School of Computing
FACULTY OF ENGINEERING

UNIVERSITY OF LEEDS

The Robot Shopkeeper: An Integrated System for Robotic Manipulation and Perception

Kieran Marshall Haden

Submitted in accordance with the requirements for the degree of BSc Artificial Intelligence

Session 2015/2016

The candidate confirms that the following have been submitted.
<As an example>

Items	Format	Recipient(s) and Date
Deliverable 1, 2, 3	Report	SSO (DD/MM/YY)
Participant consent forms	Signed forms in envelop	SSO (DD/MM/YY)
Deliverable 4	Software codes or URL	Supervisor, Assessor (DD/MM/YY)
Deliverable 5	User manuals	Client, Supervisor (DD/MM/YY)

Type of project: Exploratory Software

The candidate confirms that the work submitted is their own and the appropriate credit has been given where reference has been made to the work of others.

I understand that failure to attribute material which is obtained from another source may be considered as plagiarism.

(Signature of Student) _____

Summary

<Concise statement of the problem you intended to solve and main achievements (no more than one A4 page)>

Acknowledgements

<The page should contain any acknowledgements to those who have assisted with your work. Where you have worked as part of a team, you should, where appropriate, reference to any contribution made by other to the project.>
Note that it is not acceptable to solicit assistance on 'proof reading' which is defined as the "the systematic checking and identification of errors in spelling, punctuation, grammar and sentence construction, formatting and layout in the test";
see http://www.leeds.ac.uk/gat/documents/policy/Proof-reading-policy.pdf.

Contents

Chapter 1

Introduction

1.1 Aim

To produce a piece of software using the robotic operating system (ROS) that will allow Baxter to interact with people and give them sweets within a sweet shop setting. The personal aim of this project is to learn the algorithms and software systems used in autonomous systems whilst doing this project. Practically, the aim is to allow the School of Computing to be able to use this system as an interactive demonstration on Open Days.

1.2 Objectives

1. Develop a vision system to allow Baxter to recognise sweets and the container they are in

   (a) Deliverables - Software to allow Baxter to be able to identify and locate a bowl and individual sweet shapes/colours (depending on the recognition method). A section on the report comparing vision methods and deciding the most appropriate one to use.

   (b) Provide a section in the results section of the report showing the successful recognition of the objects by Baxter visualized using rViz/images.

2. Develop an algorithm for the singulation of sweets

   (a) Deliverables - Code to allow Baxter to singulate specified sweets from a bowl. A section on the report comparing sweet singulation algorithms, determining which one was the most efficient to use.

   (b) Provide results from the sweet singulation algorithm running in different environments and scenarios.

3. Develop a method for Baxter to manipulate and count out change

   (a) Deliverables - code allowing Baxter to identify and recognise certain coins/notes and then being able to manipulate the money to grab a particular amount.

   (b) Provide results in the report from the money manipulation algorithm running in various environments.

4. Develop a method of Human-Interaction

   (a) Deliverables - code allowing Baxter to interact with a human customer and recognise some way of telling which sweets the human wants.

   (b) Provide evidence of different methods of robot-human interaction tested in the report and an analysis on which method works best.

1.3 Methodology

Deliverables to be included:

- A Github repository of working code to allow Baxter to serve sweets to a customer with instructions on how to set up. The Github should be separated into different software packages, for example: perception, manipulation and interaction.
- A working demo of the Robot Shopkeeper to be used at Open Days in the School of Computing, with a video of the working demo.
- An analysis of object recognition methods - using OpenCV or point cloud methods.
- An analysis of object manipulation methods for Baxter to grab and separate sweets into the specified type and amount.
- An analysis of human interaction methods, with a choice on which is best/appropriate to implement.
- For each analysis above, include referencing and comparisons to academic papers, that will be studied at the appropriate points in the project.
- Aside from analyses, results from the techniques used should be provided in an appropriate form, such as images/graphs and video recordings.

1.4 Tasks, milestones and timeline

Here are each of the tasks explained in more detail:

Project setup - Setup the ROS system and look learn the basics of the software. Complete ROS and Baxter tutorials and make some example programs to test and move Baxter in different ways.

Setup Kinect recognition - Setup the Kinect with Baxter and look into different computer vision methods to recognise the bowl and sweets and their positions in 3d space.

Sweet singulation - Trial multiple object manipulation algorithms to allow Baxter to grab and select specified sweet combinations.

Human interaction - Create some form of human interaction between Baxter and a human to allow the human to ask for specific sweets. This could be gestures or voice control based.

