Learning to Look by Self-Prediction

Anonymous authors
Paper under double-blind review

Abstract

We present a method for learning active vision skills, to move the camera to observe a robot's sensors from informative points of view, without external rewards or labels. We do this by jointly training a visual predictor network, which predicts future returns of the sensors using pixels, and a camera control agent, which we reward using the negative error of the predictor. The agent thus moves the camera to points of view that are most predictive for a chosen sensor, which we select using a conditioning input to the agent. We observe that despite this noisy learned reward function, the learned policies a exhibit competence by reliably framing the sensor in a specific location in the view, an emergent location which we call a behavioral fovea. We find that replacing the conventional camera with a foveal camera further increases the policies' precision.

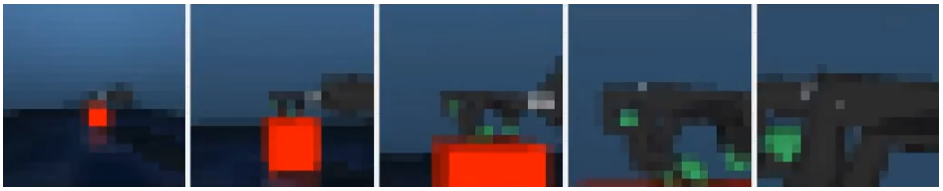
1 Introduction

Computer vision, as commonly employed in embodied RL and robotics, does not closely resemble human vision. Human vision has moving eyes, fovea, movements such as saccades to frame targets in view, and smooth pursuit to track them (Dodge, 1903). By contrast, in robotics and embodied RL, the camera is often rigidly fixed to the environment, limiting the agent to operate within the camera's fixed field of view (Levine et al., 2016). With the camera fixed to the environment or robot, moving objects cause large amounts of input variance as they traverse pixels. This can be a source of training instability (Cetin et al., 2022), often leading researchers to avoid learning vision altogether and use object features or off-the-shelf vision modules instead (OpenAI et al., 2019).

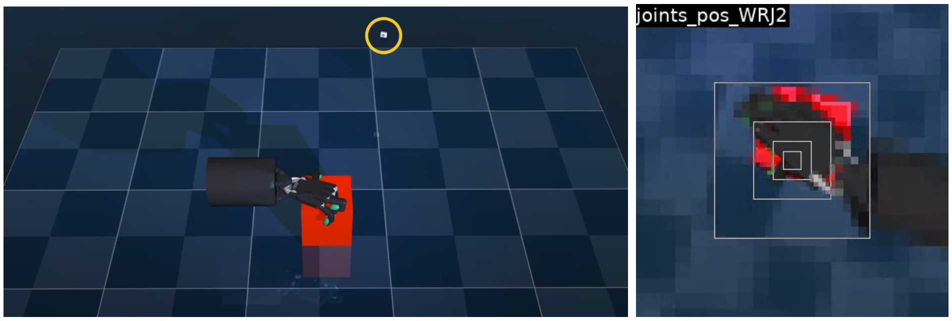
An agent that has learned to visually frame objects in a consistent image location could simplify the acquisition of visually-guided manipulation policies, as they can then focus on the manipulation aspect of the policy. This intuition has recently seen evidence in robotic manipulation research on hand-mounted cameras, where the object's apparent position roughly stays the same as the hand is about to grasp it (Hsu et al., 2022; Cheng et al., 2018; Gualtieri & Platt, 2018; Szot et al., 2021; Jangir et al., 2022). This "hand-chosen" camera mount consistently frames objects about to be grasped, but not necessarily other elements of the environment, such as footholds to step on, obstacles to duck, or other agents to collaborate or compete with.

By contrast, humans benefit from decoupling the kinematic chain of the eye from those of the limbs, allowing them to flexibly choose their visual focus in highly dynamic tasks ranging from fielding baseballs (McBeath et al., 1995) to traversing challenging terrain (Matthis et al., 2018). These visual policies all share common building blocks in the form of fixation and tracking (saccades and smooth pursuit). One inspiration to our paper is the question: if fixation is a general visual skill that is key to acquiring more task-specific visual policies, how can an agent learn it independently of specific tasks? This paper demonstrates how an embodied agent can acquire visual skills in the absence of external rewards.

We propose that learning to observe one's interactions with the world can start with learning to look at the interface to those interactions, namely the parts of one's own body. We bootstrap this by training a predictor network to predict the body's sensors from vision alone. At the same time, we use this predictor's errors as negative rewards for an RL agent that moves the camera. In other words, we reward the agent for moving the camera to viewpoints that yield better predictions for a chosen sensor. We communicate this choice of "target sensor" to the agent using a simple one-hot conditioning input. In this manner, we show



(a) View from the foveal camera.



(b) The environment with hand, block prop and camera (highlighted by the yellow circle).

(c) Stacked visualization of scaled foveal images.

Figure 1: The environment, with hand, box prop, and foveal camera. Fig. 1a shows the foveal image, a stack of 5 RGB images, with shape (5, height, width, 3). The images have the same dimension, but cover differing fields of view: 90, 45, 22, 11, and 5 degrees. Fig. 1c shows another foveal view.

that a single agent can learn distinct look-at policies, one per target sensor, without hand-designed external rewards. The predictor network, and the agent's policy and critic networks, can be trained simultaneously without manual scheduling. The learned camera policies are competent relative to baselines, navigating around occluding objects and precisely framing the targeted sensor in a consistent location within the view. We call this emergent area of the retina a "behavioral fovea".

Motivated by this emergent behavioral fovea, we implement a foveated camera (Cheung et al., 2017; Harris et al., 2019; Deza & Konkle, 2020a;b) with exponentially higher resolution near the center of the field of view. We show that this results in more precise framing behavior, as the agent learns to position the subject near the center, where it can observe it at the highest resolution.

2 Related Work

Unsupervised Learning of Tracking. Among the wider literature on tracking objects on video, some recent work focused on unsupervised learning: conditional object-centric tracking given an initial bounding box cue (Kipf et al., 2021), segmentation using Object Discovery and Representation Networks (Hénaff et al., 2022) and segmentation using motion-based (optical flow) and appearance-based information (Choudhury et al., 2022). Our unsupervised approach learns to execute large viewpoint changes in 3D environments, rather than tracking or annotating within the fixed view given by a video.

