Robust Scene Estimation for Goal-directed Robotic Manipulation in Unstructured Environments

by

Zhiqang Sui

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Computer Science and Engineering)
in the University of Michigan
2020

Doctoral Committee:

Professor Odest Chadwicke Jenkins, Chair
Professor Ella Atkins
Professor Benjamin Kuipers
Professor Edwin Olson
Associate Professor Robert Platt

Zhiqiang Sui

zsui@umich.edu

ORCID iD: 0000-0003-1286-9734

ACKNOWLEDGMENTS

First of all, I would like to give my deepest appreciation to my advisor, Professor Odest Chadwicke Jenkins, who has mentored and helped me along my Ph.D. journey. Chad's immense enthusiasm and wisdom for robotics have shaped my perspectives on conducting independent research. His constant encouragement and support provided guidance not just in research, but also in my life. I would also like to thank my thesis committee members, Professor Edwin Olson, Professor Benjamin Kuipers, Professor Robert Platt, and Professor Ella Atkins, for their keen insights in shaping the documents and being supportive. I am also thankful to Professor Xiaoping Chen, who has led the way and provided lots of opportunities in my earlier research career.

Thank you to all my labmates and collaborators for being supportive and provide immense contributions to this work. Thanks to the support and encouragement by Karthik Desingh for all the time from Brown to Michigan. Special thanks to great collaborations with my colleagues Zheming Zhou, Zhen Zeng, Lingzhu Xiang, Xiaotong Chen, Haonan Chang, and Ning Xu. The achievement would not be possible without their endeavors and contributions. Many thanks to Kevin French, Jana Pavlasek, Emily Sheetz, and Zhefan Ye for discussing and developing research projects.

Above all, my parents have supported me throughout my life. Without their support and sacrifice, none of these would be possible. Thanks for their endless love and support.

TABLE OF CONTENTS

LIST OF FIGURES

FIGURE

LIST OF TABLES

TABLE

ABSTRACT

To make autonomous robots “taskable” so that they function properly and interact fluently with human partners, they must be able to perceive and understand the semantic aspects of their environments. More specifically, they must know what objects exist and where they are in the unstructured human world. Progresses in robot perception, especially in deep learning, have greatly improved for detecting and localizing objects. However, it still remains a challenge for robots to perform a highly *reliable* scene estimation in unstructured environments that is determined by *robustness*, *adaptability* and *scale*. In this dissertation, we address the scene estimation problem under uncertainty, especially in unstructured environments. We enable robots to build a *reliable* object-oriented representation that describes objects present in the environment, as well as inter-object spatial relations. Specifically, we focus on addressing following challenges for *reliable* scene estimation: 1) *robust* perception under uncertainty results from noisy sensors, objects in clutter and perceptual aliasing, 2) *adaptable* perception in adverse conditions by combined deep learning and probabilistic generative methods, 3) *scalable* perception as the number of objects grows and the structure of objects becomes more complex (e.g. objects in dense clutter).

Towards realizing *robust* perception, our objective is to ground raw sensor observations into scene states while dealing with uncertainty from sensor measurements and actuator control . Scene states are represented as scene graphs, where scene graphs denote parameterized axiomatic statements that assert relationships between objects and their poses. To deal with the uncertainty, we present a pure generative approach, Axiomatic Scene Estimation (*AxScEs*). *AxScEs* estimates a probabilistic distribution across plausible scene graph hypotheses describing the configuration of objects. By maintaining a diverse set of possible states, the proposed approach demonstrates the robustness to the local minimum in the scene graph state space and effectiveness for manipulation-quality perception based on edit distance on scene graphs.

To *scale* up to more unstructured scenarios and be *adaptable* to adversarial scenarios, we present Sequential Scene Understanding and Manipulation (*SUM*), which estimates the scene as a collection of objects in cluttered environments. *SUM* is a two-stage method that leverages the accuracy and efficiency from convolutional neural networks (CNNs) with probabilistic inference methods. Despite the strength from CNNs, they are opaque in understanding how the decisions are made and fragile for generalizing beyond overfit training samples in adverse conditions (e.g., changes in illumination). The probabilistic generative method complements these weaknesses and provides an avenue for adaptable perception.

To *scale* up to densely cluttered environments where objects are physically touching with severe occlusions, we present *GeoFusion* , which fuses noisy observations from multiple frames by exploring geometric consistency at the object level. Geometric consistency characterizes geometric compatibility between objects and geometric similarity between observations and objects. It reasons about geometry at the object-level, offering a fast and reliable way to be robust to semantic perceptual aliasing. The proposed approach demonstrates greater robustness and accuracy than the state-of-the-art pose estimation approach.

CHAPTER 1

Introduction

1.1 Aims and Motivation

With the recent advances in AI and robotics research, the realization of general-purpose robots that can perform useful tasks in the human environment is within reach. Unlike traditional industrial robots that perform pre-programmed actions in a confined workspace, general purpose robots are expected to autonomously and intelligently complete a variety of tasks that are helpful to humans across a diversity of user preferences and environments, including household chores, entertainment, medical domains, education, and emergency response. In these environments, they are expected to be "taskable" so that they function properly, meet human expectations, and interact fluently with human partners.

Ideally, human users can specify their intent as a goal state (i.e., the desired configuration of the world) without being required to specify how this goal should be achieved (either as motions and actions). Such human-expressed goals could then be translated into symbols that interpret as computationally assertable *axioms*. The symbolic description of the goal could be used by well-developed goal-directed symbolic planners [30, 55] in the planning stage of the well-known sense-plan-act cycle. Upon receiving the symbolic requests, robots should be able to perceive the world, reason over actions towards a satisfying goal, and carry out these actions in terms of physical motion. More importantly, lots of tasks like preparing meals, doing laundry, or setting up tables require robots to physically interact with objects in the environments. Examples of common ma-

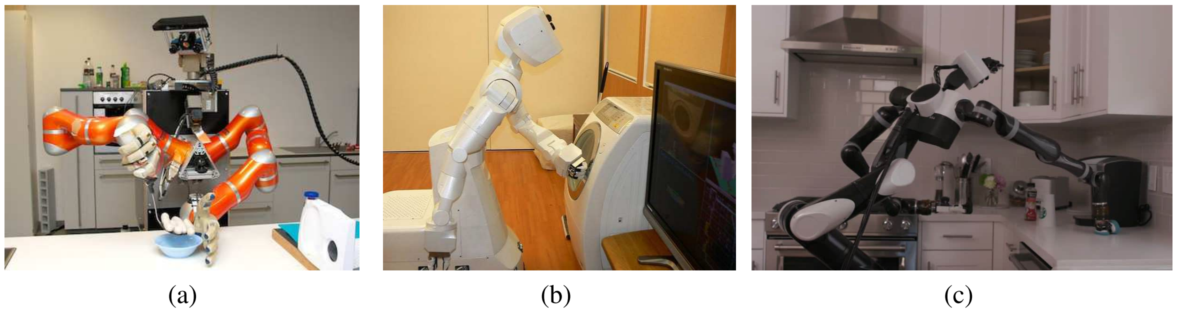


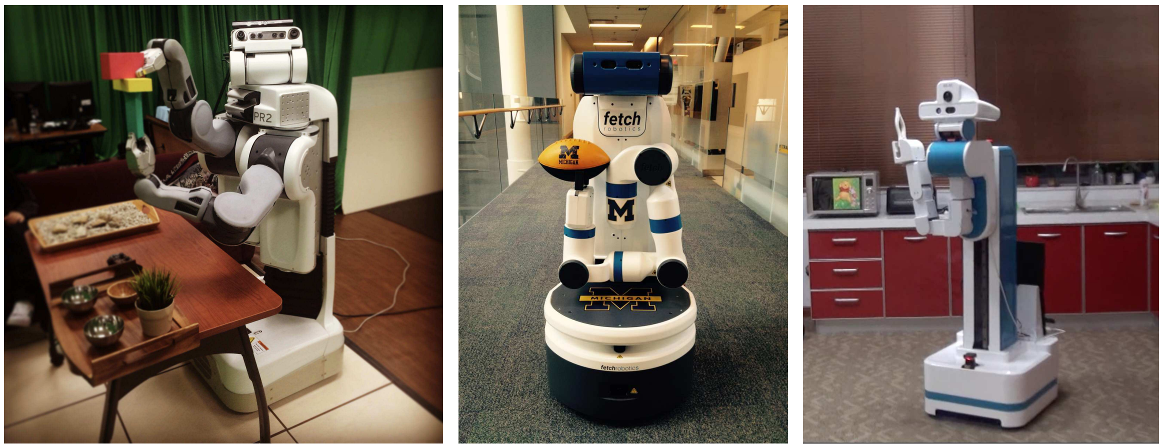
Figure 1.1: Common household manipulation tasks. (a) The robot from Technical University of Munich is preparing meals [8], (b) The robot from University of Tokyo is doing laundry, (c) The robot from Toyota Research Institute is making coffee [7].

nipulation tasks are shown in Figure 1.1 Robots need to perform manipulation actions with objects to satisfy task requirements. This level of autonomy is defined as *goal-directed manipulation*.

For extended goal-directed autonomy, we specifically face the problem of anchoring [20], a case of symbol grounding [37], to associate physical objects in the real world and relationships between these objects with computationally assertable axioms, from the robot's perception of the world. Anchoring and symbol grounding are at the heart the emerging area of semantic mapping [52] and its accelerated growth due to advancements in 3D RGBD mapping [92, 41]. Given raw sensor observations, taskable robots must know what objects exist and where they are. They should be able to build an object-oriented symbolic representation that describes objects present in the environment, as well as inter-object spatial relations. This representation provides a natural and efficient way for robots to reason and plan at the task level and communicate with human users naturally. In this dissertation, our objective is to enable robots to perceive their surroundings and build an object-oriented representation, including object classes, poses, and spatial inter-object relations for goal-directed manipulation.

1.2 Challenges

Great progress has been made in the field of scene perception, especially in perception for robotic manipulation with the renaissance of deep learning. Convolutional Neural Networks (CNNs), a



(a) (b) (c)

Figure 1.2: Examples of mobile manipulation robots. (a) Brown PR2 robot [103] (b) Michigan Progress Fetch Robot [104] (c) Kejia robot from University of Science and Technology of China [14]. They are equipped with low-cost cameras like Microsoft Kinect [121] or Asus Xtion, which have limited field-of-view and produce noisy RGB and depth data. Unlike the industrial arms, their manipulators also introduce imprecisions in actuator controls.

prevalent class of neural networks for visual data, have achieved promising accuracy and real-time inference speed for object detection [86, 60, 40], 6D object pose estimation [116, 109, 111], and spatial relations estimation in 2D images [57, 118, 122]. It is worth noting that these successes depend on well-designed models and adequate training resources. The robustness and generalization ability of CNNs depends largely on the training data, which represents a certain range of conditions that the robot may face.

However, the real world that robots attempt to perceive is dominated by uncertainty. Uncertainty comes at three levels: 1) low-cost hardwares on robots, 2) noisy or erroneous measurements from imperfect algorithms, and 3) complex and dynamic nature of the real unstructured environments. Uncertainty from low-cost hardwares arises as a result of both the observations from the noisy sensors and the performance of the motors that control the robot. Examples of mobile manipulation robots are shown in Figure 1.2. They are equipped with low-cost cameras like Microsoft Kinect [121] or Asus Xtion, which have limited field-of-view and produce partial and noisy RGB and depth observations. Unlike the industrial arms, their manipulators introduce imprecisions in actuator controls.

Uncertainty also comes from imperfect algorithms. CNN-based object detection and pose estimation methods often produce noisy or even false output due to the effects of overfitting during the training process and the biases and insufficient diversity in the training dataset. The performance of such discriminative systems deteriorates in unforeseen environmental conditions (which makes them especially vulnerable in adverse conditions). These errors (both benign and malicious) are another major source of uncertainty, semantic perceptual aliasing, as such CNNs serve as building blocks in scene perception for robotic manipulation.

Uncertainty from on-board sensors and algorithms increases in unstructured environments. The complex and dynamic nature of the environments (e.g., objects in clutter or varying illumination) limits the observability of the on-board sensors and adaptability of the CNNs recognition systems. Occlusions and physical contact between objects in clutter make Kinect produce data with greater depth discontinuities and straight edges appear as zig-zag lines. Distorted objects and/or objects captured under poor lighting conditions could be enough to defeat the recognition abilities of a CNN [27].

Under such perception uncertainty, our goal is to enable autonomous robots to perform robust goal-directed manipulation, which requires *reliable* scene perception. Specifically, robots should be able to build a reliable object-oriented representation that describes objects present in the environment, as well as inter-object spatial relations. Towards making a reliable scene perception, we must first answer: *What are the key factors that restrict the reliability of scene perception methods for robotic goal-directed manipulation?*

The first factor is *robustness*. Scene perception methods should be able to provide robustness to uncertainty. However, a majority of the CNNs recognition systems [86, 116, 57] are discriminative methods, which perceive exact single estimates of scene state. The single estimate is fragile to uncertainty, especially in the manipulation tasks with a long horizon plan. Once the estimate is erroneous, it can lead to potentially disastrous outcomes, and the robot is difficult to recover from the failure as the single estimate does not preserve any other possibilities. Generative models provide a means to address uncertainty probabilistically. Possible world states can be hypothesized to explain possibilities for the true world state that could have generated the robot?s observations. These generated hypotheses form an approximate probability distribution (or belief) over possible states of the world. The probabilistic generative approaches [23, 31, 107] have shown great success in the traditional robotics state estimation problem such as localization and mapping. The sample principle can be applied to perform reliable scene perception under uncertainty for goal-directed manipulation.

The second factor is *adaptability*. Due to the complex and dynamic nature of the unstructured environments, robots are subject to unforeseen conditions, which are not present in the training data. Discriminative CNNs recognition systems are not adaptable to such conditions, especially to adversarial scenarios due to the effects of overfitting to the training data. Generative approaches are inherently resilient to these conditions through the process of generating, evaluating, and maintaining a distribution of many hypotheses representing possible decisions. However, adaptability comes at the cost of computational efficiency. Discriminative-generative algorithms [97, 95, 72] offer a promising avenue for adaptable perception in unforeseen and adverse conditions. Such methods combine inference by deep learning (or other discriminative techniques) with sampling and probabilistic inference models. The weaknesses of one can be addressed by the strengths of the other.

The last factor is the *scale*. Lots of scene perception algorithms can work near perfectly in structured environments with objects separately placed on the table. But they don't work well in scenarios where objects are in dense clutter as the problem becomes rapidly difficult when dealing with severe occlusions, physical contact, and semantic perceptual aliasing. One promising approach [94, 65, 119] for building object-level representation of objects in dense clutter is to fuse semantic measurements from different viewpoints. Robots can take advantage of their ability to move around the environment and provide continuous observations. This approach is particularly advantageous of objects in dense clutter as it performs inference directly on object instances instead of low-level primitives, which offers faster inference, more compact map representation, and the potential for dynamic object reasoning.

The major challenge of enabling autonomous robots perform reliable scene perception is to break the limitation that is determined by *robustness*, *adaptability* and *scale*. In this dissertation, we describe several methods that expand the reliability of scene perception for goal-directed manipulation.

1.3 Thesis Contributions

In this dissertation, we address the scene estimation problem under uncertainty, especially in unstructured environments. We enable robots to build an object-oriented representation that describes objects present in the environment, as well as inter-object spatial relations. The object-oriented representation is expressed as scene graphs, where scene graphs denote parameterized axiomatic statements that assert relationships between objects and their poses and geometry. Given the arbitrary initial scene, the robot is able to ground the raw sensor observations to scene graphs, generate a sequence of actions by symbolic planners and execute the motions to complete the task. We require the scene graph estimation to be *reliable* under uncertainty with respect to *robustness*, *adaptability*, and *scale*.

In particular, this dissertation makes following contributions:

1. **AxScEs: Axiomatic Scene Estimation for Goal-directed Manipulation** We propose pure generative approaches [102, 103] to estimate the approximate distribution of the tree-structured scene graph describing the configuration of objects observed from robot sensing under sensor and actuation uncertainty. Our generative approaches iteratively hypothesizes possible scene configurations (as lists of axioms) and evaluates these hypotheses against the robot's observations. The result of this inference is an approximate posterior probability distribution over possible scenes, where the scene with maximum likelihood is taken as the estimate of the scene configuration. Though probabilistic in nature, a principal motivation for providing estimates represented in axiomatic form is the "closing the loop" between goal-directed symbolic planners [30, 55] and modern robotics. Such planners reason over manipulation

actions for the robot to execute from the estimate towards realizing a given goal scene, also expressed axiomatically. Planned sequences of actions are then executed by motion control/planning systems for pick-and-place manipulation [17] or general manipulation affordances [38]. By maintaining a diverse set of possible states, the proposed approach demonstrates the robustness to the local minimum in the axiomatic state space and effectiveness for manipulation-quality perception based on edit distance on scene graphs.

2. **SUM: Sequential Scene Understanding and Manipulation** However, *AxScEs* assumes known object identifications and works in moderately structured environments. To scale up to more unstructured scenarios and be adaptable to unforeseen and adverse conditions, we propose a combined discriminative-generative approach [104], which estimates the scene as a collection of objects in cluttered environments. *SUM* is a two-stage method that leverages the accuracy and efficiency from convolutional neural networks (CNNs) with generative probabilistic inference methods to deal with scene complexity and uncertainty. The approach utilizes candidates from the discriminative object detector to guide the generative process of sampling scene hypothesis, and each scene hypothesis is evaluated against observations. We demonstrate the effectiveness and robustness of *SUM* with respect to both perception and manipulation errors in a cluttered tabletop scenario for an object sorting task with a Fetch mobile manipulator. Moreover, we show that the proposed two-stage method is also adaptive to example adversarial scenarios.

3. **GeoFusion: Geometric Consistency informed Scene Estimation in Dense Clutter** *SUM* ignores physical interactions between objects by assuming objects are independent with each other. To reason about physical interactions between objects, especially in densely cluttered environments, we explore geometric consistency, which characterizes geometric compatibility between objects and geometric similarity between observations and objects. We present a geometric consistency informed scene estimation method [101] by fusing observations from multiple frames in dense clutter (*GeoFusion*). The geometric consistency in *GeoFusion*

reasons about geometry at the object-level, offering a fast and reliable way to be robust to semantic perceptual aliasing. The proposed approach demonstrates greater robustness and accuracy than the state-of-the-art pose estimation approach.

1.4 Dissertation Outlines

The remainder of this dissertation is organized into five chapters as follows.

Chapter 2 (Background) will survey work related to symbol grounding for robotics, perception for manipulation, scene estimation, and semantic mapping.

Chapter 3 (Axiomatic Scene Estimation) presents our generative approaches for scene inference and the evaluation of scene graphs and robot manipulation demonstrations.

Chapter 4 (Sequential Scene Understanding and Manipulation) describes the combined discriminative-generative approach for sequential scene estimation and manipulation and provides the scene estimation and sequential manipulation tasks evaluation.

Chapter 5 (Geometric Consistency informed Scene Estimation in Dense Clutter) describes the SLAM-based scene estimation method for building a semantic map at object-level from continuous noisy semantic measurements.

Chapter 6 (Conclusion) concludes the dissertation with a summary and talks about open problems and future directions.

CHAPTER 2

Background

The aim of this dissertation is for robots to perform robust scene estimation in unstructured environments for goal-directed manipulation. This objective is related to and/or motivated by previous work in several areas, including symbol grounding, robotic perception for manipulation, scene estimation, and semantic mapping. By surveying the related work, it places our work within perception for manipulation and semantic mapping.

2.1 Symbol Grounding for Robotics

2.1.1 Shared Autonomy for Manipulation

In order for autonomous robots to interact fluidly with human partners, a robot must be able to interpret scenes in the context of a human user's model of the world. The challenge is that many aspects of the human's world model are difficult or impossible for the robot to sense directly. We posit the critical missing component is the grounding of symbols that conceptually tie together low-level perception and high-level reasoning for extended goal-directed autonomy. We specifically face the problem of anchoring [20], a case of symbol grounding [37], to associate physical objects in the real world and relationships between these objects with computationally assertable facts (or axioms), from the robot's perception of the world. Anchoring and symbol grounding are at the heart the emerging area of semantic mapping [52] and its accelerated growth due to advancements in 3D RGBD mapping [92, 41]. With a working memory of grounded axioms about the world, robot manipulators will be able to flexibly and autonomously perform goal-directed tasks that require reasoning over sequential actions. Just as important, human users will be able to more intuitively specify goals for robots, as desired states of the world, through spatial configurations.

In the greater context of shared autonomy, goal-directed manipulation offers the opportunity to extend the boundaries of the "neglect curve" [33]. The neglect curve is a conceptual expression of robot effectiveness with respect to delegation (or user neglect), codifying tradeoffs between the extremes of full autonomy and manual teleoperation. While teleoperation can often yield high-levels of robot effectiveness, the performance of such systems relies heavily upon the training, aptitude, and stamina of a human operator. Conversely, systems for autonomous robots place much less burden on a human operator but are often limited to generalized trajectories over controls [3, 11, 84, 42], reactive policies [35, 22, 15, 78, 47, 110], or goals as combinations of hardcoded features [77, 1, 6, 49, 98]. As evidenced during the recent DARPA Robotics Challenge [117], shared autonomy is especially onerous and error prone for control of humanoids and mobile manipulators due to the complexity of goal-directed control. Similar to the pointing work by Kemp et al. [48], our long-term conjecture is that shared autonomy through the expression of goals will greatly reduce the complexity for human operation of robots, improving robot effectiveness during periods of delegation.

2.1.2 Goal-directed Manipulation

Our aim is to estimate axiomatic state representations that will allow robotics to build on the body of work in sequential planning algorithms, which have over a five-decade history. Described in early work, such as STRIPS [30] and SHRDLU [113], classical planning algorithms adapted theorem-provers to "prove" conclusions about goals based on symbolic axioms that describe the world through assertable logical statements. A classical planner can compute actions for a physical robot to perform arbitrary sequential tasks assuming full perception of the environment, which is often an untenable assumption in general.

However, in structured perceivable environments, systems based on classical planning have demonstrated the ability to reliably perform goal-directed manipulation. Recent work by Mohan et al. [70] uses the Soar cognitive architecture for teaching a robot arm to play games such as tic-tac-toe, Connect-4, and Towers of Hanoi through language-based expressions. Similar in spirit to our *AxScEs* estimators, Soar uses an axiomatic scene graph representation [114]. We posit *AxScEs* estimates could also be used within broader cognitive architectures, such as ACT-R/E [108], that are suited to axiomatic rather than strictly metric spatial representations. The approach proposed by Chao et al. [12] performs taskable symbolic goal-directed manipulation with a focus on associating observed robot percepts with knowledge categories. This method uses background subtraction to adaptively build appearance models of objects and obtain percepts but with sensitivity to lighting and object color. The work by Narayanaswamy et al. [74] performs scene estimation and goal-directed robot manipulation for cluttered scenes of toy parts for flexible assembly of structures.

The KnowRob system by Tenorth et al. [106] performs taskable goal-directed sequential manipulation at the scale of entire buildings by automatically synthesizing sources from the semantic web and Internet. Leveraging the community of perception modules available in the Robot Operating System (ROS) [85], KnowRob focuses uncertainty at the symbolic level and relies on hard and complete state estimates from hardcoded software components. Similarly, the work by Srivastava et al.[99] relies on hardcoded perception systems to perform the joint task and motion planning, taking advantage of modifications in controlled environments, which include green screens and augmented reality tags.

2.2 Perception for Manipulation

We summarize a relevant subset of existing work with respect to perception for manipulation. The PR2 interactive manipulation pipeline by Ciocarlie et al. [17] segments objects from a flat tabletop surface through the clustering of surface normals. This pipeline can perform relatively robust pick-and-place manipulation for isolated, non-occluded, and non-touching objects. For cluttered scenes, the work by Narayanaswamy et al. [74] performs pose and structure estimation of toy parts for flexible assembly of structures. MOPED [19] is a framework for object detection and pose estimation using the clustering of features from multi-views. The method from Joho et al. [45] uses a probabilistic generative model to model the spatial arrangement of objects on a flat surface in the context of a table setting task.

In terms of sequential manipulation, the system by Tenorth et al. [106] performs task-directed manipulation at the scale of entire buildings by integrating different knowledge sources from a perception system, an internal knowledge base and Internet repositories.The approach proposed by Cosgun et al. [21] performs sequential manipulation for placing objects in a scene where the objects are separating on a clear table. The work from Papazov et al. [82] performs sequential scene estimation and manipulation through the matching of known 3D geometries with an observed point cloud. However, this method takes a bottom-up approach using a RANSAC algorithm with Iterative Closest Point registration that neither requires nor uses a model of uncertainty.

Rosman et al. [88] proposed approaches to address the same problem of estimating relational scene graphs for objects in contact as a collection of axioms. Their approach detects contact points between objects that can be directly observed from depth. Liu et al. [61] proposed KSMCMC, which uses MCMC to sample over scene graph structures represented axiomatically to estimate objects as oriented bounding boxes. Pose estimation is performed using image features for alignment. Joho et al. [45] use a generative model to cluster objects on a flat surface into semantically meaningful categories.

2.3 Scene Estimation

We consider two categories of traditional methods for model-based object recognition and pose estimation. First, feature-based methods, also known as descriptor-based methods, aim to match key features in the models to the scenes. Key features can be comprised of local or global descriptors [4]. Using local features [44, 91, 89], the pose of the object is estimated by first matching a set of extracted features from the object model to the scene. Then, every matching pair will go through the filtering process to generate the final transformation. In contrast, methods using global features [5, 64, 90] attempt to match features with high resistance to the variance of the object pose. The object pose can be estimated by comparing those pose-preserving features from the training phase with the features computed from the test scene. A limitation of feature-based methods is that the estimation quality will degrade as the number of objects (and clutter) in the scene increases due to occlusion of key features.

Alternatively, generative methods (or *analysis by synthesis*) attempt to find the state estimate that best describes the observed sensor input through iterating over comparisons with state hypotheses rendered into sensor readings. The *Render-Match-Refine* paradigm by Stevens et al. [100] applied iterative optimization method to find a rendering that best explains the input. Their early work demonstrated Render-Match-Refine for 2D images, relying upon low-level feature extraction. Recent work uses convolutional neural networks (CNNs) to compare rendered and observed images. Among this work, the approach by Krull et al. [51] casts the CNN as a probabilistic model to output energy value, whereas work by Gupta et al. [36] directly outputs a coarse object pose and use ICP to refine it later.However, these methods do not address multi-object pose estimation such as in cluttered scenes. Narayanan et al. [72, 73] proposed approaches to integrate exhaustive global reasoning with discriminatively-trained algorithm to perform scene estimation. However, their work assumes the identification of objects is given or provided by an idealized object detection and recognition system in order to perform A* search.

Great progress has been made by deep neural networks in object detection [86, 40], 6D object pose estimation [116, 111] and semantic segmentation [62, 80]. To be robust to false detections due to the effects of overfitting, discriminative-generative algorithms [46, 72] have been proposed for robust perception, especially in adversarial environments [13]. These efforts combine inference by deep neural networks with sampling and probabilistic inference models to achieve robust and adaptive perception. However, the robustness is at the cost of computation time and the assumption object independence.

