

Supplementary Materials for GraspSplats: Efficient Manipulation with 3D Feature Splatting

Anonymous Author(s)

Affiliation

Address

email

1 Supplementary Video

We include videos to animate Gaussians with features using PCA, and different robotics tasks. Please refer to the video for the following content:

1. Motivation and a brief overview of method design. (00:00-01:10 in supp.mp4)
2. System walkthrough (01:10-01:37)
3. System walkthrough (01:10-01:37)
4. Part-level Manipulation in static scenes (01:37-01:54)
5. Manipulation under dynamic scenes (01:54-02:20)

2 Mathematical Details of Feature Splatting

Point and surface splatting methods represent a scene explicitly via a mixture of 2D or 3D Gaussian ellipsoid. In the case of Gaussian Splatting, the geometry is represented as a collection of 3D Gaussian, each being the tuple $\{\mathcal{X}, \Sigma\}$ where $\mathcal{X} \in \mathbb{R}^3$ is the centroid of the Gaussian and Σ is its covariance matrix in the world frame. This gives the probability density function

$$G(\mathcal{X}, \Sigma) = \exp -\frac{1}{2} \mathcal{X}^\top \Sigma^{-1} \mathcal{X}. \quad (1)$$

Gaussian splatting decomposes it into a scaling matrix \mathbf{S} and a rotation matrix \mathbf{R} via $\Sigma = \mathbf{R} \mathbf{S} \mathbf{S}^\top \mathbf{R}^\top$. The color information in the texture is encoded with a spherical harmonics map $\mathbf{c}_i = \text{SH}_\phi(\mathbf{d}_i)$, which is conditioned on the viewing direction ϕ .

To optimize for features, existing methods tend to append an additional vector $\mathbf{f}_i \in \mathbb{R}^d$ to each Gaussian, which is rendered in a view-independent manner because the semantics of an object shall remain the same regardless of view directions. The rasterization procedure starts with culling the mixture by removing points that lay outside the camera frustum. The remaining Gaussians are projected to the image plane according to the projection matrix \mathbf{W} of the camera, which is then sorted from low to high using the distance from the virtual camera origin. This projection also induces the following transformation on the covariance matrix Σ :

$$\Sigma' = \mathbf{J} \mathbf{W} \Sigma \mathbf{W}^\top \mathbf{J}^\top, \quad (2)$$

where \mathbf{J} is the Jacobian of the projection matrix \mathbf{W} . We can then render both the color and the visual features with the splatting algorithm:

$$\{\hat{\mathbf{F}}, \hat{\mathbf{C}}\} = \sum_{i \in N} \{\mathbf{f}_i, \mathbf{c}_i\} \cdot \alpha_i \prod_{j=1}^{i-1} (1 - \alpha_j), \quad (3)$$

where α_i is the opacity of the Gaussian conditioned on Σ' and the indices $i \in N$ are in the ascending order determined by their distance to the camera origin.

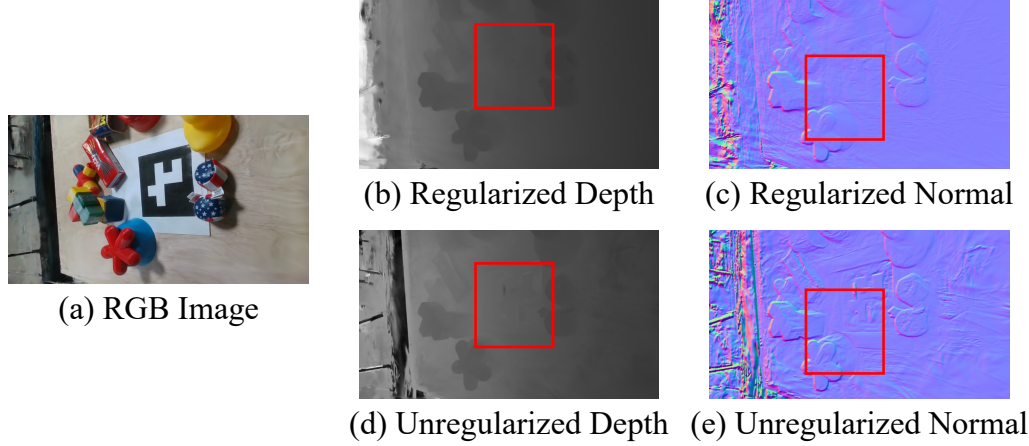


Figure 1: Qualitative results of depth supervision. (a) Rendered RGB image, which is not noticeably different with or without depth supervision. (b) Rendered depth image with depth supervision. Notice the red circle surface is accurately rendered as flat. (c) Rendered normal image with depth supervision. Notice that the object edges are smoother. (d) Unregularized depth, which contains artifacts on flat patterns. (e) Unregularized normal, which is noisy near the object edge.

