

ManiGaussian: Dynamic Gaussian Splatting for Multi-task Robotic Manipulation

Guanxing Lu¹, Shiyi Zhang¹, Ziwei Wang^{2,3✉*}, Changliu Liu⁴,
 Jiwen Lu², and Yansong Tang¹

¹ Shenzhen Key Laboratory of Ubiquitous Data Enabling, Shenzhen International Graduate School, Tsinghua University

² Department of Automation, Tsinghua University

³ Nanyang Technological University, ⁴ Carnegie Mellon University
 {lgx23@mails., sy-zhang23@mails., lujiwen@, tang.yansong@sz.}tsinghua.edu.cn
 {ziweiwa20, clius6}@andrew.cmu.edu

Abstract. Performing language-conditioned robotic manipulation tasks in unstructured environments is highly demanded for general intelligent robots. Conventional robotic manipulation methods usually learn a semantic representation of the observation for action prediction, which ignores the scene-level spatiotemporal dynamics for human goal completion. In this paper, we propose a dynamic Gaussian Splatting method named **ManiGaussian** for multi-task robotic manipulation, which mines scene dynamics via future scene reconstruction. Specifically, we first formulate the dynamic Gaussian Splatting framework that infers the semantics propagation in the Gaussian embedding space, where the semantic representation is leveraged to predict the optimal robot action. Then, we build a Gaussian world model to parameterize the distribution in our dynamic Gaussian Splatting framework, which provides informative supervision in the interactive environment via future scene reconstruction. We evaluate our ManiGaussian on 10 RLBench tasks with 166 variations, and the results demonstrate our framework can outperform the state-of-the-art methods by 13.1% in average success rate¹.

Keywords: Multi-task robotic manipulation · Dynamic Gaussian Splatting · World model

1 Introduction

Designing autonomous agents for language-conditioned manipulation tasks [2, 11, 29, 31, 59, 60, 62, 63, 77] has been highly desired in the pursuit of artificial intelligence for a long time. In realistic deployment, intelligent robots are usually required to deal with unseen scenarios in novel tasks. Therefore, comprehending complex 3D structures in the deployment scenes is necessary for the robots to achieve high task success rates across diverse manipulation tasks.

* ✉ Corresponding author.

¹ Project page: <https://guanxinglu.github.io/ManiGaussian/>

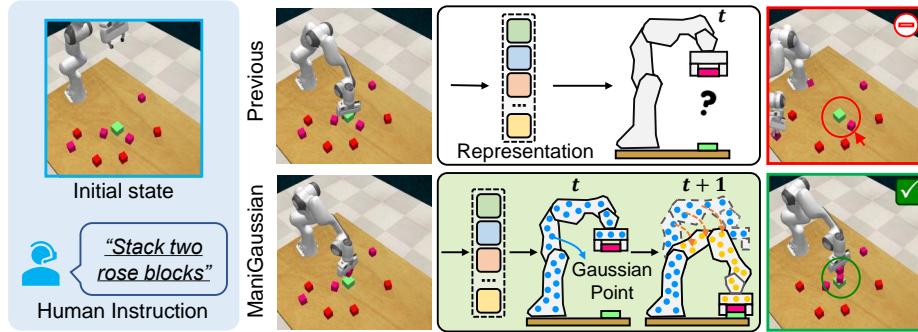


Fig. 1: Consider the human instruction “stack two rose blocks”, where the task is considered successful if two rose blocks are stacked upon the green block. The previous method (GNFactor [76]) attempts to pick up the fixed green base but fails severely due to the misunderstanding of the scene dynamics, while our ManiGaussian completes the task successfully by explicitly encoding the scene dynamics via future scene reconstruction in Gaussian embedding space.

To address the challenges, previous arts have made great progress in general manipulation policy learning, which can be divided into two categories including perceptive methods and generative methods. For the first regard, semantic features extracted by perceptive models are directly leveraged to predict the robot actions according to the visual input such as image [14, 15, 40], point cloud [7, 13, 79] and voxel [28, 60]. However, the perceptive methods heavily rely on multi-view or gripper-mounted cameras to cover the whole workbench to deal with the occlusion problem within unstructured environments, which restricts their deployment. To this end, generative methods [22, 35, 36, 47, 48, 52, 75, 76] capture the 3D scene structure information by reconstructing the scene and objects in arbitrary novel views with self-supervised learning. Nevertheless, they ignore the spatiotemporal dynamics that depict the physical interaction among objects during manipulation, and the predicted actions still fail to complete human goals without correct object interactions. Figure 1 shows a comparison of manipulation achieved by the conventional generative manipulation method (top) and the proposed method (bottom), where the conventional method fails to stack the two rose blocks due to the poor comprehension of scene dynamics.

In this paper, we propose a ManiGaussian method that leverages a dynamic Gaussian Splatting framework for multi-task robotic manipulation. Different from conventional methods which only focus on semantic representation, our method mines the scene-level spatiotemporal dynamics via future scene reconstruction. Therefore, the interaction among objects can be comprehended for accurate manipulation action prediction. More specifically, we first formulate the dynamic Gaussian Splatting framework that models the propagation of diverse semantic features in the Gaussian embedding space, and the semantic features with scene dynamics are leveraged to predict the optimal robot actions

for general manipulation tasks. We build a Gaussian world model to parameterize the distributions in our dynamic Gaussian Splatting framework. Therefore, our framework can acquire informative supervision in interactive environments by reconstructing the future scene according to the current scene and the robot actions, where we constrain consistency between reconstructed and realistic future scenes for dynamics mining. We evaluate our ManiGaussian method on the RLBench dataset [27] with 10 tasks and 166 variants, where our method outperforms the state-of-the-art multi-task robotic manipulation methods by 13.1% in the average task success rate. Our contributions can be summarized as follows:

- We propose a dynamic Gaussian Splatting framework to learn the scene-level spatiotemporal dynamics in general robotic manipulation tasks, so that the robotic agent can complete human instructions with accurate action prediction in unstructured environments.
- We build a Gaussian world model to parameterize distributions in our dynamic Gaussian Splatting framework, which can provide informative supervision to learn scene dynamics from the interactive environment.
- We conduct extensive experiments of 10 tasks on RLBench, and the results demonstrate that our method achieves a higher success rate than the state-of-the-art methods with less computation.

2 Related Work

Visual Representations for Robotic Manipulation. Developing intelligent agents for language-conditioned manipulation tasks in complex and unstructured environments has been a longstanding objective. One of the key bottlenecks in achieving this goal is effectively representing visual information of the scene. Prior arts can be categorized into two branches: perceptive methods and generative methods. Perceptive methods directly utilize pretrained 2D [14, 15, 40, 79] or 3D visual representation backbone [7, 13, 28, 60] to learn scene embedding, where optimal robot actions are predicted based on the scene semantics. For example, InstructRL [40] and Hiveformer [15] directly passed 2D visual tokens through a multi-modal transformer to decode gripper actions, but struggled to handle complex manipulation tasks due to the lack of geometric understanding. To incorporate 3D information beyond images, PolarNet [7] and Act3D [13] utilized point cloud representation, where PolarNet used a PointNeXt [49]-based architecture and Act3D designed a ghost point sampling mechanism to decode actions. Moreover, PerAct [60] fed voxel tokens into a PerceiverIO [26]-based transformer policy, demonstrating impressive performance in a variety of manipulation tasks. However, perceptive methods heavily rely on seamless camera overlay for comprehensive 3D understanding, which makes them less effective in unstructured environments. To address this, generative methods [22, 35, 36, 47, 48, 52, 75, 76] have gained attention, which learns the 3D geometry through self-supervised novel view reconstruction. For instance, Li *et al.* [36] combined NeRF and time contrastive learning to embed 3D geometry and learn fluid dynamics within an

autoencoder framework. GNFactor [76] optimized a generalizable NeRF with a reconstruction loss besides behavior cloning, and showed effective improvement in both simulated and real scenarios. However, conventional generative methods usually ignore the scene-level spatiotemporal dynamics that demonstrate the interaction among objects, and the predicted actions still fail to achieve human goals because of incorrect interaction.

