Playing optical tweezers with deep reinforcement learning: in virtual, physical and augmented environments

IOP Publishing Mach. Learn.: Sci. Technol. 2 (2021) 035024 https://doi.org/10.1088/2632-2153/abf0f6

MACHINE LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
7 January 2021

REVISED
9 March 2021

ACCEPTED FOR PUBLICATION
22 March 2021

PUBLISHED
14 June 2021

PAPER

Playing optical tweezers with deep reinforcement learning: in virtual, physical and augmented environments

Matthew Praeger[1,*], **Yunhui Xie**[1], **James A Grant-Jacob**, **Robert W Eason** and **Ben Mills**

Optoelectronics Research Centre, University of Southampton, Southampton, SO17 1BJ, United Kingdom
[1] These authors contributed equally.
[*] Author to whom any correspondence should be addressed.

E-mail: mattp@soton.ac.uk

Keywords: optical tweezers, laser trapping, machine learning, reinforcement learning

Supplementary material for this article is available online

Abstract

Reinforcement learning was carried out in a simulated environment to learn continuous velocity control over multiple motor axes. This was then applied to a real-world optical tweezers experiment with the objective of moving a laser-trapped microsphere to a target location whilst avoiding collisions with other free-moving microspheres. The concept of training a neural network in a virtual environment has significant potential in the application of machine learning for experimental optimization and control, as the neural network can discover optimal methods for problem solving without the risk of damage to equipment, and at a speed not limited by movement in the physical environment. As the neural network treats both virtual and physical environments equivalently, we show that the network can also be applied to an augmented environment, where a virtual environment is combined with the physical environment. This technique may have the potential to unlock capabilities associated with mixed and augmented reality, such as enforcing safety limits for machine motion or as a method of inputting observations from additional sensors.

1. Introduction

Neural networks have enabled a significant wave of innovation within the field of photonics, including developments in microscopy [1–3], optical fibers [4, 5], silicon photonics [6, 7], sensing [8–10] and modeling light–matter interaction [11–13]. In the field of optical tweezers, a recent article [14] used a neural network to provide high accuracy estimates of the optical forces acting on a particle within a laser trap. This was achieved with relatively low computational overhead by training on (x, y, z) particle coordinates and (F_x, F_y, F_z) force components obtained from simulation software. Predominantly, these results have been based on the machine learning technique known as supervised learning, where a neural network is trained to predict an output for a given input, such as transforming an optical microscopy image (the input) into one with a particular type of staining or fluorescence (the output). Whilst supervised learning has proved to be extremely effective when the data is formed of input–output pairs, a different approach is needed for experimental control and optimization. Reinforcement learning [15–17] is a machine learning technique where the neural network learns by making a sequence of interactions with its environment, whilst being provided with a reward signal based on its success in achieving favorable outcomes. In other words, the neural network is free to explore the environment, via trial and error, to discover an optimal process for solving a specified problem. A particularly relevant example of reinforcement learning is given in [18], where Q-learning [19] is used to choose the laser position from a set of eight discrete actions, in order to drive micro-scale particle motions via thermophoretic forces. Aside from the use of optical trapping rather than thermophoretic forces; our work differs significantly from this because it uses microscope images as input to

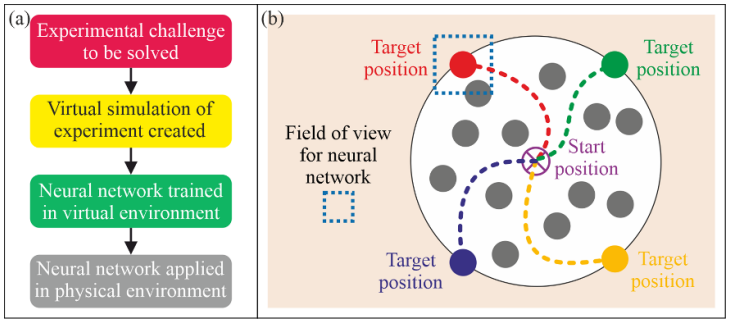

Figure 1. (a) Strategy for reinforcement learning through use of a virtual environment and (b) concept for the task described here (showing trajectories to four different target positions).

the neural network (rather than particle coordinates), it outputs continuously variable stage velocities (as opposed to discrete actions) and it makes use of a simulated environment for training.

A critical technique in reinforcement learning is the use of a virtual training environments (also known as a gyms [20]) to enable accelerated learning, where the neural network gains experience in the virtual training environment before it is applied to the physical environment (i.e. the real world). The key components of the gym concept, definition of the problem, the creation and subsequent training inside the virtual environment, and the application to the physical environment are shown in figure 1(a). The application of a gym means that the neural network training speed is dependent on computing power rather than the movement speed of physical equipment, whilst also providing other advantages such as protecting against equipment damage and alleviating material consumption. The challenge is therefore in creating a virtual environment that approximates the response of the physical experiment whilst providing a reward signal that accurately represents the consequences of equivalent actions made by the neural network in the physical environment.

Reinforcement learning in a virtual environment has recently become an important technique for robotic sample manipulation [21–23] and for collision avoidance applications such as virtual training of drones [24, 25] and autonomous driving [26]. Reinforcement learning has also recently produced human-beating performance at Atari [27], first person shooter [28, 29], strategy [30, 31], 3D multiplayer [32] and text-based [33–35] computer games, along with board games including chess, shogi, and Go [36–38]. The application of reinforcement learning to the field of photonics has recently intensified with example demonstrations including optimized laser welding [39], the control of mode-locking in an oscillator [40], the use of robotics to align an interferometer [41], perform wavefront correction [42], routing in optical transport networks [43] and coherent beam combination [44].

In this work we apply reinforcement learning at the interface of robotics and photonics to demonstrate non-contact microscale sample manipulation. The task for the neural network, as shown in figure 1(b), was to use the optical tweezers effect [45] to transport a microsphere from its initial location to a pre-defined target position without collision with other microspheres, and whilst having a limited field of view (equivalent to that observed in the physical setup). A neural network, which was trained in a virtual environment, was able to learn route finding and collision avoidance and apply this correctly when applied to the real-world physical environment. In addition, we demonstrate that it is possible to overlay additional virtual layers upon observations of the physical environment. This technique has the potential to unlock capabilities associated with mixed and augmented reality, such as adding safety limits for machine motion or for incorporating additional sensor inputs. This demonstration of reinforcement learning for control of light-matter interactions shows the potential for automation and optimization across a range of nano and microscale photonics, engineering, and bio-chemical tasks. In this manuscript, the physical environment and virtual environments are explained in sections 2 and 3, the virtual environment training process and demonstration is shown in sections 4 and 5, and the experimental results associated with the physical and then augmented environments are discussed in sections 6 and 7.

