Gaussian Splatting Visual MPC for Granular Media Manipulation

Wei-Cheng Tseng[12], Ellina Zhang[1], Krishna Murthy Jatavallabhula[3] and Florian Shkurti[12]

***Abstract*— Recent advancements in learned 3D representations have enabled significant progress in solving complex robotic manipulation tasks, particularly for rigid-body objects. However, manipulating granular materials such as beans, nuts, and rice, remains challenging due to the intricate physics of particle interactions, high-dimensional and partially observable state, inability to visually track individual particles in a pile, and the computational demands of accurate dynamics prediction. Current deep latent dynamics models often struggle to generalize in granular material manipulation due to a lack of inductive biases. In this work, we propose a novel approach that learns a visual dynamics model over Gaussian splatting representations of scenes and leverages this model for manipulating granular media via Model-Predictive Control. Our method enables efficient optimization for complex manipulation tasks on piles of granular media. We evaluate our approach in both simulated and real-world settings, demonstrating its ability to solve unseen planning tasks and generalize to new environments in a zero-shot transfer. We also show significant prediction and manipulation performance improvements compared to existing granular media manipulation methods.**

I. Introduction

Neural rendering and view synthesis methods [1], [2], [3], [4], [5], [6] have enabled a wide set of applications in scene understanding, 3D reconstruction and representation. Moreover, they have shown promise in many complex robot manipulation tasks on rigid body objects [7], [8]. Manipulating granular materials such as beans, nuts, rice, oats, and other such objects common in daily life remains a challenging problem, so in this paper we address the question of whether neural rendering methods, Gaussian Splatting in particular, provide a good representation for control of granular media.

Several factors contribute to the difficulty of granular material manipulation. First, modeling the interactions between particles is complicated due to the intricate physics involved [9] and the unknown geometry of individual particles. Second, accounting for all particles in planning requires a high-dimensional state [10], [11], which creates challenges for downstream policy learning or planning algorithms. Third, visually identifying and tracking individual granular particles in a pile is nearly impossible due to their self-similarity, which leads to data association problems, and due to the inherent partial observability of the setting. This has led to the majority of existing works that simulate granular materials doing so without ground truth data from the real world [12], [13], [14], relying on simulated data. Finally, accurately predicting particle dynamics is computationally expensive [9], [15], [16].

Corresponding author email: weicheng.tseng@mail.utoronto.ca
[1] University of Toronto, [2] Vector Institute, [3] MIT CSAIL

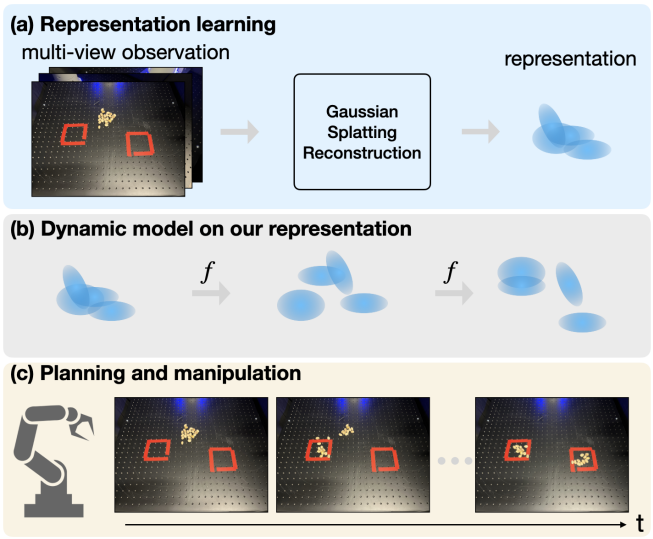

Fig. 1. Our method takes a few multi-view images of a scene and their corresponding camera poses as input (a) converts them into their Gaussian splatting representation, (b) learns a dynamics model over these representations, and (c) performs visual model-predictive control for granular material manipulation, which requires view synthesis and dynamics rollouts.

To address these challenges, recent efforts have focused on modeling the visual dynamics of granular piles using highly expressive neural network latent dynamics models directly from pixels [17]. However, these models often underperform compared to linear dynamics models due to a lack of inductive biases. In contrast, physics-inspired approaches, such as particle-based models, introduce strong inductive biases for neural network dynamics models.

In this work, we show that Gaussian splatting [4] over video frames provides effective representations for downstream model-predictive control over granular media. Gaussian splatting is an image rendering and reconstruction technique originating in computer graphics that represents a 3D scene as a collection of Gaussians (splats), each of which is centered around points in the scene and is associated with a single color. An image of the scene from a given viewpoint is rendered by projecting each splat into 2D and blending colors where they overlap. This collection of splats that can also be parameterized by position, and rotation of Gaussian provides a smooth and continuous representation in terms of space of the scene, making it particularly well-suited for modeling video sequences.

Our contribution: We use the Gaussian splats representing the scene at each time as a state vector that can be manipulated via MPC, effectively lowering the dimensionality of the image. We learn a dynamics model over Gaussian splats and show that by doing MPC with this dynamics model

and representation, robots can efficiently handle complex and precise manipulation tasks involving granular materials. This representation enables robots to optimize their actions, anticipate challenges, and adapt to dynamic environments.

