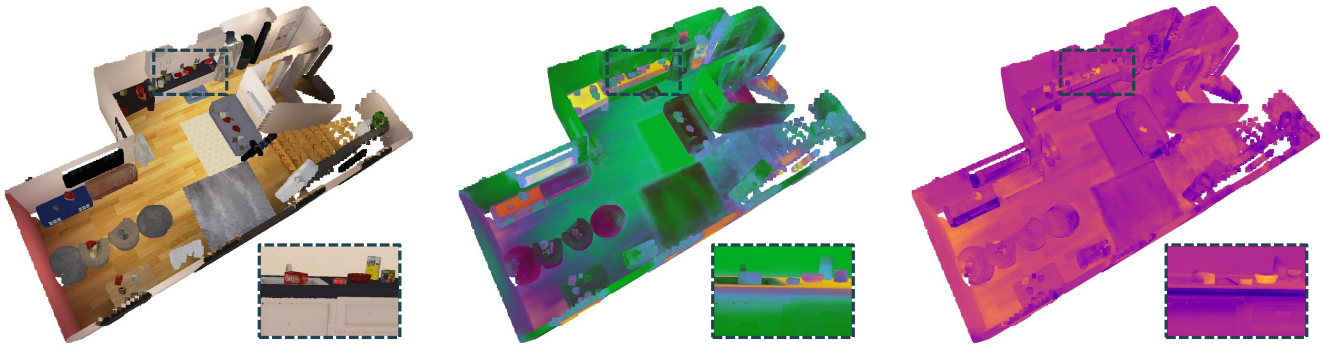
2026 IEEE International Conference on Robotics and Automation (ICRA 2026)
June 1-5, 2026. Vienna, Austria

Seeing the Bigger Picture: 3D Latent Mapping for Mobile Manipulation Policy Learning

Sunghwan Kim Woojeh Chung Zhirui Dai Dwait Bhatt Arth Shukla
Hao Su Yulun Tian Nikolay Atanasov



(a) ReplicaCAD Scene (b) 3D Latent Map (c) Attention Score

Fig. 1: (a) An RGB rendering of a ReplicaCAD [1] scene. (b) A 3D latent feature map constructed by our method, visualized with PCA. (c) Attention weights on the latent map during task execution (*e.g.*, "pick up the bowl"), highlighting regions attended by the policy model.

***Abstract*— In this paper, we demonstrate that mobile manipulation policies utilizing a *3D latent map* achieve stronger spatial and temporal reasoning than policies relying solely on images. We introduce *Seeing the Bigger Picture* (SBP), an end-to-end policy learning approach that operates directly on a 3D map of latent features. In SBP, the map extends perception beyond the robot's current field of view and aggregates observations over long horizons. Our mapping approach incrementally fuses multiview observations into a grid of scene-specific latent features. A pre-trained, scene-agnostic decoder reconstructs target embeddings from these features and enables online optimization of the map features during task execution. A policy, trainable with behavior cloning or reinforcement learning, treats the latent map as a state variable and uses global context from the map obtained via a 3D feature aggregator. We evaluate SBP on scene-level mobile manipulation and sequential tabletop manipulation tasks. Our experiments demonstrate that SBP (i) reasons globally over the scene, (ii) leverages the map as long-horizon memory, and (iii) outperforms image-based policies in both in-distribution and novel scenes, *e.g.*, improving the success rate by 15% for the sequential manipulation task.**

I. Introduction

Recent advances in robot learning have led to remarkable progress in manipulation in semi-structured environments [2]–[4]. State-of-the-art systems use large vision–language models (VLMs), whose rich semantic priors and cross-modal reasoning translate natural language commands directly into low-level actions [2]. The next frontier lies in extending these methods beyond fixed tabletop setups to long-term mobile manipulation at room, building, neighborhood, and even larger scales. However, existing methods rely on *2D image-based* designs that operate directly on raw video streams. While effective for short-term action prediction, image-based approaches struggle with consistent 3D understanding and long-horizon reasoning—two capabilities critical for spatially or temporally extended tasks.

In this work, we advocate for an alternative *3D map-based* design that conditions robot policy learning on an explicit 3D representation of the environment. A growing body of recent work explores 3D scene representations for manipulation. Some methods lift 2D foundation-model features into 3D on a per-frame basis [5]–[8]. Another line of work encodes raw point cloud observations directly with specialized 3D backbones [9], [10]. While both families preserve geometry and enhance local scene understanding, reconstructing the scene from scratch at each time step compromises temporal consistency and hinders long-horizon reasoning. Complementary efforts fuse multiview observations into feature fields offline [11]–[13]. Although these feature fields improve multiview consistency, they are confined to tabletop setups where the entire workspace remains visible at every time step and cannot adapt on the fly to novel views.

A key idea in this paper is to condition the robot policy on global features obtained from a persistent 3D map (see Fig. 1), built incrementally from streaming observations. Persistent maps have long benefited robot navigation [14], [15], yet their

We gratefully acknowledge support from NSF CCF-2402689 (ExpandAI), ONR N00014-23-1-2353, and the Technology Innovation Program (20018112, Development of autonomous manipulation and gripping technology using imitation learning based on visual and tactile sensing) funded by the Ministry of Trade, Industry & Energy (MOTIE), Korea.

S. Kim, Z. Dai, D. Bhatt, and N. Atanasov are with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA 92093, USA, e-mail: {suk063, zhdai, dhbhatt, natanasov}@ucsd.edu.

W. Chung, A. Shukla, and H. Su are with the Department of Computer Science and Engineering, University of California San Diego, La Jolla, CA 92093, USA, e-mail: {w5chung, arshukla, haosu}@ucsd.edu.

Y. Tian is with the Robotics Department, University of Michigan, Ann Arbor, MI 48109, USA, e-mail: yulunt@umich.edu.

potential to enhance manipulation remains under-explored. Such maps offer two key advantages for mobile manipulation. First, they act as *spatial memory*, offering global visibility of object locations and task goals while mitigating occlusions from the current field of view. Second, by accumulating information over time, they provide *long-term context* that enables policies to reason beyond short observation windows. Fig. 1b shows a latent map containing spatially grounded language features, generated by our method. Fig. 1c illustrates that a policy conditioned on the map may attend to the entire scene, leveraging spatially and temporally extended context.