Computational Models of Foveation. Cheung et al. (2017) introduce a neural attention model with a learnable retinal sampling lattice, with multiple extensions (Harris et al., 2019; Deza & Konkle, 2020a;b). Other work uses spatial attention (Kosiorek et al., 2017) or a virtual fovea (Burt et al., 2021) to imitate the equivalent of *what* (bottom-up saliency) and *where* (top-down attention) pathways in animal vision. Rivkind et al. (2022) introduce a low-resolution dynamical sensor that moves with drift-like tiny steps mimicking the microsaccades of a human eye. By contrast, we implement visual attention as a motor policy capable of driving the camera to informative viewpoints in 3D, rather than controlling which pixels to attend to within a given 2D image. Instead of cropping or masking out the non-foveal part of the image, we retain the low-resolution wide-angle periphery to aid the camera agent in navigating through a 3D environment.

Visual Attention in Reinforcement Learning. Attention mechanisms for agents playing RL games were introduced by Sorokin et al. (2015). They observed that top-down attention mechanisms forced agents to focus on task-relevant information by sequentially querying the environment (Mott et al., 2019) and helped generate virtual goals to replay (Liu et al., 2020). Guo et al. (2021) studied analogies between the visual attention of human experts and saliency maps in RL agents. Recently, self-supervised attention in RL agents has been used to select regions of interest without explicit annotations (Wu et al., 2021) and has provided robustness as well as increased learning efficiency and interpretability for visual tasks (Salter et al., 2020; James & Davison, 2022; James et al., 2022; Tang et al., 2020). Our active vision approach uses RL to control the camera, instead of controlling top-down attention over fixed views.

Active Vision in Reinforcement Learning. Embodied perception in a navigating agent enables it to move around an object to perceive it better (Yang et al., 2019) or to solve semantic segmentation tasks (Chaplot et al., 2020; Nilsson et al., 2021). In a panoptic camera rig, RL can be used to select the best viewpoint for human pose estimation (Gärtner et al., 2020), 3D reconstruction (Pirinen et al., 2019), forecasting the effects of motion (Jayaraman & Grauman, 2016), and more generally, learning to look around to efficiently gather information about the agent's surroundings (Ramakrishnan et al., 2019; Jayaraman & Grauman, 2018). Our work focuses on learning active vision skills without external task rewards.

3 Methods

Our method teaches a camera-controlling agent to visually frame parts of its own body. It does this by simultaneously training three networks: the agent's policy and critic networks, and a separate predictor network that predicts the body's sensor values from the camera's pixels. We use the predictor's errors as reinforcement learning penalties for the camera agent, incentivizing it to move the camera to more informative views. Below, we describe each of these components.

3.1 Environment

Our environment uses the MuJoCo (Todorov et al., 2012) physics simulator, in which we place a camera at the end of an invisible armature (the *camera bot*), controlled by the camera agent. This camera bot shares the scene with a manipulator, and a block prop randomly positioned within reach of the fingers, which gives the touch sensors at the fingertips something to touch (fig. 1b). Both the camera bot and manipulator are driven by velocity control, i.e. proportional-integral-derivative (PID) control in which the actions specify a target velocity for each actuator.

We treat the camera bot and manipulator as belonging to one robot that is conceptually split into two separate entities. The manipulator runs a fixed random behavioral policy throughout the experiment, but sensors on the manipulator are made available to define the losses of the predictor network (section 3.3), a multi-headed generalized value function (GVF) whose losses define the endogenous penalty function for the camera agent (section 3.4). The camera agent has to learn to move to look at selected parts of the manipulator to better predict the future values of the targeted sensor in the manipulator. The training process only modifies the behavior of the camera bot, not the manipulator.

At the start of each episode, we randomize the joint angles of the manipulator and camera bot, and the position and dimensions of the box prop. We also randomly choose a sensor on the manipulator for the camera agent to target. We use an episode length sufficient to allow the camera enough time to reach an arbitrary final pose from its initial randomized pose.

3.2 Cameras

The camera bot can be equipped with a conventional camera or a foveal one. Like Mnih et al. (2014), we implement the foveal camera as N conventional cameras, all with the same position, orientation, and pixel dimensions, but differing in their field of view. Fig. 1a shows an example of the resulting multi-scale images as seen by the agent, and fig. 1c demonstrates the scales' relative fields of view by stacking them on top of each other.

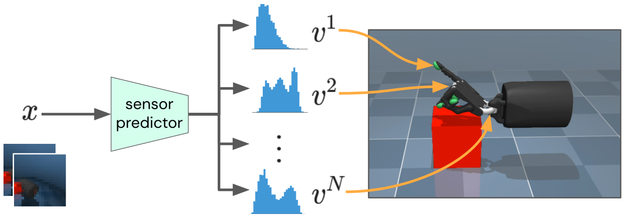
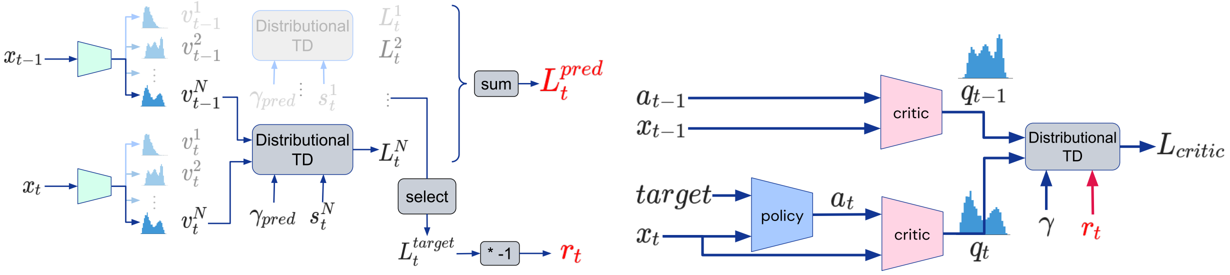


Figure 2: Predictor network input and outputs. The input x is a stack of the previous two camera observations (to enable predicting velocities). The outputs $\{v_i\}_{i=1}^N$ are the distributions of the estimated return values of N proprioceptive sensors, such as joint angles or touch sensor readings. For clarity, we have lightened the background on this high-resolution image rendered in MuJoCo.



(a) Predictor loss L_t^{pred} and agent reward r_t.

(b) Agent critic loss L_{critic}.

Figure 3: a) Distributional TD losses L_t^i are computed for each sensor reading s_t^i, in a loop over N sensors (the figure only shows the N'th sensor's reading s_t^N). The predictor loss L_t^{pred} is the sum of the N distributional TD losses. A different *target* sensor is chosen at the start of each episode, whose prediction error L_t^{target} is used as a negative agent reward r_t. b) The D4PG (Barth-Maron et al., 2018) critic loss L_{critic} depends on the learned agent reward r_t. The *target* sensor (see Section 3.4) is specified as a one-hot vector, fed to the policy as a conditioning input.

