Eye, Robot: Learning to Look to Act with a BC-RL Perception-Action Loop

Justin Kerr Kush Hari Ethan Weber Chung Min Kim Brent Yi
Tyler Bonnen Ken Goldberg Angjoo Kanazawa
https://www.eyerobot.net/
UC Berkeley

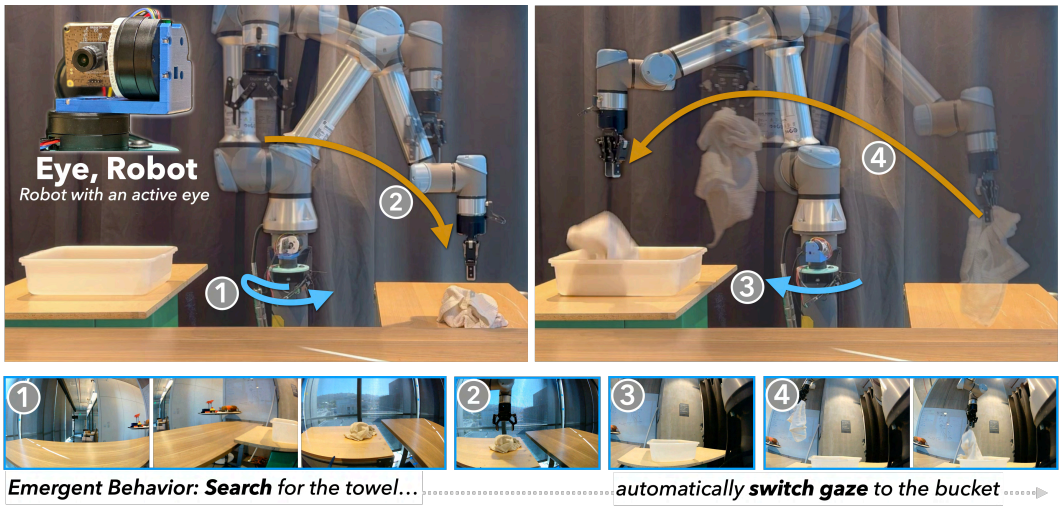

Figure 1: **EyeRobot.** We present a robotic system with an active eye, where the behavior of looking emerges from the need to act. A foveated mechanical eye, inspired by biological vision, is trained via reinforcement learning in a real-to-sim BC-RL loop. Shown here is a long-horizon pick-and-place task involving a towel and a bucket—neither visible in the initial view. The robot looks for the towel, grasps it, then switches gaze to the bucket to complete the task. This behavior emerges purely from being rewarded for looking in directions which facilitate action, without any gaze demonstration.

"We perceive in order to act and we act in order to perceive." — *JJ Gibson*

Abstract: Humans do not passively observe the visual world—we actively look in order to act. Motivated by this principle, we introduce EyeRobot, a robotic system with gaze behavior that emerges from the need to complete real-world tasks. We develop a mechanical eyeball that can freely rotate to observe its surroundings and train a gaze policy to control it using reinforcement learning. We accomplish this by first collecting teleoperated demonstrations paired with a fixed 360° camera. This data is imported into a simulation environment that supports rendering arbitrary eyeball viewpoints, allowing episode rollouts of eye gaze on top of robot demonstrations. We then introduce a BC-RL loop to train the hand and eye jointly: the hand (BC) agent is trained from rendered eye observations, and the eye (RL) agent is rewarded when the hand succeeds. In this way, hand-eye coordination emerges as the eye looks towards regions which allow the hand to complete the task. EyeRobot uses a foveated vision transformer architecture, allowing high resolution with a small compute budget, which we find leads to the emergence of stable eye fixation as well as improved ability to track objects and ignore distractors. We evaluate EyeRobot on five panoramic workspace manipulation tasks requiring manipulation in an arc surrounding the robot arm. Experiments suggest

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea.

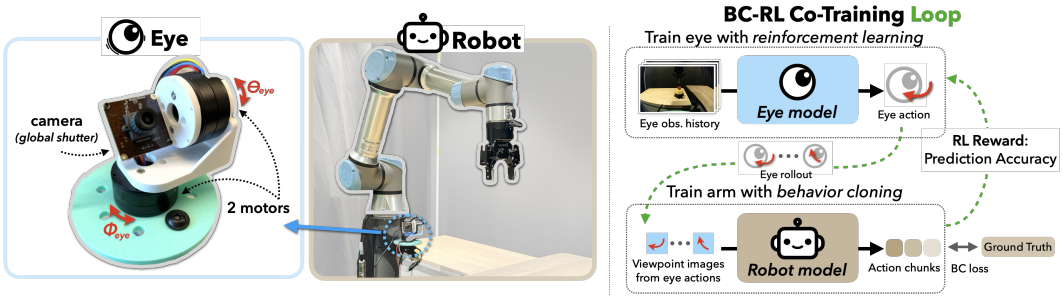

Figure 2: **EyeRobot framework.** Left: We develop a mechanical eyeball with two degrees of freedom, mounted on a high-speed gimbal and equipped with a fisheye lens and global shutter. Right: The eye policy is trained via a BC-RL loop which allows gaze to emerge to facilitate action.

EyeRobot exhibits hand-eye coordination behaviors which effectively facilitate manipulation over large workspaces with a single camera.

1 Introduction

Are you thirsty? Take a moment to reach for the nearest cup. As you do, your eyes move first—scanning the table, darting from region to region to locate the cup. Once it's in sight, your hand follows. This tight coupling between attention and eye movements is not incidental; it reflects a fundamental constraint of the human visual system. We are not built to perceive everything at once. As a result, we must look around—and in this sense, vision can be understood as a form of search. But what drives the search? The need to *act*—to accomplish something in the world. Whether gathering information about task-relevant properties or guiding the execution of a precise action, we look because we have something to do.

In this work, we present a robotic system where the same principle—looking to enable action—emerges naturally from a desired real world task without the need for gaze demonstrations, implemented on a mechanical eyeball that can freely rotate to observe its surroundings. The core challenge is how to train such a visual agent within the constraints of the physical world? To address this, we introduce a behavior cloning (BC) - reinforcement learning (RL) loop enabled by a 360° camera real-to-sim environment. This does not require any gaze demonstrations; instead, hand-eye coordination emerges solely from task supervision.

To specify what "action" we desire from the system, we build on recent advances in policy learning from behavior cloning [1]. We augment an existing teleoperation system to collect synchronized 360° video and robot trajectory data, creating an EyeGym environment that enables replay of demonstrations with renderings from simulated eyeball viewpoints. Equipped with this environment, we propose a BC-RL algorithm for training an RL eye policy with a task-based reward, which samples rollouts from the current eye agent to use as observations to supervise a behavior cloning agent for the arm. The accuracy of these action predictions are cyclically used as rewards for the eye agent (Fig. 2). As these agents co-train, the eye thus learns to look around to optimize the performance of the behavior cloning agent—gaze emerges from the need to act. Figure 1 illustrates an example of search behavior which emerges in EyeRobot to accomplish a long-horizon pick-and-place task involving a towel and a bucket—neither of which is visible in the initial camera view.

