ACTION MAP POLICY: LEARNING 3D CLOSED-LOOP MANIPULATION VIA PIXEL CLASSIFICATION

Anonymous authors
Paper under double-blind review

ABSTRACT

The action space poses a major challenge in robot learning, since it is often high-dimensional, can span long time horizons, and frequently admits multi-modal optimal solutions. A good choice of action representation and loss function can help to address these concerns, but there are often trade-offs. We propose Action Map Policy (AMP), which casts 3D closed-loop manipulation policy learning as a classification problem in image space. While classification has been an effective formulation in generative language models, applying it to robot action learning is difficult because naively discretizing high-dimensional continuous actions explodes the token vocabulary. Our key idea is to project 3D actions onto the camera image planes and treat each pixel location as a discrete class, thus controlling dimensionality while retaining multi-modality. This method supports millimeter-level precision for high-dimensional actions without requiring a prohibitively large vocabulary, while preserving fine-grained pixel-wise visual signals. Furthermore, it can predict the entire action chunk in a single forward pass, avoiding complex noise scheduling and iterative denoising while achieving substantially faster inference. Experiments on six simulated and five real-world manipulation tasks show that AMP outperforms strong baselines, improving the completion rate by 50%–70% on long-horizon real-world tasks, while running $6.8\times$ faster than Diffusion Policy and exhibiting stronger spatial reasoning.

1 INTRODUCTION

Modern large language models (LLMs) (OpenAI, 2023; Gemini Team et al., 2024; Touvron et al., 2023) have achieved remarkable success by casting sequence prediction as classification, delivering both strong predictive accuracy and powerful multi-modal distribution modeling. Extending this paradigm to robot action learning is appealing, since action distributions, like language, are inherently multi-modal and often admit several optimal solutions for a given state. A naive formulation, however, discretizes each translation and rotation component independently, producing a combinatorial explosion of tokens: with only 10 bins per dimension, a single 6-DoF action already requires 10^6 classes, and this cost compounds further over the action horizon. Alternatively, autoregressive tokenization predicts one dimension at a time to keep the vocabulary small, but it requires sequential decoding and breaks the geometric coupling between action dimensions. As a result, discrete tokenization struggles to faithfully represent high-dimensional continuous actions, failing to capture the geometric structure of large action spaces and degrading performance on fine-grained manipulation tasks (Goyal et al., 2025; Liu et al., 2026; Zitkovich et al., 2023; Kim et al., 2024).

To address these challenges, we propose Action Map Policy (AMP), a new formulation that casts 3D closed-loop manipulation policy learning as a classification problem in image space. Instead of regressing high-dimensional continuous actions (Zhao et al., 2023; Haldar et al., 2024; Mandlekar et al., 2021) or quantizing them into discrete action tokens (Goyal et al., 2025; Liu et al., 2026; Zitkovich et al., 2023; Kim et al., 2024), we represent 3D action learning as a pixel classification task in each 2D camera image plane. Specifically, we define the end-effector pose as a set of 3D keypoints and let the policy predict which pixel each keypoint projects to in every camera view. This turns policy learning into predicting a distribution over pixels for a set of coherent keypoints across future timesteps. This representation captures pixel-level precision while avoiding the combinatorial blow-up of per-dimension discretization. Our analysis shows that it supports the millimeter-level precision required for complex 3D tasks, including coffee preparation and toaster insertion. Additionally, it

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

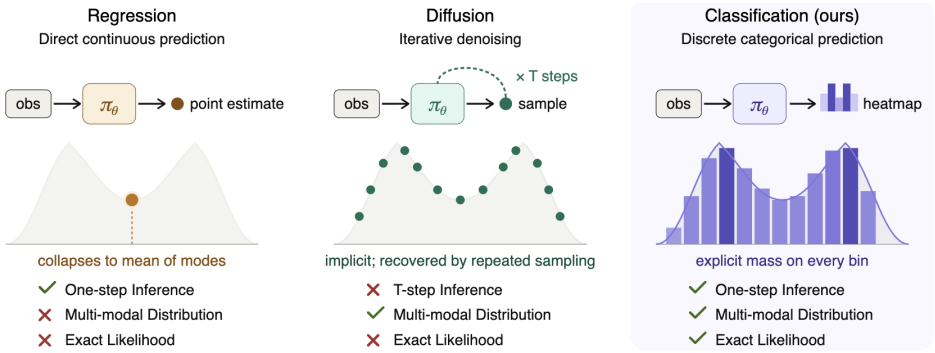


Figure 1: Comparison of different policy-learning frameworks. Regression methods can easily collapse to the mean of multiple modes. Diffusion methods can, in principle, capture multi-modal distributions, but require multiple denoising steps to generate a single sample and sampling-based procedures to recover the full distribution. In contrast, our method supports single-pass inference and captures multi-modal action distributions using a dense map that represents actions geometrically.

enables manipulation policies to be trained directly with a cross-entropy objective, which naturally models the multi-modal mass distribution over plausible trajectories. Several additional features further distinguish AMP from existing policy learning methods. As summarized in Figure 1, it generates an action chunk in a single forward pass, avoiding complex iterative sampling procedures and enabling much faster inference. Moreover, its output can be interpreted as an explicit action distribution, which may provide a natural interface for future refinement, such as reweighting actions or fine-tuning with reinforcement learning (Peng et al., 2019; Schulman et al., 2017).

To realize AMP with multi-camera geometry, we design a backbone architecture, *X-Net*, that fuses multi-view images and maps them directly to action feature maps, preserving dense spatial structure rather than collapsing the scene into a single global embedding. AMP aligns the action representation with the observation, allowing the model to reason more effectively in image space. Because each input pixel is associated with a corresponding action prediction, the policy remains grounded in the spatial structure of the scene and can exploit fine-grained, pixel-level cues. We demonstrate this feature in a designed test with several visually identical objects, where the target is specified only by a laser pointer. Our method reliably responds to this small visual cue, while Diffusion Policy (Chi et al., 2025) largely ignores it. This is important because it enables the policy to use small but task-critical visual cues, allowing the robot to act more precisely and responsively in real-world manipulation.

Our contributions are summarized as follows: (1) We propose a novel closed-loop 3D manipulation policy that represents actions as dense heatmaps over a temporal horizon, reformulating policy learning as a classification problem; (2) We design *X-Net*, which maps multi-view observations to dense action maps aligned with the input images, preserving fine-grained visual cues and supporting pixel-level precision; (3) Experiments demonstrate that our method outperforms strong baselines in both simulated and real-world settings, achieving higher success rates, faster inference, and stronger spatial reasoning ability.

2 Related Work

Manipulation Action Representation: Action representation largely determines how robot policies interact with and learn from the physical world. Existing manipulation policies mainly differ in whether actions are represented as compact low-dimensional commands or as dense spatial structures. **Low-dimensional action representations** are a common choice, where actions live in a compact continuous space such as end-effector poses or joint commands. Many policy learning methods directly regress or diffuse these low-dimensional actions (Wang et al., 2024; Mandlekar et al., 2021; Hu et al., 2025; Chi et al., 2025). Other works map actions into alternative latent or discrete spaces. For example, Zhao et al. (2023) trains a CVAE (Sohn et al., 2015) to encode actions into latent style vectors, while Liu et al. (2026); Zitkovich et al. (2023); Kim et al. (2024) tokenize continuous actions into discrete action tokens. **Dense action representation** has also been explored, especially in open-loop pick-and-place settings. For example, Zeng et al. (2021); Huang et al. (2022); Jia et al. (2025) represent pick-and-place actions as dense feature maps, where each pixel corresponds to a

spatial location and the channel dimension encodes rotation. This idea has been extended to 3D action spaces in Shridhar et al. (2023); Huang et al. (2024a); Goyal et al. (2023). Related methods also represent actions through dense goal configurations, such as target point clouds to specify desired manipulation outcomes (Pan et al., 2023; Eisner et al., 2024; Huang et al., 2024b; 2025). However, these works on dense action representation are limited to single-step prediction in open-loop settings; for example, RVT (Goyal et al., 2023) consumes point clouds, predicts a translation heatmap only for the next keyframe, and discretizes rotation separately. In contrast, AMP encodes the full pose as keypoint heatmaps over a temporal horizon, enabling closed-loop 3D policy learning through classification from image observations.

