

MSGField: A Unified Scene Representation Integrating Motion, Semantics, and Geometry for Robotic Manipulation

Yu Sheng¹, Runfeng Lin¹, Lidian Wang¹, Quecheng Qiu¹, YanYong Zhang^{1,2}
 Yu Zhang^{1,2}, Bei Hua¹, and Jianmin Ji^{1,2,†}

Abstract— Combining accurate geometry with rich semantics has been proven to be highly effective for language-guided robotic manipulation. Existing methods for dynamic scenes either fail to update in real-time or rely on additional depth sensors for simple scene editing, limiting their applicability in real-world. In this paper, we introduce MSGField, a representation that uses a collection of 2D Gaussians for high-quality reconstruction, further enhanced with attributes to encode semantic and motion information. Specially, we represent the motion field compactly by decomposing each primitive’s motion into a combination of a limited set of motion bases. Leveraging the differentiable real-time rendering of Gaussian splatting, we can quickly optimize object motion, even for complex non-rigid motions, with image supervision from only two camera views. Additionally, we designed a pipeline that utilizes object priors to efficiently obtain well-defined semantics. In our challenging dataset, which includes flexible and extremely small objects, our method achieve a success rate of 79.2% in static and 63.3% in dynamic environments for language-guided manipulation. For specified object grasping, we achieve a success rate of 90%, on par with point cloud-based methods. Code and dataset will be released at: [MSGField.github.io](https://github.com/USTC-ML/MSGField).

I. INTRODUCTION

Embodied AI has accelerated the integration of robotic systems into daily life [26], [39]. It’s critical for robots that can interact with humans and autonomously perform household tasks in real-world unstructured environments [44]. To achieve this, the robot needs to comprehend the scene’s semantics for interaction and be aware of the objects’ geometry and motion within the environment for effective manipulation.

Recently, some methods [1], [3], [8], [11] employ point clouds to capture the geometry of dynamic scenes and project them onto images to obtain semantic features. However, these methods often struggle with accurately representing small objects and precise semantics. Others [2], [6], [10], [28], [43] use Neural Radiance Field (NeRF) [5] or 3D Gaussians Splatting (3DGS) [4] to extract more detailed semantic features, leading to enhanced scene understanding. But the geometric precision is limited, and per-scene optimization or simple scene edits make them infeasible for dynamic settings.

In this context, we explore the question, “**How to design a scene representation that simultaneously integrates precise motion, semantic, and geometric while remaining**

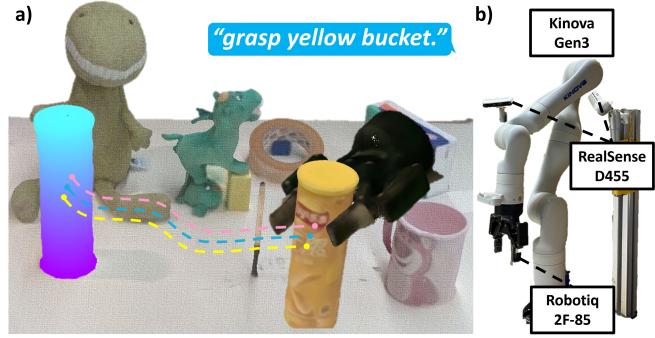


Fig. 1. a). We propose MSGField, a unified representation that enables robots to manipulate objects in real-world environments based on human instructions. b) The device we use for robotic manipulation in the real world.

compact enough for rapid training?” In this paper, we introduce MSGField, which employs extended 2D Gaussians Splatting (2DGS) [7] for high-fidelity surface reconstruction and provides a compact, comprehensive representation of scene motion and semantic, as shown in Fig. 1. Unlike existing methods that directly optimize features [28]–[30] or motions [16], [17], [20], [22] on a per-point basis, we argue that such information is consistently present within local regions, rendering per-point learning unnecessary. Specifically, for semantic filed, we design an object-centric semantic field distillation pipeline, ensuring primitives of the same object share a common semantic feature. This results in a lightweight yet well-defined semantic field, enabling fast training (30s) and precise 3D open-vocabulary segmentation. For the motion, we harness a small number of learnable motion bases to capture scene dynamics, with each Gaussian primitive’s trajectory represented as a linear composite of these bases. The motion representation, tightly integrated with Gaussian primitives, allows us to leverage the differentiable real-time rendering of 2DGS for rapid optimization (5s for rigid bodies motion) with only RGB supervision, even capturing non-rigid motions that previous methods unable to describe [2], [43]. We collect a real-world dataset comprising 30 diverse object categories, including complex surfaces objects and small cable tie only 5mm wide. Our method achieved 79.2% success in static and 63.3% in dynamic environments for language-guided manipulation, and 90.0% for target-specific tasks, comparable to point cloud-based methods.

Our main contributions are as follows:

- We introduce MSGField, a strategically designed scene representation that elegantly integrates motion, seman-

¹ School of Computer Science and Technology, University of Science and Technology of China (USTC), Hefei 230026, China

² Institute of Artificial Intelligence, Hefei Comprehensive National Science Center, Hefei, Anhui, China

† Corresponding author: jianmin@ustc.edu.cn

tics, and geometry, while meeting the robot’s need for timely response with a compact expression, significantly enhancing language-guided manipulation performance in real-world complex scenarios.

- To the best of our knowledge, MSGField is the first to apply 2DGS to robotic manipulation, with its precise reconstruction bridging the gap between 2D and 3D sensors
- We propose a well-defined semantic field and a compact motion field that facilitate precise 3D open-vocabulary segmentation and accurate recovery complex motion of dynamic objects.

