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

Rishabh Jangir[*1] **Nicklas Hansen**[*1] **Sambaran Ghosal**[1] **Mohit Jain**[1] **Xiaolong Wang**[1]

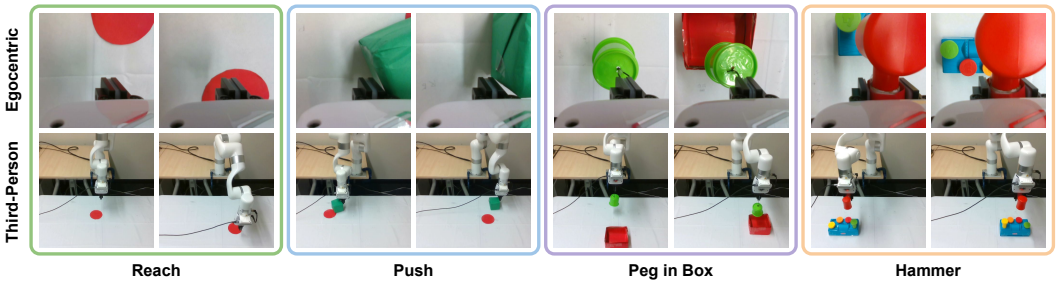

Figure 1. **Multi-view robotic manipulation.** We propose a method for vision-based robotic manipulation that fuses egocentric and third-person views using a cross-view attention mechanism. Our method learns a policy using reinforcement learning that successfully transfers from simulation to a real robot, 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.

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.

1 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 [28, 30, 44, 48], and Reinforcement Learning (RL) has emerged as a promising technique for learning flexible control policies without the need for detailed human

35th Conference on Neural Information Processing Systems (NeurIPS 2021).

engineering [43, 39, 31, 14]. 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 [29, 35, 32, 49].

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 [13, 30, 22, 2, 33], and/or rely on known camera intrinsics and meticulous calibration [44, 36] 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. [27] 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 Figure 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 [40] 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: (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 Figure 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 Figure 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 $\mathbf{75}\%$ 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.

2 Related Work

Vision-Based Robotic Manipulation. Learning RL policies for end-to-end vision-based robotic manipulation via task supervision has been widely explored [29, 31, 35, 32, 14, 25, 45, 49]. For example, Lillicrap et al. [31] solve simulated continuous control tasks directly from images with no access to state information, and Zhan et al. [49] shows that vision-based RL can solve real-world manipulation tasks efficiently. Despite this progress it is still very challenging – especially for vision-based policies – to transfer learned skills to other environments, e.g. Sim2Real [34, 9, 16]. 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 [13, 30, 22, 2, 36, 33]. In comparison, our approach learns directly from raw images and transfers to a real robot with uncalibrated cameras.

Multi-View Robotic Manipulation. Using multiple views in robotic manipulation remains a relatively unexplored topic that has only recently gained interest [1, 33, 6]. For example, Akinola et al. [1] show that third-person views from multiple cameras improves policy learning in precision-based manipulation tasks. In addition to camera inputs, they assume access to state information from the gripper. OpenAI et al. [33] 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. Concurrent to our work, Chen et al. [6] propose a framework for learning 3D keypoints for control from multiple third-person views. All of the aforementioned works only consider manipulation tasks *in simulation*. 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 [5, 42]. Traditional methods for active vision [8] either use hand-crafted utility functions [26] or are uncertainty or reconstruction based [12], whereas learning based [18, 23] 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. Our approach employs an egocentric camera tied to the wrist of the robot for active vision.

Attention Mechanisms in RL. Our multi-view fusion builds upon the soft attention mechanism which has been widely adopted in natural language processing [40, 10] and computer vision [41, 11], and has recently been popularized in the context of RL [21, 7, 20, 15, 46, 19]. For example, Jiang et al. [21] propose an attention-based communication model that scales to cooperative decision-making between a large amount of agents, and Yang et al. [46] use a cross-modal Transformer to fuse state information with a depth input in simulated locomotion tasks. James et al. [19] is a parallel work that is closest to our approach. 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.

