Model-free Reinforcement Learning with a Non-linear Reconstructor for closed-loop Adaptive Optics control with a pyramid wavefront sensor

B. Pou[a, b], J. Smith[c], E. Quinones[a], M. Martin[b], and D. Gratadour[d]

[a]Barcelona Supercomputing Center (BSC), C. Jordi Girona, 29, 08034, Barcelona, Spain
[b]Computer Science Department, Universitat Politècnica de Catalunya (UPC), C. Jordi Girona, 31, 08034, Barcelona, Spain
[c]School of Computing, Australian National University, Canberra, Australia
[d]LESIA, Observatoire de Paris, Universite PSL, CNRS, Sorbonne Universite, Univ. Paris Diderot, Sorbonne Paris Cite, 5 place Jules Janssen, 92195 Meudon, France

ABSTRACT

We present a model-free reinforcement learning (RL) predictive model with a supervised learning non-linear reconstructor for adaptive optics (AO) control with a pyramid wavefront sensor (P-WFS). First, we analyse the additional problems of training an RL control method with a P-WFS compared to the Shack-Hartmann WFS. From those observations, we propose our solution: a combination of model-free RL for prediction with a non-linear reconstructor based on neural networks with a U-net architecture. We test the proposed method in simulation of closed-loop AO for an 8m telescope equipped with a 32x32 P-WFS and observe that both the predictive and non-linear reconstruction add additional benefits over an optimised integrator.

Keywords: Reinforcement Learning, AO Control, Machine Learning, Pyramid Wavefront Sensor

1. INTRODUCTION

In recent years, the interest in adaptive optics (AO) control with machine learning (ML) methods has risen exponentially. On the one hand, there are methods built around supervised learning (SL). Among these methods, we can differentiate between non-linear reconstructors,[1–4] predictive models[5,6] or a combination of both.[7,8] In the case of predictive models, while their predictions are usually correct in open-loop, their performance dramatically decreases in closed-loop because there is a distribution mismatch between the training set and operational data. A possible solution to this problem is to introduce a regularisation so that the output of the prediction is within the bounds of the training data distribution.[8] On the other hand, a few predictive models based on reinforcement learning (RL) have been developed.[9–15] In this work, we focus on closed-loop RL methods with a wavefront sensor (WFS), in which the goal is to learn the control policy, a function that maps information of the system to commands, during operation.[11–15] In this case, there is no distribution mismatch between training and operational data as they are the same. Moreover, in contrast to most current model-based predictive controllers, there are no assumptions, and the control dynamics are learnt purely from data.

RL for AO has initially been assessed with a focus on WFS-less methods.[9,10] The current RL methods considering a WFS are mainly divided into model-based[12,15] and model-free.[11,13,14] Model-based RL methods involve building a model (usually with neural networks) of the system's dynamics and use it either to find an optimal action or to train a control policy. In its initial development, random sampled commands were propagated through a model of the system dynamics built purely on data and the ones that gave overall best performance were selected. This was shown to outperform an integrator controller. However, the large corresponding inference time limited its applicability. Recently, to reduce this inference time, the model was made to behave as a policy simultaneously[15] (i.e. directly giving as output the commands that already maximise the performance metric

Further author information: (Send correspondence to B. Pou)
B. Pou: E-mail: bartomeu.poumulet@bsc.es

selected). In the model-free counterpart, Landman et al.[11,13] used convolutional neural networks in the policy to predict zonal commands for correction. Instead, previous work from our team[14] exploited the orthogonal property of a global modal basis to factorise the problem into subproblems. This changed the problem from a single-agent RL solution to a multi-agent RL (MARL) one. Moreover, we showed that the RL controller mostly corrected temporal error via an error budget analysis.

Among the traditional model-based controllers, Linear Quadratic Gaussian (LQG) variations are considered *state-of-the-art* for AO. Initially introduced in 1993,[16] a model of the system dynamics based on its physical properties is built, which is then used to select the appropriate commands to compensate for the temporal error. This process usually takes into account many assumptions, such as the consideration of Kolmogorov turbulence, which we intend to avoid, as during an on-sky observation, this assumption might not hold. The MARL solution was compared against an implementation of LQG, showing similar performance levels, which indicates that RL-based controllers are on par with the traditional model-based *state-of-the-art* solution.

The pyramid WFS (P-WFS)[17] delivers promises for extreme AO, as the reconstruction accuracy is outstanding. This is because the P-WFS has higher sensitivity compared to the long-established Shack-Hartmann WFS (SH-WFS), allowing for a higher degree of phase correction. However, several phenomena make creating a controller for this sensor technology challenging. The most well-known phenomenon is optical gain (OG), a change in sensitivity due to residual aberrations. The CLOSE algorithm[18] provides a strategy to adjust modal gains in real-time to mitigate the impact of varying OG. Most of the RL works have focused on a SH-WFS. However, Nouisiainen et al.[15] recently presented a model-based RL method for controlling an AO system equipped with a P-WFS. In the discussion section, we argue on the differences between their approach and the one presented in this paper.

The rest of the paper continues as follows: in Section 2, we discuss how the P-WFS complexifies the training of an RL method compared to the SH-WFS. Section 3 provides a possible solution to this issue. Section 4 shows different experiments to see if we can outperform an optimised integrator. Finally, in Section 5, we discuss our results and limitations and provide possible future work.

2. DIFFICULTIES IN TRAINING AN RL CONTROLLER IN A P-WFS

In this section, we present the problems that the P-WFS pose for training an RL controller. Usually, the relationship between the phase and the WFS image is considered linear. Equation 1 shows the linear relationship described by an interaction matrix, D, between a measurement vector from the WFS, m, and a reconstruction vector of the phase projected into the deformable mirror (DM), c. D is usually obtained by pushing modes with a minimal value, seeing its effect in the WFS, and just establishing a linear relationship.

