An Optical Controlling Environment and Reinforcement Learning Benchmarks

Abulikemu Abuduweili [1] **Changliu Liu** [1]

Abstract

Deep reinforcement learning has the potential to address various scientific problems. In this paper, we implement an optics simulation environment for reinforcement learning based controllers. The environment incorporates nonconvex and nonlinear optical phenomena as well as more realistic time-dependent noise. Then we provide the benchmark results of several state-of-the-art reinforcement learning algorithms on the proposed simulation environment. In the end, we discuss the difficulty of controlling the real-world optical environment with reinforcement learning algorithms.

1. Introduction

In recent years, deep reinforcement learning (RL) has been used to solve challenging problems in various fields (Sutton & Barto, 2018), including self-driving car (Bansal et al., 2018) and robot control (Zhang et al., 2015). Among all of the applications, deep RL made significant progress in play games on a superhuman level (Mnih et al., 2013; Silver et al., 2014; 2016; Vinyals et al., 2017). Beyond playing games, deep RL has the potential to strongly impact the traditional control and automation tasks in the natural science, such as control problems in chemistry (Dressler et al., 2018), biology (Izawa et al., 2004), quantum physics (Bukov et al., 2018), optics and photonics (Genty et al., 2020).

In optics and photonics, there is particular potential for RL methods to drive the next generation of optical laser technologies (Genty et al., 2020). This is not only because there is increasing demand for adaptive control and automation (of tuning and control) for optical systems (Baumeister et al., 2018), but also because many phenomena in optics are nonlinear and multidimensional (Shen, 1984), with noise-sensitive dynamics that are extremely challenging to model using conventional methods. RL methods are able to control multidimensional environment with nonlinear function approximation (Dai et al., 2018). Thus, study the RL controller in optics becomes increasingly promising in optics and photonics as well as its applications in scientific research, medicine, and other industries (Genty et al., 2020; Fermann & Hartl, 2013).

Traditionally, many of the control problems in optics and photonics were implemented by stochastic parallel gradient descent (SPGD) algorithm with PID controller (Cauwenberghs, 1993; Zhou et al., 2009; Abuduweili et al., 2020a). The target is to maximize the reward (e.g. optical pulse energy) by adjusting and controlling the system parameters. The SPGD algorithm is one of the special cases of stochastic error descent method (Cauwenberghs, 1993; Dembo & Kailath, 1990). Stochastic error descent is based on the model-free distributed learning mechanism. A parameter update rule is proposed by which each individual parameter vector perturbation contributes to a decrease in error (or increase in reward). However, SPGD is typically a convex optimization solver, and many control problems in optics are non-convex. SPGD may be failed to search the global optimum of the optics control system unless the initial state of the system is near a global optimum. Conventionally, the initial state of the optical system was adjusted by experienced experts, then utilizing SPGD to control the manually adjusted system, which becomes extremely hard with the increasing system complexity. In order to achieve efficient control and automation, deep RL was introduced to control optical systems (Tünnermann & Shirakawa, 2019; Sun et al., 2020; Abuduweili et al., 2020b; 2021). Most of the previous works implemented Deep-Q Network (Mnih et al., 2013) and Deep Deterministic Policy Gradient (Lillicrap et al., 2015), in optical control systems to achieve the comparable performance with traditional SPGD-PID control (Tünnermann & Shirakawa, 2019; Valensise et al., 2021). But there is a lack of works on the evaluation of more RL algorithms in the more complex optical control environment.

Studying and validating RL algorithms in the real-world optical system is a challenging process because its cost is expensive and requires experienced experts to implement the optical system. Instrumenting and operating RL algo-

[1]Robotics Institute, Carnegie Mellon University. Correspondence to: Abulikemu Abuduweili <abulikea@andrew.cmu.edu>.

2nd AI4Science Workshop at the 39^{th} International Conference on Machine Learning (ICML), 2022.

rithms in a simple optical system require significant funds and manpower. An effective alternative to validate RL algorithms in optics is simulation. Simulation has been used for robotics and autonomous driving since the early days of research (Pomerleau, 1998; Bellemare et al., 2013). As learning-based robotics expands in both interest and application, the role of simulation may become ever more critical in driving research progress. But there is not any open sourced RL environment for optics control simulation by now.

In this paper, we introduce OPS (**O**ptical **P**ulse **S**tacking environment) - a scalable open simulator for controlling a typical optical system. The physics behind our OPS system is the same as many other optical problems, including coherent optical inference (Wetzstein et al., 2020) and linear optical sampling (Dorrer et al., 2003), which can be used for precise measurement, industrial manufacturing, and scientific research. A typical optical pulse stacking system directly and symmetrically stacks up the input pulses to multiply the pulse energy for output stacked pulses (Tünnermann & Shirakawa, 2017; Stark et al., 2017; Astrauskas et al., 2017; Yang et al., 2020). By providing an optical control simulation environment, we aim to encourage exploration of the application of RL on optical control tasks and furtherly explore the RL controllers in natural science. We use OPS to evaluate some important RL algorithms including twin delayed deep deterministic policy gradient (TD3, (Fujimoto et al., 2018)), soft actor-critic (SAC, (Haarnoja et al., 2018a)), and proximal policy optimization (PPO, (Schulman et al., 2017)). After reporting the results of these RL algorithms, we discuss the difficulty of RL algorithms in the real-world optical system. With the provided simulating environment OPS and the experiments of RL algorithms, we believe that this work can promote the research on RL applications in optics as well as benefit both the machine learning and the optics community. The code of the paper is available at `https://github.com/Walleclipse/Reinforcement-Learning-Pulse-Stacking`.

2. Simulation environment

2.1. Physics of the simulation

The optical pulse stacking (OPS, or also called pulse combination) system recursively stacks up the optical pulses in the time domain. The dynamics of the OPS are similar to the recurrent neural networks (RNN) or Wavenet architecture (Oord et al., 2016). We illustrate the dynamics of the OPS in RNN style as shown in Figure 1. The input of the OPS is a periodic pulse train[1] with a repetition period of T. Assume the basic function of the first pulse at time step t is $E_1 = E(t)$, then the consecutive pulses can be described as $E_2 = E(t+T), E_3 = E(t+2T)$... The OPS system recursively imposes the time delay on earlier pulses for every two consecutive pulse pairs. As an example, the 1st stage time-delay controller imposes the time delay τ_1 on pulse 1 to shift the pulse 1. With the proper time delay, pulse 1 could be stacked with the next pulse E_2 to create the stacked pulses $E_{1,2} = E(t+\tau_1) + E(t+T)$. Similarly, pulse 3 could be stacked with pulse 4 to create $E_{3,4} = E(t+2T+\tau_1) + E(t+3T)$, and so on. In 2nd stage OPS, the time delay τ_2 was further imposed to $E_{1,2}$ to make it to stack up with $E_{3,4}$ to create $E_{1,2,3,4}$. This kind of stacking is repeated in each stage OPS controller, which stacks up the pulses in geometrical progression (recursion). An N-stage OPS system simply multiplies pulse energy by 2^N times by stacking up 2^N pulses, in which N time delays $(\tau_1, \tau_2, ..., \tau_N)$ are needed to control and stabilize. Please check the more detailed illustration and configuration of the real-world OPS experiment in Appendix A.

2.2. Control objective and noise

The objective of the controlling OPS system is to maximize the final stacked (output) pulse energy by adjusting the time delays. For N-stage OPS system, let P_N denotes the final energy of N times stacked pulse, and $\tau = [\tau_1, \tau_2, \cdots, \tau_N]$ denote the time delays. Then the objective function for controlling N-stage OPS system is:

$$\arg\max_{\tau} P_N(\tau) = \arg\max_{\tau_1,\tau_2,...,\tau_N} P_N(\tau_1, \tau_2, ..., \tau_N) \tag{1}$$

If any noise were ignored, we would analyze the exact function of the final pulse energy P_N w.r.t. the time delays τ. Figure 2(a) shows the function of the pulse energy $P_1(\tau_1)$ w.r.t. the first time delay τ_1 in 1-stage OPS system. And Figure 2(b) shows the function surface of $P_2(\tau_1, \tau_2)$ w.r.t. the first and second stages time delay (τ_1, τ_2) in 2-stage OPS system. As can be seen, the control function of the OPS system is non-linear and non-convex even ignoring any noises[2]. This is a challenging problem for any controlling algorithms to achieve the global optimum (or better local optimum) from a random initial state.

In general, noise can not be ignored and the system is quite noise-sensitive. That is because the wavelength of the pulse is in μm level ($1\mu\text{m} = 10^{-6}\text{m}$). The noise in the environment, including vibration of optical devices and temperature drift of the atmosphere, could easily bring the shift of the time delay then change the output pulses. So the objective function in real-world practice is more complex than Figure 2, especially for higher stage (high-dimensional) OPS. Therefore, under (unpredictable) noise and the noise-

[1] The periodic pulse train generally emitted by lasers. The wave function of each laser pulse is almost the same except for the time delay of a period.

[2] Part of the reasons are the optical periodicity and nonlinearity of the coherent interference.