Manipulation Learning Framework: We categorize existing 3D closed-loop manipulation policy learning frameworks into three major classes. **Regression-based methods** typically predict continuous actions directly and train the policy with an MSE loss (Zhao et al., 2023; Haldar et al., 2024; Mandlekar et al., 2021). While simple and efficient, these methods can struggle to model multi-modal action distributions, since minimizing MSE may collapse multiple valid actions into an averaged prediction (Pearce et al., 2023; Pan et al., 2026). **Iterative generative policies** include diffusion- and flow-matching-based methods (Chi et al., 2025; Hu et al., 2024b; Lipman et al., 2023; Yan et al., 2025). These methods model action distributions through iterative sampling procedures, often relying on noise schedules and multiple refinement steps to generate actions. In principle, they can represent multi-modal action distributions by learning the underlying score or flow field. However, because the distribution is represented implicitly through iterative generation, it is difficult to directly inspect, constrain, or supervise the likelihood of specific action candidates. **Classification-based methods** often follow the LLM-style autoregressive paradigm (Goyal et al., 2025; Zitkovich et al., 2023; Kim et al., 2024): they discretize continuous actions into action tokens and train the policy to predict the next action token or token chunk. This formulation benefits from a stable cross-entropy objective, but it can be difficult to apply to fine-grained manipulation tasks: nearby continuous actions may be mapped to the same discrete token, while increasing the resolution can lead to a prohibitively large vocabulary. In contrast, our method addresses this limitation by representing actions as pixel-wise heatmaps, enabling fine-grained action prediction through classification without requiring an excessively large action vocabulary. We further discuss how AMP relates to vision-language-action models and world action models in Appendix A.10.

Policy Learning using Keypoints: Keypoint learning has been widely studied in computer vision for tasks such as human pose estimation, facial landmark detection, and point tracking. In robot manipulation, keypoints provide a compact and geometrically meaningful way to represent actions (Haldar et al., 2026). A set of 3D keypoints can jointly encode translation and rotation, since the end-effector pose can be recovered from their spatial configuration. Prior works have used 3D keypoints for pick-and-place manipulation (Huang et al., 2023; Hu et al., 2024a; Huang et al., 2024c; Manuelli et al., 2019) and leveraged the pointing ability of vision-language models with depth observations for manipulation control (Yuan et al., 2024; Cheng et al., 2025). Recent methods further use keypoints as intermediate action representations for policy learning. For example, Ren et al. (2025); Xu et al. (2024) generate 2D keypoint trajectories by diffusing their coordinates. Our method builds on the strength of keypoints for structured action prediction, but differs in a key way: rather than predicting low-dimensional keypoint coordinates, it predicts dense keypoint distributions over the image space across a temporal horizon, enabling closed-loop 3D manipulation policy learning through classification.

3 METHOD

Setup: We focus on 3D closed-loop manipulation policy learning from image inputs. Given the current observation $\mathcal{O}_t = (o_t^1, o_t^2, \cdots, o_t^n)$ captured from n cameras with known intrinsics and extrinsics, our objective is to learn a policy function $\pi\colon \mathcal{O}_t \mapsto \mathcal{A}_t$ from observations to an action chunk $\mathcal{A}_t = (a_t, \cdots, a_{t+l-1})$ which represents a sequence of l actions. Each action specifies an end-effector pose $a = (T, R, w)$, where $T \in \mathbb{R}^3$, $R \in \mathrm{SO}(3)$ and $w \in \mathbb{R}$ denote the translation, rotation and gripper aperture, respectively.

We represent action chunks as spatiotemporal keypoints and factorize the policy function π into two mappings: a heatmap predictor $\psi : \mathcal{O} \to \mathcal{H}$, which maps observations to keypoint distributions, and

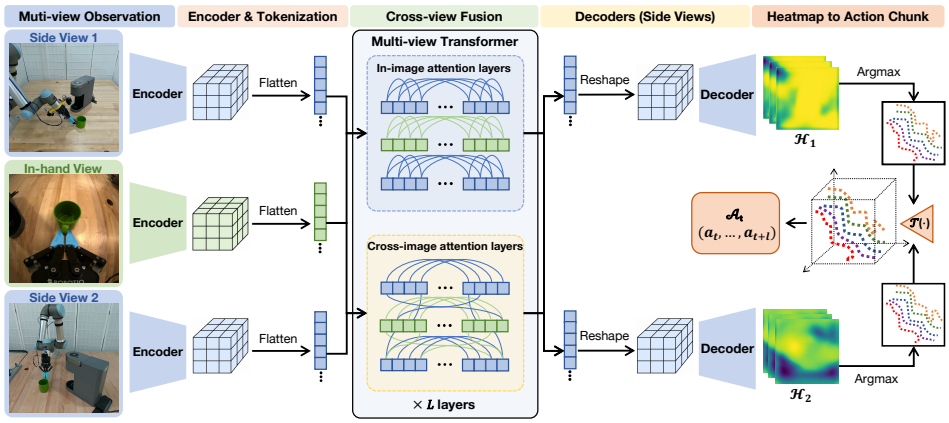


Figure 2: **Architecture of Action Map Policy.** The left branches take one in-hand image and two side-view images as input. The center features a multi-view transformer that enables communication between the in-hand features and the side-view context features. The right branch consists of two decoders that generate heatmaps of keypoints across the temporal horizon.

an action extractor $f : \mathcal{H} \to \mathcal{A}$, which converts heatmaps into executable actions. The entire policy can be expressed as the composition $\pi = f \circ \psi$.

Action Chunk Representation as 2D Heatmaps: Our model is trained entirely in 2D image space to predict actions as 2D pixel keypoints. We first convert 3D pose trajectories into 2D heatmaps to use as labels. Specifically, we map each pose $a = (T, R, w)$ to 3D keypoints $\tilde{a} = (p^1, \ldots, p^m)$ ($m = 5$ in our case) using the gripper geometry shown in Figure 3. Notably, this choice of keypoints is flexible and extends naturally to other end-effector designs. Next, we project the 3D keypoints to 2D coordinates in the camera image planes. Multi-camera projection $\mathcal{P}$ and triangulation $\mathcal{T}$ relate 3D points to 2D coordinates via the camera matrices. Under generic camera configurations, $(\mathcal{P}, \mathcal{T})$ establish a bijection between $\mathbb{R}^3$ and $\mathrm{Im}(\mathcal{P}) \subset (\mathbb{R}^2)^n$. We then *discretize* the projected coordinates into pixel locations and use them to create soft heatmap labels. For keypoint j at timestep i in camera view k, the label $\mathbf{h}_{ijk} \in \mathbb{R}^{H \times W}$ is a normalized distribution over the image grid, $\sum_{u,v} \mathbf{h}_{ijk}(u, v) = 1$.

Stacking a single keypoint's heatmaps across the l timesteps along the channel dimension yields its spatiotemporal volume $\mathbf{H}_{jk} \in \mathbb{R}^{l \times H \times W}$. Collecting these over all m keypoints and n cameras transforms the action chunk $\mathcal{A}$ into the set of heatmap volumes $\mathcal{H} = (\mathbf{H}_{jk})_{j=1,k=1}^{m,n}$. Under this representation, the policy function π is cast as a dense classification problem over the image plane. The network predicts, for each pixel, the likelihood of being traversed by each action keypoint at each timestep, thereby also unifying observation and action in a shared spatial representation.

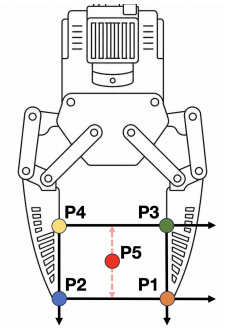


Figure 3: Keypoint-based Action Representation.

Predicting Actions as Distributions over Pixels: The keypoint-based heatmap formulates actions as a structured set of interaction primitives in image space. We introduce a heatmap predictor ψ to reason over these interactions. Specifically, we introduce what we call the *X-Net* architecture, which features a multi-view image input and multi-view heatmap output design. An example is shown in Figure 2, where the model takes two side-view images with one in-hand image for additional context information and generates distributions over pixels (normalized heatmaps) for the side views only. The left part of the X-Net uses a U-Net encoder (Ronneberger et al., 2015). The resulting dense feature maps are flattened into positionally-encoded tokens and fed into a Multi-View Transformer (MVT) (Goyal et al., 2023; Hamdi et al., 2021). The MVT consists of in-image attention layers and cross-image attention layers to enable communication across views. The output sequence of tokens is reshaped to recover the spatial dimensions and then fed into the U-Net decoder to generate the heatmap with the same resolution as the original image. While we illustrate a two-camera setup, this architecture is flexible and can be easily adapted to different numbers of cameras.