World Models. In recent years, world models have emerged as an effective approach to encode scene dynamics by predicting the future states given the current state and actions, which are explored in autonomous driving [12, 24, 25, 64], game agent [16–20, 54, 73] and robotic manipulation [21, 55, 68]. Early works [16–21, 54, 55, 73] learned a latent space for future prediction by autoencoding, which acquired notable effectiveness in both simulated and real-world situations [68]. However, learning latent for accurate future prediction requires a large amount of data and is limited to simple tasks such as robot control due to the weak representative ability of the implicit features. To address these limitations, explicit representation in the image domain [9, 45, 56, 67] and the language domain [6, 23, 38, 43, 65] has been widely studied because of the rich semantics. UniPi [9] reconstructed the future images with a text-conditional video generation model, employing an inverse dynamics model to obtain the intermediate actions. Dynalang [38] learned to predict text representations as future states, and enabled embodied agents to navigate in photorealistic home scans under human instructions. In contrast to these approaches, we generalize the world model to embedding space of dynamic Gaussian Splatting, which predicts the future state for the agent to learn scene-level dynamics from interactive environments.

Gaussian Splatting. Gaussian Splatting [32] models the scenes with a set of 3D Gaussians which are projected to 2D planes with efficient differentiable splatting. Gaussian Splatting achieves higher effectiveness and efficiency compared with implicit representations such as Neural Radiance Fields (NeRF) [8, 30, 36, 39, 46, 58, 76] with fast inference, high fidelity, and strong editability for novel view synthesis. Please refer to [5] for a comprehensive survey on 3D Gaussian Splatting. To deploy Gaussian Splatting in diverse complex scenarios, many variants have been proposed to enhance the generalization ability, enrich the semantic information, and reconstruct deformable scenes. For higher generalization ability across diverse scenes, recent works [4, 10, 61, 70, 78, 80, 83] constructed a direct mapping from pixels to Gaussian parameters from large-scale datasets. To integrate rich semantic information into Gaussian Splatting, many efforts [50, 57, 81, 84] have been demonstrated in distilling Gaussian radiance fields from pretrained foundation models [3, 34, 51, 53]. For instance, LangSplat [50] advanced the Gaussian representation by encoding language features distilled from CLIP [51] using a scene-wise language autoencoder, enabling efficient open-vocabulary localization compared with its NeRF-based counterpart [33]. For deformation modeling, time-variant Gaussian radiance fields [1, 37, 44, 66, 69, 71, 72] were reconstructed from videos instead of images, which are widely applied in applications such as surgical scene reconstruction [42, 82]. Although these approaches have achieved high-quality reconstruction from entire videos like interpolation, extrapolation

to future states conditioned on previous states and actions is unexplored, which holds significance for scene-level dynamics modeling for interactive agents. In this paper, we formulate a dynamic Gaussian Splatting framework to model the scene dynamics of object interactions, which enhances the physical reasoning for agents to complete a wide range of robotic manipulation tasks.

3 Approach

In this section, we first briefly introduce preliminaries on the problem formulation (Section 3.1), and then we present an overview of our pipeline (Section 3.2). Subsequently, we introduce a dynamic Gaussian Splatting framework (Section 3.3) that infers the propagation semantics of the manipulation scenarios in the Gaussian embedding space. To enable our dynamic Gaussian Splatting framework to learn scene dynamics from the interactive environment, we build a Gaussian world model (Section 3.4) that reconstructs future scenes according to the propagated semantics.

3.1 Problem Formulation

The demand for language-conditioned robotic manipulation is a significant aspect in the development of general intelligent robots. The agent is required to interactively predict the subsequent pose of the robot arm based on the observation and achieve the pose with a low-level motion planner to complete a wide range of manipulation tasks described in humans. The visual input at the t_{th} step for the agent is defined as $o^{(t)} = (\mathbf{C}^{(t)}, \mathbf{D}^{(t)}, \mathbf{P}^{(t)})$, where $\mathbf{C}^{(t)}$ and $\mathbf{D}^{(t)}$ respectively represent the single-view images and the depth images. The proprioception matrix $\mathbf{P}^{(t)} \in \mathbb{R}^4$ indicates the gripper state including the end-effector position, openness, and current timestep. Based on the visual input $o^{(t)}$ and the language instructions, the agent is required to generate the optimal action for the robot arm and grippers $\mathbf{a}^{(t)} = (\mathbf{a}_{\text{trans}}^{(t)}, \mathbf{a}_{\text{rot}}^{(t)}, \mathbf{a}_{\text{open}}^{(t)}, \mathbf{a}_{\text{col}}^{(t)})$, which respectively demonstrates the target translation in voxel $\mathbf{a}_{\text{trans}}^{(t)} \in \mathbb{R}^{100^3}$, rotation $\mathbf{a}_{\text{rot}}^{(t)} \in \mathbb{R}^{(360/5) \times 3}$, openness $\mathbf{a}_{\text{open}}^{(t)} \in [0, 1]$ and collision avoidance $\mathbf{a}_{\text{col}}^{(t)} \in [0, 1]$.

To learn the manipulation policy effectively, expert demonstrations as offline datasets are provided for imitation learning, where the sample triplets contain the visual input, language instruction and expert actions. Existing methods leverage powerful visual representations to learn informative latent features for optimal action prediction. However, they ignore the spatiotemporal dynamics which depicts the physical interaction among objects, and the predicted actions usually fail to complete complex human goals without correct object interactions. On the contrary, we present a dynamic Gaussian Splatting framework to mine the scene dynamics for robotic manipulation.

3.2 Overall Pipeline

The overall pipeline of our ManiGaussian method is shown in Figure 2, in which we construct a dynamic Gaussian Splatting framework that models the prop-

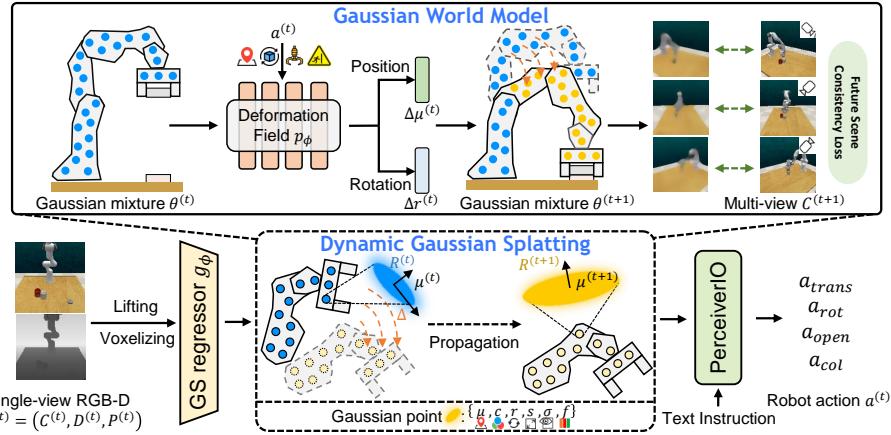


Fig. 2: The overall pipeline of ManiGaussian, which primarily consists of a dynamic Gaussian Splatting framework and a Gaussian world model. The dynamic Gaussian Splatting framework models the propagation of diverse semantic features in the Gaussian embedding space for manipulation, and the Gaussian world model parameterizes distributions to provide supervision by reconstructing the future scene for scene-level dynamics mining.

agation of diverse semantic features in the Gaussian embedding space for manipulation. We also build a Gaussian world model to parameterize distributions in our dynamic Gaussian Splatting framework, which can provide informative supervision of scene dynamics by future scene reconstruction. More specifically, we transform the visual input from RGB-D cameras to a volumetric representation by lifting and voxelization for data preprocessing. For dynamic Gaussian Splatting, we leverage a Gaussian regressor to infer the Gaussian distribution of geometric and semantic features in the scene based on the representation, propagated along time steps with rich scene-level spatiotemporal dynamics. For the Gaussian world model, we instantiate a deformation field to reconstruct the future scene according to the current scene and the robot actions, and require consistency between reconstructed and realistic scenes for dynamics mining. Therefore, the spatiotemporal dynamics indicating object correlation can be embedded into the representation learned in the dynamic Gaussian Splatting framework. Finally, we employ multi-modal transformer PerceiverIO [26] to predict the optimal robot actions for general manipulation tasks, which considers geometric, semantic, and dynamic information with human instructions.