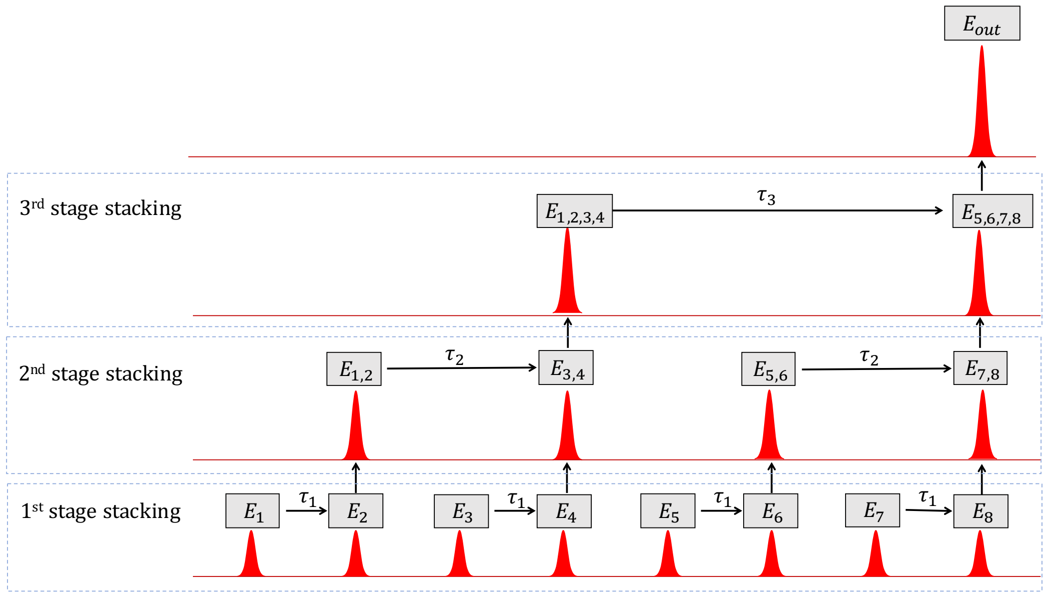
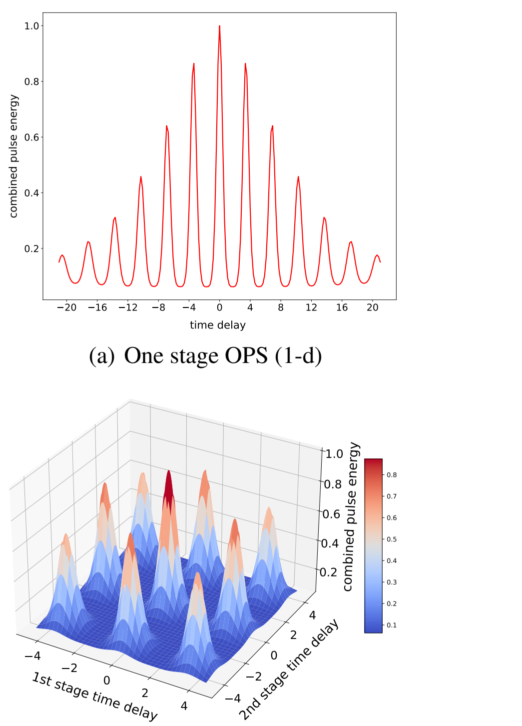


Figure 1. Illustration of the principle of optical pulse stacking. Only 3-stage pulse stacking was plotted for simplicity.



(a) One stage OPS (1-d)

(b) Two stage OPS (2-d)

Figure 2. Function plot of the (a) 1-stage OPS: pulse energy $P_1(\tau_1)$ w.r.t. delay line τ_1. (b) 2-stage OPS: pulse energy $P_2(\tau_1, \tau_2)$ w.r.t. delay lines (τ_1, τ_2).

sensitive complex system, the model of the system is hard to be achieved (Genty et al., 2020). So it is hard to implement model-based controllers. In this paper, we mainly consider model-free reinforcement learning approaches.

In this simulation, we mainly consider two kinds of noise. The first one is fast noise which comes from the vibration of devices. The noise could be formulated as a zero-mean Gaussian random noise $\mathcal{N}(0, \sigma^2)$ by following the simulation noise of (Tünnermann & Shirakawa, 2019). The second is slow noise μ_t, which comes from slow temperature drift. The influence of the temperature drift can be formulated as a piecewise linear function (Ivanova et al., 2021). By incorporating these two kinds of noise, overall noise e_t can be formulated as a random process, where:

$$\mathbb{E}\left[e_t\right] = \mu_t, \ \ \mathbb{VAR}\left[e_t\right] = \sigma^2 \tag{2}$$

2.3. Reinforcement learning environment

Interactions with RL agent. An RL agent interacts with the OPS environment in discrete time steps, as shown in Figure 3. At each time step t, the RL agent receives the current state of the OPS environment s_t. Then the RL agent chooses an action a_t to send the OPS environment. The environment conduct the action and moves to new state s_{t+1}. Then the reward r_t, which measured by the state s_{t+1}, feedback to the RL agent. The RL agent trained with the experience (s_t, a_t, s_{t+1}, r_t) to learn a policy $\pi(a, s)$ which maximizes the expected cumulative reward.

State space. State space of OPS is a continual and multidimensional vector space. The state value s_t could be described as the pulse amplitude measurement of the final

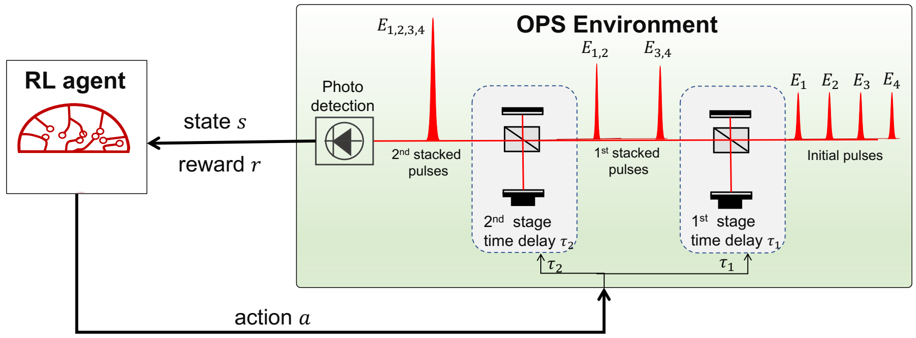


Figure 3. Illustration of the interaction between RL agent and OPS environment. Only 2-stage pulse stacking was plotted in OPS for simplicity.

stacked pulse $s_t = \text{Amplitude}(E_{\text{out}}(t))$. So s_t is the time-domain "picture" of the final stacked pulse, which directly reflects the performance of the control. In a real-world system, pulse amplitude was detected by a photo-detector then converted to digital time-series signals. In our simulation, we also implement real-time rendering of the pulse amplitude to monitor the controlling process.

Action space. Action space of N-stage OPS environment is a continual and N-dimensional vector space. At time step t, the action a_t is an additive time delay value $\Delta\tau(t)$ for N-stage OPS environment: $a_t = \Delta\tau(t) = (\tau_1(t+1) - \tau_1(t), \tau_2(t+1) - \tau_2(t), ..., \tau_N(t+1) - \tau_N(t))$ or $\tau(t+1) = a_t + \tau(t)$. The additive time delay value $a(t)$ was conducted by OPS environment to lead the next state.

Reward. As mentioned in Section 2.2, the objective of the OPS controller is maximizing the final stacked pulse energy $P_N(\tau)$. We used the reward value as normalized final pulse energy:

$$r = -\frac{(P_N(\tau) - P_{max})^2}{(P_{min} - P_{max})^2}, \tag{3}$$

where P_{max} is the maximum pulse energy achieved at the global optimum, and P_{min} is the minimum pulse energy. The maximum reward 0 achieved when $P(\tau) = P_{max}$ (peak position of Figure 2(b)) .

State transition function. The environmental noise poses direct impacts to the delay lines (including the vibration and temperature shift noise of the delay line devices). So in the state transition, real conducted delay line $\tau_{\text{real}}(t+1)$ is a combination of the action a_t and noise e_t:

$$\tau_{\text{real}}(t+1) = \tau(t+1) + e_t = \tau(t) + a_t + e_t. \tag{4}$$

Then the real time delay $\tau_{\text{real}}(t)$ imposed to some selected pulses by delay line devices (the device impose additional time delay for pulses) as conduct the action. The state transition is governed by state, action and noise. The exact form of the state transition follow the principle of the coherent pulse interference. Let f is a observation function (which observe final output pulse by time delay value). Then the state transition can be written as:

$$s_{t+1} = f(\tau_{\text{real}}(t+1)) = f(\tau_{\text{real}}(t)+a_t+e_t) = f(f^{-1}(s_t)+a_t+e_t) \tag{5}$$

Please note that $\mathbb{E}[e_t] = \mu_t$ is a slow-changed piecewise linear function, which changes very slowly with time. For episodic training for RL agents, the μ_t could be considered as a constant value for iterations within an episode. But the value of μ_t might differ from one episode to the next episode. In this case, we can assume μ_t changes very slowly, then approximate the OPS control process as a Markov decision process (MDP). If the higher accuracy is needed, one can include the noise in the state definition, $s_t = [\text{Amplitude}(E_{\text{out}}(t)); e_t]$, then the control process can be formulated as a partially observable Markov decision process (POMDP).

