


2019

## Autonomous Robotic Grasping in Unstructured Environments

Amirhossein Jabalameli  
*University of Central Florida*

 Part of the [Electrical and Computer Engineering Commons](#)  
Find similar works at: <https://stars.library.ucf.edu/etd>  
University of Central Florida Libraries <http://library.ucf.edu>

---

### STARS Citation

Jabalameli, Amirhossein, "Autonomous Robotic Grasping in Unstructured Environments" (2019).  
*Electronic Theses and Dissertations*. 6691.  
<https://stars.library.ucf.edu/etd/6691>

This Doctoral  
Dissertation (Open  
Access) is brought  
to you for free and  
open access by  
STARS. It has been  
accepted for  
inclusion in  
Electronic Theses  
and Dissertations by  
an authorized  
administrator of  
STARS. For more  
information, please  
contact  
[lee.dotson@ucf.edu](mailto:lee.dotson@ucf.edu).



# AUTONOMOUS ROBOTIC GRASPING IN UNSTRUCTURED ENVIRONMENTS

by

AMIRHOSSEIN JABALAMELI  
MS, University of Central Florida, 2015  
B.S. Isfahan University of Technology, 2013

A dissertation submitted in partial fulfilment of the requirements  
for the degree of Doctor of Philosophy  
in the Department of Electrical and Computer Engineering  
in the College of Engineering and Computer Science  
at the University of Central Florida  
Orlando, Florida

Spring Term  
2019

Major Professor: Aman Behal

© 2019 Amirhossein Jabalameli

## ABSTRACT

A crucial problem in robotics is interacting with known or novel objects in unstructured environments. While the convergence of a multitude of research advances is required to address this problem, our goal is to describe a framework that employs the robot's visual perception to identify and execute an appropriate grasp to pick and place novel objects. Analytical approaches explore for solutions through kinematic and dynamic formulations. On the other hand, data-driven methods retrieve grasps according to their prior knowledge of either the target object, human experience, or through information obtained from acquired data. In this dissertation, we propose a framework based on the supporting principle that potential contacting regions for a stable grasp can be found by searching for (i) sharp discontinuities and (ii) regions of locally maximal principal curvature in the depth map. In addition to suggestions from empirical evidence, we discuss this principle by applying the concept of force-closure and wrench convexes. The key point is that no prior knowledge of objects is utilized in the grasp planning process; however, the obtained results show that the approach is capable to deal successfully with objects of different shapes and sizes. We believe that the proposed work is novel because the description of the visible portion of objects by the aforementioned edges appearing in the depth map facilitates the process of grasp set-point extraction in the same way as image processing methods with the focus on small-size 2D image areas rather than clustering and analyzing huge sets of 3D point-cloud coordinates. In fact, this approach dismisses reconstruction of objects. These features result in low computational costs and make it possible to run the proposed algorithm in real-time. Finally, the performance of the approach is successfully validated by applying it to the scenes with both single and multiple objects, in both simulation and real-world experiment setups.

To my parents

## ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Dr. Aman Behal, for his insightful guidance, that was so helpful to the work that I have conducted at the University of Central Florida. He was incessantly encouraging, supportive, and ready to provide practical help and advice. I am also grateful to my committee members, Dr. Ladislau Boloni, Dr. Michael Haralambous, Dr. Yaser Fallah, and Dr. Yunjun Xu for their valuable time and comments. I also like to acknowledge the contribution of Yannick Roberts in implementation of this work in the Robotic Operating System.

Thanks to my friends in Orlando, for all the time we spent together, all the adventures we had, and much more. I would also like to thank my friends, Soroush, Amir and Amirali for being there, when life overflowed my plate. Because of these friends, I have never felt alone.

At the very last, I recognize the vital support of the most important people, that made all this possible: My sister-in-law and my brother, Negin and Ali, for their unconditioned kindness through these years. My mom, Farnaz, for her love, patience and self sacrifice. And my dad, Bahram, for his support, his encouragement, and for instilling in me a belief that if I am tired, I learn to rest, not to quit.

## TABLE OF CONTENTS

LIST OF FIGURES . . . . .	ix
LIST OF TABLES . . . . .	xii
CHAPTER 1: INTRODUCTION . . . . .	1
CHAPTER 2: RELATED WORK . . . . .	4
Introduction . . . . .	4
Data-driven Methods . . . . .	6
CHAPTER 3: BACKGROUND . . . . .	8
Contact Model and Force Closure Grasp . . . . .	8
Grasp Model . . . . .	12
CHAPTER 4: EDGE LEVEL GRASP PLANNING . . . . .	14
Introduction . . . . .	14
Problem Statement . . . . .	14
Object Depth Representation . . . . .	14
2D Grasp Identification . . . . .	20

Algorithm Overview . . . . .	24
Grasp Reliability . . . . .	26
<b>CHAPTER 5: EDGE LEVEL GRASPING IMPLEMENTATION AND RESULTS . . . . .</b>	<b>29</b>
Implementation Procedure . . . . .	29
Introduction . . . . .	29
Edge Detection and Line Segmentation . . . . .	30
Edge Feature Extraction and Pair Formation . . . . .	32
3D Grasp Specification . . . . .	37
Practical Issues . . . . .	38
Experiments and Discussion . . . . .	42
Introduction . . . . .	42
Simulation-Based Results . . . . .	42
Real World Experiments . . . . .	45
Grasping Policy . . . . .	47
Grasping Experiments . . . . .	47
Discussion . . . . .	53
<b>CHAPTER 6: OBJECT LEVEL GRASPING . . . . .</b>	<b>55</b>



Introduction . . . . .	55
Problem Statement . . . . .	55
Model Description . . . . .	56
Implementation . . . . .	58
CHAPTER 7: DEVELOPED ROS PACKAGE . . . . .	62
Introduction . . . . .	62
Description . . . . .	62
Instruction . . . . .	62
Application Dependencies . . . . .	62
Launching GriPIt . . . . .	63
CHAPTER 8: CONCLUSIONS . . . . .	68
APPENDIX A: IMPLEMENTED CODE DEPENDENCY . . . . .	70
APPENDIX B: PUBLICATIONS . . . . .	82
LIST OF REFERENCES . . . . .	84

## LIST OF FIGURES

Figure 2.1: Impacting Grasp Aspects [36] . . . . .	5
Figure 3.1: Planar contacts: a) Frictionless point contact b) Point contact with friction c) Soft finger contact d) Edge contact . . . . .	9
Figure 3.2: Force closure geometric interpretation for two opposing finger gripper and triangular end effector. (a) and (b) show feasible force closures grasps, while (c) and (d) illustrate impossible force closure grasps. . . . .	11
Figure 3.3: Grasp representation for a planar shape . . . . .	13
Figure 4.1: (a) RGB image of the scene, $I_c$ (b) Color map of the raw depth map. (c) Color map of computed gradient direction image, $I_\theta$ . . . . .	16
Figure 4.2: Measured depth values for a synthetic scene. . . . .	17
Figure 4.3: Geometric interpretation of a surface segment for a cube and a cylinder. . . .	19
Figure 4.4: 2D Surface segments and depth edges are marked for an isolated object. . . .	20
Figure 4.5: Overlapping test. a) shows intersection of orthogonal projection for three edges b) indicates overlapped region and edge contact regions. . . . .	23
Figure 4.6: Shapes with similar planar grasps despite different 3D friction cones. . . . .	25

Figure 5.1: Applied edge detection on an acquired depth map. (a) RGB image of the scene, $I_c$ (b) Color map of the raw depth map. White pixels imply to non-returned values from the sensor (depth shadows) (c) Color map of the processed depth map, $I_d$ (d) Color map of computed gradient direction image, $I_\theta$ (e) Detected edges before applying the morphological operations (f) Detected edges after the morphological process, $I_{DE}$ . . . . .	31
Figure 5.2: Line segmentation step is applied to a synthetic depth map. (a) detected edge pixels are marked) (b) edges are broken into line segment(s) . . . . .	32
Figure 5.3: a) Examples of parallelogram masks the sides of 2D shape ABCDE. b) Projection area and edge contact regions for a pair of edges. . . . .	36
Figure 5.4: 2D object representation ambiguity. . . . .	40
Figure 5.5: Binary images indicate extracted edges from the corresponding depth images.	41
Figure 5.6: Utilized images for obtaining simulation results. . . . .	43
Figure 5.7: Reference and detected edges for scene No.4 in the simulation-based results. Note that assigned colours are only used to distinguish the line segments visually. (a) reference graspable edges: each edge is manually marked by a line segment (b) detected graspable edges: marked points are detected by algorithm as graspable edges (c) detected line segments: each detected edge is represented by a number of line segments. . . . .	44
Figure 5.8: Utilized equipment for the real world experiments . . . . .	46
Figure 5.9: The entire set of objects used through real world experiments (16 objects). . .	48

Figure 5.10 Multi-object experiments scenes including variety of objects. (a) Scene No.2 (b) Scene No.3 (c) Scene No.6 . . . . .	51
Figure 5.11A Sequence of snapshots from the robot arm while approaching to grasp the objects in a cluttered scene. . . . .	52
Figure 6.1: A sample graph representation for based on the adjacency feature. Each node in the graph indicates an object corresponded to overlay color. . . . .	57
Figure 6.2: From a binary image to object segments by only depth information. . . . .	60
Figure 7.1: Parameters tuning panel . . . . .	64
Figure 7.2: A screenshot of the developed image. . . . .	65
Figure 7.3: Obtained graspable pairs by the developed algorithm. . . . .	66
Figure 7.4: Illustration of obtained grasp pose and orientation demonstrated on the 3- dimensional point cloud. . . . .	67

## LIST OF TABLES

- Table 5.1: Simulation section results. Columns describe the number of (G)raspable and (D)etected objects, surface segments and edge for 8 different scene. The last row indicates average accuracy rates of detection in object level, surface-level and edge-level. . . . . 44
- Table 5.2: Single object experiment results. Four attempts for each object are performed. "L" indicates the large size and "S" indicates small size objects. . . . 49
- Table 5.3: Multi object experiment results. The success rate implies number of objects grasped successfully out of total number of objects in the scene. . . . . 51

## CHAPTER 1: INTRODUCTION

A crucial problem in robotics is interacting with known or novel objects in unstructured environments. Among several emerging applications, assistive robotic manipulators seek approaches to assist users to perform a desired object motion in a partial or fully autonomous system. While the convergence of a multitude of research advances is required to address this problem, our goal is to describe a method that employs the robot's visual perception to identify and execute an appropriate grasp to pick and place novel objects.

In this dissertation, we introduce an approach to obtain stable grasps using partial depth information for an object of interest. The expected outcome is an executable end-effector configuration to grasp the object in an occluded scene. We propose a framework based on the supporting principle that potential contacting regions for a stable grasp can be found by searching for (i) sharp discontinuities and (ii) regions of locally maximal principal curvature in the depth map. In addition to suggestions from empirical evidence, we discuss this principle by applying the concept of wrench convexes. The framework consists of two phases. First, we localize candidate regions, and then we evaluate local geometric features of candidate regions toward satisfying desired grasp features. The key point is that no prior knowledge of objects is utilized in the grasp planning process; however, the obtained results show that the approach is capable to deal successfully with objects of different shapes and sizes. We believe that the proposed work is novel and interesting because the description of the visible portion of objects by the aforementioned *edges* appearing in the depth map facilitates the process of grasp set-point extraction in the same way as image processing methods with the focus on small-size 2D image areas rather than clustering and analyzing huge sets of 3D point-cloud coordinates. In fact, this approach dismisses reconstruction of objects. These features result in low computational costs and make it possible to run the algorithm in near real-time. We also see this approach as a useful solution to obtain grasp configuration according to

the given task and user constraints as opposed to the majority of data-driven methods that address this problem by locating the objects and lifting them from the top. Finally, the reliance on only a single-view depth map without the use of color information distinguishes our approach from other candidates in relevant applications. The biggest challenge in the work arises due to uncertainties arising from pixel-wise characteristics; for this matter, relying on larger pixel sets of interest has helped to make the process more robust.

In summary, the main contributions of our work are:

- Addressing the problem of grasping novel objects in unstructured environments,
- Development of a novel algorithm to construct force-closure grasps based on a single view depth map,
- Formulation of a framework to represent object boundaries and potential contact regions by using local depth descriptors,
- Development of a partially autonomous system to execute the pick and place task by using Baxter arm manipulator
- Development of a real time Robotic Operating System package according to the proposed approach.

This dissertation is organized as follows. In Chapter 2, we will categorize grasp planning approaches into analytical and data-driven methods and review the corresponding literature. Grasping preliminaries are presented in Chapter 3. The grasp planning problem is stated in Chapter 4 and the proposed approach is presented in Section 4. Specifically, in Section 4, we define the object model in the 2D image according to geometry and then introduce the employed grasp model in Section 3. Next, in Section 4, we propose an approach to find reliable contact regions for the force

closure grasp on the targeted object. Details of algorithm implementation are provided in Chapter 5. In this chapter, we also validate our proposed grasp planning approach by considering different scenarios for grasping objects, using a Kinect One sensor and a Baxter robot as a 7-DOF arm manipulator followed by a discussion of the obtained results. In chapter 6, we provide a framework to extend the grasp planning procedure to include an object level recognition. Chapter 7 is dedicated to instruction of developed Robot Operating System application. Finally, Chapter 8 concludes this study and draws the future directions.



## CHAPTER 2: RELATED WORK

### Introduction

Finding a grasp configuration relevant to a specific task has been an active topic in robotics for the past three decades. There exist multiple factors affect how a grasp can be characterized and evaluated. Figure (2.1) demonstrates a general picture of the most important ones. In a recent article by Bohg *et al.* [36], grasp synthesis algorithms are categorized into two main groups, *viz.*, analytical and data-driven. Analytical approaches explore for solutions through kinematic and dynamic formulations [25]. Object and/or robotic hand models are used in [9][8][4], [6], and [2] to develop grasping criteria such as force-closure, stability, and dexterity and also to evaluate if a grasp is satisfying them. The difficulty of modeling a task, high computational costs, and assumptions of the availability of geometric or physical models for the robot are the challenges that analytical approaches deal with in real-world experiments. Furthermore, researchers conducting experiments have inferred that classical metrics are not sufficient to tackle with grasping problems in real-world scenarios despite their efficiency in simulation environments [19][24]. In the following section, data-driven method are discussed in the details.

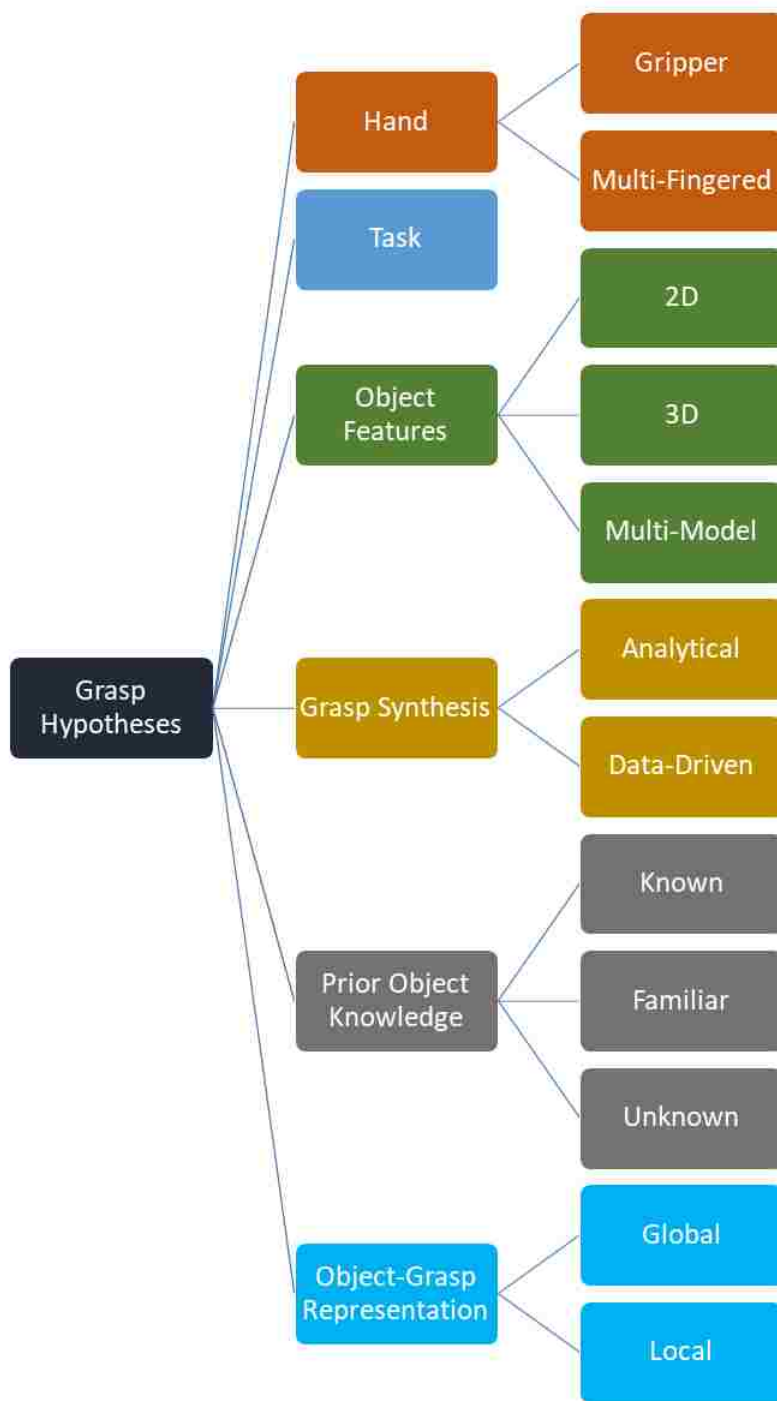


Figure 2.1: Impacting Grasp Aspects [36]

## Data-driven Methods

Data-driven methods retrieve grasps according to their prior knowledge of either the target object, human experience, or through information obtained from acquired data. In line with this definition, Bohg *et al.* [36] classified data-driven approaches based on the encountered object being considered known, familiar, or unknown to the method. Thus, the main issues relate to how the query object is recognized and then compared with or evaluated by the algorithm's existing knowledge. As an example, [11][13][22] assume that all the objects can be modeled by a set of shape primitives such as boxes, cylinders and cones. During the off-line phase, they assign a desired grasp for each shape while, during the on-line phase, these approaches are only supposed to match sensed objects to one of the shape primitives and pick the corresponding grasp. In [16], a probabilistic framework is exploited to estimate the pose of a known object in an unknown scene. Ciocarlie *et al.* [17] introduced the human operator in the grasp control loop to define a hand postures subspace known as eigen-grasp. This method finds appropriate grasp corresponded to a given task, using the obtained low-dimensional subspace. A group of methods considers the encountered object as a familiar object and employs 2D and/or 3D object features to measure the similarities in shape or texture properties [36]. In [14], a logistic regression model is trained based on labeled data sets and then grasp points for the query object are detected based on the extracted feature vector from a 2D image. The authors in [23] present a model that maps the grasp pose to a success probability; the robot learns the probabilistic model through a set of grasp and drop actions.

The last group of methods, in data-driven approaches, introduce and examine features and heuristics which directly map the acquired data to a set of candidate grasps [36]. They assume sensory data provide either full or partial information of the scene. [39] takes the point cloud and clusters it to a background and an object, then addresses a grasp based on the principal axis of the object. The authors in [37] propose an approach that takes 3D point cloud and hand geometric parameters

as the input, then search for grasp configurations within a lower dimensional space satisfying defined geometric necessary conditions. Jain *et al.* [38] analyze the surface of every observed point cloud cluster and automatically fit spherical, cylindrical or box-like shape primitives to them. The method utilizes a pre-defined strategy to grasp each shape primitive. The algorithms in [31]-[33] build a virtual elastic surface by moving the camera around the object while computing the grasp configuration in an iterative process. Similarly, [45] approaches the grasping problem through a surface-based exploration. Another approach to grasp planning problem can be performed through object segmentation algorithms to find surface patches [45][34].