2. The physical environment

The physical environment was based on an optical tweezers education kit (Thorlabs EDU-OT2/M). The beam from a laser diode operating at 658 nm with an output power of 40 mW (Thorlabs L658P040) was expanded and focused through a Zeiss, 63×, 0.8 N.A. microscope objective lens to generate the microsphere trapping force. The X and Y motorized stages each had a maximum travel of 12 mm, a maximum velocity of 2.6 mm s^{-1}, and a minimum step size of 0.05 μm but with backlash of up to 8 μm (which was partially

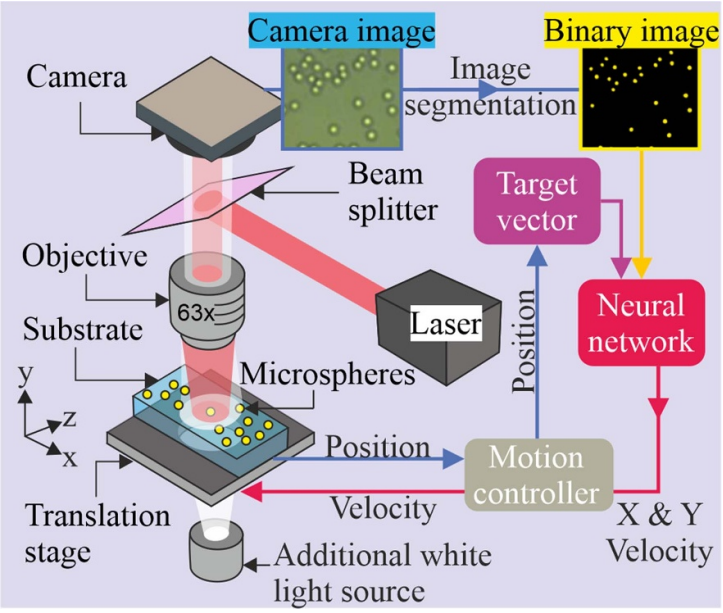

Figure 2. Schematic of physical environment.

mitigated in the control software). The motors were driven using Thorlabs KDC101 motion controllers commanded via custom python code that interfaced directly with the Thorlabs Kinesis drivers (see supplementary data for more details of the experimental apparatus, available online at stacks.iop.org/MLST/2/035024/mmedia). The sample consisted of 5 μm diameter silicon dioxide microspheres in aqueous suspension (Sigma Aldrich 44054-5ML-F) which were diluted in distilled water to achieve the desired concentration (approximately 0.125% solids by mass). The liquid sample was sealed between a glass microscope slide and a cover slip using an adhesive gasket. The stages moved the sample relative to the laser trap and microscope objective lens, hence, producing a relative movement between the trapped microsphere and all other microspheres.

As shown in figure 2, in the physical environment the neural network observed the processed microscope image of the sample and was provided with a vector to the target location, and then responded by updating the stage velocities accordingly. Specifically, at each timestep, the camera image was processed to produce a binary image that was equivalent to that created in the virtual environment. Image segmentation was achieved using a sequence of conventional image processing methods including Gaussian blur, image standardization, adaptive thresholding, erosion and dilation, where both upper and lower thresholds were applied for tolerance against small variations in the microscope focus that could cause the microsphere centers to appear either darker or lighter than the mid-tone background. The center positions of particles in the segmented image were determined using a contour-based method [46]; this information was used to render a gym-equivalent image for the neural network. At each timestep the current stage positions were used to calculate a vector in the direction of the target (this was scaled relative to the initial displacement so that it also encoded the current distance from the target). The magnitude of this vector was clipped to avoid values which were not encountered during training, where the maximum displacement was 1000 pixels. The resulting vector was provided to the neural network as a numerical observation (note that the neural networks is never directly told the current stage positions or the target position). With the present hardware, neural network actions were operated approximately four times per second, limited by the communication speed of the motion controllers.

3. The virtual environment

As shown in figure 3, the virtual gym environment was programmatically designed to simulate the response of the real-world laser tweezers experiment. When the gym environment was initialized at the start of each episode, the target location was set, a single microsphere was initialized within the laser tweezers trap and a random number of free microspheres (between 500 and 1000) were randomly positioned within the virtual environment. At each timestep of the gym environment, each free microsphere was given a random acceleration, subject to a maximum allowed velocity, in order to simulate Brownian motion. Importantly, whilst the numerical values for microsphere positions and velocities were contained within the gym

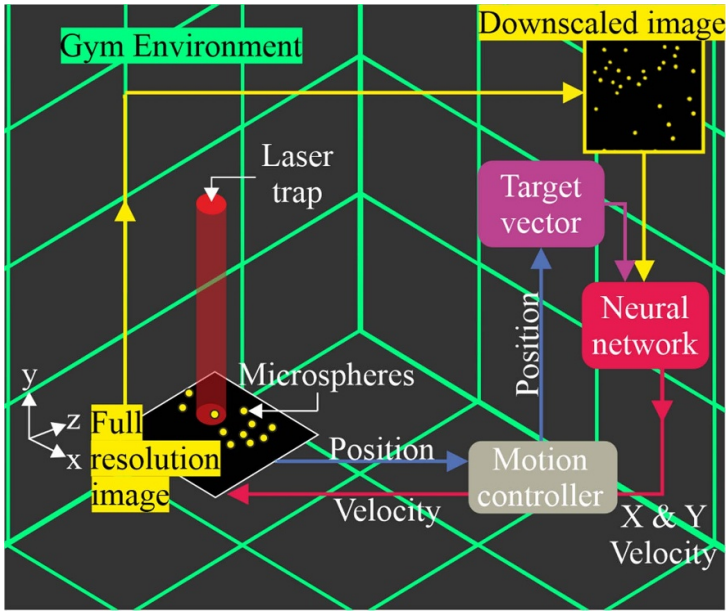

Figure 3. Schematic of the virtual gym used for testing the trained neural network.

environment (as part of the world state), these were not made available to the neural network, and instead the microsphere coordinates were used to render a binary image that simulated the field of view of the optical tweezers microscope. To provide a match to the optical tweezers apparatus in the physical environment, an image resolution of 1280 × 1024 pixels was used and microsphere diameters were set at 54 pixels (corresponding to the 5 μm diameter size of the silicon dioxide microsphere used). To reduce computational requirements, the virtual microscope images were rendered at a lower resolution via a shrink factor of 4× resulting in an image with 320 × 256 pixels. This reduced-resolution, binary image was sent to the neural network at each timestep (we term this the image observation).

