

# OBJECT SEGMENTATION METHODS FOR ONLINE MODEL ACQUISITION TO GUIDE ROBOTIC GRASPING

by

Dmitri Ignakov  
Bachelor of Aerospace Engineering  
Ryerson University, 2007

A dissertation presented to Ryerson University  
in partial fulfillment of the requirements  
for the degree of

Doctor of Philosophy  
in the Program of  
Aerospace Engineering

Toronto, Ontario, Canada, 2013  
©Dmitri Ignakov 2013



I hereby declare that I am the sole author of this dissertation. This is a true copy of the dissertation, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this dissertation to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this dissertation by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my dissertation may be made electronically available to the public.





# **Object Segmentation Methods for Online Model Acquisition to Guide Robotic Grasping**

Dmitri Ignakov, Doctor of Philosophy, Aerospace Engineering, 2013  
Ryerson University

A vision system is an integral component of many autonomous robots. It enables the robot to perform essential tasks such as mapping, localization, or path planning. A vision system also assists with guiding the robot's grasping and manipulation tasks. As an increased demand is placed on service robots to operate in uncontrolled environments, advanced vision systems must be created that can function effectively in visually complex and cluttered settings.

This thesis presents the development of segmentation algorithms to assist in online model acquisition for guiding robotic manipulation tasks. Specifically, the focus is placed on localizing door handles to assist in robotic door opening, and on acquiring partial object models to guide robotic grasping.

First, a method for localizing a door handle of unknown geometry based on a proposed 3D segmentation method is presented. Following segmentation, localization is performed by fitting a simple box model to the segmented handle. The proposed method functions without requiring assumptions about the appearance of the handle or the door, and without a geometric model of the handle.

Next, an object segmentation algorithm is developed, which combines multiple appearance (intensity and texture) and geometric (depth and curvature) cues. The algorithm is able to segment objects without utilizing any a priori appearance or geometric information in visually complex and cluttered environments. The segmentation method is based on the Conditional Random Fields (CRF) framework, and the graph cuts energy minimization technique. A simple and efficient method for initializing the proposed algorithm which overcomes graph cuts' reliance on user interaction is also developed.

Finally, an improved segmentation algorithm is developed which incorporates a distance metric learning (DML) step as a means of weighing various appearance and geometric segmentation cues, allowing the method to better adapt to the available data. The improved method also models the distribution of 3D points in space as a distribution of algebraic distances from an ellipsoid fitted to the object, improving the method's ability to predict which points are likely to belong to the object or the background.

Experimental validation of all methods is performed. Each method is evaluated in a realistic setting, utilizing scenarios of various complexities. Experimental results have

demonstrated the effectiveness of the handle localization method, and the object segmentation methods.

## Acknowledgements

First, I would like to thank my supervisor Dr. Guangjun Liu, without whose support, encouragement and guidance this thesis would not have been possible. He has always encouraged me to aim higher, and for that I am grateful. I would also like to thank my previous supervisor Dr. Galina Okouneva, who continued to provide guidance even after she moved on from the University. Thank you. I would also like to thank Dr. Donald McTavish. It is because of Dr. McTavish's Spacecraft Attitude Dynamics course that I ended up where I am today.

I would like to thank my examining committee consisting of Dr. John Enright, Dr. Matthew Kyan, Dr. David Greatrix, and Dr. Zheng Hong Zhu. Their insightful comments greatly improved the quality of this thesis. In particular, I would like to thank Dr. Enright for many helpful discussions, especially in the final few months.

During my studies at Ryerson University I have made many close friends. I would like to thank Tom Dzamba and Geoff McVittie for not attempting to keep me sane over these years, for the coffee breaks, for the random discussions, and most importantly, for making the lab worth coming to every day. I would also like to thank Tom in particular for increasing the “entropy” of my desk, and for the antics that usually followed. It seems appropriate that we are all graduating together. Thank you, my friends. I would also like to mention Chris Fernando, whose presence has made coming to the lab much more enjoyable. Marcella, Brendon, Martin; thank you for putting up with my occasional ranting. I'm glad I got to know each of you.

Some of my friends have already moved on from Ryerson. Marcin Kurillo, with whom I began my graduate career, Bryan Stuurman, Daren Lee, Kam Shahid, and Godard. You guys are missed.

To my friends outside of the University: Andrew, Andrew, Antonio, Khaled, Mike, Nina, Silvee, Stan, Vlad. Thank you for your support, and for your patience with me over the last several years.

Finally, I would like to thank my family. Without you I could not have done this. You were always there for me, and nothing I write here can adequately express how I feel.



# Dedication

To my family.



## List of Publications

### Journal papers:

Ignakov, D., Okouneva, G., and Liu, G. Localization of a door handle of unknown geometry using a single camera for door-opening with a mobile manipulator. *Autonomous Robots*, 33(4):415–426, 2012.

Ignakov, D., and Liu G. Object segmentation in cluttered and visually complex environments. *Autonomous Robots*, revisions submitted June 28, 2013.

Ignakov, D., and Liu G. Distance Metric Learning for Object Segmentation. Under preparation.

### Conference papers:

Mark, L.H., Okouneva, G., Saint-Cyr, P., Ignakov D., and English C. Near-optimal selection of views and surface regions for ICP pose estimation. In *Proceedings of the 6th International Symposium on Visual Computing*, pages 53–63, November 2010.

Ignakov D., Okouneva, G., and Liu G. Localization of door handle using a single camera on a door opening mobile manipulator. In *Proceedings of 2nd IEEE Symposium on Computational Intelligence for Security and Defence Applications*, pages 1–7, July 2009.





# Contents

|  | <b>Page</b> |
|--|-------------|
| <i>Declaration</i> . . . . .   | iii         |
| <i>Abstract</i> . . . . .  | v           |
| <i>Acknowledgements</i> . . . . .  | vii         |
| <i>Dedication</i> . . . . .  | ix          |
| <i>List of Publications</i> . . . . .  | xi          |
| <i>List of Figures</i> . . . . .   | xvii        |
| <i>List of Tables</i> . . . . .  | xxi         |
| <i>List of Abbreviations</i> . . . . .                                       | xxiii       |
| <i>Nomenclature</i> . . . . .  | xxv         |
| <b>1 Introduction</b>  | <b>1</b>    |
| 1.1 Motivation . . . . .   | 1           |
| 1.1.1 Motivation for Door Handle Localization . . . . .                      | 2           |
| 1.1.2 Motivation for Online Model Acquisition . . . . .                      | 3           |
| 1.1.3 Segmentation as a Tool for Online Model Acquisition . . . . .          | 3           |
| 1.2 Literature Review . . . . .  | 5           |
| 1.2.1 Review of Object Segmentation Methods . . . . .                        | 6           |
| 1.2.2 Review of Door Handle Localization Methods . . . . .                   | 16          |
| 1.2.3 Object Segmentation in Online Model Acquisition for Grasping . . . . . | 18          |
| 1.2.4 Distance Metric Learning . . . . .                                     | 23          |

|          |  |           |
|----------|--|-----------|
| 1.3      | Problem Statement . . . . .  | 25        |
| 1.3.1    | Segmentation for Door Handle Localization . . . . .                              | 25        |
| 1.3.2    | Object Segmentation for Model Acquisition to Guide<br>Robotic Grasping . . . . . | 27        |
| 1.4      | Contributions . . . . .  | 28        |
| 1.5      | Outline . . . . .  | 30        |
| <b>2</b> | <b>3D Segmentation for Door Handle Localization</b>                              | <b>33</b> |
| 2.1      | Algorithm Overview . . . . .   | 33        |
| 2.2      | 3D Data Acquisition . . . . .  | 36        |
| 2.3      | Handle Segmentation and Localization . . . . .                                   | 40        |
| 2.4      | Experimental Validation . . . . .  | 48        |
| 2.4.1    | Experimental Setup . . . . .   | 48        |
| 2.4.2    | Experimental Results . . . . .   | 50        |
| 2.5      | Conclusions and Discussion . . . . .   | 53        |
| <b>3</b> | <b>Object Segmentation</b>   | <b>63</b> |
| 3.1      | The Proposed Segmentation Algorithm . . . . .                                    | 64        |
| 3.2      | Markov and Conditional Random Fields in Image Segmentation . . . . .             | 66        |
| 3.2.1    | Segmentation as a Labelling Problem . . . . .                                    | 67        |
| 3.2.2    | Random Field Models . . . . .  | 67        |
| 3.2.3    | Inference Using Graph Cuts . . . . .   | 75        |
| 3.3      | Initial Object Detection . . . . .   | 80        |
| 3.4      | The <i>Unary</i> Potential . . . . .   | 82        |
| 3.4.1    | Intensity / Colour . . . . .   | 84        |
| 3.4.2    | Texture . . . . .  | 85        |
| 3.4.3    | Depth . . . . .  | 86        |

|          |   |            |
|----------|---|------------|
| 3.4.4    | Curvature . . . . .                                       | 87         |
| 3.5      | The <i>Pairwise</i> Potential . . . . .                   | 88         |
| 3.6      | Parameter Learning . . . . .                              | 91         |
| 3.7      | Experimental Results . . . . .                            | 93         |
| 3.7.1    | Object Detection . . . . .                                | 95         |
| 3.7.2    | Segmentation . . . . .                                    | 98         |
| 3.7.3    | Comparison with Prior Work . . . . .                      | 101        |
| 3.8      | Conclusions and Discussion . . . . .                      | 106        |
| <b>4</b> | <b>Distance Metric Learning for Object Segmentation</b>   | <b>117</b> |
| 4.1      | Overview of the Improved Segmentation Algorithm . . . . . | 118        |
| 4.2      | Segmentation Energy Function . . . . .                    | 120        |
| 4.3      | Ellipsoidal Object Model . . . . .                        | 124        |
| 4.4      | Learning the <i>Pairwise</i> Potential . . . . .          | 126        |
| 4.4.1    | Distance Metric Learning . . . . .                        | 126        |
| 4.5      | Experimental Results . . . . .                            | 131        |
| 4.6      | Conclusions and Discussion . . . . .                      | 144        |
| <b>5</b> | <b>Conclusions</b>  | <b>153</b> |
| 5.1      | Summary of Contributions . . . . .                        | 153        |
| 5.2      | Future Work . . . . .                                     | 155        |
|          | <b>Appendices</b>   | <b>157</b> |
| <b>A</b> | <b>Random Sample Consensus (RANSAC)</b>                   | <b>159</b> |
| <b>B</b> | <b>Kernel Density Estimation and Mean Shift</b>           | <b>163</b> |
| <b>C</b> | <b>Least Squares Ellipsoid Fitting</b>                    | <b>167</b> |

|                       |            |
|-----------------------|------------|
| <b>D Optical Flow</b> | <b>171</b> |
| <b>References</b>     | <b>191</b> |

# List of Figures

|      |   |    |
|------|---|----|
| 1.1  | Examples of cluttered or visually complex environments. . . . .   | 4  |
| 1.2  | Door handle localization. . . . .   | 26 |
| 1.3  | Object segmentation. . . . .  | 27 |
| 2.1  | Sequence of processing steps in the proposed algorithm. . . . .   | 35 |
| 2.2  | Coordinate frames associated with camera and robot body. . . . .  | 37 |
| 2.3  | The central frame of the image sequence, with the corresponding filtered depth map. . . . .   | 39 |
| 2.4  | Reconstructed point cloud, with the dominant plane, and the door handle bounding box shown. . . . .   | 40 |
| 2.5  | Geometry of the door in 3D space, and in the horizontal plane of the robot body reference frame. . . . .  | 41 |
| 2.6  | Comparison of best fit lines generated by the least squares method, and the RANSAC algorithm followed by least squares on the set of inlier points. . . . . | 43 |
| 2.7  | Histogram of points projected onto the normal of the door. . . . .  | 44 |
| 2.8  | Experimental set up. . . . .  | 48 |
| 2.9  | Examples of knob, and lever type door handles. . . . .  | 49 |
| 2.10 | Visualization of the bounding box for different types of handles. . . . .   | 56 |
| 2.11 | Handle localization with part of the handle outside of the camera's FOV . . . . .   | 61 |
| 3.1  | Major processing steps of the proposed object segmentation algorithm. . . . .   | 65 |

|      |   |     |
|------|---|-----|
| 3.2  | Flow of information in the proposed segmentation algorithm, from input to output. . . . .   | 66  |
| 3.3  | Examples of 4-connected and 8-connected neighbourhood systems. . . .  | 69  |
| 3.4  | Examples of regular and irregular neighbourhood systems. . . . .  | 70  |
| 3.5  | Comparison of the Ising and the contrast sensitive prior terms. . . . .   | 74  |
| 3.6  | An example graph with four sites. . . . .   | 78  |
| 3.7  | Steps of the object detection process. . . . .  | 81  |
| 3.8  | Examples of log likelihood ratios for three different objects. . . . .  | 83  |
| 3.9  | Comparison between the log likelihood ratios of depth and curvature models for a cylindrical object. . . . .  | 86  |
| 3.10 | Segmentation results using the initial and the final values for the energy function parameters. . . . .   | 92  |
| 3.11 | Example depth map reconstructed with the aid of a projected light pattern.  | 94  |
| 3.12 | Precision and recall scores for the object detection step. . . . .  | 97  |
| 3.13 | Sample segmentation results for the implemented segmentation methods.   | 102 |
| 3.14 | Comparison of the $F_1$ , precision, and recall scores for the proposed segmentation algorithm and the method of Bjorkman and Kragic [2010a,b].                                     | 104 |
| 3.15 | Comparison of the mean and maximum distance to the true object boundary for the proposed segmentation algorithm and the method of Bjorkman and Kragic [2010a,b]. . . . .            | 105 |
| 3.16 | Comparison of example segmentation results for the proposed method, and the algorithm presented by Bjorkman and Kragic [2010a,b]. . . . .   | 106 |
| 3.17 | Example segmentation results for scenes where the proposed segmentation algorithm exhibits better performance when compared to the method of Bjorkman and Kragic [2010a,b]. . . . . | 107 |
| 3.18 | Scenes used to evaluate the performance of the proposed method. . . . .   | 110 |

|      |   |     |
|------|---|-----|
| 4.1  | Initialized object and background regions, and segmentation results after several iterations of the algorithm. . . . .  | 119 |
| 4.2  | Log likelihood ratios of the feature models for the first iteration of the segmentation algorithm. . . . .  | 121 |
| 4.3  | Log likelihood ratios of the feature models for the last iteration of the segmentation algorithm. . . . .   | 122 |
| 4.4  | Determination of the type of junction between two surfaces. . . . .   | 128 |
| 4.5  | Normal vector differences calculated for vectors estimated using a voting algorithm, and a least squares plane fitting method. . . . .                                  | 133 |
| 4.6  | Selected pairs of similar and dissimilar points. . . . .  | 133 |
| 4.7  | Segmentation results for objects which could not be detected using the geometric object detection method. . . . .   | 134 |
| 4.8  | Results for the improved segmentation algorithm, showing the $F_1$ , precision, and recall scores. . . . .  | 135 |
| 4.9  | Results for the improved segmentation algorithm, showing the mean and maximum distance from the detected boundary to the true object boundary. . . . .                  | 136 |
| 4.10 | Comparison of segmentation results on objects with thin structure between the improved segmentation algorithm, and the method proposed in the previous chapter. . . . . | 139 |
| 4.11 | Examples of segmentation error due to shrinking of the object region. . . . .   | 140 |
| 4.12 | Examples of segmentation error due to the object region expanding onto the background. . . . .  | 141 |
| 4.13 | Learned feature weights from the first and the last iteration of the segmentation algorithm for object 16 in Figure 4.15, scene 5. . . . .                              | 142 |
| 4.14 | Learned feature weights from the first and the last iteration of the segmentation algorithm for object 22 in Figure 4.15, scene 7. . . . .                              | 143 |

|      |  |     |
|------|--|-----|
| 4.15 | Scenes used to evaluate the performance of the improved segmentation method. . . . .           | 146 |
| A.1  | Demonstration of the RANSAC algorithm . . . . .  | 160 |
| B.1  | Demonstration of the mean shift algorithm. . . . .   | 165 |
| D.1  | Results of 3D reconstruction for three types of surfaces, without a projected pattern. . . . . | 174 |
| D.2  | Results of 3D reconstruction for three types of surfaces, using a projected pattern. . . . .   | 175 |



# List of Tables

|     |  |     |
|-----|--|-----|
| 2.1 | Computational performance of the door handle segmentation and localization algorithm. . . . .  | 50  |
| 2.2 | Common error types, and their abbreviations. . . . .   | 51  |
| 2.3 | Summary of the performance of the door handle localization algorithm. .  | 52  |
| 3.1 | Example unary potentials. . . . .  | 78  |
| 3.2 | Example pairwise potentials. . . . .   | 78  |
| 3.3 | Costs of cuts and probabilities associated with all possible labellings of the example graph shown in Figure 3.6. . . . .  | 79  |
| 3.4 | Segmentation energy function parameters before and after optimization. .   | 91  |
| 3.5 | Computational performance of the proposed segmentation algorithm. . .  | 96  |
| 3.6 | Mean and standard deviation ( $\sigma$ ) of the $F_1$ , precision, and recall scores for the proposed method, and the method of Bjorkman and Kragic [2010a,b].   | 103 |
| 3.7 | Mean and standard deviation ( $\sigma$ ) of the average and maximum distances to the true object boundary for the proposed method, and the method of Bjorkman and Kragic [2010a,b]. . . . .  | 103 |
| 4.1 | Computational performance of the improved segmentation algorithm. . .  | 132 |
| 4.2 | Mean and standard deviation ( $\sigma$ ) of the $F_1$ , precision, and recall scores for the improved segmentation algorithm, as well as the method proposed in Chapter 3 and the method of Bjorkman and Kragic [2010a,b]. . . . . | 137 |

|     |   |     |
|-----|---|-----|
| 4.3 | Mean and standard deviation ( $\sigma$ ) of the average and maximum distances to the true object boundary for the improved segmentation algorithm, as well as the method proposed in Chapter 3 and the method of Bjorkman and Kragic [2010a,b]. . . . . | 138 |
| D.1 | Standard deviation of reconstructed points from a fitted planar model for different surface types. . . . .  | 176 |

# List of Abbreviations

---

|        |                              |
|--------|------------------------------|
| CRF    | Conditional Random Field     |
| DCM    | Deformable Contour Model     |
| DML    | Distance Metric Learning     |
| FN     | False Negative               |
| FOV    | Field of View                |
| FP     | False Positive               |
| GMM    | Gaussian Mixture Model       |
| HSV    | Hue-Saturation-Value         |
| LLR    | Log Likelihood Ratio         |
| MAP    | Maximum a Posteriori         |
| MRF    | Markov Random Field          |
| PCA    | Principal Component Analysis |
| RANSAC | Random Sample Consensus      |
| TP     | True Positive                |



# Nomenclature

---

|  |   |
|--|---|
| $a, b$   | Lower case letters denote scalar quantities.  |
| $\mathbf{a}, \mathbf{b}$   | Bold lower case letters denote vector quantities.   |
| $\mathbf{A}, \mathbf{B}$   | Bold capital letters denote matrix quantities.  |
| $\mathbf{A} \succeq 0$   | Indicates that a matrix $\mathbf{A}$ is positive semi-definite.   |
| $tr(\cdot)$  | Trace of a matrix.  |
| $\delta(\cdot)$  | Indicator function.   |
| $\lambda_{min}, \dots, \lambda_i, \dots, \lambda_{max}$          | Eigenvalues of a matrix.  |
| $\mathbf{e}_{min}, \dots, \mathbf{e}_i, \dots, \mathbf{e}_{max}$ | Eigenvectors corresponding to the eigenvalues $\lambda_{min}, \dots, \lambda_i, \dots, \lambda_{max}$ . |
| $\mathbf{a}$   | An appearance feature (intensity or colour).  |
| $B$  | The robot body reference frame.   |
| $\mathcal{C}$  | the set of all cliques in the graph $\mathcal{G}$ .   |
| $C$  | A cut on the a graph.   |
| $\mathbf{c}_B, \mathbf{c}'_B$                                    | Positions of the camera frames $C$ and $C'$ in the robot body reference frame.                          |

|                             |   |
|-----------------------------|---|
| $\mathbf{C}_{ij}$           | The outer product of a feature difference vector $\boldsymbol{\rho}_{ij}$ .           |
| $\mathbf{C}_{ij}^\alpha$    | The outer product of a feature difference vector $\boldsymbol{\rho}_{ij}^\alpha$ .    |
| $c$                         | A clique in $\mathcal{G}$ .   |
| $C, C'$                     | The camera reference frames.  |
| $\mathbf{d}_i$              | A vector of features at site $i$ in the image.  |
| $\mathcal{E}$               | The set of edges in the augmented graph.  |
| $E$                         | The energy function.  |
| $\mathbf{F}$                | The set of all random variables $F_i$ associated with all sites $i \in \mathcal{S}$ . |
| $\mathbf{f}$                | The set of all labels $f_i$ .   |
| $\mathbf{f}^*$              | A minimum energy labelling of an image.   |
| $\hat{\mathbf{f}}$          | True labelling of an image.   |
| $\mathbb{F}$                | The set of all possible realizations of $\mathbf{F}$ .                                |
| $f$                         | Focal length of the camera.   |
| $F(\mathbf{v}, \mathbf{p})$ | Algebraic distance.   |
| $F_1$                       | Harmonic mean of precision and recall.  |
| $F_i$                       | A random variable associated with a site $i$ .  |

|   |  |
|---|--|
| $f_i$                                   | A specific value of a random variable $F_i$ .  |
| $\mathbf{G}_t$                          | The gradient of the matrix $\mathbf{\Lambda}$ at iteration $t$ .                                   |
| $\mathcal{G}(\mathcal{S}, \mathcal{N})$ | A graph with sites defined by $\mathcal{S}$ and edges defined by $\mathcal{N}$ .                   |
| $\tilde{\mathcal{G}}$                   | Augmented graph.   |
| $h^a$                                   | The histogram of colour or intensity values.   |
| $i, j, l$                               | Indices of sites or pixels in the image.   |
| $\mathcal{L}$                           | The set of labels that each site may take.   |
| $\mathbf{M}$                            | A covariance matrix.   |
| $\mathcal{N}$                           | A neighbourhood system on the sites in $\mathcal{S}$ .   |
| $\hat{\mathbf{n}}$                      | Normal vector.   |
| $\mathbf{p}_B$                          | A 3D point in the robot body reference frame.  |
| $\mathbf{p}_C$                          | A 3D point in the camera reference frame.  |
| $\mathbf{q}$                            | A point in the image plane.  |
| $\dot{\mathbf{q}}$                      | The velocity of a point $\mathbf{q}$ in the image plane.   |
| $\mathcal{R}$                           | A set containing pairs of sites with the same label used in the calculation of $\mathbf{S}_\rho$ . |

|                    |   |
|--------------------|---|
| $\mathbf{R}_{BC}$  | The rotation matrix from the camera frame $C$ to robot body frame $B$ .   |
| $\mathbf{R}_{CC'}$ | The rotation matrix from the camera reference frames $C'$ to $C$ .  |
| $\mathcal{S}$      | A set of all indices of sites in the image.   |
| $\mathcal{S}$      | A set of edges connecting all sites in $\mathcal{S}$ to the source node $s$ .                                   |
| $\mathbf{S}_\rho$  | A covariance of feature difference vectors $\rho$ .   |
| $s$                | The source node in an augmented graph.  |
| $\mathcal{K}$      | A subset of $\mathcal{T}$ containing the set of triplets that violate the margin constraint.                    |
| $\mathcal{T}$      | A training set containing triplets of similar and dissimilar sites.   |
| $\mathcal{T}$      | A set of edges connecting all sites in $\mathcal{S}$ to the sink node $t$ .                                     |
| $t_C$              | The relative position of the camera frame $C'$ with respect to the camera $C$ , in the reference frame $C$ .    |
| $T$                | A parameter controlling the relative significance of the data and smoothness terms in the energy function $E$ . |
| $t$                | The sink node in an augmented graph.  |
| $\mathbf{v}$       | A vector parameterizing an ellipsoid.   |
| $V_1$              | Unary potential function.   |



|                  |   |
|------------------|---|
| $V_2$            | Pairwise potential function.  |
| $V_c$            | Clique potential function.  |
| $Z$              | The partition function.   |
| $\Lambda$        | A positive semidefinite matrix parameterizing a pseudo-distance function used to evaluate feature differences.    |
| $\Lambda^\alpha$ | A set of three positive semidefinite matrices parameterizing three pseudo distances.                              |
| $\Lambda^+$      | A positive semidefinite matrix used to evaluate feature differences across a concave surface junction.            |
| $\Lambda^-$      | A positive semidefinite matrix used to evaluate feature differences across a convex surface junction.             |
| $\Lambda^\times$ | A positive semidefinite matrix used to evaluate feature differences across an undetermined surface junction type. |
| $\Theta$         | A set of minimum energy labellings  |
| $\theta$         | A measure of difference between two normal vectors.   |
| $\kappa$         | Curvature measure of a 3D point sampled surface.  |
| $\kappa^+$       | Positive curvature.   |
| $\kappa^-$       | Negative curvature.   |

|                          |   |
|--------------------------|---|
| $\mu$                    | A parameter used to weigh the penalty associated with margin violations.              |
| $\nu$                    | A parameter controlling how quickly a feature difference reduces the cost of an edge. |
| $\xi_{ijl}$              | Slack parameter.  |
| $\boldsymbol{\rho}_{ij}$ | A vector of feature differences between sites $i$ and $j$ .                           |
| $\sigma$                 | Standard deviation.   |
| $\sigma_{al}$            | Standard deviation of algebraic distances between 3D points and an ellipsoid.         |
| $\boldsymbol{\tau}$      | Texture feature vector.   |
| $\boldsymbol{\omega}$    | A set of parameters of the energy function $E$ .                                      |
| $w(i, j)$                | The weight associated with an edge between sites $i$ and $j$ .                        |

# Introduction

---

## 1.1 Motivation

Service robots have already entered our lives. A number of commercial service robots are available from companies such as iRobot to assist in household tasks including vacuuming, cleaning the pool, or cleaning eavestroughs. These robots are simple, perform a single function, do not possess complex sensing capabilities, and rely entirely on the user to move from one workspace to the next.

It has long been desired to develop autonomous robots that are capable of performing more complex tasks. Robots that could be considered as “*assistants*” and not “*appliances*”, with the ability to understand commands, autonomously navigate and traverse the environment and interact with objects. Robots with the capability to perform such tasks in an uncontrolled and dynamic environment could have a vast number of applications in homes and offices, hospitals, and hazardous or dangerous environments [Reiser et al., 2009; Jain and Kemp, 2010; Tsotsos et al., 1998; Rotenstein et al., 2007; Srinivasa et al., 2010, 2012; Chung et al., 2007].

Performing complex tasks such as grasping objects requires sophisticated vision systems to guide the robot’s actions. This is especially true in uncontrolled settings where the appearance and geometry of the environment is unknown. The robot’s vision system consists of hardware and software components. The hardware components can include one or more 2D cameras as well as 3D sensors. The vision hardware acquires the raw data that represents what a robot “sees”. The software components are what allows the robot to perceive and analyze the world. The algorithms are what converts the

raw pixel intensities, colour components, or coordinates of points in a point cloud to a more meaningful and useful representation; enabling the robot to map the environment, estimate its own motion and localize itself within the environment, avoid obstacles, detect objects, and guide the grasping of known objects [Filliat and Meyer, 2003; Meyer and Filliat, 2003; Kunchev et al., 2006; Bjorkman and Eklundh, 2006].

Despite a significant body of work, a number of key problems remain unaddressed, specifically, the problem of visually localizing a door handle of unknown appearance and geometry such that it can be opened with a mobile manipulator, and the problem of guiding the grasping of previously unseen objects in a realistic setting.

### 1.1.1 Motivation for Door Handle Localization

Without the ability to open doors, the robot’s operating space and capabilities are limited. The robot is effectively trapped in its initial operating environment.

A service robot needs to be able to gain access to new workspaces. Assistant robots for homes and offices are intended to alleviate some of the workload traditionally performed by people. This implies that a user should not be required to assist the robot in moving from one room to another. Even in situations where the robot is manually operated, it may be undesirable to require the operator to manually control the robot to grasp and manipulate the handle. For example, the Playbot robotic wheelchair [Tsotsos et al., 1998; Rotenstein et al., 2007] is intended to provide assistance to disabled persons, who may be unable to produce the fine motions required to manually guide the grasping or manipulation of a door handle. For mobile robots to navigate seamlessly through human environments, a method for localizing a door handle so that it can be manipulated by the robot is required.

### 1.1.2 Motivation for Online Model Acquisition

To perform a basic task, the robot must be able to grasp and manipulate objects. Even if the robot has the ability to navigate and traverse its environment, without manipulation capabilities the robot is limited to observational roles such as patrolling or mapping.

The literature on object grasping is vast and no attempt is made to review it here. It is sufficient to say that when the pose (location and orientation) and at least a partial model of the object are available, a robotic manipulator can be controlled to execute a motion that will grasp the object (for example, Miller and Allen, 2004; Huebner et al., 2009).

In parallel, if a model of the object is available, a number of methods exist for estimating its pose [Lowe, 1987, 1991; Goddard, 1997; Rosenhahn, 2003; Srinivasa et al., 2010; Asfour et al., 2008; Prats et al., 2010]. However, because service robots are meant to be deployed in uncontrolled or unknown environments containing a large number of items, it is unlikely that a comprehensive database of object models would be available a priori. At the same time, it is impractical to expect the user to scan all of the items they wish the robot to interact with prior to its deployment.

An alternative approach is to construct a model of the object online [Hirano et al., 2005; Wang et al., 2005; Yamazaki et al., 2006, 2008; Bone et al., 2008; Kuehnle et al., 2008; Huebner and Kragic, 2008; Rusu et al., 2009a; Marton et al., 2009; Huebner et al., 2009; Rusu et al., 2010]. In this case, the process of acquiring a full or partial model requires the object to be segmented from the rest of the environment.

### 1.1.3 Segmentation as a Tool for Online Model Acquisition

Object segmentation can be used to separate the geometric structure of a target object from the rest of the environment. It is then possible to construct a full or partial model of the object, which can be used to guide grasping or manipulation tasks.

Recent advances allow for segmentation of objects using a combination of features such as intensity, colour, and texture [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006; Ilea and Whelan, 2011; Rother et al., 2004; Kim and Hong, 2009]. Notwithstanding these efforts, image segmentation in uncontrolled or visually complex environments is still a challenging task.



Figure 1.1: Examples of cluttered or visually complex environments.

Human environments are visually complex. Human environments are also cluttered (Figure 1.1). Objects are placed next to each other and on top of other objects. Many of them share multiple appearance cues with other objects, or with the background. The appearance and geometry of many common objects are chosen as much for aesthetic purposes as they are for practical ones. The result is that objects are often bright and multicoloured with many false internal edges and irregular geometry. Uncontrolled lighting conditions create artificial edges and alter region intensities due to shadows and highlights. Light sources change position and intensity based on the time of day, varying the appearance of the environment further.

In certain cases, the environment can possess a consistent geometric structure. Knowledge about the structure of the environment can significantly simplify the segmentation

problem, allowing for a simple and effective method to be developed. However, the use of such assumptions in the segmentation algorithm needs to be considered carefully; incorporating invalid assumption can have a detrimental effect on the performance of the method.

The issues discussed above make development of efficient and autonomous object segmentation algorithms very challenging. Nevertheless, segmentation methods that are able to function effectively under such conditions are required to enable online acquisition of object models.

## 1.2 Literature Review

Segmentation is the process of separating an image into two or more disjoint parts. The goal is to separate the image into components that are more meaningful or useful than the original image. This could mean simply reducing the image to a set of homogeneous regions, or separating areas that would be considered as objects by a human observer. In this work the focus is placed on the second definition, where a region in the image is thought that would be considered as an object by a person.

This section begins with a review of literature on image segmentation. The discussion is then focused on the specific problems that are examined in this thesis: door handle localization, and segmentation for online object model acquisition. Due to the use of multiple appearance and geometric features in the segmentation process, a review of distance metric learning literature is also provided as distance metric learning allows for an estimation of the relative significance of available segmentation features and their combinations.

### 1.2.1 Review of Object Segmentation Methods

Segmentation is commonly formulated as a thresholding, clustering, or optimization problem [Forsyth and Ponce, 2002; Szeliski, 2010]. These methods are used as a foundation for many segmentation techniques developed for 3D model acquisition. However, specific discussion regarding segmentation methods that incorporate 3D information is reserved for later sections.

#### Thresholding

Thresholding is one of the earliest segmentation techniques Szeliski [2010]. It is simple, generally fast to compute, and performs well in simple or controlled environments. Thresholding is commonly applied when the mean intensity of the object or objects of interest is different from the mean intensity of the background.

Thresholding involves separating the image into two or more regions based on a set of threshold values, and a set of image features, most commonly the intensity values of the pixels. More formally, a pixel  $(u, v)$  with some feature  $f(u, v)$  is assigned to a region  $k$  based on a threshold range  $[t_{k-1}, t_k)$ , resulting in the segmented image  $S(u, v)$ :

$$S(u, v) = k \quad \text{if} \quad t_{k-1} \leq f(u, v) < t_k \quad (1.1)$$

Threshold values can be selected manually but this requires strong knowledge of the appearance of the object(s) and the background. Alternatively, threshold values can be calculated from image features. When a single set of thresholds is calculated for the entire image, the techniques are referred to as *global thresholding*. When thresholds are calculated locally at different image locations, the methods are referred to as *local* or *adaptive thresholding*.



A number of global thresholding schemes are presented in the literature, including methods based on the known area of the object in the image, based on peaks or valleys in the feature histogram [Weszka, 1978; Rosenfeld and Torre, 1983], based on the curvature of the smoothed histogram [Tsai, 1995], on inter- and intra-class variance [Otsu, 1975], and methods based on histogram entropy [Kapur et al., 1985]. Image thresholding can also be considered as a classification or model fitting problem [Kittler and Illingworth, 1986a; Cho et al., 1989]. These methods are used when no clear valleys exist in the image feature histogram.

When the mean intensity changes across the image, it may not be possible to find a single set of global thresholds that can perform effectively at all locations in the image. In such situations it is more effective to calculate a set of local or adaptive thresholds whose values depend on the position in the image [Nakagawa and Rosenfeld, 1979; Sauvola and Pietikainen, 2000]. Adaptive methods generally perform better than global methods when the mean of the feature being thresholded changes based on the location in the image, for example when the mean intensity appears to change due to uneven illumination.

A drawback of thresholding methods is that spatial relationships between pixels are ignored. The resulting regions are not guaranteed to be contiguous, or represent meaningful objects. Thresholding methods are also sensitive to illumination effects such as shadows and highlights. Smooth changes in shading from illumination can be addressed with adaptive methods, however, sharp illumination changes can cause parts of the scene to be incorrectly segmented.

It is interesting to note, that more sophisticated thresholding methods utilize model fitting, distribution fitting, and mode seeking methods. Thresholding can be considered a simple form of feature space clustering, where the feature space is constrained to be one dimensional, and cluster membership is assigned based on the threshold values.

## Feature Space Clustering

Unsupervised segmentation can be formulated as a clustering problem. Each point in the image is associated with a feature vector. The feature vector encodes properties that are used for segmentation. These can include the intensity value at a pixel, colour, texture features, and other local image features that can describe the pixel itself, or the local image region surrounding it. The space used to represent these feature vectors is called the *feature space*. Assuming that regions that correspond to meaningful parts of the image exhibit similarity in their feature vectors, segmentation can be accomplished by assigning clusters to dense regions of the feature space, and labelling each pixel according to its cluster membership.

A number of clustering methods are used in image processing to perform segmentation.  $K$ -means [Szeliski, 2010] is used when the number of clusters is known, or can be determined. This method attempts to find  $K$  clusters such that the within-cluster sum of square distances of each point to the cluster mean is minimized. This can also be considered as model fitting, since it is equivalent to fitting a spherical symmetric distribution to the data. The  $K$ -medians algorithm is a variation of the  $K$ -means, where the method attempts to minimize the within cluster distance between the points and the cluster's median. The  $K$ -means method is sensitive to initialization. In certain cases, even with well separated clusters, the method can converge on an incorrect solution.

