Efficient Unicontact Grasping in Cluttered Scenes

Vishal Satish
Ken Goldberg, Ed.
Angjoo Kanazawa, Ed.

Electrical Engineering and Computer Sciences
University of California, Berkeley

Technical Report No. UCB/EECS-2021-112
http://www2.eecs.berkeley.edu/Pubs/TechRpts/2021/EECS-2021-112.html

May 14, 2021

Acknowledgement

(Abbreviated)

To Professor Ken Goldberg, for his mentorship, feedback, and support.

To Dr. Jeffrey Mahler, for his guidance and supervision.

To my fellow researchers, for providing me with an environment I could thrive in.

To my good friends for providing me with constant encouragement.

To my family for their endless love.

Efficient Unicontact Grasping in Cluttered Scenes

by

Vishal Satish

A dissertation submitted in partial satisfaction of the

requirements for the degree of

Master of Science

in

Electrical Engineering and Computer Science

in the

Graduate Division

of the

University of California, Berkeley

Committee in charge:

Professor Ken Goldberg, Chair
Professor Angjoo Kanazawa

Spring 2021

The dissertation of Vishal Satish, titled Efficient Unicontact Grasping in Cluttered Scenes, is approved:

Chair Ken Goldberg Date May 14, 2021

Ken Goldberg (May 14, 2021 13:21 PDT)

Date May 14, 2021

University of California, Berkeley

Efficient Unicontact Grasping in Cluttered Scenes

Abstract

Efficient Unicontact Grasping in Cluttered Scenes

by

Vishal Satish

Master of Science in Electrical Engineering and Computer Science

University of California, Berkeley

Professor Ken Goldberg, Chair

Mechanical search – finding and extracting a known target object from a cluttered environment – is a key challenge in automating warehouse, home, retail, and industrial tasks. In this thesis we consider contexts where occluding objects are to remain untouched to minimize disruptions and avoid toppling. We assume a 6-axis robot has an RGBD camera and suction gripper mounted on its wrist, such that it can move to an approach vector along which the suction gripper can both be inserted to grasp the target object and then retracted to extract it. We formalize the problem of efficiently finding an approach vector and present AVPLUG: Approach Vector PLanning for Unicontact Grasping: an algorithm using a fast oct-tree occupancy model and Minkowski sum computation to maximize information gain. Experiments in simulation and with a physical Fetch robot suggest that AVPLUG finds an approach vector up to 10× faster than a baseline search policy.

To my parents.

Contents

List of Figures

List of Tables

Acknowledgments

This work would not have been possible without the tremendous guidance and support of many individuals. It would be impossible to list them all here, but I would like to highlight some who were instrumental.

First, I would like to give a huge thanks to my academic advisor for the last 4 years, Professor Ken Goldberg. Professor Goldberg provided invaluable mentorship, feedback, and support throughout all of my research endeavours. He taught me the importance of performing rigorous benchmarking and ensuring reproducability - both of which I believe are what enable his research to escape the confines of a sterile lab environment and find itself actively deployed in real world settings. I was fortunate enough to see my own research follow this same trajectory.

I would like to also thank Dr. Jeffrey Mahler for taking me under his wing towards the beginning of my undergraduate career and nurturing me over the next few years by teaching me the necessary skills for performing stellar research. I would especially like to highlight his responsiveness to my never-ending questions, constantly providing thoughtful explanations that imparted years of practical experience no textbook could ever have taught me.

I would like to thank all the other researchers I was fortunate enough to collaborate with during my time at Berkeley, in particular Matt Matl, Yahav Avigal, Jeff Ichnowski, Michael Danielczuk, Richard Liaw, Eric Liang, and Michael Luo. You guys provided me with mentorship, support, encouragement, and most importantly an intellectually-stimulating environment that I could thrive in.

I would never have been able to so wholeheartedly take on such ambitious research goals with a smile on my face without the constant support of my roommates and best friends, Karthik Bharathala, Aniket Mandalik, Chaitanya Angadala, Francis Kumar, and Naren Krishna. You guys constantly kept my spirits up even in the most challenging times.

Finally, I would like to extend the deepest regards to my parents and sister, without whom I would never be where I am today.

Chapter 1

Introduction

1.1 Overview

The focus of my research over the last four years has been on robotic manipulation, in particular learning robust grasping policies that can generalize across a variety of challenging object geometries.

I started out working on the Dexterity-Network (Dex-Net) project led by Dr. Jeffrey Mahler. My initial contributions consisted of developing extensible infrastructure for fast and efficient training of Grasp Quality Convolutional Neural Networks (GQ-CNNs) on large synthetic datasets on the order of millions of samples. This work was later open-sourced as a Python package [34] to facilitate collaboration with other researchers both in academia and industry and encourage reproducibility, which is an often overlooked but essential principle of exceptional research that Professor Goldberg has taught me to strive for. Over the years I have continued to maintain the library and update it with our latest results. This work was later published as part of Dex-Net 4.0 [25] in the Science Robotics journal.

Next my focus shifted towards our core research question: "How can we improve the Mean Picks Per Hour (MPPH) of grasping policies?". I explored a two-pronged approach consisting of a more efficient on-policy training distribution coupled with a fast and efficient fully-convolutional variant of our existing network architecture dubbed the "Fully Convolutional" Grasp Quality CNN (FC-GQ-CNN). The result was a policy that could evaluate millions of grasps in under a second in a single inference pass and achieved a 20% higher MPPH than the existing policy based on iterative sampling and evaluation. This work was published in the Robotics and Automation Letters (RA-L) journal [35] and presented at the International Conference on Robotics and Automation (ICRA) in 2019. My prior work on an efficient and extensible infrastructure proved instrumental here by enabling fast iteration over different input modalities, network architectures, and training hyperparameters.

Along the way our team was fortunate enough to have the opportunity to collaborate with our peers in industry working at Siemens to showcase the Dex-Net project at the Hannover World Trade Show running in a mock production environment for warehouse order

fulfillment. To this end, I helped port the now SOTA FC-GQ-CNN and the accompanying training infrastructure over to the Intel Neon deep learning framework enabling lightning-fast inference in a small power envelope on Intel's SOTA Myriad X Vision Processing Unit (VPU). This entailed diving into the nitty-gritty details of low-level network compilation and memory profiling.

It was around this time that Dr. Mahler, my mentor up until now, finished his PhD and moved on to industry. Still eager to continue my work on grasping, I started collaborating with Dr. Jeffrey Ichnowski, an expert in motion planning. Combining Dr. Ichnowski's vast knowledge of trajectory optimization with my expertise in grasping, we published work on Grasp Optimized Motion Planning (GOMP) [17]. We further extended this work with the help of Yahav Avigal to achieve Deep Jerk-minimized Grasp Optimized Motion Planning (DJ-GOMP) that utilized deep learning to warm-start the optimization process resulting in 300x faster computation times. This work was featured in the Science Robotics journal [16].

Finally it came time for my masters, and I decided I wanted to once again focus on the problem of grasp planning. Up until now, my work had focused on planning from a single observation such as an RGBD image of the scene. However, in sequential tasks such as decluttering, we can utilize a rich history of prior observations to make a more informed decision at the current timestep. This begs the question of what is the best representation to encapsulate this history and how to both intelligently and efficiently query it for the desired objective. Yahav and I set out to explore this question in the context of unicontact suction grasping in cluttered scenes, and that is the focus of this work.