The gym environment also provided the neural network with a vector that denoted the displacement of the target location relative to the current stage position (we term this the numerical observation). At each timestep, the neural network was given the image and numerical observations and it used this information to select an action (velocity values for the *X* and *Y* stages). The world state was then calculated for the start of the next timestep, where the laser focus was assumed to remain central in the rendered camera image, with the stage motions simulated by translating all microspheres except for the one designated as trapped. The episode was immediately ended if multiple microspheres entered the laser trap region (i.e. a collision occurred) or if the laser trap was successfully maneuvered to the target location.

4. Training in the virtual environment

The reinforcement learning was carried out using the Twin Delayed Deep Deterministic Policy Gradient (TD3) algorithm [47]. As shown in figure 3, the neural network received a binary image (corresponding to the sample) and a two-component vector that pointed in the direction of the target relative to the current *X* and *Y* stage positions. TD3 is an example of an actor-critic method; the actor component is responsible for choosing the next action based on the current state (i.e. the policy μ) whilst the critic component is responsible for estimating the state-action value function Q (i.e. the expected return if a particular action is taken from a given state). For both the actor and critic, TD3 includes target networks that are updated less frequently (i.e. 'delayed') to improve learning stability (for the same reason the actor networks are also updated less frequently than the critic networks). Furthermore, a key feature of TD3 is that it maintains two independent critic networks with updates being based on the lower estimate of the value function (which further reduces errors and improves stability). Finally, TD3 also applies 'target function smoothing' by adding clipped noise to the target actions, this improves stability during training by preventing inaccurately high value estimations. TD3 therefore requires six networks (μ, μ_{target}, $Q1$, $Q1_{\text{target}}$, $Q2$, $Q2_{\text{target}}$) each of which uses the structure described below. The network design follows that used by [48], where the number of filter layers is doubled each time the feature map size is halved [49, 50]. The 256 × 320 binary image passed through three convolutional layers which, respectively, produced layer outputs with dimensions 64 × 80 × 32, 32 × 40 × 64 and 32 × 40 × 64, each with batch normalization and rectified linear unit

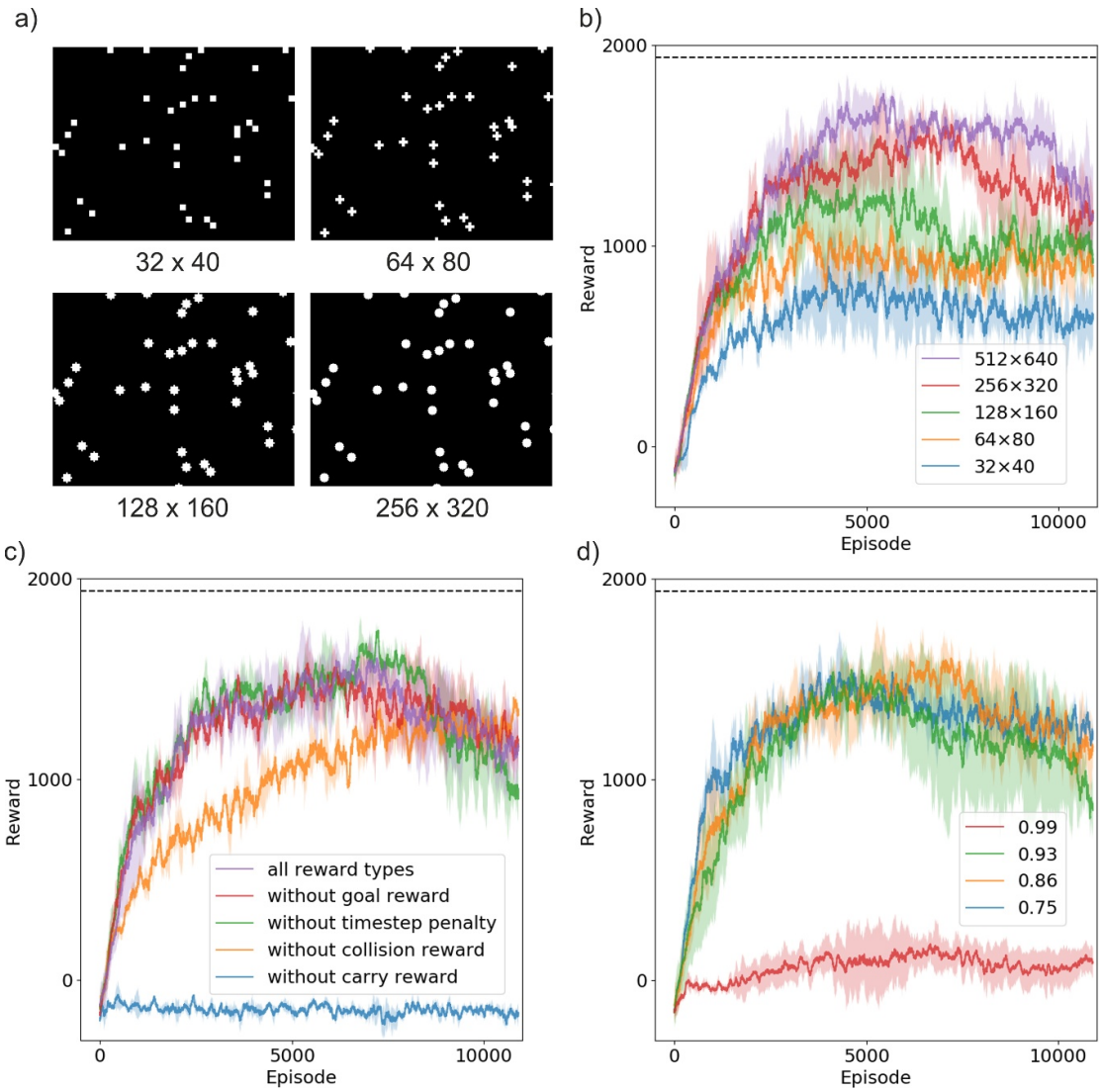

Figure 4. Analysis of training in the virtual environment. (a) Shows examples of binary images rendered at different resolutions. (b)–(d) Show the effect on reward of different: (b) image resolutions, (c) reward combinations, (d) discount factors. In (b)–(d) the horizontal dashed line shows the theoretical maximum reward, solid curves and shaded areas show the mean and standard deviation of a 100 episode, rolling-average of reward (evaluated over three training runs).