If the data is assumed to be normally distributed about the cluster means, a Gaussian mixture model (GMM) can be fitted. Each data point is assigned to a cluster implicitly as part of the fitting process [McKenna et al., 1999; Permuter et al., 2006]. The number of mixture components can be set manually, or determined during the model fitting process.

If a parametric model for the distribution of the data cannot be assumed, non-parametric methods can be used to directly search for dense regions of the feature space. The mean-shift algorithm [Fukunaga and Hostetler, 1975; Cheng, 1995; Comaniciu and Meer, 2002] is used for locating the maxima of a density function given discretely sampled data.

Given an initial guess, the method uses a predefined kernel to calculate a weighted mean of nearby points in feature space. The estimate of the mode is moved to the calculated mean, and the process is repeated, resulting in the algorithm iteratively moving to the nearest local mode of the distribution (for details see Appendix B). The kernel acts to smooth the discretely sampled data. Spherical and Gaussian kernels are typically used. The bandwidth of the kernel needs to be carefully chosen such that the desired peaks can be detected, while the noise is removed. The algorithm can be repeated for each point in the data, thus finding all the relevant modes and assigning each point to a cluster. However, restarting the procedure for every data point can be computationally slow. To increase computational performance, the algorithm is commonly initialized at a number of randomly selected seed points. Once the modes of the distribution are found, a separate method is used to determine cluster membership for all points [Comaniciu and Meer, 2002].

Clustering methods operate directly in the feature space and share a lot of the drawbacks of other global feature based methods such as histogram thresholding. The most significant drawback is that these methods do not take into consideration the spatial configuration of image pixels. This can result in clusters that are not connected in the image. To achieve better results pixel locations have to either be explicitly included in the feature vector, or a connected component search has to be performed following the clustering operation to further split the clusters into contiguous regions.

### **Region-Based Methods**

Region-based methods group pixels with similar characteristics by either growing initial seed regions outwards, splitting regions, or merging existing regions based on a homogeneity constraint. A combination of the above operations can also be used in a split-and-merge approach [Haralick and Shapiro, 1985].

Growing methods start at one or more seed pixels or regions. Initial seeds can be determined manually, or automatically. Initial seeds are expanded by including neighbouring

pixels if they meet a homogeneity condition. The candidate pixel can be compared to either its nearest neighbour in the region, to the original seed pixel, or to some region property such as mean intensity.

The watershed segmentation method [Vincent and Soille, 1991; Beucher, 1992] can be considered a form of region growing. The algorithm is an edge-based (local) technique that seeks to label regions separated by high image gradients. The technique draws its name from the process of flooding of catchment basins. The segmentation is performed by first calculating the gradient magnitude of the original grayscale image. The gradient image can be considered as a height map with regions of high gradients corresponding to ridges, and regions of low gradient corresponding to valleys. A flooding operation is then performed starting at low gradient magnitude regions. When two regions meet, the ridge is labelled, and the flooding process continues until only a single basin remains.

When regions are considered, the growing method is referred to as a “*merging*” method, but the principle is the same. Regions are merged based on a homogeneity condition, such as mean intensity, maximum intensity difference, or a statistical comparison of the distributions in the two regions [Haralick and Shapiro, 1985].

In contrast to growing methods, splitting methods are initialized to contain the whole image. The image is then split into smaller predetermined regions if a homogeneity condition across the splitting boundary is not met. Similarly to merging methods these conditions can be simple, such as the maximum intensity difference of the two regions to be separated, or can involve more complex statistics. Splitting methods can be combined with merging methods. This is done to merge adjacent regions that originated from two different parent regions. Because these regions originate from different parents, the regions will not be compared against each other if a merging step is not performed.

The advantage of this class of methods in comparison to feature space clustering techniques is that they incorporate spatial relationships between pixels by grouping and splitting directly in the image, which implicitly takes into consideration spatial connectivity of segments. The segmentation automatically results in connected components in

the image. Region-based methods can often produce better or more meaningful results than feature spaced clustering methods [Shapiro and Stockman, 2001]. Similarly to feature space clustering methods, any image feature can be used, assuming a similarity or homogeneity constraints can be defined.

Region-based methods also exhibit a number of drawbacks. The output of region growing methods heavily depend on the initial seed points used. If a bad initialization is provided, the algorithm may not converge to any meaningful solution. The method also does not possess a way of detecting or correcting cases where seed regions span across object boundaries. Finally, if the region homogeneity constraints are set too strictly the results will appear over segmented. If they are set too loosely, the resulting image will be under segmented.

In particular, the watershed algorithm is prone to over-segmentation, since it generates a unique region for every local minimum in the gradient magnitude image. For these reasons, the algorithm is often used as a preprocessing step to over segment an image into small clusters of similar pixels, referred to as *super pixels*, which are used instead of individual image pixels in more advanced methods [Szeliski, 2010].

### Deformable Contour Models: Snakes and Level Sets

Deformable contour models (DCMs) represent the segmentation either explicitly using a parametric curve, or implicitly as a level set of an embedding function. The segmentation is obtained by minimizing an energy function whose minimum corresponds to an optimal segmentation with respect to the constraints encoded in the energy. The constraints encoded in the energy function are designed to attract the contour to image features such as high intensity gradients, to enforce region homogeneity constraints, to control the shape of the contour enforcing smoothness, or to encourage the shape to match a known prior.

The original DCM introduced by Kass et al. [1988] minimized an energy function consisting of two terms: internal and external. The internal energy of the curve consisted of

an elasticity term and a curvature term. The elasticity term penalized the total length of the boundary, while the curvature term penalized sharp edges acting as a smoothness constraint. This also caused the contour to shrink into a circle, and then to disappear entirely if no image features were used. The external energy term was set to the negative inverse of the intensity gradient and was used to attract the contour to image edges. The initial snake models suffered from sensitivity to initialization, and an inability to detect distant gradients. It also relied exclusively on local edge information, and was not able to enforce any global homogeneity in the regions.

The contour's inability to detect distant image edges was addressed by modifying the algorithm to utilize edge distances [Cohen and Cohen, 1993] or gradient vector flows [Xu and Prince, 1998] in the external energy term. Statistical priors on the shape of the snake were used by Cootes et al. [1995] to control the final shape of the contour when the rough shape of the object was known.

An inherent limitation of the deformable contour method is caused by the explicit representation of the contour. The snake is not able to cross itself, or change topology. This was addressed by the introduction of the level set method as a means to implicitly represent the evolving boundary. Its first application to image segmentation was presented by Malladi et al. [1995]. The level set segmentation approach represents the contour as a zero level set of a higher dimensional embedding (or level set) function or surface. The contour evolves with a speed which depends both on the shape of the contour, and on the image features. Because the representation is implicit, the model is able to adapt its topology to represent more complex contours.

The initial level set methods performed segmentation utilizing only intensity gradients [Caselles et al., 1997]. The method has since been expanded to include both region and boundary constraints, as well as other features such as texture, colour, and motion [Cremers et al., 2007].

### Graph-Based Clustering

Graph-based clustering segmentation methods model the image as a graph. Pixels or super pixels are considered as the nodes linked to their adjacent nodes through a neighbourhood system. It is common to define the neighbourhood system to only connect immediately adjacent nodes, for example the adjacent 4 or 8 pixels in an image, or super pixels that share a common boundary. The weight of the edges connecting neighbouring nodes are set based on a similarity (sometimes referred to as affinity) measure. The similarity measure can be defined for any set of feature that can be locally assigned to describe the nodes. These features can include intensity, colour, or texture. An additional distance term can be added to weaken the affinity between nodes that are farther apart in the image.

Groups of nodes that should form a cluster are expected to be linked together by edges with high weights. Different regions are expected to be linked either by very few edges, or by edges with low weights.

Wu and Leahy [1993] performed segmentation by finding a minimum cost cut on such a graph. A cut is defined as a subset of edges that, when removed, completely partition the nodes of the induced graph into two disjoint subsets corresponding to image segments. The cost of the cut is the sum of the weights of all removed edges. The minimum cost cut was based on a formulation of a maximum flow problem. To find a minimum cut, the graph was augmented with a source and sink nodes that were iteratively attached to every pair of nodes in the image graph. The max flow problem was solved for each pair, and the maximum flow solution was chosen.

The min-cut algorithm of Wu and Leahy [1993] favoured small regions. This is due to the fact that the cost of separating a large number of weakly connected nodes is often higher than the cost of separating just a few nodes with higher weights. In an effort to address this issue, Wang and Siskind [2001], and Shi and Malik [2000] proposed to normalize the cost of the cut. Wang and Siskind [2001] normalized the cut by the total

length of the boundary, while Shi and Malik [2000] normalized the cut by the total edge weight connections to all of the nodes in the graph.

A different graph-based clustering approach was presented by Felzenszwalb and Huttenlocher [2004]. The segmentation was based on an assumption that for any two regions, the variation across the region should be larger than the variation within the region. This results in a segmentation that is defined by the authors as neither too fine nor too coarse.

Graph-based clustering has an advantage over feature space clustering methods because the spatial relationship between pixels are implicitly encoded through the neighbourhood system. This results in segments that are always contiguous.

## Probabilistic Graphical Models

Similarly to the graph based clustering methods, the image is modelled as a graph where the nodes correspond to pixels or super pixels connected by a neighbourhood system. The super pixels are small collections of pixels commonly resulting from a previous segmentation procedure, such as watershed. A set of hidden random variables is introduced. In the context of image segmentation, the state of the random variables is used to assign region labels to the nodes. The segmentation problem can then be represented as the estimation of the maximum a posteriori (MAP) probability  $P(\mathbf{f}|\mathbf{d})$  of the region labels  $\mathbf{f}$  conditioned on the observed data  $\mathbf{d}$ . Using Bayse's rule, it can be written as:

$$\mathbf{f}^* = \arg \max_{\mathbf{f}} P(\mathbf{f}|\mathbf{d}) = \frac{P(\mathbf{d}|\mathbf{f})P(\mathbf{f})}{P(\mathbf{d})} \quad (1.2)$$

where  $\mathbf{f}^*$  is the most likely set of labels. It is important to note that when the problem involves MAP estimation, the denominator does not depend on the labels and is constant. By disregarding the data probability term  $P(\mathbf{d})$  in the denominator, the problem can be re-expressed as the estimation of the mode of the joint distribution on the set of data and labels:



$$\mathbf{f}^* = \arg \max_{\mathbf{f}} P(\mathbf{f}, \mathbf{d}) = P(\mathbf{d}|\mathbf{f})P(\mathbf{f}) \quad (1.3)$$

$P(\mathbf{d}|\mathbf{f})$  is the likelihood function representing how well the observed data corresponds with the assigned labels. It can depend on the intensity of the nodes, colour, texture, or any local set of features. A Markov random field (MRF) is used as a model for the state prior  $P(\mathbf{f})$  to encode a smoothness constraint.

An issue with this formulation is that a smoothness prior on labels results in penalizing some function of the total length of the boundary. It is often more useful to encode a smoothness constraint that depends not just on the labels, but also on the image features. If the features at two nodes are very different, assigning different labels to the nodes should be penalized less. Such a constraint can be difficult to factorize into likelihood and prior terms.

An alternative formulation called the conditional random field (CRF) [Lafferty et al., 2001] does not require the factorization to be made explicit. It models the posterior distribution directly as an MRF, and allows for complex dependencies of the labels on the data to be written directly in the posterior distribution.

Initial applications of probabilistic graphical models in image processing involved binary [Greig et al., 1989], and grayscale [Geman and Geman, 1984] image restoration. Later works by Boykov et al. [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006] demonstrated how a minimum cost cut on an appropriately constructed image graph can be used to exactly solve a MAP MRF problem for certain classes of MRFs and CRFs. This method, called *graph cuts*, was initially demonstrated on grayscale images, but has since been expanded to perform segmentation using other features including colour [Rother et al., 2004], and colour and texture [Kim and Hong, 2009].

The main limitation of these methods is that they require initialization, which is commonly provided by a user, as marks on the image indicating regions which should belong to the object or the background.

### 1.2.2 Review of Door Handle Localization Methods

Without the ability to open doors, a robot’s operating environment is limited. To address this issue, a number of researchers have focused on developing methods and algorithms for robotic door opening, including both control strategies [Liu et al., 2008, 2009], as well as vision-based approaches to guide the physical actions of the robot [Andreopoulos and Tsotsos, 2008; Kragic et al., 2002; Chung et al., 2007, 2009; Rusu et al., 2009b; Klingbeil et al., 2008, 2010]. To allow for effective door handle localization the method must be able to estimate the pose (position and orientation) of a handle. Performing door handle localization using visual or 3D geometric information is a challenging task as the appearance of both the door and that handle, as well as the geometry of the handle, can change significantly from one scene to another, even within the same home or building.

Andreopoulos and Tsotsos [2008] present a computer controlled wheelchair (Playbot) capable of opening a door equipped with a lever-type handle. Playbot locates the door and navigates to it such that the door handle is entirely visible to the robot’s camera system. Door handle candidates are found using a feature-based algorithm, and the final candidate is selected using a template-matching approach in hue-saturation-value (HSV) space. Once a candidate is found, its pose is estimated by fitting a line to the handle through a feature-rich region such as the key slot. While applying a forward force, the robot’s arm executes a circular arc through the calculated contact point pushing the door open. The approach developed by Andreopoulos and Tsotsos [2008] is only applicable to doors equipped with lever-type handles that open away from the robot.

Kragic et al. [2002] modelled the handle in image space as a rectangular region and a crossed set of lines. After the vision system detects that a door handle is visible, the model is used to localize the handle in the image plane. The end-effector is then controlled to move toward the handle while keeping it in the middle of the image. When the door handle reaches a certain size in the image plane, a blind grasp is executed. The 2D model used can only represent lever-type handles.

Chung et al. [2007, 2009] continued the work started by Rhee et al. [2004]. The knob’s approximate position is identified using a single camera, and the end-effector is then used to repeatedly touch the knob. The final shape is estimated by fitting a circle to the data using a voting algorithm.

A method is presented by Rusu et al. [2009b] for localizing the position of a door and handle using a 3D point cloud obtained with a laser scanner. The door handle is detected by examining the geometric structure and the surface reflectance of the subset of points where the handle is expected to be located on the door.

Klingbeil et al. [2008, 2010] used a learning algorithm to locate all possible door handle candidates in view after the robot has navigated to a position in front of the door. 3D data is then extracted from the most likely location to contain the handle. Principal component analysis (PCA) is used to estimate the shape of the surface. Locations of the visual features are used to estimate the position of the handle and to plan a grasp.

Model-based pose estimation methods can also be used to provide an accurate location of the handle [Petrovskaya and Ng, 2007]. These methods exploit a 3D model of the object and attempt to match edges, interest points, or image gradients between the scene and the model. They usually produce accurate results, but require object models [Goddard, 1997; Lowe, 1987, 1991; Prats et al., 2010; Rosenhahn, 2003; Srinivasa et al., 2010].

### Summary

The main limitation of currently available methods is their inability to localize different types of handles. While some methods constrain the problem to lever- or knob-type handles explicitly, others are limited to a specific type of handle by the model used to represent it.

### **1.2.3 Object Segmentation in Online Model Acquisition for Grasping**

Following the review of image segmentation techniques in the previous section, the discussion is focused on methods that are developed for model acquisition used to guide robotic manipulation tasks. Methods that include depth information, but may not be used for robotic applications directly, are also discussed.

In addition to acquired images, mobile robots have access to depth information. Using stereo cameras or other 3D sensors, they are able to perceive not just the projection of the world onto a 2D plane, but also its metric 3D structure. This has the potential to remove ambiguities in images, and can enable successful object segmentation in visually complex environments.

Most of the segmentation methods used for robotic grasping build on the base of existing image segmentation techniques. For this reason, the following discussion is separated into sections corresponding to the basic segmentation techniques used as a foundation for the methods being discussed.

#### **Dominant Plane and Model Fitting**

Methods in this category function by fitting geometric models directly to the 3D data as part of the segmentation algorithm. While other segmentation techniques can be used later in the process, these methods rely on an assumption that a subset of the 3D data fits one or more geometric models, and it is always assumed that the object rests on a planar surface, often corresponding to a table or other supporting surface.

The algorithms first detect the dominant plane in the scene. Once the dominant plane is detected, points that belong to it are removed from further processing. The remaining points are then clustered into objects. The clustering is performed either by fitting

geometric primitives such as boxes or cylinders [Marton et al., 2009], by a 3D region growing algorithm [Rusu et al., 2009a], or using mean shift in 2.5D image-disparity space [Rasolzadeh et al., 2010]. To be able to segment an object resting on top of another object with a planar top, Rusu et al. [2009a] repeat the segmentation process again for each detected cluster.

In addition to the assumption that the objects rest on a planar surface, it is implicitly assumed by Rusu et al. [2009a] and Rasolzadeh et al. [2010] that the objects are separated in space. Without further processing steps, these techniques cannot be used in cluttered environments when the objects touch each other.

### **Feature Space Clustering**

Since clustering techniques do not limit the type of information included in the feature vector, they can be applied with little modification to segmentation scenarios where appearance and 3D data is available.

Harville et al. [2001] use Gaussian mixture models to calculate background and foreground models using a colour and depth feature vector. Segmentation is done by assigning each pixel to the closest model. Bleiweiss and Werman [2009] use a mean shift algorithm to segment a single target from a static background using colour and depth.

The main limitation of these methods is the way depth is incorporated into the segmentation. Because depth is used directly as part of the feature vector, segmentation at points where the object meets its supporting surface may fail if the object's appearance is similar to that of the supporting surface.

### **Graph-Based Clustering**

Similarly to feature space clustering techniques, graph-based clustering methods can be applied to segmentation problems where appearance and geometric data is available with

little modification. The only requirement is a suitable definition of similarity between adjacent graph vertices that appropriately incorporates the multi-modal data.

Cigla and Alatan [2009] use colour, depth, and motion cues. The colour image is initially over-segmented into super pixels, which are then clustered using a graph-based algorithm. Hirano et al. [2005] use mean shift to pre-segment a colour image into super pixels, which are then clustered based on curvature using the method proposed by [Shi and Malik, 2000]. The above methods perform segmentation in separate stages for colour and geometry. When appearance and geometric features are used separately, the relationship between the features is not fully exploited. Additionally, any under segmentation in the first step will be propagated to the final solution because further clustering can only combine the initial super pixels.

Rao et al. [2010] and Strom et al. [2010] present graph-based segmentation algorithms based on the work of Felzenszwalb and Huttenlocher [2004]. Rao et al. [2010] perform the segmentation using colour and depth, however the relative weight between the colour and depth features is determined off-line, and is not adaptive. In addition, depth does not provide segmentation cues in regions where surfaces meet smoothly. The segmentation method presented by Strom et al. [2010] uses a combination of colour and curvature. The graph is constructed based on a mesh derived from a 3D point cloud of the scene. The meshing process automatically removes edges from far away vertices, however the distance between vertices is not directly used for segmentation. As opposed to combining colour and curvature into a single weight, the segmentation has to satisfy two independent uniformity conditions: one for colour, and one for curvature. While this allows for the threshold to adapt uniquely to the two different features, the interrelationship between the features is not used.

## **MRF and CRF Based Methods**

Following the work on segmentation and Markov random fields by Boykov et al. [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006; Boykov et al., 1998], several segmentation

methods combining appearance and geometric cues based on random field models have been reported [Bjorkman and Eklundh, 2006; Bjorkman and Kragic, 2010a,b; Franke, 2011; Johnson-Roberson et al., 2010].

Bjorkman and Eklundh [2006] use stereo and disparity to segment objects from the background. The image is initially over-segmented into super pixels. The robot’s attention system is used to fixate the camera system on a region corresponding to the object of interest. Object’s model properties are initialized using this region. Graph cuts is then used to segment the object with super pixels acting as nodes in the graph. Since super pixels are utilized as the basis for segmentation, initial under-segmentation errors will be propagated to the final solution. Furthermore, disparity cannot be used as a segmentation cue in regions where the object meets a surface smoothly.

This issue is addressed by Bjorkman and Kragic [2010a,b] by assuming that the object rests on a flat surface. The segmentation is modelled as a labelling problem with three labels: object, surface, and background. The algorithm continuously improves the segmentation over a sequence of images. However, because only a single dominant surface is modelled, this method would encounter difficulties when segmenting objects that do not rest on a planar surface, or if one object rests on top of another object with a similar appearance.

Franke [2011] proposes an extension of the graph cuts method that uses depth data from a time of flight camera to initialize region models used for a graph cuts-based colour segmentation algorithm. In this approach, 3D data is used for initialization only and not for segmentation.

Johnson-Roberson et al. [2010] construct the graph based on a 3D mesh. The region term depends only on the colour model, and the boundary term is a colour difference divided by the Euclidean distance between vertices. No other geometric cues are used for region or boundary terms. While this can result in a significant improvement near object boundaries where a depth discontinuity is present, it does not help to distinguish boundaries at locations where an object smoothly meets a surface or another object.

Additionally, the relative importance of the depth cue is not taken into consideration. Two methods are provided to autonomously initialize the algorithm. The first method detects objects based on their distance from the dominant plane in the scene, colour and normal direction. The second method generates object seeds using image saliency techniques proposed by Rasolzadeh et al. [2010].

## Summary

Segmentation methods that use 3D information can be categorized broadly into methods based entirely on 3D data, commonly in the form of point clouds; and methods that combine appearance with 3D information.

Methods that utilize only 3D information have to make strong assumptions about the geometry of the scene. These techniques often require that objects rest on a planar surface, or be separated in space. Without further processing steps (for example, Rusu et al. [2009a]) these techniques cannot be used in the presence of clutter when the objects touch each other, or when some objects rest on top of others.

Approaches that combine appearance and geometric information do so by introducing 3D features such as depth, 3D position, or curvature into existing clustering or segmentation frameworks. The use of depth, motion, or position information in combination with appearance can result in an improved ability to identify which region in the image should belong to the object, and increase segmentation performance near object boundaries where a depth discontinuity is present.

However, depth cannot be used to detect boundaries at locations where an object meets a surface or another object. For this reason, most methods are only effective in uncluttered environments where objects are separated in space, and when the appearance of the objects differs from the surface they are placed on. To mitigate this limitation Bjorkman and Kragic [2010a,b] introduced a dominant plane model which enables effective segmentation of objects from their supporting surfaces. Nevertheless, current methods



are not capable of separating objects of similar appearance that are in contact with one another.

Depending on the appearance and geometry of the scene, some cues may be more significant for detecting object boundaries, and for separating an object from the background. Currently, when appearance and geometric features are combined, the relative significance of the various cues is either not considered, or it is assumed to be constant.

### 1.2.4 Distance Metric Learning

When multiple appearance and geometric cues are used for segmentation, it becomes important to consider which of the available cues the algorithm should pay attention to, and which cues should be disregarded. As stated in the previous section, current methods that combine appearance and geometric features do not examine the relative significance of available feature combinations.

Distance metric learning theory provides a means of learning a similarity or distance function from input data [Yang and Jin, 2006]. Provided with sets of similar and dissimilar data points, also referred to as must-link and cannot-link constraints respectively, a distance function can be estimated such that the distance between the points in the similar set is minimized, and the distance between the points in the dissimilar set is maximized. Utilizing a distance metric learned from data has been demonstrate to improve the performance of clustering and classification algorithms [Wang et al., 2006; Sobieranski et al., 2009, 2011; Xiang et al., 2008]. The application of distance metric learning to image segmentation has recently been examined by several authors [Wang et al., 2006; Xiang et al., 2008; Jia and Zhang, 2008; Sobieranski et al., 2009, 2011; Batra et al., 2010, 2011; Protiere and Sapiro, 2007].

Wang et al. [2006] present a graph based clustering method called Linear Neighbourhood Propagation. The method assumes that each point of data can be reconstructed as a linear combination of its neighbours. For each data point, the linear reconstruction weights

are calculated such that the difference between the measured data and reconstructed data is minimized. These weights are used as a measure of similarity between adjacent data points. Given a subset of labelled points, the weights are used to propagate the labels through the unlabelled data. The application of the clustering method is demonstrated on object recognition, and on supervised colour image segmentation problems.

Protiere and Sapiro [2007] present a segmentation algorithm based on a learned Geodesic distance. The Geodesic distance is determined based on the estimated probability density functions of user labelled regions. The probability that a pixel belongs to each region is then estimated as the ratio of Geodesic distances from the pixel to the region under consideration compared to the total distance from the pixel to all of the regions.

Xiang et al. [2008] present a method for learning a Mahalanobis distance from must-link and cannot-link constraints. The distance metric is learned by maximizing the ratio of must-link to cannot-link distances. The distance is used in a K-nearest neighbour classifier to perform clustering, interactive colour image segmentation, and face pose estimation.

A colour image segmentation method is presented by Jia and Zhang [2008]. A distance metric learned from partially labelled data is used to estimate edge costs for a graph based clustering algorithm. The method uses an Expectation Maximization strategy similar to the method presented by Rother et al. [2004], where the segmentation and the distance function are estimated iteratively. An initial estimate of the distance function is obtained from user input using linear discriminant analysis. The algorithm then iterates between segmentation using the current estimate of the distance function, and estimating the distance function holding the segmentation constant. The distance is estimated using gradient descent with respect to the segmentation cost function which guarantees that the cost will decrease with every iteration. The use of a learned distance is shown to improve segmentation accuracy compared to segmentation with a Euclidean distance.

Sobieranski et al. [2009, 2011] present a region based colour image segmentation algorithm based on a learned Polynomial Mahalanobis Distance. The distance is learned from

similar points only, and does not utilize points from different label classes. The learned distance is shown to improve segmentation accuracy when compared to colour space specific distance metrics, and when compared to the regular Mahalanobis Distance.

## Summary

Distance metric learning can be used to estimate the relative significance of segmentation cues, however, the application of distance metric learning method to image segmentation has been limited to segmentation of colour images. Additionally, to the best of the author's knowledge, with the exception of the work presented by Batra et al. [Batra et al., 2010, 2011], the use of distance metric learning in a segmentation method based on probabilistic graphical models is not explored in the literature, and it has not been applied to segmentation methods that combine appearance and geometric data. While Batra et al. [Batra et al., 2010, 2011] mention the use of distance metric learning, their work is focused on simultaneous co-segmentation of multiple colour images, and the effects of using distance metric learning as part of the segmentation algorithm are not discussed.

## 1.3 Problem Statement

### 1.3.1 Segmentation for Door Handle Localization

The problem of door handle localization is defined as that of estimating the geometric shape, size, and orientation (pose) of a door handle. To accomplish this, the 3D structure corresponding to the handle must be separated from the door, so that a model of the handle can be constructed. Figure 1.2 shows an example of this problem where an image of the observed scene (Figure 1.2(a)), and the final detected handle (Figure 1.2(b)) are

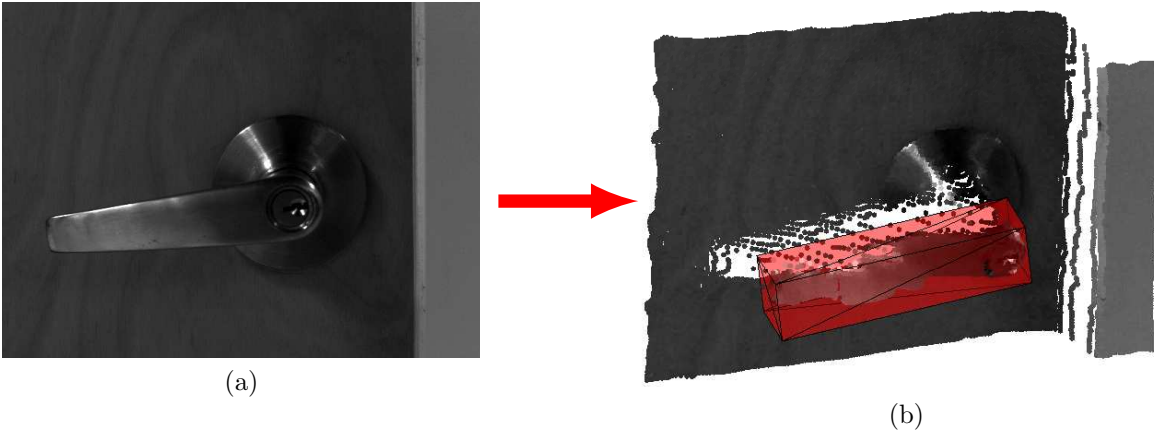


Figure 1.2: Door handle localization. (a) the image of the scene. (b) detected door handle shown over the image as a red box.

presented. The set of technical challenges that must be addressed to accomplish the above stated goals can be summarized in the following categories:

**Unknown geometry of the handle.** If a robot needs to navigate effectively in human environments, it must be able to open doors equipped with different types of handles. It must be able to operate both knob and lever type handles, and not get confused by geometric variations of each type. For this reason, the developed method must not make assumptions about the geometry of the door handle which is to be segmented and localized.

**Unknown appearance of the handle and the door.** Handles and doors can have different appearances. Even in a single home, or office it is not uncommon to have doors that are of different colour or texture, and handles of different types and appearances. More importantly, if the robot is to be deployed in unknown buildings, it would not be possible for the appearance of handles and doors to be known a priori.

### 1.3.2 Object Segmentation for Model Acquisition to Guide Robotic Grasping

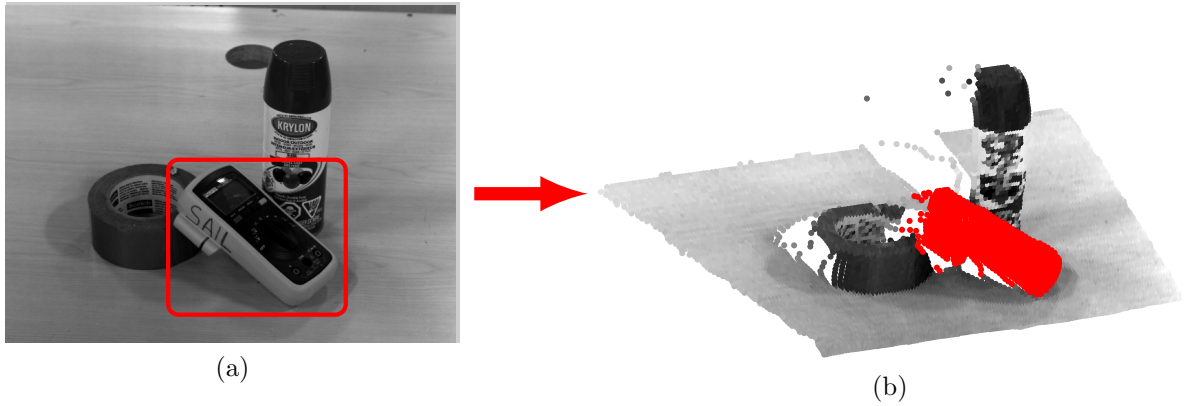


Figure 1.3: Object segmentation. (a) grayscale image of the scene. (b) reconstructed point cloud, with the segmented object highlighted in red.

Given a 2D image of the scene and its corresponding depth map, the goal is to separate the 3D structure of the object to be grasped from other objects in the scene and from the environment. Specifically, focus is placed on the segmentation of relatively small and geometrically convex objects, which can be manipulated by a human with one hand.

The segmented 3D structure of the object can then be used to plan and execute a grasp, while avoiding collisions with the environment. The process is demonstrated in Figure 1.3 where a partial 3D model of a multimeter is acquired by segmentation. For this task to be performed successfully the following challenges need to be overcome:

**Visually complex environment.** A large number of objects and surfaces in human environments are multicoloured and multitextured. When objects are placed in close proximity, it becomes difficult to distinguish object boundaries visually. Additionally, uncontrolled and varying lighting conditions create artificial edges, shadows and highlights complicating the problem further.

**Geometrically complex environment.** A large number of differently shaped objects

makes the environment complex from a geometric perspective. The complexity increases in cluttered environment when objects are in contact with one another, or when objects rest on top of other objects. Such situations can invalidate even common assumptions that an object rests on a flat surface.

**Unknown appearance and geometry of the object and the background.** Due to the large number of objects the robot may have to interact with, the visual and geometric complexity of the environment, and because the robot may be deployed in unknown locations, no appearance or geometric models of the object or the background are assumed to be available.

**Integration of available segmentation features.** When multiple segmentation features (cues) such as colour and depth are used simultaneously, the relative significance of these cues needs to be determined. This issue has to be addressed for the segmentation algorithm to be able to adapt to various scenes.

## 1.4 Contributions

### 3D Segmentation for Door Handle Localization (Chapter 2)

A method for localizing a door handle based on a 3D segmentation method, and an analysis of the geometric structure of the door and handle environment is presented. As opposed to detecting the handle, it is shown how commonalities in the geometry of the environment where the handle is mounted can be used to segment all parts of the structure except the door handle. The segmentation is performed by detecting the dominant plane in the scene, which corresponds to the door. Points are then projected onto the plane's normal. The projected points form clusters, with each cluster corresponding to one or more elements in the scene. To ensure that unique elements are detected, clusters are separated into connected components in the image. Background elements are then

removed by eliminating components that pass vertically through the field of view of the camera. The largest remaining element is taken as the door handle. The localization is then performed by fitting a bounding box around the door handle. The simple box model efficiently provides the necessary information to guide the grasping of the handle. The method is experimentally shown to be able to segment and localize door handles of multiple geometries and appearances with no prior information about their appearance or geometry.

### **Object Segmentation Using a Combination of Appearance and 3D Geometric Features (Chapter 3)**

A method for segmenting objects in cluttered environments is developed. The proposed method is based on the probabilistic conditional random fields (CRF) framework [Lafferty et al., 2001] and the graph cut energy minimization technique [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006]. To enable robust performance in visually complex and cluttered environment, multiple appearance and geometric cues are utilized. A method to autonomously initialize the algorithm based on closed contours of depth edges and high curvature edges is also presented. The algorithm takes as inputs a colour or monochrome image of a scene, and a corresponding depth map. Secondary features such as texture and curvature are calculated from the inputs. The scene is modelled as a graph with pixels acting as nodes, and edges connecting adjacent pixels in an 8-connected neighbourhood system. Following object detection, an object is selected as the target. For the target object, region and background models, as well as a covariance matrix used to weigh the similarity between adjacent pixels are constructed using the appearance and geometric features in the object and background seed regions. Segmentation is then performed using graph cuts, with graph edge weights set using the available appearance and geometric data, the object and background models, and the covariance matrix. The method is validated experimentally by examining its performance on a number of scenes of varying complexities.

## Distance Metric Learning for Object Segmentation (Chapter 4)

An improved segmentation algorithm is developed which incorporates distance metric learning (DML) [Yang and Jin, 2006] to estimate a distance function on multiple appearance and geometric features. The distance function is estimated such that it separates the points that belong to the object seed region from points that belong to the background by a unit margin. The learned distance function is then used to weight the relative significance of various segmentation cues. The improved segmentation algorithm is initialized by providing a single point on a target object. Points are assigned to object and background seed regions based on their Euclidean distance from the initially provided point. The method then operates in an iterative matter, alternating between estimating object and background models and learning the distance function, and performing segmentation. In contrast to the method presented in Chapter 3, the improved method estimates the weights using a combination of similarity and dissimilarity constraints. This allows it to better determine the ability of available features to separate the object from the background. As an additional contribution, a model for incorporating depth into the region model is presented. The model is based on distributions of algebraic distances between 3D points and an ellipsoid fitted to the object seed points using least squares. The model is incorporated into the segmentation algorithm to improve its ability to localize objects, and to remove the need to add hard constraints to object seed regions.

### 1.5 Outline

This chapter acts as an introduction, presenting the motivation for the work carried out in this thesis. It provides a review of relevant literature, presents the problem statements, and states the contributions of this work. Chapter 2 presents the development of a 3D segmentation algorithm to enable door handle localization. The design of a segmentation algorithm for acquiring object models to guide robotic grasping of unknown objects is presented in Chapter 3. An improved segmentation method incorporating distance metric



learning and an ellipsoidal object model is developed in Chapter 4. In Chapter 5 the accomplishments of this work are summarized and the direction for future research is discussed.



# 3D Segmentation for Door Handle Localization

---

Door opening is a key requirement for autonomous robotic operation in human environments. For this to be possible, a method is needed to guide robotic grasping of the door handle prior to the robot being able to manipulate it. In this chapter, a method for localizing the door handle based on a 3D segmentation method which utilizes the geometry of the background structure is proposed.