In general, knowledge level of the object, accessibility to partial or full shape information of the existing objects in a scene and type of the employed features are the main aspects that characterize data-driven methods. One of the main challenges that most of the data-driven grasping approaches deal with is uncertainties in the measured data which causes failure in real-world experiments. Thus, increasing the robustness of a grasp against uncertainties appearing in the sensed data, or during the execution phase, is the aim for a group of approaches. While [18] utilizes tactile feedback to adjust the object position deviation from the initial expectation, [30] employs visual servoing techniques to facilitate the grasping execution. Another challenge for data-driven approaches is data preparation and specifically background elimination. This matter forces some of the methods to make simplifying assumptions about an object's situation, *e.g.*, [39] is only validated for objects standing on a planar surface. Finding a feasible grasp configuration subject to the given task and user constraints is required for a group of applications. As discussed by [41]-[44], suggesting desired grasp configurations, in assistive human-robot interaction, results in increasing the users' engagement and easing the manipulator trajectory adaptation.

## CHAPTER 3: BACKGROUND

### Contact Model and Force Closure Grasp

Choosing a stable grasp is one of the key components of a given object manipulation task. According to the adopted terminology from [8], a stable grasp is defined as a grasp having force closure on the object. Force closure needs the grasp to be disturbance resistance meaning any possible motion of the object is resisted by the contact forces [35]. Thus, determining possible range of force directions and contact locations for robotic fingers is an important part of grasp planning [8]. By considering force closure as a necessary condition, [2] discussed the problem of synthesizing *planar grasps*. In the planar grasp, all the applied forces will lie in the plane of the object and shape of the object will be the only input through the process. Any contact between fingertips and the object can be described as a convex sum of three primitive contacts.

**Definition 1** *A wrench convex represents the range of force directions that can be exerted on the object and is determined depending on the contact type and the existing friction coefficient.*

Figure (3.1) shows the primitive contacts and their wrench convexes in 2D. Wrench convexes are illustrated by two arrows forming the angular sector. In the frictionless point contact, the finger can only apply force in the direction of normal. However, through a point contact with friction, the finger can apply any forces pointing into the wrench convex. Soft finger contact is capable of exerting pure torques in addition to pure forces inside the wrench convex.

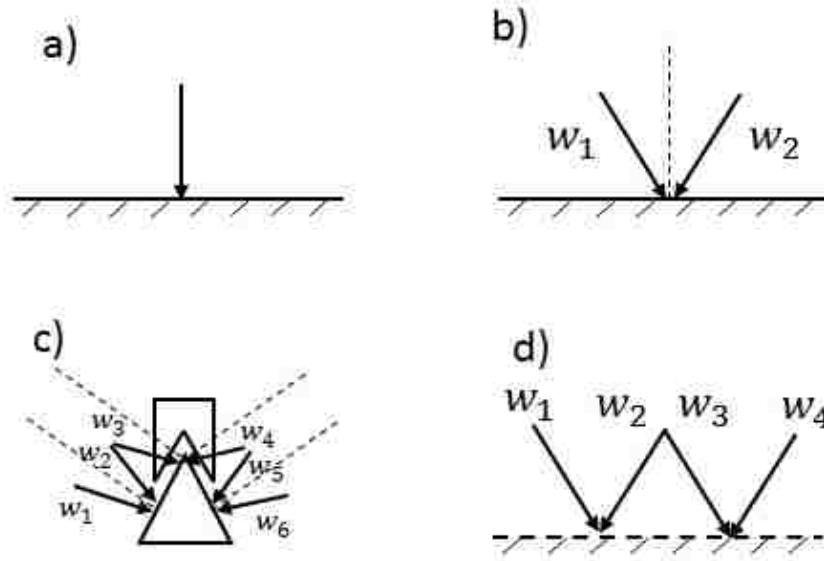


Figure 3.1: Planar contacts: a) Frictionless point contact b) Point contact with friction c) Soft finger contact d) Edge contact

**Remark 1** Any force distribution along an edge contact can be cast to a unique force at some point inside the segment. This force is described by the positive combination of two wrench convexes at the two ends of the contact edge. It is also common in this subject to refer to the wrench convex as friction cone. To resist translation and rotation motions for a 2D object, force closure is simplified to maintain force-direction closure and torque-closure [2].

**Theorem 1** (Nguyen I) A set of planar wrenches  $W$  can generate force in any direction if and only if there exists a set of three wrenches  $(w_1, w_2, w_3)$  whose respective force directions  $f_1, f_2, f_3$  satisfy: i) two of the three directions  $f_1, f_2, f_3$  are independent. ii) a strictly positive combinations of the three directions are zero:  $\sum_{i=1}^3 \alpha_i f_i = 0$

**Theorem 2** (Nguyen II) *A set of planar forces  $W$  can generate clockwise and counter-clockwise torques if and only if there exists a set of four forces  $(w_1, w_2, w_3, w_4)$  such that three of the four forces have lines of action that do not intersect at a common point or at infinity. Let  $p_{12}$  (resp.  $p_{34}$ ) be the points where the lines of action of  $w_1$  and  $w_2$  (resp.  $w_3$  and  $w_4$ ) intersect; there exist positive values of  $\alpha_i$  such that  $p_{34} - p_{12} = \pm(\alpha_1 f_1 + \alpha_2 f_2) = \mp(\alpha_3 f_3 + \alpha_4 f_4)$ .*

Basically, force-direction closure checks if the contact forces (friction cones) span all the directions in the plane. Torque-closure tests if the combination of all applied forces produces pure torques. According to Theorem I and II, existence of four wrenches with three being independent is necessary for a force closure grasp in a plane. Assuming the contacts are with friction, each point contact provides two wrenches. Thus, a planar force closure grasp is possible with at least two contacts with friction. As stated by [4] and [2], the conditions for forming a planar force closure grasp with two and three points are interpreted in geometric sense as below and illustrated in Figure (3.2):

- Two opposing fingers: A grasp by two point contacts,  $p_1$  and  $p_2$  with friction is in force closure if and only if the segment  $p_1 - p_2$  points out of and into two friction cones respectively at  $p_1$  and  $p_2$ . Mathematically speaking, assuming  $\varphi_1$  and  $\varphi_2$  are angular sectors of friction cones at  $e_1$  and  $e_2$ , term  $\arg(p_1 - p_2) \in \pm(\varphi_1 \cap -\varphi_2)$  is the necessary and sufficient condition for two point contacts with friction.
- Triangular grasp: A grasp by three point contacts,  $p_1, p_2,$  and  $p_3$  with friction is in force closure if there exists a point,  $p_f$  (force focus point) such that for each  $p_i$ , the segment  $p_f - p_i$  points out of the friction cone of the  $i^{th}$  contact. Let  $k_i$  be the unit vector of segment  $p_f - p_i$  which points out the edge; strictly positive combinations of the three directions are zero:  $\sum_{i=1}^3 \alpha_i k_i = 0$ .

An appropriate object representation and analysis on the shape of objects based on the accessed

geometry information are the steps toward finding contact regions for a stable grasp. In Section IV, we relate planar object representation to a proper grasp configuration in order to obtain the objects' possible grasps.

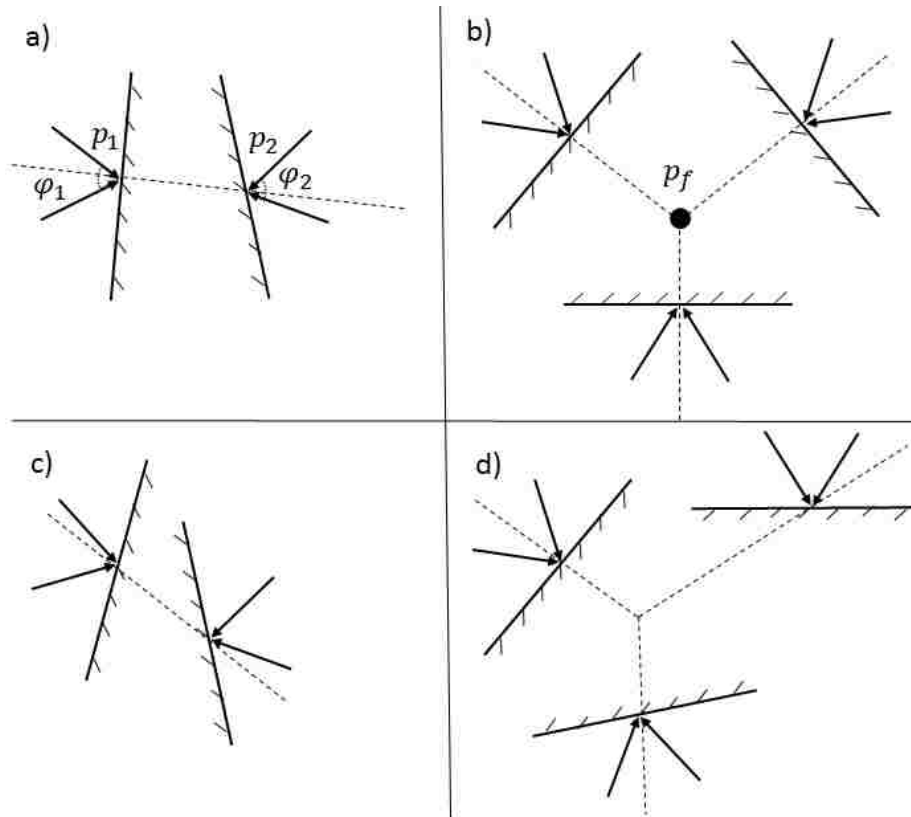


Figure 3.2: Force closure geometric interpretation for two opposing finger gripper and triangular end effector. (a) and (b) show feasible force closures grasps, while (c) and (d) illustrate impossible force closure grasps.



## Grasp Model

Generally, a precision grasp is indicated by end-effector and fingertips poses with respect to a fixed coordinate system. According to terminology adopted from [46], referring to an end-effector  $E$  with  $n_E$  fingers and  $n_\theta$  joints with the fingertips contacting an object's surface, a grasp configuration,  $G$ , is addressed as follows:

$$G = (p_G, \theta_G, C_G)$$

where  $p_G$  is the end-effector pose (position and orientation) relative to the object,  $\theta_G = (\theta_1, \theta_2, \dots, \theta_{n_\theta})$  indicates the end-effector's joint configuration, and  $C_G = \{c_i \in S(O)\}_{i=1}^{n_E}$  determines  $n_E$  point contacts on the object's surface. The contact locations set on the end-effector's fingers is  $C_E = \{\bar{c}_i \in S(E)\}_{i=1}^{n_E}$  and is obtained by a forward kinematics derived from the end-effector pose  $p_G$ .

Throughout this study, we make an assumption regarding the end-effector during the interaction with the object. Each fingertip applies a force in the direction of its normal and the exerted forces by all fingertips lie on a same plane. We refer to this plane and its normal direction, respectively, as end-effector's approach plane,  $\rho_G$ , and approach direction,  $\vec{V}_G$ . In addition, some of the end-effector geometric features, such as finger's opening-closing range, can be described according to how they appear on the approach plane. Figure (3.3) shows how a three-finger end-effector contacts points  $c_1$ ,  $c_2$ , and  $c_3$  to grasp the planar shape.

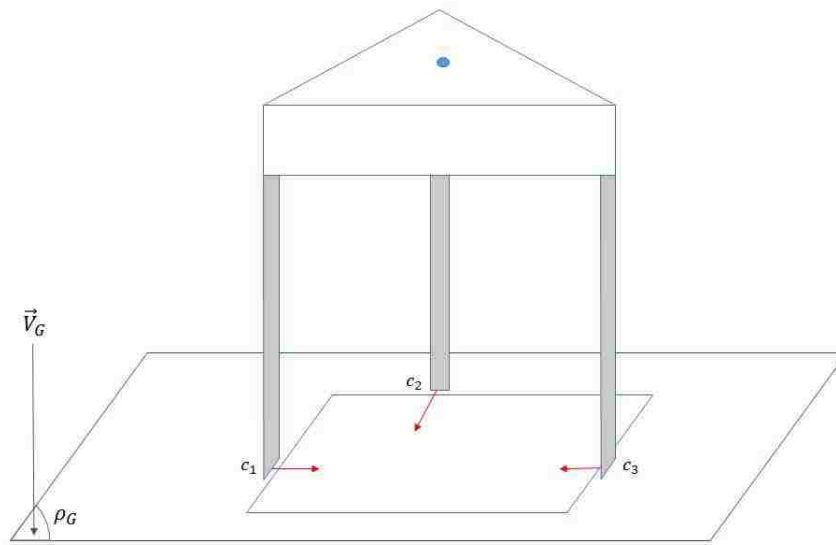


Figure 3.3: Grasp representation for a planar shape

## CHAPTER 4: EDGE LEVEL GRASP PLANNING

### Introduction

In this chapter, we first present an object representation and investigate its geometric features based on the scene depth map; then a grasp model for the end-effector is provided. In the end, pursuant to the development, we draw a relationship between an object's depth edges and force closure conditions. Finally, we specify contact location and end-effector pose to grasp the target object.

### Problem Statement

The problem addressed in this chapter is to find contacting regions for grasping unknown objects in a cluttered scene. The obtained grasp needs to exhibit force closure, be reachable, and also feasible under the specifications of a given end-effector. Partial depth information of the object, which is sensed by an RGBD camera, is the only input through this process and the proposed approach assumes that the manipulated objects have rigid and non-deformable shapes. In practice, we do not utilize objects with transparent and reflective surfaces since they cannot be sensed by the employed sensor technology.

### Object Depth Representation

Generally, 3D scanning approaches require multiple-view scans to construct complete object models. In this work, we restrict our framework to utilization of partial depth information captured from a single view and represent objects in a 2-dimensional (2D) space. As previously stated, our main premise is that potential contacting regions for a stable grasp can be found by looking for

i) sharp discontinuities or ii) regions of locally maximal principal curvatures in the depth map. A depth image can be shown by a 2D array of values which is described by an operator  $d(\cdot)$

$$z = d(r, c), d(\cdot) : R^2 \rightarrow R$$

where  $z$  denotes the depth value (distance to the camera) of a pixel positioned at coordinates  $(r, c)$  in the depth image ( $I_d$ ). Mathematically speaking, our principle suggests a search for regions holding *high gradient property* in depth or depth direction values. Gradient image, gradient magnitude image, and gradient direction image are defined as follows

$$\begin{aligned}
 \text{Depth Image:} & \quad I_d = [d(r_i, c_i)] \\
 \text{Image Gradient:} & \quad \nabla I = \left( \frac{\partial I_d}{\partial x}, \frac{\partial I_d}{\partial y} \right)^T \\
 \text{Gradient Magnitude Image:} & \quad I_M = \left[ \sqrt{\left( \frac{\partial I_d}{\partial x} \right)^2 + \left( \frac{\partial I_d}{\partial y} \right)^2} \right] \\
 \text{Gradient Direction Image:} & \quad I_\theta = \left[ \tan^{-1} \left( \left( \frac{\partial I_d}{\partial y} \right) / \left( \frac{\partial I_d}{\partial x} \right) \right) \right]
 \end{aligned} \tag{4.1}$$

where gradient magnitude image pixels describe the change in depth values in both horizontal and vertical directions. Similarly, each pixel of gradient direction image demonstrates the direction of largest depth value increase. In Figure (4.1), color maps of depth image and gradient direction image are provided. Sharp change in the color is an indication of occurrence of discontinuity in intensity values. Regions holding these specific features, in the image, locally divide the area into two sides and cause appearance of edges. In our proposed terminology, a *depth edge* is defined as a 2-dimensional collection of points in the image plane which forms a simple curve and satisfy the *high gradient property*.



Figure 4.1: (a) RGB image of the scene,  $I_c$  (b) Color map of the raw depth map. (c) Color map of computed gradient direction image,  $I_\theta$

**Definition 2** A point set  $p = (x, y)$  in the plane is called a curve or an arc if  $x = x(t)$  and  $y = y(t)$  where  $a \leq t \leq b$  while  $x(t)$  and  $y(t)$  are continuous functions of  $t$ . The points  $p(a)$  and  $p(b)$  are said to be initial and terminal points of the curve. A simple curve never crosses itself, except at its endpoints. A closed contour is defined as a piecewise simple curve in which  $p(t)$  is continuous and  $p(a) = p(b)$ . According to the Jordan curve theorem [48], a closed contour divides the plane in two sets, interior and exterior. Therefore, we define surface segment as a 2D region in the image plane which is bounded by a closed contour.

To expound on the kinds of depth edges and what they offer to the grasping problem, we investigate their properties in the depth map. To simplify the edge classification, it is worth inspecting how depth values change in a sample scene. Figure (4.2) illustrates measured depth values along the point A to G for a synthetic depth image. Please note that spikes and curvature changes toward this trajectory are cues for the potential depth edges.

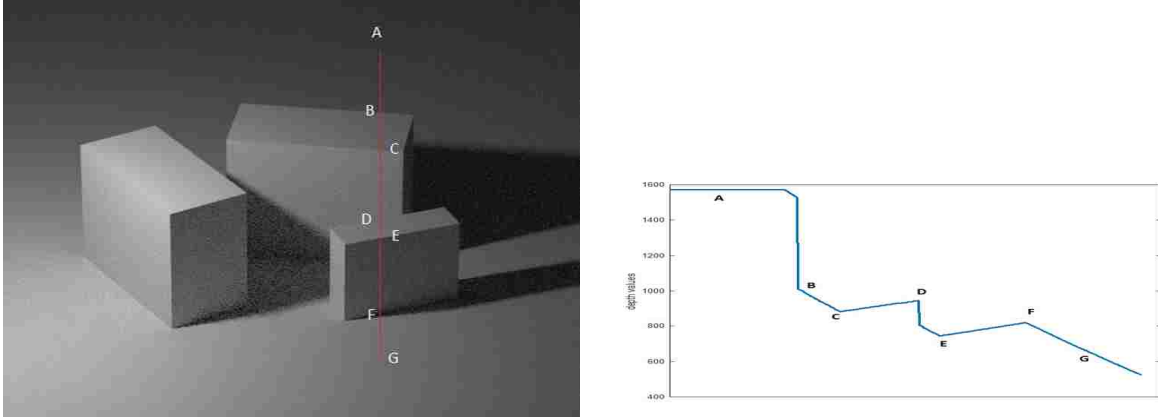


Figure 4.2: Measured depth values for a synthetic scene.

As a results, we categorize all the depth edges into two main groups: 1) Depth Discontinuity (DD) edges and 2) Curvature Discontinuity(CD) edges. A DD edge is created by high gradient in depth values or a significant depth value difference between its two sides in the 2D depth map ( $I_d$ ). It intimates a free-space between its belonged surface and its surroundings along the edge. A CD edge emerges from the directional change of depth values ( $I_\theta$ ) although it holds a continuous change in depth values on its sides. Note that the directional change of depth values is equivalent to surface orientation in 3D. In fact, a CD depth edge illustrates intersection of surfaces with different orientation characteristics in 3D. CD edges are further divided into two subtypes, namely, *concave* and *convex*. A CD edge is called convex if the following inequality is satisfied for any two points  $j_1$  and  $j_2$  belonging the convex set  $J$ , in its local neighborhood:

$$\forall j_1, j_2 \in J, \forall t \in [0, 1] : \quad (4.2)$$

$$D(tj_1 + (1 - t)j_2) \leq tD(j_1) + (1 - t)D(j_2)$$

Otherwise, it is considered as a concave edge. Simply speaking, the outer surface of the object curves like the interior of a circle at concave edges and curves like the circle's exterior at convex

edges.