We evaluate our approach in a variety of pile manipulation tasks in both simulation and real-world settings, and we show that our method outperforms existing baselines both in terms of dynamics prediction and in terms of task performance. Our model also demonstrates the ability to solve previously unseen complex planning tasks. Furthermore, we showcase the generalizability of our method by transferring a trained model to different environments with varying object shapes in a zero-shot setting. [1]

II. Related Works

3D Visual Representations for Manipulation. The use of 3D visual representations [1], [2], [3], [4], [5], [6] for robotic manipulation has gained significant traction in recent years. One of the foundational approaches in this domain involves the use of 3D point clouds, which provide a detailed geometric representation of objects in the environment. Works such as PointNet [18] and PointNet++ [19] have been pivotal in processing and understanding 3D point clouds, enabling robots to perform tasks like object recognition and grasping [20]. Voxel-based representations [3] have also been widely explored for manipulation tasks. By discretizing the workspace into a grid of voxels, these methods offer a straightforward way to model the occupancy and structure of the environment. VoxNet [21] introduced a deep learning architecture that uses 3D voxel grids for object recognition in robotic tasks. Similarly, a voxel-based deep Q-network [22], [23] was developed for robotic grasping, which demonstrated the effectiveness of 3D voxel representations in manipulation tasks.

Another prominent line of research involves the use of implicit neural representations, where 3D shapes and environments are encoded as continuous functions rather than discrete points or voxels. Neural Radiance Fields (NeRF) [1], [6] is a notable example of this approach, where scenes are represented as volumetric radiance fields that can be rendered from arbitrary viewpoints. While originally designed for view synthesis, NeRF and its variants have inspired applications in robotic manipulation, especially in scenarios where precise modeling of object geometry and appearance is critical [24], [25].

Particle Dynamics. Inductive biases, particularly object-centric representations, have been widely adopted in learning-based dynamics models. Particle-based representations serve as strong inductive biases for representing deformable objects. In particular, DPI-Net [12] combines a hierarchical particle dynamics model with MPC-based control for deformable object manipulation. However, particle-based approaches face scalability issues as the number of particles increases [26], [27], thereby making them computationally expensive and challenging to use in practical planning tasks.

[1]For more details, code, videos, and the paper's appendix please refer to our project website: https://weichengtseng.github.io/gs-granular-mani/.

Granular Material Manipulation. Manipulating granular media by pushing piles of small objects into a desired target set using visual feedback has been accomplished with models as simple as linear [17], [11]. Granular material manipulation presents unique challenges due to the complex and non-linear interactions of these materials [28]. Traditional approaches often rely on physics-based models [9], which simulate individual particles to predict bulk material behavior. While accurate, these models are computationally intensive and may not be suitable for real-time manipulation.

To address these limitations, recent research has explored data-driven methods that learn granular dynamics from observations [29]. Techniques like neural networks have been employed to approximate material behavior [28], [16], [17], enabling faster predictions during manipulation tasks. However, these approaches often struggle with generalization across different types of granular materials and varying conditions. Besides, these granular material manipulations only represent material in 2D space, which constrains the potential of 3D manipulation tasks. Our approach leverages advanced 3D reconstruction technique, which alleviates this limitation.

III. Preliminaries

Gaussian splatting [4] has emerged as a powerful rendering technique that can capture the state of the visual world with a discrete set of 3D Gaussians $G = \{n^i\}$, where $n^i = (\boldsymbol{g}^i, \boldsymbol{s}^i, \boldsymbol{R}^i, \boldsymbol{s}^i, \sigma^i, \boldsymbol{c}^i)$ represents a 3D Gaussian. Each Gaussian i is parameterized by its position $\boldsymbol{g}^i \in \mathbb{R}^3$, orientation $\boldsymbol{R}^i \in \mathbf{SO}(3)$, scale $\boldsymbol{s}^i \in \mathbb{R}^3$, opacity $\sigma^i \in \mathbb{R}^+$, and color $\boldsymbol{c}^i \in \mathbb{R}^3$. Given a viewpoint whose transform relative to the world frame is denoted by $\boldsymbol{V} \in \mathbf{SE}(3)$ and projection function from the 3D world to the view's screenspace is defined by $\pi(\boldsymbol{x})$, the color at a pixel coordinate $\boldsymbol{p}$ can be calculated by sorting the Gaussians in increasing order of their viewspace z-coordinate and then using the splatting formula:

$$C_{\mathrm{RGB}}(\boldsymbol{p}) = \sum_{i \in N} \boldsymbol{c}^i \alpha^i(\boldsymbol{p}) \prod_{j=1}^{i-1} (1 - \alpha^i(\boldsymbol{p})) \tag{1}$$

$$\alpha^i(\boldsymbol{p}) = \sigma_i \exp(g_i(\boldsymbol{p}))\,, \tag{2}$$

$$g_i(\boldsymbol{p}) = \mathbf{x}^{iT} \hat{\Sigma}_i^{-1} \mathbf{x}^i\,, \mathbf{x}^i = \boldsymbol{p} - \pi(\boldsymbol{g}^i) \tag{3}$$

$\hat{\Sigma}_i = \mathbf{J}\mathbf{V}\Sigma_i\mathbf{V}^T\mathbf{J}^T$ is the covariance of the Gaussian i projected into the viewpoint's screenspace where $\mathbf{J}$ is the Jacobian of the projection function $\pi(\boldsymbol{x})$ and $\Sigma_i = \boldsymbol{R}_i \,\mathrm{diag}(\boldsymbol{s}_i^2)\, \boldsymbol{R}_i^T$. Further details about this process can be found in [4].