3 Effect of Depth Supervision

We have provided quantitative evidence in the paper to validate the effectiveness of the depth initialization in training. For the other aspect of geometric regularization, depth supervision during the training, we provide qualitative samples in Fig. 1.

4 Failure Case

In tasks involving static scenes, aside from a few cases where objects are very similar and difficult to distinguish using text queries, resulting in segmentation failures, the system can effectively segment the objects of interest. Most failures in tasks are due to execution issues, such as collisions with surrounding objects when lifting the object, causing it to drop. These require more complex planning and are outside the scope of our research. Notably, although the system can effectively segment objects, when objects are closely packed, the generated grasp points might include other objects as is shown in Fig. 2, which will be addressed through further engineering improvements.

Additionally, In our demonstration, we included a placement task where the object is placed down with the same orientation as it was grasped. While this is reasonable if the object remains stable during grasping, in reality, the object often rotates to some extent. The placement task thus requires perception of the object’s orientation during grasping, which is complicated by the gripper obstructing the view of the object. This presents a complex challenge.

In dynamic scenes, the success rate of grasping operations primarily hinges on the accuracy of tracking. Objects with complex textures are easier to track due to the abundance of distinctive visual features, which facilitate consistent keypoint identification even during rotations. Conversely, single-color objects pose a challenge as their keypoints tend to diverge with rotation, making tracking difficult. Additionally, highly asymmetric objects are easier to track because their unique shapes provide reliable visual cues. In contrast, symmetric or nearly symmetric objects are prone to tracking failures during long-term or rapid movements due to the lack of distinctive features. Therefore, ensuring accurate tracking is crucial for improving grasping success rates in dynamic environments.

As shown in Fig. 3, the tracking process of a toy duck, which is predominantly yellow and highly symmetrical, demonstrates several key challenges. The green points indicate invalid keypoints that are either predicted as hidden or not part of the cluster. Over long-term movement, especially with

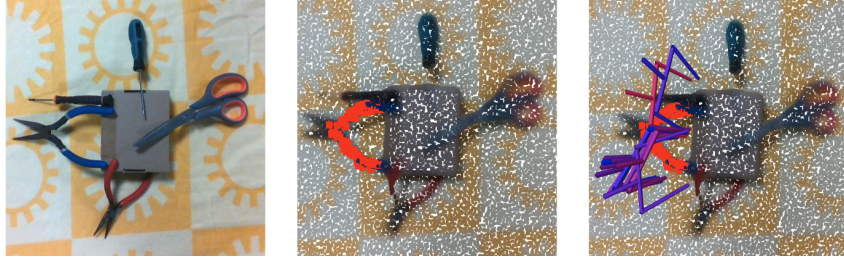


Figure 2: Grasp execution error: The leftmost image is the captured image, the middle shows the segmented grip of the pliers, and the rightmost image displays the generated grasp. The red grasp has the highest score.

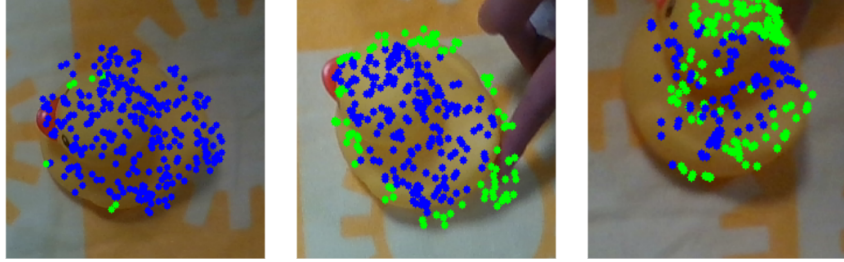


Figure 3: Tracking failure: The leftmost image shows the initially sampled feature points, with blue indicating valid feature points and green indicating lost tracking points. The two images on the right display the feature points' status as the object moves, where valid feature points are continuously lost.

56 rotation, the keypoints are progressively lost. This loss of reliable tracking features leads to track-
 57 ing failures when there are insufficient keypoints to calculate the object's movement. To address
 58 long-term tracking issues, we need to design methods to reacquire the object during task execution,
 59 including re-sampling keypoints on the object. This will be a focus of future research.

