Aligning an optical interferometer with beam divergence control and continuous action space

Anonymous Author(s)
Affiliation
Address
email

Abstract: Reinforcement learning is finding its way to real-world problem application, transferring from simulated environments to physical setups. In this work, we implement vision-based alignment of an optical Mach-Zehnder interferometer with a confocal telescope in one arm, which controls the diameter and divergence of the corresponding beam. We use a continuous action space; exponential scaling enables us to handle actions within a range of over two orders of magnitude. Our agent trains only in a simulated environment with domain randomizations. In an experimental evaluation, the agent significantly outperforms an existing solution and a human expert.

Keywords: sim-to-real, robotics, optical interferometer

1 Introduction

Reinforcement learning (RL) demonstrates incredible success in simulated environments, surpassing a human expert in Atari [1], chess, shogi and Go [2], as well as in more complicated games such as Dota2 [3] and StarCraft [4]. In robotics, reinforcement learning shows remarkable results in pushing [5], grasping [6] and stacking objects [7]. RL enables robots to walk, overcoming different obstacles [8] and learn agile animal skills [9]. However, RL agents are not yet widely used in real robotics and can hardly be compared to humans in a physical environment.

The principal problems of real-world applications of RL are nonstationarity and stochasticity of the physical environment, complexity and time intensiveness of data acquisition as well as unsafety of training and evaluation of policies on real robots [10]. A common method to handle these challenges is to train an agent in a high-fidelity simulation (source domain) of the real environment (target domain). An agent can then be transferred to a physical robot. The main limitation of this approach is inevitable discrepancy between the source and target domains that lead to performance loss after transfer. This limitation can be mitigated with the randomization of simulator parameters [5, 9, 11].

A promising application domain for smart robotics is experimental optics. Optical physicists work with installations consisting of hundreds or thousands of elements. These setups need to be thoroughly aligned before measurements can be conducted. Dependent on the setup size, such a setup can require several hours of work by a group of experts on a daily basis. The automation of this procedure could drastically enhance the productivity of research groups. Since the alignment is an iterative decision-making process with a well-defined reward function, it is natural to formulate and solve it in terms of reinforcement learning.

The alignment of a large optical setup can typically be separated into modular tasks which can be completed in sequence. A common modular task is the alignment of a Mach-Zehnder interferometer (MZI), which consists in matching the optical modes of the electromagnetic waves in two paths so that they exhibit a visibility of unity when interfering with each other. The robotization of MZI alignment was studied by Sorokin *et al.* [12], where this problem was treated as a partially observable Markov decision process, with the observation being a sequence of interference patterns observed on a camera as the relative phase of the two arms is being varied. They trained a discrete-action (VD D3QN [13]) agent, dubbed *Interferobot*, in a simulated environment with domain random-

Submitted to the 5th Conference on Robot Learning (CoRL 2021). Do not distribute.

izations and successfully transferred the trained agent to a real setup. The trained agent achieved superhuman-level performance in terms of speed and quality of alignment.

However, the task of Ref. [12] was limited to matching the spatial positions and directions of the two otherwise identical beams. It is much simpler than those alignment tasks that occur in practice, which additionally require matching the geometric sizes of the beams, their divergence, and, in the case of pulsed lasers, arrival time. These additional features of the experiment result in more complex interference patterns (a richer observation space in RL parlance), hence requiring a more sophisticated agent for their interpretation.

The main contribution of our work is to develop a new agent for interferometer alignment, which addresses this challenge. Our solution is based on a different RL algorithm and features a number of important innovations. *First,* the action space was changed from discrete to continuous, which also changed the policy task from classification to regression. This enabled us to solve the issue arising in the discrete action space, consisting in the number of actions growing exponentially with the number of degrees of freedom [14]. However, it raises a new concern: the agent's actions towards the end of the alignment process need to be much finer than in the beginning; the action space spans over two orders of magnitude. We address this with our *second* innovation — exponential action rescaling, which allows the agent to effectively explore actions of different magnitudes. *Third,* by granting the agent a continuous action space, we run a risk to leave the safe space of experimental parameters, which may even result in damaging the equipment. We solve this issue by introducing penalties that discourage the policy from approaching boundaries. *Fourth,* we enhance the set of domain randomizations that help our agent perform better in the sim-to-real transfer. Our resulting agent significantly outperforms the original Interferobot and a human expert in terms of both time and quality of the alignment.

2 Related works

RL agents for robotics can be trained either on real-world data or in simulation. The advantage of the former is that the agent receives hands-on experience of the environment in which it will be tested. The shortcoming is that the acquisition of a large dataset required for training is a complicated and time-consuming process. This is especially the case when the observations are vision-based. For example, the training of a grasping agent in Kalashnikov et al. [6] required several weeks, in spite of parallelization across 7 identical robots. This complication can be addressed with the help of behavior cloning: Vecerik et al. [15] trained an agent solving the insertion task using raw visual images using dozens of expert successful and failed trajectories. Hands-on training is simplified for agents without visual observations. For example, Haarnoja et al. [8] developed a quadruped walking robot that can generalize to unseen terrains with the training requiring as little as two hours. Another approach for training robot locomotion was presented by Yang et al. [16], who collected 4.5 minutes worth of data from a simple quadrupedal robot to model the robot's dynamics and used this model to real-time action planning.