3.3 Predictor network

The predictor network shares no weights with the policy or critic networks in the agent. It is a convolutional residual network (ResNet) followed by a multi-layer perceptron (MLP). Its architecture and layer sizes are taken from IMPALA's (Espeholt et al., 2018) convolutional residual network, replacing the LSTM at the end with an MLP. This predictor network takes two consecutive images as input, and outputs predictions for all potential target sensors. Like the critic network, its predictions take the form of discrete distributions v_t of estimated future-discounted returns y_t of sensor signals s_t rather than of rewards.

$$y_t = \sum_{t=0}^{\infty} \gamma^i s_{t+i+1} \tag{1}$$

$$v_t(y) = p(y_t = y) \tag{2}$$

This was inspired by multi-timescale nexting (Modayil et al., 2014), and the predictor network can be thought of as a multi-headed GVF (Sutton et al., 2011) with distributional outputs. Predicting a decaying sum of sensor readings, rather than a future sensor reading at a particular point in time, makes our experiments less dependent on the exact choice of frame rate or prediction timescale. Furthermore, using distributional TD losses instead of L2 losses serves as a principled means of normalizing the prediction losses across multiple sensor modalities with very different numerical ranges and distributions.

Driving behavior to maximize prediction errors has been suggested as a form of curiosity (White et al., 2014). Our agent does the opposite of this, as it is rewarded for minimizing error. Instead of maximizing short-term surprise, it explores the state space through the diversity of its prediction target sensors. This is partially a reflection of our different setting. When an agent physically interacts with the world under a fixed camera, it may make sense to explore by maximizing prediction error. When the agent moves the camera itself, it can trivially increase prediction error by staring in uninformative directions, without meaningful exploration.

3.3.1 Agent networks

The agent consists of a policy network and a critic network. Like the predictor network, each of these is an MLP stacked on an IMPALA-style convolutional ResNet. The critic and policy networks share a ResNet, but have different MLP heads. Both networks take a mix of images and other inputs. The images are fed through the ResNet, and the other inputs are concatenated with the ResNet's output and fed to the MLP. The policy network takes as input the observation (two consecutive image frames), and a one-hot vector specifying the target. It outputs bounded continuous target velocities for the camera bot's four joints (section 3.1). The state-action critic additionally takes the action, and outputs a discrete distribution over the estimated return. The agent's training objective is to minimize the predictor network's error for the target sensor, by modifying the policy network to move the camera to look at the associated body part.

3.4 Training

For each batch of transitions sampled from the replay buffer, we compute losses for the predictor, agent critic, and agent policy networks, then perform gradient updates on all three.

To train the predictor, we compute the prediction error of sensor i using the distributional TD loss (Bellemare et al., 2017) from D4PG (Barth-Maron et al., 2018). This is analogous to the standard TD error ($\delta = r_t + \gamma v_t - v_{t-1}(x)$), except that values v are represented not by scalars, but by discrete distributions over the range of possible return values (eq. 2).

$$L_t^i = DistributionalTD(s_t^i, \gamma_{pred}, v_t^i, v_{t-1}^i). \tag{3}$$

The predictor discount γ_{pred} is separate from the discount γ used for training the critic. Setting $\gamma_{pred} = 0$ amounts to performing next-frame prediction, while setting it to larger values predicts its future sum over a decaying time window with half-life $h = \Delta t \frac{ln(0.5)}{ln(\gamma_{pred})}$. It is possible to predict over multiple time windows as in Horde (Sutton et al., 2011), which may be useful in environments with predictable dynamics over multiple frames. For our environment, where the camera and manipulator have little momentum, we use decay γ_{pred} chosen to have a short half-life of 0.1s. The predictor network's loss is then the sum of prediction losses across all target sensors, $L_t^{pred} = \sum_i L_t^i$.

The camera agent networks share no parameters with the predictor. Each episode randomly chooses a proprioceptive sensor on the manipulator, to serve as the camera agent's *target* for that episode. The camera agent's task is to position the camera in a manner that reduces the predictor's error for that target sensor. We therefore define the reward to be $r_t = -L_t^{target}$, where L_t^{target} is the prediction error for that episode's target sensor, on timestep t. This is a dense reward, as it is available on every timestep, but a noisy one, as it is learned. The agent critic loss is analogous to eq. 3, substituting r_t for sensor reading s_t^i, γ for γ_{pred}, and critic outputs q_t and q_{t-1} for predictor outputs v_t and v_{t-1}. We train the policy network using the deterministic policy gradient loss (Silver et al., 2014).

4 Experimental Results

We describe our experimental setup, then present the claims supported by our results. All results are from the trained camera agent, evaluated without the exploration noise used during ϵ-greedy training.

4.1 Experimental setup

Environment The manipulator is a model of the 20 degrees of freedom (DOF) tendon-driven hand by Shadow Robotics (Shi et al., 2011; Plappert et al., 2018). We drive the manipulator using Perlin noise (Perlin, 1985), which provides temporally smooth control by interpolating random keypoint velocities with splines. The keypoint velocities are spaced 1 second apart and are uniformly sampled from the range of joint velocities.

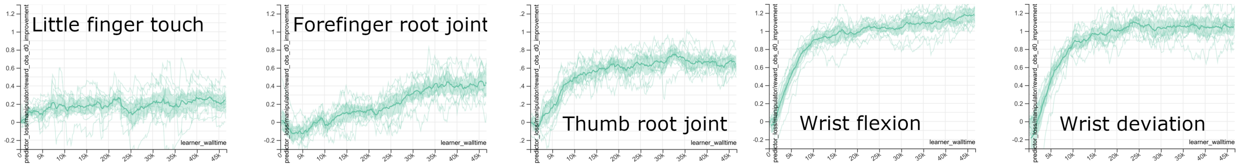


Figure 4: **Prediction improvement over an episode (higher is better).** X axis is wall time in seconds, spanning 13 hours of training. Y axis is the improvement of the target sensor's prediction error (L^{target}, eq. 3) from the beginning of the episode to the end, as the agent moves the camera to a better point of view. L^{target} is a KL divergence between distributions over possible return values, and is therefore unitless, and its range is independent of the sensor value range. All curves are taken from a single conditional agent told to target one of the above five target sensors, chosen randomly on each episode. Bold curve: the mean of 10 experiments with different RNG seeds. Shaded area: $\pm 1.96\sigma$.

Noise drives all the tendons, causing the hand to writhe about. The camera bot has four DOF with different ranges: elevation $[0, \pi]$, azimuth $[-\pi, \pi]$, distance $[.2m, 2m]$, and yaw $[-\pi, \pi]$. The first three DOFs move the camera in spherical coordinates centered around the manipulator. The yaw DOF rotates the camera around its local vertical axis, allowing it to look away from the hand. The values for these DOFs are randomized at the beginning of each episode. Randomizing the yaw DOF causes the manipulator to be entirely outside the field of view half the time. The episodes are 12 seconds long. The camera agent selects actions every 200 ms, yielding episodes of 60 timesteps.