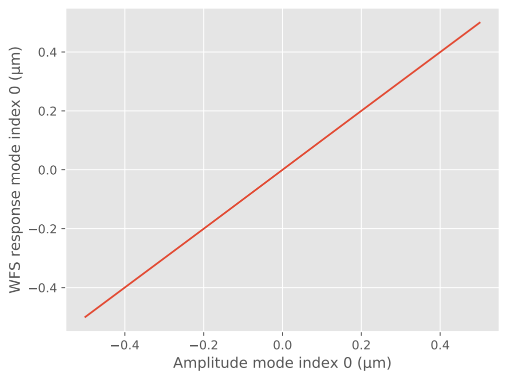
$$m = Dc. \tag{1}$$

Then given a measurement, the reconstruction can be obtained with the generalised inverse of the interaction matrix, the command matrix, $D^{\dagger}$:

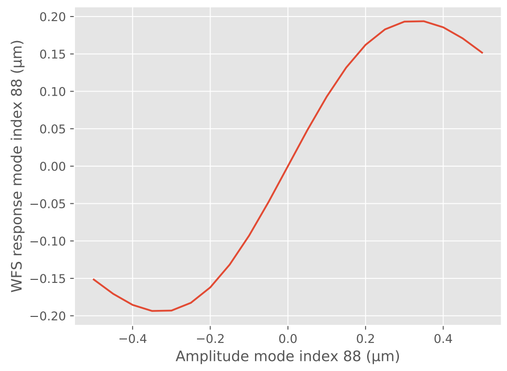
$$c = D^{\dagger}m. \tag{2}$$

For the SH-WFS, this equation holds most of the time. In contrast, the P-WFS has a non-linear behaviour, and the linear reconstruction may only be valid in a specific range. In the mentioned works of closed-loop RL control, the reward function, the performance measure which the controller uses to improve itself, and part of the state, information about the system, is derived from c. We study the linear reconstruction for both the SH-WFS and P-WFS with aberrations on both a global and zonal basis. We obtain those reconstructions using COMPASS.[19] For the SH-WFS, the simulation consists of an 8m telescope with a 40x40 SH-WFS, and the P-WFS has the same simulation parameters as the one used for the experiments (see Table 1). For the global basis, we use B_{tt} modes.[20] Inspired by the Zernike polynomials, the B_{tt} modes are desired due to their properties of pure orthogonality and that the piston is filtered out. For the zonal basis, we use the influence functions of the DM.

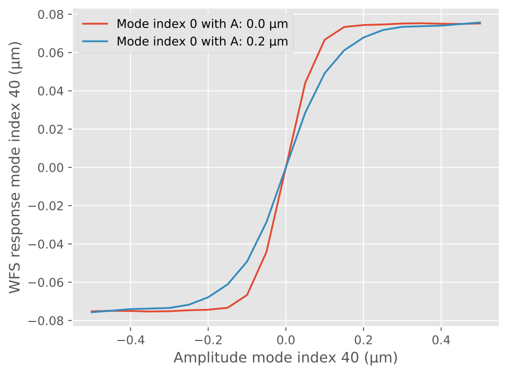
Figure 1a and Figure 1b show the reconstruction of a phase formed only by a single mode of the zonal basis from measurements of the SH-WFS and the P-WFS, respectively. Figure 1c and Figure 1d show the reconstruction of the phase formed by B_{tt} modes 40 and 200, respectively, from measurements of the P-WFS. We evaluate the P-WFS reconstructions of global modes with different values of B_{tt} mode index 0. From these figures, we can observe a set of problems which will appear when training an RL controller in an AO system equipped with a P-WFS:



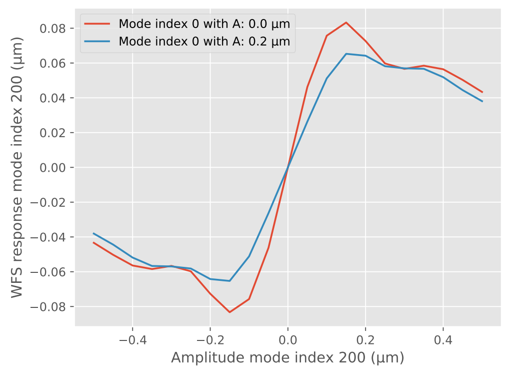
(a) Reconstruction of zonal mode 0 with a SH-WFS.



(b) Reconstruction of zonal mode 88 with a P-WFS.



(c) Reconstruction of global mode 40 a P-WFS.



(d) Reconstruction of global mode 200 with a P-WFS.

Figure 1: SH-WFS and P-WFS response to various modes. We selected the modes because they show the problematic behaviour. Not all modes show the non-monotonicity/saturation problem for the range of amplitudes we tested. For the P-WFS and global modes, we also tested the reconstruction with a presence of an aberration of mode 0 with a value of either 0 or 0.2 μm.

1. The SH-WFS response is purely linear. Therefore there should not be any problem with using the reconstruction as part of the state or reward function.
2. The P-WFS response is non-linear. If the reconstruction is monotonic, this can be an advantage for RL methods which use non-linear function approximations as neural networks.
3. Figures c) and d) show the OG problem for the P-WFS. The sensitivity of the P-WFS to mode 40 and mode 200 changes when the aberration has other residual error (in this case, an additional residual on mode 0). Therefore, the modes are not orthogonal, and the MARL approach that exploits the orthogonality between modes method will not work as it is.
4. There is a possibility of saturation (Figure 1c) or even a decrease in response (Figure 1b and Figure 1d) when the amplitude of the input mode increases. The RL methods initially use trial and error, which might lead to zones on the edges of the figures. With this, the RL approach might end with a sub-optimal solution that does not correspond to the real minimised residual. This issue will be more pronounced when the

atmospheric conditions worsen (e.g. lower r_0), or the telescope uses a shorter wavelength for observation, as the probability of ending in the non-monotonic zones is higher.

In this work, we avoid problem 3 by not focusing on methods that assume independence between modes (i.e. the MARL approach). For problem 4, we leverage a non-linear reconstruction (similar to Landman et al.,[2] and Smith et al.[4]) to mitigate the non-monotonicity.