Different control difficulty of the environment. We implemented the OPS environment for arbitrary ($N \in \{1, 2, 3, ...\}$) stage of pulse stacking. With the increase of the number of stages, the control would become more and more difficult. In addition to the customized number of stages, we also provided three modes (easy, medium, and hard) for each stage OPS, as shown in Figure 4. The mode was determined by the initial state of the system and noise distribution:

- Easy mode. The initial state of the OPS system is near the global optimum for easy mode. Figure 6(a) shows the example initial state of the easy mode of the 3-stage OPS environment. This is a case for many traditional optics control problems: the initial state of the system is tuned by "experts" to make it easy to control for convex controllers.
- Medium mode. The initial state of the system is random, as shown in Figure 6(b), which makes the control problem becomes nonconvex. But in medium mode, the noise is time-independent and we simply set the noise distribution as $e_t \sim \mathcal{N}(0, \sigma)$. This is the case for many classical reinforcement learning and typical

MDP settings. The noise distribution of each episode is the same.

- Hard mode. Similar to medium mode and Figure 6(b), the initial state of the system is random. Different from the medium mode, the noise behavior is more complicated. The mean value of the noise distribution μ_t is a time-dependent variable, which slowly changes during time. This case is not a typical MDP. The hard mode is closer to real-world settings. Because in real-world applications, we always deploy the testing environment and algorithms after training, so the noise distribution of the testing environment is different from the training environment.

Mode	Initial state	Noise
easy	near the optimum	time independent; $\mu_t \equiv 0$
medium	random	time independent; $\mu_t \equiv 0$
hard	random	time dependent; $\mathrm{d}\mu_t/\mathrm{d}t \not\equiv 0$

Figure 4. Comparison of the different game modes.

```
from optics_env import OPS_env
env = OPS_env(stage=5, mode="medium")
env.reset()
done = False
while not done:
    action = env.action_space.sample()
    observation, reward, done, info = env.step(action)
    env.render()
```

Figure 5. Example code of the OPS environment.

API & sample usage. The optical and physical principle of the simulation is based on the Nonlinear-Optics-Modeling package (Hult, 2007). The OPS environment is out of the box compatible with the widely used OpenAI Gym API (Brockman et al., 2016). We show an example code of running random agent on OPS environment as Figure 5.

Features of the OPS environment. We summarize the key features of the OPS environment as follows:

- Open source optical control environment. To the best of our knowledge, this is the first open-sourced RL environment for optics control problems. The open-source licenses enable researchers to inspect the underlying code and to modify environments if required to test new research ideas.

- Scalable and difficulty-adjustable scientific environment. Many of the recent RL environments (e.g. Atari) are easy to solve. In our OPS environment, the difficulty of the environment is flexible. And the dimension of the action space is easy to scalable with stage number N. If we choose quite larger N with hard mode, controlling the environment could become quite hard. If the hard scientific control problem could be solved effectively, which would have a broader impact on many scientific control problems (Genty et al., 2020; Fermann & Hartl, 2013).

- Realistic noise. In the hard mode of the OPS environment, we module the noise distribution as the time-dependent function. It made the noise distribution of the testing environment different from the noise distribution of the training environment. This is more realistic for noise-sensitive systems (Ivanova et al., 2021). It also increases the stochasticity of the environment.

- Extendable state and structural information. When μ_t changes very slowly, we can formulate the OPS control process as a MDP. If the higher accuracy is needed, we can include the noise in the state definition, then formulate the OPS control process as a POMDP. In addition, we can explore the structural information (or physical constrain) from the function of the OPS (Figure 2) and incorporate it with RL controllers.

3. Experiments

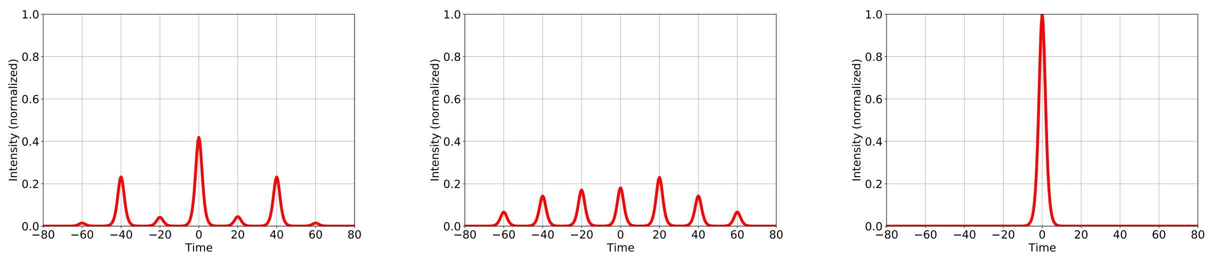
3.1. Experimental setup

As a reference, we provide benchmark results for four state-of-the-art reinforcement learning algorithms: PPO (Schulman et al., 2017), TD3 (Fujimoto et al., 2018), and SAC (Haarnoja et al., 2018b). We implement the algorithms using stable-baseline-3 (Raffin et al., 2019). The training procedure for an RL agent is divided into several episodes, each episode lasts for 200 steps. Other hyperparameters of each algorithm and training setting can be found in Appendix B.1. For each of the experimental settings, we run ten random seeds and average the results.

3.2. Results on controlling 5-stage OPS environment

In this section, we mainly report the results for the 5-stage OPS system, that stacked $2^5 = 32$ pulses. For the results of the different stage OPS system, the readers are referred to Appendix B.3. In a 5-stage OPS system, we evaluate all of the four algorithms in 3 difficulty modes of the environment: easy, medium, and hard.

Training curve (plot for training reward per step w.r.t. iterations) of PPO, TD3, and SAC algorithms has been shown in Figure 7(a) for easy mode, Figure 7(b) for medium mode, and Figure 7(c) for hard mode. As can be seen, the performance of TD3 and SAC is similar and higher than PPO for all three modes. For the difficulty mode of the environment, the convergence speed slows down and the final convergent value decreases with the increase of difficulty of the environment. As an example, in easy mode, SAC converges to the reward value of -0.04 within 100,000 steps, but it takes



(a) Initial state: easy mode (b) Initial state: medium/hard mode (c) Target optimal state

Figure 6. Rendering examples of the state of (a) initial state for easy mode, (b) initial state for medium or hard mode, (c) global optimal target state in a 2-stage OPS environment. The initial state of the easy mode has almost sacked some parts of pulses, which is more closer to the target state. The initial state of medium or hard mode is almost random and might be trapped into local optimum.

200,000 to converges to the reward value of -0.1 for hard mode.

After training the RL agents, we evaluated the performance in the testing environment. The final return (stacked pulse power P_N) under different iterations on easy mode, medium mode, and hard mode as shown in Figure 8(a), Figure 8(b), and Figure 8(c). As can be seen, although the training curve of the medium mode (Figure 7(b)) and hard mode Figure 7(c) is a bit of similar, the evaluation curve on testing environment of medium mode (Figure 8(b)) and hard mode Figure 8(c) is different. That is because the hard mode has a different noise distribution for the training and test environment. That makes the evaluation control on the testing environment for hard mode is slow to converge and achieved the lower final return. Please see the detailed results of the evaluation in Appendix B.

We reported the final return (combined pulse power P_N) of the training and testing environment on the trained policy in Table 1. Please note that the training environment and testing environment for easy and medium mode is similar, just like the classical Atari environment. The performance differences are mainly caused by randomness. We showed that the performance difference between the training and testing environment is much higher for hard mode. That is because of the different noise behavior of the training and testing environment, which makes the control complicated.

3.3. Results on different stage experiments

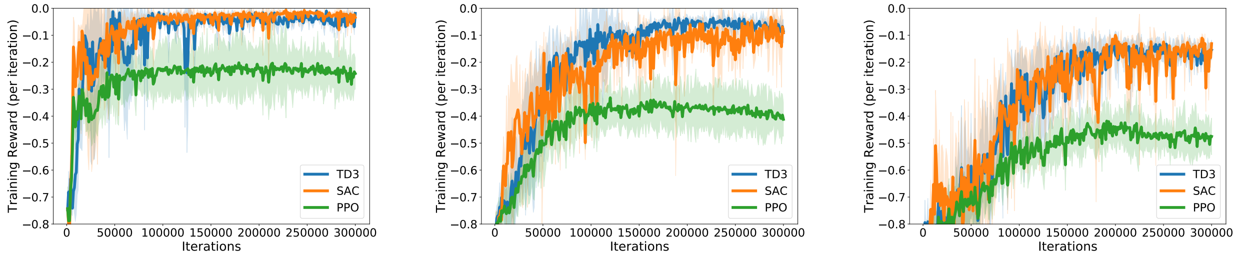
We evaluated all of the four algorithms of the different N-stage OPS environments with hard modes. Figure 9(a) shows the training curve, and Figure 9(b) shows the testing curve of TD3 on different N-stage OPS system. As can be seen, with the increase of stage number, the training convergence became slower, and the final return P_N became smaller.