Desingh et al. [25] and Mitash et al. [69] leverage physics into scene estimation to search object poses based on compatibility score from physics simulation. The above-mentioned methods take either geometric and physical constraints into account for searching object poses, at the cost of computational efficiency.

2.4 Semantic Mapping

From the emerging of semantic mapping [52], lots of work have explored this field with various semantic representations [50]. With the focus on the object-level semantic map, one widely used approach is to reconstruct a 3D map from either sparse features [71] or dense point clouds [76, 112, 87] and then augments the map with the objects. Civera et al. [18] and Ekvall et al. [26] used SURF/SIFT descriptors to register objects to the map created in a parallel thread. However, their work can not deal with objects in clutter. To deal with clutter, Li et al. [56] proposed an incremental segmentation approach to fuse segmentation across different frames based on dense reconstruction 3D map and used ObjRecRANSAC [81] to register object poses. But their method does not scale well with substantial false detections.

Salas-Moreno et al. [94] proposed an object SLAM system that recognized objects using Point-Pair Features in each frame and directly performed graph optimization on object and camera 6D poses. Their work shows promising results in the direction of building maps with 6D object poses. Fusion++ [65] constructed an online object-level SLAM system with previously unknown shape and perform pose graph optimization at object level. Both of their work assume independence between objects and hence difficult to work well in densely cluttered environments. Zeng et al. [119] proposed CT-MAP, which considers contextual relation between objects and temporal consistency of object poses in a conditional random field and maintain belief over object classes and poses. This approach is robust to false detections, however it does not scale well to the increasing number of objects due to the nature of pure generative inference.

2.5 Summary

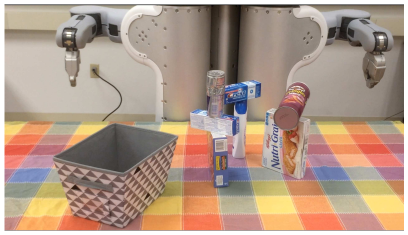
In summary, we described previous work that is related for perception for manipulation and semantic mapping. We first elaborated on the related work of symbol grounding with shared autonomy and goal-directed manipulation. We then presented the related work of perception for manipulation and scene estimation. We also discussed related work on semantic mapping.

CHAPTER 3

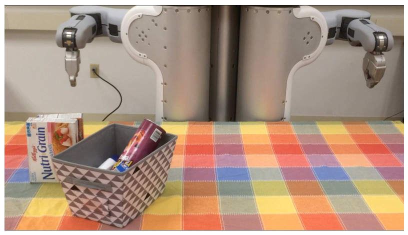
Axiomatic Scene Estimation

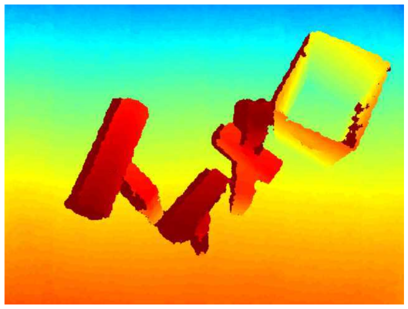
In this chapter, we describe the problem of *Axiomatic Scene Estimation (AxScEs)*, pronounced "access", for robot manipulation in cluttered scenes given raw sensor observations. We propose generative approaches to scene inference (as the axiomatic particle filter, and the axiomatic scene estimation by Markov chain Monte Carlo based sampler) of the robot's environment as a scene graph. The result from *AxScEs* estimation are axioms amenable to goal-directed manipulation through symbolic inference for task planning and collision-free motion planning and execution.

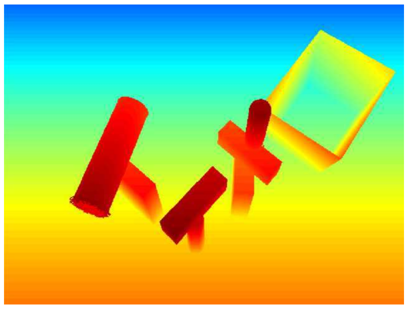
Figure 3.1 shows goal-directed manipulation in action from an *AxScEs* estimate of a cluttered scene of eight objects with a Willow Garage PR2 robot. We phrase the problem of *AxScEs* as the estimation of a tree-structured scene graph describing the configuration of objects observed from robot sensing. Similar to their use in computer graphics, scene graphs are represented as parameterized axiomatic statements that assert relationships between objects in the scene graph and the poses and geometry of each object. Our generative approach to inference for problems of *AxScEs* iteratively hypothesizes possible scene configurations (as lists of axioms) and evaluates these hypotheses against the robot's observations (currently as depth images). The result of this inference is an approximate posterior probability distribution over possible scenes, where the scene with maximum likelihood is taken as the estimate of the scene configuration. Though probabilistic in nature, a principal motivation for providing *AxScEs* estimates represented in axiomatic form is the "closing the loop" between goal-directed symbolic planners [30, 55] and modern robotics. Such planners reason over manipulation actions for the robot to execute from an *AxScEs* estimate

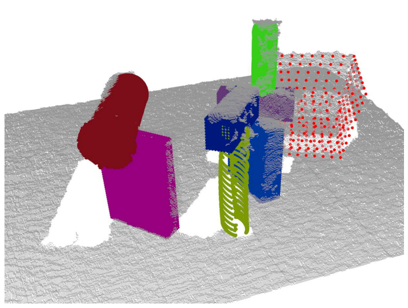

(a) robot in initial scene

(b) known geometries for each object


(c) achieved user-specified goal scene


(d) depth image observation of initial scene

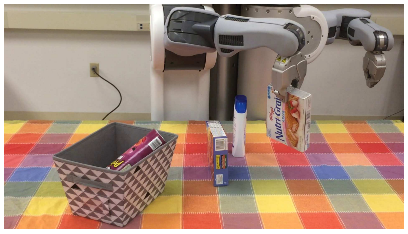

(e) estimated initial scene in depth image

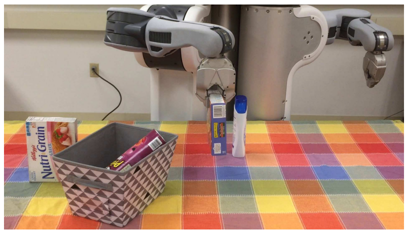

(f) estimated initial scene in point cloud

(g) action: pick red bull, place bin

(h) action: pick toothpaste box, place bin

(i) action: pick pringles, place bin


(j) action: pick nutrigrain, place side region


(k) action: pick nature_valley, place side region

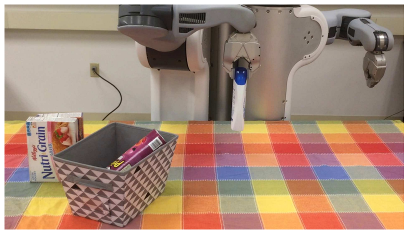

(l) reach goal: pick shampoo, place bin

Figure 3.1: An example of *AxScEs* estimation and goal-directed manipulation of a (a) cluttered scene with (b) eight objects to a (c) goal state of putting small objects in a bin and boxes to the side. *AxScEs* estimation generatively infers from robot observations (d) the maximally likely scene configuration (e,f). The resulting *AxScEs* is used to plan and execute actions (g-l) to the user-specified goal state (c).

towards realizing a given goal scene, also expressed axiomatically. Planned sequences of actions are then executed by motion control/planning systems for pick-and-place manipulation [17] or general manipulation affordances [38].

The remainder of this chapter describes the problem of axiomatic scene estimation, proposes instances of *AxScEs* estimators, and examines their use for goal-directed manipulation in scenes of increasing numbers of physically stacked objects. Section 3.1 motivates the need for generative approaches to problems of *AxScEs* as a matter of inclusion towards bridging probabilistic and symbolic inference for goal-directed manipulation. We describe a formulation for the problem of axiomatic scene estimation in Section 3.2. An analysis of the growth of possible tree-structured scene graphs is presented with respect to the number of stackable objects in a scene. In Section 3.2.3, we phrase the problem *AxScEs* as a probabilistic state estimation model that factors into the inference of scene tree relations and object pose. Within the *AxScEs* model, we propose the *Axiomatic Particle Filter (AxPF)*, as an exhaustive search over scene tree relations with particle filter inference of poses in Section 3.3. We also introduce *Axiomatic Scene Estimation by MCMC Sampling (*AxMC *)* using the Metropolis-Hastings algorithm [39] to search the scene tree relations with pose estimation likelihoods. Our GPU-optimized parallel implementation of *AxScEs* estimation for both the *AxPF* and *AxMC* are described in Section 3.5. This likelihood works directly with depth images from common ranging cameras without the need for computing discriminative features. Our experiments with this implementation are described in Section 3.6. These results indicate that *AxScEs* estimators are effective for manipulation-quality perception based on edit distance on scene graphs, estimation of currently manipulatable (clear) objects, and object pose estimation.

3.1 Motivation for Generative Approach

To motivate the problem of *AxScEs* , consider the scene in Figure 3.2 observed from 3D point clouds captured from the robot's perspective. For this scene, assume the goal for the robot is to grab the bottom green block to give to a human user. It can be clearly observed that *block1* (the top block) and *block2* (the bottom block) are two distinct objects from the perspective of human perception. A naïve perception of this scene, common to most robots, would instead perceive objects that are physically touching as a single object as shown in the right. From the perspective of common segmentation methods for 3D point clouds, estimates of the scene as two smaller objects or a single larger object are equally likely parsings of the robot's observations.

To capture this uncertainty, our approach to problems of *AxScEs* is to maintain a distribution across plausible scene graph hypotheses supported by the robot's point cloud observations. These generated hypotheses form an approximate probability distribution (or belief) over possible states of the scene. This ambiguity over possible scenes can be resolved at a later time with further information, such as after a robot action to grasp one of the objects. In addition, by maintaining diverse perspectives, the robot can use either one of these hypotheses as an estimate of the scene state to plan and execute a current course of action. If the chosen state estimate was incorrect, the alternate hypothesis of the scene should still be represented in the diversity of the belief distribution. Assuming the result of the action resolved the ambiguity, this alternate state hypothesis will now have a greater likelihood given the new point cloud observation. This distribution will now clearly distinguish the alternate as the true scene state estimate from which the robot's plan can be recomputed. This approach to scene estimation is implemented by a system architecture, whose details are described in Section 3.4.

Our approach to *AxScEs* aims to emulate and scale the highly effective and now ubiquitous pattern of decoupled decision making and probabilistic inference for autonomous navigation [23, 9] in the domain of robot manipulation. Specifically, probabilistic perception and symbolic planning are treated as independent processes, allow each to focus on what they do best. These processes interoperably interface through the communication of the maximally likely axiomatic state estimate and selected robot action (or operator). While avoiding the intractability of planning in the space of beliefs, this decoupling assumes state estimates are a plausibly accurate representation of the current state of the world.

Unlike autonomous navigation, *AxScEs* estimation faces a drastically large state space where generative inference must balance estimation accuracy and computational tractability. The scene dimensionality grows rapidly towards intractability as the number of objects in the scene increases,

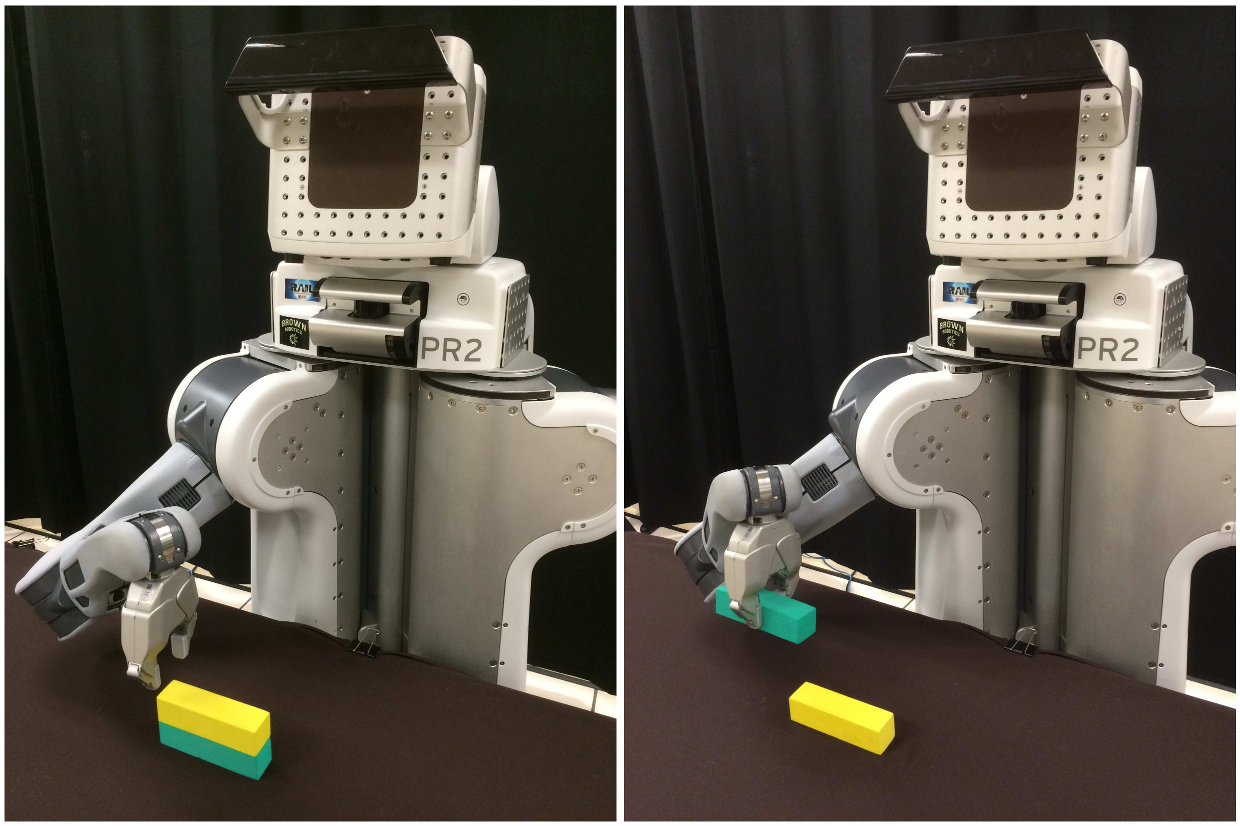
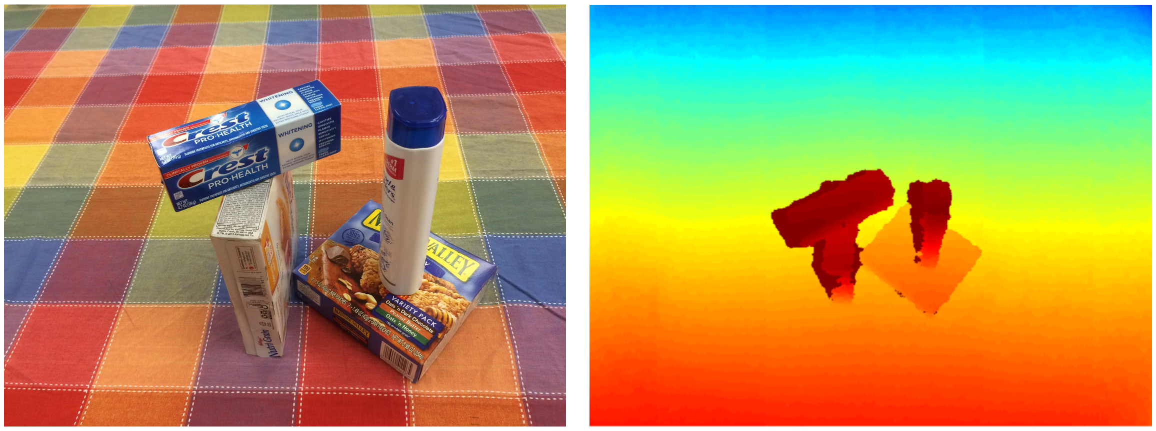


Figure 3.2: Example of a robot needing to grasp an object (the green block) in a simple stack scene. The robot needs to estimate the scene but faces ambiguity about whether there are two stacked objects or one larger object. Once estimated, the robot needs to perform a sequence of actions to move the yellow block and then, once cleared, grasp the green block.

exceeding factorial growth. Such expansive state spaces prohibit exhaustive search over scenes, even with an optimized processing pipeline. Further, our search space consists of state variables with mixed types over both non-binary tree structures and real-valued object pose parameters.As described later, we explore sampling-based algorithms, including Markov Chain Monte Carlo (MCMC) and particle filtering, suited to the diverse types and high-dimensionality of axiomatic scenes.

3.2 Problem Statement and Formulation

The objective of axiomatic scene estimation is to infer a symbolic description of the scene $\hat{\text{S}}$ from partial observations z_t by a given robot at time t. This symbolic scene description can then be read-



(table object)	(nat_val object)
(pose table pos1)	(pose nat_val pos2)
(Nutrigrain object)	(geom nat_val geom_nv)
(pose Nutrigrain pos4)	(on nat_val table)
(geom Nutrigrain geom_nut)	(shampoo object)
(on Nutrigrain table)	(pose shampoo pos3)
(toothpaste object)	(geom shampoo geom_sp)
(pose toothpaste pos5)	(on shampoo nat_val)
(geom toothpaste geom_tp)	(clear shampoo)
(on toothpaste Nutrigrain)	
(clear toothpaste)	(free right_hand robot)
(geom table geom_table)	(free left_hand robot)

Figure 3.3: Axiomatic scene estimation for an example four-object scene (top left), observed by the robot as a depth image (top right), will estimate the pose and spatial relations of objects as parameterized axiomatic assertions.

ily used by modern task and motion planners to generate sequential actions that will autonomously control the robot to achieve a user-expressed desired goal state $\mathbf{S}_G$.

Axiomatic state x_t at time t is defined as a collection of axioms expressing possible scenes $\mathbf{S}$. A scene is expressed as a scene graph $\mathbf{S}(\mathbf{W}(\mathbf{Q}, \mathbf{V}))$ as a set of axiomatic assertions describing the pose Q_i and geometry V_i of each object W_i and relations for object interactions and affordances. The Planning Domain Definition Language (PDDL) [66] is used to model axiomatic state as a formal language, which implicitly defines a tree-structured scene graph. Shown in Fig. 3.3, an example scene graph of a four object scene is represented in PDDL. To avoid ambiguity, we restrict the set of axioms to only spatial and physical expressions that can be tested geometrically or through physical simulation. These axioms assert the existence of an object W_i, as *(*W_i `object`*)*, with spatial geometry V_i, *(*`geom` W_i V_i*)*, and spatial pose configuration Q_i, *(*`pose` W_i Q_i*)*. Axioms also assert parent-child relationships between objects as whether an object W_i is inside another object W_j, *(*`in` W_i W_j*)*, or resting on another object W_k, *(*`on` W_i W_k*)*, as well as whether the object is the possession of a robot R, *(*`has` R W_i*)*. Each of these inter-object relations induces a spatial frame relation, where the frame of a child object is expressed in relation to its parent object.

Given the *AxScEs* relations above, the relation *(*`clear` W_i*)* is asserted for each object W_i that is not supporting another object. The objects asserting this relation can be picked by the robot or used as a support surface for placing other objects. For general manipulation affordances, additional axioms can be created that describe assertions for preconditions and postconditions for actions associated with objects. Precondition and postconditions axioms are envisioned to resemble collision-based "trigger" conditions, widely used to script interactive behaviors in video games through programming languages such as Lua [68].

3.2.1 Assumptions

For the methods presented in this article, we address the problem of *AxScEs* for inferring the axioms in the scene $\mathbf{S}$ and the 3 DOF poses of each object $\mathbf{Q}$. The n objects comprising $\mathbf{W}$ are assumed to have been uniquely identified with each having known spatial geometries contained in $\mathbf{V}$. This assumption is made based on using an ideal of common visual object recognition systems [28, 19] as a preprocessing step. Only the inter-object relation for stacking (`on`) is considered for *AxScEs* estimation, although the relations for enveloping (`in`) and grasping (`has`) are considered for task planning. Objects are assumed to be an upright oriented and can take on any pose on the support surface provided by its parent object. As such, the object poses of $\mathbf{Q}$ consist of a 2D position and yaw rotation (SE(2) group) in the coordinates provided by its parent object.

3.2.2 Scene Graph Enumeration

In the general case, scene estimation in this axiomatic form can lead to a very high dimensional belief space that would theoretically pose problems for probabilistic inference. For this work, we will assume scene graphs are tree-structured, have an implied base support plane, and consider only the stacking case (asserted by the ”on” relation). Thus, a single object could physically support any number of other objects, but is itself physically supported by one other object. With these assumptions, let $T(n)$ be the number of possible scene graphs, given n objects in a scene. Then, the total number of scenes can be expressed recursively:

$$T(n) = \sum_{k=1}^{n} {}^{n}C_k g(n-k,k) \tag{3.1}$$

where ${}^{n}C_k$ is the number of combinations for selecting a subset of k objects out of the n objects and $g(s,k)$ is the number of scene graphs possible for stacking s objects on top of a fixed scene on k base objects:

$$g(s,k) = \sum_{s_1=0}^{s}\sum_{s_2=0}^{s} ... \sum_{s_i=0}^{s} ... \sum_{s_k=0}^{s} \frac{s!}{s_1!s_2!...s_i!...s_k!} T(s_1)T(s_2)...T(s_i)...T(s_k) \tag{3.2}$$

where $s = \sum_{i=1}^{k} s_i$

When $n = 0$, $T(0) = 1$ as the number of scenes with no objects as 1. Similarly when $n = 1$, $T(1) = 1$ expresses the number of scenes with 1 object 1. When $n = 2$, $T(2) = 3$ breaks down into two terms ${}^{2}C_1 * g(1,1)$ and ${}^{2}C_2 * g(0,2)$ with respect to the recursion in Equation 3.2. The first term considers the number of ways 1 object can be placed on 1 supporting object $g(1,1) = 1$ times the number of ways each of the 2 objects to each of these stacking roles ${}^{2}C_1 = 2$. $g(0,2) = 1$ is the number of ways 2 objects can be placed on the table, for which there is only one combination for stacking. Relationally, $T(2) = 3$ expresses the 3 possible axiomatic scene graphs for 2 objects: objects A and B are not stacked, object A is stacked on object B, object B is stacked on object A.

When $n = 3$, $T(3) = 16$ has three terms ${}^{3}C_1 * g(2,1)$, ${}^{3}C_2 * g(1,2)$ and ${}^{3}C_3 * g(0,3)$. First term ${}^{3}C_1 * g(2,1)$ denotes the number of ways of choosing 1 object out of 3 objects, $g(2,1)$ denotes the number of ways 1 object can be placed on two supporting objects. Computing each of the terms turns out to be 9, 6, and 1 respectively and hence $T(3) = 16$.

Following this recursive expression of Equation 3.2, $T(3) = 16$, $T(4) = 125$, $T(5) = 1296$, $T(6) = 16807$, $T(7) = 262144$, $T(8) = 4782969$, and so on. This recursive expansion provides an upper bound as it assumes each object is capable of providing a support surface for all other objects. However, we speculate that most of these scene possibilities are actually implausible physically and statistically improbably to be encountered in common manipulation settings and human environments. That stated, naïvely performing state estimation in such a huge state space becomes intractable quickly as the number of objects grows. Such inference can still be of considerable use for manipulation as tabletop environments can often consist of small stacks of objects. In such cases, a tabletop segmentation algorithm, such as for the PR2 Interactive Manipulation [17] can be used to identify clusters of stacked objects, each of which can be treated as their own scene.

3.2.3 Formulation

For axiomatic scene estimation, we represent the configuration of a scene at given time $\mathbf{S}_t$ as a random state variable x_t to be inferred from a history of robot observations $z_{1..t}$. This scene state variable $x_t = [g_t, q_t]$ is comprised of both real-valued object poses, as random variable $q_t \in \Re^3$, and set-valued lists of axioms, as random variable g_t. In our case, the axioms g_t define the topology of objects in a scene as a tree.

We decompose the *AxScEs* problem as an expression of the probability of a scene into pose q_t and scene tree structure g (assuming a static scene for clarity):

$$
\begin{aligned}
p(x_t|z_{1:t}) &= p(g, q_t|z_{1:t}) && (3.3)\\
&= p(g|z_{1:t})p(q_t|g, z_{1:t}) && (3.4)\\
&\propto p(g|z_{1:t})p(g, z_t|q_t)p(g|z_{1:t}) \int p(q_t|q_{t-1}, u_{t-1})p(q_{t-1}|g, z_{1:t-1})\mathrm{d}q_{t-1} && (3.5)\\
&\approx p(g|z_{1:t})p(g, z_t|q_t) \sum_j w_{t-1}^{(j)} p(q_t|q_{t-1}^{(j)}, u_{t-1}) && (3.6)
\end{aligned}
$$

The expressions decompose the problem of *AxScEs* into a scene tree factor $p(g|z_{1:t})$ and an object pose factor $p(q_t|g, z_{1:t})$. For inference, we assume the scene tree factor is unknown and treat the object pose factor as a likelihood of pose given a scene tree. Inference of the scene tree structure allows maintenance of the distribution over scenes in relation to object poses over time, approximated by particle filtering. For goal-directed planning, our primary concern is obtaining a scene estimate $\hat{\mathbf{S}}$ from these distributions, which leads to our formulation of *AxScEs* :

$$\begin{aligned} \hat{\mathbf{S}} &= \arg\max_{x_t} p(x_t|z_{1:t}) && (3.7) \\ &= \arg\max_{g,q_t} p(g|z_{1:t})p(q_t|g, z_{1:t}) && (3.8) \\ &\approx \arg\max_{g,q_t} p(g|z_{1:t})\Big[\arg\max_{q_t} p(g, z_t|q_t)\sum_j w_{t-1}^{(j)} p(q_t|q_{t-1}^{(j)}, u_{t-1})\Big] && (3.9) \end{aligned}$$

3.3 Methods

We first propose a brute force method, the *Axiomatic Particle Filter (AxPF)*. The *AxPF* exhaustively marginalizes over combinations of scene axioms g_t and performs inference over object poses q_t through particle filtering on robot observations z_t. In the context of the *AxPF*, we additionally explore object pose estimation using MCMC sampling.

Avoiding exploration over all possible scenes, another approach to *AxScEs* inference is to search over scenes with algorithms amenable to general data structures, such as a hill climbing optimization or MCMC algorithm. Similar to [61], such an inference procedure samples over possible scenes g_t where pose estimation on q_t on robot observations z_t is performed for each sampled scene. Our proposed *Axiomatic Scene Estimation by MCMC sampling (AxScEs MCMCs)* takes this form. *AxMC* performs scene inference of g_t with the MCMC-based Metropolis-Hastings algorithm and pose inference of q_t with a particle filter. *AxMC* works directly on depth images without the need for discriminative features, as used by [61] or [19]. Further, *AxMC* provides distributions over both scene structure and object poses, which conceptually allows for updates over time as we consider for future work.