Inspired by nature, we design a Foveal Robot Transformer (FoRT) architecture which processes visual input in a foveated manner: peripheral vision provides broad, low-resolution coverage of the visual field, while foveal vision offers high-resolution input over a restricted area. Our experiments show this multi-resolution architecture results in enhanced fixation during task execution, distance invariance while tracking objects, and better ability to ignore distractors. Our experiments evaluate EyeRobot on 5 large, panoramic-workspace manipulation tasks, involving objects on a 180° area surrounding the arm. Across these tasks we observe a number of behaviors not explicitly trained

for—gaze switching between targets depending on task stage, long-range search, and independently coordinated hand-eye movements. We find that EyeRobot enables promising performance on a variety of large-workspace pick-and-place and servoing tasks with *only* a single egocentric active camera. We compare to externally and wrist-mounted cameras, and find that EyeRobot outperforms them in this large workspace either due to the limited resolution of the zoomed-out perspective or the inability of the wrist camera to search.

2 Related Work

Active Vision Active perception systems physically move sensors to not only see, but to *look* [2, 3, 4, 5]. This arises naturally from physical constraints faced by embodied agents, which have only partial observations of the world at any point in time. The key insight of active vision is that actuation lets agents shape the utility of these observations to better achieve their goals. Existing systems primarily use active vision to maximize information gain for visual tasks: examples include tracking [6, 7], search [8, 9, 10], observation completion or reconstruction [11], and view selection for 3D reconstruction [12, 13, 14, 15, 16] using voxel [17], surfel [18], or point cloud [19] representations. Other works couple active sensing setups with robot manipulators and evaluate systems in terms of downstream task performance: this includes improvements in semantic understanding [20], computational efficiency [21], information seeking [22], and occlusion robustness [23] for robot manipulation. Like these systems, EyeRobot also studies active vision for robot manipulation. Instead of relying on heuristics [20], camera action demonstrations [21, 23], or ground-truth gaze or state however, EyeRobot proposes to learn optimal eye gaze policies directly in observation space with reinforcement learning; we learn where to look in order to act.

Behavior Cloning Behavior cloning (BC) [24, 25, 26, 27] is the dominant paradigm for teaching robots manipulation skills. BC is advantageous because it does not require hand-designed behaviors, costs, and rewards—instead, BC policies are simply trained to imitate human demonstrations. Prior work has shown how this can enable new capabilities in mobile [28, 29], bimanual [1, 30, 31, 32], and language-conditioned [33, 34, 35, 36] robot manipulation. Policies of this form can also be implemented using a diverse range of architectures, including energy-based [37], diffusion [35], and autoregressive [38, 36]. In EyeRobot, we adopt a similar imitation-based system for manipulation in large workspaces. In contrast to prior systems that solely focus on learning from demonstration, however, EyeRobot's BC-RL training shows how behavior cloning objectives for robot actions can (i) be optimized jointly with an eye gaze reinforcement learning objective and (ii) used as a reward itself for reinforcement learning.

Biology-Inspired Machine Perception Many computer vision systems draw inspiration from biology to improve efficiency, adaptability, and performance. For example, simple stereo cameras [39, 40, 41] mimic the binocular vision of animals to support depth perception, while event cameras [42, 43, 44] mimic retinal spikes to enable low-latency perception. Foveated systems emulate the resource-rational nonuniformity of the retina, using either specialized hardware [45, 46, 47] or learned neural mechanisms [48, 49]. Beyond sensing, gaze control has been modeled after oculomotor behaviors like smooth pursuit and saccades [50], and implemented in hybrid pipelines that combine peripheral detection with foveal tracking [51]. Given plentiful simulation, prior works have found foveation emerges when evolving a visual sensor to complete classification or survival tasks [52, 53]. EyeRobot builds on the same principles as these prior works, drawing on the benefits of foveation and gaze control. However, its implementation differs significantly: rather than rely on specialized sensors or hardcoded behaviors, we mount a standard RGB camera on a high-speed gimbal, apply foveation using a multi-resolution transformer, and learn a gaze policy via reinforcement learning with a real-to-sim gaze rendering pipeline.

3 Approach

EyeRobot trains gaze policies for manipulation using reinforcement learning. We conduct experiments using a UR5e robot arm with a gimbal-mounted camera mounted rigidly to the base of the

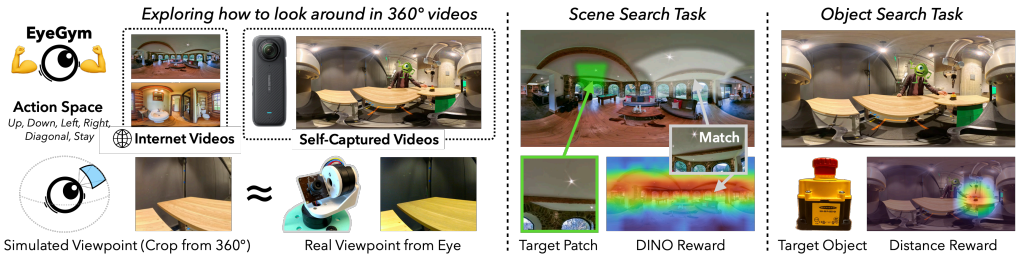

Figure 3: **Learning to Look with EyeGym.** EyeGym enables training policies on 360° internet images or robot data to perform semantic visual search tasks.

robot. To train gaze policies for hand-eye coordination, we present (i) EyeGym, an environment for eye gaze simulation, and (ii) reinforcement learning methodology—visual search rewards and a joint BC-RL loop—for learning gaze policies.

3.1 Scalable Experience Collection with EyeGym

Reinforcement learning benefits from scalable experience collection. To facilitate this for eye gaze policies, we introduce EyeGym: an RL environment that simulates eye gaze by sampling from 360° image and video data. Unlike prior approaches that rely on synthetic environments and physics simulators to render simulated views [54, 55, 56, 57, 58, 59, 60], EyeGym renders directly by sampling from real equirectangular videos and images. This has several key advantages: (i) it reduces the sim2real gap by exposing policies to authentic textures, lighting, and noise; (ii) it enables training on native 360° datasets [61, 62, 63], making EyeGym practical and scalable for learning active visual perception policies that transfer to real-world camera systems (Fig. 3); and (iii) it incurs less overhead than traditional rendering. We will release EyeGym code to support further research.

We use EyeGym for robot manipulation by first replacing our robot's physical eyeball with an off-the-shelf Insta360 X4 360° camera. We can then use this camera to capture robot demonstrations teleoperated using a GELLO system [64], where the 5.7K, 30FPS equirectangular video sequences are recorded and synchronized with robot trajectories. Paired data can then be imported into EyeGym for simulating eye gaze on top of demonstrated robot motion. Advantages of this setup include minimal additional hardware, minimal additional data bandwidth, and compatibility with existing teleoperation systems [1, 64].

3.2 Learning Gaze Policies with BC-RL

The goal of EyeRobot is to learn gaze policies that improve downstream task performance. This is done without expert gaze demonstrations. We instead use the EyeGym environment to learn gaze policies from flexible reward signals: either task-driven BC-RL or pure visual search.

The BC-RL Loop The utility of gaze policies can ultimately be measured by downstream task performance. Therefore, we propose to optimize such policies with robot task metrics, rather than hand-designed rewards like semantic visual search or user-specified gaze demonstrations. We propose a BC-RL loop to achieve this. Given a BC policy controlling a robot arm and an RL policy controlling gaze, the key idea of BC-RL is that observations flow from the RL policy to the BC policy, while task success metrics flow from the BC policy to the RL policy. At every BC-RL optimization step, the eye policy attempts to optimize the BC agent's *current performance* at matching demonstrated actions, while the BC agent learns how to perform the task best given the current gaze behavior.

