Look Closer: Bridging Egocentric and Third-Person Views With Transformers for Robotic Manipulation

Rishabh Jangir, Nicklas Hansen, Sambaran Ghosal, Mohit Jain, and Xiaolong Wang

***Abstract*—Learning to solve precision-based manipulation tasks from visual feedback using Reinforcement Learning (RL) could drastically reduce the engineering efforts required by traditional robot systems. However, performing fine-grained motor control from visual inputs alone is challenging, especially with a static third-person camera as often used in previous work. We propose a setting for robotic manipulation in which the agent receives visual feedback from both a third-person camera and an egocentric camera mounted on the robot's wrist. While the third-person camera is static, the egocentric camera enables the robot to actively control its vision to aid in precise manipulation. To fuse visual information from both cameras effectively, we additionally propose to use Transformers with a *cross-view* attention mechanism that models spatial attention from one view to another (and vice-versa), and use the learned features as input to an RL policy. Our method improves learning over strong single-view and multi-view baselines, and successfully transfers to a set of challenging manipulation tasks on a real robot with uncalibrated cameras, no access to state information, and a high degree of task variability. In a hammer manipulation task, our method succeeds in 75% of trials versus 38% and 13% for multi-view and single-view baselines, respectively. Project website can be found https://jangirrishabh.github.io/lookcloser/.**

***Index Terms*—Reinforcement Learning, Visual Learning, Manipulation Planning.**

I. Introduction

MOST solutions for robotic manipulation today rely on highly structured setups that allow for full state information, fine-grained robot calibration, and predefined action sequences. This requires substantial engineering effort, and the resulting system is intolerant to changes in the environment. Visual feedback from a mounted camera has gained popularity as an inexpensive tool for relaxing the assumption of full state information [1]–[4], and Reinforcement Learning (RL) has emerged as a promising technique for learning flexible control policies without the need for detailed human engineering [5]–[8]. Together, vision-based RL could enable use of robots in unstructured environments through flexible policies operating directly from visual feedback, without assuming access to any state information [9]–[12].

However, solving complex precision-based manipulation tasks in an end-to-end fashion remains very challenging, and current methods often rely on important state information that is difficult to obtain in the real world [2], [13]–[16], and/or rely on known camera intrinsics and meticulous calibration [3], [17] in order to transfer from simulation to the real world. We argue that for a system to be truly robust, it should be able to operate from visual feedback, succeed without the need for camera calibration, and be flexible enough to tolerate task variations, much like humans. In particular, Land *et al.* [18] find that when humans perform a complex motor task, the oculo-motor system keeps the centre of gaze very close to the point at which information is extracted, also known as *visual fixation*. However, previous work in visual RL often limit themselves to a single, static third-person monocular camera, which makes extraction of local information from areas of interest challenging, and uncalibrated cameras only exacerbate the problem.

In this work, we propose a setting for vision-based robotic manipulation in which the agent receives visual feedback from two complementary views: (i) a third-person view (*global* information), and (ii) an egocentric view (*local* information), as shown in Fig. 1. While the third-person view is static, the egocentric camera moves with the robot gripper, providing the robot with *active* vision capabilities analogous to the oculo-motor system in humans. Because the agent has control over its egocentric vision, it can learn to actively position its camera such that it provides additional information at regions of interest, e.g. accurate localization of points of contact in fine-grained manipulation tasks.

To fuse visual feedback from the two views effectively, we propose to integrate an explicit modeling of visual fixation through a network architecture with soft attention mechanisms. Specifically, we propose to use Transformers [19] with a *cross-view* attention module that explicitly models spatial attention from one view to another. Each view is encoded using separate ConvNet encoders, and we fuse their corresponding feature maps by applying our cross-view attention module bidirectionally. In this way, we let every spatial region in the egocentric view *highlight* the corresponding regions of interest in the third-person view, and vice-versa. We use the learned features as input for an RL policy and optimize the network end-to-end using a reward signal. This encourages the agent to actively control the egocentric camera in a way that maximizes success.

To demonstrate the effectiveness of our method, we conduct extensive empirical evaluation and compare to a set of strong baselines on four precision-based manipulation tasks:

Manuscript received September 9, 2021; accepted December 30, 2021. Date of publication January 21, 2022; date of current version February 4, 2022. This work was supported in part by the Grants from NSF CCF-2112665 (TILOS), in part by the NSF 1730158 CI-New: Cognitive Hardware and Software Ecosystem Community Infrastructure (CHASE-CI), in part by the NSF ACI-1541349 CC-DNI Pacific Research Platform, and in part by the gifts from Qualcomm. This letter was recommended for publication by Associate Editor W. Pan and Editor J. Kober upon evaluation of the reviewers' comments. *(Corresponding authors: Rishabh Jangir; Nicklas Hansen.)*

The authors are with the University of California, San Diego 92037, CA USA (e-mail: jangirrishabh@gmail.com; hello@nicklashansen.com; sghosal@ucsd.edu; m4jain@ucsd.edu; xiw012@ucsd.edu).

Digital Object Identifier 10.1109/LRA.2022.3144512

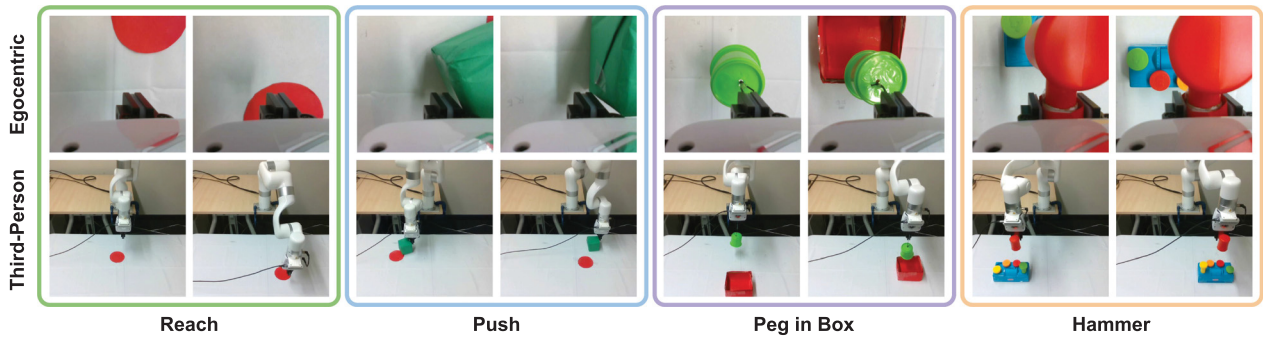

Fig. 1. Multi-view robotic manipulation. We propose a method for vision-based robotic manipulation that fuses egocentric and third-person views using a cross-view attention mechanism. Our method learns a policy using reinforcement learning that successfully transfers from simulation to a real robot, and solves precision-based manipulation tasks directly from uncalibrated cameras, without access to state information, and with a high degree of variability in task configurations.