The model outputs a heatmap $\hat{\mathbf{h}}_{ijk}$ for each timestep i, keypoint j, and camera view k. Naturally, the loss is the cross-entropy loss between the predicted heatmap $\hat{\mathbf{h}}_{ijk}$ and the soft label heatmap $\mathbf{h}_{ijk}$, a Gaussian distribution over pixels centered at the projected 3D keypoint (u, v), as described above. The loss is summed over all timesteps, keypoints, and cameras: $\mathcal{L}_{\text{CE}} = -\sum_{i=1}^{l}\sum_{j=1}^{m}\sum_{k=1}^{n}\sum_{u,v}\mathbf{h}_{ijk}(u,v)\,\log\hat{\mathbf{h}}_{ijk}(u,v)$. Since the heatmaps across timesteps form a coherent trajectory pattern and are decoded jointly from shared multi-view features, their maxima remain spatiotemporally consistent (Appendix A.9).

Heatmap-to-Action Conversion: During training, the network is supervised entirely in heatmap space. During inference, we use an action extractor $f : \mathcal{H} \to \mathcal{A}$ to recover executable actions. As shown in Figure 2, we first identify the pixel locations with the highest probability via $\arg\max$, yielding m 2D keypoints across n cameras and l timesteps, denoted as $\{(u, v)_{ijk}\}$. We then reconstruct the 3D coordinates of each keypoint via triangulation using the camera matrices, i.e., $\mathcal{T}(\{(u, v)_{ijk}\}_{k=1}^{n}) \mapsto p^j \in \mathbb{R}^3$.

After triangulation, we obtain the 3D trajectories of the m keypoints ($m = 5$ in our case), from which we analytically recover the per-step pose (T, R, w). The translation T is the centroid of the four grasp points, $T = \frac{1}{4}(p^1 + p^2 + p^3 + p^4)$. The orientation R is built from two geometric axes, the antipodal axis $v_{\text{antipodal}} = \frac{1}{2}[(p^1 - p^2) + (p^3 - p^4)]$ and the approach axis $v_{\text{approach}} = \frac{1}{2}[(p^2 - p^4) + (p^1 - p^3)]$, which we orthonormalize via Gram–Schmidt and complete into a right-handed frame. The gripper width w is the normalized ratio of p^5's distances to the two fingertip midpoints, $w = d_+/(d_+ + d_-)$, where $d_+ = \|p^5 - \frac{1}{2}(p^2 + p^4)\|$ and $d_- = \|p^5 - \frac{1}{2}(p^1 + p^3)\|$.[1] By computing the pose at each step, we obtain a sequence of l actions that can directly command the robot. The model generates new actions from the latest observation at a fixed frequency, yielding a closed-loop visuomotor policy.

Implementation Details. We report key implementation choices here. Each keypoint is rendered as a Gaussian soft label with $\sigma = 2$ pixels. The setup uses two side-view cameras and one in-hand camera, all with 224×224 images, and heatmaps are predicted only for the side views. The encoder has four ResBlocks with $16\times$ downsampling, followed by a six-layer MVT module with two intra-image and four cross-image attention layers. The decoder has four upsampling ResBlocks and outputs 60 channels per side view, representing 5 keypoints over 12 timesteps; only the first 8 timesteps are executed. Extensive data augmentation is applied by transforming images and keypoint labels jointly. This is another advantage of our formulation: actions are spatially aligned with observations, enabling equivariant augmentation. Pseudocode is provided in Appendix A.3, and details on data preprocessing and augmentation in Appendix A.2.

4 Experiments

We first measure the theoretical precision that the heatmap action representation can achieve as a function of image resolution, and then evaluate its policy-learning performance in simulation, followed by ablation studies. We then design two types of real-world experiments to test its performance and spatial reasoning ability, and finally discuss its robustness, consistency, and scalability. Note that we mainly focus on policy performance *without* using a pretrained vision encoder for any method.

4.1 Precision and Robustness Analysis

To measure the discretization effects of keypoint heatmaps, we first implement projection and triangulation $(\mathcal{P}, \mathcal{T})$ in continuous space, which can be viewed as a forward transform $\mathcal{P}$ and its inverse $\mathcal{T} = \mathcal{P}^{-1}$. The composition $\mathcal{T}\cdot\mathcal{P}$ is the identity mapping even when the real-world calibration of off-the-shelf cameras is imperfect, since the same camera matrices are used for both projection and triangulation[2]. We then discretize the continuous image coordinates into pixels, yielding $\mathcal{T} \cdot \hat{\mathcal{P}}$, where $\hat{\mathcal{P}}$ denotes a discretized projection operator determined by the image resolution. Next, we map the ground-truth action $a_{\text{gt}} = (T, R, w)$ to a set of pixel coordinates, reconstruct the corresponding 3D keypoints, and compute the reconstructed action a_{rec}. We evaluate reconstruction precision on

[1] Points p^1–p^4 are arranged in a fixed relative configuration for robust axis estimation; consequently, their pairwise distances (e.g., $\|p^3 - p^1\|$) are constant and cannot encode the gripper width.

[2] Calibration errors thus only introduce small, consistent offsets in the projected labels, to which the policy naturally adapts during training.

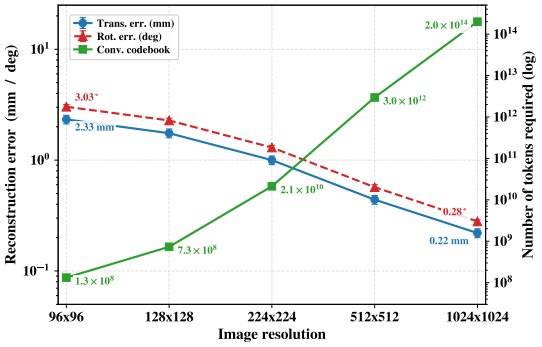


Figure 4: Reconstruction error (left) and conventional codebook size (right) vs. image resolution. The codebook grows by six orders of magnitude as resolution increases.

Table 1: Reconstruction accuracy at different image resolutions. The conventional discretization uses translation range $10\,\mathrm{mm}$ and rotation range $360°$.

Resolution	Trans. (mm)	Rot. (°)	# Tokens
96×96	2.33 ± 0.21	3.03 ± 0.14	$(\frac{10}{2.33})^3(\frac{360}{3.03})^3$
128×128	1.75 ± 0.16	2.28 ± 0.11	$(\frac{10}{1.75})^3(\frac{360}{2.28})^3$
224×224	1.00 ± 0.09	1.30 ± 0.06	$(\frac{10}{1.00})^3(\frac{360}{1.30})^3$
512×512	0.44 ± 0.04	0.57 ± 0.03	$(\frac{10}{0.44})^3(\frac{360}{0.57})^3$
1024×1024	0.22 ± 0.02	0.28 ± 0.01	$(\frac{10}{0.22})^3(\frac{360}{0.28})^3$

our real-world data in Table 1, measuring both translation and rotation errors. We compare it with conventional discretization methods that bin each action dimension and report the number of tokens required to achieve comparable accuracy.

As shown in Table 1, with a 224×224 image resolution, the heatmap representation achieves approximately $1\,\mathrm{mm}$ translation precision and $1.3°$ rotation precision. In contrast, conventional discretization would require around 10^{10} codebook tokens to achieve comparable precision, resulting in a prohibitively large number of classes. Moreover, the precision improves approximately linearly as the image resolution increases, achieving a translation error of $0.22\,\mathrm{mm}$ and a rotation error of $0.28°$ at a resolution of 1024×1024. In contrast, the number of required tokens for conventional discretization grows polynomially as n^6, leading to a rapidly increasing codebook size. This precision provides solid theoretical support for keypoint-heatmap-based policy learning.

4.2 Simulated Experiments

We evaluate Action Map Policy on six representative tasks from MimicGen (Mandlekar et al., 2023). These tasks cover both rigid and articulated objects, with medium to high precision requirements.

Task Description. As shown in Figure 5, stack-three-d1 requires the robot to stack three blocks; hammer-cleanup-d1 and mug-cleanup-d1 require the robot to open a drawer, put the target inside, and close the drawer; coffee-d2 is to place a coffee pod into the machine, while square-d2 and threading-d2 feature high-precision insertion tasks. The suffixes d0, d1, and d2 indicate the degree of spatial variation in the test scenarios, ranging from light to substantial variation. We use the largest initial-state distribution provided by MimicGen (Mandlekar et al., 2023), e.g., d2 for coffee and d1 for stack. We use two side views and one in-hand view for each task. The camera views are provided in Appendix A.1. We find that in standard MimicGen demonstrations, the gripper frequently falls outside the camera views, particularly in mug-cleanup-d1, coffee-d2, and threading-d2. To keep action keypoints within the image space during training, we truncate each action chunk at the first out-of-view keypoint and replace the rest with the last in-view configuration, while baselines are trained on the original actions. This discards part of our supervision on these three tasks, so the results of our method are likely conservative.