II. RELATED WORK

A. Scene Representation for Robot Manipulation

Based on the implicit representation, F3RM [6], NeuGraspNet [8] and LERF-TOGO [10] utilize NeRF to reconstruct target scenes and employ feature distillation to achieve language-guided grasping. GeFF [11] utilizes pixelNeRF’s [13] single-frame reconstruction capability to achieve fast mapping. For explicit representation, [1], [3] accomplish one-shot manipulation by back-projecting panoramic RGB-D images and their corresponding image features into 3D space. GaussianGasper [2] models the environment using 3DGS, where each Gaussian primitive is associated with a feature attribute to learn corresponding semantic features. However, per-scene optimization limits the applicability of these methods in dynamic scenes. Although GeFF can reconstruct scenes swiftly, its mapping paradigm struggles with rapid scene changes, and the precision of single-frame reconstruction is limited. We leverage geometrically accurate radiance fields, 2DGS [7], combined with a compact motion field representation to address the aforementioned issues.

B. Semantic Field Distilling

Most of existing semantic field distilling methods [2], [6], [10], [28]–[30] use CLIP [37] to generate semantic feature maps and supervise NeRF or 3DGS models with per-pixel feature loss. Others [1], [3] use RGB-D cameras to back-project feature maps into 3D space. However, CLIP extracts object or patch level features shared across regions, while current methods redundantly learn features point by point, leading to high memory consumption, especially in explicit representation-based approaches [2], [28], [29]. [12] uses object-centric priors to maintain semantic consistency but relies on extensive pre-training, making it hard to combine with exist scene representation. In contrast, our approach achieves similar results with significantly lower time and space complexity.

C. Dynamic Reconstruction and Tracking

Dynamic reconstruction recovers 3D scenes at each timestamp, enabling the tracking of both rigid and non-rigid motions, whereas traditional tracking or pose estimation methods [33], [34] are limited to rigid bodies. Previous dynamic reconstruction techniques relied on panoramic RGB-D

cameras [14], [15] or manual priors [25]. Recently, methods based on NeRF and 3DGS have achieved state-of-the-art results. However, most of these approaches [16], [17], [20], [22]–[24] require panoramic cameras or RGB-D cameras around scene, which are challenging to implement on robots. Some works [19], [21] have demonstrated remarkable dynamic reconstruction using monocular images or RGB-D sequences. Yet, the several hours optimization hinders their real-world utility. In contrast, our method achieves motion tracking in seconds using only two camera views, offering a practical solution for robotics.

III. METHODS

In this sections, we first introduce preliminaries on Gaussian Splatting and detail why 2D Gaussians ensure precise geometric reconstruction. Then we discuss how the MSGField efficiently and precisely represents scene semantics and motions. Finally, we demonstrate our MSGField-based manipulation algorithm.

A. Preliminaries: Gaussian Splatting

3DGS employs a set of 3D Gaussians as primitives to represent the scene’s appearance and geometry. A 3D Gaussian primitive is parameterized as $\mathbf{g} \equiv (\boldsymbol{\mu}, \mathbf{R}, \mathbf{s}, \mathbf{o}, \mathbf{c})$, where $\boldsymbol{\mu} \in \mathbb{R}^3$ and $\mathbf{R} \in \mathbb{SO}(3)$ denote the 3D mean and orientation, while $\mathbf{s} \in \mathbb{R}^3$, $\mathbf{o} \in \mathbb{R}$ and $\mathbf{c} \in \mathbb{R}^3$ correspond to scale, opacity, and color, respectively. A 2D Gaussian primitive can be regarded as a degenerate ellipsoid, retaining only the first two dimensions of \mathbf{R} and \mathbf{s} . For convenience, we use the same parameterization as for a 3D Gaussian, treating the first two dimensions of \mathbf{R} as two principal tangential vectors \mathbf{t}_u and \mathbf{t}_v of 2D Gaussian. They both employ α -blending to render a pixel color C , as detailed in Eq. (1),

$$C = \sum_{i \in \mathcal{N}} c_i \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (1)$$

where \mathcal{N} is ordered points overlapping the pixel, c_i is the color of each point and α_i is the opacity multiplied Gaussian covariance. Methods [21], [28], [29] replace c_i in Eq. (1) with each primitive’s distance to the camera or its semantic features to generate a depth or semantic feature map. Supervising the model with corresponding ground truth enables 3D reconstruction or semantic distillation; however, ensuring accuracy remains challenging. For 3D reconstruction, the absence of distance constraints between Gaussian primitives can lead to α -blended depths that fall between them. In contrast, 2DGS offer a more accurate geometric representation and incorporate a depth distortion loss to regulate inter-primitive distances. For feature distillation, α -blending complicates maintaining meaningful semantic features, as blending colors like red and green to get yellow is valid, but blending different features is meaningless. We will demonstrate this in our experiments in Section IV-E.