The parameters of the Gaussian can be updated by minimizing the $\mathscr{L}_1$ loss and structural similarity index measure (SSIM) $\mathscr{L}_{SSIM}$ between the reconstructed image and ground-truth image.

$$\begin{aligned} \mathscr{L}_{recon}(I_{recon}, I_{GT}) = & \mathscr{L}_1(I_{recon}, I_{GT}) + \\ & \beta(1 - \mathscr{L}_{SSIM}(I_{recon}, I_{GT})) \end{aligned} \tag{4}$$

where I_{GT} and I_{recon} are ground-truth and reconstructed images (with reconstructed pixel $\boldsymbol{p}$ having color $C_{\mathrm{RGB}}(\boldsymbol{p})$), respectively. Note that SSIM is calculated using various

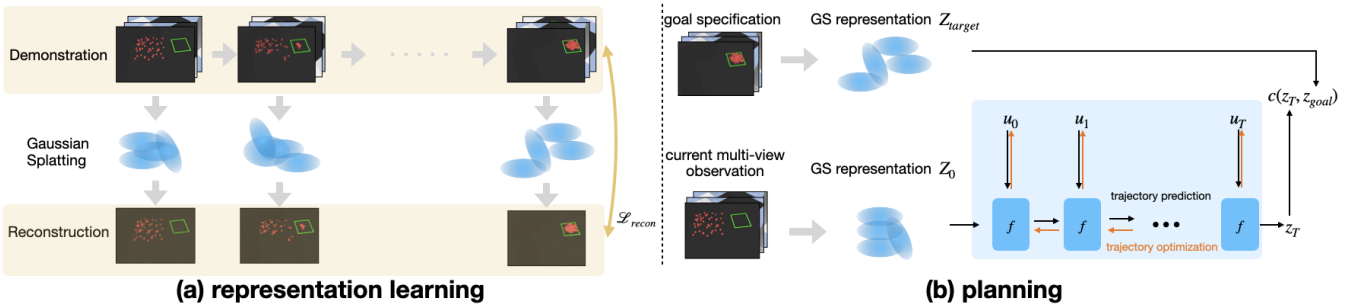

Fig. 2. (a) The dynamics model f, conditioned on the input action $\boldsymbol{u}_t$, predicts the temporal evolution of the scene representation zt. During planning time, we calculate the task objective $c(\boldsymbol{Z}_T, \boldsymbol{Z}_{target})$ and backpropagate the gradients to optimize the action sequence $\boldsymbol{u}_t$ (b) The dynamics model f, conditioned on the input action $\boldsymbol{u}_t$, predicts the temporal evolution of the scene representation $\boldsymbol{Z}_t$. During planning time, we calculate the task objective $c(\boldsymbol{Z}_T, \boldsymbol{Z}_{target})$ and backpropagate the gradients to optimize the action sequence $\boldsymbol{u}_t$.

windows of an image. The measure between two windows x and y of common size $N \times N$.

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + c_1)(2\sigma_{xy} + c_2)}{(\mu_x^2 + \mu_y^2 + c_1)(\sigma_x^2 + \sigma_y^2 + c_2)} \tag{5}$$

where μ_x and μ_y are pixel mean of x and y. σ_x and σ_y are pixel variance of x and y. σ_{xy} is the covariance of x and y. c_1 and c_2 are constant to stabilize division.

IV. Our Approach

A. Problem Formulation

Given multi-view RGBD observations $\boldsymbol{O}_{target} = \{\boldsymbol{o}^v, m^v\}_{v=1}^N$ of the target pattern of the granular material, where o^v represents the RGBD image and m^v indicates the corresponding camera pose, we would like to manipulate the granular material to minimize the cost measurement c such that the granular material can match the target pattern. We define the following trajectory optimization problem over a horizon T

$$\boldsymbol{u}_t = argmin_{\boldsymbol{u}_t} c(\boldsymbol{Z}_T, \boldsymbol{Z}_{target}) \tag{6}$$

$$\boldsymbol{Z}_0 = h(\boldsymbol{O}_0),\ \boldsymbol{Z}_{target} = h(\boldsymbol{O}_{target}),\ \boldsymbol{Z}_{t+1} = f(\boldsymbol{Z}_t, \boldsymbol{u}_t) \tag{7}$$

where h is the perception module that performs Gaussian splatting and $f()$ is the dynamic model which predicts the representation's evolution $\boldsymbol{Z}_{t+1}$ from the previous representation and action. The optimization aims to find the action sequence $\{\boldsymbol{u}_t\}$ to minimize the cost function $c(\boldsymbol{Z}_T, \boldsymbol{Z}_{target})$. Also, $\boldsymbol{u}_t \in \mathbb{R}^4$ represents the starting position and pushing direction of the end-effector. To enable learning a dynamics model for granular materials, we collect a dataset $D_{\text{RGBD}} = \{(\boldsymbol{O}_t, \boldsymbol{u}_t, \boldsymbol{O}_{t+1})\}$ via a simulator of a manipulator interacting with particles.

B. Representation of Granular Materials

We leverage Gaussian splatting [4] as the representation method for image observations of granular materials. Instead of initializing the Gaussian splats with a pointcloud formed from structure-from-motion, as is typically done in 3D scene reconstruction applications of neural rendering, we lift the RGBD image with the corresponding camera pose to 3D space and form the point cloud. To ease the downstream training of dynamic models, we downsample the point clouds used by initializing the Gaussian splatting with farthest point sampling [19], which iteratively samples the farthest point and performs distance updating. Given the multiview observation $\{\boldsymbol{O}_t\}$, we leverage Gaussian splatting for reconstruction $\{\boldsymbol{G}_t\}$. Since we are only interested in granular materials, we remove the Gaussian that represents the background or the ones that have high transparency. We then denote the remaining Gaussians as $\{\boldsymbol{Z}_t\}$. More specifically, we transform the dataset D_{RGBD} to $D_{GS} = \{(\boldsymbol{Z}_t, \boldsymbol{u}_t, \boldsymbol{Z}_{t+1})\}$ by training Gaussian splatting reconstruction for each image frame.