3.3.1 Axiomatic Particle Filter

For the Axiomatic Particle Filter (*AxPF*), we modeled the inference of axiomatic state x_t from a history of robot observations $z_{1:t}$ as a sequential Bayesian filter. This model consists of updating a prior belief from time $t-1$ with a dynamic resampling and likelihood evaluation to form a new posterior belief at time t:

$$p(x_t|z_{1:t}) \propto p(z_t|x_t) \int p(x_t|x_{t-1}, u_{t-1}) p(x_{t-1}|z_{1:t-1}) \mathrm{d}x_{t-1} \tag{3.10}$$

Although results presented are for observations of static scenes, the dynamics term $p(x_t|x_{t-1}, u_{t-1})$ in this formulation is to retain generality for tracking the scene as the robot performs an action u_{t-1}. As described by [23], the sequential Bayesian filter in Equation 3.10 is commonly approximated by a collection of N weighted particles, $\{x_t^{(j)}, w_t^{(j)}\}_{j=1}^N$, with weight $w_t^{(j)}$ for particle $x_t^{(j)}$, expressed as:

$$p(x_t|z_{1:t}) \approx p(z_t|x_t) \sum_j w_{t-1}^{(j)} p(x_t|x_{t-1}^{(j)}, u_{t-1}) \tag{3.11}$$

Over successive iterations, inference in the particle filter is performed by drawing N scene hypotheses by importance sampling, evaluating the likelihood of each hypothesis, and normalizing the weights to sum to one:

$$x_t^{(j)} \sim \sum_j w_{t-1}^{(i)} p(x_t|x_{t-1}^{(i)}, u_{t-1}) \tag{3.12}$$

$$w_t^{*(j)} = p(z_t|x_t^{(j)}) \tag{3.13}$$

$$w_t^{(j)} = \frac{w_t^{*(j)}}{\sum_k w_t^{*(k)}} \tag{3.14}$$

Within the likelihood $p(z_t|x_t^{(j)})$, the scene $\mathbf{S}^{(j)}$ associated with each particle $x_t^{(j)}$ is rendered into a depth image $\hat{z}_t^{(j)}$, through the z-buffer of a 3D graphics engine, for comparison with the robots current observation z_t:

$$p(z_t|x_t^{(j)}) = e^{-\lambda_r \cdot \mathrm{SSD}(z, \hat{z}_t^{(j)})} \tag{3.15}$$

where λ_r is a constant scaling factor and $\mathrm{SSD}(I, I')$ is the sum of squares distance function (SSD) between depth images I and I':

$$\mathrm{SSD}(I, I') = \sum_{(a,b)\in z} (I(a,b) - I'(a,b))^2 \tag{3.16}$$

where a and b are 2D image indices. Once the posterior distribution converges about a single scene hypothesis, the scene $\hat{\mathbf{S}}_t$ from the most likely particle $\hat{x}_t$ is taken as the scene estimate:

$$\hat{\mathbf{S}}_t = \arg\max_{x_t^{(j)}} p(x_t^{(j)}|z_{1:t}) \tag{3.17}$$

This axiomatic scene estimate $\hat{\mathbf{S}}_t$ is used for planning robot actions and motion towards a given goal state $\mathbf{S}_G$, which is also expressed in axiomatic form.

3.3.2 Axiomatic Monte Carlo Markov Chain

We now describe *AxMC* as a method to perform axiomatic scene estimation using MCMC and particle filtering over, respectively, scene trees g and object poses q_t. With respect to Equation 3.4, we cast inference of the unknown scene tree factor $p(g|z_{1:t})$ as the target distribution for MCMC sampling. Just as in the *AxPF*, particle filtering is performed on the pose factor $p(q_t|g, z_{1:t})$ and is treated as a likelihood for a sampled scene tree. The *AxScEs* estimate $\mathbf{S}_t$ is taken as the scene tree and pose associated with the maximally likely sample from the *AxMC* process.

MCMC sampling uses the single-site Metropolis-Hastings algorithm to approximate the target distribution. In each iteration of Metropolis-Hastings, a proposal distribution $p'(g^*|g^{(i)})$ is used to generate a new sample g^* local to the previous sample $g^{(i)}$. As we describe in Section 3.3.2.1, a common instance of Metropolis-Hastings for a real-valued vector space has this local sampling using a normally distributed proposal.

For our tree-valued variable g, generation of a proposal sample $g^* \sim \mathcal{T}(g^{(i)})$ occurs with respect to a tree kernel $\mathcal{T}(g^{(i)})$ that performs a single random edit to the tree $g^{(i)}$. The sampling of $\mathcal{T}(g^{(i)})$ randomly selects two different nodes, a and b, of $g^{(i)}$ to perform one of three permutation operations, also selected at random: 1) swap a and b, 2) move a to be the child of b, and 3) move b to the child of a. These operations are carried out by changing the `on` relations for objects associated with tree nodes for a and b.

The newly sampled scene tree g^* is either accepted or rejected with probability:

$$A(g^{(i)}, g^*) = \min \left\{ 1, \frac{\underset{q_t}{\arg\max} p(q_t|g^*, z_{1:t})}{\underset{q_t}{\arg\max} p(q_t|g^{(i)}, z_{1:t}))} \right\} \tag{3.18}$$

based on the respective maximally likely pose estimates for each scene tree. In other words, the sample g^* is accepted if $A(g^{(i)}, g^*)$ is greater than a uniformly generated random number between zero and one. After accepting a fixed number of N samples $G^* = \{g^{(i)}\}_{i=1}^N$, the scene tree estimate $\hat{g}$ is taken as the sample with the highest likelihood with respect to the likelihood over pose estimates q_i, as expressed in Equation 3.9.

3.3.2.1 MCMC Pose

As an alternative to particle filtering, we additionally investigated an MCMC approach to pose estimation of q_i as a likelihood for known scene graph g. This pose inference used a single-site Metropolis-Hastings algorithm to approximate the target distribution $p(q|z, g)$ as a sampled Markov chain, where g is a known set of scene graph axioms. In each iteration of Metropolis-Hastings, a proposal distribution $p'(q^*|q^{(i)})$ is used to generate a new sample $q^* \sim \mathcal{N}(q^{(i)}, \Sigma_q)$ from a normal distribution centered on the previous sample $q^{(i)}$ with covariance Σ_q over the space of all pose dimensions. Alternatively, this sampling can be done per object with normal distributions in the space of DOFs for each object. The newly sampled particle q^* is either accepted or rejected with acceptance probability:

$$A(q^{(i)}, q^*) = min\Big\{1, \frac{w(q^*)}{w(q^{(i)})}\Big\} \tag{3.19}$$

where $w(q)$ is the likelihood of pose state q, as specified in Equations 3.13 and 3.15.

After accepting a fixed number of N samples $Q^* = \{q^{(i)}\}_{i=1}^N$, the pose estimate $\hat{q}$ is taken as the sample with the highest likelihood with respect to the likelihood w:

$$\hat{q} = \underset{Q^{*(i)}}{\arg\max} w(Q^{*(i)}) \tag{3.20}$$

After several rounds of informal testing, we chose to focus on pose inference by particle filtering due to significantly better estimation accuracy. We attribute this preference to the relative ease of tuning parameters of the particle filter predictive density in comparison to MCMC proposal covariance Σ_q. The remainder of the discussion in this article will assume pose estimation by particle filtering, although MCMC pose estimation can be performed instead without loss of generality.

3.4 System Architecture

Our implementation for *AxScEs* perception of scenes and goal-directed manipulation is discussed in this section. This implementation follows the architecture outlined in Figure 3.4. The core of this implementation is the particle filter object pose estimation that follow a module flow of *prediction*, *diffusion*, *measurement* and *resampling*. The distribution over object poses is represented as a mixture model of particles. The *measurement* module additionally performs comparisons of relative likelihood across estimations from other scenes. We store object geometries in a database, which are expressed with respect to the parent object frame once retrieved. We further assume that invalid samples, where the center of mass for a child object is outside the support surface of its parent, is disregarded.

As shown in Figure 3.4 the *measurement* module gets the observation from the robot and hy-

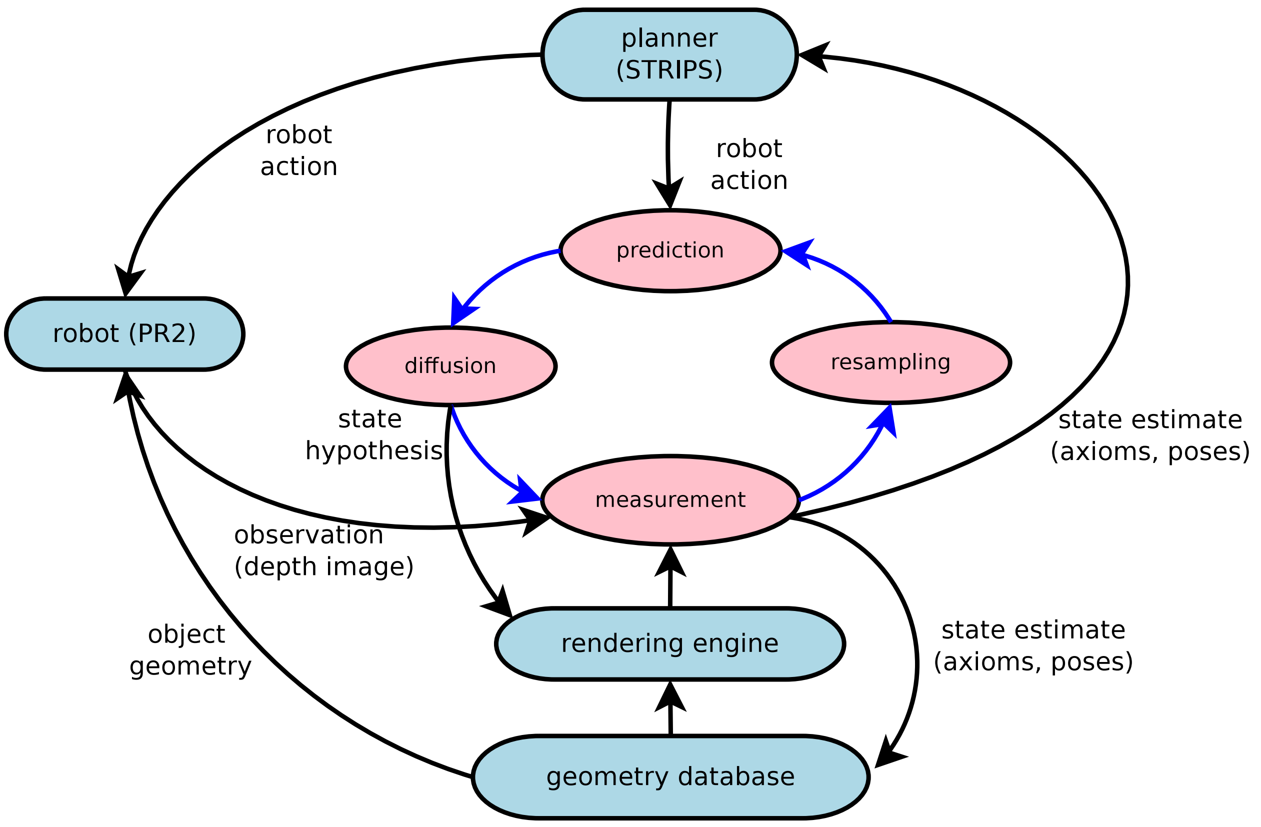


Figure 3.4: Architecture diagram for pose estimation within goal-directed manipulation with respect to the *AxPF* . Pose estimation and manipulation components are respectively highlighted in red and blue.

pothesized particles generated from the rendering engine. Robot observations are in the form of depth images from a Microsoft Kinect depth camera mounted on the head of a Willow Garage PR2 robot. The likelihood of a particle is calculated by comparing the depth images of the observation and a graphical rendering of the axiomatic state hypothesized by a particle. The comparison function is a sliding window sum of squared distance (SSD) on two images. The z-buffer of an OpenGL-based graphics rendering engine is used to generate depth images from axiomatic states. We assume a known intrinsic calibration and extrinsic pose for the Kinect camera and that all the object geometries are known and stored in a geometry database.

The principal output of the *measurement* module is the posterior distribution representing the distribution of belief for the current state of the world. If the particles converge within a threshold, the *planner* takes the maximum likely state estimate and computes a plan of action for the robot to execute. In parallel, the *resampling* module takes in the posterior distribution and performs important sampling over their states to give the new distribution of particles to the *prediction* module. Based on the robot action decided by the *planner* the *predict* module updates the state of the particles. The *diffusion* module adds noise randomly to this distribution of particles and *measurement* is performed again with a new observation from the robot. The *diffusion* module also updates the rendering engine with a new set of axiomatic states to generate particles.

A STRIPS-based system [30] was used for sequential planning in our manipulation system. With the goal and the current state of the world, the *planner* would compute a sequence of actions towards the goal and outputs the next immediate action to the robot. Actions from the planner will be pick-and-place actions for a specific object in the scene. Given this object's pose and geometry, from the *geometry database*, PR2 Tabletop Manipulation [17] is used to execute these manipulation actions.

3.5 Parallelized likelihood evaluation

As described in 3.2.2, the complexity of scene graph enumeration quickly grows beyond computational tractability. In practice, however, we can address this inference computationally viable through parallelization and constraining the space of physically viable scene estimates ([25]). Described below is one method for parallel sample generation and likelihood evaluation through leveraging hardware graphics rasterizers in modern GPUs. This parallelization provides performance beyond that is offered through general-purpose computing on graphics processing units.

We consider parallel rendering based on OpenGL ([96]) to simulate depth cameras and generate scene estimation particles rapidly. This renderer sets up the rendering pipeline using camera extrinsic and intrinsic parameters, object geometries, and estimated object transformations. During each particle filter iteration, the OpenGL renderer renders all particles in parallel onto a single render buffer, which is then passed to CUDA kernels for computing the objective metrics of particles.

A particle is a scene consisting of objects with the same geometries but of different transformations. Each particle is specified by a draw call for its object geometries and transformations. Then draw calls for rendering each particle are separated by viewport specifications `glViewport()`, which set the positions and sizes of sub-images for particle hypotheses in the output render buffer. With credit to internal GPU work scheduling, all particles are rendered in parallel and viewport specification does not reduce the parallelism of issued draw calls.

The transformations in OpenGL draw calls include model matrices, the view matrix, and the projection matrix. Model matrices specify the transformations of the objects without changing the geometry of the objects. Model matrices are constantly updated per iteration according to the changing estimates in particles. The view matrix is the extrinsic transformation of the camera within the world coordinate and can be derived using the position of the camera and the direction of the camera. The projection matrix can be derived using the camera intrinsic parameters, including the focal length f_x, f_y, the principal point c_x, c_y, and the OpenGL clipping near and far distances, z_n, z_f. We derive the following matrix P for perspective projection:

$$P = \begin{pmatrix} \frac{2f_x}{W} & 0 & 1 - \frac{2c_x}{W} & 0 \\ 0 & \frac{2f_y}{H} & 1 - \frac{2c_y}{H} & 0 \\ 0 & 0 & -\frac{z_f + z_n}{z_f - z_n} & -\frac{2z_f z_n}{z_f - z_n} \\ 0 & 0 & -1 & 0 \end{pmatrix} \tag{3.21}$$

where W and H are the image width and height.

Similar to [16], we attach the output render buffer to a framebuffer object (FBO) for efficient off screen rendering. However, we use render buffer objects (RBO) instead of textures because multisampling features in textures are not useful for our purposes and only add overhead. We also attach a depth render buffer to the FBO, which is required for depth enabled rendering. In contrast to previous research where only color information is used, we are interested in the depth information. However, RBOs in depth format are not supported by CUDA and cannot be accessed from CUDA kernels via the OpenGL interoperation interfaces. We propose an efficient multi-step process for this by modifying the OpenGL fragment shader to compute the depth values and output as float point color values in a color formatted `GL_R32F` RBO, which can be accessed from CUDA. Note that the depth rendering process described here is similar to a part of the known deferred shading pipeline where the depth is saved in an intermediate result called Geometric Buffers([93]).

Fragment shaders have access to a built-in variable $\text{gl\_FragCoord} = (x, y, z, 1/w)$ in which w is the extra dimension of the clip-space homogeneous coordinate of the fragment. Using the perspective projection matrix P, a point $[X, Y, Z, 1]^T$ in camera coordinate will be projected to clip-space coordinate $[x, y, z, w]^T$ in which $w = -Z$, and Z is the distance from the point to the X-Y plane of the camera coordinate. $w = -Z$ has a negative sign because $[x, y, z]^T$ is converted from the right-handed camera coordinate to the left-handed normalized device coordinate. The depth in the camera coordinate is then represented by $Z = w$. Thus the depth values can be computed in the fragment shader with `color = 1/gl_FragCoord.w;`. By leveraging the fragment shader that is already part of the existing rendering pipeline, this approach obtains depth values in one pass and eliminates the overhead of extra copying from a depth RBO to a color RBO.

The color RBO containing depth values is then passed to CUDA kernels through memory mapping with no data transfer and minimal overhead. The CUDA kernels compute the squared error objective for each pixel, and rearrange the memory layout to compute the sums of errors for each particle. The sums are then normalized and used as weights in particle filter resampling. Actually, the square error objectives can also be computed in the fragment shader prior to CUDA kernels, which provides some flexibility, though it should not have a big difference on the overall performance. The process of computing scores for 625 particles takes 0.027 seconds.

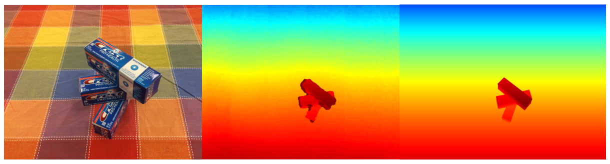
To maximize the performance of OpenGL rendering, we adopt several scene rendering best practices. [105] We use `glVertexAttribDivisor()` to specify vertex attributes format, making it only require a single model matrix for all vertices of an object. We also use the OpenGL extension `ARB_multidraw_indirect`, which allows drawing of multiple objects in a scene with a single draw call provided with parameters of multiple draw commands. With this extension, more objects in a scene would no longer require more draw calls, and all object geometries in all scenes/particles and their draw commands can be constructed and uploaded to GPU as static data during initialization. During particle filter iterations, only the model matrices will need to be updated, and the draw calls with fixed parameters reissued.

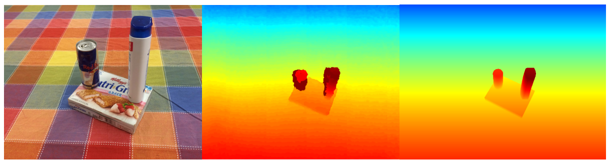
To validate the correctness of the depth value obtained by the OpenGL renderer, we also implemented a separate renderer based on Nvidia OptiX [83] ray tracing engine. We use the OptiX Prime API to implement the renderer, which solely executes on GPU compute nodes without the help of hardware rasterizers. The OptiX renderer submits parallel rendering queries that contain the scene geometries, transformations, and ray specifications corresponding to each pixel. Experiments with a toy scene of three cubes on a table show less than 10^{-5} (meter) average error in the results between OpenGL and OptiX renderers, which can be mostly attributed to floating number error. However, the performance of the OptiX renderer is much worse than the OpenGL renderer. To render 1000 images of 512 by 424 resolution, the OptiX renderer took 0.124 seconds, while for 1024 images of the same resolution, the OpenGL renderer took less than 0.005 seconds. This is because the OpenGL renderer takes advantage of the power of hardware rasterizers, while the OptiX renderer is limited to per-pixel computation on GPU compute processors.

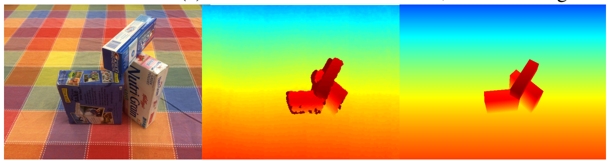
3.6 Results

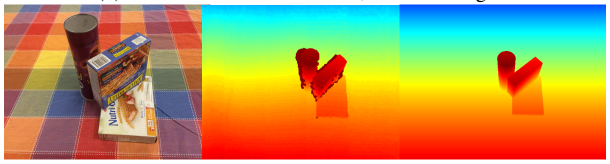
In this section, we examine our *AxScEs* estimators, the *AxPF* and the *AxMC*, with respect to 20 test scenes of interacting objects from a depth camera. These objects are common to households and vary in dimensions and geometries, as shown in Figure 3.1b. We first report the results of particle filter inference on object poses, which serve as the foundation for the inference methods over scene graphs by both *AxPF* and *AxMC* . Results are then presented for exhaustive search by the *AxPF* over scene graph, which yields estimates with high accuracy in small collections of scenes. *AxMC* results are then presented that demonstrate tractable inference with less accuracy. All the experiments are tested on a Linux PC with Intel Core i7, 32 GB memory and an Nvidia GeForce GTX Titan X Graphic Card with CUDA 7.5.

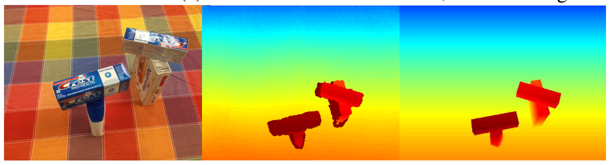
Next, we conduct three sets of experiments to demonstrate our goal-directed manipulation sys-

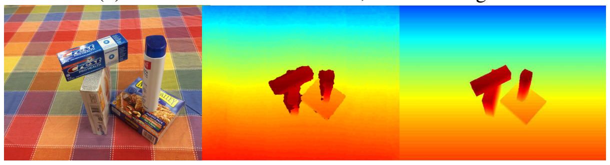

Scene (a) RMSE Translation 0.12 cm, Yaw 8.94 deg

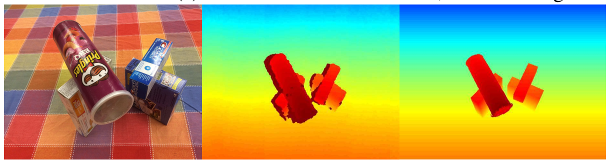

Scene (b) RMSE Translation 0.20 cm, Yaw 2.38 deg

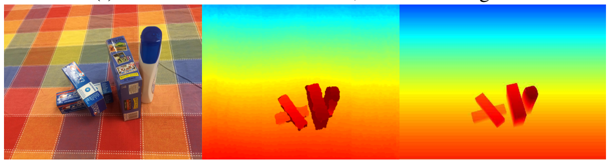

Scene (c) RMSE Translation 0.30 cm, Yaw 4.34 deg

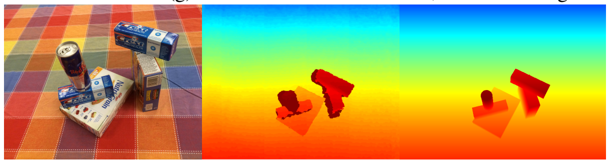

Scene (d) RMSE Translation 0.32 cm, Yaw 2.38 deg


Scene (e) RMSE Translation 0.27 cm, Yaw 9.40 deg


Scene (f) RMSE Translation 0.64 cm, Yaw 11.80 deg


Scene (g) RMSE Translation 0.34 cm, Yaw 10.90 deg


Scene (h) RMSE Translation 0.56 cm, Yaw 13.20 deg


Scene (i) RMSE Translation 0.35 cm, Yaw 8.80 deg

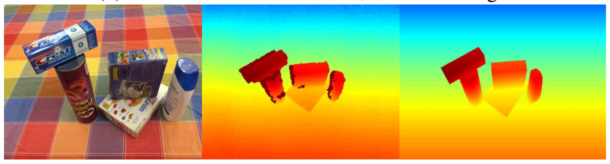
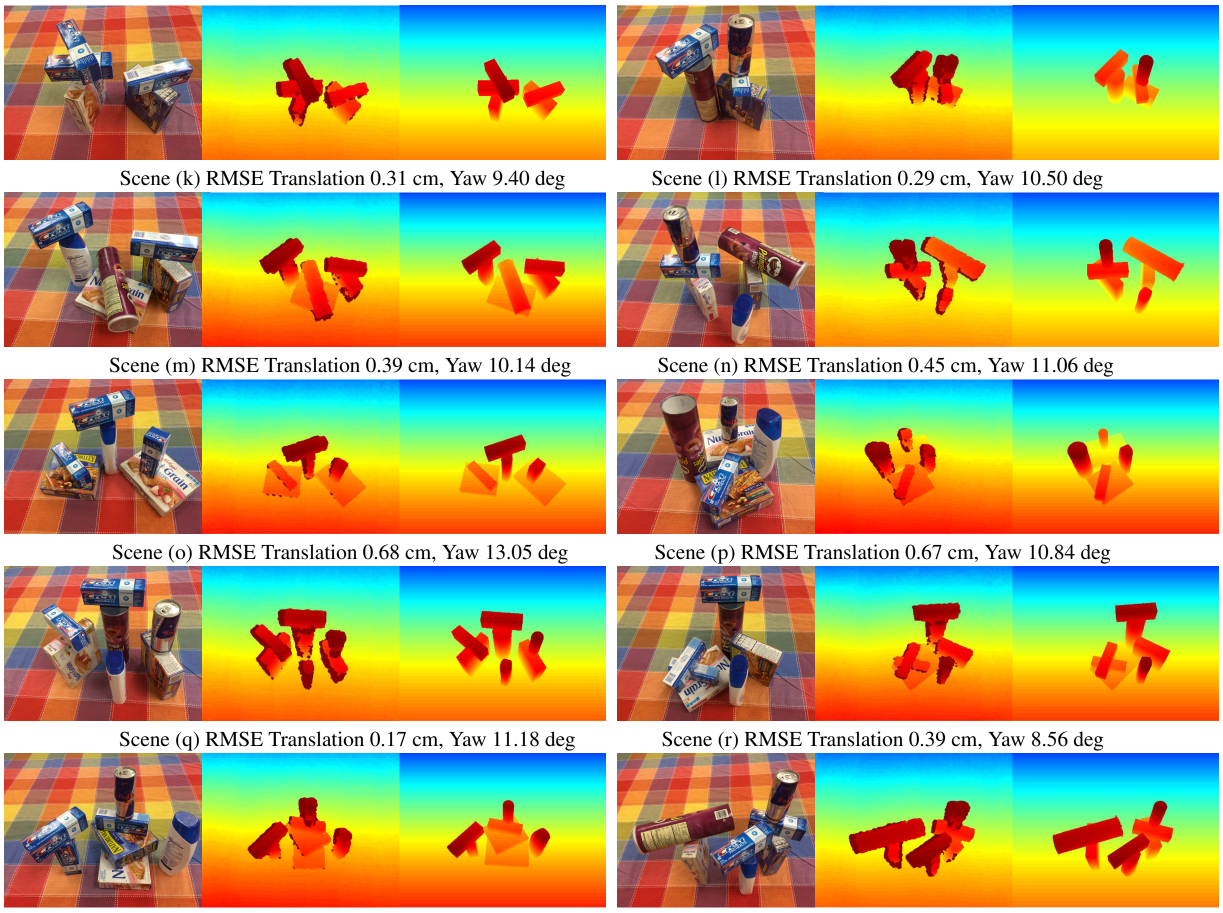

Scene (j) RMSE Translation 0.32 cm, Yaw 10.60 deg

Figure 3.5: Pose estimation results for known axiom sets in scenes containing three to four objects. Each subfigure, shows the RGB (left) and depth (middle) from an RGBD camera mounted on the head of the PR2 and estimated scene graph as a depth image (right), as well as the RMSE for translation and rotation error.

tem with *AxMC* axiomatic scene estimation. In our baseline manipulation experiment, we evaluate the manipulation system in a scenario of three blocks stacked and rotated (Figure 3.7). We then consider a more complex scenario of three stacked blocks along with a basket in the scene (Figure 3.8). At last, to test the limit of *AxScEs* , we conduct an experiment with an eight object scene as shown in Fig. 3.1. The PR2 robot was successful in achieving the goal scene: the *nature_valley* and *nutrigrain* boxes cleared to a side of the table and place all other objects into the basket.