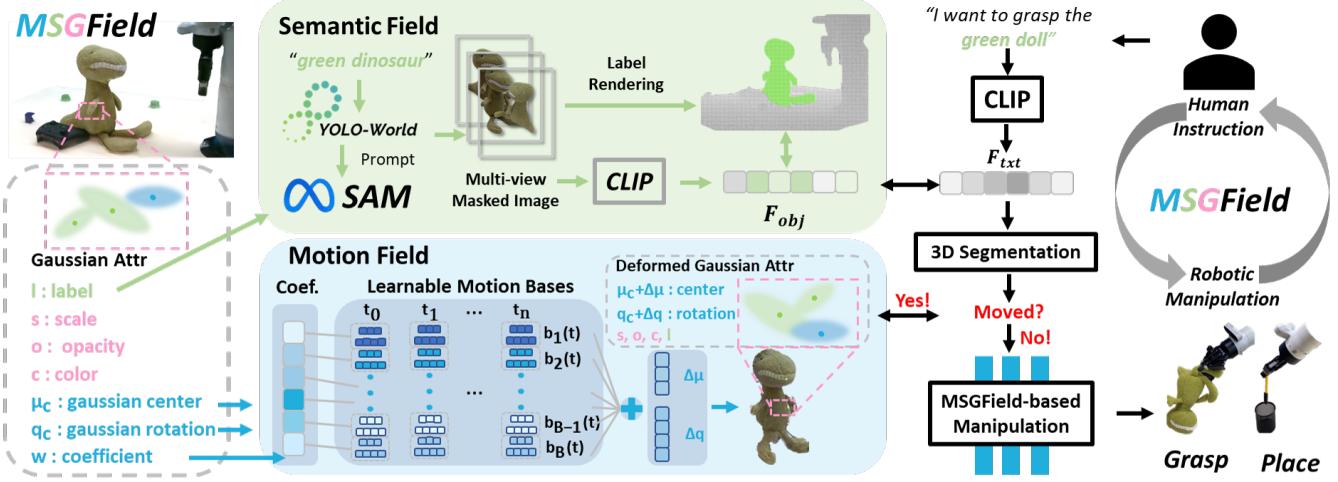


Fig. 2. The framework of MSGField. Geometry field captured by surface reconstruction from 2D Gaussian Splatting. In the semantic field, each primitive is assigned a label, which links to an object feature extract from CLIP. For the motion field, we represent scene motion with motion bases, where each primitive's motion is a combination of these bases. With human instructions, objects are segmented via text queries, and their motion is tracked. A grasp detector then identifies executable manipulations.

B. Object-Centric Semantic Field Distillation

Based on the analysis in Section III-A, we design a field distillation framework to ensure feature consistency within objects, as shown in Fig. 2. For preprocessing, YOLO-World [31] detects the object from multiple views using language prompts, and the detection results guide the Segment Anything Model (SAM) [38] to generate accurate object masks. Next, we use ray-splat intersection method [35] to determine whether each primitive belongs to the object, a process we refer to as label rendering. Specifically, for a primitive in the camera coordinate system, we define the matrix T :

$$T = \begin{pmatrix} \mathbf{t}_s & \mathbf{t}_v & \boldsymbol{\mu} \\ 0 & 0 & 1 \end{pmatrix}^T P^T \in \mathbb{R}^{3 \times 4}, \quad (2)$$

where $P \in \mathbb{R}^{4 \times 4}$ is projection matrix of camera. The projection of the 2D Gaussian onto image plane, with a center $\mathbf{p} = (p_1, p_2)$ and half extends of bounds $\mathbf{h} = (h_1, h_2)$, can be expressed as:

$$\begin{aligned} p_i &= d^{-1}(1, 1, -1)(T_i \star T_4), \\ h_i &= \sqrt{p_i^2 - d^{-1}(1, 1, -1)(T_i \star T_i)}, \\ d &= (1, 1, -1)(T_4 \star T_4), \end{aligned} \quad (3)$$

where T_i is the i -th column of T , and \star meas component-wise product. The projection boundaries can be obtained as $x_1, x_2 = p_1 \pm h_1$ and $y_1, y_2 = p_2 \pm h_2$, as illustrated in Fig. 3 a). A primitive is considered part of the object when its projection boundaries fall within the mask obtained from SAM, as shown in Fig. 3 b). However, background points within the view frustum also fall within the mask. To filter them out, we assign all primitives a black color and render the scene, then compute the loss between the mask (where only the object remains white) and the rendered image. By backpropagating, only foreground points of the object receive gradients. Points with gradients above a threshold are considered as target points and share the same feature

\mathbf{F}_{obj} extracted by CLIP. To support 3D open-vocabulary segmentation, we follow [28] to define the similarity s between the text query \mathbf{F}_{txt} and \mathbf{F}_{obj} as:

$$s = \min_i \frac{\exp(\mathbf{F}_{txt} \cdot \mathbf{F}_{obj})}{\exp(\mathbf{F}_{txt} \cdot \mathbf{F}_{obj} + \exp(\mathbf{F}_{canon}^i \cdot \mathbf{F}_{obj}))}, \quad (4)$$

where \mathbf{F}_{canon}^i is the CLIP embeddings of a predefined canonical phrase chosen from “object”, “things”, “stuff”, and “texture”. The primitives with the highest similarity is segmented. Given that the feature dimension is 512 or larger, each primitive is assigned a one-dimensional label attribute linked to object-level features to improve storage efficiency. Thanks to the rapid rendering of 2DGS, the entire process, including preprocessing, takes less than 30 seconds.