Cameras In our experiments, the foveal camera has $N = 5$ cameras with 21×21 pixels each. Camera 1 has a FOV of 90 degrees in the vertical and horizontal directions, camera 2 has a FOV of 45 degrees, and so on, down to camera 5 with a FOV of 5.26 degrees (90×2^{-4}). The conventional camera is 21×21 pixels, with a FOV of 90×90 degrees.

Training We train the agent and predictor networks simultaneously. We employ 512 processes that each run an independent copy of the environment, pushing experience transitions (labeled with their episode's sensor targets) into the replay buffer. The buffer runs in its own process, and has a maximum capacity of 1M transitions (168 GB of RAM when using for $5 \times 21 \times 21$ foveal images). A single learner process samples mini-batches of 256 transitions from the replay buffer. The policy network trains with a learning rate of 10^{-5}, while the critic and predictor networks use a learning rate of 10^{-4}. We use Adam (Kingma & Ba, 2015) to minimize the losses. Each environment process runs on a machine with 1 CPU and 1.1 GB of RAM. The learner runs on a machine with 2 CPUs, 4.9 GB of RAM, and a TPU. Training takes roughly 72 hours to converge for both foveal and conventional agents.

Networks The agent's policy and critic networks share a convolutional ResNet, which acts as a visual feature extractor. Its architecture and layer sizes are taken from IMPALA's (Espeholt et al., 2018). The policy and critic have separate MLP heads, with hidden layer sizes $(256, 256, 256)$ and $(512, 512, 256)$, respectively. The predictor uses the same layer sizes as the critic, but has its own convolutional network weights.

Inputs and outputs The predictor takes a pair of successive frames from the camera, and outputs N discrete distributions for N possible target sensors, over the range of possible return values. In our experiments, $N = 5$. These five sensors were selected to represent both touch and joint angle sensors. We selected the predicted joints to cover a range of sizes and directions of movement when the joints bend. The critic takes an N-dimensional one-hot vector in addition to the pair of pixel frames. This vector indicates which sensor to target. The critic outputs a distribution over the possible return values of the learned reward. The predictor and critic's output distributions are 51-atom discrete distributions evenly spanning the range of possible returns. The policy network takes the target one-hot and a pair of pixel frames, and outputs continuous target velocities for the camera bot's four DOFs: azimuth, elevation, distance, and yaw.

4.2 The trained camera policy improves target prediction accuracy

Figure 4 shows the improvement of the target sensor's prediction error from the start to the end of an episode, plotted throughout training. We plot the error improvement rather than the final error (seen in table 1), because the latter would jointly evaluate the predictor and the camera agent. By contrast, the error improvement within an episode evaluates the camera policy, independent of the predictor. Regardless of predictor quality, a policy selecting velocities from a uniform distribution centered at zero does not, on average, improve prediction accuracy over the course of an episode. A camera policy trained to improve this prediction error will do so, as shown. The wrist joint angles (two rightmost plots) have the most visual impact, as they move the whole hand. The agent learns to improve those errors first, while policies targeting the finer sensors improve later.

	Little finger touch	Forefinger root joint angle	Thumb root joint angle	Wrist flexion angle	Wrist deviation angle
Blind (c)	0.680 ± 0.020	4.12 ± 0.065	4.11 ± 0.073	3.84 ± 0.059	4.08 ± 0.051
Random (c)	0.667 ± 0.067	4.16 ± 0.034	4.16 ± 0.056	3.51 ± 0.069	3.61 ± 0.050
Random (f)	0.648 ± 0.048	4.07 ± 0.033	4.09 ± 0.068	3.41 ± 0.054	3.58 ± 0.077
Ours (c)	0.582 ± 0.062	3.24 ± 0.070	2.98 ± 0.061	1.73 ± 0.054	2.17 ± 0.048
Ours (f)	0.606 ± 0.046	3.11 ± 0.050	2.75 ± 0.071	1.65 ± 0.047	1.93 ± 0.051
Oracle (c)	0.480 ± 0.020	2.80 ± 0.096	2.53 ± 0.025	1.45 ± 0.017	1.62 ± 0.026
Oracle (f)	0.587 ± 0.022	2.80 ± 0.030	2.53 ± 0.032	1.45 ± 0.009	1.66 ± 0.010

Table 1: **Target sensor's prediction error at episode end (lower is better).** The "(c)" and "(f)" indicate conventional or foveal camera. *Blind* and *Oracle* give upper and lower bounds to the error, and *Random* shows the prediction error of a randomly posed camera. See section 4.2.1 for more deteail. Error is measured as the TD error for predictions given as distributions over the range of possible return values. Confidence bounds indicate $\pm 1.96\sigma$, calculated from 10 runs with different RNG seeds.

4.2.1 Comparison with baselines

Table 1 shows the target's prediction error at the end of the episode (lower is better). Unlike the relative measure of the agent plotted in fig. 4, this is an absolute measure of the joint quality of the predictor and agent at the end of training. It compares trained agents against the following baselines:

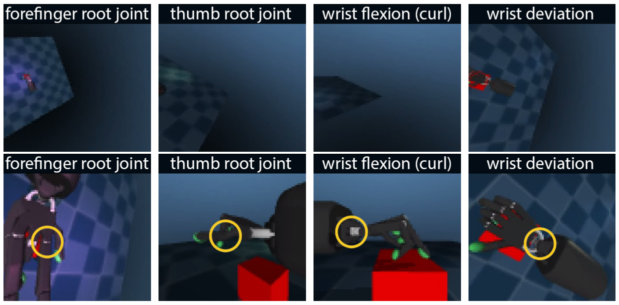
Blind: The *Blind* baseline is a predictor trained on a camera pointed away from the hand. It can do no better than learn to output each sensor's prior distribution, and serves as an upper bound to the expected agent prediction error.

Oracle: For a lower bound on the prediction error, we run a sweep over a series of hand-chosen fixed camera poses surrounding and looking at the hand, training a separate predictor for each pose. The *Oracle* entries show the minimum prediction error over all viewpoints for that target sensor. The Oracle benefits from not only specializing to a single point of view, but also from not moving, which significantly reduces data variance and improves prediction error even in the moving-camera agent. In practice, moving agents cannot always maintain a static view, nor can an agent with multiple static cameras usually know a priori which camera will yield the most accurate predictions.

