UNIVERSITY OF CALIFORNIA SAN DIEGO

Scene Reconstruction for Simulated Grasp Search in Structured Clutter

A thesis submitted in partial satisfaction of the
requirements for the degree
Master of Science

in

Computer Science

by

Runlin Guo

Committee in charge:

Professor Henrik I Christensen, Chair
Professor Manmohan Krishna Chandraker
Professor Hao Su

2022

The thesis of Runlin Guo is approved, and it is acceptable in quality and form for publication on microfilm and electronically.

University of California San Diego

2022

DEDICATION

To my dear parents

and Selene

for their unconditional love, trust, and support.

EPIGRAPH

Stay hungry, stay foolish

—Steve Jobs

The important thing is not to stop questioning. Curiosity has its own reason for existence. One cannot help but be in awe when he contemplates the mysteries of eternity, of life, of the marvelous structure of reality. It is enough if one tries merely to comprehend a little of this mystery every day.

—Albert Einstein

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGEMENTS

I am very fortunate to meet Prof. Henrik Christensen during my first quarter at UCSD. His eye-opening experiences greatly motivated me to work in the robotics field and the fundamentals learned from him helped me build a solid understanding of robotics. As a beginner in robotics research, I am grateful to join his lab and work on exciting problems. Later, I am so fortunate to meet Prof. Hao Su, who is a brave researcher with great curiosity. His deep understanding and precise opinion of 3D computer vision helped me to focus on the important aspects of my work. From both professors, I have learned to be passionate, dedicated, and ambitious in research. My interactions with them will benefit me lifelong.

I am very lucky to know many great lab members. In particular, Quan Vuong helped me to get started in robotic manipulation and taught me the importance of staying organized. Yuzhe Qin gave many insightful suggestions for my work. Fanbo Xiang, Minghua Liu, Jiayuan Gu, Jiaming Hu, Yiding Qiu, Rui Chen, and other lab members provided me support and their work greatly enlarged my research perspective.

I would also like to acknowledge Prof. Manmohan Chandraker for his help and suggestion for my thesis. I want to give thanks to many exceptional professors at UCSD, such as Prof. David Kriegman, Prof. Chung-Kuan Cheng, and Prof. Ben Ochoa, who have taught me invaluable knowledge about computer vision and optimization. Moreover, I am grateful to read research works from brilliant researchers, including but not limited to Prof. Ken Goldberg, Prof. Pieter Abbeel, Prof. Dieter Fox, Prof. Leonidas Guibas, Prof. Fei-Fei Li, Prof. Jian Sun, Prof. Li Yi, and Prof. He Wang. Their outstanding works are delightful to read and greatly motivate me to keep on pursuing high-impact research on fundamental problems.

This thesis, in part, is a coauthored unpublished material with Quan Vuong, Yuzhe Qin, Luobin Wang, Hao Su, Henrik Christensen. The thesis author is the primary investigator and author of the material appeared in this thesis.

VITA

2016-2020	B. S. in Computer Engineering and Electrical Engineering *summa cum laude*, University of California, Davis
2020-2022	M. S. in Computer Science, University of California San Diego

ABSTRACT OF THE THESIS

Scene Reconstruction for Simulated Grasp Search in Structured Clutter

by

Runlin Guo

Master of Science in Computer Science

University of California San Diego, 2022

Professor Henrik I Christensen, Chair

Robotic grasping in a complex environment is one of the fundamental challenges for home-assistant robots. Complex environment grasping has been extensively studied in industrial bin-picking scenarios, where reliably grasping objects from unorganized heaps is challenging due to sensor noise, obstructions, and occlusions. However, bin picking is still relatively easier than grasping common household objects from a structured clutter in a home environment because the robot cannot knock over neighboring objects during the grasping motion. Recently, there have been several attempts to tackle the grasping-in-structured-clutter problem. In our experiments, we found these methods either hard to adapt to our simulated environment without extra tuning or generate too few stable grasps to successfully grasp the objects. The overviews and detailed

analyses of these existing grasping approaches will appear in the first half of this thesis.

In the second half of this thesis, we investigate the idea of using a physical simulator as an intermediate step to generate a grasp trajectory proposal. At a high level, we propose a two-step approach to solve the grasping-in-structured-clutter problem. First, we collect RGB-D observations to reconstruct the environment in a physical simulator via 9 degree-of-freedom (DoF) category-level object pose estimation, CAD model matching, and physical support refinement. Then, we perform antipodal grasp sampling, collision-free motion planning, and grasp execution in the simulator and directly transfer the robot arm's motion trajectory to the original environment.

To generate a 9-DoF category-level object pose estimate, we extend a state-of-the-art 6-DoF instance-level object pose estimation network. In our experiments, we found the 9-DoF pose estimation network can reach performance comparable to the state-of-the-art on a category-level object pose estimation dataset. Relying on only the top-down view of the environment, we reconstructed the environment using the proposed two-step approach and evaluated the grasp transfer success. The results show further room for improvements in the model matching process. Future directions and some ideas will be discussed towards the end of this thesis.

We hope the work of scene reconstruction for simulated grasp search and trajectory transfer will help future research of robotics manipulation in complex environments.

Chapter 1

Introduction

Human beings develop the ability to perceive the environment and the coordination to grasp and pick up objects during the first 12 months after born. At birth, infants possess five major sensory systems: vision, audition (hearing), olfaction (smell), somatosensation (touch), and gustation (taste). Among these, vision is the least developed sense at birth [Rym14]. As infants' vision system matures, they gradually develop eye-tracking, color and depth perception, and hand-eye coordination. Combining with the sense of touch, infants become able to grasp and manipulate objects.

For robots, there are currently three major sensor modalities: visual observations from cameras, sound from microphones, and pressure from contact force sensors. At the time of this thesis, contact force sensors are mostly either very expensive and industry-oriented or cheap but provide only a low-resolution sensing area. Voice commands require speech recognition and natural language processing which recently has impressive breakthroughs thanks to vast language datasets and large language models such as GPT-3. However, obtaining adequate computational resources for even running these large language models is challenging in an academic setting. In comparison, visual observations are easier to collect from RGB-D cameras or simulated renders. Moreover, many challenges of teaching a robot to perceive and understand the world around itself

still remain unsolved. In this thesis, we investigate some challenges of learning to grasp a target object in a complex environment with a fixed-base robot arm given RGB-D observations.

Robot grasping in a complex environment has been studied extensively in industrial bin-picking scenarios [MLN$^+$17, MMS$^+$19, MCL19]. Although reliably grasping objects dissimilar in shape from unorganized heaps is challenging because of sensor noise, obstructions, and occlusions, bin-picking is still relatively an easy robot manipulation task. A large body of bin-picking works have focused on solving 3/4 degree-of-freedom (DoF) grasping, where the gripper is approaching objects from above [MPH$^+$16, MLN$^+$17, MCL19]. This vertical approaching assumption is valid in the bin-picking scenarios and greatly simplifies the grasp pose search space. However, if the target object is a horizontally placed plate, then such top-down grasping can not succeed. The ability to grasp more household objects with a larger search space for grasp planning motivates the focus of 6-DoF grasping. On the other hand, bin-picking considers unorganized objects randomly stacked in a pile where collisions with neighboring objects during grasping are allowed. However, for home-assistant robots, the common scenarios are grasping from a *structured* clutter (*i.e.*, packed configurations of objects). Kitchen cupboards or supermarket shelves are some examples. Objects in a structured clutter have fewer stable configurations because they rely only on the underlying surface (or other objects if object stacking is allowed) directly to provide support. This is different from bin-picking scenarios where objects can have support coming from neighboring objects or bin enclosure surfaces. Because of the fewer stable poses, collisions with neighboring objects during grasping can have more catastrophic effects including knocking over objects, breaking them, and hurting people nearby. Thus, avoiding collision becomes more important for grasping objects in a structured clutter.

For testing the grasping-in-structured-clutter problem, we focused on a typical kitchen table-top scenario and created a set of clutter scenes where each scene contains 5 to 10 objects. The object dataset consists of 4 categories, including bottles, bowls, cans, and mugs, with a total of around 550 unique model instances. We will use this table-top grasping environment as our

grasping-in-structured-clutter task to perform all of our experiments throughout this thesis.

Several recent works have also studied this grasping-in-structured-clutter scenario. We carefully examined their performances in our table-top clutter scenes inside a physical simulator. Overviews and detailed analyses of these existing approaches will appear in Chapter 2. In our experiments, we found these methods either hard to adapt to our simulated environment without extra tuning or generate too few stable grasps to successfully grasp the objects. In Section 2.6, we also include discussions of a recent task-and-motion-planning (TAMP) approach [LK19] that claimed to reliably grasp objects in a structured clutter.

After our experiments and analyses, we found that a successful grasp from a structured clutter not only depends on a complete accurate scene object mapping and high-quality grasp pose generation, but also relies on grasp trajectory planning and robot configuration space feasibility. In addition, data-driven grasp proposal networks are often trained on known objects with annotated grasp poses in simulation. Thus, a pretrained grasp proposal network is difficult to generate successful grasp proposals when testing on unseen objects and different object arrangements in the scene. To incorporate the full grasp trajectory and whole-scene information into the grasp search process, we think a possible solution is to reconstruct the scene in a simulator from collected observations, leverage physical simulation to evaluate grasp proposals, and finally transfer the robot's joint action trajectory back to the original scene to grasp the target object. Our detailed method and experiments will appear in Chapter 3 of this thesis.

For the scene reconstruction for grasping pipeline, we first examined a recent approach to reconstruct interactive 3D scenes in a simulator by panoptic mapping and CAD model alignments [HZJ+21]. All of their experiments were done in a room setting where most objects are large furniture (*e.g.*, tables, refrigerators, and microwaves). Because their model matching and alignment process is only based on RANSAC extracted planes from the surface of the observed objects, we found it does not perform well on smaller objects (*e.g.*, mugs, bottles, and bowls) whose geometries mostly consist of spherical and cylindrical surfaces. Despite this naive plane-based matching

approach, using the reconstructed scene from their pipeline and a simple but effective brute-force grasp sampling technique, we can already achieve a slightly higher grasping-in-structured-clutter success rate compared to current grasping state-of-the-art approaches. This motivates the research direction of interactive scene reconstruction at higher geometric accuracy.

To better reconstruct the geometry of the original scene in a physical simulator from RGB-D observations, we proposed to use a two-step approach. First, we collected RGB-D observations and performed 9-DoF category-level pose estimations of the objects in the scene. In our experiments, we demonstrated that converting 6-DoF instance-level keypoint-voting-based pose estimation into 9-DoF category-level pose estimation can produce accurate estimates by simply choosing the voted key points to be the 8 corners of the object's oriented bounding box (OBB). By extending a state-of-the-art 6-DoF instance-level object pose estimation network, we achieved comparable performance as the state-of-the-art on a category-level object pose estimation dataset. Currently, the bottleneck of our pose estimation accuracy is the semantic segmentation of observed object points. If we substitute the predicted semantic labels with the ground-truth labels, our pose estimation accuracy surpasses the state-of-the-art in the degree/cm average precision (AP) metric by almost 10%. Future plans to improve the semantic segmentation results are discussed in Section 4.1.

Given the 9-DoF category-level object pose estimations, the second step in our scene reconstruction pipeline is to match the observed object points with a CAD model from the model database while ensuring scene stability (*i.e.*, object support is adequate to not fall over and no neighboring object model mesh penetration occur). For each object model in the predicted object category dataset, we applied the predicted 9-DoF transformation to the object model and uniformly sampled points on the model surface. By computing Chamfer distance between the observed object points and the sampled model points, we selected the most geometrically similar object model. To help refine the table-object support and eliminate table penetration, we use RANSAC to estimate the supporting tabletop surface plane for the objects and refine the object

models' gravity-aligned axis position. Finally, we used the reconstructed scene and brute-force grasp sampling in a simulator to generate a grasping trajectory and then transferred the action trajectory back to the original scene.

The main contributions of this thesis include (1) detailed analyses of existing grasping-in-structured-clutter approaches; (2) the proposed two-step scene reconstruction pipeline for simulated grasp planning; (3) the OBB keypoint selection technique to convert any existing keypoint-based instance-level 6-DoF pose estimation network into a category-level 9-DoF pose estimation network; (4) state-of-the-art category-level pose estimation accuracy on an existing dataset and 10% higher than state-of-the-art degree/cm AP when given ground-truth semantic labels. Future plans to improve are discussed in Chapter 4.

Chapter 2

Baseline Approaches for Grasping in Structured Clutter

Grasping is one of the most fundamental challenges in robotics and how to reliably grasp objects in a complex environment remains an open problem. In this chapter, we will focus on grasping objects in a structured clutter and benchmark several existing baseline approaches.

2.1 Related work

Camera Viewpoint Selection Viewpoint selection or active exploration (perception) is an important aspect of vision-based grasp pose generation, especially when grasping objects in heavy occlusions or completely obstructed behind clutter. Given current observation from a camera mounted on the robot gripper, the robot plans and moves its arm to collect observation from a different camera viewpoint where previously occluded regions can be observed. By repeating this exploration process, the robot can collect a more complete geometry of the scene and use it for planning a collision-free grasp trajectory.

Viewpoint selection has been used on mobile manipulators to generate a better envi-

ronment map for navigation and manipulation [LFW+21, LLA21]. Several works have also used active exploration for object detection and pose estimation in a clutter [WRD15, SKLK17, SGHK20]. For robot grasping, [ALRC14] proposed to actively manipulate the target object to accumulate a complete view from multiple viewpoints. [KSP+15] used a trajectory optimization approach to encourage exploration of unseen space in a cluttered environment until a feasible grasp can be found. [MAC16] proposed a spatial attention approach driven by a next-best-view algorithm that computes the most promising sensor viewpoints to observe the detected salient regions. Recently, [MCL19] proposed a Multi-View Picking controller that uses active perception to choose informative viewpoints based on bin-picking grasp pose estimates, reducing the uncertainty in the grasp poses caused by clutter. [FUU+20] designed a reinforcement learning-based system that takes in an arbitrary-pose RGB image of the desired target object with a sequence of RGB wrist camera observations and generates actions consisting of a 4-DoF gripper pose change and a binary open/close gripper command.

To test the active exploration idea, we adapted the trajectory optimization approach proposed in [KSP+15] and evaluated its performance in our table-top grasping-in-structured-clutter environment. We chose this work as one of our baselines because the authors had performed similar experiments of grasping an object in a structured clutter with a 93% success rate. Also, it does not involve deep reinforcement learning networks which require large computational resources to train and much effort to tune the network performance.

Data-driven Model-free Grasping Methods Motivated by the recent success of data-driven approaches in computer vision and scene understanding, more robotics researchers began to think about data-driven grasping methods. To create an object dataset with annotations of grasp poses and corresponding grasp qualities, researchers have used manual human annotations [KBS15], probabilities of grasp pose force closure that are robust to imprecision in perception and control [WA12], and simulation-based grasp annotations under varying friction [EMF21]. With a diverse

object dataset and grasp pose annotations, many data-driven grasping methods achieved reliable grasping success in the bin-picking scenario [MPH+16, MLN+17, ZSY+19].

Grasping involves 3D geometry understanding, physics properties reasoning such as mass and friction, and complex contact physics modeling. On a high-level, grasping has been studied in two main directions: *model-based* grasping where the 3D object models or the object categories are known [CMS11, ZYS+17, MEF19], and *model-free* grasping where there is no prior knowledge about the objects [QCZ+19, MME+20, SMTF21]. For our grasping-in-structured-clutter problem, the challenges largely come from the difficulties of obtaining a complete and accurate scene model. Adapting *model-based* grasping approaches will greatly simplify our task setting and therefore we will look at *model-free* grasping methods.

For our table-top structured-clutter grasping environment, there are several existing data-driven model-free grasping approaches. [QCZ+19] proposed a single-view single-shot 6-DoF grasp proposal network trained with synthetic data and tested in real-world table-top clutter scenes. [MME+20] developed a cascaded grasping framework involving unknown instance segmentation, target object neighborhood point cloud cropping, 6-DoF grasp pose sampling with a conditional Variational Autoencoder, and a clutter-centric CollisionNet to predict grasp pose collision probability. [WFG+21] introduced a "graspness" quality metric based on geometry cues that distinguishes graspable areas in cluttered scenes and developed a two-stage neural network to predict "graspness", sample points with high graspness, and generate 6-DoF grasp poses from the selected most confident viewpoint. [SMTF21] proposed Contact-GraspNet, a grasp pose generation network which takes in a single-view full scene point cloud and generates candidate 6-DoF grasp poses each with a confidence score. Similar to [MME+20], a separate unknown instance segmentation network is used to filter the grasp contacts and a CollisionNet is used to predict collision probability.

To evaluate existing data-driven model-free grasping methods, we adapted and benchmarked the Contact-GraspNet [SMTF21] in our table-top environment. We chose this work as a

baseline because it is the most recent grasping-in-structured-clutter approach that achieves more than 90% success rate in real robot experiments.

2.2 Problem Setting & Environment Overview

In all of our experiments, we focused on a typical kitchen table-top setting where some common household objects are placed on a table. The objective is to grasp and lift the target object out of the structured clutter while not knocking over other neighboring objects. The simulated environment is created using SAPIEN [XQM+20] and one example scene is shown in Figure 2.1.

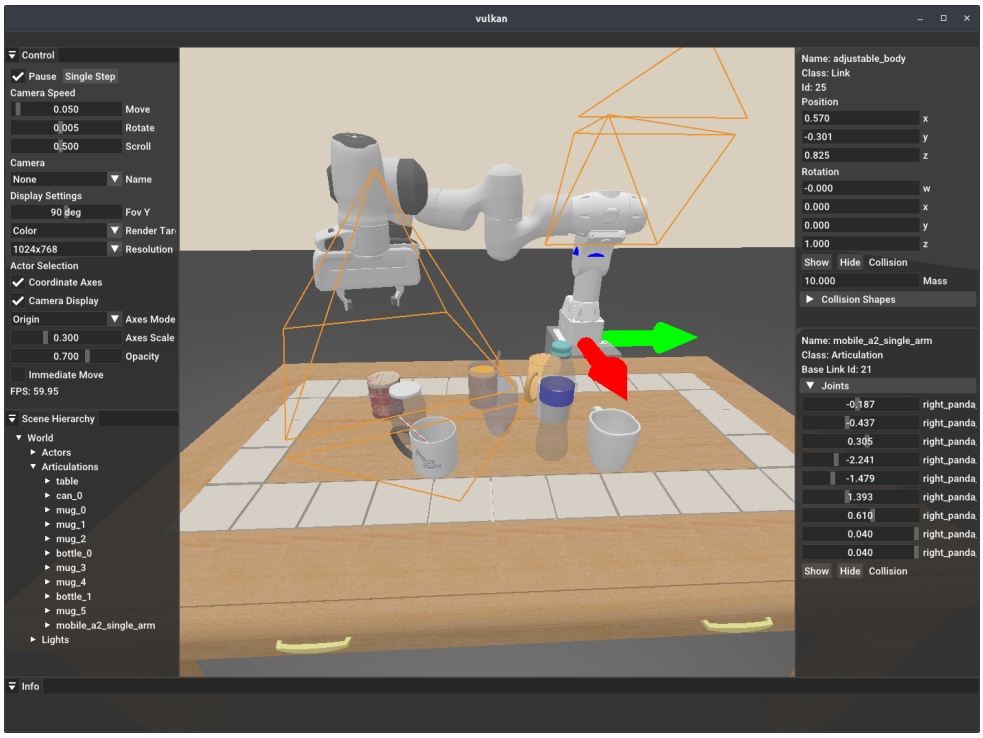


Figure 2.1: One example table-top structured-clutter scene created using SAPIEN. The position and orientation of the cameras attached to the robot arm are illustrated by the orange camera viewing pyramids where the triangles indicate the up direction in the camera frame. $\hat{z}$ axis of the world frame is in the up direction.