Scene (k) RMSE Translation 0.31 cm, Yaw 9.40 deg

Scene (l) RMSE Translation 0.29 cm, Yaw 10.50 deg

Scene (m) RMSE Translation 0.39 cm, Yaw 10.14 deg

Scene (n) RMSE Translation 0.45 cm, Yaw 11.06 deg

Scene (o) RMSE Translation 0.68 cm, Yaw 13.05 deg

Scene (p) RMSE Translation 0.67 cm, Yaw 10.84 deg

Scene (q) RMSE Translation 0.17 cm, Yaw 11.18 deg

Scene (r) RMSE Translation 0.39 cm, Yaw 8.56 deg

Scene (s) RMSE Translation 0.88 cm, Yaw 10.03 deg

Scene (t) RMSE Translation 0.44 cm, Yaw 10.71 deg

Figure 3.6: Pose estimation results for known axiom sets in scenes containing five to seven objects. Each subfigure, shows the RGB (left) and depth (middle) from an RGBD camera mounted on the head of the PR2 and estimated scene graph as a depth image (right), as well as the RMSE for translation and rotation error.

3.6.1 Object Pose Estimation

In order to validate the accuracy of particle filter pose estimation, we first started by evaluating our GPU-optimized likelihood function for estimating object poses given known scene graphs. For each scene, 40 estimation trials were performed with 400 particle filter iterations with 1250 particles. As the render buffer size supported by the graphics card is 163484 x 16384 and the size of the depth image is 640 x 480, so the maximum number of images the graphics card can render at a time is 16384 / 640 = 625. Thus, our choice of 1250 particles as two times 625.

The Root Mean Square Error (RMSE) on both translation (x and y) and rotation (yaw) are

Scene	N	Tree Edit Distance		Leaf node classification			RMS Pose Error	
		Mean	Var	Acc	Prec	Rec	Trans	Yaw
Scene(a)	3	0.00	0.00	1.00	1.00	1.00	0.47	0.99
Scene(b)	3	0.00	0.00	1.00	1.00	1.00	0.38	6.47
Scene(c)	3	0.80	1.07	0.73	0.80	0.80	3.04	11.73
Scene(d)	3	0.00	0.00	1.00	1.00	1.00	0.21	0.90
Scene(e)	4	0.00	0.00	1.00	1.00	1.00	0.23	1.69
Scene(f)	4	1.40	2.71	0.80	0.75	0.90	0.62	8.56
Scene(g)	4	1.10	1.10	0.72	0.72	0.72	0.82	0.73
Scene(h)	4	0.00	0.00	1.00	1.00	1.00	0.46	8.84

Table 3.1: Metrics calculated for *AxPF* scene estimation. The first two columns are the mean and variance of the tree edit distance which is the minimum number of node operations to transform one tree to the other. From the 3rd column to the 5th column, the accuracy, precision and recall of the leaf node classification are reported. The last two columns are the pose error of translation(x and y) and yaw a correctly classified leaf node. N is the number of objects in each scene.

computed and is denoted in each scene in Figure 3.5 and Figure 3.6. The translation error remains very low for each scene (under 1 cm) but the rotation error seems a bit high. The large angular error is primarily due to the less accurate estimation of occluded supporting objects, and not due to the accounting of object symmetry. Supporting objects, higher in the scene graph, are occluded by the top objects and, thus, have fewer pixels in the observation depth image. Further, the standing objects also have fewer observed pixels, due to taking observations directly from the robot's first person viewpoint, which leads to a larger angular error. Regardless, these errors are within our observed estimate of tolerable error for grasping with the PR2. The time taken for each particle filter iteration is 0.022s and varies with different rendering objects. The total computation time for each scene is around 9.08 seconds.

This experiment demonstrates our particle filter can estimate the object poses with high accuracy and can serve as a likelihood function for scene graph estimation methods.

3.6.2 Metrics for AxScEs

For evaluating the results of our *AxScEs* methods, we used metrics related to scene graph structure, tree edit distance [120], and leaf node classification, as the correct identification of currently manipulatable objects. Tree edit distance is the minimum number of node operations to transform one scene tree to the other. This distance uses three edit operations: replace a node, insert a node, and delete a node. Tree edit distance is used to compute the distance between an estimated scene graph tree and ground truth scene graph tree, where smaller values mean two trees are closer to each other.

In a cluttered scene, the directly manipulable objects provide support for no other objects that are immediately available to be picked or placed upon. These objects, asserted by the `clear` relation, are the leaves in a scene graph tree. We care more about these objects than the support objects higher in a scene graph from an estimation perspective because they are unoccluded. As leaf node objects are picked up and moved away, the scene will become less cluttered, and the supported objects will become clearer in the eye of the robot. Towards properly estimating leaf node objects, we introduce leaf node classification, which identifies whether a node in the estimation is a correct leaf node or not. We report the accuracy, precision, and recall for this manipulation-oriented classification, as well as their pose estimation accuracy.

3.6.3 Scene Graph Estimation

3.6.3.1 *AxPF*

For each of the 20 test scenes, we then ran the exhaustive search over scene graph (Section 3.3.1) with 625 particles for each scene. Due to the prohibitive computational complexity, the *AxPF* was not considered for scenes with more than 4 objects. From table 3.1, mean and variance of the tree edit distance remain very low for all the scenes tested. The computation time of the exhaustive set of particle filters is relatively high. For scenes with three objects, the exhaustive particle filters averaged 110.80 seconds and for four object scenes, the time grew to 1318.14 seconds on average.

3.6.3.2 *AxMC*

We ran MCMC with 200 iterations. In each MCMC iteration, a particle filter estimates object poses with 625 particles over 400 iterations. The results in table 3.2 is averaged over 10 times experiment for each scene. Based on these results, we interpreted the *AxMC* to perform well for scenes of up to six objects. The tree edit distance grows linearly with the number of objects in the scene. The average accuracy of leaf node classification is 0.78 which means only one leaf node object would get wrong on average for each scene as there are maximal four objects on the top. The RMS yaw error of the leaf nodes is relatively smaller than the error from Section 3.6.1, which are computed over all the objects in the scene. This indicates the robot can grasp the top objects more robustly. For scenes with greater numbers of objects, we found that at least one object was estimated correctly in each trial. This gives room for an active approach to perception and manipulation. From an *AxScEs* estimate, the leaf object with the highest likelihood can be grasped and moved to decrease ambiguity for another round of *AxScEs* estimation.

3.6.4 Manipulation Results

In this set of manipulation experiments, *AxMC* estimation is evaluated within the goal-directed manipulation system described in the previous section. The *AxMC* will first estimate the scene and get the scene graph and the pose for each object after convergence. The system would then reconstruct the 3D scene graph in the point cloud in camera view given the pose, object geometry, and current transformation from robot base link to camera link. The planner computes a sequence of actions towards the goal axiomatic state, and executes the first action in this plan. After performing this action, the robot re-estimates and re-plans for the resulting scene to take its next action. This process loop continues until the goal scene state is achieved.

Scene	N	Tree Edit Distance		Leaf node classification			RMS Pose Error	
		Mean	Var	Acc	Prec	Rec	Trans	Yaw
Scene(a)	3	0.00	0.00	1.00	1.00	1.00	0.41	1.35
Scene(b)	3	0.30	0.90	1.00	1.00	1.00	0.38	5.23
Scene(c)	3	1.40	0.93	0.53	0.65	0.65	1.20	18.72
Scene(d)	3	1.00	1.11	1.00	1.00	1.00	0.70	0.55
Scene(e)	4	1.20	2.84	0.82	0.81	0.85	1.16	5.97
Scene(f)	4	2.20	2.18	0.72	0.68	0.85	0.81	7.33
Scene(g)	4	2.00	0.67	0.65	0.62	0.75	3.13	3.74
Scene(h)	4	1.40	1.82	0.97	0.96	1.00	0.91	7.34
Scene(i)	5	3.00	2.00	0.72	0.60	0.90	1.20	1.04
Scene(j)	5	3.30	1.34	0.66	0.71	0.73	1.72	7.60
Scene(k)	5	2.40	1.60	0.82	0.70	0.95	0.56	1.97
Scene(l)	5	2.40	1.38	0.82	0.69	1.00	2.02	0.51
Scene(m)	6	3.90	0.77	0.73	0.68	0.87	2.31	6.77
Scene(n)	6	4.30	4.23	0.59	0.58	0.70	5.88	5.92
Scene(o)	6	3.20	2.84	0.76	0.71	0.89	3.18	13.31
Scene(p)	6	3.20	1.29	0.78	0.78	0.95	3.09	7.31
Scene(q)	7	4.30	2.01	0.60	0.62	0.75	7.81	8.80
Scene(r)	7	4.10	5.88	0.81	0.79	0.93	6.30	32.54
Scene(s)	7	5.80	1.51	0.63	0.55	0.70	5.58	25.73
Scene(t)	7	5.60	2.49	0.66	0.57	0.80	4.62	15.78
Scene	8	5.80	2.34	0.50	0.50	1.00	6.94	7.85

Table 3.2: Metrics calculated for *AxMC* scene estimation. The first two columns are the mean and variance of the tree edit distance which is the minimum number of node operations to transform one tree to the other. From the 3rd column to the 5th column, the accuracy, precision and recall of the leaf node classification are reported. The last two columns are the pose error of translation(x and y) and yaw a correctly classified leaf node. N is the number of objects in each scene.

3.6.4.1 Three stacked and rotated blocks

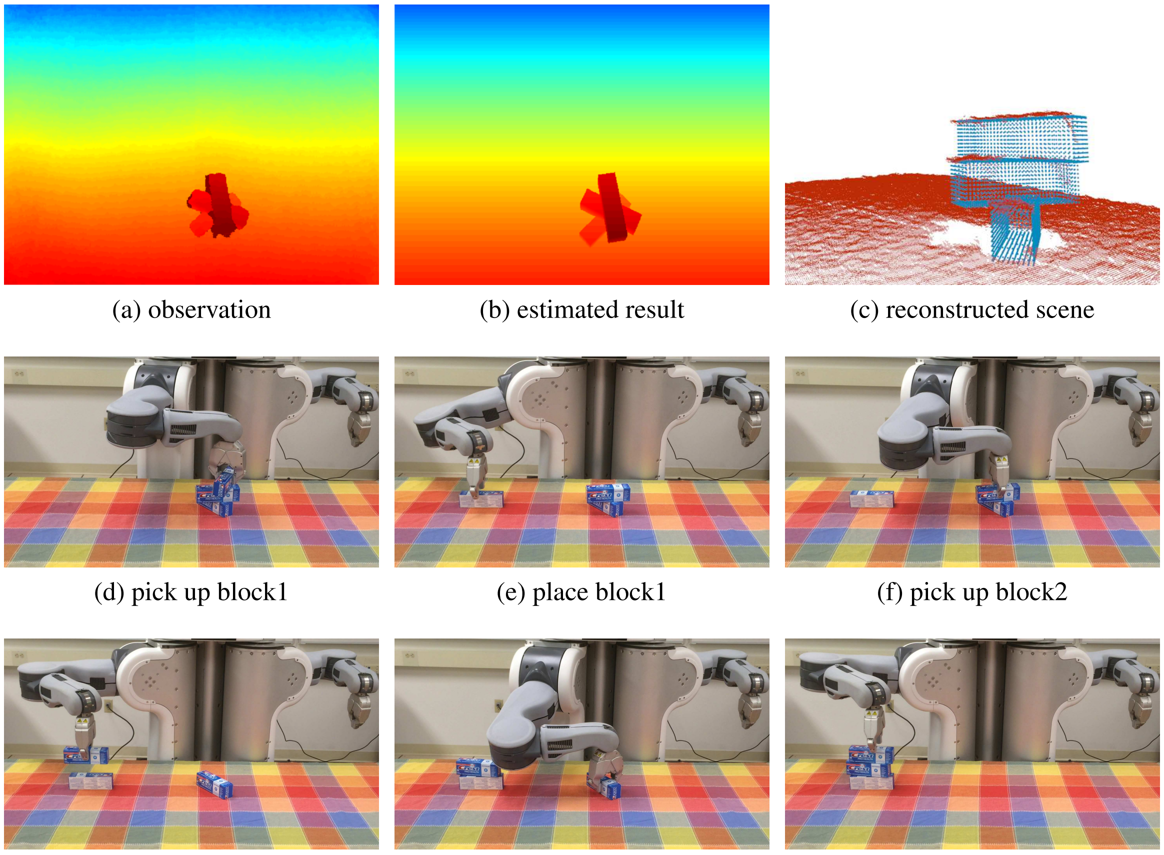
The first manipulation experiment is to rearrange three stacked and rotated blocks into a straight stack with reversed order. The observation depth image is shown in Fig. 3.7a. Fig. 3.7b shows the estimation result in the depth image and Fig. 3.7c shows the reconstructed scene in the point cloud. The STRIPS planner planned a sequence of actions towards the goal with the estimated scene graph and sent them to the robot. Then the robot executed them sequentially, shown from Fig. 3.7d to Fig. 3.7i.

3.6.4.2 Extraction of middle block

The second manipulation experiment is to extract the middle block from the three blocks sequence into a basket, which shows our system can handle complex geometries. The remaining two blocks are rearranged into a straight stack aside from the basket. Fig. 3.8a shows the perceived depth image and Fig. 3.8b and Fig. 3.8c show the estimated blocks with the basket. From Fig. 3.8d to Fig. 3.8e, the robot picked up the top block and placed it on the table. Then the middle block was picked by the robot and placed it into a basket, which is shown in Fig. 3.8f and 3.8g. Finally, the bottom block was picked and placed onto the top block, as shown in Fig. 3.8h and Fig. 3.8i.

3.6.4.3 Manipulation in Cluttered Environment

To test the limit of our approach, we conducted the manipulation experiment in a much cluttered environment with eight objects in it. The goal of the task is to place nature_valley and nutrigrain boxes to a side of the table and put all other objects into the basket. Note that as the robot gripper is not wide enough to pick up the large boxes lying on the table, nature_valley and nutrigrain are standing vertically on the table. Fig. 3.1e and Fig. 3.1f show the estimation results and scene in point cloud view. The actions performed by the robot are shown from Fig. 3.1g to Fig. 3.1l.


(a) observation (b) estimated result (c) reconstructed scene

(d) pick up block1 (e) place block1 (f) pick up block2

(g) place block2 (h) pick up block3 (i) place block3

Figure 3.7: Manipulation experiment of rearrangement of three rotated and stacked blocks. Observation and estimated depth image along with reconstructed point cloud (top row). Frames of the robot performing stacking actions to rearrange toothpaste boxes into a straight stack (bottom rows).

3.7 Summary

In this chapter, we proposed generative approaches to address the problem of *Axiomatic Scene Estimation* (*AxScEs*) as the estimation of scenes for goal-directed robot manipulation. In *AxScEs* estimation, a generative model maintains a distribution across plausible scene graph hypotheses supported by the robot's point cloud observations. These generated hypotheses form an approximate probability distribution (or belief) over possible states of the scene. We cast the problem of *AxScEs* as factors for estimating a scene graph as a tree and poses. Our *AxPF* method performs inference in this model as a brute force exhaustive search over combinations of scenes. We additionally proposed the MCMC-base *AxMC* method to avoid exploration over all possible

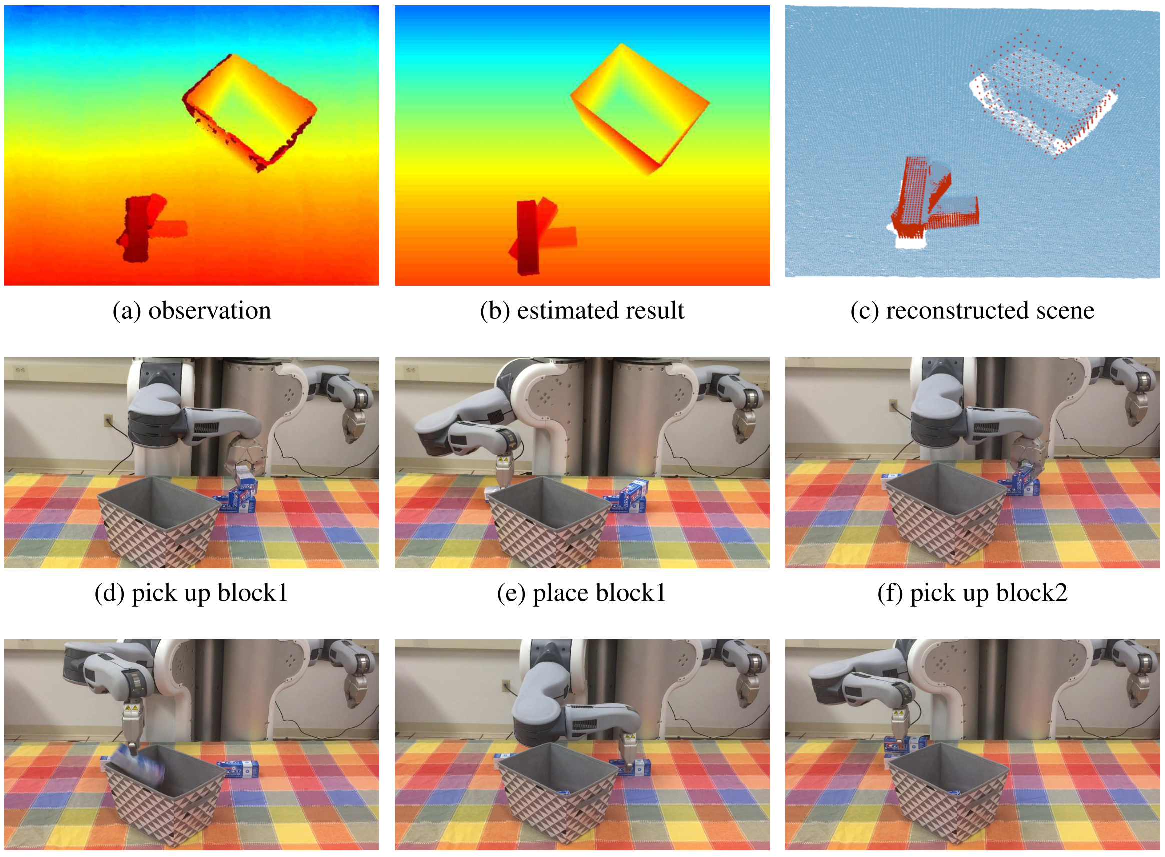

(a) observation (b) estimated result (c) reconstructed scene
(d) pick up block1 (e) place block1 (f) pick up block2
(g) place block2 (h) pick up block3 (i) place block3

Figure 3.8: Manipulation experiment of extraction of the middle block into the basket. Observation and estimated depth image along with reconstructed point cloud (top row). Frames of the robot performing stacking actions to extract the middle block into the basket (bottom rows).

scenes by random walk sampling. A parallelized GPU-optimized version of these inference methods was described and implemented. Our results indicate that *AxScEs* estimators are effective for manipulation-quality perception based on edit distance on scene graphs, estimation of currently manipulatable (clear) objects, and object pose estimation.

CHAPTER 4

Sequential Scene Understanding and Manipulation

AxScEs assumes known object identifications and works in moderately structured environments. To scale up to more unstructured scenarios and be adaptable to unforeseen and adverse conditions, we propose a combined discriminative-generative approach for robust sequential scene estimation and manipulation - Sequential Scene Understanding and Manipulation (*SUM*). *SUM* considers uncertainty due to discriminative object detection and recognition in the generative estimation of the most likely object poses maintained over time to achieve a robust estimation of the scene under heavy occlusions and unstructured environment. Our method utilizes candidates from a discriminative object detector and recognizer to guide the generative process of sampling scene hypothesis, and each scene hypothesis is evaluated against the observations. Also, *SUM* maintains beliefs of scene hypothesis over robot physical actions for better estimation and against noisy detections. We conduct extensive experiments to show that our approach is able to perform robust estimation and manipulation.

4.1 Motivation

In the previous chapter, we addressed the problem of perception for goal-directed manipulation as *axiomatic scene estimation*, sharing similar aims to existing work in scene estimation for manipulation of rigid objects [72, 73, 61, 45, 19]. These methods take a generative multi-hypothesis approach to robustly infer a tree-structured scene graph, as object poses and directed inter-object relations, from cluttered scenes observed as 3D point clouds. The inferred scene graph estimate

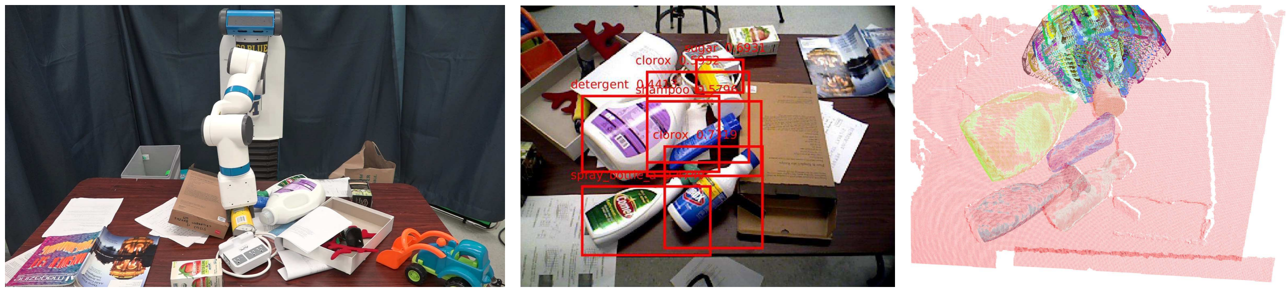


Figure 4.1: (Left) Michigan Progress Fetch robot using *SUM* for scene perception in the sorting of a cluttered set of objects on a table into cleaning (right bin) and non-cleaning categories (left bin). *SUM* performs perception by using (Middle) RGB object recognition to inform (Right) sequential pose estimation from 3D point cloud observations and the feasible grasp poses on the manipulated object.

can be expressed as parameterized axioms that allow for interoperable symbolic task and motion planning towards goals expressed as desired scenes in world space. We posit this pattern of symbolic task-level reasoning using estimates from probabilistic perception will be as applicable to scenes for autonomous manipulation as it has been for autonomous navigation.

Existing formulations of axiomatic scene estimation impose several limiting assumptions that must be relaxed for viable autonomous manipulation. First, existing axiomatic scene estimators assume the identification of objects observed in a scene is given or provided by an idealized object detection and recognition system. Object detection and recognition [43, 32, 29] has greatly improved towards feasible general use, due in part to the renaissance in convolutional neural networks. However, such recognition methods remain subject to substantial and inherent errors in discriminating false positive and negative detections. As such, our robots will need to handle uncertainty due to such recognition errors in both its scene estimation and task execution.

Second, scenes have been assumed to be static, where scene state at each moment in time is effectively decoupled sequentially from its past state. This assumption can be prohibitively costly in terms of computation, as the dimensionality of scene state space grows exponentially with the number of scene objects. We posit that robots can manage this complexity by maintaining a belief of scene state over time informed by past beliefs, a manipulation process model, and current object detections, as well as the incorporation of physical and contextual constraints [25].

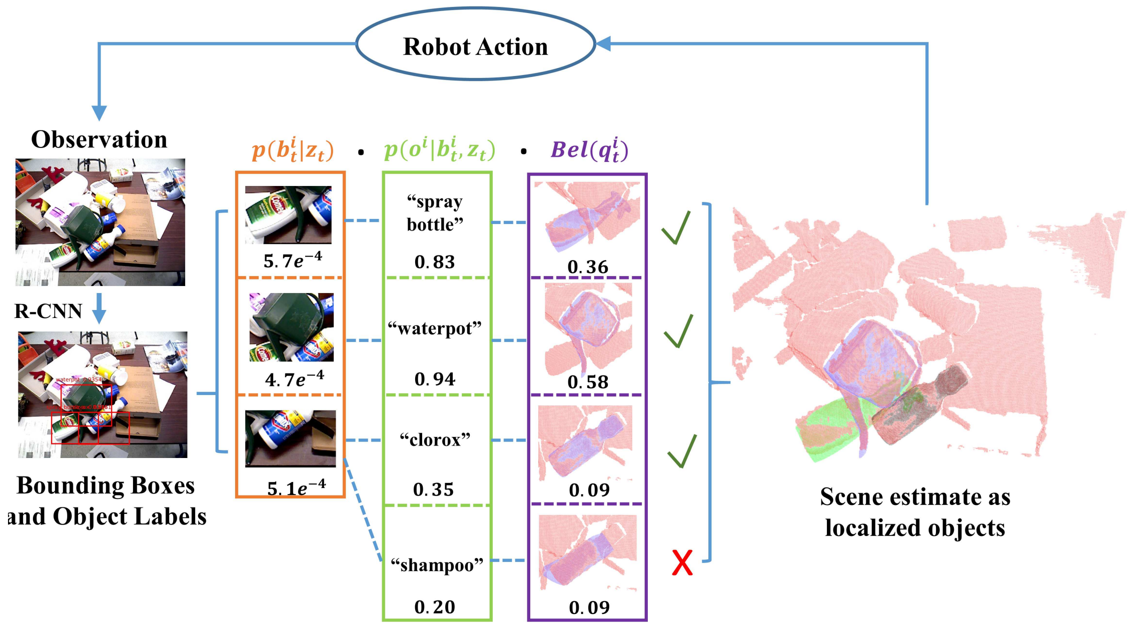


Figure 4.2: An overview of our *SUM* framework. Given an observation of the scene, pre-trained R-CNN object detector and recognizer output bounding boxes and object labels along with their confidence. Assuming that every object is independent with each other, we estimate the state of the scene by estimating the state of each object individually. The numbers in the figure denote the value of each term that composes to the posterior probability of a hypothesized object state. Multiple object pose estimator can originate from the same bounding box, for example, both the "shampoo" and "clorox" pose estimator originates from the same bounding box, and clorox is selected as the correct estimate since it has higher $p(x_t^i)$.

Lastly, existing scene estimation often assumes some scene structure, such as a canonical object orientation, a large flat surface support, or "stacking" as a single support surface per object. By maintaining belief sequentially and managing computational burden, our robots will be able to perform tractable inference in cluttered scenes with full six degree-of-freedom object poses and an arbitrary number of inter-object contacts and supports.

