Towards Generalized Dynamic Catching: Learning to Catch with a Low-Cost Robot Hand

Abstract— **Humans are capable of performing dynamic and agile maneuvers for manipulation with ease. Replicating this skill on anthropomorphic robot hands has proven challenging, with robotic hand policies failing far below human performance. As modern approaches often fail to generalize to novel objects, rely on static grasps, or rely on full state information, anthropomorphic manipulation remains an open challenge in robotics. To solve this problem, traditional methods include computing grasps from leveraging large data sets or optimizing force based closure to achieve a stable grasp. However, analytical techniques lack adaptation to unseen circumstances at deployment and do not learn how to change under these conditions. In this work, a monolithic framework is deployed in two phases to create an adaptive policy which can generalize to novel thrown objects. As prior works have explored static grasps, this work investigates the problem of dynamic catching in order to show the effectiveness of adaptation and ability to quickly respond under conditions of latency. We show this initial result in simulation on robotic arms equipped with a low cost anthropomorphic hand. The policy is trained in two phases with phase one leveraging privileged observations of incoming objects and phase 2 using only depth image observations. The resulting policy can generalize to novel objects and with high force closure across a variety of tosses.**

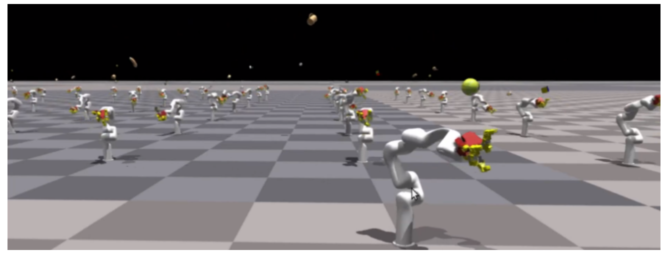

Fig. 1: IsaacGym tossing environment: 8194 robot arms trained in series using PPO to create a generalized grasping policy on training and novel objects with randomized tosses and randomized objects. Xarm6 and LeapHand deployed in sim with resulting X percent accuracy and high performance to novel objects.

I. INTRODUCTION

The ability to successfully complete dynamic tasks is a fundamental aspect of human motor skills, enabling humans to seamlessly interact with the surrounding environment. In cases of dynamic manipulation, humans rely on quick reaction times and fine control of human hands to react to real-time changes. These tasks include dynamic catching [1], tossing [2], batting [3], flinging [4], and similar fast-paced tasks. Replicating this capability in robots has been challenging, given the complexities of real-time perception, precise coordination, and dynamic adaptation required for high success [3] [5]. In order to further the ability of robots to perform dynamic tasks under these limitations, this work introduces a generalized framework for catching using adaptive reinforcement learning in simulation.

Advances in robotic manipulation have explored a foundation of manipulation methods for static and dynamic grasps. These methods include exploring catching as a dynamical system [6] and using multi agent reinforcement learning to train catching and throwing handovers [7]. Additional methods have explored how to overcome limitations in catching by investigating assistive catching tools such as nets [8] or simplifying the task to traditional robotic grippers [9]. While these works provide an initial exploration, ongoing work is needed further generalize to novel objects, work under limited world information, and adapt to wider variety of catches with anthropomorphic hands. To investigate generalization and adaption, this work deploys an adaptive reinforcement model utilizing Proximal Policy Optimization [10] to adapt in real-time to incoming objects and readjust grasps for more stable and secure grasps.

Likewise, as modern anthropomorphic robotic hands are less nimble and less dexterous than human hands, recent advances in robotic hardware have explored how to enable greater dexterity at a lower cost. From expanding off prior models of robotics hands such as Allegro[11], a low cost robotic manipulator LEAP [12] hand was introduced which outperforms Allegro on strength, versatility, and robustness. This work leverages this advancement through deploying a LEAP hand in simulation. Anthropomorphic robotic hands have been investigated due to their potential to achieve a wide range of complex tasks with variable objects. However, compared to conventional industrial grippers, the complex solution space of robotic hands has led to challenges in terms of their operational speed and dexterity [13]. With many solutions and high degrees of freedom existing, robotic hands often fail to quickly execute generalized tasks and often rely on extensive full-state information [14] [15] [16]. To work towards deployment in the real world with low-cost vision and enable a simple setup for generalization to new tasks, our method utilizes limited object observations with stereo egocentric vision.