Moreover, each surface segment in the image plane is the projection of an object's face. Particularly, projection of a flat surface maps all the belonged points to the corresponding surface segment while, in a case of curved/non-planar face, the corresponding surface segment includes that subset of the face, which is visible in the viewpoint. Assume that operator  $\lambda : R^2 \rightarrow R^3$  maps 2D pixels to their real 3D coordinates. In Figure (4.3), the  $S_i$  show 2D surface segments and  $A_i$  indicate collections of 3D points. It is clear that  $S_1$  represents a flat face of the cube and  $\lambda(S_1) = A_1$  while the surface segment  $S_2$  implies only a subset of the cylinder's lateral surface in 3D bounded between  $e_2$  and  $e_4$  such that  $\lambda(S_2) \subseteq A_2$ . Hence, a depth edge in the image plane may or may not represent an actual edge of the object in 3-dimensional space. Thus, edge type determination, in the proposed framework, relies on the viewpoint. While a concave CD edge holds its type in all the viewpoints, a convex CD edge may switch to DD edge and vice versa by changing the point of view.

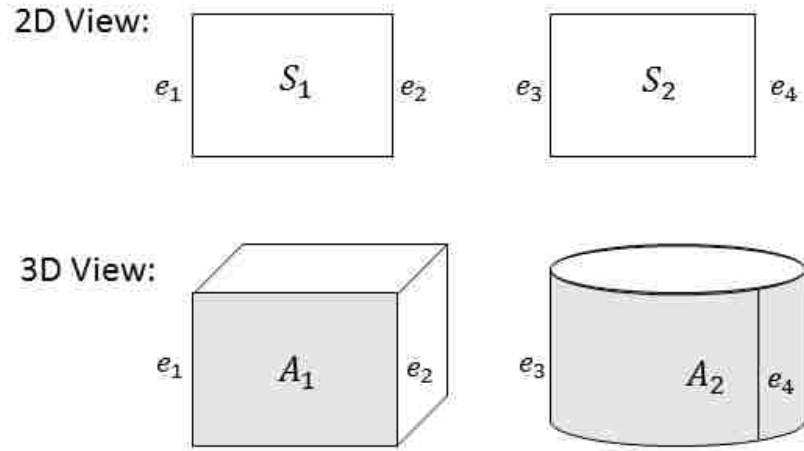


Figure 4.3: Geometric interpretation of a surface segment for a cube and a cylinder.

Thus, according to the presented model, the box in Figure ( 4.4) is projected to surfaces segments  $\{s_1, s_2, s_3\}$ , such that,  $e_1$  is a DD edge,  $e_2$  is a concave CD edge and  $e_3, e_4$  are convex CD edges.



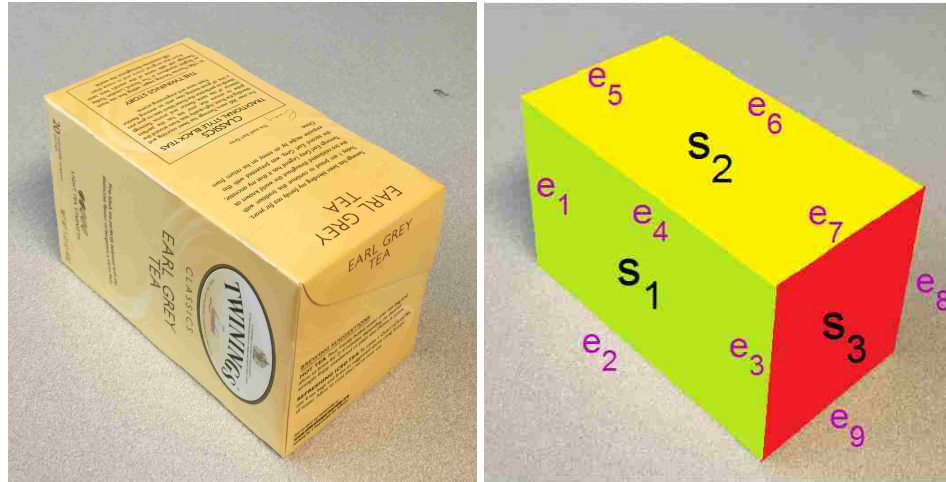


Figure 4.4: 2D Surface segments and depth edges are marked for an isolated object.

## 2D Grasp Identification

Until this point, we have discussed how to extract depth edges and form closed contours based on available partial information. In other words, objects are captured through 2D shapes formed by depth edges. Experiments show human tendency to grasp the objects by contacting its edges and corners [2]. The main reason is that edges provide a larger wrench convex and accordingly a greater capability to apply necessary force and torque directions. In this part, we aim to evaluate existence of grasps for each of the obtained closed contours as a way to contact an object. For this matter, we use contours as the input for the planar grasp synthesis process. The output grasp will satisfy reachability, force closure, and feasibility with respect to end-effector geometric properties. Next, we analyze the conversion of a planar grasp to an executable 3D grasp. Finally, we point out the emerging ambiguity and uncertainties due to the 2D representation.

If we assume that the corresponding 3D coordinates of a closed contour are located on a plane,

planar grasp helps us to find appropriate force directions lying on this virtual plane. In addition, edge type determination guides us to evaluate the feasibility of applying the force directions in 3D. *Reachability* of a depth edge is measured by the availability of a wrench convex lying in the plane of interest. A convex CD edge provides wrench convexes for possible contacting of two virtual planes while a concave CD edge is not reachable for a planar grasp. Exerting force on a DD edge, which also points to object interior, is just possible from one side. Therefore, DD and convex CD edges are remarked as reachable edges while concave CD edges are not considered as available points for planar contact.

For the purpose of simplicity in the analysis and without loss of generality, we approximate curved edges by a set of line segments. As a result, all 2D contours turn into polygonal shapes. To obtain the planar force closure grasp, we assume each polygon side represents just one potential contact. Then we evaluate all the possible combinations of polygon sides subject to the force-direction closure (Theorem I) and torque-closure (Theorem II) conditions.

We name the validation of force-direction closure, Angle-test. According to Chapter 3, force-direction closure is satisfied for a two-opposing fingers contact, if the angle made by two edges is less than twice the friction angle. The Angle-test for a three-finger end-effector is passed for a set of three contacts such that a wrench from the first contact with opposite direction overlaps with any positive combination of the other two contacts' provided wrenches (friction cones) [4].

In Chapter 3, we also discussed how to check if a set of points, corresponding to wrench convexes, holds torque-closure feature. Here, we apply the following steps to recognize regions that include such those points on each edge:

1. Form orthogonal projection areas ( $H_i$ ) for each edge  $e_i$
2. Find the intersection of projection areas by the candidate edges and output the overlapping

area ( $\bar{H}$ )

3. Back-project the overlapping area on each edge and output the contact regions ( $\bar{e}_i$ )

In fact, torque-closure is satisfied if there exists a contact region per each edge. We call this procedure Overlapping test. Figure (4.5.a) illustrates projection areas for each edge by color coded dashed lines. In Figure (4.5.b), shaded area corresponds to the overlapping area and green lines correspond to contact regions.

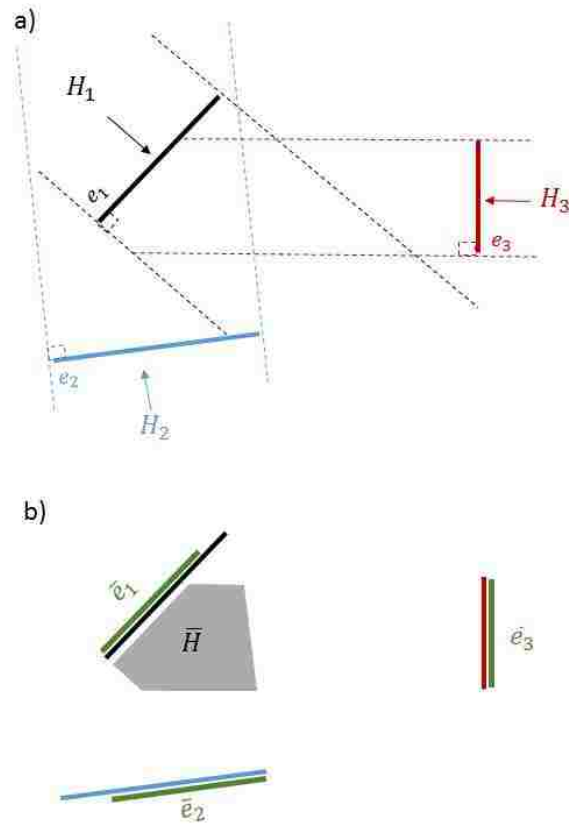


Figure 4.5: Overlapping test. a) shows intersection of orthogonal projection for three edges b) indicates overlapped region and edge contact regions.

Noting that all the procedure up to this step is performed in the image plane. In this step, we extract 3D coordinates of the involved edges in order to evaluate the feasibility of the output grasp with respect to the employed end-effector. For instance, comparison of Euclidean distance of line segments and two-fingered gripper width range specifies if the end-effector can fit around a pair of edges. Furthermore, by accessing the 3D coordinates of pixels, we find the Cartesian equation

of a plane passes through the edge contact regions ( $\bar{e}_i$ ). According to Section 3, the obtained plane determines end-effector approach plane ( $\rho_G$ ) and approach direction ( $\vec{V}_G$ ) at the grasping moment. In order to make the grasp robust to positioning errors, the center of each edge contact region ( $e_i^*$ ) denotes point contacts on the object's surface ( $c_i$ ). We discuss specification of the end-effector pose in the implementation Chapter, since it depends on end-effector kinematics and the chosen grasp policy for execution.

### Algorithm Overview

To sum up the discussed approach, we draw the steps as follows. First, we extract all the depth edges and form closed contours in the image plane. In the second step, depth edges forming each contour are evaluated to satisfy reachability and planar force closure conditions. Next, we check the feasibility with respect to the end-effector geometric properties, and find the end-effector approach plane for each combination of edges. In the final step, the grasp configuration parameters are determined based on the extracted plane and edge contact regions. The following algorithm also describes searching steps for constructing force closure using a depth image:

1. detect disc edges from depth image
2. detect curvature disc edges from gradient direction image
3. form closed contours using all depth edges
4. for each contour:
  - (a) remove the concave CD edges
  - (b) make combination of edges with desired number of contacts
  - (c) for each edges combination:

- i. perform Angle test
- ii. perform Overlapping test and output edge contact regions in 2D
- iii. perform end-effector geometric constraint test and output end-effector approach plane and contact locations
- iv. output grasp parameters w.r.t end-effector kinematics

It is worth mentioning the possible uncertainties in our method. Throughout this chapter, we assume that friction between fingertips and the object is large enough such that applied planar forces lie inside the 3D friction cones at the contacting points. To clarify, the current layout of our approach provides similar grasps for the contact pairs  $(e_1, e_2)$ ,  $(e_3, e_4)$  and  $(e_5, e_6)$  in Figure(4.6); however, the depth edges in these three cases offer different 3D wrench convexes which increases the grasping uncertainty. Another impacting factor is the relative position of force focus point with respect to the center of gravity of the object; applying sufficient level of force prevents possible **torques** that ensue from this uncertainty.

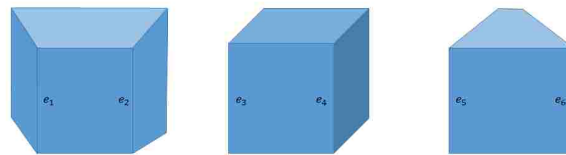


Figure 4.6: Shapes with similar planar grasps despite different 3D friction cones.

## Grasp Reliability

In this section, we aim to investigate and analyze features to predict the reliability of the constructed grasps. In general, the grasp planning component suggests a set of feasible grasp configurations per each input frame. Depends on the shape, geometry and the scene setup, each object can be located by multiple grasps. Therefore, a grasp selection procedure is required to sort the suggested configurations. This process can be formulated according to multiple factors including but not limited to the user preference, grasp uncertainty, task related constraints and robot's workspace limitation. Authors in [49] divide factors related to the position of the contact points on the object into 3 subgroups: i) grasp algebraic properties, ii) grasp geometric relations, and iii) limitation on the finger forces. Here, we focus on properties driven by the grasp geometric relations. Considering that contact polygon is the shape obtained by connecting end points of edge contact regions, the explanation and impact of chosen features to the grasp are drawn in the following:

- contact region lengths: Due to uncertainties in the perception and arm controller, larger margin for finger positioning is desired. In accordance with that, longer edge contacts indicate a more reliable grasp. For this matter, 3D length of each contact region is considered as an important factor.
- contact region distance: Wide contact surface requires a more precise positioning of the end-effector around the object. Depending on the grasp strategy and end-effector geometry, distance between the contact regions can be a measure for easiness of a grasp. In a two-opposed finger gripper setup, difference between maximum gripper width and the contact distance provides a margin for end-effector positioning margins.
- contact region relative angle: Based on the force direction closure concept, a smaller angle between the edge segments causes less uncertainty in the finger contacts. In addition to a

maximum threshold for this angle, lower values demonstrates more reliable grasp configuration

- contact polygon area: The area covered by the contact polygon is an indication of grasp robustness [50]. Assuming a 3-finger grasp, larger triangle is more resistance to external torques.
- contact region coplanarity: same finger forces, A grasp demonstrates higher force efficiency when contact points lie in a same plane. In simple words, same finger forces have larger summation when the applied forces are orthogonal to the object's boundary [49]. Deviation of contact region set points from the best estimated planes can be considered as a scale for coplanarity of the forces.
- contact region pixel density: More number of pixels per length unit illustrates more accurate depth estimation. Angle view of the depth sensor and distance of the points with respect to the camera, impact how the object projects on the image plane. Therefore, higher pixel resolution for same lengths provide more reliable coordination extraction.
- contact region curvature: Curvature along the normal of each contact edge determines how each 3D friction cone is positioned. Note that, this feature is not measurable when the contact region is extracted from a DD depth edge. In those cases, we replace it by the depth difference of interior and exterior points at the DD edge.
- contact region and center of mass distance: Assuming a unique distribution of object's weight, the center of a depth closed contour is considered as center of mass. Distance between the force focus point and estimated center of mass is a good metric to assess the torque closure property of the constructed grasp.

The above features are applicable for different hand configuration. However, the detailed calcula-



tion for each of these features rely on chosen end-effector properties.

In order to rank each grasp configuration, we need to form a feature vector per suggested grasp. For this matter, we scale each feature possible values to the range of 0 to 1. In this setup, a value of zero is in tune with the ideal scenario of the grasp with respect to that certain feature. Later in the implementation chapter, we discuss possible formulations to take advantage of the extracted features for grasp selection process.

## CHAPTER 5: EDGE LEVEL GRASPING IMPLEMENTATION AND RESULTS

In this chapter, we present and discuss algorithm implementations and obtained results in details.

### Implementation Procedure

#### *Introduction*

In this section, we describe the implementation steps to process a depth image as the input and identify appropriate grasps. Notice that the current implementation focuses on finding grasps for a two-opposing finger gripper. Therefore, we employ the described algorithm in Section 4 to construct a grasp based on forming combination of two edges to indicate a pair of contact locations. A set of pixel-wise techniques is utilized to achieve the regions of interest in a 2D image and eventually obtain the desired 3D grasp. In addition, to cope with noise effects of edge detection step in the algorithm, we utilize a tweaked procedure to follow the approach steps. In fact, we skip contour formation process in the third step of the approach and directly look for the pairs that meet the discussed conditions. Thus, if an edge is missed in the detection step, we do not lose the whole contour and its corresponding edge pairs. However, the emerging complication is expansion of the pair formation search space. Later in this section, we introduce constraints to restrict this search space.

### *Edge Detection and Line Segmentation*

According to Section 4, depth edges appear in depth image and gradient direction image. Due to the discontinuity existing by traveling in the orthogonal direction of a DD edge in depth image ( $I_d$ ), the pixels belonging to the edge are local maxima of  $I_M$  (magnitude of the gradient image  $\nabla I$ ). Alongside, a CD edge demonstrates a discontinuity in gradient direction image ( $I_\theta$ ) values, which illustrate a sudden change in normal directions corresponding to the edge neighborhood. Thus, in the first step, an edge detection method is required to be applied to  $I_d$  and  $I_\theta$  to capture all the DD and CD edges, respectively. We selected Canny edge detection method [3] that outputs the most satisfying results with our collected data.

Generally, the output of an edge detection method is a 2D binary image. Imperfect measurement in depth image yields appearance of artifacts and distorted texture in the output binary images. For instance, an ideal edge is marked out with one pixel-width. However, practically there exist non-uniform thickness along the detected edges. In order to reduce such effects and enhance the output of edge detection, a set of morphological operations is applied to the binary images. In coordination with the aforementioned attempt, logical OR operation is used to integrate all the marked pixels corresponding to depth edges from  $I_d$  and  $I_\theta$  in a single binary image called detected depth image  $I_{DE}$ . Figure (5.1) shows the output of edge detection step for an acquired depth image from the Object Segmentation Dataset [27]. Note that, the only input through the whole algorithm is  $I_d$ , and color image is merely used to visualize the obtained results. For the purpose of visualizing, a range of colors is also assigned to the values of  $I_d$  and  $I_\theta$ . Improvements made by the morphological operations is noticeable in Figure (5.1.d).

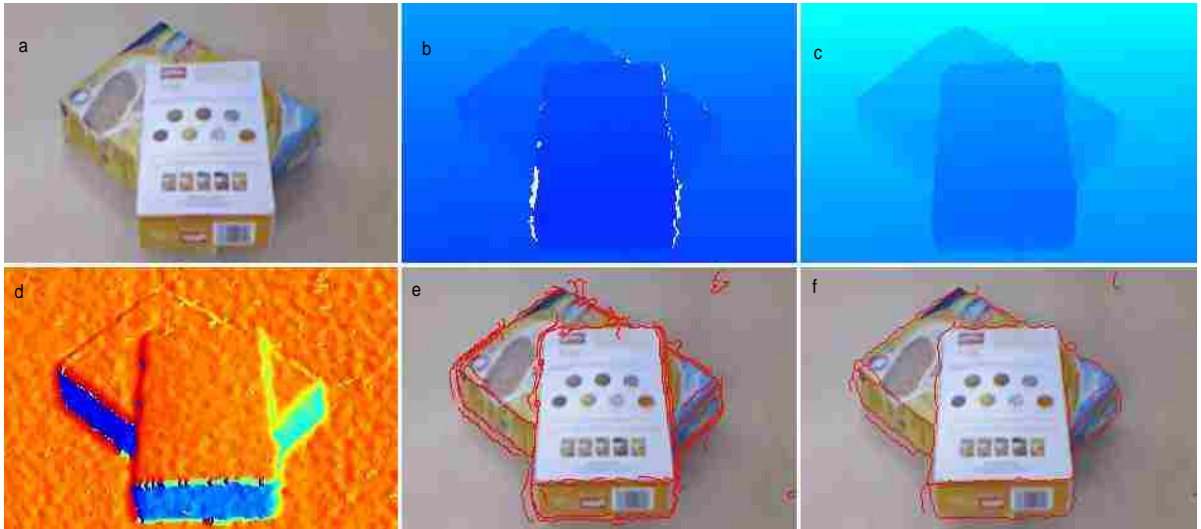


Figure 5.1: Applied edge detection on an acquired depth map. (a) RGB image of the scene,  $I_c$  (b) Color map of the raw depth map. White pixels imply to non-returned values from the sensor (depth shadows) (c) Color map of the processed depth map,  $I_d$  (d) Color map of computed gradient direction image,  $I_\theta$  (e) Detected edges before applying the morphological operations (f) Detected edges after the morphological process,  $I_{DE}$ .