activation functions. (ReLU is still the de facto standard activation function for neural networks due to its simplicity and performance across a range of architectures, despite innovations such as trainable activation functions [51]). The convolutional layers are followed by three fully connected layers of size 512, 64 (+ numerical inputs), and 64, before producing an output vector of size 2 corresponding to the intended velocities for the X and Y axes. In the case of the actor, the vector describing the direction towards the target was appended at the second fully connected layer, in the case of the critic, both the target vector and the action were appended (since action is also needed in order to learn the state-action value function). Training was conducted in an episodic manner with a maximum episode length of 1000 time steps. At the start of each episode, the target was positioned at a randomly chosen angle, 1000 pixels away from the trap. During the rollout of an episode, image and numerical states were stored in a first-in, first-out memory buffer with up to 1 050 000 quadruples [52]. Each time an episode reached a terminal state (collision, success, or time limit) the update procedure commenced, where a batch of 32 state transitions was sampled randomly (with a uniform distribution) from the memory buffer and network parameters were updated accordingly. Analysis of the neural network training is shown in figure 4, where the effects of image resolution, reward shape and discount factor are explored.

Figure 4(a) shows the effect of resolution on the appearance of microspheres when rendered as a binary image, where increasing the resolution reduces the pixelization of the microspheres. (Note that the images are scaled so that the field of view remains constant.) Figure 4(b) shows the 100 episode, rolling average of reward as a function of episode number, for binary image inputs with different resolutions. (Three training

trials were performed for each case and so the solid curves in figures 4(b) and (d) represent the mean reward whilst the shaded areas show the standard deviation of reward.) Whilst higher resolution produced a higher reward, computational limitations resulted in a maximum practical resolution of 256×320, which was the resolution used throughout this work. Such resolution is comparable to other demonstrations of state observations in the literature, such as $84 \times 84 \times 4$ [27], $64 \times 64 \times 9$ [53], and $60 \times 108 \times 3$ [29], where the third dimension arises from the use of separate RGB channels and from stacking of images from multiple timesteps. Additional experiments were conducted (not shown in figure 4) where the binary images from the microscope were cropped rather than scaled to reduce their pixel dimensions. With cropped images, the neural network was still able to learn the task, although it achieved slightly lower rewards. The reduced field of view meant that the neural network could only take avoiding action when it was in close proximity to an obstacle. Due to the random motion of the non-trapped particles, as the field of view was reduced, this resulted in an increased number of collisions. It was therefore decided to always use the maximum field of view permitted by the imaging system, allowing the neural network to be more pre-emptive in its avoidance actions.

The set of rewards that are provided to the neural network during training is critical in encouraging the preferred behavior, as a poor choice of reward structure can leave loopholes that can be exploited, leading to unexpected and undesirable behavior patterns that nonetheless yield high rewards [54] (see examples below). In this work, the rewards were (a) a universal negative reward of -1 for each timestep, (b) a carry reward equal to the number of pixels moved towards the target (positive) or away from it (negative), (c) a collision reward of -100 for hitting another microsphere, and (d) a goal reward of $+1000$ for reaching the target location. (The values of these rewards were selected based on intuition and have not been optimized—systematic adjustment of these as hyperparameters could undoubtedly improve performance but would exceed our current computation capacity). The universal negative reward was intended to exert time pressure on the neural network. The carry reward was intended to encourage movement towards the target location. The collision reward punished the neural network for collisions with the randomly moving microspheres, which also terminated the episode. The goal reward was intended to reinforce the behavior of carrying a microsphere in the laser tweezers trap to the target location.

Note: Examples of possible loopholes due to poor choice of reward structure: E.g. 1: If the negative carry reward for moving away from the target is omitted the neural network could exploit this by simply oscillating one step towards and one step away from the target, increasing its total reward at every other step and yielding a high score without reaching the target. E.g. 2: If the carry reward was proportional to the velocity of particle motion (positive for towards the target, negative for away) this could be exploited by moving at high velocity along a gradually descending spiral around the target location (this would yield a higher total reward than following a more direct path).

To quantify the effectiveness of each of the reward components, training runs were carried out in which each of the components was eliminated one at a time, as shown in figure 4(c). When the carry reward was eliminated the network failed to learn the task. This is attributed to the fact that the carry reward provides feedback to the network at every time step to encourage movement towards the target. The second largest effect was observed when the collision reward was eliminated, which had the effect of reducing the learning rate. Whilst removal of the universal timestep negative reward and the goal reward appear to have only a minimal effect on performance, the highest performance was shown to occur when all four reward types were active.

