D^{3}Fields: Dynamic 3D Descriptor Fields for Zero-Shot Generalizable Rearrangement

Yixuan Wang[1*], **Mingtong Zhang**[2*], **Zhuoran Li**[3*], **Tarik Kelestemur**[4], **Katherine Driggs-Campbell**[2], **Jiajun Wu**[5], **Li Fei-Fei**[5], **Yunzhu Li**[1]

[1]Columbia University, [2]University of Illinois, Urbana-Champaign, [3]National University of Singapore, [4]Boston Dynamics AI Institute, [5]Stanford University

Contents

* Denotes equal contribution.

8th Conference on Robot Learning (CoRL 2024), Munich, Germany.

1 Detailed Comparisons with Prior Works

1.1 Comparisons with Prior 3D Representations using Visual Foundational Models

We extend the application of state-of-the-art visual foundational models to robotic manipulation tasks, offering the following improvements over prior methods:

- **Efficiency**: Several existing works construct 3D representations from 2D models, but they are often time-consuming due to the distillation process. This limits their applicability in tasks requiring frequent visual feedback. For instance, prior works such as F3RM [1] and LERF [2] require 1.3 to 8 minutes for training. In contrast, our method takes only 0.166 seconds, making it 500 to 3,000 times faster.
- **Sparse Views**: Some existing works, such as F3RM and LERF, require dense camera views (more than 50) to build a high-quality 3D representation, which is not suitable for typical robotic manipulation workspaces with sparse camera views (1-4). In contrast, we have developed a more efficient and effective pipeline that uses only four views.
- **Planning Using Reconstructed Fields**: We explicitly use the reconstructed fields to find matches that define our planning objective, allowing us to employ model-based planning to solve rearrangement tasks. This contrasts with existing methods like F3RM, which require expert demonstrations for every new task.

Additionally, our work distinguishes itself from FeatureNeRF by applying our representation specifically to robotic manipulation tasks, whereas FeatureNeRF does not focus on manipulation [3].

1.2 Comparisons with Dense Object Nets

Our approach differs from Dense Object Nets (DON) in two major ways [4]:

- **Training Requirements**: DON is trained via contrastive learning, which requires foreground and background separation and a careful selection of positive and negative pairs. This means that deploying a DON in a new environment or on new object categories requires additional training efforts. In contrast, our pipeline does not need extra training and can be applied to a diverse set of object categories by leveraging visual foundational models.
- **Generalization**: DON is trained on small-scale datasets and may not generalize well to new object categories and scenarios. Our pipeline, however, offers stronger generalization capabilities without the need for retraining and additional data collection.

2 Method Details

2.1 Notation: Camera Transformation and Projection

We assume that there are multiple RGBD cameras with fixed viewpoints. We assume all cameras' intrinsic parameters $\mathbf{K}$ and extrinsic parameters $\mathbf{T}$ are known. The ith camera's extrinsic parameters are defined as follows:

$$\mathbf{T}_i = \begin{bmatrix} \mathbf{R}_i & \mathbf{t}_i \\ 0^T & 1 \end{bmatrix} \in \mathbb{SE}(3), \tag{1}$$

where Euclidean group $\mathbb{SE}(3) := \{\mathbf{R}, \mathbf{t} \mid \mathbf{R} \in \mathbb{SO}^3, \mathbf{t} \in \mathbb{R}^3\}$. For a 3D point $\mathbf{x}$ in the world frame, we could obtain the pixel $\mathbf{u}_i$ projected in ith camera and distance to ith camera r_i as follows:

$$\mathbf{u}_i = \text{Proj}\left(\mathbf{K}_i\left(\mathbf{R}_i\mathbf{x} + \mathbf{t}_i\right)\right), \quad r_i = [0,0,1]^T\left(\mathbf{R}_i\mathbf{x} + \mathbf{t}_i\right), \tag{2}$$

where Proj performs perspective projection, mapping a 3D vector $p = [x, y, z]^T$ to a 2D vector $q = [x/z, y/z]^T$.

Algorithm 1 Fusion Process

procedure FUSION($\mathbf{x}$) ▷ Input 3D point
  $\mathbf{u}_i, r_i \leftarrow \text{Project}(\mathbf{x}, i), r'_i \leftarrow \mathcal{R}_i[\mathbf{u}_i]$ ▷ 3D projection and depth reading
  $d_i \leftarrow r'_i - r_i, d'_i \leftarrow \text{Truncate}(d_i, \mu)$ ▷ Truncated depth difference using Eq. 3 in main paper
  $v_i, w_i \leftarrow \text{Weights}(d_i)$ ▷ Assign weights to each view using Eq. 4 in main paper
  $\mathbf{f}_i, \mathbf{p}_i \leftarrow \text{Interpolate}(\mathcal{W}_i^{\mathbf{f}}, \mathcal{W}_i^{\mathbf{P}}, \mathbf{u}_i)$ ▷ Interpolate features using Eq. 5 in main paper
  $\mathbf{f}, \mathbf{p} \leftarrow \text{Fuse}(\mathbf{f}_i, \mathbf{p}_i, v_i, w_i)$ ▷ Fuse features using Eq. 6 in main paper

2.2 Fusion Equation Explanation

Here is a detailed explanation for Equation 6.

- Signed Distance d: Assuming $\sum v_i \neq 0$ and ignoring δ, the equation simplifies to $d = \frac{\sum v_i d'_i}{\sum v_i}$. In this case, the weights sum to 1.
- Semantic Feature $\mathbf{f}$ and Instance Masks $\mathbf{p}$: Similarly, with $\sum v_i \neq 0$ and ignoring δ, we have $\mathbf{f} = \frac{\sum v_i w_i f'_i}{\sum v_i}$ and $\mathbf{p} = \frac{\sum v_i w_i p'_i}{\sum v_i}$. The weight w_i varies based on proximity to the surface: $w_i = 1$ when $\mathbf{x}$ is close, summing weights to 1, and w_i approaches 0 when $\mathbf{x}$ is far, causing $\mathbf{f}$ and $\mathbf{p}$ to converge to 0.

2.3 Correspondence Equation Explanation

Equation 8 describes the process of finding the correspondence from jth sampled point's associated feature $\mathbf{f}_j^0 \in \mathbb{R}^f$ to a 2D point $\mathbf{s}_{\text{goal},j}$ in the image space. The computation process consists of three steps:

- α_{ij} is the feature-space distance between jth sampled point and ith goal image pixel. We first extract goal image's feature map $\mathcal{W}_{\text{goal}}$ and read out ith pixel's corresponding feature $\mathcal{W}_{\text{goal}}[\mathbf{u}_i]$. Then we compute the L2 distance between $\mathcal{W}_{\text{goal}}[\mathbf{u}_i]$ and $\mathbf{f}_j^0$ and assign to α_{ij}.
- After applying the softmax to α_{ij} over the whole image, we obtain β_{ij}. The summation of β_{ij} over the image space, i.e. $\sum_{i=1}^{H \times W} \beta_{ij} = 1$.
- Then we apply the weighted sum of pixels to obtain the corresponding pixel on the goal image $\mathbf{s}_{\text{goal},j}$.