To perform further processing, a procedure is required to distinguish edges by a 2D representation in the obtained binary image ( $I_{DE}$ ). Considering a 2D image with the origin on the left bottom corner, each pixel is addressed by a pair of positive integers. We employed a method proposed in [47] to cluster binary pixels into groups and then represent them by start and end points. Given  $I_{DE}$ , we first congregate the marked pixels into connected pixel arrays such that each pixel in an array is connected only to its 8 immediate neighbor pixels of the same array. Next, an iterative line fitting algorithm is utilized to divide the pixel arrays into segments such that each segment is indicated

by its two end-points. The pixels belong to a segment, satisfy an allowable deviation threshold from the 2D line formed by the end-points. As a result, pixels corresponding to a straight edge are represented by one line segment while, curved edges are captured by a set of line segments. Figure (5.2) indicates outputs of marked edge pixels and corresponding line segmentation for a synthetic depth image; colors are randomly assigned to distinguish the captured lines. Operator  $|L_i|$  computes pixel-length of line segment and  $\angle(L_i)$  measures the angle which is made by the line segment and the positive direction of horizontal axis in the range of  $[0^\circ \sim +180^\circ)$ , where counter-clockwise is assumed as the positive orientation.

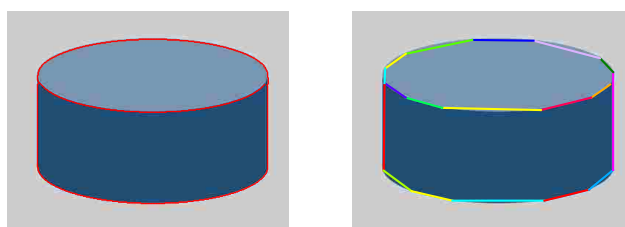


Figure 5.2: Line segmentation step is applied to a synthetic depth map. (a) detected edge pixels are marked) (b) edges are broken into line segment(s)

### *Edge Feature Extraction and Pair Formation*

At the end of the previous step, a set of pixel groups, indicated by a corresponding set of line segments, is provided. In this part, we aim to form pairs of line segments subject to mentioned constraints in Section 4. In this implementation, the geometric characteristic of a line segment is extracted from its adjacent pixels. Adjacent pixels are subsets of two surface segments in the

image plane and can be situated by rotating the line segment with either the clockwise orientation or the counter-clockwise orientation. We choose the collection of pixels in each surface segment by building 2D image masks enclosing the line segment. A parallelogram mask for a line segment is obtained by operator  $h(\cdot)$

$$h(\vec{L}_i, \vec{W}_i) \equiv h(\vec{L}_i, (w, \gamma))$$

where  $\vec{L}_i$  and  $\vec{W}_i$  are the sides of the parallelogram. In the equivalent operator representation,  $w$  and  $\gamma$ , respectively, show pixel-length of the line segment  $\vec{W}_i$  and the angle between the sides  $\vec{W}_i$  and  $L_i$  in the range of  $[-180^\circ : +180^\circ]$ . In a similar way, we provide the following predefined masks for a line segment:

$$H^0(L_i) = h(\vec{L}_i, (1, +90))$$

$$H^+(L_i) = h(\vec{L}_i, (w_0, +90))$$

$$H^-(L_i) = h(\vec{L}_i, (w_0, -90))$$

Please note that the binary mask locates the region of interest in the image and then desired functions are applied to the pixels' values. Figure (5.3.a) demonstrates masks  $H_1$  and  $H_2$  provide a positive angle parallelogram for  $L_1$  and negative angle parallelogram for  $L_2$ , respectively. Consider that increasing the value of  $w$  results in larger masks and robust feature identification.

We take advantage of the defined masks to evaluate reachability of each line segment and existence of a wrench convex for it. To do so, the line segments have to be assigned with an edge type label. Comparison of binary masks  $H^0(L_i)$  applied to  $I_d$  and  $I_\theta$  images results in distinguishing DD and CD line segments from one another. In addition, a line segment divides its local region in two sides. Therefore, the object is posed either with a positive orientation w.r.t. the line segment or a negative orientation. As discussed earlier, the wrench convex(es) is available in certain side(s) for each line segment. Note that depth value of DD edge sides hint at object relative pose with respect

to the line segment. As a result, the side with lower depth value implies object (foreground) while the side with greater depth value points out the background; correspondingly available wrench is suggested. Likewise, evaluating the sides and line segment average depth values based on Equation (4.2) specifies convexity/concavity of a CD edge. Mathematically speaking, edge type feature is determined for a DD line segment  $L_i$  and a CD line segment  $L_j$  as follows

$$\left\{ \begin{array}{l} \text{if : } \bar{d}(H^+(L_i)) < \bar{d}(H^-(L_i)) \\ \text{then : } L_i \text{ is } DD^- \\ \text{otherwise : } L_i \text{ is } DD^+ \\ \text{if : } 1/2[\bar{d}(H^-(L_j)) + \bar{d}(H^+(L_j))] > \bar{d}(H^0(L_j)) \\ \text{then : } L_j \text{ is } CD^\pm \\ \text{otherwise : } L_j \text{ is } CD^0 \end{array} \right.$$

such that  $(\pm, +, -, 0)$  signs indicate availability of wrench convex w.r.t the line segment and  $\bar{d}(\cdot)$  operator takes the average of depth values over the specified region.

According to Section 4, a pair of line segments constructs a planar force closure grasp if it satisfies the Angle test and Overlapping test. Therefore, for constructing a two-opposing fingers grasp, line segments ( $L_i$  and  $L_j$ ) need to have opposite wrench signs and satisfy the following inequality:

$$|\angle(L_i) - \angle(L_j)| < 2\alpha_f$$

where  $\alpha_f$  is determined by the friction coefficient. For checking the existence of overlapping area and edge contact regions, we utilize  $h(\cdot)$  operator to create masks and form line segment corresponded projection areas. The  $\bar{H}_\beta$  mask is the pair overlapping area which is captured by

intersection of edges projection areas and acquired by the following relations:

$$\beta = 1/2 \times |180 - |\angle(L_i) - \angle(L_j)||$$

$$H_\beta(L_i) = \begin{cases} h(\vec{L}_i, (w_{\max}, \beta)) & \text{if } DD^- \text{ or } CD^- \\ h(\vec{L}_i, (w_{\max}, -\beta)) & \text{if } DD^+ \text{ or } CD^+ \end{cases}$$

$$\bar{H}(\beta) = H_\beta(L_i) \cap H_\beta(L_j)$$

such that  $H_\beta(L_i)$  addresses projection area made by line segment  $L_i$  with the angle of  $\beta$ . In fact,  $\beta$  implies orthogonal direction of the bisector. Assuming existence of the overlapping area, edge contact regions,  $L_i^*$  and  $L_j^*$  are parts of the line segments which are enclosed by the  $\bar{H}(\beta)$  mask. Figure (5.3.b) demonstrates projection areas and contact regions for a pair of edges.

**Remark 2** *In the case that we have access to the closed contours formed by depth edges, both the line segments are required to belong to a same closed contour.*



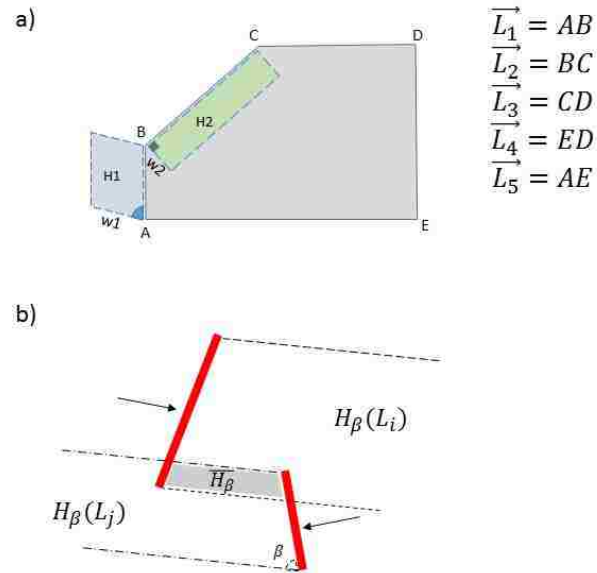


Figure 5.3: a) Examples of parallelogram masks the sides of 2D shape ABCDE. b) Projection area and edge contact regions for a pair of edges.

To this point, planar reachability and force closure features are assessed. As the final step, we check if the pair is feasible under the employed gripper constraints. We assume  $P_i = \lambda(L_i^*)$  is the set corresponding all the 3D points located on  $L_i^*$  region. Euclidean distance between the average points of two sets  $P_i$  and  $P_j$  is required to satisfy:

$$\epsilon_{\min} < \|\bar{P}_i - \bar{P}_j\|_2 < \epsilon_{\max}$$

where  $\epsilon$  denotes the width range of the gripper. In addition, to assure that  $P_i$  and  $P_j$  posed on

a plane, we fit plane model to the data. Throughout the current implementation, we utilized RANSAC [20] method to estimate the plane parameters. The advantage of RANSAC is its ability to reject the outlier points resulting from noise. If a point holds greater distance from the plane than an allowable threshold ( $t_{\max}$ ), it is considered an outlier point. The output plane and the normal unit vector pointing in the plane are referred as  $\rho_R$  and  $\vec{V}_R$ . Note that, for further processes, sets  $P_i$  and  $P_j$  are also replaced with corresponding sets excluding the outliers.

### 3D Grasp Specification

We desire to calculate grasp parameters based on the presented model in Section 3. To reduce the effects of uncertainties, we pick the centroid of the edge contact regions ( $P_i$ ) as the safest contact points. As stated by [21], a key factor to improve the grasp quality is orthogonality of the end-effector approach direction to the object surface. In addition, the fingers of a parallel-finger gripper can only move toward each other. Hence, according to the employed grasp policy, the gripper holds a certain pose such that the gripper approach direction is aligned with normal of the extracted plane. Subsequently, closing the fingers yields contact with the object at the desired contact points. Thus, for a graspable pair, grasp parameters are described by:

$$G(L_i, L_j) = (p_G, \theta_G, C_G)$$

$$= \begin{cases} p_G = (P_G, \mathbf{R}_G) \\ \theta_G = \{\theta_1, \theta_2\} \\ C_G = \{c_1, c_2\} = \{\bar{P}_i, \bar{P}_j\} \end{cases}$$

where 3D vector  $P_G$  and rotation matrix  $\mathbf{R}_G$  indicate the gripper pose. We adjust  $\theta_G$  such that fingers have maximum width before contacting and width equals to  $\|\bar{P}_i - \bar{P}_j\|_2$  during the contact.

If length of the fingers are equal to  $\epsilon_d$  and the fingers direction closure is defined by the unit vector  $\vec{V}_c = (\bar{P}_i - \bar{P}_j) / \|\bar{P}_i - \bar{P}_j\|_2$ , then we can obtain:

$$\begin{cases} P_G = 1/2 \times (\bar{P}_i + \bar{P}_j) - \epsilon_d \vec{V}_G \\ \mathbf{R}_G = \mathbf{R}_{o_1}^{o_2} \\ \vec{V}_G = \vec{V}_R \end{cases}$$

The matrix  $\mathbf{R}_{o_1}^{o_2}$  represents a rotation from the world coordinate frame  $o_1$  to the coordinate frame  $o_2$  which is captured by the three orthogonal axes  $[\vec{V}_R; \vec{V}_c \times \vec{V}_G; \vec{V}_G]$ .

### *Practical Issues*

Through the process of implementation on the real data, we face issues which are caused by uncertainties in the measured data. According to [26] error sources for imported data by depth sensors origin from imperfect camera calibration, lighting condition and properties of the object surface. RGBD sensors are subject to specific problems in measuring depth information based on the technology they use [40]. A common example of these problems is shadows or holes that appear in the depth image which point out the sensor inability to measure depth of such pixels. The main reason is some regions are visible to the emitter but not to the receiver sensor. Consequently, the sensor returns a non-value code for these regions. Since our implementation is mainly dominated by pixel-level processes, a procedure is required to handle this issue. In order to do so, we use a recursive median filter to estimate depth values for the shadow regions [29]. In Figure (5.1.b), white pixels display shadows in the sensed depth image and Figure (5.1.c) demonstrates depth image after the estimation procedure is performed.

Another issue in the case of DD edges is, if each edge pixel is rightly placed on its belonged

surface. In practice, an edge is detected as a combination of the pixels placed on both the object and the background. Since the marked pixels are utilized for the purpose of object pose estimation, we are interested to locate them on the foreground object. Although there are efficient ways to recognize the foreground pixels such as [10], but due to the computational cost we make use of a very simple pixel-level procedure. Based on the object-line relative position, we create  $H^+$  or  $H^-$  mask that orients toward the object side. Applying the mask to the gradient magnitude image ( $I_M$ ) provides accurate location of maximum depth gradient along the mask width (perpendicular to the line segment). The relative position of marked edge pixels with respect to the peak of the gradients determines if they are located on the object side or not. In the case of incorrect allocation, we move the marked pixel in the direction perpendicular to the line segment with a sufficient displacement to make sure the new marked pixel is located on the foreground side. It is important to note that this process is applied only to DD edges since there is no foreground/background concept for a CD edge.

Due to the projection occurring in camera from 3D to 2D, an ambiguity emerges causing two distinct depth edges along each other being captured as a single line segment. This issue can be handled by adding extra examination to the edge feature extraction. Considering masks  $H^0$  applied to  $I_d$  and  $I_\theta$  images demonstrate if there exist any depth or gradient orientation discontinuity along the edge. The occurrence of this discontinuity yields to breaking the edge into two line segments at the location of the abrupt change. Consider that in the existence of closed contour formation, the ambiguity does not arise and this test is discarded. In Figure (5.4) edges  $e_1$  and  $e_2$  in 2D image are considered as one line segment, while by performing the above test, they can be distinguished.

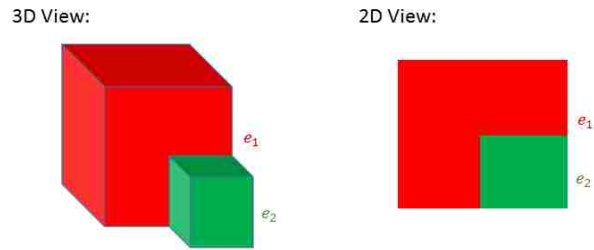


Figure 5.4: 2D object representation ambiguity.

In the following, Figure(5.5) includes extracted depth edges for a set of scenes including objects with variety of shapes and sizes. Please consider that no processing on the shadow regions in the depth image. The left column include the RGB views, the column in the center shows measured depth image, and the right column are binary images indicating depth edges before applying any morphological processing.

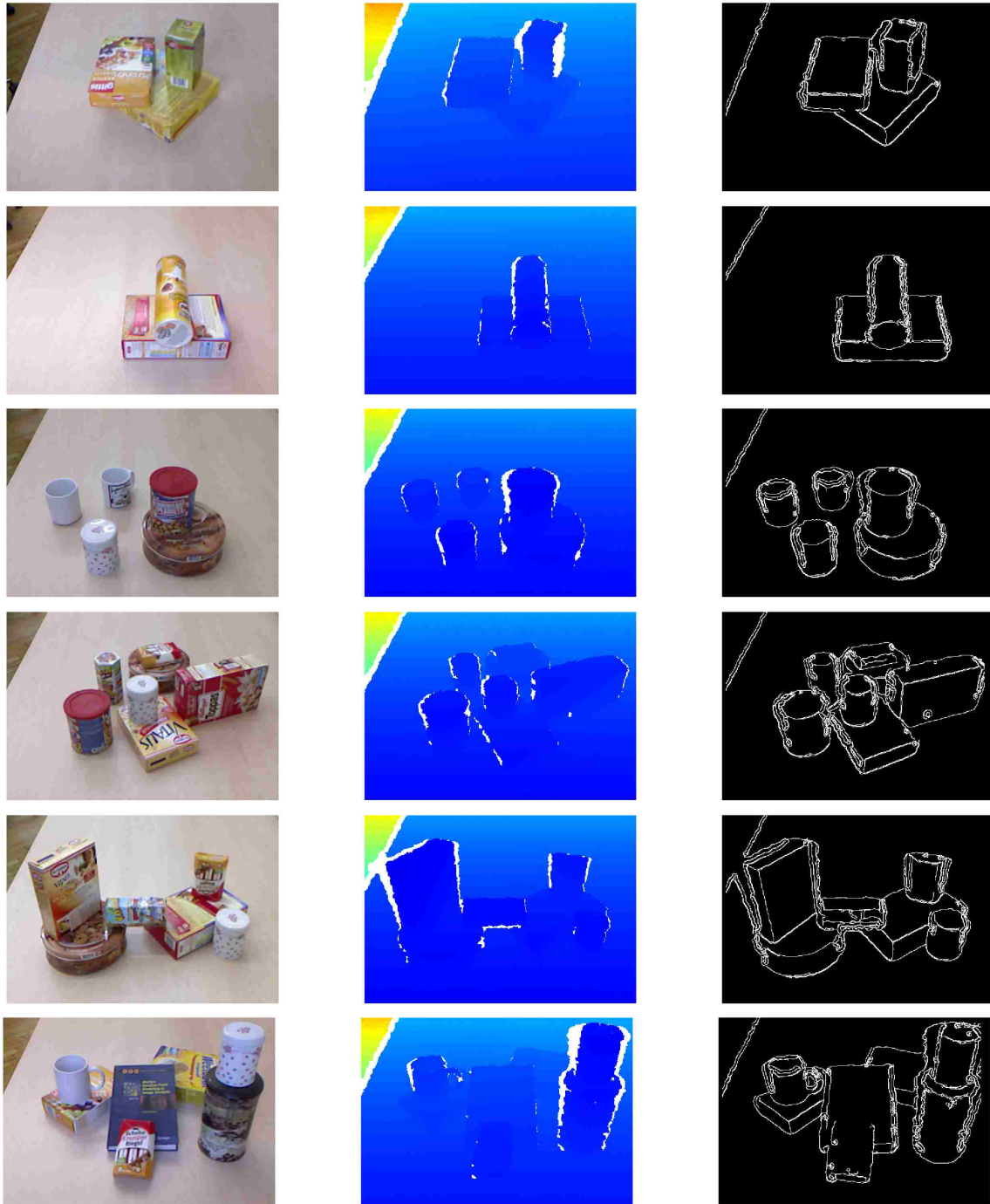


Figure 5.5: Binary images indicate extracted edges from the corresponding depth images.

## Experiments and Discussion

### *Introduction*

In this section, we first evaluate the performance of detection step of grasp planning algorithm and then conduct experiments with two setups to test the overall grasping performance using the 7-DOF Baxter arm manipulator. A standard data set named Object Segmentation Database (OSD) [27] is adopted for the simulation. Besides, we collected our own data set using Microsoft Kinect One sensor for real world experiments. The data sets include a variety of unknown objects from the aspects of shape, size, and pose. In both cases, the objects are placed on a table inside the camera view and data set provides RGBD image. The depth image is fed in the grasp planning pipeline and RGB image is just used to visualize the obtained results. Note that all the computations are performed in MATLAB.

### *Simulation-Based Results*

In this part, to validate our method, we focus on the output results of detection step in a simulation-based environment, *i.e.*, edge detection, line segmentation, and pair evaluation. To do so, we chose 8 images from OSD dataset including different object shapes and cluttered scenes. Figure (5.6) shows provided scenes.



Figure 5.6: Utilized images for obtaining simulation results.

To specify the ground truth, we manually mark all the reachable edges (DD and Convex CD) for the existing objects and consider them as *graspable edges*. If each graspable edge is detected with correct features, it is counted as a *detected edge*. Assuming there are no gripper constraints, a *graspable surface segment* is determined if it provides at least one planar force closure grasp in the camera view. In a similar way, detected surface segment, graspable object, and detected object are specified. Table (5.1) shows the obtained results by applying the proposed approach on the data set. In addition, Figure (5.7) illustrates the ground truth and detected edges for scene number 4.