Money Manipulation - Develop some vision system to recognise paper money or change and then work on algorithms for Baxter to handle/count out money to give to a person.

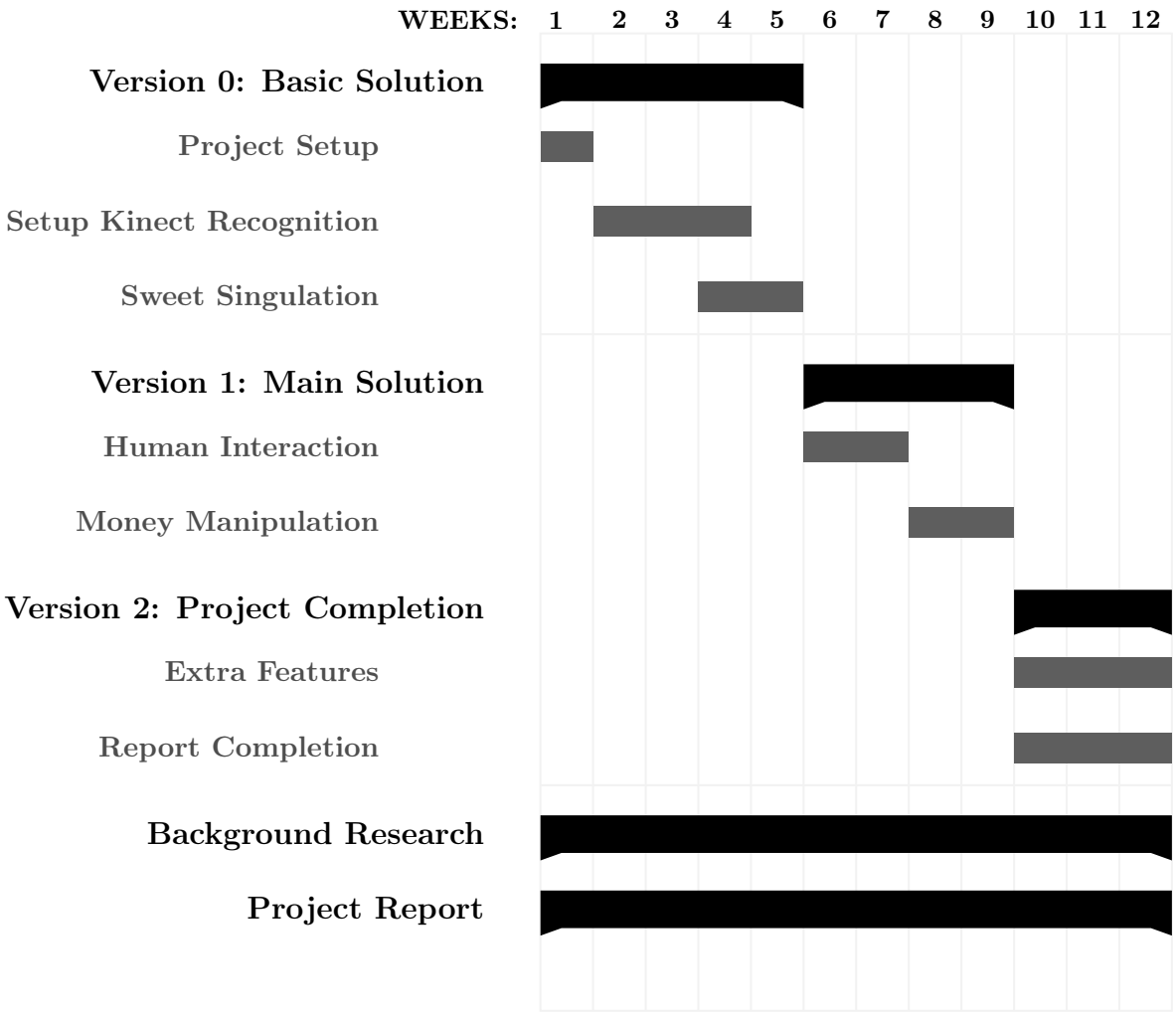
Extra Features - Allow time to further improve upon previous steps or pick one dedicated topic to further go into and develop.

Report Completion - Allow the last three weeks to further improve upon the project report.

Background Research - Throughout each stage of the project, make sure at least two academic papers have been researched, with summary paragraphs on each for use in the report.

Project Report - At each week, take a look at project report deadlines that need to be met. Make sure parts of the report are written at the time they are completed.