In all of our experiments, we will be using the custom fixed-based robot shown in Figure 2.1. It includes a fixed-base upper body from the Sciurus17 robot and a single 7-DoF Franka

Emika's Panda arm with one shoulder camera attached to the fixed base and one gripper camera attached to the arm's gripper. The cameras' position and orientation are illustrated in Figure 2.1.

2.2.1 Object Dataset

Our object dataset consists of 4 categories, including bottle, bowl, can, and mug, with a total of 546 objects. Detailed number of object instances are summarized in Table 2.1. Bottles are articulated models from PartNet-Mobility dataset [MZC+19] while bowls, cans, and mugs are rigid models from ShapeNetCorev2 dataset [CFG+15]. Some examples of object model instances are visualized in Figure 2.2.

Table 2.1: Number of object instances of each category in our dataset

Object Category	Bottle	Bowl	Can	Mug
Number of Instances	57	170	108	211

2.2.2 Table-top Scene

Using the object dataset, we generated 50 table-top object clutter scenes by rejection sampling while ensuring objects can be stably placed within a bounding box at the table center. Each scene contains 5 to 10 randomly selected objects. All objects are standing upright on the table and no stacked objects are allowed (*i.e.*, the table provides direct support to the objects). Some examples of the generated table-top scenes are visualized in Figure 2.3. In total, the 50 table-top scenes include 393 object instances where 240 of them are unique instances (21 bottles, 37 bowls, 81 cans, 101 mugs).

2.2.3 Grasping Observations & Evaluation

Observations When evaluating all grasping baselines in this chapter, we include the following observations: (1) robot joint positions and velocities; (2) RGB-D images from both the robot

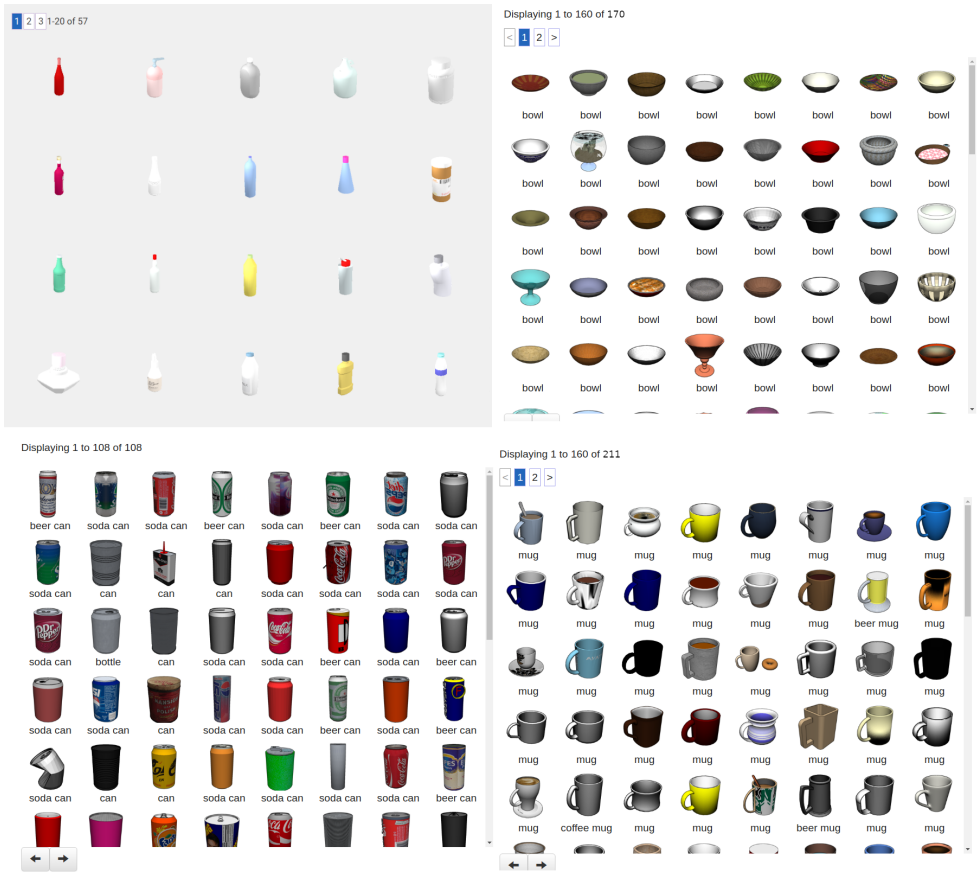


Figure 2.2: Sample object model instances in our dataset

shoulder camera and the gripper camera; (3) ground-truth segmentation masks of the objects and robot arm links for the RGB-D images. The ground-truth segmentation masks are rendered from the simulator.

Grasping Success Condition For a successful grasp in our clutter environment, we require that (1) the target object is lifted (*i.e.*, object z axis position change is > 0.1 meter) and remains static (*i.e.*, magnitude of object velocity is < 0.1 meter per second and magnitude of object angular velocity is < 0.2 radian per second or 11.46 degrees per second); (2) all other objects in the scene have not fallen over due to collision (*i.e.*, angle between the world $\hat{z}$ axis and the object $\hat{z}$ axis is $< 0.1\pi$ radian or 18 degrees). Based on our success conditions, we allow slight collisions with other objects in the scene and only consider a grasp as failed when other objects are knocked over,

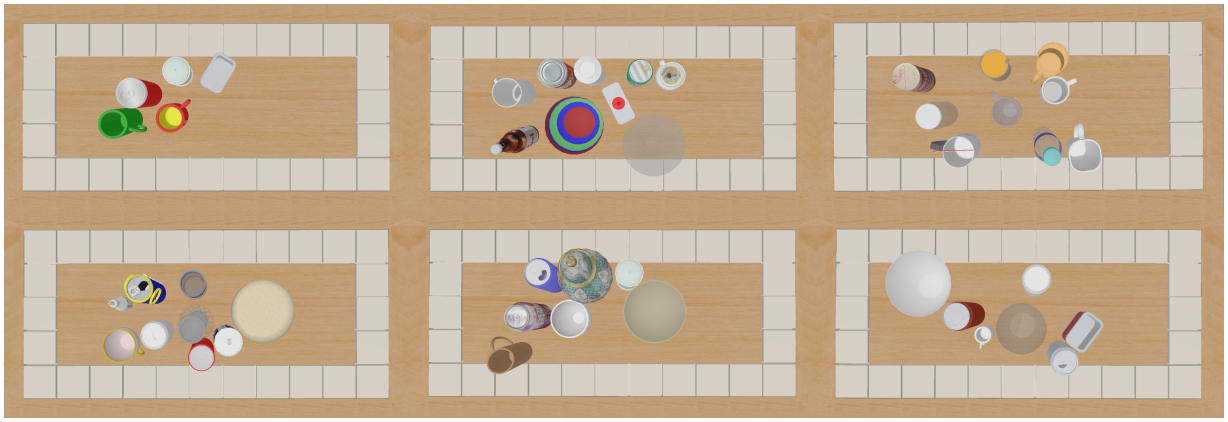


Figure 2.3: Top-down view of sample table-top scenes generated and used for benchmarking grasping performance

which is a much more catastrophic outcome.

2.2.4 Noisy Depth Generation

To better approximate the real depth noise that exists in commercial RGB-D sensors, we adapted SimKinect [HWMD14] to add noise to clean depth images rendered with our simulator. The original Microsoft Kinect for Xbox 360 uses an infrared structured light projector and an infrared camera to capture the projected light pattern. After triangulation over the known codified pattern, the resulting depth images have different kinds of noise [BM13]: (1) the depth and RGB images are often not well-aligned; (2) the disparity recovered by the Kinect is often noisy, presumably due to sensor noise or errors in the Kinect's matching algorithm; (3) the disparity is quantized, which leads to step-like artifacts in depth.

To approximate the three kinds of noise, SimKinect models the depth noise as a combination of three noise models: (1) added Gaussian shifts to pixel coordinates and perform bilinear interpolation [BM13]; (2) disparity filtering with the Kinect light pattern in a 9-by-9 correlation window [BRHS14, GKUP11]; (3) 8-bit quantization to approximate the quantized disparity [BRHS14, GKUP11]. SimKinect has been used to benchmark 3D reconstruction, RGB-D semantic segmentation, simultaneous localization and mapping (SLAM) systems in

[HWMD14, HPB+16] and found it to generate reasonable noise characteristics similar to depth images in real-world datasets. An example of the added depth noise on a clear depth image is shown in Figure 2.4. The original SimKinect implementation sequentially processes image pixels and takes around 2 seconds to process a depth image. We developed a CUDA parallel implementation to greatly decrease the processing time to 16 milliseconds per depth image, although there can be a further decrease if custom CUDA kernels are implemented.

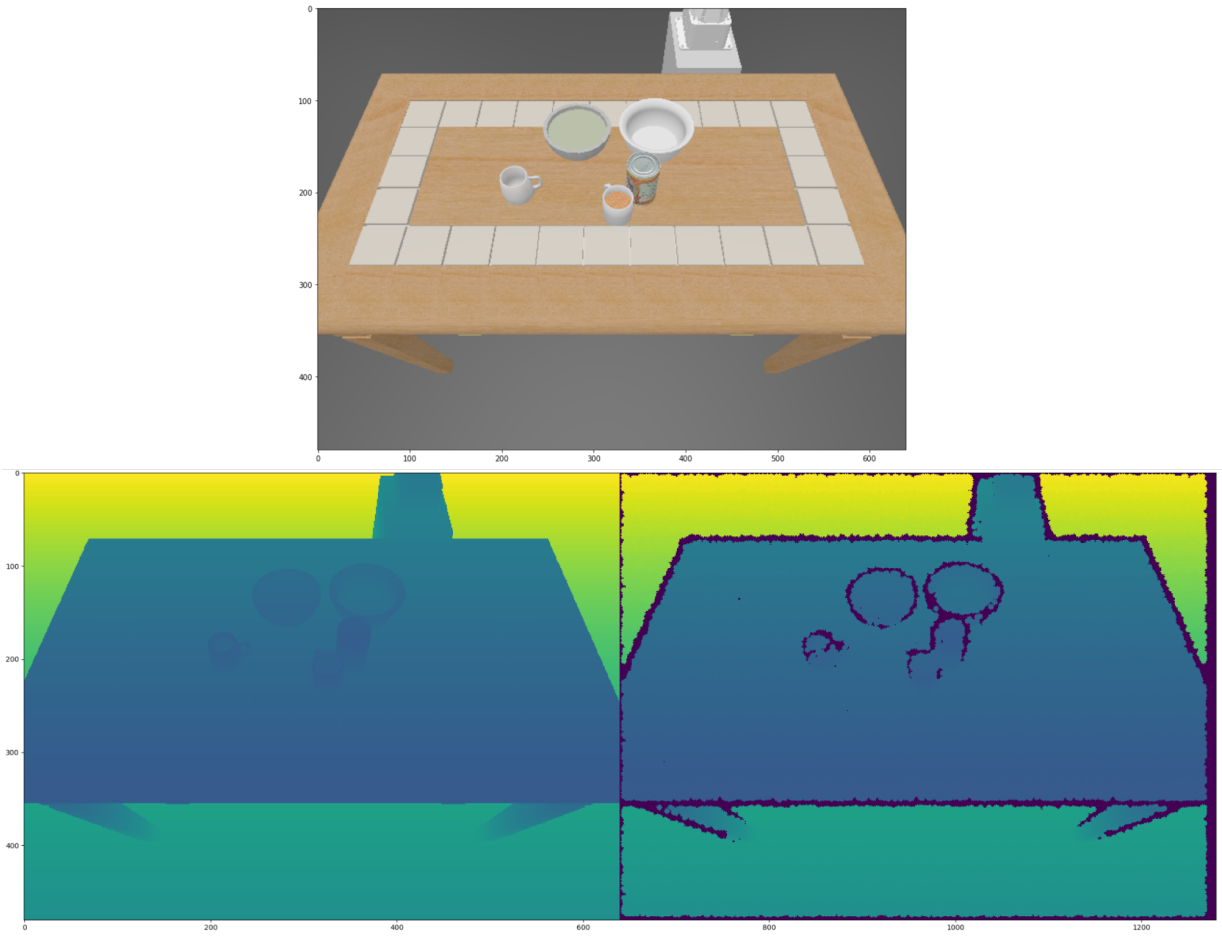


Figure 2.4: SimKinect depth noise visualization. Top-row is the rendered RGB image. Bottom-left is the rendered clean depth image. Bottom-right is the noisy depth image generated by SimKinect.

2.3 Antipodal Grasping Baseline

In this section, we implemented and tested a simple antipodal grasping baseline in our table-top grasping-in-structured-clutter environment. The high-level idea of antipodal grasp selection comes from [Ted21].

2.3.1 Antipodal Grasp Definition

In robotic manipulation, a commonly used friction model is *Coulomb friction*, which is an approximate experimental model. It states that the tangential friction force magnitude f_t is related to the normal force magnitude f_n by $f_t \leq \mu f_n$, where μ is called the *friction coefficient*. If the contact is sliding or just about to begin sliding, then $f_t = \mu f_n$ and the direction of the friction force is opposite to that of the sliding direction. For a contact normal in the $+\hat{z}$-direction, the set of forces that can be transmitted through the contact satisfies $\sqrt{f_x^2 + f_y^2} \leq \mu f_z$ where f_z is the normal force and $f_z \geq 0$. This set of contact forces forms a *friction cone*, shown in Figure 2.5a.

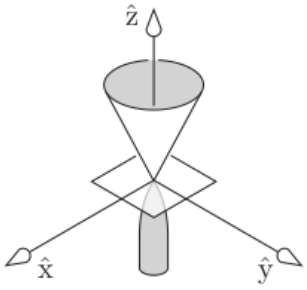
Given a single movable object with several frictional contacts, we say the contacts result in *force closure* if any external forces or torques on the object can be balanced by contact forces inside the friction cones. To increase the probability of creating force closure or at least resist the gravitational wrench (*i.e.*, the spatial force due to gravity) with a parallel gripper, a reasonable strategy is to grasp at *antipodal points* (*i.e.*, the pair of points with normals that are collinear with the line connecting the points and are pointing in opposite directions) [Ted21]. An illustration of an antipodal point pair is shown in Figure 2.5b. The grasp at antipodal points is called an *antipodal grasp*.

In practice, the contact between a parallel gripper and the object is a *patch contact* instead of a point contact due to the fingertip deflection of the gripper and any deflection of the objects being grasped. An example is shown in Figure 2.5c. Thus, we can define a grasping cost function by summing the square of the dot product between contact point normals and the gripper's $\hat{y}$ axis

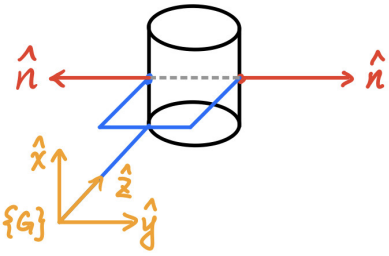
and taking its negation as

$$cost = -\sum_i (\hat{n}_i \cdot \hat{y}_{gripper})^2, \tag{2.1}$$

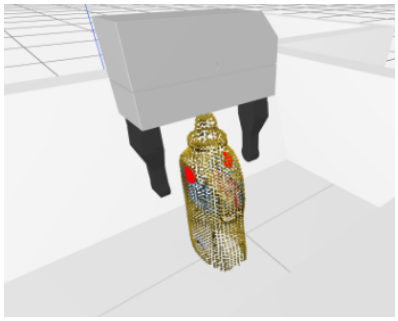
where $\hat{n}_i$ are normals of all object points between the parallel gripper.



(a) Friction cone with normal force in $+\hat{z}$-direction. Taken from Figure 12.18a of [LP17]



(b) Antipodal grasp where red arrows represent normal directions and $\{G\}$ denotes the gripper frame. The parallel gripper is sketched in blue.



(c) Patch contact where contact points between the parallel gripper are shown in red. Taken from Example 5.10 of [Ted21]

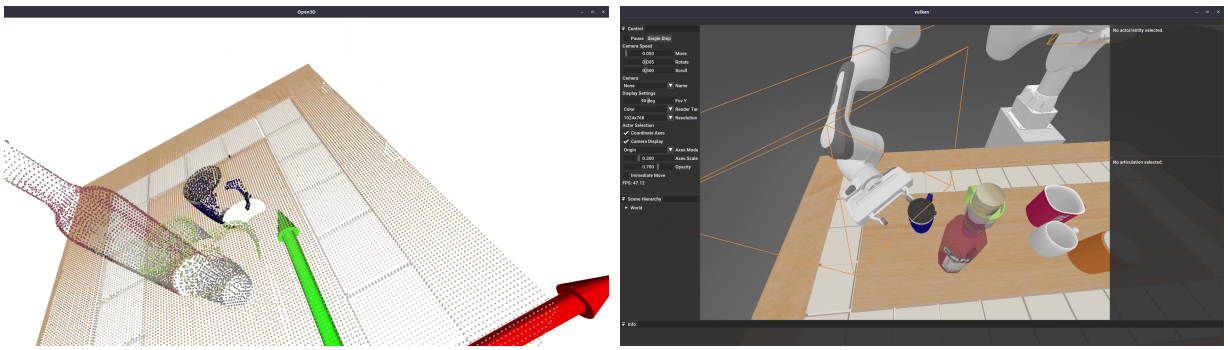
Figure 2.5: Antipodal grasp definitions: friction cone, antipodal grasp, patch contact

2.3.2 Simple Antipodal Grasping Pipeline

Using the antipodal grasping cost function, we implemented a simple grasping pipeline for our table-top grasping problem. Specifically, the pipeline consists of the following six stages: (1) Explore: move the robot gripper towards the target object to accumulate more observations of the scene by simply concatenating observed point cloud, downsampling to 5mm voxels, and estimating point normals using neighboring points; (2) Antipodal grasp sampling: sample antipodal grasps and check for collision with the observed scene point cloud, compute antipodal grasping cost function and select the grasp with lowest cost; (3) Pregrasp: plan and move to the selected grasping pose but 0.2 meter above the target object; (4) Grasp: plan and move to the grasping pose; (5) Lift: close the parallel gripper, plan and move to 0.2 meter above the object; (6) Hold: hold the grasped object in air. Both motion planning and collision checking are done with an open source library, MPlib [Liu21]. Details of MPlib can be found in Appendix A.