Finally, as an illustration, we show the evolution of training indicators for our RL method in the next section (see Section 3) without the non-linear reconstruction on an experiment with the parameters of Table 1 and an r_0 of 0.16 m at 500 nm. During training, every 2000 frames, we stop the simulation, reset the atmosphere (with another seed) and the DM shape, and perform an evaluation of 1000 frames where we record the final LE SR and the average reward obtained. In Figure 2 in red, we show the difference between LE SR obtained with an RL controller and an integrator over the evaluations done over time. In blue, we show the difference in average reward for the RL controller and the integrator for the same evaluations. As explained in Section 3, we do a correction on top of the integrator, hence the first evaluation is always similar in terms of performance to the integrator (both in reward and SR). After the RL controller starts to learn, we observe that the difference between integrator and RL both for average reward and LE SR initially follows a similar direction. However, around frame number 10^5 the SR difference starts to decrease while the difference in average reward per evaluation still increases. We hypothesise that this decoupling is because the RL agent ends up in the non-monotonic zones for some modes. In the following sections, we show that a non-linear reconstructor can help to avoid this decoupling between the reward and LE SR.

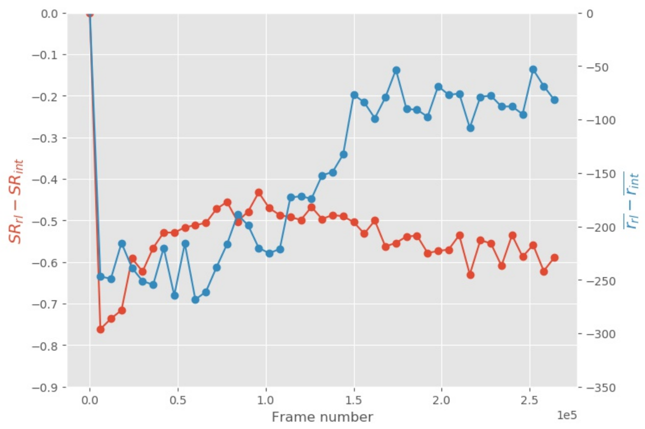


Figure 2: Decoupling of LE SR and reward. Red: evolution of difference in LE SR in the 1000-frame test between an RL controller and an integrator with a linear reconstruction. Blue: evolution of difference in average reward in the 1000-frame test between an RL controller and an integrator with a linear reconstruction.

3. METHODS

3.1 Supervised Learning Non-linear reconstructor

The task of the non-linear reconstructor is to reconstruct the phase from the WFS image. In this work, we propose to build a reconstructor based on the U-net,[21] *state-of-the-art* in many computer vision tasks. In our work, the U-net architecture receives as input the P-WFS image, x, and outputs the estimated phase, $\hat{y}$ .The P-WFS image is separated into channels, where each channel is a pupil image of the P-WFS. U-net is based on an autoencoder architecture with encoder and decoder parts. A sketch of the U-net model used for non-linear reconstruction is shown in Figure 3.

Given the network's weights, θ, the network's prediction of the phase, $\hat{y}$, and the real phase, y, the loss consists of the relative L_1 loss:

$$L(\theta) = \frac{||\hat{y} - y||_1}{||y||_1 + \epsilon}, \tag{3}$$

where ϵ is a small value to ensure that very small inputs do not have excessive significance in the loss. As reported in previous works,[2] the relative loss forces the model to consider similarly phases with a high root mean square (RMS) variance and phases with a low RMS. With a pure L_1 loss, the model would focus more on phases with a high RMS due to its overall magnitude.

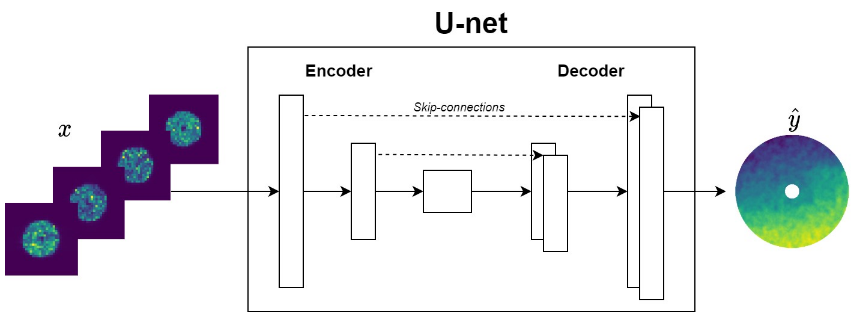


Figure 3: U-net for non-linear reconstruction. The U-net takes as input the four pupil images of the P-WFS and outputs the estimated phase.

Given a P-WFS image, x, the non-linear reconstruction, c^{nl}, is computed as the projection from the phase to any modal basis we choose as:

$$c^{nl}(x) = Pph2m \cdot (U\text{-}net(x)), \tag{4}$$

where $Pph2m$ is a matrix that projects the phase onto a modal basis.

3.2 Reinforcement learning predictive controller

RL treats control problems as an interaction between an agent and an environment. The agent executes actions, a, in the environment and receives states, s, and rewards, r, a measure of agent's performance. The agent has a policy, π, a function that maps states to actions which is improved based on data recollected interacting with the environment via trial and error. Consider the return at timestep t, R_t, the weighted sum of cumulative rewards:

$$R_t = \sum_{t'=t}^{T} \gamma^{t'-t} r_{t'+1}, \tag{5}$$

where $\gamma \in (0, 1]$ is a weighting factor giving more importance to initial or late events, and T is the number of steps in which the task ends. The goal for the agent is to learn optimal policy, π^*, the one (or set of ones) that maximises the expected return from the start of the task to the end.

$$\pi^* = arg \max_{\pi} \mathbb{E}_{\pi}[R_0] = arg \max_{\pi} \mathbb{E}_{\pi}[\sum_{t'=0}^{T} \gamma^{t'} r_{t'+1}], \tag{6}$$

where the expectation is over rewards obtained with a given policy. Usually, π is a parametric function with its parameters being updated when more data becomes available. In our case, we choose this parametric function to be a neural network due to their ability to build an implicit highly non-linear model from data.