Here is a Gantt chart representing the order of the tasks for the project:


WEEKS: 1 2 3 4 5 6 7 8 9 10 11 12
Version 0: Basic Solution
Project Setup
Setup Kinect Recognition
Sweet Singulation
Version 1: Main Solution
Human Interaction
Money Manipulation
Version 2: Project Completion
Extra Features
Report Completion
Background Research
Project Report

Chapter 2

Background And Literature Review

2.1 Machine Vision

2.2 Robotic Behaviour

2.3 Human Interaction

Chapter 3

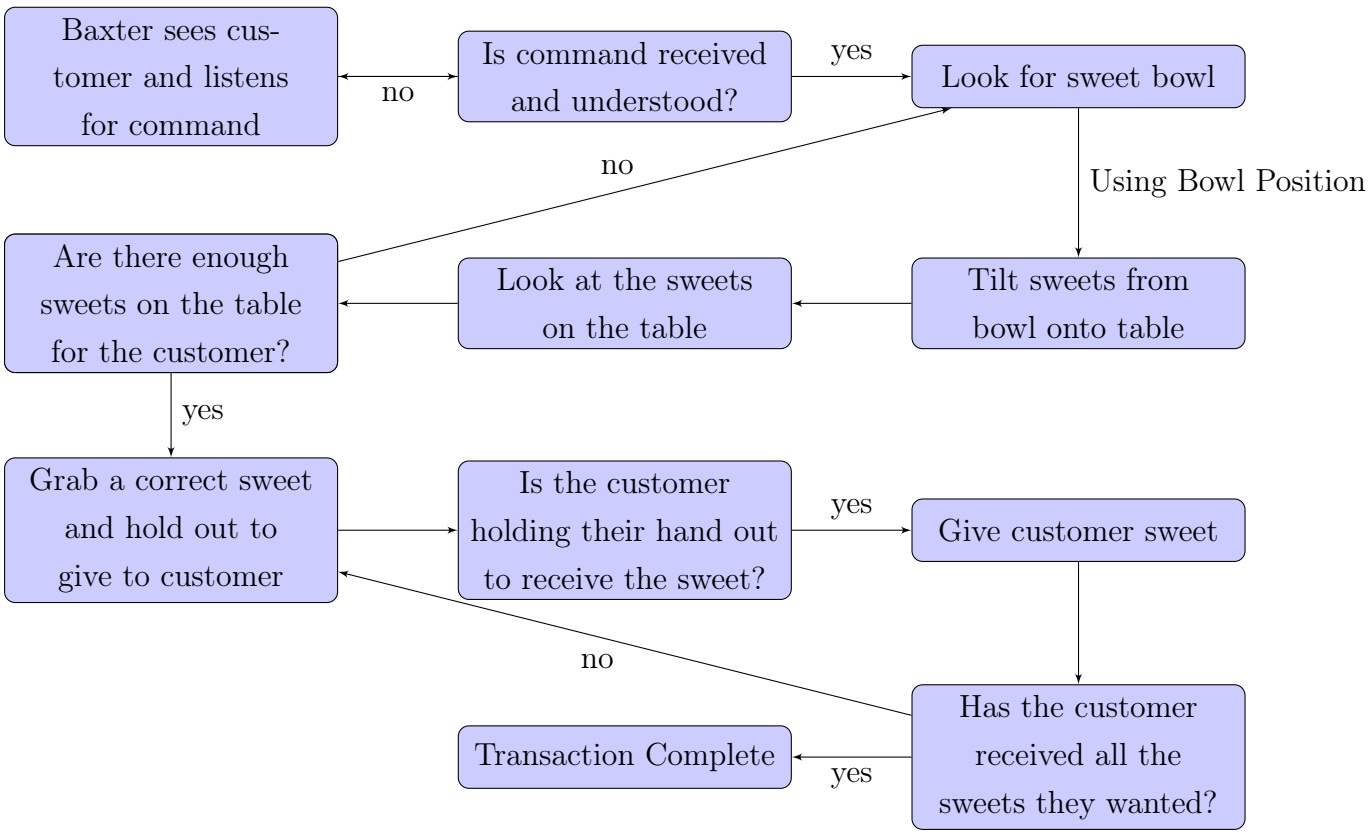
Materials and Logic

3.1 The Robot

3.2 Kinect

3.3 System Logic

To simplify the overall system logic, it was easier to look at the system as a whole and look at the simple interactions needed between a shopkeeper and their customer. Firstly, human interaction would determine what sweets the shopkeeper would need to get. Secondly, the shopkeeper would get some sweets from the bowl and individually pick out the ones the customer wanted. The shopkeeper would give those sweets to the customer and then ask for payment of some sort to complete the interaction. During the development of the project, more clear and precise logic steps came from development, shown in the flow chart below.