60 5 Text Query Comparison

61 The 2D response maps for the same text query generated by different algorithms are displayed in
 62 Fig. 4. Grounded SAM[1] in TrackAnything[2] demonstrates effective segmentation at the object
 63 level; however, it fails to distinguish parts of objects and sometimes does not respond at all. Addi-
 64 tionally, it tends to merge multiple closely positioned objects into one. LERF[3] similarly exhibits
 65 weak responses at the part level, with very unclear segmentation between objects, making it nearly
 66 impossible to separate similar objects that are close together. F3RM[4] provides higher-quality
 67 features compared to LERF and can respond to some part-level queries, but it also struggles with
 68 accurately distinguishing between similar, closely positioned objects. In contrast, our algorithm ex-
 69 hibits near 2D segmentation capabilities and is highly sensitive to part-level queries, allowing for
 70 precise differentiation of different parts within distinct objects.

71 6 GraspNet vs Grasp Sampling

72 In our comparative analysis of grasp sampling methods, we employed GraspNet[5] with collision
 73 detection as LERFTOGO[6], sampling viewpoints on a hemisphere defined by theta and phi pa-
 74 rameters. Specifically, we tested configurations with 3x3, 5x5, and 10x10 viewpoints, effectively
 75 running GraspNet 9, 25, and 100 times respectively. The resulting grasps for these configurations
 76 are illustrated in the first three images as is shown in Fig. 5. Notably, the red circles highlight regions
 77 where valid grasps were not generated. In contrast, for our sampling-based method, we configured
 78 the system to sample 3000 points within the workspace, utilizing 16 threads of 12900K for parallel

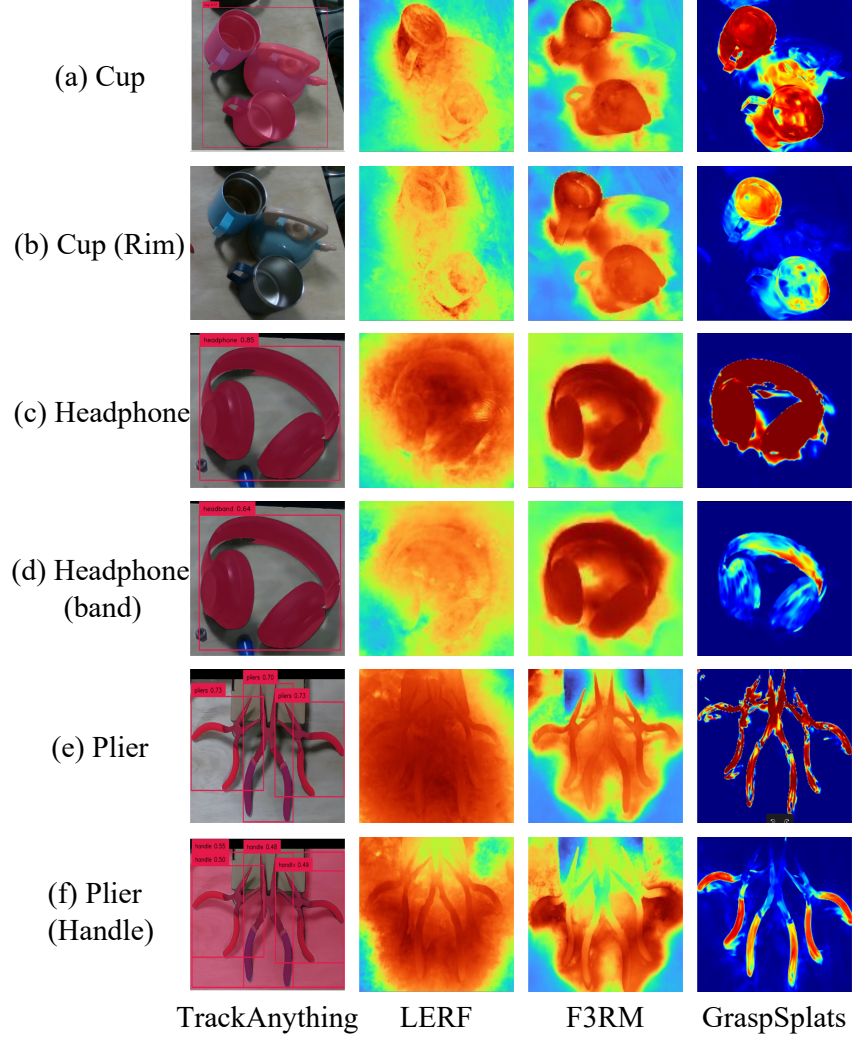


Figure 4: From left to right are the responses of TrackAnything (Grounded SAM), LERF, F3RM, and our algorithm to the same text query.