C. Learning a Visual Dynamics Model

Given the dataset D_{GS}, we learn a visual dynamics model over Gaussian splatting representations. Our dynamics model $f()$ is formed as a Graph Neural Network (GNN) [30] with iterative message passing that takes the Gaussian splatting reconstruction as input and predicts the translation and rotation for each Gaussian.

To be more specific, to transform Gaussian splitting $\boldsymbol{Z}_t$ into a graph structure $\mathscr{G}_t = (\mathscr{V}_t, \mathscr{E}_t)$, where $\mathscr{V}_t = \{v_t^i\}_{i=1...|\mathscr{V}_t|}$ indicate a set of nodes and $\mathscr{E}_t = \{e_t^i\}$ represent a set of edges. we create a graph by adding edges between Gaussians if the L_2 distance between the respective vertices is smaller than a distance threshold ω. We form the node features of the GNN as $(\boldsymbol{c}_t^i, \sigma_t^i, \boldsymbol{R}_t^i, \boldsymbol{g}_t^i, \boldsymbol{s}_t^i)$ for node $\boldsymbol{v}_t^i$. f consists of node encoder f_{enc} with node representation $\bar{v}^i$ from v_t^i:

$$\bar{\boldsymbol{v}}_t^i = f_{enc}(\boldsymbol{v}_t^i, \boldsymbol{u}_t) \tag{8}$$

Then, we have a message-passing encoder f_{msg} which allows us do multi-step message passing:

$$\boldsymbol{q}_t^{i,\gamma+1} = f_{msg}(\boldsymbol{q}_t^{i,\gamma}, \text{mean}_{j \in N_i} \boldsymbol{q}_t^{j,\gamma}),\ \boldsymbol{q}_t^{i,0} = \bar{\boldsymbol{v}}_t^i \tag{9}$$

where N_i is a set of nodes that connected to node i. Finally, we have the decoder f_{dec} that transforms node features after Γ message passing steps to dynamic information

$$\Delta \boldsymbol{r}_t^i, \Delta \boldsymbol{g}_t^i = f_{dec}(\boldsymbol{q}_t^{i,\Gamma}) \tag{10}$$

where $\Delta \boldsymbol{r}^i$ and $\Delta \boldsymbol{g}^i$ indicates the displacement and rotation of Gaussian $\boldsymbol{n}^i$. We then move and rotate the Gaussian:

$$\hat{\boldsymbol{g}}_{t+1}^i = \boldsymbol{g}^i + \Delta \boldsymbol{g}_t^i,\ \hat{\boldsymbol{r}}_{t+1}^i = \Delta \boldsymbol{r}^i \cdot \boldsymbol{r}_t^i \tag{11}$$

where $\boldsymbol{r}$ is the quaternion representation of $\boldsymbol{R}$. In the end, we obtain a set of Gaussians that represents the next image:

$$\hat{\boldsymbol{Z}}_{t+1} = \{(\boldsymbol{c}_t^i, \alpha_t^i, \hat{\boldsymbol{R}}_{t+1}^i, \hat{\boldsymbol{g}}_{t+1}^i, \boldsymbol{s}_t^i)\} \tag{12}$$

Then, we use the Chamfer distance to train the dynamics:

$$\begin{aligned}\mathscr{L}_{dyna} = &\frac{1}{|\hat{\boldsymbol{Z}}_{t+1}|}\sum_{\hat{\boldsymbol{n}}_{t+1}\in\hat{\boldsymbol{Z}}_{t+1}}\min_{\boldsymbol{n}_{t+1}\in\boldsymbol{Z}_{t+1}}\mathscr{L}_{gaussian}(\hat{\boldsymbol{n}}_{t+1},\boldsymbol{n}_{t+1}) \\ &+\frac{1}{|\boldsymbol{Z}_{t+1}|}\sum_{\boldsymbol{n}_{t+1}\in\boldsymbol{Z}_{t+1}}\min_{\hat{\boldsymbol{n}}_{t+1}\in\hat{\boldsymbol{Z}}_{t+1}}\mathscr{L}_{gaussian}(\boldsymbol{n}_{t+1},\hat{\boldsymbol{n}}_{t+1})\end{aligned} \tag{13}$$

$$\mathscr{L}_{gaussian}(n,\hat{n}) = ||\boldsymbol{g}-\hat{\boldsymbol{g}}||_2 + \lambda(1-|\boldsymbol{r}_{t+1}\cdot\hat{\boldsymbol{r}}|) \tag{14}$$

where λ is a hyperparameter that determines the importance between position and orientation.

D. Planning

Inspired by [31], we leverage the density field

$$d(\boldsymbol{x}) = \sum_{\boldsymbol{n}_i\in\boldsymbol{Z}}\sigma_i\cdot\exp((\boldsymbol{x}-\boldsymbol{g}_i)^T\Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{g}_i)) \tag{15}$$

which indicates whether a specific 3D position x is occupied by any material and we use it to form the cost function for the planning algorithm. The cost function used in the planning algorithm is the following:

$$c(\boldsymbol{Z}_t,\boldsymbol{Z}_{target}) = \frac{1}{|P|}\sum_{\boldsymbol{x}\in P}|d_t(\mathbf{x}) - d_{target}(\boldsymbol{x})|^2 \tag{16}$$

where P is a pre-defined set of points we would like to query. This cost helps us measure the difference between the occupied space in $\boldsymbol{Z}_t$ and that in $\boldsymbol{Z}_{target}$.