In AO, the state of the environment corresponds to the instantaneous residual phase. Unfortunately, we only have an approximation in the form of a reconstruction of the phase from the WFS image. From earlier works, it has been shown that a combination of reconstructions of the phase and commands from current and previous timesteps works well as the state for the agent. The history of states helps the agent understand the atmosphere's dynamics and the delay in the AO system. At timestep, t, we define the state as:

$$s_t = (c_t^l + c_t^{nl}, C_{t-1}, C_{t-2}, C_{t-3}) \tag{7}$$

where c_t^l and c_t^{nl} are the linear and non-linear reconstructions of the phase from the WFS image, and C_t are the commands at timestep t. Our method uses an RL model-free actor-critic architecture. In the current AO literature, there is both a single agent architecture with convolutional layers on a zonal basis[13] and multi-agent architecture with dense layers on a global basis.[14] For this work, we have used the first choice as it allows faster training. We plan to compare both methods in future work. The actor-critic consists of two models represented by neural networks, a policy, parameterised with weights ϕ, which takes states and outputs actions, $\pi_\phi(s)$, and a critic, parameterised with weights ψ, which outputs estimated return given the agent is in state s, produces action a, and then follows the current policy afterwards, $Q_\psi(s,a) = \mathbb{E}_\pi[R_t | s_t = s, a_t = a]$. Usually, in the actor-critic architecture, the actor's goal is to maximise the output from the critic, so that its actions maximise the cumulative reward.

In our framework, we use a variant called Soft Actor Critic (SAC). SAC uses a modification of the RL framework which maximises, in addition to the cumulative rewards, the entropy of the policy, $H(\pi)$, such that the actions have certain randomness in them for exploration purposes. Eq. 6 then becomes:

$$\pi^* = arg\max_\pi \mathbb{E}_\pi[R_0^{MaxEnt}] = arg\max_\pi \mathbb{E}_\pi[\sum_{t'=0}^{T} \gamma^{t'} (r_{t'+1} + \alpha H(\pi(s_{t'})))]. \tag{8}$$

where α is a hyperparameter that controls the trade-off between maximising reward or entropy. In SAC, the actor outputs the parameters of a Normal distribution, μ and σ^2 (α affects σ^2). Then, the action is sampled from it. Again, this is done for exploratory purposes and can be deactivated at any time by selecting the mean instead. For more details on SAC and its update rule, see the original work of Haarnoja et al.[22]The actor and critic architectures used for this work are shown in Figure 4.

The action is injected onto DM as a correction on top of the integrator with the sum of linear and non-linear reconstructions, each with a specific gain. This is done to speed up training.

$$C_t = C_{t-1} + g^l c_t^l + g^{nl} c_t^{nl} + a_t;\ a_t = \pi(s_t), \tag{9}$$

where we use the optimal gain for both reconstructions that maximises the LE SR on a 500 frames test. We build the reward such that the agent minimises the sum of linear and non-linear reconstructions.

$$r_t = -(c_t^l + c_t^{nl})^2 \tag{10}$$

The reward above is computed as a 2D map (one value per actuator position) which will be used on update time to compare against the output of the critic as shown in Figure 4. Within the saturation regime, the RL agent will be focused on reducing the non-linear error (while perhaps increasing the linear one). Within the linear regime, the RL agent will reduce both. Nonetheless, the non-linear reconstruction may not be perfect, consequently, a combination in the form of a sum might be needed. The full RL model is shown in Figure 5.

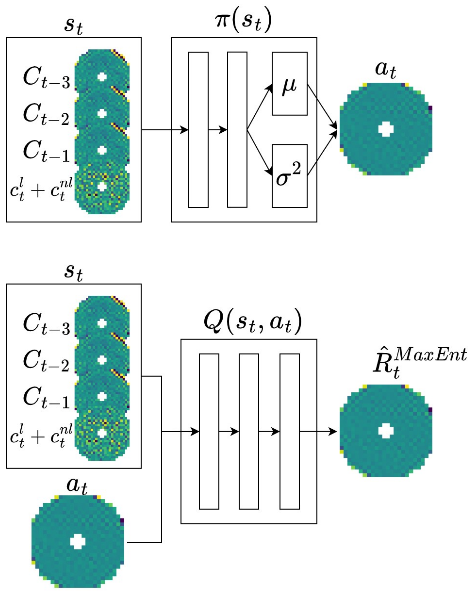


Figure 4: Architectures for both actor and critic architecture used for this work. The critic, $Q(s, a)$ takes the state and the action as input and outputs a 2D map of the predicted maximum entropy expected return, $\hat{R}^{MaxEnt}$. The actor, $\pi(s)$, takes as input the state and outputs two 2D maps corresponding to each actuator position, a mean and standard deviation one. These are the components from which the action is sampled. Both critic and actor are composed of convolutional layers. The output is always masked with invalid actuators being set to 0.

4. EXPERIMENTS

4.1 Experimental parameters

4.1.1 Simulation parameters

In this section, we show the different atmospheric parameters used in our simulation. For the simulation campaign, we use COMPASS version 5.2.0. We use a 3-layer turbulence model and an AO system on an 8m telescope with a 32x32 P-WFS and two DMs: a tip-tilt (tt) and high dimensional piezoelectric (pzt) mirror with a grid of 24x24 actuators. In the experiments, the RL agent only controls the pzt mirror. We leave for future work the problem of controlling both mirrors at the same time with the RL approach. We choose the number of pixels/subapertures of the WFS so as to oversample the DM by a factor of the order of ~ 1.4. It is empirically observed that an oversampling factor of 1.0 (corresponding to Fried's geometry) requires the user to filter out many modes to get a robust servo-loop behaviour. Conversely, a factor of 2.0 (Nyquist sampling) is usually deemed luxurious and is a waste of pixels. A factor of 1.3 to 1.4 is usually estimated to be a suitable trade-off. Thanks to this, we do not need to filter many modes to make an integrator work well enough (we only filter a single mode). We run experiments with different values of the Fried parameter, r_0, to account for different turbulence strengths, emphasising that our model can work under very different atmospheric conditions. Finally, as we want to focus on the highest performance possible, we consider a non-modulated pyramid. The remaining parameters can be seen in Table 1.