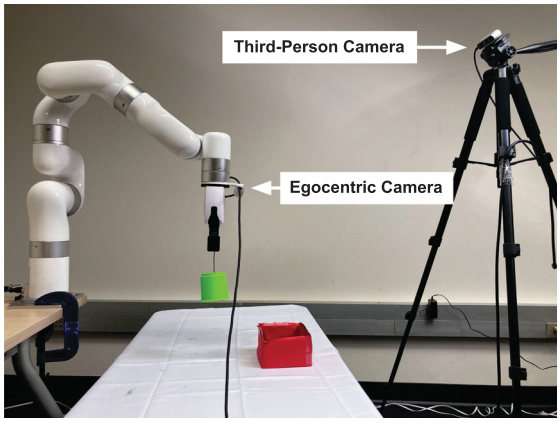

Fig. 2. Real robot setup. Image depicting our real-world environment for the *Peg in Box* task. The third-person camera is static, and the egocentric camera moves along with the robot arm. See Fig. 1 for camera view samples.

1) *Reach*, a task in which the robot reaches for a goal marked by a red disc placed on the table,
2) *Push*, a task in which the robot pushes a cube to a goal marked by a red disc, both placed on the table,
3) *Peg in Box*, a task in which the objective is to insert a peg tied to the robot's end-effector into a box placed on the table,
4) *Hammer*, a task in which the objective is to hammer in an out-of-position peg.

Each of the four tasks are shown in Fig. 1. We find that our multi-view setting and cross-view attention modules each improve learning over single-view baselines, and we further show that our method successfully transfers from simulation to a real robot setup (shown in Fig. 2) with uncalibrated cameras, no access to state information, and a high degree of task configuration variability. In the challenging *Hammer* task, our method is significantly more successful during transfer than any of our baselines, achieving a success rate of **75**% versus 38% for a multi-view baseline without cross-view attention, and only 13% and 0.0% for our single-view baselines. Finally, we qualitatively observe that (i) multi-view methods appear less prone to error due to lack of camera calibration, and (ii) our proposed cross-view attention mechanism improves precision in tasks that require fine-grained motor control.

II. Related Work

Vision-Based Robotic Manipulation: Learning RL policies for end-to-end vision-based robotic manipulation via task supervision has been widely explored [7]–[12], [20], [21]. For example, Lillicrap *et al.* [7] solve simulated continuous control tasks directly from images with no access to state information, and Zhan *et al.* [12] shows that vision-based RL can solve real-world manipulation tasks efficiently given a small number of expert demonstrations. Despite this progress it is still very challenging – especially for vision-based policies – to transfer learned skills to other environments, e.g. Sim2Real [22]–[24]. To facilitate transfer, related work also leverage important state information such as object pose, which is readily available in simulation but difficult to obtain in the real world [2], [13]–[17]. While this approach is valid, it is limited by the accuracy of pose estimation algorithms which typically require a well calibrated system and/or reference models [3], [17], [25], [26]. In comparison, our approach learns directly from raw images and transfers to a real robot with uncalibrated cameras.

Multi-View Robotic Manipulation: Multi-modal sensor fusion is a well-studied problem in robotics [9], [16], [27], [28]. For example, Lee *et al.* [27] propose a method for fusing vision and haptic feedback without explicit supervision. However, using multiple *views* of the *same* modality remains a relatively underexplored topic that has only recently gained interest [16], [29], [30]. Akinola *et al.* [29] show that third-person views from multiple, statically placed cameras improves policy learning in precision-based manipulation tasks over a single view. Specifically, their algorithm aggregates features encoded from multiple cameras by simple addition and performs RL on top of these features. In addition to camera inputs, they assume access to state information from the gripper. OpenAI *et al.* [16] explore a multi-view robotic manipulation setting similar to ours, but they require privileged state information that is not easily available in the real world, e.g. full object and goal states. Concurrent to our work, Chen *et al.* [30] propose a framework for learning 3D keypoints for control from multiple third-person views. While they similarly to Akinola *et al.* [29] demonstrate benefits from additional views, keypoint detection remains prone to error from uncalibrated cameras. All of the aforementioned works only consider manipulation tasks *in simulation*, and do *not* consider the problem setting of robotic manipulation from egocentric and third-person cameras without access to state information. In this

work, we develop a real robotic system that operates strictly from uncalibrated egocentric and third-person cameras.

Active Vision for Manipulation: Robots operating only from egocentric images suffer from incomplete state information (occlusion, small field-of-view), while when operating only from third-person vision fail to capture more accurate interactions. A natural solution to this problem is active vision. An active vision system is one that can manipulate the viewpoint of the camera(s) in order to investigate the environment and get better information from it [31], [32]. Traditional methods for active vision [33] either use hand-crafted utility functions [34] or are uncertainty or reconstruction based [35], whereas learning based [36] approaches explicitly learn these utility functions for selecting the next-best view using ground truth labels. In this work, we use the task-based reward to guide active vision. Cheng *et al.* [37] propose an active vision system that learns camera control policies directly from task-based rewards, in coordination to the manipulation policies. However, they use a separate camera to actively focus on objects of interest which requires an additional manipulator. In a similar fashion, Zaky *et al.* [38] use an additional manipulator to achieve an active perception system. Importantly, these approaches do not show real-world results. Our approach employs an egocentric camera tied to the wrist of the robot for active vision. While it is hard to both carry out manipulation and keep objects of interest in view in this perspective, our two complementary cameras enable us to achieve this through cross-view attention.

