Supplementary for TaMMa: Target-driven Multi-subscene Mobile Manipulation

Anonymous Author(s)
Affiliation
Address
`email`

A Setups

In this section, we illustrate the physical setup of the environment, robotic, and the target objects. Moreover, we list the parameters and settings of the employed models in detail.

A.1 Physical Setup

The subscenes taken as an experiment example in our paper is a multi-tabletop scene in a big room. We set up 3 tables of different sizes in the scene, arranged in a triangular pattern, with an average distance of 1 m between the tables, as shown in the diagram. The objects on the tables can be categorized as (1) daily-used containers, such as cups, bottles, plates, baskets, pen holders, and a sink, (2) geometric objects like cubes, (3) challenging objects for grasping, such as rubber toys, irregular toy models, and transparent objects. The arrangement of objects on the tables is relatively random. Regarding the robotic configuration, we employ the Franka Panda Arm as our robotic arm and the RealSense D435 camera, which is fixed at the rear of the robotic arm's end effector and calibrated with the easy-hand-eye package. The gripper is a 3D-printed model with a length of 5 cm. For the mobile base, we employ the SLAMTEC Hermes, equipped with a laser radar for simplified mapping, localization, obstacle avoidance, and navigation.

A.2 Model Setup

Gaussian Reconstruction. For the Gaussian-Splatting-based scene reconstruction, we employ the widely-used open-source 3DGS[1] code base. The initial point clouds used as the Gaussian initialization are downsampled by a factor of 5. The rendering process generates results as the origin image resolution of 640×480. Subsequently, Gaussians are optimized for 30,000 iterations across all scenes, utilizing the same loss function, Gaussian density, schedule, and hyperparameters as specified in the original implementation.

Depth Completion. The depth inpainting process is powered by a diffusion model which is built upon Latent Diffusion Models (LDMs) [2, 3, 4, 5] using a pre-trained Variational Auto-Encoder (VAE) and a U-Net-based[6] denoising architecture. The depth map is repeated at channels to form a tri-channel input as an RGB image and is normalized. A composite feature map is constructed by concatenating the encoded depth and image elements. The denoising step is set to 20 at inference as default to trade off the time consumption and the effect. The U-Net-based denoising architecture iteratively refines the depth latent by predicting and removing noise at each timestep, which is managed by the DDIM[4] scheduler to ensure that noise is progressively reduced in control.

Gaussian Merging and Fine-tuning. With the corresponding camera pose and obtained depth map of the inpainted image, the 2D inpainted data is unprojected into a 3D colored point cloud from image space. Then, features from the original and inpainted Gaussian point clouds are merged by concatenating their poses, features, and opacities. To remove floaters at the edges of the mask, the minimum number of points within a radius of 0.1 for a point to be considered not an outlier is set to 100 as default. The following fine-tuning process optimizes the model using a combination of L1 loss and D-SSIM (Differentiable Structural Similarity Index) to ensure that the final rendered results closely match the inpainted reference images. The weight parameters are set to 0.8 and 0.2 respectively, reflecting the emphasis on maintaining a balance between pixel-wise accuracy and perceptual similarity. The optimization is performed over 150 iterations to achieve the final Gaussian model.

B Task Videos

We provide videos illustrating our cross-subscene fine-grained manipulation ability on 4 tasks. Each of the interacting objects can be located in the arbitrary sub-scenes.

Pick and Place. The input query is in the form of “Move the [A] to the [B]”, where [A] and [B] are objects in the whole scene. For example, in the video, we set “Move the pink cup to the white plate”, in which objects lie on 2 separate tables.

Stacking. The input query is in the form of “Stack the [A] onto the [B]”, where [A] are objects to be stacked and [B] is the target place. In the video, we show the result of applying “Stack the orange cubes onto the blue cube”. In this case, orange cubes on different tables will be collected and placed on the blue one.

Pouring. The query is in the form of “Pour the liquid in the [A] into the [B]”, where [A] is the container that has liquid in it and [B] is the target container. We show the result of “Pour the liquid in the bottle into the white cup”. In this example, the robotic carries the bottle smoothly to the cup and rotates the bottle to pour.