4.1.2 RL hyperparameters

For the RL module, we need to describe the hyperparameters of the actor and critic. Both actor and critic have three hidden layers with 64 filters, where each filter uses a 3x3 kernel with padding and stride of one. For the activation functions in the hidden layers, both actor and critic use rectified linear units.[23] For the output layer the critic uses a linear activation function and the actor uses a tanh after sampling the action from the Normal distribution. For the critic, the output is a 24x24 map of predicted maximum entropy cumulative rewards (conditioned in the input state, action and current policy) in each actuator position. For the actor, the output

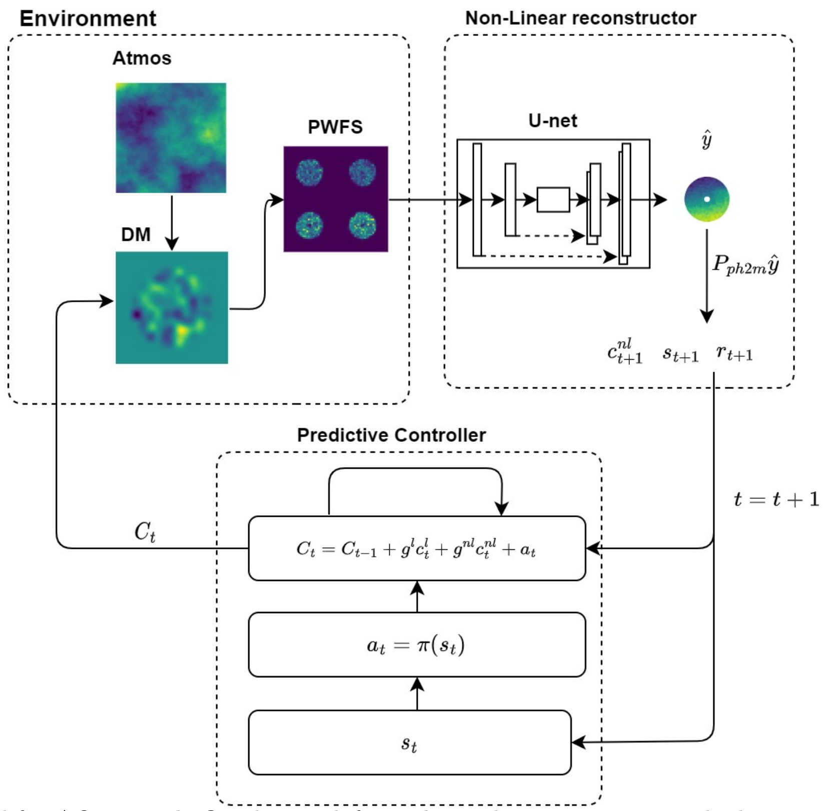


Figure 5: Full model for AO control. On the top left, we have the environment, which consists of the atmosphere, the P-WFS, and the DM. Then, the non-linear reconstructor takes the P-WFS as input to create the reconstruction, the state and the reward. Finally, the RL corrects the command that an integrator with the non-linear reconstruction does with the properties of a predictive controller, as demonstrated in our previous study.

Telescope parameters		**Atmospheric parameters**	
Diamenter (m)	8	Num. layers	3
Pupil diameter (pixels)	256	Altitude per layer (km)	0, 4, 10
AO loop parameters		r_0 (m)	variable @ 500 nm
Loop frequency (Hz)	1000	r_0 distribution	0.5/0.35/0.15
Frames of delay	2	L_0 (m)	30
Target parameters		Wind speed per layer (m/s)	15, 15, 35
λ_{target} (μm)	1.6	Wind direction per layer (°)	0 20 180
P-WFS parameters		**DM parameters**	
Num. subapertures	32x32	Mirrors	pzt and tt
Modulation	0	Num. actuators	24x24
λ_{wfs} (μm)	0.85	Modes filtered	1
Noise	None	Push4Imat [pzt (μm), tt (rad)]	[0.01, 0.00005]
		Coupling (pzt)	0.3
		Threshold response actuators (pzt)	0.3

Table 1: Simulation parameters.

is a 24x24 map of actions. Both outputs are masked according to the valid actuators. As the actions only affect close-by rewards and states in the time domain, the hyperparameter γ is set to 0.1. We manually set α to 50, which allowed us to get good results in the experiments. We filter both the state and the actions with the B_{tt} modes so that they do not contain information with higher spatial frequencies that the DM can manage. To assign the reward to a specific state-action tuple and train the agent, we skip one step before assigning, i.e. s_t and

a_t is be assigned to r_{t+2}. To avoid saturation, we set a maximum and minimum bound for the commands, which we set to [-300, 300] volts. We normalise the state by dividing between 100 (hence the command information in the state can only be between -3 and 3). Finally, we allow the value of the action in each timestep to be between [-10, 10] volts. For that, as the actor has an output tanh layer, we just multiply the output of the actor by 10.

During training, both actor and critic use ADAM optimiser[24] with a learning rate of 0.0003. Their weights are initialised with Xavier initialisation,[25] except for the last layer of the policy, which is initialised to 0 (as per the suggestion of previous work on residual policy learning[26, 27]). In this way, the commands in the initial stages of training are similar to the ones that the integrator would do.

Finally, in RL, we usually train in episodes of a certain number of steps. In this work, we have used episodes of 1000 frames. At the beginning of every episode, we reset the atmosphere (with another seed) and the DM shape. Every two training episodes, we perform an evaluation episode for the RL controller, the integrator with a linear reconstruction and the integrator with a linear plus non-linear reconstruction.

4.1.3 Non-linear reconstructor hyperparameters

The non-linear reconstructor is built with the U-net architecture explained in Section 3.1. The U-net expects inputs and output of the same size. For that reason, we pad with zeros each input channel for it to be the same size as the output, from 32x32 to 256x256. The encoder and decoder are symmetric. Each layer in the encoder (other than input/output) has a corresponding one in the decoder with the same number of filters and image size. Between those symmetric layers, there are skip-connections[28] which help train deep neural networks. All the layers have a 4x4 kernel, stride of two and padding of one. The decoder uses rectified linear units as the activation functions for the hidden layers and the encoder uses leaky rectified linear units.[29] In the case of the output, a tanh is used, which constrains the output between -1 and 1. Batch normalisation,[30] another stability trick for training deep neural networks, is used in all layers except the first and the last. Finally, we use three dropout layers[31] in the decoder's first three layers, which diminishes the possibility of overfitting. The number of filters per hidden layer is summarised below.

- **Encoder**: 64 - 128 - 256 - 512 - 512 - 512 - 512.
- **Decoder**: 512 - 512 - 512 - 512 - 256 - 128 - 64.