3 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) [24] 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, $\mathcal{P}\colon \mathcal{S} \times \mathcal{A} \mapsto \mathcal{S}$ 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, $r\colon \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$ 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 $\pi\colon \mathcal{S} \mapsto \mathcal{A}$ 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 [14] RL algorithm for policy search.

Soft Actor-Critic (SAC) [14] is an off-policy actor-critic algorithm that learns a (soft) state-action value function $Q_\theta\colon \mathcal{S} \times \mathcal{A} \mapsto \mathbb{R}$, 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 [38], π_θ is optimized using a maximum entropy objective [4, 50], and τ_θ is optimized to maintain a desired expected entropy; see [14] 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 Figure 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

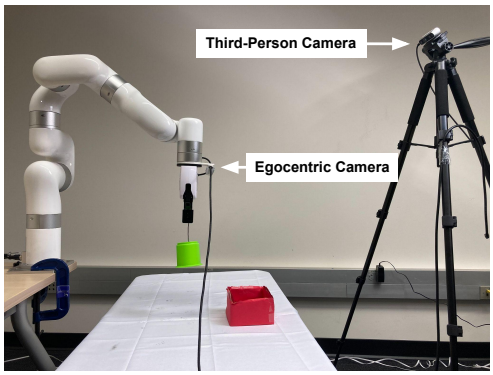

Figure 2. **Real robot setup.** Image depicting our real-world environment for the *Peg in Box* task. The third-person camera is static, and the egocentric camera moves along with the robot arm. See Figure 1 for camera view samples.

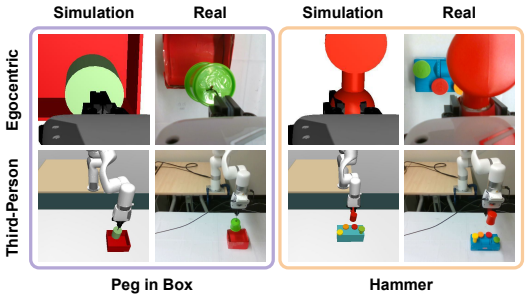

Figure 3. **Sim2Real.** Samples from our simulation and real world environments for each view and two tasks, *Peg in Box* and *Hammer*. We emphasize that the real world differs in both visuals, dimensions, dynamics, and camera views, but roughly implement the same task as in simulation. Also note that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

objects in the simulation roughly mimic that of the real-world. We 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 Figure 3. 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. 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.

4 Method

Inspired by visual fixation in the human eye, we propose to fuse egocentric and third-person views in a multi-camera robotic 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 [14] 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.

4.1 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

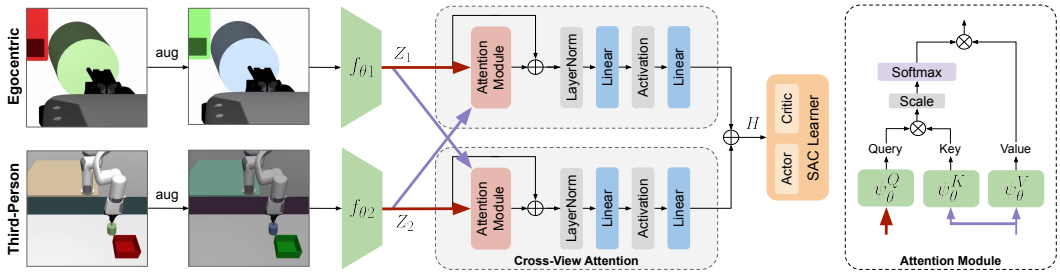

Figure 4. **Architectural overview.** Egocentric and third-person views O_1, O_2 are augmented using stochastic data augmentation, and are encoded using separate ConvNet encoders to produce spatial feature maps Z_1, Z_2. We perform cross-view attention between views using a Transformer such that features 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.