Tidy-up. The query is in the form of “Tidy up the table with [A]”, where [A] is the representative objects of a subscene. For example, the video shows the result of “Tidy up the table with toys and cups”. In this example, the small objects on the table with toys and cups will be rearranged into a basket.

C Mobile Manipulation

Point Cloud Extraction To extract a scene-wide point cloud for manipulation, we reconstructed the entire scene based on 3D Gaussian Splatting [1] and performed depth completion using diffusions [4, 2]. Starting from the completed depth map, we performed back-projection to obtain point clouds of several sub-scenes. Subsequently, we transformed the point clouds into the robot’s working coordinate system , and executed specific navigation or manipulation tasks based on the transformation matrices of the movement base and end-effector relative to the working coordinate system.

Navigation and Manipulation The navigation process is finished by employing the API provided by the SLAMTEC mobile base. Specifically, a 2D topdown occupancy map is built for the environment as the scene map. Based on the occupancy map, the trajectory planning and obstacles avoidance is completed by querying the map with the target position and the current position. By generating a set of waypoints, the mobile base is guided to the target position. As for manipulation, Franka Panda Arm is operated through the MoveIt! library [7]. We provide the 6 DoF pose of the target and employ the movement API to approach and operate the gripper to close and open depending on the width of the estimated target objects.

Motion Planning based on 6D Pose Some fine-grained robotic manipulations mainly rely on the 6D pose of the target for motion planning [8]. Based on the poses of different category-level objects and the tasks to be performed (e.g., pouring, stack), we utilized MoveIt! [7] to set up target-driven action sequences. Therefore, the success rate of robotic manipulation heavily depends on the accuracy of pose estimation. Fig. C1 presents the qualitative results of pose estimation across different scenes. The top row of Fig. C1 demonstrates the excellent poses in most cases. However, there are instances where the results are suboptimal, often due to inaccurate depth data. This can be observed in the