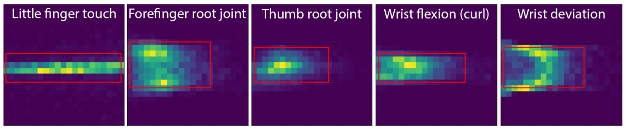
Random: The *Random* agent is our agent with its policy and critic learning rates set to zero. The camera spawns randomly, as usual, but hardly moves thereafter (brownian actions do little to move the camera bot, which has high inertia). The predictor must learn to predict from the resulting random camera views. These views are mostly static, advantageously reducing input variance in a similar manner to the still images of the Oracle. Our agent outperforms the Blind and Random baselines by a statistically significant margin.

4.3 The trained camera policy frames the subject

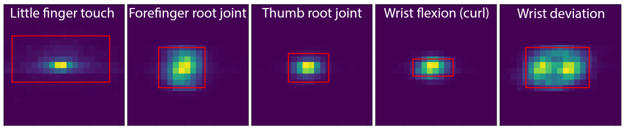
A consistent outcome in our experiments is that camera agents trained on a fixed choice of target tend to place the target in a particular position on the screen. This position varies from one training run to another, though it is usually near the center (fig. 5a). This is not an instance of the camera agent having memorized a particular set of preferred values for its own joint angles. The camera agent is unaware of the camera bot's joint angles, as the only inputs are pixels and the one-hot vector specifying the target sensor. Furthermore, it controls the camera bot joints by velocity control, unlike a position-controlled camera, which may learn to output a constant target camera pose regardless of the input. Figure 5b shows the emergence of this *behavioral fovea* from framing.



(a) First (top row) and last (bottom row) frames



(b) Histograms of targets on conventional camera



(c) Histograms of targets on foveal camera

Figure 5: Fig. **a**: the first and last frames (top and bottom row) from episodes where the sensor in the caption is the visual target. The figure highlights the target sensor with yellow circles (the agent does not see this). Note that the agent chooses to observe the wrist flexion angle (3rd column) from the side, from where it is most visible, while it observes the wrist's side-to-side deviation angle (4th column) from above. Fig. **b** and **c** show histograms of target location at the end of the episode, in image space, for conventional and foveal agents. The red box covers the 95% confidence interval (2.5% to 97.5% of the cumulative probability distribution) along each axis. The histograms accumulate the final target position over episodes collected after convergence, or the last 8 days out of an 11 day training run. This amounts to a total of 25000 episodes, or roughly 5000 episodes per target.

4.4 Training the camera policy with a foveal camera yields more precise framing

As shown in fig. 5b, agents learn to frame the target in a specific region of the image, even when equipped with a non-foveal camera. We call these regions *behavioral fovea*. Equipping agents with a foveal camera induces more focused and centered framing behavior (fig. 5c), as the agent can see a target at full resolution only when framing it at the center. A stronger tendency to center the subject further lessens the need to spend network capacity on position-equivariance. If the target moves around the workspace, tracking behavior may emerge as a side-effect of this tendency to keep it centered in the image. We discuss this future work in section 5.

Figure 5 also helps answer a fundamental question: why use RL to learn to look at sensors, as opposed to supervised policy training? After all, the positions of sensors are calculable from the robot's 3D geometry, which could provide image-space targets to direct the camera towards during training. Our RL-based method has the following advantages over such a supervised approach: (1) it needs no such prior information on the robot's geometry, (2) the agent learns not just what 3D point to look at, but from which direction, and (3) the most informative point of focus is not always the sensor itself. Fig. 5a illustrates point 2. The agent observes wrist flexion and deviation from orthogonal directions (i.e., along their orthogonal bending axes), despite looking at the same wrist. The rightmost image of fig. 5c illustrates point 3. The histogram shows two peaks; the wrist is framed off-center to the left or right, and the camera centers the hand instead. This is because the wrist primarily causes motion to the hand, making the latter more informative to look at. (Some sensors have a larger decoupling between the sensor position and the most informative view, such as an IMU-based orientation sensor attached to some arbitrary location in a large rigid body.)

4.5 The camera policy adopts distinct camera positions for different sensor targets

Sections 4.3 and 4.4 showed that the policy learns to orient the camera to frame the target sensor at a specific image location. While it is obviously important to look in the right direction, looking from the right position also matters. Fig. 6 shows the distribution of the camera position at the end of the episode for a single trained agent, showing a separate plot for each sensor target. Figure 6a shows that the camera has learned to observe the wrist flexion in profile, i.e., along the axis of rotation, from which the visual flow of flexing the wrist is most apparent. It observes the wrist from one side or the other, hence the bimodal distribution seen in the red projection. By contrast, observing the same wrist joint, but predicting its side-to-side deviation angle, behooves the agent to adopt a top-down view, as shown in fig. 6b. The differences in distribution for

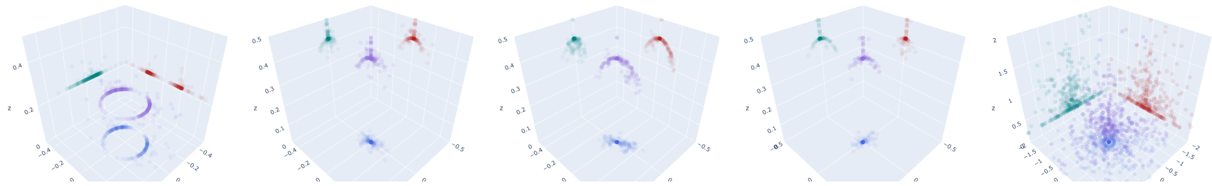

(a) Wrist flexion (curl) (b) Wrist deviation (c) Forefinger root joint (d) Thumb root joint (e) Little finger touch

Figure 6: Positions of the camera at the end of the episode, for different target sensors. Positions are in purple, with their projections to axis-aligned planes shown in red, green, and blue.

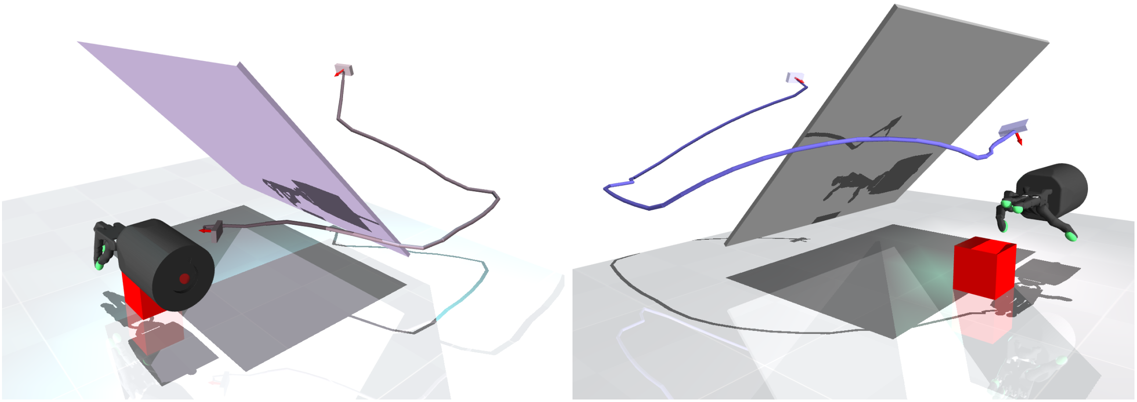

(a) Wrist flexion is viewed in profile. (b) Wrist deviation is viewed from above.