4.2 Formulation

Given past RGBD observations $(z_0, \ldots, z_t)$ and robot manipulation actions $(u_0, \ldots, u_t)$, our aim is to estimate the scene as a collection of k objects, with object labels o, 2D image-space bounding boxes b_t, and six DoF object poses q_t. Note, k is the number of objects *recognized* in the scene.

Object labels are strings containing a semantic identifier assumed to be a human-intuitive reference. In our experiments, the manipulation actions u_t are pick-and-place actions, which will invoke a motion planning process. However, our formulation is general such that u_t also applies to low level motor commands represented by joint torques, such as in scenarios for object tracking. The state of an individual object i in the scene is represented as $x_t^i = \{q_t^i, b_t^i, o^i\}$. We assume that every object is independent of all other objects, which implies there will be only one object with a given label in the eventual inferred scene estimate. Independence between objects allows us to state this scene estimation problem as:

$$p(x_t^1, \cdots, x_t^k | z_{0:t}, u_{1:t}) = \prod_{i=1}^{k} p(x_t^i | z_{0:t}, u_{1:t}) \tag{4.1}$$

where, for each object, the posterior probability is

$$\begin{aligned} p(x_t^i | z_{0:t}, u_{1:t}) =& p(q_t^i, b_t^i, o^i | z_{0:t}, u_{1:t}) \\ =& p(b_t^i | z_{0:t}, u_{1:t}) p(o^i | b_t^i, z_{0:t}, u_{1:t}) \cdot \\ & p(q_t^i | b_t^i, o^i, z_{0:t}, u_{1:t}) \qquad (4.2) \\ =& \underbrace{p(b_t^i | z_t)}_{detection} \underbrace{p(o^i | b_t^i, z_t)}_{recognition} \underbrace{p(q_t^i | b_t^i, o^i, z_{0:t}, u_{1:t})}_{Bel(q_t^i)} \qquad (4.3) \end{aligned}$$

using the statistical chain rule and independence assumptions to yield Equations 4.2 and 4.3, respectively. Equation 4.3 represents the factoring of the scene estimation problem into object detection, object recognition, and belief over object pose. The object detection factor $p(b_t^i | z_t)$ denotes the probability of object i being observed within the bounding box b_t^i given observation z_t. The object recognition factor $p(o^i | b_t^i, z_t)$ denotes the probability of this object having label o^i given the observation z_t inside the bounding box b_t^i. These distributions are generated as the output of a pre-trained discriminative object detector and recognizer that evaluates all possibilities. The implementation of these detectors and recognizers is as explained in Section 4.3. Figure 4.2 describes the framework of *SUM* showing three factors of the scene estimation. The pose belief

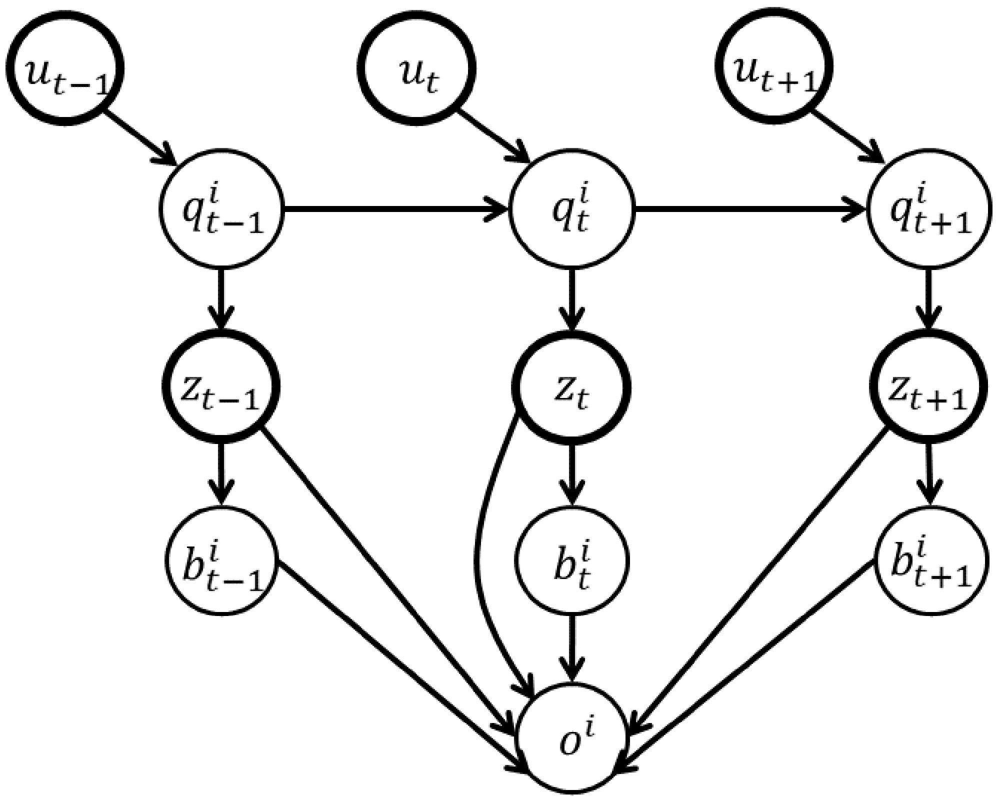


Figure 4.3: Graphical model for estimating pose of a particular object o^i given observations, actions. The bounding box and object label hypothesis at each frame is based on object detection, recognition, and data association as explained in Section 4.3.

factor for a particular object o^i is modeled over time by a recursive Bayesian filter, as illustrated in Figure 4.3. The belief over the object pose q^i_t at time t is estimated as:

$$Bel(q^i_t) \propto \underbrace{p(z_t|q^i_t, b^i_t, o^i)}_{observation\ model} \int_{q^i_{t-1}} \underbrace{p(q^i_t|q^i_{t-1}, u_t, b^i_t, o^i)}_{action\ model} Bel(q^i_{t-1}) dq^i_{t-1} \tag{4.4}$$

Data Association Across all objects, this Bayesian filtering framework also requires a data association process to correspond to previous object estimators with the current detection and recognition. Data association for *SUM* maintains independent filters for each possible object, which are spawned or terminated based on object detection and recognition. At the initial instance of time $t = 0$, the number of objects k is estimated by thresholding on the detection and recognition results for the initial observation z_0.For each recognized object o^i along with its bounding box b^i_0, we will assign an object pose estimator to localize the object within the region defined by b^i_0. When the robot manipulates the objects in the scene for the next action u_1, the objects poses change as a result. To decide within which region that each object pose estimator should continue to localize the object o^i, there is a data association stage where it is associated to a bounding box b_1^i detected at time $t = 1$ after the robot action. Thus after every robot action u_t, the robot receives a new observation z_t, a data association stage takes care of associating each object estimator T^i with a bounding box b_t detected in z_t.

4.3 Methodology

4.3.1 Object Detection Heuristics

The *SUM* model above is agnostic to the specific algorithms for objects detection and recognition as long as distributions of possible object bounding boxes b and labels o can be generated. Specifically, each proposed detection of an object will have a bounding box in the image space with a probability belonging to one of N object labels in the training database. For each generated bounding box at time t, we filter out the object proposal o_l, $1 \leq l \leq N$, if its confidence is smaller than a certain ratio σ_c of the maximum confidence in this bounding box:

$$p(o_l|z_0, b) < \max_{o_l} p(o_l|z_0, b) \cdot \sigma_c \tag{4.5}$$

where $\sigma_c \in (0 \ldots 1)$. The number of recognized objects k is determined by the above thresholding procedure. An object pose estimator T^i is associated to each recognized object i and its corresponding bounding box b_0^i and object label o^i pair.

4.3.2 Particle Filtering for Pose Belief

Particle filtering is employed with each object estimator T^i to infer the pose q_t^i of object i. A particle filter is a means of inference for the sequential Bayesian filter in Eq. 4.4 through an approximation consisting of n weighted particles, $\{q_t^{i_j}, w_t^{(j)}\}_{j=1}^n$. Weight $w_t^{(j)}$ for particle $q_t^{i_j}$ is expressed as:

$$Bel(q_t^i) \propto p(z_t|q_t^i, b_t^i, o^i) \sum_j p(q_t^{i_j}|q_{t-1}^{i_j}, u_t, b_t^i, o^i) Bel(q_{t-1}^i) \tag{4.6}$$

as described by [23]. The initial belief of object pose is uniform. At each time instance, the weight of each hypothesis is computed, normalized to one, and resampled based on importance into an updated set of n particles:

$$q_t^i \sim \sum_j w_{t-1}^{(j)} p(q_t^{i_j}|q_{t-1}^{i_j}, u_t) \tag{4.7}$$

Before each robot action, we apply iterated likelihood weighting [67] to estimate the distribution of the object pose given the bounding box and the object label. This serves as a *bootstrap filter*, where the state transition in the action model is replaced by a zero-mean Gaussian noise.

4.3.2.1 Observation Model

Our observation likelihood function measures how well a particle's rendered point cloud explains the observation point cloud. The observation model of this particle filter uses the z-buffer of a 3D graphics engine to render each particle $q_t^{i_j}$ into a depth image for comparison with the observation. This depth image, represented as $\hat{z}_t^{(j)}$, is back-projected into a point cloud $\hat{r}_t^{(j)}$ in the camera frame to simulate the camera model.The observation likelihood for each particle hypothesis with respect to the point cloud r_t associated with the observation z_t is then expressed as:

$$p(z_t|q_t^{i_j}, b_t^i, o^i) = \frac{\sum_{a,b \in \hat{r}_t^{(j)}} \text{INLIERS}(r_t(a,b), \hat{r}_t^{(j)}(a,b))}{N_{z_t}} \tag{4.8}$$

where a and b are 2D indices in the rendered point cloud $\hat{r}_t^{(j)}$, N_{z_t} is the total number of points in the observation point cloud and

$$\text{INLIERS}(p, p') = \begin{cases} 1, & \text{if } \|p - p'\|_2 < \epsilon \\ 0, \text{otherwise} & \end{cases} \tag{4.9}$$

Thus, if the Euclidean distance between an observed point and a rendered point is within a certain sensor resolution ϵ, total number of inliers will increase by 1.

4.3.2.2 Action Model

A robot manipulation action is represented by $u(j, \phi_{pick}, \phi_{place})$. This pick-and-place action is parametrized by the target object index j, object pick and place pose $\phi_{pick}, \phi_{place}$. For a particular object o^i, we use Gaussian components to model how the object pose q^i_t will change from q^i_{t-1} after a robot action $u_t(j, \phi_{pick}, \phi_{place})$,

$$\begin{aligned} &p(q^i_t | q^i_{t-1}, u_t(j, \phi_{pick}, \phi_{place}), b^i_t, o^i) \\ &\quad \propto \begin{cases} w_1 \mathcal{N}(\phi_{place}, \sigma_1^2) + w_2 \mathcal{N}(q^i_{t-1}, \sigma_2^2), & \text{if } i = j \\ \mathcal{N}(q^i_{t-1}, \sigma_3^2), & \text{if } i \neq j \end{cases} \end{aligned} \tag{4.10}$$

If the action u_t is targeted on object o^i, then either the action succeeds and the object is moved to the place pose ϕ_{place} with uncertainty characterized by σ_1^2, or the action fails and the object stays at its previous pose q_{t-1} with uncertainty characterized by σ_2^2. If the action u_t is not targeted on object o^i, then we assume that the object stays at its previous pose q^i_{t-1} with uncertainty characterized by σ_3^2. In cases where the action fails due to the manipulated object being accidentally mishandled, the new pose q_t far from its previous pose q_{t-1} or its expected pose from manipulation success. This possibility is currently not modeled. Instead, the data association will handle this object through the spawning of a new estimator.

4.3.3 Greedy Data Association

Data association is needed to associate each currently detected object bounding box with at most one object estimator at each moment in time. We use a greedy algorithm for our data association problem, which yields similar results at the lower computational cost compared to the Hungarian algorithm (as reported by [10]). First, a matching score matrix S of every pair (T^i, b^l_t) of the object

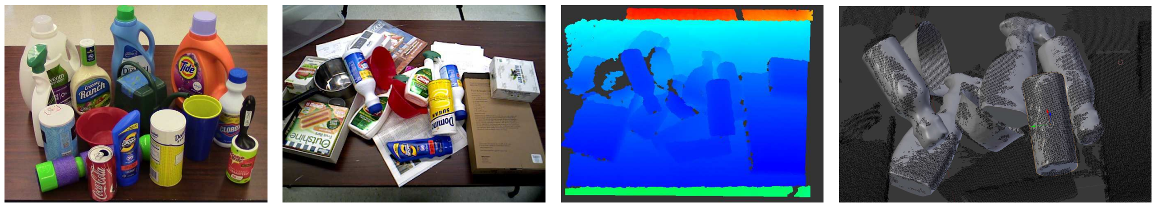


Figure 4.4: The *SUM* dataset objects (a). Eight of these objects in a cluttered scene (b) viewed as an observed depth image (c) and as ground truth (d).

estimator and the bounding box is calculated, with the matching score defined as

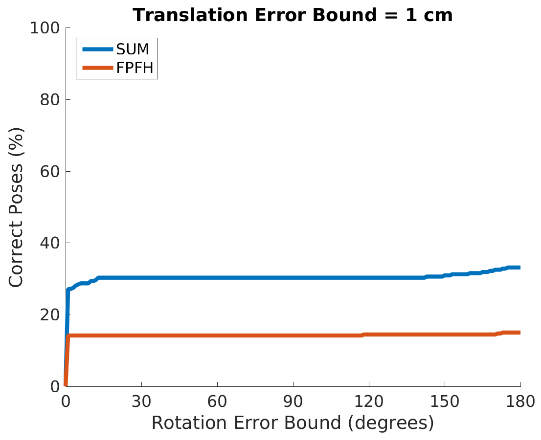
$$s(T^i, b_t^l) = IoU(b_{t-1}^i, b_t^l)p(b_t^l|z_t)p(o^i|b_t^l, z_t) \tag{4.11}$$

which consists of three factors: the overlap between b_t^l and b_{t-1}^i by Intersection over Union (IoU), the likelihood of an object to be in bounding box b_t^l, the likelihood of object o^i inside the bounding box b_t^l. The pair (T^{i*}, b_t^{l*}) with the maximum score in S is selected as an established association. The rows and columns belonging to the object estimator T^{i*} and the bounding box b_t^{l*} are removed from S. This process is repeated until no further pairing is possible. In the end, we only keep the established associations with a matching score above a chosen threshold. A new object estimator is spawned for a bounding box b_t^l not associated with any existing object estimators. An object estimator will be terminated if it is not associated with any bounding boxes for K consecutive frames.

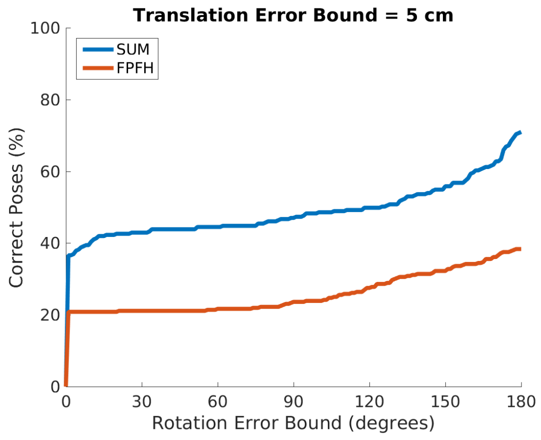
4.4 Results

We first examined *SUM* on single scenes with the estimation on static images without robot actions. We compare *SUM* with a local descriptor, Fast Point Feature Histograms [90], on 10 test scenes of cluttered unstructured environments. For sequential manipulation, eight experiments for sorting objects into two groups were performed with a Fetch mobile manipulation robot.

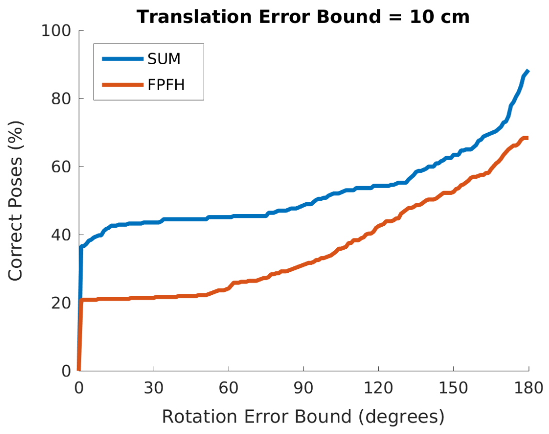
SUM was run on a Ubuntu 14.04 system with a Titan X Graphics card and CUDA 7.5 with 625 particles and 20 resampling iterations for all trials. σ_c is set to 0.1. Sensor resolution ϵ is set



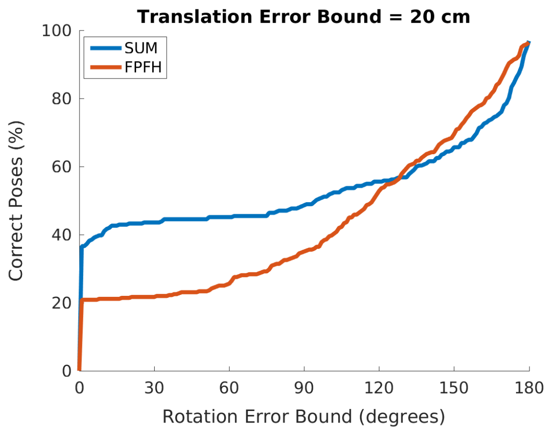
(a)



(b)



(c)



(d)

Figure 4.5: The plots compare the performance between our method *SUM* and FPFH with respect to the accuracy of correct poses. In each plot, there is a fixed translation error bound (1cm, 5cm, 10cm and 20cm), the x-axis is the changing rotation error bound and the y-axis is the percentage of the correct poses. Each point in the plot shows the accuracy of correctly localized objects with a fixed translation and rotation error bound.

to 0.008 in meters. σ_1, σ_2 and σ_3 are set to 0.04, 0.02, 0.01 respectively. A custom dataset of 15 household objects (Figure 4.4) was used for testing, as well as 3D model generation.For CNN training, 8-10 streams of each object in the dataset was captured in a variety of different poses with different backgrounds. The whole training dataset contains 8366 ground truth images and 60563 background images. The Caffenet model [43] was used for network fine-tuning, which was trained on ImageNet [24].

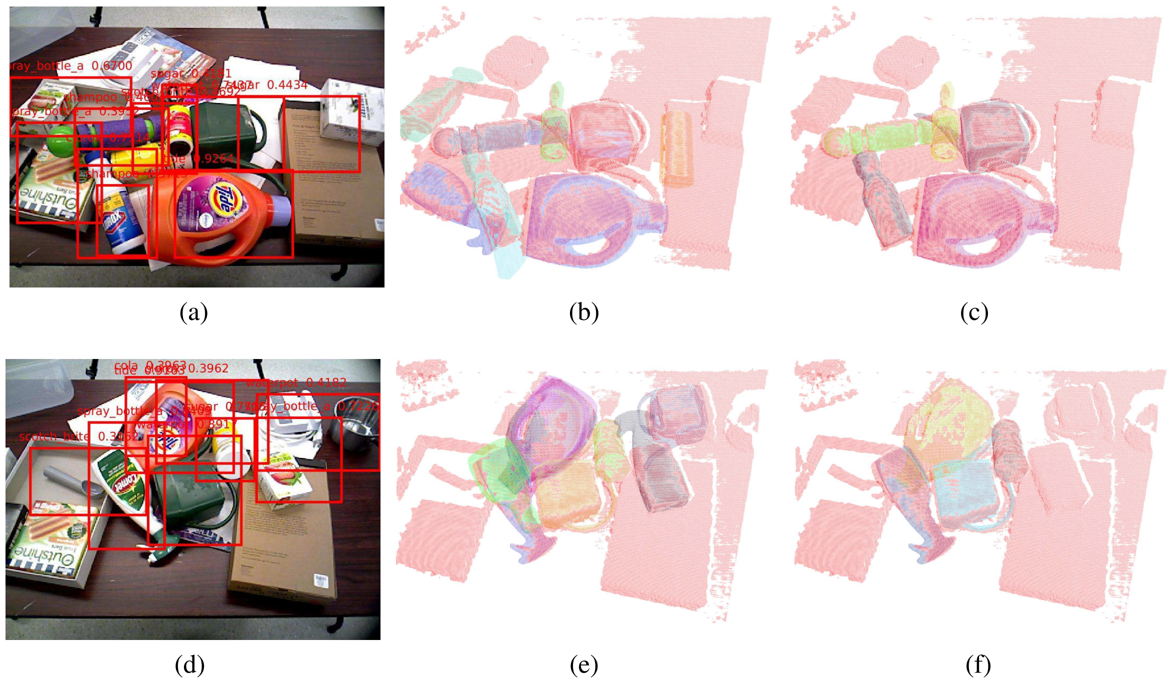


Figure 4.6: (a)(d) The detection results from EgdeBox and R-CNN object detector . (b)(e) Intermediate results from *SUM* which contain false positive estimation (c) The scene estimate result of (a) after thresholding, correct the false positives: two ”spray bottle”, a ”shampoo” and a ”sugar”. (f) The scene estimate result of (d), correct the false positives: ”ranch”, ”waterpot”, ”spray bottle” and ”tide”.

4.4.1 Single Scene Estimation

SUM was evaluated and compared with FPFH on 10 single scenes, with 10 trials each, with respect to the accuracy of estimated object poses.We compute the accuracy of correct poses over all the test scenes and all the runs. Accuracy is defined as the number of correctly localized objects over the total number of detection true positives from the RCNN object detector, where a true positive has IoU greater than 0.5 between estimated and ground truth bounding boxes. We deem an object as correctly localized if its translation error and rotation error fall with chosen error bounds. The translation error is the Euclidean distance between estimated object position (x, y, z) and ground truth pose (x_{gt}, y_{gt}, z_{gt}) and the orientation error is the shortest angle error between estimate object $(roll, pitch, yaw)$ and ground truth $(roll_{gt}, pitch_{gt}, yaw_{gt})$.

The four plots in Fig. 4.5 depict the comparison between *SUM* and the baseline method. In

	Sequence(a)	Sequence(b)	Sequence(c)	Sequence(d)	Sequence(e)	Sequence(f)	Sequence(g)	Sequence(h)
Number of total objects	5	5	5	5	5	5	5	5
Number of Manipulation Errors	1	1	2	0	0	1	1	0
Number of Manipulation Trials	4	6	7	5	5	5	6	5
Completion Ratio	0.80	1.0	1.0	1.0	1.0	0.8	1.0	1.0

Table 4.1: The table shows results of manipulation experiments for 8 sequences. The first row shows the number of objects in each scene. The number of the manipulation errors and trials for the sequence are shown in row two and row three. The last row shows the ratio of how much the recognizable objects on the table are successfully sorted by the robot.

each plot, there is a fixed translation error bound (1cm, 5cm, 10cm and 20cm) and the x-axis is the changing rotation error bound. The y-axis is the percentage of the correct poses. We can see in Fig. 4.5a, 4.5b and 4.5c, our method performs better than FPFH in the small error bounds (translation error smaller than 1cm, 5cm 10cm). *SUM* can also reject false positives from detection results. Fig 4.6 shows two examples of how *SUM* corrected false positives from detection results. We also calculated the mean ratio of rejected detection false positives. The mean ratio of rejected detection false positives over all the test scenes is 0.84 and the standard error over 10 runs is 0.0126.

4.4.2 Estimation and Manipulation on Sequential Scenes

In the manipulation evaluation, the robot must sort objects on a cluttered tabletop into cleaning and non-cleaning categories by picking and placing the object into the right or left bin. In order to make a natural unstructured scene, we avoid manually placing objects in the scene by indiscriminately pouring the objects onto the cluttered table. After scene estimation by *SUM*, the object with the most likely estimate is selected to be grasped and sorted into the appropriate bin. No matter whether the robot succeeds or not, *SUM* updates the pose hypotheses by the action model, associates the object estimators with current detection results and estimates the scene iteratively.

The manipulation results are shown in Table 4.1. Each scene contains five recognizable objects. We evaluate the method by the completion ratio of each sequence shown in the last row of the table. The completion ratio is how much the recognizable objects on the table are successfully sorted by the robot. The robot successfully completed six out of eight sequences. In sequence(a), failure occurred when the robot was trying to pick up ”downy”, it swept ”sugar can” onto the ground. In sequence(f), the robot failed to pick up “waterpot” as the feasible grasp poses are out of the joint limits of the arm. As shown in the second row, a manipulation action error occurs about once on average per trial. Despite such errors, *SUM* performs robustly to not only detection uncertainties but also manipulation failures. As shown in Figure 4.7, there is a manipulation error in the fourth action of the sequence, where the “spray bottle” slipped from the gripper. *SUM* subsequently estimated this object again and the robot picked it up successfully. More experiments are provided in the video[1].

4.4.3 Robotic Manipulation Experiment in Adversarial Scenarios

In this experiment, we compare the robot grasping results using Faster-RCNN [86] and our proposed method for the object detection and pose estimation stages to get an initial understanding of how these two approaches might perform under adversarial attack. The task for the robot is to recognize the Coke can and then pick and place it on the black tray.

Figure 4.8 shows the results from these two methods. We first make a basic scene with normal light condition and minimum occlusion of objects in our dataset (”Coke”, ”clorox”, ”downy” and ”ranch”). We then mimic three adversarial scenes by changing the light conditions, making the Coke can partially occluded and altering the surface of the Coke. In the basic scene, both Faster-RCNN and our method correctly recognize all the objects and the robot successfully picks and places the Coke on the tray. However, in the adversarial scenes, the performance of Faster-RCNN is not satisfactory while the two-stage method remains reliable. In the dark scene, Faster-RCNN only detects one object, but it is a false positive, while our method finds three objects correctly though missing the ”ranch”. When the Coke is partially occluded or wrapped with a cover, Faster-RCNN fails to detect it, while our approach gives not only the correct label and bounding box but also a good pose estimation, so that the robot succeeds in picking and placing.

[1]https://youtu.be/ry0mqY5I-04

4.5 Summary

In this chapter, we propose *SUM* as a combined generative and discriminative approach to robust sequential scene estimation and manipulation. *SUM* utilizes output from a discriminative object detector and recognizer to guide the generative process of sampling scene hypothesis for 6DOF pose estimation. By maintaining a belief over object poses over a sequence robot actions, *SUM* is able to perform robust estimation and manipulation in a cluttered and unstructured tabletop scenario.