Contributions. We present *Seeing the Bigger Picture* (SBP), an end-to-end policy learning method operating directly on an incrementally constructed 3D latent feature map. We propose a modular design that comprises (i) a *scene-specific* latent feature grid that aggregates and compresses multiview visual observations, and (ii) a *pre-trained*, *scene-agnostic* decoder that reconstructs target embeddings (*e.g.*, CLIP [16]) from the latent features and generalizes to diverse scenes. During task execution, our approach enables online optimization of latent features by using the pre-trained decoder. To harness the 3D latent map in mobile manipulation tasks, we propose a 3D feature aggregator that summarizes the spatially distributed features into a compact *global map token* that provides global context to support robot policy learning. The global map token can be integrated into a behavior cloning or reinforcement learning policy, and we show that the resulting policy improves long-horizon reasoning and task execution. Our contributions are summarized as follows.

- We propose a mapping approach that incrementally builds a 3D map of latent features. Its modular design decouples scene-specific feature optimization from a scene-agnostic feature decoder, enabling generalization across different environments.
- We design a policy that treats the map as a state and tokenizes its features with a 3D feature aggregator to improve spatial and temporal reasoning. The model supports both behavior cloning and reinforcement learning.
- We demonstrate that our SBP method reasons globally and leverages the map as spatiotemporal memory in long-horizon manipulation tasks, outperforming image-based policies in both in-distribution and novel scenes.

II. Related Work

A. 3D Feature Mapping with Large Vision Models

A growing body of work distills vision model features, *e.g.*, CLIP [16] and DINOv2 [17], into neural scene representations for grounding semantics in 3D. Early work such as LERF [18] embeds vision-language features into implicit neural fields by enforcing multiview consistency, enabling open-vocabulary queries. To achieve fine-grained segmentation, subsequent works adopt explicit 3D representations (*e.g.*, Gaussian splatting [19]) and leverage SAM [20] to obtain precise object boundaries [21], [22]. Recent work extends these representations to dynamic environments via online updates [23] and 4D scene representations for moving objects [24], [25].

Several works structure the 3D features to support downstream manipulation tasks, such as 6-DoF grasping. F3RM [12] distills CLIP features into a hierarchical 3D feature grid, enabling few-shot grasping by optimizing grasp poses based on language queries. LERF-TOGO [11] addresses non-uniform activations in LERF [18] via DINO-based masking, improving part-aware grasp selection. GeFF [26] eliminates the need for per-scene optimization by introducing a generalizable NeRF [27] encoder that predicts language-aligned features. OK-Robot [28] presents a modular pipeline for language-conditioned household manipulation, while DynaMem [29] utilizes a dynamic 3D voxel map with semantic vectors for object localization and navigation. More recently, UAD [30] distills visual affordance maps from VLMs into a task-conditioned predictor that generalizes in the wild.

Despite their strong 3D grounding capabilities, these methods focus on localizing objects or predicting affordances from language queries rather than leveraging the representations for end-to-end policy learning. To bridge this gap, we introduce a 3D latent mapping approach designed to facilitate policy training for long-horizon tasks.

B. Manipulation Policy Learning with 3D Representations

3D scene representations encode explicit geometric structure (*e.g.*, free space, surfaces) that is absent from 2D images, providing a stronger spatial inductive bias for manipulation. As a result, conditioning manipulation policies on 3D representations improves generalization and sample efficiency.

One line of work lifts features from 2D large vision models into 3D structures. PerAct [31] encodes RGB-D observations into voxel grids and predicts end-effector poses via a transformer. Act3D [5] lifts multi-scale CLIP features into a 3D feature cloud and scores candidate points via cross-attention to regress manipulator poses. This has been extended to generate action trajectories by conditioning diffusion models on the featurized 3D scene [6]. GNFactor [7] trains a neural feature field that jointly reconstructs 2D foundation-model features and predicts robot actions in 3D.

Another line of work bypasses 2D feature lifting and learns policies directly from geometric observations like point clouds. DP3 encodes raw point clouds into compact embeddings that condition a diffusion policy for manipulation [3], [9]. Some approaches render the scene and apply multi-stage transformers that zoom in on regions of interest [32]. Recent work fuses point clouds with DINO features, forming semantic fields that enable part-level generalization across object instances [8]. EquiBot [10] combines $\mathrm{SIM}(3)$-equivariant architectures with a point cloud diffusion policy for scale, rotation, and translation invariance, while ActionFlow [33] leverages $\mathrm{SE}(3)$-invariant point relations with flow matching for symmetry-preserving trajectories.

Despite recent progress, most approaches treat 3D representations as instantaneous observations, reconstructing them at each step. We address this by learning end-to-end policies conditioned on a persistent 3D latent map, enabling global and long-horizon reasoning.

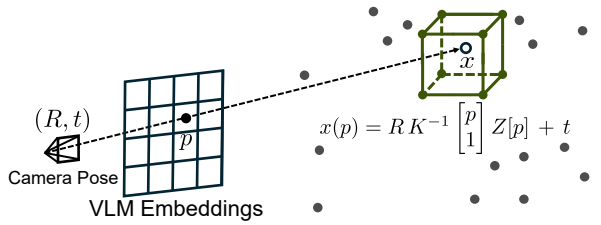


Fig. 2: Visualization of per-patch VLM embeddings back-projected into the 3D world frame using depth $Z[p]$ and camera pose (R, t).

III. Problem Formulation

This paper has two primary objectives: first, to incrementally construct a 3D latent feature map of a robot's workspace (Sec. III-A), and second, to design manipulation policies that use the map as a state variable to execute tasks specified in natural language (Sec. III-B).

A. Latent Feature Mapping

Let $\mathcal{X} \subseteq \mathbb{R}^3$ denote the robot's workspace and $\mathcal{Y} \subseteq \mathbb{R}^k$ a target embedding space (*e.g.*, of language features such as CLIP [16]). We seek an efficient representation $\mathcal{M} : \mathcal{X} \to \mathcal{Y}$ that maps 3D workspace points to target embeddings.

Problem 1. Given a dataset of coordinate-feature pairs $\mathcal{D} = \{(x, y)\} \subset \mathcal{X} \times \mathcal{Y}$, find a map $\mathcal{M}$ that minimizes the reconstruction loss:

$$\min_{\mathcal{M}} \mathbb{E}_{(x,y)\sim\mathcal{D}}\big[\mathcal{L}\big(\mathcal{M}(x), y\big)\big], \tag{1}$$

where $\mathcal{L} : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ is a distance function on $\mathcal{Y}$.