2.4 Grounded-SAM Masks Association

Using Grounded-SAM [5, 6], we could extract instance segmentation masks from each view. However, masks from different views can contain a different number of instances, and the instance IDs may not be consistent. To tackle this problem, we need to post-process the instance segmentation results. The high-level idea is to merge instances from different viewpoints based on their geometric distance.

We will save all merged instances to a list. Specifically, we will first start from the first viewpoint. For each instance mask, we map them to 3D point clouds and save them into a list. Then, we move to the next viewpoint and map all instance masks to 3D point clouds. We will compare each instance with the merged instances in the list. If they have significant overlap, measured by the Intersection of Union (IoU) of the two point clouds, we will merge the instance from the new viewpoint with the merged instances in the list. This process will continue until all viewpoints have been iterated through.

After merging all instances, we will filter out instances that are not stably detected. Specifically, instances that meet one of the following criteria will be filtered out:

- The instance has little point cloud.
- The instance is known to be a background, such as the table.

- The instance overlaps with other instances, while other instances have a higher confidence.

After filtering, we will assign consistent instance IDs to the instance masks in each viewpoint.

2.5 Mask Tracking using XMem

After using Grounded-SAM to obtain initial masks, we use XMem to track instance masks in later frames so that we can skip time-consuming segmentation detection for subsequent frames. By applying XMem, we significantly reduce the computational cost and processing time, making the pipeline more efficient.

2.6 Dynamics Training Details

We instantiate the dynamics model $f(\cdot, \cdot)$ as graph neural networks (GNNs) that predict the evolution of particles $\boldsymbol{s}^t \in \mathbb{R}^{2\times n_{\boldsymbol{s}}}$ under external actions a_t. We also construct edges $\boldsymbol{e}^t \in \mathbb{N}^{2\times n_{\boldsymbol{e}}}$ according to particle distance, where $\boldsymbol{e}_j^t = (u_j^t, v_j^t)$ represents an edge connecting from particle u_j^t to particle v_j^t. $f(\cdot, \cdot)$ consists of node and edge encoders $f_{\mathcal{O}}^{\text{enc}}(\cdot, \cdot)$, $f_{\mathcal{E}}^{\text{enc}}(\cdot, \cdot)$ to obtain node and edge representations:

$$\begin{aligned} p_i^t &= f_{\mathcal{O}}^{\text{enc}}(\boldsymbol{s}_i^t, a_t), \quad i = 1, \ldots, n_{\boldsymbol{s}}, \\ q_j^t &= f_{\mathcal{E}}^{\text{enc}}(\boldsymbol{s}_{u_j^t}^t, \boldsymbol{s}_{v_j^t}^t), \quad j = 1, \ldots, n_{\boldsymbol{e}}. \end{aligned} \tag{3}$$

Then, we use node and edge decoders $f_{\mathcal{O}}^{\text{dec}}(\cdot, \cdot)$, $f_{\mathcal{E}}^{\text{dec}}(\cdot, \cdot)$ to predict the next time step's particle states:

$$\begin{aligned} r_j^t &= f_{\mathcal{E}}^{\text{dec}}(q_j^t), \quad j = 1, \ldots, n_{\boldsymbol{e}}, \\ \hat{\boldsymbol{s}}_i^{t+1} &= f_{\mathcal{O}}^{\text{dec}}(p_i^t, \sum_{j \in \mathcal{N}_i} r_j^t), \quad i = 1, \ldots, n_{\boldsymbol{s}}, \end{aligned} \tag{4}$$

where $\mathcal{N}_i$ is the index set of the edges that connect to particle i. In practice, we follow Li et al. [7] and use multi-step message passing over the graph to approximate the propagation of action impacts.

2.7 Keypoints Tracking Initialization

To initialize keypoint tracking, we first densely sample points, which can either come from grid sampling or instances' 3D point clouds. Then, we evaluate these points using our D^3Fields and mask out those not belonging to the desired instance. Finally, we downsample these points to the desired number using farthest point sampling.

2.8 Model-Predictive Control (MPC) Details

As described in Section 3.4 of the main paper, our MPC framework needs a reference camera to bridge the gap between 3D representation and 2D representation. In our work, the reference camera's extrinsic parameters are manually defined according to the tasks. Typically, it looks down at the workspace from above. Its intrinsic parameters are the same as the real camera's intrinsic parameters. The detailed MPC algorithm we used is described in Algorithm 2.

For the pick-and-place tasks, our dynamics model is simplified, as the object is rigidly attached to the end-effector.

2.9 Discussion of Backbone Choice

We choose DINOv2 as our feature backbone because there are several great and unique properties of DINOv2, including:

- Generalization: DINOv2 has demonstrated consistent feature extraction across diverse object categories and scenes, which allows us to apply our pipeline to various object categories and bridge the gap between goal image and workspace.

Algorithm 2 Trajectory optimization at each MPC step

Input: Current state $\mathbf{s}_0$, goal $\mathbf{s}_{\text{goal}}$, time horizon T, gradient descent iteration N
the perception module h, and the dynamics module f
Output: Actions $a_{0:T-1}$

Sample current action sequence $\hat{a}^*_{0:T-1}$
for $i = 1, \ldots, N$ **do**
  Sample M action sequences $\hat{a}^{1:M}_{0:T-1}$ near current action sequence
  for $m = 1, \ldots, M$ **do**
    for $t = 0, \ldots, T-1$ **do**
      Predict the next step $\mathbf{s}_{t+1} \leftarrow f(\mathbf{s}_t, \hat{a}^m_t)$
    Calculate the task loss $c^m \leftarrow c(\mathbf{s}_T, y_g)$
  Calculate the current action sequence $\hat{a}^*_{0:T-1}$ using the task loss $c^{1:M}$
Return $\hat{a}^*_{0:T-1}$

- Refined Correspondence: DINOv2 demonstrates the capability to establish a refined correspondence, even when two images have quite different backgrounds and contexts. This is important for our tasks since we need refined correspondence between the goal image and the workspace to define the objective function for the manipulation.

However, our framework is not specific to DINOv2. If there is a better visual foundational model in the future, we could replace it easily in a plug-and-play manner.

3 Additional Experiments

3.1 Implementation Details

For the truncation threshold μ, we set it to 0.02 across all experiments. For the prompts used for Grounded-SAM [5, 6] in Figure 7 in the main paper, from left to right, they are "shoe", "mug", "spoon", "can", and "toothpaste" respectively. The confidence threshold used for Grounded-SAM is 0.2.

