Learning to Catch Reactive Objects with a Behavior Predictor

Kai Lu[1], Jia-Xing Zhong[1], Bo Yang[2], Bing Wang[2], Andrew Markham[1]

Abstract*— Tracking and catching moving objects is an important ability for robots in a dynamic world. Whilst some objects have highly predictable state evolution e.g., the ballistic trajectory of a tennis ball, reactive targets alter their behavior in response to motion of the manipulator. Reactive applications range from gently capturing living animals such as snakes or fish for biological investigations, to smoothly interacting with and assisting a person. Existing works for dynamic catching usually perform target prediction followed by planning, but seldom account for highly non-linear reactive behaviors. Alternatively, Reinforcement Learning (RL) based methods simply treat the target and its motion as part of the observation of the world-state, but perform poorly due to the weak reward signal. In this work, we blend the approach of an explicit, yet learned, target state predictor with RL. We further show how a tightly coupled predictor which 'observes' the state of the robot leads to significantly improved anticipatory action, especially with targets that seek to evade the robot following a simple policy. Experiments show that our method achieves an 86.4% (open plane area) and a 73.8% (room) success rate on evasive objects, outperforming monolithic reinforcement learning and other techniques. We also demonstrate the efficacy of our approach across varied targets and trajectories. All code, data, and additional videos: https://kl-research.github.io/dyncatch.

I. INTRODUCTION

Reactive object catching with a mobile robot finds numerous applications, from capturing living objects e.g., snakes in the wild or a fish in a domestic setting, to delivering tools to humans. However, despite recent advancements in embodied mobile manipulation [1]–[6], there is limited research on catching reactive targets. This task poses significant challenges for mobile manipulators, given the rapid and less-predictable movements of such targets (see Fig. 1).

Common methods for dynamic catching often combine a pose predictor and a motion planner, such as ball-catching algorithms [7]–[14] and real-time grasping approaches [14], [15], but they usually rely on known trajectories [7]–[12] or assume objects that follow fixed paths [14], [15]. These trajectories can be fitted by parameterized functions or learned by only observing the sequential object position over a few time-steps before planning. However, these methods rarely consider reactive behaviors, where the object alters its trajectory in response to the motion of the robot itself. This leads to a coupled problem which is challenging to model with classical techniques.

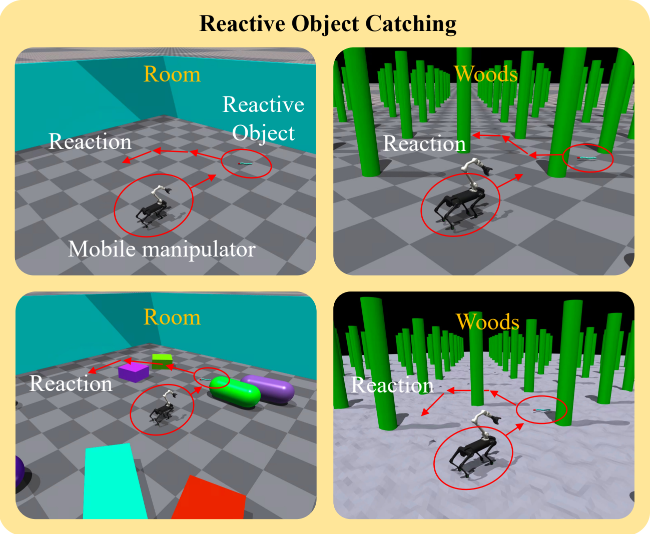


Fig. 1. **Reactive object catching for mobile manipulator.** In this work, we study a challenging task for a robotic mobile manipulator. Our robot learns to catch reactive objects e.g., evasive animals that have rapid and less-predictable movements in relation to the dynamics of the robot.

Reinforcement Learning (RL) methods have recently been studied in reactive object pursuit and catching [16]–[21], where a robotic control policy is learned from interactions. However, RL-based catching also struggles with rapid reactive behaviors, where the time-horizon of future predictability is small in relation to the speed of the robot. To address these challenges, most works choose to simplify the robotic platform to non-holonomic circles [16], [21], overlook visual latencies [17] and limit object movements to a small area [18], [20]. In these cases, they can learn the robotic catching policy by using monolithic RL, demonstrating that pursuit-evasion can be tackled in a data-driven manner. However, these assumptions limit their potential for application to embodied mobile manipulators as the dynamics of the robot itself (e.g., its ability to turn rapidly) impact the policy that it needs to learn.

In this work, we propose a prediction-based RL method with a learned behavior predictor for reactive object pursuit and catching. Instinctively, understanding a target's reactive movements (i.e., what will happen next) before making control decisions (i.e., what to do) should enable robots to learn the good policies for catching more effectively.

As we have no access to the internal state of the target, we have to infer and learn the relevant states of the target (e.g., future velocity) through observation. Rather than doing this explicitly using a classical pose predictor with knowledge of the target state-space model, we implement this as a learnable

[1]: K. Lu, J.-X. Zhong, and A. Markham are with the Department of Computer Science, University of Oxford, Oxford, UK. {kai.lu, jiaxing.zhong, andrew.markham}@cs.ox.ac.uk

[2]: B. Yang is with vLAR Group, Department of Computing, and B. Wang is with Department of Aeronautical and Aviation Engineering, Hong Kong Polytechnic University, HKSAR. {bo.yang, bing.wang}@polyu.edu.hk

state predictor. We further note that as the target alters its actions in response to the robot's actions, if we supply the known state of the robot into the learnable state predictor we obtain greatly improved predictions (see Fig. 2). In this way, we are learning the model/policy of the target. For simplicity, we trained this first through robot-object interactions, leaving the catching policy learning to the second stage.

The catching task is then treated as an embodied hierarchical policy learning problem. The high-level RL policy is learned by taking both the current observations and the predicted behavior of the object as input, and then generating robot moving commands to pursue and catch the target object. The low-level joint control is pre-trained using another RL model on uneven terrains and is not changed afterwards.