To instantiate Problem 1 for learning language-grounded latent maps, we use dense visual features extracted from a VLM as the target labels y. Given an RGB image o, a depth image Z, and camera pose (R, t), we compute per-patch embeddings $G \in \mathbb{R}^{k\times(H\times W)}$ by feeding the image through the VLM's vision encoder. Following the ViT [34] convention, we partition the image into $H \times W$ non-overlapping patches, each producing a k-dimensional feature embedding at its corresponding spatial location. We back-project each patch p with valid depth $Z[p]$ into the world frame using the camera intrinsics K and pose (R, t), as shown in Fig. 2:

$$x(p) = R\,K^{-1}\begin{bmatrix}p\\1\end{bmatrix} Z[p] + t. \tag{2}$$

Each 3D point $x(p)$ is paired with its embedding $y(p) = G[p]$ from the VLM's vision encoder, yielding a coordinate-feature pair (x, y). By aggregating these pairs across multiple viewpoints, we construct a training set $\mathcal{D}$. Minimizing (1) ensures that the latent map captures the semantics provided by the VLM and associates them with 3D spatial locations.

B. Map-Conditioned Policy Learning

We consider a manipulator that executes language-specified instructions. The task specification is embedded in a vector e_ℓ described below. At each time step τ, the robot receives its proprioceptive state s_τ (*e.g.*, joint angles) and observations o_τ (*e.g.*, RGB images). A key feature of our formulation is the 3D latent map $\mathcal{M}_\tau$, which is built incrementally from the observations and serves as persistent spatial memory of the environment (see Fig. 1). A policy π_ϕ with parameters ϕ maps $\mathcal{M}_\tau$, state s_τ, the observation o_τ, and the task embedding e_ℓ to an action a_τ (*e.g.*, joint velocities).

The policy is trained in two settings. In behavior cloning (BC) [35], the policy learns to mimic expert actions from a demonstration dataset $\mathcal{D}^*$. For BC, we use a text embedding of a command (*e.g.*, "pick up the bowl") as the task embedding e_ℓ. In reinforcement learning (RL), the policy is trained to maximize expected sum of discounted rewards, $\mathbb{E}_{\mathcal{T}\sim\pi_\phi}[\sum_\tau \gamma^\tau R_\tau]$. The reward R_τ is shaped to facilitate learning, with bonuses for completing subgoals such as reaching, grasping, or placing an object. For RL, we use a learnable task embedding specifying the target object [36].

Problem 2. Determine policy parameters ϕ that enable the robot to complete its task by solving $\min_\phi \mathcal{J}(\pi_\phi)$, where the form of $\mathcal{J}(\pi_\phi)$ depends on the learning method:

$$\mathcal{J}(\pi_\phi) = \begin{cases} -\mathbb{E}_{(\cdot,a_\tau^*)\sim\mathcal{D}^*}\left[\log \pi_\phi(a_\tau^* \mid \mathcal{M}_\tau, o_\tau, s_\tau, e_\ell)\right] & \text{(BC)} \\ -\mathbb{E}_{\mathcal{T}\sim\pi_\phi}\left[\sum_\tau \gamma^\tau R_\tau\right] & \text{(RL)} \end{cases} \tag{3}$$

IV. Latent Feature Mapping

In this section, we present our latent mapping approach, grounded in two principles. (1) *Incremental updates*: A 3D latent map continuously integrates multiview observations to serve as persistent spatial memory. (2) *Modularity*: A scene-specific representation is decoupled from a scene-agnostic mapping function to enable generalization across scenes.

To realize this modularity, we model the latent map as an encoder-decoder architecture, $\mathcal{M}{=}(F_\psi, D_\theta)$. The encoder $F_\psi : \mathcal{X} \to \mathcal{F}$ lifts workspace points $x \in \mathcal{X}$ to a latent space $\mathcal{F}$, and the decoder $D_\theta : \mathcal{F} \to \mathcal{Y}$ projects latent features $f \in \mathcal{F}$ to the target space $\mathcal{Y}$. The intermediate space $\mathcal{F}$ enables the map to capture the geometric and semantic structure of the environment more effectively than a direct $\mathcal{X} \to \mathcal{Y}$ mapping [18], [37], [38]. This architecture disentangles the scene-specific encoder parameters ψ from the scene-agnostic decoder parameters θ. Pre-training the decoder on diverse environments enables fast adaptation to new environments.

A. Multiresolution Feature Grid

We represent the scene as learnable latent vectors anchored at the vertices of a 3D grid. These vectors act as spatial memory that is updated incrementally as new observations arrive. Let $\mathcal{G} = \{(z_i, f_i)\}_{i=1}^M$, where each vertex position $z_i \in \mathcal{X}$ stores a latent vector $f_i \in \mathcal{F}$. For a query point $x \in \mathcal{X}$, its feature is retrieved by trilinear interpolation of the eight vertex features of the voxel containing x,

$$f(x) = \sum_{i\in\mathcal{N}(x)} w(x, z_i)\, f_i, \tag{4}$$

where $\mathcal{N}(x)$ is the index set of the neighboring vertices and $w(\cdot,\cdot)$ is the trilinear interpolation function.

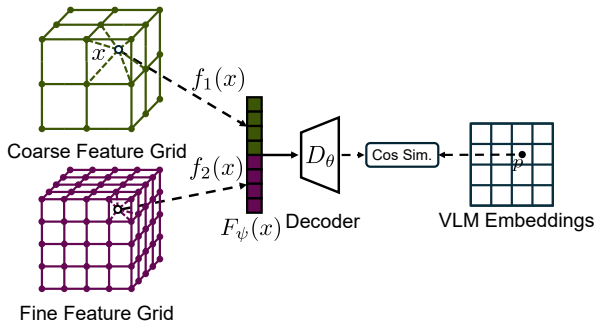


Fig. 3: **Latent feature mapping.** We represent the scene with a multiresolution feature grid. For any query point x, we retrieve features from each level via trilinear interpolation, concatenate them to form $F_\psi(x)$, and decode with D_θ to reconstruct the target embedding. The model is trained to maximize similarity between predicted and ground-truth embeddings.