The discount factor $\gamma \in (0, 1)$ and the concept of discounted rewards R_t are used to set the relative weighting of immediate rewards compared to those that may be received at a future time [47]. This is illustrated in the equation below where $r(s_t, a_t)$ is the reward received at timestep t whilst in state s_t and taking action a_t and where T is the episode length. If $\gamma = 1$ then all future rewards are of equal weight, as γ is reduced the weighting of distant future rewards is greatly reduced:

$$R_t = r(s_t, a_t) + \gamma r(s_{t+1}, a_{t+1}) + \gamma^2 r(s_{t+2}, a_{t+2}) + \ldots = \sum_{i=t}^{T} \gamma^{i-t} r(s_i, a_i)\,.$$

The effectiveness of the terminal goal and collision rewards used here relies heavily upon the use of discounted rewards. For example, the direction of the last step that takes the microsphere into the target zone may be relatively unimportant, however, the continuous progress towards the target, over the previous hundred or so steps, is critical. The numerical value of the discount factor effectively determines the number of steps into the future that the network attempts to predict future rewards (its time horizon). Figure 4(d) shows a comparison of the training performance for different discount factors, in the case where all reward components are included. The often-used value of 0.99 [47] was found to be inappropriate for this task and achieved very poor performance; this is attributed to the fact that higher discount factors require prediction

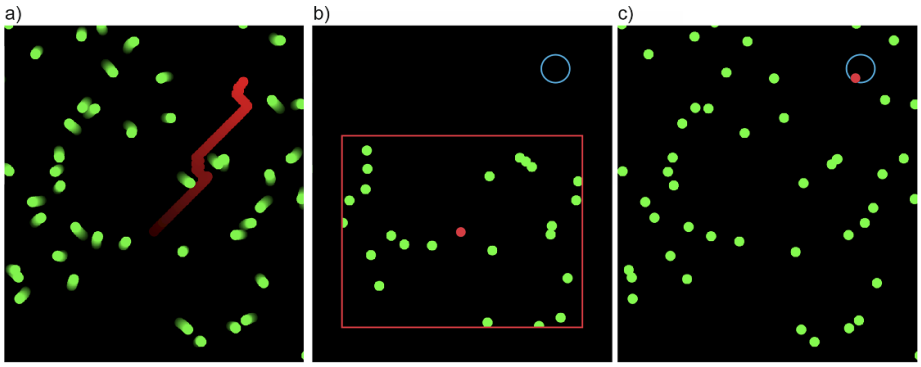

Figure 5. Demonstration of the virtual gym environment for dynamic microspheres, showing (a) the chosen trajectory and movement of microspheres during the episode (b) the field of view and target position at the start of the episode and (c) the environment state at the end of the episode.

of rewards over a longer horizon, where a large fraction of that trajectory may be outside of the limited field of view of the microscope. Consequently, the variance of these predictions is greater, which slows the rate of convergence and in severe cases causes the agent to completely fail to learn the task. As shown in figure 4(d) as the value of discount factor was reduced, the rate of convergence increased, with the highest average reward being achieved with a discount factor of 0.86. In all cases, the reward reaches a peak at around 6000 epochs, after which the performance declines. The precise cause of this effect is difficult to determine, however, it may represent an instance of catastrophic forgetting [55] for which one contributing factor can be the finite buffer size. However, as the models were saved periodically during training, the model that yielded the highest reward was applied to the physical environment.

5. Demonstration in the virtual environment

Once trained, the neural network was evaluated in the gym environment for a range of starting conditions. An example of a single trajectory is shown in figure 5, where the microsphere of interest and its associated trajectory is red, other microspheres are green circles, and the target position is a blue ring. Figure 5(a) shows the trajectory of the trapped microsphere, along with the movement of the other microspheres. Figure 5(b) shows the extent of the field of view (which does not include the entire path to the target position) and the starting positions of the free microspheres within that field of view. Figure 5(c) shows the environment state at the end of the episode. As observed, the neural network is clearly capable of avoiding collisions whilst continuing to progress towards the target position, as the field of view is updated.

To provide a more general demonstration of the capability of the neural network inside the gym, figure 6 shows 12 trajectories, each under identical starting conditions but with different target positions. Here, the simulated Brownian motion of the free microspheres was kept the same for each of the 12 trials, and therefore all trajectories can be overlaid on a single figure. Where the red and green paths appear to cross in the figure, this corresponds to a different time step position rather than a collision. Figure 6(a) shows the trajectory paths for all microspheres, figure 6(b) shows the initial environment state, with associated field of view and target positions, and figure 6(c) presents the environment state after completion of all trajectories. Although the action space of the environment was continuous, the neural network was observed to frequently output actions corresponding to the combination of maximum positive or negative stage velocities for both axes. This is attributed to the combination of the time based reward (which encourages maximum velocity in many cases) and the use of the tanh hyperbolic function as the network activation function (which can result in saturation of actions towards their upper and lower bounds).

6. Demonstration in the physical environment

Once the neural network was trained, it was applied to the physical environment where, importantly, no further training took place. An example trajectory is presented in figure 7, which shows the field of view (red rectangle) at time steps of (a) 0, (b) 74, and (c) 146, corresponding to the initial, middle and end states. For visual clarity, the figures also show the environment state outside the field of view for positions already explored by the neural network (although the network has no mechanism to retain this information). As before, for each time step, the neural network was provided only with a binary image corresponding to the field of view, along with a vector denoting the direction towards the target position. The trajectory in the

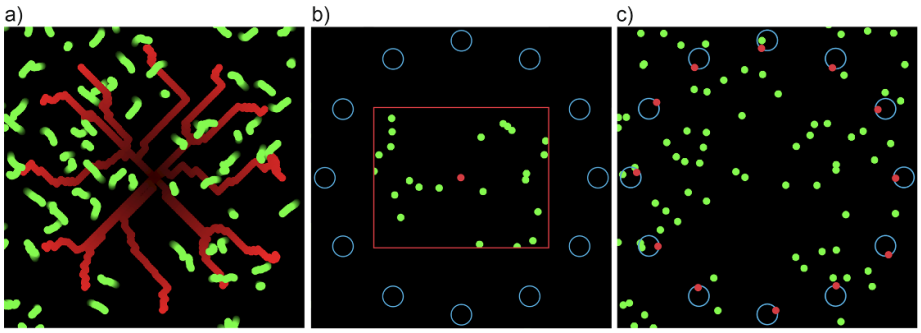

Figure 6. The virtual gym environment showing trajectories to 12 target locations, subject to the same free microsphere motions. (a) The trajectories chosen by the neural network (red) and the movements of the other microspheres (green). (b) The start of all episodes, showing microsphere initial positions, target positions and the field of view permitted to the neural network. (c) The entire environment at the end of the episodes.