The chapter is organized as follows. Section 2.1 presents an overview of the proposed algorithm, and describes the geometry of the problem. Section 2.2 describes the method used to reconstruct 3D data from optical flow. Section 2.3 presents the method used to segment and then localize the door handle. Section 2.4 presents experimental validation of the proposed method. Concluding remarks are given in Section 2.5.

## 2.1 Algorithm Overview

In this section, an overview of the proposed algorithm, any assumptions made, and a brief description of major calculation steps are provided.

The presented approach utilizes the common structure of the door-handle environment. While the geometry and appearance of the handle may vary, the geometry of the supporting structures (consisting of the door, walls, columns and door frames) is consistent. These structures, together with the door handle, will be referred to as the elements in

the scene. By detecting and segmenting the background elements, it becomes possible to isolate and localize the door handle. This is in contrast to most published methods, which attempt to localize the handle directly, and as result often require models that restrict their applicability to a specific class of handles. The approach presented here does not require any knowledge of the handle’s appearance or geometry, allowing it to be applicable to any type of door and handle assuming the 3D structure of the scene can be reconstructed. Following segmentation, a simple geometric model can be constructed to guide the grasping of the handle by a mobile robot.

An assumption is made that the mobile robot observing a door and handle scene is already positioned in front of the door, and the handle is visible in the camera’s field of view (FOV). This assumption is justifiable for several reasons. First, a number of approaches have been presented that enable a mobile robot to position itself in front of a door, such that the door handle would be visible to the camera [Andreopoulos and Tsotsos, 2008; Klingbeil et al., 2008]. Second, the system under consideration utilizes an end-effector mounted camera which can be re-positioned to obtain a more favourable view of the handle. This re-positioning procedure could be performed by the operator if the robot is manually controlled, or autonomously (e.g. Kragic et al., 2002).

The proposed algorithm operates in several steps which are shown in Figure 2.1. First, optical flow is used to extract 3D information and generate a point cloud of the scene. Next, door handle segmentation is performed in several stages. A random sample consensus (RANSAC) algorithm [Fischler and Bolles, 1981] is used to find a plane that best fits the points corresponding to the door in the point cloud. All of the points are projected onto the normal to this plane, and then separated into clusters using a mean shift algorithm [Fukunaga and Hostetler, 1975; Cheng, 1995; Comaniciu and Meer, 2002]. A brief description of the mean shift algorithm is provided in Appendix B. A region with a high density of points is considered a cluster, and any point within that region is said to belong to that cluster. To ensure that the clusters represent contiguous region in the image, the clusters are further separated into connected components in the image. A connected component is formed by a group of pixels that are all adjacent to each other

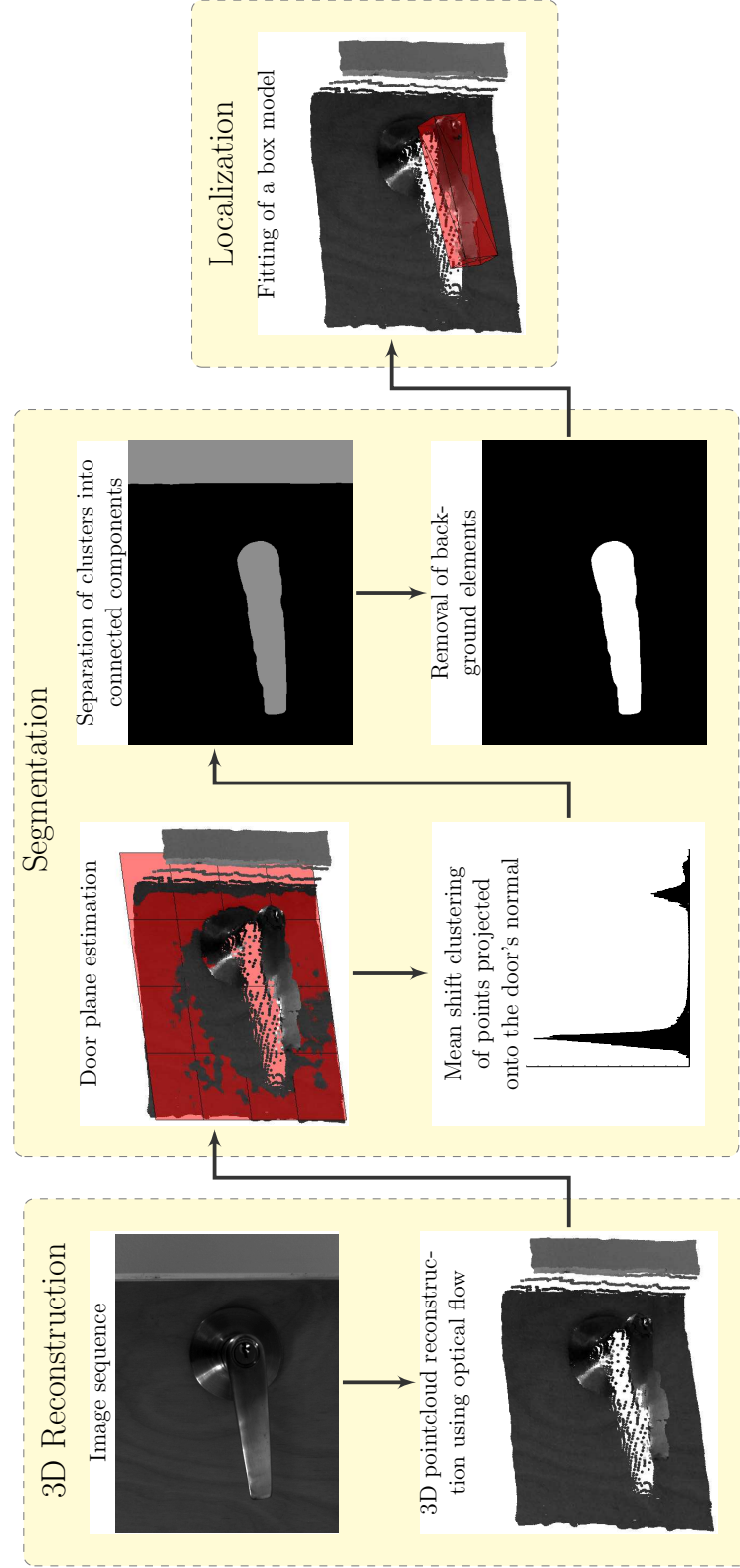


Figure 2.1: Sequence of processing steps in the proposed algorithm.

vertically, horizontally, or diagonally. The door, and any other structural elements are removed by detecting component that extend vertically past the FOV of the camera. The largest component that does not vertically extend to the top and bottom edges of the image is selected as the door handle. All other points are discarded. Finally, a bounding box is fitted around the set of handle points along their principal components. The bounding box is aligned along the largest principal component of the handle point cloud, and its top face is aligned to be parallel with the door plane.

The box efficiently provides the necessary information for the mobile manipulator to grasp the handle enclosed in it. Door opening can then be performed using the strategy reported by Liu et al. [2008, 2009]. By segmenting the handle, and constructing the model online, the algorithm presented here allows the robot to localize a handle of arbitrary shape and appearance, while not requiring a large library of door handle models. Additionally, this procedure utilizes only a single end-effector mounted camera.

## 2.2 3D Data Acquisition

This section describes the process used to construct the point cloud of the scene (see Cyganek and Siebert [2009]). The reconstructed point cloud is then used in the segmentation of the door handle, and the construction of the handle box model.

The vision system used is a single camera mounted on the end-effector of the robot. The camera is positioned to take several images of the door handle. These images are used in pairs, such that optical flow can be used with known position and orientation of the camera at each frame in order to recover a dense depth map of the scene.

Figure 2.2 shows position vectors  $\mathbf{p}_C$  and  $\mathbf{p}_B$  of a point in space with respect to the camera reference frame  $C$  and the robot body reference frame  $B$ . The frame  $C$  is located at  $\mathbf{c}_B$  with respect to the robot body frame  $B$ . The rotation matrix from frame  $C$  to

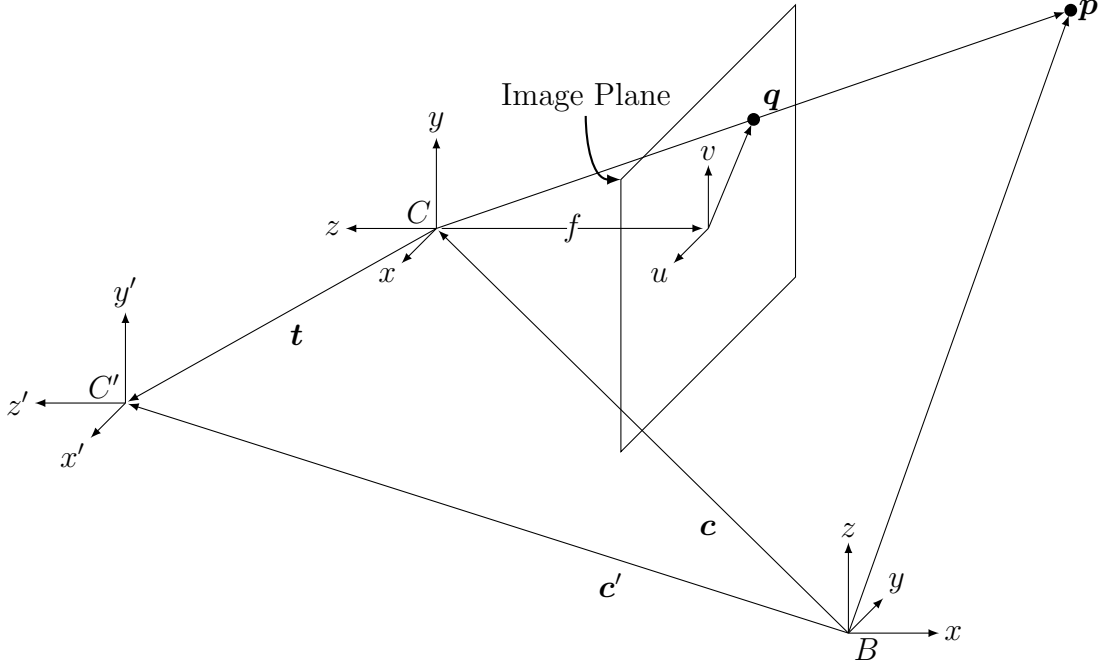


Figure 2.2: Coordinate frames associated with camera and robot body.

frame  $B$  is denoted as  $\mathbf{R}_{BC}$ . The vector  $\mathbf{q} = [u, v]^T$  denotes the projection of the point  $\mathbf{p}_C = [x_C, y_C, z_C]^T$  onto the image plane in frame  $C$ :

$$\mathbf{q} = \frac{f}{z_C} [x_C, y_C]^T \quad (2.1)$$

where  $f$  is the focal length of the camera lens. The *motion field*  $\dot{\mathbf{q}}$  is the motion of points in the scene as observed in the image plane. Given a sequence of images, the motion field  $\dot{\mathbf{q}}$  can be estimated using optical flow, and the depth of each pixel in the image can be recovered. From Equation (2.1),  $\mathbf{q}$  is the projection of a point  $\mathbf{p}$  when the camera reference frame  $C$  is located at  $\mathbf{c}_B$ . Let  $\mathbf{q}' = \mathbf{q} + \dot{\mathbf{q}}$  be the estimated projection of the same point captured when the camera frame  $C'$  is at position  $\mathbf{c}'_B$ . If the rotation between the two cameras is  $\mathbf{R}_{CC'}$ , and the relative position of the cameras is  $\mathbf{t}_C = \mathbf{R}_{BC}^T (\mathbf{c}'_B - \mathbf{c}_B)$ , then the location of the point  $\mathbf{p}_C$  can be determined by finding the point of closest approach between the two rays created by the points  $[\mathbf{q}^T, f]^T$ ,  $[\mathbf{q}'^T, f]^T$ , and the camera centres

$\mathbf{c}$  and  $\mathbf{c}'$ . Let  $\mathbf{r} = \alpha \mathbf{v}$  be the ray projected through the centre of the camera  $\mathbf{c}$  and the point on the image plane  $\mathbf{v} = [\mathbf{q}^T, f]^T$ . Let the second ray be  $\mathbf{r}' = \mathbf{t}_C + \beta \mathbf{v}'$  where  $\mathbf{v}' = \mathbf{R}_{CC'}[\mathbf{q}'^T, f]^T$ . The point at which the distance between the two rays is minimized can be found as (see Cyganek and Siebert [2009]):

$$\alpha \mathbf{v} - \beta \mathbf{v}' + \gamma (\mathbf{v} \times \mathbf{v}') = \mathbf{t}_C \quad (2.2)$$

When the rays  $\mathbf{v}$  and  $\mathbf{v}'$  are not parallel, the parameters  $\alpha$ ,  $\beta$ , and  $\gamma$  in Equation (2.2) can be determined as:

$$[\alpha, \beta, \gamma]^T = (\mathbf{A}^T \mathbf{A})^{-1} \mathbf{A}^T \mathbf{t}_C \quad (2.3)$$

where

$$\mathbf{A} = [\mathbf{v}, \mathbf{v}', (\mathbf{v} \times \mathbf{v}')] \quad (2.4)$$

The position of  $\mathbf{p}_C$  can then be calculated as:

$$\mathbf{p}_C = \alpha \mathbf{v} + (\beta/2)(\mathbf{v} \times \mathbf{v}') \quad (2.5)$$

Only the depth value from each point  $\mathbf{p}_C$  is used.

The optical flow algorithm used for 3D reconstruction is based on the approach outlined in Zach et al. [2007]. This algorithm uses variational methods to calculate the optical flow between two image frames and was chosen for the present algorithm because of its proven performance [Baker et al., 2007, 2009], and because it can be implemented in real-time on modern graphics cards. For a demonstration of the accuracy of 3D reconstruction



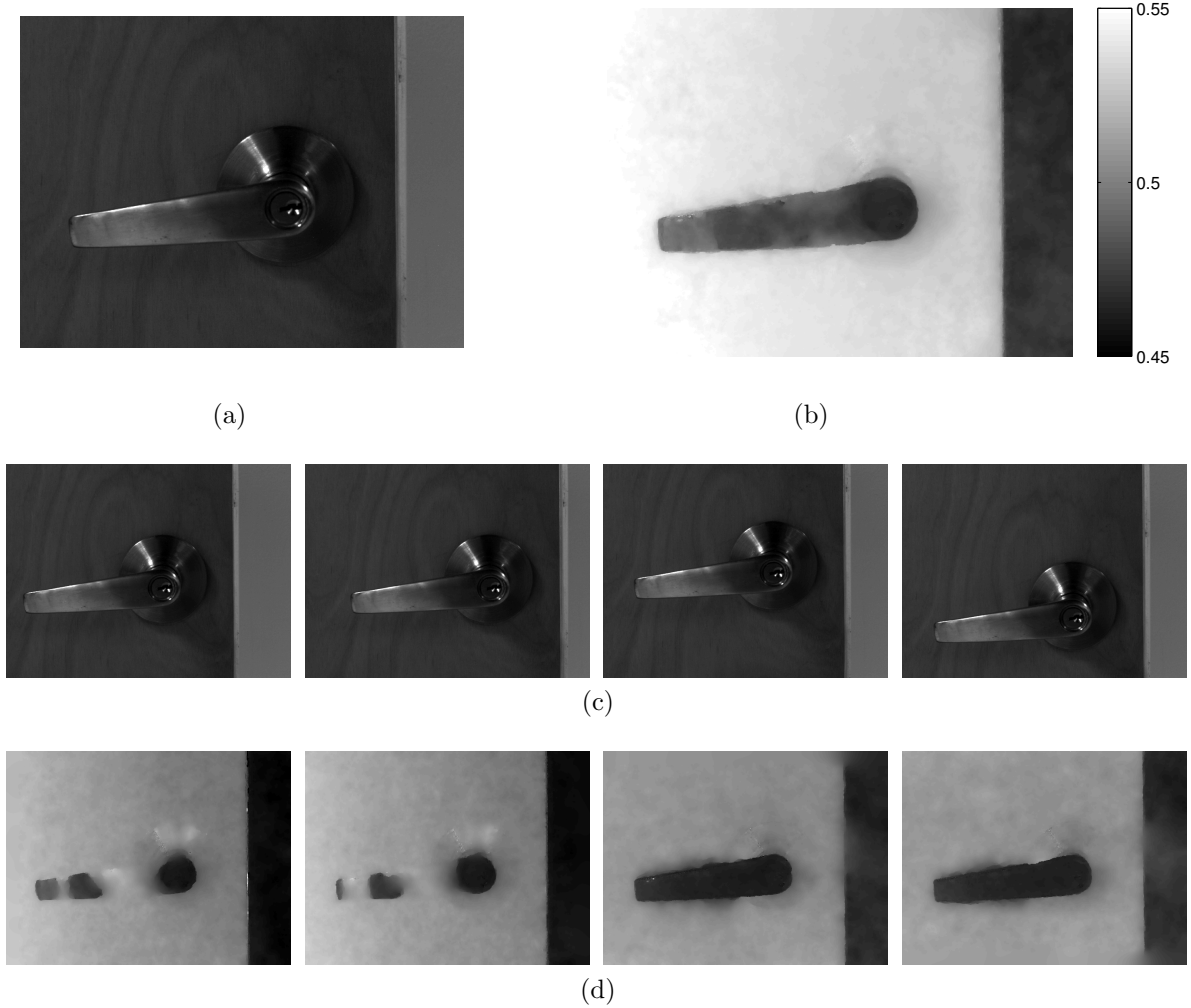


Figure 2.3: The central frame of the image sequence (a), with the corresponding filtered depth map (b). Several frames of the image sequence (c), and their associated unfiltered depth maps (d) are also shown. A portion of a wall protruding forward from the plane of the door can be seen on the right-hand side of the images.

using the chosen optical flow algorithm see Appendix D. For details of implementation, see Zach et al. [2007].

The final depth estimate is obtained by applying a median filter to a  $5 \times 5 \times N_{maps}$  window around each pixel, where  $N_{maps}$  is the number of available depth maps. The median filter is used to smooth the depth maps while preserving edges, and to remove

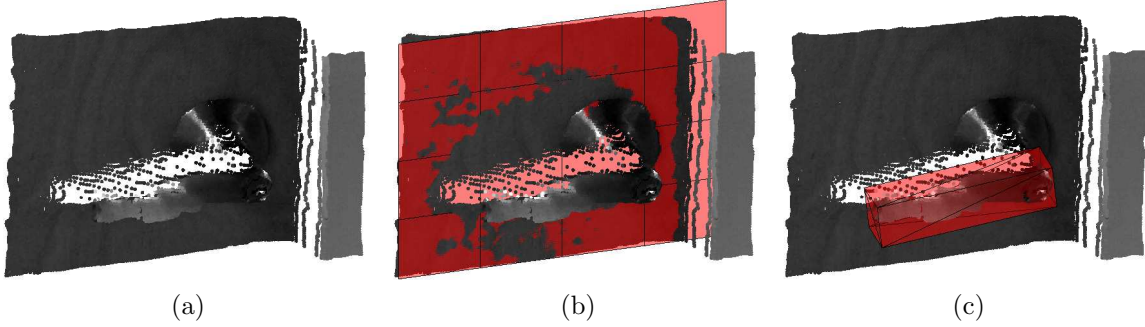


Figure 2.4: Reconstructed point cloud (a), with the dominant plane (b), and the door handle bounding box (c) shown.

individual pixel outliers. Figure 2.3(a) shows one of the images used in the experiment. The final filtered depth map is shown in Figure 2.3(b). Several images of the sequence used for 3D reconstruction are shown in Figure 2.3(c). One image in each direction of the center is shown. Depth maps obtained from these images are shown in Figure 2.3(d). The position of each pixel in the camera frame is calculated using the filtered depth map. As a final step, the points are transformed into the robot body reference frame  $B$ , and the point cloud  $\{\mathbf{p}_B\}$  is obtained as:

$$\mathbf{p}_B = \mathbf{c}_B + \mathbf{R}_{BC}\mathbf{p}_C \quad (2.6)$$

Figure 2.4(a) shows the point cloud  $\{\mathbf{p}_B\}$  obtained from the experimental images (Figure 2.3).

## 2.3 Handle Segmentation and Localization

The current section describes the process of segmenting and localizing the door handle. This task is accomplished by detecting and removing all background elements, leaving the door handle as the only significant element in the scene.

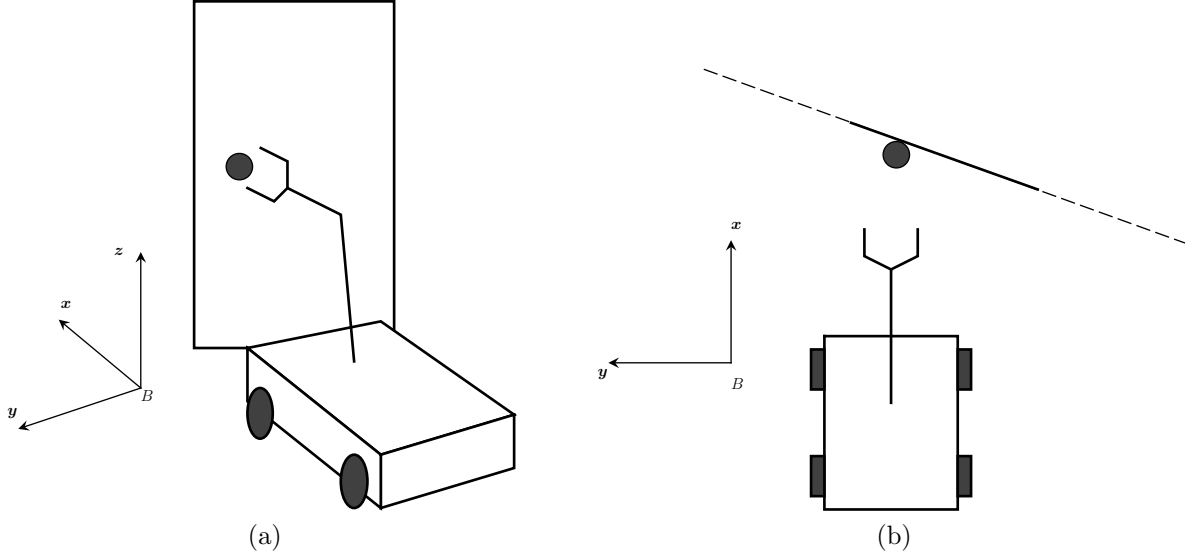


Figure 2.5: Geometry of the door in 3D space (a), and in the horizontal plane of the robot body reference frame (b).

The door is expected to be the largest element in the view. Due to its shape it is detected by finding the dominant plane in the scene. The plane of the door is found by projecting the entire point cloud onto a horizontal plane (in our case the  $x$ - $y$  plane of the robot body reference frame  $B$ ), and then finding a line that best fits the projected point cloud using the RANSAC algorithm [Fischler and Bolles, 1981]. The calculation is performed in 2D to ensure that the resulting plane is vertical in the robot body frame. This is shown in Figure 2.5 where the door in 3D (Figure 2.5(a)) can be seen to form a line in the horizontal plane (Figure 2.5(b)).

The RANSAC is a method that attempts to find a set of model parameters to maximize the number of points that are considered inliers with respect to the model. Let  $\{\mathbf{p}_H\}$  be the set of points in the horizontal plane of the robot body reference frame. The model to be found consists of a point on the best fit line  $\mathbf{p}_l$  and a vector perpendicular to the line, called the normal  $\hat{\mathbf{n}}_l$  that maximize the number of points within a distance  $t$  of the line which are considered inliers:

$$\max_{\mathbf{p}_l, \hat{\mathbf{n}}_l} \sum_{\mathbf{p}_k \in \{\mathbf{p}_H\}} \delta(|(\mathbf{p}_k - \mathbf{p}_l)^T \hat{\mathbf{n}}_l| < t) \quad (2.7)$$

where  $\delta(\cdot)$  is a function which is equal to one if the condition is true, and zero otherwise. The threshold  $t$  was set to 1cm based on experimental observations of the noise present in the point clouds, and based on the geometry of a typical door and handle scene.

The RANSAC procedure consists of iteratively choosing two points  $\mathbf{p}_a, \mathbf{p}_b \in \{\mathbf{p}_H\}$ . One of the points is taken as the point on the line  $\mathbf{p}_l = \mathbf{p}_a$ . The normal is estimated as a vector perpendicular to the ray connecting the two points:

$$\mathbf{n} = \begin{bmatrix} 0 & -1 \\ -1 & 0 \end{bmatrix} (\mathbf{p}_b - \mathbf{p}_a) \quad (2.8)$$

The vector  $\mathbf{n}$  is normalized as a last step in the model calculation  $\hat{\mathbf{n}} = \mathbf{n}/|\mathbf{n}|$ . Once the model is obtained Equation (2.7) is evaluated to determine the number of inliers that support the current model. The procedure is iterated, and the model with the largest number of inliers is chosen. The final model is fitted using least squares to the set of inlier points only.

The final normal vector is estimated using a least squares fitting procedure on the set of inlier points. Let  $\{\mathbf{p}_I\}$  be the set of inlier points obtained using the RANSAC procedure. The covariance matrix of the set of inlier points can be calculated as:

$$\mathbf{M} = \frac{1}{|\{\mathbf{p}_I\}|} \sum_{\mathbf{p}_k \in \{\mathbf{p}_I\}} \mathbf{p}_k \mathbf{p}_k^T - \bar{\mathbf{p}}_I \bar{\mathbf{p}}_I^T \quad (2.9)$$

where  $|\{\mathbf{p}_I\}|$  denotes the number of points in the set, and  $\bar{\mathbf{p}}_I$  is the centroid:

$$\bar{\mathbf{p}}_I = \frac{1}{|\{\mathbf{p}_I\}|} \sum_{\mathbf{p}_k \in \{\mathbf{p}_I\}} \mathbf{p}_k \quad (2.10)$$

The normal vector to a line that best fits these points is the eigenvector corresponding to the smallest eigenvalue of the covariance matrix  $\mathbf{M}$ .

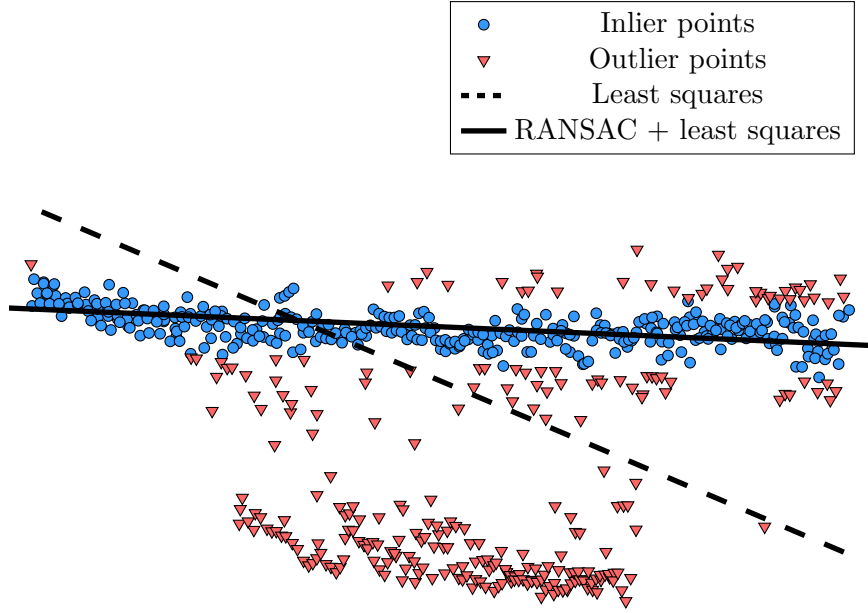


Figure 2.6: Comparison of best fit lines generated by the least squares method, and the RANSAC algorithm followed by least squares on the set of inlier points.

The RANSAC algorithm is used for this stage of the method because of its ability to deal with a large number of outliers in the data. Given that the door handle points and any part of the wall offset from the door are considered outliers, and since these points are expected to be distributed asymmetrically about the centroid of the point cloud, a least squares algorithm may return an incorrect estimate (Figure 2.6). The RANSAC algorithm performs much better under these conditions. Details of the RANSAC algorithm are provided in Appendix A.

Figure 2.4(b) shows a calculated door plane overlaid onto the point cloud of the door handle scene. Note that all of the important elements in the scene, including the handle and any other structural element, will form clusters at specific distances from the surface of the door. By clustering the points according to their projection in the direction of the plane’s normal the various elements in the scene can be identified.

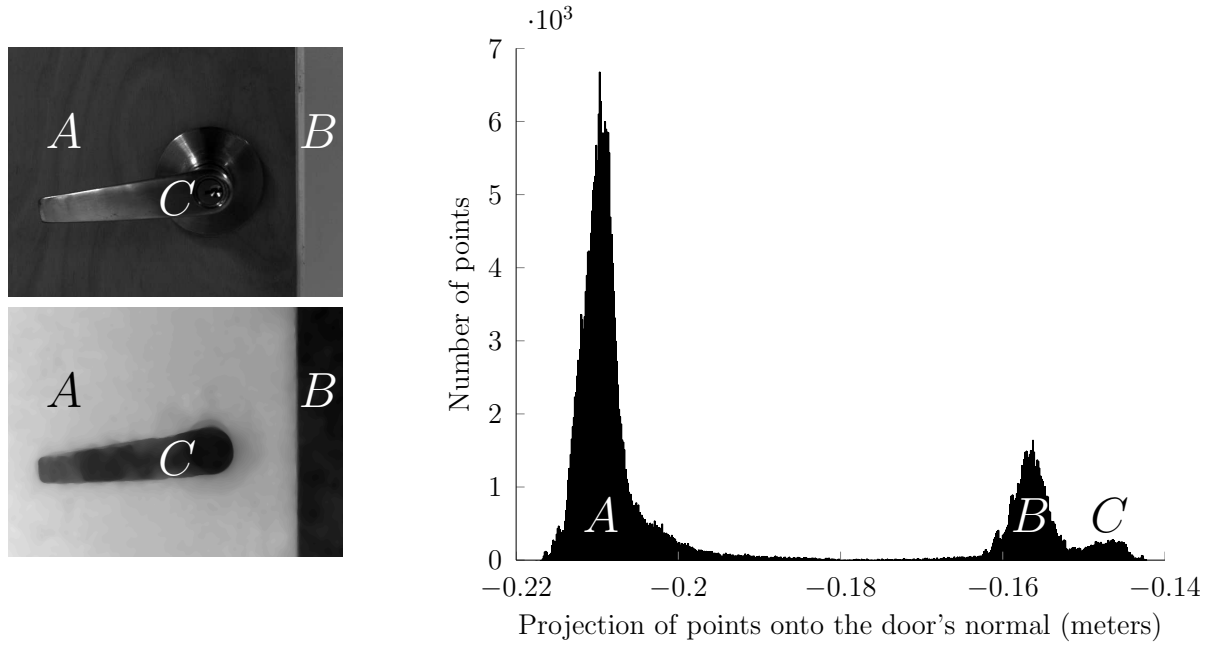


Figure 2.7: Histogram of points projected onto the normal of the door. The large peak (A) corresponds to points that lie in the door plane, while the smaller peaks correspond to points from a protruding portion of a wall (B), and the door handle (C).

Projecting the point cloud onto the normal vector of the door plane results in a set of one dimensional data. A histogram of points projected onto the normal of the door plane is shown in Figure 2.7. The values on the horizontal axis of the histogram are negative because of the orientation of door’s normal vector with respect to the robot’s body frame. Several clusters are expected to be present in the projected data, including a large cluster corresponding to the door, and a smaller cluster corresponding to the door handle. Additionally, other clusters may be present which would correspond to any other extruding element in the camera’s FOV, such as an extruding portion of a wall. To

identify these clusters, the mean shift algorithm is used [Fukunaga and Hostetler, 1975; Cheng, 1995].

The mean shift algorithm is an iterative mode seeking method. If the space of the one dimensional data resulting from projecting the point cloud into the door's normal is considered as a feature space, the mean shift algorithm allows for locating locally dense regions of the feature space, or clusters. Temporarily, let  $\{x_i\}$  be the set of point projections onto the door's normal. Mean shift clustering is performed by moving each point toward a locally denser region. Starting with an estimate of a cluster center  $y_t$  at some iteration  $t$  the estimate is iteratively moved toward the weighted mean of points surrounding it. The weights are determined by the kernel function. A Gaussian kernel is used resulting in the following update rule for the cluster center:

$$y_{t+1} = \frac{\sum_{x_i} \exp\left(\frac{(y_t - x_i)^2}{h^2}\right) x_i}{\sum_{x_i} \exp\left(\frac{(y_t - x_i)^2}{h^2}\right)} \quad (2.11)$$

In the above equation,  $h$  is called the bandwidth of the kernel. It defines the region and weight of points that contribute to the estimate of the new cluster center. A value of  $h = 1\text{cm}$  is used. The process can be initialized at each point. Points which converge to a cluster center close to each other are assigned to the same cluster. Performing this procedure for each point in the set can be slow. For this reason, a multi-start approach is used. An initial set of 1000 points evenly spaced through  $\{x_i\}$  is chosen, and mean shift is used to determine the cluster centers for these initial points. Any point in between one of the initial points and its cluster center is also assigned to the same cluster. This allows for most of the points to be assigned to a cluster efficiently. The mean shift algorithm is then applied to all remaining points individually. Additional information regarding the mean shift algorithm is provided in Appendix B. If only a single cluster is found following the mean shift procedure, then it is concluded that the handle is indistinguishable from the door based on the available 3D data.

It is possible that several features were initially grouped together into a single cluster,

either because the bandwidth of the mean shift kernel is too large, or because these objects are at the same distance away from the door and naturally form a cluster. An example of this scenario would be the handle and the extruding portion of the wall shown in Figure 2.3 and Figure 2.4. To address this issue and to ensure that the clusters represent contiguous elements in the image, the clusters are further separated into connected components (see Ritter and Wilson [2001]). The purpose of clustering the points and separating them into connected components is not to detect the handle, but to remove the points that belong to the door and any other structural elements. While RANSAC is able to detect a subset of points that belong to the door, it is unable to identify any of the other structural element. Additionally, any noisy points that may have been part of the door but fell outside of the threshold value would not be identified by RANSAC.

Structural elements extend from the floor to at least the top of the door, and pass vertically through the field of view of the camera. Any component that vertically passes through the image, simultaneously touching the top and bottom edges of the screen is considered as part of the background structure, and removed. This is achieved by determining whether any component contains pixels corresponding to the top and bottom rows of the image; these components are removed. Points belonging to the largest remaining component are labelled as the handle points  $\{\mathbf{p}_B^h\}$ .

As a final step, the handle is localized by fitting a bounding box to the handle points. The pose of the bounding box is calculated by determining the largest principal component of the handle point cloud  $\{\mathbf{p}_B^h\}$  when projected onto the plane of the door. The points in  $\{\mathbf{p}_B^h\}$  are orthographically projected onto the door by representing them in a new coordinate system  $D$ , resulting in a 2D set of points  $\{\mathbf{p}_D^h\}$ . This reference frame has the first axis as the cross product of the up-vector  $[0, 0, 1]^T$  and the door's normal  $\hat{\mathbf{n}} = [\hat{n}_1, \hat{n}_2, \hat{n}_3]^T$ , and the second axis as the up-vector

$$\mathbf{p}_D^h = \begin{bmatrix} -\hat{n}_2 & \hat{n}_1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{p}_B^h \quad (2.12)$$



The centroid  $\bar{\mathbf{p}}_D^h$  of the 2D handle set  $\{\mathbf{p}_D^h\}$  is

$$\bar{\mathbf{p}}_D^h = \frac{1}{|\{\mathbf{p}_D^h\}|} \sum_{\mathbf{p}_k \in \{\mathbf{p}_D^h\}} \mathbf{p}_k \quad (2.13)$$

The  $2 \times 2$  covariance matrix of  $\{\mathbf{p}_D^h\}$  can be calculated as

$$\mathbf{M} = \frac{1}{|\{\mathbf{p}_D^h\}|} \sum_{\mathbf{p}_k \in \{\mathbf{p}_D^h\}} \mathbf{p}_k \mathbf{p}_k^T - \bar{\mathbf{p}}_D^h (\bar{\mathbf{p}}_D^h)^T \quad (2.14)$$

The principal component vectors  $\mathbf{e}_{D,1}$  and  $\mathbf{e}_{D,2}$  of the 2D handle set are the eigenvectors corresponding to the eigenvalues,  $\lambda_1 > \lambda_2$ , of the covariance matrix  $\mathbf{M}$ . The principal component vectors are transformed back into the robot body frame  $B$  by augmenting each 2D vector with a zero in the 3rd component, and rotating the vector into the  $B$  reference frame

$$\mathbf{e}_B = \begin{bmatrix} -\hat{n}_2 & 0 \\ \hat{n}_1 & 0 \\ 0 & 1 \end{bmatrix} \mathbf{e}_D \quad (2.15)$$

After these vectors are transformed back to the  $B$  reference frame, a bounding box is constructed such that its long axis is aligned with the largest principal component  $\mathbf{e}_{B,1}$ , and its top face is aligned with the normal direction of the door  $\hat{\mathbf{n}}$ . The box is scaled such that it encompasses all of the points in the handle set (Figure 2.4(c)).