For training, we use ADAM optimisation with a learning rate of 0.0002, which is decayed linearly as more data is processed. The weights are initialised with a Normal distribution of mean 0 and variance 1. We use a batch size of 32. We perform an early stop procedure to select the best model available. Finally, we normalise the training data so that input is between 0 and 1 and output is between -1 and 1. During closed-loop operation, we must normalise the input and denormalise the output.

We want a model that can be used for any atmospheric condition. For that, and similarly to Landman et al.,[2] we use a dataset that consists of random phases created by reshaping the mirrors randomly. This is done by pushing both pzt and tt actuators randomly with values obtained from a Normal distribution with $\mu = 0$. The σ^2 is obtained by sampling values with a log-uniform distribution with minimum and maximum of $(0.015, 3)$ μm for the pzt mirror and $(0.00085, 0.15)$ rad for the tt mirror. Interestingly, as we use the phase as the output, we are able to create the dataset with a pzt mirror that has more actuators than the one used in the loop for a dataset that contains a higher frequency content. Using this method, we recollect a dataset with an 8m telescope equipped with a 32x32 pzt mirror and a tt mirror. The final dataset contains 500000 samples for training and 100000 samples for evaluation.

4.2 Results non-linear reconstructor

In this section, we evaluate the performance of the non-linear reconstructor. Figure 6 compares the performance of an integrator with a purely non-linear reconstruction, an integrator with a linear reconstruction and an integrator with the sum of both linear and non-linear for a 1000 frames. As observed in previous works,[2] the combination of both linear and non-linear reconstruction is able to obtain better performance than the linear and non-linear only. This provides the means to build our state and reward.

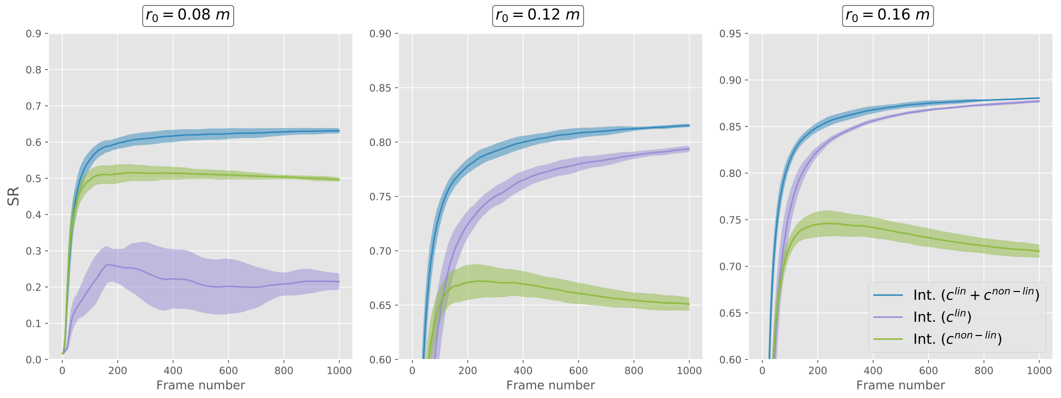


Figure 6: LE SR over frame number comparison of an integrator with linear, non-linear and linear plus non-linear reconstructions. The experiment was conducted for three different seeds. In the figure, we show the mean result of the three seeds shadowed by the standard deviation.

4.3 Results RL

In this section, we provide the results for an RL controller with the combination of linear and non-linear reconstruction.

4.3.1 Training

Figure 7 shows the evolution of the LE SR in the evaluation episodes for the RL controller compared against an integrator controller with a linear reconstruction and an integrator controller with the combination of linear and non-linear reconstructions. Initially, at frame 0, the RL controller has a similar performance as the integrator (recall that we have initialised the last layer of the policy to 0), and just after a few thousand frames, the RL approach is able to outperform both the integrator with a linear reconstruction and the integrator with a non-linear reconstruction. The gain in performance compared to the integrator increases as the seeing conditions worsen.

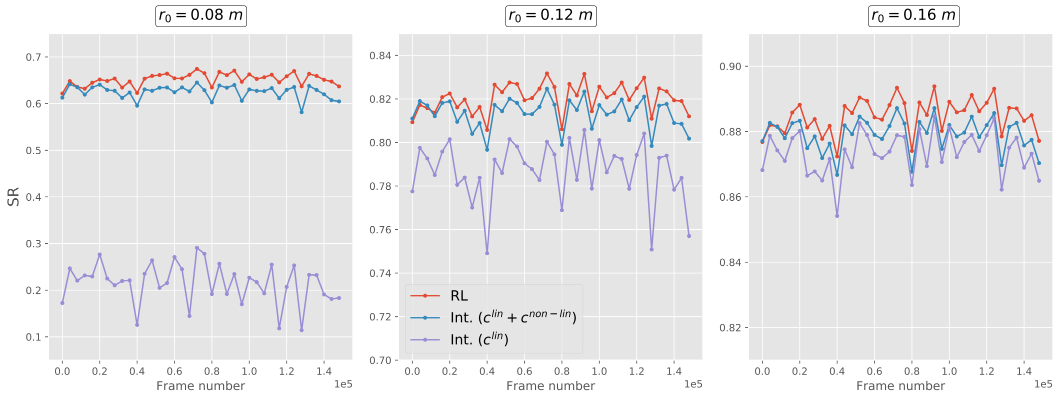


Figure 7: Training curves of the proposed model. The y-axis is the LE SR, and the x-axis is the frame number. Every point shows the final LE SR of a 1000-frame evaluation episode. We observe that the RL agent can generalise and is not prone to overfitting.

Moreover, we test if the new reward function is able to overcome the decoupling of SR and reward (seen in Figure 2). In Figure 8, we show both the evolution of the difference of average reward (in red) and the difference of LE SR (in blue) between the RL controller and the integrator with linear plus non-linear reconstruction over the different evaluation episodes. This is shown for the experiment of $r_0 = 0.08$ m. As we observe, most of the time, both functions are monotonic, when one increases, so does the other and vice-versa, effectively solving the problem we showed in the motivation section.