Attention Mechanisms in RL: Our multi-view fusion builds upon the soft attention mechanism which has been widely adopted in natural language processing [19], [39] and computer vision [40], [41], and has recently been popularized in the context of RL [28], [42]–[46]. For example, Jiang *et al.* [42] propose an attention-based communication model that scales to cooperative decision-making between a large amount of agents, and Yang *et al.* [28] use a cross-modal Transformer to fuse state information with a depth input in simulated locomotion tasks. James *et al.* [46] is a parallel work that is closest to our approach in their idea of using visual attention to attend to parts of an input RGB image in robotic manipulation tasks. However, like most other works on image-based RL, they restrict their study to a single, fixed third-person view. We propose a *cross-view* attention mechanism that fuses information from multiple camera views, and we are – to the best of our knowledge – the first to demonstrate the effectiveness of attention-based policies in transfer from simulation to a real robot setup.

III. Background

Vision-Based Reinforcement Learning. We formulate interaction between the agent's vision-based control policy and environment as an infinite-horizon Partially Observable Markov Decision Process (POMDP) [47] described by a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{P}, p_0, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ is a transition function that defines a conditional probability distribution $\mathcal{P}(\cdot|\mathbf{s}_t, \mathbf{a}_t)$ over possible next states given current state $\mathbf{s}_t \in \mathcal{S}$ and action $\mathbf{a}_t \in \mathcal{A}$ taken at time t, P_0 is a probability distribution over initial states, $r : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ is a scalar reward function, and $\gamma \in [0, 1)$ is a discount factor. In a POMDP, the underlying state $\mathbf{s}$ of the system is assumed unavailable, and the agent must therefore learn to implicitly model states from raw image observations. Our goal is to learn a policy $\pi : \mathcal{S} \mapsto \mathcal{A}$ that maximizes return $\mathbb{E}_{\Gamma\sim\pi}[\sum_{t=0}^{\infty} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)]$ along a trajectory $\Gamma = (\mathbf{s}_0, \mathbf{s}_1, \ldots, \mathbf{s}_\infty)$ where $\mathbf{s}_0 \sim P_0$, $\mathbf{a}_t \sim \pi(\cdot|\mathbf{s}_t)$, and $\mathbf{s}_{t+1} \sim \mathcal{P}(\cdot|\mathbf{s}_t, \mathbf{a}_t)$, and we use the Soft Actor-Critic [8] RL algorithm for policy search.

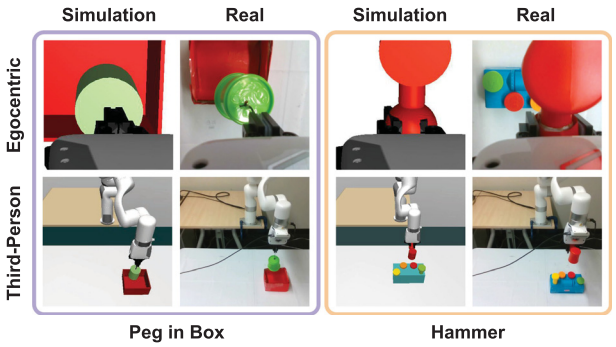

Fig. 3. Sim2Real. Samples from our simulation and real world environments for each view and two tasks, *Peg in Box* and *Hammer*. We emphasize that the real world differs in both visuals, dimensions, dynamics, and camera views, but roughly implement the same task as in simulation. Also note that there is only an approximate correspondence between the samples from simulation and the real world shown here.

Soft Actor-Critic: (SAC) [8] is an off-policy actor-critic algorithm that learns a (soft) state-action value function $Q_\theta : \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, a stochastic policy π_θ as previously defined, and optionally a temperature parameter τ_θ. Q_θ is optimized to minimize the (soft) Bellman residual [48], π_θ is optimized using a maximum entropy objective [49], [50], and τ_θ is optimized to maintain a desired expected entropy; see [8] for further details. For brevity, we generically refer to learnable parameterization by a subscript θ throughout this work.

Robot Setup: Our real-world setup is shown in Fig. 2. We use a Ufactory xArm 7 robot with an xArm Gripper as the robot platform in both our real-world and simulation experiments, although our method is agnostic to the specific robot hardware. The camera poses and the shape and size of objects in the simulation roughly mimic that of the real-world. We use Intel RealSense cameras for our experiments and apply the same pre-processing to both the simulated and real images. Image samples from the simulation and real-world robot setup can be seen in Fig. 3. Using the Mujoco [51] simulation engine, we create a simulation environment with the xArm robot and other objects as necessary in the robotic tasks. Two cameras are mounted to overlook the robot workspace where the task is being performed. With respect to the robot, one camera is directly placed in front of the whole setup (third-person view) with a large field of view covering the robot and the workspace, and the other camera (egocentric view) is attached to the robot at its wrist, i.e., right above where the gripper meets the robot. While the third-person camera is fixed, the egocentric camera moves along with the robot and provides a top-down view of the robot workspace. We emphasize that the cameras are *uncalibrated*, that the policy operates strictly from RGB images (84×84 pixels) captured by the two cameras, and that the policy has *no* access to state information.

IV. Method

Inspired by visual fixation in the human eye, we propose to fuse egocentric and third-person views in a multi-camera robotic