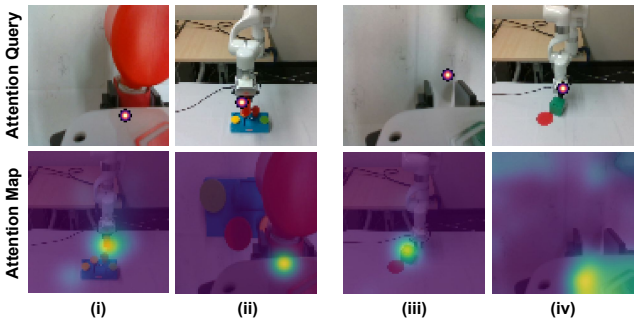

Figure 5. **Attention maps.** Visualization of attention maps learned by our policy for **(i), (ii)** *Hammer* task, and **(iii), (iv)** *Push* task. For samples **(i), (iii)** a spatial location in egocentric view (red highlight; top) is used as query and its corresponding attention map for the third-person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

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 Figure 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.

4.2 Network Architecture

Figure 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 [40] 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 [47, 37, 17, 25]. In the following section, we describe our cross-view attention mechanism in more detail.

4.3 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 $\mathrm{LayerNorm}$ [3] normalization, and A_{12}, A_{21} are normalized (soft) scaled dot-product attention [40] 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 $\mathrm{Softmax}$ normalization. Q_i, K_j, V_j are view-dependent embeddings

$$Q_i = \psi^Q_{\theta i}(\mathrm{LN}(Z_i)),\ K_j = \psi^K_{\theta j}(\mathrm{LN}(Z_j)) \tag{4}$$

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

where $\psi^Q_{\theta i}, \psi^K_{\theta j}, \psi^V_{\theta j}$ 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 Equation 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 Figure 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.

4.4 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 shift [25] and color jitter augmentations to each view independently after sampling from the replay buffer, i.e. $O_1^{\mathrm{aug}} = \mathrm{aug}(O_1, \zeta_1),\ O_2^{\mathrm{aug}} = \mathrm{aug}(O_2, \zeta_2),\ \zeta_1, \zeta_2 \sim \Omega$ where ζ_1, ζ_2 parameterize the augmentations and Ω is a uniform distribution over the joint augmentation space.

5 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 3. During evaluation, agents have no access to reward signals and are expected to generalize without any trials nor prior knowledge about the test conditions. To ensure 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. We first detail our experimental setup and then discuss our findings.

Tasks. Figure 1 provides samples for each task and view considered. Using the simulation setup as described in Section 3, 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

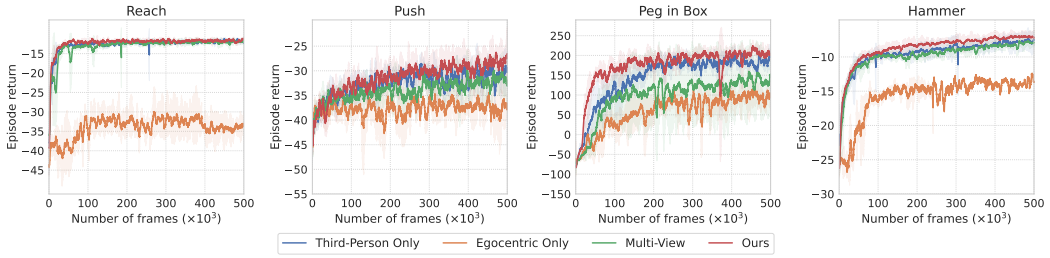

Figure 6. **Training performance.** Episode return as a function of the number of frames during training, averaged over 3 seeds and shaded area is ± 1 std. deviation. Our method consistently performs on par or better than all baselines considered.

Table 1. **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	0.00	0.50	**0.86**

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 all baselines using Soft Actor-Critic (SAC) [14] as learning algorithm and, whenever applicable, we similarly use the same network architecture and hyperparameters adopted from previous work on image-based RL [17, 15]. As in previous work, 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 500k frames and evaluated for 30 trials (24 in *Hammer*: 6 for each peg across 3 object locations), both in simulation and on the real robot. 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, whenever applicable. Specifically, we compare our method to the following baselines: **(1) Third-Person View** (*3rd*) 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 [25], 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. [1] 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 design choices in our cross-view attention mechanism.