To capture information at multiple scales, we use a hierarchy of L grids $\{\mathcal{G}_l\}_{l=1}^{L}$, ranging from coarse ($l=1$) to fine ($l=L$) resolutions, based on the design proposed in [37], [38] (see Fig. 3). Let $f_{l,i} \in \mathbb{R}^c$ denote the latent vector at vertex $z_{l,i}$ of grid $\mathcal{G}_l = \{(z_{l,i}, f_{l,i})\}_{i=1}^{M_l}$, where M_l is the number of vertices at level l. The collection of all latent vectors $\psi = \{\, f_{l,i} \mid l{=}1{:}L,\ i{=}1{:}M_l\}$ constitutes the scene-specific map parameters. The interpolated feature at level l is $f_l(x)$. Concatenating the features across all levels yields the final feature representation:

$$F_\psi(x) = \bigoplus_{l=1}^{L} f_l(x) \in \mathcal{F} \subseteq \mathbb{R}^d, \quad d = L\,c. \tag{5}$$

where $\bigoplus$ denotes concatenation. For efficient implementation, we use a hash-based voxel grid, following [37].

B. Latent Map Optimization

The decoder D_θ maps a latent feature $F_\psi(x)$ to the target space $\mathcal{Y}$. The predicted feature of any query point $x \in \mathcal{X}$ is

$$\hat{y}(x) = D_\theta\big(F_\psi(x)\big) \in \mathcal{Y} \subseteq \mathbb{R}^k. \tag{6}$$

We implement D_θ as a multilayer perceptron (MLP). The decoder is pre-trained on scenes from diverse environment configurations to learn a general mapping from the latent space $\mathcal{F}$ to the target space $\mathcal{Y}$. Intuitively, $F_\psi(x)$ serves as a multiview-aggregated, compressed representation of target feature embeddings, while D_θ is trained to reconstruct them back into the target space.

During task execution, the pre-trained decoder can be kept frozen. Adaptation to new environments is accelerated by focusing optimization on the grid parameters ψ. The parameters ψ and, optionally, θ are optimized by minimizing the loss $\mathcal{L}$ in (1). We align the predicted feature $\hat{y}(x)$ with the reference y using the cosine distance loss, which empirically outperforms alternatives such as the L_2 loss. Given a dataset $\mathcal{D}$ of coordinate-feature pairs (x, y), the objective is:

$$\min_{\psi,\theta} \frac{1}{|\mathcal{D}|} \sum_{(x,y)\in\mathcal{D}} \big[1 - \cos\big(\hat{y}(x), y\big)\big]. \tag{7}$$

Algorithm 1 Online Latent Map Update

Require: pre-trained decoder D_θ, learning rate η.
Initialize map features ψ_0 from a pre-trained offline map.
for each time step $\tau = 1, 2, \ldots$ **do**
  Set map features $\psi_\tau \leftarrow \psi_{\tau-1}$.
  if $\tau \pmod{T_{\text{update}}} \equiv 0$ **then**
    Generate $\mathcal{D}_\tau$ from $\big(o_\tau, Z_\tau, (R_\tau, t_\tau)\big)$.
    for $k = 1, \ldots, K_{\text{update}}$ **do**
      $\psi_\tau \leftarrow \psi_\tau - \eta \nabla_\psi \mathcal{L}(\mathcal{D}_\tau; \psi_\tau)$
  $a_\tau \sim \pi_\phi(\cdot|\mathcal{M}_\tau, o_\tau, s_\tau, e_\ell)$, where $\mathcal{M}_\tau = (F_{\psi_\tau}, D_\theta)$.

For each environment configuration (*i.e.*, scene and object arrangement), we optimize a dedicated feature grid. The decoder is jointly pre-trained across all configurations for a given task. Our mapping approach omits 3D positional encoding to avoid overfitting to absolute coordinates and to enable cross-scene generalization between different environments. Fig. 3 summarizes the overall mapping approach, instantiated for language-grounding as described in Sec. III-A.

C. Online Latent Mapping

To account for changes in the environment as the task state progresses (*e.g.*, object reconfiguration in multi-stage tasks), we update the latent map online as detailed in Algorithm 1. At the start of each policy rollout ($\tau = 0$), we initialize both the feature grid and the decoder from a pre-trained offline map serving as an environment prior. The map is updated online every T_{update} steps using streaming robot observations. Here, the decoder parameters θ may be trained or frozen. We consider the frozen-decoder setting. To preserve the consistency of the static scene, we segment out dynamic elements (*e.g.*, robot arm) and exclude them from the updates, and we suspend updates while the robot is grasping. Each update performs K_{update} optimization steps. The policy is conditioned on the updated map $\mathcal{M}_\tau$ and generates an action $a_\tau \sim \pi_\phi(\cdot \mid \mathcal{M}_\tau, o_\tau, s_\tau, e_\ell)$. Notably, the map parameters ψ are not optimized via the policy objective in (3).

D. Implementation Details

We use a two-level feature grid ($L = 2$) for both room-scale and table-scale scenes, with resolutions of 0.4 m and 0.2 m (room) and 0.24 m and 0.12 m (table). We use EVA-02-Large [39] to obtain target VLM embeddings for supervision. For online mapping, the map is updated every $T_{\text{update}} = 5$ environment steps, with each update consisting of $K_{\text{update}} = 20$ optimization steps on the map features.

V. Map-Conditioned Policy

This section explains how the robot manipulation policy is conditioned on the latent feature map. We first introduce a *global map token* that captures scene-wide context from the latent map, enabling global and long-horizon reasoning (Sec. V-A). We then describe how the global map token is integrated into the policy, learned either with behavior cloning or reinforcement learning (Sec. V-B).

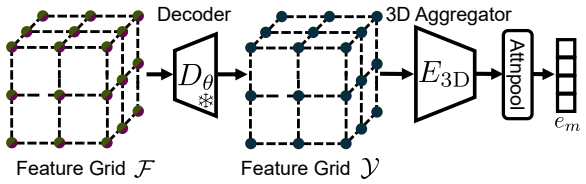


Fig. 4: **Global map token.** Latent features from $\mathcal{F}$ are decoded to $\mathcal{Y}$ via the decoder D_θ at the finest grid vertices. The 3D feature aggregator processes the coordinate-feature pairs, and its outputs are attention-pooled to produce the global map token e_m.