We also compare with several baselines, with details listed below:

- Dense Object Nets (DON) [4]: We compare the effects of using different feature backbones, with DON as one of the baseline backbones. We use the pre-trained DON model since our method also uses off-the-shelf models with no re-training or additional data. Specifically, we use the model trained on the shoe class, which can be found here.
- DINO [8]: Another feature backbone we baseline on is DINO, which is the precursor to DINOv2. We use the code provided by [9] to extract dense DINO features.
- RGBD+DINOv2: We also compare our method to simply merging point clouds with DINOv2 features from multiple viewpoints [10].
- FeatureNeRF [3]: We compare our representation with other state-of-the-art 3D implicit semantic representations, including FeatureNeRF. We trained the model on the car example dataset provided by the authors. For comparison with our model, we do not distill the model from the DINO model but from DINOv2. It is worth noting that FeatureNeRF only uses one RGB image to generate the neural fields. We found that providing more views to the FeatureNeRF model during inference time leads to worse performance. Therefore, we keep its input as a single-view RGB image.
- Distilled Feature Fields (F3RM) [1]: Another 3D implicit semantic representation we compare to is F3RM. We use the same training code provided by the authors, except that we distill from DINOv2 models to make it comparable with our model. In addition, we use four camera views as F3RM inputs instead of dense views as the original paper.

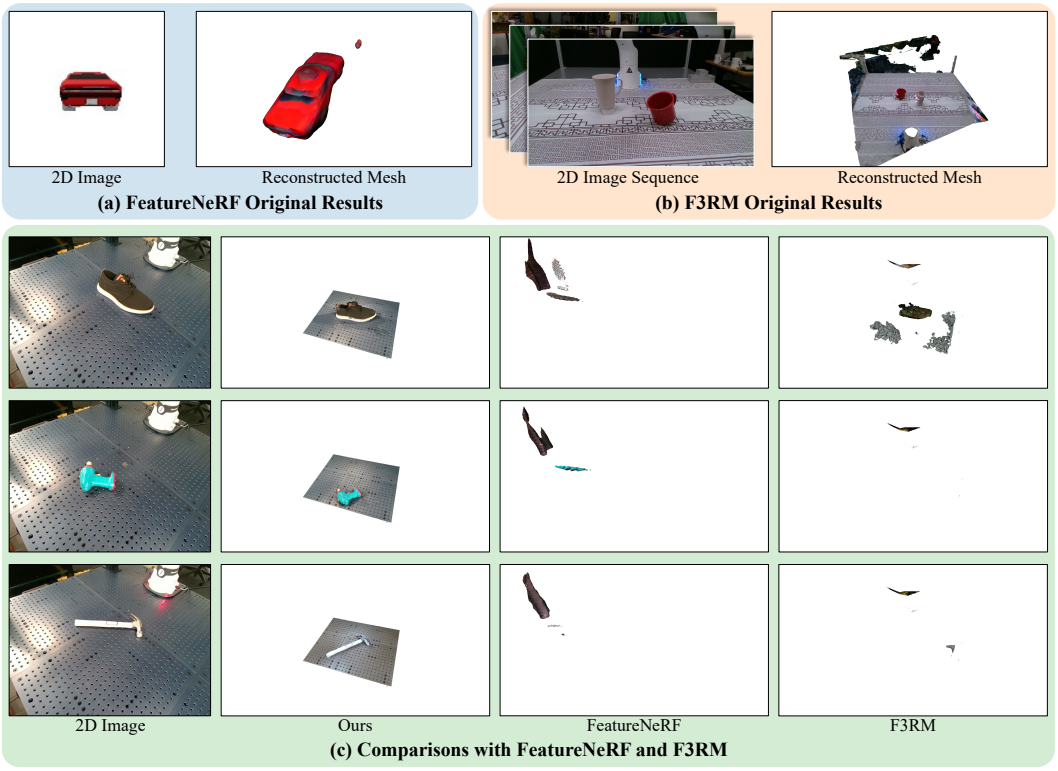

Figure 1: **Mesh Reconstruction Comparison.** (a) shows the reconstructed mesh of the FeatureNeRF, given a 2D image from the training distribution. This reflects that our mesh extraction process works well when the input image is within the training distribution. Given a sequence of 2D images densely scanning the workspace, (b) also shows good reconstruction quality of the scene. However, when given sparse views containing novel instances, both FeatureNeRF and F3RM fail to generate accurate meshes for the scene, which demonstrates the effectiveness of our method.

For the dynamics training of the shoe-pushing example, we collected 20 episodes of pushing one shoe. Then, we trained a dynamics model that can take in current particles and a pushing action and predict particles in the next step.

For the evaluation in the real world, we summarize the details of our tasks in Table 1.

Environment	Task Name	Objects
Real World	Organize Shoes	Shoe
	Collect Debris	Almonds
	Organize Office Table	Mouse, Pen, Mug
	Organize Utensils	Knife, Spoon, Fork, Bowl
	Organize Fruits	Apple, Banana
	Push Shoes	Shoe
Simulation	Serve Food	Cupcake, Bread, Tomato, Lemon, Banana
	Organize Mugs	Mug
	Organize Shoes	Shoe
	Organize Utensils	Knife, Spoon, Fork

Table 1: **Task Details Summary.** This table summarizes our task environment, specific tasks, and objects. We evaluate our framework on eight tasks and fifteen object categories, where each object category covers several object instances with diverse appearances and shapes.

3.2 Keypoint Tracking Results

We show two examples of 3D keypoint tracking in Figure 2. In the first scenario, we track a shoe as it is pushed and subsequently flipped. The second example demonstrates tracking a shoe that is lifted and then placed down. Our system robustly tracks the shoe in 3D space. These examples underscore the effectiveness of D^{3}Fields in maintaining accurate tracking in dynamic scenarios, which enables our dynamics learning capabilities.

3.3 Mesh Comparisons with FeatureNeRF and F3RM

We qualitatively compare the descriptor fields generated by the three methods. We extract the mesh from these fields using marching cubes, as shown in Figure 1. We observe that our D^{3}Fields could generate accurate color meshes given sparse views. FeatureNeRF could reconstruct a reasonable mesh given a single 2D image from the training distribution, as shown in Figure 1 (a). However, when it encounters a new object outside the training distribution, the reconstructed mesh will be completely off, even when we apply the image preprocessing to align the testing images with the training set in terms of image sizes, data range, and background color. Although Figure 1 (b) shows that we can reconstruct a clear mesh with dense image sequences as in the original paper, its color mesh is quite inaccurate given sparse viewpoints in our experiment setting, except for the shoe case.