## 2.4 Experimental Validation

### 2.4.1 Experimental Setup

Experimental validation was performed using a 5-axis robotic arm which can move the camera mounted on its end-effector in 5-DOFs (Figure 2.8). The scene was captured

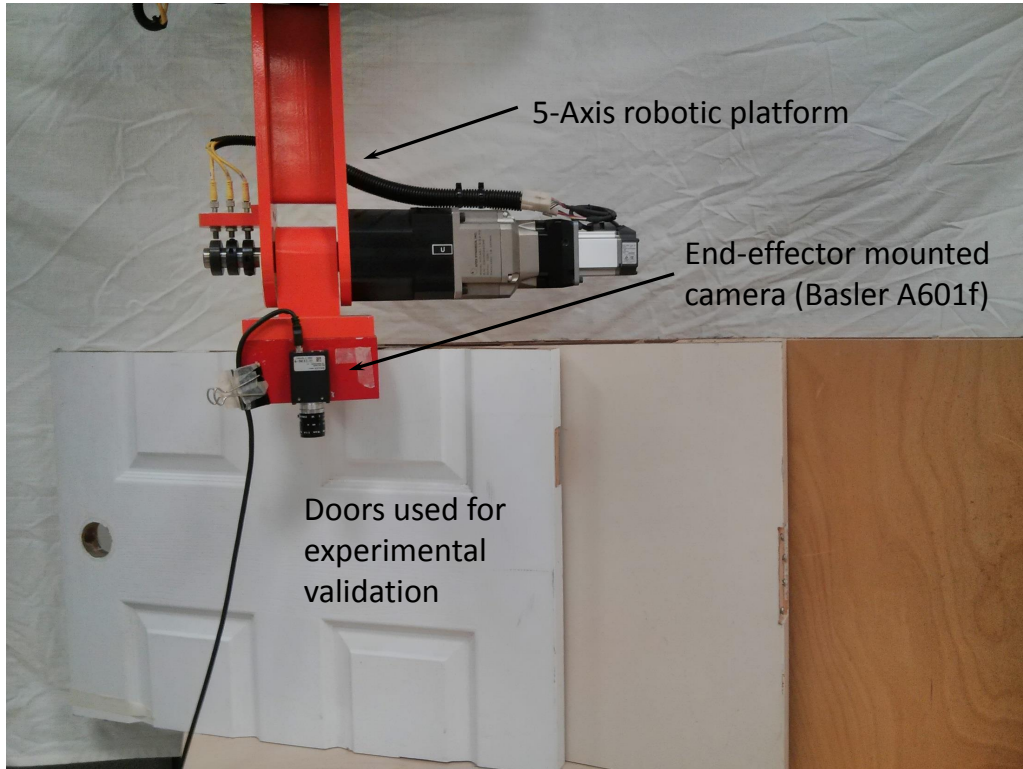


Figure 2.8: Experimental set up.

with a Basler A601f camera, with a resolution of  $656 \times 491$  pixels. Seven sample handles ranging from brushed metal, to specular and reflective were used. An example of a featureless, reflective knob-type handle is shown in Figure 2.9(a). Examples of brushed metal, lever-type handles are shown in Figure 2.9(b), and Figure 2.9(c). The handle in Figure 2.9(c) has a distinct feature in the form of a key hole on the front surface of the

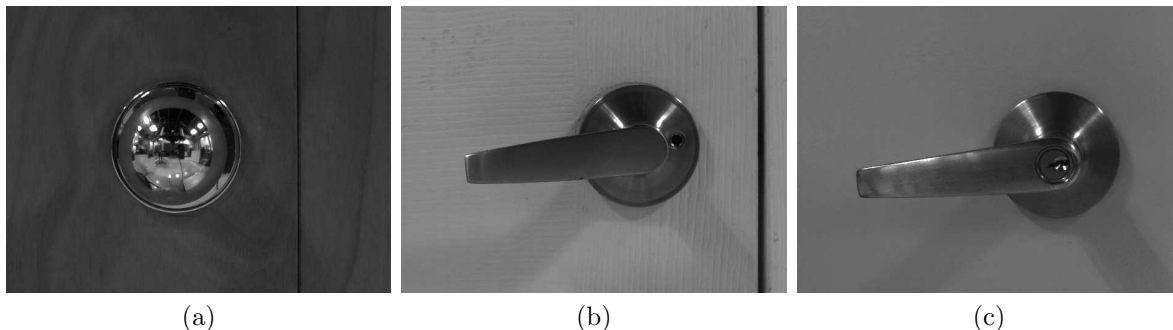


Figure 2.9: Examples of knob (a), and lever (b) and (c) type door handles. Handles shown in (a) and (b) are considered featureless. The handle shown in (c) has a lock feature on the front surface. A strongly textured door is shown in (a). A door with visible but not strong texture is shown in (b). A door with almost no texture is shown in (c).

handle. Three types of doors were used in the experiment: one with a strong wooden texture (Figure 2.9(a)), one with a light texture resulting from monochrome paint on wood grain (Figure 2.9(b)), and a featureless door with a smooth monochrome painted surface (Figure 2.9(c)). This was done to clearly demonstrate the effective range of the algorithm, where it fails and where it succeeds. A series of images were taken of each door handle attached to each of the doors under normal indoor lighting conditions. The door and handle were positioned approximately 0.5m from the camera. Encoders on the arm were used to gather position data of each joint. This data was used to calculate the pose of the camera mounted on the end-effector at the time each image was taken.

All steps of the algorithm were executed on a computer with an Intel Q9550 processor, with 8 GB of memory. With the exception of the optical flow method, all the steps of the algorithm were implemented in Matlab. Specifically, evaluation of the method's computational performance was performed in Matlab 8.1 (release R2013a). A summary of the method's computational performance is shown in Table 2.1. The table shows the mean and maximum time required for the algorithm to execute, as well as the time required to execute each step of the algorithm. The standard deviation associated with the mean execution time is also shown.

| Algorithm Step              | Time Required (s)    |                      |                      |
|-----------------------------|----------------------|----------------------|----------------------|
|                             | Mean                 | $\sigma$             | Maximum              |
| Optical flow for 16 images  | 307.788              | 3.494                | 318.604              |
| Depth map calculation       | 523.302              | 5.452                | 534.497              |
| Point cloud construction    | 36.036               | 9.042                | 50.141               |
| Plane estimation (RANSAC)   | 0.051                | 0.002                | 0.057                |
| Mean shift clustering       | 6.367                | 0.409                | 7.11                 |
| Connected components search | 0.027                | $4.6 \times 10^{-4}$ | 0.028                |
| Bounding box fitting        | $1.6 \times 10^{-5}$ | $9.2 \times 10^{-6}$ | $4.9 \times 10^{-5}$ |
| Combined algorithm          | 873.574              | 6.776                | 889.346              |

Table 2.1: Computational performance of the door handle segmentation and localization algorithm.

## 2.4.2 Experimental Results

An example of one of the experimental images, and its corresponding filtered depth map can be seen in Figure 2.3(a) and Figure 2.3(a) respectively. The 3D point cloud reconstruction of the scene is shown in Figure 2.4(a). Images in Figure 2.10 show the bounding box projected onto the front and top views of each handle. Figure 2.11 shows two instances where the door handle had to be localized while a part of it was outside of the camera’s field of view. Table 2.3 presents a summary of the most common errors observed in the shape and position of the bounding box. The types of errors, and their abbreviations are described in Table 2.2.

The most commonly encountered bounding box errors was when the box did not fully encompass the volume occupied by the door handle (error types F and S). While one case was observed where this occurred at the side of the handle (Figures 2.10(f)), in all other cases the handle protruded through the front face of the bounding box.

This defect occurred frequently with flat handles (Figures 2.10(k), and 2.10(r)) and spher-

| Error Type   | Abbreviation |
|--|--------------|
| Bounding box is larger than the handle                 | L            |
| Handle protrudes through the front of the bounding box | F            |
| Handle protrudes through the side of the bounding box  | S            |
| Bounding box is misaligned with the door plane         | M            |

Table 2.2: Common error types, and their abbreviations.

ical knobs (Figures 2.10(p), and 2.10(u)) with no visually distinct features on the front surface. When features or texture are not visible, the most prominent remaining feature that can be tracked between images is the edge of the handle with respect to the door. When the handles do not have any distinct features on the front surface, the optical flow algorithm is unable to accurately estimate the motion field on the surface of the handle. As a result, the bounding boxes for these cases do not fully capture the front surface of the handles. When the handle has features such as locks or key holes, the additional detail allows for a more accurate estimate the depth of the handle, resulting in the bounding box that much closer approximates the volume occupied by the handle (e.g. Figures 2.10(c), 2.10(m), and 2.10(q)).

While the generated bounding boxes in the above examples are sufficient to roughly localize the handle, the manipulator would need to allow for a margin of error when executing a grasp. In most cases the handle protrudes to a small extent, which is unlikely to cause difficulties. In certain cases the handle protrudes to a significant extent, and a collision of the end effector with the handle is possible. In such cases, the manipulator would require a means of detecting contact with the handle.

In Figures 2.10(a) and 2.10(e), the bounding box appears to extrude further forward than the handle. This occurs when a set of outlier points are not fully removed by the median filter. If the pixels corresponding to these points are adjacent to the handle in the image, these points end up being grouped together with the handle points in one component. The resulting bounding box appears larger than the true handle. However, this type of

| Door Type           | Handle Type | Surface  | Keyhole | Error Type |   |   |   |
|---------------------|-------------|----------|---------|------------|---|---|---|
|                     |             |          |         | L          | F | S | M |
| painted, monochrome | spherical   | gold     | yes     | x          |   |   | x |
| painted, monochrome | spherical   | gold     | no      |            | x |   | x |
| painted, monochrome | lever       | metallic | yes     |            |   |   |   |
| painted, monochrome | lever       | metallic | no      |            |   |   |   |
| painted, monochrome | cylindrical | metallic | no      | x          |   |   | x |
| painted, monochrome | spherical   | metallic | yes     |            |   | x | x |
| painted, monochrome | spherical   | metallic | no      |            |   |   | x |
| painted, wood       | spherical   | gold     | yes     |            |   |   |   |
| painted, wood       | spherical   | gold     | no      |            | x |   |   |
| painted, wood       | lever       | metallic | yes     |            |   |   |   |
| painted, wood       | lever       | metallic | no      |            | x |   |   |
| painted, wood       | cylindrical | metallic | no      |            |   |   |   |
| painted, wood       | spherical   | metallic | yes     |            | x |   | x |
| painted, wood       | spherical   | metallic | no      |            | x |   |   |
| wood                | spherical   | gold     | yes     |            | x |   |   |
| wood                | spherical   | gold     | no      |            | x |   |   |
| wood                | lever       | metallic | yes     |            |   |   |   |
| wood                | lever       | metallic | no      |            | x |   |   |
| wood                | cylindrical | metallic | no      |            |   |   |   |
| wood                | spherical   | metallic | yes     |            | x |   |   |
| wood                | spherical   | metallic | no      |            | x |   |   |

Table 2.3: Summary of the performance of the door handle localization algorithm. Error types are defined in Table 2.2.

error was observed in only two of the tested cases.

In several cases, a small misalignment between the orientation of the bounding box and the normal to the door was noted. This occurs when the normal to the door plane

cannot be accurately estimated. Optical flow algorithms attempt to estimate how far each image pixel has moved between two images taken from slightly different viewpoints, under the assumption that the intensity of the pixel should remain constant. Since the constant intensity constraint alone is insufficient, a smoothness constraint is introduced. In regions where no image features are visible the smoothness constraint is dominant, and the surface of the door can appear to curve when it is not parallel to the image plane. This was noticed to occur in scenes with the plane painted door, where the detail used for the optical flow estimation came largely from the line where the door was closed, from detail where the handle was mounted to the door, and from shadows cast by the handle (Figure 2.10 (a), (b), and (f)).

Figure 2.11 shows two cases where the handle is partially outside of the FOV of the camera. In each case a structural element is present on the right side of the image. The algorithm was able to detect the handle, and construct a bounding box in both of these cases. The accuracy of the localization is similar to what was demonstrated in Figure 2.10.

## 2.5 Conclusions and Discussion

This chapter presents a method for localizing a door handle of arbitrary geometry and identifying its pose. The method is based on segmentation algorithm that separates the background structure in the scene, allowing for the handle to be localized. The pose and geometry of the handle are estimated by fitting a bounding box to its principal components when projected onto the dominant plane in the scene. The bounding box can then be used to guide robotic grasping and manipulation of the handle. The effectiveness of the proposed method was demonstrated experimentally. The method performed well under conditions similar to those which would be encountered in a normal indoor environment. The method was able to successfully perform segmentation and localization of multiple types of door knobs and handles mounted on doors with a varying amount of texture.

The performance of the algorithm is limited by the performance of the vision system. If the vision system is able to accurately reconstruct the 3D structure of the observed scene, the proposed algorithm should be able to segment and localize any door handle, of arbitrary shape. However, 3D reconstruction based on optical flow becomes unreliable when dealing with featureless or highly reflective surface regions, as demonstrated by some of the experimental results. While in these cases the method was still able to segment the handle from the background environment, error in 3D reconstruction reduced the accuracy with which the bounding box was able to approximate the volume occupied by the door handle. Passive vision systems, such as those using stereo or optical flow, can not be used to accurately estimate the geometry of featureless surfaces. Recovery of surface shape when no features are visible requires the use of active vision [Ihrke et al., 2010; Zhang et al., 2003; Kosov et al., 2011; Zickler et al., 2003; Wang et al., 2007; Scharstein and Szeliski, 2003]. Additionally, incorporating features that are not effected by illumination effects could improve the performance of the algorithm.

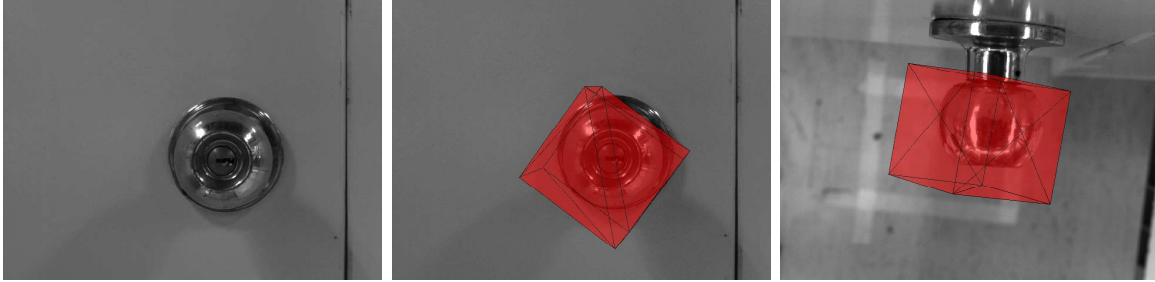
The location where the vision system can be mounted also has to be considered. It is not always possible to mount a stereo camera or a 3D camera in the optimal position for door opening. Often, stereo or 3D cameras have to be mounted at the base of a mobile robot. The small size of the handle, combined with the high angle at which a base-mounted vision system would have to observe it, make 3D reconstruction challenging. Alternatively, a camera mounted on the end-effector can be moved to observe the handle from a more favourable position.

The algorithm was shown to be able to function successfully when a portion of the handle was outside of the field of view of the camera. When a significant part of the handle is not visible, the bounding box may not be effective for guiding the grasping of the handle. However the ability to move the camera can be used to repeat the operation from a better view point, or to avoid this situation entirely.

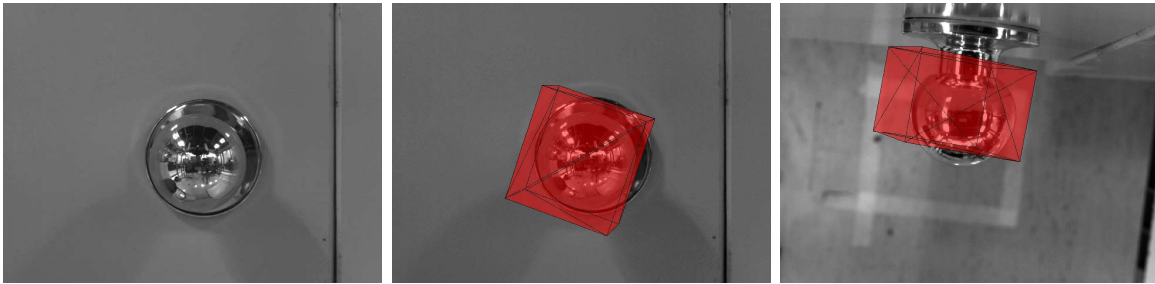
Even with a low resolution of  $656 \times 491$  there were sufficient details captured for optical flow to be used to obtain a rough estimate of the door's and handle's geometry. While most results presented with defects in the shape or orientation of the bounding box, in a



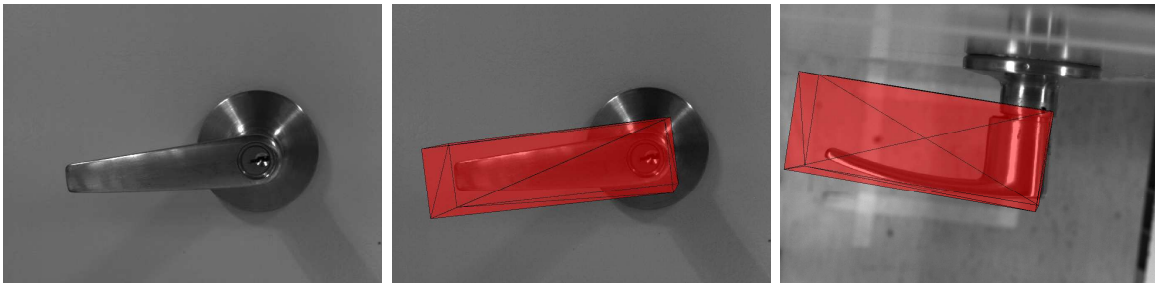
majority of cases these errors are considered minor, and are not likely to cause difficulties to later grasping operations. In the worst cases, where the handle was reflective, or had no visible features, the handle's surface appeared to be pushed closer toward the door. When features such as locks, or key holes are visible, the algorithm performs very well, with the bounding volume closely approximating the boundaries of the handle.



(a) Gold spherical handle with a lock on a painted monochrome door.



(b) Gold spherical handle on a painted monochrome door.

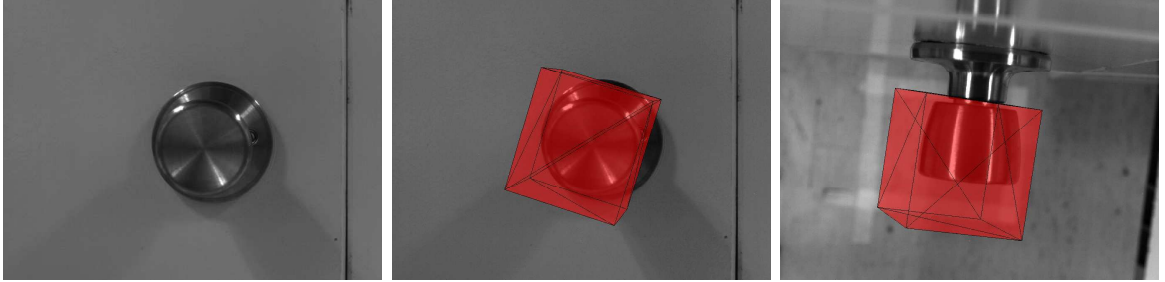


(c) Metallic handle with a lock on a painted monochrome door.

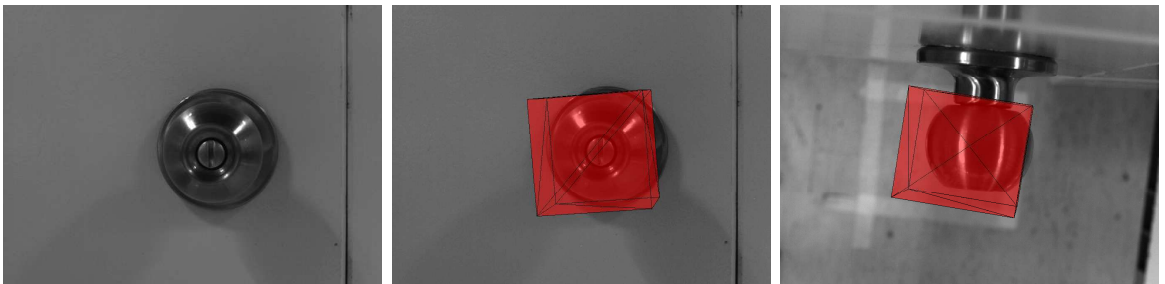


(d) Metallic handle on a painted monochrome door.

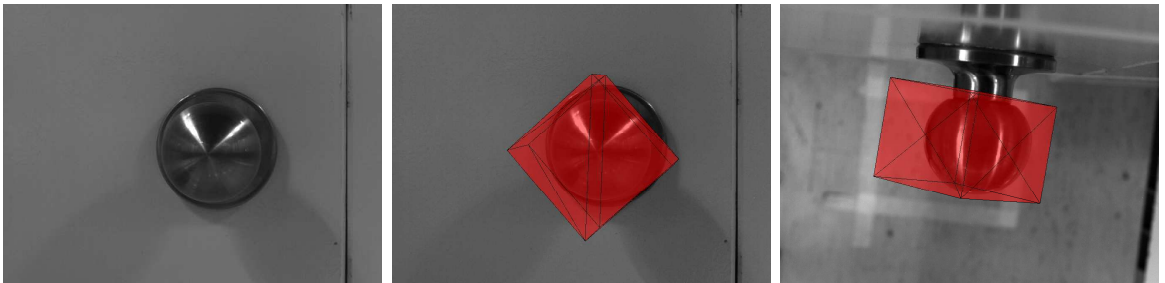
Figure 2.10: Visualization of the bounding box for different types of handles.



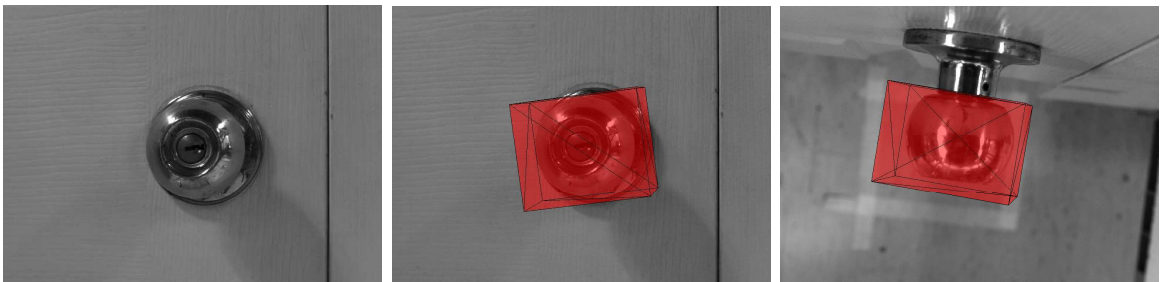
(e) Cylindrical metallic handle on a painted monochrome door.



(f) Spherical metallic handle with a lock on a painted monochrome door.

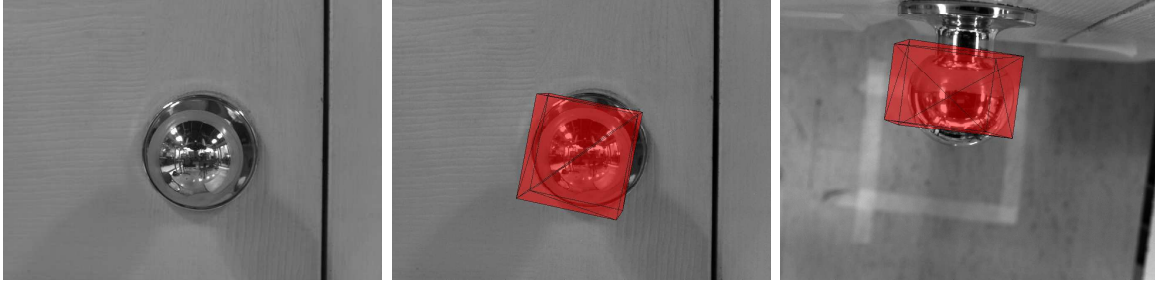


(g) Spherical metallic handle on a painted monochrome door.



(h) Gold spherical handle with a lock on a painted wooden door.

Figure 2.10: Visualization of the bounding box for different types of handles.



(i) Gold spherical handle on a painted wooden door.



(j) Metallic handle with a lock on a painted wooden door.



(k) Metallic handle on a painted painted wooden door.

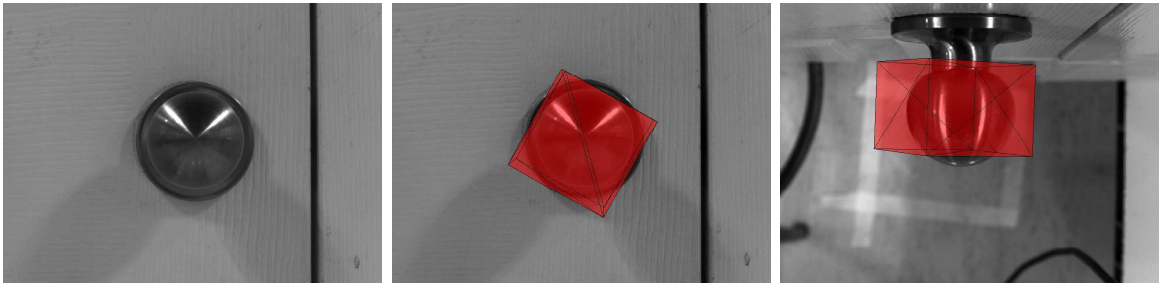


(l) Cylindrical metallic handle on a painted wooden door.

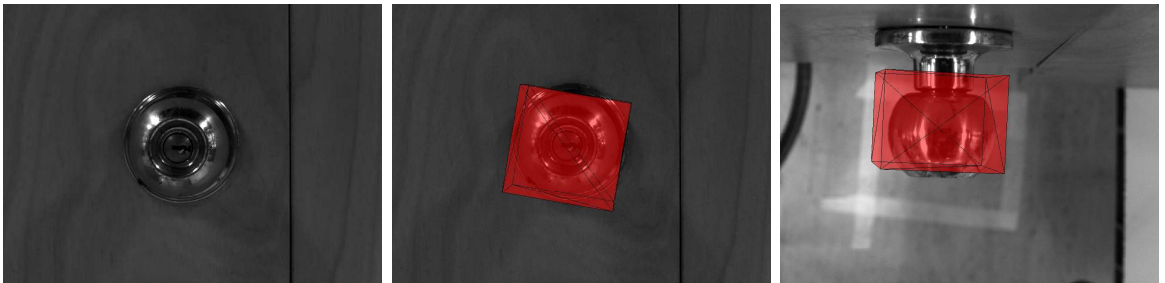
Figure 2.10: Visualization of the bounding box for different types of handles.



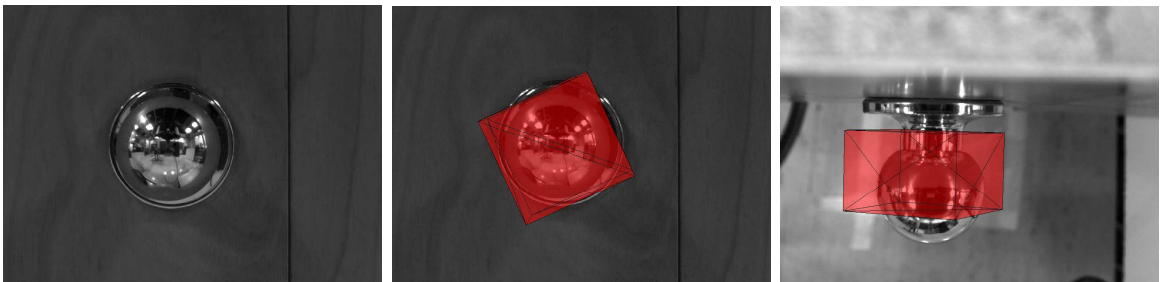
(m) Spherical metallic handle with a lock on a painted wooden door.



(n) Spherical metallic handle on a painted wooden door.



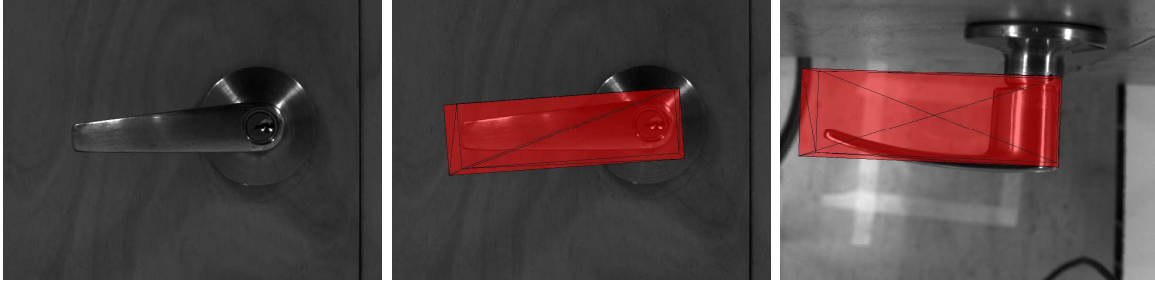
(o) Gold spherical handle with a lock on a door with strong wooden texture.



(p) Gold spherical handle on a door with strong wooden texture.

Figure 2.10: Visualization of the bounding box for different types of handles.

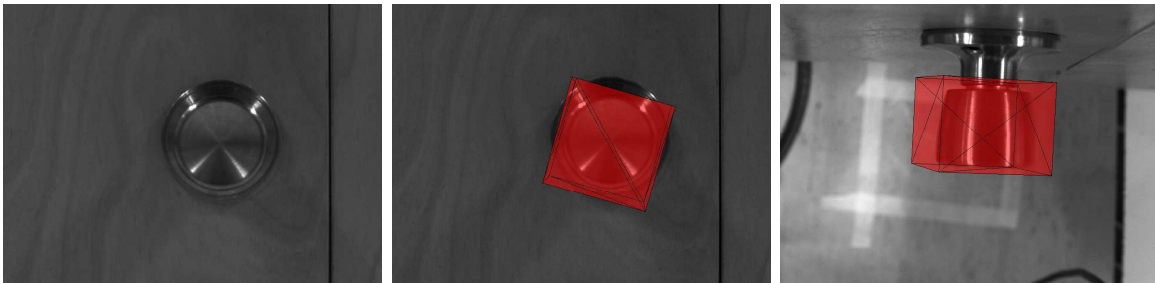




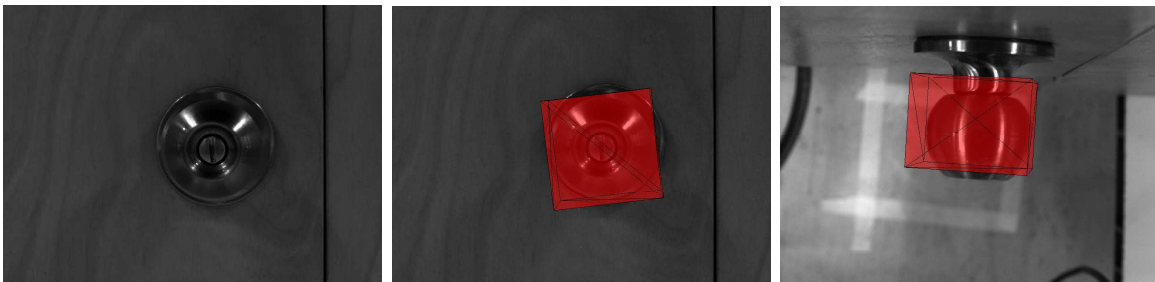
(q) Metallic handle with a lock on a door with strong wooden texture.



(r) Metallic handle on a door with strong wooden texture.

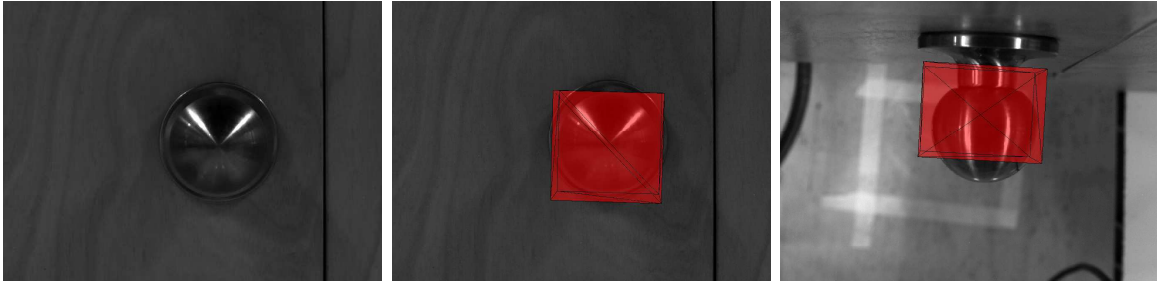


(s) Cylindrical metallic handle on a door with strong wooden texture.



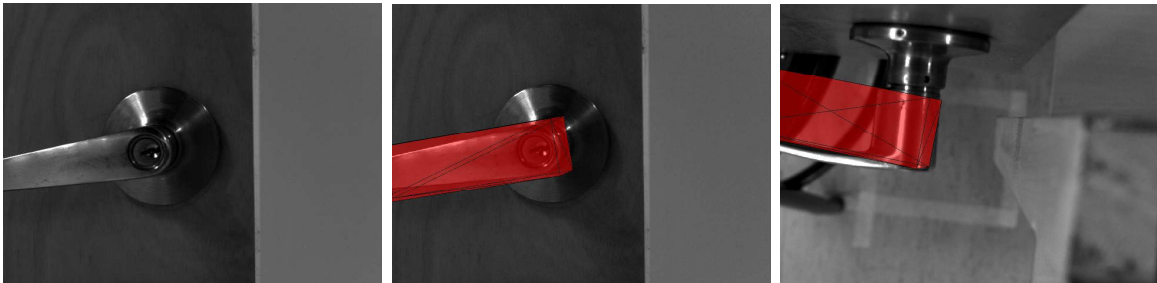
(t) Spherical metallic handle with a lock on a door with strong wooden texture.

Figure 2.10: Visualization of the bounding box for different types of handles.

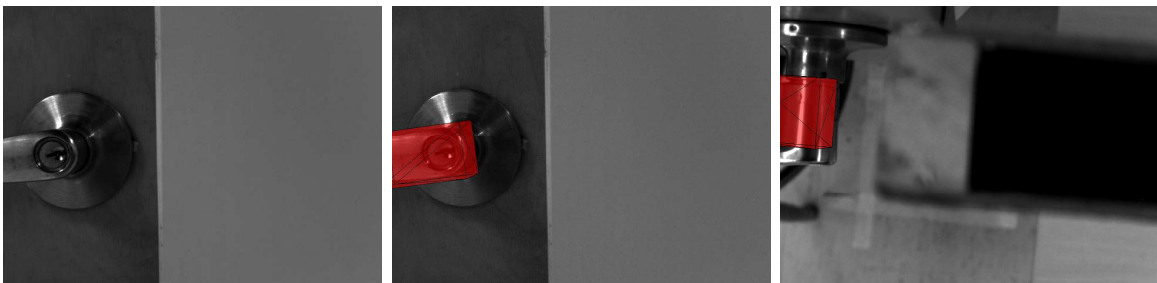


(u) Spherical metallic handle on a door with strong wooden texture.