We integrate the solution to Equation 6 into a MPC framework using the optimized action sequence in a closed-loop system. At each MPC step, we follow the procedure outlined in Algorithm 1, which first determines the appropriate resolution for representing the environment. Next, a combination of sampling and gradient descent is used to derive the action sequence through trajectory optimization, employing the shooting method. After executing the first action from the sequence in the real world, new observations are obtained, and Algorithm 1 is applied again. This iterative process allows the system to continuously incorporate environmental feedback and adaptively choose the optimal resolution as the task progresses. Further details on the task objectives and MPC hyperparameters are provided in the supplementary materials.

V. Experimental Results

A. Implementation Details

We implement the entire framework using PyTorch [32] and PyTorch-Geometric [33].

Simulation Setup. The simulation environment is based on Pybullet [34], adapted from the Ravens [35] framework. Throughout both data generation and evaluation, unless otherwise specified, we use a set of 50 cubic blocks, each measuring 1 cm in size, along with a planar pusher of 5 cm in length.

Algorithm 1: Planning algorithm

Data: Current observation $\boldsymbol{O}_t$, target $\boldsymbol{O}_{target}$, planning horizon T, the dynamics module f, and gradient descent iteration N
Result: a sequence of action actions $\boldsymbol{u}_{0:T-1}$

Get current representation $\boldsymbol{Z}_t$ from observation $\boldsymbol{O}_t$;
and target representation $\boldsymbol{Z}_{target}$ from $\boldsymbol{O}_{target}$;
Sample K action sequences $\boldsymbol{u}_{0:T-1}^{1:K}$;

for k = *1,...,K* **do**
  for i = *1,...,N* **do**
    for t = *0,...,T-1* **do**
      Predict the next step $\boldsymbol{Z}_{t+1} = f(\boldsymbol{Z}_t, \boldsymbol{u}_t)$;
    end
    Compute the task loss $c^k = c(\boldsymbol{Z}_T, \boldsymbol{Z}_{target})$;
    update the $\boldsymbol{u}_{0:T-1}^{1:K}$ with the task loss;
  end
end
$k_{opt} = argmin_k c^k$;
return $\boldsymbol{u}_{0:T-1}^{k_{opt}}$;

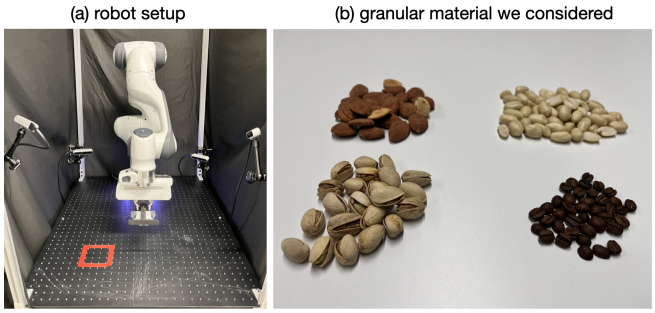

Fig. 3. **Real-world experiment setup**. (a) The robotic manipulator, equipped with a pusher at the end-effector, moves object piles within the workspace. Four calibrated RGBD cameras mounted around the workspace provide visual observations of the environment. (b) The granular materials used in real-world experiments include coffee beans, peanuts, pistachios, and almonds.

Physical Robot Setup. The real-world setup features a Franka Panda manipulator [36] and four Intel RealSense cameras (see Appendix for further details). The camera captures a top-down view of the workspace, with images rectified through homographic warping. Color and depth thresholding are then applied to extract the object density field. We directly transfer a model trained in simulation to real-world experiments. To ensure consistency between simulation and reality, we resize the input images to match the workspace size used in the simulation. The pusher has the same width as its simulated counterpart and is attached to the robot's gripper. Fig. 3 shows the granular materials tested.

Baselines. We compare our approach against several baselines, providing a brief description of each below:

- **Dyn-Res**[16] constructs dynamic-resolution particle representations of the environment and learns a unified

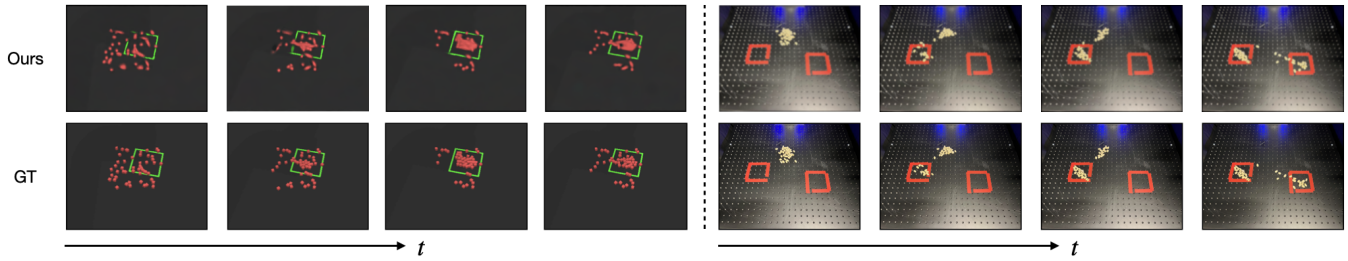

Fig. 4. **Rollout of our dynamics model.** We show the rollout predictions of dynamic model in both simulation (left) and real-world data (right). Both of the rollout results show that the dynamics model prediction is accurate for a few steps.

dynamics model using GNNs. This model allows for continuous adjustment of the abstraction level. During testing, the agent adaptively determines the optimal resolution at each MPC step.