C. Motion Field Parameterization

The Motion Field aims to recover primitive positions and orientations over time from a sequence of video frames $\{\mathbf{I}_{t_i} \in \mathbb{R}^{H \times W \times 3}\}_{i=0}^t$. Specifically, for a 2D Gaussian primitive, the attributions at time t_0 are represented as $\mathbf{g}^{t_0} \equiv (\boldsymbol{\mu}^{t_0}, \mathbf{q}^{t_0}, \mathbf{s}, \mathbf{o}, \mathbf{c})$. Here, we use the unit quaternion $\mathbf{q} \in \mathbb{S}^3$ to represent orientation for efficient optimization, as done in [32]. Assuming that \mathbf{s} , \mathbf{o} and \mathbf{c} remain unchanged over time, the position and orientation at time t_i are:

$$\boldsymbol{\mu}^{t_i} = \boldsymbol{\mu}^{t_0} + \Delta\boldsymbol{\mu}_{0 \rightarrow t_i}, \quad \mathbf{q}^{t_i} = \text{normalize}(\mathbf{q}^{t_0} + \Delta\mathbf{q}_{0 \rightarrow t_i}), \quad (5)$$

where $\Delta\boldsymbol{\mu}_{0 \rightarrow t_i}$ and $\Delta\mathbf{q}_{0 \rightarrow t_i}$ represent the motion changes we aim to capture. We observe that the motion remaining consistent within a given area. In the extreme case of rigid body motion, all primitives share the same movement. Thus, optimizing each primitive's motion individually is redundant and may lead to uncontrolled deformations. Inspired by [32], we use a set of learnable motion bases $\{(\boldsymbol{\mu}_{0 \rightarrow t_i}^{(b)}, \mathbf{q}_{0 \rightarrow t_i}^{(b)})\}_{b=1}^B$ to compactly represent scene motion. Each primitive's motion is decomposed into a combination of these bases, as expressed in Eq. (6), where $\{w^{(b)}\}_{b=1}^B$ represents the motion

Algorithm 1: Robot Manipulation with MSGField

```

1: Initial: robot.move( $P_{\text{home}}$ ), robot.grasp_open
2: Input:  $L_{\text{operation}}, L_{\text{obj}} \leftarrow \text{language}$ 
3:  $G_{\text{obj}} \leftarrow \text{MSGField}.\text{semantic}(L_{\text{obj}})$ 
4:  $P_{\text{obj}} \leftarrow \text{MSGField}.\text{geometry}(G_{\text{obj}})$ 
5: while NOT (robot.current_pose ==  $P_{\text{obj}}$ ) do
6:    $I \leftarrow \text{Camera}.\text{perception}$ 
7:   if detect_motion( $I$ ) then
8:     robot.stop
9:      $M_{\text{obj}} \leftarrow \text{MSGField}.\text{motion}(G_{\text{obj}}, I)$ 
10:    MSGField.update( $M_{\text{obj}}$ )
11:     $P_{\text{obj}} \leftarrow \text{MSGField}.\text{geometry}(G_{\text{obj}})$ 
12:   else
13:     robot.move( $P_{\text{obj}}$ )
14:   end if
15: end while
16: if  $L_{\text{operation}} == \text{'grasp'}$  then
17:   robot.grasp_close
18: else if  $L_{\text{operation}} == \text{'place'}$  then
19:   robot.grasp_open
20: else
21:   Raise NotImplementedException
22: end if

```

coefficient of the primitive.

$$\Delta \mu_{0 \rightarrow t_i} = \sum_{b=1}^B w^{(b)} \mu_{0 \rightarrow t_i}^{(b)}, \Delta \mathbf{q}_{0 \rightarrow t_i} = \sum_{b=1}^B w^{(b)} \mathbf{q}_{0 \rightarrow t_i}^{(b)}. \quad (6)$$

We focus on optimizing the dynamic primitives, treating other as static. To accelerate optimization, we employ sparse frame sampling to \mathbf{I}_{t_i} . However, this leads to sampled frames with minimal overlap, causing most primitives to compute losses with the background and hindering convergence. To address this, we introduce Dice loss L_{Dice} , which emphasizes global pixel distribution over pixel-wise RGB or SSIM losses [7]. The L_{Dice} is:

$$L_{\text{Dice}} = 1 - \text{Dice}(\mathbf{I}_{\text{opacity}}, \mathbf{I}_{\text{mask}}). \quad (7)$$

Here, $\mathbf{I}_{\text{opacity}}$ represents the image rendered with the dynamic primitives' opacity, and \mathbf{I}_{mask} is the dynamic object mask for the current frame. This loss provides a global optimization direction, allowing the primitives to quickly adjust to the region corresponding to the current frame. We retained the rigidity loss from [32] to encourage rigidity between primitives. Our total loss L is:

$$L = L_{\text{RGB}} + \lambda_1 L_{\text{SSIM}} + \lambda_2 L_{\text{Dice}} + \lambda_3 L_{\text{Rigidity}}. \quad (8)$$

For rigid body motion, where all primitives share the same $\text{SE}(3)$ transformation, we assign each primitive the same fixed motion coefficient and only optimize the motion bases. Rotation and translation changes are applied as an $\text{SE}(3)$ matrix to the original primitives.

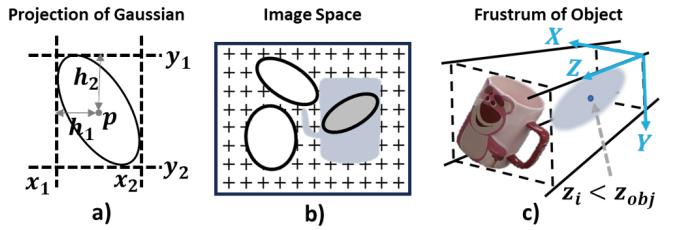


Fig. 3. a). Projection of the 2D Gaussian on the image. b). Determining whether a Gaussian primitive falls within the mask. The gray primitive is inside. c). The gray primitives are the ones occluding the object.