Figure 2.10: Visualization of the bounding box for different types of handles.



(a) Flat metallic knob on a door with strong wooden texture. A small portion of the handle is outside of the camera's FOV.



(b) Metallic handle on a door with strong wooden texture. Most of the handle is outside of the camera's FOV.

Figure 2.11: Handle localization with part of the handle outside of the camera's FOV. An extruding portion of a wall can be seen on the right-hand side of the images.





# Object Segmentation

---

The door handle localization method developed in the previous chapter has the potential to improve the mobility of autonomous robots in human environments. However, the segmentation method that allows for the door handle model to be acquired incorporates strong assumptions about the specific geometry of the door and handle scene. This makes it not applicable to acquiring models of objects other than door handles.

Without the ability to manipulate objects, the usefulness of a robot is severely limited. In this chapter, a segmentation method is developed to assist in guiding autonomous robotic grasping and manipulation tasks.

This chapter is organized as follows. Section 3.1 presents an overview of the segmentation algorithm. Section 3.2 describes the use of probabilistic graphical models in image segmentation. It provides the necessary theoretical background, and describes the formulation of the energy function used to solve the segmentation problem. It also describes the graph cuts optimization method used. Section 3.3 presents the method for detecting the initial object seeds which are used to construct object and background models, and to estimate the covariance matrix of the Mahalanobis-like distance used for measuring similarity between pixels. Section 3.4 describes how the *unary* term of the segmentation energy function is defined, and explains how the features used in the segmentation algorithm are modelled. Section 3.5 describes *pairwise* term used as a smoothness constraint, and defines the Mahalanobis-like distance used to evaluate similarity between image feature vectors. Section 3.6 describes the process used to learn the parameters of the segmentation energy function from training data. Section 3.7 discusses experimental validation, and presents experimental results. Concluding remarks are given in Section 3.8.

### 3.1 The Proposed Segmentation Algorithm

Segmentation in uncontrolled environments is challenging. Algorithms that rely on image features alone can perform poorly under certain lighting conditions, or if the object and the background have similar appearance. Object segmentation methods that rely exclusively on three dimensional (3D) geometric data are derived under strong assumptions about the geometry of the scene. While a significant improvement in segmentation robustness and accuracy can be achieved by utilizing both appearance and geometric cues, current methods that combine appearance and geometry still have limitation: they either do not utilize all available information, or use this information at different stages of the algorithm. The result is that these methods must still rely on certain geometric assumptions about the scene, the most common of which is that the object rests on a planar surface.

To address the challenges in performing segmentation in complex environment, the algorithm proposed in this chapter utilizes both appearance and geometric features jointly in a probabilistic framework. Specifically, combining appearance features such as intensity or colour and texture, with geometric features such as depth and curvature, enables the proposed algorithm to function effectively when the object and the background have similar appearance. Texture is utilized to allow for a richer description of the appearance properties of both the object and the background, particularly in situations where the object and the background have the same mean intensity or colour. Curvature is used to model the objects' geometric properties.

In contrast to methods that use depth to model object regions directly, which can cause the segmentation to spill past the object of interest, curvature allows for a more accurate description of the objects geometry. It also allows the proposed algorithm to detect regions where objects contact supporting surfaces or other objects, removing the need to assume that objects rest on the dominant planar surface. In the proposed algorithm, appearance and geometric cues are used jointly, avoiding segmentation errors inherent

### 3.1. The Proposed Segmentation Algorithm

---

in algorithms which use such cues independently at different stages of the algorithm. In addition, the method presented here uses a Mahalanobis-like distance to weight relative significance of the available segmentation cues. In different scenes, some features are more significant for detecting the boundary of the object. For example, intensity should be considered more significant when the object and the background are uniformly shaded, and less significant when the object is multi-shaded. The use of the Mahalanobis-like distance allows the algorithm to decide which feature differences are assigned greater significance, while also allowing features of different scales to be effectively combined. The term Mahalanobis-like is used because the feature differences are evaluated using feature-specific functions, and not as differences of feature vectors. This allows the algorithm to better adapt to the available data. The proposed method can function in visually complex and cluttered environments, and only requires the use of image and 3D data that can be easily available to many mobile robots.

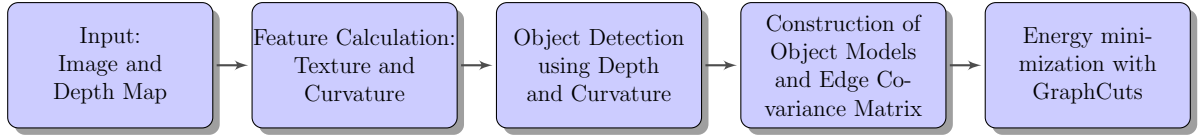


Figure 3.1: Major processing steps of the proposed object segmentation algorithm.

The proposed algorithm operates in several steps, which are shown in Figure 3.1. The flow of information between components of the proposed method is shown in Figure 3.2. Prior to segmentation, a vision system is used to acquire a colour or monochrome image of the scene and a corresponding depth map. The scene is modelled as a graph, with pixels in the image acting as nodes, and edges connecting adjacent pixels in an 8-connected neighbourhood system. Initial object seed regions are detected using closed contours of depth edges and high curvature edges. Once a target object has been selected, foreground and background region models are built from the depth, curvature, image, and texture data in the detected object and background seed regions. A covariance matrix used to weigh the similarity between adjacent pixels is also constructed based on image, texture, depth, and curvature information within object and background seed regions. Segmentation is then performed using graph cuts, with graph edge weights assigned using the

available appearance and geometry data, the calculated region models, and covariance matrix.

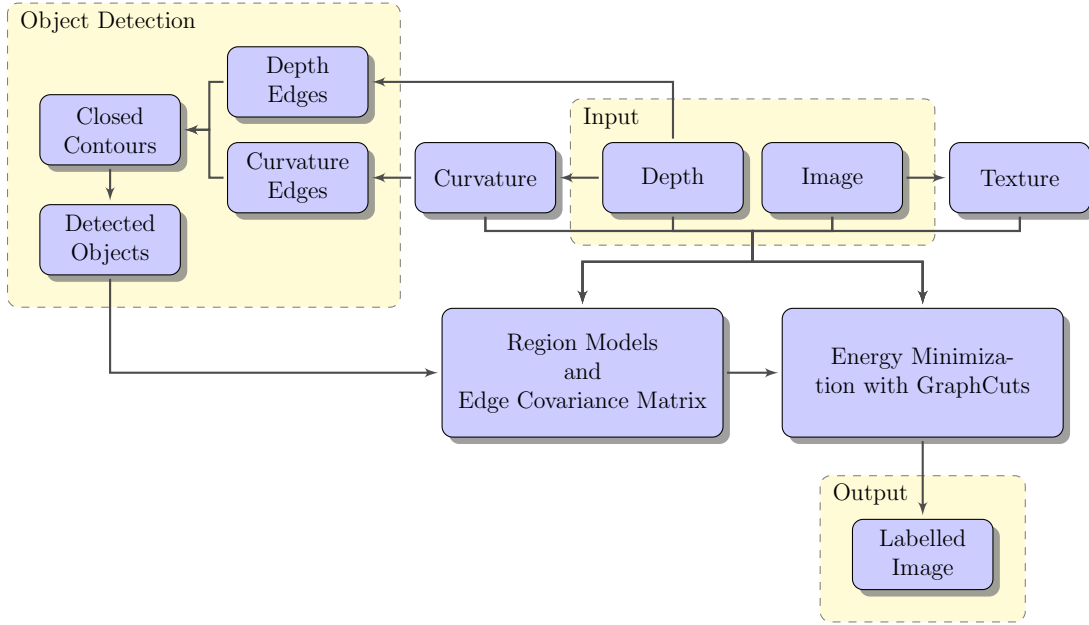


Figure 3.2: Flow of information in the proposed segmentation algorithm, from input to output.

## 3.2 Markov and Conditional Random Fields in Image Segmentation

This section introduces Markov and Conditional Random Field (MRF and CRF, respectively) models for image segmentation. An energy function based on a conditional random field model of the segmentation problem is defined, and the graph cuts method is presented as a means of obtaining the optimal solution to the segmentation problem.

### 3.2.1 Segmentation as a Labelling Problem

Image segmentation can naturally be modelled as a labelling problem. Segmentation is the partitioning of the image into a number of disjoint segments. This is equivalent to assigning a label to each pixel, such that each label corresponding to a specific image segment. Pixels with the same label therefore represent a single segmented region of the image, and the total number of labels is equal to the number of regions in the image.

More formally, let the set  $\mathcal{S} = \{1, \dots, i, \dots, |\mathcal{S}|\}$  index the sites in the image, where  $|\mathcal{S}|$  denotes the total number of elements in  $\mathcal{S}$ . The term site is used as opposed to pixel because it is possible to perform segmentation utilizing different structures based not only on image pixels, but also on superpixels, which are collections of small segments resulting from a pre-processing step; or vertices in a 3D mesh. Each site  $i \in \mathcal{S}$  must be assigned a label out of the set of all possible labels  $\mathcal{L}$ . As the focus of this chapter is the segmentation of a single object from the background the label set is binary  $\mathcal{L} = \{0, 1\}$ , where 0 corresponds to the object, and 1 to the background.

### 3.2.2 Random Field Models

Let  $F_i$  be a random variable associated with a site  $i \in \mathcal{S}$  that may take on a value from the previously defined label set  $\mathcal{L}$ . The family of such random variables  $\mathbf{F} = \{F_1, \dots, F_i, \dots, F_{|\mathcal{S}|}\}$  defined on the set  $\mathcal{S}$  is called a random field. The event that the random variable  $F_i$  takes on a specific value  $f_i$  is denoted as  $F_i = f_i$ . The joint event that all random variables in a field assuming specific values ( $F_1 = f_1, \dots, F_i = f_i, \dots, F_{|\mathcal{S}|} = f_{|\mathcal{S}|}$ ) is denoted as  $\mathbf{F} = \mathbf{f}$ , where  $\mathbf{f} = \{f_1, \dots, f_i, \dots, f_{|\mathcal{S}|}\}$  is called the configuration of  $\mathbf{F}$ . The set of all possible configurations that a random field can take is denoted  $\mathbb{F}$ . The probability that a random variable  $F_i$  takes on a specific value  $f_i$  is denoted  $P(F_i = f_i)$ , and the probability of the joint event is denoted  $P(\mathbf{F} = \mathbf{f})$ . It is common to abbreviate the probability of a realization of the random variable or field  $P(F_i = f_i)$  as  $P(f_i)$  and

$P(\mathbf{F} = \mathbf{f})$  as  $P(\mathbf{f})$  respectively. For convenience, this notation will be used from now on.

The goal of segmentation is to separate an image into meaningful or useful regions. Therefore a labelling is sought which is optimal with respect to some criterion defined on the set of labels and the observed image data. Given a set of data  $\mathbf{d} = \{\mathbf{d}_i, \dots, \mathbf{d}_i, \dots, \mathbf{d}_{|\mathcal{S}|}\}$  where  $\mathbf{d}_i$  corresponds to an observation at the site  $i \in \mathcal{S}$ . A labelling is sought that maximizes the posterior distribution of labels conditioned on the available data  $P(\mathbf{f}|\mathbf{d})$ . Using Bays' rule, the posterior distribution can be decomposed into a product of the observation model  $P(\mathbf{d}|\mathbf{f})$  and the prior  $P(\mathbf{f})$ :

$$P(\mathbf{f}|\mathbf{d}) = \frac{P(\mathbf{d}|\mathbf{f})P(\mathbf{f})}{P(\mathbf{d})} \quad (3.1)$$

The data term  $\mathbf{d}$  is fixed and the denominator of Equation (3.1) can be ignored. This reduces the problem to finding the joint probability of the labelling and the data:

$$P(\mathbf{f}|\mathbf{d}) \propto P(\mathbf{f}, \mathbf{d}) = P(\mathbf{d}|\mathbf{f})P(\mathbf{f}) \quad (3.2)$$

## Markov Random Fields

The prior distribution  $P(\mathbf{f})$  in Equations (3.1), and (3.2) encodes a set of (smoothness) assumptions on the distribution of labels. It can be modelled as an MRF.

A Markov random field is a probabilistic model on the graph  $\mathcal{G}(\mathcal{S}, \mathcal{N})$  where the set of sites  $\mathcal{S}$  is defined as above, and the relationship between various sites is described by the neighbourhood system  $\mathcal{N}$ . The neighbourhood system  $\mathcal{N} = \{\mathcal{N}_i | \forall i \in \mathcal{S}\}$ , where  $\mathcal{N}_i = \{j | j \neq i, j \sim i\}$  contains the set of all neighbours  $j$  of the site  $i$ . A site cannot be a neighbour to itself, and the notation  $i \sim j$  indicates that sites  $i$  and  $j$  are neighbours and share an edge in the graph. The neighbourhood relationship is mutual, if  $j$  is a

neighbour of  $i$ , then symmetrically,  $i$  is also a neighbour of  $j$ . In segmentation problems the neighbourhood is commonly defined as the 4- or 8-neighbourhood system on image pixels for a regular set of sites (Figure 3.3). In a 4-neighbourhood (Figure 3.3(a)), each pixel is connected to its four immediate neighbours vertically and horizontally. In the 8-neighbourhood (Figure 3.3(b)), the diagonally located pixels are also considered neighbours. When irregular sites are used, such as vertices formed by a mesh, or superpixels from a pre-processing step, the size of the neighbourhood may not be constant. Neighbours in these situations are defined based on a shared boundary in case of superpixels, or based on their distance from the site under consideration, their connectivity in the mesh, or other criteria. Figure 3.4 show examples of regular and irregular sites defined for images and point clouds, along with sample neighbourhood systems defines on those sites.

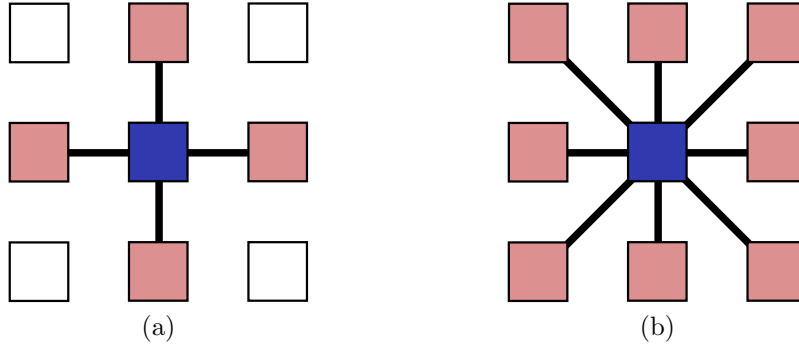


Figure 3.3: Examples of 4-connected (a), and 8-connected (b) neighbourhood systems.

For the random field  $\mathbf{F}$  to be an MRF it must satisfy two additional condition. The first condition is called Markovianity, and it states that the probability of a any random variable taking on a specific value depends only on the values in the neighbourhood of that random variable:

$$P(f_i | \{f_j | j \neq i\}) = P(f_i | \{f_j | (i, j) \in \mathcal{N}_i\}) \quad (3.3)$$

The set of neighbours of a site  $i$  is also referred to as its Markov blanket. The second

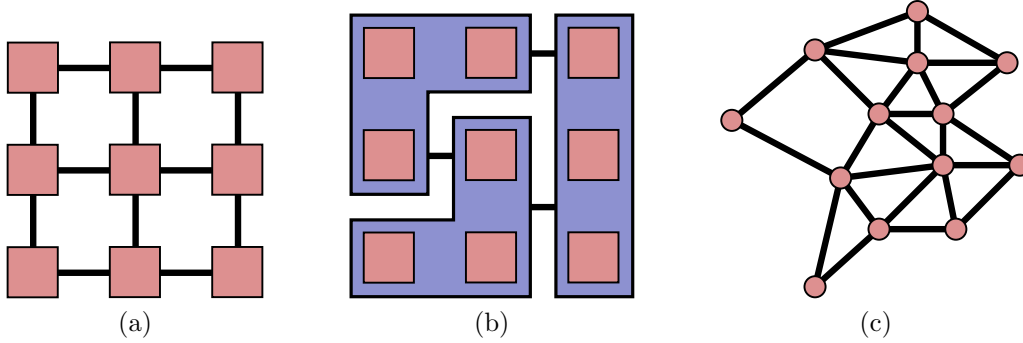


Figure 3.4: Examples of regular and irregular neighbourhood systems. (a) Regular neighbourhood system. (b) Irregular neighbourhood system resulting from pre-segmentation into superpixels. (c) Irregular neighbourhood system resulting from triangulation.

condition is positivity. It states that probability density of the field must be strictly positive:

$$P(\mathbf{f}) > 0, \quad \forall \mathbf{f} \in \mathbb{F} \quad (3.4)$$

For an MRF, the Hammersley–Clifford theorem, states that the prior distribution  $P(\mathbf{f})$  can be written as a Gibbs distribution:

$$P(\mathbf{f}) = \frac{1}{Z} \exp \left( -\frac{1}{T} E(\mathbf{f}) \right) \quad (3.5)$$

where  $Z$  is the normalizing constant of the distribution called the partition function:

$$Z = \sum_{\mathbf{f} \in \mathbb{F}} \exp \left( -\frac{1}{T} E(\mathbf{f}) \right) \quad (3.6)$$

The temperature  $T$  controls the spread of the distribution. For high values of  $T$ , the distribution is spread out, and all configurations of  $\mathbf{f}$  are equally likely. For low values



of  $T$  the distribution is narrow, with a defined peak. The energy function  $E(\mathbf{f})$  is [Li, 2009]:

$$E(\mathbf{f}) = \sum_{c \in \mathcal{C}} V_c(\mathbf{f}) \quad (3.7)$$

The term  $V_c$  represents the clique potential energy corresponding to a specific configuration of labels in a clique, and a clique is a fully connected subgraph of  $\mathcal{G}(\mathcal{S}, \mathcal{N})$ . The size of the clique is equal to the number of sites it contains. Trivially, a single site is considered a clique of size one  $c \in \mathcal{C}_1 = \{i\}$ . Pairs of connected sites are cliques of size two  $c \in \mathcal{C}_2 = \{i, j\}$ . The maximum size of a clique depends on the neighbourhood system of the graph. The energy function measures the likelihood of a particular combination or realization of labels in the field  $\mathbf{F}$ . The more likely the realization of the field, the lower the corresponding energy will be. The prior model is commonly limited to pairwise terms only [Blake et al., 2011; Li, 2009]:

$$E(\mathbf{f}) = \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j) \quad (3.8)$$

Combining Equation (3.2) and (3.8), the joint distribution can be written as:

$$P(\mathbf{f}, \mathbf{d}) = P(\mathbf{d}|\mathbf{f}) \frac{1}{Z} \exp \left( -\frac{1}{T} \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j) \right) \quad (3.9)$$

The observation model  $P(\mathbf{d}|\mathbf{f})$  is determined based on the the knowledge of the data formation process, or based on a model of the data distributions. A detailed description of the models used in this chapter will be presented in Section 3.4. A common assumption is that observations at different sites are conditionally independent when the label of the site is available. This allows the likelihood to be written as a product of likelihoods for individual sites:

$$P(\mathbf{d}|\mathbf{f}) = \prod_{i \in \mathcal{S}} P(d_i|f_i) \quad (3.10)$$

resulting in a the joint probability density function of the form:

$$P(\mathbf{d}, \mathbf{f}) = \frac{1}{Z(\mathbf{d})} \exp \left( -\ln \left( \sum_{i \in \mathcal{S}} P(d_i|f_i) \right) - \frac{1}{T} \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j) \right) \quad (3.11)$$

The joint distribution (3.11) is also an MRF of the form:

$$P(\mathbf{f}, \mathbf{d}) = \frac{1}{Z(\mathbf{d})} \exp(E(\mathbf{f}, \mathbf{d})) \quad (3.12)$$

where

$$E(\mathbf{f}, \mathbf{d}) = \sum_{i \in \mathcal{S}} V_1(d_i, f_i) + \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j) \quad (3.13)$$

$V_1$  is called the *unary* or the *data* term which evaluates how well the data agrees with the assigned label, and  $V_2$  is the *pairwise* smoothness or *prior* term encoding prior assumptions about the distribution of labels. The *pairwise* term is commonly used to enforce a smoothness constraint on the solution.

## Conditional Random Fields

In segmentation problems, the set of labels is unordered. There is no meaningful way of expressing the smoothness of a solution with the exception of a binary condition indicating whether the labels are the same or not. This has lead to the wide adoption of the Ising prior model [Blake et al., 2011; Li, 2009]:

$$V_2(f_i, f_j) = \begin{cases} \beta_1, & f_i = f_j \\ \beta_2, & \text{otherwise} \end{cases} \quad (3.14)$$

where,  $(i, j) \in \mathcal{N}$  and  $\beta_2 > \beta_1$ . This prior model penalizes any discontinuity between labels at neighbouring sites, and encodes the basic prior assumption that regions should be contiguous. This is equivalent to penalizing the total boundary length of an object.

When performing segmentation, it would be desirable to weaken the penalty imposed by Equation (3.14) when the image data indicates the existence of an edge. For example, in grayscale image segmentation, it may be desirable to fully penalize the separation of two sites that share the same intensity. However a reduced penalty may be preferable when separating two sites that have an intensity of 0, and 255 respectively. Such a smoothness constraint can be written as a modulated Ising prior:

$$V'_2 = g(\mathbf{d})V_2 \quad (3.15)$$

where the function  $0 \leq g(\mathbf{d}) \leq 1$  weakens the usual pairwise smoothness prior of Equation (3.14) based on the image data. This is illustrated in Figure 3.5. An image of three pop cans on a light table is presented, along with the corresponding Ising prior, and the contrast sensitive pseudo “prior”. For each site, the sum of all pairwise potentials in the pixel’s neighbourhood is shown:

$$intensity(i) = \sum_{(i,j) \in \mathcal{N}_i} V_2(i, j) \quad (3.16)$$

Brighter regions corresponding to higher affinity between the pixel and its neighbours.

The main issue that arises in weakening the prior term depending on the image data, is that it can no longer be considered a prior on the distribution. This results in potentials

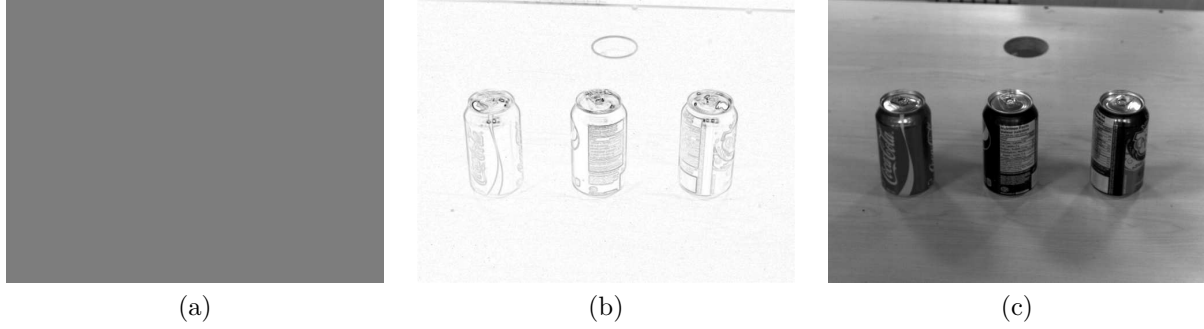


Figure 3.5: Comparison of the Ising (a) and the contrast sensitive (b) prior terms for a sample scene (c). Darker values indicate a lower cost for separating sites associated with the image pixel. Note that a uniform penalty is applied to any discontinuity by the Ising prior (a), demonstrated by the uniform grey image. The contrast sensitive model (b) penalizes boundary placement at image edges less compared to placing boundaries in regions of constant intensity.

that are not fully factored into observation and prior. A model that allows for this is called a conditional random field (CRF) [Lafferty et al., 2001; Li, 2009; Blake et al., 2011]. The CRF models the posterior conditional distribution  $P(\mathbf{f}|\mathbf{d})$  directly without requiring the factorization of the prior and the observation terms:

$$E(\mathbf{f}, \mathbf{d}) = \sum_{i \in \mathcal{S}} V_1(\mathbf{d}, \mathbf{f}) + \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j, \mathbf{d}) \quad (3.17)$$

While the CRF model allows for arbitrary dependence of each of the unary and pairwise terms on all of the data, this is not required. It is sufficient to allow the pairwise term to utilize the data only at sites directly involved in the evaluation of the clique potential:

$$E(\mathbf{f}, \mathbf{d}) = \sum_{i \in \mathcal{S}} V_1(f_i, \mathbf{d}_i) + \frac{1}{T} \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j, \mathbf{d}_i, \mathbf{d}_j) \quad (3.18)$$

Similarly to the case of the MRF, the *unary* potential function  $V_1$  indicates how well the data  $\mathbf{d}_i$  at site  $i$  matches the assigned label  $f_i$ . It's formulation will be described in

Section 3.4. The *pairwise* potential  $V_2$  acts as a smoothness constraint (data dependent “prior”) penalizing discontinuities between labels. To preserve edges the value of  $V_2$  must be high when the features at  $i$  and  $j$  are similar and low when they are different. The process of estimating this term will be presented in Section 3.5. The term  $1/T$  controls the relative significance of the smoothness term. It’s value is determined offline during training, and remains constant when segmentation is performed.

#### 3.2.3 Inference Using Graph Cuts

Given a random field model, the problem of inference is to estimate the most likely realization of the random field.

$$\mathbf{f}^* = \arg \max_{\mathbf{f} \in \mathbb{F}} P(\mathbf{f}|\mathbf{d}) \quad (3.19)$$

This is referred to as maximum a posteriori (MAP) estimation. An important fact about obtaining the MAP estimation of a labelling is that the partition function is evaluated over all possible labelling  $\mathbb{F}$ . As such, it is not a function of any particular labelling, and can be ignored. The MAP estimate can therefore be obtained by finding the labelling with the minimum energy:

$$\mathbf{f}^* = \arg \min_{\mathbf{f} \in \mathbb{F}} E(\mathbf{f}, \mathbf{d}) = \arg \min_{\mathbf{f} \in \mathbb{F}} \sum_{i \in \mathcal{S}} V_1(f_i, \mathbf{d}_i) + \frac{1}{T} \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j, \mathbf{d}_i, \mathbf{d}_j) \quad (3.20)$$

A number of algorithms exist to solve MAP problems in graphical models. For example, belief propagation can be used to obtain exact solutions for acyclic graphs, as well as high quality approximate solutions for graphs with cycles like the ones used to model images. [Blake et al., 2011; Mudigonda, 2008]. Alternative methods to approximate MAP

solutions in cyclic graphs have included simulated annealing, mean field approximation, and Gibbs sampling. While classical approaches such as simulated annealing have been shown to perform poorly in practise, often converging far from the globally optimal solution, belief propagation and graph cuts have become very popular in recent literature [Mudigonda, 2008].

Of particular interest is the graph cuts approach, which can be used to obtain an exact global solution to a MAP-MRF problem provided that the energy function meets certain condition. This method of inference has become widely used in the image segmentation literature, as energy functions commonly formulated for segmentation meet the necessary conditions of being optimizable with graph cuts, resulting in solutions that are exact and numerically stable [Mudigonda, 2008; Blake et al., 2011].

The graph cuts algorithm can be used to obtain an exact minimum of a submodular quadratic pseudo-boolean energy function. A pseudo-boolean energy function maps a set of  $n$  boolean variables to a single real number. Energy functions formulated for binary image segmentation where each label is only allowed to take on one of two possible values in  $\mathcal{L} = \{0, 1\}$  can therefore be optimized using graph cuts if they can be shown to be submodular. For functions of up to two variables, it is sufficient to show:

$$V_2(1, 0) + V_2(0, 1) \geq V_2(1, 1) + V_2(0, 0) \quad (3.21)$$

Since the indicator function is zero whenever  $f_i = f_j$ , and since the edge costs are otherwise greater or equal to zero, the condition above is satisfied.

It was shown by Boykov and Jolly [2001], that submodular quadratic pseudo-boolean energy functions can be efficiently minimized by finding the maximum flow through a special graph. This is also equivalent to finding a minimum cost cut on the graph. This procedure is referred to as graph cuts.

To implement the graph cuts algorithm [Boykov and Jolly, 2001; Boykov and Funka-Lea, 2006], two special vertices called the source  $\{s\}$  and sink  $\{t\}$  are introduced. These

vertices represent the target object to be segmented and the background, respectively. The *augmented* graph  $\tilde{\mathcal{G}}(\mathcal{V}, \mathcal{E})$  which is used to segment the image is described as the vertex set  $\mathcal{V} = \mathcal{S} \cup \{s\} \cup \{t\}$  containing the set  $\mathcal{S}$  of vertices corresponding to image pixels as well as the source and sink vertices, and the set of edges  $\mathcal{E} = \mathcal{N} \cup \mathcal{S} \cup \mathcal{T}$ , where  $\mathcal{S}$  and  $\mathcal{T}$  represent the edges between the vertices in  $\mathcal{S}$ , and the source and sink vertices respectively. Each edge  $(i, j) \in \mathcal{E}$  connecting vertices  $i$  and  $j$  is assigned a weight  $w(i, j) \geq 0$ . An  $s$ - $t$  cut  $\mathcal{C}$  is a subset of edges  $\mathcal{C} \subset \mathcal{E}$ , such that source and sink nodes become completely separated on the induced subgraph  $\tilde{\mathcal{G}}(\mathcal{V}, \mathcal{E} \setminus \mathcal{C})$ . Such a cut can be considered as a solution to the binary labelling problem where each  $i$  in  $\mathcal{S}$  can be connected (labelled) to either the sink vertex or the source vertex, corresponding to assigning  $i$  to the object or the background respectively, but not both. The cost of a cut is the sum of the weights of all edges in  $\mathcal{C}$ :

$$cost = \sum_{(i,j) \in \mathcal{C}} w(i, j) \quad (3.22)$$

A minimum cost  $s$ - $t$  cut is defined as an  $s$ - $t$  cut that minimizes (3.22). By setting the weights corresponding to the source and sink edges ( $\mathcal{S}$  and  $\mathcal{T}$ ) to be equal to the first term of Equation (3.18), and by setting the weights corresponding to the neighbourhood edges ( $\mathcal{N}$ ) to be equal to the second term of Equation (3.18), the minimum cost  $s$ - $t$  cut on the augmented graph  $\tilde{\mathcal{G}}$  will produce a labelling that minimizes the energy function (3.18).

To demonstrate the graph cuts algorithm, consider a problem of minimizing an energy function defined by Equation (3.18), over four sites  $\mathcal{S} = \{a, b, c, d\}$ , with a neighbourhood system  $\mathcal{N} = \{(a, b), (b, c), (c, d)\}$ . Let the *unary* ( $V_1$ ) and *pairwise* ( $V_2$ ) potentials be defined as shown in Table 3.1 and Table 3.2 respectively. The augmented graph associated with this labelling problem is shown in Figure 3.6(a).

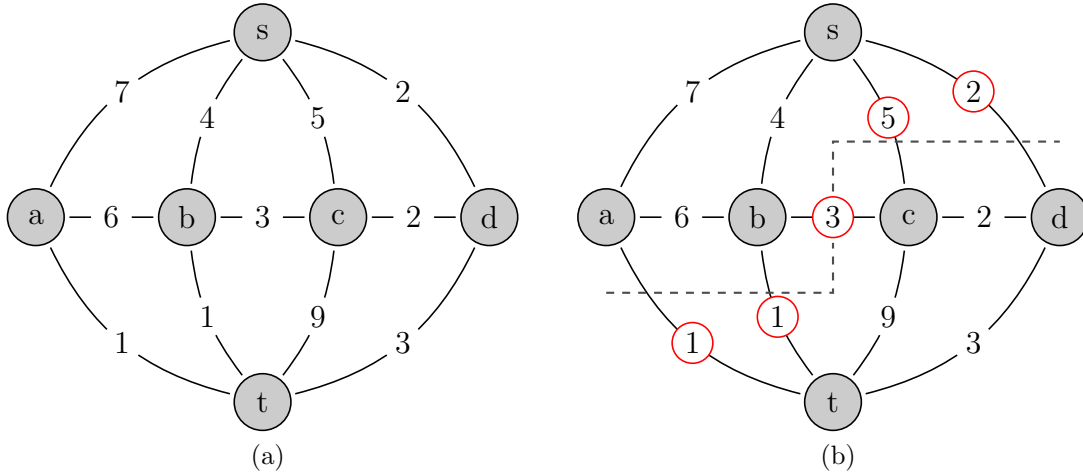
The weights connecting the sites in  $\mathcal{S}$  to the source  $s$  and sink  $t$  nodes correspond to the assignment of the node to one of the two labels. If a site is assigned a label  $f_i = 0$  then

| Site ( $i$ ) | $V_1(f_i = 0)$ | $V_1(f_i = 1)$ |
|--------------|----------------|----------------|
| a            | 7              | 1              |
| b            | 4              | 1              |
| c            | 5              | 9              |
| d            | 2              | 3              |

Table 3.1: Example unary potentials.

| Neighbours ( $i, j$ ) | $V_2(f_i \neq f_j)$ | $V_2(f_i = f_j)$ |
|-----------------------|---------------------|------------------|
| a, b                  | 6                   | 0                |
| b, c                  | 3                   | 0                |
| c, d                  | 2                   | 0                |

Table 3.2: Example pairwise potentials.


 Figure 3.6: An example graph with four sites (a). The minimum cost cut  $s$ - $t$  cut is shown with a dashed line (b), with severed edges highlighted.

it's connection with the source node  $s$  is severed, and the corresponding *unary* cost is incurred. Similarly, if a site is assigned a label  $f_i = 1$ , its connection with the sink  $t$  node is severed.



The weights shown connecting neighbouring sites correspond to the *pairwise* potentials. When two adjacent sites are assigned different labels, the edge connecting them must be severed, and the corresponding cost is incurred.

| Labelling |          |          |          | Cost of Cut | Probability   |
|-----------|----------|----------|----------|-------------|---------------|
| a         | b        | c        | d        |             |               |
| 0         | 0        | 0        | 0        | 18          | 0.0020        |
| 0         | 0        | 0        | 1        | 21          | 0.0001        |
| 0         | 0        | 1        | 0        | 27          | 0.0000        |
| 0         | 0        | 1        | 1        | 26          | 0.0000        |
| 0         | 1        | 0        | 0        | 24          | 0.0000        |
| 0         | 1        | 0        | 1        | 27          | 0.0000        |
| 0         | 1        | 1        | 0        | 27          | 0.0000        |
| 0         | 1        | 1        | 1        | 26          | 0.0000        |
| 1         | 0        | 0        | 0        | 18          | 0.0020        |
| 1         | 0        | 0        | 1        | 21          | 0.0001        |
| 1         | 0        | 1        | 0        | 27          | 0.0000        |
| 1         | 0        | 1        | 1        | 26          | 0.0000        |
| <b>1</b>  | <b>1</b> | <b>0</b> | <b>0</b> | <b>12</b>   | <b>0.8064</b> |
| 1         | 1        | 0        | 1        | 15          | 0.0401        |
| 1         | 1        | 1        | 0        | 15          | 0.0401        |
| 1         | 1        | 1        | 1        | 14          | 0.1091        |

Table 3.3: Costs of cuts and probabilities associated with all possible labellings of the example graph shown in Figure 3.6.

An  $s$ - $t$  cut must completely separate the source node  $s$  from the sink node  $t$ . It severs exactly one of the  $s$  or  $t$  edges for every site, and all edges which connect neighbouring sites with different labels  $f_i \neq f_j$ . This corresponds directly with a binary labelling, and the cost of the cut is equal to the energy of the labelling. A minimum cost  $s$ - $t$  cut finds a partitioning of the graph while minimizing the total cost of the severed edges, which is equivalent to finding a minimum energy labelling. The minimum cost  $s$ - $t$  cut for

the example problem is shown in Figure 3.6(b). The energy associated with all possible labellings, as well as the probability of each labelling is shown in Table 3.3. Note that the minimum cost  $s$ - $t$  cut is equal to the labelling associated with the lowest energy, and highest probability. This labelling is shown highlighted in bold in Table 3.3.

The minimum cut algorithm used is based on the work presented by Boykov and Kolmogorov [2004].