- **NFD**[28] employs a fully convolutional neural network that operates on a density field-based representation of object piles and pushers. This approach leverages the spatial locality of inter-object interactions and translation equivariance through convolutional operations.
- **NeRF-dy** [37] integrates NeRF with time contrastive learning in an autoencoding framework. It learns viewpoint-invariant, 3D-aware scene representations. By constructing a dynamics model over this learned representation space, NeRF-dy enables visuomotor control for challenging manipulation tasks.
- **DVF** [17] is a strong dynamics model that uses field-based state representation without inductive biases. DVF only predicts the start and end position of a straight push.

We evaluate the entire framework on the following tasks:

- **Collecting**: pushing the piles into a target region.
- **Splitting**: pushing the piles into multiple target regions.
- **Redistributing**: redistributing the piles to match a specific pattern.

Metrics. In simulation, we perform 100 trials, while for real-world experiments, we conduct 20 trials. We use two metrics to assess the performance of the framework.

- **Success rate**: success is defined as moving all materials to the target region.
- **State error**: in simulation experiments, we also measure the Chamfer distance between the particles in the target observation and the particles after manipulation.

B. Reconstruction and Dynamics Prediction Results

In Fig. 4, we show the reconstruction of the rollout trajectories with our dynamic model. We can find that our approach does capture the granular material, and the movement of the granular material is also correctly predicted by our dynamic model. We also provide more novel-view synthesis results in supplementary.

C. Manipulation Results

As shown in Tables I and II, our approach consistently outperforms all baseline methods. When comparing our method to NeRF-dy, we observe significant improvements, which we attribute to our incorporation of particle clustering.

TABLE I

MANIPULATION SUCCESS RATE IN SIMULATION (MAX = 1.0)

	Collection	Splitting	Redistributing
NeRF-dy [37]	0.67	0.43	0.31
Dyn-Res [16]	0.79	0.72	0.67
NFD [28]	**0.89**	0.82	0.46
DVF [17]	0.78	0.67	0.55
Ours	**0.89**	**0.88**	**0.78**

TABLE II

STATE ERROR IN SIMULATION

	Collection	Splitting	Redistributing
NeRF-dy [37]	0.0096	0.0620	0.0717
Dyn-Res [16]	0.0179	0.0533	0.0901
NFD [28]	0.0073	0.0310	0.0660
DVF [17]	0.0093	0.0340	0.0919
Ours	**0.0027**	**0.0041**	**0.0081**

This clustering allows for a better understanding of inter-particle interactions. In contrast, NeRF-dy relies on learning dynamics through NeRF reconstruction, which lacks the use of physics-based priors. NFD performs well on simpler tasks like collection and splitting but struggles with more complex target patterns.

TABLE III

REAL-WORLD MANIPULATION SUCCESS RATE (MAX = 1.0).

	Collection	Splitting
Pistachios	0.85	0.80
Almonds	0.85	0.75
Peanuts	0.85	0.85
Coffee Beans	0.65	0.60

In real-world experiments, as shown in Table III, we observe high manipulation performance across most materials. Qualitative results are presented in Fig. 5, and additional demonstrations are available on our project website.

D. Generalization Studies

In this section, we conduct ablation studies to evaluate the effectiveness of each component.

Number of Views. Figure 6 demonstrates that increasing the number of viewpoints improves manipulation performance by offering a more accurate reconstruction of the granular material. Our approach achieves higher performance than NeRF-dy while requiring fewer views.

Generalization to Different Numbers of Particles. Though our approach was trained with 50 particles, we found

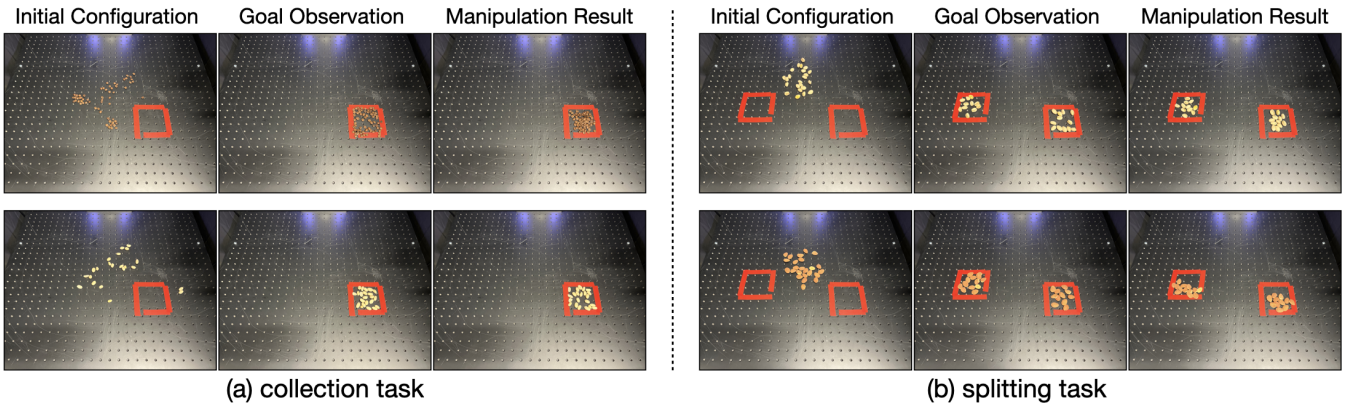

Fig. 5. **Qualitative results from real-world experiments.** (a) Evaluation of our method on a collection task with different objects than what it was trained on. The objects vary in scale and physical properties (e.g., almonds and pistachios remain quasi-static during MPC steps, while peanuts and coffee beans may roll after being pushed). (b) Pushing object piles into two separate target configurations. Our method successfully pushes randomly scattered objects into the desired locations.