D. MSGField-Based Language-Guided Manipulation

Leveraging the precise motion, semantic, and geometric information provided by MSGField, robots can accurately manipulate objects in dynamic real-world environments based on natural language instructions. The pseudocode of our language-guided manipulation algorithm is illustrated in Algorithm 1. We perform both grasping and placing operations, primarily by controlling the end-effector's pose and the gripper's opening and closing using the robot API. Specifically, the object primitives G_{obj} are segmented according to the text query L_{obj} , and the end-effector's pose P_{obj} is determined based on [39] (Line 3-4 in Algorithm 1). We convert the precise mesh from MSGField into a pseudo RGB-D point cloud, which serves as the input for [39]. We use a straightforward feature matching method to detect object motion. If the object moves, the MSGField will update, allowing the robot to adjust and execute the manipulation effectively (Line 5-15 in Algorithm 1). We observed that [39] is sensitive to variations in camera coordinate systems, and a single camera perspective often encounters occlusion issues. Therefore, we sample multiple camera perspectives that minimize occlusion and combine the grasps derived from them. Specifically, for a camera view with a projection matrix P , we apply perspective division transform the positions of primitives μ_i into Normalized Device Coordinates (NDC) space:

$$\mu_i^{ndc} = \mu_i^{clip}[:, 3] / \mu_i^{clip}[3], \quad \mu_i^{clip} = P \begin{pmatrix} \mu_i \\ 1 \end{pmatrix}. \quad (9)$$

The viewing frustum transforms into a standard cuboid in NDC space, where occlusion points can be easily identified, as shown in Fig. 3 c). We select views with minimal occlusion and filter out grasps misaligned with the robotic arm's orientation to obtain more robust grasp poses.

IV. EXPERIMENTS

A. Setup, Dataset and Metric

1) Setup: Fig. 1 illustrates our experimental setup. We use a Kinova-Gen3 robotic arm equipped with a Robotiq 2F-85 gripper. Two RealSense-D455 depth cameras are positioned on either side to capture object motion. Notably, our method relies solely on RGB data. Both cameras are precisely calibrated using hand-eye calibration [41] to ensure alignment with the robot arm's base. Another camera of the same type, along with the two cameras, provides panoramic views. The optimization runs on an RTX 3090 remote server, while the

TABLE I
LANGUAGE-GUIDED MANIPULATION SUCCESS RATE

Scene	Success Rate(%) ↑		
	F3RM [6]	GraspNet(RGB-D) [39]	MSGField
Daily Objects	76.7	-	90.0
Small Objects	30.0	-	56.7
Flexible Objects	23.3	-	80.0
Place	66.7	-	90.0
Rigid Body Motion [†]	-	86.7	83.3
Non-Rigid Motion [†]	-	90.0	43.3
Static Avg.	49.2	-	79.2
Dynamic Avg.	-	88.4	63.3

[†] Without language guide.

laptop is used to control the robot and transfer messages. We integrate and manage the hardware components with ROS.

2) Dataset: Our dataset includes both static and dynamic scenes. The static scenes are categorized into three groups based on object type: daily objects, small objects, and flexible objects, and the dynamic scenes are divided into rigid and non-rigid body motion. Each category contains two distinct scenarios, with the entire dataset spanning up to 30 distinct object types, as shown in Fig. 5. For each scene, we capture 16 panoramic images and use COLMAP [42] to estimate the extrinsic parameters, which were then scaled and aligned to the robot arm’s base with extrinsic parameters of the two pre-calibrated cameras. For dynamic scenes, we capture an additional video sequence using two calibrated cameras.

3) Metric: We assess the methods based on the operational success rate, consistent with [6], [39]. For each scene, we conduct a total of 30 manipulations attempts. A successful grasp occurs when the gripper picks up the specified object without external intervention and holds it suspended. A successful placement requires the robot to place the object within the designated area without disturbing the existing arrangement.

B. Implant Details

We first use 2D Gaussian Splatting to train a geometry field. We train it for 20k iterations in our experiments (10k iterations is enough for most scene). The feature field and motion field are generated on-site. For the feature field, we use OpenCLIP ViT-B/16 model and SAM ViT-H model to extract features. The motion field is optimized using the Adam optimizer, with total 300 iterations for rigid body motion and 1000 iterations for non-rigid motion per frame. The number of motion bases is set to 10. We choose F3RM [6], a recent work which employs NeRF to distill a feature field enabling language-guided manipulation, as our baseline. Following the setting in [6], we collect 40 panoramic images and optimize each scene for 2,000 iterations for it. Since F3RM cannot handle dynamic scenes, we directly compare our method with point cloud-based approach GraspNet [39] to assess the accuracy of the motion field. The real-time calibrated RGB-D frames are used as input for [39], with the target object for grasping manually specified.

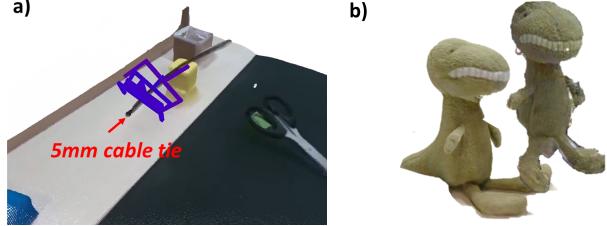


Fig. 4. a) MSGField produces accurate meshes and can generate robust grasp poses, even for a 5mm cable tie. b) MSGField effectively optimizes non-rigid motion, recovering a toy’s transition from sitting to standing.