1.2 Introduction

In many automation tasks, such as extracting a product from a warehouse shelf, removing an ingredient from a refrigerator, or picking up a tool on a cluttered workbench, desired objects are often hidden behind other objects, presenting a challenge to find both the object and a way of grasping it. To automate such tasks, robots require both the ability to perform an informed visual search, and to robustly grasp and manipulate target objects once found. Although several methods have been proposed for grasping objects in isolation [21, 22, 27], finding a robust grasp becomes significantly more challenging in a cluttered environment where the target object may be partially or fully occluded.

Mechanical Search [8] aims to find a desired object in clutter and focuses on clearing a view to the target by pushing or removing occluding objects [8, 10, 15, 19, 31]. This approach requires planning and executing multiple collision-free motion of the arm, adding to the overall time taken in the form of both motion planning and execution. In many environments, the placement of occluding objects is structured, such as objects resting on a kitchen counter or a supermarket shelf [28], and pushing or removing objects may be undesirable. In addition, in places where objects have few stable equilibria, even glancing contacts can lead to undesired toppling which can damage delicate objects such as glass bottles. In contrast, this work focuses on servoing a wrist-mounted camera with a unicontact

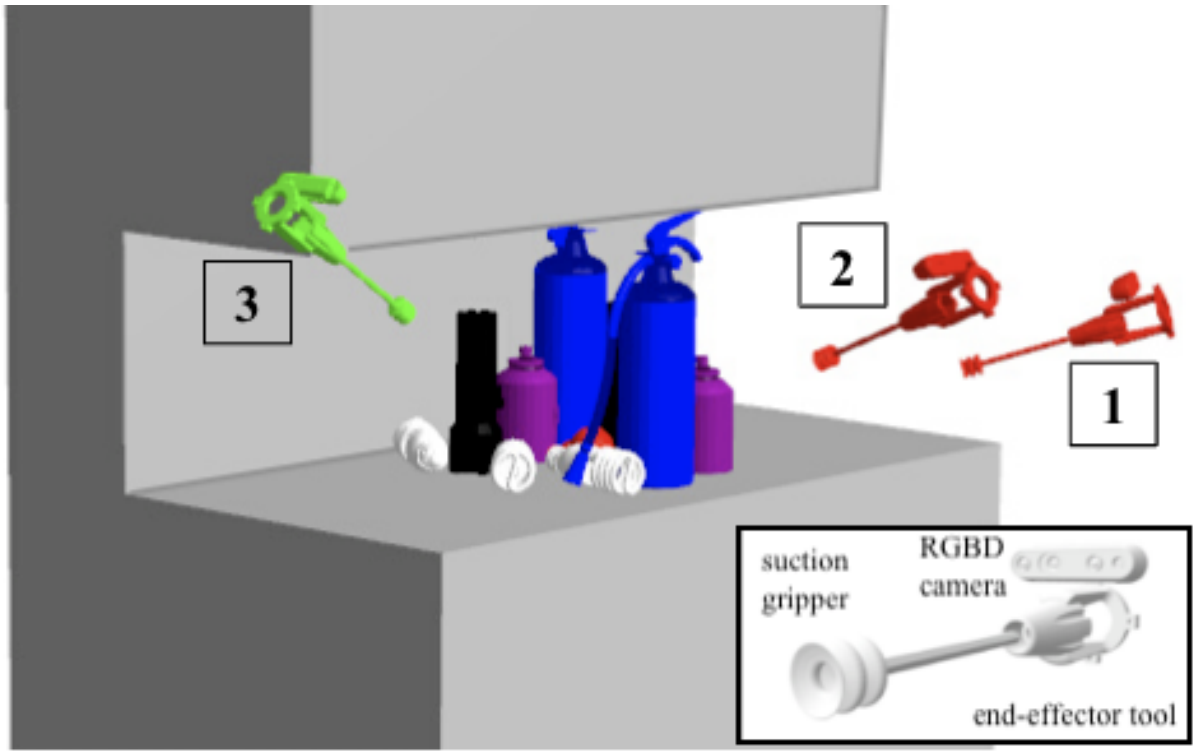


Figure 1.1: Target object (on the worksurface in red) is occluded. AVPLUG moves the wrist-mounted tool to a view from which it can find an approach vector for unicontact grasping (in green). AVPLUG uses an occupancy map and a Minkowski sum to track the previously explored regions of the scene and to evaluate the information gain of candidate vectors. **Inset:** The end-effector tool used by AVPLUG, comprised of an RGBD camera with its optical axis aligned to a unicontact suction gripper.

suction gripper tool (see Fig. 1.1) to a view from which the target object is visible and a single robust grasp can be planned without the risk of toppling other objects.

Efficiently searching for a view of a target object is related to next-view planning, the problem of finding a single additional sensor placement to improve a current full scene reconstruction [11] and has a rich history in computer vision [32, 3, 4]. Next-view planning requires keeping track of which regions of the scene have already been explored and which have not. For the task of unicontact grasp planning, a full scene reconstruction is computationally expensive and unnecessary; thus, we propose using an efficient 3D voxel-based occupancy map (e.g., Octomap [14]) as it provides the required information and can be computed rapidly.

We focus on unicontact (suction) grasping. As opposed to parallel-jaw grasping, in which the contact points are rarely visible from the approach vector, in unicontact grasping the

grasp quality is highly correlated with the visible surface normals [23]. Accordingly, by aligning the wrist-mounted camera's optical axis with the gripper approach vector, finding a view from which we can plan a unicontact grasp corresponds to finding a gripper's approach vector and therefore a unicontact gripper is well suited for a visibility-based grasp exploration policy.

We present Approach Vector Planning for Unicontact Grasping (AVPLUG), an algorithm that leverages an occupancy map based on an octree and Minkowski sum computation to find an approach vector for unicontact grasping of a partially or fully occluded target object in a structured clutter scene, without changing the scene. First, AVPLUG efficiently casts rays outwards from potential locations of the target object in the unknown regions of the occupancy map to identify unobstructed candidate vectors. Next, it casts rays inwards from each vector to the neighborhood of the target object to simulate the field-of-view (FOV) of the camera and evaluate the information-gain from positioning the camera in each vector. Then, it moves to the view that will gain the most information, repeating the process until a view on a graspable area of the target object is revealed, or report failure.

We also consider the scenario in which the target object is fully occluded and there is no model of the environment. The additional challenge here is that there is no clear signal to guide exploration. Systematic exploring approach vectors in which the target object can be hidden is inefficient. Instead, we propose to efficiently compute the Minkowski sum [13] between the target object and the region of the occupancy map that we have explored thus far to constrain the potential locations of the target object on the worksurface.

Experiments with simulation and on a physical Fetch robot suggest that AVPLUG can find an approach vector in up to 10× fewer steps compared to a baseline policy that systematically visits views on a discretized grid, even in the presence of dense occlusions and in tight spaces. This thesis makes three contributions:

1. A formulation of the problem of efficiently finding an approach vector for unicontact grasping a target object in the presence of partial or full occlusions.

2. AVPLUG, an efficient algorithm that uses an octree-based occupancy map and Minkowski sum computation for the above problem.

3. Experiments in simulation and on a physical robot comparing AVPLUG with a grid search baseline.

My primary contributions to this end were in developing the simulation environment in which we evaluate different policies and ablations thereof, helping to design the inside→out and outside→in raycasting algorithm, policy implementation, and performance optimizations for fast parallel batched raycasting.