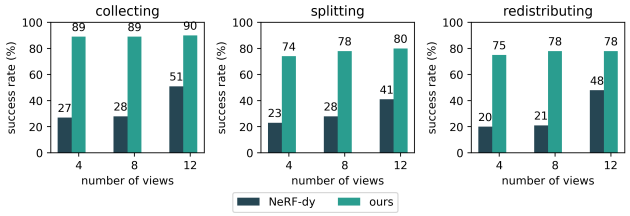

Fig. 6. **Manipulation performance with different numbers of viewpoints as input.** Performance increases with more views, providing more accurate granular material reconstruction.

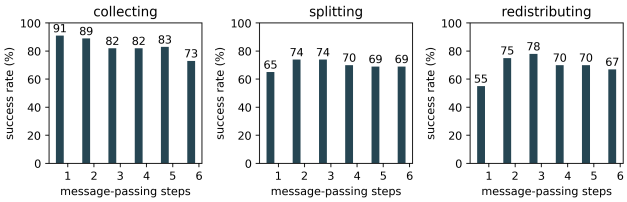

Fig. 8. **Manipulation performance with different numbers of message-passing steps.** More steps lead to better performance.

that it generalizes well to varying numbers of particles (Fig. 7), largely due to the graph neural network we use to model dynamics. In contrast, NeRF-dy and NFD, which rely on latent vectors or visual observations, struggle to generalize across different data distributions.

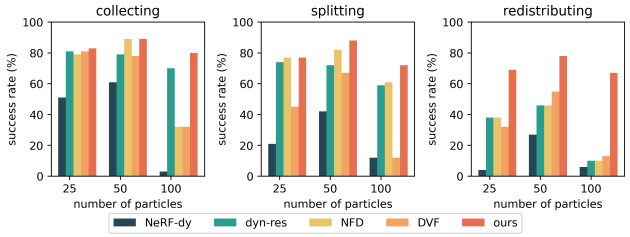

Fig. 7. **Manipulation performance with different numbers of particles in the workspace.** Our approach demonstrates superior generalization compared to other baselines.

Message Passing in Dynamics. We find that message passing plays a crucial role in capturing granular material dynamics. Tasks requiring accurate future state predictions benefit from additional message-passing steps for precise manipulation.

VI. Limitations

Manipulation Efficiency. Granular material dynamics are inherently complex due to the intricate interactions between particles. To manage this challenge, we limit the robot arm's movement speed, reducing the intensity of particle interactions and promoting more stable and controllable manipulation. **Particle Size Limitation**. Our framework is less suited for manipulating very small particles, such as salt, sugar, or rice. This limitation stems from the difficulty in accurately reconstructing such tiny particles using Gaussian splatting, which struggles to maintain precision at smaller scales. We anticipate however that this would be an issue for any vision-based policy. **Shortsighted Planning**. While our method effectively optimizes for the next best trajectory based on a global cost function, it focuses on short-term decision-making. In more complex scenarios, planning several steps ahead may be necessary to determine truly optimal actions. Future work will explore longer-horizon predictions and the development of more efficient sampling and optimization strategies to facilitate long-term planning.

VII. Conclusion

This work presents a novel approach to granular material manipulation using Gaussian splatting as a latent representation. By encoding the material's state into a probabilistic form, we effectively model and predict the dynamics of granular interactions. Our integration of this learned model with Model Predictive Control enables precise and adaptive manipulation in real-time.

Experiments demonstrate that our method significantly improves manipulation accuracy and stability over existing approaches. This highlights the potential of Gaussian splatting as a powerful tool for advancing robotic manipulation, especially in complex environments. Future work could extend this framework to other non-rigid materials, further enhancing the capabilities of robotic systems in dynamic tasks.

REFERENCES

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

[2] V. Sitzmann, M. Zollhöfer, and G. Wetzstein, "Scene representation networks: Continuous 3d-structure-aware neural scene representations," in *Advances in Neural Information Processing Systems*, 2019.

[3] L. Mescheder, M. Oechsle, M. Niemeyer, S. Nowozin, and A. Geiger, "Occupancy networks: Learning 3d reconstruction in function space," in *Proceedings IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3d gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, July 2023. [Online]. Available: https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/

[5] J. J. Park, P. Florence, J. Straub, R. Newcombe, and S. Lovegrove, "Deepsdf: Learning continuous signed distance functions for shape representation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[6] W.-C. Tseng, H.-J. Liao, L. Yen-Chen, and M. Sun, "Cla-nerf: Category-level articulated neural radiance field," in *2022 International Conference on Robotics and Automation (ICRA)*. IEEE Press, 2022, p. 8454–8460. [Online]. Available: https://doi.org/10.1109/ICRA46639.2022.9812272

[7] Q. Dai, Y. Zhu, Y. Geng, C. Ruan, J. Zhang, and H. Wang, "GraspNeRF: Multiview-based 6-DoF grasp detection for transparent and specular objects using generalizable NeRF," *arXiv [cs.RO]*, Oct. 2022.

[8] Z. Jiang, Y. Zhu, M. Svetlik, K. Fang, and Y. Zhu, "Synergies between affordance and geometry: 6-dof grasp detection via implicit representations," *Robotics: science and systems*, 2021.

[9] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *ICLR*, 2019.