Figure 7: Occlusion-avoidant trajectories, by the camera agent trained with random occlusions.

the camera positions when observing the finger root joints (fig. 6c, fig. 6d are more subtle, observable in the blue projection along the floor. Fig. 6e shows the agent to be more position-agnostic when targeting the little finger's touch sensor. As shown in fig. 4a, the little finger's touch sensor is the most difficult to predict, due to the severe label imbalance presented to the predictor (the touch sensor touches the block infrequently under the hand's random policy). This leads to a noisy learned reward, which is good enough to teach the agent to center the touch sensor in the view (fig. 5c), but is insufficient to narrow down the camera position. In section 6 we propose a method to improve this.

4.6 The camera policy learns to circumnavigate occlusions

We introduced a randomized occluder to our training environment to test the generality of our self-supervised training. The occluder is a flat rectangle with random color, position, and dimensions. We uniformly sampled its height and width from the range $[0.5, 1.0]$, and sampled its position in spherical coordinates from $azimuth \sim [-\pi, \pi]$, $elevation \sim [0, \pi]$, $distance \sim [.4, .6]$. These spherical coordinates are roughly centered around the hand. For reference, the size of the environment's floor is 2×2. To maximize its effectiveness as an occluder, we orient the board to face the hand. Fig. 7 shows two trajectories of the camera agent, starting from the same initial conditions. The resulting camera policy is able to sidestep the occluder when it blocks the camera's view, thereafter exhibiting similar viewpoint preferences as in the unoccluded case.

5 Discussion

Many animals do not have a fovea. Some, like rats, have a roughly even distribution of optical receptors spread over a nearly spherical field of view. When objects in all directions are seen with equal acuity, one might ask if rats need to move their eyes to look *at* subjects at all. Yet they do, exhibiting similar visual skills to humans, such as centering and fixating the subject in a specific area of the retina (Holmgren et al., 2021). Like the rat, we find that our non-foveal agents exhibit framing behavior, positioning the subject in a specific area of the image, giving rise to a *behavioral fovea* despite there being no intrinsic acuity advantage in one area of the image versus another.

This suggests that the dynamics of the training encourage a positive feedback loop between the predictor and agent: the predictor improves its expertise in a specific region of the field of vision, and the agent learns to move the relevant subject into this visual region to receive the better prediction reward. This in turn provides to the predictor even more training data with the target in that image position, further improving its predictions there. This behavior is in contrast to the usual emphasis placed on learning position-invariant or position-equivariant representations in computer vision research that uses static datasets of images rather than an active camera (LeCun et al., 1998). That said, even static facial recognition has been shown to benefit from normalizing the facial feature locations (Taigman et al., 2014), a domain-specific form of framing.

Limitations We simulated the Shadow hand in the MuJoCo environment and thus have yet to investigate real-world complexities such as sensor timing, synchronization, noise, resolution, and discretization. Predicting distributions of returns instead of sensor readings isolates our framework from some of these real-world complexities but not all (e.g. motion blur). We use a random policy for the hand, under which the fingertips rarely touch the block, resulting in highly skewed distributions for their touch sensors. This results in relatively poor touch prediction compared to joint angle prediction. In future work, we plan to reward the hand for maximizing target sensor entropy, yielding more pedagogical hand behavior with a flatter sensor distribution. We randomize the lighting and expect that other standard data-randomization techniques could enable more robust sim2real transfers. At a higher level, this work does not address the open research question of how an agent can form an expanding conceptual space of internal GVF questions beyond predicting its own sensors.

Future work Giving our agent a foveal camera incentivizes it to position the subject more precisely at the center of the image. One direction for future work is to investigate whether the agent can maintain this centering as the target body part makes large movements, resulting in tracking behavior from the camera as a side-effect of framing. We demonstrate that a single agent can be trained to fixate on one of several targets, specified with a conditioning input. In this work, we limit the sensors to a single hand, but a realistic body provides a rich variety of visual scales and distances, from shoulders to toe tips. We posit that these present a means to learn a rich repertoire of visual fixation and tracking skills without the need to design their reward functions. These skills could serve as a basis for exploration while learning higher-level skills, as demonstrated with SAC-X (Riedmiller et al., 2018), which used manually designed reward functions. Parisi et al. (2022) show that visual RL agents benefit from pretraining on static image datasets, even when the dataset imagery looks nothing like the tasks. Self-supervised RL presents a means of feature learning that is more natural, in a lifelong learning sense, than using human-annotated image datasets. Our method learns features that are sufficient for the task of precise visual fixation. Whether they improve performance on general control tasks remains to be seen.

6 Conclusion

Learning motor skills without externally defined rewards or tasks is one path towards lifelong learning. Externally specified rewards can require privileged information that is unavailable to a real embodied agent. This is especially true of skills such as active vision, employed across a wide variety of embodied visual tasks.

In this paper, we demonstrate a means of learning visual fixation from self-prediction alone. We train a single embodied agent to visually fixate on different parts of its own body, as chosen by a conditioning input from a set of proprioceptive sensors. We show that this encourages the emergence of a *behavioral fovea*, where the fixated body part typically appears in a specialized region of the image, even when using conventional cameras. We show that when provided with an actual foveated camera, the same agent more strongly constrains the target to the center of the fovea.

We show that the agent learns to adopt distinct points of view for observing different sensor targets. Taken together, these results present a means to learn a variety of visual skills, up to one per proprioceptive sensor, without the need for hand-designed rewards.

References

Gabriel Barth-Maron, Matthew W Hoffman, David Budden, Will Dabney, Dan Horgan, Dhruva Tb, Alistair Muldal, Nicolas Heess, and Timothy Lillicrap. Distributed distributional deterministic policy gradients. *6th International Conference on Learning Representations, ICLR*, 2018. URL `http://arxiv.org/abs/1804.08617`.

Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. In *International Conference on Machine Learning*, pp. 449–458. PMLR, 2017.