We operationalize BC-RL in EyeGym using the synchronized demonstration and 360° video pairs detailed in Section 3.1. One episode rollout consists of a video trajectory sampled from a demonstration; beginning with the eye gaze randomly initialized $\pm 90°$ and $\pm 15°$ from the neutral azimuth and elevation positions. At each step, the eye receives a reward comparing the predicted action chunk

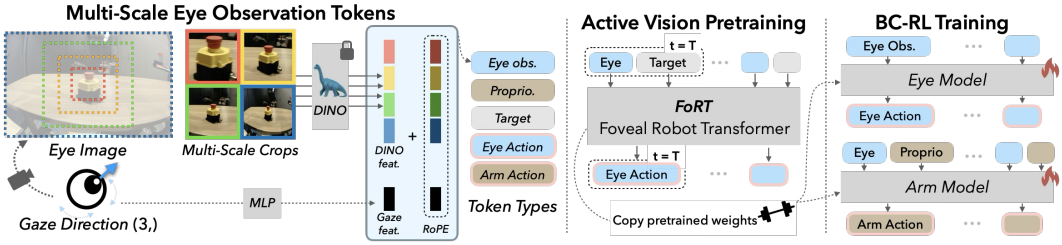

Figure 4: **Foveal Robot Transformer (FoRT)**. Eye observations are processed in a *foveal* manner, where each image is processed at multiple scales and concatenated together with the gaze direction.

to the ground-truth action chunk. We use a reward based on the action chunk's forward-kinematics, which is better normalized than joint-space error. Specifically, we compute the end effector position over predicted and ground-truth trajectories, and assign reward to be the negative Fréchet distance between these splines (penalizing deviation). At the beginning of each demonstration, we pause time for 30 frames without BC supervision, to allow the eye to visually search before advancing time in the video. The BC agent is trained with a standard L1 loss between predicted and ground truth chunks.

Active Visual Pretraining (AVP) Randomly initializing networks during BC-RL results in an ill-conditioned BC objective, as often times the target object is invisible given random eye movements. We thus use a visual object search reward as a pretraining task to initialize network weights and ease BC-RL convergence. We pretrain policies in EyeGym using randomly sampled static 360° video frames, allowing the eye to look around and rewarding it when the object is centered in view. During pretraining, we condition on CLIP embeddings of search targets to give networks the ability to learn about multiple objects; this input is replaced with zero tokens after pretraining. During the BC-RL phase, we initialize network weights of both the hand and eye with pretrained weights from AVP.

3.3 Foveal Robot Transformer (FoRT)

EyeRobot uses transformer architectures for its eye and robot policies (Fig. 4), which convert all observations to tokens and predicts all outputs as tokens. The eye and robot policies share projection matrices for shared inputs, but otherwise use separate transformer weights.

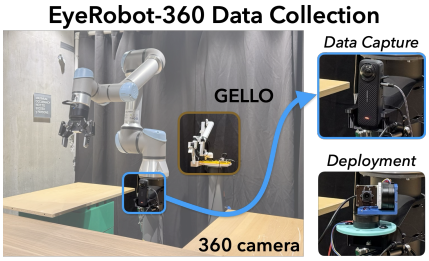

Figure 5: **Teleop Data Setup with EyeRobot.** We collect actions with GELLO [64], and the EyeGym environment with a time-synced 360 camera.

Observation Feature Extraction Though sophisticated mechanisms for multi-resolution foveation have been proposed in prior vision work [48, 65], we wish to leverage pretrained vision backbones and thus opt for a simpler architecture involving multi-cropping. We process input images into an image pyramid of crops with N scales centered at the center pixel, rescaling all images to the same 224 resolution. We embed all patches independently with a frozen DINOv2-ViT/S [66] encoder, and flatten them token-wise as inputs to the transformer. Each policy additionally takes as input the eye gaze direction as a 3D vector, the current joint proprioception, and an optional "target" token for visual search. Each are projected to the input token dimension with small MLPs. We apply 10% dropout on the proprioceptive tokens to avoid overfitting.

Outputs Action outputs are decoded from the transformer with lightweight projection heads. Eye actions are parameterized as a categorical distribution over 8 azimuth-elevation directions and $\vec{0}$. We use a learnable input token for each output type, which is shared across timesteps in the output action chunk.

4 Experiments

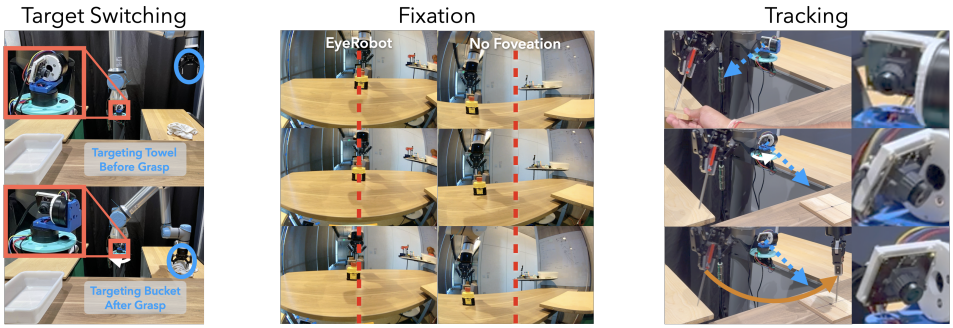

Figure 7: **Emergent Eye Behavior.** (Left) The eye learns to shift its gaze from towel to bucket during grasping depending on the state of the arm. (Middle) Foveation causes object-centering fixation to arise. (Right) Gaze follows target objects independently of the hand.

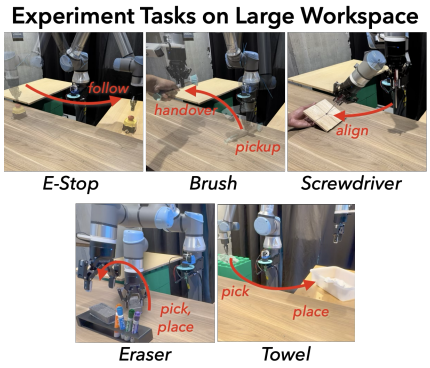

Figure 6: **Tasks.** We evaluate EyeRobot on 5 large-workspace tasks.

Our experiments aim to (1) evaluate the ability of using RL to visually search in scenes and find objects, (2) compare EyeRobot's performance on manipulation tasks to conventional camera placements, and (3) investigate emergent properties of the hand-eye coordination learned during BC-RL.

4.1 Using EyeGym for Visual Search

Here we perform an RL-only experiment to evaluate the effectiveness of the EyeGym at training an eyeball to scan the environment for a desired object.

Visual Search Rewards Search is a critical step of vision for robot manipulation, especially for large workspaces. As an initial gaze policy study, we evaluate two search tasks with explicit visual rewards. *(a) Scene Search* policies attempt to locate image patches that are visually similar to a given target patch. For this, we use DINO feature similarity between current and target views as a reward signal. See the Appendix for results and discussion. *(b) Object Search* policies are more fine-grained, and aim to locate specific objects within scenes. For Object Search, we primarily investigate a "truncated distance reward". This reward is zero if the target object is outside the camera's field of view (FOV) and increases linearly to 1 as the agent perfectly centers the target in its view.