Table 5.1: Simulation section results. Columns describe the number of (G)raspable and (D)etected objects, surface segments and edge for 8 different scene. The last row indicates average accuracy rates of detection in object level, surface-level and edge-level.

Scene	Objects	G. Object	D. Object	G. Surface	D. Surface	G. Edge	D. Edge
No.1	Boxes	3	3	6	6	17	14
No.2	Boxs	3	3	8	8	20	17
No.3	Cylinders	3	3	6	5	12	10
No.4	Cylinders	5	5	10	9	20	19
No.5	Mixed - low cluttered	6	6	13	9	28	21
No.6	Mixed - low cluttered	7	7	13	9	28	22
No.7	Mixed - high cluttered	11	11	24	17	55	42
No.8	Mixed - high cluttered	12	10	22	16	49	33
Average detection accuracy rate		97%		81%		80%	



Figure 5.7: Reference and detected edges for scene No.4 in the simulation-based results. Note that assigned colours are only used to distinguish the line segments visually. (a) reference graspable edges: each edge is manually marked by a line segment (b) detected graspable edges: marked points are detected by algorithm as graspable edges (c) detected line segments: each detected edge is represented by a number of line segments.

According to the provided results, although 20% of the graspable edges are missed in the detection steps, 97% of the existed objects are detected and represented by at least one of their graspable surface segments. This emphasizes how skipping the contour formation step has positive effects through the grasp planning. Obtained results also indicate that the efficiency of the proposed approach decreases as the scene becomes more cluttered. Addressing how exactly the performance of these pixel-wise techniques, such as edge detection and morphology operations, affect the efficiency of the our approach is complex. Output quality and setting of these methods strongly depend on characteristics of the image view and scene. Therefore here, we only analyze edge length effects and avoid detailing other effective parameters.

In fact, an edge appearing longer in a 2D image is composed of a greater number of pixels. Thus, it has a smaller chance of being missed in the detection step. In addition, since there is uncertainty in the measured data, a longer 2D edge signifies more reliable information in the grasp extraction step. On the other hand, appearance of an edge in the image relies on the distance and orientation of the object w.r.t. the camera view. Thus, depth pixel density of an object in 2D image affects the detection performance and reliability of its corresponding grasp.

### *Real World Experiments*

For the real world experiments, the approach is run in two phases, namely grasp planning and grasp execution. In the first phase, the proposed approach is applied to the sensed data and extracted grasping options are presented to the user by displaying the candidate pairs of contact regions. Based on the selected candidate, a 3D grasp is computed for the execution phase and the grasp strategy is performed. We collected our RGBD images using Microsoft Kinect Sensor and conducted our experiments using Baxter by Rethink Robotics. During all the experiments, arm manipulator, RGBD camera, and the computer station are connected through a ROS network. The

right arm of Baxter is fitted out with a parallel gripper. The gripper is controlled with two modes, in its "open mode" fingers distance is manually adjusted ,  $\epsilon_{\max} = 7cm$  based on the size of the utilized objects. During the "closed mode", fingers take either minimum distance,  $\epsilon_{\min} = 2cm$  or hold a certain force value in the case of contacting.



Figure 5.8: Utilized equipment for the real world experiments

## *Grasping Policy*

The grasp strategy is described for the end-effector by taking the following steps:

Step 1) Move from an initial pose to the planned pre-grasp pose.

Step 2) Wend through a straight line from pre-grasp pose to final grasp pose with fingers in the open-mode.

Step 3) Contact the object by switching the fingers to the close-mode.

Step 4) Lift the object and move to post-grasp pose.

In the current implementation, pre-grasp and post-grasp poses have the same orientation as the final grasp pose while they take a height  $20cm$  above the final grasp position. In this way, the end-effector approaches the object while holding a fixed orientation. Consequently, the fingers are prevented from colliding with the object prior to the grasp. Please note that a motion planner is utilized to find feasible trajectories for the arm joints.

## *Grasping Experiments*

We defined two scenarios to examine algorithm's overall performance, single object and multiple objects setups. In all the experiments, we assume target object is placed in the camera field of view, there exists at least one feasible grasp based on the employed gripper configuration, and planned grasps are in the workspace of the robot. An attempt is considered as a *successful grasp*, if the robot could grasp the target object and hold it for a 5 sec duration after elevating. In the cases, where the user desired object does not provide a grasp choice, the algorithm acquires a new image from the sensor. If the grasp does not show up even in the second try, we consider the attempt as a failed case. In the case, where planned grasp is valid but the motion planner fails to plan or execute the trajectory, the iteration is discarded and a new query is called.



Table 5.2: Single object experiment results. Four attempts for each object are performed. "L" indicates the large size and "S" indicates small size objects.

Object	% Succ.	Object	% Succ.
Toothpaste Box	100	L Box	100
S Blue Box	75	L Paper Cup	100
Banana	100	L Plastic Cup	100
S Paper Cup	75	Green Cylinder	100
Apple Charger	75	L Pill Container	100
Tropicana Bottle	100	Chips Container	75
S Pill Container	100	Smoothie Bottle	100
Mouse	50	Fruit Can	100
Average: 90.62 %			

According to the provided rates, 90% of the robot attempts were successful for the entire set where 11 objects were grasped successfully in all 4 iterations, 4 objects failed to be grasped successfully in 1 out of 4 iterations, while one object (mouse) had 2 successful and 2 unsuccessful attempts. In the unsuccessful attempts, the inappropriate orientation of the gripper during approaching moment is observed as the main reason of failure (4 out of 6) preventing the fingers from forming force closure on the desired contact regions. Basically, this relates performance of plane extraction from the detected contact regions. Observations during the experiments illustrate high sensitivity of the plane retrieval step to existence of unreliable data in the case of curved shape objects. For instance, in grasping the toothpaste box, although estimated normal direction ( $\vec{V}_R$ ) made a  $19^\circ$  angel with the expected normal direction (actual normal of the surface), the object was lifted successfully. However, a  $9^\circ$  degree error resulted in failure to grasp the mouse. Impact of force closure uncertainties

on the mouse case is also noticeable. For the other 2 unsuccessful attempts in the single object experiment, inaccurate positioning of the gripper was the main reason for the failure. For grasping the apple charger, gripper could not contact the planned regions, due to noisy values retrieved from low number of pixels on the object edges.

Multi object experiments are conducted to demonstrate the algorithm overall performance in a more complex environment. In each scene, a variety of objects are placed on the table and the robot approaches the object of interest in each attempt. Measuring the reliability and quality of candidate grasps is not in the scope of this paper. Hence, the order of grasping objects is manually determined such that:

- i) the objects which are not blocked by other objects in the view, are attempted first.
- ii) the objects, pursuant to lifting, result in scattering the other objects are attempted last.

Therefore, the user chooses one of the candidate grasps and robot attempts the target object unless there are no feasible grasps in the image. This experiment includes 6 different scenes, two scenes with box shaped objects, two scenes with cylinder shaped objects and two scenes with a variety of shapes. Table (5.3) indicates the obtained results of multi object experiment while Figure (5.10) demonstrates the setups of three of the scenes.

Table 5.3: Multi object experiment results. The success rate implies number of objects grasped successfully out of total number of objects in the scene.

Scene No.	Objects	Grasped Objects	Total attempts
1	Boxes	4 out of 4	4
2	Boxes	5 out of 5	5
3	Cylinders	4 out of 5	6
4	Cylinders	5 out of 5	6
5	Mix.	5 out of 6	8
6	Mix.	5 out of 8	8



Figure 5.10: Multi-object experiments scenes including variety of objects. (a) Scene No.2 (b) Scene No.3 (c) Scene No.6

Based on the obtained results, the proposed approach yields a 100% successful rate for box shaped objects, 90% for curved shapes, and 72% for very cluttered scenes with mixed objects. Figure (5.11) indicates a sequence of images during the grasp execution for the scene #3 in the multi



object experiment. During the grasp execution for scene #3 in (Figure (5.10)) in the multi object experiment. The robotic arm attempted to grasp the cylinder shaped objects located on the table. In the first two attempts, orange and blue bottles were successfully grasped, lifted and removed from the scene. Although in the third attempt the gripper contacted the paste can and elevated it, the object was dropped caused by lack of sufficient friction between the fingertips and object surface. Then, the arm approached the remaining objects (Green cylinder and large paper cup) and grasped them successfully. In the last attempt, another grasp was planned for the paste can by capturing a new image. This attempts also failed because of inaccurate estimated pose. Finally, the experiment was finished with one extra attempt and a successful rate of 4 out of 5. A video of the robot executing the tasks can be found online at [28].

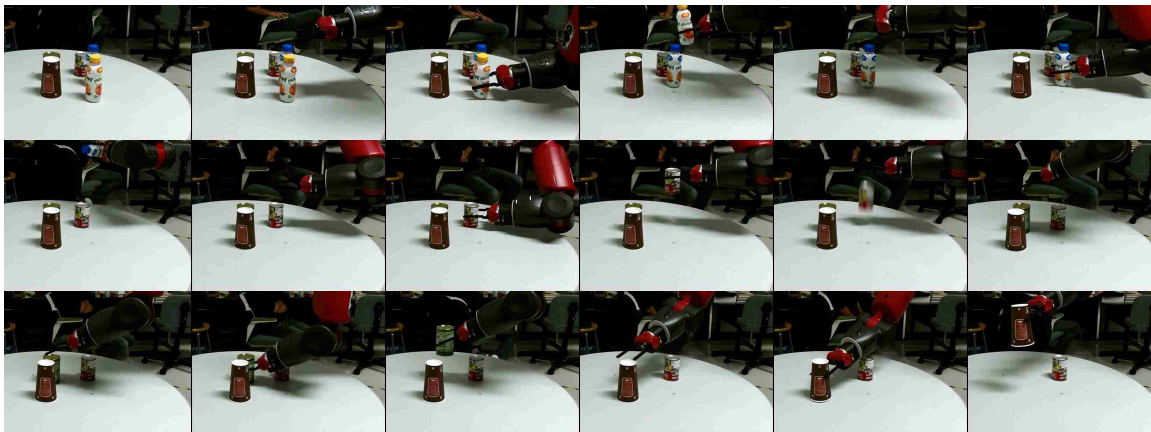


Figure 5.11: A Sequence of snapshots from the robot arm while approaching to grasp the objects in a cluttered scene.

## Discussion

According to the implemented approach, we discuss the performance of the approach and failure reasons at different levels, namely 2D contact region detection, 3D grasp extraction, and execution. In the detection phase, the output is a pair of 2D line segments. False positive and false negative pairs are caused by the following reasons: i) inefficiency of edge detection, ii) incorrect identification of edge type feature(DD/CD), and iii) incorrect identification of wrench direction feature  $(\pm, +, -, 0)$ . The aforementioned errors ensue from measurement noise and appearance of artifacts in the data. However, objective modification based on specific datasets can yield performance improvement. Note that if we perform detections on a synthetic dataset without adding noise, these reasons do not affect the output.

Since the user selects a desired pair, false positive output of detection phase, do not impact the grasp attempt in the conducted experiments. As a matter of fact, in the 3D grasp extraction step, the approach provides grasp parameters  $(P_G, \vec{V}_R, \vec{V}_C)$  based on a true positive pair of contact regions. Overall, the grasp parameter estimation errors can be sourced to the following underlying reasons: i) inaccurate DD edge pixel placement(background/foreground), ii) unreliable data for low pixel density objects, and iii) noise in the captured data. Since we derive contact regions instead of contact points, deviation of  $P_G$  in certain directions is negligible unless the finger collides with an undesired surface while approaching the object. Width of the gripping area with respect to the target surface determines limits for this deviation. Further, error in estimation of  $\vec{V}_R$  also results in force exertion on improper regions and consequently results in an unsuccessful grasp. Sensitivity of a grasp to this parameter depends on the surface geometry and finger kinematics. Compliant fingers show high flexibility to the estimated plane error, while firm wide fingertips do not tolerate the error. Uncertainties and assumptions regarding the friction coefficient, robot calibration, and camera calibration errors are among the factors impacting the performance of the execution step.

Future work will focus on three directions: 1) extracting more geometric features from the available data to control the uncertainties, 2) employing efficient techniques to reduce the noise effects and, 3) equipping the approach with a process to evaluate the grasp quality and reliability.

## CHAPTER 6: OBJECT LEVEL GRASPING

### Introduction

In this chapter, we develop a framework to extend the proposed grasp planning system with the aim of facilitating object-level grasping rather than the discussed edge-level. In fact, we plan to achieve a structure per each image to relate depth edges, surface segments, and objects together.

Generally speaking, having access to an object-level recognition system can result in the following directions:

- Improving user interface experience and providing options for high level user interactions,
- Boosting grasp ranking system to include high level depth descriptors and RGB information in the decision process,
- Increasing reliability in the grasp planning system.

### Problem Statement

The problem addressed in this chapter is to map obtained closed contours to a weighted graph representing surface segments geometric, adjacency and similarity relations, and then to utilize this graph for locating objects in an RGBD image. Please note that, the mentioned closed contours are formed by the identifying depth edges in accordance to the previous chapter explanation.

## Model Description

According to the presented model in Chapter 4, visible faces of objects appear as distinct surface segments in the image plane. The basic principal of this chapter is *inter* and *intra* features can guide us to associate the surface segments and build object segments in the image. Please note that based on utilized terminology in this chapter, an object segment contains all the pixels in the image, corresponded to an individual object in the scene.

To systematically analyze the pairwise surface segment relations, we take advantage of mathematical graph structure. Universally, a graph is expressed as following:

$$G_I = ( N_I, A_I )$$

where  $N$  is a set of nodes  $N_I = \{N_i\}$ , and  $A_I = \{A_{jk}\}$  is a set of arcs. As a matter of fact, each arc is an unordered pair of two nodes. For instance, Figure (6.1) illustrates a graph representation for the three object masks in the image. In this example, existence of arcs refers to adjacency of the objects.

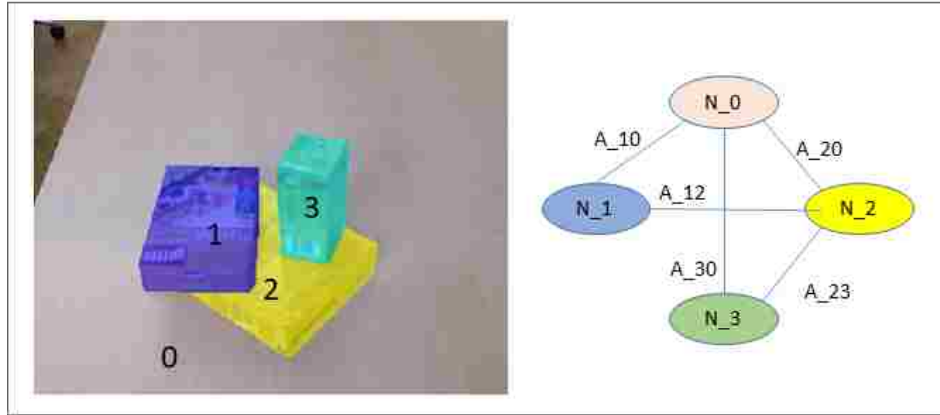


Figure 6.1: A sample graph representation for based on the adjacency feature. Each node in the graph indicates an object corresponded to overlay color.

To construct a graph in the proposed framework, given an image  $I$ , we assign each closed contour and its interior pixels to one single node,  $N_i$ . Possibilities of nodes  $N_i$  and  $N_j$  belong to a same object segment is determined by the arc,  $A_{jk}$ . The connecting arcs can be structured in multiple ways. However, to specify if two surface segments are originated by one object segment in the scene, we focus our investigations on three main inter-surface relations:

- 2D adjacency: For an isolated object setup, adjacent surface segments belong to a same object segment. Although, in occluded scenes, two surface segments may not share any common neighborhood.
- Curvature of common edge: Boundaries of all the convex shaped objects are formed by DD and convex CD depth edges. Therefore, depth edge type is a reliable metric for constructing object segments.
- Color similarity: Combination of color and texture features extracted from the closed contour

interior points are common metrics in RGB image segmentation. Dominant color is an instance with low computation cost.

In general, each graph arc can be a binary function of metric driven by the above features.

### Implementation

In this section, we describe the implementation steps to utilize the presented graph model based on a single geometry feature. Notice that, similar to the other chapters, RGB images are used for the matter of visualization, and we only employ depth information through the prediction process. Derived by edge detection methods applied to the depth and gradient direction image, a binary image is obtained. Given a binary image,  $I_BW$ , the following steps are taken

- Step1: Separate interior pixels of each closed contour, cluster them as one node and construct surface segment graph
- Step2: Determine common boundaries between closed contour and form adjacency matrix
- Step3: Connect the nodes through arcs in the surface segment graph according to the adjacency matrix
- Step4: Assign depth edge type to the graph adjacency arcs
- Step5: Cut the arcs where not holding convex CD type
- Step6: Merge the connected nodes and form object segment graph

In practice, largest surface segment in the binary image is considered as background cluster. Moreover, small closed regions appeared in the binary image due to the artifacts in the depth measurement. For this matter, we remove regions with smaller than a threshold area and then form the

surface segment grasp. Another point in this implementation setup is regions made by visible shadows are also omitted from the graph construction and assumed as a part of background.

In Figure (6.2), 3 example scenes from results of our algorithm is drawn. Row A shows 3 scenes with different object shapes. While, images in row B illustrate a color map measured depth image. In row C constructed binary image demonstrates extracted depth edges. All the interior points in each closed contour are clustered as one surface segment and assigned a unique color in row D. However, the noisy binary image caused appearance of artifact regions. Row E and F indicate resulted surface segment and object segment representations of the input images.



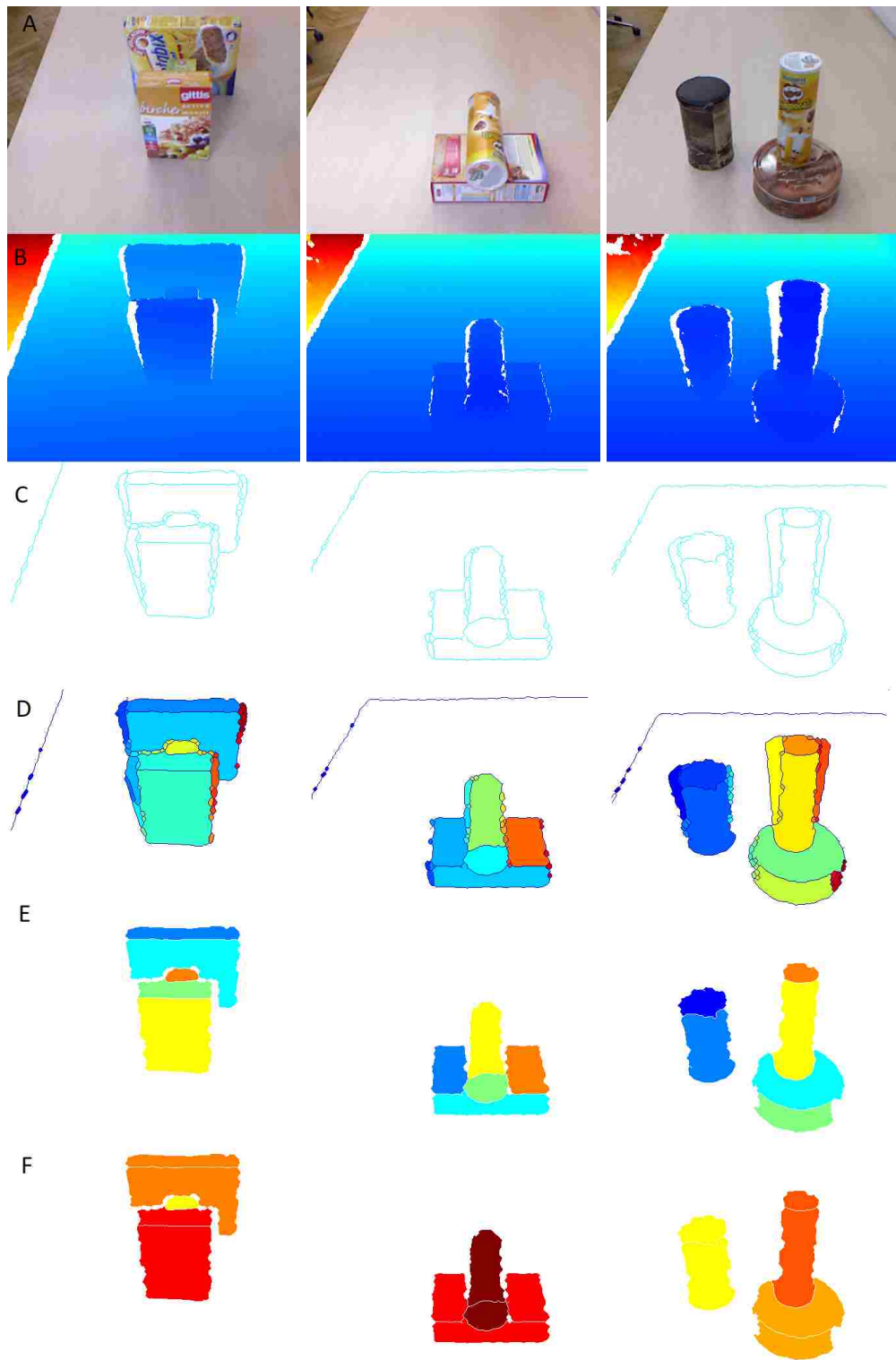


Figure 6.2: From a binary image to object segments by only depth information.