3.3 Dynamic Gaussian Splatting for Robotic Manipulation

In order to capture the scene-level dynamics for general manipulation tasks, we propose a dynamic Gaussian Splatting framework that models the propagation of diverse semantic features within the Gaussian embedding space. While the

vanilla Gaussian Splatting has remarkable effectiveness and efficiency in reconstructing static environments, it fails to capture the scene dynamics for manipulation due to the lack of temporal information. To this end, we formulate a dynamic Gaussian Splatting framework based on the vanilla Gaussian Splatting methodology by enabling the Gaussian points of scene representation to move with robotic manipulation, which demonstrates the physical interactions between objects. The scene representation contains geometric information depicting the explicit visual clues, semantic information illustrating the implicit high-level visual features, and dynamic information encoding the physical properties of the scene, which are utilized to predict the optimal actions.

Dynamic Gaussian Splatting. Gaussian Splatting [32] is a promising approach for multi-view 3D reconstruction, which exhibits fast inference, high fidelity, and strong editability of generated content compared with Neural Radiance Field (NeRF) [46]. Gaussian Splatting represents a 3D scene explicitly with multiple Gaussian primitives, where the i_{th} Gaussian primitive is parameterized by $\theta_i = (\mu_i, c_i, r_i, s_i, \sigma_i)$, where respectively represent the positions, color, rotation, scale, and opacity for the Gaussian primitive. To render a novel view, we project Gaussian primitives onto the 2D plane by differential tile-based rasterization. The value of the pixel \mathbf{p} can be rendered by the alpha-blend rendering:

$$C(\mathbf{p}) = \sum_{i=1}^N \alpha_i c_i \prod_{j=1}^{i-1} (1 - \alpha_j) \quad \text{where, } \alpha_i = \sigma_i e^{-\frac{1}{2}(\mathbf{p} - \mu_i)^\top \Sigma_i^{-1} (\mathbf{p} - \mu_i)}, \quad (1)$$

where C is the rendered image, N denotes the number of Gaussians in this tile, α_i represents the 2D density of the Gaussian points in the splatting process, and Σ_i stands for the covariance matrix acquired from the rotation and scales of the Gaussian parameters. However, the vanilla Gaussian Splatting encounters difficulties in reconstructing changing environments, which limits the ability to model scene-level dynamics that is crucial for manipulation tasks. To address this, we enable the Gaussian particles to be propagated with time to capture the spatiotemporal dynamics of the scene. The parameters of the i_{th} Gaussian primitive at the t_{th} step can be expressed as follows:

$$\theta_i^{(t)} = (\mu_i^{(t)}, c_i^{(t)}, r_i^{(t)}, s_i^{(t)}, \sigma_i^{(t)}, f_i^{(t)}). \quad (2)$$

The positions, colors, rotations, scales, and opacities with the superscript t represent their counterparts at the t_{th} step in the propagation, and $f_i^{(t)}$ is the high-level semantic feature distilled from the Stable Diffusion [53] visual encoder based on the RGB images of the scene. In robotic manipulation, all objects are regarded as rigid bodies without inherent properties including colors, scales, opacities, and semantic features. $c_i^{(t)}, s_i^{(t)}, \sigma_i^{(t)}$ and $f_i^{(t)}$ are therefore regarded as time-independent parameters. The positions and rotation of Gaussian particles change during the manipulation due to the physical interaction between objects and robot grippers, which can be formulated as follows:

$$(\mu_i^{(t+1)}, r_i^{(t+1)}) = (\mu_i^{(t)} + \Delta\mu_i^{(t)}, r_i^{(t)} + \Delta r_i^{(t)}) \quad (3)$$

where $\Delta\mu_i^{(t)}$ and $\Delta r_i^{(t)}$ demonstrate the change of positions and rotation from the t_{th} step to the next one for the i_{th} Gaussian primitive. With the time-dependent parameters of the Gaussian mixture distribution, the pixel values in 2D views of the scene can still be rendered by (1).

Gaussian World Model. In our implementation, we present a Gaussian world model to parameterize the Gaussian mixture distribution in dynamic Gaussian Splatting, through which the future scene can be reconstructed via parameter propagation. Therefore, the dynamic Gaussian Splatting model can acquire informative supervision in the interactive environment by considering the consistency between the reconstructed and realistic feature scenes. World models are effective to learn the environmental dynamics for downstream tasks by anticipating the future state $s^{(t+1)}$ based on the current state $s^{(t)}$ and action $a^{(t)}$ at the t_{th} step, which have been applied to a variety of tasks including autonomous driving [12, 24, 25, 64] and game agent [16–20, 54, 73]. For our robotic manipulation tasks, we instantiate the current state in the world model as the visual observation in the current step, and actions refer to those of the robot arm and grippers. They are leveraged to predict the visual scenes observed in the next step that represent the future state. More specifically, the Gaussian world model contains a representation network q_ϕ that learns high-level visual features with rich semantics for the input observation, a Gaussian regressor g_ϕ that predicts the Gaussian parameters of different primitives based on the visual features, a deformation predictor p_ϕ that infers the difference of Gaussian parameters during the propagation, and a Gaussian renderer \mathcal{R} (1) that generates the RGB images for the predicted future state:

$$\begin{cases} \text{Representation model: } & \mathbf{v}^{(t)} = q_\phi(o^{(t)}), \\ \text{Gaussian regressor: } & \theta^{(t)} = g_\phi(\mathbf{v}^{(t)}), \\ \text{Deformation predictor: } & \Delta\theta^{(t)} = p_\phi(\theta^{(t)}, a^{(t)}), \\ \text{Gaussian renderer: } & o^{(t+1)} = \mathcal{R}(\theta^{(t+1)}, w), \end{cases} \quad (4)$$

where $o^{(t)}$ and $\mathbf{v}^{(t)}$ mean the visual observation and the corresponding high-level visual features at the t_{th} step. w is the camera pose for the view where we project the Gaussian primitives. We leverage multi-head neural networks as the Gaussian regressor, where each head predicts a specific feature for the Gaussian parameters shown in (2). By inferring the changes of positions and rotations between consecutive steps, we acquire the propagated Gaussian parameters in the future step based on (3). Finally, the Gaussian renderer projects the propagated Gaussian distribution in a specific view for future scene reconstruction.

3.4 Learning Objectives

Current Scene Consistency Loss. Reconstructing the current scene based on the current Gaussian parameters accurately can enhance the performance of the Gaussian regressor. To achieve this goal, we introduce the consistency objective between the realistic current observation and the rendered according

to the current Gaussian parameters:

$$\mathcal{L}_{\text{Geo}} = \|\mathbf{C}^{(t)} - \hat{\mathbf{C}}^{(t)}\|_2^2, \quad (5)$$

where $\mathbf{C}^{(t)}$ and $\hat{\mathbf{C}}^{(t)}$ respectively mean the groundtruth and prediction of observation images from different views at the t_{th} step.

Semantic Feature Consistency Loss. The semantic features contain high-level visual information of the observed scenes. Since the foundation models can extract informative semantic features for general scenes, we expect the semantic features in our Gaussian parameters to mimic those acquired by large pre-trained models such as Stable Diffusion [53], so that the knowledge learned by the pre-trained models can be distilled to our Gaussian world model according to the following objective:

$$\mathcal{L}_{\text{Sem}} = 1 - \sigma_{\text{cos}}(\mathbf{F}^{(t)}, \hat{\mathbf{F}}^{(t)}), \quad (6)$$

where $\mathbf{F}^{(t)}$ and $\hat{\mathbf{F}}^{(t)}$ are projected map of semantic features in Gaussian parameters and the feature map learned by pre-trained models. σ_{cos} means the cosine distance between variables.

Action Prediction Loss. The distribution parameters in our dynamic Gaussian framework are leveraged to predict the optimal action of the robot arm and grippers for general manipulation tasks. We employ a multi-modal transformer PerceiverIO [26] to infer the selection probability of different action candidates based on the Gaussian parameters and the human language instructions, and leverage the cross-entropy loss CE for accurate action prediction:

$$\mathcal{L}_{\text{Act}} = CE(p_{\text{trans}}, p_{\text{rot}}, p_{\text{open}}, p_{\text{col}}), \quad (7)$$

where p_{trans} , p_{rot} , p_{open} , p_{col} represent the probability of the groundtruth actions in expert demonstrations for translation, rotation, gripper openness and collision avoidance of the robot, respectively.