We evaluate the effectiveness of our approach in Isaac Gym environment [22] with reactive dynamic objects and a quadruped robot with a robotic arm. We achieve a success rate of 86.4% on catching evasive objects in the open plane setting and 73.8% in the room setting, outperforming the monolithic reinforcement learning method and the classical predictor-based methods. Furthermore, we demonstrate the effectiveness of our methods in different targets and trajectories including bouncing and fixed paths.

In summary, the main contributions of our work are:

- We learn to predict the reactive behavior of the target using a learning-based state prediction model. This is then enhanced using a tightly-coupled predictor which takes in knowledge of the robot's current positional state.
- We propose a prediction-based RL approach for dynamic catching with a mobile robotic manipulator.
- We introduce a new challenging task for embodied mobile manipulation research. To the best of our knowledge, we are the first to study this task using RL, and our approach shows superiority across different environments and targets. We also believe our method can be applied to many RL-based dynamic manipulation tasks.

II. RELATED WORK

Traditional approaches for dynamic object catching utilize object trajectory predictions for robotic planning [7]–[9], [11], [13], [23], [24]. However, the predictive model is usually learned or estimated from passive observations since the high-speed objects are not evasive from catching. In the tasks of catching a flying ball [7], [12], [13], [23], [24] or tool [11], high-speed external cameras [23], [24] or optical tracking systems [7] are often used to capture the object and fit it into a ballistic trajectory. Learning-based approaches exploit linear parameter varying (LPV) systems to predict the full trajectory [12], or use deep learning techniques such as visual encoder-decoder to predict future location [25]. Therefore, recent works typically focus on the visual tracking of the objects using high-speed event-cameras [23], [24], and then fed the estimated poses to the target predictor.

On the other hand, in the relevant robotic grasping area, there are research works on grasping dynamic objects [14], [15], [18], [20], [26], [27]. They propose to learn grasping skills for the fixed-base robotic arms with grippers or dexterous hands by adversarial learning [27], inverse RL [18], and meta-controller learning [15]. However, some of them consider relatively slow scenarios [26] and small movements [18], [20], while the others are limited to fixed or known trajectories [14], [15], [27].

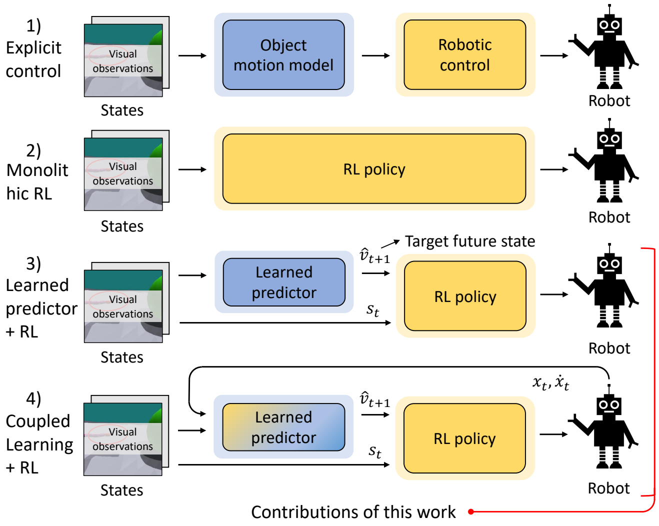


Fig. 2. **Various methods for dynamic object catching for robots.** 1) Classical pose predictor with object motion model followed by an off-the-shelf robotic planner and controller. 2) Monolithic RL generates robotic actions from current observations. 3) Learned predictors with visual observations of targets followed by an RL policy. 4) (Ours) Learned predictors leveraging both visual cues and robot states to predict target behaviors e.g., $\hat{v}_{t+1}$, combined with an RL policy taking both current observations and future prediction before making a decision.

Recently, embodied policy learning research has demonstrated significant progress in enabling robots to interact more naturally and effectively with their environment [28]. Key achievements include learning about the properties of objects through interactions and applying this knowledge to manipulate various rigid objects [29] and articulated objects [1]–[3]. Latest works learn to predict object motion flow [2], [3] from visual observations, but they usually study non-reactive objects. Other related topics are navigation [6], and legged robot locomotion [30], [31], but they either focus on a different task such as the navigation goal is a static position, or concentrate on whole-body control [4], [30], rather than the reactive moving targets. Therefore, although there are many mobile manipulation benchmarks proposed in recent years [5], [6], [32]–[34], the task of tracking and reaching the reactive and fast-moving targets has been less explored.

Another related area is pursuit-evasion games (PEG) [35], which usually focuses on solving optimal pursuit in the worst case of evasive motions. Deterministic solutions such as pursuit curve analysis and differential games require high abstraction of the problems [36], while heuristic methods [37] usually use assumptions of oracle information of the evaders. RL methods [16], [38] for pursuit consider more robotic constraints, but also simplify the robot and targets to non-holonomic circles [16] or points [21], and neglect visual delays [38]. These methods are limited for whole-body robot catching tasks due to their abstractions and assumptions.