We evaluate the performance of this algorithm using Precision Recall metrics. The expected outcome is a 2D mask for each detected object and calculation of the precision and recall are based on the following formulations:

$$Precision = True\ Positive / (True\ Positive + False\ Positive)$$

$$Recall = True\ Positive / (True\ Positive + False\ Negative)$$

Consider that, true positive value indicates number of detected mask pixels overlay with the ground truth object mask. Provided data by Object Segmentation Database is used for this section. In total, 60 depth images are utilized in this test, including 346 objects, with a variety of shapes and sizes. The selected scenes also show different clutter levels. Applying the proposed approach on the mentioned dataset yields in Precision value of 98% and Recall value of 87%. Integrating an efficient procedure to avoid under-segmentation and over-segmentation in our object segmentation method can improve the obtained results.

According to the current implementation, one may extend the work to add more features, including color and geometric similarity, into the object localization framework. Moreover, integrating machine learning techniques can make the proposed approach more robust and adaptable to unstructured environments.

## CHAPTER 7: DEVELOPED ROS PACKAGE

### Introduction

In this chapter, the developed application, named GripIt, in Robotic Operating System is introduced and its capabilities are discussed.

### Description

Provided the depth map of an arbitrary scene, GripIt can extract the geometric edges of objects in said scene and further calculate the optimal approach vector for a 2-finger pinch-based robotic grippers. This information is presented in an interactive 3D point cloud view. Furthermore, GripIt provides editable parameters which governs particular features of the 2D and 3D scene. As a high-level overview, GripIt relies on machine vision algorithms to define the edges within a depth map. These edges are then paired and a normal vector calculated based on the underlying surface's depth map representation.

### Instruction

#### *Application Dependencies*

Currently, GripIt was built using Anaconda's build environment. GripIt also relies on the following packages:

- OpenCV3 3.1.0

- Matplotlib 2.1.0
- Numpy 1.13.3
- PyQt 5.6.0
- PqQtGraph 0.10.0
- Scipy 0.19.1
- Scikit-image 0.13.0

### *Launching GriPIt*

Currently, GriPIt must be launched from a terminal and depends on application arguments to load a scene.

Arguments:

- -m database[blend,real] Selects a database where a set of scenes are stored. The "Blend," stores synthetic data produced by blender while "Real" hosts an array of real images.
- -n imageNumber Scenes stored in the database are selected by their numeric index.

For instance, to load the second scene from the database, "real", the following commands must be used:

```
Python ./application.py -m real -n 2
```

Scene Parameters:

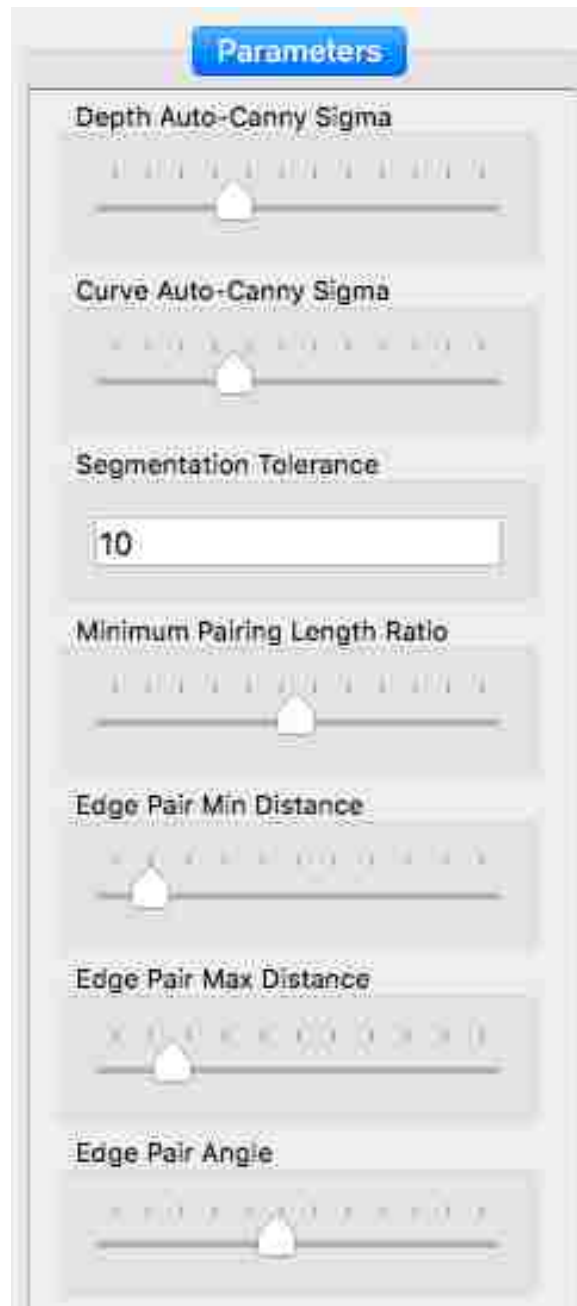


Figure 7.1: Parameters tuning panel

GriptIt incorporates a set of parameters which may be used to alter the edge detection and point-

cloud representation of the scene.

Parameters: Auto-Canny Sigma Controls the sensitivity for edge detection algorithms that are used. Lower value may exclude some edges, while a higher value may present noise. The default value of 33 Segmentation Tolerance Influences at what angle an edge may be divided. Minimum Pairing Length Ratio Edge Pair Min Distance Sets the minimum distance that an edge-pair has to be in order to be processed Edge Pair Max Distance Sets the maximum distance that an edge-pair has to be in order to be processed Edge Pair Angle The maximum angle between 2 edge pair vectors

Processing a Scene:

On loading a scene, GripIt will launch the Base view as show in figure 1. Here the program parameters are edited, a region of interest is established under the crop rectangle of the image, and the scene processed by clicking the process button.

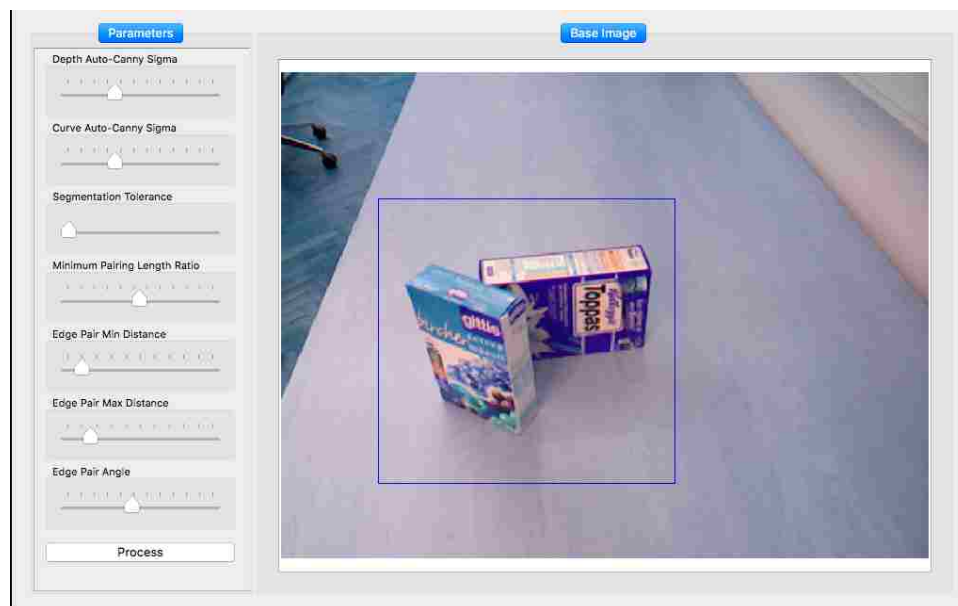


Figure 7.2: A screenshot of the developed image.

After a scene has been processed, a set of views will be added to the base window as tabs. These views present the calculated edge-pairs and approach vector as a 2D image and a 3D point-cloud. At any time the parameters could be re-edited and the scene re-updated with leaving the application.

#### Edge-Pair View:

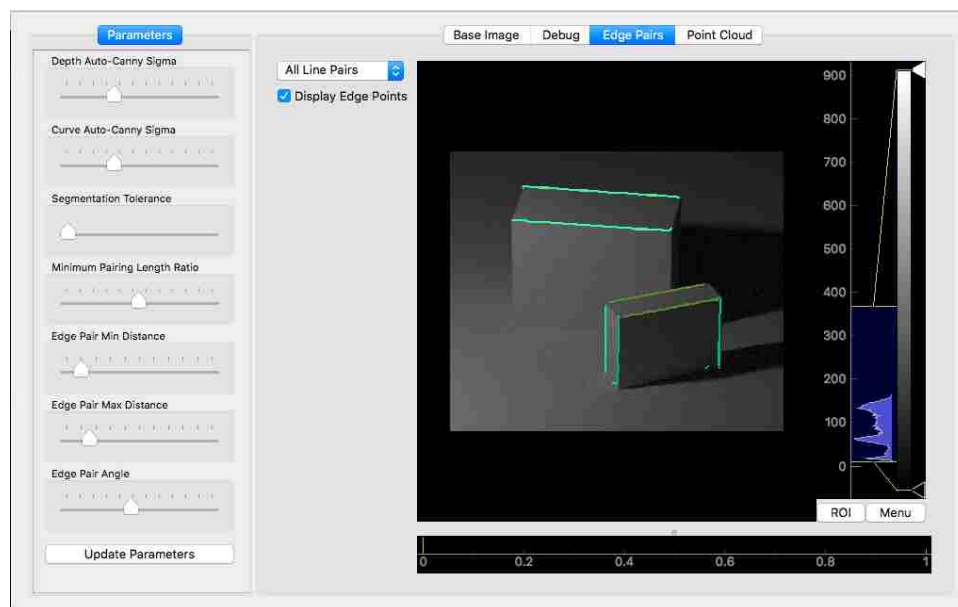


Figure 7.3: Obtained graspable pairs by the developed algorithm.

The first of these tabs, EdgePairs, displays the edge-pairs located in the cropped scene. These edge pairs are color coded, and given numeric names. The underlying points defining an edge could also be viewed by selecting "Display Edge Points." By pressing the left or right keys, a correspond 2d depth map image will be presented in the image view. To view the approach vector of an edge-pair, an edge-pair must be selected from the drop-down menu. Clicking "process face" generates an approach vector for the selected edge-pair. This vector could be viewed by switching to the Point-Cloud tab.

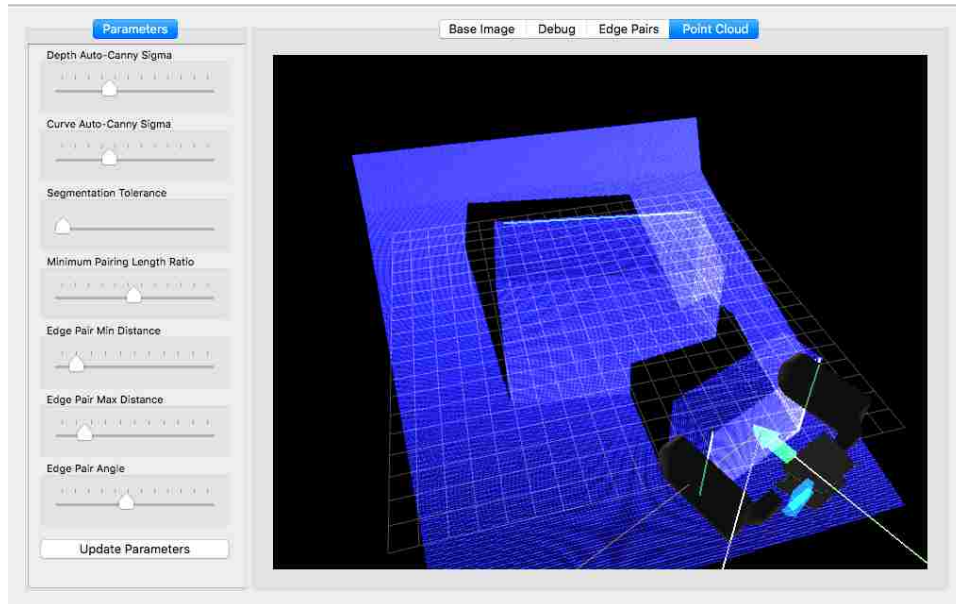


Figure 7.4: Illustration of obtained grasp pose and orientation demonstrated on the 3-dimensional point cloud.

In the point cloud tab, GripIt presents the scene in a 3D point cloud. The edge select tool of the EdgePair tab is synchronized with the edges that are shown in the PointCloud view. When an edge is processed the calculated approach vector is represented by a 2-finger gripper. This gripper dynamically resizes to grasp the selected edges. The scene presented in the Point-Cloud View could be panned and rotated as needed.



## CHAPTER 8: CONCLUSIONS

In this dissertation, we have discussed a variety of analytical and data-driven approaches that address the grasping problem, and pointed out the challenges each category tackles with. In addition, analytical concepts, related to stability of a grasp, are reviewed in chapter 3. In chapter 4, we have developed a framework to represent objects by depth edges and proposed a novel approach to obtain stable grasps based on partial geometry information. In fact, the proposed algorithm identifies reliable regions and possible force directions to contact the object of interest based on geometric features extracted from a captured single view depth map. In chapter 5, we have described the implementation steps to process a depth image as the input and identify feasible force-closure grasps. We have evaluated the performance of grasp planning approach in a simulation-based setup, and also have conducted multiple experiments to test the real-world performance by using the 7-DOF Baxter arm manipulator. Real world experiments demonstrate the ability of the proposed method to successfully grasp a variety of objects of different shapes, sizes, and colors. In chapter 6, we have extended our grasp planning process to localize objects in the image by adding a framework to find the relation between the surface segments. In chapter 7, the developed ROS package based on our approach is introduced and presented.

Based on our contributions in this dissertation, possible future directions may be:

- Developing an algorithm for tracking depth edges in an active vision system in order to increase the robustness,
- Formulating a learning-based framework to predict the obtained grasps quality by geometric and visual descriptors,
- And developing a probabilistic framework to construct invisible object surfaces according to

the extracted object

## APPENDIX A: IMPLEMENTED CODE DEPENDENCY

The following pages shows dependencies among MATLAB code files required for running the algorithm. All the scripts and source codes are available online at :

[https://github.com/amirjabal/GraspIt\\_MatlabScripts](https://github.com/amirjabal/GraspIt_MatlabScripts)

## Dependency Report

The Dependency Report shows dependencies among MATLAB files in a folder ([Learn More](#)).

- Show child functions  Show parent functions (current folder only)  
 Show subfunctions

Built-in functions and files in toolbox/matlab are not shown

Report for Folder H:\2019 Research\GraspingMatlabCodes\Edge-Grasping-Manus

MATLAB File List	Children (called functions)
<a href="#">DrawLineFeature</a>	
<a href="#">DrawSelectedPoints</a>	
<a href="#">DrawingScript</a>	unknown : <a href="#">BW20</a> unknown : <a href="#">BW30</a> unknown : <a href="#">BWn</a> unknown : <a href="#">DE10</a> unknown : <a href="#">DE3</a> unknown : <a href="#">DE_o</a> current dir : <a href="#">DrawLineFeature</a> current dir : <a href="#">DrawSelectedPoints</a> unknown : <a href="#">Ic</a> unknown : <a href="#">L00</a> unknown : <a href="#">L11</a> unknown : <a href="#">L22</a> unknown : <a href="#">LineFeature</a> unknown : <a href="#">Line_new</a> unknown : <a href="#">ListPair</a> unknown : <a href="#">TargetPoints</a> unknown : <a href="#">TargetPoints_noshift</a> toolbox : <a href="#">\images\images\labeloverlay.m</a> current dir : <a href="#">draw_2LogicalOnImage</a> current dir : <a href="#">draw_LogicalOnImage</a> current dir : <a href="#">draw_disc_curv</a> current dir : <a href="#">draw_pair_seperate</a> current dir : <a href="#">draw_pairs_v2</a> unknown : <a href="#">p_best</a> unknown : <a href="#">set_points</a>
<a href="#">LSE</a>	
<a href="#">LabelLineCurveFeature_v3</a>	toolbox : <a href="#">\images\images\roipoly.m</a>
<a href="#">LabelLineFeature_1026</a>	toolbox : <a href="#">\images\images\roipoly.m</a>
<a href="#">LabelLine_EdgeType</a>	toolbox : <a href="#">\images\images\roipoly.m</a> current dir : <a href="#">make_RecMask</a>
<a href="#">Load_Image_Win</a>	unknown : <a href="#">imgnum</a>
<a href="#">Lseq_to_lfeat_v2</a>	
<a href="#">MANUS_main</a>	unknown : <a href="#">Cloud_B2</a> unknown : <a href="#">Dir_vecRAN</a> current dir : <a href="#">DrawSelectedPoints</a> unknown : <a href="#">Ic</a> unknown : <a href="#">Id</a> unknown : <a href="#">LL1</a> unknown : <a href="#">LL2</a> unknown : <a href="#">Line_new</a> unknown : <a href="#">ListPoint_new</a> unknown : <a href="#">P_centerRAN</a> unknown : <a href="#">TargetPoints</a> unknown : <a href="#">TargetPoints_noshift</a> current dir : <a href="#">algorithm_part1</a> current dir : <a href="#">algorithm_part2</a> current dir : <a href="#">algorithm_part3h1</a> current dir : <a href="#">algorithm_part3h2</a>

```

current dir      : auto\_pair\_selection
current dir      : clean\_vars
current dir      : drawPclMarkedPts
current dir      : draw\_pairs\_v2
current dir      : draw\_pcl
current dir      : draw\_pcl\_Xaxis
unknown          : f\_success
current dir      : manus\_initial\_parameters
current dir      : manus\_pcl\_process
unknown          : n\_best
unknown          : original\_points
unknown          : p\_best
unknown          : ro\_best
unknown          : set\_points
unknown          : sorted\_pairs
current dir      : user\_selection