Chapter 2

Related Work

2.1 Target-driven grasping in clutter

Danielczuk et al. [8] defined the Mechanical Search Problem and proposed a pipeline to iteratively search for a partially occluded object through a series of parallel-jaw grasping, suction, and pushing actions. Huang et al. [15] and Danielczuk et al. [10] then extended the Mechanical Search problem by learning an occupancy distribution to guide the search process to recover the occluded target. Xiao et al. [39] formulate the object search in clutter task as a POMDP and suggest an algorithm that takes into account the robot's current belief to evaluate the success of a manipulation task. Murali et al. [28] leveraged a variational autoencoder [27] to plan 6-DoF parallel-jaw grasps on a partially occluded target object in a cluttered scene, and remove occluding objects if no feasible grasp is found. Boroushaki et al. [5] identify and locate a fully occluded target object using RFID tags. In this work, we focus on aligning the optical axis of a wrist-mounted camera with an approach vector from which the target object can be grasped.

2.2 View planning for grasping

In active perception [3, 4, 2, 1] the position of the sensor changes to reveal more of the scene's geometry for tasks such as 3D scene reconstruction [30] and mapping [7]. Accordingly, the next-best-view planning problem computes the optimal next view with respect to the desired goal. In the context of manipulation, a camera mounted on a robot end-effector can guide the motion. Kahn et al. [18] models the occluded regions where the target object may be located as a mixture of Gaussians and encourages exploration during the trajectory optimization by penalizing for uncertainty. Other works constrain the action space to top-down (4-DoF) grasps. Morrison et al. [26] proposed a top-down grasp planning controller with active perception to choose the next-best-view of the camera as it approaches the target object along the z-axis to reveal more robust grasps. Novokovic et al. [29], propose a reinforcement learning based active and interactive perception system from a top-down view to uncover

a hidden target cube in a pile of cubes. In contrast, in this work we consider approach directions on a sphere centered on the clutter centroid and consider candidate 5-DoF grasps, since suction grasps have symmetry about the approach vector.

2.3 Occupancy maps

Occupancy maps are 3D representations of the environment for next-best-view planning that posses information about which regions have already been explored and which have not. Hornung et al. [14] presented Octomap, an efficient implementation of an octree-based occupancy map that given a point cloud updates a 3D voxelized representation of the scene with one of three labels per voxel: occupied, empty or unknown. Santos et al. [33] used an octree while exploring a scene with a robotic arm and a wrist-mounted camera for generating a 3D reconstruction of the scene. Octrees are also used for grasping a target object in a cluttered scene in [19, 31, 12], however in contrast to moving the camera, these works remove an occluding object from the scene to expose the target object.

Chapter 3

Approach Vector Planning

3.1 Problem Statement

Given:

- An RGBD camera with known intrinsics, mounted in alignment with a vacuum suction cup gripper on a robot arm (see Fig. 1.1).
- A target object of known geometry.
- An environment of unknown objects resting on a planar worksurface, partially or fully occluding the target object.
- A target object detector that returns a binary mask of the target object if it is visible from the RGBD camera.
- A suction grasp planner that samples candidate suction points on a depth image and returns the point with the highest associated grasp quality value.

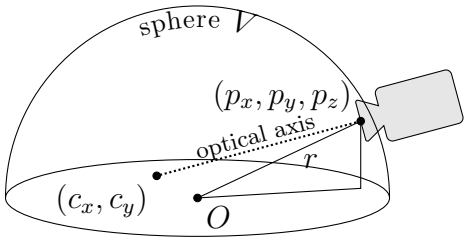


Figure 3.1: States in AVPLUG consist of a camera location (p_x, p_y, p_z) on a sphere V with radius r, and a point on the worksurface (c_x, c_y) (with implicit $c_z = 0$) representing a potential target location, at which the camera's optical axis points.

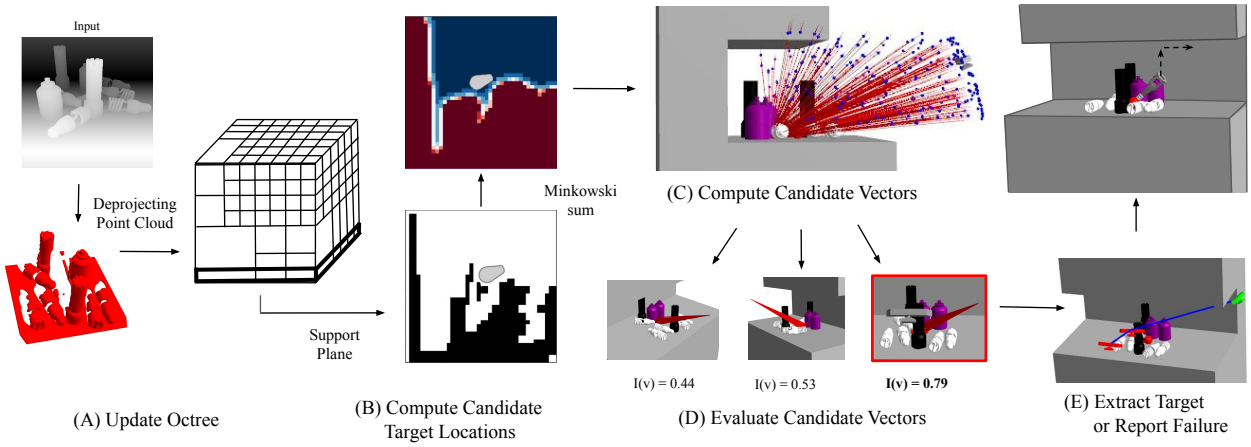


Figure 3.2: **AVPLUG Overview.** The update octree stage (A) deprojects an input depth image taken from the current state to a point cloud and inserts it to the octree. The support plane is then queried from the octree to identify candidate target locations in the unknown regions (white is unknown, black is occupied) where the target object might hide. The target object is illustrated in gray for the reader, although in practice its location is unknown to AVPLUG. Computing the Minkowski sum between the convex hull of the known target object and the support plane reduces the number of candidate target locations (B). AVPLUG casts rays outwards from candidate target locations in order to find candidate vectors on a sphere around the workspace (marked by blue points in (C)). AVPLUG evaluates the information gain from each vector (D), i.e., the amount of unknown voxels visible from that vector by simulating the camera field-of-view and chooses the vector that maximizes objective 3.1. AVPLUG uses a grasp planner to find a grasp aligned with the approach vector and extracts the grasped target object with an upward motion (E).

Output: an approach vector $\mathbf{v}$ along which a collision-free linear motion can achieve a unicontact grasp of the target object in a minimum number of steps or report failure.

Definitions

We consider the following states, actions and observations in the problem definition:

Worksurface. We define the worksurface to be a planar surface orthogonal to the z-axis which is aligned to gravity. The worksurface may be bounded due to reachability, and from above by a ceiling plane.

Sphere. We define V to be a sphere with radius r centered on the origin of the worksurface (see Fig. 3.1).

States ($\mathcal{S}$). Let $\mathbf{s}_t \in \mathcal{S}$ denote a state at timestep t defining the position and orientation of the camera on V. We restrict motion of the camera focal point to be on the bounds of the

worksurface. The camera can rotate about its placement on V to look at any point on the planar surface, but does not roll around its optical axis. The state space is thus $\mathcal{S} = S^2 \times S^2$, which we represent with a pair of Cartesian coordinates $(\mathbf{p}, \mathbf{c})$, where $\mathbf{p} \in \mathbb{R}^3$ is the location on V, and $\mathbf{c} \in \mathbb{R}^3$ is the point on the worksurface that the camera's optical axis intersects, thus $c_z = 0$ (see Fig. 3.1). Let $\mathbf{v} = \mathbf{p} - \mathbf{c}$ be the approach vector, defined as the direction from the camera to the target.