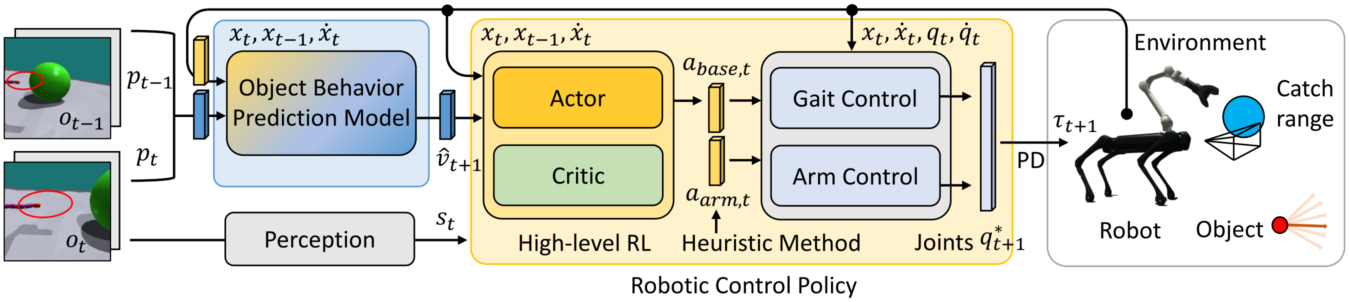


Fig. 3. **Overall framework of our method.** The left blue panel shows our object prediction model in the learning process, which involves both the current observations of the target and the robot states as inputs. The yellow panel shows our prediction-based RL policy, where the high-level control model takes both current observations and predicted states as input to generate base moving commands. The arm will move automatically when the object is close to the robot. These high-level actions are then fed into low-level control modules to generate joint control signals for the robot.

III. Learning Dynamic Object Behaviors

A. Problem Statement

Robotic catching policy learning can be formulated as a Markov decision process (MDP), which is represented as (S, A, R, T, γ), where S is the set of states, A is the set of actions, $R(s_t, a_t, s_{t+1})$ is the reward function, $T(s_{t+1}|s_t, a_t)$ is the transition function as a probability distribution, and γ is the discount factor for the future rewards. The agent policy $\pi(a|s)$ is the action selecting probability under a given state s. The goal of RL is to maximize the return under the policy $G_\pi = \mathbb{E}_\pi[\sum_t \gamma^t R(s_t, a_t, s_{t+1})]$. In robot learning tasks, we usually need to estimate the task-relevant states from observation O, regarded as $s = f(o)$. This setting is viewed as a partially observable Markov decision process (POMDP) where the policy is $\pi(a|f(o))$.

In this work, we study the task of reactive object catching with a mobile manipulator, where the state space $s = [s_{obj}, s_{rob}, s_{env}]$ and the action space $a = [a_{base}, a_{arm}]$. We represent the object state as $s_{obj} = [p_{obj}, \dot{p}_{obj}]$, where $p_{obj} \in \mathbb{R}^3$ is the position of the object. We hope to predict the object's reactive behaviors i.e., the object's desired velocity in the next step $\hat{v}_{obj} = \frac{\Delta p}{\Delta t} = \frac{\hat{p}_{t+1} - p_t}{\Delta t}$ from current observations. We will discuss the predictor learning for $\hat{v}_{obj}$ in the current section, leaving the discussion of RL policy in Section IV.

B. Learning Target Behavior Prediction Model

Dynamic objects in our task exhibit reactive movements in response to the robot's actions. The object's control policy is determined by both current and past states. The desired velocity of the object is given by:

$$v^*_{obj} = \pi_{obj}(s_{rob}, s_{obj}, s_{env}) \tag{1}$$

Here, π_{obj} can represent non-evasive policies like bouncing or evasive policies such as probabilistic evasions.

Given the consideration of visual latency, the robot captures the object's position at a reduced frequency (2Hz in this study). This leads to significant variations in observed object positions. Due to the inherent flexibility and stability constraints of legged robots with full dynamics, predicting an object's behavior becomes vital for making catching motion decisions. However, in our case, the real desired velocity of the object isn't directly observable; we can only measure resulting position changes. Consequently, we frame the state predictor problem as,

$$p_{obj,t+1} = h(v^*_{obj,t}, p_{obj,t}) \tag{2}$$

To anticipate such reactions, the robot needs to learn the object's behavior through a coupled model to predict $\hat{v}_{obj,t}$. The model can be learned from the observable positions $\{p_{obj,t}, t = 0, 1, 2, 3 \ldots\}$. Here, we represent the model as $\Phi(\cdot)$. As the object's reaction considers both the robot's actions and its states, this model's input state space includes:

$$s_{\phi,t} = \{x_{rob,t}, x_{rob,t-1}, \dot{x}_{rob,t}, p_{obj,t}, p_{obj,t-1}, s_{env,t}\} \tag{3}$$

where $x_{rob} \in SE(3)$ represents the robot's pose and s_{env} contains the wall positions in the room setting. The output of the prediction model represents the approximated object's desired velocity at the next time step $\hat{v}_{t+1} = \Delta p / \Delta t$. Note that time step means $\Delta t = 1/f_{vis}$, where f_{vis} is the visual observation frequency, not the simulation step $dt = 1/f_{physx}$. This is because the robot can only infer the object's behaviors from its own observations (subject to latency and field-of-view limitations) and cannot access real-time object behaviors.

This model is trained with samples collected through robot-object interactions. As the object's reactions cannot be obtained from stationary camera observations and its movements are coupled with the robot's actions, we randomly place the robot around the object for interactions and collect a sample buffer, represented as $B_{sample} = \{(s_{\phi,t}, v_{obj,t}) | t = 0, 1, 2, \ldots, n\}$. We then use L2 loss for the supervised learning of this model, represented as:

$$Loss = ||v_{obj,t} - \hat{v}_{obj,t}||_2, \ \ where\ \hat{v}_{obj,t} = \Phi(s_{\phi,t}) \tag{4}$$

We currently use samples where the object is observed in consecutive frames and do not consider the uncertainty caused when the object leaves the field of view. In the future, we hope to leverage the probabilistic estimation of object positions as a weak supervision to alleviate this limitation.

IV. Prediction-Based Reinforcement Learning for Legged Mobile Manipulator

In this section, we present our prediction-based RL approach for the legged mobile manipulator. The robotic control follows a hierarchical framework and the high-level policy is learned using our approach with the behavior predictor. The low-level joints are controlled by a pre-trained gait model and an inverse differential kinematic solver.