Chapter 4

Methods

4.1 Main ROS Principles

4.1.1 Inverse Kinematics

dsaokdaskdaskldaskl;dsad s d;asdaskldksal;das dsadsamdaskdkadklas;dsa lkjsakdsadjksajkdasjldas

4.1.2 Visualisation using RViz

dsaokdaskdaskldaskl;dsad s d;asdaskldksal;das dsadsamdaskdkadklas;dsa lkjsakdsadjksajkdasjldas

4.1.3 Fixed Frame Transforms

A key aspect of creating the integrated movement system was understanding the concept of fixed frames and transforms. In ROS, Baxter contains information for multiple transforms between every joint and sections of his body. These transforms are known at all times to help know where a point is relative to one part, say the hand, against Baxter's torso. This method works similarly for the Kinect, where Baxter's torso coordinate system, is linked to the Kinect's coordinate system via a transform, which can be accessed by certain methods in ROS.

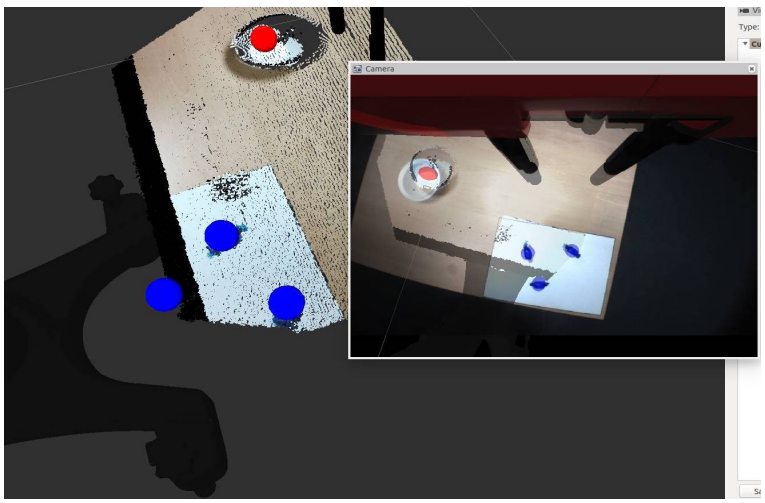


Figure 4.1: *Example RVIZ Image*

Fixed coordinate systems were constantly used in testing the vision systems in this project, mainly within rViz. Everytime a coordinate or shape was recognised within the vision system, they could be published to rViz using a PointStamped object, which can be published in a particular coordinate system. Therefore publishing in one frame, transforming then publishing it another, rViz can show whether a point is correctly detected and whether it is in the correct frame.

This principle was used in the bowl recognition system, where after the centre of the bowl was obtained, Baxter needed to know where the centre of the bowl was in it's main body coordinate system before a movement command could be made. The system then looked up the transform between the Kinect and Baxter's torso to transform the 3D coordinate between coordinate systems.

4.1.4 Custom Service Requests

The idea that Baxter's movements would all be based on the current states of his vision system - depending upon where both the sweets and the bowl were, there needed to be a method developed for Baxter's movement system to be able to request information from the vision system. This was implemented via Services, which is a method in ROS that allows a custom message to be sent, and then received between two nodes.

4.2 The Vision System

4.2.1 Calibration and Setup

4.2.2 Basic Vision Techniques

Throughout this section, there is going to be information on the complex vision techniques used to help Baxter identify the position of the bowl and the sweets on the table. Firstly, here is some background information on the simpler, smaller algorithms used with these:

- **RANSAC** - dsjadsjda
- **Contours** - dsjadsjda
- **Gaussian Blurring** - dsjadsjda

4.2.3 Bowl Recognition

Pointcloud Segmentation and Recognition

The Kinect produces a pointcloud, a set of 3D points in the Kinect's coordinates system. To recognise the bowl of sweets on the table, the vision sytem first separates objects from the points representing the table, then separates those points into individiual object clusters. To eliminate noise and find the bowl cluster, a colour segmentation is performed to find only the white objects on the table. After that, the bowl can then be found by looking for the rim of the bowl as a circle in 3D space. This process is carried out by multiple algorithms, as discussed here:

1. Tabletop Object Detection/Segmentation

This method works by detecting the main table plane in the Pointcloud by detecting the dominant plane using RANSAC. Points above this plane are then considered to be objects on top of the table, which are then segmented from the points within the table plane. This results in a pointcloud devoid of any points belonging to the table.