```

[Main\\_dissertation](#)

```

unknown          : Cloud\_B2
unknown          : Dir\_vecRAN
current dir      : DrawSelectedPoints
unknown          : Ic
unknown          : Id
unknown          : LL1
unknown          : LL2
unknown          : Line\_new
unknown          : ListPoint\_new
current dir      : MakeFeatureVector
unknown          : P\_centerRAN
current dir      : PairsLabelAssignment
unknown          : TargetPoints
unknown          : TargetPoints\_noshift
current dir      : algorithm\_part1
current dir      : algorithm\_part2
current dir      : algorithm\_part3h1
current dir      : algorithm\_part3h2
current dir      : auto\_pair\_selection
current dir      : clean\_vars
current dir      : dissertation\_initial\_parameters
current dir      : drawPclMarkedPts
current dir      : draw\_disc\_curv
current dir      : draw\_pairs\_v2
current dir      : draw\_pcl
current dir      : draw\_pcl\_Xaxis
unknown          : f\_success
current dir      : manus\_pcl\_process
unknown          : n\_best
unknown          : original\_points
unknown          : p\_best
unknown          : ro\_best
unknown          : set\_points
unknown          : sorted\_pairs
current dir      : user\_selection

```

[Main\\_sim](#)

```

unknown          : Cloud\_B2
unknown          : Dir\_vecRAN
current dir      : DrawSelectedPoints
unknown          : Ic
unknown          : Id
unknown          : LL1
unknown          : LL2
unknown          : Line\_new
unknown          : ListPoint\_new
current dir      : MakeFeatureVector
unknown          : P\_centerRAN
current dir      : PairsLabelAssignment
unknown          : TargetPoints
unknown          : TargetPoints\_noshift
current dir      : algorithm\_part1
current dir      : algorithm\_part2
current dir      : algorithm\_part3h1
current dir      : algorithm\_part3h2
current dir      : auto\_pair\_selection
current dir      : clean\_vars
current dir      : drawPclMarkedPts
current dir      : draw\_disc\_curv
current dir      : draw\_pairs\_v2

```

	<pre> current dir      : <a href="#">draw_pcl</a> current dir      : <a href="#">draw_pcl_Xaxis</a> unknown          : <a href="#">f_success</a> current dir      : <a href="#">manus_initial_parameters</a> current dir      : <a href="#">manus_pcl_process</a> unknown          : <a href="#">n_best</a> unknown          : <a href="#">original_points</a> unknown          : <a href="#">p_best</a> unknown          : <a href="#">ro_best</a> unknown          : <a href="#">set_points</a> unknown          : <a href="#">sorted_pairs</a> current dir      : <a href="#">user_selection</a> </pre>
<a href="#">MakeFeatureVector</a>	<pre> unknown          : <a href="#">Id</a> unknown          : <a href="#">ListPair</a> toolbox          : 2 Multiple class methods match <a href="#">imgradient.m</a> toolbox          : 2 Multiple class methods match <a href="#">imgradientxy.m</a> toolbox          : <a href="#">\images\images\roipoly.m</a> current dir      : <a href="#">draw_pair_seperate</a> current dir      : <a href="#">make_RecMask</a> </pre>
<a href="#">NextMove_v2</a>	
<a href="#">PairsLabelAssignment</a>	<pre> ERROR           : H:\2019 Research\GraspingMatlabCodes\Edge-Grasping- Manus\PairsLabelAssignment.m Syntax Error: L 11 (C 6): SYNER: Parse error at '=': usage might be invalid MATLAB syntax. </pre>
<a href="#">TrainingPairs</a>	<pre> unknown          : <a href="#">PairFeatures_table</a> unknown          : <a href="#">PairTScore</a> </pre>
<a href="#">absor</a>	<pre> subfunction      : <a href="#">dealr</a> subfunction      : <a href="#">matmvecHandle</a> subfunction      : <a href="#">mattvecHandle</a> </pre>
<a href="#">absor&gt;matmvecHandle</a>	
<a href="#">absor&gt;mattvecHandle</a>	
<a href="#">absor&gt;dealr</a>	
<a href="#">affine_fit</a>	
<a href="#">algorithm_part1</a>	<pre> unknown          : <a href="#">Id_background</a> unknown          : <a href="#">Id_o</a> current dir      : <a href="#">LabelLineCurveFeature_v3</a> current dir      : <a href="#">LabelLine_EdgeType</a> current dir      : <a href="#">Lseq_to_Lfeat_v2</a> toolbox          : 2 Multiple class methods match <a href="#">edge.m</a> toolbox          : 2 Multiple class methods match <a href="#">imgradient.m</a> toolbox          : 2 Multiple class methods match <a href="#">imgradientxy.m</a> toolbox          : <a href="#">\images\images\label2rgb.m</a> toolbox          : <a href="#">\images\images\wiener2.m</a> current dir      : <a href="#">edqelink</a> current dir      : <a href="#">line_match_1026</a> current dir      : <a href="#">lineseq</a> current dir      : <a href="#">mergeLines_hardCond</a> current dir      : <a href="#">merge_lines_1101</a> current dir      : <a href="#">morpho_modify_0712</a> current dir      : <a href="#">remove_boundries</a> current dir      : <a href="#">zeroElimMedianHoleFill</a> </pre>
<a href="#">algorithm_part1_test_combineDDCD</a>	<pre> unknown          : <a href="#">BW20</a> unknown          : <a href="#">BW30</a> unknown          : <a href="#">Id</a> unknown          : <a href="#">L00</a> current dir      : <a href="#">LabelLineCurveFeature_v3</a> current dir      : <a href="#">LabelLineFeature_1026</a> current dir      : <a href="#">Lseq_to_Lfeat_v2</a> toolbox          : 2 Multiple class methods match <a href="#">bwmorph.m</a> current dir      : <a href="#">edqelink</a> current dir      : <a href="#">line_match_1026</a> current dir      : <a href="#">lineseq</a> current dir      : <a href="#">merge_lines_1101</a> current dir      : <a href="#">morpho_modify_0712</a> current dir      : <a href="#">shadows_detection</a> </pre>

<a href="#">algorithm_part2</a>	unknown : <a href="#">DE3</a> unknown : <a href="#">Gmag</a> unknown : <a href="#">Ic</a> unknown : <a href="#">Id</a> unknown : <a href="#">ListPair</a> current dir : <a href="#">depth_shift_v20805</a> current dir : <a href="#">line2region</a> current dir : <a href="#">pair_connectionSingle_v2</a> unknown : <a href="#">pair_no</a>
<a href="#">algorithm_part3h1</a>	unknown : <a href="#">TargetPoints</a> unknown : <a href="#">TargetPoints_noshift</a> current dir : <a href="#">local_ransac_tim_v2</a>
<a href="#">algorithm_part3h2</a>	unknown : <a href="#">Cloud_B2</a> unknown : <a href="#">ListPair</a> unknown : <a href="#">TargetLines</a> toolbox : <a href="#">\aero\aedcm2angle.m</a> current dir : <a href="#">absor</a> current dir : <a href="#">distance2d</a> unknown : <a href="#">error_ransac</a> unknown : <a href="#">flag_orientation</a> current dir : <a href="#">manus_orientation_check</a> unknown : <a href="#">n_best</a> unknown : <a href="#">p_best</a> unknown : <a href="#">pair_no</a> current dir : <a href="#">ransacfitline</a> unknown : <a href="#">theta20</a> unknown : <a href="#">theta30</a>
<a href="#">auto_pair_selection</a>	unknown : <a href="#">ListPair</a> unknown : <a href="#">feat_vec</a> unknown : <a href="#">feat_vecN</a>
<a href="#">auto_pair_selection_test</a>	unknown : <a href="#">dis_L1L2_3d</a> unknown : <a href="#">error_plane</a> unknown : <a href="#">flag_orientation</a> unknown : <a href="#">pair_no</a> unknown : <a href="#">set_points</a> unknown : <a href="#">set_points2</a> unknown : <a href="#">set_points3</a> unknown : <a href="#">x0</a>
<a href="#">check_overlap</a>	toolbox :? Multiple class methods match <a href="#">pdist2.m</a>
<a href="#">clean_vars</a>	
<a href="#">compare_cameras</a>	unknown : <a href="#">BW20</a> unknown : <a href="#">BW30</a> unknown : <a href="#">L11</a> unknown : <a href="#">L22</a> current dir : <a href="#">draw_2LogicalOnImage</a> current dir : <a href="#">manus_initial_parameters</a> current dir : <a href="#">part1_test</a>
<a href="#">contour_detection</a>	unknown : <a href="#">BW20</a> unknown : <a href="#">BW30</a> unknown : <a href="#">Ic</a> unknown : <a href="#">Id</a> unknown : <a href="#">Id_background</a> current dir : <a href="#">LabelLine_EdgeType</a> current dir : <a href="#">Lseg_to_Lfeat_v2</a> toolbox : <a href="#">\images\images\bwboundaries.m</a> toolbox :? Multiple class methods match <a href="#">edge.m</a> toolbox :? Multiple class methods match <a href="#">regionprops.m</a> toolbox : <a href="#">\simulink\simulink\sl.m</a> toolbox :? Multiple class methods match <a href="#">pca.m</a> current dir : <a href="#">lineseg</a> current dir : <a href="#">morpho_modify_0712</a> unknown : <a href="#">pcloudL1</a> unknown : <a href="#">pcloudL2</a> unknown : <a href="#">pcloudL3</a>



<a href="#">depthToCloud_v2</a>	
<a href="#">depth_shift_v20805</a>	
<a href="#">dissertation_initial_parameters</a>	unknown : <a href="#">Id_o</a> current dir : <a href="#">Load_Image_Win</a> toolbox : <a href="#">\aero\aero\angle2dcm.m</a> unknown : <a href="#">cc</a> unknown : <a href="#">device_data</a>
<a href="#">distance2d</a>	
<a href="#">drawPclMarkedPts</a>	
<a href="#">draw_2LogicalOnImage</a>	
<a href="#">draw_LogicalOnImage</a>	
<a href="#">draw_Spairs</a>	
<a href="#">draw_disc_curv</a>	
<a href="#">draw_pair_seperate</a>	unknown : <a href="#">Ic</a> unknown : <a href="#">Line_new</a>
<a href="#">draw_pair_seperate_0617</a>	unknown : <a href="#">Ic</a> unknown : <a href="#">Line_new</a>
<a href="#">draw_pairs_v2</a>	
<a href="#">draw_pcl</a>	
<a href="#">draw_pcl_Xaxis</a>	
<a href="#">draw_pcl_pca</a>	toolbox :2 Multiple class methods match <a href="#">pca.m</a>
<a href="#">draw_shifted_pointes</a>	
<a href="#">drawedqelist</a>	subfunction : <a href="#">midedge</a>
<a href="#">drawedqelist&gt;midedge</a>	
<a href="#">edge_modified</a>	toolbox :2 Multiple class methods match <a href="#">im2single.m</a> subfunction : <a href="#">parse_inputs</a> subfunction : <a href="#">selectThresholds</a> subfunction : <a href="#">smoothGradient</a> subfunction : <a href="#">thinAndThreshold</a>
<a href="#">edge_modified&gt;parse_inputs</a>	toolbox : <a href="#">\images\images\+images\+internal\parseNonStringInputsEdge.m</a>
<a href="#">edge_modified&gt;smoothGradient</a>	toolbox :2 Multiple class methods match <a href="#">imfilter.m</a>
<a href="#">edge_modified&gt;selectThresholds</a>	toolbox :2 Multiple class methods match <a href="#">imhist.m</a>
<a href="#">edge_modified&gt;thinAndThreshold</a>	toolbox : <a href="#">\images\images\bwselect.m</a> toolbox : <a href="#">\images\images\private\cannyFindLocalMaxima_mexw64</a>
<a href="#">edgelinek</a>	toolbox :2 Multiple class methods match <a href="#">bwmorph.m</a> subfunction : <a href="#">availablepixels</a> current dir : <a href="#">findendsjunctions</a> subfunction : <a href="#">trackededge</a>
<a href="#">edgelinek&gt;trackededge</a>	subfunction : <a href="#">availablepixels</a> subfunction : <a href="#">unitvector</a>
<a href="#">edgelinek&gt;availablepixels</a>	
<a href="#">edgelinek&gt;unitvector</a>	
<a href="#">edgelist2image</a>	
<a href="#">eyeonhand</a>	
<a href="#">find_point</a>	
<a href="#">findendsjunctions</a>	toolbox :2 Multiple class methods match <a href="#">applylut.m</a> toolbox : <a href="#">\images\images\makelut.m</a> unknown : <a href="#">edgeim</a> subfunction : <a href="#">ending</a> subfunction : <a href="#">junction</a>

<a href="#">findendsjunctions&gt;iunction</a>	
<a href="#">findendsjunctions&gt;ending</a>	
<a href="#">findisolatedpixels</a>	toolbox :2 Multiple class methods match <a href="#">applylut.m</a> toolbox : <a href="#">\images\images\makelut.m</a> subfunction : <a href="#">isolated</a>
<a href="#">findisolatedpixels&gt;isolated</a>	
<a href="#">fitline3d</a>	
<a href="#">fun_AlignToAnnotation</a>	unknown : <a href="#">I_ref</a> unknown : <a href="#">seq_obj_image</a>
<a href="#">fun_DepthEdgeExtractor</a>	unknown : <a href="#">Id_o</a> toolbox :2 Multiple class methods match <a href="#">edge.m</a> toolbox :2 Multiple class methods match <a href="#">imgradient.m</a> toolbox :2 Multiple class methods match <a href="#">imgradientxy.m</a> toolbox : <a href="#">\images\images\label2rgb.m</a> toolbox : <a href="#">\images\images\wiener2.m</a> current dir : <a href="#">remove_boundries</a> current dir : <a href="#">zeroElimMedianHoleFill</a>
<a href="#">fun_Surf2Obj</a>	
<a href="#">fun_SurfEdgeType</a>	toolbox :2 Multiple class methods match <a href="#">bwmorph.m</a>
<a href="#">fun_SurfSegProcess</a>	toolbox : <a href="#">\images\images\bwboundaries.m</a> toolbox :2 Multiple class methods match <a href="#">imdilate.m</a> toolbox :2 Multiple class methods match <a href="#">regionprops.m</a>
<a href="#">fun_SurfSubMerge</a>	unknown : <a href="#">Gdir</a> unknown : <a href="#">Id_o</a> toolbox : <a href="#">\images\images\label2rgb.m</a> toolbox :2 Multiple class methods match <a href="#">kmeans.m</a> toolbox : <a href="#">\stats\stats\silhouette.m</a> unknown : <a href="#">idx</a> unknown : <a href="#">seq_image</a>
<a href="#">func_refToObj</a>	unknown : <a href="#">I_ref</a> unknown : <a href="#">seq_obj_image</a>
<a href="#">hhf_pad</a>	
<a href="#">line2region</a>	toolbox : <a href="#">\images\images\roipoly.m</a>
<a href="#">lineMergeOrientationFlag</a>	toolbox : <a href="#">\images\images\roipoly.m</a> current dir : <a href="#">make_RecMask</a>
<a href="#">line_match_1026</a>	current dir : <a href="#">check_overlap</a> current dir : <a href="#">distance2d</a> current dir : <a href="#">relative_pos_1026</a>
<a href="#">line_plane_checker</a>	
<a href="#">lineseq</a>	current dir : <a href="#">maxlinedev</a>
<a href="#">local_ransac_tim_v2</a>	current dir : <a href="#">LSE</a>
<a href="#">make_RecMask</a>	
<a href="#">manus_initial_parameters</a>	unknown : <a href="#">Id_o</a> current dir : <a href="#">Load_Image_Win</a> toolbox : <a href="#">\aero\aero\angle2dcm.m</a> unknown : <a href="#">cc</a> unknown : <a href="#">device_data</a>
<a href="#">manus_orientation_check</a>	unknown : <a href="#">n_best</a>
<a href="#">manus_pcl_process</a>	unknown : <a href="#">Id</a> unknown : <a href="#">Id_o</a> current dir : <a href="#">depthToCloud_v2</a>

<a href="#">maxlinedev</a>	
<a href="#">mergeLines_hardCond</a>	current dir : <a href="#">LabelLine_EdgeType</a> current dir : <a href="#">lineMergeOrientationFlag</a>
<a href="#">merge_lines_1101</a>	
<a href="#">morpho_modify_0712</a>	toolbox :2 Multiple class methods match <a href="#">bwmorph.m</a> toolbox :2 Multiple class methods match <a href="#">imclose.m</a> toolbox : <a href="#">\images\images\strel.m</a>
<a href="#">pair_connectionSingle_v2</a>	current dir : <a href="#">NextMove_v2</a> toolbox : <a href="#">\images\images\bwconncomp.m</a>
<a href="#">part1_test</a>	unknown : <a href="#">Id_o</a> toolbox :2 Multiple class methods match <a href="#">edge.m</a> toolbox :2 Multiple class methods match <a href="#">imgradient.m</a> toolbox :2 Multiple class methods match <a href="#">imgradientxy.m</a> toolbox : <a href="#">\images\images\label2rgb.m</a> toolbox : <a href="#">\images\images\wiener2.m</a> current dir : <a href="#">remove_boundries</a> current dir : <a href="#">zeroElimMedianHoleFill</a>
<a href="#">point2pair</a>	toolbox :2 Multiple class methods match <a href="#">sortrows.m</a>
<a href="#">prepare_data</a>	
<a href="#">prepare_data2</a>	unknown : <a href="#">N</a>
<a href="#">ransac</a>	toolbox : <a href="#">\stats\stats\ransac.m</a> unknown : <a href="#">ransac</a>
<a href="#">ransacfitline</a>	subfunction : <a href="#">defineline</a> current dir : <a href="#">fitline3d</a> subfunction : <a href="#">isdegenerate</a> subfunction : <a href="#">lineptdist</a> current dir : <a href="#">ransac</a>
<a href="#">ransacfitline&gt;defineline</a>	
<a href="#">ransacfitline&gt;lineptdist</a>	
<a href="#">ransacfitline&gt;isdegenerate</a>	
<a href="#">ransacfitplane</a>	subfunction : <a href="#">defineplane</a> unknown : <a href="#">fitplane</a> subfunction : <a href="#">isdegenerate</a> subfunction : <a href="#">planeptdist</a> current dir : <a href="#">ransac</a>
<a href="#">ransacfitplane&gt;defineplane</a>	
<a href="#">ransacfitplane&gt;planeptdist</a>	
<a href="#">ransacfitplane&gt;isdegenerate</a>	unknown : <a href="#">iscolinear</a>
<a href="#">relative_pos_1026</a>	
<a href="#">remove_boundries</a>	
<a href="#">shadows_detection</a>	toolbox : <a href="#">\images\images\bwboundaries.m</a> toolbox :2 Multiple class methods match <a href="#">bwmorph.m</a> toolbox :2 Multiple class methods match <a href="#">findpeaks.m</a>
<a href="#">surfature</a>	
<a href="#">test2</a>	current dir : <a href="#">DrawLineFeature</a> unknown : <a href="#">Ic</a> unknown : <a href="#">Id</a> toolbox :2 Multiple class methods match <a href="#">imgradient.m</a> toolbox :2 Multiple class methods match <a href="#">imgradientxy.m</a> toolbox : <a href="#">\images\images\roipoly.m</a> current dir : <a href="#">draw_2LogicalOnImage</a> current dir : <a href="#">make_RecMask</a>
<a href="#">test_Main20180205</a>	unknown : <a href="#">algorithm_part1_contour</a>