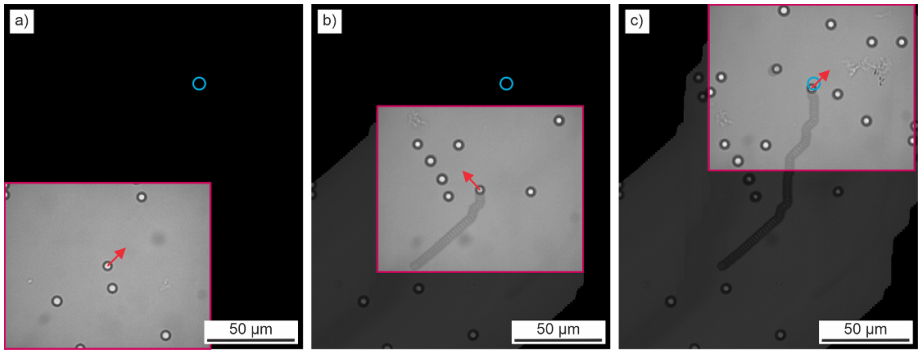

Figure 7. Demonstration of the neural network in the physical environment, showing the field of view and trajectory path up to that point, at time steps of (a) 0, (b) 41, and (c) 81.

physical environment resulted in the transportation of the microsphere to the target position with only slight deviations from the ideal path, as shown by the small degree of movement in the left-direction in the time steps before reaching the target. The movement direction selected by the neural network (the velocity action) at each of the featured timesteps is shown by the red arrow.

As the X and Y movement stages were independently controlled, the fastest microsphere movement corresponded to the hypotenuse where both X and Y stages were at their maximum speeds. As such, for a vertical movement (i.e. Y axis only), the X movement velocity value had no effect on the vertical component of the speed of travel. Therefore, for a desired vertical movement, the neural network was able to move continuously in the Y axis at the maximum speed, whilst alternating between maximum positive and negative movement directions in the X axis (a technique familiar to many players of older computer games). This effect is observed in figure 7, where diagonal movements (i.e. a combination of X and Y stage movements) are shown by the red arrows. Although the neural network was found to use such diagonal movements extremely frequently, it is important to realize that the network could choose any possible value for each of the two stages (i.e. the action space is continuous) and a distribution of actions was observed, as opposed to discrete diagonal actions. Whilst the cause of this movement strategy is challenging to identify due to the nature of obfuscation in a neural network, we attribute this effect to the fact that this movement technique can increase the field of view, whilst having no negative effect on the movement speed.

7. Augmented physical environment

Training a neural network in a virtual environment for application in a physical environment is only possible if the neural network treats both environments equally. This process therefore has the interesting consequence that a combination of a virtual and a physical environment is also treated equally. This prospect is explored in this section where the capability to overlay a virtual environment onto the physical environment is demonstrated. Such a technique has the potential to unlock capabilities associated with mixed and augmented reality, such as adding safety limits for machine motion or using observation inputs from additional sensing techniques.

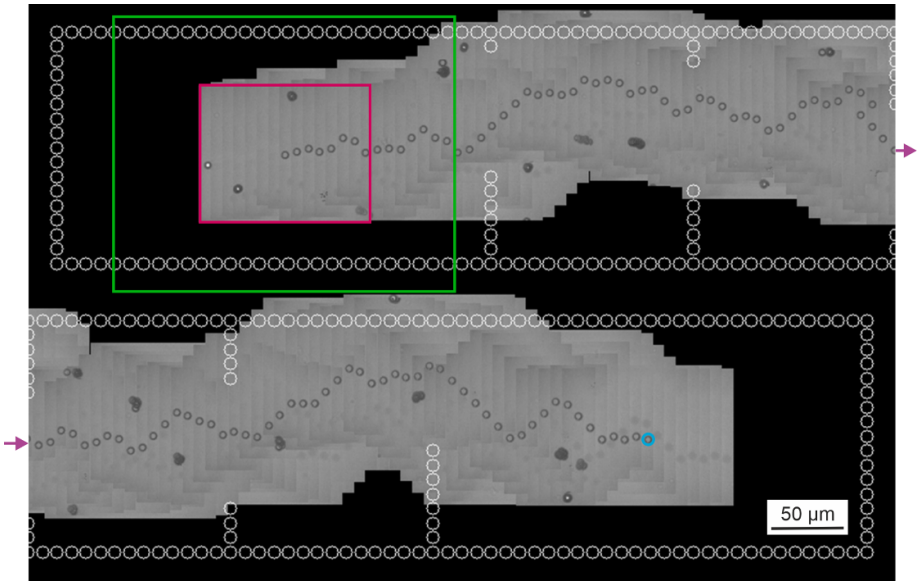

Figure 8. Demonstration of neural network in the augmented physical environment, showing the virtual obstacles (white circles), the physical free-moving microspheres and the physical laser-trapped microsphere. The virtual obstacles are presented in the form of hollow circles in the figure for visual clarity. The solid rectangles represent the field of view of the physical microscope and the expanded field of view possible for the virtual environment.

Figure 8 shows a trajectory of a trapped microsphere controlled by the neural network in such an augmented physical environment, where there are both free-moving microspheres (that are physically present) and gate-shaped obstacles (that are virtually overlaid). (Note that stage velocity is higher than was used in figure 7 and so the distance moved per timestep is larger.) The trajectory, which is displayed as a series of overlapping experimentally measured camera images, shows the successful movement of the laser-trapped microsphere to the opposite side of the environment. This is a clear demonstration that the neural network has learnt the general principles of obstacle avoidance and motion planning, as the neural network was trained on virtual free-moving microspheres but is shown here to be able to apply this learnt capability to a maze-like path formed of virtual microspheres. The virtual obstacles were rendered in the form of solid circles, equivalent to the appearance of the physical microspheres, to be treated equivalently to the free-moving physical microspheres. For visual clarity, the figure shows these virtual microspheres as white circles whilst the target position is show as a blue circle. The initial field of view of the microscope (the physical component of the augmented environment) is highlighted by the pink rectangle. In the augmented environment it is possible for the virtual components to be rendered over a wider field of view; this is the case here, with the augmented field of view illustrated by the green rectangle. The purple arrow shows the position where the environment has been split in two, for the purpose of displaying over two lines in the figure.