Object Search We define a task where the goal is to locate and track target object(s). We investigate this setting using self-captured videos from robot demonstrations, train with the "Truncated Distance" reward, and deploy the learned policy to the physical eyeball. We train the robot to locate the towel and assess its ability to find it. The towel is deliberately placed outside the initial field of view, requiring the eyeball to actively explore the scene. We achieve a success rate of 87% with an average search time of 1.8 seconds, with failures occurring when the towel lies near the far edges of the workspace. To investigate qualitative tracking speed capabilities, we also train on a simple stuffed bear, and find the eye can track even rapid movements like tossing, see project site for videos.

4.2 Hand-Eye Coordination

We evaluate EyeRobot on 5 tasks (Fig. 6) which involve manipulation over a 210° workspace, to probe the limits of hand-eye coordination over a large region. All experiments are performed on a UR5e with a Robotiq gripper. Data was collected using GELLO [64], totaling 100-500 demos for each task. This workspace represents a significant challenge: some tasks require tolerance on the order of cm, while the workspace is on the order of $1000cm^2$. For all trials, the robot is pro-

Table 1: **Camera Comparisons**. We report success rate (Eraser & E-Stop) and distance to target (E-Stop).

Task	EyeRobot	Exo	Wrist	Wrist+Exo	Task	EyeRobot	Exo	Wrist	Wrist+Exo
Eraser (Pick & Place)	60%	0%	**100%**	60%	Eraser (Perturb Place)	**100%**	-	10%	40%
E-Stop (Slow)	**100%**	**100%**	80%	**100%**	E-Stop (Fast)	**100%**	**100%**	60%	**100%**
E-Stop (Slow)	**4.0cm**	7.8cm	5.3cm	7.1cm	E-Stop (Fast)	**4.7cm**	9.9cm	**4.7cm**	5.5cm

grammatically reset to the same pose (shared across ablations and baselines). We evaluate on 5 large-workspace manipulation tasks: eraser-on-shelf, e-stop reaching, brush handoff, screwdriver servoing, and towel-in-bucket. See Appendix for details on these task definitions.

Results Results are reported in Tables 1, 2, 3 and results are best viewed in the included execution videos to better understand qualitative performance. EyeRobot can consistently perform manipulation tasks over a large workspace, For the **Towel** task, Most of EyeRobot's failures are in narrowly missing grasps on the towel, and we notice it tends to struggle more in switching gaze from bucket to towel, as evidenced by worse performance in trials where only the bucket is visible. Sometimes it grasps only a corner of the towel, leading to a difficult bucket drop where half the towel dangles outside the bucket. In our **E-Stop** trials, EyeRobot never loses track of the object, and its main error is in z-distance towards the eyeball, owing to the difficulty of resolving depth with a monocular viewpoint. In the **Eraser** task, EyeRobot primarily fails by narrowly missing grasps on the eraser. In the place only trials where the eraser begins in the same location each time, EyeRobot can robustly follow the perturbed location of the shelf post-grasp. In **Screwdriver**, EyeRobot achieves a mean error of 4.0cm when the target is flat, and 5.2cm when the target is tilted $45°$, compared to a total test area spanning 115cm left to right. The **Brush** task successfully grasps the brush at the correct orientation 15/20 trials, and successfully completed the human handover 14/15 times.

4.2.1 Eye Behavior

We observe three key behaviors that the eye learns while being rewarded for enabling action: switching, search, and independent tracking. In multi-step tasks, the eye learns to automatically switch its gaze towards the next relevant object, depending on the state of the robot (Fig 1). For long-horizon tasks this requires that the eye search for the subsequent target objects when it is not in view, or even just make smaller fixation adjustments for objects that are only partially visible (Fig 7). Its search strategy tends to oscillate back and forth in a sweeping motion. When the target location is moved mid placement (e.g., 'perturb' condition in Table 2) the eye tracks the new target location, independently of the robot arm. Finally, in more dynamic tasks, we note that the eye learns to attend to predictive cues in the environment (e.g., humans placing a target object). Taken together, these qualitative results suggest the potential of an active vision agent trained with RL to naturally complement a behavior cloning agent in achieving tasks.

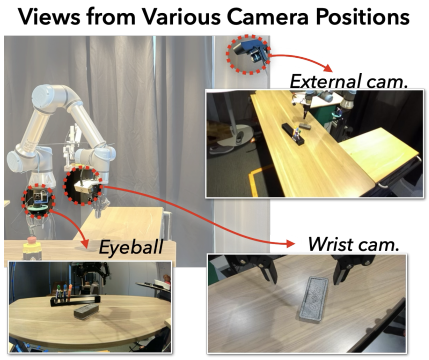

Figure 8: **Effect of Camera Placement.** Placing the camera on a gimbal allows it to observe across the whole workspace at a higher resolution.

4.2.2 Wrist, Exo Comparisons

An important question is how EyeRobot compares to more conventional camera placements, namely fixed externally-mounted, and wrist-mounted cameras (Fig. 8). We perform comparisons on the E-Stop and Eraser tasks, by training on *the exact same data* with the same architecture. For exo-only and wrist-only baselines, we input images at 640 resolution, while for wrist+exo we use 360. Both comparisons have *more* input image tokens than FoRT, and the same number of model parameters.

Results are reported for the eraser and e-stop tasks in Table 1. When the wrist camera can view the eraser, the wrist camera achieves a very high success rate, although this performance greatly

suffers in the perturbation trials where the wrist cannot search. In contrast, EyeRobot can adjust its viewpoint to maintain the shelf in view at all times. The exo camera baseline struggles to achieve precise enough grasps given its low resolution; while often touching the eraser, it never successfully grasps it. The exo+wrist baseline also performs worse at servoing compared to wrist-only, likely due to the complexity of merging image tokens from multiple cameras–though it can occasionally succeed in locating the perturbed shelf.

4.3 Ablations

Foveation: To understand the contribution of foveated inputs on EyeRobot's behavior, we train and evaluate a BC-RL gaze model that operates over a uniform image resolution. To control for the number of image tokens we use inputs of 1x448x448 instead of 4x224x224. We find the foveated model consistently maintains the target object near the center of its field of view, whereas the non-foveated model only loosely keeps it within view (Fig. 7). Quantitatively, this uniform image resolution leads to degraded performance on all metrics evaluated for E-Stop, while settling significantly slower due to its unstable viewpoint (Table 2).

Table 2: **Foveation Ablation.** (Error ↓ / Speed ↓)

Task	EyeRobot	No Foveal
E-Stop (Slow)	**4.0cm / 4.2s**	5.9cm / 5.3s
E-Stop (Fast)	**4.7cm / 5.8s**	6.9cm / 7.1s
Average	**4.4cm / 5.0s**	6.4cm / 6.2s

Table 3: **AVP Ablation.** (Success Rate)

Visible At Start	EyeRobot	No AVP
Both Visible	**80%**	73%
Towel Visible	**95%**	40%
Bucket Visible	50%	**70%**
Neither Visible	**60%**	**60%**
Average	**72.2%**(±13.1%)	**62.1%**(±14.2%)

The foveated model is also significantly more robust to distractor objects; when a human places a distractor (yellow cup) in the scene, the uniform resolution model tends to split attention between the two and pays more attention to the human hand, and by slowly moving the cup one can force the robot to completely lose the E-Stop. On the other hand, the foveated model is able to ignore this distractor and remains fixated on the E-Stop (see videos). These data suggest potential performance benefits in adopting a foveal architecture for manipulation owing to the physical token distribution making it easier to attend to the right region.

Foveation also results in significantly improved robustness to objects' distance during visual tracking. This phenomenon is best seen in experiment videos; while slowly moving the stuffed bear away from the eyeball, the uniform resolution model quickly loses track after about 2 meters as the bear vanishes from input tokens, whereas the foveated model can maintain tracking much farther (around $5\times$) because its pyramidal representation maintains tokens visually recognizable as the bear.