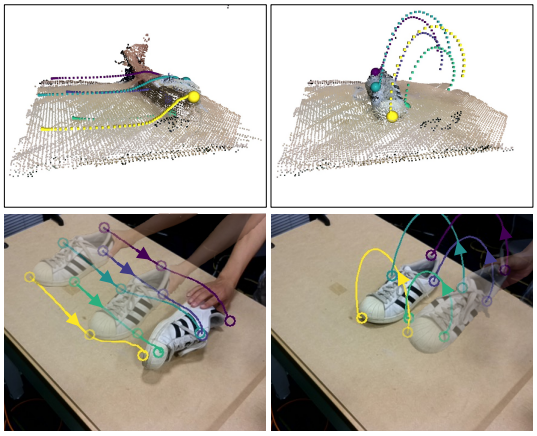

Figure 2: **Keypoint Tracking.** We apply D^{3}Fields to tracking tasks and showcase two tracking examples, both of which involve 3D motions and partial observations from single viewpoints. This shows that our representation is 3D, dynamic, and semantic.

3.4 Quantitative Correspondence Comparisons with FeatureNeRF and F3RM

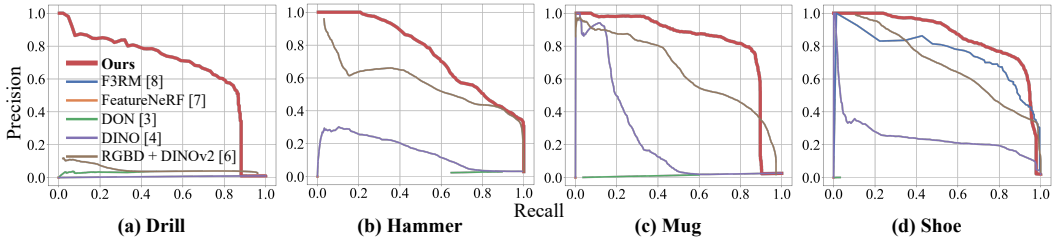

Figure 3: **Precision-Recall of Various Thresholds for Different Instances.** The curves show how D^{3}Fields compares with 3 baseline methods in terms of matching quality, tested on 4 different instances: mug, bag, pan, and shoe. We use the precision-recall curve to measure the correspondence quality. Our method shows to consistently exceeds the performance of the baseline approaches, which demonstrates our method's capability to encode semantic information accurately and establish precise correspondences using the semantic information.

In addition, we also measure the quantitative correspondence accuracy of our method, FeatureNeRF, and F3RM. We manually label the ground truth correspondence keypoints on the source image and the target descriptor fields. We measure the correspondence quality using the precision-recall curve, as shown in Figure 3 A larger area under the curve indicates better correspondence quality. Details regarding the precision-recall curve are provided later. We could see that F3RM and FeatureNeRF collapse to the origin point except for the shoe example for F3RM. This is because these two methods fail to reconstruct meshes given sparse observations and unseen instances. In contrast, our method shows a much better correspondence quality.

To generate the precision-recall curve, we manually label one point $\mathbf{x}_{\text{src}}$ on the 2D source image, and a set of corresponding 3D points $\mathbf{x}_{\text{tgt}}$. For 2D points, we obtain the associated semantic feature $\mathbf{f}_{\text{src}}$. For vertices on the reconstructed mesh, we can obtain a set of semantic features $\{\mathbf{f}_{\text{tgt},0}, ..., \mathbf{f}_{\text{tgt},N}\}$. Then we compute the cosine similarity between features $\mathbf{f}_{\text{src}}$ and $\{\mathbf{f}_{\text{tgt},0}, ..., \mathbf{f}_{\text{tgt},N}\}$. For one similarity threshold τ, we filter out a set of points $\mathbf{F}_\tau$ with similarity scores higher than τ. Additionally, we identify the set of points $\mathbf{G}$ that are close to $\mathbf{x}_{\text{tgt}}$. We then define precision P_τ and R_τ as follows:

$$P_\tau = \frac{|\mathbf{F}_\tau \cap \mathbf{G}|}{|\mathbf{F}_\tau|}, \quad R_\tau = \frac{|\mathbf{F}_\tau \cap \mathbf{G}|}{|\mathbf{G}|}. \tag{5}$$

By varying τ, we can plot the precision-recall curve as shown in Figure 3.

3.5 Comparisons with F3RM under Dense Views

We compare our approach using 4 views against F3RM [1], which uses 50 views (with and without Grounding DINO + SAM supervision), on the Dense Object Nets dataset [4].

Our qualitative results are shown in Figure 4. Similar to the main paper, We compare the reconstructed mesh, descriptor fields, and mask fields for our approach (4 views), F3RM without Grounding DINO + SAM (50 views), and F3RM with Grounding DINO + SAM (50 views). We observed that F3RM with 50 views reconstructs high-quality meshes, confirming our F3RM implementation is bug-free. Additionally, there is no significant qualitative difference between our method with 4 views and F3RM with 50 views, demonstrating the effectiveness of our representation. Lastly, we did not observe notable differences between F3RM with and without Grounding DINO + SAM, which is consistent with our previous conclusion that this additional supervision does not significantly contribute to F3RM training.

Similar to our paper, we also evaluated quantitative correspondence quality using the Precision-Recall (P-R) curve, as shown in Figure 5. A larger area under the P-R curve indicates better correspondence quality. Our results show no significant correspondence differences between our approach with 4 views and F3RM with 50 views, indicating that our representation is effective even with sparse views.

3.6 Ablation Study: Qualitative Correspondence Comparisons

In this section, we first study how different feature backbones could affect the correspondence quality. Then we show the qualitative correspondence results of our method. As mentioned in Section 3.1, we substitute our method's backbone with other pre-trained models, like DON and DINO [4, 8]. We also compare with RGBD+DINOv2 to demonstrate its effectiveness.

Figure 6 shows the qualitative correspondence results of all ablation baselines. There are three key observations we can make from this figure.

- Compared with RGBD+DINOv2, our method's correspondence quality is better. We achieve more accurate correspondence since our representation can amortize noise from single views by considering 3D consistency. Although RGBD+DINOv2 can achieve some spatial consistency, there are still variances in results from different viewpoints, while our method guarantees spatial consistency.
- Compared with DINO, our correspondence is more fine-grained and accurate. Thanks to the advancements in foundational visual models, DINOv2 encodes more fine-grained features and enables correspondence with higher accuracy.
- DON struggles to generalize to novel scenes and unseen object categories. Although the original DON shows good correspondence quality, it is trained on one type of object with a relatively small dataset. Compared with visual foundational models, it shows limited generalization capabilities in terms of scenes and object categories.