2. Region Growing Segmentation

The purpose of this algorithm is to separate the points in the pointcloud into clusters, ones which are close enough to separate into individual objects on the table. The theory behind this algorithm is it takes in the indices and estimated normals of the Pointcloud and for each possible region, calculates a K-nearest neighbour search over the indices. During this search, it compares each point with another, checking first for a specified smoothness constraint, checking if the deviation between normals is within a specified angle. If that is satisfied, then the difference in curvature is tested, allocating the points to the appropriately separated clusters.

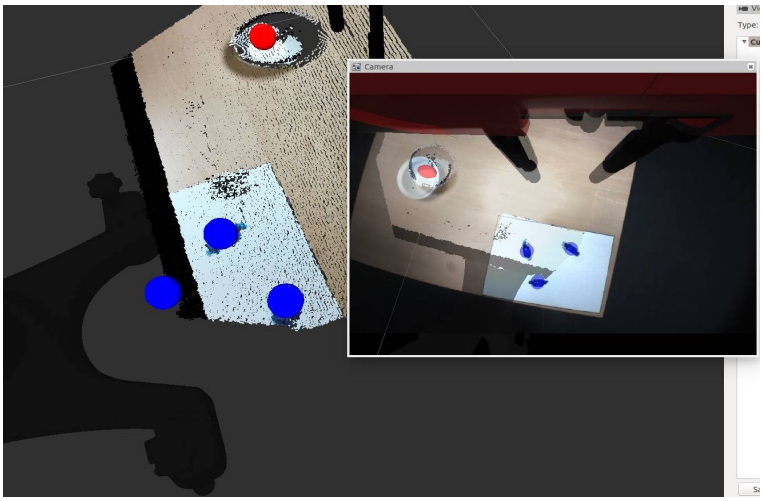


Figure 4.2: *An example of segmented objects*

During the implementation of this algorithm, multiple number of neighbours, smoothness and curvature constraints were used until the individual objects in the pointcloud were sufficiently separated.

3. Colour-Based Segmentation

After separation of the Pointcloud into object clusters, a key thing to do was to get rid of noise caused by movement of Baxter's arms in front of the Kinect.

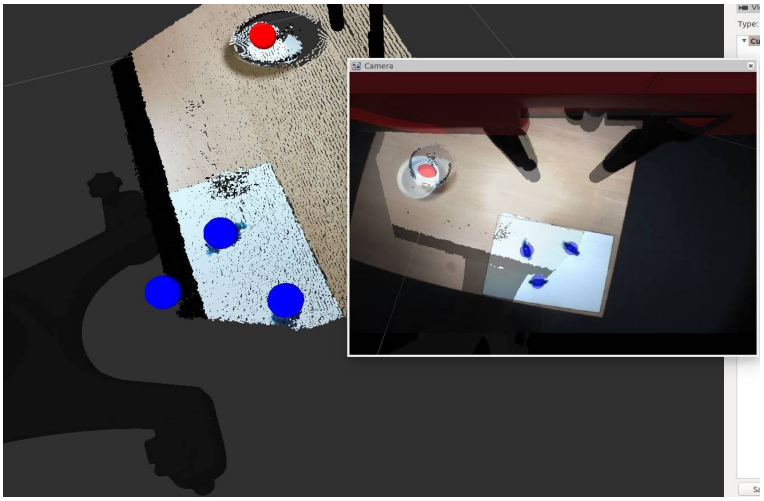


Figure 4.3: *An example of white colour segmented objects*

To do this, black noise/points were eliminated by segmenting the objects further by colour. This was done by looping over the points in each cluster and averaging the RGB values of each cluster. Then, only the white clusters (those with a high enough R, G and B threshold), were segmented out so that the Pointcloud contains only the points for the bowl and any other white object on the table. This method could be expanded to recognise other colour bowl relatively easily, by finding the correct thresholds for other colour bowls.

4. Detection of the Bowl Rim

The most significant part of the bowl that tended to show up in the PointCloud was the bowl's rim. From that information, it was decided that the simplest solution was to find the rim of the bowl by finding a specific sized circle within a plane of the 3D PointCloud. This method was implemented by using the RANSAC algorithm to detect the points that best fit a circle shape within any plane. This method in the PCL library is especially good at allowing variation in it's detection. By varying distance thresholds and minimum/maximum circle radius values, the bowl was able to be found successfully, even with gaps in the rim, which occurred when Baxter's hands went in front of the bowl. After the circle model had been found in the white PointCloud, the x, y z of the centre of the bowl's rim was then known in the Kinect's frame coordinates.