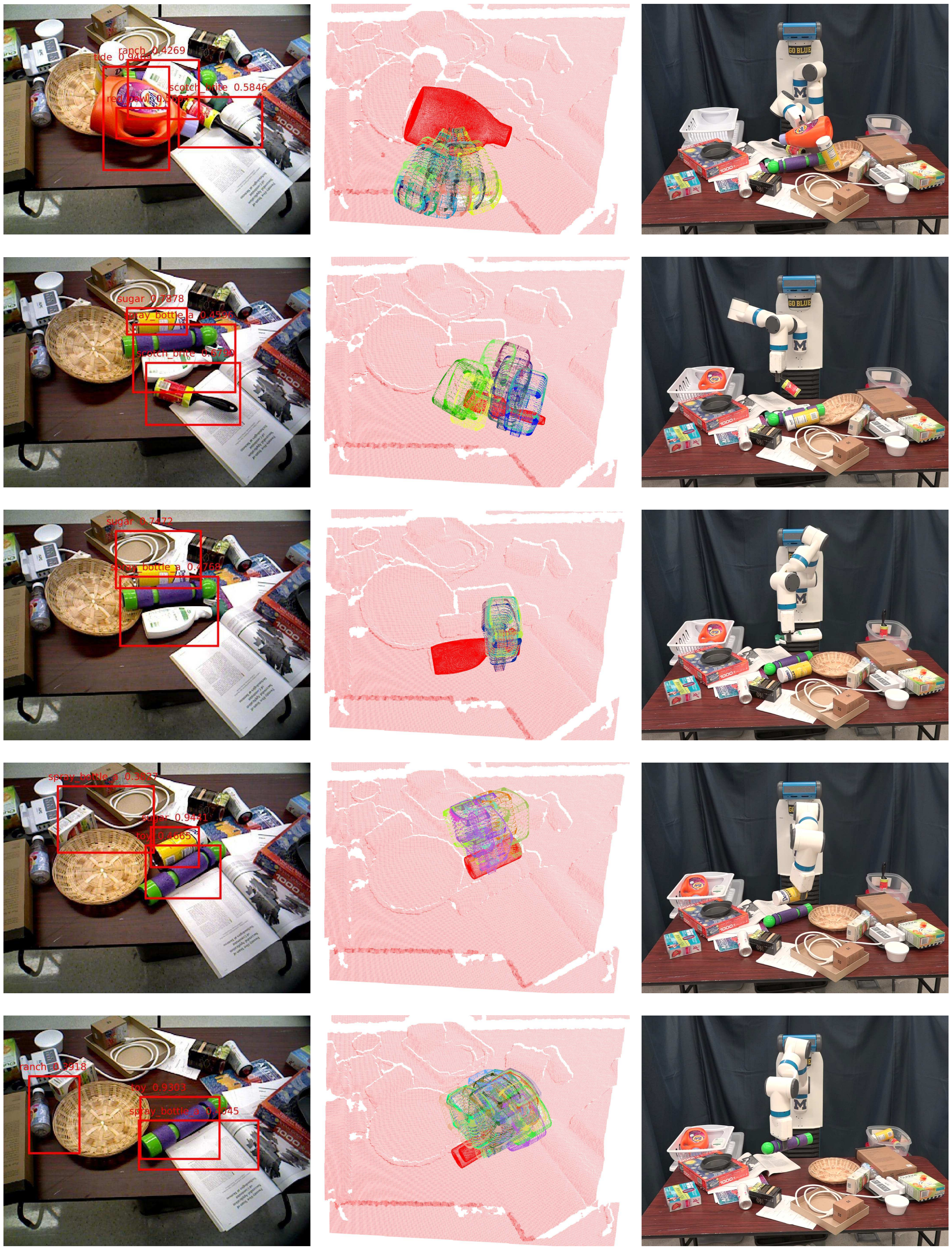
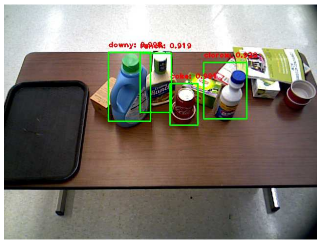
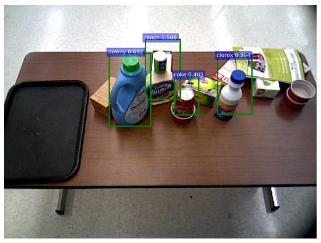


Figure 4.7: Sequential manipulation by a robot to sort 5 objects on a cluttered tabletop into two bins: cleaning (right bin) and non-cleaning (left bin). From the left to right is the detection results from RCNN object detector, most likely object from *SUM*, computed collision-free grasp poses and robot manipulating the object in action. Our system estimated and manipulated "tide", "scotch brite", "spray bottle", "sugar" and "toy" sequentially.

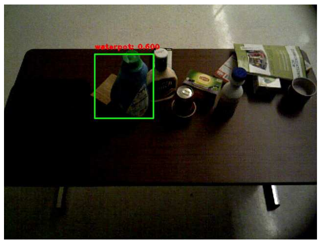
(a) Basic Scene

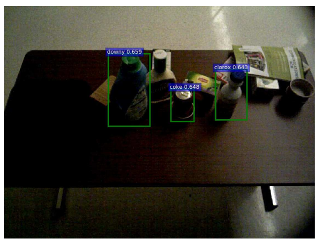

(b) Faster-RCNN Detection


(c) Two-Stage Detection

(d) Grasping

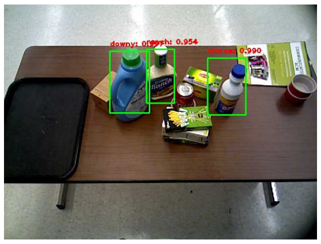
(e) Dark Scene

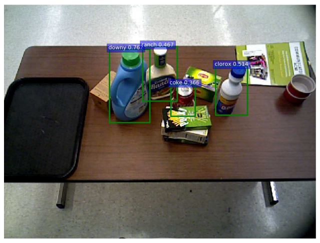

(f) Faster-RCNN Detection


(g) Two-Stage Detection

(h) Grasping

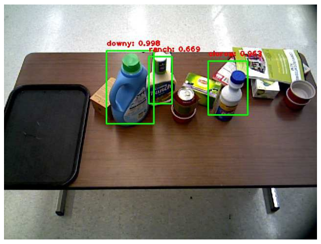
(i) Cluttered Scene

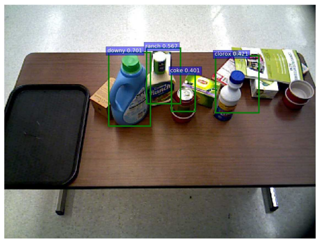

(j) Faster-RCNN Detection


(k) Two-Stage Detection

(l) Grasping

(m) Surface-Change scene


(n) Faster-RCNN Detection


(o) Two-Stage Detection

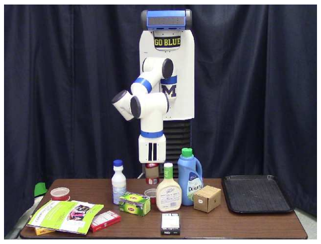

(p) Grasping

Figure 4.8: The robot task is to pick the can of Coke and place it on the black tray under various adversarial scenes. One basic scene (a) and three adversarial scenes (b)(c)(d) are shown above. (b) is a dark scene, (c) is a cluttered scene in which the Coke is partially occluded, and (d) is a scene in which the Coke is wrapped with a brown cover. Figures (b)(f)(j)(n) show the detection results from the Faster-RCNN detector with a threshold of 0.5. The Faster-RCNN detector was not able to detect the Coke in those adversarial scenes. Figures (c)(g)(k)(o) show the detection results from our method. The detector missed the bottle of ranch in the dark scene, but successfully detected all the Coke cans and other objects in our dataset. Figure (d)(h)(i)(p) show the moment when the robot successfully picked up the Coke can, given the detection from our method.

CHAPTER 5

Geometric Consistency informed Scene Estimation in Dense Clutter

SUM ignores physical interactions between objects by assuming objects are independent with each other. However, as the structure of the objects becomes more complex as in densely cluttered environments, taking physical interactions into account benefits reliable scene perception. In this chapter, We propose *GeoFusion*, a SLAM-based scene estimation method for building an object-level semantic map in dense clutter. In dense clutter, objects are often in close contact and severe occlusions, which brings more false detections and noisy pose estimates from existing perception methods. To solve these problems, our key insight is to consider geometric consistency at the object level within a general SLAM framework. The geometric consistency is defined in two parts: geometric consistency score and geometric relation. The geometric consistency score describes the compatibility between the object geometry model and observation point cloud. Meanwhile, it provides a reliable measure to filter out false positives in data association. The geometric relation represents the relationship (e.g., contact) between geometric features (e.g., planes) among objects. The geometric relation makes the graph optimization for poses more robust and accurate. *GeoFusion* can robustly and efficiently infer the object labels, 6D object poses, and spatial relations from continuous noisy semantic measurements. We quantitatively evaluate our method using observations from a Fetch mobile manipulation robot. Our results demonstrate greater robustness against false estimates than frame-by-frame pose estimation from the state-of-the-art convolutional neural network.

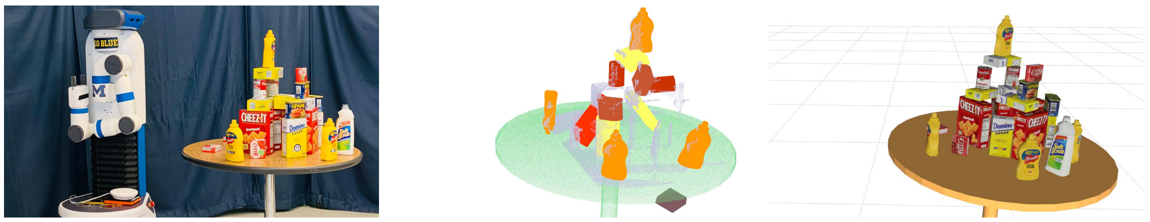


Figure 5.1: Given a robot moving around a table of cluttered objects (left), and noisy semantic measurements from state-of-the-art neural networks (middle), the robot builds an accurate and robust object-level semantic map (right).

5.1 Motivation

To make autonomous robots taskable such that they function properly, meet human expectations, and interact fluently with human partners, they must be able to perceive and understand the semantics of their environments [54]. More specifically, as robots move around in the environment, they must know what objects are presented as well as their locations. It is desired for robots to understand the semantic aspects of the scene and build a map at the object-level. An object-oriented representation of the world is a natural and efficient way for robots to make high-level decisions using task-level planners and help them to communicate with human users.

The challenge is that many semantic aspects of the world are difficult for robots to sense directly due to the limited field-of-view and noisy observations from their onboard sensors. Great progress has been achieved by the advances in deep neural networks for object detection [86, 60, 40] and 6D object pose estimation [116, 109, 111]. However, building a semantic map at the object-level remains a challenging problem, especially in densely cluttered environments. Because in such dense clutter, objects are often in close contact, causing severe or complete occlusions.

One promising approach for building object-level maps is to fuse semantic measurements from different viewpoints. Robots can take advantage of their ability to move around the environment and provide continuous observations. More specifically, the data association between measurements and objects are first determined [115] and probabilistic model are built over objects and robot poses to infer the semantic map (SLAM++ [94], Fusion++ [65]) or belief over objects (CT-MAP [119]). Such methods perform inference directly on the object instance instead of low-level primitives (points, surfels). This approach to inference offers faster inference, more compact map representation, and the potential for dynamic object reasoning. We posit that this approach is particularly advantageous in dense clutter. The geometric properties of object instances and the geometric relationship between objects can be used for more robust data association and accurate pose estimation.

In this work, we present *GeoFusion* , a SLAM-based approach for inferring object labels and 6D poses from continuous noisy measurements in dense clutter by exploring the *geometric consistency* of the object instances. The geometric consistency is defined in two parts: geometric consistency score and geometric relation. The geometric consistency score describes the compatibility between object geometry model and observation point cloud. Meanwhile, it provides a reliable measure to filter out false positives in data association. The geometric relation represents the relationship (e.g., contact) between geometric features (e.g., planes) among objects. Moreover, geometric relation is directly amenable to make better decisions for high-level task planners [66, 58].

Given noisy measurements from state-of-the-art object detection [40] and pose estimation [111] systems, our method first determines the correct correspondences between measurements and objects with geometric consistency between measurements and point cloud observations. The associations are then used to build a factor graph along with the geometric relations between objects. We use the fast optimization technique to get the maximum likelihood estimation of object and robot poses. We quantitatively evaluate our method and demonstrate that *GeoFusion* is able to estimate geometrically stable scene and be robust against false estimates from state-of-art frame-by-frame estimation system and outperforms the baseline methods that do not consider geometric consistency.

5.2 Formulation

Our aim is to estimate the semantic map composed of a collection $\mathcal{O} = \{o_j\}_{j=1}^{M}$ of M static objects as well as geometric relationships $\mathcal{R} = \{r_{ij} | i, j \in [1, M]\}$ between them with the assumption of known 3D object geometries. Note that the number of objects M is unknown. Each object $o_j = (o_j^c, o_j^p)$ contains object class $o_j^c \in [1, C]$ and 6D object pose $o_j^p \in SE(3)$. Each r_{ij} describes the geometric spatial relation between geometric features of two objects (e.g. support), more of which will be discussed at Sec. 5.4.2.

When the robot moves around in the environment, it observes a set of semantic measurements $\mathcal{Z} = \{\{z_t^k\}_{k=1}^{N_t}\}_{t=1}^{T}$, where T is the total time step robot has travelled, and N_t is the number of semantic measurements at each time step. Similar to the definition of object, each semantic measurement $z_t^k = (z_t^{k,c}, z_t^{k,p})$ is comprised of object class and 6D pose. The robot poses represent as $\mathcal{X} = \{x_t\}_{t=1}^{T}$, where $x_t \in SE(3)$ is also 6D pose in our case. In addition, the correspondence between objects $\mathcal{O}$ and measurements $\mathcal{Z}$ also needs to be determined and is defined as $\mathcal{D} = \{\{d_t^k\}_{k=1}^{N_t}\}_{t=1}^{T}$ where $d_t^k = j$ stipulates that measurement z_t^k corresponds to object o_j.

A complete statement of this problem is the maximum likelihood estimation of $\mathcal{X}$, $\mathcal{O}$, and $\mathcal{D}$ given the semantic measurements $\mathcal{Z}$ and the geometric relation $\mathcal{R}$ is computed heuristically from $\mathcal{O}$:

$$\hat{\mathcal{X}}, \hat{\mathcal{O}}, \hat{\mathcal{D}} = \underset{\mathcal{X},\mathcal{O},\mathcal{D}}{\arg\max} \log p(\mathcal{Z} | \mathcal{X}, \mathcal{O}, \mathcal{D}) \tag{5.1}$$

As the joint estimation of objects, data association and robot poses suffers from the high dimensionality, the most common approach is to decompose Eq. 5.1 into two separate estimation problems: data association and graph optimization. The maximum likely estimation data association $\mathcal{D}$ is first computed given initial estimates of robot poses $\mathcal{X}^{(0)}$ and objects $\mathcal{O}^{(0)}$. Then given computed $\hat{\mathcal{D}}$, the most likely robot poses and objects are estimated:

$$\hat{\mathcal{D}} = \arg\max_{\mathcal{D}} p(\mathcal{D}|\mathcal{X}^{(0)}, \mathcal{O}^{(0)}, \mathcal{Z}) \tag{5.2}$$

$$\hat{\mathcal{X}}, \hat{\mathcal{O}} = \arg\max_{\mathcal{X},\mathcal{O}} \log p(\mathcal{Z}|\mathcal{X}, \mathcal{O}, \hat{\mathcal{D}}) \tag{5.3}$$

Again, we decompose Eq. 5.3 to a two-step optimization where the object and robot poses are first optimized and the geometric relation R will be computed heuristically from O and these geometric relations will, in turn, pose constraints on factor graph for estimating geometrically stable scene:

$$\hat{\mathcal{X}}, \hat{\mathcal{O}}' = \arg\max_{\mathcal{X},\mathcal{O}} \log p(\mathcal{Z}|\mathcal{X}, \mathcal{O}, \hat{\mathcal{D}}) \tag{5.4}$$

$$\hat{\mathcal{O}} = \arg\max_{\mathcal{O}} \log p(\mathcal{Z}|\mathcal{O}, \mathcal{R}) \tag{5.5}$$

5.3 Data Association

As the number of objects is not known as prior knowledge, we proposed a non-parametric clustering based approach for data association based on DPmeans algorithm [53], as shown in Algorithm 1. Given initial estimates of objects $\mathcal{O}^{(0)}$, robot poses $\mathcal{X}^{(0)}$ and measurements $\mathcal{Z}$, we compute how likely each measurement belongs to current objects, being a new object or a false positive. The objects are created, updated and deleted dynamically as the data association process moves forward.

5.3.0.1 Association

For each measurement z_t^k in frame t, the likelihood of z_t^k being assigned to object o_j will be calculated as follows:

$$P(d_t^k = j) \propto \mathbb{I}_{z_t^{k,c} = o_j^c} \, f_{gc}(z_t^k|o_j) \, f(z_t^k|o_j, x_t), \tag{5.6}$$

Algorithm 1: Clustering Based Data Association

```
   Input: Measurements $\mathcal{Z}$, Robot poses $\mathcal{X}$, Objects $\mathcal{O}$
   Output: Data Associations $\mathcal{D}$
 1 for Each measurement $z_t^k$ do
 2     for Each object $o_j$ do
 3         Compute likelihood of $z_t^k$ of being object $o_j$:
 4         $p(d_t^k = j) \propto \mathbb{I}_{z_t^{k,c}=o_j^c} f_{gc}(z_t^k|o_j) f(z_t^{k,p}|o_j^p, x_t)$;
 5     end
 6     if $\max p_j \leq \epsilon_{new}$ then
 7         NewObjectInit($z_t^k$);
 8     else
 9         $d_t^k = \arg\max p_j$;
10     end
11 end
12 for Each object $o_j$ do
13     Compute false positive score $f_j$ and remove it if score is above $\epsilon_{fp}$
14 end
```

where $\mathbb{I}_{z_t^{k,c}=c_i^c}$ is the label factor and it is obtained using an indicator function that evaluates to 1 if $z_t^{k,c} = c_i^c$ and 0 otherwise, meaning that each measurement will only be assigned to a object with the same label.

The second term $f_{gc}(z_t^k|o_j)$ captures the geometric consistency score between measurement and the observation point cloud and provides a measure of how reliable the given measurement is. The motivation behind this term is that the confidence score given by neural networks sometimes are not accurate enough to describe the reliability of the measurement. Given a measurement z_t^k, a point cloud is back-projected from the rendered depth image and compared with the observation point cloud to compute projective inlier ratio r_i, outlier ratio r_{out} and occlusion ratio r_{occ}. A point in the rendered point cloud is first considered as an occlusion point if it is occluded by the projective point shooting from the camera ray. If not, it is considered as an inlier if the distance with the projective point is less than certain sensor resolution ϵ_{res}, or outlier if the distance is greater than ϵ_{out}. Fig. 5.2 shows an example of different types of points. The ratios are computed over the total number of rendered points. And they are used as follows:

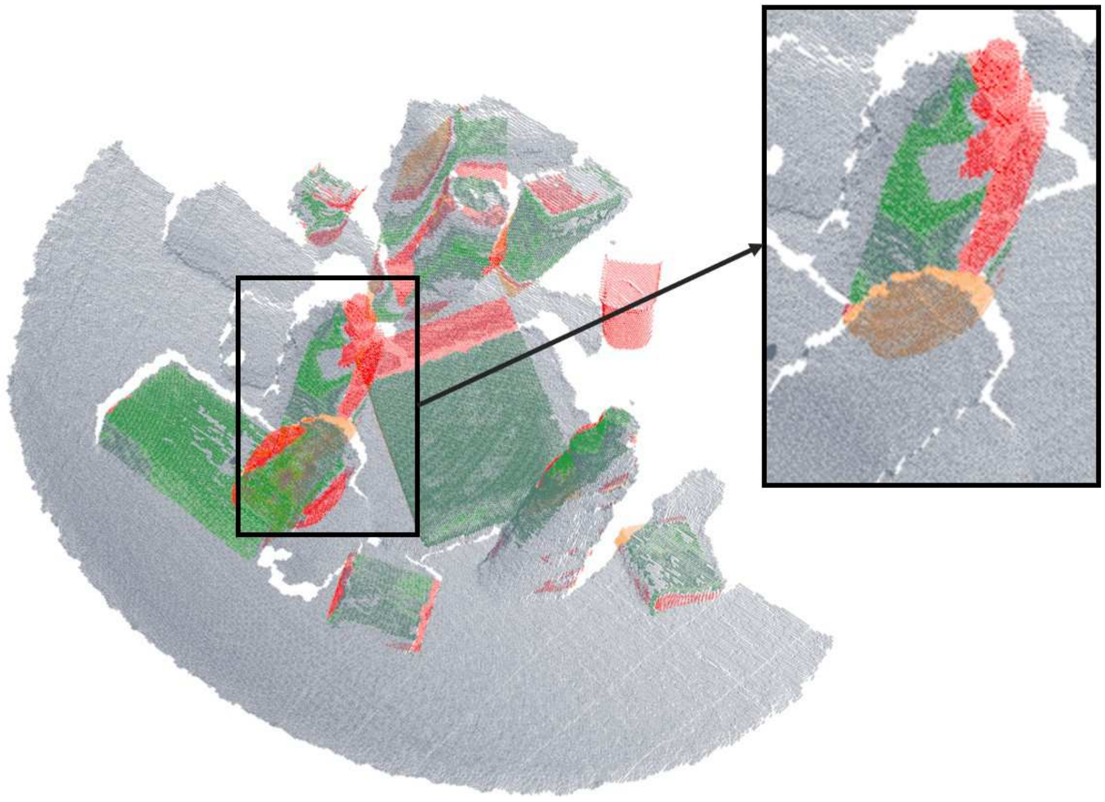


Figure 5.2: Geometric consistency score between measurements and point cloud observation. The zoomed in picture shows the inlier, outlier and occlusion points of a mustard bottle. Point cloud observations are shown in blue. Inlier points are shown in green, outlier in red and occlusion in orange.

$$f_{gc}(z_t^k|o_j) = S(r_i)S(1 - r_{out})S(1 - r_{occ}) \tag{5.7}$$

where S is the modified sigmoid function providing an S-shaped logistic curve that is well-adapted to reflect the changing tendency of confidence on the given measurement over different values of each ratio.

$f(z_t^k|o_j, x_t)$ is the pose factor that reflects the similarity of measurement pose and object pose, given the current robot pose x_t. Here, we make the assumption that the likelihood of each measurement z_t^k given the robot pose and the object pose follows a multivariate Gaussian distribution.

$$f(z_t^k|o_j, x_t) \sim \mathcal{N}(x_t^{-1} \cdot o_j^p, Q) \tag{5.8}$$

where Q is the measurement noise matrix and this factor can thus be drawn from the Gaussian probability density function. With all the factors described above, z_t^k is assigned to be the maximum likelihood object if the object is within a certain threshold ϵ_{new}, otherwise, it is assigned to a new object.

The One Measurement Per Object (OMPO) constraint [115] is also applied in this process so that two measurements in the same frame will never be assigned to the same object and the latter will be assigned to a new object instead.

5.3.0.2 False positive removal

False positive score For each object candidate created or updated in the association step, we compute its false positive score f_j as follows:

$$f_j = 1 - \frac{R_j}{1 + e^{(-n_i)}} \tag{5.9}$$

where n_j is the number of measurements assigned to o_j and R_j is the maximum geometric consistency score among all of these measurements. If there are more measurements assigned to one object, it is reasonable to consider this object as more reliable among others thus it deserves a lower false-positive score. All objects with false-positive scores higher than ϵ_{fp} will then be deleted.

Overlapped objects merging We notice that most of the false positive measurements usually overlap with other objects, especially for symmetric objects, like a cylinder-shape tomato soup can. Object-level geometric consistency is incorporated again to address this problem as we applied the Separating Axis Theorem (SAT) [34] to detect the collision ratio of the three-dimensional bounding boxes of any two objects. Corner points of each bounding box are projected to several main axes and the overall collision ratio is decided based on the percentage of overlap along each axis. One object with a high collision ratio will be merged to the other one that has a lower false-positive score. A Non-Maximum Suppression (NMS) algorithm [75] is also adopted to ensure we find the

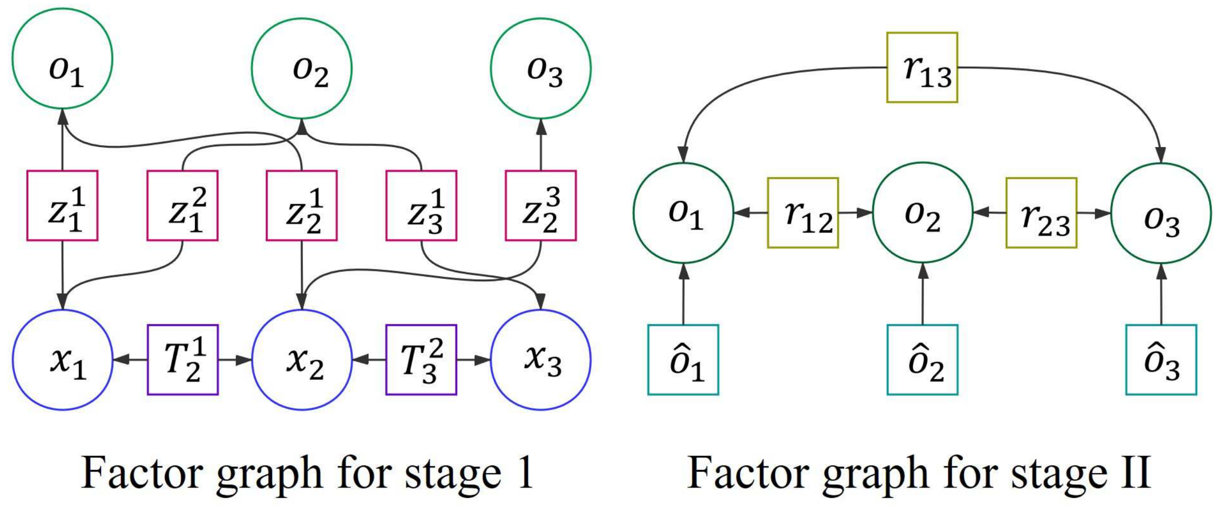


Figure 5.3: Factor graphs for our two stage optimization method.

best candidate when there are multiple objects that overlap with each other.

5.4 Graph Optimization

After the associations $\mathcal{D}$ between objects $\mathcal{O}$ and semantic measurements $\mathcal{Z}$ have been determined, the most likely robot and object poses are then inferred in the graph optimization process. The factor graph is used to model robot and object states and express the conditional independence between them. It gained a huge success in solving SLAM problems [**?**] due existing computational tools that allow efficient optimization [**?**]. In a factor graph, there exists a set of vertices $\mathcal{V}$ that represent optimization random variables and edges represent factors $\mathcal{F}$ among a subset of random variables. A factor f represents probabilistic dependencies among the random variables and is associated with a cost function. Graphically, the vertex is a circle and the factor is a square in the factor graph, as shown in Fig. 5.3. The joint probability of the optimization random variables can be expressed as the product of factors:

$$p(\mathcal{V}) \propto \prod_{f \in \mathcal{F}} f(\mathcal{V}) \tag{5.10}$$

The optimization process in *GeoFusion* is decomposed into two stages, as shown in Fig. 5.3. First, 6D poses of the robot and the objects are optimized over odometry and semantic measurements. In the second stage, 6D poses of objects are then fine-tuned over the computed geometric relations to produce a geometric consistent scene estimate.