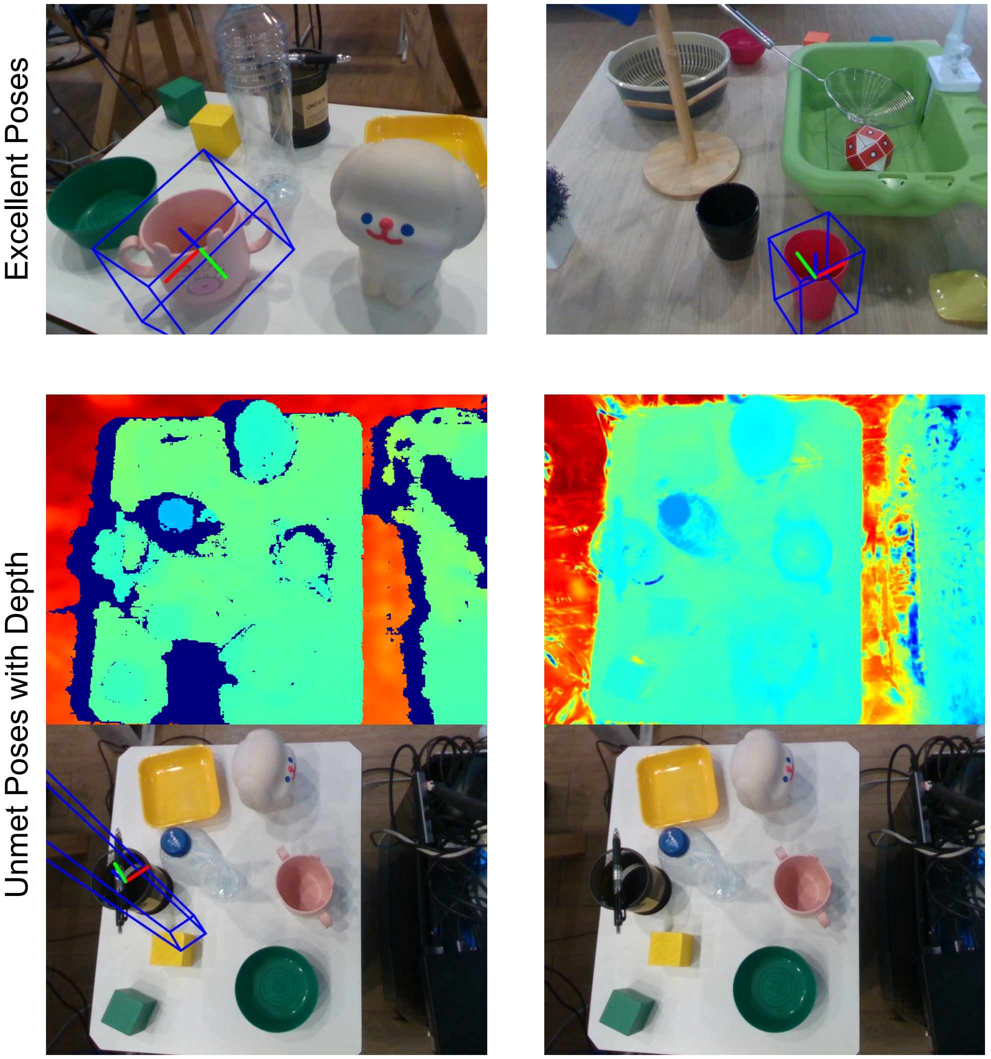

Figure C1: The qualitative results of pose estimation under different scenes. The top row shows the excellent poses under the corresponding views. The two bottom rows show unmet poses with depth. In these bottom rows, the left column presents the original depths acquired by the depth camera, while the right column shows the depths completed using our method.

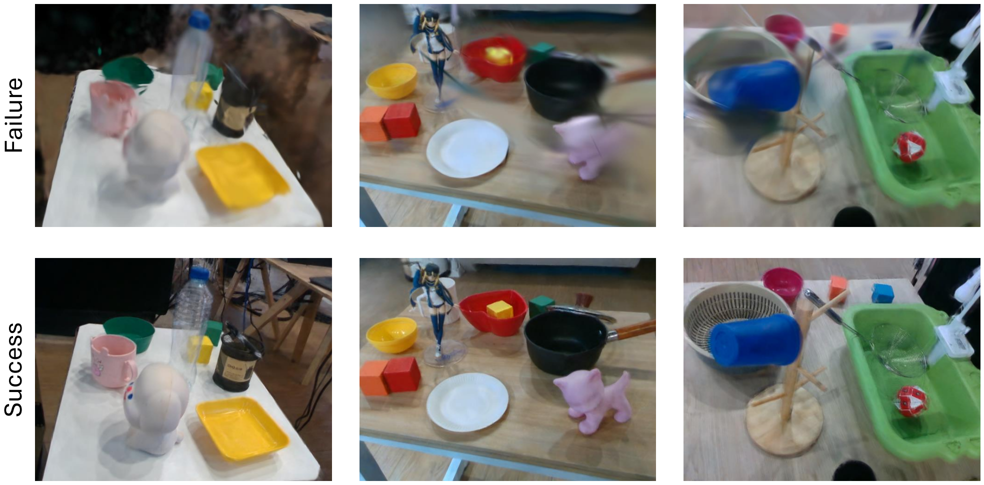

Figure D1: The comparison of reconstruction failure caused by camera jittering and the successful results.

bottom two rows. When we attempt to acquire the pose of the ”black pen,” the original depth (left column) and the completed depth (right column) are both unsatisfactory, affecting the subsequent pose estimation process.

D Failure Cases

Capture Failures. Accurate camera pose is crucial for scene reconstruction. During the experimentation process, we discovered that the jittering and offset errors of the mobile base and robotic arm could result in inaccurate camera poses used for scene reconstruction. Consequently, this leads to errors in the initialization and optimization processes of Gaussians, resulting in issues such as ghosting in the reconstructed scene. One possible solution is to reduce the movement speed of the base and robotic arm, capture data only when they are stable, and minimize frequent movements of the chassis as much as possible. The visualization of comparison is shown in Fig. D1.