	current dir	: <a href="#">manus_initial_parameters</a>
<a href="#">test_Main_autolabel</a>	unknown	: <a href="#">Cloud_R2</a>
	unknown	: <a href="#">Dir_vecRAN</a>
	current dir	: <a href="#">DrawSelectedPoints</a>
	unknown	: <a href="#">Ic</a>
	unknown	: <a href="#">Id</a>
	unknown	: <a href="#">LL1</a>
	unknown	: <a href="#">LL2</a>
	unknown	: <a href="#">Line_new</a>
	unknown	: <a href="#">ListPoint_new</a>
	unknown	: <a href="#">P_centerRAN</a>
	unknown	: <a href="#">TargetPoints</a>
	unknown	: <a href="#">TargetPoints_noshift</a>
	unknown	: <a href="#">algorithm_part1_newMerge</a>
	current dir	: <a href="#">algorithm_part2</a>
	current dir	: <a href="#">algorithm_part3h1</a>
	current dir	: <a href="#">algorithm_part3h2</a>
	current dir	: <a href="#">auto_pair_selection</a>
	current dir	: <a href="#">clean_vars</a>
	current dir	: <a href="#">drawPclMarkedPts</a>
	current dir	: <a href="#">draw_pairs_v2</a>
	current dir	: <a href="#">draw_pcl</a>
	current dir	: <a href="#">draw_pcl_Xaxis</a>
	unknown	: <a href="#">f_success</a>
	current dir	: <a href="#">manus_initial_parameters</a>
	current dir	: <a href="#">manus_pcl_process</a>
	unknown	: <a href="#">n_best</a>
	unknown	: <a href="#">original_points</a>
	unknown	: <a href="#">p_best</a>
	unknown	: <a href="#">ro_best</a>
	unknown	: <a href="#">set_points</a>
	current dir	: <a href="#">user_selection</a>
<a href="#">test_ambiguity</a>	toolbox	: <a href="#">\images\images\roipoly.m</a>
<a href="#">test_ambiguity_ver2</a>	unknown	: <a href="#">Gdir</a>
	unknown	: <a href="#">Ic</a>
	unknown	: <a href="#">Id</a>
	unknown	: <a href="#">Line_new</a>
	current dir	: <a href="#">draw_LogicalOnImage</a>
<a href="#">test_combineImages</a>		
<a href="#">test_convex_objects</a>	unknown	: <a href="#">Id_o</a>
	toolbox	: <a href="#">\images\images\bwboundaries.m</a>
	toolbox	:2 Multiple class methods match <a href="#">edge.m</a>
	toolbox	:2 Multiple class methods match <a href="#">imgradient.m</a>
	toolbox	:2 Multiple class methods match <a href="#">imgradientxy.m</a>
	toolbox	: <a href="#">\images\images\label2rgb.m</a>
	toolbox	:2 Multiple class methods match <a href="#">regionprops.m</a>
	toolbox	: <a href="#">\images\images&gt;wiener2.m</a>
	toolbox	:2 Multiple class methods match <a href="#">pca.m</a>
	current dir	: <a href="#">manus_initial_parameters</a>
	current dir	: <a href="#">morpho_modify_0712</a>
	unknown	: <a href="#">pcloudI1</a>
	unknown	: <a href="#">pcloudI2</a>
	unknown	: <a href="#">pcloudI3</a>
	current dir	: <a href="#">remove_boundries</a>
	current dir	: <a href="#">zeroElimMedianHoleFill</a>
<a href="#">test_draw_pcl</a>	toolbox	:2 Multiple class methods match <a href="#">pca.m</a>
	unknown	: <a href="#">indx1</a>
	unknown	: <a href="#">indx2</a>
<a href="#">test_fig</a>	unknown	: <a href="#">BW20</a>
	unknown	: <a href="#">BW30</a>
	toolbox	: <a href="#">\images\images\bwboundaries.m</a>
	toolbox	: <a href="#">\images\images\label2rgb.m</a>
	current dir	: <a href="#">morpho_modify_0712</a>
<a href="#">test_filters</a>	toolbox	:2 Multiple class methods match <a href="#">bwmorph.m</a>
	toolbox	:2 Multiple class methods match <a href="#">edge.m</a>
	toolbox	:2 Multiple class methods match <a href="#">imgradient.m</a>

	<pre> toolbox      :? Multiple class methods match <a href="#">imgradientxy.m</a> toolbox     : <a href="#">\images\images\label2rgb.m</a> toolbox     : <a href="#">\images\images\wiener2.m</a> current dir : <a href="#">draw_LogicalOnImage</a> current dir : <a href="#">morpho_modify_0712</a> current dir : <a href="#">remove_boundaries</a> current dir : <a href="#">shadows_detection</a> current dir : <a href="#">zeroElimMedianHoleFill</a> </pre>
<a href="#">test_hole_fix</a>	<pre> unknown     : <a href="#">Id_o</a> unknown     : <a href="#">L00</a> toolbox     : <a href="#">\images\images\bwconncomp.m</a> toolbox     :? Multiple class methods match <a href="#">bwlabel.m</a> toolbox     :? Multiple class methods match <a href="#">bwmorph.m</a> </pre>
<a href="#">test_hole_fix2</a>	<pre> unknown     : <a href="#">Id_o</a> unknown     : <a href="#">Iq</a> toolbox     : <a href="#">\images\images\bwboundaries.m</a> toolbox     :? Multiple class methods match <a href="#">bwmorph.m</a> current dir : <a href="#">drawedgelist</a> current dir : <a href="#">edgelist</a> </pre>
<a href="#">test_hole_fix3</a>	<pre> toolbox     : <a href="#">\images\images\bwboundaries.m</a> toolbox     :? Multiple class methods match <a href="#">bwmorph.m</a> toolbox     : <a href="#">\images\images\label2rgb.m</a> unknown     : <a href="#">cc</a> </pre>
<a href="#">test_hole_fix_loop</a>	<pre> unknown     : <a href="#">Id_o</a> toolbox     : <a href="#">\images\images\bwconncomp.m</a> toolbox     :? Multiple class methods match <a href="#">bwlabel.m</a> toolbox     :? Multiple class methods match <a href="#">bwmorph.m</a> toolbox     : <a href="#">\images\images\label2rgb.m</a> </pre>
<a href="#">test_imp2_2019</a>	<pre> unknown     : <a href="#">BW20</a> unknown     : <a href="#">BW30</a> unknown     : <a href="#">I_obj_new</a> unknown     : <a href="#">I_ref</a> unknown     : <a href="#">Ic</a> unknown     : <a href="#">Id</a> unknown     : <a href="#">Id_o</a> toolbox     : <a href="#">\images\images\bfscore.m</a> toolbox     : <a href="#">\images\images\label2rgb.m</a> current dir : <a href="#">fun_DepthEdgeExtractor</a> current dir : <a href="#">fun_Surf2Obj</a> current dir : <a href="#">fun_SurfEdgeType</a> current dir : <a href="#">fun_SurfSegProcess</a> current dir : <a href="#">func_refToObj</a> current dir : <a href="#">manus_initial_parameters</a> current dir : <a href="#">morpho_modify_0712</a> </pre>
<a href="#">test_line_merg_extension</a>	<pre> current dir : <a href="#">DrawLineFeature</a> unknown     : <a href="#">Gdir</a> unknown     : <a href="#">Ic</a> unknown     : <a href="#">Id</a> toolbox     : <a href="#">\images\images\roipoly.m</a> current dir : <a href="#">make_RecMask</a> </pre>
<a href="#">test_overlapRegions</a>	<pre> unknown     : <a href="#">Ic</a> unknown     : <a href="#">Id</a> unknown     : <a href="#">ListPair</a> toolbox     : <a href="#">\images\images\imoverlay.m</a> toolbox     : <a href="#">\images\images\roipoly.m</a> </pre>
<a href="#">test_pair2surface</a>	<pre> unknown     : <a href="#">Id</a> unknown     : <a href="#">ListPair</a> toolbox     : <a href="#">\images\images\label2idx.m</a> toolbox     : <a href="#">\images\images\roipoly.m</a> toolbox     : <a href="#">\simulink\simulink\sl.m</a> </pre>
<a href="#">transform_pcl_1025</a>	<pre> current dir : <a href="#">depthToCloud_v2</a> </pre>

<a href="#">user_selection</a>	toolbox : <a href="#">\images\imuitools\imfreehand.m</a>
	unknown : <a href="#">createMask</a>
	current dir : <a href="#">draw_Spairs</a>
	Java method : 2 Multiple class methods match <a href="#">wait.m</a>
<hr/>	
<a href="#">zeroElimMedianFilter</a>	current dir : <a href="#">hhf_pad</a>
<hr/>	
<a href="#">zeroElimMedianHoleFill</a>	current dir : <a href="#">zeroElimMedianFilter</a>

## APPENDIX B: PUBLICATIONS

- A. Jabalameli, N. Etehad, and A. Behal, “Near Real-Time Robotic Grasping of Novel Objects in Cluttered Scenes,” SAI Computer Vision Conference (CVC), Las Vegas, NV, April 2019, accepted, to appear.
- A. Jabalameli and A. Behal, “Edge-Based Recognition of Novel Objects for Robotic Grasping”, Submitted to International Journal of Computer Vision.
- E. L. Parkhurst and M. A. Rupp and A. Jabalameli and A. Behal and J. A. Smither, “Compensations for an Assistive Robotic Interface”, Proceedings of the Human Factors and Ergonomics Society Annual Meeting, Oct 2017.
- S. Manaffam and A. Jabalameli, ”RF-localize: An RFID-based localization algorithm for Internet-of-Things,” 2016 Annual IEEE Systems Conference (SysCon), Orlando, FL, 2016, pp. 1-5. doi: 10.1109/SYSCON.2016.7490643
- R. Rahmatizadeh, P. Abolghasemi, A. Jabalameli, A. Behal, and L. Bölöni. “Trajectory adaptation of robot arms for head-pose dependent assistive tasks”. In Proc. of the 29th International FLAIRS Conference, May 2016.
- A. Jabalameli and A. Behal, ”System Identification of a Multi-timescale Adaptive Threshold Neuronal Model”, CsRO, 2018. [Online]. Available: arXiv: 1803.04236 [q-bio. NC], Feb 2018.
- A. Jabalameli and A. Behal, “A constrained linear approach to identify a multi-timescale adaptive threshold neuronal model”, IEEE 5th International Conference on Computational Advances in Bio and Medical Sciences (ICCABS), Oct 2015.



## LIST OF REFERENCES

- [1] A. Jabalameli, N. Ettehad, and A. Behal, "Near Real-Time Robotic Grasping of Novel Objects in Cluttered Scenes," *SAI Computer Vision Conference (CVC)*, Las Vegas, NV, April 2019, accepted, to appear.
- [2] V. D. Nguyen, "Constructing force-closure grasps," *Proceedings of the 1986 IEEE International Conference on Robotics and Automation*, pp. 1368-1373, 1986.
- [3] J. Canny, "A Computational Approach to Edge Detection," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, no. 6, pp. 679-698, Nov. 1986.
- [4] Y. C. Park and G. P. Starr, "Grasp Synthesis of Polygonal Objects Using a Three-Fingered Robot Hand," *Proceedings of the International Journal of Robotics Research*, vol. 11, no. 3, pp. 163-184, 1992.
- [5] Richard M. Murray, S. Shankar Sastry, and Li Zexiang, *A Mathematical Introduction to Robotic Manipulation (1st ed.)*, CRC Press, Inc., Boca Raton, FL, USA, 1994.
- [6] W. S. Howard and V. Kumar, "On the stability of grasped objects," in *IEEE Transactions on Robotics and Automation*, vol. 12, no. 6, pp. 904-917, Dec 1996.
- [7] K. Shimoga, "Robot grasp synthesis algorithms: A survey," *Int. J. Robot. Res.*, vol. 15, no. 3, pp. 230-266, 1996.
- [8] A. M. Okamura, N. Smaby and M. R. Cutkosky, "An overview of dexterous manipulation," *Proceedings of the IEEE Conference on Robotics and Automation*, San Francisco, CA, 2000, vol. 1, pp. 255-262, 2000.

- [9] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," *Proceedings of the 2000 ICRA Millennium Conference, Symposia Proceedings* (Cat. No.00CH37065), San Francisco, CA, vol. 1, pp. 348-353, 2000.
- [10] O. Carmichael and M. Hebert, "Shape-based recognition of wiry objects," *Proceedings of the 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. II-401-8 vol.2. doi: 10.1109/CVPR.2003.1211496, 2003.
- [11] A. T. Miller, S. Knoop, H. I. Christensen, and P. K. Allen, "Automatic grasp planning using shape primitives," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 1824–1829, 2003.
- [12] E. Chinellato, A. Morales, R. B. Fisher and A. P. del Pobil, "Visual quality measures for Characterizing Planar robot grasps," in *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 35, no. 1, pp. 30-41, Feb. 2005.
- [13] K. Hübner and D. Kragic, "Selection of robot pre-grasps using box-based shape approximation," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 1765-1770, 2008.
- [14] A. Saxena, J. Driemeyer, and A. Y. Ng, "Robotic grasping of novel objects using vision," *Int. J. Robot. Res.*, vol. 27, no. 2, pp. 157-173, Feb. 2008.
- [15] C. Dunes, E. Marchand, C. Colloret, and C. Leroux, "Active rough shape estimation of unknown objects," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, 2008, pp. 3622–3627
- [16] R. Detry, N. Pugeault and J. H. Piater, "A Probabilistic Framework for 3D Visual Object Representation," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 10, pp. 1790-1803, Oct. 2009
- [17] M. Ciocarlie and P. Allen, "Hand posture subspaces for dexterous robotic grasping," *Int. J. Robot. Res.*, vol. 28, pp. 851-867, Jul. 2009.

- [18] K. Hsiao, S. Chitta, M. Ciocarlie, and E. G. Jones, "Contact-reactive grasping of objects with partial shape information," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, Taipei, Taiwan, Oct. 2010, pp. 1228-1235.
- [19] R. Diankov, *Automated construction of robotic manipulation programs*, Ph.D. dissertation, Robotics Inst., Carnegie Mellon Univ., Pittsburgh, PA, USA, Aug. 2010.
- [20] M.A. Fischler and R.C. Bolles, "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography," *Communications of the ACM*, vol. 24, no. 6, pp. 381-395, June 1981.
- [21] R. Balasubramanian, L. Xu, P. D. Brook, J. R. Smith and Y. Matsuoka, "Human-guided grasp measures improve grasp robustness on physical robot," *2010 IEEE International Conference on Robotics and Automation*, Anchorage, AK, pp. 2294-2301, 2010.
- [22] M. Przybylski, T. Asfour, and R. Dillmann, "Planning grasps for robotic hands using a novel object representation based on the medial axis transform," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, pp. 1781-1788, Sep. 2011.
- [23] R. Detry, D. Kraft, O. Kroemer, *et al.*, *Learning grasp affordance densities*, Paladyn, 2011.
- [24] J. Weisz and P. K. Allen, "Pose error robust grasping from contact wrench space metrics," in *Proc. IEEE Int. Conf. Robot. Autom.*, pp. 557-562, 2012.
- [25] A. Sahbani, S. El-Khoury, P. Bidaud, "An overview of 3D object grasp synthesis algorithms," *Robotics and Autonomous Systems*, vol. 60, no. 3, pp. 326-336, March 2012.
- [26] K. Khoshelham and S.O. Elberink, "Accuracy and resolution of kinect depth data for indoor mapping applications," *Sensors*, vol. 12, pp. 1437-1454; doi:10.3390/s120201437, 2012.

- [27] A. Richtsfeld, T. Mörwald, J. Prankl, M. Zillich and M. Vincze, “Segmentation of unknown objects in indoor environments,” *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Vilamoura, pp. 4791-4796, 2012.
- [28] A. Jabalameli, *Sample Grasping Execution*, available at <https://youtu.be/7r0KaxpQk6E>
- [29] Brian Hudson, *Emphasis on Filtering & Depth Map Occlusion Filling*, Clarkson University Computer Vision Course CS 611, available online from <http://people.clarkson.edu/~hudsonb/courses/cs611/>, Fall 2012.
- [30] X. Gratal, J. Romero, J. Bohg, and D. Kragic, “Visual servoing on unknown objects,” *Mechatronics*, vol. 22, no. 4, pp. 423–435, 2012.
- [31] V. Lippiello, F. Ruggiero, B. Siciliano and L. Villani, “Visual Grasp Planning for Unknown Objects Using a Multifingered Robotic Hand,” in *IEEE/ASME Transactions on Mechatronics*, vol. 18, no. 3, pp. 1050-1059, June 2013.
- [32] G. Ardeshiri, H. Yazdani and A. Vosoughi, “Optimal Local Thresholds for Distributed Detection in Energy Harvesting Wireless Sensor Networks,” *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Anaheim, CA, USA, 2018, pp. 813-817.
- [33] H. Yazdani and A. Vosoughi, “On cognitive radio systems with directional antennas and imperfect spectrum sensing,” *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, New Orleans, LA, 2017, pp. 3589-3593.
- [34] A. Ückermann, R. Haschke and H. Ritter, “Realtime 3D segmentation for human-robot interaction,” *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Tokyo, pp. 2136-2143, 2013.
- [35] B. León, A. Morales, J. Sancho-Bru, *From Robot to Human Grasping Simulation*, Springer International Publishing, 2014

- [36] J. Bohg, A. Morales, T. Asfour and D. Kragic, "Data-Driven Grasp Synthesis—A Survey," in *IEEE Transactions on Robotics*, vol. 30, no. 2, pp. 289-309, April 2014.
- [37] A. ten Pas and R. Platt, "Using geometry to detect grasp poses in 3d point clouds," *Int'l Symp. on Robotics Research*, 2015.
- [38] S. Jain and B. Argall, "Grasp detection for assistive robotic manipulation," *2016 IEEE International Conference on Robotics and Automation (ICRA)*, Stockholm, pp. 2015-2021, 2016.
- [39] T. Suzuki and T. Oka, "Grasping of unknown objects on a planar surface using a single depth image," *2016 IEEE International Conference on Advanced Intelligent Mechatronics (AIM)*, Banff, AB, pp. 572-577, 2016.
- [40] G. Moyà-Alcover, A. Elgammal, A. Jaume-i-Capó, and J. Varona, "Modeling depth for nonparametric foreground segmentation using RGBD devices," *Pattern Recognition Letters*, available online, 21 September 2016.
- [41] N. Etehadhi and A. Behal, "Implementation of Feeding Task via Learning from Demonstration," *2018 Second IEEE International Conference on Robotic Computing (IRC)*, Laguna Hills, CA, 2018, pp. 274-277.
- [42] N. Etehadhi and A. Behal, "A learning from demonstration framework for implementation of a feeding task", *Encyclopedia with Semantic Computing and Robotic Intelligence* Vol. 02, No. 01, 1850001, June 2018.
- [43] M. Haghshenas, J. A. Wilson and R. Kumar, "Algebraic coupled level set-volume of fluid method for surface tension dominant two-phase flows", *International Journal of Multiphase Flow*, Volume 90, 2017, Pages 13-28, ISSN 0301-9322.

- [44] M. Haghshenas, J. A. Wilson, R. Kumar, "Finite volume Ghost Fluid Method implementation of interfacial forces in PISO loop", *Journal of Computational Physics*, V 376, 2019, Pages 20-27.
- [45] Z. Teng and J. Xiao, "Surface-Based Detection and 6-DoF Pose Estimation of 3-D Objects in Cluttered Scenes," in *IEEE Transactions on Robotics*, vol. 32, no. 6, pp. 1347-1361, Dec. 2016.
- [46] A. Stork, *Representation and Learning for Robotic Grasping, Caging, and Planning*, PhD dissertation, Stockholm, 2016.
- [47] P. D. Kovesi, *MATLAB and Octave Functions for Computer Vision and Image Processing*, available online from <http://www.peterkovesi.com/matlabfns>, 2000.
- [48] Do Carmo, P. Manfredo, *Differential Geometry of Curves and Surfaces: Revised and Updated Second Edition*. Courier Dover Publications, 2016.
- [49] Roa, Máximo A., and Raúl Suárez. "Grasp quality measures: review and performance." *Autonomous robots* 38.1 (2015): 65-88.
- [50] Chinellato, Eris, Robert B. Fisher, Antonio Morales, and Angel P. Del Pobil. "Ranking planar grasp configurations for a three-finger hand." In *ICRA*, pp. 1133-1138. 2003.