Future Scene Consistency Loss. We require consistency between the reconstructed and realistic scenes, so that the dynamic Gaussian Splatting framework can accurately embed scene-level spatiotemporal dynamics in the Gaussian parameters. Specifically, the training objective aligns the predicted future scenes based on different observations and actions with the realistic ones, which can be formulated as follows:

$$\mathcal{L}_{\text{Dyna}} = \|\hat{\mathbf{C}}^{(t+1)}(a^{(t)}, o^{(t)}) - \mathbf{C}^{(t+1)}\|_2^2, \quad (8)$$

where $\hat{\mathbf{C}}^{(t+1)}(a^{(t)}, o^{(t)})$ means the predicted future image of the scene at the t_{th} step based on the action $a^{(t)}$ and current observation $o^{(t)}$, and $\mathbf{C}^{(t+1)}$ is the realistic counterpart. By imposing the Gaussian world model to predict future scenes based on the representation, the representation is required to encode the physical properties of the scene. This is important for the action decoder to predict effective actions with such representation.

The overall objective for our ManiGaussian agent is written as a weighted combination of different loss terms:

$$\mathcal{L} = \mathcal{L}_{\text{Act}} + \lambda_{\text{Geo}} \mathcal{L}_{\text{Geo}} + \lambda_{\text{Sem}} \mathcal{L}_{\text{Sem}} + \lambda_{\text{Dyna}} \mathcal{L}_{\text{Dyna}}, \quad (9)$$

where λ_{Geo} , λ_{Sem} , λ_{Dyna} are the hyperparameters that control the importance of different terms during training. In training, we set a warm-up phase that freezes the deformation predictor to learn a stable representation model and a Gaussian regressor during the first 3k iterations. After the warm-up phase, we jointly train the whole Gaussian world model with the action decoder.

4 Experiments

In this section, we first introduce the experiment setup including datasets, baseline methods, and implementation details (Section 4.1). Then we compare our method with the state-of-the-art approaches to show the superiority in success rate (Section 4.2), and conduct an ablation study to verify the effectiveness of different components in our dynamic Gaussian Splatting framework and the Gaussian world model (Section 4.3). Finally, we also illustrate the visualization results to depict our intuition (Section 4.4). Further results and case studies can be found in the supplementary material.

4.1 Experiment Setup

Simulation. Our experiments are conducted in the popular RLBench [27] simulated tasks. Following [76], we utilize a curated subset of 10 challenging language-conditioned manipulation tasks from RLBench, which includes 166 variations in object properties and scene arrangement. The diversity of these tasks requires the agent to acquire generalizable knowledge about the intrinsical scene-level spatial-temporal dynamics for manipulation, rather than solely relying on mimicking the provided expert demonstrations to achieve high success rates. We evaluated 25 episodes in the testing set for each task to avoid result bias from noise. For visual observation, we employ RGB-D images captured by a single front camera with a resolution of 128×128 . We use the same number of cameras (*i.e.*, 20) as GNFactor to provide multi-view supervision for fair comparisons. During the training phase, we use 20 demonstrations for each task.

Baselines: We compare our ManiGaussian with the previous state of the arts including the perceptive method PerAct [60] and its modified version using 4 camera inputs to cover the workbench, as well as the generative method GNFactor [76]. The evaluation metric is the task success rate, which measures the percentage of completed episodes. An episode is considered successful if the agent completes the goal specified in natural language within a maximum of 25 steps.

Implementation Details. We use the SE(3) [60, 76] augmentation for the expert demonstrations in the training set to enhance the generalizability of agents. To mitigate the impact of parameter size, we utilize the same version of Perceive-IO [26] as the action decoder across all baselines. All the compared methods are trained on two NVIDIA RTX 4090 GPUs for 100k iterations with a batch size of 2. We employ LAMB optimizer [74] with an initial learning rate 5×10^{-4} . We also adopt a cosine scheduler with warmup in the first 3k steps.

Table 1: Multi-task Test Results. We evaluate 25 episodes per task for the final checkpoint on 10 challenging tasks from RLBench and report the success rates (%), where the second results are underlined and the best results are bold.

Method / Task	close jar	open drawer	sweep to dustpan	meat off grill	turn tap
PerAct	18.7	<u>54.7</u>	0.0	40.0	38.7
PerAct (4 cameras)	21.3	44.0	0.0	65.3	46.7
GNFactor	<u>25.3</u>	76.0	<u>28.0</u>	57.3	<u>50.7</u>
ManiGaussian (ours)	28.0	76.0	64.0	60.0	56.0

Method / Task	slide block	put in drawer	drag stick	push buttons	stack blocks	Average
PerAct	18.7	2.7	5.3	<u>18.7</u>	<u>6.7</u>	20.4
PerAct (4 cameras)	16.0	<u>6.7</u>	12.0	9.3	5.3	22.7
GNFactor	<u>20.0</u>	0.0	<u>37.3</u>	<u>18.7</u>	4.0	<u>31.7</u>
ManiGaussian (ours)	24.0	16.0	92.0	20.0	12.0	44.8

Table 2: Comparison of Methods with Different Techniques. Following [15], we manually group the 10 RLBench tasks into 6 categories according to their main challenges to demonstrate the improvement reason. The 6 categories are detailed in the supplementary file.

Geo.	Sem.	Dyna.	Planning	Long	Tools	Motion	Screw	Occlusion	Average
✗	✗	✗	36.0	2.0	25.3	52.0	4.0	28.0	23.6
✓	✗	✗	46.0	4.0	52.0	52.0	<u>24.0</u>	60.0	39.2
✓	✓	✗	46.0	8.0	<u>53.3</u>	64.0	28.0	56.0	41.6
✓	✗	✓	54.0	<u>10.0</u>	49.3	64.0	<u>24.0</u>	<u>72.0</u>	<u>43.6</u>
✓	✓	✓	40.0	14.0	60.0	<u>56.0</u>	28.0	76.0	44.8

4.2 Comparison with the State-of-the-Art Methods

In this section, we compare our ManiGaussian with previous state-of-the-art methods on the RLBench tasksuite. Table 1 illustrates the comparison of the average success rate of each task. Our method achieves the best performance with an average success rate of 44.8%, which is state-of-the-art, outperforming the previous arts including both perceptive and generative-based methods by a sizable margin. The dominated generative-based method GNFactor leveraged a generalizable NeRF to learn informative latent representation for optimal action prediction, which showed effective improvement beyond the perceptive-based method PerAct. However, it ignores the scene-level spatiotemporal dynamics that demonstrate the interaction among objects, and the predicted actions still fail to achieve human goals because of the incorrect interaction. On the contrary, our ManiGaussian learns the scene dynamics with the proposed dynamic Gaussian Splatting framework, so that the robotic agent can complete human instructions with accurate action prediction in unstructured environments. As a result, our method outperforms the second-best GNFactor method by a relative improvement of 41.3%. In the task **meat off grill** where the best performance was not reached, our method also ranks as second best. The experimental results

illustrate the effectiveness of our proposed method across multiple language-conditioned robotic manipulation tasks.

4.3 Ablation Study

Our dynamic Gaussian Splatting framework models the propagation of diverse features in the Gaussian embedding space, and the Gaussian world model reconstructs the future scene according to the current scene by constraining the consistency between reconstructed and realistic scenes for dynamics mining. We conduct an ablation study to verify the effectiveness of each presented component in Table 2. We first implement a vanilla baseline without any proposed technique, where we directly train the representation model and the action decoder to predict the robot actions. By adding the Gaussian regressor to predict the Gaussian parameters, the performance improves by 15.6% compared with the baseline. Especially, in the tasks that require geometric reasoning such as Occlusion, Tools and Screw, it outperforms the vanilla version by sizable margins, which proves the ability of the Gaussian Splatting technology to model the spatial information for manipulation tasks. We then add semantic features distilled from the pretrained foundation model into the dynamic Gaussian Splatting framework. By adding the semantic features and the related consistency loss, we observe that the average success rate increases by 2.4% than the only geometric features version, which indicates the benefits of the high-level semantic information for robotic manipulation. Besides, we implement the deformation predictor and the corresponding future scene consistency loss, resulting in a dramatic performance improvement of 4.4%. Particularly, the proposed deformation predictor improves the task completion of 4 out of 6 task types, which demonstrates the importance of the scene-level dynamics encoded by the deformation predictor in the Gaussian world model, especially in long-horizon tasks (Long). Though the dynamic loss may slightly impact short-term results due to the balance of different loss items, it significantly improves overall performance. After combining all the techniques in our dynamic Gaussian Splatting framework, the performance increases from 23.6% to 44.8%, which verifies the necessity of the scene-level spatiotemporal dynamics mined by the proposed dynamics Gaussian Splatting framework with the Gaussian world model.