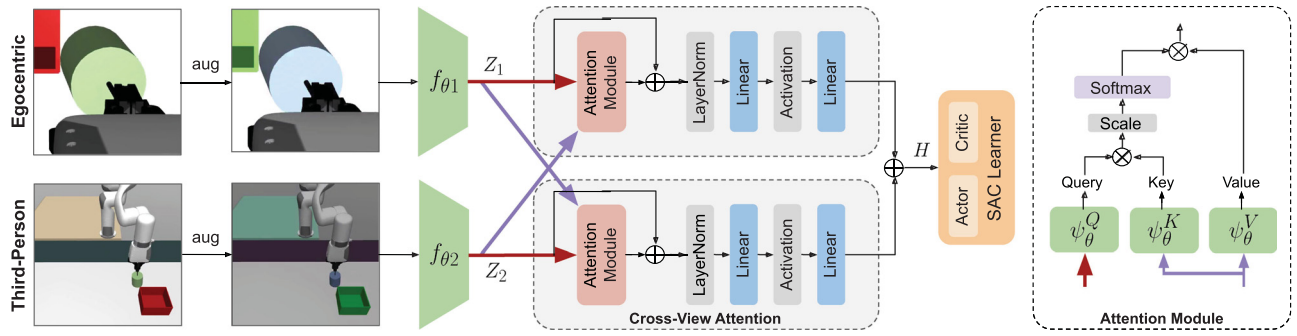

Fig. 4. Architectural overview. Egocentric and third-person views O_1, O_2 are augmented using stochastic data augmentation, and are encoded using separate ConvNet encoders to produce spatial feature maps Z_1, Z_2. We perform cross-view attention between views using a Transformer such that features in Z_1 are used as queries for spatial information in Z_2, and vice-versa. Features are then aggregated using simple addition ($\oplus$ in the figure), and used as input for a Soft Actor-Critic (SAC) policy.

manipulation setting, leveraging Transformers for explicit modeling of fixation through its attention mechanism. Our method is a general framework for multi-view robotic manipulation that is simple to both implement and deploy in a real-world setting. For simplicity, we describe our method using SAC [8] as backbone learning algorithm as used in our experiments, but we emphasize that our method is agnostic to the particular choice of learning algorithm. We introduce each component of our method in the following.

A. Multi-View Robotic Manipulation

We propose to learn vision-based RL policies for robotic manipulation tasks using raw, uncalibrated RGB inputs from two complementary camera views: (i) an egocentric view O_1 from a camera mounted on the wrist of the robot, and (ii) a fixed third-person view O_2. The third-person view provides global information about the scene and robot-object relative configurations, while the egocentric view serves to provide local information for fine-grained manipulation; see Fig. 3 for samples. Because the egocentric camera is attached to the robot, it provides the robot with *active* vision capabilities – control over one of the two views. Our method learns a joint latent representation from the two camera views, encouraging the model to relate global and local information, and actively position itself such that the egocentric view provides valuable local information for fine-grained manipulation.

B. Network Architecture

Fig. 4 provides an overview of our method and architecture. Our proposed architecture takes two raw RGB images $O_1, O_2 \in \mathbb{R}^{C\times H\times W}$ from the egocentric and third-person cameras, respectively (s.t. the system state $\mathbf{s}$ is approximated by $\{O_1, O_2\}$), and encodes them separately using ConvNet encoders $f_{\theta 1}$ and $f_{\theta 2}$ to produce latent feature maps $Z_1, Z_2 \in \mathbb{R}^{C'\times H'\times W'}$. We propose to fuse the two feature maps using a Transformer [19] with a cross-view attention module that takes Z_1, Z_2 as input and produces a single feature map $H = T_\theta(Z_1, Z_2)$. The aggregated features H are flattened and fed into the policy (actor) π_θ and state-action value function (critic) Q_θ s.t. $\mathbf{a} \sim \pi_\theta(\cdot|T_\theta(f_{\theta 1}(O_1), f_{\theta 2}(O_2)))$ and similarly for the critic. In practice, we optimize all components end-to-end using the actor and critic losses of SAC, but only back-propagate gradients from the critic to encoder components $T_\theta, f_{\theta 1}, f_{\theta 2}$, as is common practice in image-based RL [20], [52]–[54]. In the following section, we describe our cross-view attention mechanism in more detail.

C. Cross-View Attention

Our Transformer T_θ learns to perform cross-view attention by associating spatial information between intermediate feature maps Z_1, Z_2 from the egocentric view O_1 and third-person view O_2, respectively. T_θ takes as input Z_1, Z_2 and produces a joint embedding

$$H = T_\theta(Z_1, Z_2) \tag{1}$$

$$\triangleq g_{\theta 1}\left(\mathrm{LN}(Z_1 + V_2 A_{12})\right) + g_{\theta 2}\left(\mathrm{LN}(Z_2 + V_1 A_{21})\right) \tag{2}$$

where $g_{\theta 1}, g_{\theta 2}$ are Multi-Layer Perceptrons (MLP), LN is a LayerNorm [55] normalization, and A_{12}, A_{21} are normalized (soft) scaled dot-product attention [19] weights

$$A_{12} = \sigma\left(K_2^\top Q_1/\sqrt{C'}\right),\ A_{21} = \sigma\left(K_1^\top Q_2/\sqrt{C'}\right) \tag{3}$$

for $Q_i, K_j, V_j \in \mathbb{R}^{C'\times H'\times W'}$ denoted as the queries, keys, and values, respectively, for the cross-view attention between Z_i and Z_j, and σ is a Softmax normalization. Q_i, K_j, V_j are view-dependent embeddings

$$Q_i = \psi_{\theta i}^Q(\mathrm{LN}(Z_i)),\ K_j = \psi_{\theta j}^K(\mathrm{LN}(Z_j)) \tag{4}$$

$$V_j = \psi_{\theta j}^V(\mathrm{LN}(Z_j))\,, \tag{5}$$

where $\psi_{\theta i}^Q, \psi_{\theta j}^K, \psi_{\theta j}^V$ are 1×1 convolutional layers.

Intuitively, each of the two cross-view attention mechanisms between feature maps Z_i, Z_j can be interpreted as a differential (spatial) lookup operation, where Z_i is used as query to retrieve information in Z_j. By performing cross-view attention bidirectionally as shown in (2), T_θ enables flow of spatial information both from the egocentric view to the third-person view and vice-versa. With spatial attention, the policy can learn to associate concepts present in both views, e.g. objects or the gripper as shown in Fig. 5, and enhance both object-centric features as well as the overall 3D geometry of the scene. We note that, while we consider two views in this work, our method can trivially be extended to n views by letting T_θ compute cross-view attention bidirectionally between all pairs of views.