TABLE II
STATIC SCENE GRASPING WITHOUT LANGUAGE GUIDING

Scene	Success Rate(%) ↑	
	GraspNet(RGB-D based) [39]	MSGField
Daily Objects	100	93.3
Small Objects	50.0	90.0
Flexible Objects	90.0	86.7
All Avg.	80.0	90.0

C. Overall Results and Analysis

Static Scene. As shown in Table I, our method achieved an average success rate of 79.2% in static scenes, surpassing [6] by a large margin, demonstrating its accuracy and generalization capability. [6] performs well with daily objects but struggles with the small and flexible object. This is mainly due to insufficient modeling accuracy and the limitations of its matching-based grasp detection algorithm, which struggles to generalize to flexible items with complex surfaces. Our method’s success rate also significantly declines with small objects, primarily because their indistinct features make natural language matching difficult. For example, “black cable tie” is frequently mismatched, which is a major source of fail in our approach. The main reason we outperform [6] in placement tasks is that querying with “black pen container” or “yellow bowl” in [6] often includes many desktop points, leading to inaccurate position estimation and failure. In contrast, our object-centered semantic field effectively isolates the queried objects.

Dynamic Scene. In dynamic scenes, our method is nearly on par with GraspNet for rigid motion, without the requirement for a depth sensor. When the object moves farther from the camera or to the camera’s side, the accuracy of [39] begins to decline, whereas our method does not encounter this issue. Our method often fails when an object reveals previously unseen sides, such as a side contacted with table. For non-rigid motion, the motion field is well-optimized in camera-supervised views within 2 minutes, as shown in Fig. 4 b), resulting in a success rate of 43.3%. The success rate drops as the limited supervision from only two viewpoints makes it difficult to optimize the motion of primitives in unsupervised views. [39] directly uses real-time RGB-D as inputs, it can achieve the same performance as in static scenes.

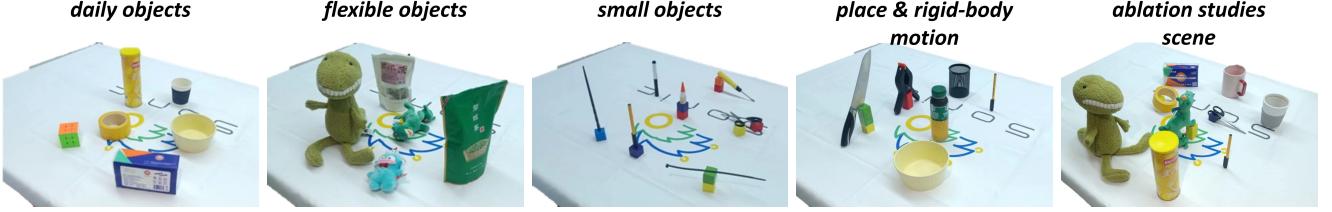


Fig. 5. Visualization of certain scenes from the dataset. We strongly recommend watching the supplementary video for more detailed information.

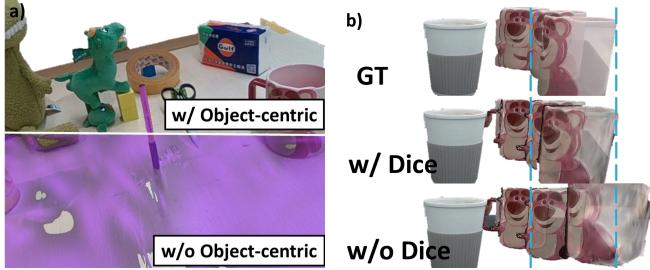


Fig. 6. a) Use a “yellow pen” to query and project the semantic field onto the image. Object-centric feature distillation focuses on the corresponding primitives as the purple visualization results. b) Without the constraint of the Dice loss, the motion fails to converge.

D. Reconstruction Accuracy Analysis

To further assess our reconstruction accuracy, we compared our method with GraspNet [39]. We convert the geometric information from MSGField to match the same perspective as GraspNet inputs and manually specify the objects to be grasped. As shown in Table II, our method achieves much higher accuracy than [39] for small objects. The main challenge with RGB-D point clouds is their difficulty in accurately modeling small objects, especially as distance increases. In contrast, the MSGField provides highly precise geometry, clearly capturing objects as small as a 5mm cable tie, as shown in Fig. 4 a). For regular-sized and flexible objects, our method’s modeling accuracy is on par with RGB-D. The slight difference in success rate compared to [39] is primarily due to minor distortions in some scenes caused by low lighting conditions.

E. Ablation Studies

In this section, we demonstrate the effectiveness and efficiency of different components of our method in terms of grasping success rate and optimization time. As shown in Fig. 5, we designed a scene with various objects, moving the pink mug for simulating rigid body motion and the green dinosaur toy for non-rigid motion scenarios.

Impact of 2D Gaussian. We replaced the geometric component of the scene representation with 3D Gaussian, as done in GaussianGrasper [2]. To eliminate the effects of other factors, we manually specified the target object for grasping in a static environment. As shown in Table IV, using 3D Gaussian results in a 63.4% decrease in performance. Visualization of the reconstruction results reveals significant surface distortions with 3D Gaussians, causing the generated grasp to shift, leading to failed grasps. While 2D Gaussian has a longer optimization time per iteration, it achieves much better surface reconstruction quality.