In this paper, we study generalized catching with anthropomorphic robotic hands using egocentric vision through deploying an adaptive policy. This work investigates how to enable catching with randomized throws between .5 and 9 m/s, reaction times, latency, across thirty novel and training objects. The policy is deployed in IsaacGym [17] to leverage large-scale information gained in simulation.

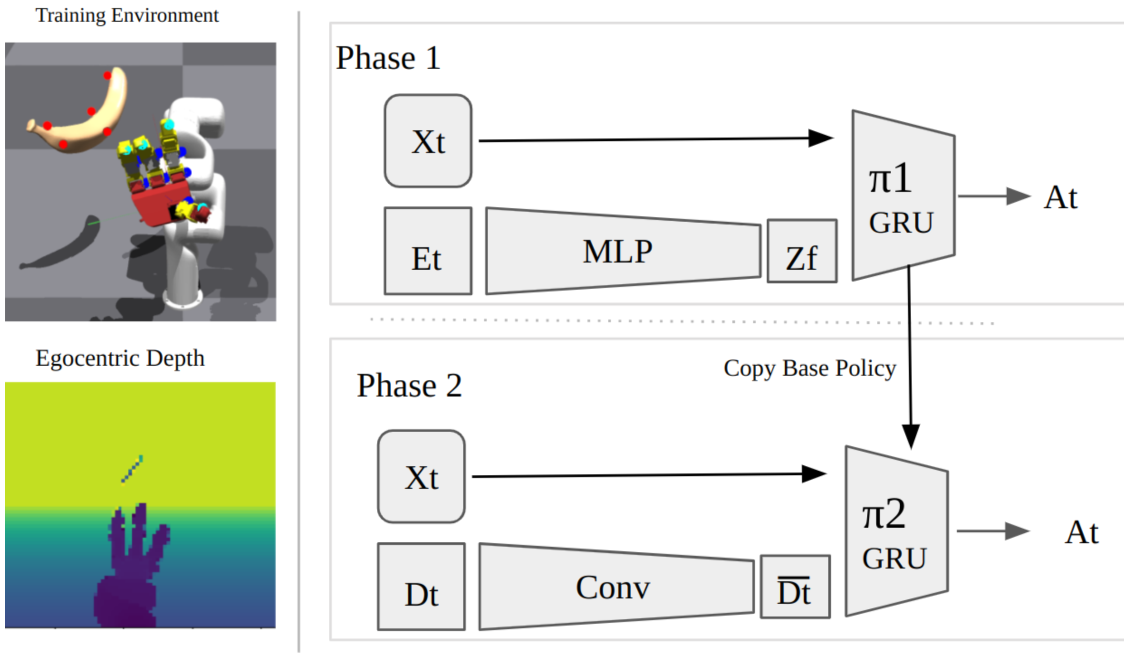

Fig. 2: The training pipeline deployed in IsaacGym. Inputs: Object Pose ($R \epsilon 3$), Object Linear Velocity ($R \epsilon 3$), Robot States ($R \epsilon 26$) as privileged information, Et, and state information, Xt. The policy uses is Proximal Policy Optimization in combination with GRU. This approach allows for real time adaption which allows the policy to readjust grasp position post catch and to re-catch rebounds or recover from near catches.

II. Related Works

Prior work on robotic hands includes generalizing grasps for variable objects, including tool use, generalized grasping, and autonomous grasping. Approaches have included collecting datasets of human grasps and finding corresponding robotic grasps [14], using video learning [18], and mapping [16]. These approaches have shown that grasping across objects is feasible but may require large datasets. Likewise, several hardware improvements have been made to anthropomorphic hands. To maximize dexterity and create a low-cost solution, the LEAP hand [12] was introduced, enabling finer dexterity compared to similar counterparts, including Allegro.

Dynamic Tasks Catching has recently become more popular in robotics literature. Early work on catching with anthropomorphic robotic hands by Kim et al. [6] utilized dynamical systems to map projectiles of uneven objects alongside human demonstrations. While this work was pivotal in introducing catching with anthropomorphic hands, the controller created has limitations in adaptation and relied on high-speed cameras. More recent work in robotic catching utilized a Multi-Agent framework to create a tossing and catch policy for two robot arms trained together [7]. This framework resulted in small catches and tosses between each robot across similar ball-shaped objects. This framework showed real-time reinforcement policies could be used in the real world and overcome Sim2Real challenges.

Beyond robotic hands, catching has been explored through net catching. In 2019, using a robot dog and net, Bryner et al. [8] used event-based cameras to match projectiles of incoming objects. This work relied on a known environment for high-speed objects and utilized IsaacGym [17] for simulation. In order to likewise leverage large-scale data collection from simulation, this work similarly relies on Isaac. More classical approaches often relied on control-intensive and predictive modeling of projectiles; however, these works fail to adapt dynamically at test time.