### 3.3 Initial Object Detection

This section describes the method for detecting object seed regions, which are used for constructing object and background models used to evaluate the *unary* potential function  $V_1$ . The seed regions are also used for estimating the parameters of the *pairwise* potential function  $V_2$ , used to evaluate the similarity between neighbouring sites.

Initial object seed regions are detected by searching for regions enclosed in contours corresponding to depth discontinuity and high curvature. Depth discontinuity allows for the identification of the top edges of an object. High values of negative curvature allow for the detection of boundaries where the object contacts the surface. Only points with negative curvature are considered as they are more likely to correspond to an object boundary.

The detection method operates as follows. First, a gradient magnitude image of the depth map is calculated. The depth gradient image, and the negative curvature map are thresholded to find high depth gradients and high negative curvature values. The thresholds are individually determined for each of the maps using the method proposed by Kittler and Illingworth [1986b]. The resulting thresholded boundary images are combined into a single boundary map.

Initial boundaries are dilated. Regions enclosed by the boundaries are filtered by size to remove small regions which are likely generated by contours of edges caused by noise in

### 3.3. Initial Object Detection

---

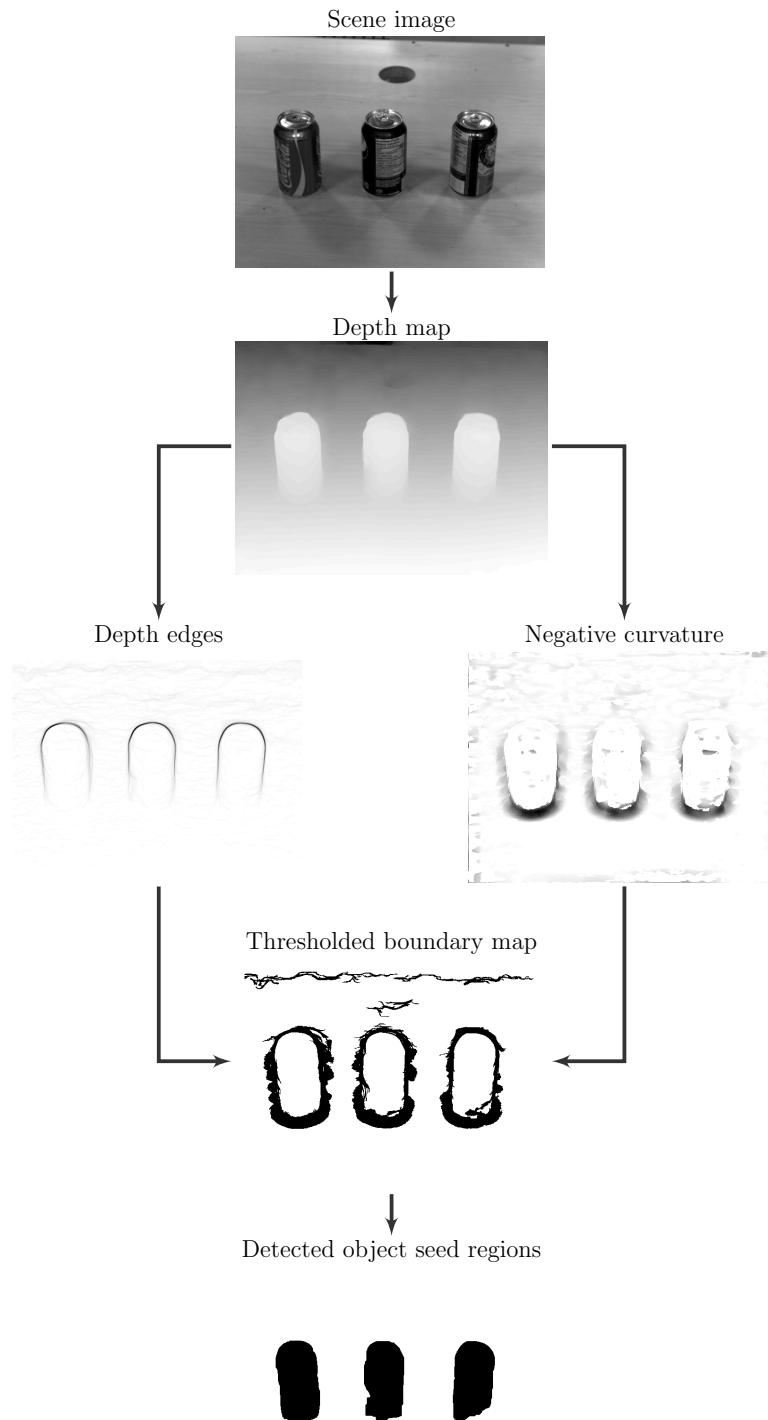


Figure 3.7: Steps of the object detection process.

depth data. The remaining regions are taken as object seeds, and the area outside of the detected and dilated contours is taken as the background. The steps involved in the object detection process are shown in Figure 3.7.

### 3.4 The *Unary* Potential

This section defines the *unary* potential, and shows how appearance and geometric features are modelled.

The *unary* potential function  $V_1$  is the negative log likelihood of the feature vector  $\mathbf{d}_i$  conditioned on the label  $f_i$  of the site  $i$ . Assuming that the appearance and geometric cues are independent,  $V_1$  can be expressed as:

$$V_1(f_i, \mathbf{d}_i) = \sum_{\mathbf{d}_i^n \in \mathbf{d}_i} -\ln P(\mathbf{d}_i^n | f_i) \quad (3.23)$$

where  $\mathbf{d}_i^n$  represents the  $n$ -th feature in the feature vector  $\mathbf{d}_i$ .

The performance of each feature in different situations is demonstrated using the log likelihood ratios (LLRs) which are presented in Figure 3.8. The LLR is used to indicate whether a given site in the image is more likely have been generated by the object or the background model. This is indicated by brighter and darker intensities in Figure 3.8 respectively. For a site  $i \in \mathcal{S}$  the LLR can be written as:

$$LLR_i = \ln \left( \frac{P(\mathbf{d}_i^n | f = 0)}{P(\mathbf{d}_i^n | f = 1)} \right) \quad (3.24)$$

where  $\mathbf{d}_i^n$  represents the  $n$ -th feature in the feature vector  $\mathbf{d}_i$ , and the value of  $f = 0$  or  $f = 1$  indicates whether the foreground or background model is used respectively.

### 3.4. The Unary Potential

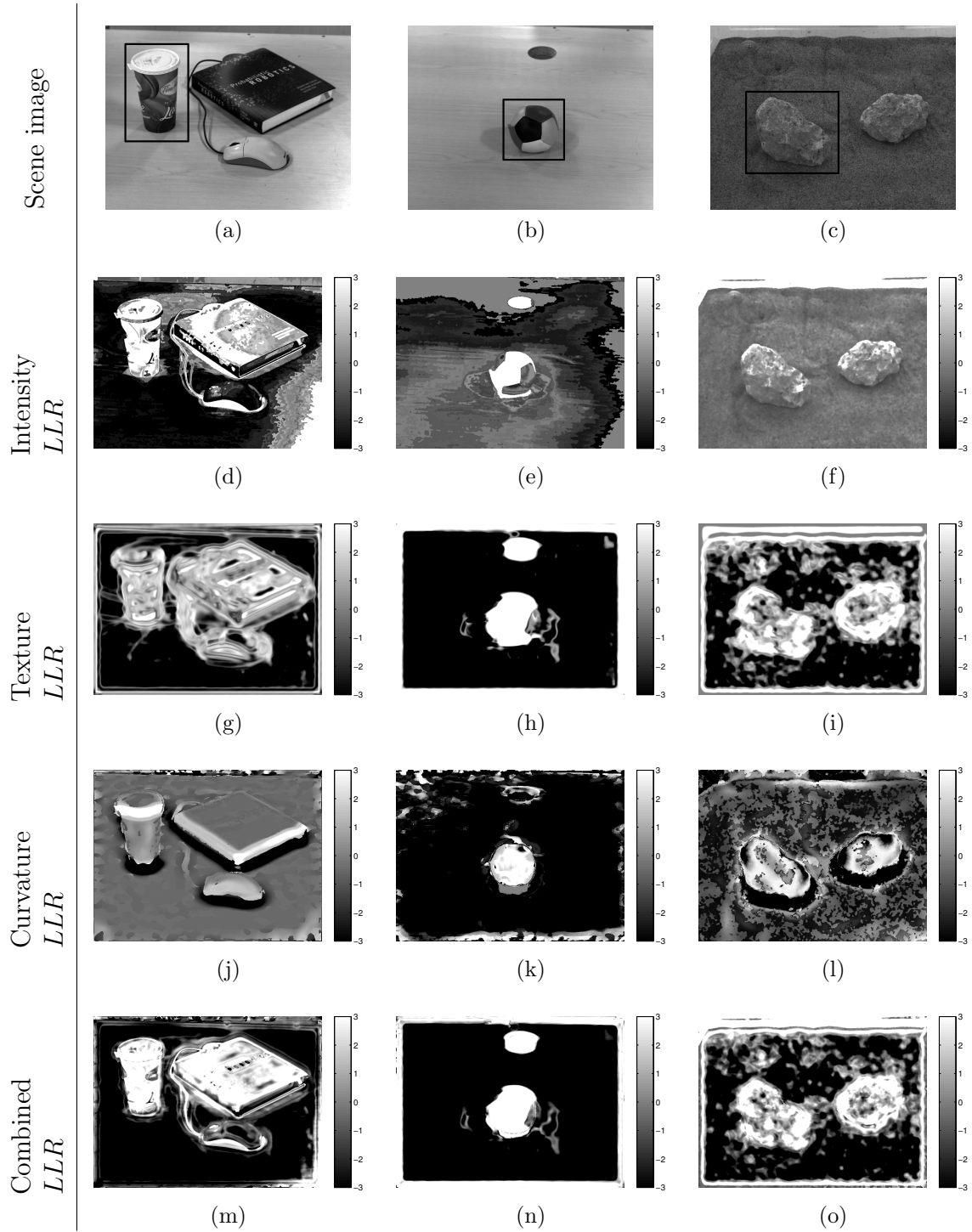


Figure 3.8: Examples of log likelihood ratios for three different objects (a)–(c) calculated using intensity models (d)–(f), texture models (g)–(i), curvature models (j)–(l) and the likelihood ratio of the combined models (m)–(o).

### 3.4.1 Intensity / Colour

Whether colour or grayscale images are available depends on the hardware. However, both features are modelled similarly. A median filter is initially applied to the image to reduce noise. A histogram of intensity values or colour vectors is then constructed for the object and the background regions. The negative log likelihood of the colour or intensity component can then be calculated as:

$$-\ln P(\mathbf{a}_i | f_i) = -\ln(h_{f_i}^{\mathbf{a}}(\mathbf{a}_i)) \quad (3.25)$$

where  $h_{f_i}^{\mathbf{a}}$  is the histogram of colour or intensity values ( $\mathbf{a}$ ) corresponding to the foreground or the background model indicated by the subscript  $f_i \in \{0, 1\}$ , and  $\mathbf{a}_i$  is the intensity value or the colour vector of the site  $i$ .

If a colour image is available additional consideration needs to be given to the colour model used. While the RGB colour space is commonly used, it suffers from several drawbacks including device dependency, perceptual non-uniformity, and a poor ability to represent observed colour separately from the lightness component. This results in two challenges for segmentation. First, models of the object and background colour become depended on the illumination. And second, the magnitudes of colour differences between two RGB vectors do not correspond to the colour difference perceived by a human.

Alternatively, colour models based on the separation of colour into hue, saturation, and intensity or lightness components can be used. While still perceptually non-uniform and device dependent, these models separate the “colour” of the object from its lightness or saturation components. This can be used to reduce the dimensionality of the model by only including the hue component, and to remove some of the model’s sensitivity to illumination and shading.

Finally, several perceptually uniform colour spaces exist including the CIE-XYZ, the CIE-Lab, and the CIE-Luv colour spaces. These spaces represent all colours by two

chroma components, and a single intensity component. The colour spaces are specifically designed to mimic human visual perception, and colour differences correlate well with human perception. For recent reviews of colour models and their applications see [Tkalcic and Tasic, 2003; Busin et al., 2008].

Figures 3.8(d)–(f) shows the LLRs for a set of objects using only the intensity histograms. Brighter intensities indicate that a pixel is more likely to belong to the object.

#### 3.4.2 Texture

Texture can be a valuable visual cue to distinguish regions that otherwise have the same mean color or intensity.

From the available texture models [Szeliski, 2010; Shapiro and Stockman, 2001; Ojala et al., 1996, 2002; Leung and Malik, 2001; Varma and Zisserman, 2005], a model based on a distribution of filter responses is used [Jain and Farrokhnia, 1991; Malik and Perona, 1990]. It should be noted that, in general, the segmentation model only requires that the texture feature can be calculated from a local region surrounding the pixel. This allows for other texture features to be used to replace or augment the selected model.

The image is first convolved with a set of frequency and orientation selective filters [Leung and Malik, 2001; Varma and Zisserman, 2005]. The set used consists of first and second derivatives of Gaussians at 6 orientations, and 5 scales ( $\sigma = \{3, 5, 7, 9, 15\}$  pixels), 8 Laplacian of Gaussian filters, and 4 Gaussian filters. The filters are  $L1$  normalized and each image patch being convolved is intensity normalized to minimize global illumination effects. The set of filtered images is also contrast normalized. As a final step, each magnitude filter response image is convolved with a large support Gaussian filter [Varma and Zisserman, 2005].

This process results in a high dimension texture feature vector for every pixel. To reduce the dimensionality of the texture features, the filter magnitude responses are projected

onto the principal components of a subset of feature vectors located within the object and background seed regions. Only the projections onto the three largest principal components are used.

A Gaussian mixture model is generated for each seed region, one for the object and one for the background. Each mixture model is set to use five components.

Figures 3.8(i)–(l) shows the log likelihood ratios for the texture models. Brighter intensities correspond to a higher likelihood that the pixel belongs to the object.

### 3.4.3 Depth

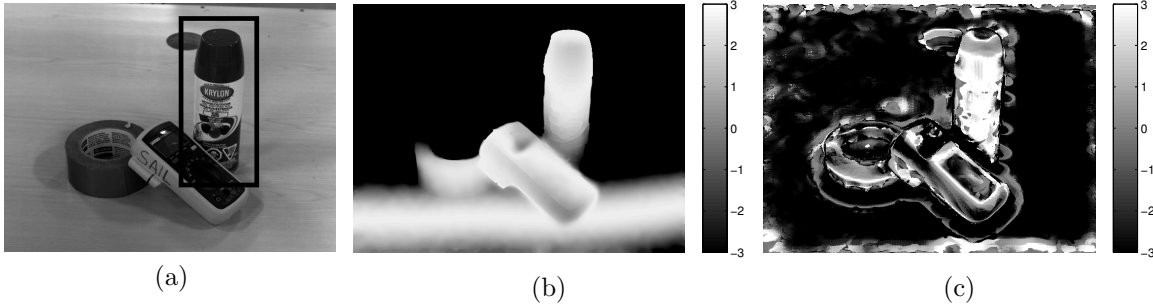


Figure 3.9: Comparison between the log likelihood ratios of depth (b) and curvature (c) models for a cylindrical object.

Depth discontinuities are strong indicators of object boundaries. Depth can be used to determine the true object boundary when multiple strong image edges are present, or when the object is very similar to the background in appearance. However, depth is not an effective cue to model object regions. Figure 3.9 shows this effect. It shows the likelihood ratio of pixels based on the depth models. The object model is constructed using the detected object seed, while the background likelihood is assumed to be uniform across the entire depth range. Figure 3.9 demonstrates that a large part of the background near the object's supporting surface is incorrectly expected to belong to the object (Figure 3.9(b)). If other cues are not sufficiently strong, this could lead to the segmentation



spilling into the background region. For this reason, depth is used only in the boundary term as described in Section 3.1.

#### 3.4.4 Curvature

Curvature is used to model the object’s geometric properties. It is also used in the boundary term to allow the algorithm to segment an object from its supporting surface without the necessity to make assumptions about the surfaces’ geometry.

The chosen curvature measure is based on the ratio of eigenvalues (see Gumhold et al., 2001) of the covariance matrix for a small neighbourhood around a point. This ratio is chosen because it is less sensitive to noise than mean or Gaussian curvature calculated using derivatives on depth images, and it requires less computation than fitting more complex surfaces to local neighbourhoods.

Let  $\{\mathbf{p}_k\}_1^{m+1}$  be a set of points in 3D corresponding to some query point  $\mathbf{p}_i$  and its  $m$  nearest neighbours, the covariance matrix  $\mathbf{M}$  can be calculated as:

$$\mathbf{M} = \frac{1}{m+1} \sum_{k=1}^{m+1} \mathbf{p}_k \mathbf{p}_k^T - \bar{\mathbf{p}} \bar{\mathbf{p}}^T \quad (3.26)$$

where  $\bar{\mathbf{p}}$  is the centroid of  $\{\mathbf{p}_k\}$ .

$$\bar{\mathbf{p}} = \frac{1}{m+1} \sum_{k=1}^{m+1} \mathbf{p}_k \quad (3.27)$$

If  $\lambda_1 < \lambda_2 < \lambda_3$  are the eigenvalues and  $\mathbf{e}_1$ ,  $\mathbf{e}_2$  and  $\mathbf{e}_3$  are the corresponding eigenvectors of the matrix  $\mathbf{M}$ , then the eigenvector  $\mathbf{e}_1$  corresponding to the smallest eigenvalue  $\lambda_1$  is also the normal  $\hat{\mathbf{n}}$  to the least squares plane fitted to the point set  $\{\mathbf{p}_k\}$ . Correct

orientation of the normal vector is enforced by making sure that it points toward the camera,  $\mathbf{p}_i^T \hat{\mathbf{n}} < 0$ . The curvature measure  $\kappa$  is calculated by dividing the smallest eigenvalue of the covariance matrix by the sum of all of the eigenvalues:

$$|\kappa| = \frac{\lambda_1}{\lambda_1 + \lambda_2 + \lambda_3} \quad (3.28)$$

The sign of the curvature can be determined by checking whether the query point  $\mathbf{p}_i$  lies in front or behind the centroid  $\bar{\mathbf{p}}$  in the direction of the surface normal:

$$\text{sign}(\kappa) = \text{sign}\left((\mathbf{p}_i - \bar{\mathbf{p}})^T \hat{\mathbf{n}}\right) \quad (3.29)$$

A two-component Gaussian mixture model is used to model curvature likelihoods. Two models are constructed from seed regions; one for the object and one for the background. The LLRs for the curvature feature can be seen in Figures 3.8(j)–(l), as well as Figure 3.9(c). Similarly to the previous two features, brighter intensities correspond to a higher likelihood that the pixel belongs to the object. In contrast to the depth model, the likelihood model based on curvature is able to identify the region belonging to the object more accurately (Figure 3.9(c)).

## 3.5 The *Pairwise* Potential

This section defines the form of the *pairwise* potential, and shows how it is evaluated using a Mahalanobis-like distance.

The *pairwise* potential  $V_2$  acts as a smoothness constraint (data dependent pseudo “prior”) penalizing discontinuities between labels, and  $T$  is a parameter controlling the relative significance of the smoothness term. It is analogous to the temperature variable

in the MRF prior. To preserve edges, the value of  $V_2$  must be high when the features at  $i$  and  $j$  are similar and low when they are different. We use a function of the form:

$$V_2(f_i, f_j, \mathbf{d}_i, \mathbf{d}_j) = \exp\left(-\frac{\boldsymbol{\rho}_{ij}^T \mathbf{S}_\rho^{-1} \boldsymbol{\rho}_{ij}}{\nu}\right) \cdot \frac{1}{|q_i - q_j|} \delta(f_i, f_j) \quad (3.30)$$

where  $\mathbf{d}_i$  and  $\mathbf{d}_j$  are feature vectors at sites  $i$  and  $j$  respectively,  $\delta(f_i, f_j)$  is one if  $f_i \neq f_j$ , and zero otherwise. The term  $|q_i - q_j|$  represents the distance in the image between the pixels  $q_i$  and  $q_j$  associated with sites  $i$  and  $j$  respectively. The difference between feature vectors is denoted as  $\boldsymbol{\rho}_{ij}$ . The components of  $\boldsymbol{\rho}_{ij}$  include the intensity or colour difference, the difference between texture features, depth difference of the two sites, and the maximum negative curvature of either site:

$$\boldsymbol{\rho}_{ij} = [\text{diff}(\mathbf{a}_i, \mathbf{a}_j), \text{diff}(\boldsymbol{\tau}_i, \boldsymbol{\tau}_j), |z_i - z_j|, \max(\kappa_i^-, \kappa_j^-)]^T \quad (3.31)$$

where  $\mathbf{a}$  is the appearance feature which represents the intensity value or colour vector of the site,  $\boldsymbol{\tau}$  is the texture feature vector, and  $\kappa^-$  is the magnitude of the negative curvature:

$$\kappa^- = \begin{cases} |\kappa|, & \kappa < 0 \\ 0, & \kappa \geq 0 \end{cases} \quad (3.32)$$

For grayscale images,  $\text{diff}(\mathbf{a}_i, \mathbf{a}_j)$  is the difference of the gray values, and for colour images it is the appropriate difference of the colour vectors. Texture difference is calculated as a Euclidean distance between the texture vectors. The patches for calculating the texture vectors are taken from the opposite, non-overlapping sides of the edge connecting sites  $i$  and  $j$ . Maximum negative curvature is used as the last element in the difference vector because it corresponds to creases in the surface that indicate object boundaries.

Finally,  $\mathbf{S}_\rho$  is the covariance matrix of feature differences in the same region. It is estimated using feature differences from within the object and background seed regions. Let  $\mathcal{R} = \{(i, j) | f_i = f_j\}$  be a set of pairs of sites such that both sites are in the same region. The matrix  $\mathbf{S}_\rho$  is the covariance matrix of feature differences associated with sites in the set  $\mathcal{R}$ :

$$\mathbf{S}_\rho = \frac{1}{|\mathcal{R}|} \sum_{(i,j) \in \mathcal{R}} \boldsymbol{\rho}_{ij}^T \boldsymbol{\rho}_{ij} \quad (3.33)$$

The exponent's numerator in (3.30) is analogous to a squared Mahalanobis distance, with the main difference being that we do not take simple feature vector differences but instead allow for more complex similarity evaluations within the components of  $\boldsymbol{\rho}_{ij}$ . This Mahalanobis-like distance weighs the differences of features across an edge (edge patterns) by the inverse of their covariance, allowing the algorithm to determine which edge patterns are assigned greater significance when the edge costs are calculated.

Combining Equations (3.18), (3.23), and (3.30) results in the following energy function:

$$\begin{aligned} E(\mathbf{f}, \mathbf{d}, \boldsymbol{\omega}) &= \sum_{i \in \mathcal{S}} \sum_{\mathbf{d}_i^n \in \mathbf{d}_i} -\ln P(\mathbf{d}_i^n | f_i) \\ &+ \frac{1}{T} \sum_{(i,j) \in \mathcal{N}} \exp \left( -\frac{\boldsymbol{\rho}_{ij}^T \mathbf{S}_\rho^{-1} \boldsymbol{\rho}_{ij}}{\nu} \right) \cdot \frac{1}{|q_i - q_j|} \delta(f_i, f_j) \end{aligned} \quad (3.34)$$

This energy function contains a set of parameters  $\boldsymbol{\omega} = \{T, \nu\}$  that control its behaviour. They are assumed to be constant and will be determined offline using a set of training images. When  $\boldsymbol{\omega}$  is known, the energy function (3.34) can be minimized using graph cuts, resulting in a solution to the segmentation problem.

## 3.6 Parameter Learning

This Section describes the process of learning the unknown parameters for the energy function.

| Parameter | Initial Value | Final Value          |
|-----------|---------------|----------------------|
| $T$       | 1             | $5.8 \times 10^{-3}$ |
| $\nu$     | 1             | 47                   |

Table 3.4: Segmentation energy function parameters before and after optimization.

The energy function (3.34) has two unknown parameters  $\omega = \{T, \nu\}$  that need to be determined before the algorithm can be used. When the task is MAP inference, the partition function  $Z(\mathbf{d}, \omega)$  is constant, and can be ignored. However when the task involves finding the optimal set of parameters given training data, the partition function depends on the parameter vector  $\omega$ , and can no longer be disregarded. This constitutes a difficult problem as the calculation of the partition function is intractable for this problem.

To avoid the calculation of the partition function when learning the parameters  $\omega$ , an energy based learning approach is used [Szummer et al., 2008; LeCun et al., 2006]. This approach seeks to find a set of parameters such that the probability of the true labelling is greater than the probability of any other possible labelling. This can be represented as:

$$P(\hat{\mathbf{f}}|\mathbf{d}, \omega) \geq P(\mathbf{f}|\mathbf{d}, \omega) \quad \forall \mathbf{f} \neq \hat{\mathbf{f}} \quad (3.35)$$

where  $\mathbf{d}$  is the set of data from the training image,  $\hat{\mathbf{f}}$  is the true labelling, and  $\mathbf{f}$  is any other labelling. Because the data  $\mathbf{d}$  and the parameter vector  $\omega$  are the same on both sides of the inequality, the partition function can be ignored. The problem can be

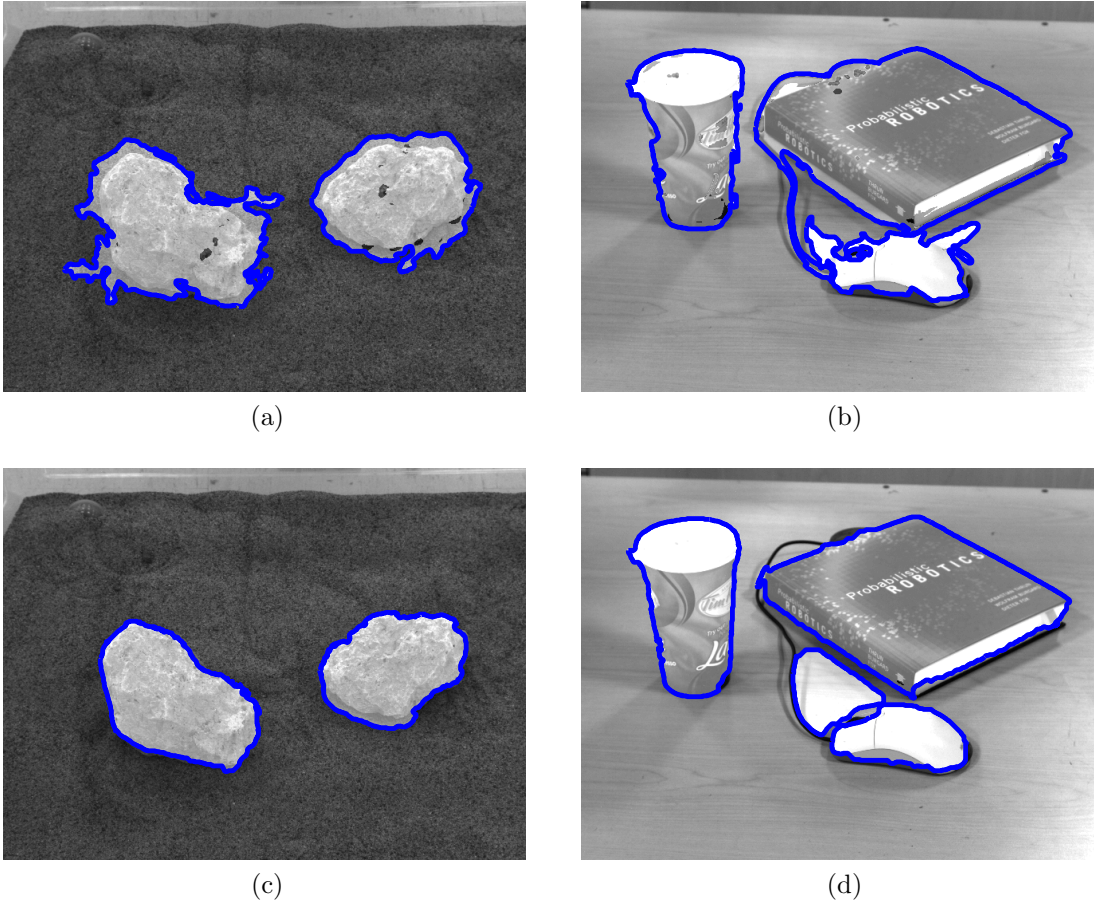


Figure 3.10: Segmentation results using the initial (a)–(b) and the final (c)–(d) values for the energy function parameters.

restated as finding the set of parameters  $\omega$  that result in the energy of the true labelling being as low as possible with respect to all other possible incorrect labellings:

$$\min_{\omega} E(\hat{\mathbf{f}}, \mathbf{d}, \omega) - E(\mathbf{f}, \mathbf{d}, \omega) \quad \forall \mathbf{f} \neq \hat{\mathbf{f}} \quad (3.36)$$

Although checking all of the possible labellings  $\mathbf{f} \neq \hat{\mathbf{f}}$  is intractable, it is sufficient to check the labellings with the lowest energy  $\mathbf{f}^* \neq \hat{\mathbf{f}}$ , which can be found using graph cuts and the current estimate of  $\omega$  as described in Section 3.1. The algorithm loops over all of the training images. For all images where the calculated labelling  $\mathbf{f}^*$  is not within a

small margin of the ground truth labelling  $\hat{\mathbf{f}}$ , the labelling  $\mathbf{f}^*$  is added to the constraint set  $\Theta = \Theta \cup \mathbf{f}^*$ . The parameters  $\omega$  are then adjusted so that the ground truth labelling has the lowest possible energy relative to the constraint set:

$$\min_{\omega} E(\hat{\mathbf{f}}, \mathbf{d}, \omega) - E(\mathbf{f}, \mathbf{d}, \omega) \quad \forall \mathbf{f} \in \Theta \quad (3.37)$$

Because the energy function (3.34) is not linear in the parameters  $\omega$ , a direct search algorithm is used to minimize (3.37) (see Kolda et al., 2003). The process is repeated until  $\omega$  stops changing, the iteration limit is reached, or the algorithm is unable to find a set of parameters to satisfy Equation (3.37). Table 3.4 shows the initial and learned parameters of the energy function. Figure 3.10 shows initial (Figure 3.10(a), Figure 3.10(b)) and final (Figure 3.10(c), Figure 3.10(d)) segmentation results for a pair of experimental images.

## 3.7 Experimental Results

To investigate the effectiveness of the proposed segmentation method, it is used to segment scenes of various complexity levels. Several application cases were considered. Case one consisted of detecting and segmenting objects in an office or domestic setting. It involved detecting various household objects placed in a variety of environments. The less complex scenes involved a single object, while the more complex scenes involved multiple objects in close proximity, with some objects resting on top of other objects of similar appearance. Examples of this category are shown in Figure 3.18 scenes 1–12, 16–20, and 24–16. Case two was intended to simulate what a rover may observe while segmenting a sample rocks resting in sand. The complexity ranged from rocks well separated in space and resting on top of a planar surface, to multiple rocks buried to various extents in the sand and located in close proximity to each other. Example of scenes from this category can be seen in Figure 3.18 scenes 13–15. Additional challenging scenes, where

the object and background had the same appearance are shown in Figure 3.18 scenes 21–23 and 28–30. Finally, to demonstrate the applicability of the proposed segmentation method to door and handle environments, four door handle segmentation scenes are shown (Figure 3.18 scenes 31–34).

In total, the algorithm was applied to 34 scenes. The scenes, along with ground truth labellings, and segmentation results are shown in Figure 3.18. Note that objects in Figure 3.18 marked with an “X” were not detected using the geometric object detection method.

For each scene, the ground truth segmentations were obtained manually. In all cases, the region models are computed from seed regions generated by the object detection algorithm. The parameters in the segmentation energy function are determined as described in Section 3.6. All parameters are kept constant for all of the experiments.

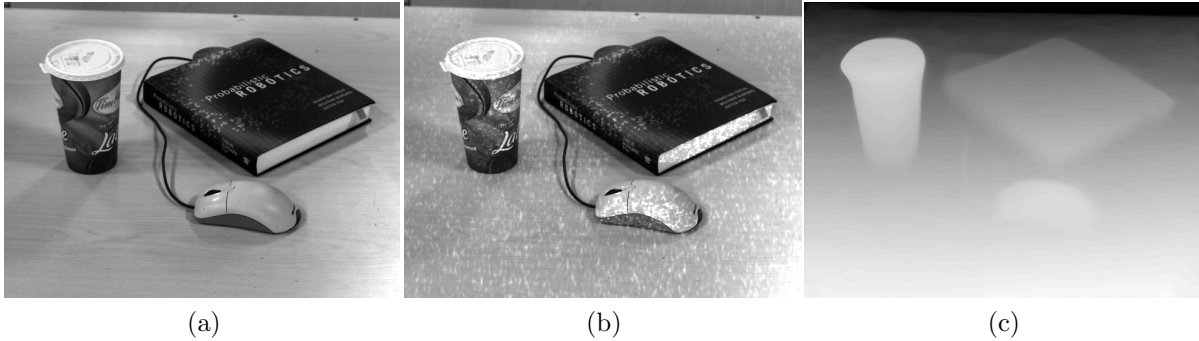


Figure 3.11: Example depth map reconstructed with the aid of a projected light pattern. (a) Image of the scene. (b) Image of the scene with a projected light pattern. (c) Reconstructed depth map.

The scenes are captured with a Basler A601f camera, with a resolution of 656x491 pixels. The camera is mounted on the end-effector of a 5-axis robotic arm. A series of images are taken of each scene. Encoders on the arm are used to gather position data of each joint. The data is used to calculate the pose of the end-effector mounted camera at the time each image is taken. Optical flow [Zach et al., 2007] is extracted from image pairs, and used with the pose of the camera to reconstruct the 3D structure of the scene. To improve the



quality of 3D reconstruction, a random pattern is projected onto the scene after the image used for appearance and texture models is captured [Rusu et al., 2009a]. The pattern adds additional detail to surfaces that would otherwise be untextured, allowing the optical flow algorithm to obtain better 3D reconstruction results. Figure 3.11, which shows an example scene without the projected light pattern (Figure 3.11(a)), with the projected light pattern (Figure 3.11(c)), and the reconstructed depth map (Figure 3.11(c)).

All steps of the algorithm were executed on a computer with an Intel Q9550 processor, with 8 GB of memory. With the exception of the optical flow algorithm, and the graph cuts algorithm, all other the steps of the segmentation method were implemented in Matlab 8.1 (release R2013a). A summary of the method's computational performance is shown in Table 3.5. The table shows the mean time required to complete each step of the algorithm, as well as the standard deviation in the computation time. The maximum time observed when computing each step is also shown.

#### 3.7.1 Object Detection

Figure 3.12 shows the results of the object detection process for the tested scenes. It shows the precision (Figure 3.12(a)) and recall (Figure 3.12(b)) measures for each scene.

Precision and recall measures are used to evaluate the performance of the object detection algorithm, as well as part of the performance evaluation of the segmentation algorithm. Precision and recall are defined as:

$$precision = \frac{TP}{TP + FP} \quad (3.38)$$

$$precision = \frac{TP}{TP + FN} \quad (3.39)$$

| Algorithm Step                    | Time Required (s)    |                      |                      |
|-----------------------------------|----------------------|----------------------|----------------------|
|                                   | Mean                 | $\sigma$             | Maximum              |
| Optical flow for 16 images        | 307.788              | 3.494                | 318.604              |
| Depth map calculation             | 523.302              | 5.452                | 534.497              |
| Point cloud construction          | 36.036               | 9.042                | 50.141               |
| Curvature calculation             | 78.334               | 0.768                | 80.622               |
| Texture features calculation      | 27.568               | 2.131                | 31.128               |
| Object detection                  | 1.897                | 0.003                | 1.901                |
| Intensity model construction      | $1.2 \times 10^{-4}$ | $7.3 \times 10^{-6}$ | $1.4 \times 10^{-4}$ |
| Texture model construction        | 3.246                | 0.327                | 3.614                |
| Curvature model construction      | $1.3 \times 10^{-4}$ | $8.5 \times 10^{-6}$ | $1.6 \times 10^{-4}$ |
| Graph construction                | 1.135                | 0.003                | 1.138                |
| Edge covariance matrix estimation | 4.993                | 0.699                | 6.751                |
| Edge cost evaluation              | 0.193                | 0.035                | 0.281                |
| Graph cuts                        | 0.269                | 0.020                | 0.304                |
| Combined algorithm                | 981.521              | 10.237               | 1005.204             |

Table 3.5: Computational performance of the proposed segmentation algorithm.

where TP, FP, and FN stand for *true positive*, *false positive*, and *false negative*, respectively. A true positive corresponds to a correctly detected object in the scene. A false positive is any detected object region, where in reality an object is not present at that location. A false negative is counted when an object in the scene is not detected by the method. If a single object is detected in multiple parts, one part is considered a true positive detection, while the remaining parts are considered false positives.