4.3.2 Final results on a single episode

Figure 9 shows the performance comparison for previously unseen conditions of an integrator with a linear reconstruction, an integrator with a combined linear plus non-linear reconstruction, the RL controller trained for 10^5 frames and the direct projection of the phase into the DM. As we observe, the RL agent is able to outperform both forms of the integrator. This can be easily explained as: the non-linear reconstruction reduces errors related to the reconstruction from the pyramid image, and the RL controller mitigates the temporal error. Again, this occurs for all atmospheric conditions. Nevertheless, there is room for improvement as the direct projection performs better than any of the aforementioned methods.

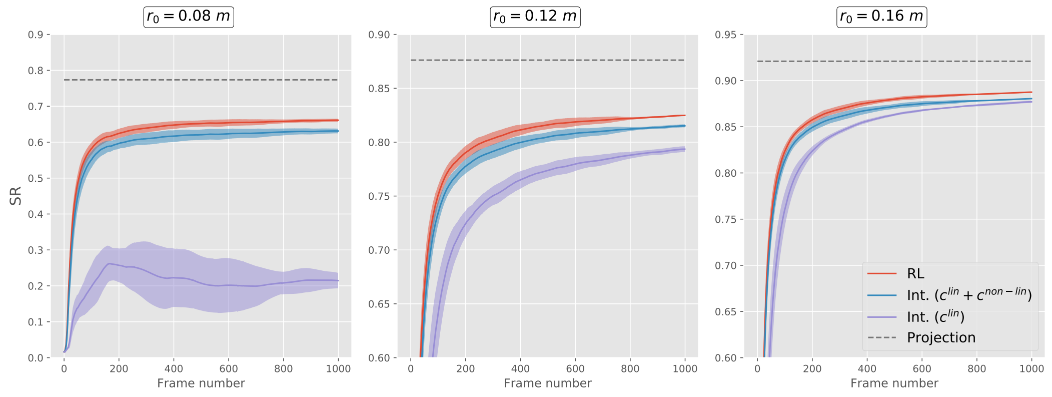


Figure 9: Performance comparison of an RL controller, an integrator with a linear reconstruction and an integrator with a linear plus non-linear reconstruction. We also have added to the graph the maximum performance a controller could achieve, the direct projection of the phase into the DM.

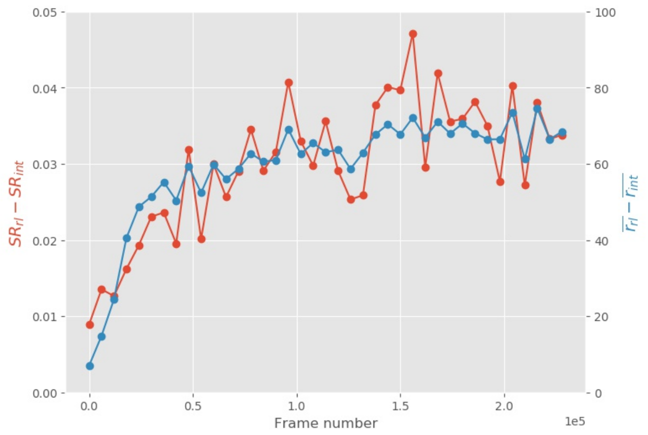


Figure 8: Solving the problem of decoupling between LE SR and reward. Red: evolution of difference in LE SR in evaluation episodes between RL agent and an integrator with linear plus non-linear reconstruction. Blue: evolution of difference in average reward in evaluation episodes between RL agent and an integrator with linear plus non-linear reconstruction.

5. DISCUSSION AND FUTURE WORK

We have presented a model composed of two parts: a non-linear reconstructor with SL and a predictive controller with RL. The non-linear reconstructor allows the RL model to learn without being confused with incorrect states and rewards. This combination is able to outperform an optimised integrator, correcting both errors due to reconstruction and temporal error.

Other researchers have presented their work with RL and the P-WFS.[15] We could not make it work properly without a non-linear reconstruction. We hypothesise that their exploratory commands are always within the linear range; therefore, their RL model can learn from a linear reconstruction of the P-WFS data. However, this might not work for bad seeing conditions.

For our next steps, we plan to optimise the hyperparameters of all the networks involved in the model to drive its performance to the limit. Moreover, we want to test the MARL approach with the non-linear reconstructor and observe its performance compared to the convolutional neural network single agent approach presented here. Another possible direction to explore is to derive the state and reward directly from the P-WFS pixels. The mentioned SR and reward decoupling would probably be avoided as it is an artefact of the reconstruction. Moreover, in this way, the RL agent would both do the job of non-linear reconstruction and predictive model, which would bring online adaptation to both parts. However, this might diminish training time. Still, these methods that use only information from the WFS can not avoid errors unseen by it, such as non-common path aberrations. For that, it could be helpful to derive a reward signal from, for example, the point spread function. Finally, validating the results on a test bench and later on-sky will help bring this method for science operations.

ACKNOWLEDGMENTS

We would like to thank Eric Gendron and Fabrice Vidal from Observatoire de Paris for their useful comments and discussion about the P-WFS. We also would like to thank Jesse Cranney and Hao Zhang from Australian National University for the fruitful discussions of potential uses of Artificial Intelligence in AO.

FUNDING

This project has received funding from the European Union's Horizon 2020 research and innovation program under the Marie Sklodowska-Curie grant agreement No 873120.

REFERENCES

[1] Guo, H., Korablinova, N., Ren, Q., and Bille, J., "Wavefront reconstruction with artificial neural networks," *Optics Express* **14**(14), 6456–6462 (2006).

[2] Landman, R. and Haffert, S., "Nonlinear wavefront reconstruction with convolutional neural networks for fourier-based wavefront sensors," *Optics Express* **28**(11), 16644–16657 (2020).

[3] Hu, L., Hu, S., Gong, W., and Si, K., "Deep learning assisted shack–hartmann wavefront sensor for direct wavefront detection," *Optics Letters* **45**(13), 3741–3744 (2020).

[4] Smith, J., Cranney, J., Gretton, C., and Gratadour, D., "Enhanced adaptive optics control with image to image translation," in [*The 38th Conference on Uncertainty in Artificial Intelligence*], (2022).