5. Improvement on Detection

Due to the noise in the Kinect's Pointcloud recordings, some improvements were made further into development. The problem with the circle detection was that noise caused the centre of the bowl to shift each frame. To counteract the noise in detection, a cumulative average was taken over 20 frames and outputted as an average bowl centre. Then a cumulative centre point was taken further over time to de-noise the results, resulting in a fixed bowl centre that Baxter could reliably use as the actual centre of the bowl on the table.

4.2.4 Sweet Recognition

The task of Baxter recognising the sweets involved him being able to look at an area of the table with sweets retrieved from the bowl and determine how many sweets there were in that area along with what type/colour each sweet was. The problem with recognising the sweets using the Kinect is that the sweets were such small objects, that the noise in the Kinect meant that a significant number of sweets weren't picked up after segmenting objects on the table. Therefore an alternative method was proposed using OpenCV. The idea behind this method was for Baxter to capture an image of the table with the sweets on using his hand camera. Then from that image, OpenCV image processing techniques could be applied to separate the sweets into individual objects, from which the shapes, centres and colours could be obtained.

The first task in separating the sweets was to have Baxter to only look in the area the sweets were placed on the table, so the sweets remaining in the bowl would not interfere. It was decided that the easiest way to segment this area out was to use a white piece of paper as the background, making it easier to segment out the rest of the image. The piece of paper used was A3 in size and had a black edge drawn on, to help detect the border of the page. Multiple vision methods were trialled to try and recognise this sweet placement area, explained here:

Image Pre-processing

Before the image from the camera could first be used, some pre-processing was taken place to reduce noise and make it easier to detect the area on the table. A Gaussian Blur was performed with a small kernel to firstly blur the image to reduce possible noise. Then, Canny Edge Detection was used to detect edges within the image, along with dilation and erosion to close any incomplete edges, resulting in a reasonably successful edge detection for the entire image.

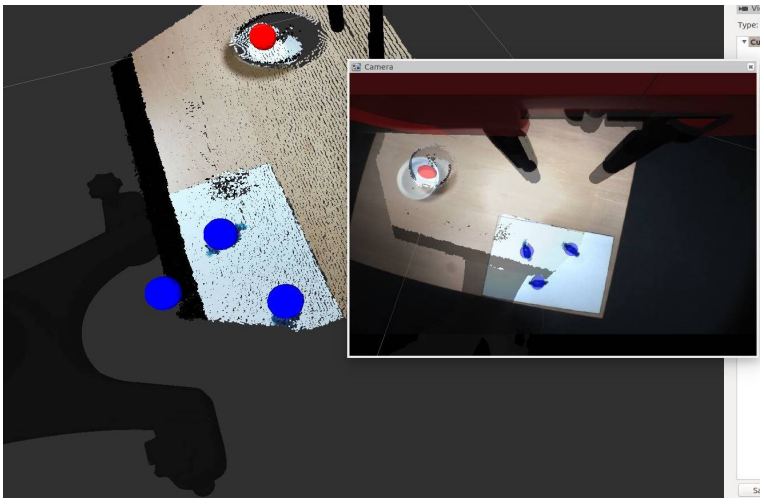


Figure 4.4: *An example of segmented objects*

Hough Line Transform

Firstly, to find a rectangle in the image, Hough Line Transform was attempted to be used to find the individual edges of the paper. This technique was somewhat successful when attempting to distinguish between the edges however, by varying and optimising the line detection threshold, it was difficult to detect all four edges of the paper constantly. Either one or two edges kept being detected then undetected or too many edges were found. Due to the inaccuracy of this edge detection, the four edges could not always be found to form the rectangle. Therefore other methods were explored.