We also evaluated the trained TD3 and SAC on the different N stage testing environment, as shown in Figure 10(a) and Figure 10(b). Figure 10(a) illustrated the final return P_N under different stages OPS and different difficulty modes. For 1-stage OPS, the final return P_N could reach 1. That means TD3 and SAC are able to search the global optimum for 1-stage OPS. But for 5-stage OPS with hard mode, the SAC and TD3 only could achieve the 0.8 final return. That means the controlling algorithms were trapped into a local optimum. In the real-world experiments, this case means the 20% energy loss. Figure 10(a) illustrated the training-convergence step for different stages OPS. As the stage number increase, the number of steps to training convergence increases significantly, which will slow down the training for higher stage OPS.

4. Discussion

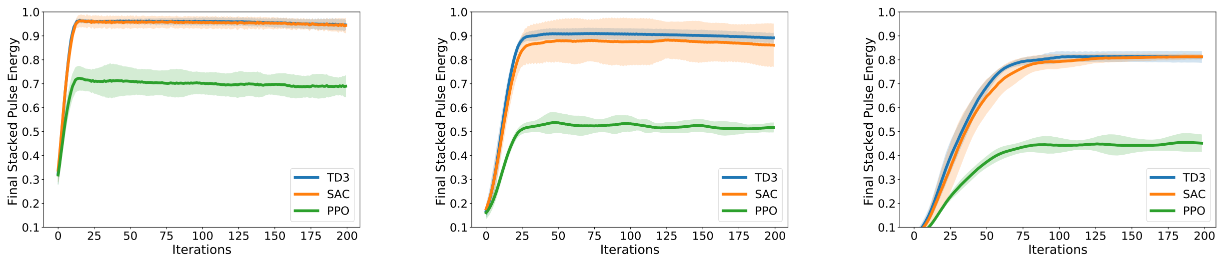
As far as we know, the previous real-world experiments of RL algorithms on complicated OPS systems are not very successful (Tünnermann & Shirakawa, 2019). One reason is slow training in a real environment. To deploy the RL algorithm in an optics system, we need to convert optical signal to analog signal using photo-detector, then convert the analog signal to digital signal using an analog-to-digital converter. These two conversions not only cost some additional time to process the signal but also introduce some noises. Another reason is that it is hard to find satisfactory RL algorithms to handle such a complex and noise-sensitive system with non-stationary noise (or non-stationary state). So one of the promising approaches is sim2real: exploring RL algorithms in the simulation environment then deploying them to the real-world control system. Note that sim2real is not easy for optical control systems. Because of the coherent interference (which is a root for many optical control problems), there are many states with zero observation signal (coherent cancellation of the pulses with opposite phases). More of the controller failed when encountering many zero observations[3]. Compared to directly training on physical

[3]Under the zero observations, the system controlled by experts conventionally.



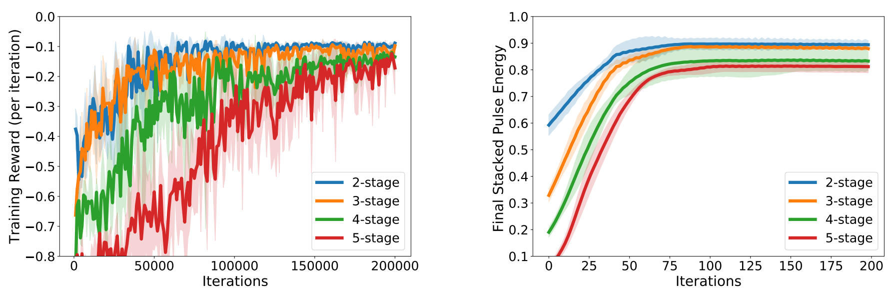
(a) Training curve: easy mode (b) Training curve: medium mode (c) Training curve: hard mode

Figure 7. Training curve for SAC, TD3, and PPO on 5-stage OPS environment for (a) easy mode, (b) medium mode, and (c) hard mode. The dashed region shows the area within the standard deviation.



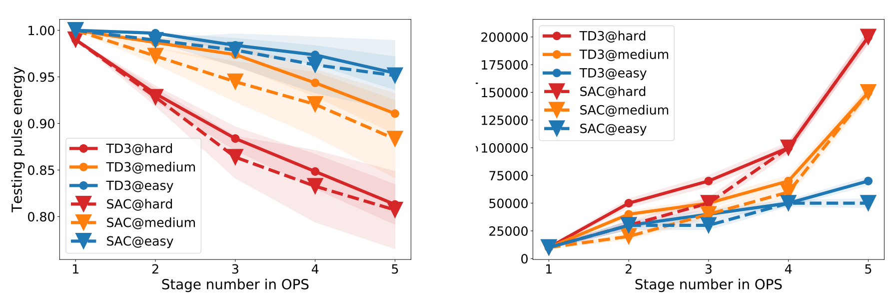
(a) Testing curve: easy mode (b) Testing curve: medium mode (c) Testing curve: hard mode

Figure 8. Evaluation of the stacked pulse power P_N (normalized) of testing environment for (a) easy mode, (b) medium mode, and (c) hard mode.



(a) Training curve: TD3 with hard mode (b) Testing curve: TD3 with hard mode

Figure 9. Comparison of the results on hard mode N-stage OPS environment with TD3 algorithms. (a) shows the training curve; (b) shows the evaluation of final return P_N of the testing environment.



(a) Testing curve (final return P_N) (b) Training convergence step

Figure 10. (a) Final return P_N of different stage OPS on testing environment controlled with TD3 or SAC. (b) Convergence steps for the training of TD3 and SAC on different stage OPS environments.

Mode	Evaluation on which environment	PPO	TD3	SAC
easy	training	0.7684 ± 0.0884	0.9580 ± 0.0189	0.9637 ± 0.0172
	testing	0.7439 ± 0.0463	0.9541 ± 0.0177	0.9514 ± 0.0231
medium	training	0.6210 ± 0.0828	0.9204 ± 0.0351	0.8945 ± 0.0501
	testing	0.6182 ± 0.0229	0.9106 ± 0.0217	0.8833 ± 0.0838
hard	training	0.5473 ± 0.0680	0.8524 ± 0.0380	0.8515 ± 0.0375
	testing	0.4461 ± 0.0300	0.8130 ± 0.0215	0.8071 ± 0.0164

Table 1. Evaluation performance of PPO, TD3, and SAC on three (easy, medium, hard) modes. Final return P_N on both the training environment and testing environment was evaluated.

OPS system, sim2real relief the problem, but it still exists. So the fast training and noise-robust RL algorithms, that are able to handle non-stationary noise and non-convex control objectives, are critical to controlling tasks in optics. Which is our main concern about implementing OPS simulation environments. Please check the further discussion about real-world system in Appendix C.

5. Conclusion

In this paper, we introduce OPS – an open-sourced simulator for controlling the pulse stacking system. To the best of our knowledge, this is the first open-sourced RL environment for optics control problems. Then we evaluated the SAC, TD3, and PPO on our proposed simulation environment. By providing an optical control simulation environment and RL benchmarks, we aim to encourage exploration of the application of RL on optical control tasks and furtherly explore the RL controllers in natural science.

In the future, we will explore the sim2real experiments: training RL algorithms in the simulation environment then deploying them to the real-world control system. Another important topic is that optical systems (or other scientific control problems) typically provide us with much richer structural information. So it is promising to incorporate additional structural information behind the OPS into the RL controllers.

References

Abuduweili, A., Yang, B., and Zhang, Z. Modified stochastic gradient algorithms for controlling coherent pulse stacking. In *Conference on Lasers and Electro-Optics*, pp. STh4P.1, 2020a.

Abuduweili, A., Yang, B., and Zhang, Z. Control of delay lines with reinforcement learning for coherent pulse stacking. In *Conference on Lasers and Electro-Optics*, pp. JW2F.33, 2020b.

Abuduweili, A., Wang, J., Yang, B., Wang, A., and Zhang, Z. Reinforcement learning based robust control algorithms for coherent pulse stacking. *Opt. Express*, 29(16):26068–26081, Aug 2021.

Agarwal, A., Kakade, S. M., Lee, J. D., and Mahajan, G. On the theory of policy gradient methods: Optimality, approximation, and distribution shift. *Journal of Machine Learning Research*, 22(98):1–76, 2021.

Astrauskas, I., Kaksis, E., Flöry, T., Andriukaitis, G., Pugžlys, A., Baltuška, A., Ruppe, J., Chen, S., Galvanauskas, A., and Balčiūnas, T. High-energy pulse stacking via regenerative pulse-burst amplification. *Optics letters*, 42(11):2201–2204, 2017.

Bansal, M., Krizhevsky, A., and Ogale, A. Chauffeurnet: Learning to drive by imitating the best and synthesizing the worst. *arXiv preprint arXiv:1812.03079*, 2018.

Baumeister, T., Brunton, S. L., and Kutz, J. N. Deep learning and model predictive control for self-tuning mode-locked lasers. *JOSA B*, 35(3):617–626, 2018.