Method	Query Time↓
GraspNet-100	10.3
GraspNet-25	3.2
GraspNet-9	1.2
Ours	0.5

Table 1: Time efficiency comparison of different grasp sampling methods including ours and GraspNet with different number of executions.

79 processing. The outcome of this approach is depicted in the last image. We also compare the time
80 efficiency of the algorithms as is shown in Tab. 1

81 It is worth noting that while the GS grasps are directly generated with Gaussians, the point cloud
82 data (pcd) is used solely for visualization purposes. This comparison underscores the efficiency and
83 potential limitations of each method in terms of grasp generation and computational demands.

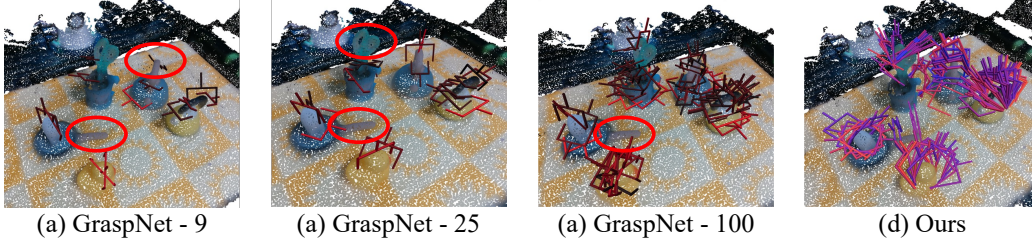


Figure 5: The first three images from left to right show the grasps obtained by running GraspNet with 9, 25, and 100 different viewpoints, respectively. The last image shows the results obtained by our method, which sampled 3000 points in the workspace.

7 Limitation

We present a comprehensive analysis of limitation, or failure modes, of the current implementation of GraspSplats. We hope they will help inspire future research.

- **General Deformation.** The current scene deformation algorithm based on the Kabsch algorithm assumes that objects undergo *rigid* transformation. While the Gaussian representation is conceptually applicable to more general deformable objects (*e.g.*, dough and clay), this is not investigated in the current work.
- **Tracking Failure.** In addition, the tracking is sensitive to fast rotation and the resulting visual occlusions and motion blurs, which could be potentially addressed as an optimization problem using semantic and geometric priors fused in GeFF without assuming consistent object views.
- **Manipulation Policy.** GraspSplats focuses on the grasping of the object parts via language guidance. However, in reality, it can be interesting to incorporate a more complex manipulation policy. For example, though GraspSplats supports the manipulation of articulated object, such as cabinets, by heuristic policy to pull with respect to the normal of the surface, it currently has the quasi-static assumption. Thus, it does not support more complex manipulation. Future work may explore if it is possible to learn more complex manipulation policies on top of the powerful representations of GraspSplats.
- **Requirement for Scene Scanning.** GraspSplats requires scanning of initial objects on the tabletop, where the scanning poses are generated from pre-programmed poses. Future work may include an active reconstruction policy to automate such a process.

References

- [1] T. Ren, S. Liu, A. Zeng, J. Lin, K. Li, H. Cao, J. Chen, X. Huang, Y. Chen, F. Yan, et al. Grounded sam: Assembling open-world models for diverse visual tasks. *arXiv preprint arXiv:2401.14159*, 2024.
- [2] H. K. Cheng, S. W. Oh, B. Price, A. Schwing, and J.-Y. Lee. Tracking anything with decoupled video segmentation. In *ICCV*, 2023.
- [3] J. Kerr, C. M. Kim, K. Goldberg, A. Kanazawa, and M. Tancik. Lerf: Language embedded radiance fields. In *ICCV*, 2023.
- [4] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot language-guided manipulation. In *Conference on Robot Learning*. PMLR, 2023.
- [5] H.-S. Fang, C. Wang, M. Gou, and C. Lu. Graspnet-1billion: A large-scale benchmark for general object grasping. In *CVPR*, 2020.

- 117 [6] A. Rashid, S. Sharma, C. M. Kim, J. Kerr, L. Y. Chen, A. Kanazawa, and K. Goldberg. Lan-
118 guage embedded radiance fields for zero-shot task-oriented grasping. In *Conference on Robot*
119 *Learning*. PMLR, 2023.