2.3.3 Grasping Evaluation Results

To evaluate the simple antipodal grasping pipeline in our 50 table-top object clutter scenes discussed in Section 2.2.2, we used noise-free RGB-D observations rendered directly by our simulator. We found that the overall grasping success rate is **71.25%** and most of the failure cases are due to incomplete object models from partial observations despite the exploration stage of trying to accumulate more observations about the target object model. Thus, based on the incomplete observations, the collision checking module failed to filter antipodal grasps that would lead to collisions. An example of this failure case is shown in Figure 2.6.



(a) Mug body surface around its handle is not observed

(b) The robot gripper attempts to insert and grasp the mug by its body surface

Figure 2.6: Simple antipodal grasping failed to grasp the mug due to the incomplete object model from partial observations. The algorithm mistakenly thought the mug does not have a complete body surface and planned a grasp to insert into its side body.

In addition, some of the sampled antipodal grasps are unstable (Figure 2.7) because the antipodal grasping cost function considers all object points between the parallel gripper which include the extra points on the mug's handle. Actually, when computing the grasping cost function, only gripper contact points should be considered. We leave the task of improving this antipodal grasp sampling process as a potential future work.

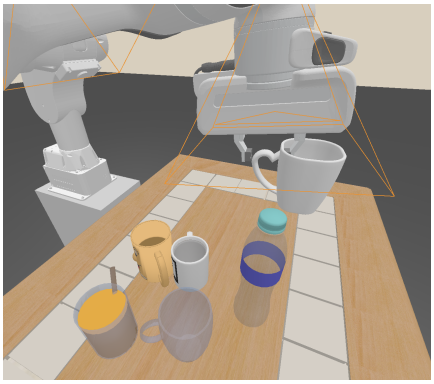


Figure 2.7: Unstable antipodal grasp. All points between the parallel gripper were considered in the grasp cost function.

2.4 Active Exploration with Antipodal Grasping

The evaluation results of the simple antipodal grasping pipeline in the previous section demonstrate the importance of complete scene geometry to the grasp sampling process. In this section, we adapted a trajectory-optimization-based active exploration approach [KSP+15] with our antipodal grasp sampling process and evaluated it in our table-top grasping-in-structured-clutter environment.

2.4.1 Active Exploration using Trajectory Optimization

The active exploration work [KSP+15] aimed to plan the robot arm motion in order to search for feasible grasp handles occluded by object clutters in the initial camera view. They designed an active exploration and grasping system (shown in Figure 2.8a) that continuously updates the environment map from RGB-D observations and performs frontier detection, active exploration, and grasp handle detection. When a grasp handle is detected and a collision-free grasp trajectory is found, the system executes the grasp. Otherwise, the system continues exploring.

Map Construction The map construction used KinectFusion (KinFu) [NIH+11] to fuse the streaming point clouds into a single map. Internally, KinFu represents the environment map using

a truncated signed-distance function (TSDF) where each voxel contains a confidence weight w_i and the truncated signed-distance d_i to the nearest surface. Voxels with $w_i = 0$ are unobserved voxels and voxels with $w_i > 0$ are either in free space ($d_i > 0$) or near a surface ($d_i \approx 0$). Figure 2.8d illustrates a 2D TSDF example.

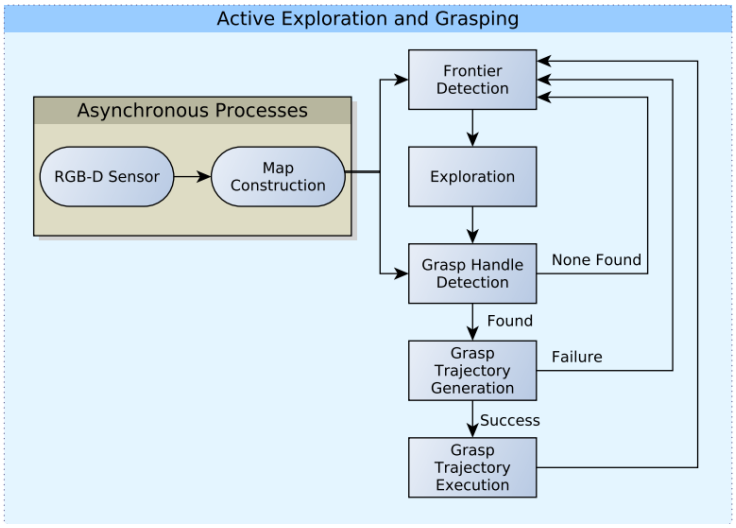
Frontier Detection The active exploration starts with frontier detection. The *frontier* denotes the surfaces that separate known regions and occluded regions. An example is shown in Figure 2.8b and Figure 2.8c. During frontier extraction, the surface voxels and zero-weight voxels are first extracted from the TSDF map. Then, clustering is applied to the surface voxels. For each surface voxel cluster, an oriented bounding box (OBB) is fitted and the extracted frontiers are the OBB faces closest to the current RGB-D sensor position. Finally, the zero-weight voxels occluded by each OBB are parameterized by Gaussian fitting.

Trajectory Optimization Objective Given the current frontiers and M fitted Gaussians $(\mathbf{x}^m, \Sigma_0^m)$ of the occluded regions, researchers in [KSP+15] formulate a discrete-time trajectory optimization problem over time horizon T ($T = 5$ is chosen for our experiments). The current joint state and uncertainty of the robot arm are denoted as $(\mathbf{x}_0^R, \Sigma_0^R)$. Let $\mathbf{u}$ be the joint velocity control input applied to the joint state $\mathbf{x}^R$. To minimize the uncertainty of the occluded regions and penalizes the control effort, they design the cost functions as

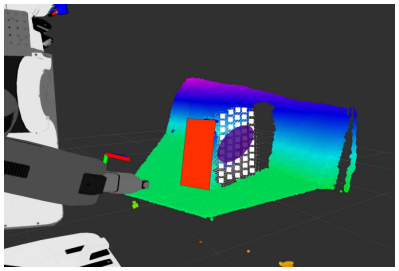
$$c_t(\mathbf{x}_t^R, \Sigma_t^{1:M}, \mathbf{u}_t) = \alpha \|\mathbf{u}_t\|_2^2 + \sum_{m=1}^{M} \beta_t \mathrm{tr}(\Sigma_t^m) \tag{2.2}$$

$$c_T(\mathbf{x}_T^R, \Sigma_T^{1:M}) = \sum_{m=1}^{M} \beta_T \mathrm{tr}(\Sigma_T^m)$$

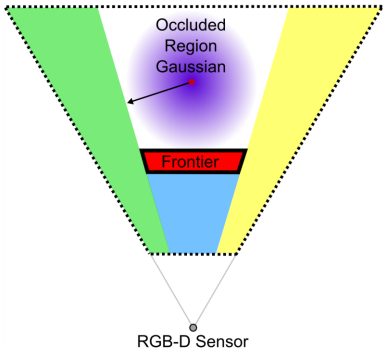
where α and β_t are user-defined scalar weighting parameters.



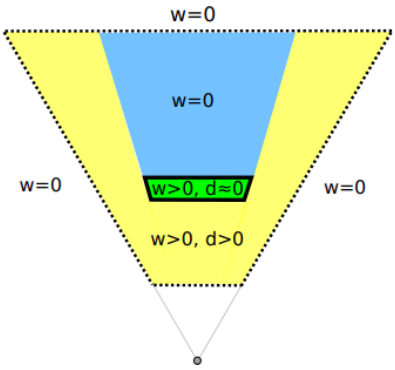
(a) Active exploration and grasping system



(b) Frontier detection



(c) Signed-distance computation



(d) Truncated signed-distance function

Figure 2.8: Active exploration related figures taken from [KSP+15]. (a) Overview of the active exploration and grasping system. (b) The output of the frontier detection algorithm given a single object in the environment. The frontier is represented by the red rectangle and the occluded region voxels are shown in white. The purple ellipsoid denotes the fitted Gaussian to the occluded voxels. (c) A simplified 2D illustration of (b). The RGB-D sensor view frustum is outlined by the dotted line. The view frustum is geometrically truncated against the frontier into the green, blue, and yellow convex regions. The signed-distance is negative if the occluded region mean is inside one of the visible convex regions and positive otherwise. The magnitude of the signed-distance is the distance to the nearest convex region border. The signed-distance of the illustrated occluded region is shown by the black arrow. (d) A 2D illustration of the weights w and truncated signed-distance values d of the KinectFusion (KinFu) truncated signed-distance function (TSDF). The RGB-D sensor view frustum is outlined by the dotted line and the frontier (observed surface) is shown in green. The voxels in the yellow region are known free space while the voxels in the blue region are the unobserved region.

The optimization objective is then

$$\min_{\mathbf{x}^R_{0:T},\mathbf{u}_{0:T-1}} \mathbf{E}\left[c_T(\mathbf{x}^R_T,\Sigma^{1:M}_T)+\sum_{t=0}^{T-1} c_t(\mathbf{x}^R_t,\Sigma^{1:M}_t,\mathbf{u}_t)\right] \tag{2.3}$$

$$\text{s.t.} \quad \mathbf{x}^R_{t+1}=\mathbf{x}^R_t+(\mathbf{u}_t+\mathbf{q}_t)dt$$

$$\mathbf{x}^R_t \in \mathcal{X}_{feasible},\ \mathbf{u}_t \in \mathcal{U}_{feasible}$$

$$\mathbf{x}^R_0=\mathbf{x}^R_{init},\ \Sigma^m_0=\Sigma^m_{init}$$

where $\mathbf{q}_t$ is the robot dynamics noise.

Uncertainty Model Given the current system belief $(\mathbf{x}^R_t,\Sigma^R_t)$, occluded region beliefs $(\mathbf{x}^m,\Sigma^m_t)$, control input $\mathbf{u}_t$, and measurements $\mathbf{z}^m_{t+1}$, the occluded region beliefs are evolved using an extended Kalman filter (EKF). In this work, they assumed the occluded regions are independent and thus their covariances evolve separately. Also, a static environment is assumed and the occluded region means $\mathbf{x}^m$ do not evolve in the EKF.

2.4.2 Grasping Evaluation Results

Grasping Evaluation Details To adapt the trajectory-optimization-based active exploration approach in our table-top grasping environment, we substituted the grasp handle detection and grasp trajectory generation modules (shown in Figure 2.8a) with our simple antipodal grasp sampling process described in Section 2.3.2. Also, we added an extra antipodal grasp cost threshold to select good grasps and switch from trajectory optimization to grasp execution. The trajectory optimization is re-implemented in Python using sequential quadratic programming (SQP) from CVXPY [DB16] and we initialized the joint velocity control input $\mathbf{u}_t$ and intermediate joint state $\mathbf{x}^R_t$ with a heuristic that guides towards the mean of occlusion centers. Map construction used a Python implementation of KinFu [ZSN+17], which removes the iterative closest point (ICP) pose estimation step from the original KinFu work [NIH+11] and instead requires a camera

pose for each input RGB-D image.

Object Interior as Occlusion Issue Because our table-top environment only includes small household objects, if we extract the occluded region purely based on the closest OBB faces to the current RGB-D sensor position, the object's interior space will always be recognized as an occluded region. However, these object-interior occluded regions can never be observed from the outside regardless of camera movements. Given that the trajectory optimization process is already difficult to find the optimal joint trajectory due to many local minima and the frontier-extracted occluded regions are heavily viewpoint-dependent, we observed many cases where the trajectory optimization cannot find any good joint action and always returns a zero vector because of these additional object-interior occluded regions. Therefore, we added an extra simple step to remove most of these object-interior occluded voxels by filtering out the voxels inside the object axis-aligned bounding box (AABB). However, this is not guaranteed to filter out all object interior voxels (as shown in Figure 2.10a after this filtering step) and will sometimes remove surface points if the estimated AABB is not exact.

Eval Results The grasping success rate of the active exploration with antipodal grasping approach in our 50 table-top object clutter scenes with rendered noise-free RGB-D observations is **64.38%**, 6.87% worse than the simple antipodal grasping pipeline described in Section 2.3. After detailed analyses, we found a few improved examples with more complete observed object models compared to the simple antipodal grasping pipeline but observed several failure cases. The most frequent failure case is that the trajectory optimization process often plans a joint trajectory that is very close to inverse kinematic singularities, making it difficult to plan the subsequent antipodal grasping motion.

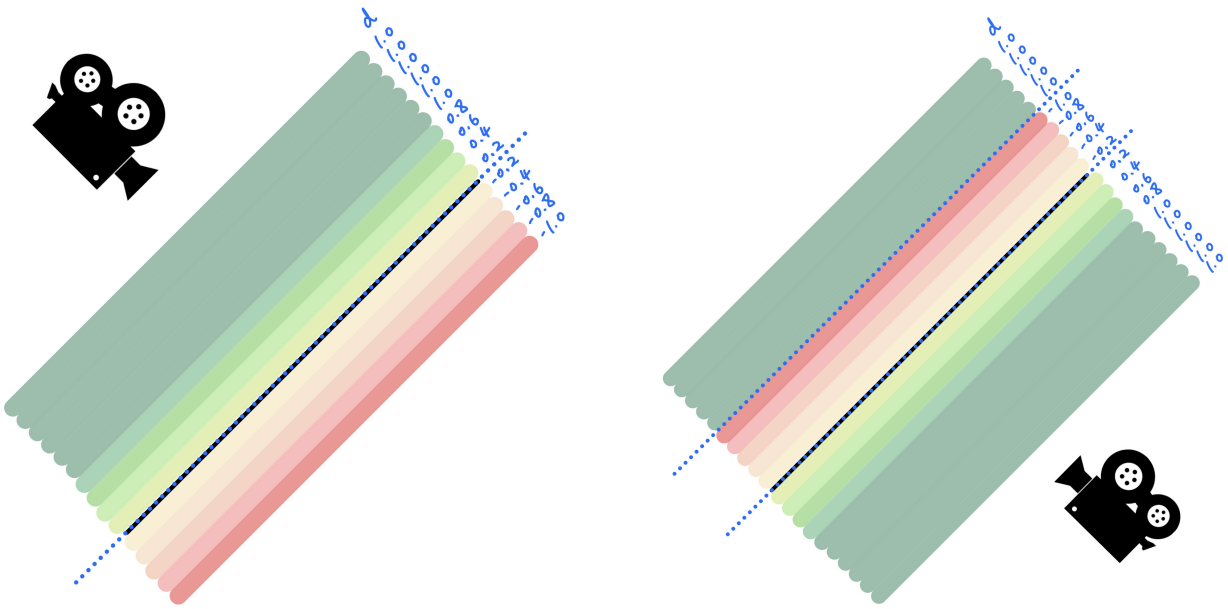
The second frequent failure case relates to an inherent issue of the TSDF map representation used by KinFu [NIH+11]. In a voxel grid constructed by KinFu TSDF, each voxel contains a confidence weight $w \geq 0$ and a truncated signed-distance value $d \in [-1.0, 1.0]$. The

surface meshes are then extracted from the TSDF by running the marching cubes algorithm [LC87] on d to extract the zero-crossing surfaces. Given a point cloud observation at time t, the truncated signed-distance values d are updated using a simple weighted moving average with $d_t \in [-1.0, 1.0]$ computed from the observation, where d_t is normalized by dividing the depth distance between the voxel and the observed point on a camera ray with a truncation margin μ and d_t is positive if in front and negative if behind the observed point. The d values of a planar surface from a single viewpoint are illustrated in Figure 2.9a.

However, in Figure 2.9b, if the camera moves to the opposite side of the planar surface, the d values obtained from the first viewpoint will be eventually overwritten by the weighted moving average, causing the marching cubes algorithm to mistakenly extract an extra surface plane between the $d = -1.0$ and $d = 1.0$ crossing. This inherent issue of the TSDF update process appears quite often in our environment because all of our objects are relatively small objects with finer geometry and the observations are collected with a trajectory-optimization-based active exploration approach where a surface is very likely to be observed from opposite viewpoint directions. Two of such double surface examples from our scenes are shown in Figure 2.10.

Finally, there are a few failed grasps where the robot arm collides with scene objects during the active exploration trajectory optimization process because no collision checking is done during the optimization to validate the robot joint states. The antipodal grasping algorithm deficiency illustrated in Figure 2.7 also occurs in some scenes.

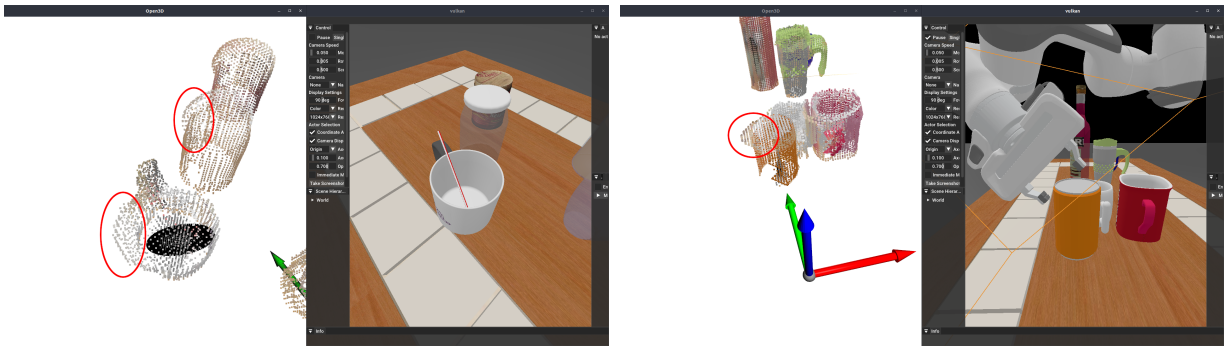
Overall, getting the trajectory-optimization-based active exploration approach to work properly is tricky due to many reasons including the joint state initializing heuristic, optimization local minima, viewpoint-dependent frontier extraction, object interior voxels as occlusions issue, TSDF inherent double surface issue, kinematic singularities and no collision checking during trajectory optimization. Although researchers proposed directional TSDF [SB19] to solve the double surface issue by encoding the surface orientations and preventing overwrites, implicit surface reconstruction with approaches such as neural radiance fields (NeRF) [MST$^+$20] and



(a) Truncated signed-distance values d from a single viewpoint

(b) d values after more observations from a second viewpoint

Figure 2.9: A planar surface example illustrating the issue of TSDF truncated signed-distance values d. The planar surface denoted by the black solid lines is the same in (a) and (b). The marching-cubes extracted surfaces are denoted with blue dotted lines. Camera viewpoints are illustrated with the camcorder icon. (a) shows the d values after observations from a single viewpoint. (b) shows the d values after more observations from a second viewpoint on the opposite side of the planar surface. An extra surface plane is mistakenly extracted because the observations from the second viewpoint overwrite the d values from the first viewpoint.