Learning Dexterity In studying robotic hands for for real world In studying robotic hands for real-world grasping, several robotic hands have been deployed and tested. These hands include Allegro [11], designed as a low-cost alternative to Shadow Hand [19]. In OpenAI et. al., ShadowHand successfully completed dynamic rotation tasks of a Rubik's Cube through a learned policy [20]. Similarly, learned dexterity has been shown through applying rapid motor adaptation using Allegro [21]. In order to expand on learned dexterity, Shaw et al. developed a more dexterous open-source robotic hand [12], which has successfully demonstrated dexterity across objects.

As robotic hands have a complex solution space, several works have explored solving grasps through data. By using teleoperated hands, several works have improved the sample efficiency of human data [22], [23] and tested the generation of robotic grasps from human data [18] [14]. However, these works rely on static grasps with objects remaining at a fixed state.

III. Methodology

The objective is to create a generalized and adaptive catching policy that uses minimal real-world object observations with high performance under high noise and latency, enabling successful deployment in the real world. IsaacGym allows for widescale data collection with information that mimics real robot configurations, and using limited information for input allows the policy to be deployed zero-shot without needing real-world learning. By implementing an additional object and tossing curriculum, the policy first learns a grasping policy and then transitions to arm control with a generalized grasping policy.

This work first deploys in simulation using IsaacGym with a custom environment to leverage large scale training and assist with sim to real. By training with ranges of high latency and noise on the robot position, end effector, and object position, the policy adapts to real-time observations, including readjustment of the hand and quick reaction to the changing environment.

To leverage privileged observations, a wide variety of enviromental observations are used in phase 1. These include simulated gravity, latency, object scale, friction, object and robot mass, object angular and linear velocity, object bounding box, fingertip to object distance, and palm to object vector. State information of the robot's end effect or and joint positions are fed as Xf from Figure 2. This is used to train a base policy using the same rewards that are also used in Phase 2. Once Phase 1 in trained, the base policy is frozen and Phase 2 is executed using depth from egocentric vision.

In Phase 2, The policy trains from a single camera behind the robot. As the robot hand is not equiped with force sensors, this trains using vision only until a satisfactory accuracy is reached. Phase 2 was trained using Isaacgym's built in depth estimation with camera observations of 1920 by 1080 and field of view of 100 degrees to mimic the real set stereo camera. At this point, it is ready to deploy in real once hyperparameters sim to real are tuned.

Rewards Custom rewards were created to optimize catch success, generalization to novel objects, and ensure force closure through experimental testing in simulation. These rewards include distance to palm, fingertip to object distance, force closure, palm contact, major axis of object to palm orientation, and costs on objects caught outside of the inside of the hand. These rewards act together to ensure tight grasps and stable by ensuring the object center of mass is close to the center of the palm of the hand. Objects outside of the robot's reach range or on the floor resulted in a auto reset. Force closure was calculated by finding distance and angle from evenly displayed points on the projectile to points on the robot's fingers and force applied. Actions were outputted in joint space as commands for the robotic hand and arm, and controls were computed using inverse kinematics.

Curriculums The model deploys an object curriculum along with a throwing curriculum to assist with training and generalization to novel objects. The object curriculum on initialization starts with a random chance between ball and cuboid objects. After a determined percentage of tosses are completed for each set of objects, an additional object is added, progressing from simple to complex objects. Likewise, the throw curriculum sets an increasing range of tosses based on the percentage of epoch tosses completed. This range increases until the full robot's reach range is met.

Reward	Scale
alive	1
episode_length	3.0
toques	-1e-8
palm_to_obj_perp	1
action_rate	-2e-6
outer_hand_contact	-0.25
fingertip_palm_norm	1
force_closure	.2
fingertip_obj_dist	-1
obj_palm_dist	-2

TABLE I: Rewards used for tuning catching.

Policy In order to work in the real world, noise was added to all observations. In particular, noise was added to latency (computed at 190 ms), object pose, end effector start, robot rotation, and start noise. This led to initial simulation results below. The policy utilized is PPO with an LSTM network. 8194 environments were used, with 400 epochs used at training time.