5.4.1 Stage I Optimization

The right hand side of Eq. 5.4 is rewritten as:

$$\log p(\mathcal{Z}|\mathcal{X},\mathcal{O},\hat{\mathcal{D}}) = \sum_{t=1}^{T} \phi(T_t^{t-1}; x_{t-1}, x_t) + \sum_{t=1}^{T}\sum_{k=1}^{N^t} \phi(z_t^{k,p}; x_t, o_{d_t^k}^p)$$

where $\phi(T_t^{t-1}; x_{t-1}, x_t)$ is the odometry factor and $\phi(z_t^{k,p}; x_t, o_{d_t^k}^p)$ is the object measurement factor. T_t^{t-1} is the odometry measurement between robot pose x_{t-1} and x_t. $z_t^{k,p}$ is the semantic measurement of an object between the robot pose x_t and the object pose $o_{d_t^k}^p$. With the standard assumption of additive Gaussian noise, each factor follows a quadratic form:

$$\begin{aligned}\phi(T_t^{t-1}; x_{t-1}, x_t) &= -\frac{1}{2}(x_t \ominus x_{t-1} - T_t^{t-1})Q^{-1}(x_t \ominus x_{t-1} - T_t^{t-1}) \\ \phi(z_t^{k,p}; x_t, o_{d_t^k}^p) &= -\frac{1}{2}(x_t \ominus o_{d_t^k}^p - z_t^{k,p})R^{-1}(x_t \ominus o_{d_t^k}^p - z_t^{k,p})\end{aligned}$$

where $\ominus$ represents the operator that computes the relative transformation. Q and R is the corresponding covariance noise matrix. Thus, The maximum likelihood estimation of X and O can be written as the nonlinear least-squares problem:

$$\begin{aligned}\hat{\mathcal{X}}, \hat{\mathcal{O}}' = \underset{\mathcal{X},\mathcal{O}}{\arg\min} &\sum_{t=1}^{T} \|x_{t-1} \ominus x_t - T_t^{t-1}\|_{\Sigma_Q} \\ &+ \sum_{t=1}^{T}\sum_{k-1}^{N_t} \|x_t \ominus o_{d_t^k}^p - z_t^{k,p}\|_{\Sigma_R}{}^1 \end{aligned} \tag{5.11}$$

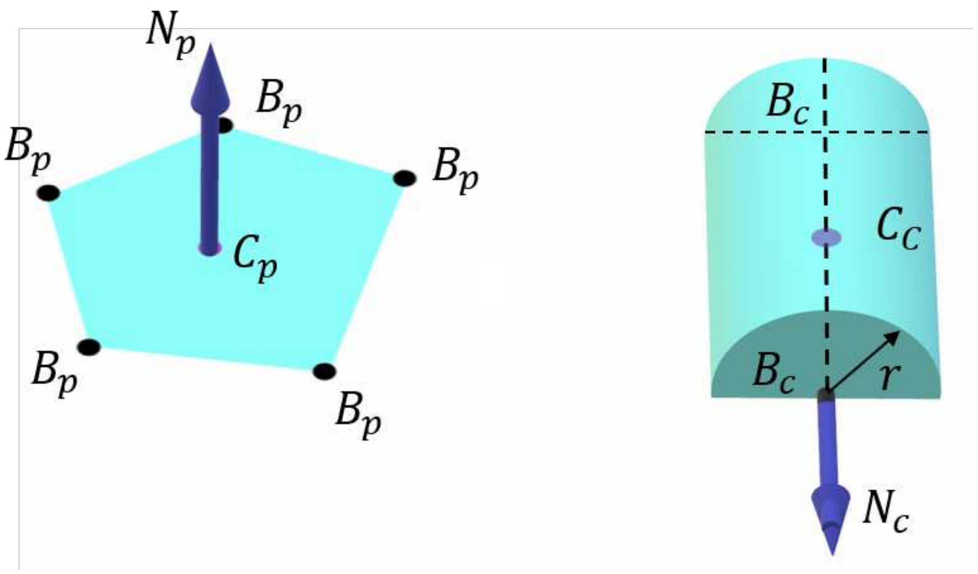


Figure 5.4: Two types of the surface features: the left one is the plane surface feature, and the right one is the curved surface feature.

where Σ_Q and Σ_R is the information matrix. Note that the as the metrics for translations and rotations are different, the information matrix for them is also different. We use Σ_p for translations and Σ_q for rotations which represent as quaternions. Σ_p and Σ_q are diagonal matrices and represent as $Diag(\omega_p, \omega_p, \omega_p)$ and $Diag(\omega_q, 0, 0, 0)$. For symmetric objects in the object measurement factor, we design a separate $Sigma_q$ for the rotation. For example, object with rotation axis in z-axis is associated with $\Sigma_q = Diag(0, \omega_q, \omega_q, 0)$ to exclude the influence from z-axis.

5.4.2 Object Geometric Relationship Inference

In stage I optimization, the maximum likelihood estimation of robot and object poses is obtained by minimizing the odometry and object measurement cost. However, the resulting object poses are still noisy and not accurate enough for applications like precise robot manipulation. The noisy object poses lead to geometrically inconsistent scene estimate, e.g., floating or intersection between objects. To get a highly accurate and geometrically consistent scene estimate, we posit that if the geometric relationship between objects can be inferred and these relations can serve as the

[1] $\|X\|_\Omega = \sqrt{X^T \Omega X}$

constraints (factors) between objects in the factor graph model, as shown in the right of Fig. 5.3. Given the estimated object poses $\mathcal{O}$ from Eq. 5.11, $\mathcal{R}$ is heuristically computed.

5.4.2.1 Geometric Surface Feature

For common rigid household objects, we only consider *contact* relations between surfaces for generality in representing the geometry of the object and efficiency in the graph optimization process. In this paper, geometric surface features are divided into two classes: plane surface feature and curved surface feature. An illustration of surface features is shown in Fig 5.4.

Given an arbitrary convex polyhedron, geometric surface features can be extracted. For example, there are six plane surfaces and twelve curved surfaces for a cube-like geometry, as it has six planes and twelve edges. The edge is treated as a zero radius curve surface. For cylinder-like geometry, there are two plane surfaces and one curved surface. Although the two classes of surfaces cannot represent all of the objects (e.g., articulated or non-rigid objects), they are general and sufficient for common rigid household objects.

Plane Surface Feature Plane surface feature refers to those planes on polyhedrons. There are three attributes associated with plane surface feature: a center point $C_p \in R^3$, a set of points $B_p = \{b_p^i\}_{i=1}^{N_b}$ with $b_p^i \in R^3$, and the normal direction $N_p \in R^3$. C_p describes the 3D position of the plane center. Each b_p^i is the 3D position of a boundary point on the surface and N_b is the number of boundary points.

Curved Surface Feature The curved surface feature describes those curved surfaces in 3D shapes like cylinders. Noticeably, edges of polyhedrons are also regarded as a curved surface feature with a radius equal to zero. Five attributes are associated with the curved surface feature: $C_c \in R^3$ and $N_c \in R^3$, determining the 3D position of center and direction of the rotation axis of the curved surface. $B_c = \{b_c^i\}_{i=1}^{2}$ with $b_c^i \in R^3$ refers to the 3D position of boundary points in the rotation axis. Radius r describes the radius of the curved surface.

Accordingly, we define three types of spatial relations between these two surfaces: Plane2Plane

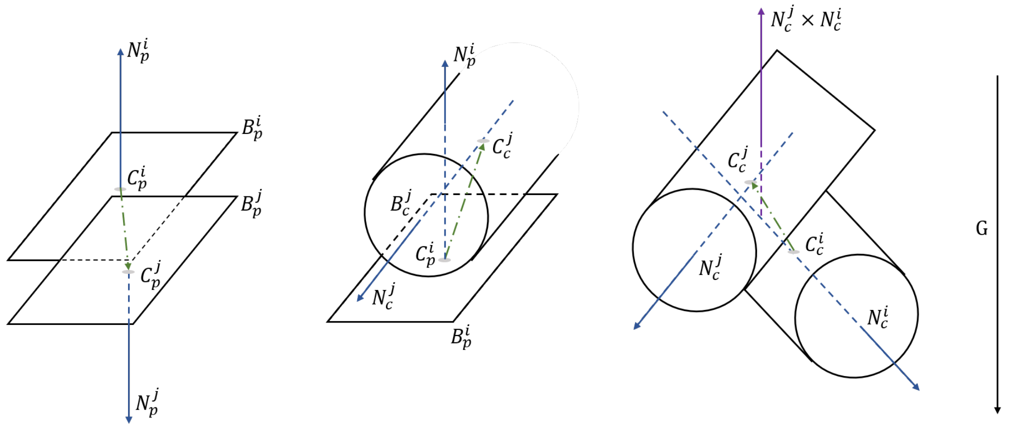


Figure 5.5: Illustrations for geometric relationships. Plane to plane (P2P) contact, plane to curved surface (P2C) contact, and curved to curved surface (C2C) contact are shown from the left to the right.

Contact (P2P), Plane2Curve Contact (P2C), and Curve2Curve Contact (C2C). Fig 5.5 illustrates these geometric relations.

5.4.2.2 Geometric Feature Relationship Inference

P2P contact If two different plane surface features $P_u,\ P_v$ from two objects $o_i,\ o_j$ are contacting each other, their attributes should follow the following rules:

$$\begin{aligned} |T_{o_i} N^i_{P_u} \cdot T_{o_j} N^j_{P_v} + 1| &< \epsilon^{pp}_n \\ |T_{o_i} N^i_{P_u} \cdot (T_{o_j} C^j_{P_v} - T_{o_i} C^i_{P_u})| &< \epsilon^{pp}_c \end{aligned} \tag{5.12}$$

where $\epsilon^{pp}_n, \epsilon^{pp}_c$ are the direction threshold and distance threshold for P2P contact.

P2C contact If a plane surface feature P_u and a curved surface feature C_v are contacting each other, their attributes should follow the following rules:

$$\begin{aligned} |T_{o_i} N^i_{P_u} \cdot T_{o_j} N^j_{C_v}| &< \epsilon^{pc}_n \\ |T_{o_i} N^i_{P_u} \cdot (T_{o_j} C^j_{C_v} - T_{o_i} C^i_{P_u}) - r_j| &< \epsilon^{pc}_c \end{aligned} \tag{5.13}$$

where $\epsilon^{pc}_n, \epsilon^{pc}_c$ are the direction threshold and distance threshold for P2C contact.

C2C contact If two different curved surface features $C_u,\ C_v$ are contacting each other, their attributes should follow the following rules:

$$|(T_{o_i}N^i_{C_u} \times T_{o_j}N^j_{C_v}) \cdot (T_{o_j}C^j_{C_v} - T_{o_i}C^i_{C_u}) - (r_i + r_j)| < \epsilon^{cc}_c \tag{5.14}$$

where ϵ^{cc}_c is the distance threshold for C2C contact.

5.4.2.3 Physical Relationship Check

We also add two physical based checks in complement to the above geometric methods to ensure eliminating false detected contact. For example, if two cubes are placed parallel and close to each other on the table, a P2P contact can be detected. However, there is a chance that those objects are just near to each other but not contacting each other.

Support direction check If a plane feature can support another plane or be supported, its normal direction can not be horizontal to the gravity direction. Assume the direction of gravity is N_G, the support direction check of plane feature i is defined as:

$$N_G \cdot N^i_p > \epsilon_G \tag{5.15}$$

where ϵ_G is the threshold for support direction check. Here we make an implicit assumption that there is at least one grounded object, e.g., table, which supports all the other objects. Similar assumption can be found in SLAM++ [94].

Qualitative Support projection check If there is a support relationship between two features, their 2D projections along the direction of gravity must be overlapped. For those contact feature candidates, their 2D projections are the polygon or the edge projected from their boundary points. The Separating Axis Theorem is applied again to check the overlapping of those 2D contours. Those feature candidates whose 2D projections are not overlapped will be removed.

5.4.3 Stage II Optimization

After spatial relations $\mathcal{R}$ between objects have been computed from the estimated object poses in stage I optimization, these relations are used as constraints in the factor graph optimization. These constraints will enforce proper contact between objects and fix the issues of floating and interpenetration. We can rewrite the right hand side of Eq. 5.5 as:

$$\log p(\mathcal{Z}|\mathcal{O}, \mathcal{R}) = \sum_{i=1}^{M} \sum_{j=1}^{M} \phi(r_{ij}; o_i^p, o_j^p) \sum_{i=1}^{M} \phi(o_i^{'}; o_i) \tag{5.16}$$

where $\phi(r_{ij}; o_i^p, o_j^p)$ is the geometric relation factor and $\phi(o_i^{'}; o_i)$ is the object prior factor. The object prior factor comes from the estimated object poses in stage I optimization. The geometric relation factor encodes the contact constraint between two objects computed from the previous section. Accordingly, there are three constraints: P2P, P2C, and C2C. For P2P constraint, the cost function is defined as:

$$E_{P2P} = \omega_q |T_{o_i} N_{P_u}^i \cdot T_{o_j} N_{P_v}^j + 1| + \omega_p |T_{o_i} N_{P_u}^i \cdot (T_{o_j} C_{P_v}^j - T_{o_i} C_{P_u}^i)|$$

Similarly, the cost function of P2C constraint is defined as:

$$E_{P2C} = \omega_q |T_{o_i} N_u^{o_i} \cdot T_{o_j} N_v^{o_j}| + \omega_p |T_{o_i} N_u^{o_i} \cdot (T_{o_i} C_u^{o_i} - T_{o_j} C_v^{o_j})|$$

and the cost function of C2C is defined as:

$$E_{p2c} = \omega_p |(T_{o_i} N_u^{o_i} \times T_{o_j} N_v^{o_j}) \cdot (T_{o_i} C_u^{o_i} - T_{o_j} C_v^{o_j}) - (r_v + r_u)|$$

	FbF	B-SLAM	R-Front	*GeoFusion*
mAP$_{50}$	58.4	56.0	72.3	**73.4**
mAP$_{75}$	28.6	24.6	50.3	**59.5**
mAP$_{50:95}$	30.5	27.9	45.2	**50.3**

Table 5.1: Object detection performance of different methods with mAP under different IoU.

5.5 Experiments

5.5.1 Implementation

We use Mask R-CNN [40] and DenseFusion [111] to generate semantic measurements where the former detects objects along with the instance segmentation which the latter takes in to estimate the 6D pose. We finetune the Mask R-CNN with the YCB-Video Dataset [116] with the pretrained Microsoft coco model [59]. We use the public available DenseFusion weights and implementation without fine-tuning. The front-end in our implementation selects every 10th camera frame as a keyframe. The camera visual odometry is provided by ORB-SLAM2 [**?**]. We use Ceres [2] as the optimization backend. In our implementation, we perform stage I optimization every 10th keyframe and perform stage II optimization every keyframe.

5.5.2 Dataset and Baseline

To test the performance of our method, we collect a testing dataset from six RGBD video streams in which our Michigan Progress Robot moves around the table. In each scene, we put around 18 objects in dense clutter on the table and collect around 200 keyframe RGB-D observations from the robot's sensor. The groundtruth object classes and poses are labelled using LabelFusion [63]. For baseline methods, we compare our method with frame-by-frame (FbF) estimation from Mask R-CNN and DenseFusion. It only considers the single frame and does not take any temporal or spatial constraints into account. We also compare with the variations of our proposed *GeoFusion* : 1) a baseline SLAM (B-SLAM) method without considering geometric consistency in both data

	FbF		B-SLAM		R-Front		*GeoFusion*	
	Pr	Rec	Pr	Rec	Pr	Rec	Pr	Rec
mAP_{50}	0.87	**0.60**	0.87	0.27	0.91	0.43	**0.99**	0.51
mAP_{75}	0.56	0.38	0.63	0.19	0.75	0.35	**0.91**	**0.46**
$mAP_{50:95}$	0.52	0.35	0.57	0.17	0.65	0.30	**0.77**	**0.40**

Table 5.2: Object detection performance on precision (Pr) and recall (Rec) for different methods under different mAP. The detection confidence threshold is set to 0.5.

association and graph optimization. The confidence score from DenseFusion instead of the geometric consistent score is used in computing data association. The back-end graph optimization only optimizes over odometry and object poses without considering spatial constraints between objects. 2) a robust front-end (R-Front) method where only our data association is used and not the graph optimization.

5.5.3 Evaluation

5.5.3.1 Object Detection

We first compare object detection performance with baseline methods on each keyframe collected in the dataset.

mAP We use the common object detection metric: mean average precision (mAP), to evaluate object detection. Average precision (AP) is the area under the Precision-Recall curve of an object class and mean average precision is the mean of all the AP. The subscript under mAP is the ratio of intersection over union (IoU) between the estimated bounding box and the ground truth bounding box. We also use mAP@[50:95] which corresponds to the average AP for IOU from 0.5 to 0.95 with a step size of 0.05.

Results are shown in Table 5.1. *GeoFusion* significantly outperforms the frame-by-frame method in all mAP metrics indicating that our method is capable of identifying false detections and keeping true ones from noisy semantic measurements. Robust front-end method achieves the

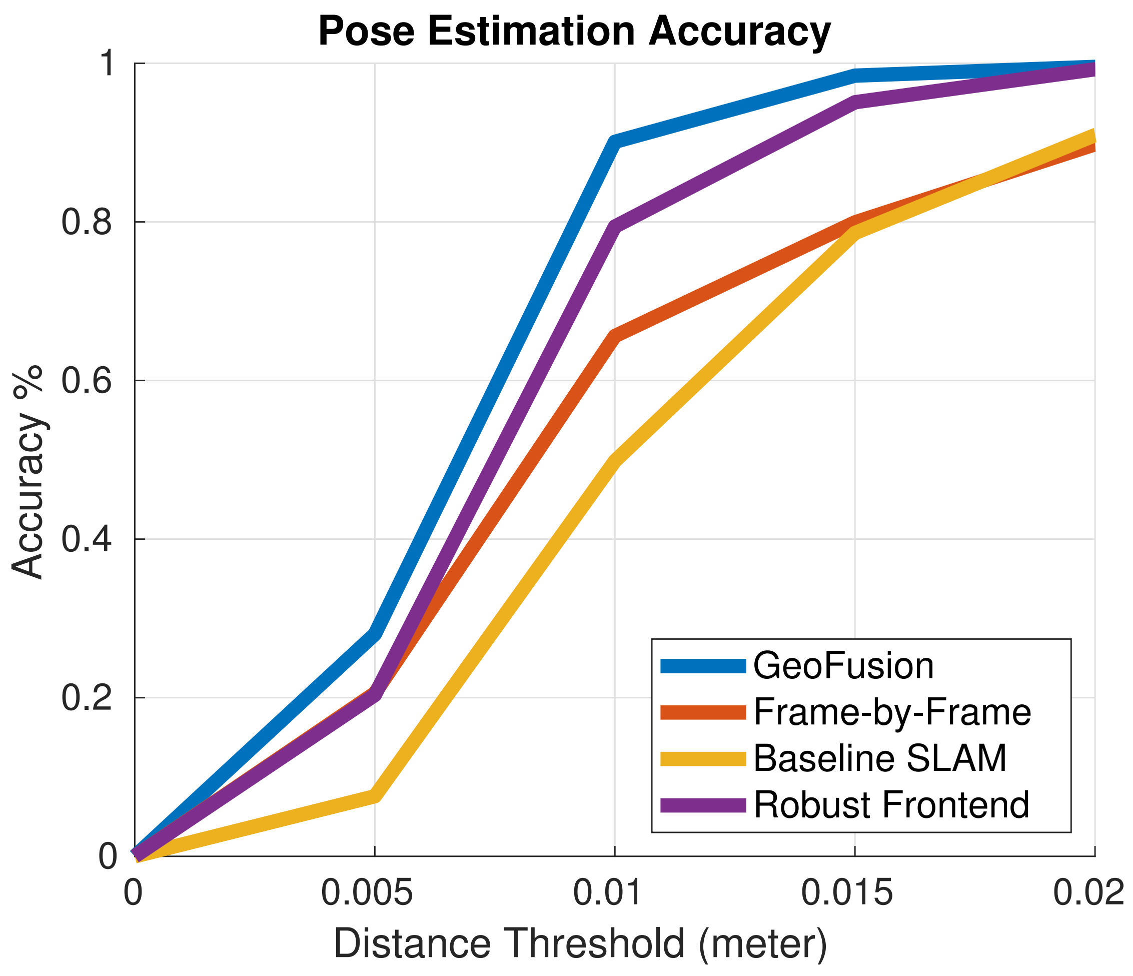


Figure 5.6: Pose accuracy comparison between different methods.

similar mAP$_{50}$ with *GeoFusion* however underperforms in mAP$_{75}$ and mAP$_{50:95}$. It shows that the back-end optimization improves the 6D poses of objects, which in turn increases the mAP with a tighter threshold in IOU. The baseline SLAM method does not take any geometric consistency into account and the performance is even worse than the frame-by-frame method. This is because the false and noisy measurements accumulate in the baseline slam and without the help of geometric consistency, they are difficult to be eliminated from the estimates.

Precision and Recall For a more intuitive evaluation on the object detection, we also report numbers on precision (Pr) and recall (Rec) with a confidence score threshold set to 0.5. As shown in Table 5.2, *GeoFusion* outperforms all the baseline methods under different mAP metrics. With the ratio of IOU increases for checking true positives, the performance gap between *GeoFusion*

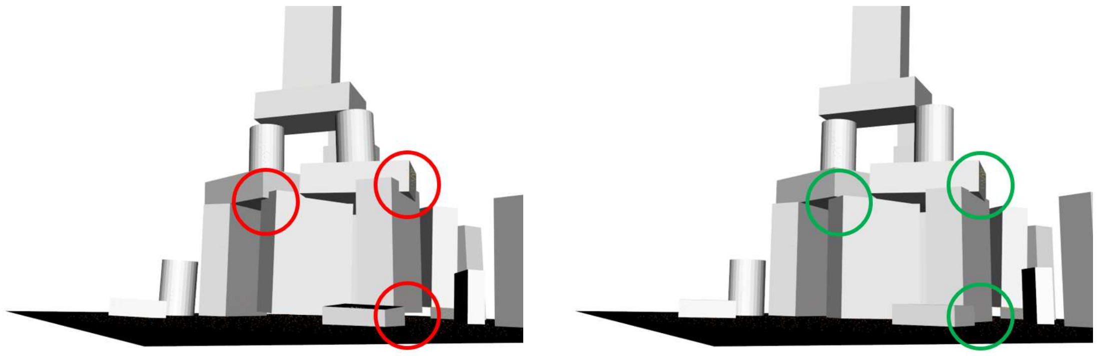


Figure 5.7: The comparison of the two-stage optimization of 230th keyframe of the scene in Fig. 5.1.

and baseline methods becomes larger. Interestingly, the recall of the Frame-by-frame method under $mAP_{50}$is higher than *GeoFusion*. This is because *GeoFusion* has a strict way to filter out false and noisy detections. In other words, detections are chosen very conservatively by *GeoFusion* which leads to more false negatives in a looser metric.

5.5.3.2 Pose Accuracy

For 6D object pose accuracy, we use the ADD-S metric [116] to compute the average point distance between the estimated pose and the correct pose. Only true positives are computed at each frame for each method. As our aim is to build a high accuracy object-level map and the precision required by the robot to grasp objects are relatively high, we plot the accuracy-threshold curves within the range of [0.00m, 0.02m] as shown in Fig. 5.6. *GeoFusion* achieves the highest pose accuracy among all the methods and about 90% percent of the objects have achieved pose error under 1cm. The pose accuracy of the baseline SLAM method is even worse than the Frame-by-Frame method indicating that the confidence score from perception module is sometimes not reliable.

5.5.3.3 Run Time

We show an analysis of run time in Fig. 5.8 on each frame for data association and optimization of an example scene in Fig. 5.1. Note that our method only runs on CPU (without measurements generation), the average time for each frame is under 200 ms

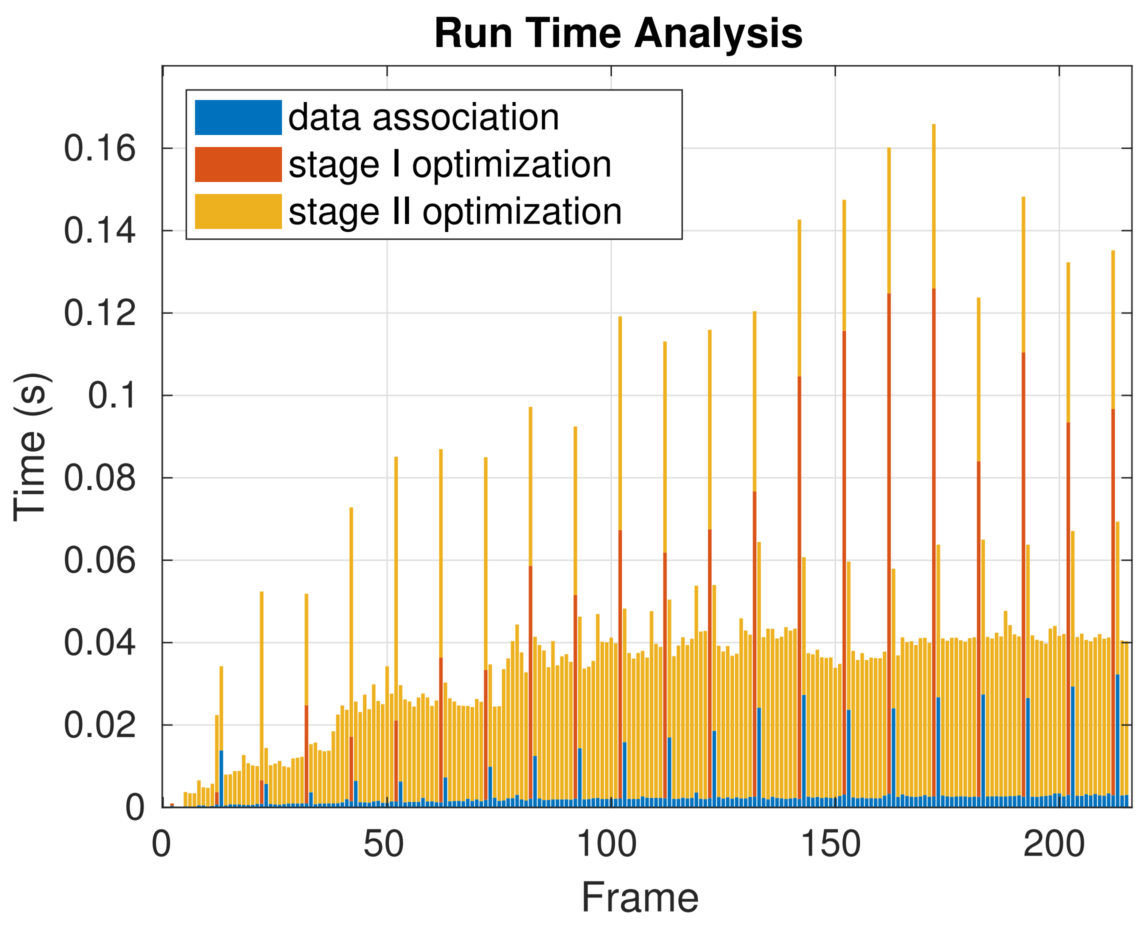


Figure 5.8: Time plot for each frame of the scene in the Fig. 5.1

5.5.3.4 Geometric Relation

Fig. 5.7 compares the results of two optimization stage of the scene in Fig. 5.1. The left figure is the result of the optimization stage I without geometric relation constraints and the right figure is the result after optimized over geometric constraints. The geometric constraints pulled objects to satisfy the constraints. For example, the objects on the right are more aligned with the table and the intersection between objects is fixed.