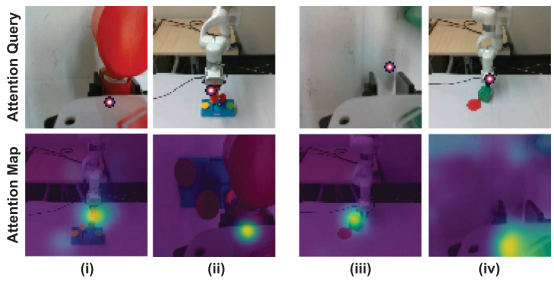

Fig. 5. Attention maps. Visualization of attention maps learned by our policy for **(i), (ii)** *Hammer* task, and **(iii), (iv)** *Push* task. For samples **(i), (iii)** a spatial location in egocentric view (red highlight; top) is used as query and its corresponding attention map for the third-person image (green highlight; bottom) is shown. For samples **(ii), (iv)** the third-person view is used as query and attention maps for the egocentric view are shown.

D. Data Augmentation

To aid both learning in simulation and transfer to the real world, we apply data augmentation to image observations, both in our method and across all baselines. During training, we sequentially apply stochastic image shifts of 0-4 pixels as in Kostrikov *et al.* [20], as well as the color jitter augmentation from Hansen *et al.* [45]. Augmentations are applied independently to each view, i.e., $O_1^{\text{aug}} = \text{aug}(O_1, \zeta_1)$, $O_2^{\text{aug}} = \text{aug}(O_2, \zeta_2)$, $\zeta_1, \zeta_2 \sim \Omega$ where ζ_1, ζ_2 parameterize the augmentations and Ω is a uniform distribution over the joint augmentation space.

V. Experiments

We investigate the effects of our proposed multi-view setting as well as our Transformer's cross-view attention mechanism on a set of precision-based robotic manipulation tasks from visual feedback. Both our method and baselines are trained entirely in simulation using dense rewards and randomized initial configurations of the robot, goal, and objects across the workspace. We evaluate methods both in the simulation used for training, and a real robot setup as described in Section III. For consistent results, we report success rate over a set of pre-defined goal and object locations both in simulation and the real world. Goal locations vary between tasks, and the robot is reset after each trial.

Tasks: Fig. 1 provides samples for each task and view considered. Using the simulation setup as described in Section III, we consider the following tasks: **(1) Reach**, a task in which the robot reaches for a goal marked by a red disc placed on the table. Success is considered when the robot gripper reaches within 5 cm of the goal. **(2) Push**, a task in which the robot pushes a cube to a goal marked by a red disc, both placed on the table. Success is considered when the cube reaches within 10 cm of the goal. **(3) Peg in Box**, a task in which the objective is to insert a peg tied to the robot's end-effector into a box placed on the table. Success is considered when the peg reaches within 5 cm of the goal and as a result gets inserted into the box. **(4) Hammer**, a task in which the objective is to hammer in an out-of-position peg. At each episode, one peg is randomly selected from 4 differently colored pegs. Success is considered when the out-of-position peg is hammered back within 1 cm of the box. Both the *Reach* and *Push* tasks use an XY action space, with movement along Z constrained. *Peg in Box* and *Hammer* use an XYZ action space. Episodes are terminated when the policy succeeds, collides with the table, or a maximum number of time steps is reached. An episode is counted as a success if the success criteria defined above is met for any time step in the episode.

Setup: We implement our method and baselines using Soft Actor-Critic (SAC) [8] as learning algorithm and, whenever applicable, we use the same network architecture and hyperparameters as in previous work on image-based RL [45], [54], i.e., f_θ consists of 11 convolutional layers. Observations are RGB images of size 84×84 from either one or two cameras depending on the method. Although related works commonly approximate the system state $\mathbf{s}$ using a stack of frames, we empirically find a single frame sufficient for both learning and transfer. All methods are trained for 500 k frames and evaluated for 30 trials across 5 object locations, both in simulation and on the real robot. However, for the *Hammer* task, we conduct 24 trials: 2 trials for each of the 4 pegs, repeated across 3 object locations, $2 \times 4 \times 3 = 24$. In simulation, all results are averaged over 3 model seeds; in real world experiments, we report transfer results for the best seed of each method due to real world constraints.

Baselines: We compare our method to a set of strong baselines, all using Soft Actor-Critic (SAC) as learning algorithm and the same choice of network architecture, hyperparameters, and data augmentations as our method. Specifically, we compare our method to the following baselines: **(1) Third-Person View** (*3 rd*) using visual feedback from a fixed third-person camera, **(2) Egocentric View** (*Ego*) using only the egocentric camera mounted on the robot's wrist, **(3) Multi-View** (*Multi*) that uses both views as input, encodes each view using separate encoders, and aggregates extracted features using addition. We emphasize that all baselines are implemented using the same image augmentations (image shift and color jitter) as our method, which makes them largely equivalent to DrQ [20], a state-of-the-art algorithm for image-based RL that builds on SAC. Lastly, we note that (3) implements our method without cross-view attention and is therefore analogous to the method proposed by Akinola *et al.* [29] but for a combination of third-person and egocentric views, without state information from the gripper, and including recent advances in image-based RL such as augmentations. In addition to these baselines, we also ablate our design choices.

A. Robotic Manipulation in Simulation

Before discussing our real robot experiments, we first consider methods in simulation. Training performances are shown in Fig. 6. Our method using both multi-view inputs and a Transformer performs on par or better than baselines on all tasks, and we particularly observe improvements in sample efficiency on the *Peg in Box* task that requires 3D geometric understanding. All methods except the egocentric-only baselines trivially solve the *Reach* task, but we include it to better ground our results. We conjecture that the egocentric baseline performs significantly worse than other methods due to its lack of global scene information.

The corresponding success rates for each task and method after training for 500 k frames are shown in Table I. Interestingly, while a third-person view is sufficient for learning to solve 3 out of 4 tasks, we observe substantial improvements in success rate in both our method ($+\mathbf{56}\%$) and multi-view ($+20\%$), which is not immediately obvious from the episode returns. Qualitatively,

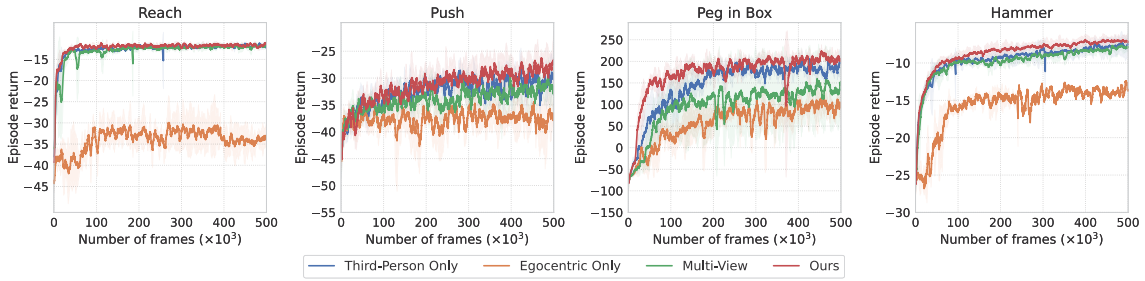

Fig. 6. Training performance. Episode return as a function of the number of frames during training, averaged over 3 seeds and shaded area is ±1 std. deviation. Our method consistently performs on par or better than all baselines considered.