Actions (A). Let $\mathbf{a}_t = (\Delta p_x, \Delta p_y, \Delta p_z, \Delta c_x, \Delta c_y)$ denote the change from the state $\mathbf{s}_t$ to state $\mathbf{s}_{t+1}$, where $\mathbf{s}_{t+1}$ is restricted to remain on V.

Observations (Ω). Let $\mathbf{y}_t = \mathbb{R}_+^{H \times W \times 4}$ be an RGBD image with height H and width W taken from state $\mathbf{s}_t$ at timestep t.

Surrogate Objective

Optimizing the above objective directly is difficult due to the large number of possible approach vectors and the partial observability of the environment. Instead, we formulate a surrogate objective describing the information gained from an approach vector $\mathbf{v}$:

$$\mathbf{v}_t = \arg\max_{\mathbf{v}} \left(\alpha \cdot \mathcal{I}(\mathbf{v}) - (1 - \alpha) \cdot L(\mathbf{v}_{t-1}, \mathbf{v})\right) \tag{3.1}$$

We define information $\mathcal{I}$ as the number of known voxels in the octree (labeled occupied or empty) and accordingly the information-gain from vector $\mathbf{v}_t$ is the amount of unknown voxels at timestep t in the FOV of a camera at state $\mathbf{s}_t$ corresponding to vector $\mathbf{v}_t$. Given observations $(\mathbf{y}_1, \ldots, \mathbf{y}_{t-1})$ of the environment corresponding to states $(\mathbf{s}_1, \ldots, \mathbf{s}_{t-1})$ at timestep t, our goal is to find an approach vector $\mathbf{v}_t$ that maximizes the information-gain $\Delta\mathcal{I}(\mathbf{v}_t)$. Since multiple approach vectors can produce high information gain, we add a term for the L geodesic distance from the previous view along the sphere V, where α is a constant such that $0 < \alpha < 1$.

3.2 Approach Vector Planning

Given observation $\mathbf{y}_t$, AVPLUG attempts to find a grasp aligned within a tolerance angle ψ of the camera optical axis. In experiments we set $\psi = 15^\circ$ to account for precision errors in sensing and actuation. After detecting and segmenting the target object, AVPLUG samples and evaluates grasps from its visible surface using a provided grasp planner, $\mathcal{G} : \mathbb{R}_+^{H \times W} \rightarrow (\mathbb{R}^5, \mathbb{R})$, mapping grasps parameterized by a 5-DOF pose $g \in \mathbb{R}^3 \times S^2$ to the corresponding grasp quality $q \in [0, 1]$ with a higher value indicating a more robust grasp. If a termination condition $\mathcal{T}$ is not reached, i.e. the highest grasp quality found in $\mathbf{y}_t$ does not meet a certain threshold, AVPLUG finds the next approach vector.

Finding an approach vector for unicontact grasping that maximizes the surrogate objective in equation (3.1) is challenging since the visual signal, e.g, the visibility of the target object, is low due to inter-object and environmental occlusions, and without full knowledge

of object poses and geometries, it is difficult to estimate which view will uncover a graspable part of the target object. We address this with an occupancy map based on an octree, $\mathcal{M} : \mathbb{R}^3 \to \{-1, 0, 1\}$, that maps voxels, i.e., minimum-size boxes in the octree, to occupancy values, with -1 meaning unknown occupancy, 0 meaning known to be empty, and 1 meaning known to be occupied. We aggregate point clouds of the scene taken from different states into the occupancy map. This allows the policy to efficiently keep track of unexplored unknown regions of the scene and prioritize them in subsequent steps. The resolution of the octree is configurable on initialization, with higher resolution allowing for a more accurate search at the expense of increased processing time. Accordingly, if the target object is initially fully occluded, the octree is initialized with a higher resolution compared to the partially occluded case. This empirically leads to higher success rates and a lower number of steps to success.

Updating Octree

To update AVPLUG's representation of the occupancy map, the depth image in observation $\mathbf{y}_t$ is deprojected to a point cloud using the known camera intrinsics. It is then transformed to a global coordinate frame centered at the center of the worksurface using the known camera extrinsics and inserted into the occupancy map $\mathcal{M}$ (Fig. 3.2(A)).

Finding Candidate Target Object Locations

Assuming that all objects are resting on a planar worksurface, AVPLUG reduces the computational complexity of the problem by limiting the search for candidate target object locations to a 2D plane in the worksurface. We define a 2D occupancy map as the 2D slice of the octree that corresponds to the support plane, i.e., the worksurface which is defined by $z = 0$ in the global coordinate frame (see Fig. 3.2(B)). We note that AVPLUG does not project the occupancy map onto the support plane, as this can result in missing a target object that is hidden beneath another object (e.g., a small object hidden below a large bowl). We also observe that there are only occupied and unknown voxels on the support plane. Using the above, AVPLUG searches for a set of candidate target object locations $\mathcal{U}$ in the unknown region (Fig. 3.2(B)).

Partially Occluded Target Case

Here AVPLUG restricts the search to unknown regions closest to the visible portion of the target object by computing the set of points $\mathcal{U}$ for which the Euclidean distance to a point occupied by the target object is lower than a threshold. These will have the highest probability of uncovering more of the object.

Fully Occluded Target Case

When the target object is fully occluded, finding candidate vectors is more challenging due to the initially large number of unknown voxels on the support plane. Given the geometry of

the target object, and assuming it has a finite set of feasible stable poses on the worksurface, we use a Minkowski sum [13] to estimate an occupancy distribution for the location of the target object. To compute the Minkowski sum we first generate polygons from the occupied region by converting the 2D occupancy map to a binary image and finding the contours of the occupied components (See Fig. 3.2(B)). If the contours form self-intersecting polygons, we smooth them using erosion and dilation [36]. To create a polygon of the target object, we project the vertices of the known mesh to the support plane and compute the convex hull of the projected vertices. We then compute the Minkowski sum between the polygons to eliminate unknown points that are less likely to occupy the target object. Since the stable pose of the target object and its rotation about the z axis are unknown, we discretize the rotations into 8 bins and compute a Minkowski sum per stable pose and discretized rotation, sum the results and normalize to estimate a distribution for the location of the target object. We then use this distribution to uniformly sample a set of points $\mathcal{U}$ that maximize the likelihood of occupying a portion of the target object.

Finding Candidate Vectors

To find a candidate approach vector $\mathbf{v}$ such that an unknown point $\mathbf{u} \in \mathcal{U}$ is in the FOV of the camera when positioned in $\mathbf{v}$, AVPLUG casts rays outwards from candidate target locations, drawing inspiration from Lozano-Perez et al. [20] who describe an approach to fine motion synthesis by chaining backwards from a known goal towards the current position (Fig. 3.2(C)). If ray i in direction $\mathbf{d}$ does not intersect any occupied voxels, it represents a clear line-of-sight, and the intersection point $\mathbf{p} \in \mathbb{R}^3$ of the ray with V is computed (see the blue points in Fig. 3.2(C)). There may be few or many candidate vectors, in correspondence with the levels of occlusion in the scene. During the process of backward ray casting, a valid candidate vector consists of an intersection point and the inverse direction leading to it: $\mathbf{v} = (\mathbf{p}_i, -\mathbf{d})$.