Bellemare, M. G., Naddaf, Y., Veness, J., and Bowling, M. The arcade learning environment: An evaluation platform for general agents. *Journal of Artificial Intelligence Research*, 47:253–279, 2013.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Bukov, M., Day, A. G., Sels, D., Weinberg, P., Polkovnikov, A., and Mehta, P. Reinforcement learning in different phases of quantum control. *Physical Review X*, 8(3): 031086, 2018.

Cauwenberghs, G. A fast stochastic error-descent algorithm for supervised learning and optimization. *Advances in neural information processing systems*, 5:244–251, 1993.

Chen, Y. and Chi, Y. Harnessing structures in big data via guaranteed low-rank matrix estimation: Recent theory and fast algorithms via convex and nonconvex optimization. *IEEE Signal Processing Magazine*, 35(4):14–31, 2018. doi: 10.1109/MSP.2018.2821706.

Chi, Y., Chen, Y., and Lu, M. Y. Recent advances in nonconvex methods for high-dimensional estimation. *ICASSP tutorial*, 2018.

Dai, B., Shaw, A., Li, L., Xiao, L., He, N., Liu, Z., Chen, J., and Song, L. Sbeed: Convergent reinforcement learning with nonlinear function approximation. In *International Conference on Machine Learning*, pp. 1125–1134. PMLR, 2018.

Dembo, A. and Kailath, T. Model-free distributed learning. *IEEE Transactions on Neural Networks*, 1(1):58–70, 1990.

Dorrer, C., Kilper, D., Stuart, H., Raybon, G., and Raymer, M. Linear optical sampling. *IEEE Photonics Technology Letters*, 15(12):1746–1748, 2003.

Dressler, O. J., Howes, P. D., Choo, J., and deMello, A. J. Reinforcement learning for dynamic microfluidic control. *ACS omega*, 3(8):10084–10091, 2018.

Du, Q., Zhou, T., Doolittle, L. R., Huang, G., Li, D., and Wilcox, R. Deterministic stabilization of eight-way 2d diffractive beam combining using pattern recognition. *Optics letters*, 44(18):4554–4557, 2019a.

Du, S. S., Luo, Y., Wang, R., and Zhang, H. Provably efficient q-learning with function approximation via distribution shift error checking oracle. *arXiv preprint arXiv:1906.06321*, 2019b.

Dulac-Arnold, G., Mankowitz, D., and Hester, T. Challenges of real-world reinforcement learning. *arXiv preprint arXiv:1904.12901*, 2019.

Fermann, M. E. and Hartl, I. Ultrafast fibre lasers. *Nature photonics*, 7(11):868–874, 2013.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1587–1596. PMLR, 2018.

Genty, G., Salmela, L., Dudley, J. M., Brunner, D., Kokhanovskiy, A., Kobtsev, S., and Turitsyn, S. K. Machine learning and applications in ultrafast photonics. *Nature Photonics*, pp. 1–11, 2020.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In Dy, J. and Krause, A. (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pp. 1861–1870. PMLR, 10–15 Jul 2018a.

Haarnoja, T., Zhou, A., Hartikainen, K., Tucker, G., Ha, S., Tan, J., Kumar, V., Zhu, H., Gupta, A., Abbeel, P., et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018b.

Hult, J. A fourth-order runge–kutta in the interaction picture method for simulating supercontinuum generation in optical fibers. *J. Lightwave Technol.*, 25(12):3770–3775, Dec 2007.

Ivanova, Y. M., Pallubinsky, H., Kramer, R., and van Marken Lichtenbelt, W. The influence of a moderate temperature drift on thermal physiology and perception. *Physiology & Behavior*, 229:113257, 2021.

Izawa, J., Kondo, T., and Ito, K. Biological arm motion through reinforcement learning. *Biological cybernetics*, 91(1):10–22, 2004.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

Miryoosefi, S., Brantley, K., Daumé III, H., Dudík, M., and Schapire, R. Reinforcement learning with convex constraints. *arXiv preprint arXiv:1906.09323*, 2019.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. Playing atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*, 2013.

Oord, A. v. d., Dieleman, S., Zen, H., Simonyan, K., Vinyals, O., Graves, A., Kalchbrenner, N., Senior, A., and Kavukcuoglu, K. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

Pomerleau, D. An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems; Morgan Kaufmann Publishers Inc.: Burlington, MA, USA*, 1998.

Raffin, A., Hill, A., Ernestus, M., Gleave, A., Kanervisto, A., and Dormann, N. Stable baselines3. https://github.com/DLR-RM/stable-baselines3, 2019.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shen, Y.-R. The principles of nonlinear optics. *New York*, 1984.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. Deterministic policy gradient algorithms. In Xing, E. P. and Jebara, T. (eds.), *Proceedings of the 31st International Conference on Machine*

Learning, volume 32 of *Proceedings of Machine Learning Research*, pp. 387–395, Bejing, China, 22–24 Jun 2014. PMLR. URL http://proceedings.mlr.press/v32/silver14.html.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., Van Den Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 529(7587):484–489, 2016.

Stark, H., Müller, M., Kienel, M., Klenke, A., Limpert, J., and Tünnermann, A. Electro-optically controlled divided-pulse amplification. *Optics express*, 25(12):13494–13503, 2017.

Sun, C., Kaiser, E., Brunton, S. L., and Kutz, J. N. Deep reinforcement learning for optical systems: A case study of mode-locked lasers. *Machine Learning: Science and Technology*, 1(4):045013, 2020.

Sutton, R. S. and Barto, A. G. *Reinforcement learning: An introduction*. MIT press, 2018.

Tünnermann, H. and Shirakawa, A. Delay line coherent pulse stacking. *Opt. Lett.*, 42(23):4829–4832, Dec 2017.

Tünnermann, H. and Shirakawa, A. Deep reinforcement learning for coherent beam combining applications. *Opt. Express*, 27(17):24223–24230, Aug 2019.

Valensise, C. M., Giuseppi, A., Cerullo, G., and Polli, D. Deep reinforcement learning control of white-light continuum generation. *Optica*, 8(2):239–242, 2021.

Vinyals, O., Ewalds, T., Bartunov, S., Georgiev, P., Vezhnevets, A. S., Yeo, M., Makhzani, A., Küttler, H., Agapiou, J., Schrittwieser, J., et al. Starcraft ii: A new challenge for reinforcement learning. *arXiv preprint arXiv:1708.04782*, 2017.

Wetzstein, G., Ozcan, A., Gigan, S., Fan, S., Englund, D., Soljačić, M., Denz, C., Miller, D. A., and Psaltis, D. Inference in artificial intelligence with deep optics and photonics. *Nature*, 588(7836):39–47, 2020.

Yang, B., Liu, G., Abulikemu, A., Wang, Y., Wang, A., and Zhang, Z. Coherent stacking of 128 pulses from a ghz repetition rate femtosecond yb:fiber laser. In *Conference on Lasers and Electro-Optics*, pp. JW2F.28, 2020.

Zhang, F., Leitner, J., Milford, M., Upcroft, B., and Corke, P. Towards vision-based deep reinforcement learning for robotic motion control. *arXiv preprint arXiv:1511.03791*, 2015.

Zhou, P., Liu, Z., Wang, X., Ma, Y., Ma, H., Xu, X., and Guo, S. Coherent beam combining of fiber amplifiers using stochastic parallel gradient descent algorithm and its application. *IEEE Journal of Selected Topics in Quantum Electronics*, 15(2):248–256, 2009.

A. Additional Information of Optical Pulse Stacking

A.1. System configuration

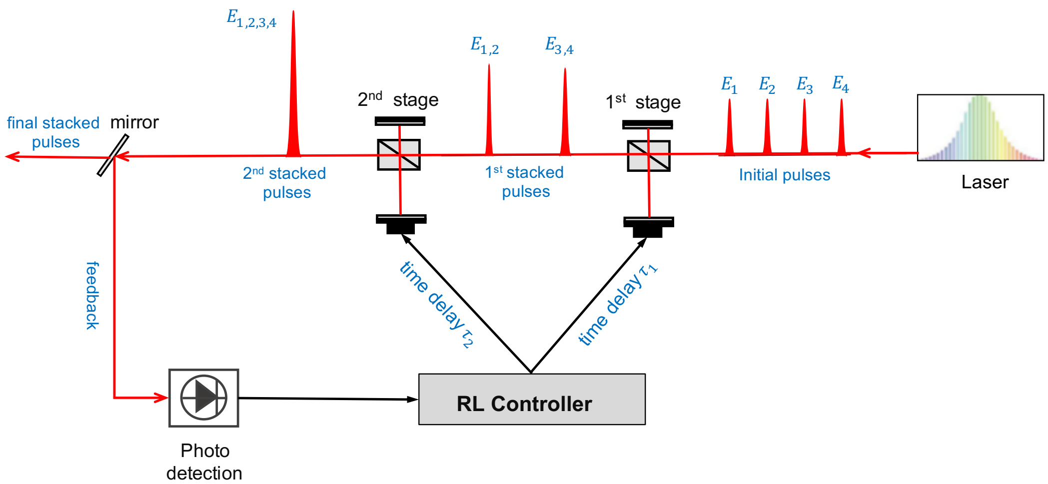


Figure 11. Configuration of optical pulse stacking (OPS) system. Only a 2-stage OPS system was plotted for simplicity.