Baselines. We compare AMP against four strong baselines that cover different policy-learning frameworks: (1) Diffusion Policy (Chi et al., 2025), which uses an iterative denoising approach to predict actions conditioned on extracted observation embeddings; (2) ACT (Zhao et al., 2023), which directly maps observations to actions and is trained with an MSE loss, using a CVAE to model an action-style latent; (3) OAT (Liu et al., 2026), a token-based method that first learns a codebook with an autoencoder on demonstration actions and then predicts action tokens; and (4) Motion Tracks (Ren et al., 2025), a diffusion-based policy variant that takes a single view as input, denoises pixel keypoint coordinates independently, and triangulates the denoised coordinates from two views. Baselines are trained for 800 epochs on 100 demonstrations per task and evaluated every 40 epochs on 50 *unseen* test episodes. Since our method converges faster, it is trained for 200 epochs and evaluated the same

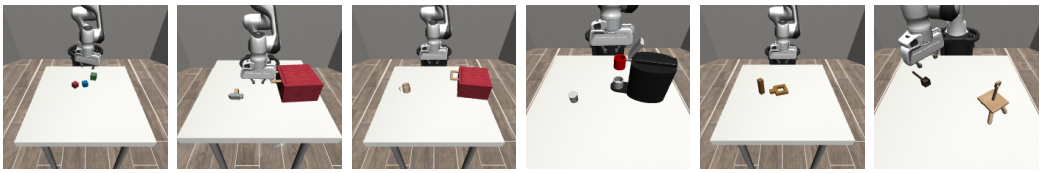


Figure 5: Representative tasks from MimicGen. From left to right: stack-three-d1, hammer-cleanup-d1, mug-cleanup-d1, coffee-d2, square-d2, and threading-d2.

Table 2: Performance Comparisons on MimicGen Benchmark. Success rate (mean%) over 50 unseen tests with 100 demonstration episodes used for training.

	stack-three-d1	hammer-cleanup-d1	mug-cleanup-d1	coffee-d2	square-d2	threading-d2
Method	100 Demo	100 Demo	100 Demo	100 Demo	100 Demo	100 Demo
Diffusion Policy (Chi et al., 2025)	38	58	**60**	62	20	26
ACT (Zhao et al., 2023)	14	60	44	42	6	20
OAT (Liu et al., 2026)	34	32	22	30	18	16
Motion Tracks (Ren et al., 2025)	8	40	24	36	14	12
Action Map Policy	**90**	**88**	52	**78**	**50**	**30**

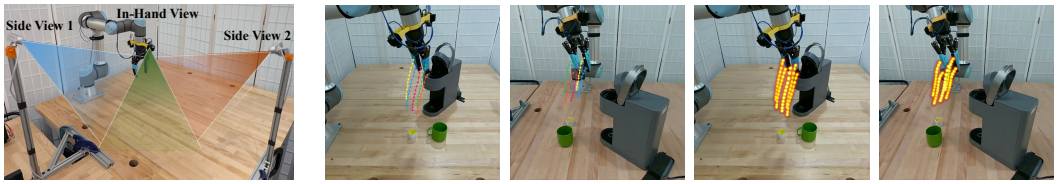
number of times (20, i.e., every 10 epochs). Following Chi et al. (2025), we report the best mean success rate across these checkpoints using two random seeds.

Results. Table 2 summarizes the results, from which several conclusions can be drawn. **(1)** AMP outperforms all baselines on five of the six tasks, with an average absolute gain of 20.7% over the second-best baseline, Diffusion Policy (44.0%). The improvement on individual tasks is up to 52% on stack-three-d1, demonstrating its advantage as a general policy-learning framework. **(2)** Compared with the regression-based method ACT and the token-based method OAT, AMP achieves better performance across all tasks without requiring any action pretraining. **(3)** AMP substantially outperforms the keypoint-based diffusion method Motion Tracks. We find that Motion Tracks can produce inconsistent trajectories across views due to its independent per-view inference, whereas AMP maintains consistent predictions through its cross-view design.

Table 3: Ablation study results.

Method	stack-three-d1	hammer-cleanup-d1	coffee-d2
Action Map Policy	**90**	**88**	**78**
w/o equiv. data aug.	58 (↓32)	62 (↓26)	66 (↓12)
w/o in-hand camera	88 (↓2)	74 (↓14)	64 (↓14)
one-hot label ($\sigma = 0$)	82 (↓8)	78 (↓10)	70 (↓8)
wide soft label ($\sigma = 4$)	86 (↓4)	78 (↓10)	72 (↓6)

Ablation Study. We ablate three key design choices and report the results in Table 3. **(1) Equivariant data augmentation.** Removing it causes a drop of 12–32%, highlighting the advantage of leveraging this augmentation. It is enabled by a distinctive property of AMP: actions are represented as heatmaps in the same image space as the observations, so each view and its heatmap labels can be rotated or translated jointly, which low-dimensional action vectors do not naturally support. This enlarges the support of the training data and encourages symmetry-consistent predictions (Appendix A.4). Note that all baselines follow the data augmentation settings of their official implementations, and even without equivariant augmentation, AMP still outperforms the strongest baseline on all three tasks (Table 2). **(2) In-hand camera.** Removing the in-hand view reduces success by 14% on hammer-cleanup-d1 and coffee-d2 but only 2% on stack-three-d1, suggesting that it provides complementary visual information for tasks with heavier occlusion. **(3) Soft-label width.** Both one-hot ($\sigma = 0$) and wider ($\sigma = 4$) labels



(a) workspace setup (b) side view 1 (c) side view 2 (d) keypoint prediction (e) keypoint prediction

Figure 6: Real-world system setup, ground-truth keypoint projection visualization, and action map prediction.

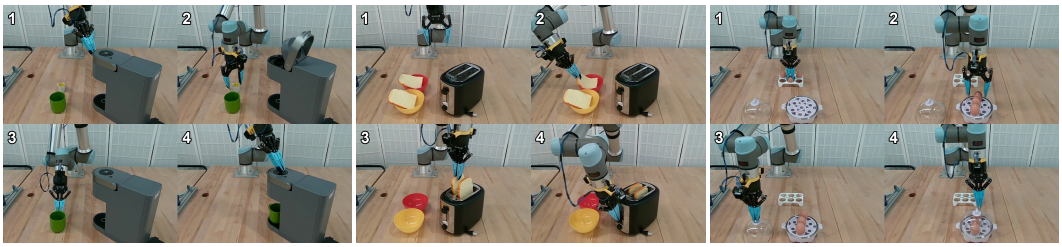


Figure 7: Real-world tasks: make-coffee, toast-bread, and steam-egg.

degrade performance, by up to 10%. σ controls how sharply the target probability mass concentrates around the projected keypoint: $\sigma = 0$ gives sparse supervision, whereas $\sigma = 4$ blurs the target; the default $\sigma = 2$ balances the two, retaining a well-localized peak with a smooth surrounding gradient (Appendix A.2). We further ablate the number of keypoints in Appendix A.5.

4.3 Real-World Experiments

We conduct five real-world experiments: three long-horizon tasks to evaluate our method as a closed-loop visuomotor policy, and two tasks to test its spatial reasoning ability under multi-modal action distributions. The system setup is shown in Figure 6a. We use a UR5 arm equipped with dual in-hand cameras, of which only one is used, and two RealSense D455 cameras as side views pointing toward the workspace. We calibrate the side-view cameras, collect demonstrations using Gello (Wu et al., 2024), and project the keypoint trajectories onto the images, as shown in Figures 6b and 6c.

Table 4: Real-world performance on long-horizon tasks, evaluated by completion rate and inference speed.

Method	Speed	Coffee	Toast	Egg
DiffPo (DDIM)	93.53 ms	25% (5/20)	40% (8/20)	25% (5/20)
ACT	7.16 ms	15% (3/20)	35% (7/20)	15% (3/20)
Action Map Policy	13.80 ms	**80%** (16/20)	**90%** (18/20)	**85%** (17/20)



Figure 8: Variation in initial object placements during testing

4.3.1 Real-World Evaluation of Visuomotor Policy Learning

We design three challenging long-horizon real-world tasks and compare our method with the two strongest baselines from our simulation experiments, Diffusion Policy and ACT. As shown in Figure 7, (1) make-coffee requires the robot to open the coffee machine lid, insert a coffee capsule, place the mug, and press the button. (2) toast-bread requires the robot to place each bread slice into the toaster slot and press the lever down. (3) steam-egg requires the robot to iteratively place three eggs into the slots and place the lid onto the egg steamer. For each task, we collect 70 demonstrations and evaluate each method over 20 runs with varied initial object placements. These tasks are long-horizon and require precise manipulation, and we report the task completion rates in Table 4.