Evaluating Candidate Views

Next, we evaluate the information gain $\Delta\mathcal{I}(\mathbf{v})$ from each candidate vector $\mathbf{v}$. Each candidate vector has a clear line-of-sight to at least a single unknown point on the support plane, namely, the point on the support plane in direction $-d$ from the camera view. To rank a vector $\mathbf{v}$, we determine the information gain $\Delta\mathcal{I}(\mathbf{v})$ by casting rays from each candidate vector in directions uniformly sampled from a cone (with angle β) corresponding to the camera FOV (Fig. 3.2(D)). We then find the intersection between the rays and the support plane and compute the proportion of unknown to occupied points to get $\Delta\mathcal{I}(\mathbf{v})$.

Ranking the next view only by the level of information gain might be suboptimal in the overall exploration time for two reasons: there might be more than one vector that leads to maximal information gain, and on the first steps of the algorithm when most of the scene is still unknown, distant vectors provide high information gain by default although they might not reveal more of the target object. To encourage efficient motions that minimize the

exploration time, we also compute the geodesic distance from each candidate vector to the current vector and penalize distant views. This forms the ranking objective in equation (3.1). The tradeoff between distance and information gain is controlled by the weight α.

Chapter 4

Experiments

To evaluate AVPLUG, we run experiments in both simulated and physical environments and compare to a baseline policy.

Simulation Experiments

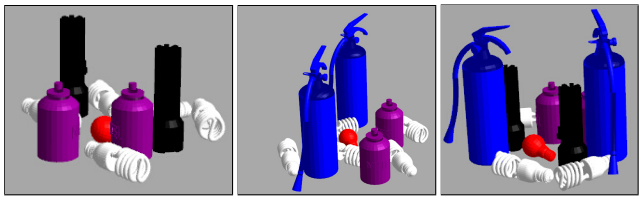
We use $R = 0.01\,\text{m}$ resolution if the initially the target obj is fully occluded, and $R = 0.03\,\text{m}$ otherwise. This resolution empirically allows for a more accurate Minkowski sum. To implement the octree for the occupancy map, we use the open-source OctoMap [14, 38].

We use ground truth segmentation to generate a binary mask of the target object. Since AVPLUG relies on an external instance segmentation algorithm, in practice, one can use an off-the-shelf object segmentation algorithm such as SD Mask R-CNN [9] with an additional matching phase for classification.

To decide whether the target object is graspable from the current state we use a grasp planner based on Dex-Net 3.0 [24] as an oracle. Dex-Net 3.0 pre-computes suction grasps and associates quasi-static wrench-resistance quality metrics to a target object mesh, then matches these to pre-computation results at evaluation time.

Environments in Simulation

We first evaluate AVPLUG on two simulated scenes: 1) a tabletop, for which the potential vectors on the sphere V range between elevation angle $\theta \in [0°, 85°]$ and azimuth angle $\phi \in [-90°, 90°]$, and 2) a counter with a cabinet above that constrains the possible grasp approach directions to a slice of V ranging between elevation angle $\theta \in [55°, 85°]$ and azimuth angle $\phi \in [-90°, 90°]$, with a radius $r = 0.6\,\text{m}$ for both. To generate varying levels of occlusions we use $N = 10$ objects from Thingiverse [37] and YCB [6] of varying heights. We generate initial object poses by uniformly sampling the locations of the objects in a bounded $0.4\,\text{m} \times 0.4\,\text{m}$ worksurface and positioning them in a stable pose. We also randomize the initial camera view by rejection sampling until a non-graspable view is found. We generate 3 tiers of scene complexity (Fig. 4.1) by altering the proportion of low-to-high objects which



(a) Tier 1 (b) Tier 2 (c) Tier 3

Figure 4.1: **Difficulty Tiers with Varying Levels of Occlusions.** The target is shown in red on a tabletop environment. The difficulty tiers define scenes with increasing levels of complexity due to increased occlusions. **Left:** Tier 1 includes two flashlights, two spray bottles and five spiral bulbs. **Center:** Tier 2 includes two fire extinguishers, two spray bottles and five spiral bulbs. **Right:** Tier 3 includes two fire extinguishers, two flashlights, two spray bottles and three spiral bulbs.

Figure 4.2: Comparisons of steps to completion of successful rollouts in simulation. The scale on the horizon axis is 10× larger for the *GridSearch* policy.

affects the level of occlusions in the scene: **Tier 1** consists of 2 flashlights, 2 spray bottles and 5 spiral bulbs. **Tier 2** replaces the flashlights from tier 1 with 1.6× higher and 1.7× wider fire extinguishers. **Tier 3** consists of 2 fire extinguishers, 2 flashlights, 2 spray bottles, and 3 spiral bulbs. For all tiers, the target object is a light bulb, which is easily occluded due to its small proportions compared to the other objects in the scene. After executing a successful grasp, the target object is extracted in an upwards motion (Fig 3.2(E)).

			# Steps		Distance [m]	
Scene	**Tier**	**Policy**	Median	IQR	Median	IQR
Tabletop	1	GridSearch	16.0	22.2	**0.8**	**1.0**
		AVPLUG	**2.0**	**3.5**	2.7	1.0
	2	GridSearch	20.0	26.0	**1.0**	1.4
		AVPLUG	**2.0**	**2.8**	2.6	**1.1**
	3	GridSearch	16.0	16.0	**0.8**	**0.9**
		AVPLUG	**3.0**	**3.0**	1.7	1.6
Counter	1	GridSearch	14.5	19.0	**0.7**	**0.9**
		AVPLUG	**2.0**	**3.0**	1.8	1.6
	2	GridSearch	15.5	18.0	**0.7**	**0.9**
		AVPLUG	**3.0**	**6.0**	2.6	4.3
	3	GridSearch	18.0	19.5	**0.9**	**1.0**
		AVPLUG	**3.0**	**6.5**	2.7	3.8

Table 4.1: **Simulation Experiments.** Median and interquartile range (IQR) of the number of steps to success and the distance traveled for each policy over 100 rollouts in two simulated environments. The metrics are reported for successful rollouts. The success rate is 100% for the *GridSearch* baseline and 95-100% for AVPLUG. Failure modes are described in Fig. 4.3

Grid Search Baseline

We compare AVPLUG to a *GridSearch* baseline that systematically searches for a graspable view by discretizing the sphere V into 212 fixed-spaced views (the distance between neighboring views is $l = 5°$ in both elevation and azimuth) and visits each view in a raster sequence until reaching a view from which it can plan a grasp. This baseline visits all the views to the right of the initial view, moves up to the next row of views once it reaches the boundary of V ($\phi \in [-90°, 90°]$), continues the search by moving left until it reaches the next boundary, and so on. Once it reaches the top-most row and it cannot move up, it continues the search from the bottom-most row. GridSearch stops when it finds a view from which it can plan a grasp, or after visiting all the discretized views.