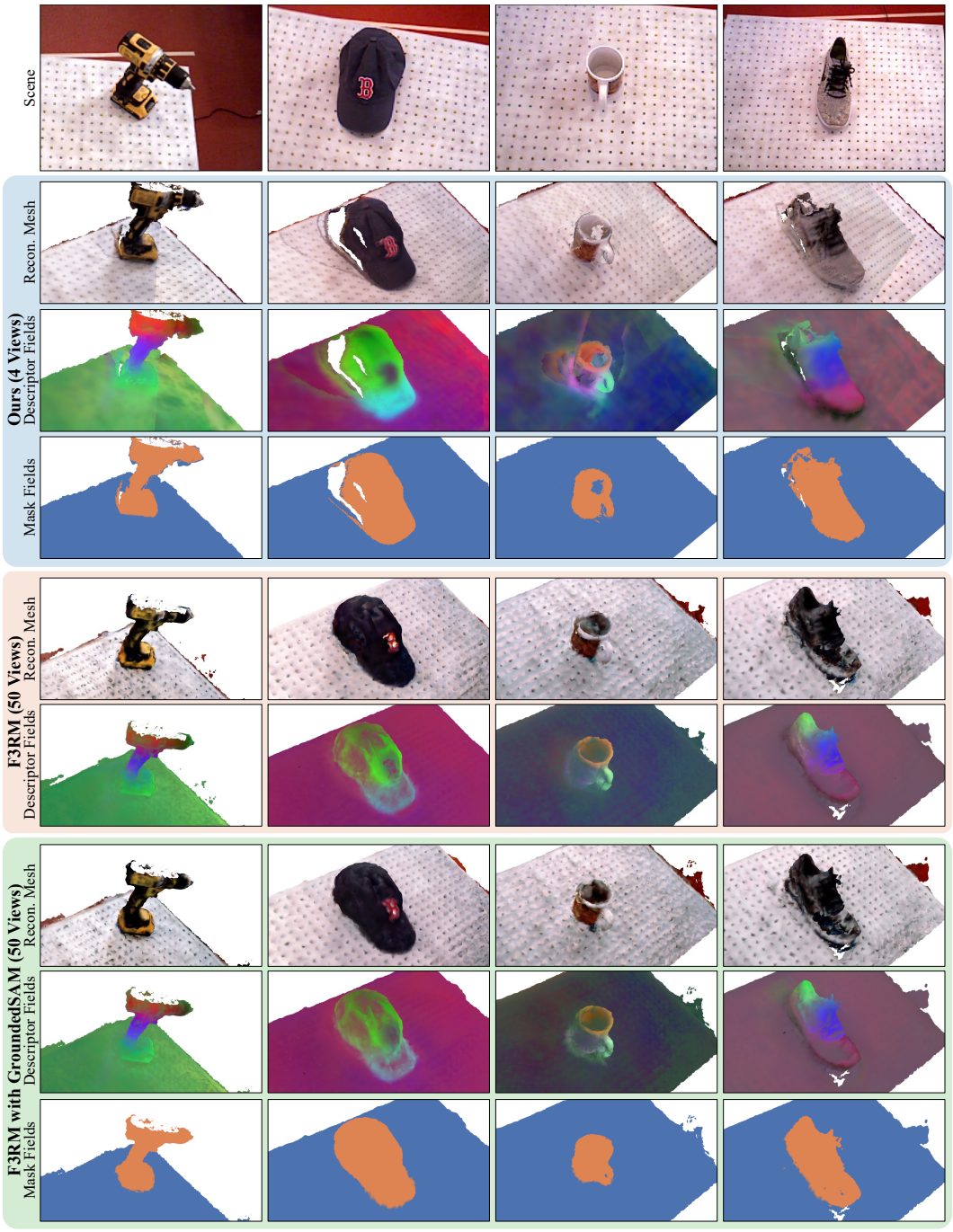

Figure 4: **Comparisons with F3RM under Dense Views.** We compared the reconstructed mesh, feature fields, and mask fields with F3RM using dense views. Our results showed that, despite using sparse inputs, our method did not exhibit significant qualitative performance degradation compared to F3RM under dense views, demonstrating the efficiency of our approach.

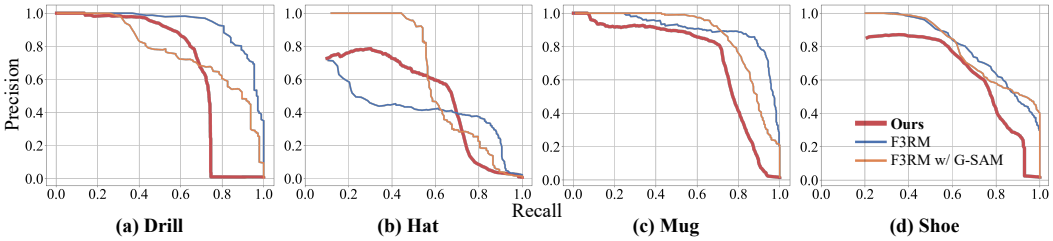

Figure 5: **Quantitative Comparisons with F3RM under Dense Views.** We use the P-R curve to measure the correspondence accuracy. Our results show no significant differences in correspondence between our approach with 4 views and F3RM with 50 views, indicating that our representation remains effective even with sparse views.

We also visualize the correspondence from 2D images to our workspace as shown in Figure 7. Specifically, we extract the DINOv2 feature of the selected pixel in the 2D image. Then we highlight the part of the 3D mesh with features close to the query feature. There are two observations regarding the qualitative correspondence results. First, the semantically similar parts are correctly matched across different instances and contexts. For example, when we select the rim of the plate in the 2D image, the corresponding part in the 3D mesh is highlighted. This matching is consistent across different object parts, such as the head and tail of the shoe, the handle and blade of the knife, and the tip and bar of the drill. Second, the correspondence is multimodal when there are multiple semantically similar object parts in the workspace. For example, when we select the spoon handle in the 2D image, multiple utensil handles in the workspace are highlighted. The correspondence qualitative results show that our D^3Fields could establish meaningful correspondences across different instances and contexts, so that we can rely on correspondence to define the planning objective function.

3.7 Ablation Study: Quantitative Correspondence Comparisons

Similar to the main paper, we generate the precision-recall curve to quantitatively compare the correspondence quality with ablation baselines. We can make the following observations regarding the baseline correspondence results.

- Compared with RGBD+DINOv2, our method shows more accurate correspondence results. This is because our D^3Fields can average out noise from each viewpoint, while RGBD+DINOv2 accumulates noises.
- DINO faces challenges in accurately distinguishing specific object components. This limitation results in less precise correspondence, as shown in Figure 3.
- Although DON can encode semantic features in seen environments and instances, it fails to generalize to novel environments and object categories. Therefore, its correspondence results are even worse than DINO, as shown in Figure 3.