[10] C. Schenck, J. Tompson, S. Levine, and D. Fox, "Learning robotic manipulation of granular media," in *Proceedings of the 1st Annual Conference on Robot Learning*, ser. Proceedings of Machine Learning Research, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., vol. 78. PMLR, 2017, pp. 239–248.

[11] N. Tuomainen, D. B. Mulero, and V. Kyrki, "Manipulation of granular materials by learning particle interactions," *CoRR*, vol. abs/2111.02274, 2021. [Online]. Available: https://arxiv.org/abs/2111.02274

[12] Y. Li, J. Wu, R. Tedrake, J. B. Tenenbaum, and A. Torralba, "Learning particle dynamics for manipulating rigid bodies, deformable objects, and fluids," in *ICLR*, 2019.

[13] W. F. Whitney, T. Lopez-Guevara, T. Pfaff, Y. Rubanova, T. Kipf, K. Stachenfeld, and K. R. Allen, "Learning 3d particle-based simulators from rgb-d videos," in *ICLR*, 2024.

[14] X. Li, Y. Cao, M. Li, Y. Yang, C. Schroeder, and C. Jiang, "Plasticitynet: Learning to simulate metal, sand, and snow for optimization time integration," in *Advances in Neural Information Processing Systems*, A. H. Oh, A. Agarwal, D. Belgrave, and K. Cho, Eds., 2022. [Online]. Available: https://openreview.net/forum?id=_WqHmwoE7Ud

[15] C. Jiang, C. Schroeder, J. Teran, A. Stomakhin, and A. Selle, "The material point method for simulating continuum materials," in *ACM SIGGRAPH 2016 Courses*, ser. SIGGRAPH '16. New York, NY, USA: Association for Computing Machinery, 2016. [Online]. Available: https://doi.org/10.1145/2897826.2927348

[16] Y. Wang, Y. Li, K. Driggs-Campbell, L. Fei-Fei, and J. Wu, "Dynamic-resolution model learning for object pile manipulation," in *Robotics: Science and Systems*, 2023.

[17] H. J. T. Suh and R. Tedrake, "The surprising effectiveness of linear models for visual foresight in object pile manipulation," in *Workshop on Algorithmic Foundations of Robotics (WAFR)*, 2020.

[18] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3d classification and segmentation," *arXiv preprint arXiv:1612.00593*, 2016.

[19] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *arXiv preprint arXiv:1706.02413*, 2017.

[20] M. Liu, X. Li, Z. Ling, Y. Li, and H. Su, "Frame mining: a free lunch for learning robotic manipulation from 3d point clouds," in *6th Annual Conference on Robot Learning*, 2022.

[21] D. Maturana and S. Scherer, "Voxnet: A 3d convolutional neural network for real-time object recognition," in *IROS*, 2015, pp. 922–928.

[22] S. James, K. Wada, T. Laidlow, and A. J. Davison, "Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation," *CoRR*, vol. abs/2106.12534, 2021. [Online]. Available: https://arxiv.org/abs/2106.12534

[23] S. James and A. J. Davison, "Q-attention: Enabling efficient learning for vision-based robotic manipulation," *CoRR*, vol. abs/2105.14829, 2021. [Online]. Available: https://arxiv.org/abs/2105.14829

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

[25] L. Yen-Chen, P. Florence, J. T. Barron, T.-Y. Lin, A. Rodriguez, and P. Isola, "NeRF-Supervision: Learning dense object descriptors from neural radiance fields," in *IEEE Conference on Robotics and Automation (ICRA)*, 2022.

[26] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," *CoRR*, vol. abs/2002.09405, 2020. [Online]. Available: https://arxiv.org/abs/2002.09405

[27] K. Kumar and J. Vantassel, "Gns: A generalizable graph neural network-based simulator for particulate and fluid modeling," 2022. [Online]. Available: https://arxiv.org/abs/2211.10228

[28] S. Xue, S. Cheng, P. Kachana, and D. Xu, "Neural field dynamics model for granular object piles manipulation," in *CoRL*, 2023.

[29] T. Xie, Z. Zong, Y. Qiu, X. Li, Y. Feng, Y. Yang, and C. Jiang, "Physgaussian: Physics-integrated 3d gaussians for generative dynamics," *arXiv preprint arXiv:2311.12198*, 2023.

[30] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," *CoRR*, vol. abs/1706.02216, 2017. [Online]. Available: http://arxiv.org/abs/1706.02216

[31] J. Tang, J. Ren, H. Zhou, Z. Liu, and G. Zeng, "Dreamgaussian: Generative gaussian splatting for efficient 3d content creation," *arXiv preprint arXiv:2309.16653*, 2023.

[32] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer, "Automatic differentiation in pytorch," in *NIPS-W*, 2017.

[33] M. Fey and J. E. Lenssen, "Fast graph representation learning with PyTorch Geometric," in *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[34] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2021.

[35] A. Zeng, P. Florence, J. Tompson, S. Welker, J. Chien, M. Attarian, T. Armstrong, I. Krasin, D. Duong, V. Sindhwani, and J. Lee, "Transporter networks: Rearranging the visual world for robotic manipulation," *Conference on Robot Learning (CoRL)*, 2020.

[36] K. Zhang, M. Sharma, J. Liang, and O. Kroemer, "A modular robotic arm control stack for research: Franka-interface and frankapy," *arXiv preprint arXiv:2011.02398*, 2020.

[37] Y. Li, S. Li, V. Sitzmann, P. Agrawal, and A. Torralba, "3d neural scene representations for visuomotor control," in *5th Annual Conference on Robot Learning*, 2021. [Online]. Available: https://openreview.net/forum?id=zv3NYgRZ7Qo