A. High-Level Catching Policy Learning

To effectively catch reactive objects, a mobile manipulator must operate in two phases: 1) pursuing and tracking the object, and 2) within a close range, moving the arm to reach the object rapidly. In this work, we focus on the first phase and treat the second arm-moving phase as an automatic process. While the object is close to the mobile robot as the distance $d < d_{arm,act}$, the arm will move directly to the object given $x_{goal,ee} = x_{obj}$, where $x_{goal,ee}$ is the goal pose of the gripper (as shown in Fig. 8). Based on this heuristic solution, the high-level RL policy generates base commands to control the robotic motion (as shown in Fig. 3).

The action space for the RL policy is then represented as $a = [v_x, v_y, \omega_{yaw}]$, where v_x, v_y are the desired base linear velocities, and ω is the yaw angular velocity. The state space is $s = [s_{rob}, s_{obj}, \hat{v}_{obj}, s_{env}]$, where $s_{rob} = [p_{rob}, \dot{p}_{rob}, R_{rob}, \dot{R}_{rob}]$ includes robot position and orientation, linear velocity and angular velocity, and $s_{obj} = [p_{obj}]$ is the position of the target. The predicted $\hat{v}_{obj}$ is the future velocity of the target. We assume the robot has an accurate segmentation mask to obtain object positions from depth cameras. Therefore, the goal is to optimize the policy π_{catch} where

$$a = \pi_{catch}(s_{rob}, s_{obj}, s_{env}, \Phi(s_\phi)) \tag{5}$$

The reward function in this work consists of: 1) a dense pursuit and tracking reward, 2) a visibility reward, and 3) a task success reward. The pursuit and tracking reward is,

$$R_{pursuit,t} = ||x_{obj,t-1} - x_{rob,t-1}|| - ||x_{obj,t} - x_{rob,t}|| \tag{6}$$

where x represents the world-frame coordinates. This reward in (6) encourages the robot to explore different paths to catch the object. The visibility and success rewards are,

$$R_{visibility} = \begin{cases} 0, \ \theta > \theta_{FoV} \\ 1, \ \theta \leq \theta_{FoV} \end{cases} \tag{7}$$

$$R_{success} = \begin{cases} 0, \ d_{obj,hand} > d_{suc} \\ 1, \ d_{obj,hand} \leq d_{suc} \end{cases} \tag{8}$$

where θ is the yaw-angle of the distance vector from the robot to the object. The final reward is weighted as,

$$R = w_1 R_{pursuit} + w_2 R_{visibility} + w_3 R_{success} \tag{9}$$

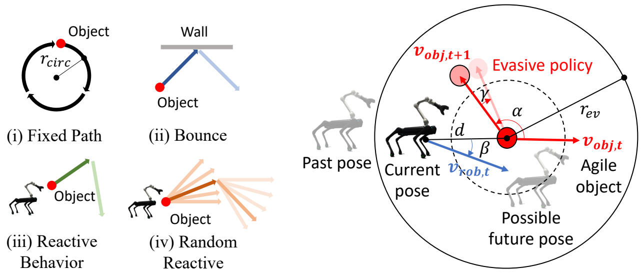
In this work, we set $w_1 = 1, w_2 = 0.2, w_3 = 5, \theta_{FoV} = 120°$.

We train the RL model using Proximal Policy Optimization (PPO) [39] algorithm, with the Gaussian head for the continuous action space. Both the actor and critic functions are approximated using a Multilayer Perceptron (MLP). The massively parallel RL paradigm is deployed for training.

The high-level actions are then generated by π_{catch} for the open plane setting during training and testing. To further evaluate our method in woods and rooms, we integrate a pre-trained visual navigation model [40] for collision avoidance (not used for RL training) into high-level control,

$$A_t = w_p \pi_{catch}(s_t) + w_c \pi_{collid}(s_t) \tag{10}$$

where $w_p, w_c = g(d_{rob,obst})$ are weights based on the distance of the robot and its closest obstacle in the environment.



(a) Trajectories and behaviors (b) Reactive policy for evasion

Fig. 4. **Illustration of reactive object behaviors.** (a) Four types of object movements representing commonly seen trajectories and reactive behaviors. (b) A demonstration of how a reactive target evades being caught by a robot through rapid turning. The policy is described in Sec. V-A.

B. Low-Level Joint Control

We use another RL model to train the low-level gait controller on rough terrains following the approach from [31]. The robotic arm is static during gait training but random forces are added to the mobile base of the quadruped robot for robust locomotion. During catching policy learning and evaluation, the arm is controlled by the joint-space inverse differential kinematics (IK) controller [22] with the damped least-square method, and the legged mobile base is controlled by the trained gait model. They take the robot's proprioceptive states as input to generate desired joint angles $q*$ and then use a PD controller for torques τ to the robot.

V. Experiments

In this section, we analyze the performance of the learned target predictor, evaluate the effectiveness of our predictor in RL training, and demonstrate the improvements of our prediction-based RL policy for this embodied dynamic catching task. We also present the adaptability of our approach in diverse environments and object trajectories.

A. Experimental Setups

We evaluate our approach in the Isaac Gym simulated environment [22] using the quadruped Aliengo robot [41] equipped with a Z1 robotic arm and two onboard depth cameras. The visual observation frequency is 2Hz, and the high-level catching policy is running at 50Hz to generate the base commands and arm commands. The low-level joint control is at 200Hz. The reactive objects are actuated by three virtual joints for the x, y, and yaw-rotation. The actuation joints are attached to the center of the first link. In Fig. 4, we present four object behaviors: fixed path, bounce, reactive behavior, and reactive behavior with a random heading angle change. The fixed path is a 5m radius circle. The reactive policy depicted in Fig. 4b aims to evade the robot's capture,

$$v^*_{obj} = \pi_{obj}(s) = ||v_{obj}|| \cdot u(\theta^*_{obj}) \tag{11}$$

where v^*_{obj} is the object's desired velocity, u is the unit vector of θ angle on xy-plane of the environment,

$$\theta^*_{obj} = \begin{cases} \arctan(d_y/d_x) + \gamma, \ d > r_{ev}, \\ \arctan(d_y/d_x) + \text{sign}(\beta)\alpha + \gamma, \ d \leq r_{ev} \end{cases} \tag{12}$$

where $d = x_{obj} - x_{rob}$ is the distance vector, and β is the yaw angle difference, $\gamma \sim U(-22.5°, 22.5°)$ is a random angle from the uniform distribution.