3.8 Abaltion Study: Quantitative Manipulation Results

In Figure 8 (a), we measure performance using the IoU between the mask of the goal image and the mask of the final state post-manipulation. Higher IoU values indicate a greater degree of alignment between the intended and achieved configurations. Our method demonstrates superior performance across five distinct object categories, consistently outshining the baseline methods. For each category, we performed 5 experiments for the evaluation results. This not only highlights its exceptional manipulation accuracy but also its robust generalization capabilities. While the DINO model exhibits some struggles, particularly in distinguishing specific object components and consequently yielding less precise results, it still performs better than DON. Although DON shows commendable results with familiar objects and configurations, its performance dips in novel scenarios, revealing a lack of generalization. These results collectively emphasize the significant advantages of our method in diverse and accurate object manipulation.

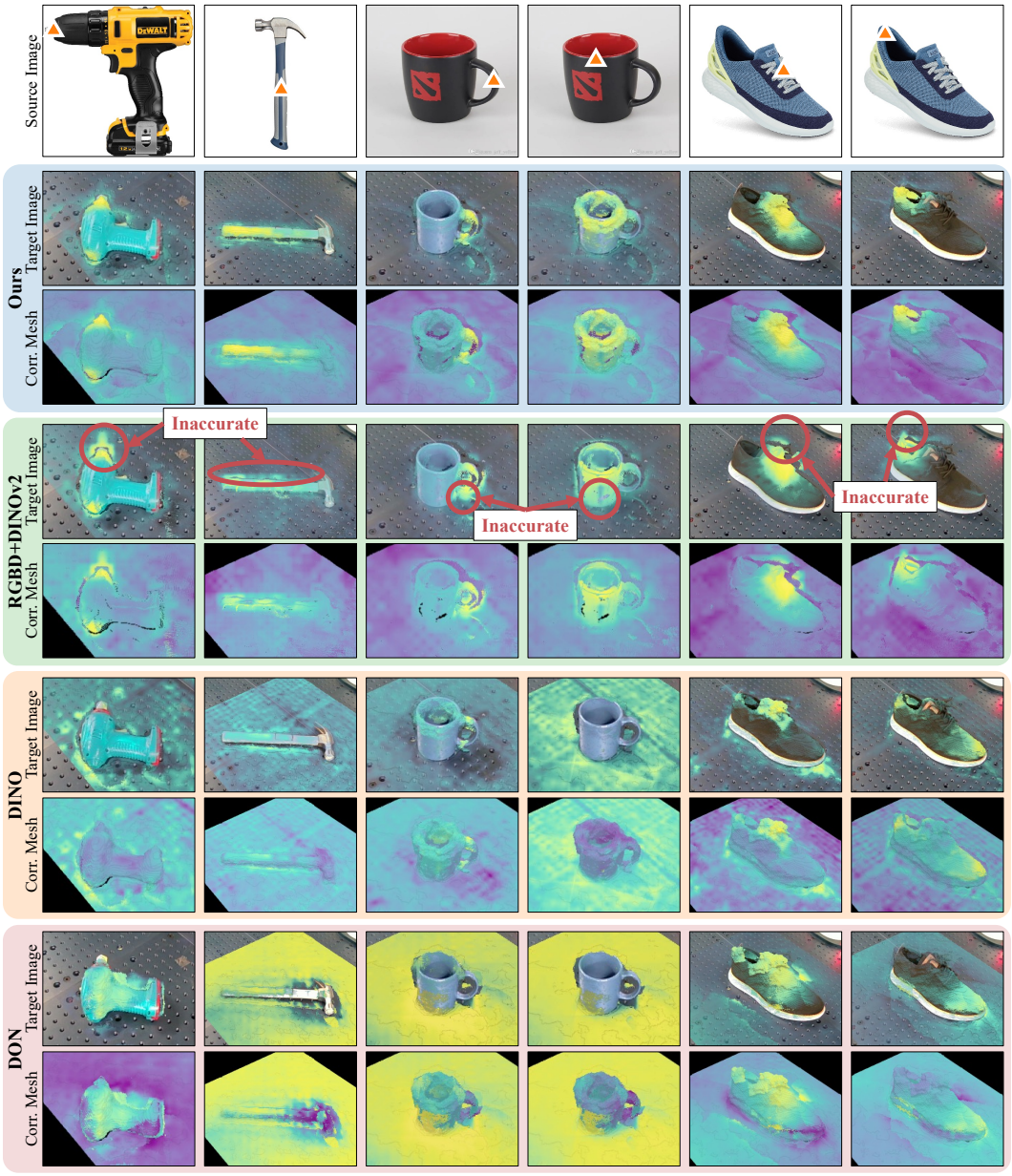

Figure 6: **Correspondence Quantitative Comparison.** The top row shows the selected pixel from the source image, and the following rows show the corresponding areas for different methods. While our method could have accurate correspondence, RGBD+DINOv2 corresponds to some points in the background. For example, the drill tip is not accurately highlighted in the RGBD+DINOv2 example, while ours can accurately highlight the drill tip. Ours with DINO feature backbones fail to identify objects accurately, while ours with DON fail to generalize to novel scenes and novel instances.

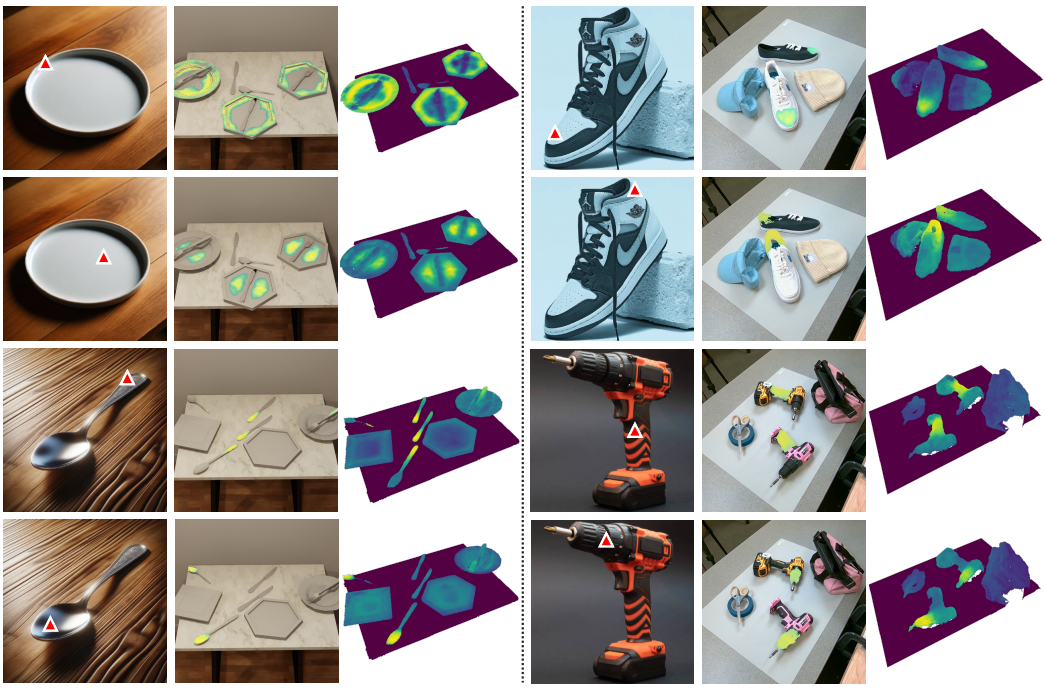

Figure 7: **Cross-Domain Correspondence.** The red triangles represent query points in the source image, and the corresponding areas are highlighted in the 3D mesh. First, we observe that our representation can encode features for object parts and establish the correspondence, such as spoon tips and spoon handles. In addition, we found the correspondence can be multimodal. When the shoe head is selected, multiple shoe heads in the workspace are highlighted. At last, the correspondence is generalizable across different contexts, instances, and domains, which demonstrates our method's generalization capabilities.