An alternative approach is to train the agent in simulation and transfer it to a real-world system afterwards. To eliminate the discrepancy between the real and simulated environments, Tobin et al. [11] proposed a simple technique called domain randomization: they trained an object detection model on simulated images with different textures, lights, object and camera positions, and demonstrated that such a model achieves high accuracy in the real world. Peng et al. [5] trained a robotic hand to push an object using randomization of physical parameters such as friction, mass, damping, etc. Another example is the vision-based task of dexterous in-hand manipulation [17], which demonstrated the effectiveness of applying randomizations to both the physical parameters and image observations.

The automation of optical system alignment is constantly evolving. In 1987 Gabler et al. [18] demonstrated automated alignment of an optical fibre used an iterative algorithm that consequently finds the maximum of a photo-detector signal with respect to each of the three fibre movement axis. Fang and Savransky [19] considered automated alignment of a system of two lenses with 8 degrees of freedom. Acquiring a focal plane image, their algorithm performs principal component analysis and Kalman filtering to calculate proper control inputs.

Deep machine learning, particularly RL, has become increasingly popular in experimental optics. For example, Sun et al. [20] used an RL agent to stabilize a mode-locked laser by controlling wave-

plates and polarizers. RL algorithms are also routinely used to optimize optical communications [21], e.g. to route traffic in optical transport networks [22, 23].

3 Mach-Zehnder interferometer

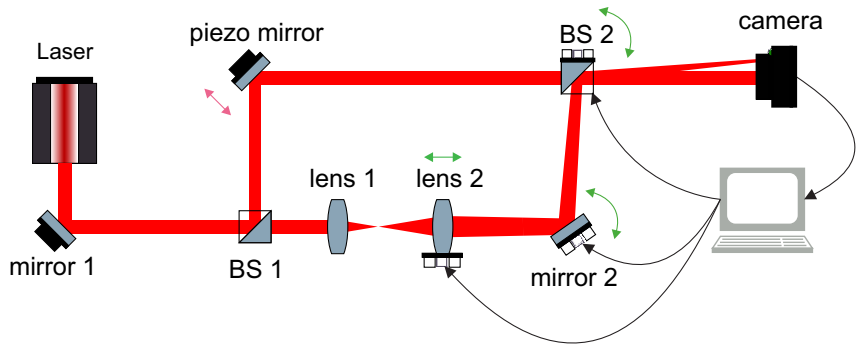


Figure 1: Conceptual scheme of the Mach-Zehnder interferometer. Lens 2, mirror 2 and BS 2 are motorized optical elements controlled by an RL agent.

Interference is a physical phenomenon that results from coherent addition of amplitudes of two or more overlapping waves; the resulting amplitude depends on the relative phase of the component waves. Interferometers, which are among the main instruments of experimental optics, use interference to precisely measure this phase difference.

In this paper we consider an MZI displayed in Fig. 1. A collimated (parallel) laser beam is divided by a beam splitter (BS 1). The two resulting beams propagate through different paths before being recombined, with the help of steering mirrors, by another beam splitter (BS 2) and viewed by a camera. One of the mirrors is mounted on a piezoelectric transducer to vary the relative phase φ of the beams. One of the beam paths (the lower one in Fig. 1) contains a system of two lenses (telescope), which control the beam divergence and size. We use motorized mounts for mirror 2 and BS 2; lens 2 is mounted on a motorized translation stage. Aligning the interferometer requires precise matching of the two beams in terms of their transverse positions, directions, radii and divergences.

The images from the camera serve as observations for our RL agent. Examples of acquired images are shown in Fig. 2. The curved shape of the interference fringes is a consequence of different divergences of the two beams. We apply an asymmetric sawtooth voltage pattern to the piezo, akin to Ref. [12], to view the temporal dynamics of fringes. The amplitude of the piezo motion corresponds to a phase difference of about 2π. For a misaligned interferometer, the piezo motion causes transverse displacement of the fringes visible in Fig. 2 (b-d), which permit the agent to extract the information about the sign of the difference of the beams' angles. When fully aligned, the interference will appear as a single blinking spot [Fig. 2 (a)].

The complexity of alignment arises because changing the angular orientations of the mirrors simultaneously affects the position and angle of the lower beam, while the lens movement changes the lower beam radius and divergence. Moreover, images received from the camera can be corrupted by noise, aberrations, ambient light and dust. The problem is additionally complicated by the inaccuracy of mounts and positioners whose relative action noise is about 4%.

The position-dependent intensity of the interference pattern can be written as $I(x, y, \varphi) = \frac{1}{2}|E_{\text{upper}}(x, y)e^{i\varphi} + E_{\text{lower}}(x, y)|^2$, where $E_{\text{upper/lower}}(x, y)$ is the amplitude of each beam. The quality metric of alignment, *visibility*, is defined as

$$V = \frac{\max_\varphi(I_{\text{tot}}(\varphi)) - \min_\varphi(I_{\text{tot}}(\varphi))}{\max_\varphi(I_{\text{tot}}(\varphi)) + \min_\varphi(I_{\text{tot}}(\varphi))} \tag{1}$$

where $I_{\text{tot}}(\varphi) = \iint I(x, y, \varphi)dxdy$ is a total light intensity in the image plane with the phase difference φ between the arms. The intensity depends on time because of periodic piezo movement that varies the phase difference between the two interferometer arms. The visibility lies in $[0, 1]$ range, where $V = 1$ corresponds to a perfectly aligned interferometer.