TABLE I

Task success rates and catching time on random reactive objects of different methods. (Success rate↑ / Time (s)↓)

Behavior	Method	Low Speed ($v_{obj} \geq 40\%\ v_{rob}$)		Medium Speed ($v_{obj} \geq 60\%\ v_{rob}$)		High Speed ($v_{obj} \geq 80\%\ v_{rob}$)		Average
		$v_{obj} = 0.8m/s$	$v_{obj} = 1.0m/s$	$v_{obj} = 1.2m/s$	$v_{obj} = 1.4m/s$	$v_{obj} = 1.6m/s$	$v_{obj} = 1.8m/s$	
Reactive	RL	0.952 / 3.24	0.716 / 8.46	0.522 / 11.50	0.236 / 16.30	0.026 / 19.56	0. / 20.00	0.409 / 13.18
	P+RL	0.871 / 4.62	0.803 / 5.88	0.493 / 11.40	0.200 / 16.70	0.005 / 19.10	0.009 / 20.00	0.397 / 12.95
	SP+FP	0.969 / 3.02	0.943 / 3.40	0.891 / 5.22	0.625 / 10.32	0.463 / 12.88	0.275 / 15.90	0.694 / 8.46
	SP+RL (Ours)	**0.982 / 2.48**	**0.978 / 2.62**	**0.978 / 2.82**	**0.987 / 2.94**	**0.973 / 3.70**	**0.908 / 5.76**	**0.968 / 3.39**
Random Reactive	RL	0.769 / 6.67	0.622 / 9.73	0.489 / 12.49	0.245 / 16.36	0.105 / 18.33	0.022 / 19.70	0.375 / 13.88
	P+RL	0.791 / 5.82	0.655 / 8.39	0.345 / 13.93	0.148 / 17.47	0.096 / 18.36	0.061 / 19.01	0.349 / 13.83
	SP+FP	**0.996 / 2.13**	**0.998 / 2.42**	**0.991 / 3.04**	0.925 / 4.84	0.725 / 8.97	0.502 / 13.44	0.856 / 5.81
	SP+RL (Ours)	0.987 / 2.38	0.978 / 2.86	0.978 / 3.10	**0.961 / 4.13**	**0.925 / 5.48**	**0.864 / 7.34**	**0.949 / 4.22**

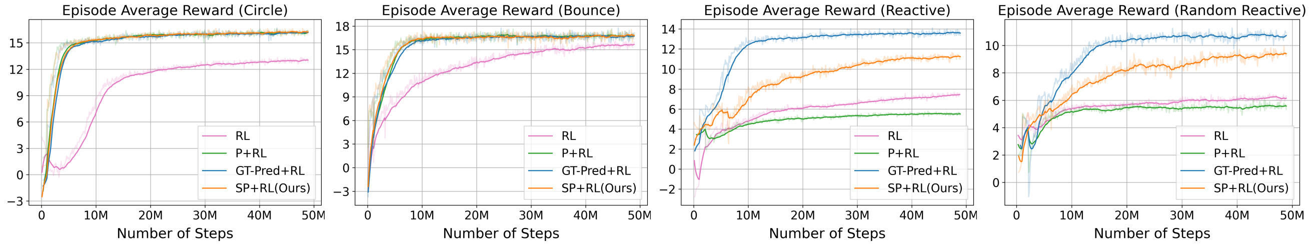


Fig. 5. **Learning curves of baseline methods for different object behaviors.**

TABLE II

Comparison of prediction error $||\hat{v} - v||_2$ for reactive targets

Behavior	Predictor	Number of Training Steps			
		0.2M	0.5M	1M	2M
Circle	Vanilla	0.001	0.	0.	0.
	Self-aware	0.002	0.001	0.001	0.
Bounce	Vanilla	0.048	0.022	0.024	0.022
	Self-aware	0.027	0.039	0.026	0.022
Reactive	Vanilla	1.251	1.201	1.261	1.197
	Self-aware	0.398	0.211	0.205	0.128

The evaluation metrics are: 1) Average success rate: the ratio of episodes where the object body link is inside a 5cm radius spherical space under 10cm of the robot's gripper. 2) Average catching time: the time taken for the catching task, where the maximum episode time is 20s for open planes and 40s for rooms and woods.

B. Analysis of The Learned Target Predictors

We first compare our self-aware predictor using additional robot state input, with the vanilla predictor that relies solely on target observations (Table II). During training and testing, the object's maximum velocity $||v_{obj}||$ is randomly set within $[-1.8m/s, 1.8m/s]$. While vanilla predictors converge faster on fixed trajectories, they falter on reactive targets. This is because the object considers the robot's state and reacts rapidly to avoid being caught. By leveraging this, our predictor can accurately anticipate $\hat{v}_{t+1}$ within only 2M robot-object interaction steps. Compared with the 20M to 50M steps required for high-level RL, predictor learning is an effective way to achieve better catching performance.

C. Comparison of Different Methods on Reactive Objects

We assess the benefits of prediction-based RL and the effectiveness of our predictor. The baseline methods are: 1) Monolithic RL (RL): The policy model will not explicitly predict target future movements. 2) Vanilla predictor w/o robot states + RL (P+RL). 3) Self-aware predictor with robot states + Frozen control policy (SP+FP): The RL control policy is trained with ground truth prediction (GT-Pred) of the targets which is then replaced by our learned predictor for testing. 4) Self-aware predictor with robot states + RL (SP+RL) that is our proposed method.