(a) TSDF surface issue (black ellipsoid is the occluded region Gaussian covariance)

(b) Another TSDF surface issue causing an unstable grasp

Figure 2.10: TSDF erroneous double surface issue (red circles) causes unstable grasps. (a) and (b) show the point cloud extracted from TSDF on the left and the actual scene in simulation on the right. (a) shows both the mug and the bottle have erroneous surfaces around the top rim. (b) shows that the erroneous points around the can's rim cause an unstable grasp.

occupancy networks [MON+19] would be more promising, efficient, and scalable to large scenes.

2.5 Contact-GraspNet

In this section, we evaluated Contact-GraspNet [SMTF21], the most recent state-of-the-art grasping-in-clutter approach in our table-top environment.

2.5.1 Method Overview

The Contact-GraspNet (CGN) [SMTF21] is an end-to-end grasp pose generation network which takes in a single-view full scene point cloud and generates candidate 6-DoF grasp poses each with a confidence score. The full inference pipeline is shown in Figure 2.11.

Grasp Representation In this work, researchers proposed to map the grasp pose to a contact point $c \in \mathbb{R}^3$ and reduce the 6-DoF grasp learning problem to estimating the 3-DoF grasp rotation $R_g \in SO(3)$ and the 1-DoF grasp width $w \in \mathbb{R}$ of a parallel gripper. Starting from a contact point $c \in \mathbb{R}^3$, a 6-DoF grasp pose g defined by $(R_g, \mathbf{t}_g) \in SE(3)$ can be recovered as

$$\begin{aligned} \mathbf{t}_g &= \mathbf{c} + \frac{w}{2}\mathbf{b} - d\mathbf{a} \\ R_g &= \begin{bmatrix} \mathbf{b} & \mathbf{a} \times \mathbf{b} & \mathbf{a} \end{bmatrix}, \end{aligned} \tag{2.4}$$

where $\mathbf{a} \in \mathbb{R}^3$, $\|\mathbf{a}\| = 1$ is the approach direction, $\mathbf{b} \in \mathbb{R}^3$, $\|\mathbf{b}\| = 1$ is the grasp baseline direction perpendicular to $\mathbf{a}$, $w \in \mathbf{R}$ is the predicted grasp width, and $d \in \mathbf{R}$ is the constant distance from the gripper baseline to the gripper base frame. Their proposed grasp representation is illustrated in Figure 2.12.

Training Data CGN is trained with the ACRONYM dataset [EMF21], which consists of 8872 object meshes from the Shapenet dataset [CFG+15] and 17.7 million simulated grasps under

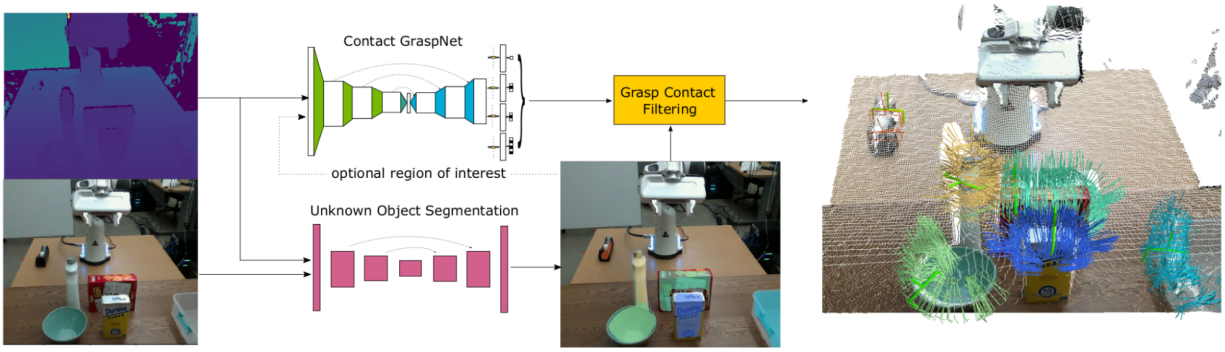


Figure 2.11: Contact-GraspNet full inference pipeline (taken from [SMTF21]). The unknown object segmentation network is replaced with ground-truth segmentation rendered from our simulator. The CGN processes a local region point cloud around the target object. Predicted 6-DoF grasps are associated with the target object by filtering their contact points. The predicted 6-DoF grasp distribution is shown on the right, with the most confident grasp per object segment in bold.

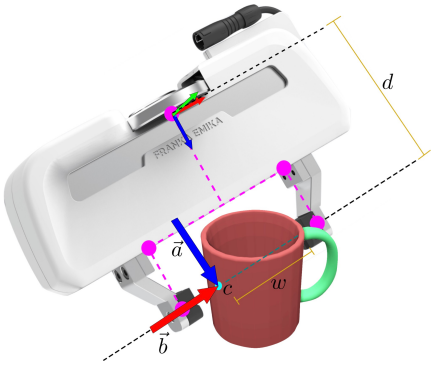


Figure 2.12: Contact-GraspNet grasp representation (taken from [SMTF21]). c denotes the observed contact point. The approach direction **a** and the grasp baseline direction **b** constitute the 3-DoF rotation. w is the predicted grasp width and d is the fixed distance from grasp baseline to the gripper base frame. The five gripper points **v** used in the l_{add-s} loss are shown in pink.

varying friction. The 10000 training clutter scenes are generated by placing 8 to 12 object meshes from the ACRONYM dataset on a table at random stable poses. Annotated grasp poses in collision are removed and the remaining grasps are mapped to their contact points on the mesh surface. During training, a single-view point cloud is rendered from a randomly sampled virtual camera pose, and the points are considered as positive contacts if there exist collision-free grasp pose mesh contacts in a 5mm radius and are associated with the grasp pose of the closest mesh contact. These per-point binary successes and grasp pose annotations are used to supervise the CGN.

CGN Architecture and Losses The CGN used the set abstraction and feature propagation layers proposed in PointNet++ [QYSG17] to build a U-shaped network. The network has four output heads each with two 1D-Conv layers and outputs per-point the grasp success probability $s \in \mathbb{R}$, $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^3$, and $\mathbf{o} \in \mathbb{R}^{10}$. The predicted grasp width $w \in [0, w_{max}]$ is split into 10 equidistant bins $\mathbf{o} \in \mathbb{R}^{10}$ and formulated as a multiclass classification problem. The grasp width w used to recover the 6-DoF grasp pose is the upper-bound value of the bin intervals with the highest confidence (to allow grasps at maximum grasp width). The approach direction $\mathbf{a} \in \mathbb{R}^3$ and the baseline direction $\mathbf{b} \in \mathbb{R}^3$ are orthonormalized from $\mathbf{z}_1, \mathbf{z}_2 \in \mathbb{R}^3$ by the Gram-Schmidt process.

Besides the binary cross entropy loss l_{bce} of grasp success $s \in \mathbb{R}$ and the multiclass cross entropy loss l_{width} of grasp width bins $\mathbf{o} \in \mathbb{R}^{10}$, CGN introduces a 6-DoF grasp loss l_{add-s} as a weighted minimum average distance of the 5 gripper points (shown in pink in Figure 2.12) between ground-truth $\mathbf{v}^{gt}$ and prediction $\mathbf{v}^{pred}$:

$$l_{add-s} = \frac{1}{n^+} \sum_{i}^{n^+} \hat{s}_i \min_u \|\mathbf{v}_i^{pred} - \mathbf{v}_u^{gt}\|_2, \tag{2.5}$$

where n^+ is the number of points with ground-truth positive grasps, and $\hat{s}_i$ is the predicted contact grasp success confidence. The total loss is $l = \alpha l_{bce} + \beta l_{add-s} + \gamma l_{width}$ with $\alpha = 1$, $\beta = 10$, and $\gamma = 1$.

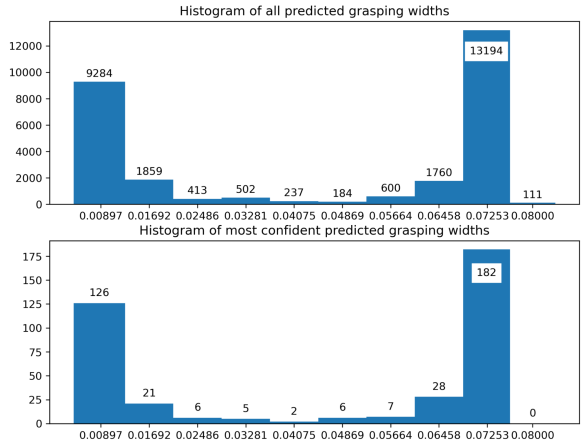
2.5.2 Grasping Evaluation Results

Grasping Evaluation Details To evaluate CGN in our table-top grasping environment, we replaced the unknown object segmentation network shown in Figure 2.11 with ground-truth segmentation rendered from our simulator. Also, the CGN work utilizes learned implicit collision functions [DMEF21] to generate collision-free robot arm motion. In our evaluation, we used the same library, MPlib (details see Appendix A) as in previous grasping evaluations for motion planning and collision checking. Starting from the CGN-generated grasp pose with highest

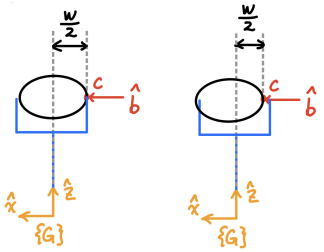
confidence, we executed the first collision-free grasp to determine grasping success. We chose the camera viewpoint to be a side view with most objects visible similar to CGN's evaluation environment.

Eval Results The grasping success rate of the pretrained CGN in our 50 table-top object clutter scenes with rendered noise-free RGB-D observations is **74.81%**, the grasping pipeline with the highest success rate so far. To examine its performance with depth sensor noise, we also evaluated the pretrained CGN with RGB + noisy depth observations obtained with SimKinect (details see Section 2.2.4) and the success rate in our 50 scenes is **62.60%**. After analyzing the results, we found that most of the grasps failed because the grasp candidates generated by CGN result in collisions, and the remaining collision-free grasp candidates are unstable.

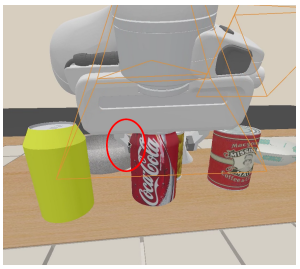
The predicted grasp poses in collision can be categorized into several cases depending on their causes. Figure 2.13a shows that very few predicted grasp widths w are at the maximum width and thus will result in collision if the object is thick and only allow grasps almost at the maximum grasp width (Figure 2.13b-2.13c). This observation is in line with the CGN paper [SMTF21] and they conjectured its reason to be the discontinuous decision boundary of the grasp width bins. Also, if the predicted grasp baseline direction deviates too much from being perpendicular to the contact surface, the gripper fingertip can collide with the target object as shown in Figure 2.13d-2.13e. The remaining collisions include colliding with the table (Figure 2.13f), a nearby object (Figure 2.13g), and the target object if the selected contact point is too low (Figure 2.13h). In addition, because the CGN is single-view-based, heavy occlusions of the target object in the camera viewpoint might cause the predicted grasp to be in collision or unstable (Figure 2.13i-2.13j). Finally, the predicted grasp pose can be unstable and misleading by trying to grasp the mug shown in Figure 2.13k by the straw, probably because CGN was not trained on such adversarial object models.



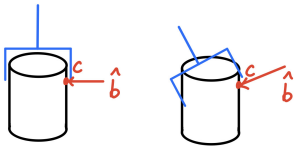
(a) Histograms of all and the most confident predicted grasp widths w



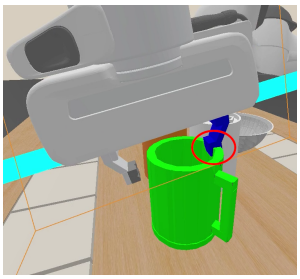
(b) Wrong grasp width w (right one) causes collision



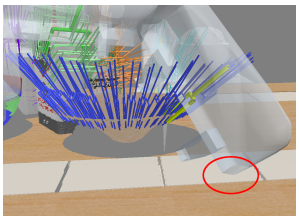
(c) Wrong grasp width collision



(d) Deviated grasp baseline **b** causing collision

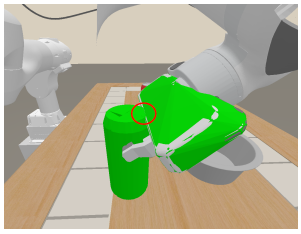


(e) Deviated grasp baseline collision



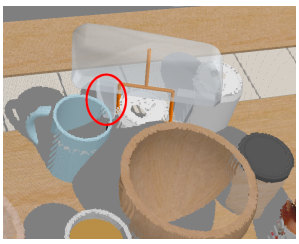
(f) Collides with the table

(g) Collides with a nearby object

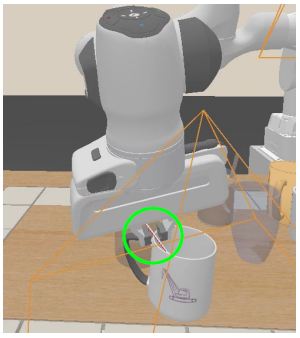


(h) Contact point too low, gripper-object collision

(i) Occluded viewpoint



(j) Collision from occlusion



(k) Unstable grasp by straw

Figure 2.13: Causes of collision for grasps predicted by CGN. Red circles denote collisions and green circles denote an unstable grasp. (a) shows that only a tiny portion of both all and the most confident predicted grasp widths w is the maximum grasp width. (b)-(c) show that the wrong predicted grasp width w causes collision for thick objects. (d)-(e) show that the deviated predicted grasp baseline causes a collision with the target object. (f) shows the gripper collides with the table at the predicted grasp pose. (g) shows the gripper collides with a nearby object at the predicted grasp pose. (h) shows the gripper collides with the target object because the predicted contact point is too low. (i)-(j) show a collision caused by heavy occlusion in the camera viewpoint. (k) shows an unstable predicted grasp of the mug by the straw.

In summary, the data-driven model-free grasping method, Contact-GraspNet, shows the highest grasping success rate in our 50 table-top clutter scenes when using noise-free RGB-D observations. However, there is still a wide gap between CGN's current performance (noise-free or noisy observations) and a reliable grasping-in-structured-clutter pipeline with a $\geq 95\%$ success rate. Admittedly, the proper way to test the true performance of the CGN approach with noisy depth observations is to collect a similar object model dataset with simulation-annotated grasp poses and retrain the CGN on a new set of train/test table-top clutter scenes. But even then, some of the observed failure cases of CGN will still remain, including the issue of very few predicted grasp widths at the maximum width, no supervision of aligning the gripper baseline to be perpendicular to the contact surface, single-viewpoint occlusions, and the unstable misleading grasps from adversarial object models.

2.6 Discussion & Conclusions

In this chapter, we closely examined a simple antipodal grasping algorithm and two existing baseline approaches: trajectory-optimization-based active exploration [KSP$^+$15] with antipodal grasping and the Contact-GraspNet [SMTF21]. Their grasping success rates in our 50 table-top clutter scenes are summarized in Table 2.2 below.

Table 2.2: Grasping success rate of tested baselines in our 50 table-top clutter scenes

Baseline	Observation Type	Success Rate
Simple antipodal grasping	Noise-free RGB-D	71.25%
Active exploration with antipodal grasping	Noise-free RGB-D	64.38%
Contact-GraspNet (pretrained)	Single-view Noise-free RGB-D	74.81%
Contact-GraspNet (pretrained)	Single-view RGB + Noisy depth	62.60%

From the table, we can see that none of the baselines achieves performance close to an ideal reliable grasping-in-structured-clutter method with a $\geq 95\%$ success rate. We also found a recent task-and-motion-planning (TAMP) approach [LK19] that we will briefly discuss next.

TAMP for grasping in clutter In *unstructured* environments such as homes and hospitals, directly specifying the full behavior policy for a robot is not practical due to complexity and edge cases. TAMP divides the complete behavior policy into high-level *task* planning (*e.g.*, move robot base, pick up object, place object) and low-level motion planning (*e.g.*, how to move the robot arm to grasp and pick up the target object). In [LK19], researchers proposed to use TAMP to first plan a motion trajectory to rearrange any existing obstacle that prevents access to the target object and then move to grasp the target object to the placement location. An example of this process is illustrated in Figure 2.14. In our point of view, this TAMP approach can be a quite strong method for our cluttered environment even if the clutter is denser. However, searching in a dense clutter for the optimal set of obstacles to move and the corresponding placement areas for rearrangement is probably a difficult time-consuming optimization problem. In [LK19], researchers also assumed a planar approach direction to the objects instead of a full 6-DoF grasp that can approach and lift the object from above. Moreover, implementing this TAMP approach requires lots of effort and the resulting pipeline will be very specific to this grasping-in-structured-clutter task without much generalizability to similar grasping tasks.

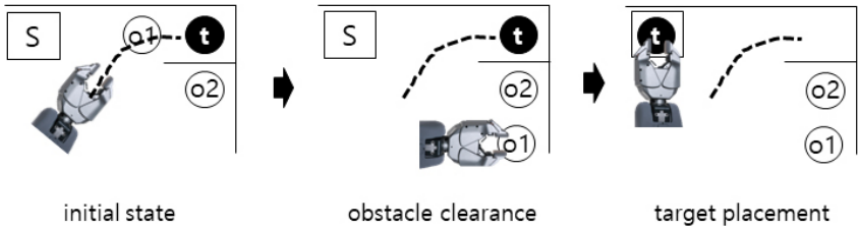


Figure 2.14: Example of grasping obstructed object using TAMP (taken from [LK19])

In our baseline experiments and analyses, we found that a successful grasp not only depends on a complete accurate scene mapping and high-quality collision-free grasp pose generation, but also relies on grasp trajectory planning and robot configuration space feasibility. Thus, we next explore the idea of bringing the simulator in the loop of planning a grasp by first reconstructing the observed scene in a physical simulator.

Chapter 3

Scene Reconstruction for Grasp Planning

For planning a robotic manipulation task in a physical simulator and directly transferring the planned motion trajectory, the reconstructed scene needs to have not only accurate geometric reconstruction but also close-enough dynamic properties predictions, stable physical supports with no penetration, and correct joint information for articulated objects. In this chapter, we will focus on reconstructing rigid objects in 3D scenes with accurate geometry and plausible physical constraints as a starting point. The table model and pose are fixed with ground-truth and not being reconstructed.

3.1 Related work

Digital Twin A digital twin is a virtual representation that serves as the real-time digital counterpart of a physical object or process. Recent work has studied the recreation of the digital twin of articulated objects through interactive perception [JHZ22]. Their category-agnostic implicit neural representation model is able to reconstruct part-level geometry and estimate the articulation model of an articulated object given a pair of point cloud observations before and after an interaction. The reconstructed digital twins can be directly imported into a physical simulator and used for planned motion transfer to the real world. However, they assumed a simple

scene where no other object exists and the reconstructed object surface generated by the marching cubes algorithm [LC87] from the implicit neural representation is not guaranteed to be smooth enough for stable placement on a supporting surface.