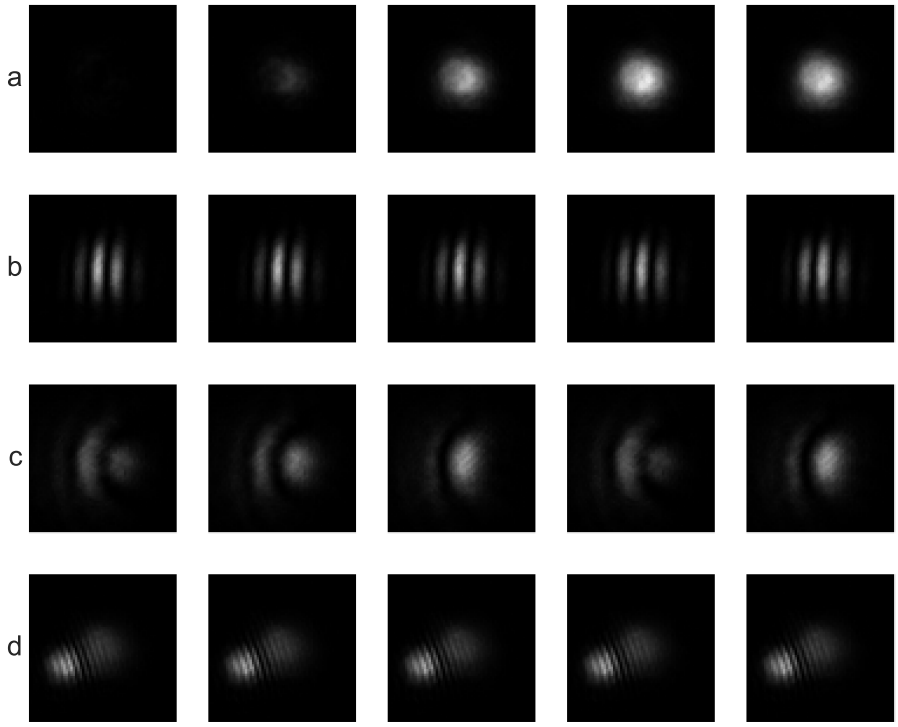


Figure 2: Camera images from the Mach-Zehnder interferometer. (b-d) Example images acquired with a misaligned interferometer. The images in each row correspond to varying path length differences of the interferometer arms.

4 Background

We consider a standard partially observable Markov decision process (POMDP) setting where the purpose of an agent is to maximize its cumulative reward during the policy execution. The observation of the environment at timestep t is defined as $o_t \in \mathcal{O}$ and is sampled from the distribution $o_t \sim U(o_t|s_t)$, where $s_t \in \mathcal{S}$ is the state. The policy of our agent is a deterministic function $a_t = \pi(o_t)$ yielding an action, which is an element of the action space $\mathcal{A}$. Following the action, the agent receives a reward $r_t : \mathcal{S} \times \mathcal{A} \rightarrow R$ and the next observation $o_{t+1} \sim U(o_{t+1}|s_{t+1})$, where s_{t+1} is produced according to a latent transition distribution $s_{t+1} \sim F(s_{t+1}|s_t, a_t)$. The discounted reward sum (return) from timestep t in each episode is $R_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$, where $\gamma \in [0, 1]$ is the discount factor and T is the horizon of the episode.

The agent's objective is to learn the policy π^* that maximizes the expected return $J(\pi) = E_{\tau \sim p(\tau|\pi)}[R_0]$ where $p(\tau|\pi)$ is the distribution of trajectories $\tau = (o_0, a_0, o_1, a_1, ..., a_{T-1}, o_T)$ produced by policy π:

$$\pi^* = \arg\max_{\pi} J(\pi). \tag{2}$$

Policy gradient. The policy π is defined by parameters θ (i.e. $\pi = \pi_\theta$), so the optimization (2) is performed with respect to the parameters: $\theta^* = \arg\max_\theta J(\pi_\theta)$. A common approach to such a problem is policy gradient [24], which iteratively improves the policy in terms of expected return via gradient ascent:

$$\theta_{t+1} = \theta_t + \alpha \frac{\partial J(\pi_\theta)}{\partial \theta}, \tag{3}$$

where α is the learning rate. The algorithms of this family vary by the method of approximating the unknown $J(\pi_\theta)$.

A common approach to continuous-action Markov decision processes is TD3 [25] which is an extension of another popular algorithm DDPG [26]. This algorithm uses three neural networks: one

(actor) for a deterministic policy and two (critics) for evaluating the action-state values $Q(s_t, a_t)$. Using two critics allows the algorithm to suffer less from overestimating the Q-values; in addition, the algorithm smoothens the Q-functions by adding Gaussian noise to the target actions when updating the parameters of the critics. Gaussian noise is also added to each policy action during the training to encourage exploration.