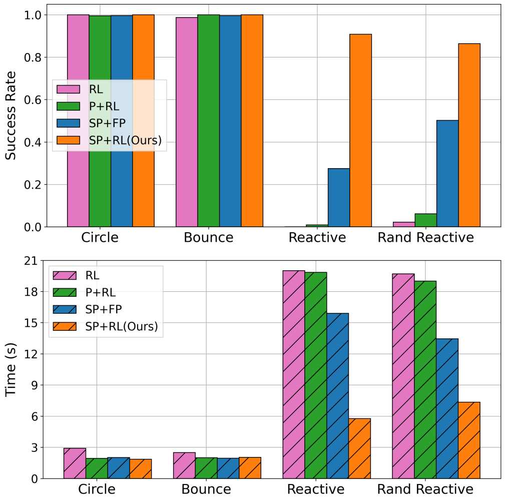


Fig. 6. **Average success rates and catching time of different methods.**

The learning curves of different methods on various object behavior settings are shown in Fig. 5. For objects with fixed trajectories, prediction-based RL converges faster than the monolithic method. For reactive targets, the RL agent with ground truth prediction obtains the highest final reward, followed by the agent using our learned predictor.

The evaluation results of reactive targets with varying speeds are shown in Table I. Notably, our method (SP+RL) consistently outperforms other methods in the high-speed scenarios. Even when the object's velocity approaches 80% of the robot's velocity, our approach remains resilient with the highest average success rate. At lower object speeds, our method maintains competitive success rates and catching times, often achieving the best results. The SP+FP method

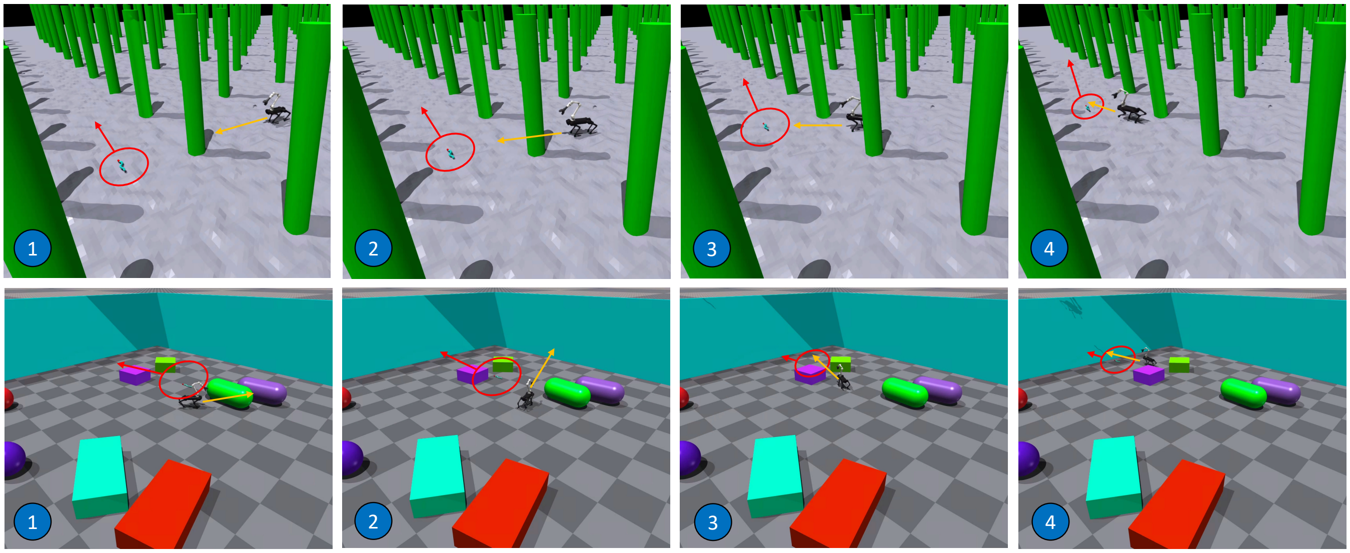


Fig. 7. **Visualization of typical scenarios: robot catching a snake indoors and in the woods.**

consistently outperforms other conventional methods, second only to our proposed approach. This highlights the potential of integrating self-awareness into prediction models.

D. Versatility Across Behaviors

This experiment compares the model performance on high-speed objects with different movements (As shown in Fig. 6). For all four types of object behaviors, our method (SP+RL) consistently showcases superior or comparable performance. Particularly in reactive targets, our method achieves the highest success rates with the minimum catching time. For fixed path trajectories of 'Circle Path' and 'Bounce', our method also costs less catching time than the monolithic RL. As shown in Table. II, our predictor also learns the fixed trajectories. These show the versatility of our method across behaviors not limited to evasive policies.

E. Adaptability in Diverse Environments

We evaluate the adaptability of our method in a variety of settings, including indoor rooms with obstacles, and outdoor forests with uneven terrains. Note that all models are trained exclusively on planar terrain, even when applied to environments with uneven terrain. Despite this, the proposed method exhibits a notable degree of adaptability when deployed in environments with uneven terrain, as shown in Fig. 9.

In indoor environments devoid of obstacles (Room-0), the Circle movement exhibits the highest success rate at 99.9%, followed by Bounce and Reactive movements at 86.5% and 73.8%, respectively. The average success rate for this environment is 86.8%. As the number of obstacles within the room increases (Room-4 and Room-8), there is a concomitant decrease in the success rate across all movement types. In outdoor forest environments with uneven terrain (Wood-R, and Wood-S), success rates are generally lower than those observed in plane environments, but our model still has decent performance. These results demonstrate the adaptability of the proposed method from planar to uneven terrain. Fig. 7 and Fig. 8 is the visualization of the robot catching evasive animals indoors and outdoors.