Table 2. **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**

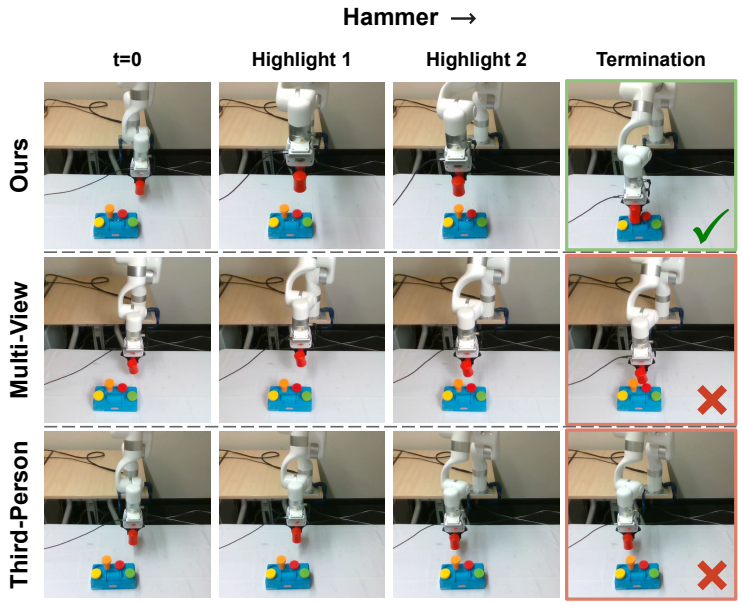

Figure 7. **Qualitative results.** Sample trajectories from the *Hammer* task for our method, the multi-view baseline using both camera views, as well as the baseline using only a third-person view. Episodes are terminated when the policy succeeds, collides with the table, or a maximum number of time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

5.1 Robotic Manipulation in Simulation

Before discussing our real robot experiments, we first consider methods in simulation. Training performance (in terms of episode return) of our method and baselines is shown in Figure 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 500k frames are shown in Table 1. 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 (+**56**%) and multi-view (+20%), which is not immediately obvious from the episode returns. Qualitatively, 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.

5.2 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 Figure 3. Success rates on the real robot are shown in Table 2.

Table 3. **Ablations.** Success rate of our method and ablations when trained and evaluated in simulation. 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 for *Hammer*).

Simulation	A_{12}	A_{21}	Ours
Reach	**1.00**	**1.00**	**1.00**
Push	0.63	0.65	**0.80**
Peg in Box	0.70	0.70	**0.80**
Hammer	0.60	0.50	**0.86**

We find that the 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 *Push* and *Peg in Box* tasks. Finally, we observe no drop in success for our method on the 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. As such, view aggregation using Transformers appear to be a promising research direction.

Similar to our simulation experiments, we also study the qualitative behavior of policies when transferring to the real world. Figure 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 Figure 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.

5.3 Ablations

Our experiments discussed in Section 5.1 and 5.2 ablate the choice of camera views. Our 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. Our ablations indicate that unidirectional cross-view attention achieves similar success rates to that of the multi-view baselines, only improving marginally in the *Hammer* task for the egocentric $\rightarrow$ third-person direction. In comparison, our proposed bidirectional formulation of cross-view attention succeeds more often than either ablation. 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.

6 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] Iretiayo Akinola, Jacob Varley, and Dmitry Kalashnikov. Learning precise 3d manipulation from multiple uncalibrated cameras. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4616–4622. IEEE, 2020. 3, 7

[2] OpenAI Marcin Andrychowicz, Bowen Baker, Maciek Chociej, R. Józefowicz, Bob McGrew, Jakub W. Pachocki, Arthur Petron, Matthias Plappert, Glenn Powell, Alex Ray, Jonas Schneider, S. Sidor, Joshua Tobin, P. Welinder, Lilian Weng, and Wojciech Zaremba. Learning dexterous in-hand manipulation. *The International Journal of Robotics Research*, 39:20 – 3, 2020. 2, 3

[3] Jimmy Ba, J. Kiros, and Geoffrey E. Hinton. Layer normalization. *ArXiv*, abs/1607.06450, 2016. 6