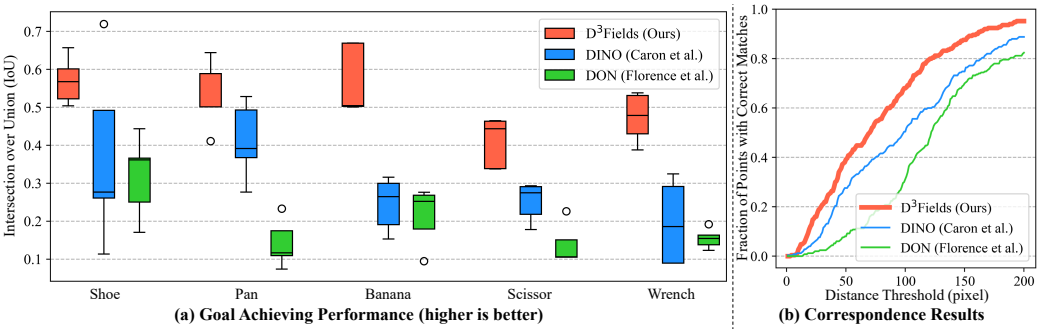

Figure 8: **Quantitative Evaluation.** We perform real-world quantitative evaluations by measuring final goal-achieving performance and keypoints correspondence accuracy. (a) We use IoU to measure goal-achieving performance. Results indicate that our method aligns with the goal configurations much better than DON and DINO across various object categories and scenarios. (b) We measure the keypoints correspondence accuracy according to the fraction of points with accurate matches, with correct matches determined by a distance threshold. Our method is consistently better at aligning with the goal image, regardless of the chosen threshold.

In Figure 8 (b), we present the correspondence results. We label 10 corresponding keypoint pairs on both the goal image and the final manipulation result to sufficiently evaluate the correspondence accuracy. The accuracy of correspondence was determined by calculating the proportion of keypoints that were accurately matched, using a predefined distance threshold as the criterion. If the distance between corresponding keypoints exceeds this threshold, they are determined as unmatched. Our method shows superior performance across various thresholds, consistently outperforming the baseline models. DINO emerges as the second-best in terms of performance, exhibiting broad applicability but with a lower precision compared to our method. Meanwhile, DON lags in performance, primarily due to its struggles with generalization in novel scenarios. These results, in conjunction with those from Figure 8 (a), reiterate our method's outstanding capabilities in both generalization

and accuracy. While DINO provides reasonable applicability, it lacks the precision of our approach, and the performance of DON is hindered by its limited adaptability.

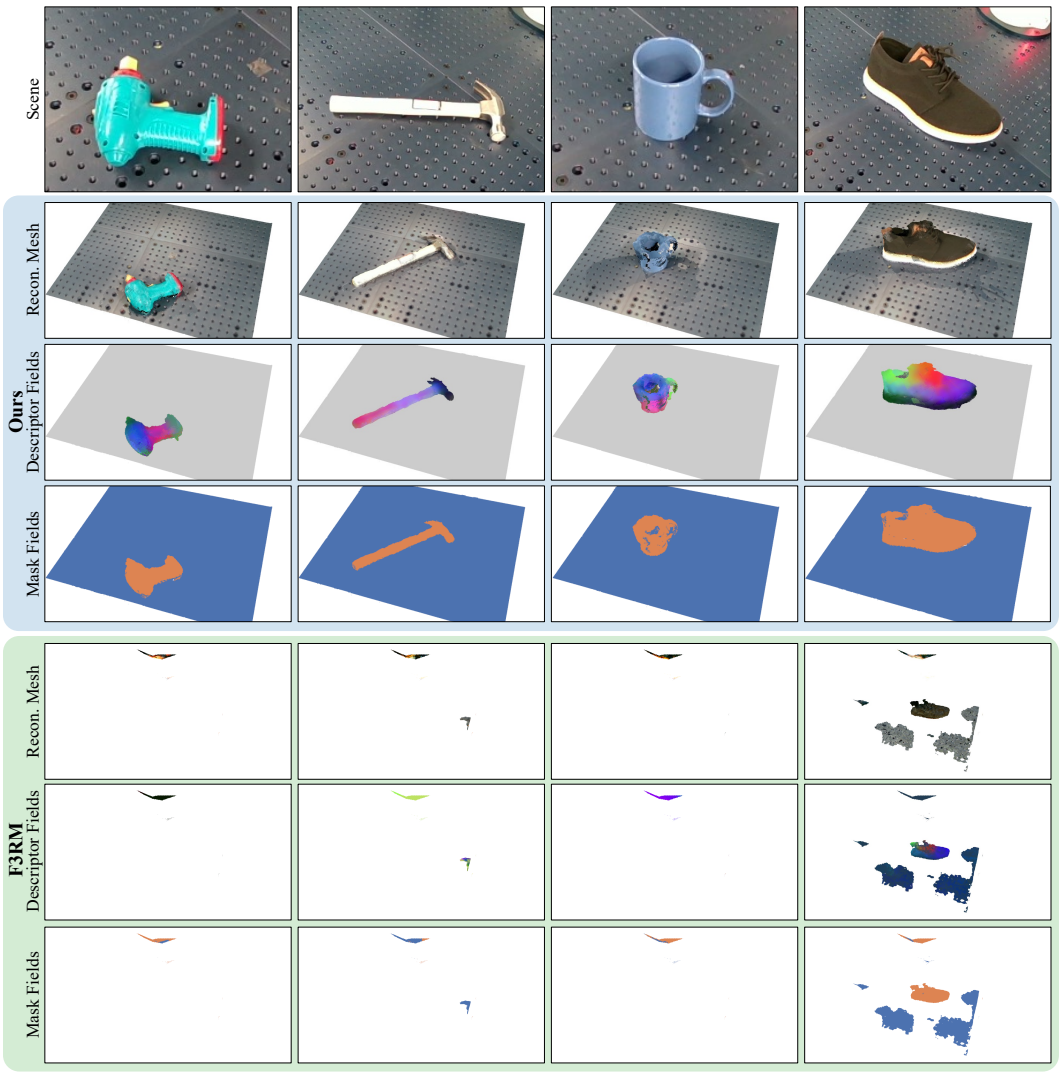

Figure 9: **Qualitative Comparison with F3RM.** We compare our representation with F3RM trained with additional Grounded-SAM on four scenes. We qualitatively evaluate their performance by reconstructing their mesh and visualizing descriptor fields and mask fields. Although Grounded-SAM supervision can help F3RM to segment out objects as shown in the shoe example, it does not contribute too much to reconstructing a quality mesh. In contrast, our representation can consistently reconstruct meshes and generate quality descriptor fields and mask fields across four scenes.