Active Visual Pretraining: We ablate AVP on the towel task, and find that long-range visual search can emerge *purely* from task driven BC-RL. We note, however, that grasping performance suffers compared to the model with AVP, which we attribute partially to poorly initialized network weights and partially to poor training data distribution early in BC-RL training, as the eye fails to fixate correctly at initialization. Convergence speed of BC-RL is also aided by AVP as it initializes gaze looking towards task relevant objects.

5 Conclusion

EyeRobot presents a method for training a mechanical eyeball to look around to achieve physical robot manipulation via a real-to-sim-to-real pipeline utilizing 360° videos. This simulation environment allows the training of active visual policies with RL on top of teleop demonstrations, with which we train a visual agent to look around to maximize task-based BC performance. We find this agent learns to look to facilitate action, resulting in emergent eye behaviors such as search and fixation, and that the eye enables manipulation across a large workspace. In addition, we find that physically allocating tokens towards target objects via foveation results in improved robustness to distractors as well as increased robustness to distance while maintaining a cheap compute budget.

6 Limitations

The primary limitation of EyeRobot is the inability of 360° video to simulate motion parallax, i.e, how a neck can provide the ability to move around occlusions for a better perspective. Another drawback of EyeRobot is its eagerness to learn strategies which match simulation very well, but fail in real due to narrow data distributions. Concretely, one behavior we observe in the towel task is "blind grasping", where the robot will sometimes look left, and upon observing no towel, grasps an average location to the right. This arises from the demonstration data distribution, owing to the fact there are fewer demos at the edges of the workspace. Finally, training BC-RL converges significantly slower than vanilla BC. This is because of the co-training of two models which are mutually used in the others' train loop. While our current system is limited to a stationary workstation, an exciting future direction could be to mount the eyeball on a mobile robot, further increasing the need for active vision. Likewise, adding a second eye for stereo vision or incorporating monocular depth estimators could help overcome current limitations in depth perception and support more fine-grained manipulation.

7 Acknowledgements

This research was performed at the AUTOLAB at UC Berkeley in affiliation with the Berkeley AI Research (BAIR) Lab, which is supported in part by donations from Bosch, Google, Autodesk, and Siemens. Justin Kerr, Kush Hari, and Chung Min Kim are supported by the National Science Foundation Graduate Research Fellowship Program under Grant No. DGE 2146752. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation. The authors were supported in part by equipment grants from NVIDIA. We deeply appreciate Jon Kuroda, Ion Stoica, and Joey Gonzalez for their generous support with computing hardware during some downtime on compute.

References

[1] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning fine-grained bimanual manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.

[2] R. Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988.

[3] R. Bajcsy, Y. Aloimonos, and J. K. Tsotsos. Revisiting active perception. *Autonomous Robots*, 42:177–196, 2018.

[4] K. Y. Goldberg and R. Bajcsy. Active touch and robot perception. *Cognition and Brain Theory*, 7(2):199–214, 1984.

[5] J. Aloimonos, I. Weiss, and A. Bandyopadhyay. Active vision. *International journal of computer vision*, 1:333–356, 1988.

[6] N. P. Papanikolopoulos, P. K. Khosla, and T. Kanade. Visual tracking of a moving target by a camera mounted on a robot: A combination of control and vision. *IEEE transactions on robotics and automation*, 9(1):14–35, 1993.

[7] N. Papanikolopoulos, P. K. Khosla, and T. Kanade. Vision and control techniques for robotic visual tracking. In *ICRA*, pages 857–864, 1991.

[8] Y. Zhu, R. Mottaghi, E. Kolve, J. J. Lim, A. Gupta, L. Fei-Fei, and A. Farhadi. Target-driven visual navigation in indoor scenes using deep reinforcement learning. In *2017 IEEE international conference on robotics and automation (ICRA)*, pages 3357–3364. IEEE, 2017.

[9] X. Ye, Z. Lin, H. Li, S. Zheng, and Y. Yang. Active object perceiver: Recognition-guided policy learning for object searching on mobile robots. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6857–6863. IEEE, 2018.

[10] E. Kolve, R. Mottaghi, W. Han, E. VanderBilt, L. Weihs, A. Herrasti, M. Deitke, K. Ehsani, D. Gordon, Y. Zhu, et al. Ai2-thor: An interactive 3d environment for visual ai. *arXiv preprint arXiv:1712.05474*, 2017.

[11] D. Jayaraman and K. Grauman. Learning to look around: Intelligently exploring unseen environments for unknown tasks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1238–1247, 2018.

[12] R. Zeng, Y. Wen, W. Zhao, and Y.-J. Liu. View planning in robot active vision: A survey of systems, algorithms, and applications. *Computational Visual Media*, 6(3):225–245, Sep 2020. ISSN 2096-0662. doi:10.1007/s41095-020-0179-3. URL https://doi.org/10.1007/s41095-020-0179-3.

[13] J. E. Banta, L. Wong, C. Dumont, and M. A. Abidi. A next-best-view system for autonomous 3-d object reconstruction. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 30(5):589–598, 2000.

[14] M. Mendoza, J. I. Vasquez-Gomez, H. Taud, L. E. Sucar, and C. Reta. Supervised learning of the next-best-view for 3d object reconstruction. *Pattern Recognition Letters*, 133:224–231, 2020.

[15] E. Smith, D. Meger, L. Pineda, R. Calandra, J. Malik, A. Romero Soriano, and M. Drozdzal. Active 3d shape reconstruction from vision and touch. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P. Liang, and J. W. Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 16064–16078. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/8635b5fd6bc675033fb72e8a3ccc10a0-Paper.pdf.

[16] W. Jiang, B. Lei, and K. Daniilidis. Fisherrf: Active view selection and uncertainty quantification for radiance fields using fisher information. *arXiv*, 2023.

[17] S. Isler, R. Sabzevari, J. Delmerico, and D. Scaramuzza. An information gain formulation for active volumetric 3d reconstruction. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3477–3484, 2016. doi:10.1109/ICRA.2016.7487527.

[18] R. Monica and J. Aleotti. Surfel-based next best view planning. *IEEE Robotics and Automation Letters*, 3(4):3324–3331, Oct 2018. ISSN 2377-3766. doi:10.1109/LRA.2018.2852778.

[19] R. Zeng, W. Zhao, and Y.-J. Liu. Pc-nbv: A point cloud based deep network for efficient next best view planning. *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7050–7057, 2020.

[20] V. Sripada, S. Carter, F. Guerin, and A. Ghalamzan. Ap-vlm: Active perception enabled by vision-language models. *arXiv preprint arXiv:2409.17641*, 2024.

[21] X. Cheng, J. Li, S. Yang, G. Yang, and X. Wang. Open-television: Teleoperation with immersive active visual feedback. *arXiv preprint arXiv:2407.01512*, 2024.

[22] S. Dass, J. Hu, B. Abbatematteo, P. Stone, and R. Martín-Martín. Learning to look: Seeking information for decision making via policy factorization, 2024. URL https://arxiv.org/abs/2410.18964.

[23] I. Chuang, A. Lee, D. Gao, M. Naddaf-Sh, I. Soltani, et al. Active vision might be all you need: Exploring active vision in bimanual robotic manipulation. *arXiv preprint arXiv:2409.17435*, 2024.