IV. Results

Initial results are created in simulation using IsaacGym on a simulated Xarm and LeapHand. All objects, masses, tosses, object scale, and velocities are randomized, with noise and latency added to all observations. The resulting policy predictively tracks incoming objects and adapts in real time to optimize catch success. In training and in test, arm and hand joints are constrained by real-world joint limits, velocities, and inertia.

The IsaacGym environment initializes a random object, toss, and end position for the toss within the robot's maximum reach range of 0.1 to 0.45 meters forward from the robot base, -0.4 to 0.4 meters in range, and 0.2 to 0.55 meters high. Tosses were generated in this range to avoid joint locking in real and operating outside of the feasible work range for the robot.

For training, 12 objects were selected for variable throwing behavior based on variable center of mass, aspect ratio, and grasp difficulty. Additionally, 18 novel objects were tested at test time. Each environment initializes a random toss between 0.5 to 9 m/s based on the randomized initial throw position. Each toss starts at a minimum of 0.5 meters forward from the robot and 0.25 meters above the ground. At a maximum, tosses are generated 2.75 meters away and 2 meters high at the start. Tosses range between 0 and 150 degrees and have an initial angular velocity between 0 and 40 degrees per second. Both an object curriculum and throw curriculum were used for all environments, with 8,194 environments with 400 epochs at 60 steps each. Training time for the policy averaged 30 minutes using GTX 3090 GPU. At test time,

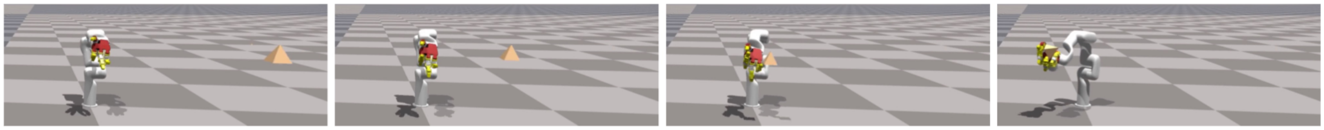

Fig. 3: A fast catch with a low throw. Object thrown at an initial velocity of 8 m/s to a position on the low side of the robot's reach range.

the policy is executed on 64 environments with randomized noise, tosses, objects, and latency. Benchmarked tosses are a set of 25 randomized tosses saved and tested through training and test time.

Objects at a higher velocity (8-10 m/s) had the lowest success rate, with a slow decline between objects at slow to fast velocities. The accuracy averages 87 percent across tosses for known training objects. Tosses across objects remained nearly constant for the training set. On novel objects, accuracy drops depending on how similar the novel test objects are to the test set of objects. Across object scale, success remained constant with a slight drop-off for objects above 1.5x scale due to the objects being too large for the hand to catch above that range.

As only the object pose is specified for each observation, the policy performs similar grasps for each object. In simulation videos, the hand sometimes adjusts moments after the catch to better grasp the object for full closure. On smaller objects, grasps fully close around the object. However, as the object scale was randomized, on large objects, the hand is forced to morph to the object during the catch. Accuracy remained nearly constant across more complex objects.

While successes were high at X percent, failure cases occurred X percent of the time, typically in the form of objects slipping out of the hand during attempted catches. This also includes object ricochets, which were less often but occurred for very fast and long throws (8-10 m/s). The policy performed X well across novel and X percent on training objects but with high variance between objects. Irregular and thin objects were the hardest to catch. However, the policy generalized to symmetrical objects and objects with a variable center of mass, including flashlights.

When initialized in Overhand mode, the policy had X percent success with higher success than underhand on fast moving objects and objects at the top of the robot's reach range. Underhand performed X percent success on training objects for comparison.

V. Discussion and Limitations

Current policies have high accuracy, but in a real-world setup, it can be difficult to determine from a depth map alone which object to track in such short time frames. Future work will continue to analyze methods for reducing noise, latency, and testing in the real world. Current policies execute in overhand and underhand modes with

In future work, we plan to test the setup on the real world with the target hardware described above. Additionally, we plan to further implement methods for sim-to-real transfer for egocentric tracking and test additional tasks, including overhand throws. We may also explore the incorporation of privileged information with masking to ensure proper catching in the real world. Future work will encompass adding additional tasks, expanding the methodology, and testing generalization across a wider range of dynamic manipulation tasks.