Domain randomization. To lower the performance losses associated with the transferring of the agent from a simulated to real-world environment, domain randomization is used. The agent is trained for a set of tasks with different dynamics $\hat{F}(s_t|s_{t-1}, a_{t-1}, \mu_{\text{sim}}) \approx F(s_t|s_{t-1}, a_{t-1}, \mu_{\text{real}})$ and sensor noise models $\hat{U}(o_t|s_t, \mu_{\text{sim}}) \approx U(o_t|s_t, \mu_{\text{real}})$ where $\mu_{\text{sim/real}}$ are sets of environment parameters. Whereas we do not know the actual μ_{real} of the real environment, varying μ_{sim} in a range containing μ_{real} helps to improve generalization and transfer quality.

5 Our method

In this work, we apply continuous control reinforcement learning methods to align the MZI. As mentioned above, the alignment is performed step by step and the actions depend on the interference patterns observed by the agent, so it can be naturally viewed as a POMDP.

The agent is trained in simulation and evaluated on a real interferometer. The training algorithm is listed in Appendix A and the MZI simulator is described in Appendix B.

State, observation and action. The state of the environment is a vector consisting of the transverse position (x, y), direction angle (α_x, α_y), radius r_{lower} and divergence curvature ρ of the lower beam in the plane of the camera. The position and transverse direction of the upper beam are assumed to be zero. This state is fully determined by the angles of the mirror and the beam splitter and the position of the lens. The range of allowed states, listed in Table 1, is restricted by the requirement that the beam remains visible on the camera and the position of the lens stays within the travel range of the translation stage.

The observation is a set of 16 consecutive 64×64 images acquired by the camera during one period of the piezo mirror. The actions are five-dimensional vectors specifying the relative angular deflections of both mirrors along the x and y axes and the linear displacement of the lens with respect to their current positions. Each action vector component lies in the $[-1, 1]$ range, where the values of ± 1 correspond to the maximum and minimum values as listed in Table 1. The absolute values of each action are restricted to the interval $[2.5 \cdot 10^{-3}, 1]$, because smaller actions do not produce observable changes in the interference pattern and fall within the uncertainty range of motorized mounts.

Episode and reset.

To compare the performance of our agent with that of Sorokin et al. [12], we keep the episode length equal to 100 actions. At the beginning of each episode, the beam in the lower path is misaligned by setting the mirror angles and lens position to random values within the allowed range.

Reward. Under the normal alignment procedure, the reward at each step is positive and consists of two terms:

$$R = V - \log(1 - V). \tag{4}$$

The first term is the visibility, which, as discussed above, is the primary metric for the interferometer alignment quality. The second term rewards high quality of the final alignment, which is critical for optical experiments; it tends to infinity for $V \to 1$ [12].

However, if the agent proposes an action that takes one of the controls out of the boundaries defined in Table 1, the episode is terminated to avoid damage to the equipment. Additionally, the agent

Element angle / position	mirror 2, x	mirror 2, y	BS 2, x	BS 2, y	lens 2
Amplitude of max deflection	$2.6 \cdot 10^{-3}$	$1.8 \cdot 10^{-3}$	$1.3 \cdot 10^{-3}$	$0.9 \cdot 10^{-3}$	7.5

Table 1: Maximum deflection of each optical element. Mirrors angles are given in radians, lens positions in millimeters.

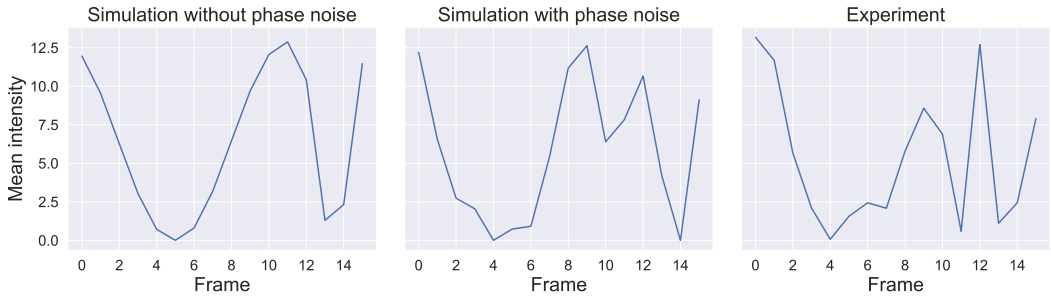


Figure 3: Effect of phase noise. The plots display the integrated intensity of the interference pattern as the path length difference is varied. The behavior becomes closer to real after applying phase noise.

is penalized with the reward of $P = -0.04$. This penalty is important during early stages of the training, when the reward (4) is normally close to zero. A small negative reward for unsafe actions will train the agent to be aware of the bounds, but will not discourage exploration. The specific value of P was handpicked via experimentation. On the other hand, when the agent is well-trained, it normally receives a significant positive reward for each step. At this stage, terminating an episode has a major negative effect on the return, thereby strongly discouraging unsafe actions.

Domain randomization. To learn a more generalized policy and facilitate sim-to-real transfer, we introduce several domain randomizations, which are used during the training. The only randomization that changes the dynamic of the environment $F(s_{t+1}|s_t, a_t)$ is the beam radius randomization within $\pm 20\%$ of the measured radius $r = 0.71$ mm. The randomization is applied at the beginning of each episode.