We find that **(1)** AMP outperforms Diffusion Policy (Chi et al., 2025) and ACT (Zhao et al., 2023) by **50%–70%** on all three tasks. The baselines degrade significantly under spatial variations (Figure 8) and in precision-demanding steps, whereas AMP maintains strong spatial generalization and achieves high completion rates (80%–90%) on these long-horizon tasks, including high-precision steps such as inserting the coffee capsule and placing bread slices into the toaster slots; **(2)** AMP generates the entire action chunk in a single forward pass with an end-to-end latency of 13.80 ms, which is 6.8× **faster** than Diffusion Policy (93.53 ms, 16 DDIM steps) and close to the regression-based ACT (7.16 ms), while achieving substantially higher completion rates. Latency measurement details and heatmap visualizations are provided in Appendices A.7 and A.6.

4.3.2 Real-World Evaluation of Spatial Reasoning

We tailor two tasks, stack-target-cup (transparent) and stack-target-block, to evaluate the spatial reasoning ability of our method and baselines. Specifically, we present four identical objects, and a human uses a laser pointer to indicate the target object to manipulate. As shown in Figure 9, these

Table 5: Real-world spatial reasoning results.

Method	stack-target-cup	stack-target-block
DiffPo	25% (5/20)	15% (3/20)
ACT	75% (15/20)	65% (13/20)
Action Map Policy	**100%** (20/20)	**100%** (20/20)



Figure 9: Spatial reasoning tasks under multi-modal action distributions.

tasks feature small laser-point cues and multi-modal object distributions caused by identical objects. We collect 40 demonstrations for each task and evaluate each model over 20 test runs.

We report the results in Table 5. Our method achieves a success rate of 100%, while Diffusion Policy only achieves an average of 20%. We observe two main failure cases for the baselines: (1) the policy ignores the laser point and grasps a non-target object; and (2) the policy approaches the target object but fails to manipulate it due to incorrect action prediction or collapses toward the middle location between two identical objects. We hypothesize that the root cause is that Diffusion Policy denoises actions *conditioned* on observation features, which may make it difficult to directly respond to fine-grained geometric cues. ACT directly maps observations to actions and shows better spatial reasoning ability than Diffusion Policy. However, its regression-based formulation may collapse multi-modal behaviors, especially when identical objects are present. In contrast, our method captures fine-grained observation signals and responds correctly, while expressing complex action distributions through a classification objective.

4.4 Robustness, Consistency and Scalability

These practical aspects are orthogonal to our main contribution, a new policy-learning formulation; we discuss them here and provide details and results in Appendix A.9. **Robustness.** AMP performs well in the real world despite imperfect off-the-shelf camera calibration and extrinsic perturbations caused by table vibration, suggesting its practical robustness. Moreover, unlike conventional methods that implicitly absorb camera information into the network, AMP decouples the camera model from the network, which could facilitate transfer across camera setups, e.g., when trained with heterogeneous camera configurations. This decoupling also allows us to attribute errors to each source (Appendix A.9): the network's keypoint error on transformed observations is below one pixel, and triangulation stays at the millimeter level under small calibration noise. **Consistency.** Our multi-view design predicts all views jointly through cross-view attention, generating multi-modal distributions while keeping the per-view heatmap maxima geometrically consistent. This yields smooth rollouts in our supplementary videos and a low triangulated-action MSE ($1\text{–}2 \times 10^{-4}$) on the validation sets of all five real-world tasks. See Appendix A.9 for details. **Scalability.** AMP is not tied to a specific robot or camera placement, as reflected by the different setups across MimicGen tasks and our real-world system. While scalability is not the focus of this work, we verify within our current resources that AMP benefits from more data (200 demonstrations).

5 Conclusion and Limitations

In this work, we propose a novel policy-learning framework that casts closed-loop action learning as classification using action keypoint heatmaps. Experiments show that AMP substantially outperforms several baselines, while enabling fast single-pass inference and strong responsiveness to fine-grained geometric cues. Our precision analysis and real-world tests show that AMP supports high-precision tasks and generalizes well under spatial variations. There are several future directions we plan to explore. AMP requires at least two calibrated cameras for triangulation; although calibration is a one-time procedure that is readily obtained with standard tools, the workspace is limited by the calibrated camera coverage, which prevents actions outside the observed scope. This could be alleviated with larger-FOV cameras or dual in-hand cameras for dense feature prediction. While we focus on a parallel-jaw gripper, the method extends naturally to dexterous or bimanual setups by adapting the keypoint design. In addition, since AMP generates an explicit action distribution, it provides a promising opportunity to learn Q-functions for reinforcement learning. Finally, while our experiments train only from scratch, exploring how pretrained models could further strengthen the framework is an interesting future direction.

AI USE STATEMENT

We did not use generative AI tools for tasks requiring disclosure under the ICLR 2027 AI Policy for Authors; translation was not applicable. We used these tools to improve readability, assist with LaTeX formatting, and format and check references. The authors reviewed all AI-assisted work and take responsibility for the final content, including AI-assisted text, claims, and artifacts.

ETHICS STATEMENT

All real-world demonstrations were collected by the authors through robot teleoperation in a laboratory, and an author operated the laser pointer in the spatial-reasoning tasks. No human-subject study was conducted, and no personal data were collected. Learned robot policies may produce unsafe motions outside the training distribution, requiring appropriate safeguards in real-world deployment.

REPRODUCIBILITY STATEMENT

Section 3 describes the action representation, architecture, training objective, and action extraction. Appendices A.2 and A.3 provide implementation details and pseudocode, while Appendices A.1 and A.7 provide the simulated camera setup and additional real-world experimental details. Simulated experiments use the public MimicGen benchmark (Mandlekar et al., 2023). We plan to release the code and, where possible, trained models and real-world demonstration data upon publication.

REFERENCES

Long Cheng, Jiafei Duan, Yi Ru Wang, Haoquan Fang, Boyang Li, Yushan Huang, Elvis Wang, Ainaz Eftekhar, Jason Lee, Wentao Yuan, et al. PointArena: Probing multimodal grounding through language-guided pointing. *arXiv preprint arXiv:2505.09990*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion Policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10-11):1684–1704, 2025.

Ben Eisner, Yi Yang, Todor Davchev, Mel Vecerik, Jonathan Scholz, and David Held. Deep $\mathrm{SE}(3)$-equivariant geometric reasoning for precise placement tasks. In *International Conference on Learning Representations (ICLR)*, 2024.

Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

Ankit Goyal, Jie Xu, Yijie Guo, Valts Blukis, Yu-Wei Chao, and Dieter Fox. RVT: Robotic view transformer for 3D object manipulation. In *Conference on Robot Learning (CoRL)*, pp. 694–710, 2023.

Ankit Goyal, Hugo Hadfield, Xuning Yang, Valts Blukis, and Fabio Ramos. VLA-0: Building state-of-the-art VLAs with zero modification. *arXiv preprint arXiv:2510.13054*, 2025.

Siddhant Haldar, Zhuoran Peng, and Lerrel Pinto. BAKU: An efficient transformer for multi-task policy learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 141208–141239, 2024.

Siddhant Haldar, Lars Johannsmeier, Lerrel Pinto, Abhishek Gupta, Dieter Fox, Yashraj Narang, and Ajay Mandlekar. Point Bridge: 3D representations for cross domain policy learning. *arXiv preprint arXiv:2601.16212*, 2026.

Abdullah Hamdi, Silvio Giancola, and Bernard Ghanem. MVTN: Multi-view transformation network for 3D shape recognition. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 1–11, 2021.

Boce Hu, Xupeng Zhu, Dian Wang, Zihao Dong, Haojie Huang, Chenghao Wang, Robin Walters, and Robert Platt. OrbitGrasp: $\mathrm{SE}(3)$-equivariant grasp learning. In *Conference on Robot Learning (CoRL)*, pp. 2456–2474, 2024a.

Boce Hu, Dian Wang, David Klee, Heng Tian, Xupeng Zhu, Haojie Huang, Robert Platt, and Robin Walters. 3D equivariant visuomotor policy learning via spherical projection. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 38, pp. 5685–5715, 2025.

Xixi Hu, Bo Liu, Xingchao Liu, and Qiang Liu. AdaFlow: Imitation learning with variance-adaptive flow-based policies. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 37, pp. 138836–138858, 2024b.