TABLE I
SIMULATION EXPERIMENTS. SUCCESS RATE OF OUR METHOD AND BASELINES WHEN TRAINED AND EVALUATED IN SIMULATION. AVERAGED ACROSS 3 SEEDS AND 30 TRIALS (24 FOR *HAMMER*). OUR METHOD IS SIGNIFICANTLY MORE SUCCESS IN *HAMMER*

Simulation	3rd	Ego	Multi	Ours
Reach	**1.00**	0.15	**1.00**	**1.00**
Push	0.75	0.50	**0.80**	**0.80**
Peg in Box	**0.80**	0.20	**0.80**	**0.80**
Hammer	0.30	0.04	0.50	**0.86**

TABLE II
REAL ROBOT EXPERIMENTS. SUCCESS RATE OF OUR METHODS WHEN TRAINED IN SIMULATION AND TRANSFERRED TO A REAL ROBOT. AVERAGED ACROSS 30 TRIALS (24 FOR *HAMMER*)

Real robot	3rd	Ego	Multi	Ours
Reach	0.83	0.17	0.83	**1.00**
Push	0.10	0.17	0.23	**0.80**
Peg in Box	0.23	0.27	0.50	**0.80**
Hammer	0.13	0.00	0.38	**0.75**

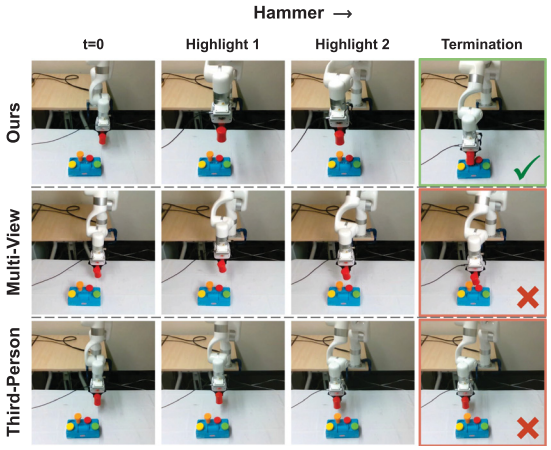

Fig. 7. Qualitative results. Sample trajectories from the *Hammer* task for our method, the multi-view baseline using both camera views, as well as the baseline using only a third-person view. Episodes are terminated when the policy succeeds, collides with the table, or a maximum number of time steps is reached. Our method succeeds in 75% of trials.

we find that the third-person baseline often approaches the peg but fails to hit it, whereas our method is comparably better at precise manipulation.

B. Sim2Real Transfer

We now consider deployment of the learned policies in a Sim2Real setting, i.e., the policies trained in simulation are transferred to a real robot setup as shown in Fig. 3. Success rates on the real robot are shown in Table II.

We find that success of the single-view baselines drops considerably when transferring to the real world. For example, third-person baseline (*3rd*) achieves success rates of 75% and 80% for the *Push* and *Peg in Box* tasks, respectively, in simulation, while merely 10% and 23% in the real world. We conjecture that this drop in performance is due to the reality gap – and the lack of camera calibration in particular. With the addition of an egocentric view, the multi-view baseline improves transfer to 23% and 50% on the two tasks. Finally, we observe no drop in success for our method on the same two tasks, achieving a success rate of 80% in both tasks, and only a small drop in success rate on the challenging *Hammer* task that requires 3D understanding and a high level of precision. Specifically, our method succeeds in 75% of trials in the *Hammer* task versus only 38% and 13% for the multi-view and single-view baselines, respectively.

We also study the *qualitative* behavior of policies when transferring to the real world. Fig. 7 shows sample trajectories for our method, the multi-view baseline, and the third-person baseline on the *Hammer* task. In the *Hammer* task, we observe that the multi-view baseline frequently misses its target by a small margin (presumably due to the reality gap), and the third-person baseline systematically fails to reach the peg, which we conjecture is due to error in 3D perception from the uncalibrated camera. Using our method, we observe that the robot frequently positions its gripper above the peg, such that it is visible from the egocentric view during the hammering motion. Finally, we also evaluate the qualitative behavior of the cross-view attention module. Attention maps for a set of spatial queries are shown in Fig. 5. We find that the agent often attends to regions of interest, such as objects or the gripper. Transformer-based policies could therefore also be a promising technique for explainability in RL.

C. Ablations

Attention mechanism: Our experiments discussed in Section V-A and V-B ablate the choice of camera views. Our

TABLE III
ABLATIONS. SUCCESS RATE OF OUR METHOD AND ABLATIONS WHEN TRAINED IN SIMULATION AND EVALUATED IN *(LEFT)* THE SIMULATED ENVIRONMENT, AND *(RIGHT)* OUR REAL ROBOT SETUP. A_{12} ABLATES OUR METHOD BY ONLY PERFORMING CROSS-ATTENTION FROM THE EGOCENTRIC VIEW TO THE THIRD-PERSON VIEW, AND A_{21} CORRESPONDS TO THE OPPOSITE DIRECTION. AVERAGED ACROSS 3 SEEDS AND 30 TRIALS (24 TRIALS FOR *HAMMER*)

	Simulation			Real robot		
	A_{12}	A_{21}	Ours	A_{12}	A_{21}	Ours
Reach	**1.00**	**1.00**	**1.00**	0.63	0.73	**1.00**
Push	0.63	0.65	**0.80**	0.26	0.37	**0.80**
Peg in Box	0.70	0.70	**0.80**	0.23	0.53	**0.80**
Hammer	0.60	0.50	**0.86**	0.42	0.50	**0.75**