The configuration of the optical pulse stacking (OPS) system is shown in Figure 11. The source laser delivers a train of periodic optical pulses. Given the base wave function of the laser pulse $E(t)$ and period T, each laser pulse in Figure 11 can be described as:

$$E_1 = E(t), E_2 = E(t+T), E_3 = E(t+2T), E_4 = E(t+3T). \tag{6}$$

Then the laser pulses were sent to the n-stage OPS system. In each OPS time delay stacking module, a time delay should be given to former pulses for every two consecutive pulse pairs.

The demonstration of each OPS time delay stacking module is shown in Figure 12. Figure 12(a) shows the initial state of the two pulses before processing by this stage OPS time delay. Figure 12(b) $\sim$ Figure 12(e) show the (chronological) procedures of stacking two pulses by imposing additional time delay. The former pulse E_1 was refracted by the splitter to undergo additional delay lines (vertical path between "mirror" and "time delay controller and mirror" in Figure 12(a)). The latter pulse directly transmitted the splitter. If the displacement of the additional delay line is d_1, then the additional time delay imposed to E_1 is $\tau_1 = d_1/c$, where c is the light speed. Thus, in experimental implementation, the time delay of the pulse was imposed by additional delay line displacement. The value of the delay line displacement is controlled by the RL controller.

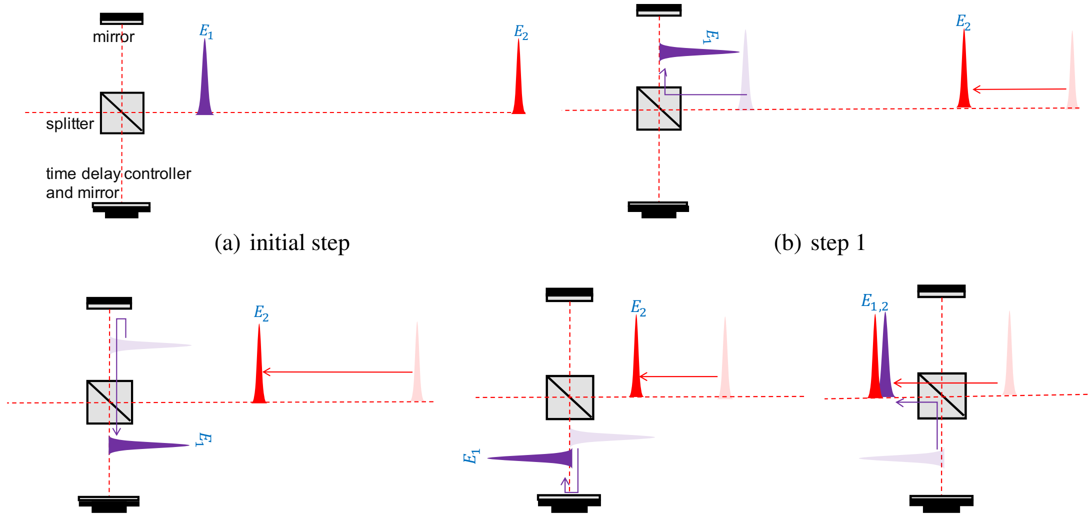
In OPS system, the 1st stage time-delay controller imposes the time delay τ_1 on pulse E_1 to stack with pulse E_2 to create the stacked pulses $E_{1,2} = E(t+\tau_1) + E(t+T)$. Similarly, After imposing time delay, pulse E_3 could be stacked with pulse E_4 to create $E_{3,4} = E(t+2T+\tau_1) + E(t+3T)$. In the 2nd stage OPS, the time delay τ_2 was further imposed to $E_{1,2}$ to make it stacking up with $E_{3,4}$ to create $E_{1,2,3,4}$:

$$E_{1,2,3,4} = E(t+\tau_1+\tau_2) + E(t+T+\tau_2) + E(t+2T+\tau_1) + E(t+3T). \tag{7}$$

If noise was ignored, when $\tau_1 = T, \tau_2 = 2T$, $E_{1,2,3,4}$ achieved the maximum value $4E(t+3T)$. Furthermore, for N-stage OPS system, 1st, 2nd, $\ldots$, N-th time delay $\tau_1, \tau_2, ..., \tau_N$ matches to $2^0, 2^1, \cdots, 2^{N-1}$ times of the pulse period T, the 2^N pulses will be perfectly stacked, and the power of the output pulse reaches the global maximum. In practice, noise can not be ignored, so the exact value of time delay $\tau_1, \tau_2, ..., \tau_N$ could be adjusted according to the feedback.

A.2. Real physical system

The real physical OPS system is shown in Figure 13. RL algorithm computes the value of each time delay $\tau_1, \tau_2, ..., \tau_N$ and sends the values to each stage delay line controller. (RL controller connected with the electric signal line of 1st, 2nd, 3rd delay line located at the bottom of the Figure 13.) Real-world OPS control experiments are quite costly and slow.



(a) initial step (b) step 1

(c) step 2 (d) step 3 (e) step 4

Figure 12. Demonstration of stacking two pulses with additional time delay. (a) shows the initial state of the 2 pulses before stacking. (b)-(e) show the (chronological) procedure of the stacking 2 pulses with imposing additional time delay. The former (latter) pulse was plotted with purple (red). The solid (transparent) plotted pulse shows the pulse position at the current (last) step. The arrow denotes the shifting value of a pulse.

B. Additional Experiments

B.1. Experimental setting

We evaluated the performance of PPO, TD3, and SAC in our OPS environment. For each of PPO, TD3, and SAC, we performed hyperparameter search to achieve better performance. For the search, we trained on 5-stage OPS environment with medium difficulty. Each of the hyperparameter sets was repeated with 3 random seeds. For each algorithm, the best hyperparameter set was decided based on the final performance in the testing environment. After the search, each of the best hyperparameter sets was used to run experiments with 10 different random seeds on all scenarios. The hyperparameter range and selected value of TD3 can be found in Table 2, hyperparameter range and selected value of SAC can be found in Table 3, and hyperparameter range and selected value of PPO can be found in Table 4.

Hyperparameter	Range	Best-selected
Size of the replay buffer	{1000,10000,100000}	10000
Step of collect transition before training	{100, 1000, 10000}	1000
Unroll Length/n-step	{1,10, 100}	100
Training epochs per update	{1,10, 100}	100
Discount factor (γ)	{0.98, 0.99, 0.999}	0.98
Noise type	{'normal', 'ornstein-uhlenbeck', None}	'normal'
Noise standard value	{0.1, 0.3, 0.5, 0.7, 0.9}	0.7
Learning rate	{0.0001, 0.0003, 0.001,0.003,0.01}	0.001
Policy network hidden layer	{1, 2, 3}	2
Policy network hidden dimension	{64, 128, 256}	256
Optimizer	Adam	Adam

Table 2. TD3: ranges used during the hyperparameter search and the final selected values.

B.2. Transfer trained policy

It is possible to transfer the trained policy between different OPS environments. The major difference between the simulation and real-world environments is the different noise levels. We conduct an simulated experiment to show the transferability between different noise levels.

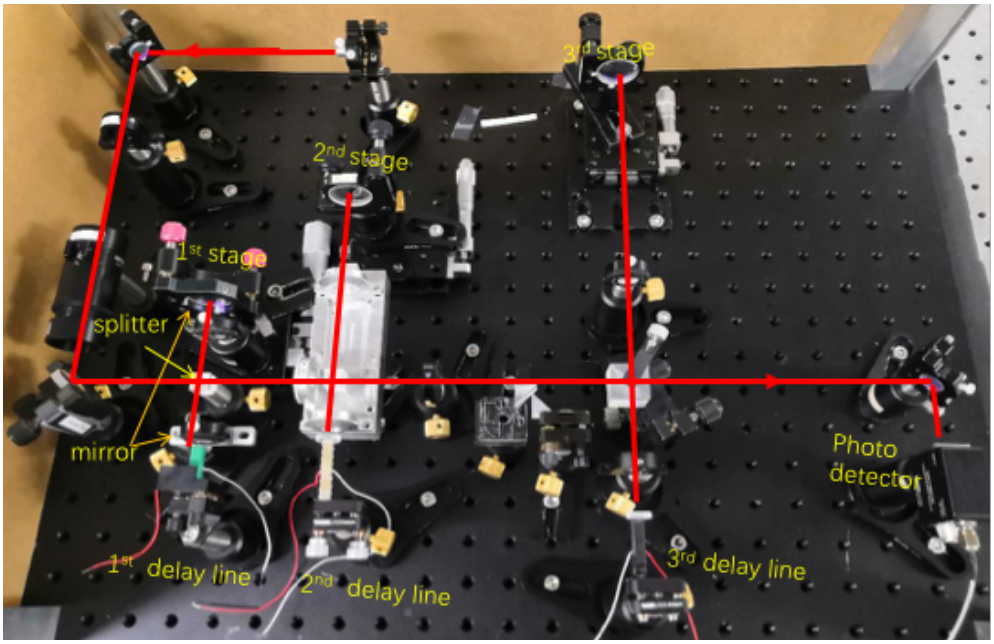


Figure 13. Real world optical pulse stacking system.