Precision (Equation (3.38)), measures the fraction of correctly detected objects with respect to the total number of detected objects. A high precision score indicates that all of the detected objects were correct, while a low precision score indicates a high number of false detections.

### 3.7. Experimental Results

---

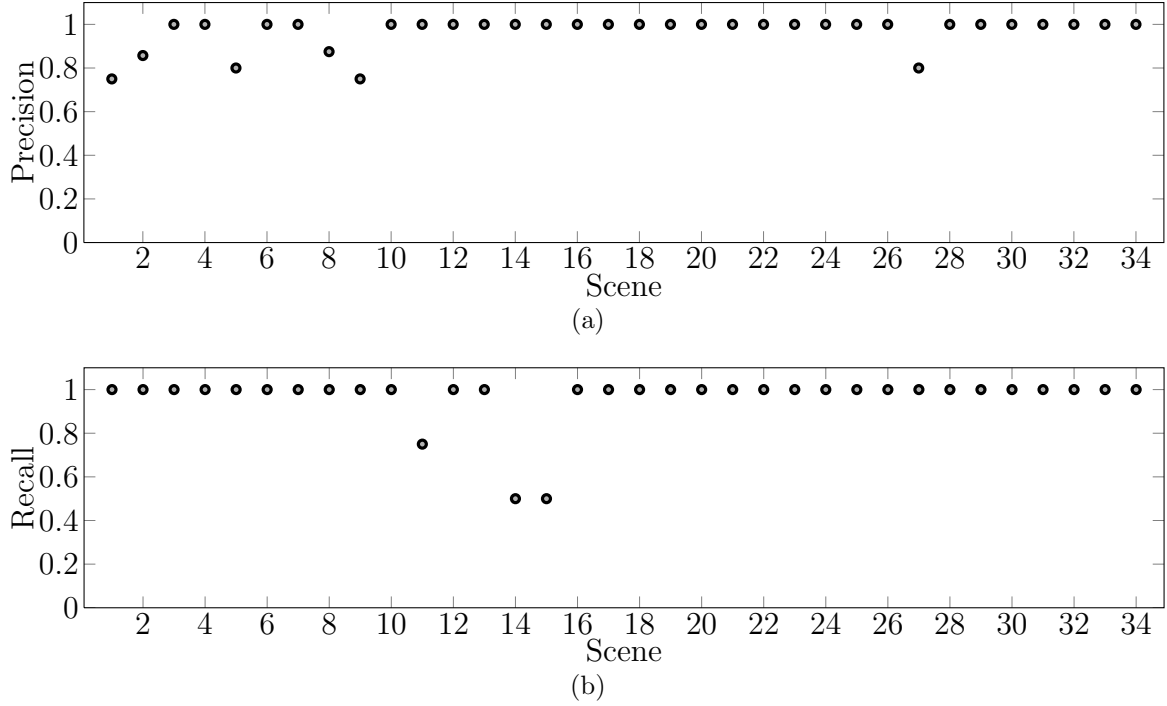


Figure 3.12: Precision (a) and recall (b) scores for the object detection step. Scene numbers are shown in the first column of Figure 3.18.

Recall (Equation (3.39)), measures the fraction of correctly detected objects with respect to the number of objects that are known to be present in the scene. A high recall rate indicates that most objects are detected, while a low recall rate indicates that the method is not able to detect objects that are present.

High recall rates are noted in the majority of cases, indicating that the method was able to detect most objects with few false negatives. Lower recall rates are observed in the rock sample segmentation cases (Figure 3.18 scenes 14 and 15).

When the rocks are mostly buried in the sand, there is no crease region of high curvature to complete the contour required for object detection. This resulted in the algorithm not being able to detect the objects.

In Figure 3.18 scene 11, two coffee cups were detected as a single object. The top surfaces of the cups are on the same plane and touching. In such a scenario, neither a depth

or curvature edge is present at the contact point of the two objects, and the detection method is unable to differentiate the two objects.

Precision rates for object detection are lower, indicating that the method is more likely to detecting false positives. Edges that appear due to noise in the 3D data can form closed contours with themselves or parts of objects in the scene, resulting in false detections. For example the small region between the book and the mouse in scene 1 of Figure 3.18.

In Figure 3.18 scene 8, the tea cup is detected as two separate objects, resulting in one of the two parts being considered a false positive. Due to the thin structure of the cup, there is a sharp depth discontinuity between the front and the back surfaces. Because the connecting region is thin, it is filled in during the object detection process. This results in the algorithm detecting two objects, one corresponding to the front surface of the cup, and a second one corresponding to the back surface. This is not the case when the object is thicker, such as the duct tape in Figure 3.18, scene 12, which is successfully detected as a single object. Two other instances where an object is detected in multiple parts can be seen in Figure 3.18, scene 9, where the teddy bear and the plush Stewie doll are each detected as multiple objects. Each detected part of these objects is separated by a either depth edges as in the case of the teddy bear, or by curvature edges, as in the case of the plush doll. These edges cause the algorithm to detect multiple objects, where only one complex object is present.

### 3.7.2 Segmentation

Segmentation results can be seen in Figure 3.18. A quantitative evaluation of the segmentation method's performance is shown in Figure 3.14, and Figure 3.15. Figure 3.14 shown the  $F_1$  (Figure 3.14(a)), precision (Figure 3.14(b)) and recall (Figure 3.14(c)) scores. Figure 3.15 shows the mean (Figure 3.15(a)) and maximum (Figure 3.15(b)) distances between the true and detected object boundaries. These measures provide for an evaluation of the method's performance both in terms of area and perimeter errors.

### 3.7. Experimental Results

---

The precision and recall values are calculated as described in Equation (3.38) and Equation (3.39) respectively. In the context of segmentation a true positive is any object pixel that was correctly labelled as an object. Background pixels that were correctly labelled as background are not counted. A false positive is any background pixel that was incorrectly labelled as an object pixel, and a false negative is any object pixel that was incorrectly labelled as background.

The precision score can be considered as an evaluation of how much of the object region spills onto the background. A high precision indicates that most of the pixels labelled as belonging to the object are correct. A low precision rate indicates that a large portion of pixels predicted to belong to the object, in reality, are part of the background.

The recall rate allows for evaluation of how much of the object was detected. A high recall rate indicates that a large portion of the object was detected correctly. A low recall rate indicates that parts of the object were missed.

A combined measure of these two metrics is the  $F_1$  score. It is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \quad (3.40)$$

The  $F_1$  measure is an overall indicator of the algorithm's performance, taking into account both how much of the true object was detected, and to what extent the segmentation spills onto the background.

It was found that the algorithm was able to accurately segment most detected objects in the test scene. Specifically, the method was able to perform well in several challenging scenarios, including cases where the object and the background had a similar appearance and cases where multiple objects were in contact with one another or when one object was resting on top of another. The proposed method was also able to function in cluttered environments.

The method's performance when the object's appearance is nearly identical to that of the background is demonstrated in Figure 3.18 scene 5, and scenes 21 through 30. The mean intensities of the school bus and its background, the book and the pop can in scene 5, as well as most of the objects in scenes 21 through 30, are nearly identical. While intensity edges can still be seen between the eraser box and the tissue box, such edges are not present for a large portion of the school buses' and the pop can's boundaries. In scenes 21 through 23, the checker pattern makes it difficult to distinguish intensity edges between objects. It is important to note, that a number of objects in these scenes are placed on surfaces that do not correspond to a dominant plane. In these cases, the assumption that the objects rest on a dominant plane would be invalid. In each case, the objects were successfully segmented from the background, and from each other.

Several examples of segmentation in cluttered environments are shown in Figure 3.18 (for example scenes 2, 8, 9, 23, and 27, as well as others). In these cases, a number of objects are placed in a scene with a complex background. In several scenes, the appearances of the background and the objects are similar (for example scene 23). The rest of the scenes contain multiple objects in contact with one another. Under these challenging conditions, the algorithm was able to accurately segment most of the detected objects in each scene.

While the algorithm performed well overall, a number of issues were noted. In several cases the segmentation can be seen spilling onto the object's supporting surface (the school bus in Figure 3.18, scene 2), or cutting off a part of an object near its supporting surface (the toy frog in Figure 3.18, scene 2). Even when intensity edges separating the object from the background are strong and well defined, there can be a large amount of intensity variation within seed regions. This is not so for curvature and depth, as these values tend to be more constant or change slowly. Additionally, high negative curvature regions are not common on most small graspable objects. In these situations, the algorithm tends to follow high depth or high curvature edges which can be less precise than intensity edges. Because curvature is calculated based on neighbourhoods of points taken from a noisy depth map, its maximum does not always correspond to that of the

true boundary, resulting in the segmentation appearing to be less accurate. This can also be seen to a lesser extent near depth edges, but since the depth data used for this experiment was obtained from optical flow, depth discontinuities are likely to coincide with intensity edges, thus the effect is lessened.

An example of a poor segmentation can be seen in Figure Figure 3.18, scene 8, between the tea box in the foreground, and the cereal box in the background. In this example, the appearance of the two objects is very similar in the area where the segmentation is spilling from the tea box onto the cereal box. At the same time, the tea box is close enough to the cereal box for depth difference to be small, and just far enough away that the curvature in the area is near zero. This results in the segmentation algorithm following the strong intensity edge, which is located on the surface of the cereal box itself. When other cues are present, for instance where the tea box meets the pop cans or the plate, segmentation remains accurate.

Thin or non convex objects can be detected as two or more separate objects. For thin objects, parts may be missed altogether. Examples of this type of error are the segmentation results of the tea cups in Figure 3.18, scenes 6 (object 20), scene 7 (object 21), and scene 8 (object 25), as well as the teddy bear and the Stewie doll in Figure 3.18, scene 9 (objects 31 and 35 respectively). In these situations the segmentation algorithm is unable to recover from the detection error. Since different parts of these objects are separated from other parts by contours of curvature or depth, and because of the added constraints that are based on the initial region detection, the segmentation algorithm is unable to expand into adjacent object regions. Correspondingly, these scenes are observed to have the lowest recall, and  $F_1$  scores (Figure 3.14), as well as the highest mean and maximum distance between the predicted and the true boundaries (Figure 3.15).

#### 3.7.3 Comparison with Prior Work

To compare the performance of the proposed segmentation method, three methods for object segmentation were implemented. Implemented methods included the graph-based

clustering method of Rao et al. [2010], the CRF based methods presented by Johnson-Roberson et al. [2010], and the method proposed by Bjorkman and Kragic [2010a,b]. The methods presented in the original works were modified to use grayscale images. For each of the methods, any free parameters were optimized using the same training set that was used to determine the parameters for the proposed method. Methods requiring initialization were provided with the same object seed region as was used by the proposed algorithm.

While all three tested methods performed well in simple environments where the objects had different intensities, the methods of Johnson-Roberson et al. [2010] and Rao et al. [2010] were unable to cope with situations where the objects had strong internal intensity edges, or if the intensity distributions of multiple objects were similar. The method proposed by Bjorkman and Kragic [2010a,b], however, performed exceptionally well on a number of challenging test scenes. For this reason, it is used as the benchmark to measure the performance of the proposed segmentation algorithm. The other two methods are not considered further.

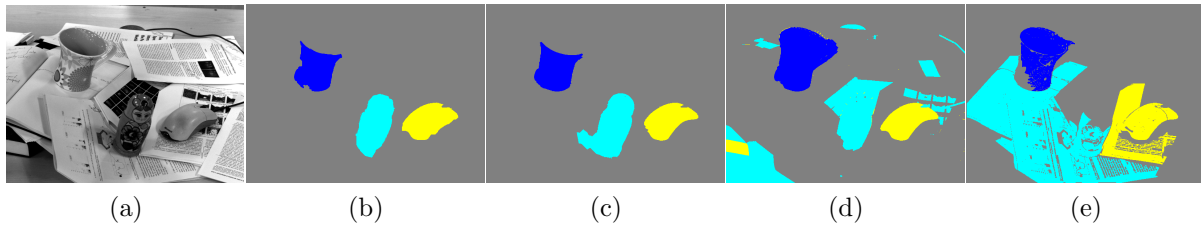


Figure 3.13: Sample segmentation results for the implemented segmentation methods. (a) Scene image. (b) Proposed method. (c) Bjorkman and Kragic [2010a,b]. (d) Johnson-Roberson et al. [2010]. (e) Rao et al. [2010].

Results comparing the performance of the proposed segmentation algorithm to the method of Bjorkman and Kragic [2010a,b] are shown in Figure 3.14 and Figure 3.15. A summary of the results is shown in Table 3.6, and Table 3.7. These tables show the mean and standard deviations of the performance metrics for the proposed algorithm and the method of Bjorkman and Kragic [2010a,b].



### 3.7. Experimental Results

The results demonstrate that the proposed segmentation algorithm performs on average better than the benchmark method of Bjorkman and Kragic [2010a,b]. The proposed algorithm shows a higher average  $F_1$  and precision scores, with less variation in these measures. The recall scores for the two methods are similar. The proposed method also shows lower mean and maximum distances between the detected and true object boundaries, with less variation in these values as well.

| Method                        | F1     |          | Precision |          | Recall |          |
|-------------------------------|--------|----------|-----------|----------|--------|----------|
|                               | Mean   | $\sigma$ | Mean      | $\sigma$ | Mean   | $\sigma$ |
| Proposed method               | 0.8979 | 0.0896   | 0.9780    | 0.0540   | 0.8441 | 0.1331   |
| Bjorkman and Kragic [2010a,b] | 0.8367 | 0.1671   | 0.8564    | 0.2341   | 0.8753 | 0.1302   |

Table 3.6: Mean and standard deviation ( $\sigma$ ) of the  $F_1$ , precision, and recall scores for the proposed method, and the method of Bjorkman and Kragic [2010a,b].

| Method                        | Mean Distance to True Object Boundary |          | Maximum Distance to True Object Boundary |          |
|-------------------------------|---------------------------------------|----------|--|----------|
|                               | Mean                                  | $\sigma$ | Mean                                     | $\sigma$ |
| Proposed method               | 5.8074                                | 8.4696   | 23.7206                                  | 28.6687  |
| Bjorkman and Kragic [2010a,b] | 13.9709                               | 19.5428  | 46.7471                                  | 51.1294  |

Table 3.7: Mean and standard deviation ( $\sigma$ ) of the average and maximum distances to the true object boundary for the proposed method, and the method of Bjorkman and Kragic [2010a,b].

The proposed method exhibits similar performance to the method of Bjorkman and Kragic [2010a,b] in situations where the objects are well separated in space, or where a strong intensity edge exists between objects (Figure 3.16).

In cluttered scenes where multiple objects of similar intensity are in contact with one another, or when one of the object rests on top of another, the proposed segmentation

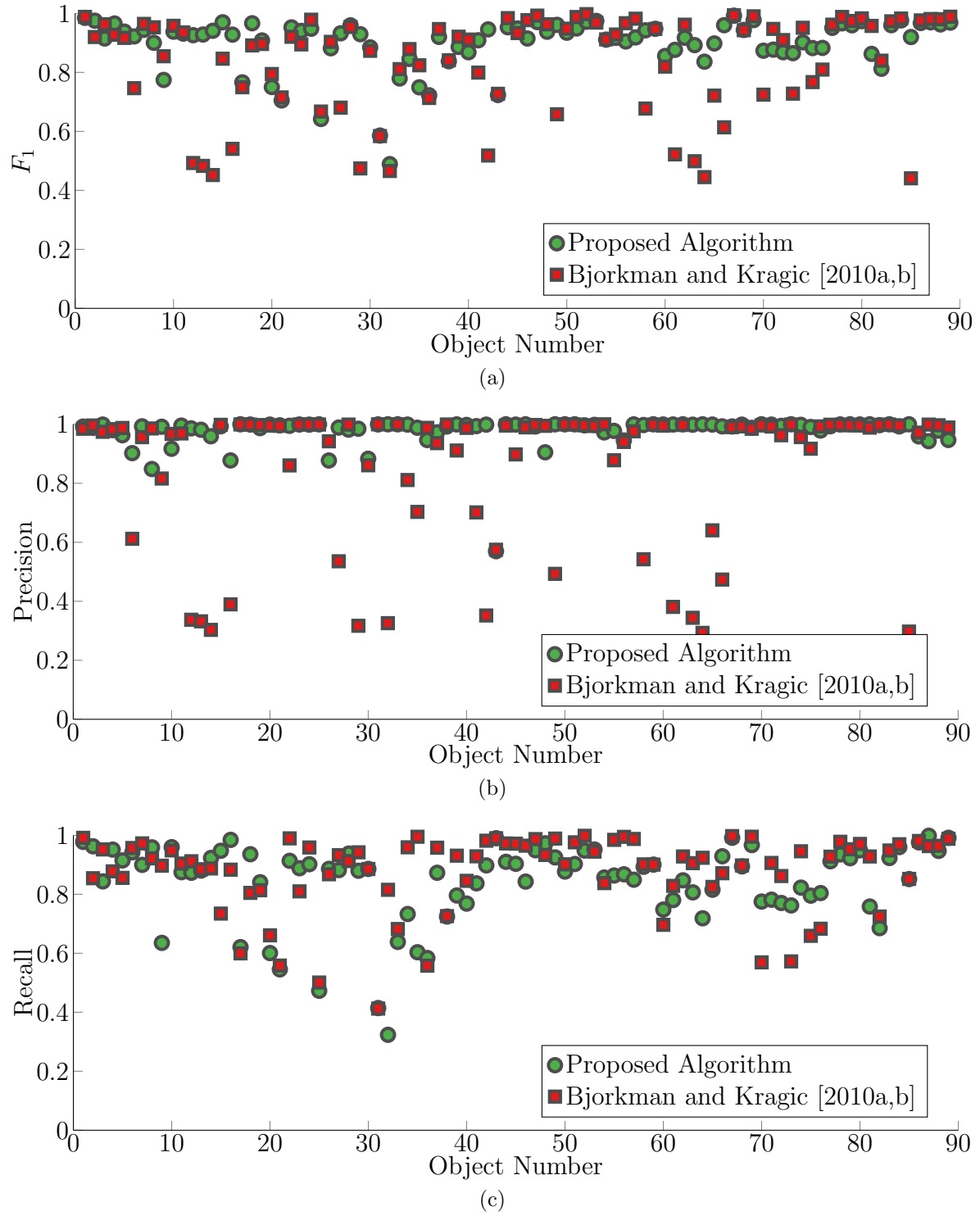


Figure 3.14: Comparison of the  $F_1$  (a), precision (b), and recall (c) score for the proposed segmentation algorithm and the method of Bjorkman and Kragic [2010a,b].

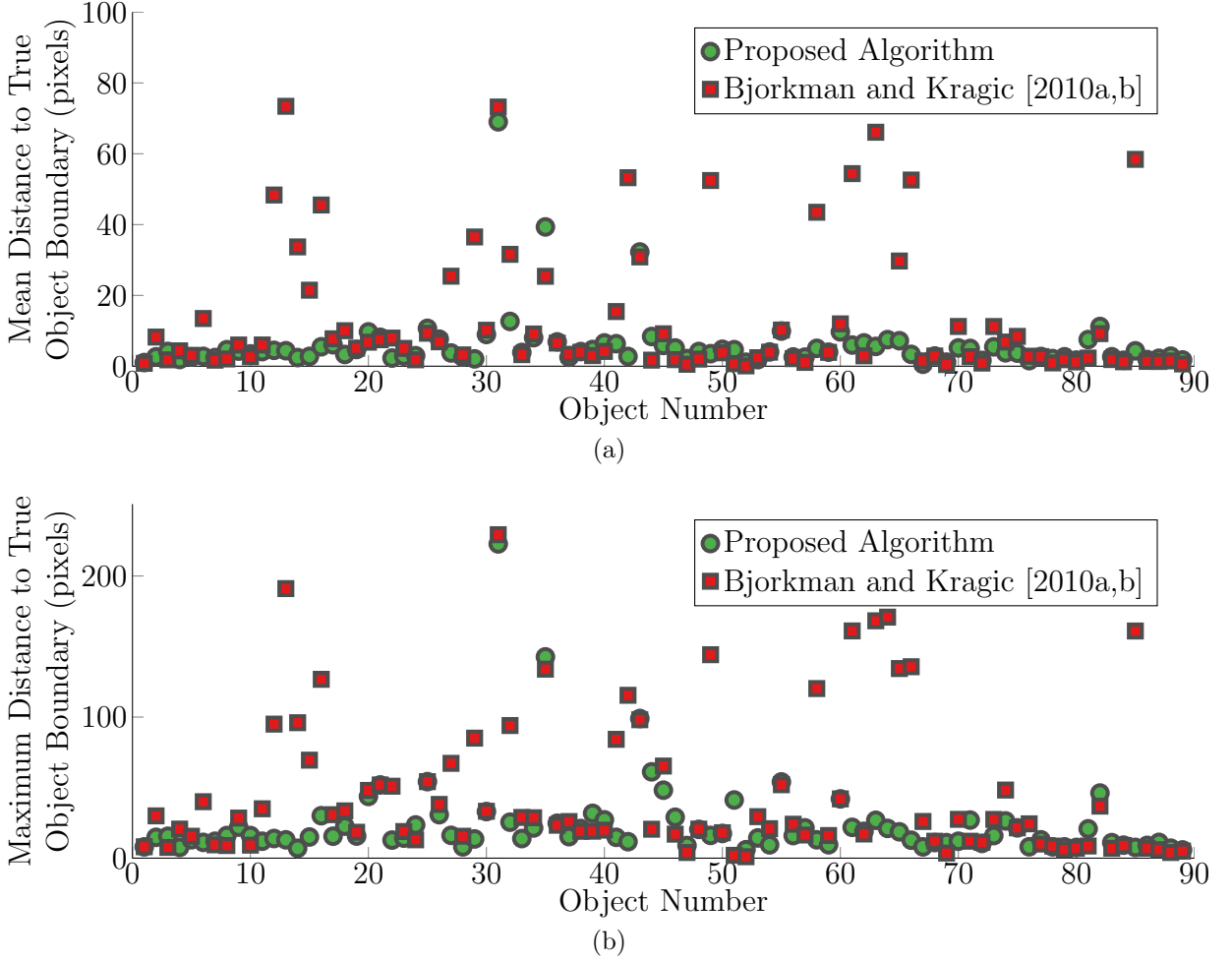


Figure 3.15: Comparison of the mean (a), and maximum (b) distance to the true object boundary for the proposed segmentation algorithm and the method of Bjorkman and Kragic [2010a,b].

method shows superior performance (Figure 3.17).

The dominant plane, and Gaussian position-depth models used by the method of Bjorkman and Kragic [2010a,b] are not useful in segmenting objects from other contacting objects of similar appearance. In contrast, the proposed method is able to utilize the additional information provided by curvature and texture to better predict which regions should belong to the object. Additionally, by employing the curvature feature, the proposed method is able to detect boundaries between contacting objects even when they

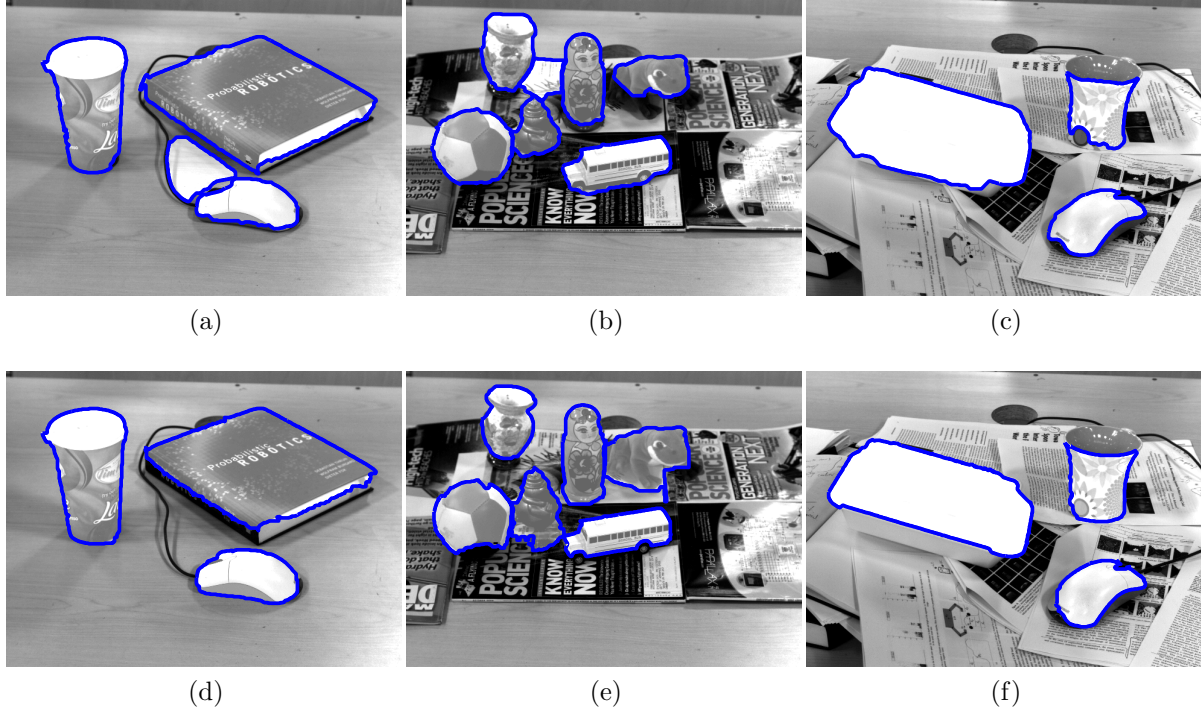


Figure 3.16: Comparison of example segmentation results for the proposed method (a)–(c), and the algorithm presented by Bjorkman and Kragic [2010a,b] (d)–(f).

otherwise have the same appearance.

### 3.8 Conclusions and Discussion

This chapter presents a method for segmenting objects in cluttered environments. The proposed method has incorporated both appearance and geometric cues, which allow it to be more robust in situations where some of the data is ambiguous. The problem is formulated using the Conditional Random Fields framework, and a globally optimal solution is obtained efficiently using the graph cuts energy minimization technique. A simple object detection method based on closed contours of depth edges and high curvature edges is used to initialize the segmentation algorithm. The effectiveness of the

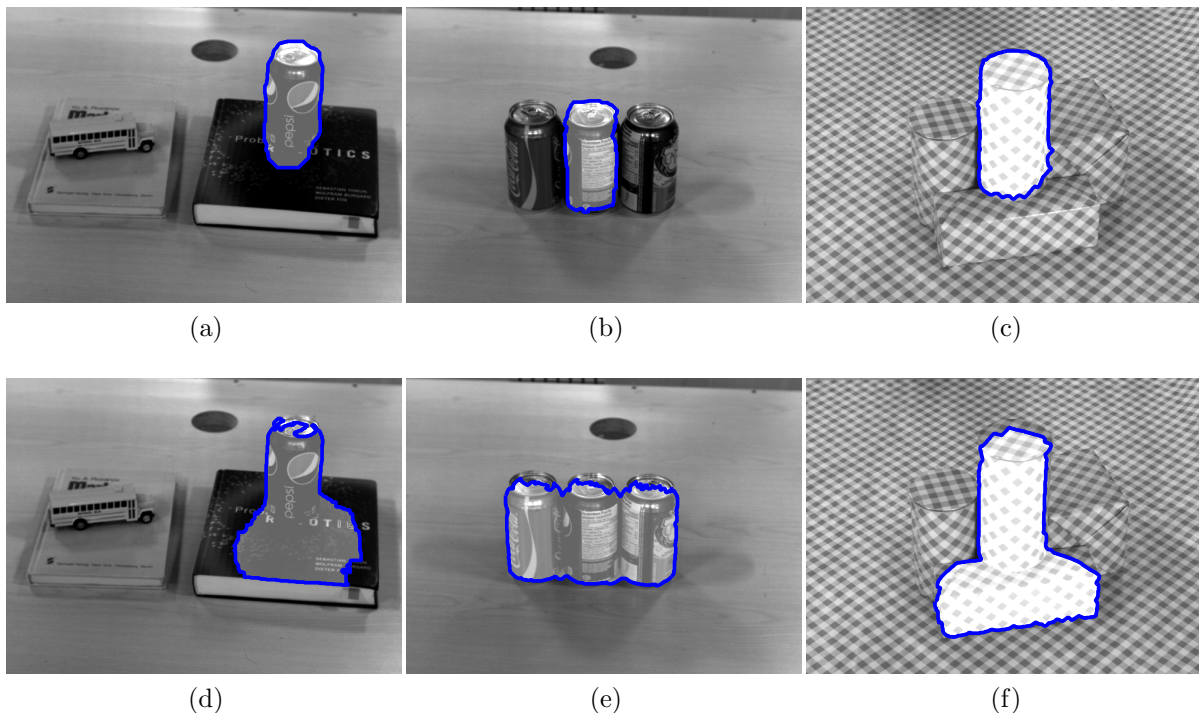


Figure 3.17: Example segmentation results for scenes where the proposed segmentation algorithm (a)–(c) exhibits better performance when compared to the method of Bjorkman and Kragic [2010a,b] (d)–(f).

proposed method has been verified by experiments performed on a number of scenes.

The method was able to detect and segment most of the objects it was tested on, even in challenging scenarios where the object’s appearance was nearly identical to that of the background and when objects were in contact with each other. The method was able to perform segmentation without requiring the assumption that the objects rest on the dominant planar surface in the scene, which allowed for the segmentation of objects when one was placed on top of the other.

When compared against other methods in the literature, the proposed method matched or exceeded their performance, specifically, the proposed method exhibited superior performance in cluttered scenes where objects were in contact with one another, or where some objects were placed on top of other objects of similar appearance.

A limitation of the proposed object detection system is its reliance on 3D data. If the 3D data is too noisy, smaller objects may be missed or regions that do not correspond to actual objects may be incorrectly detected. If an object is not detectable with the available 3D data, its segmentation will not be performed, even if it would have been possible based on appearance cues.

In the presented method, depth information was only used in the *pairwise* smoothness term to detect object edges. While incorporating depth can increase the accuracy of the region term where the object is far removed from any other nearby object or from the background, at points where the object is in close proximity to other object, depth can cause the segmentation to spill. Excluding depth or location from the *unary* term avoids the issue of segmentation spilling onto adjacent surfaces, but also ignores potentially useful information. A method to incorporate depth into the regional terms of the segmentation energy function, while avoiding issues with segmentation spilling onto adjacent surfaces could improve the accuracy of the segmentation results.

The use of multiple segmentation cues requires for the algorithm to be able to determine which of the cues, or combinations of cues, should be assigned greater significance. Since the unary region term  $V_1$  is a negative log of a product of individual feature likelihoods, the relative significance of the individual features is implicitly accounted for. However, in the boundary term  $V_2$ , the relative significance of features has to be addressed explicitly.

To weigh the relative significance of available segmentation cues in the boundary term of the energy function  $V_2$ , the proposed method utilizes a squared Mahalanobis-like distance in the exponent of the term. The covariance matrix used by the distance is calculated based on the edges found inside of the object and background seed regions. A lower distance (higher cost) is assigned to edges where the feature difference across the edge is small compared to the variation of the feature differences within the object and the background region. While this approach is effective at finding edge patterns that do not occur commonly within object or background regions, it ignores the feature similarity or difference between the different regions. The method is also unable to properly judge the importance of edges where multiple cues vary simultaneously. For example, one may

expect that edges where depth and intensity, or depth and curvature vary simultaneously to be more likely to correspond to object boundaries than edges where any feature varies individually. Such a relationship cannot be observed within either the object or the background seed regions.

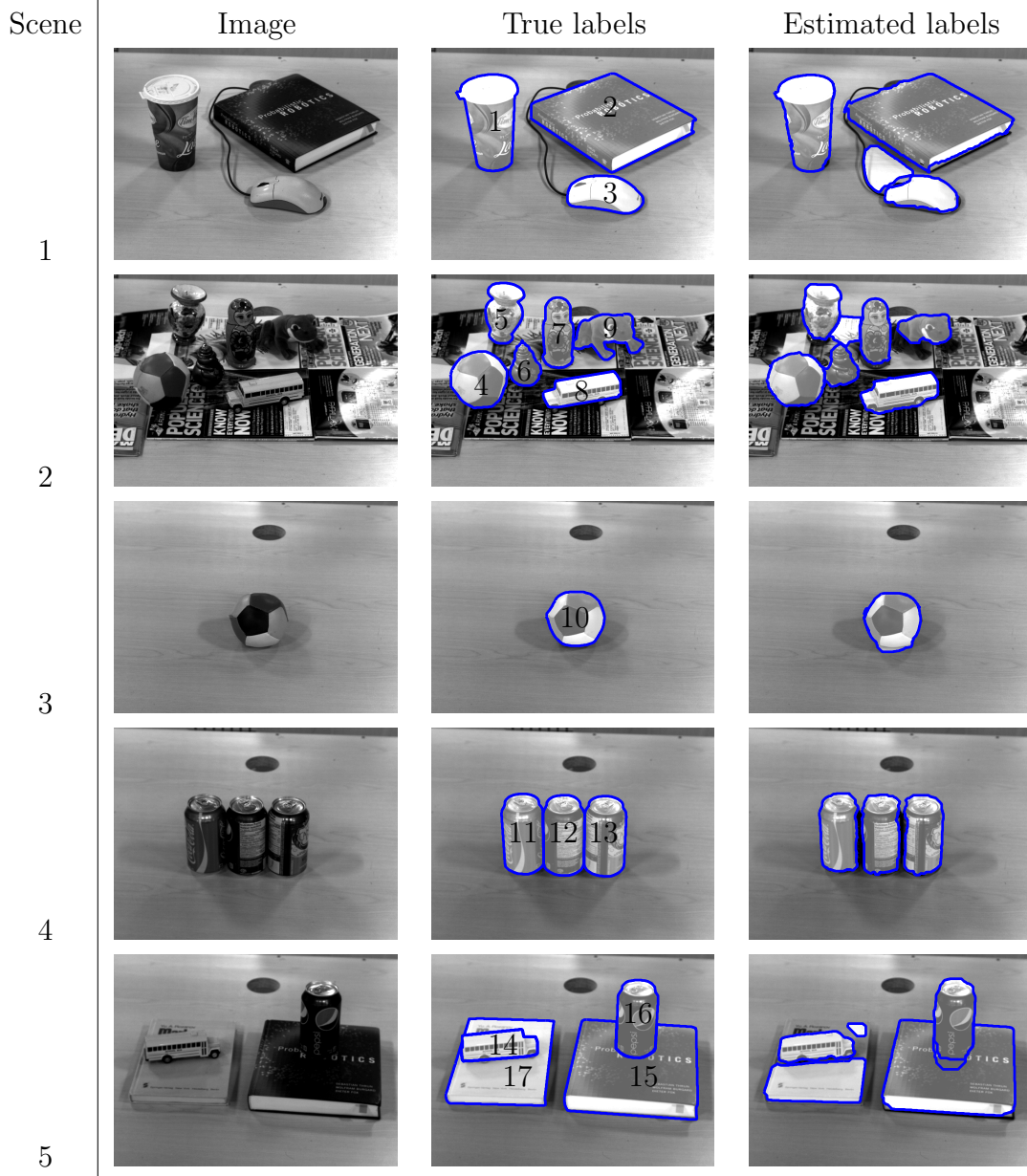


Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.