Ryan Burt, Nina N Thigpen, Andreas Keil, and Jose C Principe. Unsupervised foveal vision neural architecture with top-down attention. *Neural Networks*, 141:145–159, 2021. URL `https://arxiv.org/abs/2010.09103`.

Edoardo Cetin, Philip J Ball, Stephen Roberts, and Oya Celiktutan. Stabilizing off-policy deep reinforcement learning from pixels. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 2784–2810. PMLR, 17–23 Jul 2022. URL `https://proceedings.mlr.press/v162/cetin22a.html`.

Devendra Singh Chaplot, Helen Jiang, Saurabh Gupta, and Abhinav Gupta. Semantic curiosity for active visual learning. In *European Conference on Computer Vision*, pp. 309–326. Springer, 2020.

Ricson Cheng, Arpit Agarwal, and Katerina Fragkiadaki. Reinforcement learning of active vision for manipulating objects under occlusions. In *Conference on Robot Learning*, pp. 422–431. PMLR, 2018. URL `http://proceedings.mlr.press/v87/cheng18a/cheng18a.pdf`.

Brian Cheung, Eric Weiss, and Bruno Olshausen. Emergence of foveal image sampling from learning to attend in visual scenes. *5th International Conference on Learning Representations, ICLR*, 2017. URL `https://arxiv.org/abs/1611.09430`.

Subhabrata Choudhury, Laurynas Karazija, Iro Laina, Andrea Vedaldi, and Christian Rupprecht. Guess what moves: Unsupervised video and image segmentation by anticipating motion. *arXiv preprint arXiv:2205.07844*, 2022. URL `https://arxiv.org/abs/2205.07844`.

Arturo Deza and Talia Konkle. Emergent properties of foveated perceptual systems. *arXiv preprint arXiv:2006.07991*, 2020a. URL `https://arxiv.org/abs/2006.07991`.

Arturo Deza and Talia Konkle. Foveation induces robustness to scene occlusion in deep neural networks. *Journal of Vision*, 20(11):442–442, 2020b. URL `https://jov.arvojournals.org/article.aspx?articleid=2771566`.

Raymond Dodge. Five types of eye movement in the horizontal meridian plane of the field of regard. *American journal of physiology-legacy content*, 8(4):307–329, 1903.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Vlad Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, et al. Impala: Scalable distributed deep-rl with importance weighted actor-learner architectures. In *International conference on machine learning*, pp. 1407–1416. PMLR, 2018.

Erik Gärtner, Aleksis Pirinen, and Cristian Sminchisescu. Deep reinforcement learning for active human pose estimation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 10835–10844, 2020. URL `https://ojs.aaai.org/index.php/AAAI/article/view/6714`.

Marcus Gualtieri and Robert Platt. Learning 6-dof grasping and pick-place using attention focus. In *Conference on Robot Learning*, pp. 477–486. PMLR, 2018. URL `http://proceedings.mlr.press/v87/gualtieri18a/gualtieri18a.pdf`.

Suna Sihang Guo, Ruohan Zhang, Bo Liu, Yifeng Zhu, Dana Ballard, Mary Hayhoe, and Peter Stone. Machine versus human attention in deep reinforcement learning tasks. *Advances in Neural Information Processing Systems*, 34:25370–25385, 2021. URL `https://arxiv.org/abs/2010.15942`.

Ethan William Albert Harris, Mahesan Niranjan, and Jonathon Hare. Foveated convolutions: improving spatial transformer networks by modelling the retina. In *Shared Visual Representations in Human & Machine Intelligence Workshop at NeurIPS 2019*, 2019. URL `https://eprints.soton.ac.uk/441204/`.

Olivier J Hénaff, Skanda Koppula, Evan Shelhamer, Daniel Zoran, Andrew Jaegle, Andrew Zisserman, João Carreira, and Relja Arandjelović. Object discovery and representation networks. *arXiv preprint arXiv:2203.08777*, 2022. URL `https://arxiv.org/abs/2203.08777`.

Carl D Holmgren, Paul Stahr, Damian J Wallace, Kay-Michael Voit, Emily J Matheson, Juergen Sawinski, Giacomo Bassetto, and Jason ND Kerr. Visual pursuit behavior in mice maintains the pursued prey on the retinal region with least optic flow. *Elife*, 10:e70838, 2021. URL `https://elifesciences.org/articles/70838`.

Kyle Hsu, Moo Jin Kim, Rafael Rafailov, Jiajun Wu, and Chelsea Finn. Vision-based manipulators need to also see from their hands. *10th International Conference on Learning Representations, ICLR*, 2022. URL `https://sites.google.com/corp/view/seeing-from-hands`.

Stephen James and Andrew J. Davison. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 7(2):1612–1619, 2022. doi: 10.1109/LRA.2022.3140817.

Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J. Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 13739–13748, June 2022.

Rishabh Jangir, Nicklas Hansen, Sambaran Ghosal, Mohit Jain, and Xiaolong Wang. Look closer: Bridging egocentric and third-person views with transformers for robotic manipulation. *IEEE Robotics and Automation Letters*, 2022. URL `https://arxiv.org/abs/2201.07779`.

Dinesh Jayaraman and Kristen Grauman. Look-ahead before you leap: End-to-end active recognition by forecasting the effect of motion. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling (eds.), *Computer Vision – ECCV 2016*, pp. 489–505, Cham, 2016. Springer International Publishing. ISBN 978-3-319-46454-1.

Dinesh Jayaraman and Kristen Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. In *Conference on Computer Vision and Pattern Recognition*, pp. 1238–1247, 06 2018. doi: 10.1109/CVPR.2018.00135.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *3rd International Conference on Learning Representations, ICLR*, 2015. URL `http://arxiv.org/abs/1412.6980`.

Thomas Kipf, Gamaleldin F Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional object-centric learning from video. *arXiv preprint arXiv:2111.12594*, 2021. URL `https://arxiv.org/abs/2111.12594`.

Adam Kosiorek, Alex Bewley, and Ingmar Posner. Hierarchical attentive recurrent tracking. *Advances in neural information processing systems*, 30, 2017. URL `https://arxiv.org/abs/1706.09262`.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Sergey Levine, Peter Pastor, Alex Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, 2016. URL `https://arxiv.org/abs/1603.02199`.

Peng Liu, Chenjia Bai, Yingnan Zhao, Chenyao Bai, Wei Zhao, and Xianglong Tang. Generating attentive goals for prioritized hindsight reinforcement learning. *Knowledge-Based Systems*, 203:106140, 2020. URL `https://www.sciencedirect.com/science/article/pii/S0950705120303920`.

Jonathan Samir Matthis, Jacob L Yates, and Mary M Hayhoe. Gaze and the control of foot placement when walking in natural terrain. *Current Biology*, 28(8):1224–1233, 2018. URL `https://www.sciencedirect.com/science/article/pii/S0960982218303099`.