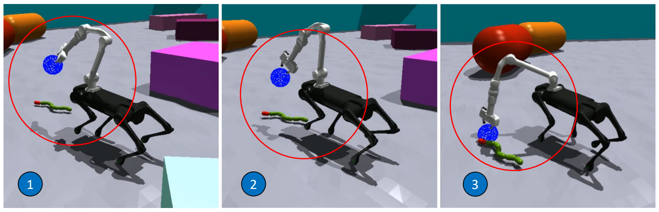


Fig. 8. **Visualization of the catching process of our mobile manipulator.**

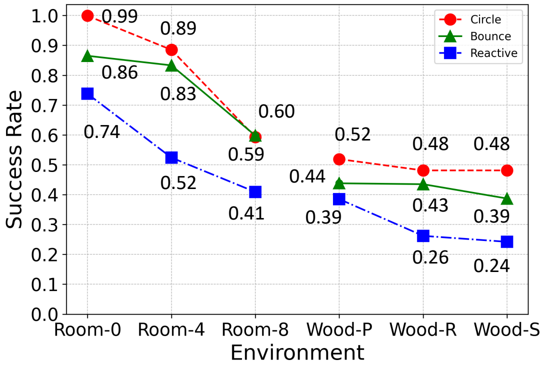


Fig. 9. **Generalization of our method to different scenes.** The notations of 0,4,8 for the rooms are the number of obstacles, and P,R,S for the woods are the terrain types of plane, rough, and stairs, respectively.

VI. CONCLUSION

In conclusion, we study the challenging task of robotic catching for reactive objects using a mobile manipulator. We proposed a tightly-coupled approach to learn object behaviors using both target positions and robot states. Building on this, we developed a prediction-based RL method for high-level catching policy learning. Our results demonstrate the superiority of our self-aware behavior predictor and prediction-based RL in enhancing robotic catching, especially in high-speed and reactive contexts. Additionally, our method adapts effectively in obstacle-rich and uneven terrains. In the future, we would like to extend the generalizability of our method to more challenging and interesting scenarios where multiple target objects move concurrently.

References

[1] Zhenjia Xu, Zhanpeng He, and Shuran Song. UMPNet: Universal Manipulation Policy Network for Articulated Objects. *IEEE Robotics and Automation Letters*, 7(2):2447–2454, 2022.

[2] Ben Eisner, Harry Zhang, and David Held. FlowBot3D: Learning 3D Articulation Flow to Manipulate Articulated Objects, 2022.

[3] Mayank Mittal, David Hoeller, Farbod Farshidian, Marco Hutter, and Animesh Garg. Articulated Object Interaction in Unknown Scenes with Whole-Body Mobile Manipulation, 2022.

[4] Zipeng Fu, Xuxin Cheng, and Deepak Pathak. Deep whole-body control: Learning a unified policy for manipulation and locomotion, 2022.

[5] Tongzhou Mu, Zhan Ling, Fanbo Xiang, Derek Yang, Xuanlin Li, Stone Tao, Zhiao Huang, Zhiwei Jia, and Hao Su. ManiSkill: Generalizable Manipulation Skill Benchmark with Large-Scale Demonstrations - Sep. *arXiv:2107.14483 [cs]*, 2021.

[6] Andrew Szot, Alex Clegg, Eric Undersander, Erik Wijmans, Yili Zhao, John Turner, Noah Maestre, Mustafa Mukadam, Devendra Chaplot, Oleksandr Maksymets, Aaron Gokaslan, Vladimir Vondrus, Sameer Dharur, Franziska Meier, Wojciech Galuba, Angel Chang, Zsolt Kira, Vladlen Koltun, Jitendra Malik, Manolis Savva, and Dhruv Batra. Habitat 2.0: Training Home Assistants to Rearrange their Habitat. *arXiv:2106.14405 [cs]*, 2021.

[7] Won Hong and Jean-Jacques E. Slotine. Experiments in hand-eye coordination using active vision. In Oussama Khatib and J. Kenneth Salisbury, editors, *Experimental Robotics IV*, volume 223, pages 130–139. Springer-Verlag, London, 1997.

[8] Ga-Ram Park, KangGeon Kim, ChangHwan Kim, Mun-Ho Jeong, Bum-Jae You, and Syungkwon Ra. Human-like catching motion of humanoid using Evolutionary Algorithm(EA)-based imitation learning. In *RO-MAN 2009 - The 18th IEEE International Symposium on Robot and Human Interactive Communication*, pages 809–815, 2009.

[9] Katharina Muelling, Jens Kober, and Jan Peters. Learning table tennis with a Mixture of Motor Primitives. In *2010 10th IEEE-RAS International Conference on Humanoid Robots*, pages 411–416, 2010.

[10] Jens Kober, Katharina Muelling, and Jan Peters. Learning throwing and catching skills. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5167–5168, 2012.

[11] Seungsu Kim, Ashwini Shukla, and Aude Billard. Catching Objects in Flight. *IEEE Transactions on Robotics*, 30(5):1049–1065, 2014.

[12] Seyed Sina Mirrazavi Salehian, Mahdi Khoramshahi, and Aude Billard. A Dynamical System Approach for Softly Catching a Flying Object: Theory and Experiment. *IEEE Transactions on Robotics*, 32(2):462–471, 2016.

[13] Kunyue Su and Shaojie Shen. Catching a Flying Ball with a Vision-Based Quadrotor. In Dana Kulić, Yoshihiko Nakamura, Oussama Khatib, and Gentiane Venture, editors, *2016 International Symposium on Experimental Robotics*, volume 1, pages 550–562. Springer International Publishing, Cham, 2017.

[14] Iretiayo Akinola, Jingxi Xu, Shuran Song, and Peter K. Allen. Dynamic grasping with reachability and motion awareness, 2021.

[15] Yinsen Jia, Jingxi Xu, Dinesh Jayaraman, and Shuran Song. Learning a meta-controller for dynamic grasping, 2023.