Additionally, we introduce the following randomizations of the observation $U(o_t|s_t)$. First, to address the noise of the camera detector, scattering, air fluctuations and dust effects, we add Gaussian noise to each pixel acquired by the camera. The standard deviation of the noise is 20% with respect to the simulated intensity value. Second, we rearrange cyclically the images within the video sequence to account for the randomness of the camera trigger. In addition, we randomize the duty cycle (fraction of time spent on the forward and backward passes) of the piezo. Third, we vary the beams' brightness by $\pm 30\%$ to model the variance of the camera exposure. All these randomizations are applied to each time step (i.e. they are constant for the 16 frames acquired in each step).

The randomizations described above have been used in Sorokin et al. [12] and have proven their effectiveness. An additional randomization we introduce in this work addresses irregularities in the motion of the piezo mirror and the frame rate, as well as fluctuations in the air density in different arms of the interferometer, all of which lead to random variation of the optical path length difference between the two MZI arms. To simulate these effects, we add Gaussian noise with the standard deviation of 0.5 rad to this phase difference.

Action rescaling. As the alignment progresses, the agent's actions become smaller and more precise. The typical action magnitude decreases during an episode by about two orders (as illustrated in the experimental Section 6 below). It is therefore desirable that the exploration noise also decrease with the action magnitude. We satisfy this by setting the agent's neural network to output a "raw action" value $a_0 \in [-1, 1]$, from which the actual action is calculated according to

$$a = \begin{cases} \text{sign}(a_0) \cdot 1000^{|a_0|-1} & \text{if } |a_0| > 0.17 \\ 0 & \text{if } |a_0| \leq 0.17 \end{cases} \tag{5}$$

This transformation produces rescaled actions with absolute values $|a| \in \{0\} \cup [2.5 \cdot 10^{-3}, 1]$.

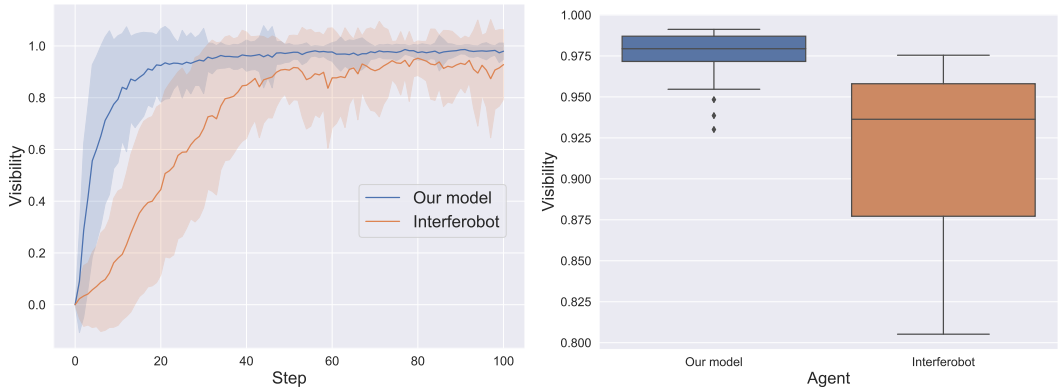
Algorithm and network architecture. We use the standard TD3 algorithm with handpicked hyperparameters to produce a deterministic policy, which yields the raw action a_0 as described above. For both critics and the actor, we use the VGG-16 [27] architecture, modified as follows. The number of convolutional layers in the encoder is set to 8, followed by three MLP layers, with no dropout. We have chosen this architecture because max pooling operations help reducing the overfitting and sensitivity to individual pixel noise. We use orthogonal initialization in all models as we found it to ensure faster convergence.

	$V \geq 0.92$	$V \geq 0.95$	$V \geq 0.98$
Human	93.9 (**0%**)	103.6 (**0%**)	129.6 (10%)
TD3 (our agent)	**56.16** (**0%**)	**75.06** (**0%**)	**120.1** (**4%**)
Interferobot	98.7 (7.6%)	116.1 (7.6%)	156.4 (10.6%)

Table 2: Comparison with a human expert and the original Interferobot. The time required to reach the visibility thresholds of 0.92, 0.95 and 0.98 are shown, together with the percentage of episodes the threshold has not been reached (in parentheses).

We train the agent with a discount factor $\gamma = 0.8$; this relatively short reward sight inspires our agent to reach high visibility faster. A Gaussian exploration noise is added to the raw action a_0 with the standard deviation decreasing exponentially from 0.5 to 0.02 during the training. Although the variance of this noise is independent of the magnitude of a_0, its effect on the actual action a is proportional to its magnitude due to the exponential dependence (5). The total number of steps is 10^6 and the replay buffer size is 10^5. Updates are performed every ten steps. The whole training on an NVidia RTX 2080 GPU takes about 26 hours.