3.9 F3RM with Grounded-SAM

We also trained F3RM [1] on Grounded-SAM [5, 6] and compare with our method, as shown in Figure 9. For F3RM, we use the same setting as the main paper, except for the additional Grounded-SAM labeling. We reconstruct meshes and visualize the corresponding descriptor meshes and mask meshes, similar to what we did in the main paper. We have the following observations:

- F3RM could segment out objects given additional Grounded-SAM supervision, as shown in the shoe example. This indicates that our F3RM training pipeline works as expected.
- Additional Grounded-SAM supervision does not influence too much on results. We could see that reconstructed meshes are quite similar to the results from the main paper because

additional Grounded-SAM supervision does not contribute too much to NeRF training. Instead, sparse camera views are the main reason for the poor performance of F3RM.

- Our representation can consistently construct quality meshes and generate clear descriptor fields and mask fields, which demonstrates the effectiveness of our method.

3.10 Debris Experiment Details

In this section, we provide more details and visualization for our debris experiments. Our initial multi-view observations and goal images are visualized in Figure 10. In this task, we want the robot to push spreading almonds into one object pile. We could see that initially, the debris spreads over the workspace, which is quite different from the goal states. This task is challenging since the dynamics of object piles is hard to predict. In addition, such manipulation capability needs an efficient perception module so that it can gain visual feedback from the environment. Our zero-shot rearrangement framework can collect these almonds into one pile successfully using our dynamic representation and the learned dynamics model.

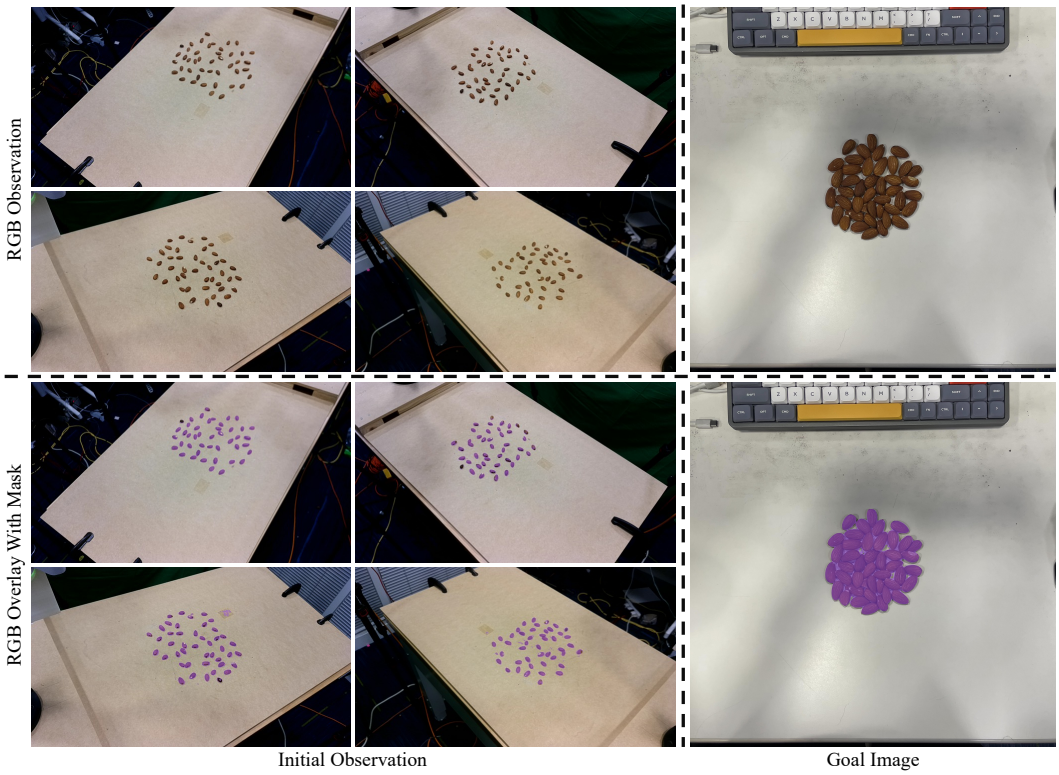

Figure 10: **Debris Experiments Details.** We visualize initial multi-view observations in our debris experiments and the corresponding goal image. The bottom row overlays the RGB observation with the mask. In this task, our objective is to collect spreading debris into an object pile, which is useful for tasks like cleaning.

References

[1] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot manipulation. In *7th Annual Conference on Robot Learning*, 2023.

[2] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In *International Conference on Computer Vision (ICCV)*, 2023.

[3] J. Ye, N. Wang, and X. Wang. Featurenerf: Learning generalizable nerfs by distilling foundation models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 8962–8973, 2023.

[4] P. R. Florence, L. Manuelli, and R. Tedrake. Dense object nets: Learning dense visual object descriptors by and for robotic manipulation. In A. Billard, A. Dragan, J. Peters, and J. Morimoto, editors, *Proceedings of The 2nd Conference on Robot Learning*, volume 87 of *Proceedings of Machine Learning Research*, pages 373–385. PMLR, 29–31 Oct 2018.

[5] S. Liu, Z. Zeng, T. Ren, F. Li, H. Zhang, J. Yang, C. Li, J. Yang, H. Su, J. Zhu, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. *arXiv preprint arXiv:2303.05499*, 2023.

[6] 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. *arXiv:2304.02643*, 2023.

[7] Y. Li, J. Wu, J.-Y. Zhu, J. B. Tenenbaum, A. Torralba, and R. Tedrake. Propagation networks for model-based control under partial observation. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1205–1211. IEEE, 2019.

[8] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9650–9660, 2021.

[9] S. Amir, Y. Gandelsman, S. Bagon, and T. Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2021.

[10] M. Oquab, T. Darcet, T. Moutakanni, H. V. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, R. Howes, P.-Y. Huang, H. Xu, V. Sharma, S.-W. Li, W. Galuba, M. Rabbat, M. Assran, N. Ballas, G. Synnaeve, I. Misra, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski. Dinov2: Learning robust visual features without supervision, 2023.