Segmentation Failures. Our experiments have shown that when using a segmentation model based on SAM[9] as a mask for image inpainting and depth completion, the inaccuracy of the mask can result in incomplete object editing, leaving behind edges or blurry traces. A more precise mask can achieve better visual effects. The comparison inpainting results between using inaccurate masks and refined masks are shown in Fig. D2

E Detailed Illustration of Limitations

Monocular Depth Estimation In practical applications of robotic arm manipulation, it is common to use the depth values directly captured by a depth camera as the input for scene modeling and object pose estimation. This is because the depth values provided by a depth camera are absolute depth values, which differ from the relative depth typically obtained through monocular depth estimation. This absolute depth information is highly valuable for practical operations. However, depth camera captured results often exhibit errors and significant uncertainty, particularly at the edges of the camera’s field of view and when dealing with transparent or reflective objects. These factors raise issues in performing operations in challenging scenarios. In this paper, we utilize a diffusion-based depth completion method to partially address the discontinuities and errors present in the depth camera results. However, the completed depth results still contain certain errors compared to the original absolute depth input. How to leverage depth maps from multiple viewpoints to supplement depth maps from a single viewpoint and obtain a dense and reliable point cloud remains an open question.

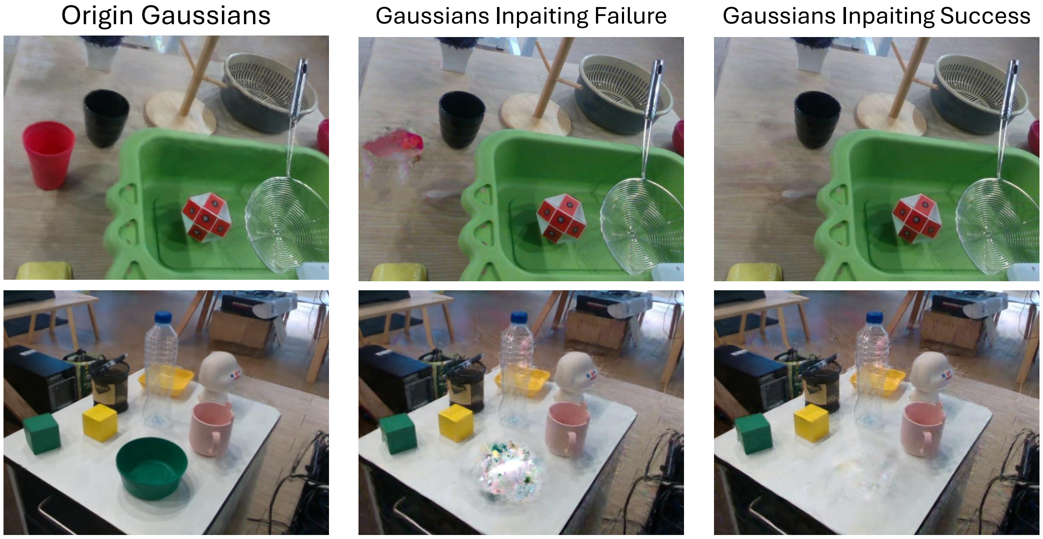

Figure D2: The comparison of inpainting failures caused by inaccurate masks and the successfully inpainted Gaussians results.

What's more, the object occlusion may also cause depth completion failure. As mentioned above, even the diffusion-based method is also unable to solve the depth prediction problem for objects with severe occlusion. Additionally, for transparent objects, objects located behind them can also affect depth and pose estimation. To manage this issue in 3D space, amodal-based method will be our future work direction.

3D Objects Inpainting Calculating precise poses for robotic arm manipulation has always been an open and challenging problem, especially in scenarios lacking constraints. For example, it is difficult for a robotic arm to place a cup in the exact middle of the plate to achieve the same level of precision as a human. Similarly, stacking blocks together is challenging for a robotic arm to achieve a perfect appearance. These results stem from two main reasons: the difficulty of endowing the robotic arm with real-time adjustment capabilities and the inability of the arm to obtain accurate pose information of the target location. Through a comparison with existing successful robotic arm grasping tasks, we observed that while the estimation of graspable object poses can now yield fairly accurate results, it remains challenging to estimate the precise poses required for tasks such as placing a cup in the center of a plate or achieving perfect block stacking. Consequently, we propose using inpainting techniques, employing a ”think before you do it” approach, to address this issue. We first edit the images and depth maps, followed by editing Gaussians to create a 3D representation of the desired task-completed scenario. We then estimate the poses of the target objects in this scene to obtain a more accurate target pose for interaction. However, implementing this approach still presents difficulties. The current Gaussians inpainting methods perform well for editing relatively planar objects but struggle with editing voluminous and complex-shaped objects. Therefore, our future research direction will focuse on how to edit objects in 3D space to obtain accurate results that can be used for interactive operations.