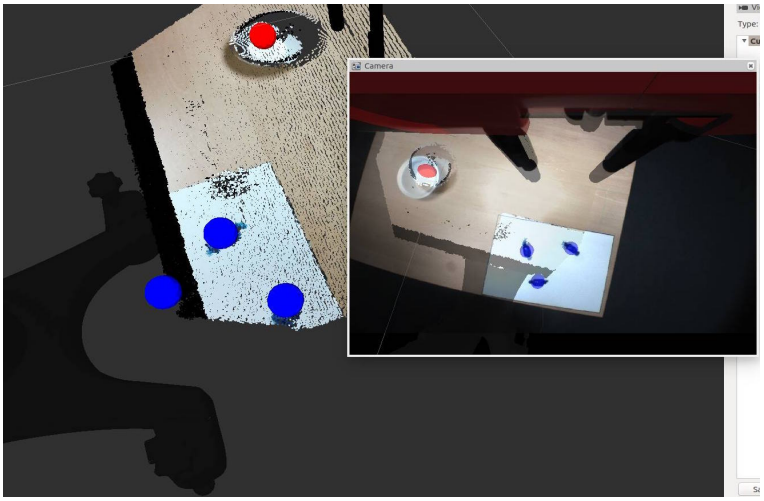


Figure 4.5: *An example of segmented objects*

Contour Detection

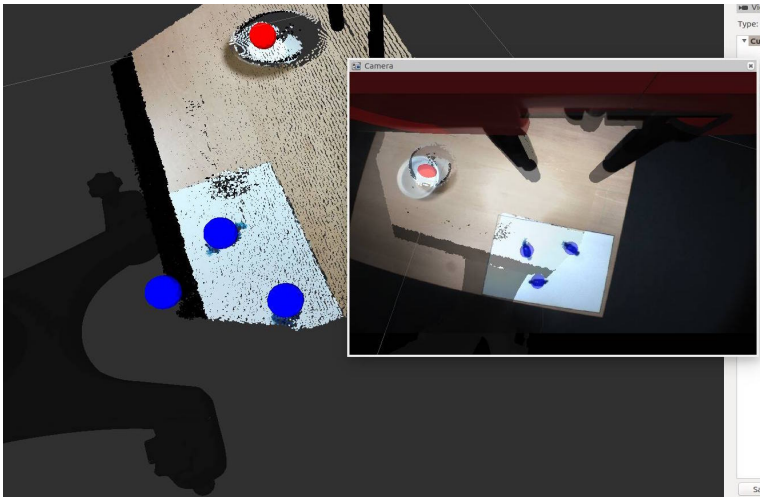


Figure 4.6: *An example of white colour segmented objects*

Once the black border on the area was defined enough to be consistent during canny edge detection, a contour successfully managed to detect the four edges of the paper. Due to there being many contours detected in the image, certain constraints had to be put on the detection to segment out the sweet area. The main method of segmenting out the rectangular area then was to first eliminate the smaller contours by size (using the in-built countourArea) and then approximate a polygon for the countours to detect which countour was a rectangle. This contour then produced a mask, which could segment out the rest of the image from the sweet area.

Once the sweet placement area was reliably segmented out from the image, multiple

methods were used to attempt to try and identify the individual sweets. These methods are explained below:

Hough Circle Transform

For the simple, round sweets initially used in trials, a Hough Circle detection algorithm seemed like a sensible way to detect the simpler, round sweets. However, like the Hough Line detection used earlier, it was hard to get a constant solution, with circles being detected and then undetected throughout various received image frames, therefore other methods needed to be tried to get a more reliable vision system.

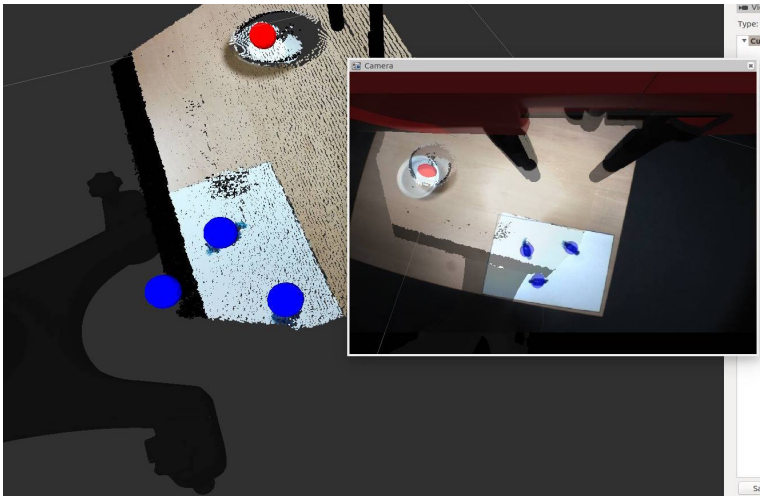


Figure 4.7: *An example of segmented objects*

Contour Detection

A better method was to do some image processing, Gaussian blurring, closing and opening to produce a lot better results, producing a mask of a white background with some black sweets in front. The problem with using this method is due to reflections on the sweets wrappers, this caused an issue of multiple broken contours within an individual sweet, whereas a preferred method would have been to capture the whole sweet with one contour, as Baxter needs to be able to count each sweet once.