5.6 Summary

In this work, we present *GeoFusion* , a SLAM-based scene understanding method for building a semantic map at object-level in dense clutter while taking into account geometric consistency. Our method is able to infer object labels and 6D poses from noisy semantic measurements robustly and efficiently. The reasoning at object-level with geometry offers a fast and reliable way to filter out false positives and constraint the object through geometric relation. The computed geometric relations are also directly amenable to high-level task planners for robots to reason over actions for goal-directed manipulation tasks.

CHAPTER 6

Discussion and Conclusion

6.1 Conclusion

This dissertation presents several methods that perform *reliable* scene perception under uncertainty for robotic goal-directed manipulation. The proposed methods expand the reliability of the scene perception towards *robustness*, *adaptability*, and *scale*: 1) robust perception under uncertainty results from low-level hardwares, imperfect algorithms and unstructured environments, 2) adaptable perception to unforeseen and adverse conditions, 3) scalable perception when the number of objects grows, and the structure of objects becomes more complex.

In Chapter 3, we present a pure generative approach, *AxScEs* , to estimate the approximate distribution of a tree-based scene graph to capture the uncertainty from sensor measurements. *AxScEs* generates hypotheses to maintain a diverse set of perspectives over possible scene graph states. The proposed generative approach demonstrates its robustness to the local minimum in the axiomatic state space and object pose space. Our results indicate that *AxScEs* estimators are effective for manipulation-quality perception based on edit distance on scene graphs, estimation of currently manipulatable objects, and object pose estimation.

In Chapter 4, we present *SUM* as a combined discriminative and generative approach to robust sequential scene estimation and manipulation. *SUM* combines the efficiency of the discriminative methods and the robustness of the generative methods. The method achieves robustness and adaptiveness to unstructured and adversarial environments.

In Chapter 5, we present *GeoFusion* that explores physical interactions between objects to build a geometric consistent scene estimate from objects in dense clutter. The interactions between objects are characterized as geometric consistency. *GeoFusion* takes the geometric consistency into account within a SLAM framework. The geometric consistency plays a vital role in both data association and graph optimization modules in the SLAM framework. Our results demonstrate high robustness, accuracy, and efficiency of scene estimation of household objects in dense clutter.

6.2 Future Directions

In this section, we present some future directions for researchers to carry on ideas from this dissertation.

6.2.1 Explore space and physics for real-time inference for AxScEs

In addressing problems of *AxScEs* , one of our primary aims is to enable axiomatic perception that will enable a greater convergence of symbolic inference for task planning and collision-free motion planning and execution. Overcoming the divides between perception, planning, and action is a critical challenge for realizing the next generation of task-oriented mobile manipulators. In this regard, our *AxScEs* estimators are only a step towards this goal. There are still many issues to address, given the computational and spatial complexity that limit our current *AxPF* and *AxMC* methods. Our methods focus on the space of potential scenes. We have yet to exploit the space of plausible scenes, where the constraints of physics and space could bring scene inference into tractability. Ideally, such scene inference could occur in real-time, similar to localization for modern autonomous navigation. While our models incorporate notions of dynamics for tracking, we have left exploration of this issue as future work.

6.2.2 Maintain multi-modal distribution of object poses with multiple instances for SUM

The two-stage method presented in Chapter 4 has an assumption that only one instance appears in the scene, and the final object pose is taken wherever particle filter converges. The case becomes different when multiple instances appear simultaneously in the scene, and one of the directions is to explore more advanced particle selection techniques that maintain multiple modes in the probability distribution. This insight has been used as diverse particle selection in [79] to preserve multiple modes during belief propagation for multiple human pose estimation, where optimization is used to select those particles that naturally maintain multiple modes instead of the common re-sampling process that is easy to lose modes with lower probability.

6.2.3 Explore point-wise error in optimization process

In *GeoFusion* , geometric constraints between objects are considered into factor-graph based optimization to force proper contact between objects. When the poses of two objects are interpenetrated with each other, the geometric constraint in the optimization will pull these two objects out from each other until there is no intersection. If one object is floating on the other one, the optimization process will push them together until a proper contact. The final estimate is a geometric consistent scene estimate. However, the direction of the 'pull' or 'push' is purely random and could be guided by the point-wise error between the observation point cloud and the rendered point cloud. In this case, the guided refine direction can provide a more accurate scene estimate that matches the observations.

BIBLIOGRAPHY

[1] P. Abbeel and A. Y. Ng. Apprenticeship learning via inverse reinforcement learning. In *Proceedings of the Twenty-first International Conference on Machine Learning*, ICML '04, pages 1–, New York, NY, USA, 2004. ACM.

[2] S. Agarwal, K. Mierle, and Others. Ceres solver.

[3] B. Akgun, M. Cakmak, K. Jiang, and A. L. Thomaz. Keyframe-based learning from demonstration. *International Journal of Social Robotics*, 4(4):343–355, 2012.

[4] A. Aldoma, Z.-C. Marton, F. Tombari, W. Wohlkinger, C. Potthast, B. Zeisl, R. B. Rusu, S. Gedikli, and M. Vincze. Point cloud library. *IEEE Robotics & Automation Magazine*, 1070(9932/12), 2012.

[5] A. Aldoma, F. Tombari, R. B. Rusu, and M. Vincze. Our-cvfh–oriented, unique and repeatable clustered viewpoint feature histogram for object recognition and 6dof pose estimation. In *Joint DAGM (German Association for Pattern Recognition) and OAGM Symposium*, pages 113–122. Springer, 2012.

[6] C. G. Atkeson and S. Schaal. Robot learning from demonstration. In *Proceedings of the Fourteenth International Conference on Machine Learning*, ICML '97, pages 12–20, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[7] M. Bajracharya, J. Borders, D. Helmick, T. Kollar, M. Laskey, J. Leichty, J. Ma, U. Nagarajan, A. Ochiai, J. Petersen, et al. A mobile manipulation system for one-shot teaching of complex tasks in homes. *arXiv preprint arXiv:1910.00127*, 2019.

[8] M. Beetz, U. Klank, I. Kresse, A. Maldonado, L. Mösenlechner, D. Pangercic, T. Rühr, and M. Tenorth. Robotic roommates making pancakes. In *2011 11th IEEE-RAS International Conference on Humanoid Robots*, pages 529–536. IEEE, 2011.

[9] J. Biswas and M. M. Veloso. Localization and navigation of the cobots over long-term deployments. *Int. J. Rob. Res.*, 32(14):1679–1694, Dec. 2013.

[10] M. D. Breitenstein, F. Reichlin, B. Leibe, E. Koller-Meier, and L. Van Gool. Robust tracking-by-detection using a detector confidence particle filter. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1515–1522. IEEE, 2009.

[11] S. Calinon and A. Billard. Incremental learning of gestures by imitation in a humanoid robot. In *Second Conference on Human-Robot Interaction (HRI)*, Arlington, Virginia, 2007.

[12] C. Chao, M. Cakmak, and A. L. Thomaz. Towards grounding concepts for transfer in goal learning from demonstration. In *Proceedings of the Joint IEEE International Conference on Development and Learning and on Epigenetic Robotics (ICDL-EpiRob)*. IEEE, 2011.

[13] X. Chen, R. Chen, Z. Sui, Z. Ye, Y. Liu, R. Bahar, and O. C. Jenkins. Grip: Generative robust inference and perception for semantic robot manipulation in adversarial environments. *arXiv preprint arXiv:1903.08352*, 2019.

[14] X. Chen, J. Xie, J. Ji, and Z. Sui. Toward open knowledge enabling for human-robot interaction. *J. Hum.-Robot Interact.*, 1(2):100–117, Jan. 2013.

[15] S. Chernova and M. Veloso. Interactive policy learning through confidence-based autonomy. *Journal of Artificial Inelligence Research*, 34(1):1–25, 2009.

[16] C. Choi and H. I. Christensen. RGB-d object tracking: A particle filter approach on GPU. In *Intelligent Robots and Systems (IROS), 2013 IEEE/RSJ International Conference on*, pages 1084–1091, 2013.

[17] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan. Towards reliable grasping and manipulation in household environments. In *Experimental Robotics*, pages 241–252. Springer Berlin Heidelberg, 2014.

[18] J. Civera, D. Gálvez-López, L. Riazuelo, J. D. Tardós, and J. Montiel. Towards semantic slam using a monocular camera. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1277–1284. IEEE, 2011.

[19] A. Collet, M. Martinez, and S. S. Srinivasa. The moped framework: Object recognition and pose estimation for manipulation. *Int. J. Rob. Res.*, 30(10):1284–1306, Sept. 2011.

[20] S. Coradeschi and A. Saffiotti. An introduction to the anchoring problem. *Robotics and Autonomous Systems*, 43(2):85–96, 2003.

[21] A. Cosgun, T. Hermans, V. Emeli, and M. Stilman. Push planning for object placement on cluttered table surfaces. In *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4627–4632, 2011.

[22] C. Crick, S. Osentoski, G. Jay, and O. C. Jenkins. Human and robot perception in large-scale learning from demonstration. In *Proceedings of the 6th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, 2011.

[23] F. Dellaert, D. Fox, W. Burgard, and S. Thrun. Monte carlo localization for mobile robots. In *IEEE International Conference on Robotics and Automation (ICRA 1999)*, May 1999.

[24] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[25] K. Desingh, O. C. Jenkins, L. Reveret, and Z. Sui. Physically plausible scene estimation for manipulation in clutter. In *IEEE-RAS International Conference on Humanoid Robots (Humanoids 2016)*, 2016.

[26] S. Ekvall, P. Jensfelt, and D. Kragic. Integrating active mobile robot object recognition and slam in natural environments. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5792–5797. IEEE, 2006.

[27] I. Evtimov, K. Eykholt, E. Fernandes, T. Kohno, B. Li, A. Prakash, A. Rahmati, and D. Song. Robust physical-world attacks on deep learning models. *arXiv preprint arXiv:1707.08945*, 2017.

[28] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(9):1627–1645, Sept 2010.

[29] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32(9):1627–1645, 2010.

[30] R. E. Fikes and N. J. Nilsson. Strips: A new approach to the application of theorem proving to problem solving. *Artificial intelligence*, 2(3):189–208, 1972.

[31] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte carlo localization: Efficient position estimation for mobile robots. *AAAI/IAAI*, 1999(343-349):2–2, 1999.

[32] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[33] M. A. Goodrich, D. R. O. Jr., J. W. Crandall, and T. J. Palmer. Experiments in adjustable autonomy. In *in Proceedings of the IJCAI Workshop on Autonomy, Delegation and Control: Interacting with Intelligent Agents*, pages 1624–1629, 2001.

[34] S. Gottschalk, M. C. Lin, and D. Manocha. Obbtree: A hierarchical structure for rapid interference detection. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 171–180. Citeseer, 1996.

[35] D. H. Grollman and O. C. Jenkins. Sparse incremental learning for interactive robot control policy estimation. In *International Conference on Robotics and Automation (ICRA 2008)*, pages 3315–3320, Pasadena, CA, USA, May 2008.

[36] S. Gupta, P. Arbeláez, R. Girshick, and J. Malik. Inferring 3d object pose in rgb-d images. *arXiv preprint arXiv:1502.04652*, 2015.

[37] S. Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena*, 42(1):335–346, 1990.

[38] S. Hart, P. Dinh, and K. Hambuchen. The affordance template ros package for robot task programming. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pages 6227–6234, May 2015.

[39] W. K. Hastings. Monte carlo sampling methods using markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[40] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 2961–2969, 2017.

[41] E. Herbst, X. Ren, and D. Fox. Rgb-d object discovery via multi-scene analysis. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, pages 4850–4856. IEEE, 2011.

[42] O. C. Jenkins and M. J. Matarić. Performance-derived behavior vocabularies: Data-driven acqusition of skills from motion. *International Journal of Humanoid Robotics*, 1(2):237–288, Jun 2004.

[43] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *Proceedings of the 22nd ACM international conference on Multimedia*, pages 675–678. ACM, 2014.

[44] A. E. Johnson and M. Hebert. Using spin images for efficient object recognition in cluttered 3d scenes. *IEEE Transactions on pattern analysis and machine intelligence*, 21(5):433–449, 1999.

[45] D. Joho, G. D. Tipaldi, N. Engelhard, C. Stachniss, and W. Burgard. Nonparametric bayesian models for unsupervised scene analysis and reconstruction. In *Proceedings of Robotics: Science and Systems*, Sydney, Australia, July 2012.

[46] D. Joho, G. D. Tipaldi, N. Engelhard, C. Stachniss, and W. Burgard. Nonparametric bayesian models for unsupervised scene analysis and reconstruction. *Robotics*, page 161, 2013.

[47] R. P. Jr., A. H. Fagg, and R. A. Grupen. Null-space grasp control: Theory and experiments. *IEEE Transactions on Robotics*, 26(2):282–295, April 2010.

[48] C. C. Kemp, C. D. Anderson, H. Nguyen, A. J. Trevor, and Z. Xu. A point-and-click interface for the real world: Laser designation of objects for mobile manipulation. In *Human-Robot Interaction (HRI), 2008 3rd ACM/IEEE International Conference on*, pages 241–248, March 2008.

[49] J. Kober and J. Peters. Policy search for motor primitives in robotics. *Mach. Learn.*, 84(1-2):171–203, July 2011.

[50] I. Kostavelis and A. Gasteratos. Semantic mapping for mobile robotics tasks: A survey. *Robotics and Autonomous Systems*, 66:86–103, 2015.

[51] A. Krull, E. Brachmann, F. Michel, M. Ying Yang, S. Gumhold, and C. Rother. Learning analysis-by-synthesis for 6d pose estimation in rgb-d images. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 954–962, 2015.

[52] B. Kuipers. The spatial semantic hierarchy. *Artificial Intelligence*, 119(1):191–233, 2000.

[53] B. Kulis and M. I. Jordan. Revisiting k-means: New algorithms via bayesian nonparametrics. *arXiv preprint arXiv:1111.0352*, 2011.

[54] J. E. Laird, K. Gluck, J. Anderson, K. D. Forbus, O. C. Jenkins, C. Lebiere, D. Salvucci, M. Scheutz, A. Thomaz, G. Trafton, et al. Interactive task learning. *IEEE Intelligent Systems*, 32(4):6–21, 2017.

[55] J. E. Laird, A. Newell, and P. S. Rosenbloom. Soar: An architecture for general intelligence. *Artificial intelligence*, 33, 1987.

[56] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager. Incremental scene understanding on dense slam. In *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 574–581. IEEE, 2016.

[57] Y. Li, W. Ouyang, B. Zhou, K. Wang, and X. Wang. Scene graph generation from objects, phrases and region captions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1261–1270, 2017.

[58] V. Lifschitz. What is answer set programming?. In *AAAI*, volume 8, pages 1594–1597, 2008.

[59] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[60] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[61] Z. Liu, D. Chen, K. M. Wurm, and G. von Wichert. Table-top scene analysis using knowledge-supervised mcmc. *Robotics and Computer-Integrated Manufacturing*, 33:110 – 123, 2015. Special Issue on Knowledge Driven Robotics and Manufacturing.

[62] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[63] P. Marion, P. R. Florence, L. Manuelli, and R. Tedrake. Label fusion: A pipeline for generating ground truth labels for real rgbd data of cluttered scenes. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3325–3242. IEEE, 2018.

[64] Z.-C. Marton, D. Pangercic, N. Blodow, and M. Beetz. Combined 2d–3d categorization and classification for multimodal perception systems. *The International Journal of Robotics Research*, 30(11):1378–1402, 2011.

[65] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger. Fusion++: Volumetric object-level slam. In *2018 International Conference on 3D Vision (3DV)*, pages 32–41. IEEE, 2018.

[66] D. Mcdermott, M. Ghallab, A. Howe, C. Knoblock, A. Ram, M. Veloso, D. Weld, and D. Wilkins. Pddl - the planning domain definition language. Technical Report TR-98-003, Yale Center for Computational Vision and Control,, 1998.

[67] S. J. McKenna and H. Nait-Charif. Tracking human motion using auxiliary particle filters and iterated likelihood weighting. *Image and Vision Computing*, 25(6):852–862, 2007.

[68] F. P. Miller, A. F. Vandome, and J. McBrewster. *Lua (Programming Language)*. Alpha Press, 2009.

[69] C. Mitash, A. Boularias, and K. Bekris. Physics-based scene-level reasoning for object pose estimation in clutter. *The International Journal of Robotics Research*, page 0278364919846551, 2019.

[70] S. Mohan, A. H. Mininger, J. R. Kirk, and J. E. Laird. Acquiring grounded representations of words with situated interactive instruction. In *Advances in Cognitive Systems*. Citeseer, 2012.

[71] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.

[72] V. Narayanan and M. Likhachev. Discriminatively-guided deliberative perception for pose estimation of multiple 3d object instances. In *Proceedings of Robotics: Science and Systems*, AnnArbor, Michigan, June 2016.

[73] V. Narayanan and M. Likhachev. Perch: perception via search for multi-object recognition and localization. In *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pages 5052–5059. IEEE, 2016.

[74] S. Narayanaswamy, A. Barbu, and J. M. Siskind. A visual language model for estimating object pose and structure in a generative visual domain. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 4854–4860. IEEE, 2011.

[75] A. Neubeck and L. Van Gool. Efficient non-maximum suppression. In *18th International Conference on Pattern Recognition (ICPR'06)*, volume 3, pages 850–855. IEEE, 2006.

[76] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. W. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *ISMAR*, volume 11, pages 127–136, 2011.

[77] M. Nicolescu and M. J. Matarić. Natural methods for robot task learning: Instructive demonstration, generalization and practice. In *in: 2nd International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 241–248, 2003.

[78] S. Niekum, S. Osentoski, G. Konidaris, and G. Andrew Barto. Learning and generalization of complex tasks from unstructured demonstrations. *Intelligent Robots and Systems*, 2012.

[79] J. Pacheco, S. Zuffi, M. Black, and E. Sudderth. Preserving modes and messages via diverse particle selection. In *International Conference on Machine Learning*, pages 1152–1160, 2014.

[80] G. Papandreou, L.-C. Chen, K. P. Murphy, and A. L. Yuille. Weakly-and semi-supervised learning of a deep convolutional network for semantic image segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1742–1750, 2015.

[81] C. Papazov and D. Burschka. An efficient ransac for 3d object recognition in noisy and occluded scenes. In *Asian Conference on Computer Vision*, pages 135–148. Springer, 2010.

[82] C. Papazov, S. Haddadin, S. Parusel, K. Krieger, and D. Burschka. Rigid 3d geometry matching for grasping of known objects in cluttered scenes. *The International Journal of Robotics Research*, page 0278364911436019, 2012.

[83] S. G. Parker, J. Bigler, A. Dietrich, H. Friedrich, J. Hoberock, D. Luebke, D. McAllister, M. McGuire, K. Morley, A. Robison, and M. Stich. Optix: A general purpose ray tracing engine. *ACM Trans. Graph.*, 29(4):66:1–66:13, July 2010.

[84] P. Pastor, M. Kalakrishnan, S. Chitta, E. Theodorou, and S. Schaal. Skill learning and task outcome prediction for manipulation. In *Proceedings of the 2011 IEEE International Conference on Robotics & Automation(ICML 2011)*, 2011.

[85] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. Ros: an open-source robot operating system. In *ICRA Workshop on Open Source Software*, 2009.

[86] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[87] A. Rosinol, M. Abate, Y. Chang, and L. Carlone. Kimera: an open-source library for real-time metric-semantic localization and mapping. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, 2020.

[88] B. Rosman and S. Ramamoorthy. Learning spatial relationships between objects. *Int. J. Rob. Res.*, 30(11):1328–1342, Sept. 2011.

[89] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *Robotics and Automation, 2009. ICRA'09. IEEE International Conference on*, pages 3212–3217. IEEE, 2009.

[90] R. B. Rusu, G. Bradski, R. Thibaux, and J. Hsu. Fast 3d recognition and pose using the viewpoint feature histogram. In *Intelligent Robots and Systems (IROS), 2010 IEEE/RSJ International Conference on*, pages 2155–2162. IEEE, 2010.

[91] R. B. Rusu, Z. C. Marton, N. Blodow, and M. Beetz. Persistent point feature histograms for 3d point clouds. In *Proc 10th Int Conf Intel Autonomous Syst (IAS-10), Baden-Baden, Germany*, pages 119–128, 2008.

[92] R. B. Rusu, Z. C. Marton, N. Blodow, M. Dolha, and M. Beetz. Towards 3d point cloud based object maps for household environments. *Robotics and Autonomous Systems*, 56(11):927–941, 2008.

[93] T. Saito and T. Takahashi. Comprehensible rendering of 3-d shapes. In *Proceedings of the 17th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '90, pages 197–206, New York, NY, USA, 1990. ACM.

[94] R. F. Salas-Moreno, R. A. Newcombe, H. Strasdat, P. H. Kelly, and A. J. Davison. Slam++: Simultaneous localisation and mapping at the level of objects. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1352–1359, 2013.

[95] M. Salzmann and R. Urtasun. Combining discriminative and generative methods for 3d deformable surface and articulated pose reconstruction. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 647–654. IEEE, 2010.

[96] D. Shreiner and T. K. O. A. W. Group. Opengl programming guide: The official guide to learning opengl, versions 3.0 and 3.1. 2009.

[97] L. Sigal, A. Balan, and M. J. Black. Combined discriminative and generative articulated pose and non-rigid shape estimation. In *Advances in neural information processing systems*, pages 1337–1344, 2008.

[98] W. D. Smart and L. P. Kaelbling. Effective reinforcement learning for mobile robots. In *Robotics and Automation, 2002. Proceedings. ICRA '02. IEEE International Conference on*, volume 4, pages 3404–3410 vol.4, 2002.

[99] S. Srivastava, L. Riano, S. Russell, and P. Abbeel. Using classical planners for tasks with continuous operators in robotics. In *Proceedings of the ICAPS Workshop on Planning and Robotics (PlanRob)*, 2013.

[100] M. R. Stevens and J. R. Beveridge. Localized scene interpretation from 3d models, range, and optical data. *Computer Vision and Image Understanding*, 80(2):111–129, 2000.

[101] Z. Sui, H. Chang, N. Xu, and O. C. Jenkins. Geofusion: Geometric consistency informed scene estimation in dense clutter. In *IEEE Robotics and Automation Letters (In Submission)*, 2020.

[102] Z. Sui, O. C. Jenkins, and K. Desingh. Axiomatic particle filtering for goal-directed robotic manipulation. In *International Conference on Intelligent Robots and Systems (IROS 2015)*, Hamburg, Germany, Oct 2015.

[103] Z. Sui, L. Xiang, O. C. Jenkins, and K. Desingh. Goal-directed robot manipulation through axiomatic scene estimation. *The International Journal of Robotics Research*, 36(1):86–104, 2017.

[104] Z. Sui, Z. Zhou, Z. Zeng, and O. C. Jenkins. Sum: Sequential scene understanding and manipulation. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3281–3288, Sept 2017.

[105] M. Tavenrath and C. Kubisch. Advanced scenegraph rendering pipeline. In *GPU Technology Conference*, San Jose, May 2013.

[106] M. Tenorth and M. Beetz. Knowrob: A knowledge processing infrastructure for cognition-enabled robots. *Int. J. Rob. Res.*, 32(5):566–590, Apr. 2013.

[107] S. Thrun, D. Fox, W. Burgard, and F. Dellaert. Robust monte carlo localization for mobile robots. *Artificial intelligence*, 128(1-2):99–141, 2001.

[108] G. Trafton, L. Hiatt, A. Harrison, F. Tamborello, S. Khemlani, and A. Schultz. Act-r/e: An embodied cognitive architecture for human-robot interaction. *Journal of Human-Robot Interaction*, pages 30–55, 2013.

[109] J. Tremblay, T. To, B. Sundaralingam, Y. Xiang, D. Fox, and S. Birchfield. Deep object pose estimation for semantic robotic grasping of household objects. *arXiv preprint arXiv:1809.10790*, 2018.

[110] M. Vondrak, L. Sigal, J. Hodgins, and O. Jenkins. Video-based 3d motion capture through biped control. *ACM Transaction of Graphics (TOG) (Proceedings of ACM SIGGRAPH)*, 2012.

[111] C. Wang, D. Xu, Y. Zhu, R. Martín-Martín, C. Lu, L. Fei-Fei, and S. Savarese. Densefusion: 6d object pose estimation by iterative dense fusion. *arXiv preprint arXiv:1901.04780*, 2019.

[112] T. Whelan, S. Leutenegger, R. Salas-Moreno, B. Glocker, and A. Davison. Elasticfusion: Dense slam without a pose graph. In *Robotics: Science and Systems*, 2015.

[113] T. Winograd. Understanding natural language. *Cognitive psychology*, 3(1):1–191, 1972.

[114] S. Wintermute and J. E. Laird. Bimodal spatial reasoning with continuous motion. In *Proceedings of the 23rd National Conference on Artificial Intelligence - Volume 3*, AAAI'08, pages 1331–1337. AAAI Press, 2008.

[115] L. L. Wong, L. P. Kaelbling, and T. Lozano-Pérez. Data association for semantic world modeling from partial views. *The International Journal of Robotics Research*, 34(7):1064–1082, 2015.

[116] Y. Xiang, T. Schmidt, V. Narayanan, and D. Fox. Posecnn: A convolutional neural network for 6d object pose estimation in cluttered scenes. *arXiv preprint arXiv:1711.00199*, 2017.

[117] H. Yanco, A. Norton, W. Ober, D. Shane, A. Skinner, and J. Vice. Analysis of human-robot interaction at the darpa robotics challenge trials. *Journal of Field Robotics*, 32(3):420–444, 2015.

[118] R. Zellers, M. Yatskar, S. Thomson, and Y. Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5831–5840, 2018.

[119] Z. Zeng, Z. Zhou, Z. Sui, and O. C. Jenkins. Semantic robot programming for goal-directed manipulation in cluttered scenes. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7462–7469. IEEE, 2018.

[120] K. Zhang and D. Shasha. Simple fast algorithms for the editing distance between trees and related problems. *SIAM J. Comput.*, 18(6):1245–1262, Dec. 1989.

[121] Z. Zhang. Microsoft kinect sensor and its effect. *IEEE multimedia*, 19(2):4–10, 2012.

[122] B. Zhuang, L. Liu, C. Shen, and I. Reid. Towards context-aware interaction recognition for visual relationship detection. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 589–598, 2017.