F Implementation Details of Comparison Methods

For the cross-subscene mobile manipulation task, only a few works have been open-sourced. In this paper, we choose F3rm[10] and HomeRobot[11] as the comparison methods. To enable them with the cross-subscene ability in our environment, we re-implement and fine-tune these methods for the aiming tasks.

F3rm. We implement F3rm to expand its workspace to the cross-subscene environment. For the data source, the same images from RealSense D435 are used as input, and to be fair, the depth images are also employed to train the implicit representation as an additional loss. The camera poses come from the calibrated camera on the robotic arm. As the environment expands, the need for VRAM increases obviously. Fairly, we maximize the VRAM usage of the RTX 3090 by employing a smaller feature resolution, as mentioned in F3rm, and ignoring the regions out of the tabletop workspace. As for the manipulation process, we first drive the mobile base to ensure the manipulation targets are reachable for the robotic arm and optimize the target pose for manipulation.

HomeRobot. We implement HomeRobot in our environment for the cross-subscene task. The provided Detic and Grounded-SAM libraries are employed to get semantics. The exploration of the environment is replaced by feeding the recorded RGB-D sequences to the HomeRobot, and the navigation and planning of the mobile base is implemented in the same way as our proposed *TaMMa*. For mobile manipulation, the mobile base is first guided to approach the target, ensuring the targets are reachable, and the Contact Graspnet is employed to generate the 6 DoF grasp pose of the target. The pose of the receptacle, the location on which the object is placed, comes from adjusting from a reference object. For example, to put the cup on a plate, we optimize the pose of the plate and add a bias on the y-axis to get the target pose of the cup.

References

[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis. 3d gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics*, 42(4), July 2023. URL https://repo-sam.inria.fr/fungraph/3d-gaussian-splatting/.

[2] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[3] A. Blattmann, R. Rombach, H. Ling, T. Dockhorn, S. W. Kim, S. Fidler, and K. Kreis. Align your latents: High-resolution video synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22563–22575, 2023.

[4] J. Ho, A. Jain, and P. Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arxiv:2006.11239*, 2020.

[5] D. Podell, Z. English, K. Lacey, A. Blattmann, T. Dockhorn, J. Müller, J. Penna, and R. Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis. *arXiv preprint arXiv:2307.01952*, 2023.

[6] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*, pages 234–241. Springer, 2015.

[7] S. Chitta, I. Sucan, and S. Cousins. Moveit![ros topics]. *IEEE Robotics & Automation Magazine*, 19(1):18–19, 2012.

[8] S. Stevšić, S. Christen, and O. Hilliges. Learning to assemble: Estimating 6d poses for robotic object-object manipulation. *IEEE Robotics and Automation Letters*, 5(2):1159–1166, 2020.

[9] T. Wang, Y. Li, H. Lin, X. Xue, and Y. Fu. Wall-e: Embodied robotic waiter load lifting with large language model. *arXiv preprint arXiv:2308.15962*, 2023.

[10] W. Shen, G. Yang, A. Yu, J. Wong, L. P. Kaelbling, and P. Isola. Distilled feature fields enable few-shot language-guided manipulation. *arXiv preprint arXiv:2308.07931*, 2023.

[11] S. Yenamandra, A. Ramachandran, K. Yadav, A. Wang, M. Khanna, T. Gervet, T.-Y. Yang, V. Jain, A. W. Clegg, J. Turner, et al. Homerobot: Open-vocabulary mobile manipulation. *arXiv preprint arXiv:2306.11565*, 2023.