Scene Reconstruction Scene reconstruction has been studied extensively on indoor datasets of furniture such as beds, tables, and chairs [ADD+19, AKC+20, LYS+21]. These relatively large objects are less likely to result in collision (*i.e.*, object mesh penetration) in the reconstructed scene. However, for smaller household objects, a small error in the predicted pose or scale can incur mesh penetration, making the scene unstable for a physical simulator. On a high level, scene reconstruction methods can be categorized into implicit surface reconstruction and object pose estimation followed by CAD model alignments. For reconstruction with implicit representation, [AXW+20] worked on reconstructing a table-top setup for manipulation similar to ours. To encourage scene stability, they introduced a stability loss by geometric centroid analysis and a connectivity loss for bridging the disconnected object parts. The reconstructed scene is used for manipulation trajectory planning. Although their idea seemed promising, the reconstructed object surfaces from implicit representation are not smooth and their grasping success rate is $< 45\%$. [JZS+21] proposed to jointly learn the gripper grasp pose and scene reconstruction with an implicit neural representation. However, their network input is a TSDF volume fused from depth images which will unavoidably suffer from its inherent double surface issue illustrated in Figure 2.9. Also, their reconstructed scene was extracted using the marching cubes algorithm [LC87] and was not intended for physical simulation, meaning that potential collisions can occur.

For the two-step reconstruction approach of object pose estimation and CAD model alignments, [HZJ+21] is the most recent work. From an RGB-D image sequence, their panoptic mapping module generates reconstructed meshes for all object instances. Then, for each object mesh, they perform matching and optimization-based pose alignment with a CAD model dataset of the same object category. Finally, they perform physical violation checks to resolve nearby

object mesh penetrations and refine object supports. They demonstrated robot interactions in a ROS simulator and interactions in a VR environment using the reconstructed scene. Based on their promising reconstruction results of indoor scenes (*e.g.*, kitchen and room), we chose their approach as our scene reconstruction baseline and examined its performance in our table-top clutter environment.

3.2 Plane-based Scene Reconstruction Baseline

In this section, we evaluated [HZJ+21], the most recent scene reconstruction by pose estimation and CAD model alignment approach in our table-top clutter environment.

3.2.1 Method Overview

In [HZJ+21], researchers proposed to use a three-step pipeline to reconstruct interactive 3D scenes in a physical simulator: (1) panoptic mapping; (2) CAD matching and alignment; (3) physical violation check and contact graph generation. To represent the object-object relations, they proposed the contact graph scene representation which consists of a tree-like structure encoding the object support relationship and proximal edges for ensuring the non-penetration constraint between nearby objects.

Panoptic Mapping Given a sequence of RGB-D observations, they performed per-frame object segmentation by fusing panoptic segmentation from Mask R-CNN [HGDG17] and geometric depth segmentation based on surface convexity and depth discontinuity. Then, data fusion across image frames associates the instance and semantic labels to generate a set of consistent global instance labels. After processing all observations, the initial contact graph is constructed by estimating supporting relations between the objects based on extracted 3D oriented bounding boxes (OBB).

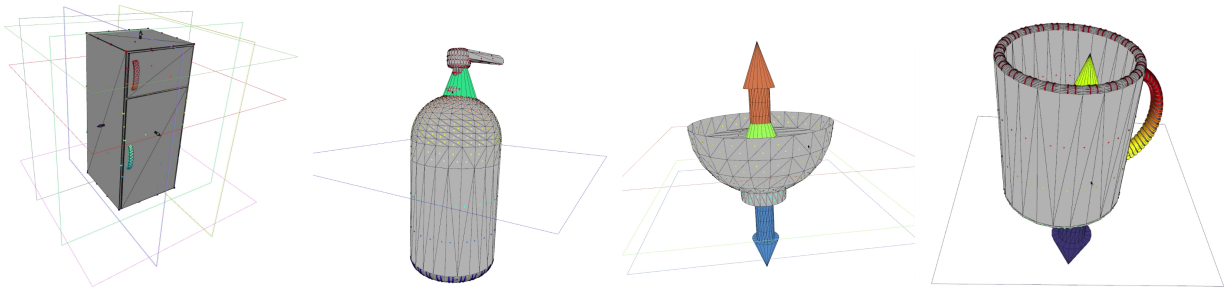
CAD Matching & Alignment Before matching, dominant planes are extracted from the reconstructed object meshes using RANSAC [FB81]. Then, for each model in the CAD object dataset of the same semantic category, a coarse pose matching is done by posing the CAD model in 24 discretized orientations and matching them with the reconstructed mesh using three similarity distance metrics, including the 3-DoF OBB size difference, the relative pose difference of the extracted RANSAC planes, and the up-direction matching error to encourage upright standing models. Next, a fine-grained pose alignment step takes the matched poses as initializations and performs a least-squares optimization to minimize the OBB error and the plane alignment errors using the Levenberg-Marquardt algorithm [Mor78]. To simplify the pose optimization process, it only involves a 4-DoF variable: a 1-DoF uniform scale, the yaw angle around the gravity-aligned axis, and the 2-DoF translation in the gravity-perpendicular plane.

Physical Violation Check & Contact Graph Generation With a set of well-aligned CAD model candidates, they performed a global physical violation check to refine the object position in the gravity-aligned axis and select CAD candidates for each segmented object to prevent object-object collision with a min-conflict constraint satisfaction algorithm [MJPL92] using the sum of mesh penetration depths. Finally, a contact graph with all matched CAD models and their poses is generated.

3.2.2 Reconstruction & Grasping Evaluation Results

Evaluation Details To reconstruct the table-top clutter scenes, we set the TSDF voxel resolution used in the voxblox++ [GFN+19] object segmentation module in [HZJ+21] to 5 mm. The RGB-D observation sequences were collected along the trajectories generated by the active exploration approach described in Section 2.4.1. We used the RGB + noisy depth observations obtained with SimKinect (details see Section 2.2.4). For now, the object dataset used for matching contains the same objects as the dataset we used in the 50 table-top scenes.

Reconstruction Results After examining the reconstructed scenes, we noticed that the CAD matching and alignment process does not perform well with our object dataset. In [HZJ+21], the scene being reconstructed are mostly indoor kitchens or rooms which mostly consist of relatively large furniture such as refrigerators, microwaves, and tables. These objects mostly have flat enclosing surfaces and RANSAC extracted planes are adequate approximations for their geometries. In our table-top clutter environment, the objects have much fewer flat surfaces, causing very few extracted planes. Also, a significant portion of objects have no extracted planes because their geometries mostly consist of spherical or cylindrical surfaces. Since the pose matching and alignment process heavily relies on the RANSAC extracted planes, these CAD models without any extracted planes would not be used, limiting the size of the actual object dataset used for matching. A few object model examples are shown in Figure 3.1.



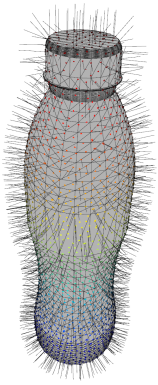
(a) Refrigerator (b) Bottle (c) Bowl (d) Mug

Figure 3.1: RANSAC extracted planes of objects for pose matching used in [HZJ+21]. Planes and their normals are illustrated with colored squares and arrows. (a) shows a refrigerator with flat enclosing surfaces, a common case for relatively large furniture. (b)-(d) show objects used in our table-top scenes with much fewer flat surfaces and hence very few RANSAC extracted planes.

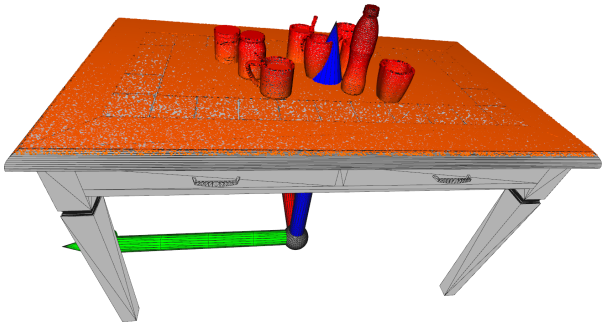
In our 50 table-top clutter scenes, we found that there are still 3 unstable reconstructed scenes where all of them are due to slight nearby-object mesh penetration. An example of such mesh penetration is shown in Figure 3.2. A possible reason is that the global physical violation check step only involves CAD candidate selection with no pose refinement and thus cannot guarantee object-object non-collision if no pair of CAD candidates is collision-free. Also, it could be that their mesh penetration checks are not accurate enough for slight penetrations.

Figure 3.2: Unstable scene reconstruction due to slight nearby-object mesh penetration

Grasping Results Using the reconstructed scenes, we performed brute-force sampling with the scene models and antipodal grasp sampling (see Section 2.3) to plan and execute grasping joint trajectories in our physical simulator. An example of brute-force sampled points on an object surface and a table-top scene is shown in Figure 3.3. The executed joint trajectories, if successful, will be directly transferred to the original scene to grasp the target object. The overall success rate in our 50 table-top scenes is **65.01%**, around 2.5% better than CGN with noisy depth observations from Section 2.5.



(a) An object model (with sampled point normals)

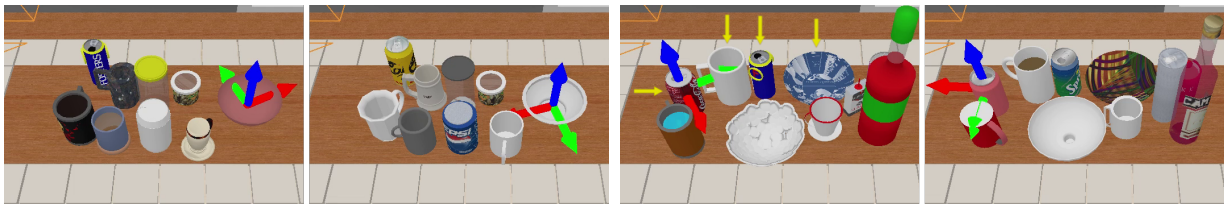


(b) A clutter scene (only the top-surface of the table is sampled for efficiency)

Figure 3.3: Brute-force sampled points on (a) an object surface and (b) a clutter scene. Red points indicate the sampled points.

Two examples of the reconstructed scene along with the original scene are shown in Figure 3.4. In Figure 3.4a, most of the objects are reconstructed with geometrically-similar different model instances but the object positions are all quite accurately reconstructed, which helps to

successfully transfer the grasping joint trajectories for all objects in the scene. However, in Figure 3.4b, several reconstructed objects have large geometric differences: the bottom-left upside-down mug is reconstructed as standing upright, the gray can is reconstructed with a completely different small oil dispenser (most likely to avoid nearby object collisions), and the wine bottle is replaced with a much wider bottle instance. These failure cases are great examples to show the deficiency of both the RANSAC-plane-based model matching and pose alignment process and the naive candidate selection algorithm to avoid nearby object collisions.



(a) A scene where all 9 transferred grasps succeed (b) A scene where 6/10 transferred grasps fail

Figure 3.4: Examples of the reconstructed scenes by [HZJ+21]. Left is the reconstructed scene and right is the original scene. (b) Objects marked by the yellow arrows are the ones with successfully transferred grasps.

Takeaways The [HZJ+21] work demonstrates a simple and efficient scene reconstruction pipeline by panoptic mapping, RANSAC-plane-based CAD model matching and alignment, and physical violation checking by candidate selection. We have seen its potential in reconstructing a geometrically similar scene (as shown in Figure 3.4a); however, it has several drawbacks: (1) plane-based model matching and pose alignment are not well suited for small objects without many flat enclosing surfaces; (2) the naive model candidate selection algorithm for preventing object-object collision usually sacrifices geometric similarity for physically stable non-penetration which is unideal if the reconstructed scene is going to be used for geometry-based planning; (3) the TSDF-based fusion of RGB-D observations will inevitably suffer from its inherent double surface issue illustrated in Figure 2.9. Also, although the results are not included, we have observed that TSDF-based scene representation is hard to filter out depth observation noises:

either the sensor noise needs to be filtered out before TSDF voxel updates, or a manual threshold of the TSDF observation weights w needs to be tuned to filter out the noisy voxels, assuming the noisy voxels will all have relatively lower observation weights compared to the rest.

Nevertheless, there are some takeaway ideas. First, the two-step scene reconstruction approach of object pose estimation followed by CAD model matching can make it much easier to generate a reconstructed scene with adequate object support because most CAD models are carefully designed for stable placement on a supporting plane. Second, to resolve nearby object penetration, a naive model candidate selection is probably not the right direction to pursue. It would be better to perform 3-DoF rescaling or deformation based on observations with some pose refinement steps to avoid nearby object collision and preserve geometric similarity. Third, a contact graph can be a good representation of the scene structure. Although if the scene has more complex support relations (*e.g.*, two objects both provide support to a third object on top of them), generalizing the tree-like support structure to a directed acyclic graph would be necessary. In our simple table-top environment where currently no object stacking is allowed, we do not need to explicitly generate the contact graph for our scene representation. Fourth, if the TSDF-based scene representation is not the right direction for fine-grained geometry representation and observation viewpoints from opposing directions, then in-network RGB-D observation feature fusion or observation keyframe selections might be worth to explore. Implicit neural representations also contain lots of potentials.

In the next section, we continue to explore the two-step scene reconstruction approach by 9-DoF category-level object pose estimation and CAD model matching focusing on geometric similarity.

3.3 Scene Reconstruction by Category-level Pose Estimation and Model Matching

In this section, we explore the scene reconstruction approach by 9-DoF category-level object pose estimation and CAD model matching in our table-top clutter environment.

3.3.1 Objective, Assumptions & Evaluation Metrics

The objective of our scene reconstruction approach is to reconstruct the scene in a physical simulator to match the geometry of the original scene as close as possible from a sequence of (or only one) RGB-D observations. The reconstructed scene can be used to plan for downstream manipulation tasks.

Our proposed two-step method is as follows: (1) perform a 9-DoF category-level pose estimation of the objects in the scene, and (2) match the observed object at the 9-DoF estimated pose with the most geometrically similar CAD model from the model database of the predicted semantic category while ensuring scene stability. Currently, the original and reconstructed environments are both created and simulated using SAPIEN [XQM+20].

Our environment assumptions include (1) objects are relatively small household objects (*e.g.*, mugs and bowls) but not large home furniture (*e.g.*, chairs and beds); (2) all objects are rigid and standing upright on the table with no object stacking allowed; (3) our environment is assumed to remain static during observation collection. During our reconstruction, we also do not attempt to reconstruct any visual properties (*e.g.*, colors, BRDFs) or dynamic properties (*e.g.*, frictions, mass distributions).

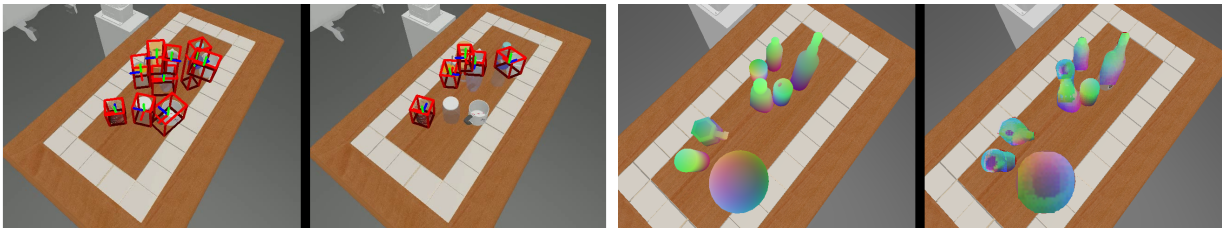
To evaluate the reconstruction, we need metrics to measure the geometry difference between the reconstructed scene and the original scene. We proposed to use a scene-level Chamfer distance by densely sampling the objects on the table (similar to Figure 3.3b). To account for scene stability, we simply use a binary metric to record whether the reconstructed

scene is stable in a physical simulator by forward-stepping. If needed, we can compute and use the sum of mesh penetration depths of all scene objects. We will also keep the grasping success rate metric in our table-top clutter scenes to evaluate the downstream manipulation task performance.

3.3.2 Related Work on Pose Estimation

Instance-level Instance-level object pose estimation has great practical importance in industrial applications and has been extensively studied in the literate both using traditional point cloud matching methods [AHB87, MAM14] and using learning-based methods [XSNF18, WXZ+19, HSH+20, HHF+21]. The task is to infer the 6-DoF object pose (3D position and 3D rotation) of the objects, assuming exact 3D CAD models and their sizes are available. The main challenges come from partial observations due to inter- and intra-object occlusions. Recently, the most notable work is PVN3D [HSH+20], a point-wise keypoint-voting-based network that takes in an RGB-D image and performs semantic segmentation, object center offset prediction, and 3D keypoint offset prediction. By clustering and voting the per-point semantic segmentations and center offset predictions, the model generates instance semantic segmentations which are combined with keypoint offset predictions to generate 3D object keypoint positions. A least-squares fitting is used at the end to produce 6-DoF object pose estimations from the keypoint positions. Their follow-up work, FFB6D [HHF+21], achieves state-of-the-art performance on 6-DoF instance-level pose estimation by adding a bidirectional fusion of the RGB image and depth point cloud features in the network flow. In their experiments, even when 90% of the object's surface is invisible in view, the FFB6D network can still achieve a remarkable pose estimation accuracy of 99.8%. However, instance-level object pose estimation needs *exact 3D models* of the objects during training and testing time. Besides the practical limitation in storing all 3D CAD models or learned networks in memory at test time, capturing and designing high-fidelity 3D models of many objects is a challenging task.

Category-level Category-level object pose estimation does not rely on the exact 3D models. Instead, it utilizes a canonical object frame for each object category, shared per-category keypoint positions, or part information as a bridging step to generate 9-DoF pose estimations for all object instances of the same category. One of the earliest works in category-level object pose estimation is NOCS [WSH+19], which defines a normalized object coordinate space (NOCS) for each object category and uses a Mask-RCNN backbone [HGDG17] to predict object category, instance segmentation mask, and the NOCS coordinates from an RGB-D image. The 9-DoF object pose is extracted with RANSAC [FB81] followed by the Umeyama algorithm [Ume91]. Two examples of the pretrained and retrained NOCS model performance are shown in Figure 3.5. The performance is not great mainly because the original NOCS network relies only on RGB images to perform classification, segmentation, and coordinate prediction while important geometric information contained in depth images is not utilized.



(a) Pretrained NOCS model pose predictions (b) Retrained NOCS model coordinate predictions

Figure 3.5: NOCS predicted results in our scenes. Left are ground-truths and right are predictions. (a) shows the pretrained model failed to detect 4 objects and generates inaccurate pose estimations for the 5 detected objects. (b) shows the retrained model contains severe noises in the NOCS coordinate predictions which will result in inaccurate pose estimations.