Haojie Huang, Dian Wang, Robin Walters, and Robert Platt. Equivariant Transporter Network. In *Robotics: Science and Systems (RSS)*, 2022.

Haojie Huang, Dian Wang, Xupeng Zhu, Robin Walters, and Robert Platt. Edge Grasp Network: A graph-based $\mathrm{SE}(3)$-invariant approach to grasp detection. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3882–3888, 2023.

Haojie Huang, Owen Howell, Dian Wang, Xupeng Zhu, Robert Platt, and Robin Walters. Fourier Transporter: Bi-equivariant robotic manipulation in 3D. In *International Conference on Learning Representations (ICLR)*, 2024a.

Haojie Huang, Karl Schmeckpeper, Dian Wang, Ondrej Biza, Yaoyao Qian, Haotian Liu, Mingxi Jia, Robert Platt, and Robin Walters. Imagination Policy: Using generative point cloud models for learning manipulation policies. In *Conference on Robot Learning (CoRL)*, pp. 5150–5165, 2024b.

Haojie Huang, Haotian Liu, Dian Wang, Robin Walters, and Robert Platt. Match Policy: A simple pipeline from point cloud registration to manipulation policies. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 16907–16914, 2025.

Wenlong Huang, Chen Wang, Yunzhu Li, Ruohan Zhang, and Li Fei-Fei. ReKep: Spatio-temporal reasoning of relational keypoint constraints for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pp. 4573–4602, 2024c.

Mingxi Jia, Haojie Huang, Zhewen Zhang, Chenghao Wang, Linfeng Zhao, Dian Wang, Jason Xinyu Liu, Robin Walters, Robert Platt, and Stefanie Tellex. Learning efficient and robust language-conditioned manipulation using textual-visual relevancy and equivariant language mapping. *IEEE Robotics and Automation Letters*, 10(8):8204–8211, 2025.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Pannag Sanketi, Quan Vuong, et al. OpenVLA: An open-source vision-language-action model. In *Conference on Robot Learning (CoRL)*, pp. 2679–2713, 2024.

Xinqing Li, Xin He, Le Zhang, Min Wu, Xiaoli Li, and Yun Liu. A comprehensive survey on world models for embodied ai. *arXiv preprint arXiv:2510.16732*, 2025.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

Chaoqi Liu, Xiaoshen Han, Jiawei Gao, Yue Zhao, Haonan Chen, and Yilun Du. OAT: Ordered action tokenization. In *Robotics: Science and Systems (RSS)*, 2026.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, pp. 1678–1690, 2021.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *Conference on Robot Learning (CoRL)*, pp. 1820–1864, 2023.

Lucas Manuelli, Wei Gao, Peter Florence, and Russ Tedrake. kPAM: KeyPoint affordances for category-level robotic manipulation. In *International Symposium of Robotics Research (ISRR)*, pp. 132–157, 2019.

OpenAI. GPT-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

Chaoyi Pan, Giridharan Anantharaman, Nai-Chieh Huang, Claire Jin, Daniel Pfrommer, Chenyang Yuan, Frank Permenter, Guannan Qu, Nicholas Boffi, Guanya Shi, and Max Simchowitz. Much ado about noising: Dispelling the myths of generative robotic control. In *International Conference on Learning Representations (ICLR)*, 2026.

Chuer Pan, Brian Okorn, Harry Zhang, Ben Eisner, and David Held. TAX-Pose: Task-specific cross-pose estimation for robot manipulation. In *Conference on Robot Learning (CoRL)*, pp. 1783–1792, 2023.

Tim Pearce, Tabish Rashid, Anssi Kanervisto, Dave Bignell, Mingfei Sun, Raluca Georgescu, Sergio Valcarcel Macua, Shan Zheng Tan, Ida Momennejad, Katja Hofmann, and Sam Devlin. Imitating human behaviour with diffusion models. In *International Conference on Learning Representations (ICLR)*, 2023.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Juntao Ren, Priya Sundaresan, Dorsa Sadigh, Sanjiban Choudhury, and Jeannette Bohg. Motion Tracks: A unified representation for human-robot transfer in few-shot imitation learning. In *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 8802–8810, 2025.

Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pp. 234–241, 2015.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-Actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pp. 785–799, 2023.

Kihyuk Sohn, Honglak Lee, and Xinchen Yan. Learning structured output representation using deep conditional generative models. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 28, pp. 3483–3491, 2015.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. LLaMA: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

Dian Wang, Stephen Hart, David Surovik, Tarik Kelestemur, Haojie Huang, Haibo Zhao, Mark Yeatman, Jiuguang Wang, Robin Walters, and Robert Platt. Equivariant Diffusion Policy. In *Conference on Robot Learning (CoRL)*, pp. 48–69, 2024.

Philipp Wu, Yide Shentu, Zhongke Yi, Xingyu Lin, and Pieter Abbeel. GELLO: A general, low-cost, and intuitive teleoperation framework for robot manipulators. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 12156–12163, 2024.

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. In *Conference on Robot Learning (CoRL)*, pp. 2475–2499, 2024.

Ge Yan, Jiyue Zhu, Yuquan Deng, Shiqi Yang, Ri-Zhao Qiu, Xuxin Cheng, Marius Memmel, Ranjay Krishna, Ankit Goyal, Xiaolong Wang, and Dieter Fox. ManiFlow: A general robot manipulation policy via consistency flow training. In *Conference on Robot Learning (CoRL)*, pp. 2268–2293, 2025.

Seonghyeon Ye, Yunhao Ge, Kaiyuan Zheng, Shenyuan Gao, Sihyun Yu, George Kurian, Suneel Indupuru, You Liang Tan, Chuning Zhu, Jiannan Xiang, et al. World action models are zero-shot policies. *arXiv preprint arXiv:2602.15922*, 2026.

Wentao Yuan, Jiafei Duan, Valts Blukis, Wilbert Pumacay, Ranjay Krishna, Adithyavairavan Murali, Arsalan Mousavian, and Dieter Fox. RoboPoint: A vision-language model for spatial affordance prediction in robotics. In *Conference on Robot Learning (CoRL)*, pp. 4005–4020, 2024.

Andy Zeng, Pete Florence, Jonathan Tompson, Stefan Welker, Jonathan Chien, Maria Attarian, Travis Armstrong, Ivan Krasin, Dan Duong, Vikas Sindhwani, and Johnny Lee. Transporter Networks: Rearranging the visual world for robotic manipulation. In *Conference on Robot Learning (CoRL)*, pp. 726–747, 2021.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Robotics: Science and Systems (RSS)*, 2023.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning (CoRL)*, pp. 2165–2183, 2023.

A Appendix

A.1 Simulation Details

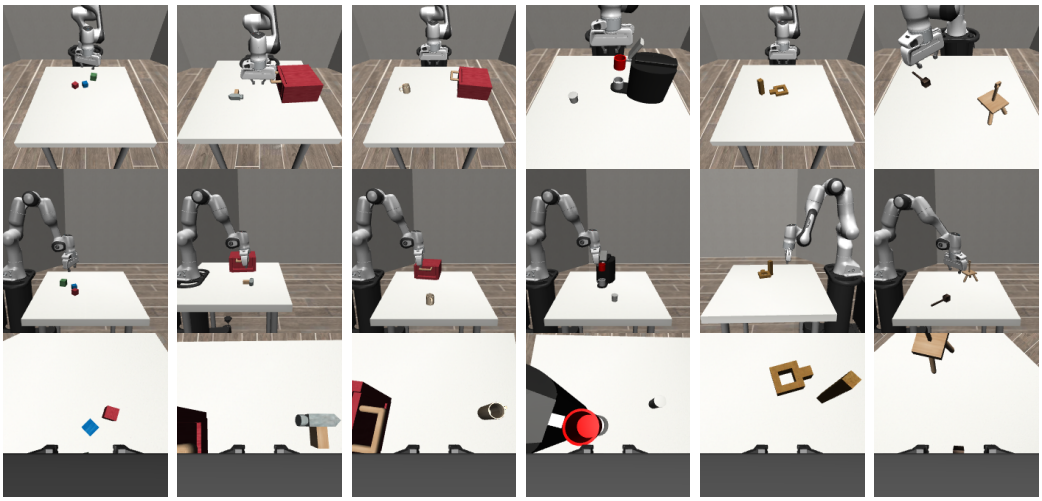


Figure 10: Simulation camera setup for the six MimicGen tasks. The first row shows side-view camera 1, the second row shows side-view camera 2, and the last row shows the in-hand view. All images are rendered at a resolution of 224×224.