Figure 3 depicts the learning curve of the proposed ManiGaussian and the state-of-the-art method GNFactor, where we save checkpoints and test them

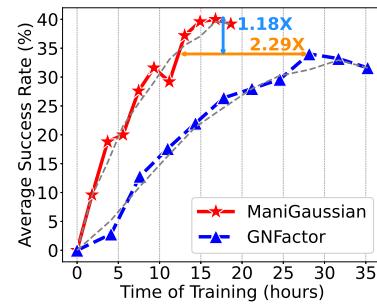


Fig. 3: Learning Curve. Comparison of our ManiGaussian with GNFactor in performance and speed. For a fair comparison, we exclude auxiliary losses from the reconstruction loss. The grey dotted lines represent the results using a moving average.

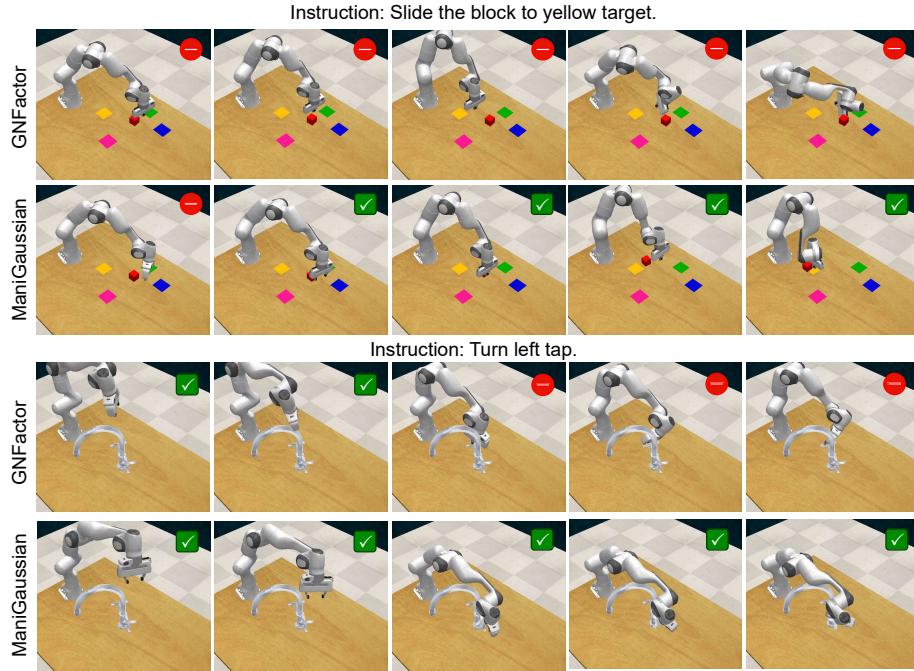


Fig. 4: Case Study. The red mark signifies the pose deviates severely from the expert demonstration, whereas the green mark indicates that the pose aligns with the expert trajectory. Our ManiGaussian can successfully complete the human goal with the physical understanding of scene-level spatial-temporal dynamics.

every 10k parameter updates. Both the compared methods get convergence within 100k training steps. As shown in Figure 3, our ManiGaussian outperforms the state-of-the-art method GNFactor, achieving $1.18\times$ better performance and $2.29\times$ faster training. This result proves that our ManiGaussian not only performs better but also trains faster, which also shows the efficiency of the explicit Gaussian scene reconstruction than the implicit approach like NeRF.

4.4 Qualitative Analysis

Visualization of Whole Trajectories. We present two qualitative examples of the generated action sequence in Figure 4 from GNFactor and our ManiGaussian. In the top case, the agent is instructed to “*Slide the block to yellow target*”. The results show that the previous agent struggles to complete the task since it imitates the expert’s backward pulling motion, even though the claw is already leaning towards the right side of the red block. In contrast, ManiGaussian returns to the red square and successfully slides the square to the yellow target, owing to that our method can correctly understand the scene dynamics of objects in contact. In the bottom case, the agent is instructed to “*Turn left tap*”. The results

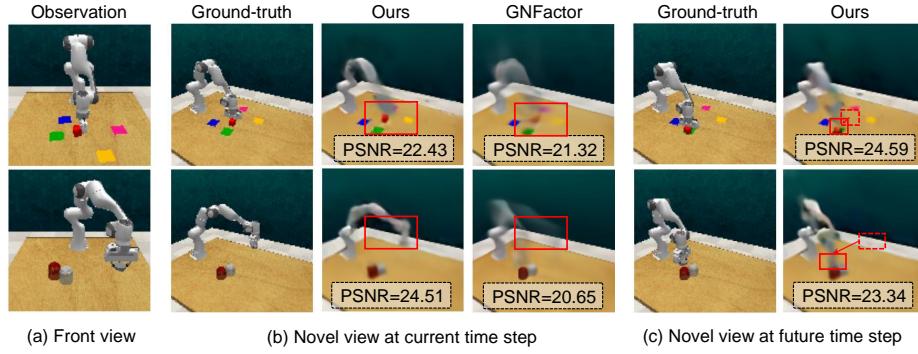


Fig. 5: Novel View Synthesis Results. We remove the action loss here for better visualization. Our ManiGaussian is capable of both current scene reconstruction and future scene prediction.

show that GNFactor misunderstands the meaning of “left”, and instead operates the right tap, and also fails to turn on the tap. In contrast, our ManiGaussian successfully completes the task, which shows that ManiGaussian can not only understand the semantic information, but also execute operations accurately.

Visualization of Novel View Synthesis. Figure 5 shows the novel view image synthesis results. First, based on the front view observation where the gripper shape cannot be seen, our ManiGaussian offers superior detail in modeling cubes in novel views. Second, our method accurately predicts future states based on the recovered details. For example, in the top case of the `slide block` task, our ManiGaussian not only predicts the future gripper position that corresponds to the human instruction, but also predicts the future cube location influenced by the gripper based on the understanding of the physical interaction among objects. This qualitative result demonstrates that our ManiGaussian learns the intricate scene-level dynamics successfully.

5 Conclusion

In this paper, we have presented a ManiGaussian agent that encodes the scene-level spatiotemporal dynamics for language-conditioned manipulation agents. We design a dynamic Gaussian Splatting framework that models the propagation of features in the Gaussian embedding space, and the latent representation with scene dynamics is leveraged to predict the robot actions. Subsequently, we build a Gaussian world model to parameterize the distributions in the dynamic Gaussian Splatting framework to mine scene-level dynamics by reconstructing the future scene. Experiments in diverse manipulation tasks demonstrate the superiority of ManiGaussian. The limitations stem from the necessity of multiple view supervision with camera calibration for the Gaussian Splatting framework.

Acknowledgements

This work was supported in part by the National Key Research and Development Program of China under Grant 2022ZD0114903 and Shenzhen Ubiquitous Data Enabling Key Lab under grant ZDSYS20220527171406015. We would like to greatly thank Yanjie Ze for his kind response to the GNFactor issues.