[5] Montera, D. A., Welsh, B. M., Roggemann, M. C., and Ruck, D. W., "Prediction of wave-front sensor slope measurements with artificial neural networks," *Applied optics* **36**(3), 675–681 (1997).

[6] Liu, X., Morris, T., Saunter, C., Juez, F. J. d. C., González-Gutiérrez, C., and Bardou, L., "Wavefront prediction using artificial neural networks for open-loop adaptive optics," *Monthly Notices of the Royal Astronomical Society* (2020).

[7] Swanson, R., Lamb, M., Correia, C., Sivanandam, S., and Kutulakos, K., "Wavefront reconstruction and prediction with convolutional neural networks," in [*Adaptive Optics Systems VI*], **10703**, 107031F, International Society for Optics and Photonics (2018).

[8] Swanson, R., Lamb, M., Correia, C. M., Sivanandam, S., and Kutulakos, K., "Closed loop predictive control of adaptive optics systems with convolutional neural networks," *Monthly Notices of the Royal Astronomical Society* **503**(2), 2944–2954 (2021).

[9] Hu, K., Xu, Z., Yang, W., and Xu, B., "Build the structure of wfsless ao system through deep reinforcement learning," *IEEE Photonics Technology Letters* **30**(23), 2033–2036 (2018).

[10] Ke, H., Xu, B., Xu, Z., Wen, L., Yang, P., Wang, S., and Dong, L., "Self-learning control for wavefront sensorless adaptive optics system through deep reinforcement learning," *Optik* **178**, 785–793 (2019).

[11] Landman, R., Haffert, S. Y., Radhakrishnan, V. M., and Keller, C. U., "Self-optimizing adaptive optics control with reinforcement learning," in [*Adaptive Optics Systems VII*], **11448**, 1144849, International Society for Optics and Photonics (2020).

[12] Nousiainen, J., Rajani, C., Kasper, M., and Helin, T., "Adaptive optics control using model-based reinforcement learning," *Optics Express* **29**(10), 15327–15344 (2021).

[13] Landman, R., Haffert, S. Y., Radhakrishnan, V. M., and Keller, C. U., "Self-optimizing adaptive optics control with reinforcement learning for high-contrast imaging," *Journal of Astronomical Telescopes, Instruments, and Systems* **7**(3), 039002 (2021).

[14] Pou, B., Ferreira, F., Quinones, E., Gratadour, D., and Martin, M., "Adaptive optics control with multi-agent model-free reinforcement learning," *Optics express* **30**(2), 2991–3015 (2022).

[15] Nousiainen, J., Rajani, C., Kasper, M., Helin, T., Haffert, S., Vérinaud, C., Males, J., Van Gorkom, K., Close, L., Long, J., et al., "Towards on-sky adaptive optics control using reinforcement learning," *arXiv preprint arXiv:2205.07554* (2022).

[16] Paschall, R. N. and Anderson, D. J., "Linear quadratic gaussian control of a deformable mirror adaptive optics system with time-delayed measurements," *Applied optics* **32**(31), 6347–6358 (1993).

[17] Ragazzoni, R., "Pupil plane wavefront sensing with an oscillating prism," *Journal of modern optics* **43**(2), 289–293 (1996).

[18] Deo, V., Gendron, É., Vidal, F., Rozel, M., Sevin, A., Ferreira, F., Gratadour, D., Galland, N., and Rousset, G., "A correlation-locking adaptive filtering technique for minimum variance integral control in adaptive optics," *arXiv preprint arXiv:2103.09921* (2021).

[19] Gratadour, D., Puech, M., Vérinaud, C., Kestener, P., Gray, M., Petit, C., Brulé, J., Clénet, Y., Ferreira, F., Gendron, E., Lainé, M., Sevin, A., Rousset, G., Hammer, F., Jégouzo, I., Paillous, M., Taburet, S., Yang, Y., Beuzit, J.-L., Carlotti, A., Westphal, M., Epinat, B., Ferrari, M., Gautrais, T., Lambert, J. C., Neichel, B., and Rodionov, S., "Compass: an efficient, scalable and versatile numerical platform for the development of elt ao systems," in [*Adaptive Optics Systems IV*], **9148**, 91486O, International Society for Optics and Photonics (2014).

[20] Ferreira, F., Gendron, E., Rousset, G., and Gratadour, D., "Numerical estimation of wavefront error breakdown in adaptive optics," *Astronomy & Astrophysics* **616**, A102 (2018).

[21] Ronneberger, O., Fischer, P., and Brox, T., "U-net: Convolutional networks for biomedical image segmentation," in [*International Conference on Medical image computing and computer-assisted intervention*], 234–241, Springer (2015).

[22] 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* (2018).

[23] Nair, V. and Hinton, G. E., "Rectified linear units improve restricted boltzmann machines," in [*Icml*], (2010).

[24] Kingma, D. P. and Ba, J., "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980* (2014).

[25] Glorot, X. and Bengio, Y., "Understanding the difficulty of training deep feedforward neural networks," in [*Proceedings of the thirteenth international conference on artificial intelligence and statistics*], 249–256, JMLR Workshop and Conference Proceedings (2010).

[26] Silver, T., Allen, K., Tenenbaum, J., and Kaelbling, L., "Residual policy learning," *arXiv preprint arXiv:1812.06298* (2018).

[27] Johannink, T., Bahl, S., Nair, A., Luo, J., Kumar, A., Loskyll, M., Ojea, J. A., Solowjow, E., and Levine, S., "Residual reinforcement learning for robot control," in [*2019 International Conference on Robotics and Automation (ICRA)*], 6023–6029, IEEE (2019).

[28] He, K., Zhang, X., Ren, S., and Sun, J., "Deep residual learning for image recognition," in [*Proceedings of the IEEE conference on computer vision and pattern recognition*], 770–778 (2016).

[29] Maas, A. L., Hannun, A. Y., Ng, A. Y., et al., "Rectifier nonlinearities improve neural network acoustic models," in [*Proc. icml*], **30**(1), 3, Citeseer (2013).

[30] Ioffe, S. and Szegedy, C., "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in [*International conference on machine learning*], 448–456, PMLR (2015).

[31] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R., "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research* **15**(1), 1929–1958 (2014).