3.3.3 Method Overview

Inspired by our discussions with geometry experts, we found that simply choosing the voted keypoints to be the 8 corners of the object's oriented bounding box (OBB) can convert a 6-DoF instance-level keypoint-voting-based pose estimation network into a 9-DoF category-level pose estimation approach. In our following experiments, we adapted the state-of-the-art instance-

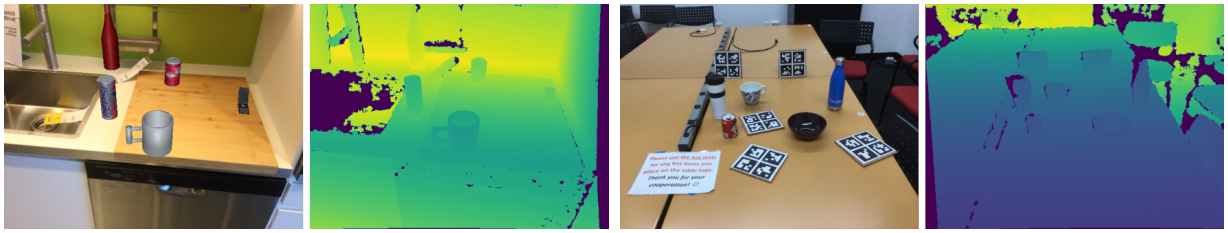
level pose estimation FFB6D architecture [HHF+21] to our category-level pose estimation problem. Specifically, given an RGB-D image, the FFB6D network produces per-point semantic segmentations, object center offset predictions, and keypoint offset predictions to the 8 OBB corners. Then, clustering and voting are performed to recover the 3D OBB corner positions which we used to estimate a 2-DoF object aspect ratio and a 7-DoF similarity transform using the Umeyama algorithm [Ume91]. We will refer to this category-level pose estimation approach as FFB6D-8OBB for the rest of this thesis.

3.3.4 Pose Estimation Evaluation Results on NOCS dataset

Evaluation Details To evaluate the category-level pose estimation of our FFB6D-8OBB approach, we chose the dataset introduced in the NOCS paper [WSH+19] because it is the only existing dataset for benchmarking category-level pose estimation approaches and it allows us to compare with existing baselines. The NOCS dataset contains two sets of RGB-D images: (1) a Contact-Aware MixEd ReAlity (CAMERA) dataset with real IKEA background images and synthetically rendered foreground objects in a *context-aware* matter (*i.e.*, objects are rendered with plausible physical locations, lighting, and scale); (2) real RGB-D images of 18 indoor scenes. The objects consist of 6 categories, including bottles, bowls, cameras, cans, laptops, and mugs. Detailed data distributions are summarized in Table 3.1 below and an example for both a CAMERA scene and a REAL scene are shown in Figure 3.6. Following the same training method used in NOCS [WSH+19], we train on both the CAMERA and REAL datasets and test on the REAL dataset.

Table 3.1: NOCS dataset distribution. Dataset splits are formatted as "Train / (Val) / Test"

Dataset Name	Models (6 categories)	Scenes	RGB-D Images
CAMERA	901 / 184	27 / 4	275K / 25K
REAL	18 / 6 / 18 (3 / 1 / 3 for each category)	7 / 5 / 6	4300 / 950 / 2750



(a) Example NOCS CAMERA scene (b) Example NOCS REAL scene

Figure 3.6: RGB-D observations of an example CAMERA scene and an example REAL scene in the NOCS dataset

NOCS 3DIoU Computation Bug As we started looking into the evaluation metrics used in NOCS, we found an unnoticed bug in computing the OBB IoU. The code and an example incorrect value are shown in Figure 3.7. At a glance, it seems like the authors of NOCS want to compute an enclosing AABB IoU as an approximation to the actual OBB IoU. However, the NumPy axis indices are incorrect, causing the significant numeric difference shown in Figure 3.7b. Unfortunately, all approaches evaluated on the NOCS dataset used this same incorrect implementation to compute 3D IoU. We have adopted a verified correct 3D IoU implementation from PyTorch3D [RRN+20]. For a fair comparison, all 3D IoUs shown in our later results are computed in the same way (Figure 3.7a) as NOCS (unless explicitly noted).

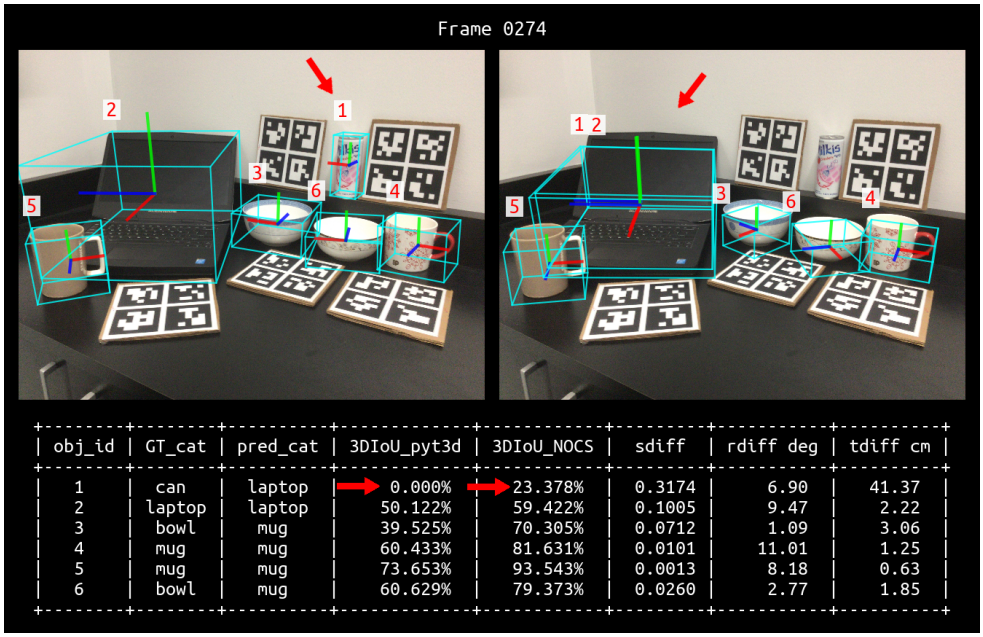
Evaluation Metrics The evaluation metrics for the category-level pose estimation are mean average precision (mAP) proposed in [EGW+10] of the 3D OBB IoU, rotation error (in degree), and translation error (in centimeter) at several thresholds. In order to compute the mAP values, one-to-one instance matches between the ground-truth objects and the predicted objects are computed by checking if they are of the same object category and if the 3DIoU is above the threshold (or the rotation and translation errors are below the thresholds). From the object instance matches, the precision/recall curve and average precision are computed. Finally, mAP is computed as the mean of APs for all object categories.

```python
def nocs_iou_3d(bbox_3d_1, bbox_3d_2):
    """bbox_3d_1, bbox_3d_2: [3, 8]"""
    bbox_1_max = np.amax(bbox_3d_1, axis=0)
    bbox_1_min = np.amin(bbox_3d_1, axis=0)
    bbox_2_max = np.amax(bbox_3d_2, axis=0)
    bbox_2_min = np.amin(bbox_3d_2, axis=0)

    overlap_min = np.maximum(bbox_1_min, bbox_2_min)
    overlap_max = np.minimum(bbox_1_max, bbox_2_max)

    # intersections and union
    if np.amin(overlap_max - overlap_min) < 0:
        intersections = 0
    else:
        intersections = np.prod(overlap_max - overlap_min)
    union = np.prod(bbox_1_max - bbox_1_min) + np.prod(bbox_2_max - bbox_2_min) - intersections
    overlaps = intersections / union
    return overlaps
```

(a) Function definition of 3D IoU computation in NOCS



obj_id	GT_cat	pred_cat	3DIoU_pyt3d	3DIoU_NOCS	sdiff	rdiff deg	tdiff cm
1	can	laptop	0.000%	23.378%	0.3174	6.90	41.37
2	laptop	laptop	50.122%	59.422%	0.1005	9.47	2.22
3	bowl	mug	39.525%	70.305%	0.0712	1.09	3.06
4	mug	mug	60.433%	81.631%	0.0101	11.01	1.25
5	mug	mug	73.653%	93.543%	0.0013	8.18	0.63
6	bowl	mug	60.629%	79.373%	0.0260	2.77	1.85

(b) Example of an incorrect 3D IoU

Figure 3.7: NOCS incorrect OBB IoU computation. (a) shows the code used in NOCS. `axis=0` should have been `axis=1` to compute an enclosing AABB IoU. (b) shows an example of incorrect 3D IoU. Left is ground-truth OBB and right is predicted OBB. For object 1, the OBB IoU between GT and pred should be 0% but NOCS gives 23.378% while the result from PyTorch3D 3DIoU implementation is correct.

Pose Estimation Results The category-level pose estimation results on the NOCS REAL-Test dataset are summarized in Table 3.2. From the row "FFB6D-8OBB", we can clearly see a large difference in IoU mAPs but only a small gap in deg/cm mAPs between our approach and the current state-of-the-art on NOCS REAL-Test dataset, FS-Net [CJC+21]. After inspection, we found our FFB6D-8OBB approach failed to generate correct semantic labels for many objects in

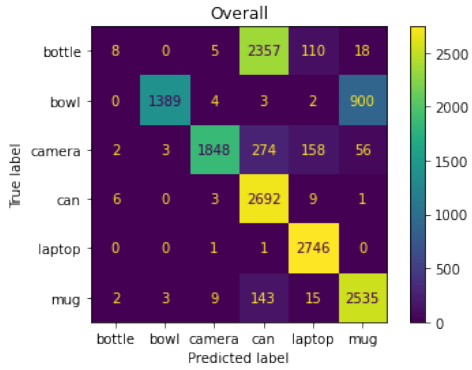
the dataset. The semantic classification confusion matrix is shown in Figure 3.8a. An example scene with per-point semantic prediction is shown in Figure 3.8b and 3.8c.

Because the semantic segmentation is not great and the mAP metrics couple pose accuracies with semantic classification accuracies, we performed another evaluation where the predicted semantic labels are substituted with the ground-truth semantic labels. The results are shown in Table 3.2 as "FFB6D-8OBB GTseg". We can see that our degree/cm mAPs surpass FS-Net by a large margin of 10% while the IoU mAPs are still $\sim$3% lacking behind, indicating possible room for improvements in the 8OBB corner position prediction.

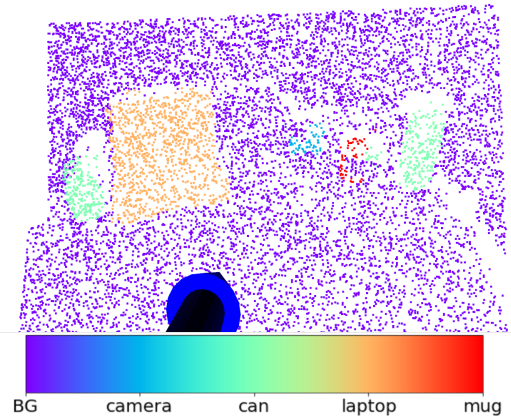
Table 3.2: Category-level pose estimation performance on NOCS REAL-Test dataset and our custom SAPIEN dataset, partly taken from [CJC+21]. All metrics are mAP. '-' means no results are reported. 'GTseg' means ground-truth semantic segmentation labels are used. Baseline references: NOCS [WSH+19], CASS [CLWX20], Shape-Prior [TAL20], 6-PACK [WMMX+20], FS-Net [CJC+21].

Method	IoU_{25}	IoU_{50}	IoU_{75}	5°5 cm	10°5 cm	10°10 cm	Speed (fps)
NOCS	84.9%	80.5%	30.1%	9.5%	26.7%	26.7%	5
CASS	84.2%	77.7%	-	23.5%	58.0%	58.3%	-
Shape-Prior	83.4%	77.3%	53.2%	21.4%	54.1%	-	4
6-PACK	94.2%	-	-	33.3%	-	-	10
FS-Net	**95.1%**	**92.2%**	**63.5%**	28.2%	60.8%	64.6%	**20**
FFB6D-8OBB	62.2%	61.3%	39.4%	26.7%	57.7%	61.8%	10
FFB6D-8OBB GTseg	91.1%	89.9%	61.3%	**35.4%**	**70.9%**	**73.4%**	10
Performance on our SAPIEN dataset (Section 3.3.5)							
FFB6D-8OBB	77.3%	71.9%	56.8%	71.2%	76.8%	78.1%	10
FFB6D-8OBB GTseg	96.7%	92.0%	72.1%	78.6%	83.9%	84.5%	10

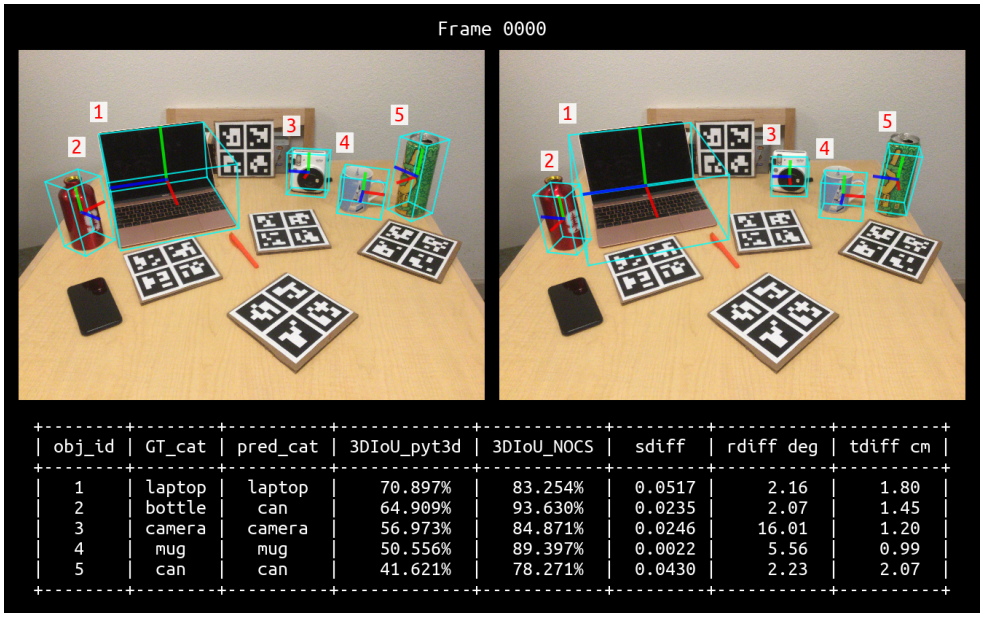
Possible Reasons for Semantic Segmentation Deficiency We summarized some potential reasons for the semantic segmentation deficiency we observed. (1) Comparing the NOCS CAMERA and REAL dataset as shown in Figure 3.6, we see that the CAMERA dataset has no depth noise for the object models while the REAL dataset has realistic depth discontinuity noises around the object boundaries. Also, the size of the CAMERA dataset is much larger than that of the REAL dataset as shown in Table 3.1 but the RGB-D images are sampled with equal weighting



(a) Object semantic classification confusion matrix



(b) Per-point semantic predictions of (c)



(c) Ground-truth (left) and predicted (right) object OBBs

Figure 3.8: Pose estimation visualizations of FFB6D-8OBB on NOCS REAL-Test. (a) shows bottles are being mostly misclassified as cans and many bowls are being misclassified as mugs. (b) shows that all points of the leftmost bottle are mispredicted as can points. Many points of the camera and the mug are also mispredicted as background. (c) shows that the OBB corner position predictions still have room for improvement for all 5 objects.

during training. It would probably make sense to sample more often in the REAL dataset or use SimKinect to add some simple depth noises to depth images in the CAMERA dataset. (2) In many baselines shown in Table 3.2, instance semantic segmentation is done by a separate Mask-RCNN-based network and additional RGB image datasets (*e.g.*, COCO [LMB+14]) are used to

train the 2D segmentation model. Comparing the per-point semantic segmentation approach with the 2D image approach, it is harder for the point-based network to learn more global object-level information which is valuable for instance semantic segmentation. If point-based network is necessary or end-to-end training is desired, the recent Point Group approach [JZS+20] might be worth looking into. More future works will be discussed in Chapter 4.

3.3.5 Pose Estimation Evaluation Results on Custom SAPIEN dataset

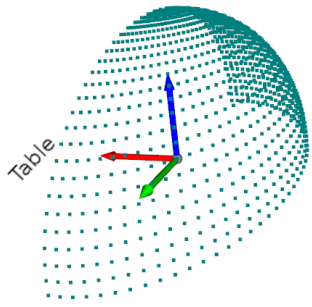
In this section, we present the results of applying FFB6D-8OBB in our table-top clutter environment for object pose estimation.

Custom SAPIEN Dataset To evaluate the category-level pose estimation approach in our table-top environment, we first divided our object models into train/test sets and then generate 80 training clutter scenes and 30 testing clutter scenes using the object sets. Then we rendered RGB-D images from the 1201 camera viewpoints shown in Figure 3.9b for each of the scenes. The depth images are processed with SimKinect (described in Section 2.2.4) to add close-to-real depth noises. The dataset details are summarized in the table shown in Figure 3.9a.

```
Bottle: 28 train, 22 test. 50 total. train/test: 56.00% / 44.00%
Bowl: 121 train, 41 test. 162 total. train/test: 74.69% / 25.31%
Can: 76 train, 32 test. 108 total. train/test: 70.37% / 29.63%
Mug: 126 train, 84 test. 210 total. train/test: 60.00% / 40.00%
```

Train/Test	Models (4 categories)	Scenes	Images
SAPIEN	351 / 179	80 / 30	96080 / 3630 (36030)

(a) Dataset distribution



(b) Camera positions

Figure 3.9: Custom SAPIEN dataset details. (a) shows the train/test splits of object models and clutter scenes. RGB-D images in the test scenes are interleavedly sampled down to 121 images per scene for speeding up the evaluation process. (b) The 1201 camera positions around the one-quarter ellipsoid centered at the table center for rendering RGB-D images. The camera always looks towards the table center.

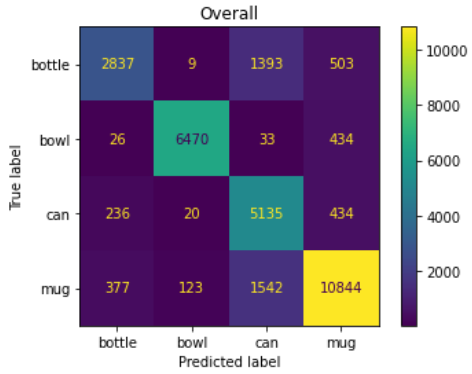
Pose Estimation Results The category-level pose estimation results on our testing images are summarized in the last two rows of Table 3.2. We see that the performance gap between using predicted semantic labels and using ground-truth labels is $\sim$10% smaller than that on the NOCS REAL-Test dataset. The semantic classification confusion matrix shown in Figure 3.10a also shows a better performance. Admittedly, we note that our custom SAPIEN dataset can be easier to achieve a good semantic classification accuracy because of the same distributed train/test sets and the simple background in our table-top environment. To further improve the semantic segmentation, one additional attempt can be adding category frequency weighting in the semantic segmentation Focal Loss [LGG$^+$20] used by FFB6D given that there are larger differences in the number of model instances of the object category. However, a portion of the misclassifications might be unavoidable due to category ambiguity from the camera viewing angle. An example scene with per-point semantic prediction is shown in Figure 3.10b and 3.10c.