Simulation Results

We roll out AVPLUG on 100 scenes, until the policy reaches a termination condition $\mathcal{T}$ and it finds a grasp with a high quality value ($q \geq 0.75$) on the target object, or it fails to find a grasp within a maximal number of steps H and the experiment fails. We set $H = 212$ to account for the total number of grid points in the countertop environment,

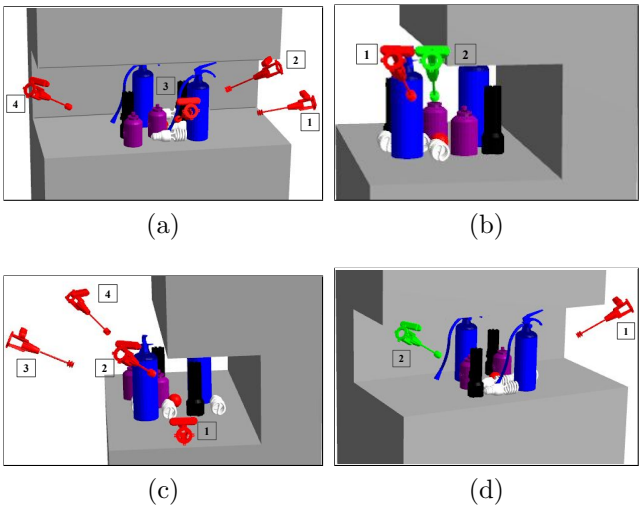
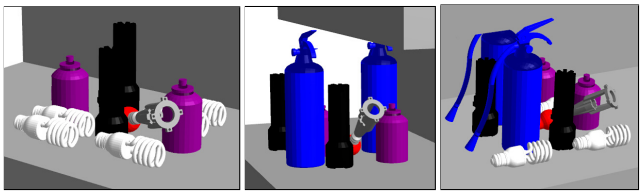


Figure 4.3: **Number of Steps (left) and Distance Traveled (right) failure cases.** AVPLUG chooses approach vectors according to their potential information gain. As a result, in certain cases this can lead to suboptimal behaviour with respect to the metrics we consider: number of steps to success and traveled distance. **Left:** In steps 1 and 3 AVPLUG misses a successful approach vector by few centimeters, and it next moves further away to unsuccessful vectors that reveal more of the scene. **Right:** The initial vector is near a successful approach vector, and the GridSearch baseline (top) finds it in a singel step. In contrast, AVPLUG (bottom) chooses a different vector that is also successful but requires excessive travel.

so that if a successful approach vector exists the baseline will find it. We benchmark the experiments using the following metrics: median and interquartile range for number of steps to success and distance traveled. The number of steps to success influences data acquisition and computation time and the distance traveled by the robot arm may result in a higher travel time and a potential loss in precision. The results are summarized in Table 4.1 and Fig. 4.2. While GridSearch occasionally finds a successful approach vector in several steps, AVPLUG consistently finds an approach vector in up to 10× fewer steps. In Fig. 4.2 we observe that the baseline suffers from high variance, as it is sensitive to the initial view—if it starts near a successful approach vector it can terminate quickly, otherwise it may search the grid exhaustively. In contrast, AVPLUG travels 2 to 3× further than the baseline due to the fact that further vectors often provide higher information gain.

Although AVPLUG's success rate is between 95 % and 100 %, there are certain cases in which it chooses suboptimal actions that lead to a high number of steps to success or a high travel distance. Fig. 4.3 shows two such examples, where AVPLUG may be close to a successful vector but chooses a distant vector that provides more information.

We note that average computation time of AVPLUG for finding an approach vector is



(a) (b) (c)

Figure 4.4: **Unicontact grasping in tight spaces.** AVPLUG can find approach vectors for unicontact grasping even in tight spaces due to the high resolution of the occupancy map.

1.05 s, benchmarked on a server with an Intel Xeon CPU @ 2.20 GHz.

Recovery Mode

After several steps, the support plane voxels may contain only occupied labels despite AVPLUG not yet finding a graspable view. If all subsequent views have zero information gain, AVPLUG will exit without finding a graspable view. To address this case, we add a recovery step that identifies these situations and resets the occupancy map. On a tier 1 counter scene, this recovery step led to an increase in success rate from 69% to 100%, while increasing the median number of steps to success from 1 to 2.

Physical Experiments

We evaluate AVPLUG on physical scenes in the countertop setting using a Fetch mobile robot. We use a planarity-based grasp planner that first samples candidate suction points from a depth image by computing surface normals and pruning those within 10 degrees of the optical axis and then ranks them by a planarity metric that projects a ring with the diameter of the suction cup centered on the suction point and minimizes the distance from the ring to the surface depth [24]. We filter out any grasps that will collide with the scene when approaching and exiting using collision checking between the gripper mesh and the observed point cloud. A grasp is considered successful if it is not in collision, and its associated quality value is above 0.8.

We construct 3 tiers of scenes with matching difficulty to those in simulation. For each tier we evaluate a single scene, and for each scene we choose 5 random starting views. At each view, the baseline is evaluated once and AVPLUG is evaluated 3 times and averaged to account for its inherent stochasticity. We use the elevation angle $\theta \in [45°, 75°]$, azimuth angle $\phi \in [-45°, 45°]$ and radius $r = 0.5\,\text{m}$ for kinematic feasibility. The target object is a red lightbulb as used in simulation and the distractor objects are common objects found around the house—or in this case the lab. We use a HSV color detector to get the binary target segmentation mask. The experimental setup is shown in Figure 4.5. Results in Table 4.2