[4] J. Bagnell and Brian D. Ziebart. Modeling purposeful adaptive behavior with the principle of maximum causal entropy. 2010. 3

[5] Ruzena Bajcsy. Active perception. *Proceedings of the IEEE*, 76(8):966–1005, 1988. 3

[6] Boyuan Chen, P. Abbeel, and Deepak Pathak. Unsupervised learning of visual 3d keypoints for control. *ArXiv*, abs/2106.07643, 2021. 3

[7] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, M. Laskin, P. Abbeel, A. Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *ArXiv*, abs/2106.01345, 2021. 3

[8] Shengyong Chen, Youfu Li, and Ngai Ming Kwok. Active vision in robotic systems: A survey of recent developments. *The International Journal of Robotics Research*, 30(11):1343–1377, 2011. 3

[9] Karl Cobbe, Oleg Klimov, Christopher Hesse, Taehoon Kim, and J. Schulman. Quantifying generalization in reinforcement learning. In *ICML*, 2019. 2

[10] J. Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 3

[11] A. Dosovitskiy, L. Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, M. Dehghani, Matthias Minderer, G. Heigold, S. Gelly, Jakob Uszkoreit, and N. Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. *ArXiv*, abs/2010.11929, 2021. 3

[12] Friedrich Fraundorfer, Lionel Heng, Dominik Honegger, Gim Hee Lee, Lorenz Meier, Petri Tanskanen, and Marc Pollefeys. Vision-based autonomous mapping and exploration using a quadrotor mav. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4557–4564. IEEE, 2012. 3

[13] S. Gu, E. Holly, T. Lillicrap, and Sergey Levine. Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates. *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3389–3396, 2017. 2, 3

[14] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *arXiv preprint arXiv:1801.01290*, 2018. 2, 3, 4, 7

[15] Nicklas Hansen, H. Su, and Xiaolong Wang. Stabilizing deep q-learning with convnets and vision transformers under data augmentation. *ArXiv*, abs/2107.00644, 2021. 3, 7

[16] Nicklas Hansen, Yu Sun, P. Abbeel, Alexei A. Efros, Lerrel Pinto, and Xiaolong Wang. Self-supervised policy adaptation during deployment. *ArXiv*, abs/2007.04309, 2021. 2

[17] Nicklas Hansen and Xiaolong Wang. Generalization in reinforcement learning by soft data augmentation. In *International Conference on Robotics and Automation*, 2021. 5, 7

[18] Benjamin Hepp, Debadeepta Dey, Sudipta N Sinha, Ashish Kapoor, Neel Joshi, and Otmar Hilliges. Learn-to-score: Efficient 3d scene exploration by predicting view utility. In *Proceedings of the European conference on computer vision (ECCV)*, pages 437–452, 2018. 3

[19] Stephen James, Kentaro Wada, Tristan Laidlow, and A. Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. *ArXiv*, abs/2106.12534, 2021. 3

[20] Michael Janner, Qiyang Li, and Sergey Levine. Reinforcement learning as one big sequence modeling problem. *ArXiv*, abs/2106.02039, 2021. 3

[21] Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. In *NeurIPS*, 2018. 3

[22] T. Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, J. A. Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. *2019 International Conference on Robotics and Automation (ICRA)*, pages 6023–6029, 2019. 2, 3

[23] Mustafa Devrim Kaba, Mustafa Gokhan Uzunbas, and Ser Nam Lim. Supplementary material for paper entitled: A reinforcement learning approach to view planning problem. 3

[24] Leslie Pack Kaelbling, Michael L. Littman, and Anthony R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 1998. 3

[25] Ilya Kostrikov, Denis Yarats, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. *ArXiv*, abs/2004.13649, 2021. 2, 5, 6, 7

[26] Simon Kriegel, Christian Rink, Tim Bodenmüller, and Michael Suppa. Efficient next-best-scan planning for autonomous 3d surface reconstruction of unknown objects. *Journal of Real-Time Image Processing*, 10(4):611–631, 2015. 3

[27] M. Land, Neil Mennie, and Jennifer Rusted. The roles of vision and eye movements in the control of activities of daily living. *Perception*, 28:1311–28, 02 1999. 2