A.2 Implementation Details

To keep the action keypoints within the image space during training, we truncate each action chunk at the first out-of-view keypoint and replace the remaining keypoint configurations with the last valid in-view configuration. Since the baselines are trained on the original actions, this truncation removes part of the supervision available to AMP on the affected tasks (mug-cleanup-d1, coffee-d2, and threading-d2), so our reported results on these tasks are likely conservative. We also apply extensive data augmentation to AMP. Since the actions are represented in the same image space, each image can be jointly rotated and translated together with its corresponding action heatmap labels. Specifically, for each view, including side view 1, side view 2, and the in-hand view, we apply a random rotation and translation with probability 0.5, transforming the labels accordingly. Rotations are sampled uniformly from $[-\pi/6, \pi/6]$, and translations are sampled uniformly from $[-H/6, H/6]$, where H is the image size. This exposes the model to multiple transformed variants of similar training samples, improving robustness and reducing overfitting. We discuss the benefits of equivariant data augmentation in more detail in Appendix A.4. Our transformer consists of 6 attention layers with 8 heads each and a hidden dimension of 768, yielding approximately 41.2M parameters in total. We train the model for 200 epochs on a single NVIDIA RTX 5090 GPU, using a learning rate of $1\mathrm{e}{-4}$ with a cosine annealing schedule.

Table 3 also reports a parameter study on the soft-label width $\sigma \in \{0, 2, 4\}$. The width σ controls how sharply the target probability mass is concentrated around the projected keypoint. We find that an intermediate value strikes the best balance. When $\sigma = 0$, the label approaches a one-hot target with all mass on a single pixel: the supervision signal is sparse and the loss is dominated by exact-pixel matches, which provides little gradient for nearby predictions and makes training sensitive to small projection errors. When $\sigma = 4$, the label becomes overly diffuse and spreads mass across a large neighborhood, blurring the target and reducing localization precision, which in turn degrades the accuracy of the triangulated 3D keypoints. A moderate $\sigma = 2$ retains a well-localized peak while still providing a smooth gradient in the surrounding region, yielding the highest success rate across all three tasks.

A.3 Pseudocode

Algorithms 1 and 2 summarize the training and inference of Action Map Policy in the notation of Section 3. The X-Net predictor ψ_θ and the action extractor f compose into the policy $\pi = f \circ \psi$.

POSE2KP maps a pose to its $m{=}5$ keypoints $p^{1:m}{:=}(p^1,\dots,p^m)$ (Figure 3) and f is its geometric inverse; $\mathcal{P}_k$ projects a 3D point into side view k, and $\mathcal{T}$ triangulates the matched pixels from the $n{=}2$ side views back to 3D. Soft labels $\mathbf{h}_{ijk}$ are $\sigma{=}2$ Gaussians and a chunk spans l steps. For brevity each line is written once but applies over all timesteps $i{\le}l$, keypoints $j{\le}m$, and side views $k{\le}n$. In training, lines 1–4 turn the action chunk into heatmap labels while lines 5–7 predict the heatmaps and update θ; CROSSENTROPY is the pixel-wise loss of Section 3.

Algorithm 1 Action Map Policy – Training step

Require: demo $(\mathcal{O},\mathcal{A})$; ψ_θ; rate η
1: $p_i^{1:m} \leftarrow$ POSE2KP(a_i)
2: $(u,v)_{ijk} \leftarrow \mathcal{P}_k(p_i^j)$
3: jointly augment $\mathcal{O}$ and $(u,v)_{ijk}$
4: $\mathbf{h}_{ijk} \leftarrow \mathcal{N}\big((u,v)_{ijk},\sigma^2\big)$
5: $\hat{\mathbf{h}}_{ijk} \leftarrow \operatorname{softmax}\psi_\theta(\mathcal{O})$
6: $\mathcal{L}_{\text{CE}} \leftarrow$ CROSSENTROPY$(\hat{\mathbf{h}}_{ijk},\mathbf{h}_{ijk})$
7: $\theta \leftarrow \theta - \eta\,\nabla_\theta\mathcal{L}_{\text{CE}}$

Algorithm 2 Action Map Policy – Inference $(\pi = f\circ\psi)$

Require: obs $\mathcal{O}$; ψ_θ, $\mathcal{T}$, f
1: $\hat{\mathbf{h}}_{ijk} \leftarrow \operatorname{softmax}\psi_\theta(\mathcal{O})$
2: $(u,v)_{ijk} \leftarrow \arg\max_{(u,v)}\hat{\mathbf{h}}_{ijk}$
3: $p_i^j \leftarrow \mathcal{T}\big(\{(u,v)_{ijk}\}_{k=1}^n\big)$
4: $a_i{=}(T_i,R_i,w_i) \leftarrow f(p_i^{1:m})$
5: **return** $\mathcal{A}=(a_1,\dots,a_l)$
6: execute first l_e steps, then re-plan

A.4 EQUIVARIANT DATA AUGMENTATION FOR HEATMAP PREDICTION

Representing actions as heatmaps in observation space preserves the task's geometric symmetries. Specifically, rotations and translations applied to the input images induce corresponding transformations in the predicted heatmaps. Moreover, transformations applied independently to each view result in independent transformations of the corresponding heatmaps without affecting others. This property corresponds to the n-equivariance, which we extend to heatmap prediction under a classification formulation:

$$\psi(GO) = G\,\psi(O), \tag{1}$$

where $G=\operatorname{diag}(g_1,g_2,g_3)$ with $g_i\in\mathrm{SE}(2)$ denotes independent transformations applied to each observation, and $O=\operatorname{diag}(o_1,o_2,o_3)$ denotes the stacked multi-view observations. Since heatmaps are predicted only for the two side views, the output is equivariant to transformations of the side views and invariant to those of the in-hand view.

We apply equivariant data augmentation to realize the symmetries by jointly transforming each observation and its corresponding ground-truth heatmaps with random rotations and translations. This encourages the policy to learn symmetry-consistent mappings, enlarges the support of the data distribution, and biases the model toward responding to local image features, as it must steer the end effector to manipulate objects under varying transformations. As shown in Table 3, removing equivariant data augmentation reduces the success rate by 12–32%, the largest drop among all ablated components.

A.5 ABLATION ON THE NUMBER OF KEYPOINTS

The number of keypoints is a design choice of our action representation. Recovering a full end-effector action requires at least four keypoints: three non-collinear points determine the 6-DoF pose, and one additional point encodes the gripper width. We compare this minimal 4-keypoint configuration with our default 5-keypoint design (Figure 3) in Table 6. Using 5 keypoints improves the success rate by 6% on stack-three-d1 and 4% on square-d2. We attribute this gain to redundancy in pose recovery: with four grasp points p^1–p^4, both the antipodal and approach axes are computed as the average of two difference vectors, and the translation as the centroid of four points (Section 3), which reduces the influence of localization errors of individual keypoints. In contrast, the minimal configuration estimates each axis from a single pair of points. Nevertheless, the 4-keypoint variant still outperforms all baselines on both tasks (Table 2), indicating that the number of keypoints is a hyperparameter worth exploring rather than a limitation of AMP; additional keypoints may further improve robustness at the cost of more output channels.

Table 6: Ablation on the number of keypoints: success rate (%).

# Keypoints	stack-three-d1	square-d2
4 (minimal)	84	46
5 (default)	**90**	**50**

A.6 Real-World Heatmap-Prediction Visualization

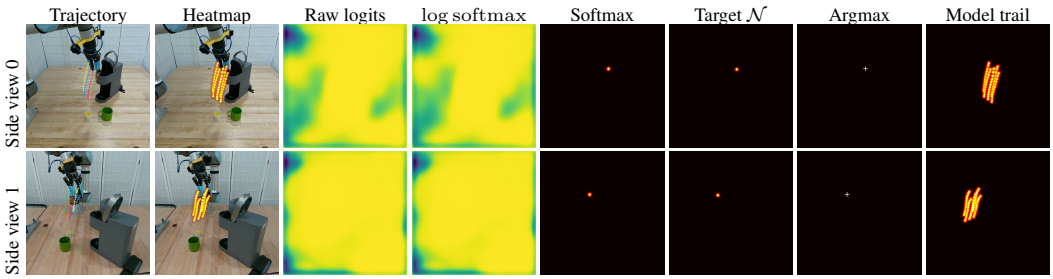


Figure 11: Heatmap prediction visualization on the MAKE-COFFEE task, evaluated at the midpoint of a downward grasp toward the coffee pod. Each row corresponds to one calibrated side view. The columns show the dense prediction process, including the raw decoder output, cross-entropy supervision, spatial softmax, argmax pixel selection, and the final horizon-wide trajectory. Columns (1)–(2) summarize the full 5×12 keypoint-step output of the policy, while Columns (3)–(7) visualize the image-space distribution of a single keypoint.

