

Eye, Robot: Learning Hand-Eye Coordination with Reinforcement Learning Supplementary Material

Anonymous Author(s)

Affiliation

Address

email

1 Video Results

Please see the attached supplement video for videos of all task executions, including eye-perspective videos, ablations, baselines, and scene-level search.

2 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.

3 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

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 1: Dataset size of teleop demonstrations

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).

4 Demonstration Data Details

All data will be made public upon acceptance.

4.0.1 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 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.

4.0.2 Dataset Statistics

Table 1 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.

5 BC-RL Implementation Details

All code will be made public upon acceptance.

5.0.1 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.

5.0.2 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.

5.1 Scene and Object Search

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 view-points. To test this behavior, we move a target through 360° images in an S-shaped pattern, as shown in Figure 1. 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.

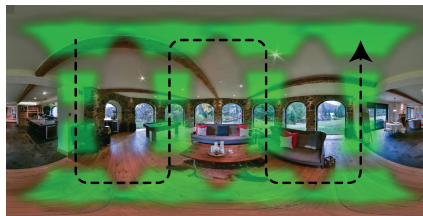


Figure 1: **S-shaped pattern for scene search evaluation.** We move a target across crops in a sweeping pattern to probe our scene search behavior.

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.

5.2 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.

5.3 FoRT Positional Embedding

We use 3D rotary positional embeddings [2] to assign (t, x, y) coordinates to each token. We assign each image token its corresponding image pixel, which allows attention to seamlessly transfer from one crop within the image pyramid to others. All other 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 $t + A_{\text{size}}$

6 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 relative due to the wrist’s highly local viewpoint, and absolute actions outperform on the exo only comparison.

7 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 [3, 4, 5, 6], 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 [7, 8, 9]. iGibson couples the Bullet physics engine with textured reconstructions of real homes and offices, offers domain randomization, and exposes interactive objects for embodied learning [10, 11]. AI2-THOR provides Unity-based apartments with rich object affordances [12]; its extensions add real-world counterparts for sim-to-real transfer [13], a tabletop manipulator arm [14], and large procedurally generated layouts [15]. 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 [16, 17]. 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 [17], 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.

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] J. Su, M. Ahmed, Y. Lu, S. Pan, W. Bo, and Y. Liu. Roformer: Enhanced transformer with rotary position embedding. *Neurocomputing*, 568:127063, 2024.
- [3] E. Coumans and Y. Bai. Pybullet, a python module for physics simulation for games, robotics and machine learning. <http://pybullet.org>, 2016–2023.
- [4] 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.

- [5] 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.
- [6] 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.
- [7] 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.
- [8] 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.
- [9] 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.
- [10] 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.
- [11] 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.
- [12] 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.
- [13] 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.
- [14] 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.
- [15] M. Deitke, E. Vanderbilt, A. Herrasti, L. Weihs, K. Ehsani, J. Salvador, W. Han, E. Kolve, A. Kembhavi, and R. Mottaghi. Proctor: Large-scale embodied ai using procedural generation. *Advances in Neural Information Processing Systems*, 35:5982–5994, 2022.
- [16] 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.
- [17] 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.