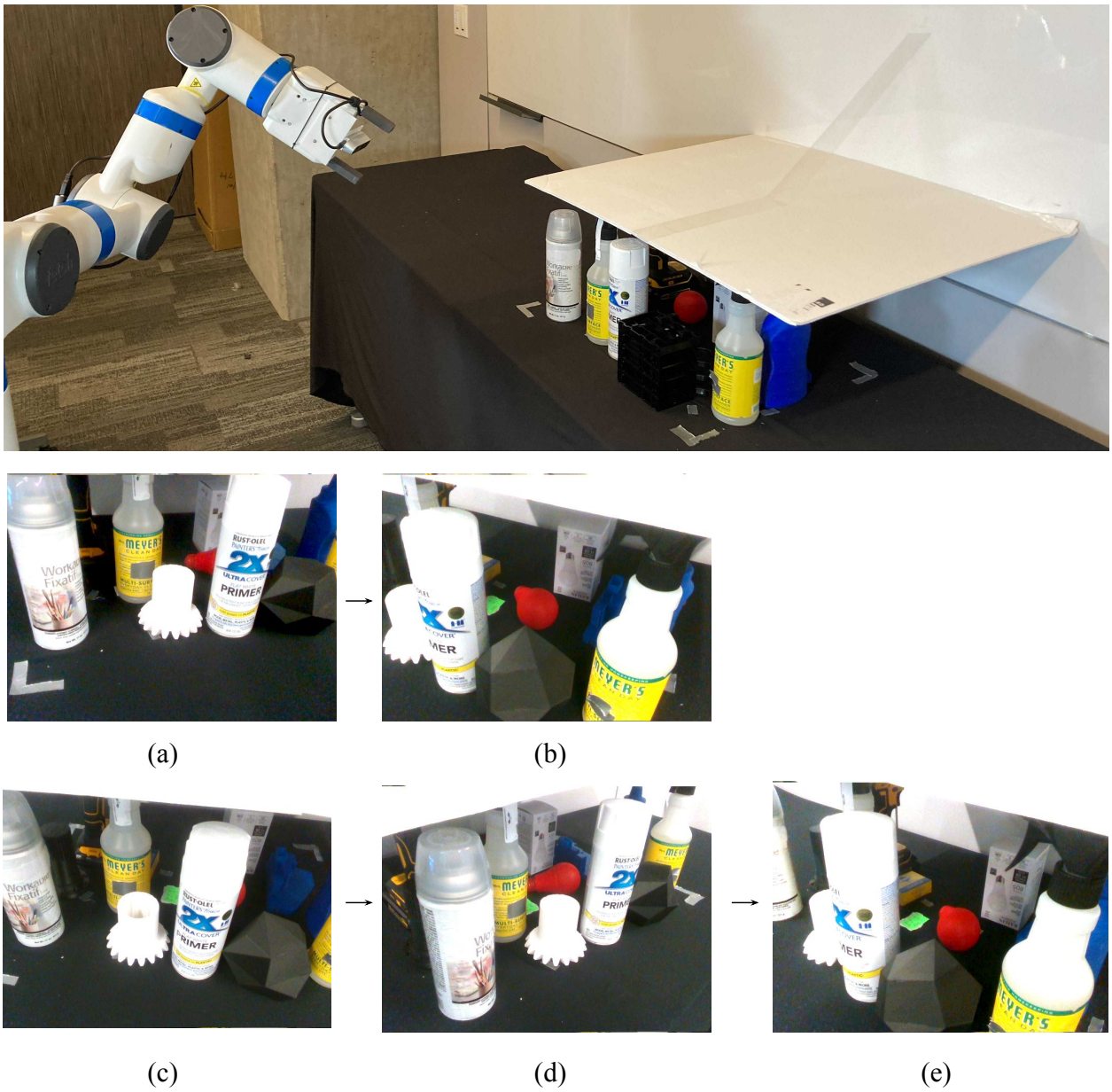


Figure 4.5: **Physical Experiments. Top:** Physical counter setup with a Fetch mobile manipulator for grasping. **Bottom:** Two AVPLUG rollouts. In the first experiment (a-b) the visible part of the target object (in red) is not graspable from the initial position, but is graspable from the next position. In the second experiment (c-e), although a successful grasp is found from the second position, it leads to a collision between the gripper and the environment. AVPLUG then finds a collision free approach vector on the following step.

			# Steps		Distance [m]	
Scene	**Tier**	**Policy**	Median	IQR	Median	IQR
Counter	1	GridSearch	5.0	10.0	**0.6**	0.5
		AVPLUG	**2.0**	**1.0**	0.7	**0.4**
	2	GridSearch	6.5	9.2	0.6	0.5
		AVPLUG	**2.0**	**1.0**	**0.6**	**0.3**
	3	GridSearch	12.0	3.0	1.1	0.6
		AVPLUG	**2.0**	**1.0**	**0.6**	**0.4**

Table 4.2: **Physical Experiments.** Median and interquartile range (IQR) of the number of steps to success and the distance traveled for each policy over 5 rollouts in a physical counter environment. The metrics are reported for successful rollouts. The success rate is 100% for both the *GridSearch* baseline and AVPLUG.

suggest that AVPLUG can consistently find an approach vector in fewer steps (median 2.0) than the baseline (median between 5.0 and 12.0). While the number of search steps taken by baseline policy highly depends on the starting view (with a higher IQR between 3.0 and 10.0), AVPLUG is able to achieve more consistent high performance among random starting views (with a lower IQR of 1.0).

Chapter 5

Conclusions and Future Work

We present AVPLUG, an algorithm that employs an octree-based occupancy map and Minkowski sum computation to find an approach vector for unicontact grasping. AVPLUG takes advantage of the visibility and graspability in suction grasping by servoing a wrist-mounted camera to a view that maximizes information gain to reveal a fully or partially occluded known target object and extract it without the risk of toppling other objects. Experiments with simulation and on a physical robot suggest that AVPLUG can find an approach vector in up to 10× fewer steps compared to a baseline policy and extract objects from tight spaces.

One avenue for future work is in shape completion for reasoning about occluded object geometry for situations where the object cannot be extracted with an upward motion due to overhead obstructions. In these scenarios, reasoning about the rest of the object and jointly optimizing an objective over graspability and extractability will be key.

Another avenue for future work is in pose estimation. In many industrial settings, it is reasonable to assume that we have CAD models for the objects in the scene. We can thus pre-compute grasps on these models using robust analytic methods and then use pose registration to map these grasps onto objects in the scene. Now it is highly likely that some of these grasps will be on occluded regions, and this is where a combination of shape completion and pose registration will prove useful. Even when we do not have CAD models on-hand, we can assume priors about the graspable regions of certain common objects based on semantic cues such as handles of mugs on a home countertop.

One more avenue for future work is towards leveraging the representational power of a learning-based occupancy map. Presumably a deep network can lift observations to a higher-dimensional space best suited for the task of graspability and efficiently aggregate them in this space. Such a representation would also be fast to query thanks to recent advances in hardware acceleration for neural network inference. Architectures such as PointNet++ could prove to be a good starting point as they have shown to empirically lend themselves well to related 3D spatial reasoning tasks.

We hope that in addition to further improvements to AVPLUG itself, AVPLUG will see adoption in many other downstream applications in complex cluttered environments

where we wish to not only plan grasps rapidly and efficiently but also minimize or avoid entirely disruption to the environment itself. Some example applications where this degree of precision is necessary are:

1. **Home robotics**: Robots must be able to fetch objects on cluttered countertops or from densely packed cupboards without toppling nearby objects.
2. **Medical assistance**: Medical robots inherently operate in sterile environments with expensive and fragile equipment that must be carefully handled.
3. **Industrial manufacturing**: Robots must be able to pick up a single part from a tight arrangement of many identical parts without knocking others over.

Chapter 6

Final thoughts

In completing this thesis, I learned the importance of simulation in robotics research in enabling fast iteration as it obviates the need for slow physical robot motion and can be massively parallelized. However, I also learned that it is by no means a substitute for physical experiments as there are many factors that cannot tractably be simulated such as complex physical phenomena.

I also learned the relevance of classical methods even to this day. We initially attempted to learn an end-to-end solution but were faced with poor performance and more importantly a lack of interpretability. We then pivoted to much more transparent occupancy maps and ray tracing which have been around for decades. Although more classical methods like these are a bit lackluster when compared to SOTA learning-based approaches, they are far more interpretable and can thus be finely architected for the task at hand.

Another thing I learned is the importance of rigorous benchmarking and reproducability that Professor Goldberg constantly strives for in all of his projects. Although at first it can seem excessive, I have since learned that it is instrumental to producing high-quality research that will stand the test of time and gives others the confidence to build upon both your current and future projects. I have come to realize that this is one of the defining characteristics that sets the best researchers apart from the rest of the crowd as it is no trivial task to try and break work you have already heavily invested so much time and effort into. This ranges the gamut from crafting strong baselines to compare against that go past simple strawman approaches to both properly open-sourcing code and datasets and maintaining them. I have learned from first-hand experience that the latter is not easy as it involves heavily interacting with the larger research community and becoming aware of the limitations of your research as others try to use it.

One more thing I learned not only through the masters but even from my earlier research is the importance of being able to clearly convey your ideas not only in the papers you write but also in the presentations you give. You can be the best researcher in the world, but no-one will know it and it won't matter if you cannot clearly present your research to your colleagues in a clear and digestible manner. Professor Goldberg taught me this through his extensive draft review process with many back-and-forth revisions that can seem daunting

at first, but in the end produced a very polished result and made me feel confident that I was really submitting my best work.

I am grateful for everything I have learned thus far and the people who have taught me it. Now I hope to both apply it out in the real world and impart this knowledge upon others!

Bibliography

[1] John Aloimonos. "Purposive and qualitative active vision". In: *[1990] Proceedings. 10th International Conference on Pattern Recognition*. Vol. 1. IEEE. 1990, pp. 346–360.

[2] Yiannis Aloimonos. *Active perception*. Psychology Press, 2013.

[3] Ruzena Bajcsy. "Active perception". In: *Proceedings of the IEEE* 76.8 (1988), pp. 966–1005.

[4] Ruzena Bajcsy, Yiannis Aloimonos, and John K Tsotsos. "Revisiting active perception". In: *Autonomous Robots* 42.2 (2018), pp. 177–196.

[5] Tara Boroushaki et al. "Robotic Grasping of Fully-Occluded Objects using RF Perception". In: *arXiv preprint arXiv:2012.15436* (2020).