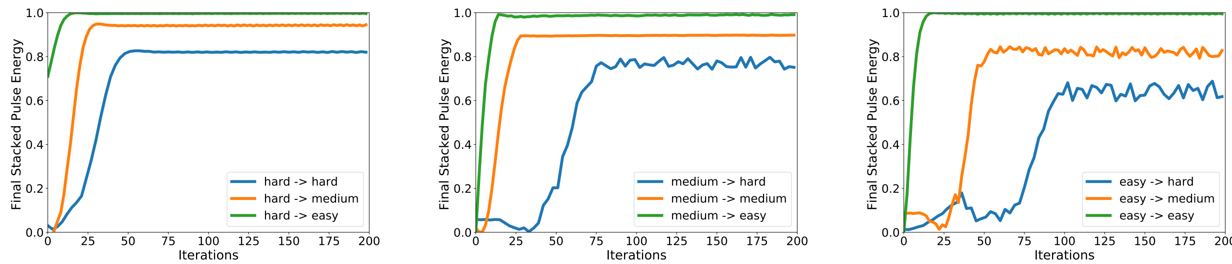
Hyperparameter	Range	Best-selected
Size of the replay buffer	{1000,10000,100000}	10000
Step of collect transition before training	{100, 1000, 10000}	1000
Unroll Length/n-step	{1,10, 100}	1
Training epochs per update	{1,10, 100}	1
Discount factor (γ)	{0.98, 0.99, 0.999}	0.98
Generalized State Dependent Exploration (gSDE)	{True, False}	True
Soft update coefficient for "Polyak update" (τ)	{0.002,0.005, 0.01, 0.02}	0.005
Learning rate	{0.0001, 0.0003, 0.001,0.003,0.01}	0.001
Policy network hidden layer	{1, 2, 3}	2
Policy network hidden dimension	{64, 128, 256}	256
Optimizer	Adam	Adam

Table 3. SAC: ranges used during the hyperparameter search and the final selected values.

Hyperparameter	Range	Best-selected
Unroll Length/n-step	{128,256,512,1024,2048}	1024
Training epochs per update	{1,5,10}	10
Clipping range	{0.1,0.2,0.4}	0.2
Discount factor (γ)	{0.98, 0.99, 0.999}	0.98
Entropy Coefficient	{0, 0.001, 0.01, 0.1}	0.01
GAE (λ)	{0.90, 0.95, 0.98, 0.99}	0.95
Value function coefficient	{0.1,0.3,0.5,0.7,0.9}	0.5
Learning rate	{0.0001, 0.0003, 0.001,0.003,0.01}	0.001
Gradient norm clipping	{0.1, 0.5, 1.0, 5.0}	0.5
Policy network hidden layer	{1, 2, 3}	2
Policy network hidden dimension	{64, 128, 256}	256
Optimizer	Adam	Adam

Table 4. PPO: ranges used during the hyperparameter search and the final selected values.

In our simulation environment, the noise level is dependent on the difficulty mode. Then we explore the transferability of the trained policy between ”easy”, ”medium”, and ”hard” modes. After trained the policy on ”hard” mode environment, we tested the trained policy on ”hard”, ”medium”, and ”easy” environment, as shown in Figure 14 (a). Transfer results of ”medium” trained policy and ”easy” trained policy are shown in Figure 14 (b) and Figure 14 (c) respectively. As can be seen, We can perfectly transfer the harder mode trained policy to easier mode environments. When the easier mode training strategy is transferring to a harder mode environment, the performance may drop, and there is a jitter in pulse energy. Please see the discussion about the real-world and simulated experiments in Appendix C.



(a) hard trained (b) medium trained (c) easy trained

Figure 14. Demonstration of the transfer performance of the trained policy on (a) hard mode training env; (b) medium mode training env; (c) easy mode training env.

Figure 14 shows that it is possible to transfer the trained policy between different noise levels, but it is more useful to train the policy in a harder environment than tested on the easier environment. Thus, we could explore fast and robust controlling algorithms in more harder simulation environment (with introducing more noise and more uncertainty in the simulation) then deploy the trained policy to real-world physical systems.

B.3. Results on controlling OPS environment

We reported the training curve (training reward w.r.t. iterations) and testing curve (return (stacked pulse power P_N) w.r.t. testing iterations) on the 4-stage OPS environment in Figure 15, and on the 6-stage OPS environment in Figure 16. As can be seen, the performance of TD3 and SAC is higher than PPO. Compared with Figure 15 (4-stage), and Figure 7 (5-stage) to Figure 16 (6-stage), with the increase of stage number, the training convergence became slower, and the final return P_N became smaller, especially for medium mode and hard mode difficulty.

B.4. Demonstration of the controlling OPS environment

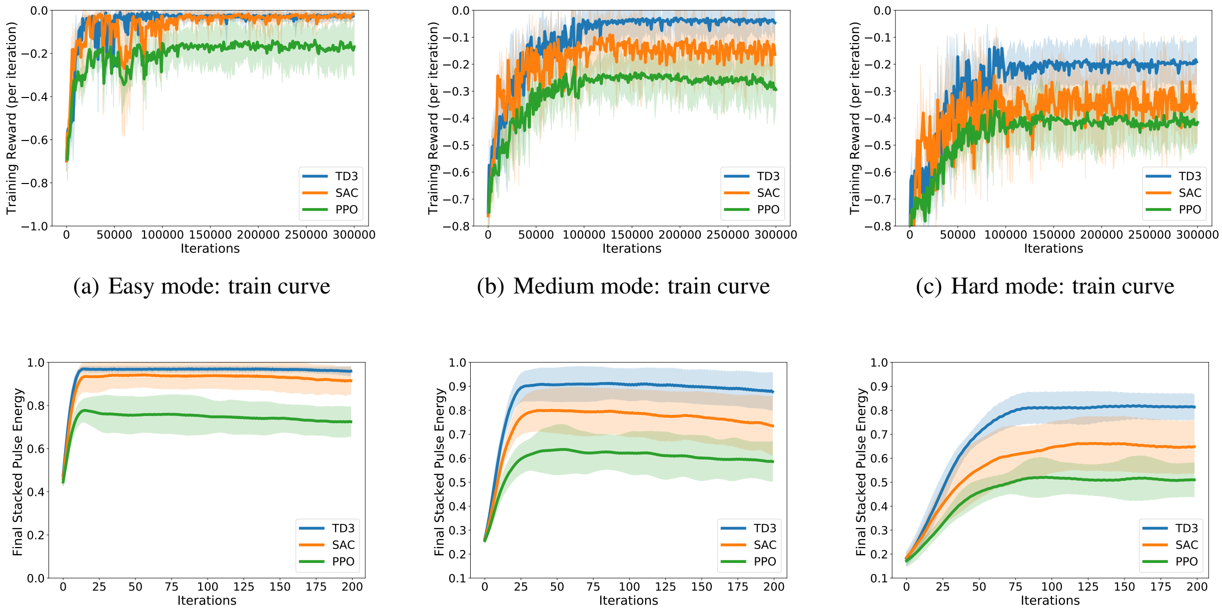
Figure 17 shows the pulse trains on a 5-stage hard mode OPS system controlled by TD3 from the random initial state. It is seen that TD3 algorithm could achieve (local) maximum power within 40 iterations.

C. Discussion

C.1. Real-world environment and simulation environment

Deploying the RL algorithm in the real-world optics system requires converting optical signal to electrical analog signal using photo-detector (PD), then converting the analog signal to digital signal using an analog-to-digital converter (ADC). These two conversions cost some additional time to process the signal and cause feedback delay. At a conservative estimate, the regular PD and ADC processing takes 0.01s per step. Then it would be possible to implement deep reinforcement algorithms on FPGAs to create control output by the feed of digital observation signal. In a proper implementation, FPGA computing time would be less than 0.01s per step. Including signal converting, neural-network inference time, and time-delay of the optical-mirror driver (controller), the time cost per control step is in the magnitude of 0.1s [4]. But in our simulation system, we could speed up the control step by at least 10 times (with GPUs). More importantly, for RL training on real-world OPS systems, it needs to manually tune the optical devices when the optical beams are totally misaligned caused by the exploring process of RL. The initial alignment of the complex OPS system is usually tuned by experts to take several hours even

[4]With expensive high-speed PD, AD/DA cards, and optical mirror driver, as well as efficient FPGA implementation, the control-speed time would be reduced to 0.01s. But it will increase the budget of the devices.