6 Experimental evaluation



(a) Mean visibility per step

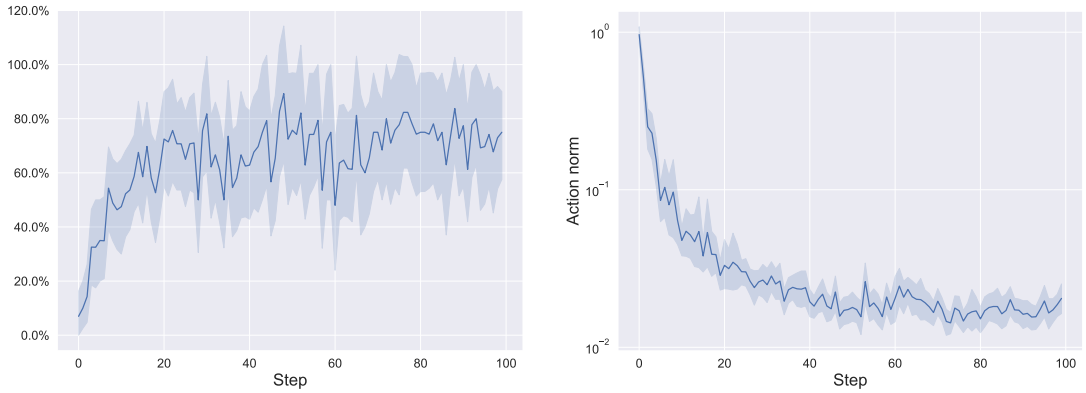
(b) Box plot of visibility

Figure 4: Comparison of Interferobot and our agent, demonstrating the advantage of the latter in convergence speed and the final alignment quality. As evident from the box plot (b), the 25-75 interquartile range for our agent is significantly narrower than that of Interferobot, indicating higher stability of the former. All results are averaged over 50 episodes.

For our experiments, we build an MZI according to the scheme shown in Fig. 1. We use a continuous HeNe laser with wavelength $\lambda = 632$ nm, Newport Picomotor mirror mounts, a Standa 8MT167-25LS linear translation stage, a CMOS camera with 16 fps acquisition rate, and a high-bandwidth photodetector to precisely measure the visibility. In addition to our agent, we also train and evaluate the original Interferobot [12] with the action space extended to include the lens movement. For each agent, we run 50 episodes of evaluation with 100 timesteps. Figure 4 shows the evaluation results. Our agent significantly outperforms Interferobot in terms of the alignment time, final visibility (averaged over the last 40 steps of each episode) and its variance.

Additionally, our agent has been compared with a human expert, who aligned the interferometer 10 times by manually turning the controls on the physical setup. As evidenced by Table 2, our agent surpasses the human both in the speed and quality of the alignment.

A common trick used by human experts in aligning optics is to turn two mirrors steering the same beam by equal and opposite angles. This action moves the position of the beam without changing its direction. Fig. 5a shows that our agent learns this method and uses it extensively: by the end of the procedure, about 70% of the actions contain such parallel movements. Fig. 5b shows the action norm as a function of the step number. The descending trend means that at the beginning



(a) Percentage of parallel actions.

(b) Norm of actions.

Figure 5: Interpreting the agent's policy. a) Percentage of actions that change the distance between beams while conserving the directions increases with the step number. b) Mean norm of actions decreases with the step number.

our agent uses large actions to obtain rough alignment and decreases the action size to fine-tune the interferometer.

Table 3 presents the results of an ablation study. A standard TD3 agent does not outperform the original Interferobot and achieves a mean visibility of $V = 0.83$. Action rescaling and phase noise randomizations significantly boost the performance of our agent, leading to a visibility of $V = 0.98$. It is notable that the phase noise randomization slightly decreases the mean visibility of the Interferobot but improves the standard deviation. This bias-variance trade-off can be due to the relative simplicity of the Interferobot model which cannot capture richer observations produced by this randomization.

Model	mean visibility for last 40 step	standard deviation
TD3 + AR + PN	**0.98**	**0.03**
TD3 + AR	0.95	0.06
Interferobot	0.92	0.12
Interferobot + PN	0.90	0.08
TD3	0.83	0.18

Table 3: Comparative evaluation with ablated agents and the original Interferobot [12]. PN: phase noise; AR: action rescaling.

7 Conclusion

We demonstrated a novel RL algorithm for the automatic alignment of an optical interferometer, which contains a system of lenses in one of its optical paths. Such a setup results in richer observation and action spaces and constiututes a better approximation of a general problem of achieving mode matching between two arbitrary Gaussian beams. Our reinforcement learning agent with a continuous action space solves the alignment problem successfully, surpassing both a previous solution and a human expert. Important innovations include exponential scaling of the action space, an additional domain randomization that helps the agent generalize to a real setup after training in simulation as well as a discrete reward structure that trains the agent to avoid unsafe actions. Our solution will work for any Mach-Zehnder interferometer with 2×2 degrees of freedom (mirror angles) controlling the position and direction, and one additional degree of freedom (lens position) controlling the width and divergence of one of the beams.

8 Acknowledgments

We acknowledge support from Russian Science Foundation (19-71-10092).

References

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

[2] D. Silver, T. Hubert, J. Schrittwieser, I. Antonoglou, M. Lai, A. Guez, M. Lanctot, L. Sifre, D. Kumaran, T. Graepel, et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144, 2018.