8. Conclusions

Reinforcement learning offers the capability for a neural network to find a solution to a problem through self-exploration of the environment. However, in many cases, it is beneficial to train the neural network in a virtual environment, so that this exploration can occur faster and without the possibility of damage to the physical apparatus. However, this leads to the challenge, programmatically, of designing a sufficiently accurate simulation of the physical world. Here, an optical tweezers environment is designed, and a neural network is trained, with the objective of moving a microsphere to the desired position using the optical tweezers effect, whilst simultaneously avoiding collisions with other microspheres. The neural network, which was provided with limited field of view camera images and a vector indicating the direction towards the target, was shown to be able to achieve continuous control of the X, Y stage velocities and ensure that the laser-trapped microsphere reached the target destination without collision. The results presented demonstrate the effective transference of capability learned in the virtual environment to the physical environment and, furthermore, highlight the potential for applications in augmented environments, where a virtual environment was overlaid upon the physical one.

See the supplementary data for a study of the performance of the trained neural network in the case of barriers with concave geometry. It is concluded that to tackle more complex optical tweezers tasks (such as those that require maze navigation) it would be necessary to alter the neural network architecture. For

example, an RNN or LSTM type structure [56] could introduce memory of past states and actions which could dramatically improve performance on this type of task (although at the expense of additional computation). Where a variety of tasks must be undertaken, it has been proposed (using robot swarms as an example) that automatic off-line design [57] could be used to tailor neural network structures and hyperparameters to specific tasks on a task-by-task basis (without human intervention). Such a system, if it could be realized, could greatly extend the range of capabilities for our neural network controlled optical tweezers, or in fact, of any equipment controlled via neural networks.

The concept of training neural networks within virtual environments clearly has significant potential in the application of machine learning for experimental optimization and control. Fully unlocking this potential will require widespread application of these transformative techniques across a broad range of disciplines—a trend that is now entering the field of optics and photonics.

Data availability statement

All data that support the findings of this study are included within the article (and any supplementary files).

Acknowledgments

We gratefully acknowledge the support of NVIDIA Corporation with the donation of the Titan X GPU used for this research. The authors acknowledge the use of the IRIDIS High Performance Computing Facility, and associated support services at the University of Southampton, in the completion of this work.

Funding

Engineering and Physical Sciences Research Council (EP/N03368X/1).

Conflict of interest

The authors declare no conflicts of interest.

ORCID iDs

Matthew Praeger https://orcid.org/0000-0002-5814-6155
Yunhui Xie https://orcid.org/0000-0002-8841-7235
James A Grant-Jacob https://orcid.org/0000-0002-4270-4247
Robert W Eason https://orcid.org/0000-0001-9704-2204
Ben Mills https://orcid.org/0000-0002-1784-1012

References

[1] Rivenson Y, Göröcs Z, Günaydin H, Zhang Y, Wang H and Ozcan A 2017 Deep learning microscopy *Optica* **4** 1437–43
[2] Grant-Jacob J A, Mackay B S, Baker J A G, Xie Y, Heath D J, Loxham M, Eason R W and Mills B 2019 A neural lens for super-resolution biological imaging *J. Phys. Commun.* **3** 065004
[3] Matsumoto T *et al* 2019 Deep-UV excitation fluorescence microscopy for detection of lymph node metastasis using deep neural network *Sci. Rep.* **9** 16912
[4] Rahmani B, Loterie D, Konstantinou G, Psaltis D and Moser C 2018 Multimode optical fiber transmission with a deep learning network *Light: Sci. Appl.* **7** 69
[5] Närhi M, Salmela L, Toivonen J, Billet C, Dudley J M and Genty G 2018 Machine learning analysis of extreme events in optical fibre modulation instability *Nat. Commun.* **9** 4923
[6] Tait A N, De Lima T F, Zhou E, Wu A X, Nahmias M A, Shastri B J and Prucnal P R 2017 Neuromorphic photonic networks using silicon photonic weight banks *Sci. Rep.* **7** 7430
[7] Tahersima M H, Kojima K, Koike-Akino T, Jha D, Wang B, Lin C and Parsons K 2019 Deep neural network inverse design of integrated photonic power splitters *Sci. Rep.* **9** 1368
[8] Grant-Jacob J A, Mackay B S, Baker J A G, Heath D J, Xie Y, Loxham M, Eason R W and Mills B 2018 Real-time particle pollution sensing using machine learning *Opt. Express* **26** 27237–46
[9] Paine S W and Fienup J R 2018 Machine learning for improved image-based wavefront sensing *Opt. Lett.* **43** 1235–8
[10] Mennel L, Symonowicz J, Wachter S, Polyushkin D K, Molina-Mendoza A J and Mueller T 2020 Ultrafast machine vision with 2D material neural network image sensors *Nature* **579** 62–6
[11] Heath D J, Grant-Jacob J A, Xie Y H, Mackay B S, Baker J A G, Eason R W and Mills B 2018 Machine learning for 3D simulated visualization of laser machining *Opt. Express* **26** 21574–84
[12] Mills B, Heath D J, Grant-Jacob J A and Eason R W 2018 Predictive capabilities for laser machining via a neural network *Opt. Express* **26** 17245–53
[13] Zhou J, Huang B, Yan Z and Bünzli J-C G 2019 Emerging role of machine learning in light–matter interaction *Light: Sci. Appl.* **8** 84