TABLE IV
ROBOT STATE AS ADDITIONAL INPUT. SUCCESS RATE COMPARISON WHEN TRAINED IN SIMULATION AND EVALUATED ON THE REAL ROBOT, ALONG WITH THE %-CHANGE IN PERFORMANCE AS COMPARED TO IMAGE-ONLY OBSERVATIONS

Real robot	3rd	Multi	Ours
Peg in Box	0.30	0.77	**0.87**
Hammer	0.16	0.43	**0.76**

method learns cross-view attention between the egocentric view and the third-person view and models each of the directions individually, i.e., an attention map A_{12} is computed for egocentric $\rightarrow$ third-person direction, and A_{21} is likewise computed for the opposite direction. We ablate each of these two modules such that cross-view attention is only learned unidirectionally; results are shown in Table III. Our ablations indicate that unidirectional cross-view attention achieves similar success rates to that of the multi-view baselines in simulation, only improving marginally in the *Hammer* task. In our real robot experiments, we observe performance gains in both of our unidirectional ablations over the multi-view baseline, but not consistently. We observe the biggest improvements in *Push* and *Hammer*, where global-local coordination is especially important. However, we find that our proposed formulation succeeds more often than either ablation and produces more consistent gains across tasks. We conjecture that letting *both* views attend to each other improves flow of information, which in turn improves the expressiveness of the learned joint representation.

Robot state utilization: We study the effect of including robot state along with visual inputs in Table IV. Robot state includes end-effector position and gripper state, and is readily available both in simulation and the real robot. Following the architecture as described in [29] for fusing state and visual inputs, we train the top 3 performing methods in simulation and test on the real robot on our two most difficult tasks. Adding robot state information brings performance gains across all methods, while *3rd* and *Multi* benefit comparably more from the added states. However, our method still outperforms all in this setting. We hypothesize that the baselines have difficulty extracting state information from visual inputs, whereas our proposed fusion mechanism alleviates this issue and is therefore less dependent on the provided state information. The remaining gap in performances can be attributed to the ability of each method to extract object state information from visual inputs.

TABLE V
CAMERA RANDOMIZATION. SUCCESS RATES WHEN TRAINED IN SIMULATION AND EVALUATED IN A CAMERA-RANDOMIZED SIMULATION, ALONG WITH %-CHANGE IN PERFORMANCE

Simulation	Multi	Ours
Reach	0.85 (−15.0%)	**0.98** (−2.0%)
Push	0.46 (−42.5%)	**0.72** (−10.0%)
Peg in Box	0.24 (−70.0%)	**0.55** (−31.3%)
Hammer	0.18 (−64.0%)	**0.47** (−45.3%)

Robustness analysis: Camera calibration can be a time consuming process, and typically needs to be repeated every time the camera or environment changes. Therefore, it is desirable to develop methods that can operate directly from uncalibrated cameras, and we hypothesize that our proposed method increases robustness to such settings. To test this hypothesis, we further evaluate methods in simulation under random camera perturbations, simulating deployment with a variety of uncalibrated cameras. Perturbations include camera properties such as camera pose and FOV. Results are shown in Table V. We find that our proposed cross-view attention mechanism is more robust to camera perturbations than the multi-view baseline across all tasks, presumably due to better information fusing across views.

VI. CONCLUSION

Precise robotic manipulation from visual feedback is challenging. Through experiments in both simulation and the real world, we observe that both our multi-view setting as well as our proposed cross-view attention mechanism improves learning and in particular improves transfer to the real world, even in the challenging setting of Sim2Real with uncalibrated cameras, no state information, and a high degree of task variability. Therefore, we believe our proposed problem setting and method is a promising direction for future research in robotic manipulation from visual feedback.

REFERENCES

[1] I. Lenz, H. Lee, and A. Saxena, "Deep learning for detecting robotic grasps," *Int. J. Robot. Res.*, vol. 34, no. 4-5, pp. 705–724, 2015.
[2] S. Levine, P. Pastor, A. Krizhevsky, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, no. 4-5, pp. 421–436, 2018.
[3] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox, "Posecnn: A convolutional neural network for 6D object pose estimation in cluttered scenes," 2017, *arXiv:1711.00199*.
[4] A. Zeng, S. Song, J. Lee, A. Rodriguez, and T. Funkhouser, "Tossingbot: Learning to throw arbitrary objects with residual physics," *IEEE Trans. Robot.*, vol. 36, no. 4, pp. 1307–1319, Aug. 2020.
[5] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.
[6] R. Sutton, D. A. McAllester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. Neural Inf. Process. Syst.*, 1999, pp. 1057–1063.
[7] T. Lillicrap *et al.*, "Continuous control with deep reinforcement learning," *Comput. Res. Repository*, 2016, *arXiv:1509.02971*.
[8] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
[9] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, pp. 1334–1373, 2016.

[10] L. Pinto and A. Gupta, "Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2016, pp. 3406–3413.

[11] A. Nair, V. H. Pong, M. Dalal, S. Bahl, S. Lin, and S. Levine, "Visual reinforcement learning with imagined goals," in *Proc. Neural Inf. Process. Syst.*, 2018, vol. 31, pp. 9191–9200.

[12] A. Zhan, P. Zhao, L. Pinto, P. Abbeel, and M. Laskin, "A framework for efficient robotic manipulation," 2020, *arXiv:2012.07975*.

[13] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 3389–3396.

[14] T. Johannink *et al.*, "Residual reinforcement learning for robot control," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 6023–6029.

[15] O. M. Andrychowicz *et al.*, "Learning dexterous in-hand manipulation," *Int. J. Robot. Res.*, vol. 39, pp. 20–3, 2020.

[16] O. OpenAI *et al.*, "Asymmetric self-play for automatic goal discovery in robotic manipulation," 2021, *arXiv:2101.04882*.

[17] G. Shi *et al.*, "Fast uncertainty quantification for deep object pose estimation," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 5200–5207.