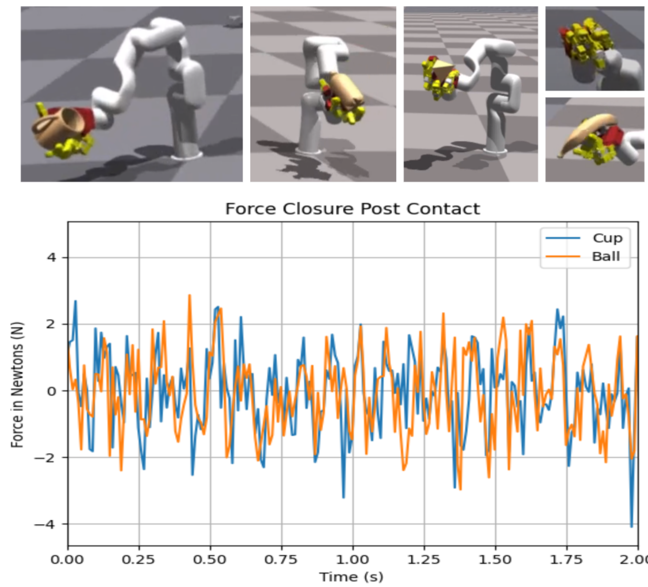

Fig. 4: Example grasps on mixed novel and training objects and force closure applied after catching each item.

Acknowledgment

A portion of this work is supported by the NSF Graduate Research Fellowship under Grant No. DGE2140739.

References

[1] A. Namiki, Y. Imai, M. Ishikawa, and M. Kaneko, "Development of a high-speed multifingered hand system and its application to catching," in *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, vol. 3, 2003, pp. 2666–2671 vol.3.

[2] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," 2019.

[3] F. Ruggiero, V. Lippiello, and B. Siciliano, "Nonprehensile dynamic manipulation: A survey," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1711–1718, 2018.

[4] H. Ha and S. Song, "Flingbot: The unreasonable effectiveness of dynamic manipulation for cloth unfolding," in *Conference on Robotic Learning (CoRL)*, 2021.

[5] A. Agarwal, A. Kumar, J. Malik, and D. Pathak, "Legged locomotion in challenging terrains using egocentric vision," in *Conference on Robot Learning*. PMLR, 2023, pp. 403–415.

[6] S. Kim, A. Shukla, and A. Billard, "Catching objects in flight," *Robotics, IEEE Transactions on*, vol. 30, pp. 1049–1065, 10 2014.

[7] B. Huang, Y. Chen, T. Wang, Y. Qin, Y. Yang, N. Atanasov, and X. Wang, "Dynamic handover: Throw and catch with bimanual hands," 07 2023.

[8] S. Bryner, G. Gallego, H. Rebecq, and D. Scaramuzza, "Event-based, direct camera tracking from a photometric 3d map using nonlinear optimization," 07 2019.

[9] B. Hove and J.-J. E. Slotine, "Experiments in robotic catching," in *1991 American Control Conference*, 1991, pp. 380–386.

[10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[11] W. Robotics, "Allegro, the allegro logo, roboticslab," 2008-2023.

[12] K. Shaw, A. Agarwal, and D. Pathak, "Leap hand: Low-cost, efficient, and anthropomorphic hand for robot learning," 07 2023.

[13] K. Shaw, S. Bahl, and D. Pathak, "Videodex: Learning dexterity from internet videos," in *Conference on Robot Learning*. PMLR, 2023, pp. 654–665.

[14] R. Wang, J. Zhang, J. Chen, Y. Xu, P. Li, T. Liu, and H. Wang, "Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation," *arXiv preprint arXiv:2210.02697*, 2022.

[15] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, "Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 5954–5961.

[16] W. Wei, P. Wang, and S. Wang, "Generalized anthropomorphic functional grasping with minimal demonstrations," 03 2023.

[17] 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.

[18] A. Sivakumar, K. Shaw, and D. Pathak, "Robotic telekinesis: Learning a robotic hand imitator by watching humans on youtube," 2022.

[19] "Shadowhand," https://ninjatek.com/shop/edge/.

[20] I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, A. Paino, M. Plappert, G. Powell, R. Ribas, *et al.*, "Solving rubik's cube with a robot hand," *arXiv preprint arXiv:1910.07113*, 2019.

[21] V. Sládek, J. Macháček, C. B. Ruff, E. Schuplerová, R. Přichystalová, and M. Hora, "Population-specific stature estimation from long bones in the early medieval pohansko (czech republic)," *American journal of physical anthropology*, vol. 158, no. 2, pp. 312–324, 2015.

[22] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang, "Dexmv: Imitation learning for dexterous manipulation from human videos," *arXiv preprint arXiv:2108.05877*, 2021.

[23] A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, and S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations," *arXiv preprint arXiv:1709.10087*, 2017.