TABLE III
ABLATION RESULTS ABOUT SUCCESS RATE

2D-GS	Component		Success Rate(%) ↑	
	Object-Centric	Dice Loss	Static Obj.	Dynamic Obj.
✓	✓	✓	86.7	56.7
		✓	23.3	-
✓		✓	0.0	0.0
✓	✓	-	-	13.3

TABLE IV
ABLATION RESULTS ABOUT TRAINING TIME

2D-GS	Component		Training Time(min) ↓		
	Object-Centric	Dice Loss	Geometry	Semantic	Motion
✓	✓	✓	20.6	0.5	0.1
		✓	11.3	0.5	0.1
✓		✓	20.6	16.9	0.1
✓	✓	-	20.6	0.5	0.1

Impact of Object-Centric Feature Distillation. To validate the effectiveness of Object-Centric feature distillation, we replaced our method with LangSplat [28], a recent per-primitive feature distillation approach. Following the setup in [28], we use the CLIP and SAM models as mentioned in Section IV-B to extract image features and compress the CLIP features to 3 dimensions with autoencoder. The feature field was trained for 30,000 iterations. [28] achieves excellent semantic segmentation performance on 2D images, consistent with the results reported in the paper. However, for 3D segmentation, using the same decoder to recover features to 512 dimensions, it becomes nearly impossible to directly query the corresponding primitives using natural language, as shown in Fig. 6 a). Even increasing the compressed feature dimensions (up to 32 dimensions on RTX 3090) to reduce information loss yielded unsatisfactory results. This supports our analysis in Section III-A, where we showed that features lose their meaning after α -blending. And the time consumption (without preprocessing) is also unacceptable.

Impact of Dice Loss. To assess the effectiveness of Dice loss, we remove it and optimize the object’s motion. As shown in Fig. 6 b), optimization can be converged with minor translations without Dice loss. However, with rotations or larger movements, relying solely on per-pixel loss fails to converge, as even slight movements can significantly affect RGB loss. Integrating Dice loss improves motion estimation, ensuring better position optimization and a 43.4% increase in manipulation success rate.

V. CONCLUSIONS

This paper presents a unified scene representation that integrates motion, semantic, and geometric information. It effectively captures object-level semantics and spatial motion, and its compact nature allows rapid optimization. However, the method has its limitations. Firstly, incorporating 2D Gaussian Splatting results in higher computational costs. Moreover, under supervision by only two RGB cameras, it has difficulty precisely optimizing highly complex non-rigid motions. Nevertheless, advancements in robotics and compute vision are expected to address these challenges. Moving forward, we aim to enhance reconstruction speed and explore the application of this system in mobile robots.