[28] I. Lenz, Honglak Lee, and Ashutosh Saxena. Deep learning for detecting robotic grasps. *The International Journal of Robotics Research*, 34:705 – 724, 2015. 1

[29] Sergey Levine, Chelsea Finn, Trevor Darrell, and Pieter Abbeel. End-to-end training of deep visuomotor policies. *The Journal of Machine Learning Research*, 17(1):1334–1373, 2016. 2

[30] Sergey Levine, P. Pastor, A. Krizhevsky, and Deirdre Quillen. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. *The International Journal of Robotics Research*, 37:421 – 436, 2018. 1, 2, 3

[31] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. *CoRR*, abs/1509.02971, 2016. 2

[32] Ashvin Nair, Vitchyr H. Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. In *NeurIPS*, 2018. 2

[33] OpenAI OpenAI, Matthias Plappert, Raul Sampedro, Tao Xu, I. Akkaya, V. Kosaraju, P. Welinder, Ruben D'Sa, Arthur Petron, Henrique Pondé de Oliveira Pinto, Alex Paino, Hyeonwoo Noh, Lilian Weng, Qiming Yuan, Casey Chu, and Wojciech Zaremba. Asymmetric self-play for automatic goal discovery in robotic manipulation. *ArXiv*, abs/2101.04882, 2021. 2, 3

[34] Lerrel Pinto, Marcin Andrychowicz, P. Welinder, Wojciech Zaremba, and P. Abbeel. Asymmetric actor critic for image-based robot learning. *ArXiv*, abs/1710.06542, 2018. 2

[35] Lerrel Pinto and Abhinav Gupta. Supersizing self-supervision: Learning to grasp from 50k tries and 700 robot hours. *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3406–3413, 2016. 2

[36] Guanya Shi, Yifeng Zhu, Jonathan Tremblay, Stan Birchfield, F. Ramos, Animashree Anandkumar, and Yuke Zhu. Fast uncertainty quantification for deep object pose estimation. *ArXiv*, abs/2011.07748, 2020. 2, 3

[37] Aravind Srinivas, Michael Laskin, and Pieter Abbeel. Curl: Contrastive unsupervised representations for reinforcement learning. *arXiv preprint arXiv:2004.04136*, 2020. 5

[38] R. Sutton. Learning to predict by the methods of temporal differences. *Machine Learning*, 3:9–44, 2005. 3

[39] R. Sutton, David A. McAllester, Satinder Singh, and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. In *NIPS*, 1999. 2

[40] Ashish Vaswani, Noam M. Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv*, abs/1706.03762, 2017. 2, 3, 5, 6

[41] X. Wang, Ross B. Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7794–7803, 2018. 3

[42] David Wilkes and John K Tsotsos. *Active object recognition*. University of Toronto, 1994. 3

[43] Ronald J. Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 2004. 2

[44] Yu Xiang, Tanner Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *ArXiv*, abs/1711.00199, 2018. 1, 2

[45] Wilson Yan, Ashwin Vangipuram, P. Abbeel, and Lerrel Pinto. Learning predictive representations for deformable objects using contrastive estimation. *ArXiv*, abs/2003.05436, 2020. 2

[46] Ruihan Yang, Minghao Zhang, Nicklas Hansen, Huazhe Xu, and Xiaolong Wang. Learning vision-guided quadrupedal locomotion end-to-end with cross-modal transformers. *ArXiv*, abs/2107.03996, 2021. 3

[47] Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images, 2019. 5

[48] Andy Zeng, Shuran Song, Johnny Lee, Alberto Rodriguez, and T. Funkhouser. Tossingbot: Learning to throw arbitrary objects with residual physics. *ArXiv*, abs/1903.11239, 2019. 1

[49] Albert Zhan, Philip Zhao, Lerrel Pinto, P. Abbeel, and M. Laskin. A framework for efficient robotic manipulation. *ArXiv*, abs/2012.07975, 2020. 2

[50] Brian D. Ziebart, Andrew L. Maas, J. Bagnell, and A. Dey. Maximum entropy inverse reinforcement learning. In *AAAI*, 2008. 3