be effective in such cases, too.

VI. CONCLUSIONS

We proposed a reinforcement learning-based laser welding control method for minimizing the bead width error. The proposed method is simple but efficient for training policies for controlling welding parameters including speed. Our method can be easily implemented by slightly modifying the Q-function update in off-policy RL algorithms.

Our motivation is the fact that the standard RL formulation does not minimize the bead width or penetration depth error when the welding speed varies. Such issues are common in industrial applications where autonomous velocity control is desired to minimize positional error from a target. Therefore, we believe the same approach can be applied to a broad range of tasks.

REFERENCES

[1] J. Günther, P. M. Pilarski, G. Helfrich, H. Shen and K. Diepold, Intelligent laser welding through representation, prediction, and control learning: An architecture with deep neural networks and reinforcement learning, Mechatronics, vol. 34, pp. 1-11, Mar. 2016.

[2] B. Wang, S. J. Hu, L. Sun, T. Freiheit, Intelligent welding system technologies: State-of-the-art review and perspectives, Journal of Manufacturing Systems 56, pp.373-391, 2020.

[3] G. Masinelli, T. Le-Quang, S. Zanoli, K Wasmer and S. A. Shevchik, Adaptive Laser Welding Control: A Reinforcement Learning Approach, ," IEEE Access, vol. 8, pp. 803–814, 2020.

[4] M. Schmitz, F. Pinsker, A. Ruhri, B. Jiang and G. Safronov, Enabling Rewards for Reinforcement Learning in Laser Beam Welding processes through Deep Learning, In 2020 19th IEEE Int. Conf. on Machine Learning and Applications (ICMLA), 2020.

[5] Richard S. Sutton and Andrew G. Barto, Reinforcement Learning: An Introduction, Second Edition, MIT Press, Cambridge, MA, 2018.

[6] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, Sergey Levine, Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, In 35th Int. Conf. on Machine Learning (ICML), Stockholm, Sweden, 2018.

[7] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, Sergey Levine, Soft Actor-Critic Algorithms and Applications, arXiv:1812.05905, 2018.

[8] Kenji Doya, Reinforcement Learning in Continuous Time and Space, Neural Computation, 12(1):219-245, 2000.

[9] Corentin Tallec, Léonard Blier, Yann Ollivier, Making Deep Q-learning Methods Robust to Time Discretization, In 36th Int. Conf. on Machine Learning (ICML), Long Beach, USA, 2019.

[10] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with Deep Reinforcement Learning, NIPS Deep Learning Workshop, 2013.

[11] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, D. Wierstra, Continuous Control With Deep Reinforcement Learning, In 2016 Int. Conf. on Learning Representations (ICLR), 2016.

[12] Danial Kamran, Junyi Zhu, Martin Lauer, Learning Path Tracking for Real Car-like Mobile Robots From Simulation, In 2019 European Conf. on Mobile Robots (ECMR), 2019.

[13] Johannes Ultsch, Jonas Mirwald, Jonathan Brembeck, Ricardo de Castro, Reinforcement Learning-based Path Following Control for a Vehicle with Variable Delay in the Drivetrain, In 2020 IEEE Intelligent Vehicles Systems (IV), Las Vegas, USA, 2020.

[14] Haiqing Shen and Chen Guo,Path-Following Control of Underactuated Ships using Actor-Critic Reinforcement Learning with MLP Neural Networks, In Sixth Int. Conf. on Information Science and Technology, Dalian, China, 2016.

[15] Andreas B. Martinsen and Anastasios M. Lekkas, Curved Path Following with Deep Reinforcement Learning: Results from Three Vessel Models, In OCEANS 2018 MTS/IEEE Charleston, 2018.

[16] Chunyu Nie, Zewei Zheng and Ming Zhu, Three-Dimensional Path-Following Control of a Robotic Airship with Reinforcement Learning, Int. Journal of Aerospace Engineering, vol. 2019, pp.1-12, 2019.

[17] Bartomeu Rubí, Bernardo Morcego and Ramon Pérez, A Deep Reinforcement Learning Approach for Path Following on a Quadrotor, In 2020 European Control Conf. (ECC), Saint Petersburg, Russia, 2020.

[18] Naoto Yoshida, Eiji Uchibe and Kenji Doya, Reinforcement Learning with State-Dependent Discount Factor, The third IEEE Int. Conf. on Development and Learning and Epigenetic Robotics (ICDL), Osaka, Japan, 2013.

[19] Martha White, Adaptive Laser Welding Control: A Reinforcement Learning Approach, In 34th Int. Conf. on Machine Learning (ICML), Sydney, Australia, 2017.

[20] Abhinav Sharma, Ruchir Gupta, K. Lakshmanan and Atul Gupta, Transition Based Discount Factor for Model Free Algorithms in Reinforcement Learning, Symmetry 2021, 13(7), 1197, 2021.