[24] A. Billard, S. Calinon, R. Dillmann, and S. Schaal. Survey: Robot programming by demonstration. *Springer handbook of robotics*, 2008.

[25] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys*, 2017.

[26] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard. Recent advances in robot learning from demonstration. *Annual review of control, robotics, and autonomous systems*, 2020.

[27] D. A. Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in neural information processing systems*, 1, 1988.

[28] Y. Du, D. Ho, A. Alemi, E. Jang, and M. Khansari. Bayesian imitation learning for end-to-end mobile manipulation. In *International Conference on Machine Learning*, pages 5531–5546. PMLR, 2022.

[29] Z. Fu, T. Z. Zhao, and C. Finn. Mobile ALOHA: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. *arXiv:2401.02117*, 2024.

[30] T. Lin, Y. Zhang, Q. Li, H. Qi, B. Yi, S. Levine, and J. Malik. Learning visuotactile skills with two multifingered hands. *arXiv preprint arXiv:2404.16823*, 2024.

[31] J. Grannen, Y. Wu, B. Vu, and D. Sadigh. Stabilize to Act: Learning to coordinate for bimanual manipulation. In *CoRL*, 2023.

[32] C. Chi, Z. Xu, C. Pan, E. Cousineau, B. Burchfiel, S. Feng, R. Tedrake, and S. Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv:2402.10329*, 2024.

[33] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, et al. Rt-1: Robotics transformer for real-world control at scale. *arXiv preprint arXiv:2212.06817*, 2022.

[34] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, et al. Rt-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

[35] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, et al. $\pi 0$: A vision-language-action flow model for general robot control, 2024. *URL https://arxiv. org/abs/2410.24164*.

[36] G. R. Team, S. Abeyruwan, J. Ainslie, J.-B. Alayrac, M. G. Arenas, T. Armstrong, A. Balakrishna, R. Baruch, M. Bauza, M. Blokzijl, et al. Gemini robotics: Bringing ai into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

[37] P. Florence, C. Lynch, A. Zeng, O. A. Ramirez, A. Wahid, L. Downs, A. Wong, J. Lee, I. Mordatch, and J. Tompson. Implicit behavioral cloning. In *Conference on robot learning*, pages 158–168. PMLR, 2022.

[38] K. Pertsch, K. Stachowicz, B. Ichter, D. Driess, S. Nair, Q. Vuong, O. Mees, C. Finn, and S. Levine. Fast: Efficient action tokenization for vision-language-action models. *arXiv preprint arXiv:2501.09747*, 2025.

[39] D. Marr, T. Poggio, E. C. Hildreth, and W. E. L. Grimson. *A computational theory of human stereo vision*. Springer, 1991.

[40] T. Kanade, A. Yoshida, K. Oda, H. Kano, and M. Tanaka. A stereo machine for video-rate dense depth mapping and its new applications. In *Proceedings CVPR IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 196–202. IEEE, 1996.

[41] D. Scharstein and R. Szeliski. High-accuracy stereo depth maps using structured light. In *2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2003. Proceedings.*, volume 1, pages I–I. IEEE, 2003.

[42] H. Rebecq, R. Ranftl, V. Koltun, and D. Scaramuzza. High speed and high dynamic range video with an event camera. *IEEE transactions on pattern analysis and machine intelligence*, 43(6):1964–1980, 2019.

[43] G. Gallego, T. Delbrück, G. Orchard, C. Bartolozzi, B. Taba, A. Censi, S. Leutenegger, A. J. Davison, J. Conradt, K. Daniilidis, et al. Event-based vision: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(1):154–180, 2020.

[44] H. Kim, S. Leutenegger, and A. J. Davison. Real-time 3d reconstruction and 6-dof tracking with an event camera. In *European conference on computer vision*, pages 349–364. Springer, 2016.

[45] C. Bandera and P. D. Scott. Foveal machine vision systems. In *Conference Proceedings., IEEE International Conference on Systems, Man and Cybernetics*, pages 596–599. IEEE, 1989.

[46] S. Minut, S. Mahadevan, J. M. Henderson, and F. C. Dyer. Face recognition using foveal vision. In *Biologically Motivated Computer Vision: First IEEE International Workshop, BMCV 2000 Seoul, Korea, May 15–17, 2000 Proceedings 1*, pages 424–433. Springer, 2000.

[47] G. Killick, P. Henderson, P. Siebert, and G. Aragon-Camarasa. Foveation in the era of deep learning. *arXiv preprint arXiv:2312.01450*, 2023.

[48] A. Jonnalagadda, W. Y. Wang, B. Manjunath, and M. P. Eckstein. Foveater: Foveated transformer for image classification. *arXiv preprint arXiv:2105.14173*, 2021.

[49] B. Cheung, E. Weiss, and B. Olshausen. Emergence of foveal image sampling from learning to attend in visual scenes. *arXiv preprint arXiv:1611.09430*, 2016.

[50] E. Rivlin and H. Rotstein. Control of a camera for active vision: Foveal vision, smooth tracking and saccade. *International Journal of Computer Vision*, 39:81–96, 2000.

[51] S. Gould, J. Arfvidsson, A. Kaehler, B. Sapp, M. Messner, G. R. Bradski, P. Baumstarck, S. Chung, A. Y. Ng, et al. Peripheral-foveal vision for real-time object recognition and tracking in video. In *Ijcai*, volume 7, pages 2115–2121. Citeseer, 2007.

[52] K. Tiwary, A. Young, Z. Tasneem, T. Klinghoffer, A. Dave, T. Poggio, D.-E. Nilsson, B. Cheung, and R. Raskar. What if eye...? computationally recreating vision evolution, 2025. URL https://arxiv.org/abs/2501.15001.

[53] B. Cheung, E. Weiss, and B. A. Olshausen. Emergence of foveal image sampling from learning to attend in visual scenes. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017. URL https://openreview.net/forum?id=SJJKxrsgl.

[54] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. http://pybullet.org, 2016–2023.

[55] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, et al. Isaac gym: High performance gpu-based physics simulation for robot learning. *arXiv preprint arXiv:2108.10470*, 2021.

[56] E. Todorov, T. Erez, and Y. Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pages 5026–5033. IEEE, 2012.

[57] K. Zakka, B. Tabanpour, Q. Liao, M. Haiderbhai, S. Holt, J. Y. Luo, A. Allshire, E. Frey, K. Sreenath, L. A. Kahrs, et al. Mujoco playground. *arXiv preprint arXiv:2502.08844*, 2025.

[58] M. Savva, A. Kadian, O. Maksymets, Y. Zhao, E. Wijmans, B. Jain, J. Straub, J. Liu, V. Koltun, J. Malik, et al. Habitat: A platform for embodied ai research. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9339–9347, 2019.

[59] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets, et al. Habitat 2.0: Training home assistants to rearrange their habitat. *Advances in neural information processing systems*, 34:251–266, 2021.

[60] X. Puig, E. Undersander, A. Szot, M. D. Cote, T.-Y. Yang, R. Partsey, R. Desai, A. W. Clegg, M. Hlavac, S. Y. Min, et al. Habitat 3.0: A co-habitat for humans, avatars and robots. *arXiv preprint arXiv:2310.13724*, 2023.

[61] S.-H. Chou, C. Sun, W.-Y. Chang, W.-T. Hsu, M. Sun, and J. Fu. 360-indoor: Towards learning real-world objects in 360deg indoor equirectangular images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 845–853, 2020.