References

1. Abou-Chakra, J., Rana, K., Dayoub, F., Sünderhauf, N.: Physically embodied gaussian splatting: Embedding physical priors into a visual 3d world model for robotics. In: CoRL (2023)
2. Brohan, A., Brown, N., Carbajal, J., Chebotar, Y., Dabis, J., Finn, C., Gopalakrishnan, K., Hausman, K., Herzog, A., Hsu, J., et al.: Rt-1: Robotics transformer for real-world control at scale. arXiv (2022)
3. Caron, M., Touvron, H., Misra, I., Jégou, H., Mairal, J., Bojanowski, P., Joulin, A.: Emerging properties in self-supervised vision transformers. In: ICCV (2021)
4. Charatan, D., Li, S., Tagliasacchi, A., Sitzmann, V.: pixelsplat: 3d gaussian splats from image pairs for scalable generalizable 3d reconstruction. arXiv preprint arXiv:2312.12337 (2023)
5. Chen, G., Wang, W.: A survey on 3d gaussian splatting. arXiv preprint arXiv:2401.03890 (2024)
6. Chen, J., Jiang, Y., Lu, J., Zhang, L.: S-agents: self-organizing agents in open-ended environment. arXiv preprint arXiv:2402.04578 (2024)
7. Chen, S., Garcia, R., Schmid, C., Laptev, I.: Polarnet: 3d point clouds for language-guided robotic manipulation. arXiv preprint arXiv:2309.15596 (2023)
8. Driess, D., Schubert, I., Florence, P., Li, Y., Toussaint, M.: Reinforcement learning with neural radiance fields. NeurIPS (2022)
9. Du, Y., Yang, S., Dai, B., Dai, H., Nachum, O., Tenenbaum, J., Schuurmans, D., Abbeel, P.: Learning universal policies via text-guided video generation. NeurIPS **36** (2024)
10. Fu, Y., Liu, S., Kulkarni, A., Kautz, J., Efros, A.A., Wang, X.: Colmap-free 3d gaussian splatting. arXiv preprint arXiv:2312.07504 (2023)
11. Fu, Z., Zhao, T.Z., Finn, C.: Mobile aloha: Learning bimanual mobile manipulation with low-cost whole-body teleoperation. arXiv preprint arXiv:2401.02117 (2024)
12. Gao, Z., Mu, Y., Shen, R., Chen, C., Ren, Y., Chen, J., Li, S.E., Luo, P., Lu, Y.: Enhance sample efficiency and robustness of end-to-end urban autonomous driving via semantic masked world model. arXiv preprint arXiv:2210.04017 (2022)
13. Gervet, T., Xian, Z., Gkanatsios, N., Fragkiadaki, K.: Act3d: 3d feature field transformers for multi-task robotic manipulation. In: CoRL. pp. 3949–3965 (2023)
14. Goyal, A., Xu, J., Guo, Y., Blukis, V., Chao, Y.W., Fox, D.: Rvt: Robotic view transformer for 3d object manipulation. arXiv preprint arXiv:2306.14896 (2023)
15. Guhur, P.L., Chen, S., Pinel, R.G., Tapaswi, M., Laptev, I., Schmid, C.: Instruction-driven history-aware policies for robotic manipulations. In: CoRL. pp. 175–187. PMLR (2023)
16. Ha, D., Schmidhuber, J.: Recurrent world models facilitate policy evolution. NeurIPS **31** (2018)
17. Hafner, D., Lee, K.H., Fischer, I., Abbeel, P.: Deep hierarchical planning from pixels. NeurIPS **35**, 26091–26104 (2022)

18. Hafner, D., Lillicrap, T., Ba, J., Norouzi, M.: Dream to control: Learning behaviors by latent imagination. arXiv preprint arXiv:1912.01603 (2019)
19. Hafner, D., Lillicrap, T., Norouzi, M., Ba, J.: Mastering atari with discrete world models. arXiv preprint arXiv:2010.02193 (2020)
20. Hafner, D., Pasukonis, J., Ba, J., Lillicrap, T.: Mastering diverse domains through world models. arXiv preprint arXiv:2301.04104 (2023)
21. Hansen, N., Su, H., Wang, X.: Td-mpc2: Scalable, robust world models for continuous control. arXiv preprint arXiv:2310.16828 (2023)
22. Hansen, N., Wang, X.: Generalization in reinforcement learning by soft data augmentation. In: ICRA (2021)
23. Hong, S., Lin, Y., Liu, B., Wu, B., Li, D., Chen, J., Zhang, J., Wang, J., Zhang, L., Zhuge, M., et al.: Data interpreter: An llm agent for data science. arXiv preprint arXiv:2402.18679 (2024)
24. Hu, A., Corrado, G., Griffiths, N., Murez, Z., Gurau, C., Yeo, H., Kendall, A., Cipolla, R., Shotton, J.: Model-based imitation learning for urban driving. NeurIPS **35**, 20703–20716 (2022)
25. Hu, A., Russell, L., Yeo, H., Murez, Z., Fedoseev, G., Kendall, A., Shotton, J., Corrado, G.: Gaia-1: A generative world model for autonomous driving. arXiv preprint arXiv:2309.17080 (2023)
26. Jaegle, A., Gimeno, F., Brock, A., Vinyals, O., Zisserman, A., Carreira, J.: Perceiver: General perception with iterative attention. In: ICML (2021)
27. James, S., Ma, Z., Arrojo, D.R., Davison, A.J.: Rlbench: The robot learning benchmark & learning environment. RA-L (2020)
28. James, S., Wada, K., Laidlow, T., Davison, A.J.: Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In: CVPR (2022)
29. Jang, E., Irpan, A., Khansari, M., Kappler, D., Ebert, F., Lynch, C., Levine, S., Finn, C.: Bc-z: Zero-shot task generalization with robotic imitation learning. In: CoRL (2022)
30. Jiang, Z., Zhu, Y., Svetlik, M., Fang, K., Zhu, Y.: Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. arXiv preprint arXiv:2104.01542 (2021)
31. Kalashnikov, D., Irpan, A., Pastor, P., Ibarz, J., Herzog, A., Jang, E., Quillen, D., Holly, E., Kalakrishnan, M., Vanhoucke, V., et al.: Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv (2018)
32. Kerbl, B., Kopanas, G., Leimkuhler, T., Drettakis, G.: 3d gaussian splatting for real-time radiance field rendering. TOG **42**(4) (2023)
33. Kerr, J., Kim, C.M., Goldberg, K., Kanazawa, A., Tancik, M.: Lerf: Language embedded radiance fields. arXiv preprint arXiv:2303.09553 (2023)
34. Kirillov, A., Mintun, E., Ravi, N., Mao, H., Rolland, C., Gustafson, L., Xiao, T., Whitehead, S., Berg, A.C., Lo, W.Y., et al.: Segment anything. In: ICCV. pp. 4015–4026 (2023)
35. Laskin, M., Srinivas, A., Abbeel, P.: Curl: Contrastive unsupervised representations for reinforcement learning. In: ICML (2020)
36. Li, Y., Li, S., Sitzmann, V., Agrawal, P., Torralba, A.: 3d neural scene representations for visuomotor control. In: CorL. pp. 112–123 (2022)
37. Liang, L., Bian, L., Xiao, C., Zhang, J., Chen, L., Liu, I., Xiang, F., Huang, Z., Su, H.: Robo360: A 3d omnispective multi-material robotic manipulation dataset. arXiv preprint arXiv:2312.06686 (2023)
38. Lin, J., Du, Y., Watkins, O., Hafner, D., Abbeel, P., Klein, D., Dragan, A.: Learning to model the world with language. arXiv preprint arXiv:2308.01399 (2023)