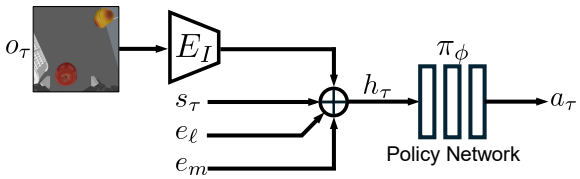


Fig. 5: **Map-conditioned policy network.** Proprioceptive state s_τ, image features $E_I(o_\tau)$, task embedding e_ℓ, and global map token e_m are concatenated to form a joint embedding h_τ, which is mapped to an action a_τ by the policy network π_ϕ.

A. Global Map Token

The latent map enables the robot to retrieve global and long-term context about the environment beyond its immediate field of view. We introduce a 3D feature aggregator that distills features distributed across the scene-wide map into a compact *global map token*, which then conditions the policy. An overview of this process is shown in Fig. 4.

First, we extract features from the map at the vertices of its finest grid level. Let $\mathcal{Z}_L = \{z_{L,i}\}_{i=1}^{M_L}$ be the coordinates of these vertices. We compute a decoded feature at each vertex using the multiresolution feature $F_\psi(z_{L,i})$ from (5) and the decoder D_θ: $\hat{y}(z_{L,i}) = D_\theta\big(F_\psi(z_{L,i})\big)$. We then form the set $\mathcal{S}_L$ of coordinate–feature pairs at selected vertices:

$$\mathcal{S}_L = \big\{\big(z_{L,i}, \hat{y}(z_{L,i})\big)\big|\, v_i = 1,\ i{=}1,\dots,M_L\big\}, \tag{8}$$

where $v_i \in \{0,1\}$ is a binary mask selecting task-relevant vertices (*e.g.*, excluding unoccupied or background regions). These are fed into a 3D feature aggregator, E_{3D}, and its outputs are attention-pooled to produce a global map token:

$$e_m = \text{Attn-Pool}\Big(E_{3D}(\mathcal{S}_L)\Big). \tag{9}$$

The token e_m is used to condition the policy π_ϕ. With an offline map, the global map token is time-invariant; with online mapping (Sec. IV-C), it depends on the time step τ as the task state changes. The parameters of E_{3D} are included in ϕ and jointly optimized via the objective in (3).

The E_{3D} architecture matches the environment's scale. For large room-scale environments, we use a Point Transformer [40], whose hierarchical attention captures long-range spatial relationships. For smaller tabletop scenes, we utilize a lightweight PointNet [41], whose simpler structure is sufficient for compact environments and its computational efficiency is advantageous for sample-intensive RL. We encode coordinates with sinusoidal positional encodings [27], concatenate them with features, and input them to PointNet.

B. Map-Conditioned Policy Network

We treat the map token e_m in (9) as an additional state input to the policy networks. We first outline a generic integration scheme and, then, specify its instantiation for BC and RL. The policy inputs are image features $E_I(o_\tau)$, proprioceptive state s_τ, task embedding e_ℓ, and the global map token e_m. Concatenating these inputs yields a joint embedding h_τ:

$$h_\tau = \text{Concat}\big(E_I(o_\tau), s_\tau, e_\ell, e_m\big), \tag{10}$$

which is provided as input to the policy network π_ϕ to produce an action a_τ. The action generation and concatenation details depend on the policy architecture and learning method. The overall architecture is illustrated in Fig. 5.

a) Map-Conditioned BC: For BC, we formulate the policy architecture using ACT [4]. Image features from a Transformer encoder are concatenated channel-wise with the proprioceptive state, task embedding, and global map token to form the joint embedding h_τ. Then, h_τ is fed into a Transformer decoder, which predicts an action sequence $a_{\tau:\tau+H-1}$. The task embedding e_ℓ is obtained by encoding a natural language command (*e.g.*, "pick up the bowl") using a VLM text encoder [39]. Unlike the original ACT, we use an exponentially decaying loss weight on future action steps, prioritizing accuracy on immediate actions. This change is empirically effective for our expert demonstrations, which are produced by an RL policy and less smooth than human demonstrations. At test time, only the first action of each predicted chunk is executed. Our BC policy uses DINOv2-ViT-S [17] as the visual backbone E_I. The Transformer has a 4-layer encoder and 6-layer decoder, predicting action sequences over a 16-step horizon ($H = 16$).

b) Map-Conditioned RL: For RL, we use PPO [42] with an actor-critic architecture where both heads are two-layer MLPs. The actor head outputs an action distribution $\pi_\phi(\cdot\,|\,h_\tau)$, while the critic head estimates the value $V(h_\tau)$, used to compute the advantage in the PPO objective. Both heads are conditioned on the joint embedding h_τ, formed via feature-wise concatenation of the proprioceptive state, the map token, a learnable task embedding e_ℓ specifying the target object, and image features flattened then projected by an MLP. We use a frozen DINOv2-ViT-S [17] whose output is processed by a two-layer MLP as the visual backbone E_I. To improve sample efficiency, we adopt a two-stage curriculum. First, we pre-train a map-agnostic, image-based policy by replacing the global map token with a zero vector. We then fine-tune the policy with the map token enabled via a learnable gating mechanism. A trainable sigmoid gate applies element-wise scaling to the map token, allowing the policy to gradually incorporate map features during fine-tuning.

VI. Evaluation

We evaluate SBP on its ability to (1) reason globally, particularly for targets beyond the field of view, and (2) handle multi-stage tasks requiring long-term context. We test SBP in two settings: a home-rearrangement mobile manipulation task