[62] M. Wallingford, A. Bhattad, A. Kusupati, V. Ramanujan, M. Deitke, A. Kembhavi, R. Mottaghi, W.-C. Ma, and A. Farhadi. From an image to a scene: Learning to imagine the world from a million 360° videos. *Advances in Neural Information Processing Systems*, 37:17743–17760, 2024.

[63] H.-N. Hu, Y.-C. Lin, M.-Y. Liu, H.-T. Cheng, Y.-J. Chang, and M. Sun. Deep 360 pilot: Learning a deep agent for piloting through 360 sports videos. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1396–1405. IEEE, 2017.

[64] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel. Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 12156–12163. IEEE, 2024.

[65] A. Deza and T. Konkle. Emergent properties of foveated perceptual systems. *arXiv preprint arXiv:2006.07991*, 2020.

[66] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, et al. Dinov2: Learning robust visual features without supervision. *arXiv preprint arXiv:2304.07193*, 2023.

[67] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.

[68] J. Dong, B. Feng, D. Guessous, Y. Liang, and H. He. Flex attention: A programming model for generating optimized attention kernels, 2024. URL https://arxiv.org/abs/2412.05496.

[69] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR, 2021.

[70] B. Shen, F. Xia, C. Li, R. Martín-Martín, L. Fan, G. Wang, C. Pérez-D'Arpino, S. Buch, S. Srivastava, L. Tchapmi, et al. igibson 1.0: A simulation environment for interactive tasks in large realistic scenes. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 7520–7527. IEEE, 2021.

[71] C. Li, F. Xia, R. Martín-Martín, M. Lingelbach, S. Srivastava, B. Shen, K. Vainio, C. Gokmen, G. Dharan, T. Jain, et al. igibson 2.0: Object-centric simulation for robot learning of everyday household tasks. *arXiv preprint arXiv:2108.03272*, 2021.

[72] M. Deitke, W. Han, A. Herrasti, A. Kembhavi, E. Kolve, R. Mottaghi, J. Salvador, D. Schwenk, E. VanderBilt, M. Wallingford, et al. Robothor: An open simulation-to-real embodied ai platform. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3164–3174, 2020.

[73] K. Ehsani, W. Han, A. Herrasti, E. VanderBilt, L. Weihs, E. Kolve, A. Kembhavi, and R. Mottaghi. Manipulathor: A framework for visual object manipulation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 4497–4506, 2021.

[74] M. Deitke, E. VanderBilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi. Procthor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022.

A Eye Hardware

The physical eye uses a 90fps, 1900x1200 global shutter RGB camera with a 110° FOV fisheye lens, which allows for wide periphery view, with smooth and unwarped images during rapid movements. The eye is mounted on a gimbal consisting of two direct-drive brushless DC motors independently controlling pan and tilt, which are capable of a stopped-to-stopped saccade speed of $20ms$ at a range of 60°. In practice, we limit and smooth the speed of motion during inference to minimize motion blur. The rapid motion capability of the eyeball aids sim2real transfer by reducing the gap between commanded and executed motion. CAD and hardware instructions will be made available.

B Task Descriptions

E-Stop Reaching: The robot must continuously servo its gripper to hover above the button at all times, tracking large perturbations up to 180°. We measure the settling time after each perturbation and metric error from tooltip to button. We perform one "slow" trial where the E-Stop is moved to 10 locations sequentially spaced 20° apart, and a "fast" trial where these same locations are tested in an adversarial pattern up to 90° apart.

Screwdriver Servoing: While grasping a 7cm screwdriver, the robot must servo to align its tip with the center of a piece of wood marked by a black dot, which can be oriented between flat on the table to 45° inclined. This task requires orientation control and servoing of a very thin (3mm) tool shaft. Similar to the E-stop task, we measure final screwdriver tip distance for this task.

Eraser on Shelf: The robot grasps and transports an EXPO eraser onto a shelf, whose position may be perturbed during execution. The orientation of both pick and place locations can be randomized $\pm 45^\circ$. We consider an experiment a success if the resting state of the eraser is fully supported by the shelf. We test on a static case, where both objects are static but their location varies 180° radially around the robot, and a case where objects begin centered but we perturb the shelf location post-grasp by 40cm left or right. This evaluates the policies' robustness to searching for the target place location.

Towel in Bucket: The robot must transport a towel from the tabletop to a bucket, *both of whose locations may vary* up to 180°. This task is the most difficult for visual search, as some cases (Fig. 1) begin with *neither* target object in view. We split evaluation into 4 tiers with 10 trials each, corresponding to which objects are initially in view. We pre-define locations for each tier covering the entire workspace, and initialize the eye facing forward on cases involving search. A trial succeeds if the towel is fully inside the bucket when the bucket is lifted by the experimenter, with half credit assigned if the towel is partially inside the bucket when lifted.

Brush Handoff: The robot grasps a brush and servos it towards a human, releasing its grasp when the human grasps the handle. The brush orientation may vary $\pm 90^\circ$, and its position is tested along a 75cm line in the center of the table. A handoff is considered successful if the robot picks up the brush, moves it within arms reach of the human, and releases its grasp once the human grasps the handle (but not before).

C Demonstration Data Details

All data, code, and data collection hardware will be made public.

C.0.1 Data Collection Methodology

We collect teleop demonstrations for servoing tasks (E-Stop, Screwdriver) in one continuous take, while a human moves the target periodically around. For the towel task, data is collected with uniformly randomized positions of the towel and bucket, with care to ensure they are each placed everywhere around the workspace. For the Eraser task, the location of the eraser relative to the shelf is maintained roughly the same within a 20x20cm square, while the global positions of both vary

Task	Number of Demos	Total Hours
Eraser	242	0.95
E-Stop	–	0.50
Screwdriver	–	0.71
Towel	599	1.69
Brush	265	0.83

Table 4: Dataset size of teleop demonstrations. Continuous servoing tasks like E-Stop and Screwdriver were collected in a few, long demos, hence we only report total time.

around the whole workspace. In half of our demos, the positions of one or both of the eraser and shelf are perturbed by a human during execution to provide more rich servoing data to the robot. For the brush task, the brush is uniformly randomized on the front of the reachable workspace, and at each demo the human waits for a random amount of time before grabbing the brush for hand-off. This encourages the robot to learn when to release the brush rather than prematurely dropping it.

C.0.2 Dataset Statistics

Table 4 shows the size of our teleop datasets. These numbers account for the total amount of post-processed "live" robot time, which the BC-RL agent sees during training.

D FoRT Implementation Details

D.0.1 Positional Embeddings

We use 3D rotary positional embeddings (RoPE) [67] to assign (t, x, y) coordinates to each token. We assign each image token its corresponding image pixel coordinate, which allows attention to seamlessly transfer from one crop within the image pyramid to others. This allows the transformer to re-use features across scales, which provides a mechanism for correlating observations of the same object across distances. For example, an object close-up fills the periphery, while an object far away fills the fovea, allowing distance to be recognized via spatial attention while underlying token embeddings themselves remain similar. All other non-image tokens are assigned $(\frac{I_W}{2}, \frac{I_H}{2})$ as their spatial coordinate to encourage preferential attention to the image center. Each t coordinate is assigned to the corresponding timestep of the observation or output, for example a proprioceptive observation token at time t would produce output joint tokens from t to $t + A_{\text{size}}$

D.0.2 Attention Masking and Memory

EyeRobot masks significant portions of self-attention to increase the throughput of inference and minimize overfitting. First, we mask all image-image attention since DINOv2 outputs have already undergone self-attention. Second, we utilize sliding window attention to train on long rollouts efficiently (up to 100 time-steps), with a window size of 10 for the eye to provide history of motion, and 1 for the hand to effectively make it single-frame. We use Flex Attention [68] to efficiently compute these custom masks.