Michael K McBeath, Dennis M Shaffer, and Mary K Kaiser. How baseball outfielders determine where to run to catch fly balls. *Science*, 268(5210):569–573, 1995. URL `https://www.bioteach.ubc.ca/TeachingResources/GeneralScience/BaseballPaper.pdf`.

Volodymyr Mnih, Nicolas Heess, Alex Graves, and koray kavukcuoglu. Recurrent models of visual attention. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K.Q. Weinberger (eds.), *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014. URL `https://proceedings.neurips.cc/paper/2014/file/09c6c3783b4a70054da74f2538ed47c6-Paper.pdf`.

Joseph Modayil, Adam White, and Richard S Sutton. Multi-timescale nexting in a reinforcement learning robot. *Adaptive Behavior*, 22(2):146–160, 2014. URL `https://arxiv.org/pdf/1112.1133.pdf`.

Alexander Mott, Daniel Zoran, Mike Chrzanowski, Daan Wierstra, and Danilo Jimenez Rezende. Towards interpretable reinforcement learning using attention augmented agents. *Advances in Neural Information Processing Systems*, 32, 2019. URL `https://arxiv.org/abs/1906.02500`.

David Nilsson, Aleksis Pirinen, Erik Gärtner, and Cristian Sminchisescu. Embodied visual active learning for semantic segmentation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. URL `https://arxiv.org/abs/2012.09503`.

OpenAI, Ilge Akkaya, Marcin Andrychowicz, Maciek Chociej, Mateusz Litwin, Bob McGrew, Arthur Petron, Alex Paino, Matthias Plappert, Glenn Powell, Raphael Ribas, Jonas Schneider, Nikolas Tezak, Jerry Tworek, Peter Welinder, Lilian Weng, Qiming Yuan, Wojciech Zaremba, and Lei Zhang. Solving rubik's cube with a robot hand, 2019. URL `https://arxiv.org/abs/1910.07113`.

Simone Parisi, Aravind Rajeswaran, Senthil Purushwalkam, and Abhinav Gupta. The (un)surprising effectiveness of pre-trained vision models for control. In *Workshop on Learning from Diverse, Offline Data*, 2022. URL `https://openreview.net/forum?id=0AYsHncySL4`.

Ken Perlin. An image synthesizer. *SIGGRAPH Comput. Graph.*, 19(3):287–296, jul 1985. ISSN 0097-8930. doi: 10.1145/325165.325247. URL `https://doi.org/10.1145/325165.325247`.

Aleksis Pirinen, Erik Gärtner, and Cristian Sminchisescu. Domes to drones: Self-supervised active triangulation for 3d human pose reconstruction. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, 2019.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018. URL `https://arxiv.org/abs/1802.09464`.

Santhosh K. Ramakrishnan, Dinesh Jayaraman, and Kristen Grauman. Emergence of exploratory look-around behaviors through active observation completion. *Science Robotics*, 4(30):eaaw6326, 2019. doi: 10.1126/scirobotics.aaw6326. URL `https://www.science.org/doi/abs/10.1126/scirobotics.aaw6326`.

Martin Riedmiller, Roland Hafner, Thomas Lampe, Michael Neunert, Jonas Degrave, Tom van de Wiele, Vlad Mnih, Nicolas Heess, and Jost Tobias Springenberg. Learning by playing solving sparse reward tasks from scratch. In Jennifer Dy and Andreas Krause (eds.), *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4344–4353. PMLR, 10–15 Jul 2018. URL `https://proceedings.mlr.press/v80/riedmiller18a.html`.

Alexander Rivkind, Or Ram, Eldad Assa, Michael Kreiserman, and Ehud Ahissar. Visual hyperacuity with moving sensor and recurrent neural computations. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=p0rCmDEN_-`.

Sasha Salter, Dushyant Rao, Markus Wulfmeier, Raia Hadsell, and Ingmar Posner. Attention-privileged reinforcement learning. *Conference on Robot Learning (CoRL)*, 2020. URL `https://arxiv.org/abs/1911.08363`.

J. Shi, J. Z. Woodruff, and P. B. Umb. The shadow dextrous hand. `https://www.shadowrobot.com/products/dexterous-hand/`, 2011. [Online; accessed 24-January-2020].

David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In Eric P. Xing and Tony Jebara (eds.), *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pp. 387–395, Bejing, China, 22–24 Jun 2014. PMLR. URL `https://proceedings.mlr.press/v32/silver14.html`.

Ivan Sorokin, Alexey Seleznev, Mikhail Pavlov, Aleksandr Fedorov, and Anastasiia Ignateva. Deep attention recurrent q-network. *eep Reinforcement Learning Workshop, NIPS*, 2015. URL `https://arxiv.org/abs/1512.01693`.

Richard S Sutton, Joseph Modayil, Michael Delp, Thomas Degris, Patrick M Pilarski, Adam White, and Doina Precup. Horde: A scalable real-time architecture for learning knowledge from unsupervised sensorimotor interaction. In *The 10th International Conference on Autonomous Agents and Multiagent Systems-Volume 2*, pp. 761–768, 2011. URL `https://aamas.csc.liv.ac.uk/Proceedings/aamas2011/papers/A6_R70.pdf`.

Andrew Szot, Alexander Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Singh Chaplot, Oleksandr Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in Neural Information Processing Systems*, 34, 2021. URL `https://arxiv.org/abs/2106.14405`.

Yaniv Taigman, Ming Yang, Marc'Aurelio Ranzato, and Lior Wolf. Deepface: Closing the gap to human-level performance in face verification. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1701–1708, 2014. doi: 10.1109/CVPR.2014.220.

Yujin Tang, Duong Nguyen, and David Ha. Neuroevolution of self-interpretable agents. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, pp. 414–424, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371285. doi: 10.1145/3377930.3389847. URL `https://doi.org/10.1145/3377930.3389847`.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5026–5033. IEEE, 2012. URL `https://ieeexplore.ieee.org/document/6386109`.

Adam White, Joseph Modayil, and Richard S Sutton. Surprise and curiosity for big data robotics. In *Workshops at the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 2014.

Haiping Wu, Khimya Khetarpal, and Doina Precup. Self-supervised attention-aware reinforcement learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. URL `https://www.aaai.org/AAAI21Papers/AAAI-1137.WuH.pdf`.

Jianwei Yang, Zhile Ren, Mingze Xu, Xinlei Chen, David J Crandall, Devi Parikh, and Dhruv Batra. Embodied amodal recognition: Learning to move to perceive objects. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2040–2050, 2019. URL `https://ieeexplore.ieee.org/document/9008379`.