39. Lin, Y.C., Florence, P., Zeng, A., Barron, J.T., Du, Y., Ma, W.C., Simeonov, A., Garcia, A.R., Isola, P.: Mira: Mental imagery for robotic affordances. In: CoRL. pp. 1916–1927 (2023)
40. Liu, H., Lee, L., Lee, K., Abbeel, P.: Instruction-following agents with jointly pre-trained vision-language models. arXiv preprint arXiv:2210.13431 (2022)
41. Liu, S., James, S., Davison, A.J., Johns, E.: Auto-lambda: Disentangling dynamic task relationships. arXiv preprint arXiv:2202.03091 (2022)
42. Liu, Y., Li, C., Yang, C., Yuan, Y.: Endogaussian: Gaussian splatting for deformable surgical scene reconstruction. arXiv preprint arXiv:2401.12561 (2024)
43. Lu, G., Wang, Z., Liu, C., Lu, J., Tang, Y.: Thinkbot: Embodied instruction following with thought chain reasoning. arXiv preprint arXiv:2312.07062 (2023)
44. Luiten, J., Kopanas, G., Leibe, B., Ramanan, D.: Dynamic 3d gaussians: Tracking by persistent dynamic view synthesis. arXiv preprint arXiv:2308.09713 (2023)
45. Mendonca, R., Bahl, S., Pathak, D.: Structured world models from human videos. arXiv preprint arXiv:2308.10901 (2023)
46. Mildenhall, B., Srinivasan, P.P., Tancik, M., Barron, J.T., Ramamoorthi, R., Ng, R.: Nerf: Representing scenes as neural radiance fields for view synthesis. CACM **65**(1), 99–106 (2021)
47. Nair, S., Rajeswaran, A., Kumar, V., Finn, C., Gupta, A.: R3m: A universal visual representation for robot manipulation. arXiv (2022)
48. Parisi, S., Rajeswaran, A., Purushwalkam, S., Gupta, A.: The unsurprising effectiveness of pre-trained vision models for control. In: ICML (2022)
49. Qian, G., Li, Y., Peng, H., Mai, J., Hammoud, H., Elhoseiny, M., Ghanem, B.: Pointnext: Revisiting pointnet++ with improved training and scaling strategies. NeurIPS **35**, 23192–23204 (2022)
50. Qin, M., Li, W., Zhou, J., Wang, H., Pfister, H.: Langsplat: 3d language gaussian splatting. arXiv preprint arXiv:2312.16084 (2023)
51. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al.: Learning transferable visual models from natural language supervision. In: ICML (2021)
52. Radosavovic, I., Xiao, T., James, S., Abbeel, P., Malik, J., Darrell, T.: Real-world robot learning with masked visual pre-training. In: CoRL (2023)
53. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B.: High-resolution image synthesis with latent diffusion models. In: CVPR (2022)
54. Schrittwieser, J., Antonoglou, I., Hubert, T., Simonyan, K., Sifre, L., Schmitt, S., Guez, A., Lockhart, E., Hassabis, D., Graepel, T., et al.: Mastering atari, go, chess and shogi by planning with a learned model. Nature **588**(7839), 604–609 (2020)
55. Seo, Y., Hafner, D., Liu, H., Liu, F., James, S., Lee, K., Abbeel, P.: Masked world models for visual control. In: CoRL. pp. 1332–1344 (2023)
56. Seo, Y., Lee, K., James, S.L., Abbeel, P.: Reinforcement learning with action-free pre-training from videos. In: ICML. pp. 19561–19579 (2022)
57. Shi, J.C., Wang, M., Duan, H.B., Guan, S.H.: Language embedded 3d gaussians for open-vocabulary scene understanding. arXiv preprint arXiv:2311.18482 (2023)
58. Shim, D., Lee, S., Kim, H.J.: Snerl: Semantic-aware neural radiance fields for reinforcement learning. ICML (2023)
59. Shridhar, M., Manuelli, L., Fox, D.: Cliprot: What and where pathways for robotic manipulation. In: CoRL (2022)
60. Shridhar, M., Manuelli, L., Fox, D.: Perceiver-actor: A multi-task transformer for robotic manipulation. In: CoRL (2023)
61. Szymanicz, S., Rupprecht, C., Vedaldi, A.: Splatter image: Ultra-fast single-view 3d reconstruction. arXiv preprint arXiv:2312.13150 (2023)

62. Tellez, S., Kollar, T., Dickerson, S., Walter, M., Banerjee, A., Teller, S., Roy, N.: Understanding natural language commands for robotic navigation and mobile manipulation. In: AAAI (2011)
63. Tenorth, M., Nyga, D., Beetz, M.: Understanding and executing instructions for everyday manipulation tasks from the world wide web. In: ICRA. pp. 1486–1491. IEEE (2010)
64. Wang, X., Zhu, Z., Huang, G., Chen, X., Lu, J.: Drivedreamer: Towards real-world-driven world models for autonomous driving. arXiv preprint arXiv:2309.09777 (2023)
65. Wang, Z., Cai, S., Chen, G., Liu, A., Ma, X., Liang, Y.: Describe, explain, plan and select: Interactive planning with large language models enables open-world multi-task agents. arXiv preprint arXiv:2302.01560 (2023)
66. Wu, G., Yi, T., Fang, J., Xie, L., Zhang, X., Wei, W., Liu, W., Tian, Q., Wang, X.: 4d gaussian splatting for real-time dynamic scene rendering. arXiv preprint arXiv:2310.08528 (2023)
67. Wu, J., Ma, H., Deng, C., Long, M.: Pre-training contextualized world models with in-the-wild videos for reinforcement learning. NeurIPS **36** (2024)
68. Wu, P., Escontrela, A., Hafner, D., Abbeel, P., Goldberg, K.: Daydreamer: World models for physical robot learning. In: CoRL. pp. 2226–2240 (2023)
69. Xie, T., Zong, Z., Qiu, Y., Li, X., Feng, Y., Yang, Y., Jiang, C.: Physgaussian: Physics-integrated 3d gaussians for generative dynamics. arXiv preprint arXiv:2311.12198 (2023)
70. Xu, D., Yuan, Y., Mardani, M., Liu, S., Song, J., Wang, Z., Vahdat, A.: Agg: Amortized generative 3d gaussians for single image to 3d. arXiv preprint arXiv:2401.04099 (2024)
71. Yang, Z., Yang, H., Pan, Z., Zhu, X., Zhang, L.: Real-time photorealistic dynamic scene representation and rendering with 4d gaussian splatting. arXiv preprint arXiv:2310.10642 (2023)
72. Yang, Z., Gao, X., Zhou, W., Jiao, S., Zhang, Y., Jin, X.: Deformable 3d gaussians for high-fidelity monocular dynamic scene reconstruction. arXiv preprint arXiv:2309.13101 (2023)
73. Ye, W., Liu, S., Kurutach, T., Abbeel, P., Gao, Y.: Mastering atari games with limited data. NeurIPS **34**, 25476–25488 (2021)
74. You, Y., Li, J., Reddi, S., Hseu, J., Kumar, S., Bhojanapalli, S., Song, X., Demmel, J., Keutzer, K., Hsieh, C.J.: Large batch optimization for deep learning: Training bert in 76 minutes. arXiv (2019)
75. Ze, Y., Hansen, N., Chen, Y., Jain, M., Wang, X.: Visual reinforcement learning with self-supervised 3d representations. RA-L (2023)
76. Ze, Y., Yan, G., Wu, Y.H., Macaluso, A., Ge, Y., Ye, J., Hansen, N., Li, L.E., Wang, X.: Gnfactor: Multi-task real robot learning with generalizable neural feature fields. In: CoRL. pp. 284–301. PMLR (2023)
77. Zeng, A., Florence, P., Tompson, J., Welker, S., Chien, J., Attarian, M., Armstrong, T., Krasin, I., Duong, D., Sindhwani, V., et al.: Transporter networks: Rearranging the visual world for robotic manipulation. In: CoRL (2021)
78. Zhang, B., Cheng, Y., Yang, J., Wang, C., Zhao, F., Tang, Y., Chen, D., Guo, B.: Gaussiancube: Structuring gaussian splatting using optimal transport for 3d generative modeling. arXiv preprint arXiv:2403.19655 (2024)
79. Zhang, T., Hu, Y., Cui, H., Zhao, H., Gao, Y.: A universal semantic-geometric representation for robotic manipulation. arXiv preprint arXiv:2306.10474 (2023)

80. Zheng, S., Zhou, B., Shao, R., Liu, B., Zhang, S., Nie, L., Liu, Y.: Gps-gaussian: Generalizable pixel-wise 3d gaussian splatting for real-time human novel view synthesis. arXiv preprint arXiv:2312.02155 (2023)
81. Zhou, S., Chang, H., Jiang, S., Fan, Z., Zhu, Z., Xu, D., Chari, P., You, S., Wang, Z., Kadambi, A.: Feature 3dgs: Supercharging 3d gaussian splatting to enable distilled feature fields. In: CVPR. pp. 21676–21685 (2024)
82. Zhu, L., Wang, Z., Jin, Z., Lin, G., Yu, L.: Deformable endoscopic tissues reconstruction with gaussian splatting. arXiv preprint arXiv:2401.11535 (2024)
83. Zou, Z.X., Yu, Z., Guo, Y.C., Li, Y., Liang, D., Cao, Y.P., Zhang, S.H.: Triplane meets gaussian splatting: Fast and generalizable single-view 3d reconstruction with transformers. arXiv preprint arXiv:2312.09147 (2023)
84. Zuo, X., Samangouei, P., Zhou, Y., Di, Y., Li, M.: Fmgs: Foundation model embedded 3d gaussian splatting for holistic 3d scene understanding. arXiv preprint arXiv:2401.01970 (2024)

ManiGaussian: Dynamic Gaussian Splatting for Multi-task Robotic Manipulation

Supplementary Material

In this supplementary material, we provide additional details and experiments not included in the main paper due to limitations in space.