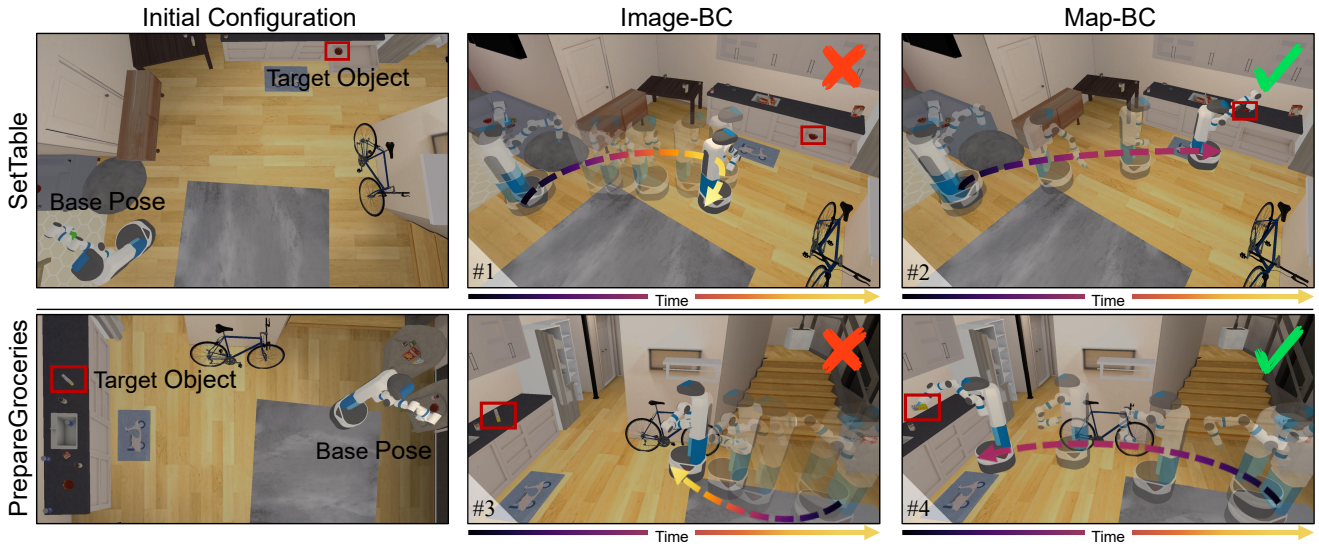


Fig. 6: Qualitative comparison on home-rearrangement tasks under out-of-distribution conditions. At the start of each episode, the robot is placed at a distant base pose unseen during training, with the target object completely outside of the robot's current field of view. Image-BC (#1 and #3) fails to localize the object in these settings, resulting in inefficient trajectories that fail to reach the target. In contrast, Map-BC (#2 and #4) successfully navigates to and grasps the object, completing the task with direct and efficient trajectories.

TABLE I: Performance of BC policies using different visual representations for an object-picking task from three home-rearrangement benchmarks. For each benchmark, we report success rate (SR↑) and episode reward (ER↑) on both the training scene (ID) and the novel scene (OOD), averaged over three runs. Best and second-best results are highlighted in **bold** and underlined, respectively.

	TidyHouse-Pick				PrepareGroceries-Pick				SetTable-Pick			
Method	SR (ID)	SR (OOD)	ER (ID)	ER (OOD)	SR (ID)	SR (OOD)	ER (ID)	ER (OOD)	SR (ID)	SR (OOD)	ER (ID)	ER (OOD)
Image-BC [4]	0.31	0.29	0.50	0.50	0.30	0.25	0.46	0.44	0.55	0.49	0.58	0.56
Uplifted [6]	0.30	0.30	0.48	0.50	0.31	0.28	**0.48**	0.44	**0.56**	0.51	**0.59**	0.57
Point Cloud [9]	0.15	0.13	0.42	0.41	0.16	0.16	0.38	0.37	0.41	0.32	0.49	0.45
Map-BC (offline)	**0.33**	**0.31**	**0.53**	**0.52**	**0.33**	**0.30**	0.47	**0.46**	0.55	**0.54**	0.59	**0.60**

(Sec. VI-A) and a sequential pick-and-place task (Sec. VI-B). For the latter, we also demonstrate zero-shot sim-to-real transfer of our learned policy to a physical robot. In both settings, SBP demonstrates superior performance over policies that rely solely on image-based reasoning.

Across all experiments, performance is evaluated using success rate (SR) and episode reward (ER). ER comprises object reach shaping, a grasp bonus, and a success bonus. We report both in-distribution (ID) and out-of-distribution (OOD) results. All experiments run in the ManiSkill simulator [43].

A. Mobile Manipulation Experiment

Setup. Our setup is based on Pick Subtasks from the ManiSkill-HAB benchmark [1], [44]. Unlike the original benchmarks, we train our policy on demonstrations from only a single scene (`sc1-13`), which are generated using the RL policy from [44]. We evaluate on the training scene (ID, `sc1-13`) and a novel, unseen scene with a different layout and object arrangement (OOD, `sc1-10`). To test generalization with a map, OOD evaluation uses a pre-generated latent map of the novel scene, without using any expert demonstrations from the scene.

We compare three baseline policies that use different visual representations and our map-conditioned BC policy.

- Image-BC: Image-based policy (ACT [4]).
- Uplifted: Image-based policy whose features are lifted into transient 3D tokens via 3D RoPE [5], [6].
- Point Cloud: Point cloud-based policy with point cloud observations processed by a 3D feature encoder [9], [41].
- Map-BC (offline): The proposed map-conditioned policy that uses a pre-generated offline map.

To ensure a fair comparison, all methods use the same policy architecture and hyperparameters.

Results. Table I shows that Map-BC outperforms the baselines on most benchmarks, with especially large gains on challenging tasks such as TidyHouse, which includes nine target objects. We attribute this to stronger scene understanding, enabling efficient target localization and navigation.

Fig. 6 shows two illustrative runs comparing Map-BC and Image-BC policies in mobile manipulation tasks that require global reasoning. In the original dataset, localization is simplified by initializing the robot near the target and orienting it towards the target. Instead, we apply substantial translational and rotational perturbations to the robot's initial pose, resulting in the target being far outside the robot's initial field of view. The Image-BC baseline produces erratic and inefficient trajectories and fails to reach the target, whereas Map-BC drives toward the target object and completes the task. These results indicate that the latent map enables the policy to perform global reasoning over the scene.

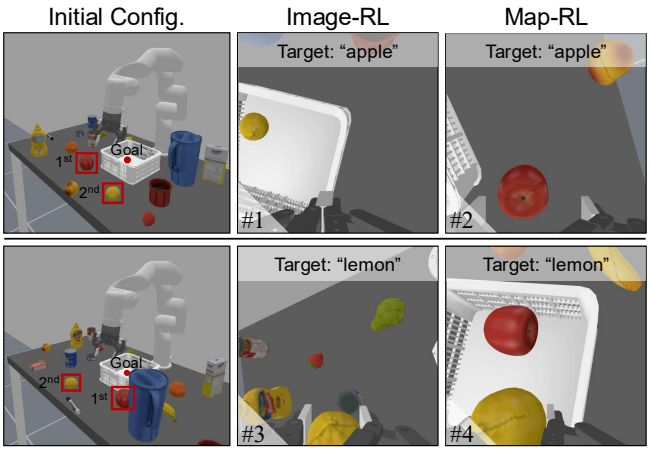


Fig. 7: Qualitative results from sequential manipulation using only egocentric views. Image-RL fails to localize the second object (#1) or the goal (#3) once they are out of view. In contrast, Map-RL leverages the map as spatial memory, locating the object and goal and completing the task sequence (#2 and #4).