REFERENCES

- [1] Q. Wang, H. Zhang, C. Deng, Y. You, H. Dong, Y. Zhu, and L. Guibas, “Sparsedff: Sparse-view feature distillation for one-shot dexterous manipulation,” *arXiv preprint arXiv:2310.16838*, 2023.
- [2] Y. Zheng, X. Chen, Y. Zheng, S. Gu, R. Yang, B. Jin, P. Li, C. Zhong, Z. Wang, L. Liu, *et al.*, “Gaussiangrasper: 3d language gaussian splatting for open-vocabulary robotic grasping,” *arXiv preprint arXiv:2403.09637*, 2024.
- [3] Y. Wang, Z. Li, M. Zhang, K. Driggs-Campbell, J. Wu, L. Fei-Fei, and Y. Li, “D3fields: Dynamic 3d descriptor fields for zero-shot generalizable robotic manipulation,” *arXiv preprint arXiv:2309.16118*, 2023.
- [4] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, “3d gaussian splatting for real-time radiance field rendering.” *ACM Trans. Graph.*, vol. 42, no. 4, pp. 139–1, 2023.
- [5] 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,” *Communications of the ACM*, vol. 65, no. 1, pp. 99–106, 2021.
- [6] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola, “Distilled feature fields enable few-shot language-guided manipulation,” *arXiv preprint arXiv:2308.07931*, 2023.
- [7] B. Huang, Z. Yu, A. Chen, A. Geiger, and S. Gao, “2d gaussian splatting for geometrically accurate radiance fields,” in *ACM SIGGRAPH 2024 Conference Papers*, 2024, pp. 1–11.
- [8] S. Jauhri, I. Lunawat, and G. Chalvatzaki, “Learning any-view 6dof robotic grasping in cluttered scenes via neural surface rendering,” *arXiv preprint arXiv:2306.07392*, 2023.
- [9] G. Tzias, Y. Xu, Z. Li, and H. Kasaei, “3d feature distillation with object-centric priors,” *arXiv preprint arXiv:2406.18742*, 2024.
- [10] 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 *7th Annual Conference on Robot Learning*, 2023.
- [11] R.-Z. Qiu, Y. Hu, G. Yang, Y. Song, Y. Fu, J. Ye, J. Mu, R. Yang, N. Atanasov, S. Scherer, *et al.*, “Learning generalizable feature fields for mobile manipulation,” *arXiv preprint arXiv:2403.07563*, 2024.
- [12] G. Tzias, Y. Xu, Z. Li, and H. Kasaei, “3d feature distillation with object-centric priors,” *arXiv preprint arXiv:2406.18742*, 2024.
- [13] A. Yu, V. Ye, M. Tancik, and A. Kanazawa, “pixelnerf: Neural radiance fields from one or few images,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 4578–4587.
- [14] A. Bozic, M. Mollhofer, C. Theobalt, and M. Nießner, “Deepdeform: Learning non-rigid rgb-d reconstruction with semi-supervised data,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 7002–7012.
- [15] M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escalano, C. Rhemann, D. Kim, J. Taylor, *et al.*, “Fusion4d: Real-time performance capture of challenging scenes,” *ACM Transactions on Graphics (ToG)*, vol. 35, no. 4, pp. 1–13, 2016.
- [16] Y.-H. Huang, Y.-T. Sun, Z. Yang, X. Lyu, Y.-P. Cao, and X. Qi, “Scgs: Sparse-controlled gaussian splatting for editable dynamic scenes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 4220–4230.
- [17] J. Luiten, G. Kopanas, B. Leibe, and D. Ramanan, “Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis,” in *2024 International Conference on 3D Vision (3DV)*. IEEE, 2024, pp. 800–809.
- [18] K. R. Barad, A. Richard, J. Dentler, M. Olivares-Mendez, and C. Martinez, “Object-centric Reconstruction and Tracking of Dynamic Unknown Objects using 3D Gaussian Splatting. [Online]. Available: <http://arxiv.org/abs/2405.20104>
- [19] D. Cai, J. Heikkilä, and E. Rahtu, “Gs-pose: Cascaded framework for generalizable segmentation-based 6d object pose estimation,” *arXiv preprint arXiv:2403.10683*, 2024.
- [20] T. Fischer, J. Kulhanek, S. R. Buld, L. Porzi, M. Pollefeys, and P. Kortscheder, “Dynamic 3d gaussian fields for urban areas,” *arXiv preprint arXiv:2406.03175*, 2024.
- [21] Q. Wang, V. Ye, H. Gao, J. Austin, Z. Li, and A. Kanazawa, “Shape of motion: 4d reconstruction from a single video,” *arXiv preprint arXiv:2407.13764*, 2024.
- [22] A. Bansal, M. Vo, Y. Sheikh, D. Ramanan, and S. Narasimhan, “4d visualization of dynamic events from unconstrained multi-view videos,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020, pp. 5366–5375.
- [23] M. Broxton, J. Flynn, R. Overbeck, D. Erickson, P. Hedman, M. Duval, J. Dourgarian, J. Busch, M. Whalen, and P. Debevec, “Immersive light field video with a layered mesh representation,” *ACM Transactions on Graphics (TOG)*, vol. 39, no. 4, pp. 86–1, 2020.
- [24] A. Cao and J. Johnson, “Hexplane: A fast representation for dynamic scenes,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 130–141.
- [25] S. Kumar, Y. Dai, and H. Li, “Monocular dense 3d reconstruction of a complex dynamic scene from two perspective frames,” in *Proceedings of the IEEE international conference on computer vision*, 2017, pp. 4649–4657.
- [26] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” *arXiv preprint arXiv:2304.13705*, 2023.
- [27] C. Wang, L. Fan, J. Sun, R. Zhang, L. Fei-Fei, D. Xu, Y. Zhu, and A. Anandkumar, “Mimicplay: Long-horizon imitation learning by watching human play,” *arXiv preprint arXiv:2302.12422*, 2023.
- [28] M. Qin, W. Li, J. Zhou, H. Wang, and H. Pfister, “Langsplat: 3d language gaussian splatting,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 20051–20060.
- [29] S. Zhou, H. Chang, S. Jiang, Z. Fan, Z. Zhu, D. Xu, P. Chari, S. You, Z. Wang, and A. Kadambi, “Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 21676–21685.
- [30] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik, “Lerf: Language embedded radiance fields,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 19729–19739.
- [31] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, “Yoloworld: Real-time open-vocabulary object detection,” in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024.
- [32] A. Kratimenos, J. Lei, and K. Daniilidis, “Dynmf: Neural motion factorization for real-time dynamic view synthesis with 3d gaussian splatting,” *arXiv preprint arXiv:2312.00112*, 2023.
- [33] B. Huang, J. Yu, and S. Jain, “Earl: Eye-on-hand reinforcement learner for dynamic grasping with active pose estimation,” in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 2963–2970.
- [34] B. Wen, W. Yang, J. Kautz, and S. Birchfield, “Foundationpose: Unified 6d pose estimation and tracking of novel objects,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024, pp. 17868–17879.
- [35] T. Weyrich, S. Heinzle, T. Aila, D. B. Fasnacht, S. Oetiker, M. Botsch, C. Flraig, S. Mall, K. Rohrer, N. Felber, *et al.*, “A hardware architecture for surface splatting,” *ACM Transactions on Graphics (TOG)*, vol. 26, no. 3, pp. 90–es, 2007.
- [36] M. Zwicker, H. Pfister, J. Van Baar, and M. Gross, “Ewa volume splatting,” in *Proceedings Visualization, 2001. VIS’01*. IEEE, 2001, pp. 29–538.
- [37] A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, *et al.*, “Learning transferable

- visual models from natural language supervision,” in *International conference on machine learning*. PMLR, 2021, pp. 8748–8763.
- [38] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, *et al.*, “Segment anything,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 4015–4026.
- [39] H.-S. Fang, C. Wang, M. Gou, and C. Lu, “Graspnet-1billion: A large-scale benchmark for general object grasping,” in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2020, pp. 11444–11453.
- [40] J. K. S. B. Bowen Wen, Wei Yang, “FoundationPose: Unified 6d pose estimation and tracking of novel objects,” in *CVPR*, 2024.
- [41] K. Daniilidis, “Hand-eye calibration using dual quaternions,” *The International Journal of Robotics Research*, vol. 18, no. 3, pp. 286–298, 1999.
- [42] J. L. Schonberger and J.-M. Frahm, “Structure-from-motion revisited,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 4104–4113.
- [43] O. Shorinwa, J. Tucker, A. Smith, A. Swann, T. Chen, R. Firooz, M. D. Kennedy, and M. Schwager, “Splat-mover: Multi-stage, open-vocabulary robotic manipulation via editable gaussian splatting,” in *8th Annual Conference on Robot Learning*, 2024.
- [44] Y. Liu, W. Chen, Y. Bai, J. Luo, X. Song, K. Jiang, Z. Li, G. Zhao, J. Lin, G. Li, *et al.*, “Aligning cyber space with physical world: A comprehensive survey on embodied ai,” *arXiv preprint arXiv:2407.06886*, 2024.