[16] Cristino de Souza, Rhys Newbury, Akansel Cosgun, Pedro Castillo, Boris Vidolov, and Dana Kulić. Decentralized Multi-Agent Pursuit Using Deep Reinforcement Learning. *IEEE Robotics and Automation Letters*, 6(3):4552–4559, 2021.

[17] Yuanda Wang, Lu Dong, and Changyin Sun. Cooperative control for multi-player pursuit-evasion games with reinforcement learning. *Neurocomputing*, 412:101–114, 2020.

[18] Zhe Hu, Yu Zheng, and Jia Pan. Grasping living objects with adversarial behaviors using inverse reinforcement learning. *IEEE Transactions on Robotics*, 39(2):1151–1163, 2023.

[19] Ruilong Zhang, Qun Zong, Xiuyun Zhang, Liqian Dou, and Bailing Tian. Game of Drones: Multi-UAV Pursuit-Evasion Game With Online Motion Planning by Deep Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–10, 2022.

[20] Zhe Hu, Yu Zheng, and Jia Pan. Living object grasping using two-stage graph reinforcement learning. *IEEE Robotics and Automation Letters*, 6(2):1950–1957, 2021.

[21] Selim Engin, Qingyuan Jiang, and Volkan Isler. Learning to Play Pursuit-Evasion with Visibility Constraints. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3858–3863, 2021.

[22] Viktor Makoviychuk, Lukasz Wawrzyniak, Yunrong Guo, Michelle Lu, Kier Storey, Miles Macklin, David Hoeller, Nikita Rudin, Arthur Allshire, Ankur Handa, and Gavriel State. Isaac Gym: High Performance GPU-Based Physics Simulation For Robot Learning. *arXiv:2108.10470 [cs]*, 2021.

[23] Benedek Forrai, Takahiro Miki, Daniel Gehrig, Marco Hutter, and Davide Scaramuzza. Event-based Agile Object Catching with a Quadrupedal Robot, 2023.

[24] Ziyun Wang, Fernando Cladera Ojeda, Anthony Bisulco, Daewon Lee, Camillo J. Taylor, Kostas Daniilidis, M. Ani Hsieh, Daniel D. Lee, and Volkan Isler. EV-Catcher: High-Speed Object Catching Using Low-Latency Event-Based Neural Networks. *IEEE Robotics and Automation Letters*, 7(4):8737–8744, 2022.

[25] Pierluigi Cigliano, Vincenzo Lippiello, Fabio Ruggiero, and Bruno Siciliano. Robotic ball catching with an eye-in-hand single-camera system. *IEEE Transactions on Control Systems Technology*, 23(5):1657–1671, 2015.

[26] Naresh Marturi, Marek Kopicki, Alireza Rastegarpanah, Vijaykumar Rajasekaran, Maxime Adjigble, Rustam Stolkin, Aleš Leonardis, and Yasemin Bekiroglu. Dynamic grasp and trajectory planning for moving objects. *Autonomous Robots*, 43(5):1241–1256, 2019.

[27] Tianhao Wu, Fangwei Zhong, Yiran Geng, Hongchen Wang, Yongjian Zhu, Yizhou Wang, and Hao Dong. Grasparl: Dynamic grasping via adversarial reinforcement learning, 2022.

[28] Jiafei Duan, Samson Yu, Hui Li Tan, Hongyuan Zhu, and Cheston Tan. A survey of embodied ai: From simulators to research tasks. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 6(2):230–244, 2022.

[29] Ziyuan Liu, Wei Liu, Yuzhe Qin, Fanbo Xiang, Minghao Gou, Songyan Xin, Maximo A. Roa, Berk Calli, Hao Su, Yu Sun, and Ping Tan. OCRTOC: A Cloud-Based Competition and Benchmark for Robotic Grasping and Manipulation. *arXiv:2104.11446 [cs]*, 2021.

[30] Yujin Tang, Jie Tan, and Tatsuya Harada. Learning agile locomotion via adversarial training, 2020.

[31] Nikita Rudin, David Hoeller, Philipp Reist, and Marco Hutter. Learning to Walk in Minutes Using Massively Parallel Deep Reinforcement Learning, 2021.

[32] Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Pooria Poorsarvi Tehrani, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments, 2023.

[33] Yuke Zhu, Josiah Wong, Ajay Mandlekar, Roberto Martn-Martn, Abhishek Joshi, Soroush Nasiriany, and Yifeng Zhu. robosuite: A modular simulation framework and benchmark for robot learning, 2022.

[34] Kiana Ehsani, Winson Han, Alvaro Herrasti, Eli VanderBilt, Luca Weihs, Eric Kolve, Aniruddha Kembhavi, and Roozbeh Mottaghi. Manipulathor: A framework for visual object manipulation, 2021.

[35] Timothy H. Chung, Geoffrey A. Hollinger, and Volkan Isler. Search and pursuit-evasion in mobile robotics: A survey. *Autonomous Robots*, 31(4):299–316, 2011.

[36] Isaac E. Weintraub, Meir Pachter, and Eloy Garcia. An Introduction to Pursuit-evasion Differential Games. In *2020 American Control Conference (ACC)*, pages 1049–1066, 2020.

[37] Milán Janosov, Csaba Virágh, Gábor Vásárhelyi, and Tamás Vicsek. Group chasing tactics: how to catch a faster prey. *New Journal of Physics*, 19(5):053003, 2017.

[38] Charles H. Wu, Donald A. Sofge, and Daniel M. Lofaro. Crafting a robotic swarm pursuit–evasion capture strategy using deep reinforcement learning. *Artificial Life and Robotics*, 27(2):355–364, 2022.

[39] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[40] Simar Kareer, Naoki Yokoyama, Dhruv Batra, Sehoon Ha, and Joanne Truong. Vinl: Visual navigation and locomotion over obstacles, 2023.

[41] Unitree. Unitree, 2023. Accessed: Date of Access.