[18] M. Land, N. Mennie, and J. Rusted, "The roles of vision and eye movements in the control of activities of daily living," *Perception*, vol. 28, no. 11, pp. 1311–1328, 1999.

[19] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5998–6008.

[20] I. Kostrikov, D. Yarats, and R. Fergus, "Image augmentation is all you need: Regularizing deep reinforcement learning from pixels," 2021, *arXiv:2004.13649*.

[21] W. Yan, A. Vangipuram, P. Abbeel, and L. Pinto, "Learning predictive representations for deformable objects using contrastive estimation," 2020, *arXiv:2003.05436*.

[22] L. Pinto, M. Andrychowicz, P. Welinder, W. Zaremba, and P. Abbeel, "Asymmetric actor critic for image-based robot learning," 2017, *arXiv:1710.06542*.

[23] K. Cobbe, O. Klimov, C. Hesse, T. Kim, and J. Schulman, "Quantifying generalization in reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2019, pp. 1282–1289.

[24] N. Hansen, Y. Sun, P. Abbeel, A. A. Efros, L. Pinto, and X. Wang, "Self-supervised policy adaptation during deployment," 2021, *arXiv:2007.04309*.

[25] C. Li, J. Bai, and G. Hager, "A unified framework for multi-view multi-class object pose estimation," in *Proc. IEEE Eur. Conf. Comput. Vis.*, 2018, pp. 254–269.

[26] C. Wang *et al.*, "Densefusion: 6D object pose estimation by iterative dense fusion," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 3338–3347.

[27] M. A. Lee *et al.*, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 8943–8950.

[28] R. Yang, M. Zhang, N. Hansen, H. Xu, and X. Wang, "Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers," 2021, *arXiv:2107.03996*.

[29] I. Akinola, J. Varley, and D. Kalashnikov, "Learning precise 3D manipulation from multiple uncalibrated cameras," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2020, pp. 4616–4622.

[30] B. Chen, P. Abbeel, and D. Pathak, "Unsupervised learning of visual 3D keypoints for control," 2021, *arXiv:2106.07643*.

[31] R. Bajcsy, "Active perception," *Proc. IEEE*, vol. 76, no. 8, pp. 966–1005, 1988.

[32] D. Wilkes and J. K. Tsotsos, *Active Object Recognition*. Univ. of Toronto, 1994.

[33] S. Chen, Y. Li, and N. M. Kwok, "Active vision in robotic systems: A survey of recent developments," *Int. J. Robot. Res.*, vol. 30, no. 11, pp. 1343–1377, 2011.

[34] S. Kriegel, C. Rink, T. Bodenmüller, and M. Suppa, "Efficient next-best-scan planning for autonomous 3D surface reconstruction of unknown objects," *J. Real-Time Image Process.*, vol. 10, no. 4, pp. 611–631, 2015.

[35] F. Fraundorfer *et al.*, "Vision-based autonomous mapping and exploration using a quadrotor mav," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 4557–4564.

[36] B. Hepp, D. Dey, S. N. Sinha, A. Kapoor, N. Joshi, and O. Hilliges, "Learn-to-score: Efficient 3D scene exploration by predicting view utility," in *Proc. Eur. Conf. Comput. Vis.*, 2018, pp. 437–452.

[37] R. Cheng, A. Agarwal, and K. Fragkiadaki, "Reinforcement learning of active vision for manipulating objects under occlusions," in *Proc. Conf. Robot Learn.*, 2018, pp. 422–431.

[38] Y. Zaky, G. Paruthi, B. Tripp, and J. Bergstra, "Active perception and representation for robotic manipulation," 2020, *arXiv:2003.06734*.

[39] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proc. North Amer. Chapter Assoc. Comput. Linguistics, Hum. Lang. Technol.*, 2019, pp. 4171–4186.

[40] X. Wang, R. B. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 7794–7803.

[41] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," 2021, *arXiv:2010.11929*.

[42] J. Jiang and Z. Lu, "Learning attentional communication for multi-agent cooperation," in *Proc. Neural Inf. Process. Syst.*, 2020, vol. 31, pp. 7254–7264.

[43] L. Chen *et al.*, "Decision transformer: Reinforcement learning via sequence modeling," 2021, *arXiv:2106.01345*.

[44] M. Janner, Q. Li, and S. Levine, "Reinforcement learning as one big sequence modeling problem," 2021, *arXiv:2106.02039*.

[45] N. Hansen, H. Su, and X. Wang, "Stabilizing deep q-learning with convnets and vision transformers under data augmentation," *Adv. Neural Inf. Process. Syst.*, vol. 34, 2021.

[46] S. James, K. Wada, T. Laidlow, and A. Davison, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," 2021, *arXiv:2106.12534*.

[47] L. P. Kaelbling, M. L. Littman, and A. R. Cassandra, "Planning and acting in partially observable stochastic domains," *Artif. Intell.*, vol. 101, no. 1-2, pp. 99–134, 1998.

[48] R. Sutton, "Learning to predict by the methods of temporal differences," *Mach. Learn.*, vol. 3, no. 1, pp. 9–44, 2005.

[49] J. Bagnell and B. D. Ziebart, "Modeling purposeful adaptive behavior with the principle of maximum causal entropy," Pittsburgh, PA, USA: Carnegie Mellon University, 2010.

[50] B. D. Ziebart, A. L. Maas, J. Bagnell, and A. Dey, "Maximum entropy inverse reinforcement learning," in *Proc. AAAI Conf. Artif. Intell.*, 2008, pp. 1433–1438.

[51] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2012, pp. 5026–5033.

[52] D. Yarats, A. Zhang, I. Kostrikov, B. Amos, J. Pineau, and R. Fergus, "Improving sample efficiency in model-free reinforcement learning from images," 2019, *arXiv:1910.01741*.

[53] A. Srinivas, M. Laskin, and P. Abbeel, "Curl: Contrastive unsupervised representations for reinforcement learning," 2020, *arXiv:2004.04136*.

[54] N. Hansen and X. Wang, "Generalization in reinforcement learning by soft data augmentation," in *Proc. Int. Conf. Robot. Automat.*, 2021, pp. 13611–13617.

[55] J. Ba, J. Kiros, and G. E. Hinton, "Layer normalization," 2016, *arXiv:1607.06450*.