[14] Lenton I C D, Volpe G, Stilgoe A B, Nieminen T A and Rubinsztein-Dunlop H 2020 Machine learning reveals complex behaviours in optically trapped particles *Mach. Learn.: Sci. Technol.* **1** 045009
[15] Henderson P, Islam R, Bachman P, Pineau J, Precup D and Meger D 2017 Deep reinforcement learning that matters (arXiv:1709.06560)
[16] Li Y 2017 Deep reinforcement learning: an overview (arXiv:1701.07274)
[17] Sutton R S and Barto A G 2018 *Reinforcement Learning: An Introduction* (Cambridge, MA: MIT Press)
[18] Muiños-Landin S, Ghazi-Zahedi K and Cichos F 2018 Reinforcement learning of artificial microswimmers (arXiv:1803.06425)
[19] Watkins C J C H 1989 Learning from delayed rewards (Cambridge, UK: University of Cambridge)
[20] Brockman G, Cheung V, Pettersson L, Schneider J, Schulman J, Tang J and Zaremba W 2016 OpenAI gym (arXiv:1606.01540)
[21] James S, Wohlhart P, Kalakrishnan M, Kalashnikov D, Irpan A, Ibarz J, Levine S, Hadsell R and Bousmalis K 2019 Sim-to-real via sim-to-sim: Data-efficient robotic grasping via randomized-to-canonical adaptation networks (arXiv:1812.07252)
[22] Andrychowicz O M *et al* 2019 Learning dexterous in-hand manipulation *Int. J. Rob. Res.* **39** 3–20
[23] Akkaya I, Andrychowicz M, Chociej M, Litwin M, McGrew B, Petron A, Paino A, Plappert M, Powell G and Ribas R 2019 Solving Rubik's cube with a robot hand (arXiv:1910.07113)
[24] Sadeghi F and Levine S 2016 CAD2RL: real single-image flight without a single real image (arXiv:1611.04201)
[25] Sampedro C, Bavle H, Rodriguez-Ramos A, De La Puente P and Campoy P 2018 Laser-based reactive navigation for multirotor aerial robots using deep reinforcement learning *2018 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS) (Madrid, Spain, 1–5 October 2018)* (IEEE) pp 1024–31
[26] You Y, Pan X, Wang Z and Lu C 2017 Virtual to real reinforcement learning for autonomous driving (arXiv:1704.03952)
[27] Mnih V, Kavukcuoglu K, Silver D, Graves A, Antonoglou I, Wierstra D and Riedmiller M 2013 Playing Atari with deep reinforcement learning (arXiv:1312.5602)
[28] Bhatti S, Desmaison A, Miksik O, Nardelli N, Siddharth N and Torr P H 2016 Playing doom with slam-augmented deep reinforcement learning (arXiv:1612.00380)
[29] Lample G and Chaplot D S 2016 Playing FPS games with deep reinforcement learning (arXiv:1609.05521)
[30] Vinyals O, Ewalds T, Bartunov S, Georgiev P, Vezhnevets A S, Yeo M, Makhzani A, Küttler H, Agapiou J and Schrittwieser J 2017 StarCraft II: a new challenge for reinforcement learning (arXiv:1708.04782)
[31] Vinyals O *et al* 2019 Grandmaster level in StarCraft II using multi-agent reinforcement learning *Nature* **575** 350–4
[32] Jaderberg M *et al* 2019 Human-level performance in 3D multiplayer games with population-based reinforcement learning *Science* **364** 859
[33] Narasimhan K, Kulkarni T and Barzilay R 2015 Language understanding for text-based games using deep reinforcement learning (arXiv:1506.08941)
[34] Ammanabrolu P and Riedl M O 2018 Playing text-adventure games with graph-based deep reinforcement learning (arXiv:1812.01628)
[35] Kanagawa Y and Kaneko T 2019 Rogue-Gym: a new challenge for generalization in reinforcement learning *2019 IEEE Conf. on Games (CoG)* (*London, UK, 20–23 August 2019*) (IEEE) (https://doi.org/10.1109/CIG.2019.8848075)
[36] Silver D *et al* 2016 Mastering the game of Go with deep neural networks and tree search *Nature* **529** 484–9
[37] Silver D *et al* 2018 A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play *Science* **362** 1140
[38] Schrittwieser J, Antonoglou I, Hubert T, Simonyan K, Sifre L, Schmitt S, Guez A, Lockhart E, Hassabis D and Graepel T 2019 Mastering atari, go, chess and shogi by planning with a learned model (arXiv:1911.08265)
[39] Günther J, Pilarski P M, Helfrich G, Shen H and Diepold K 2016 Intelligent laser welding through representation, prediction, and control learning: an architecture with deep neural networks and reinforcement learning *Mechatronics* **34** 1–11
[40] Sun C, Kaiser E, Brunton S L and Kutz J N 2020 Deep reinforcement learning for optical systems: a case study of mode-locked lasers *Mach. Learn.: Sci. Technol.* **1** 045013
[41] Sorokin D, Ulanov A, Sazhina E and Lvovsky A 2020 Interferobot: aligning an optical interferometer by a reinforcement learning agent (arXiv:2006.02252)
[42] Ke H, Xu B, Xu Z, Wen L, Yang P, Wang S and Dong L 2019 Self-learning control for wavefront sensorless adaptive optics system through deep reinforcement learning *Optik* **178** 785–93
[43] Suarez-Varela J, Mestres A, Yu J, Kuang L, Feng H, Cabellos-Aparicio A and Barlet-Ros P 2019 Routing in optical transport networks with deep reinforcement learning *IEEE/OSA J. Opt. Commun. Netw.* **11** 547–58
[44] Tünnermann H and Shirakawa A 2019 Deep reinforcement learning for coherent beam combining applications *Opt. Express* **27** 24223–30
[45] Grier D G 2003 A revolution in optical manipulation *Nature* **424** 810–6
[46] Suzuki S and Abe K 1985 Topological structural analysis of digitized binary images by border following *Comput. Vision Graph. Image Process.* **30** 32–46
[47] Fujimoto S, Van Hoof H and Meger D 2018 Addressing function approximation error in actor-critic methods (arXiv:1802.09477)
[48] Mnih V *et al* 2015 Human-level control through deep reinforcement learning *Nature* **518** 529–33
[49] He K, Zhang X, Ren S and Sun J 2015 Deep residual learning for image recognition (arXiv:1512.03385)
[50] Simonyan K and Zisserman A 2014 Very deep convolutional networks for large-scale image recognition (arXiv:1409.1556)
[51] Nwankpa C, Ijomah W, Gachagan A and Marshall S 2018 Activation functions: comparison of trends in practice and research for deep learning (arXiv:1811.03378)
[52] Lin L J 1992 Self-improving reactive agents based on reinforcement learning, planning and teaching *Mach. Learn.* **8** 293–321
[53] Lillicrap T P, Hunt J J, Pritzel A, Heess N, Erez T, Tassa Y, Silver D and Wierstra D 2015 Continuous control with deep reinforcement learning (arXiv:1509.02971)
[54] Amodei D, Olah C, Steinhardt J, Christiano P, Schulman J and Mané D 2016 Concrete problems in AI safety (arXiv:1606.06565)
[55] Kirkpatrick J *et al* 2017 Overcoming catastrophic forgetting in neural networks *Proc. Natl Acad. Sci.* **114** 3521–6
[56] Sherstinsky A 2020 Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network *Physica D* **404** 132306
[57] Birattari M *et al* 2019 Automatic off-line design of robot swarms: a manifesto *Front. Robot. AI* **6** 59