E BC-RL Implementation Details

E.0.1 Optimization and Hyperparameters

We use PPO to optimize the eye agent during all training, with default parameters and an entropy regularization coefficient of 0.01. We use separate optimizers for the hand and eye policies, both of which are optimized with AdamW with default parameters except a β_2 coefficient of 0.95. The learning rate is initialized to $1e-3$ for the BC agent and $5e-4$ for the RL agent, to incentivize the RL agent to move slower so the BC agent has more time to converge. Both learning rates following

a cosine decay schedule to a $10\times$ lower LR, and are trained for 7M environment steps. We use an action chunk size of 30, corresponding to 1 second of real-world time. The BC agent outputs raw joint positions to execute (which are normalized to the range $[-1, 1]$), which are parameterized either by absolute or relative joint commands. We find that for tasks with persistent large motions such as the E-Stop, Towel, or Screwdriver, absolute actions out-perform relative as the latter tends to get "lost" in joint configuration space and has no data to recover from such states. In tasks with shorter horizon motion such as the Eraser and Brush, we find that relative actions can achieve more precise servoing performance.

E.0.2 EyeGym Rollouts

Each episode is rolled out for 130 steps, 30 of which are paused in the beginning to give the eye agent 1 second to explore and find relevant objects. The demonstration video is played at $2\times$ real-time speed to increase the diversity of batches seen by the BC agent. In effect, this means the agent sees 100 frames at 15fps, or 6.7 seconds of data each rollout. Each rollout is initialized at a random time-step from a randomly sampled demonstration, with an added buffer to ensure the rollout doesn't extend beyond the length of data available. We rollout 16 environments in parallel across GPUs, and use NVIDIA A6000s to train.

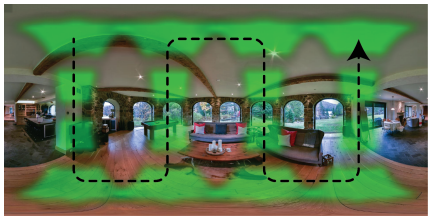

Figure 9: **S-shaped pattern for scene search evaluation.** We move a target across crops in a sweeping pattern to probe our scene search behavior.

E.1 Scene and Object Search

Scene Search We propose a scene-level semantic visual search task on 2,000 360° images from [61]. For each training episode, we select a randomly located target crop with a field of view between 10° and 65° and condition FoRT on pooled extracted DINO tokens. Evaluation is performed on 500 unseen images, each with 24 equally space target crops, and results are reported in Table 5. We move the target in an S-shaped pattern, and for each new location, we allow the policy 20 steps to move before evaluating CLIP [69] similarity and "Exact Match" rate, which marks how often it can put the target object within its field of view at the end of the rollout. We find the eye learns to look towards semantically similar viewpoints, but struggles to find exact matches due to the inherent repeating nature of many scenes.

Table 5: **Scene Search Results.**

Method	CLIP ↑	Exact Match ↑
Random Walk	0.629	28.1%
Distance Reward	0.704	64.8%
DINO Reward	0.715	60.2%
DINO+Distance	0.711	66.5%

We train policies with DINO feature similarity and truncated distance rewards and look for interesting emergent behavior. The results are best visualized in our video, where the policy learns to find semantically similar viewpoints. To test this behavior, we move a target through 360° images in an S-shaped pattern, as shown in Figure 9. The scene is divided into 24 crops arranged in a 4×6 grid (4 along elevation, 6 along azimuth), with each crop covering a 40° field of view. The target starts at the top left, and the policy has 20 steps to move around and look for similar-looking regions before the target moves to its next position, at which point it again has 20 steps. This setup allows us to observe visual search behavior. We evaluate this procedure across 500 images and report CLIP similarity and exact match rate, computed at the final step after the policy has used its 20 steps to move. Our numbers are in the main paper, where we compare our method variants against a random movement baseline. We chose the truncated distance reward for active visual pretraining since it leads to more search behavior, where the policy is only rewarded when it can find the target object.

For the object search experiment using our real camera to find the towel, we run 15 trials: 5 with the towel placed on the right, 5 in the center, and 5 on the left. In each case, we initialize the eyeball looking away from the towel to ensure a non-trivial starting point. The model successfully finds

the towel in 87% of the trials, as reported in the main paper, indicating it has learned a valid search behavior that can help initialize the policy for active vision pretraining in robotic manipulation tasks.

E.2 Robot Inference

When deploying on the robot, we predict eye and arm actions at 30Hz from the trained neural networks, and follow the temporal ensembling proposed in Zhao et al. [1] to interpolate robot actions. Arm and gripper actions are predicted as continuous positions, and a continuous servo loop commands the robot arm to follow these commands. The predicted eye actions are converted to motor velocities and smoothed with EMA before being sent to motor controllers. We use random sampling during eye inference, to lessen the distribution shift between train and test. In total, the inference control loop including all neural network passes runs at 30hz, and we employ temporal ensembling to interpolate the output actions. We use $k = 0.05$ for the exponential smoothing coefficient. During test rollouts, we keep the eye paused for 2 seconds before allowing the hand to move to allow the eye to search. We allow an extra 1 second during inference since in practice the physical eye moves slightly slower because of its inertia than the inertia-free simulated eye.

F Camera Comparison Details

Our camera comparisons use the same transformer-architecture as FoRT, except we remove all eye-related tokens and pass in a $2\times$ higher resolution (448) image to match the number of image tokens as FoRT. When two images (wrist and exo) are present, to enable the same batch size training we set both exo and wrist images to 360 resolution. Positional encoding for baseline image tokens is the same as for FoRT; with learnable query tokens assigned the x,y coordinates of the center of the image. When two images are present, we offset the x coordinates of the exo image to be side-by-side with the wrist image, and set the learnable coordinates to the center of the wrist image. For wrist and wrist+exo comparisons, we find that relative actions out-perform absolute due to the wrist's highly local viewpoint, and absolute actions outperform on the exo only comparison.

G Additional Related Work

Simulation environments are critical for modern robotics research, where physical experiments can be costly and difficult to reproduce. While many environments are focused on physics [54, 55, 56, 57], others are tailored to the embodied visual tasks central to active vision. Habitat renders photorealistic RGB–D scans of indoor scenes, enabling large-scale navigation and manipulation studies [58, 59, 60]. iGibson couples the Bullet physics engine with textured reconstructions of real homes and offices, offers domain randomization, and exposes interactive objects for embodied learning [70, 71]. AI2-THOR provides Unity-based apartments with rich object affordances [10]; its extensions add real-world counterparts for sim-to-real transfer [72], a tabletop manipulator arm [73], and large procedurally generated layouts [74]. We introduce EyeGym as an alternative environment for training physical eyes to look around in any 360 direction. Rather than rendering 3D reconstructions, EyeGym builds on the insight that camera observations can be simulated by sampling from any 360° image or video [63, 62]. This approach has two key advantages: (i) it minimizes the sim-to-real gap by exposing agents to real-world textures, lighting, and noise, and (ii) it enables large-scale pretraining using native 360° video datasets such as 360-1M [62], which contains one million panoramic YouTube videos with unconstrained egocentric camera motion. These properties make EyeGym practical and scalable for learning active visual perception policies that transfer to real-world camera systems.