TABLE II: Performance of RL policies on the sequential pick-and-place task. We evaluate on 100 training (ID) and 20 novel (OOD) scenes. We report success rate (SR↑) and episode reward (ER↑), averaged over three runs. Best results are highlighted in **bold**.

	Sequential Pick-and-Place			
Method	SR (ID)	SR (OOD)	ER (ID)	ER (OOD)
Image-RL [42]	0.82	0.75	0.77	0.7
Map-RL (offline)	0.94	0.95	0.85	0.84
Map-RL (online)	**0.97**	**1.00**	**0.87**	**0.88**

B. *Sequential Manipulation Experiment*

Setup. We design a two-stage sequential pick-and-place task to evaluate whether the latent map improves long-horizon manipulation. The task requires picking objects from a cluttered tabletop and placing them in a basket in a prescribed order (Fig. 7). For each episode, we sample an environment from a predefined set of scenes that differ in the arrangements and poses of clutter and target objects. The target set is fixed across episodes, and the execution order of targets is randomized per episode. To mimic limited visibility in mobile manipulation, the robot relies solely on an egocentric camera (*i.e.*, no global view of the scene). The scenes contain occlusions, making it challenging to localize the target objects. Experiments are conducted on a uFactory xArm6 robot. We evaluate on 100 training (ID) scenes and 20 novel (OOD) scenes with unseen object arrangements.

We compare an image-based baseline with two variants of the map-conditioned RL policy.

- Image-RL: Image-based policy.
- Map-RL (offline): The proposed map-conditioned policy that uses a pre-generated offline map.
- Map-RL (online): The proposed map-conditioned policy that uses an online-updated map, as described in Sec. IV-C.

All policies are trained with PPO [42] under the same policy architecture and hyperparameters.

Results. The results of the sequential manipulation experiment are summarized in Table II and illustrated in Fig. 7.

Fig. 8: Real-world setup mirroring the sequential manipulation task from simulation. Our policy is transferred from simulation to the real robot in a zero-shot manner and completes the task.

Map-RL (offline) and Map-RL (online) both outperform Image-RL in SR and ER, with a larger margin in OOD settings. In Fig. 7, Image-RL fails to localize the second object or the goal once they leave the egocentric view. In contrast, our Map-RL policy leverages the latent map as spatial memory, enabling it to locate the objects and complete the task sequence. Furthermore, Map-RL (online) shows an advantage over Map-RL (offline). The latter relies on an offline map capturing only the initial object arrangement, whereas the former's online updates provide temporal memory that allows the policy to track the task state.

Real robot deployment. We deploy the Map-RL (offline) policy, trained in simulation, on a uFactory xArm6 robot in a zero-shot manner. Our real-world setup, shown in Fig. 8, closely mirrors the simulation environment with aligned coordinate frames, identical objects (*e.g.*, table, basket, and targets), and matched egocentric camera pose. We first build an offline latent map of the real scene from a sequence of egocentric RGB-D images and camera poses estimated from the robot's forward kinematics. We then deploy the policy using this map as input. We do not use additional sim-to-real transfer techniques; instead, we rely on a frozen DINOv2 visual backbone that is robust to the visual domain gap. The policy successfully completes the sequential manipulation task on the real robot. The sim-to-real gap still makes grasping challenging in scenes with distant targets. Fig. 8 shows snapshots from the policy rollout.

VII. Conclusion

While 3D maps have long been core components of robot navigation, they have largely been overlooked in learning manipulation policies. In this paper, we presented a 3D latent map formulation that offers key advantages for manipulation: (i) perception beyond the robot's field of view and (ii) observation aggregation over long horizons. Building on these advantages, we proposed an end-to-end approach that couples a 3D latent map with mobile manipulation policy learning, providing the robot with extended spatial and temporal context. The experiments demonstrate that the map-conditioned policy is able to reason globally, leveraging the map as spatiotemporal memory for scene-level mobile manipulation and sequential manipulation tasks.

Several avenues for future work are possible. First, while our latent feature map provides global context, the policy still relies on local image features. Future work could reduce this dependency by developing dynamic scene representations to capture the motion of the robot and other objects. Second, the map-conditioned RL policy is trained using an on-policy method, which can be sample-inefficient, requiring pre-training from an image-based policy. Integrating our method with off-policy or model-based RL could mitigate these problems. Third, our map-conditioned policy uses Point Transformer [40] for behavior cloning but a lighter PointNet [41] for RL, as the former is too expensive for online training. Developing a unified and efficient 3D aggregation model would enable online policy training. Finally, evaluating our approach on larger-scale, longer-horizon mobile manipulation tasks is needed to further validate its effectiveness. Extending to more complex dexterous manipulation scenarios could also demonstrate its broader applicability.

REFERENCES

[1] A. Szot, A. Clegg, E. Undersander, E. Wijmans, Y. Zhao, J. Turner, N. Maestre, M. Mukadam, D. S. Chaplot, O. Maksymets, A. Gokaslan, V. Vondrus, S. Dharur, F. Meier, W. Galuba *et al.*, "Habitat 2.0: Training home assistants to rearrange their habitat," in *NeurIPS*, 2021.

[2] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell *et al.*, "π_0: A vision-language-action flow model for general robot control," in *RSS*, 2025.

[3] C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, and S. Song, "Diffusion Policy: Visuomotor policy learning via action diffusion," *IJRR*, 2023.

[4] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *RSS*, 2023.

[5] T. Gervet, Z. Xian, N. Gkanatsios, and K. Fragkiadaki, "Act3D: 3D feature field transformers for multi-task robotic manipulation," in *CoRL*, 2023.

[6] T.-W. Ke, N. Gkanatsios, and K. Fragkiadaki, "3D Diffuser Actor: Policy diffusion with 3D scene representations," in *CoRL*, 2024.

[7] Y. Ze, G. Yan, Y.-H. Wu, A. Macaluso, Y. Ge, J. Ye, N. Hansen, L. E. Li, and X. Wang, "GNFactor: Multi-task real robot learning with generalizable neural feature fields," in *CoRL*, 2023.