HSV Colour Segmentation with Contour Detection

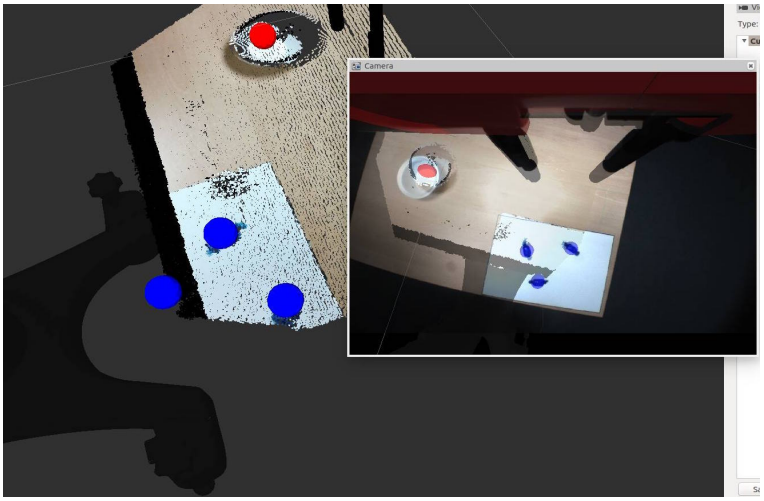


Figure 4.8: *An example of white colour segmented objects*

A more accurate method was found by using an existing tool called objectfinder. This tool uses multiple sliders to segment an image and produce a mask using lower and upper bounds for RGB values. Since each sweet wrapper had different identifiable colours and they were placed onto a separatable white background, using a scale of possible HSV values could be used to identify main sweet wrapper colours - blue, green, red etc. The only limitation of this approach was that similar colours under light could be mixed up, for example, red and pink wrappers had overlapping HSV RBG ranges, and therefore could not be both separated and identified by this method. It did however result in very clear contours for the significantly different colours so this method was used for the sweet recognition. Possible improvements could be made with shape recognition then colour analysis, which may have been able to identify a typical sweet shape and then separate by colour after.

Now there was a method for the sweets to be detected, by extracting moments from the sweet contours, the 2D pixel coordinate could be retrieved from the 2D image. The problem then was converting the 2D points in that image into 3D world coordinates. This was done using an altered version of a pinhole camera method, where the u, v coordinate within the 2D image could be converted to a 3D world coordinate using Baxter's in-built calibrated camera matrix. The equation uses the x and y camera offset values, with the focal lengths to scale the initial point. Then using the distance from the camera to the table (which is fixed), the points can then be converted into 3D world coordinates. This was tested using rViz and Baxter to make sure it worked.

4.2.5 Sweet Singulation

4.3 The Manipulation System

4.3.1 Manipulating the Bowl

4.3.2 Sweet Manipulation

4.4 Human Interaction

4.4.1 Voice Recognition

4.4.2 Hand Recognition

Pages to be referenced for practical applications:
http://docs.opencv.org/3.0-beta/doc/py_tutorials/py_imgproc/py_houghlines/py_houghlines.html#py-hough-lines http://docs.opencv.org/2.4/modules/calib3d/doc/camera_calibration_and_3d_reconstruction.html http://pointclouds.org/documentation/tutorials/region_growing_segmentation.php
http://pointclouds.org/documentation/tutorials/region_growing_rgb_segmentation.php
http://wiki.ros.org/tabletop_object_detector
http://docs.pointclouds.org/trunk/group__sample__consensus.html

[3] [2] [4] [1]

References

[1] M. Ciocarlie, K. Hsiao, E. G. Jones, S. Chitta, R. B. Rusu, and I. A. Şucan. *Experimental Robotics: The 12th International Symposium on Experimental Robotics*, chapter Towards Reliable Grasping and Manipulation in Household Environments, pages 241–252. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[2] K. Lai, L. Bo, X. Ren, and D. Fox. Detection-based object labeling in 3d scenes. In *Robotics and Automation (ICRA), 2012 IEEE International Conference on*, pages 1330–1337, May 2012.

[3] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohi, J. Shotton, S. Hodges, and A. Fitzgibbon. Kinectfusion: Real-time dense surface mapping and tracking. In *Mixed and Augmented Reality (ISMAR), 2011 10th IEEE International Symposium on*, pages 127–136, Oct 2011.

[4] S. S. Srinivasa, D. Berenson, M. Cakmak, A. Collet, M. R. Dogar, A. D. Dragan, R. A. Knepper, T. Niemueller, K. Strabala, M. V. Weghe, and J. Ziegler. Herb 2.0: Lessons learned from developing a mobile manipulator for the home. *Proceedings of the IEEE*, 100(8):2410–2428, Aug 2012.

Appendices

Appendix A

External Material

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Appendix B

Ethical Issues Addressed