[6] Berk Calli et al. "Benchmarking in manipulation research: The YCB object and model set and benchmarking protocols". In: *arXiv preprint arXiv:1502.03143* (2015).

[7] Henry Carrillo, Ian Reid, and José A Castellanos. "On the comparison of uncertainty criteria for active SLAM". In: *2012 IEEE International Conference on Robotics and Automation*. IEEE. 2012, pp. 2080–2087.

[8] Michael Danielczuk et al. "Mechanical search: Multi-step retrieval of a target object occluded by clutter". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 1614–1621.

[9] Michael Danielczuk et al. "Segmenting unknown 3d objects from real depth images using mask r-cnn trained on synthetic data". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 7283–7290.

[10] Michael Danielczuk et al. "X-ray: Mechanical search for an occluded object by minimizing support of learned occupancy distributions". In: *arXiv preprint arXiv:2004.09039* (2020).

[11] Enrique Dunn and Jan-Michael Frahm. "Next Best View Planning for Active Model Improvement." In: *BMVC*. 2009, pp. 1–11.

[12] Megha Gupta et al. "Interactive environment exploration in clutter". In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE. 2013, pp. 5265–5272.

[13] Dan Halperin. "Robust geometric computing in motion". In: *The International Journal of Robotics Research* 21.3 (2002), pp. 219–232.

[14] Armin Hornung et al. "OctoMap: An efficient probabilistic 3D mapping framework based on octrees". In: *Autonomous robots* 34.3 (2013), pp. 189–206.

[15] Huang Huang et al. "Mechanical Search on Shelves using Lateral Access X-RAY". In: *arXiv preprint arXiv:2011.11696* (2020).

[16] Jeffrey Ichnowski et al. "Deep learning can accelerate grasp-optimized motion planning". In: *Science Robotics* 5.48 (2020).

[17] Jeffrey Ichnowski et al. "GOMP: Grasp-optimized motion planning for bin picking". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 5270–5277.

[18] Gregory Kahn et al. "Active exploration using trajectory optimization for robotic grasping in the presence of occlusions". In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 4783–4790.

[19] Jue Kun Li, David Hsu, and Wee Sun Lee. "Act to see and see to act: POMDP planning for objects search in clutter". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2016, pp. 5701–5707.

[20] Tomas Lozano-Perez, Matthew T Mason, and Russell H Taylor. "Automatic synthesis of fine-motion strategies for robots". In: *The International Journal of Robotics Research* 3.1 (1984), pp. 3–24.

[21] Jeffrey Mahler et al. "Dex-net 1.0: A cloud-based network of 3d objects for robust grasp planning using a multi-armed bandit model with correlated rewards". In: *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2016, pp. 1957–1964.

[22] Jeffrey Mahler et al. "Dex-net 2.0: Deep learning to plan robust grasps with synthetic point clouds and analytic grasp metrics". In: *arXiv preprint arXiv:1703.09312* (2017).

[23] Jeffrey Mahler et al. "Dex-Net 3.0: computing robust robot vacuum suction grasp targets in point clouds using a new analytic model and deep learning". In: *arXiv preprint arXiv:1709.06670* (2017).

[24] Jeffrey Mahler et al. "Dex-net 3.0: Computing robust vacuum suction grasp targets in point clouds using a new analytic model and deep learning". In: *2018 IEEE International Conference on robotics and automation (ICRA)*. IEEE. 2018, pp. 5620–5627.

[25] Jeffrey Mahler et al. "Learning ambidextrous robot grasping policies". In: *Science Robotics* 4.26 (2019).

[26] Douglas Morrison, Peter Corke, and Jürgen Leitner. "Multi-view picking: Next-best-view reaching for improved grasping in clutter". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 8762–8768.

[27] Arsalan Mousavian, Clemens Eppner, and Dieter Fox. "6-dof graspnet: Variational grasp generation for object manipulation". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019, pp. 2901–2910.

[28] Adithyavairavan Murali et al. "6-dof grasping for target-driven object manipulation in clutter". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 6232–6238.

[29] Tonci Novkovic et al. "Object finding in cluttered scenes using interactive perception". In: *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2020, pp. 8338–8344.

[30] Richard Pito. "A solution to the next best view problem for automated surface acquisition". In: *IEEE Transactions on pattern analysis and machine intelligence* 21.10 (1999), pp. 1016–1030.

[31] Andrew Price, Linyi Jin, and Dmitry Berenson. "Inferring occluded geometry improves performance when retrieving an object from dense clutter". In: *arXiv preprint arXiv:1907.08770* (2019).

[32] Sumantra Dutta Roy, Santanu Chaudhury, and Subhashis Banerjee. "Active recognition through next view planning: a survey". In: *Pattern Recognition* 37.3 (2004), pp. 429–446.

[33] João Santos et al. "Autonomous Scene Exploration for Robotics: A Conditional Random View-Sampling and Evaluation Using a Voxel-Sorting Mechanism for Efficient Ray Casting". In: *Sensors* 20.15 (2020), p. 4331.

[34] Vishal Satish. *GQ-CNN*. `https://github.com/BerkeleyAutomation/gqcnn`. 2018.

[35] Vishal Satish, Jeffrey Mahler, and Ken Goldberg. "On-policy dataset synthesis for learning robot grasping policies using fully convolutional deep networks". In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 1357–1364.

[36] Pierre Soille. "Erosion and dilation". In: *Morphological Image Analysis*. Springer, 2004, pp. 63–103.

[37] *Thingiverse online 3D object database.* Thingiverse. URL: `https://www.thingiverse.com/` (visited on 03/09/2021).

[38] Kentaro Wada. *Octomap-python*. `https://github.com/wkentaro/octomap-python`. 2013.

[39] Yuchen Xiao et al. "Online planning for target object search in clutter under partial observability". In: *2019 International Conference on Robotics and Automation (ICRA)*. IEEE. 2019, pp. 8241–8247.

5th_Year_MS_Thesis_Vishal_Satish_final

Final Audit Report 2021-05-14

Created:	2021-05-14
By:	Vishal Satish (vsatish@berkeley.edu)
Status:	Signed
Transaction ID:	CBJCHBCAABAALgUCGgPm6TbA4AD5IhcSjB-yupdm8x8f

"5th_Year_MS_Thesis_Vishal_Satish_final" History

Document created by Vishal Satish (vsatish@berkeley.edu)
2021-05-14 - 4:50:22 PM GMT- IP address: 99.73.36.50

Document emailed to Ken Goldberg (goldberg@berkeley.edu) for signature
2021-05-14 - 4:56:47 PM GMT

Email viewed by Ken Goldberg (goldberg@berkeley.edu)
2021-05-14 - 8:21:22 PM GMT- IP address: 66.249.84.69

Document e-signed by Ken Goldberg (goldberg@berkeley.edu)
Signature Date: 2021-05-14 - 8:21:38 PM GMT - Time Source: server- IP address: 67.169.14.106

Document emailed to Angjoo Kanazawa (kanazawa@berkeley.edu) for signature
2021-05-14 - 8:21:40 PM GMT

Email viewed by Angjoo Kanazawa (kanazawa@berkeley.edu)
2021-05-14 - 10:37:53 PM GMT- IP address: 66.249.84.215

Document e-signed by Angjoo Kanazawa (kanazawa@berkeley.edu)
Signature Date: 2021-05-14 - 10:38:11 PM GMT - Time Source: server- IP address: 192.31.105.191

Agreement completed.
2021-05-14 - 10:38:11 PM GMT