[3] C. Berner, G. Brockman, B. Chan, V. Cheung, P. Debiak, C. Dennison, D. Farhi, Q. Fischer, S. Hashme, C. Hesse, et al. Dota 2 with large scale deep reinforcement learning. *arXiv preprint arXiv:1912.06680*, 2019.

[4] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature*, 575(7782):350–354, 2019.

[5] X. B. Peng, M. Andrychowicz, W. Zaremba, and P. Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pages 3803–3810. IEEE, 2018.

[6] D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, et al. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on Robot Learning*, pages 651–673. PMLR, 2018.

[7] T. Haarnoja, V. Pong, A. Zhou, M. Dalal, P. Abbeel, and S. Levine. Composable deep reinforcement learning for robotic manipulation. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6244–6251. IEEE, 2018.

[8] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine. Learning to walk via deep reinforcement learning. *arXiv preprint arXiv:1812.11103*, 2018.

[9] X. B. Peng, E. Coumans, T. Zhang, T.-W. Lee, J. Tan, and S. Levine. Learning agile robotic locomotion skills by imitating animals. *arXiv preprint arXiv:2004.00784*, 2020.

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

[11] 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 *2017 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 23–30. IEEE, 2017.

[12] D. Sorokin, A. Ulanov, E. Sazhina, and A. Lvovsky. Interferobot: aligning an optical interferometer by a reinforcement learning agent. In *Advances in Neural Information Processing Systems*, volume 33, pages 13238–13248. Curran Associates, Inc., 2020.

[13] Y. Huang, G. Wei, and Y. Wang. Vd d3qn: the variant of double deep q-learning network with dueling architecture. In *2018 37th Chinese Control Conference (CCC)*, pages 9130–9135. IEEE, 2018.

[14] G. Dulac-Arnold, R. Evans, H. van Hasselt, P. Sunehag, T. Lillicrap, J. Hunt, T. Mann, T. Weber, T. Degris, and B. Coppin. Deep reinforcement learning in large discrete action spaces. *arXiv preprint arXiv:1512.07679*, 2015.

[15] M. Vecerik, O. Sushkov, D. Barker, T. Rothörl, T. Hester, and J. Scholz. A practical approach to insertion with variable socket position using deep reinforcement learning. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 754–760. IEEE, 2019.

[16] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani. Data efficient reinforcement learning for legged robots. In *Conference on Robot Learning*, pages 1–10. PMLR, 2020.

[17] O. M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. McGrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, A. Ray, et al. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39(1):3–20, 2020.

[18] C. Gabler, K. Li, S. Hackwood, and G. Beni. An optical alignment robot system. In *Integration and Packaging of Optoelectronic Devices*, volume 703, pages 8–28. International Society for Optics and Photonics, 1987.

[19] J. Fang and D. Savransky. Automated alignment of a reconfigurable optical system using focal-plane sensing and kalman filtering. *Applied optics*, 55(22):5967–5976, 2016.

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

[21] D. Wang and M. Zhang. Artificial intelligence in optical communications: from machine learning to deep learning. *Frontiers in Communications and Networks*, 2:9, 2021.

[22] J. Suárez-Varela, A. Mestres, J. Yu, L. Kuang, H. Feng, A. Cabellos-Aparicio, and P. Barlet-Ros. Routing in optical transport networks with deep reinforcement learning. *Journal of Optical Communications and Networking*, 11(11):547–558, 2019.

[23] J. Li, D. Wang, M. Zhang, and S. Cui. Digital twin-enabled self-evolved optical transceiver using deep reinforcement learning. *Optics Letters*, 45(16):4654–4657, 2020.

[24] R. S. Sutton, D. A. McAllester, S. P. Singh, Y. Mansour, et al. Policy gradient methods for reinforcement learning with function approximation. In *NIPs*, volume 99, pages 1057–1063. Citeseer, 1999.

[25] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning*, pages 1587–1596. PMLR, 2018.

[26] 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. *arXiv preprint arXiv:1509.02971*, 2015.

[27] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

Appendix

A. The pseudo code and hyperparameters

Algorithm 1: TD3 with action rescaling

Input: initial policy parameters θ, Q-function parameters ϕ_1, ϕ_2, empty replay buffer $\mathcal{D}$
Set target parameters equal to main parameters $\theta_{\text{targ}} \leftarrow \theta$, $\phi_{\text{targ},1} \leftarrow \phi_1$, $\phi_{\text{targ},2} \leftarrow \phi_2$
for i in range(number of steps) **do**
  Get observation o and select action $a_0 = \text{clip}(\pi_\theta(o) + \epsilon, -1, 1)$, where $\epsilon \sim \mathcal{N}(0, \sigma_{\text{explore}})$
  Compute rescaled action $a(a_0)$ according to Eq. (5)
  Execute a in the environment
  Get next observation o', reward r, and done signal d to indicate whether o' is terminal
  Store (o, a_0, r, o', d) in replay buffer $\mathcal{D}$
  If o' is terminal, reset environment state.
  if $i >$ `start_train_step` **then**
    for j in range(`num_epochs`) **do**
      Randomly sample a batch of transitions, $B = \{(o, a_0, r, o', d)\}$ from $\mathcal{D}$
      Compute target actions

$$a'(o') = \text{clip}\left(\pi_{\theta_{\text{targ}}}(o') + \text{clip}(\epsilon, -c, c), -1, 1\right), \qquad \epsilon \sim \mathcal{N}(0, \sigma_{\text{targ}})$$