3.3.6 Scene Reconstruction Evaluation Results on Custom SAPIEN dataset

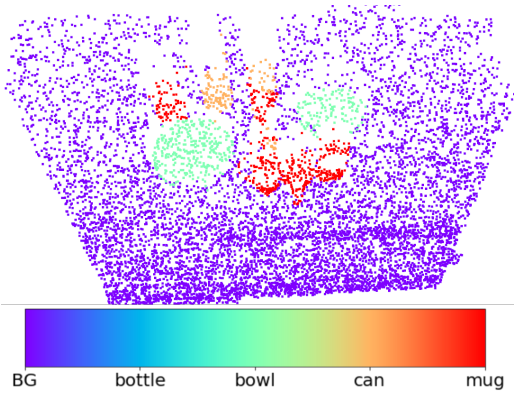
In this section, we complete the scene reconstruction pipeline with CAD model matching and present some evaluation results.

Top-down View Only To reconstruct the table-top clutter scene, we started our experiment by only using the top-down view observations. We chose the top-down view because it is guaranteed that all objects are observed in this view whereas other side views might have complete occlusions. A comparison of pose estimation accuracy by using all views and using the top-down view only is shown in Figure 3.11. We see that all metric performances are better when using the top-down view only, probably because of no occlusions.

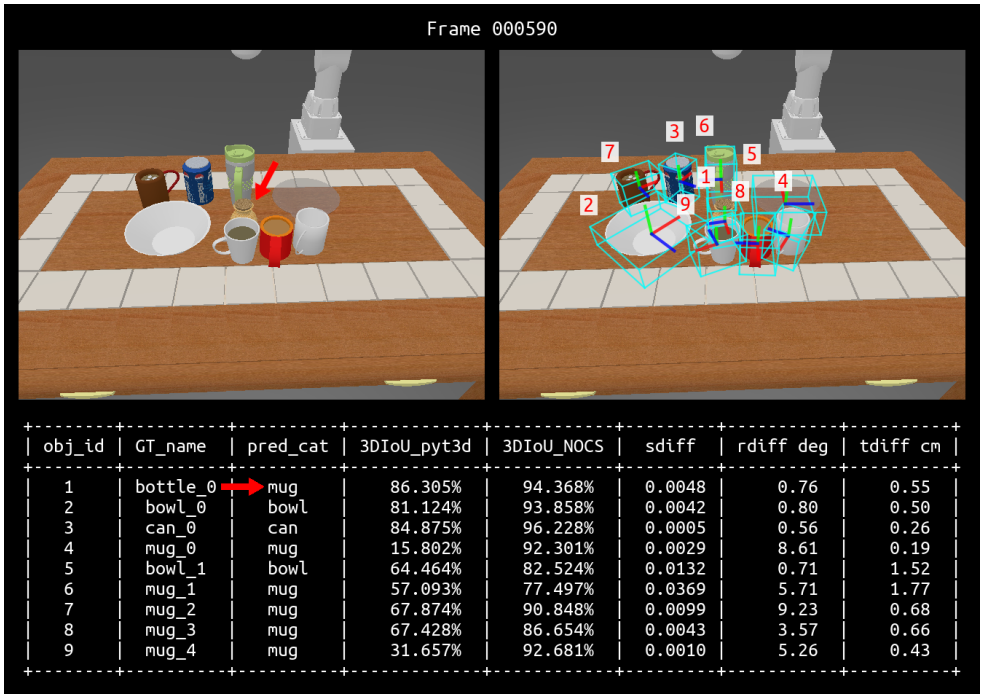
Model Matching Details Given 9-DoF pose estimations of the scene objects, we proceed to use CAD model matching to reconstruct the original scene. A naive solution for matching models



(a) Object semantic classification confusion matrix



(b) Per-point semantic predictions of (c)



obj_id	GT_name	pred_cat	3DIoU_pyt3d	3DIoU_NOCS	sdiff	rdiff deg	tdiff cm
1	bottle_0	mug	86.305%	94.368%	0.0048	0.76	0.55
2	bowl_0	bowl	81.124%	93.858%	0.0042	0.80	0.50
3	can_0	can	84.875%	96.228%	0.0005	0.56	0.26
4	mug_0	mug	15.802%	92.301%	0.0029	8.61	0.19
5	bowl_1	bowl	64.464%	82.524%	0.0132	0.71	1.52
6	mug_1	mug	57.093%	77.497%	0.0369	5.71	1.77
7	mug_2	mug	67.874%	90.848%	0.0099	9.23	0.68
8	mug_3	mug	67.428%	86.654%	0.0043	3.57	0.66
9	mug_4	mug	31.657%	92.681%	0.0010	5.26	0.43

(c) Original RGB image (left) and predicted object OBBs (right)

Figure 3.10: Pose estimation visualizations of FFB6D-8OBB on our custom SAPIEN dataset. (a) shows many bottles are being misclassified as cans and many mugs are being misclassified. (b) shows that mug with `obj_id=6` and the center bottle have half of the points misclassified. With only a slight number difference, mug 6 is classified correctly and the center bottle is not. (c) shows that the center bottle is mispredicted as a mug although no handle is visible.

at the predicted pose is to use the 7-DoF similarity transform which considers a uniform 1-DoF scaling and select the most geometrically similar object model. However, the 1-DoF scaling can limit the matching to only CAD models with similar 3D aspect ratios and will more likely

Method	5deg1cm	5deg5cm	mIoU_pyt3d	mIoU_nocs	scale err	rot err (deg)	trans err (cm)
FFB6D-8OBB	0.456587	0.629504	0.606689	0.794331	0.015299	17.480230	1.172080
FFB6D-8OBB top-down	0.521073	0.754789	0.645740	0.845374	0.013197	13.331079	0.905782

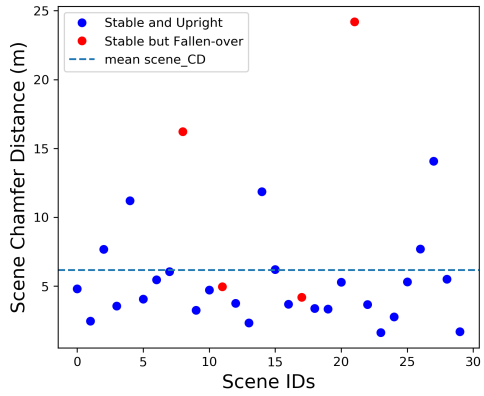
Figure 3.11: Mean pose estimation accuracy from all views vs. from the top-down view only. The degree/cm numbers are accuracies while others are means of the metric values.

cause nearby object mesh penetration (as demonstrated in Section 3.2.2 for the plane-based scene reconstruction approach). Thus, we proposed to utilize the full 9-DoF pose estimations by performing an additional 3-DoF rescaling.

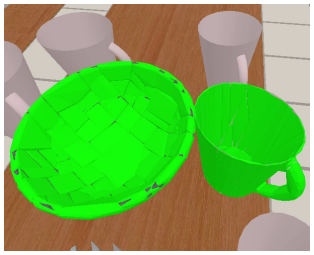
Specifically, we decomposed the 9-DoF pose into a 2-DoF 3D aspect ratio $\hat{a} \in \mathbb{R}^3, \|\hat{a}\| = 1$ and a 7-DoF similarity transformation. Given a CAD model, we performed rescaling to match the predicted 3D aspect ratio $\hat{a}$ and then inserted the scaled CAD model into the scene at the predicted 7-DoF pose. Next, we uniformly sampled points from the posed CAD model, computed the Chamfer distance between the observed object points and the sampled CAD points, and selected the model with the smallest Chamfer distance for reconstruction. Similar to [HZJ+21], we refined the table-object support relation by estimating the table top surface plane using RANSAC [FB81] and resetting the CAD model's $\hat{z}$-position accordingly.

Scene Reconstruction Results We computed the scene-level Chamfer distance metric described in Section 3.3.1 for the 30 testing table-top clutter scenes. The results are plotted in Figure 3.12a. As we can see, there are 4 scenes with fallen-over reconstructed objects. One of them (shown in Figure 3.12b and 3.12c) is because of a slight nearby object mesh penetration caused by the errors in the predicted object scales for both objects. The other 3 scenes contain reconstructed objects that have slight rotation errors and are not completely standing upright on the table. These unstable scenes inspire a future direction of estimating the uncertainty of the 9-DoF predicted pose and formulating a pose refinement optimization step to resolve such collisions and instabilities (details discussed in Section 4.2). For the remaining stably reconstructed scenes with no fallen-over objects, several scenes still have relatively large scene-level Chamfer distance. An example is

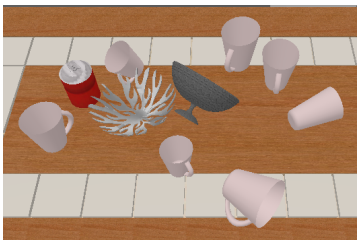
shown in Figure 3.12d and 3.12e. This is probably because using observations only from the top-down view is hard to find a good match for the scene objects.



(a) Scene-level Chamfer distance

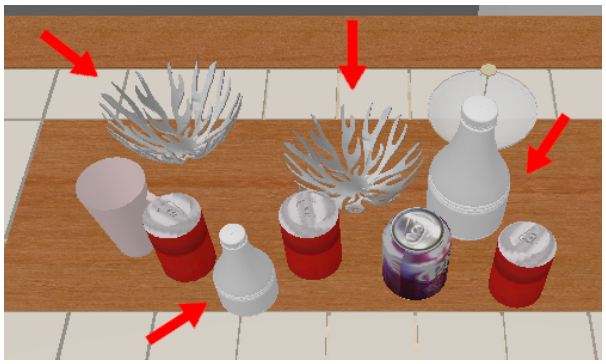


(b) Slight mesh penetration



(c) Forward-stepping result of scene shown in (b)

(d) Example of an original scene



(e) Reconstruction of (d) with a large scene-level CD

Figure 3.12: Reconstruction results of our table-top clutter scenes. (a) shows the scene-level Chamfer distance metric with 4 scenes containing fallen-over objects. (b)-(c) show slight nearby object mesh penetration due to the errors in the predicted scale, causing many fallen-over objects. (d)-(e) show an example of a stably reconstructed scene with a large scene-level Chamfer distance. Red arrows indicate objects with large Chamfer distances.

Grasp Planning Results To test the performance of using the reconstructed scene for downstream manipulation motion planning, we used the same method as we did when testing the plane-based scene reconstruction method in Section 3.2.2: brute-force sampling with the reconstructed scene models and antipodal grasp sampling to plan and execute grasping joint trajectories in our physical simulator. The executed joint trajectories, if successful, will be directly transferred to the original scene to grasp the target object. After evaluation, the overall success rate in our 30

testing table-top scenes is **46.36%**. The grasping performance is quite low compared to previous approaches but it is understandable given the matched objects' geometries are not similar enough to the geometries of the original objects (Figure 3.12a). Another failure case we found is that some matched bowls with too detailed geometries (*e.g*, the two hollow bowls shown in Figure 3.12e) will cause the antipodal grasp sampler to find an unstable grasp due to the current antipodal cost design deficiency mentioned in Section 2.3.3.

Chapter 4

Discussions & Future Work

Given that our FFB6D-8OBB 9-DoF category-level pose estimation approach achieves promising results as the current state-of-the-art approach, we plan to continue working on improving the pose estimation performance and then the scene reconstruction accuracy. In this chapter, we discuss several future directions we plan to pursue.

4.1 Pose Estimation

Currently, the bottleneck of our category-level pose estimation is the instance semantic segmentation accuracy, which has plenty of room for improvements on both the NOCS REAL-Test dataset and our custom SAPIEN dataset as shown in Table 3.2. For the NOCS dataset, we need to get a fairer evaluation by verifying the training procedure of the existing baselines to get aware of any particular tricks for good semantic segmentation performance on the NOCS dataset (*e.g.*, CAMERA/REAL dataset weighting, additional training data from COCO). This might require us to employ a separate 2D RGB instance semantic segmentation network pretrained on abundant real-world images or utilize point-based techniques like PointGroup [JZS+20]. We also need to decouple the mAP evaluation metrics from depending on semantic classification results and generate separate metrics for segmentation, IoU, and degree/cm accuracies to help better

understand the pose estimation performance. If there is a need, we need to re-benchmark the pose estimation performance for all existing baselines on the NOCS REAL-Test dataset with the correct implementation of 3DIoU to eliminate any uncertainty from the incorrect implementation.

For our custom SAPIEN dataset, we need to add a class-imbalance weighting in the segmentation Focal Loss and check the performance improvements. There is also the viewpoint-dependent object category ambiguity issue that we can verify by training a standard state-of-the-art object classification network to see the performance upper bound.

Better Keypoint Selection In PVN3D [HSH+20], researchers performed ablation studies and demonstrated a $\sim$1.5% ADD-S metric (*i.e.*, AUC of mean pair-wise distances between ground-truth and predicted object vertices considering symmetry) improvement if the selected keypoints for prediction switched from the 8 OBB corners to 8 farthest-point-sampled (FPS) points on the object surface. They claimed it is because the bounding box corners are virtual points relatively far away from the object surface and point-based networks are difficult to aggregate scene context in the vicinity of these virtual corner points. Thus, a better keypoint selection can help the pose estimation performance and possibly the semantic segmentation performance as well given that they are simultaneously learned during training. A simple starting point can be keypoints close to the semantic parts of the object category (*e.g.*, mug's handle, bottle's cap on top). These keypoints can be transformed and mapped to each model instance of the same category using the defined NOCS coordinates. However, a better keypoint selection alone probably will not improve the pose estimation performance a lot.

Multi-view Observations Utilizing multi-view observations and designing methods to fuse the visual information would almost certainly improve the pose estimation performance because relying on a single view will potentially suffer from object occlusions, object pose and category ambiguity. However, there is the challenge of efficiently acquiring multi-view observations while traversing a collision-free robot joint trajectory. Moreover, there have been several methods

studying the problem of category-level object pose tracking [WMMX+20, WWZ+21, WB21]. The most recent BundleTrack work [WB21] achieved close to 100% pose tracking accuracy in the NOCS REAL-Test dataset while not relying on any instance-level or category-level information by using a keypoint detection network similar to [SSTN18] and a keyframe memory pool for multi-frame pose joint-optimization similar to techniques used in the simultaneous-localization-and-mapping (SLAM) field. However, it assumes a static scene which essentially reduces the object pose tracking problem to a simple camera pose tracking problem widely studied in the computer vision community. I think the next important challenge along this direction is to perform 9-DoF object pose tracking in dynamic environments similar to recent research in the HOI4D dataset [LLJ+22]. Thus, working on object pose tracking in a static environment is of fewer merits and probably not something we want to pursue.

4.2 Pose Refinement Optimization for Physically Stable Scene

From the nearby object mesh penetration in the reconstruction scenes shown in Figure 3.12b and 3.12c, we see potential catastrophic outcomes if the 9-DoF object pose estimations contain small errors. Given that the FFB6D is a per-point prediction network, each observed object point will generate offset predictions to the 8 OBB corners. It seems natural to estimate a Gaussian distribution for the 8 OBB keypoint 3D position predictions and develop a technique to propagate the keypoint location uncertainty to the extracted 9-DoF pose parameters generated by the Umeyama algorithm [Ume91]. With uncertainties of the pose parameters, we can formulate a pose refinement optimization problem that attempts to prevent nearby object mesh penetration and encourage stable object placement (*i.e.*, standing upright on the table). This pose refinement step has the potential to guarantee stable scene reconstructions and thus is worthy of further consideration.

4.3 Miscellaneous

Some miscellaneous improvements are included in this section.

Antipodal Cost Design As shown in Figure 2.7, the current antipodal cost function used for grasp sampling can generate unstable grasp poses because all points between the gripper contribute to the antipodal cost. However, the correct implementation should only consider points on the contact surfaces between the gripper and the object.

Re-evaluate Grasping Baselines The grasping-in-clutter baselines discussed in Chapter 2 were benchmarked on the old 50 table-top clutter scenes while the scene reconstruction grasping approaches were evaluated on the new 30 testing clutter scenes. If we want to compare against the grasping baselines, we should re-evaluate their success rates on the 30 testing scenes.

This thesis, in part, is a coauthored unpublished material with Quan Vuong, Yuzhe Qin, Luobin Wang, Hao Su, Henrik Christensen. The thesis author is the primary investigator and author of the material appeared in this thesis.

Appendix A

MPlib: a Lightweight Motion Planning Package

To reduce the barrier to entry of robotic motion planning framework, the SAPIEN team release a lightweight Python package for motion planning, called *MPlib* [Liu21]. With less focus on low-level hardware interface, MPlib is decoupled from ROS and is a standalone Python package. In its experiments, MPlib demonstrated close to 100% success rates in motion planning with collision checking. The success rates are higher than the RRTConnect algorithm in MoveIt! [CSCC14] (a commonly used ROS motion planning package) while being 2 to 4 times faster.

During initialization, MPlib takes in a Unified Robotic Description Format (URDF) file that describes a robot's geometric representation, kinematics, and dynamics properties. It also takes in a Semantic Robotic Description Format (SRDF) file which provides extra information useful for motion planning, such as certain pairs of links to ignore collision checking. When planning a robot arm joint path, users need to provide current joint positions and specify the end-effector link and its SE(3) goal pose. To enable collision checking, users also need to update an environment model in the form of point clouds, attached boxes, geometric primitives, or meshes. Then MPlib generates time-optimal joint trajectories, which include desired joint

positions, velocities, and accelerations.

A high-level pipeline of MPlib is shown in Figure A.1. MPlib utilizes Pinocchio [CSB+19] and KDL for solving robot forward and inverse kinematics, FCL [SCP12] for detecting collisions between model geometries, RRTConnect algorithm from OMPL [SMK12] for sampling-based planning, and toppra [PP18] for time-optimal trajectory parameterization.

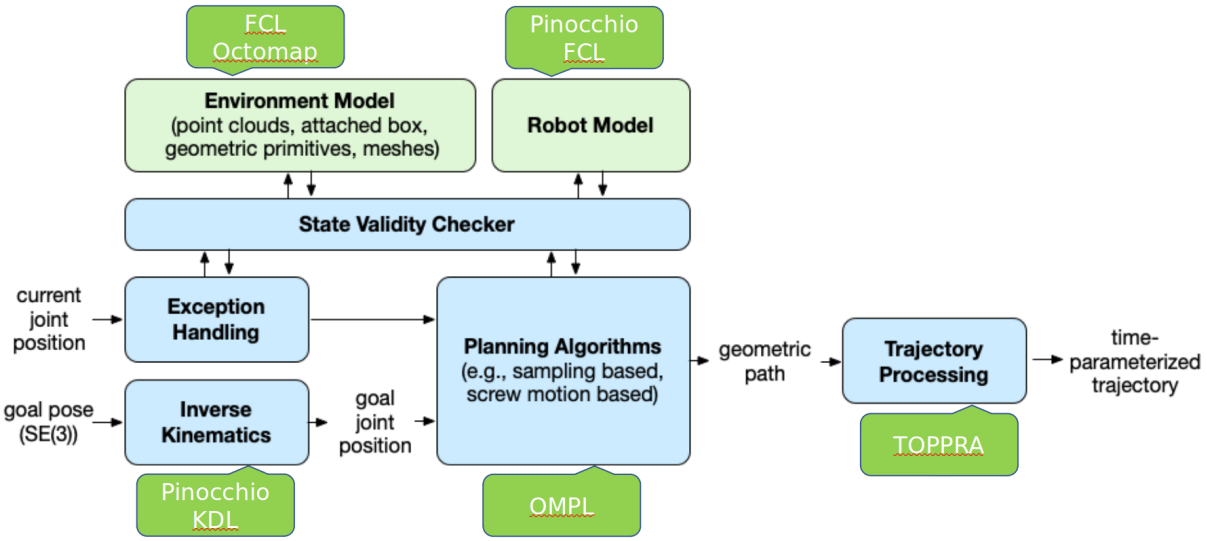


Figure A.1: MPlib high-level pipeline with third-party libraries. Taken from [Liu21].