(a) Easy mode: train curve (b) Medium mode: train curve (c) Hard mode: train curve

(d) Easy mode: testing curve (e) Medium mode: testing curve (f) Hard mode: testing curve

Figure 15. 4-stage OPS experiments. Training reward was plotted for (a) easy mode, (b) medium mode, and (c) hard mode. Evaluation of the stacked pulse power P_4 (normalized) of testing environment was plotted for (d) easy mode, (e) medium mode, and (f) hard mode.

several days, that the time-cost is depending on the system complexity[5]. But in our simulation system, the initial alignment could be done by simply "reset" the environment. So the value of the simulation environment can be summarized as:

- Faster control process than a real-world experiment.
- Easy to "reset" (and initial align) the environment, while it takes a lot of works to reset or initial align a real-world experiment.
- It is safer and cheaper. In real-world experiments, it has potential risk when the optical beams are totally misaligned, because refracted light is non-predictable and may shed on experimenters.

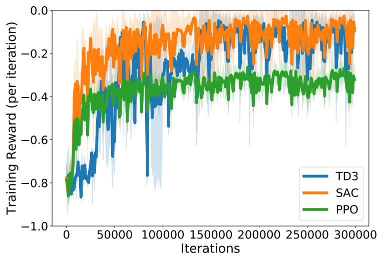
C.2. Real-world Evaluation

The impact of the simulation must be valued by the real measurement. Part of the correctness of our simulation has been evaluated by the simplified beam combining experiments (Tünnermann & Shirakawa, 2019; Yang et al., 2020). Specifically, (Tünnermann & Shirakawa, 2019) implemented an simple real experiment and the same simulation, the authors found the simulation is valuable. Our simulation and experimental settings are complicated than (Tünnermann & Shirakawa, 2019), but the physics behind them is the same. Actually, if we set stage number =1, our simulation is almost the same as (Tünnermann & Shirakawa, 2019). We will do detailed real experiments and justification in the near future.

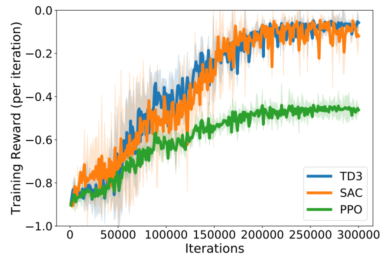
C.3. Potential Impact and additional Related works

Machine learning community. High-dimensional real-world reinforcement learning problems are extremely challenging(Dulac-Arnold et al., 2019). In our simulation environment, if we choose a quite large N-stage number with hard mode, controlling the environment could become high-dimensional and difficult. Few recent works studied the distribution shift in RL(Agarwal et al., 2021; Du et al., 2019b). In the hard mode of the OPS environment, the noise distribution of the testing environment is different from the noise distribution of the training environment. Therefore our simulation environment

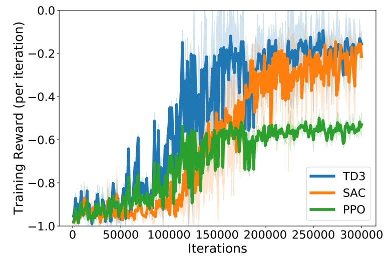
[5] We cannot detect any stacking signal when the optical beams are totally misaligned. So the RL algorithms would fail. It needs to align manually in this scenario.



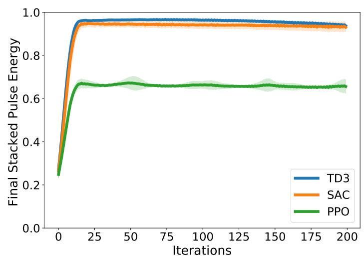
(a) Easy mode: train curve



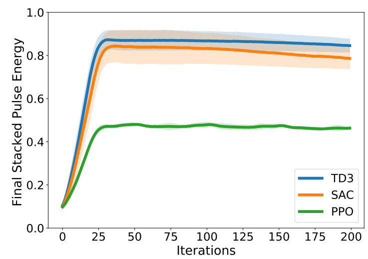
(b) Medium mode: train curve



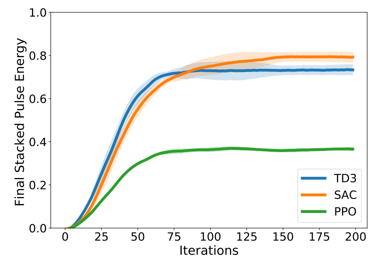
(c) Hard mode: train curve



(d) Easy mode: testing curve



(e) Medium mode: testing curve

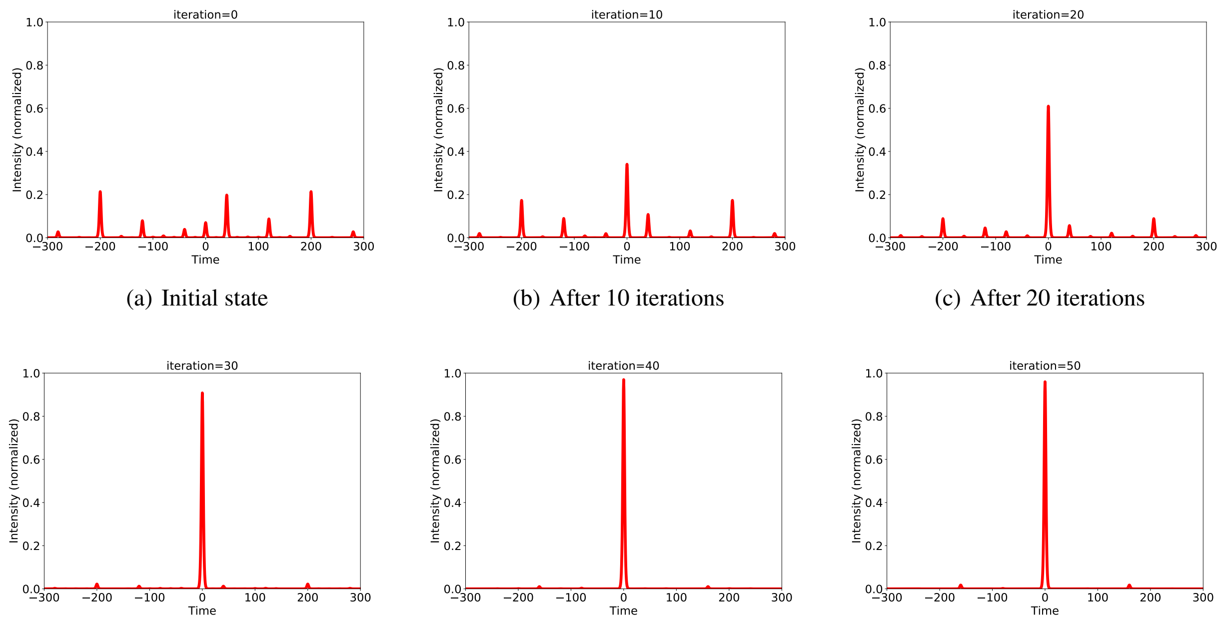


(f) Hard mode: testing curve

Figure 16. 6-stage OPS experiments. Training reward was plotted for (a) easy mode, (b) medium mode, and (c) hard mode. Evaluation of the stacked pulse power P_6 (normalized) of testing environment was plotted for (d) easy mode, (e) medium mode, and (f) hard mode.

is beneficial to solve the hard and realistic reinforcement learning problems. In recent years, statistical procedures have been developed to promote low-dimensional structures using convex relaxations, rather than directly solving the nonconvex problems (Chen & Chi, 2018; Chi et al., 2018). As shown in Figure 2, we know the function of the OPS objective (if ignoring noise). The function typically provides us with much richer structural information and physical constraints. So it is possible to explore the additional information about the function of the OPS and incorporating it with RL algorithms. In many of the real-world cases, we are not interested in "generic" nonconvex problems, but rather, we focus on more specific nonconvex control with physical constrain or some known objective function (Miryoosefi et al., 2019). Exploring the nonconvex and periodic objective of OPS would benefit the real-world RL problems that including some structural information.

Optics community. High pulse energy lasers can be used in laser accelerators, large-scale material processing, and medicine (Fermann & Hartl, 2013). Optical (coherent) pulse stacking is one of the easiest and promising ways to scale the pulse energy (Tünnermann & Shirakawa, 2017). However, the conventional control algorithms for OPS are not very effective (Du et al., 2019a). RL methods are able to control this kind of multi-dimensional and nonlinear environment. Similar to our OPS control system, all of the optical control problems are affected by the nonlinearity and periodicity of the light inference (as shown in Figure 2), including coherent optical inference (Wetzstein et al., 2020) and linear optical sampling (Dorrer et al., 2003), which can be used for precise measurement, industrial manufacturing, and scientific research. We believe our simulation is one of the important and typical optical control environments. Beyond OPS, RL methods have the potential to drive the next generation of optical laser technologies even the next generation of scientific control technologies(Genty et al., 2020). This is because many phenomena in optics are nonlinear and multidimensional, with noise-sensitive dynamics that are extremely challenging to model using conventional methods.



(a) Initial state (b) After 10 iterations (c) After 20 iterations

(d) After 30 iterations (e) After 40 iterations (f) After 50 iterations

Figure 17. Demonstration of the controlling 5-stage OPS hard mode testing environment by TD3 algorithm after training. (a): initial state of pulses; (b) pulse state after 10 control iterations; (c) pulse state after 20 control iterations; (d) pulse state after 30 control iterations; (e) pulse state after 40 control iterations; (e) pulse state after 50 control iterations.