[8] Y. Wang, G. Yin, B. Huang, T. Kelestemur, J. Wang, and Y. Li, "GenDP: 3D semantic fields for category-level generalizable diffusion policy," in *CoRL*, 2024.

[9] Y. Ze, G. Zhang, K. Zhang, C. Hu, M. Wang, and H. Xu, "3D Diffusion Policy: Generalizable visuomotor policy learning via simple 3D representations," in *RSS*, 2024.

[10] J. Yang, Z.-a. Cao, C. Deng, R. Antonova, S. Song, and J. Bohg, "EquiBot: SIM(3)-equivariant diffusion policy for generalizable and data efficient learning," *CoRL*, 2024.

[11] A. Rashid, S. Sharma, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. Goldberg, "Language embedded radiance fields for zero-shot task-oriented grasping," in *CoRL*, 2023.

[12] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola, "Distilled feature fields enable few-shot language-guided manipulation," *CoRL*, 2023.

[13] Y. Wang, M. Zhang, Z. Li, K. R. Driggs-Campbell, J. Wu, L. Fei-Fei, and Y. Li, "D^3 Fields: Dynamic 3d descriptor fields for zero-shot generalizable robotic manipulation," in *CoRL*, 2024.

[14] H. Durrant-Whyte and T. Bailey, "Simultaneous localization and mapping: Part I," *RAM*, 2006.

[15] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard, "Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age," *T-RO*, 2016.

[16] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, and I. Sutskever, "Learning transferable visual models from natural language supervision," in *ICML*, 2021.

[17] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang *et al.*, "DINOv2: Learning robust visual features without supervision," *TMLR*, 2023.

[18] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, "LERF: Language embedded radiance fields," in *ICCV*, 2023.

[19] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *SIGGRAPH*, 2023.

[20] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick, "Segment anything," in *ICCV*, 2023.

[21] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister, "LangSplat: 3D language Gaussian splatting," in *CVPR*, 2024.

[22] R.-Z. Qiu, G. Yang, W. Zeng, and X. Wang, "Feature Splatting: Language-driven physics-based scene synthesis and editing," in *ECCV*, 2024.

[23] S. Katragadda, C.-Y. Wu, Y. Guo, X. Huang, G. Huang, and L. Ren, "Online language splatting," *ICCV*, 2025.

[24] J. Yang, B. Ivanovic, O. Litany, X. Weng, S. W. Kim, B. Li, T. Che, D. Xu, S. Fidler, M. Pavone, and Y. Wang, "EmerNeRF: Emergent spatial-temporal scene decomposition via self-supervision," *ICLR*, 2024.

[25] W. Li, R. Zhou, J. Zhou, Y. Song, J. Herter, M. Qin, G. Huang, and H. Pfister, "4D LangSplat: 4D language Gaussian splatting via multimodal large language models," in *CVPR*, 2025.

[26] R.-Z. Qiu, Y. Hu, G. Yang, Y. Song, Y. Fu, J. Ye, J. Mu, R. Yang, N. Atanasov, S. Scherer, and X. Wang, "Learning generalizable feature fields for mobile manipulation," *IROS*, 2025.

[27] B. Mildenhall, P. P. Srinivasan, M. Tancik, J. T. Barron, R. Ramamoorthi, and R. Ng, "NeRF: Representing scenes as neural radiance fields for view synthesis," in *ECCV*, 2020.

[28] P. Liu, Y. Orru, J. Vakil, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "OK-Robot: What really matters in integrating open-knowledge models for robotics," *RSS*, 2024.

[29] P. Liu, Z. Guo, M. Warke, S. Chintala, C. Paxton, N. M. M. Shafiullah, and L. Pinto, "DynaMem: Online dynamic spatio-semantic memory for open world mobile manipulation," *ICRA*, 2025.

[30] Y. Tang, W. Huang, Y. Wang, C. Li, R. Yuan, R. Zhang, J. Wu, and L. Fei-Fei, "UAD: Unsupervised affordance distillation for generalization in robotic manipulation," in *ICRA*, 2025.

[31] M. Shridhar, L. Manuelli, and D. Fox, "Perceiver-Actor: A multi-task transformer for robotic manipulation," in *CoRL*, 2023.

[32] A. Goyal, V. Blukis, J. Xu, Y. Guo, Y.-W. Chao, and D. Fox, "RVT-2: Learning precise manipulation from few demonstrations," in *RSS*, 2024.

[33] N. Funk, J. Urain, J. Carvalho, V. Prasad, G. Chalvatzaki, and J. Peters, "ActionFlow: Equivariant, accurate, and efficient policies with spatially symmetric flow matching," *arXiv preprint arXiv:2409.04576*, 2024.

[34] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, and N. Houlsby, "An image is worth 16x16 words: Transformers for image recognition at scale," in *ICLR*, 2021.

[35] D. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *NeurIPS*, 1989.

[36] N. Hansen, H. Su, and X. Wang, "TD-MPC2: Scalable, robust world models for continuous control," *ICLR*, 2024.

[37] T. Müller, A. Evans, C. Schied, and A. Keller, "Instant neural graphics primitives with a multiresolution hash encoding," *TOG*, 2022.

[38] T. Takikawa, J. Litalien, K. Yin, K. Kreis, C. Loop, D. Nowrouzezahrai, A. Jacobson, M. McGuire, and S. Fidler, "Neural geometric level of detail: Real-time rendering with implicit 3D shapes," in *CVPR*, 2021.

[39] Y. Fang, Q. Sun, X. Wang, T. Huang, X. Wang, and Y. Cao, "EVA-02: A visual representation for neon genesis," *Image Vis. Comput.*, 2024.

[40] H. Zhao, L. Jiang, J. Jia, P. H. Torr, and V. Koltun, "Point Transformer," in *ICCV*, 2021.

[41] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "PointNet: Deep learning on point sets for 3D classification and segmentation," in *CVPR*, 2017.

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

[43] S. Tao, F. Xiang, A. Shukla, Y. Qin, X. Hinrichsen, X. Yuan, C. Bao, X. Lin, Y. Liu, T.-K. Chan, Y. Gao, X. Li, T. Mu, N. Xiao, A. Gurha *et al.*, "ManiSkill3: GPU parallelized robotics simulation and rendering for generalizable embodied AI," *RSS*, 2025.

[44] A. Shukla, S. Tao, and H. Su, "ManiSkill-HAB: A benchmark for low-level manipulation in home rearrangement tasks," in *ICLR*, 2025.