Collision-checking Details for Our Use Case For all of our motion planning use cases, we used sampled or observed point clouds acquired from CAD models or depth camera observations. For collision checking, MPlib constructs an octree representation from the point clouds with a specified voxel resolution, generates boxes for each octree node based on the resolution, and uses FCL [SCP12] to check collision between robot link convex collision meshes and the octree boxes.

Bibliography

[ADD+19] Armen Avetisyan, Manuel Dahnert, Angela Dai, Manolis Savva, Angel X. Chang, and Matthias Nießner. Scan2cad: Learning cad model alignment in rgb-d scans. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2609–2618, 2019.

[AHB87] K. S. Arun, T. S. Huang, and S. D. Blostein. Least-squares fitting of two 3-d point sets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-9(5):698–700, 1987.

[AKC+20] Armen Avetisyan, Tatiana Khanova, Christopher Choy, Denver Dash, Angela Dai, and Matthias Nießner. Scenecad: Predicting object alignments and layouts in rgb-d scans. In *The European Conference on Computer Vision (ECCV)*, August 2020.

[ALRC14] Jacopo Aleotti, Dario Lodi Rizzini, and Stefano Caselli. Perception and grasping of object parts from active robot exploration. *Journal of Intelligent & Robotic Systems*, 76(3-4):401–425, 12 2014.

[AXW+20] William Agnew, Christopher Xie, Aaron Walsman, Octavian Murad, Caelen Wang, Pedro M. Domingos, and Siddhartha S. Srinivasa. Amodal 3d reconstruction for robotic manipulation via stability and connectivity. In *Conference on Robot Learning*, 2020.

[BM13] Jonathan T. Barron and Jitendra Malik. Intrinsic scene properties from a single rgb-d image. In *2013 IEEE Conference on Computer Vision and Pattern Recognition*, pages 17–24, 2013.

[BRHS14] Jeannette Bohg, Javier Romero, Alexander Herzog, and Stefan Schaal. Robot arm pose estimation through pixel-wise part classification. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3143–3150, 2014.

[CFG+15] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, Jianxiong Xiao, Li Yi, and Fisher Yu. Shapenet: An information-rich 3d model repository. *arXiv preprint arXiv:1512.03012*, 2015.

[CJC+21] Wei Chen, Xi Jia, Hyung Jin Chang, Jinming Duan, Linlin Shen, and Aleš Leonardis. Fs-net: Fast shape-based network for category-level 6d object pose estimation with decoupled rotation mechanism. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1581–1590, 2021.

[CLWX20] Dengsheng Chen, Jun Li, Zheng Wang, and Kai Xu. Learning canonical shape space for category-level 6d object pose and size estimation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11970–11979, 2020.

[CMS11] Alvaro Collet, Manuel Martinez, and Siddhartha S Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *The International Journal of Robotics Research*, 30(10):1284–1306, 2011.

[CSB+19] Justin Carpentier, Guilhem Saurel, Gabriele Buondonno, Joseph Mirabel, Florent Lamiraux, Olivier Stasse, and Nicolas Mansard. The pinocchio c++ library : A fast and flexible implementation of rigid body dynamics algorithms and their analytical derivatives. In *2019 IEEE/SICE International Symposium on System Integration (SII)*, pages 614–619, 2019.

[CSCC14] David Coleman, Ioan Alexandru Sucan, Sachin Chitta, and Nikolaus Correll. Reducing the barrier to entry of complex robotic software: a moveit! case study. *CoRR*, abs/1404.3785, 2014.

[DB16] Steven Diamond and Stephen Boyd. CVXPY: A Python-embedded modeling language for convex optimization. *Journal of Machine Learning Research*, 17(83):1–5, 2016.

[DMEF21] Michael Danielczuk, Arsalan Mousavian, Clemens Eppner, and Dieter Fox. Object rearrangement using learned implicit collision functions. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6010–6017, 2021.

[EGW+10] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *International Journal of Computer Vision*, pages 303–338, 2010.

[EMF21] Clemens Eppner, Arsalan Mousavian, and Dieter Fox. Acronym: A large-scale grasp dataset based on simulation. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6222–6227, 2021.

[FB81] Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, jun 1981.

[FUU+20] Yasuhiro Fujita, Kota Uenishi, Avinash Ummadisingu, Prabhat Nagarajan, Shimpei Masuda, and Mario Ynocente Castro. Distributed reinforcement learning of targeted grasping with active vision for mobile manipulators. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 9712–9719, 2020.

[GFN+19] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto. Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery. *IEEE Robotics and Automation Letters*, 4(3):3037–3044, July 2019.

[GKUP11] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: Blender sensor simulation toolbox. In *Advances in Visual Computing*, pages 199–208, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[HGDG17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[HHF+21] Yisheng He, Haibin Huang, Haoqiang Fan, Qifeng Chen, and Jian Sun. Ffb6d: A full flow bidirectional fusion network for 6d pose estimation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3002–3012, 2021.

[HPB+16] Ankur Handa, Viorica Patraucean, Vijay Badrinarayanan, Simon Stent, and Roberto Cipolla. Understanding realworld indoor scenes with synthetic data. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4077–4085, 2016.

[HSH+20] Yisheng He, Wei Sun, Haibin Huang, Jianran Liu, Haoqiang Fan, and Jian Sun. Pvn3d: A deep point-wise 3d keypoints voting network for 6dof pose estimation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11629–11638, 2020.

[HWMD14] Ankur Handa, Thomas Whelan, John McDonald, and Andrew J. Davison. A benchmark for rgb-d visual odometry, 3d reconstruction and slam. In *2014 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1524–1531, 2014.

[HZJ+21] Muzhi Han, Zeyu Zhang, Ziyuan Jiao, Xu Xie, Yixin Zhu, Song-Chun Zhu, and Hangxin Liu. Reconstructing Interactive 3D Scenes by Panoptic Mapping and CAD Model Alignments. In *ICRA*, 2021.

[JHZ22] Zhenyu Jiang, Cheng-Chun Hsu, and Yuke Zhu. Ditto: Building Digital Twins of Articulated Objects from Interaction. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2022.

[JZS+20] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. Pointgroup: Dual-set point grouping for 3d instance segmentation. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4866–4875, 2020.

[JZS+21] Zhenyu Jiang, Yifeng Zhu, Maxwell Svetlik, Kuan Fang, and Yuke Zhu. Synergies between affordance and geometry: 6-dof grasp detection via implicit representations. *Robotics: science and systems*, 2021.

[KBS15] Daniel Kappler, Jeannette Bohg, and Stefan Schaal. Leveraging big data for grasp planning. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4304–4311, 2015.

[KSP+15] Gregory Kahn, Peter Sujan, Sachin Patil, Shaunak Bopardikar, Julian Ryde, Ken Goldberg, and Pieter Abbeel. Active exploration using trajectory optimization for robotic grasping in the presence of occlusions. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4783–4790, 2015.

[LC87] William E. Lorensen and Harvey E. Cline. Marching cubes: A high resolution 3d surface construction algorithm. *SIGGRAPH Comput. Graph.*, 21(4):163–169, aug 1987.

[LFW+21] Liyang Liu, Simon Fryc, Lan Wu, Thanh Long Vu, Gavin Paul, and Teresa Vidal-Calleja. Active and interactive mapping with dynamic gaussian process implicit surfaces for mobile manipulators. *IEEE Robotics and Automation Letters*, 6(2):3679–3686, 2021.

[LGG+20] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(2):318–327, 2020.

[Liu21] Minghua Liu. MPlib: a Lightweight Motion Planning Package, 2021.

[LK19] Seokjun Lee and Incheol Kim. Task and Motion Planning for Grasping Obstructed Object in Cluttered Environment. *Journal of Korea Robotics Society*, 14:104–113, 2019.

[LLA21] Iker Lluvia, Elena Lazkano, and Ander Ansuategi. Active mapping and robot exploration: A survey. *Sensors*, 21(7), 2021.

[LLJ+22] Yunze Liu, Yun Liu, Che Jiang, Zhoujie Fu, Kangbo Lyu, Weikang Wan, Hao Shen, Boqiang Liang, He Wang, and Li Yi. Hoi4d: A 4d egocentric dataset for category-level human-object interaction. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[LMB+14] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *European Conference on Computer Vision*, pages 740–755, Cham, 2014. Springer International Publishing.

[LP17] Kevin M. Lynch and Frank C. Park. *Modern Robotics: Mechanics, Planning, and Control*, chapter 12. Cambridge University Press, 2017.

[LYS+21] Zhengqin Li, Ting-Wei Yu, Shen Sang, Sarah Wang, Meng Song, Yuhan Liu, Yu-Ying Yeh, Rui Zhu, Nitesh Gundavarapu, Jia Shi, Sai Bi, Hong-Xing Yu, Zexiang Xu, Kalyan Sunkavalli, Miloš Hašan, Ravi Ramamoorthi, and Manmohan Chandraker. Openrooms: An open framework for photorealistic indoor scene datasets. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7186–7195, 2021.

[MAC16] Riccardo Monica, Jacopo Aleotti, and Stefano Caselli. A kinfu based approach for robot spatial attention and view planning. *Robotics and Autonomous Systems*, 75:627–640, 2016.

[MAM14] Nicolas Mellado, Dror Aiger, and Niloy J. Mitra. Super 4pcs fast global pointcloud registration via smart indexing. In *Proceedings of the Symposium on Geometry Processing*, SGP '14, page 205–215, Goslar, DEU, 2014. Eurographics Association.

[MCL19] Douglas Morrison, Peter Corke, and Jürgen Leitner. Multi-View Picking: Next-best-view Reaching for Improved Grasping in Clutter. In *2019 IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[MEF19] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. 6-dof graspnet: Variational grasp generation for object manipulation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 2901–2910, 2019.

[MJPL92] Steven Minton, Mark D. Johnston, Andrew B. Philips, and Philip Laird. Minimizing conflicts: a heuristic repair method for constraint satisfaction and scheduling problems. *Artificial Intelligence*, 58(1):161–205, 1992.

[MLN+17] Jeffrey Mahler, Jacky Liang, Sherdil Niyaz, Michael Laskey, Richard Doan, Xinyu Liu, Juan Aparicio Ojea, and Ken Goldberg. Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics. 2017.

[MME+20] Adithyavairavan Murali, Arsalan Mousavian, Clemens Eppner, Chris Paxton, and Dieter Fox. 6-dof grasping for target-driven object manipulation in clutter. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6232–6238, 2020.

[MMS+19] Jeffrey Mahler, Matthew Matl, Vishal Satish, Michael Danielczuk, Bill DeRose, Stephen McKinley, and Ken Goldberg. Learning ambidextrous robot grasping policies. *Science Robotics*, 4(26):eaau4984, 2019.

[MON+19] Lars Mescheder, Michael Oechsle, Michael Niemeyer, Sebastian Nowozin, and Andreas Geiger. Occupancy networks: Learning 3d reconstruction in function space. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4455–4465, 2019.

[Mor78] Jorge J. Moré. The levenberg-marquardt algorithm: Implementation and theory. In G. A. Watson, editor, *Numerical Analysis*, pages 105–116, Berlin, Heidelberg, 1978. Springer Berlin Heidelberg.

[MPH+16] Jeffrey Mahler, Florian T Pokorny, Brian Hou, Melrose Roderick, Michael Laskey, Mathieu Aubry, Kai Kohlhoff, Torsten Kröger, James Kuffner, and Ken Goldberg. Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards. In *IEEE International Conference on Robotics and Automation (ICRA)*, pages 1957–1964. IEEE, 2016.

[MST+20] Ben Mildenhall, Pratul P. Srinivasan, Matthew Tancik, Jonathan T. Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. In *ECCV*, 2020.

[MZC+19] Kaichun Mo, Shilin Zhu, Angel X. Chang, Li Yi, Subarna Tripathi, Leonidas J. Guibas, and Hao Su. PartNet: A large-scale benchmark for fine-grained and hierarchical part-level 3D object understanding. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019.

[NIH+11] Richard A. Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J. Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pages 127–136, 2011.

[PP18] Hung Pham and Quang-Cuong Pham. A new approach to time-optimal path parameterization based on reachability analysis. *IEEE Transactions on Robotics*, 34(3):645–659, 2018.

[QCZ+19] Yuzhe Qin, Rui Chen, Hao Zhu, Meng Song, Jing Xu, and Hao Su. S4G: Amodal Single-view Single-Shot SE(3) Grasp Detection in Cluttered Scenes. In *Conference on Robot Learning (CoRL)*, 2019.

[QYSG17] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5105–5114, Red Hook, NY, USA, 2017. Curran Associates Inc.

[RRN+20] Nikhila Ravi, Jeremy Reizenstein, David Novotny, Taylor Gordon, Wan-Yen Lo, Justin Johnson, and Georgia Gkioxari. Accelerating 3d deep learning with pytorch3d. *arXiv:2007.08501*, 2020.

[Rym14] Kylie Rymanowicz. Infant vision development: Helping babies see their bright futures!, Dec 2014.

[SB19] Malte Splietker and Sven Behnke. Directional tsdf: Modeling surface orientation for coherent meshes. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1727–1734, 2019.

[SCP12] Ioan Sucan, Sachin Chitta, and Jia Pan. FCL: Flexible Collision Library, 2012.

[SGHK20] Juil Sock, Guillermo Garcia-Hernando, and Tae-Kyun Kim. Active 6d multi-object pose estimation in cluttered scenarios with deep reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 10564–10571, 2020.

[SKLK17] Juil Sock, S. Hamidreza Kasaei, Luis Seabra Lopes, and Tae-Kyun Kim. Multi-view 6d object pose estimation and camera motion planning using rgbd images. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 2228–2235, 2017.

[SMK12] Ioan A. Sucan, Mark Moll, and Lydia E. Kavraki. The open motion planning library. *IEEE Robotics Automation Magazine*, 19(4):72–82, 2012.

[SMTF21] Martin Sundermeyer, Arsalan Mousavian, Rudolph Triebel, and Dieter Fox. Contact-graspnet: Efficient 6-dof grasp generation in cluttered scenes. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13438–13444, 2021.

[SSTN18] Supasorn Suwajanakorn, Noah Snavely, Jonathan J Tompson, and Mohammad Norouzi. Discovery of latent 3d keypoints via end-to-end geometric reasoning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[TAL20] Meng Tian, Marcelo H. Ang, and Gim Hee Lee. Shape prior deformation for categorical 6d object pose and size estimation. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *European Conference on Computer Vision*, pages 530–546, Cham, 2020. Springer International Publishing.

[Ted21] Russ Tedrake. *Robotic Manipulation*, chapter 5. 2021.

[Ume91] Shinji Umeyama. Least-squares estimation of transformation parameters between two point patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(4):376–380, 1991.

[WA12] Jonathan Weisz and Peter K. Allen. Pose error robust grasping from contact wrench space metrics. In *2012 IEEE International Conference on Robotics and Automation*, pages 557–562, 2012.

[WB21] Bowen Wen and Kostas Bekris. Bundletrack: 6d pose tracking for novel objects without instance or category-level 3d models. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 8067–8074, 2021.

[WFG+21] Chenxi Wang, Hao-Shu Fang, Minghao Gou, Hongjie Fang, Jin Gao, and Cewu Lu. Graspness discovery in clutters for fast and accurate grasp detection. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 15944–15953, 2021.

[WMMX+20] Chen Wang, Roberto Martín-Martín, Danfei Xu, Jun Lv, Cewu Lu, Li Fei-Fei, Silvio Savarese, and Yuke Zhu. 6-pack: Category-level 6d pose tracker with anchor-based keypoints. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 10059–10066, 2020.

[WRD15] Kanzhi Wu, Ravindra Ranasinghe, and Gamini Dissanayake. Active recognition and pose estimation of household objects in clutter. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4230–4237, 2015.

[WSH+19] He Wang, Srinath Sridhar, Jingwei Huang, Julien Valentin, Shuran Song, and Leonidas J. Guibas. Normalized object coordinate space for category-level 6d object pose and size estimation. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2637–2646, 2019.

[WWZ+21] Yijia Weng, He Wang, Qiang Zhou, Yuzhe Qin, Yueqi Duan, Qingnan Fan, Baoquan Chen, Hao Su, and Leonidas J. Guibas. Captra: Category-level pose tracking for rigid and articulated objects from point clouds. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 13189–13198, 2021.

[WXZ+19] Chen Wang, Danfei Xu, Yuke Zhu, Roberto Martín-Martín, Cewu Lu, Li Fei-Fei, and Silvio Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3338–3347, 2019.

[XQM+20] Fanbo Xiang, Yuzhe Qin, Kaichun Mo, Yikuan Xia, Hao Zhu, Fangchen Liu, Minghua Liu, Hanxiao Jiang, Yifu Yuan, He Wang, Li Yi, Angel X. Chang, Leonidas J. Guibas, and Hao Su. SAPIEN: A simulated part-based interactive environment. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

[XSNF18] Yu Xiang, Tanner Schmidt, Venkatraman Narayanan, and Dieter Fox. PoseCNN: A convolutional neural network for 6d object pose estimation in cluttered scenes. 2018.

[ZSN+17] Andy Zeng, Shuran Song, Matthias Nießner, Matthew Fisher, Jianxiong Xiao, and Thomas Funkhouser. 3dmatch: Learning local geometric descriptors from rgb-d reconstructions. In *CVPR*, 2017.

[ZSY+19] Andy Zeng, Shuran Song, Kuan-Ting Yu, Elliott Donlon, Francois R. Hogan, Maria Bauza, Daolin Ma, Orion Taylor, Melody Liu, Eudald Romo, Nima Fazeli, Ferran Alet, Nikhil Chavan Dafle, Rachel Holladay, Isabella Morona, Prem Qu Nair, Druck Green, Ian Taylor, Weber Liu, Thomas Funkhouser, and Alberto Rodriguez. Robotic pick-and-place of novel objects in clutter with multi-affordance grasping and cross-domain image matching. *The International Journal of Robotics Research*, 0(0):0278364919868017, 2019.

[ZYS+17] Andy Zeng, Kuan-Ting Yu, Shuran Song, Daniel Suo, Ed Walker Jr, Alberto Rodriguez, and Jianxiong Xiao. Multi-view self-supervised deep learning for 6d pose estimation in the amazon picking challenge. In *Proceedings of the IEEE International Conference on Robotics and Automation*, 2017.