The X-Net casts trajectory prediction as a dense per-pixel classification problem. For every keypoint $j\in\{1,\dots,m\}$ and every future timestep $i\in\{1,\dots,l\}$, it outputs a single 224×224 score map for each side view. We do not apply any non-linearity to this map before supervision. During training, the raw logits are optimized with a spatial cross-entropy loss against a Gaussian target. During inference, we take the $\arg\max$ pixel and lift the predicted pixels from two calibrated views into a 3D keypoint cloud via triangulation.

To make this pipeline transparent, we visualize each intermediate quantity for the MAKE-COFFEE task on a single frame captured midway through a downward grasping motion. From left to right in Figure 11, the eight columns are:

- **(1) Trajectory.** Per-keypoint $\arg\max$ pixels over the full $l=12$ prediction horizon, connected as polylines and overlaid on the native-resolution side-view image. This is the 2D output the policy commits to *before* triangulation.
- **(2) Heatmap.** The same prediction in the heatmap domain: the per-step spatial-softmax maps are summed over the 5 keypoints, peak-normalized, max-merged across the horizon, and alpha-blended onto the native background. The bright "trail" is the heatmap analogue of the trajectory in column (1).
- **(3) Raw logits.** The unactivated output of the X-Net decoder, $\mathrm{logit}(u,v)\in\mathbb{R}$, rendered by linearly stretching its values to a colormap.
- **(4) Log-softmax.** The quantity used by the cross-entropy loss, obtained by applying $\log\mathrm{softmax}$ over all image pixels. Its range is $(-\infty,0]$, with 0 at the peak. Since $\log\mathrm{softmax}$ differs from the raw logits only by an additive normalization term, its spatial pattern matches column (3).
- **(5) Spatial softmax.** Exponentiating the logits and normalizing them to unit mass over all pixels converts the raw score map into a probability distribution, often appearing as a sharply peaked heatmap.
- **(6) Training target.** The supervision used during training: a $\sigma=2$ Gaussian centered at the ground-truth keypoint pixel and normalized to unit mass.
- **(7) Argmax.** The single highest-scoring pixel, which is the only information passed downstream to two-view triangulation.

- **(8) Model trail.** A horizon-wide rendering of the policy prediction, obtained by stacking the spatial-softmax maps for all 5 keypoints and l=12 steps, peak-normalizing each step, and taking the maximum across steps.

Two observations follow. First, columns (3) and (5) encode the same ordering over pixels, but their visual appearance can be very different: spatial softmax can turn a nearly flat logit map into a compact, Gaussian-like blob. Second, the model prediction in column (5) closely matches the training target in column (6), suggesting that the cross-entropy objective has converged well in the heatmap domain.

A.7 Real-World Experiment Details

In the real-world experiments, we apply color jittering and use exactly the same observations for our method and all baselines to ensure a fair comparison. The results in Table 4 show that our method substantially outperforms the baselines across all tasks. In addition, our method achieves an inference latency of 13.80 ms. This latency is measured from receiving the observation to outputting the full trajectory commands, rather than only the model forward-pass time. All inference times are measured on a single NVIDIA RTX 3090 GPU. We note that the SVD-based triangulation step contributes non-negligible overhead, and can be further optimized. We additionally evaluate Motion Tracks (Ren et al., 2025) on toast-bread using the authors' official implementation, which achieves only a 5% completion rate, since its independently denoised per-view keypoint trajectories are often inconsistent across views and yield erroneous triangulated actions.

A.8 Real-World Experiments: Failure Case Analysis

We report the completion rate without partial credit for each task. The main failure mode of our method is imprecise grasping and insertion, occurring primarily under out-of-distribution evaluation. These failures are more severe for the Diffusion Policy and ACT baselines, which commonly either get stuck at a fixed location or collapse toward the averaged position of several candidate objects. In contrast, AMP rarely exhibits this collapse, as its classification-based formulation preserves the multi-modal distribution over candidate grasps.

A.9 Robustness, Consistency, and Scalability Details

Robustness in practice. Our real-world results already provide practical evidence of robustness: we use off-the-shelf RealSense cameras whose calibration deviates from the true camera matrices, and robot motion slightly shakes the table during execution, perturbing the extrinsics. AMP performs well under both conditions. Moreover, whereas most visuomotor policies absorb camera geometry implicitly into the network, AMP decouples the camera model from the network, which allows us to attribute errors to their two sources, as analyzed below.

Network error under transformed observations. We evaluate the network on observations transformed by rotations of up to $\pm 30^\circ$ and translations of up to $\pm H/6$ (within the augmentation range). The error of the argmax keypoint predictions w.r.t. the correspondingly transformed labels is only 0.51 ± 0.03 px, i.e., below one pixel, showing that the network itself is robust to such transformations.

Triangulation under camera noise. To isolate the effect of calibration errors from the network, we perturb the camera extrinsic translation or focal length used for SVD-based triangulation and measure the position and rotation errors of the reconstructed actions. As shown in Table 7, small noise ($2\,\mathrm{mm}$ extrinsic translation or 0.5% focal length) leads to errors within $3\,\mathrm{mm}$ and 0.2°, whereas large noise ($8\,\mathrm{mm}$ or 8%) increases them to about $1\,\mathrm{cm}$ and up to 1.6°, which can degrade performance in extreme cases.

Table 7: Triangulation precision under camera extrinsic and intrinsic noise (small / large noise).

Noise	Pos. (cm)	Rot. (deg)
extr. trans. $2/8\,\mathrm{mm}$	$0.28/1.14$	$0.18/0.73$
focal len. $0.5\%/8\%$	$0.11/0.88$	$0.20/1.63$

Consistency. Since X-Net predicts the heatmaps of all views jointly through cross-view attention, AMP naturally encourages cross-view consistency. We observe no inconsistent motion in the rollouts,

which remain smooth (see supplementary videos). Quantitatively, the MSE between the triangulated and ground-truth actions (6D rotation, translation in meters, and gripper width) is only 1×10^{-4}–2×10^{-4} on the validation sets of all five real-world tasks. As inconsistent per-view predictions would lead to much larger triangulation errors, this indicates that the maxima of the learned multi-modal distributions are geometrically consistent across views. We also tried an explicit consistency loss (Jensen–Shannon divergence between the predictions of two consecutive steps), which brought no further improvement, suggesting that consistency is already learned implicitly.

Hardware and camera placement. Although scalability is not the focus of this work, nothing in AMP ties it to a specific robot or camera placement: MimicGen already uses different camera placements across tasks (e.g., coffee-d2 and threading-d2 in Figure 5; all views in Figure 10), and our real-world setup differs from simulation in both the robot and the cameras.

Data scaling. To study data scaling within our computational budget, Table 8 reports the success rates of AMP with 100 and 200 demonstrations on two simulated tasks. Doubling the demonstrations improves the success rate by 8% on stack-three-d1 and 14% on square-d2, suggesting that AMP continues to benefit from additional data. Beyond these results, we believe AMP has strong potential for scaling up for two reasons: (1) observations and actions live in the same image space, which provides a natural interface to pretrained vision and vision-language models; and (2) its classification objective is the same type of objective that has been shown to scale effectively in large language models (OpenAI, 2023; Touvron et al., 2023), and it aligns naturally with vision-language models, offering a unified interface for jointly formulating vision, language, and action.

Table 8: Data scaling in simulation: success rate (%) with 100 and 200 demonstrations.

Task	100 demos	200 demos
stack-three-d1	90	98
square-d2	50	64

A.10 Discussion: AMP, VLAs, and World Action Models

To better position AMP within current policy-learning frameworks, we view it as a *geometric* approach: it first predicts a geometrically rich intermediate output and then extracts low-level actions from it. World action models (WAMs) (Ye et al., 2026) follow a similar two-stage principle, first generating videos of future states and then extracting actions from them. In contrast, vision-language-action models (VLAs) (Zitkovich et al., 2023; Kim et al., 2024) directly map observations to low-level actions such as end-effector poses. Compared with video-action generation approaches (Ye et al., 2026; Li et al., 2025), which predict future observations at the image level and tend to be computationally heavy, AMP only predicts pixel-level distributions over action keypoints, making it more efficient. Moreover, AMP generates explicit action distributions in a single forward pass, whereas VLAs and WAMs typically require multiple sequential steps, such as autoregressive token decoding or iterative denoising, which slows down inference and leaves the action distribution implicit or factorized token by token.