      Compute targets

$$y(r, o', d) = r + \gamma(1 - d) \min_{i=1,2} Q_{\phi_{\text{targ},i}}(o', a'(o'))$$

      Update Q-functions by one step of gradient descent using

$$\nabla_{\phi_i} \frac{1}{|B|} \sum_{(o,a_0,r,o',d)\in B} \left(Q_{\phi_i}(o, a_0) - y(r, o', d)\right)^2 \qquad \text{for } i = 1, 2$$

      if $j \mod$ `policy_delay` $= 0$ **then**
        Update policy by one step of gradient ascent using

$$\nabla_\theta \frac{1}{|B|} \sum_{o\in B} Q_{\phi_1}(o, \pi_\theta(o))$$

        Update target networks with

$$\phi_{\text{targ},i} \leftarrow \rho_{\text{polyak}}\phi_{\text{targ},i} + (1 - \rho_{\text{polyak}})\phi_i \qquad \text{for } i = 1, 2$$
$$\theta_{\text{targ}} \leftarrow \rho_{\text{polyak}}\theta_{\text{targ}} + (1 - \rho_{\text{polyak}})\theta$$

      end if
    end for
  end if
end for

batch_size	32	π lr	10^{-5}
num_epochs	10	Q lr	10^{-4}
policy_delay	1	σ_{targ}	0.2
max_grad_norm	10	c	0.5
start_train_step	10^4 steps	optimizer	Adam
ρ_{polyak}	0.995		

Table 4: Training hyperparameters

	Distance, mm
BS 1 $\rightarrow$ Mirror 2	300
Mirror 2 $\rightarrow$ BS 2	200
BS 2 $\rightarrow$ Camera	100
BS 1 $\rightarrow$ Lens 1	50
$\text{f}_{\text{lens1}} = f_{\text{lens2}}$	50

Table 5: Setup parameters

In this section we show a full listing of our algorithm as well as the hyperparameters (aside from those mentioned in the main text) and the parameters of the physical setup.

The exponential action scaling described in the article is performed in step 5 of Algorithm 1. Importantly, while the action a is executed in the environment, the raw action a_0 is stored in the replay buffer $\mathcal{D}$ (step 8).

B. The simulator

The laser beams in the upper and lower arms of the interferometer are modelled with a Gaussian transverse profile. Their electric field amplitudes at a point in space with the coordinates (x, y, z) are given by

$$E_u = \mathrm{Re}\left[\exp\left(-\frac{x^2+y^2}{r_u^2(z)}\right)\exp\left(-i\left(k_z z + k\frac{x^2+y^2}{2\rho_u^2(z)} + \phi_{\mathrm{piezo}}(t)\right)\right)\right] \tag{6a}$$

$$E_l = \mathrm{Re}\left[\exp\left(-\frac{(x-x_0)^2+(y-y_0)^2}{r_l^2(z)}\right)\exp\left(-i\left(k_x x + k_y y + k_z z + k\frac{x^2+y^2}{2\rho_l^2(z)}z\right)\right)\right] \tag{6b}$$

where the subscripts u and l respectively refer to the upper and lower beam, (x_0, y_0) is the position of the centre of the lower beam [the upper beam is assumed centered at $(x, y) = (0, 0)$], z is the direction of beams propagation, $r(z)$ the beam radius, $\rho(z)$ the wavefront curvature radius, (k_x, k_y, k_z) the wave vector with $k = \sqrt{k_x^2 + k_y^2 + k_z^2} = 2\pi/\lambda$, and $\phi_{\mathrm{piezo}}(t)$ the phase shift that produced by the piezo mirror periodic movement. We work in the paraxial approximation, such that $k_z \gg k_x, k_y$.

Prior to the first beam splitter, the two beams have identical parameters. The propagation of the Gaussian beams is analyzed in the framework of the ABCD formalism, where the beam is characterized by a complex parameter $\frac{1}{q} = \frac{1}{\rho} - \frac{i\lambda}{\pi r^2}$ and the transformations of the beam as it propagates through the setup are given by $q' = \frac{Aq+B}{Cq+D}$, where, e.g., $\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 1 & d \\ 0 & 1 \end{bmatrix}$ for free propagation over a distance d and $\begin{bmatrix} A & B \\ C & D \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1/f & 1 \end{bmatrix}$ for a thin lens of focal length f.

The transverse field profiles (6) are calculated for both beams, after which the interference pattern and the visibility are evaluated according to

$$I(x, y, z, t) = |E_u(x, y, z) + E_l(x, y, z)|^2$$

and

$$V = \frac{\max_t(I_{\mathrm{tot}}) - \min_t(I_{\mathrm{tot}})}{\max_t(I_{\mathrm{tot}}) + \min_t(I_{\mathrm{tot}})},$$

where $I_{\mathrm{tot}}(t) = \iint_{-\infty}^{+\infty} I(x, y, t)\mathrm{d}x\mathrm{d}y$ is the intensity integrated over the beam area; the maximum and minimum are taken over the period of piezo movement.