- Appendix A: Details of the RLBench dataset and the training pipeline used in our experiments.
- Appendix B: Additional implementation details of our ManiGaussian.
- Appendix C: Supplementary quantitative analysis.
- Appendix D: Supplementary qualitative analysis.

A Details of RLBench

RLBench Dataset. In this section, we provide a concise overview of the RLBench [27] dataset and our training pipeline. Table 3 is an overview of the 10 selected tasks we use in the experiments. Our task variations include randomly sampled colors, sizes, counts, placements, and categories of objects. We have a color palette of 20 shades, including red, maroon, lime, green, blue, navy, yellow, cyan, magenta, silver, gray, orange, olive, purple, teal, azure, violet, rose, black, and white. The size of the objects is categorized into two types: short and tall. The number of objects can be either 1, 2, or 3. Other properties vary depending on the specific task. Furthermore, objects are randomly arranged on the tabletop within a certain range, adding to the diversity of the tasks. In the ablation study, we adopt the task classification from [15] to group the RLBench tasks of Table 3 into 6 categories according to their key challenges. The task groups include:

- The **Planning** group contains tasks with multiple subtasks. The included tasks are: `meat off grill` and `push buttons`.
- The **Long** group includes long-term tasks that require more than 10 keyframes. The included tasks are: `put in drawer` and `stack blocks`.
- The **Tools** group requires the agent to grasp an object to interact with the target object. The included tasks are: `slide block`, `drag stick` and `sweep to dustpan`.
- The **Motion** group requires precise control, which often causes failures due to the predefined motion planner. The included task is: `turn tap`.
- The **Screw** group requires gripper rotation to screw an object. The included task is: `close jar`.
- The **Occlusion** group involves tasks with severe occlusion problems from certain views. The included task is: `open drawer`.

Training Pipeline. To learn the policy, we uniformly sample a group of expert episodes from all the task variations, and then randomly choose an input-action pair for each of the tasks to form a batch. Other sampling strategies (*e.g.*, Auto- λ [41]) are also available. To simplify the tasks, the agent is assumed to access a

Table 3: Selected tasks.

Task	Type	Variations	Keyframes	Instruction	Template
close jar	color	20	6.0	"close the _ jar"	
open drawer	placement	3	3.0	"open the _ drawer"	
sweep to dustpan	size	2	4.6	"sweep dirt to the _ dustpan"	
meat off grill	category	2	5.0	"take the _ off the grill"	
turn tap	placement	2	2.0	"turn _ tap"	
slide block	color	4	4.7	"slide the block to _ target"	
put in drawer	placement	3	12.0	"put the item in the _ drawer"	
drag stick	color	20	6.0	"use the stick to drag the cube onto the _ target"	
push buttons	color	50	3.8	"push the _ button, [then the _ button]"	
stack blocks	color, count	60	14.6	"stack _ _ blocks"	

predefined motion planner (*e.g.*, RRT-Connect), so that the input-action pairs are determined as the bottleneck end-effector poses (*i.e.*, keyframes) within each demonstration based on empirical rules: a pose is determined as a keyframe if the end-effector changes state (*e.g.* close the gripper) or its velocities approach zero [7, 13, 28, 60, 76]. This setting simplifies the sequential decision-making problem into predicting the next optimal keyframe action based on the current observation, which can also be interpreted as a classification task.

B Additional Implementation Details

In this section, we detail the architectural design of each submodule in our Gaussian world model. For more details, please refer to our code.

Representation model. The representation model q_ϕ is the same with [76], which is not the main contribution in this paper. The representation model utilize a shallow 3D UNet to encode the voxel $\in \mathbb{R}^{100^3 \times 10}$ (RGB features, coordinates, indices, and occupancy) into the high-level visual features $\mathbf{v}^{(t)} \in \mathbb{R}^{100^3 \times 128}$.

Gaussian regressor. Given the current features $\mathbf{v}^{(t)}$ encoded by the representation model q_ϕ , we pass it through a generalizable Gaussian regressor g_ϕ to infer the Gaussian distribution $\theta^{(t)}$ directly. The Gaussian regressor is designed as a lightweight multi-head neural network, where each head is responsible for predicting a specific feature. It consists of: (1) a position offset head that predicts the per-pixel 3D center offset $\in \mathbb{R}^3$, (2) a color head that predicts the coefficients of the spherical harmonic basis $\in \mathbb{R}^{12}$, (3) a rotation head with normalization that predicts the rotation quaternion $\in \mathbb{R}^4$, (4) a scaling head with exponential activation that outputs the scaling factor $\in \mathbb{R}^3$, (5) an opacity head with sigmoid activation that predicts the opacity $\in \mathbb{R}^1$. (6) a semantic head that predicts the semantic feature $\in \mathbb{R}^3$.

Deformation predictor. After obtaining the current visual features $\mathbf{v}^{(t)}$, Gaussian embedding $\theta^{(t)}$ and action $a^{(t)}$, we parameterize the transition process as a deformation predictor p_ϕ to predict the deformation $\Delta\mu_i^{(t)} \in \mathbb{R}^3$ and $\Delta r_i^{(t)} \in \mathbb{R}^4$ of each Gaussian, resulting in the future Gaussian embedding $\theta^{(t+1)}$. The deformation predictor is a fully-connected network with residual connections.

Hyperparameters. The hyperparameters used in ManiGaussian are shown in Table 4. To train the robotic manipulation agent, we use $\lambda_{\text{Geo}} = 0.01$, $\lambda_{\text{Sem}} = 0.0001$ and $\lambda_{\text{Dyna}} = 0.001$ to focus on the action prediction. Other hyperparameters are in line with previous works [60, 76] for fair comparison.

Table 4: Hyperparameters.

Hyperparameter	Value
training iteration	100k
image resolution	128×128
voxel resolution	$100 \times 100 \times 100$
batch size	2
optimizer	LAMB
learning rate	0.0005
weight decay	0.000001
Number of Gaussian points	16384
λ_{Geo}	0.01
λ_{Sem}	0.0001
λ_{Dyna}	0.001

C Additional Quantitative Analysis

We provide further ablation study on different implementation choices in our ManiGaussian. Table 5 presents the impact of different balance hyperparameters on the overall performance, from which we can conclude that the balance of each loss item is important to learn an optimal manipulation policy.

Table 5: Impact of Balance Hyperparameters.

λ_{Geo}	λ_{Sem}	λ_{Dyna}	Planning	Long	Tools	Motion	Screw	Occlusion	Average
0.01	0	0.00001	42.0	24.0	48.0	48.0	28.0	72.0	42.4
0.01	0	0.0001	54.0	12.0	44.0	52.0	28.0	80.0	42.4
0.01	0	0.001	54.0	10.0	49.3	64.0	24.0	72.0	43.6
0.01	0.00001	0	48.0	8.0	34.7	48.0	24.0	64.0	35.2
0.01	0.0001	0	46.0	8.0	53.3	64.0	28.0	56.0	41.6
0.01	0.001	0	46.0	2.0	37.3	60.0	40.0	68.0	37.6
0.01	0.0001	0.001	40.0	14.0	60.0	56.0	28.0	76.0	44.8

D Additional Qualitative Analysis

We provide 9 additional comprehensive episodes generated by our ManiGaussian and the state-of-the-art generative method GNFactor [76] in the attached video file (*demo.mp4*). In the long-term “stack 2 rose blocks”, “put the item in the bottom drawer” and “take the steak off the grill” tasks, ManiGaussian completes the human instructions in the correct order with the understanding of high-level scene dynamics mined by the Gaussian world model. In the “sweep dirt to the short dustpan” and “use the stick to drag the cube onto the azure target” tasks that involve tool usage, our ManiGaussian succeeds in solving the tasks by correctly understanding the low-level scene dynamics of objects in contact. In the “slide the block to green target”, “turn left tap”, “close the azure jar” and “open the bottom drawer” tasks that require semantic understanding and precise control, our ManiGaussian can successfully comprehend the semantic information to interact with the correct object instance, while the baseline method often confuses different instances.