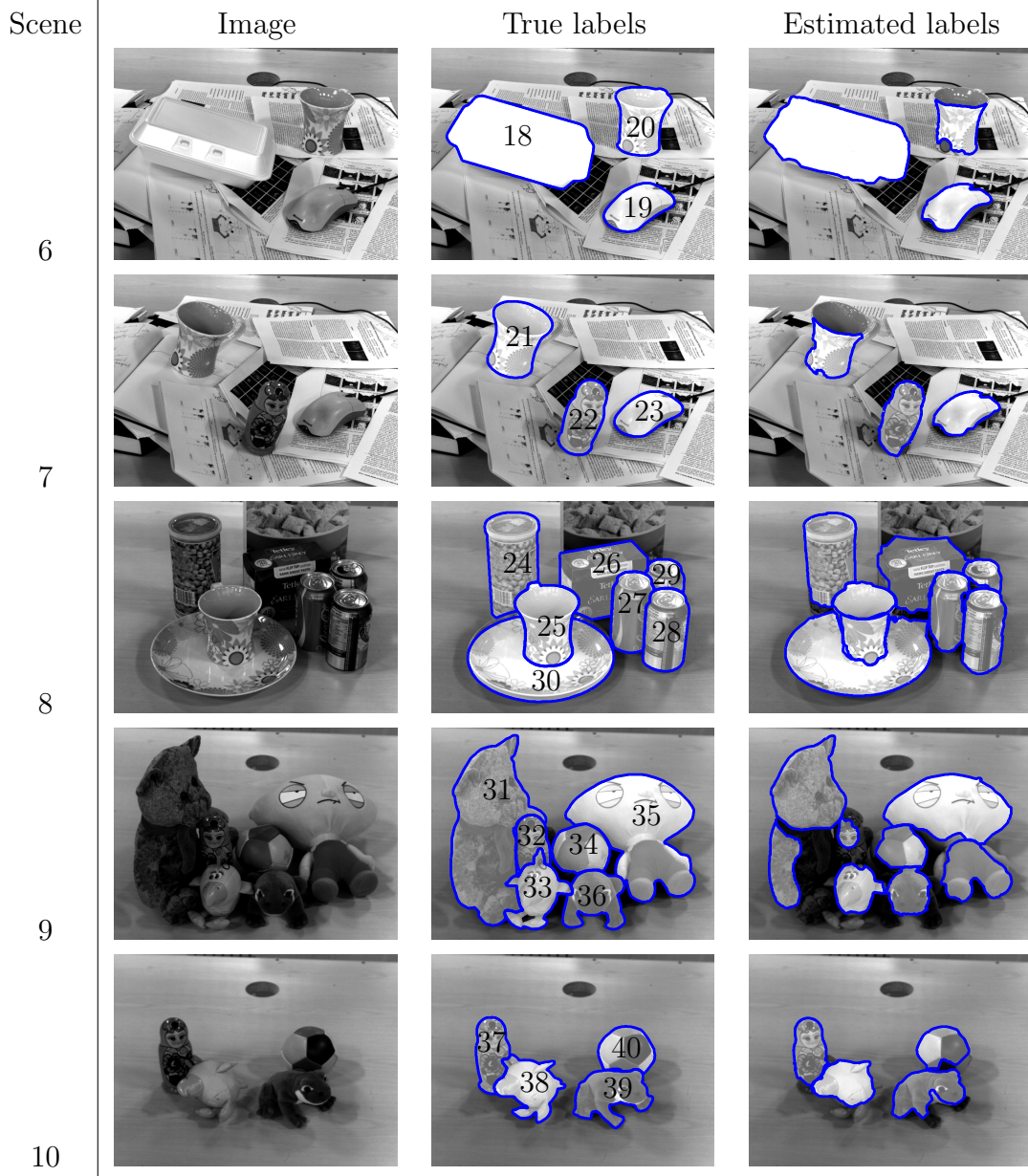


Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.

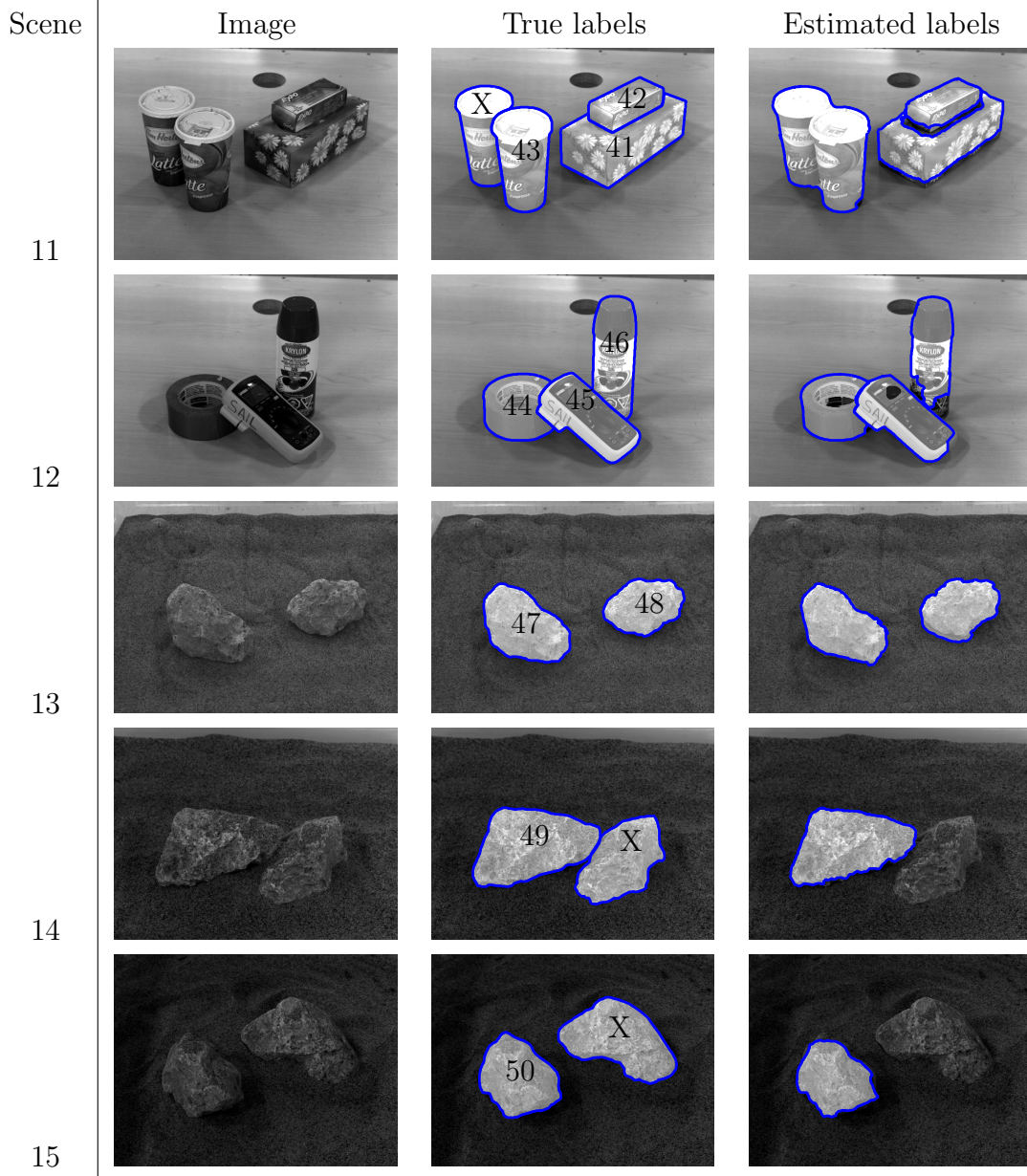


Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.

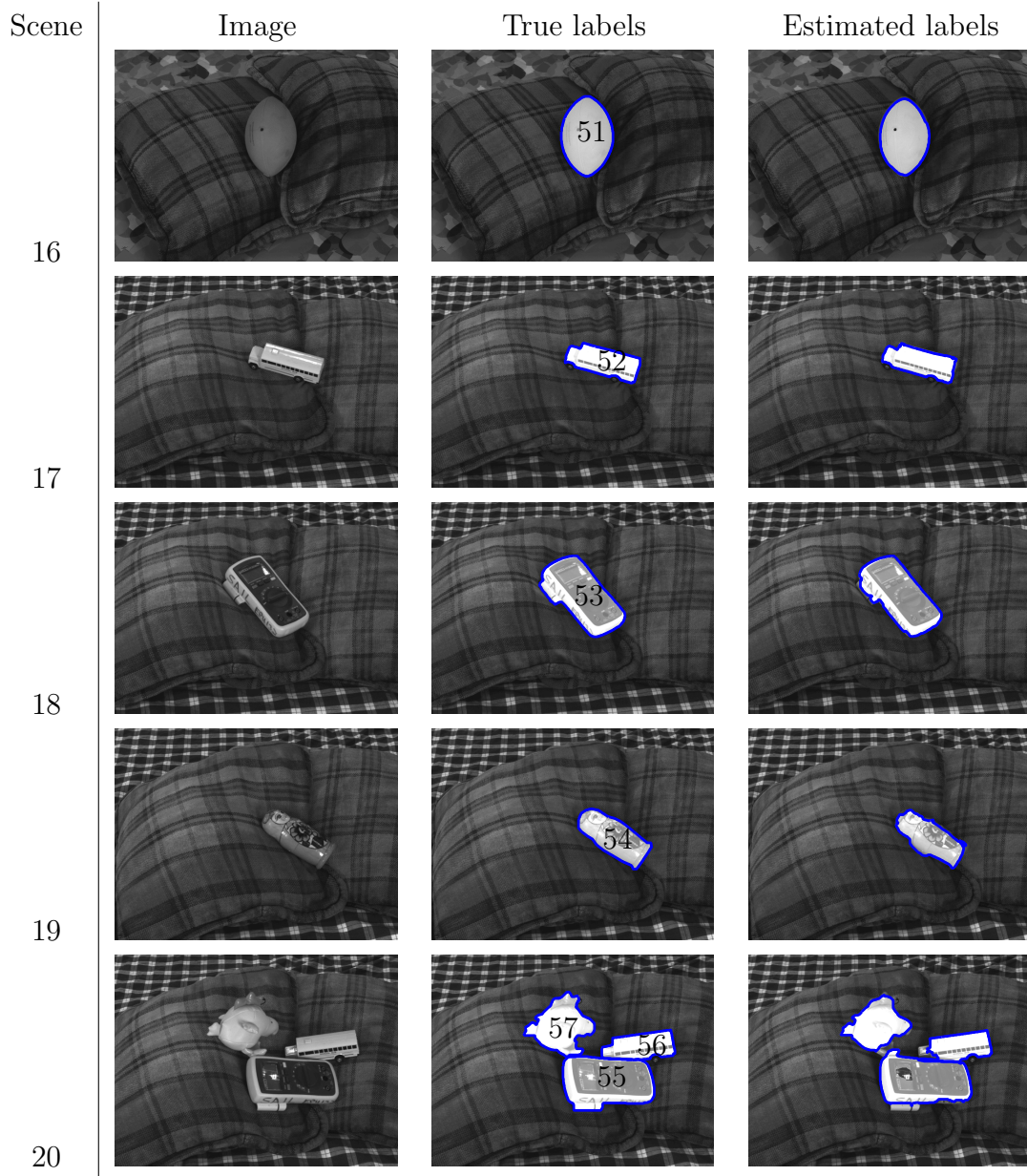


Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.

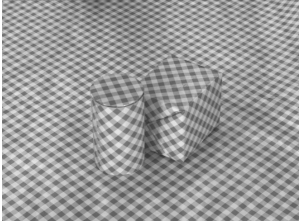
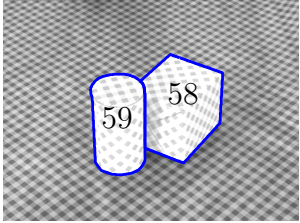
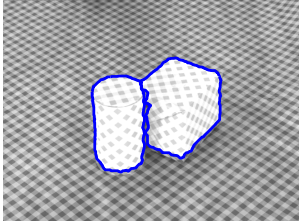
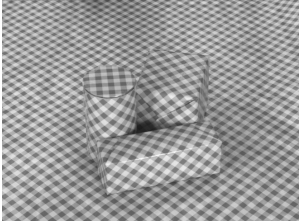
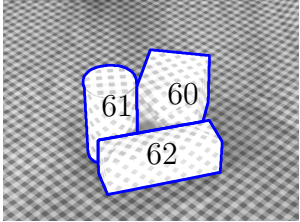
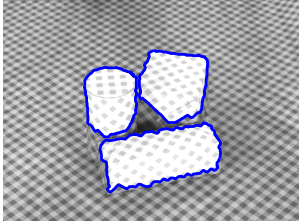
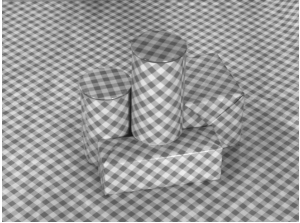
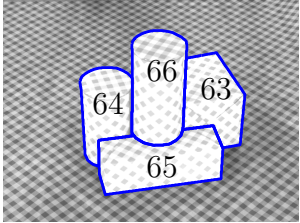
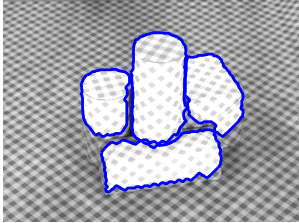




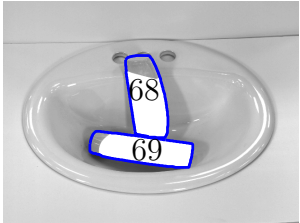
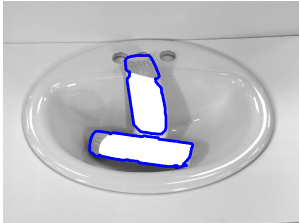
| Scene | Image   | True labels   | Estimated labels  |
|-------|---|---|---|
| 21    |    |    |    |
| 22    |    |    |    |
| 23    |   |   |   |
| 24    |  |  |  |
| 25    |  |  |  |

Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.




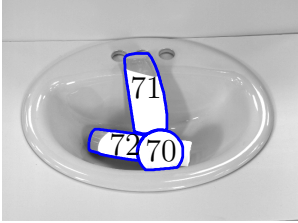
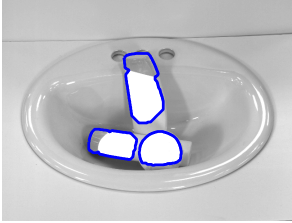

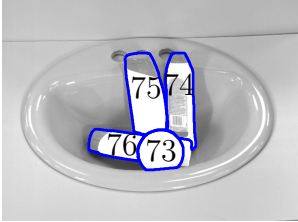
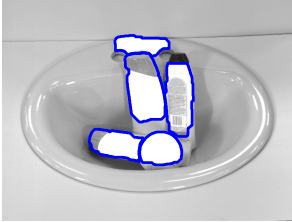

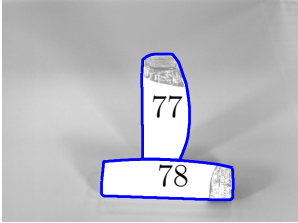
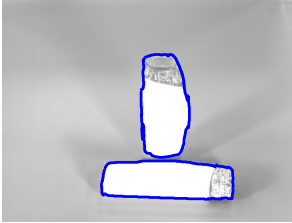

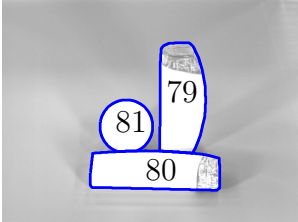
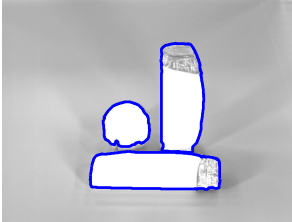
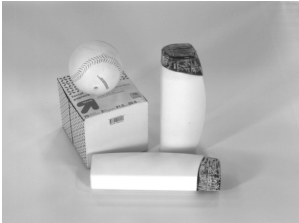
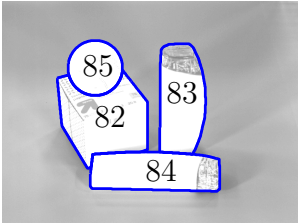
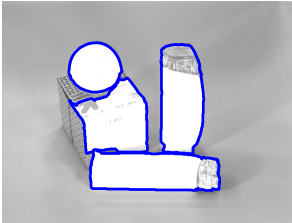
| Scene | Image   | True labels  | Estimated labels  |
|-------|---|--|---|
| 26    |    |    |    |
|       |    |    |    |
| 28    |   |   |   |
|       |  |  |  |
| 30    |  |  |  |
|       |   |  |   |

Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.

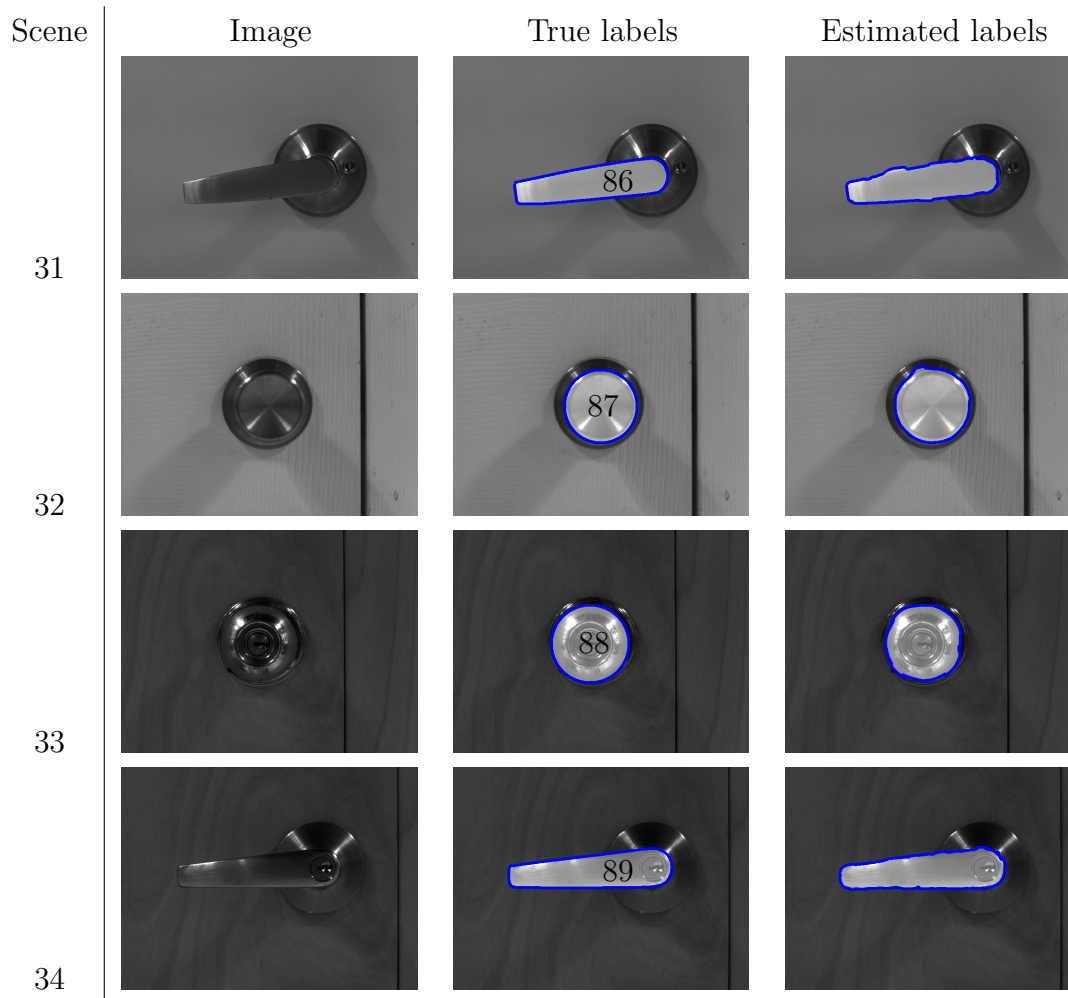


Figure 3.18: Scenes used to evaluate the performance of the proposed method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected. Column 3 shows results of applying the proposed segmentation method.

# Distance Metric Learning for Object Segmentation

---

Combining multiple appearance and geometric cues for segmentation has the potential to remove the ambiguity associated with individual cues, resulting in a more robust segmentation algorithm capable of functioning in complex environments. However, using multiple cues introduces new challenges. Depending on the appearance and geometry of the scene, certain cues are more useful for differentiating the object from the background, and for predicting the location of the object's boundaries. Additionally, when using appearance and geometric features jointly, correlation between these features becomes significant. While the various features are considered independent within object or background regions, a correlation between features is expected at object boundaries. For a segmentation algorithm to adapt effectively to the available data it needs the ability to select which combination of features are the most useful in a given situation.

Few of the current segmentation methods which combine appearance and geometry explicitly address the ranking of features. In this chapter, an improved segmentation algorithm is developed which incorporates distance metric learning to estimate the distance function between multiple appearance and geometric features.

The segmentation algorithm proposed here is based on the method presented in the previous chapter. A learned distance function is introduced into the *pairwise* smoothness term to weight the relative significance of various segmentation cues according to their ability to separate points that belong to the object seed region from the background.

As a secondary contribution, a geometric region model of the object is presented. The model is similar to the one proposed by Bjorkman and Kragic [2010a,b]. In their work,

Bjorkman and Kragic [2010a,b] modelled the object as an ellipsoid. The distribution of points belonging to the object was modelled as a normal distribution in 2.5D image–disparity space; object points were assumed to be normally distributed in space about a mean position. The same assumption is followed here. The object is assumed to be ellipsoidal in shape. However, the distribution of 3D points in space is modelled as a function of the points distances from an ellipsoid fitted to sample points from the object.

The rest of this chapter is organized as follows. Section 4.1 provides an overview of the segmentation algorithm. Section 4.2 reviews the terms in the segmentation energy function. Section 4.3 presents the elliptical object model used to incorporate 3D location information into the *unary* term of the energy function. Section 4.4 introduces distance metric learning, and describes how it is used to learn a distance function used in the evaluation the *pairwise* smoothness term of the segmentation energy function. Section 4.5 presents experimental validation of the improved segmentation method. Concluding remarks are given in Section 4.6.

## 4.1 Overview of the Improved Segmentation Algorithm

The improved segmentation algorithm operates in an iterative manner, similar to the method presented by Rother et al. [2004]. As an initial step, the grayscale image, and the depth map are used to calculate secondary features such as texture, curvature and normal vectors at each pixel.

The method is initialized by providing a single point on the image. The 3D location of this point is determined, and the object region is initialized to include all points within a small distance from the seed point. The background region is initialized by centring a large sphere on the location of the seed point. A set of points in a ring outside of this sphere is taken as a set of initial background seed points. An example of this is



#### 4.1. Overview of the Improved Segmentation Algorithm

---

shown in Figure 4.1(a) and Figure 4.1(b), where object and background seed regions (respectively) are shown highlighted in white. The algorithm then iteratively estimates the object and the background models, and learns a distance metric. The models and the distance metric are used to obtain a new estimate of the segmentation using graph cuts.

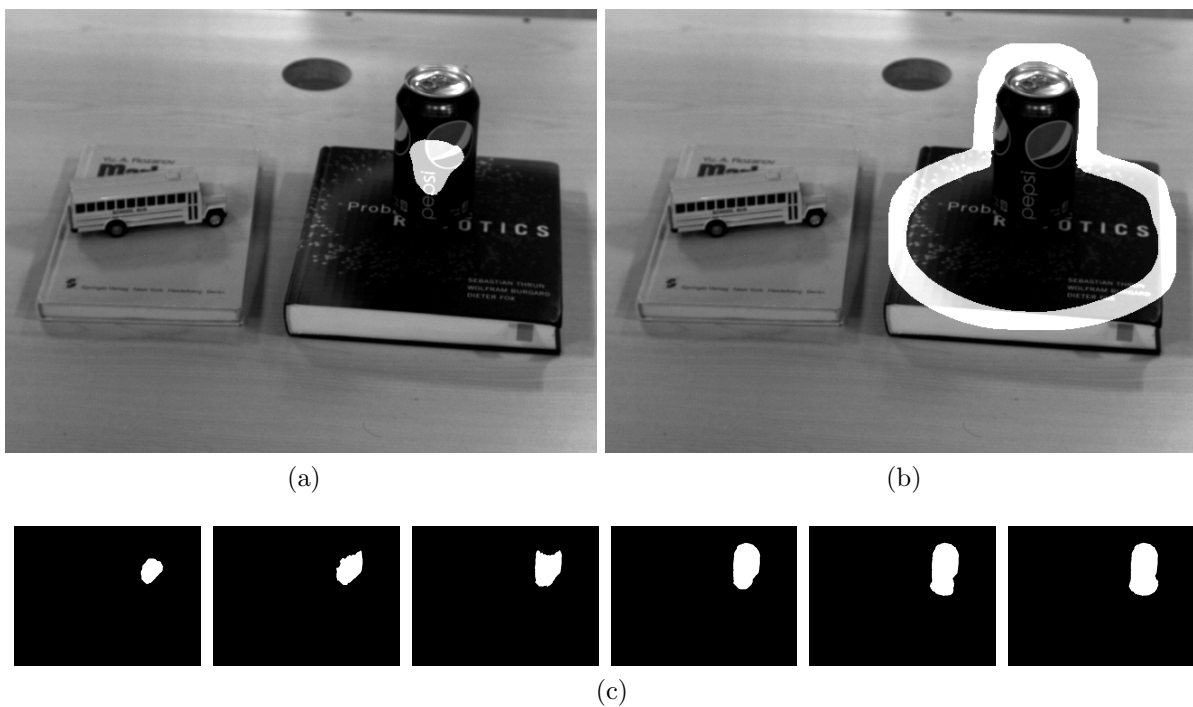


Figure 4.1: Initialized object (a), and background (b) regions. Segmentation results after several iterations of the algorithm are also shown (c).

The new estimates of the object and background regions are then passed as seeds to the next iteration of the segmentation algorithm. The process is repeated until the segmentation stops changing or until a set number of iterations is reached. Several iterations of the segmentation process are shown in Figure 4.1(c), with the object and background initial seed regions shown in Figure 4.1(a) and Figure 4.1(b), respectively.

## 4.2 Segmentation Energy Function

In Chapter 3, an energy function was defined the minimum of which corresponds to an optimal labelling of the image into foreground and background regions. This labelling is considered optimal with respect to the posterior probability of label assignments conditioned on the available image and geometric data. The energy function has the following form:

$$E(\mathbf{f}, \mathbf{d}) = \sum_{i \in \mathcal{S}} V_1(f_i, \mathbf{d}_i) + \frac{1}{T} \sum_{(i,j) \in \mathcal{N}} V_2(f_i, f_j, \mathbf{d}_i, \mathbf{d}_j) \quad (4.1)$$

where  $\mathbf{f} = \{f_1, \dots, f_i, \dots, f_{|\mathcal{S}|}\}$  and  $\mathbf{d} = \{\mathbf{d}_1, \dots, \mathbf{d}_i, \dots, \mathbf{d}_{|\mathcal{S}|}\}$  denote the sets of assigned labels, and observed data, respectively, for each site (pixel)  $i \in \mathcal{S}$ . The set  $\mathcal{S}$  contains all sites in the image, and the total number of sites is denoted as  $|\mathcal{S}|$ . The set  $\mathcal{N}$  represents the neighbourhood system, and contains pairs of sites that have direct influence over each other. The parameter  $T$  controls the relative weight between the terms  $V_1$  and  $V_2$ .

The energy function consists of two terms: the *unary* and *pairwise* potentials, corresponding to  $V_1$  and  $V_2$  respectively. The *unary* term measures the agreement between the data and the assigned labelling. The term is modelled as a negative log likelihood of a feature vector  $\mathbf{d}_i$  conditioned on the label at the site  $f_i$ :

$$V_1(f_i, \mathbf{d}_i) = \sum_{\mathbf{d}_i^n \in \mathbf{d}_i} -\ln P(\mathbf{d}_i^n | f_i) \quad (4.2)$$

where  $\mathbf{d}_i^n$  represents the  $n$ -th feature in the feature vector  $\mathbf{d}_i$

The available features include image intensity, texture, depth, and curvature. Instead of using depth directly, the likelihood of a given point belonging to the object is calculated as a function of its distance from an ellipsoidal object model. This model will be described

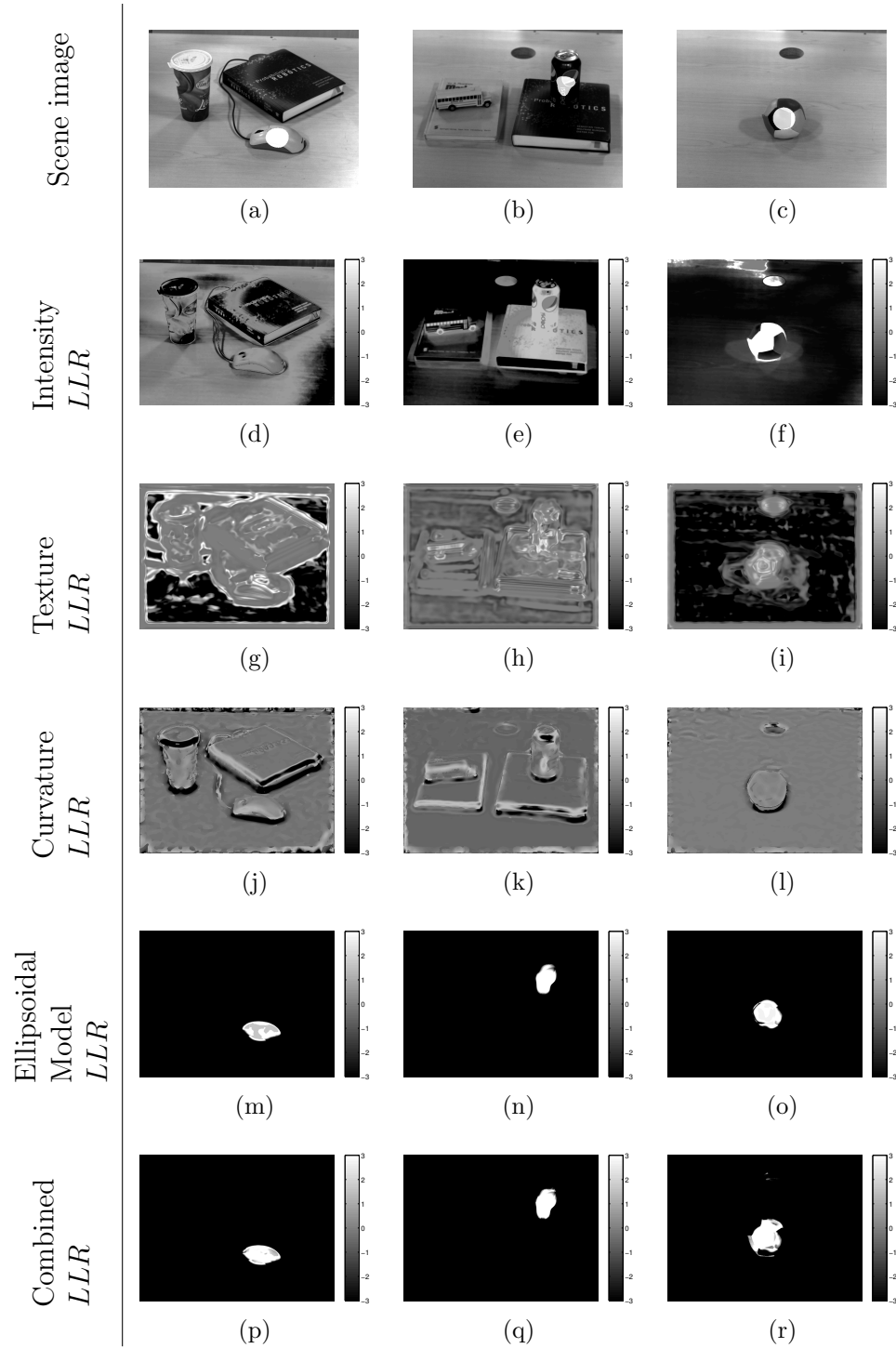


Figure 4.2: Log likelihood ratios of the feature models for the first iteration of the segmentation algorithm. (a)–(c) Objects being modelled. (d)–(f) Intensity models. (g)–(i) Texture models. (j)–(l) Curvature models. (m)–(o) Ellipsoidal models. (p)–(r) Log likelihood ratio of the combined models.

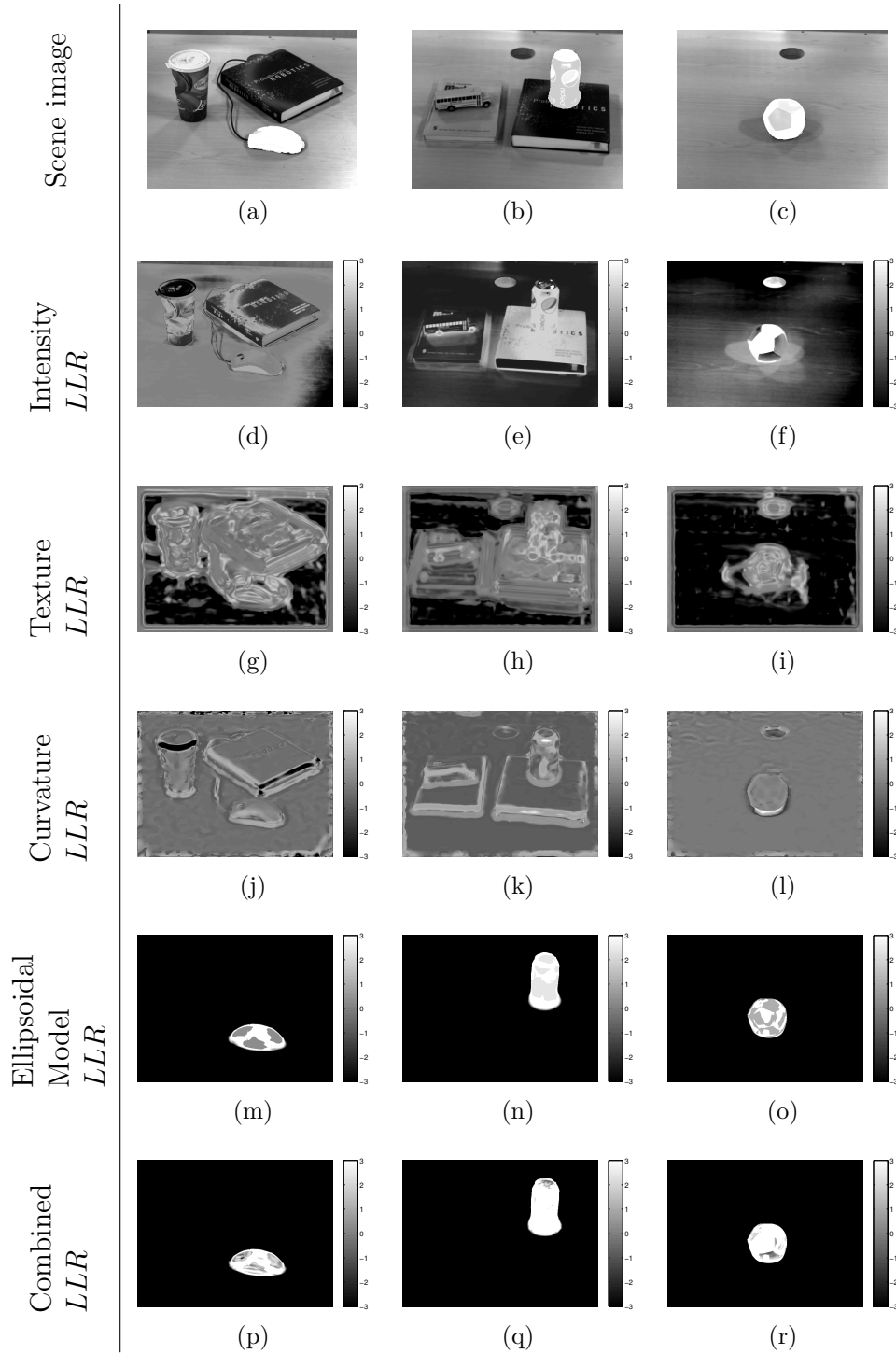


Figure 4.3: Log likelihood ratios of the feature models for the last iteration of the segmentation algorithm. (a)–(c) Objects being modelled. (d)–(f) Intensity models. (g)–(i) Texture models. (j)–(l) Curvature models. (m)–(o) Ellipsoidal models. (p)–(r) Log likelihood ratio of the combined models.

in more detail in Section 4.3. The probability density functions for all features are modelled using kernel density estimation (see Appendix B). For each feature, the kernel size is selected experimentally. Gaussian kernels are used for all features. Examples of the performance of the nonparametric feature models are shown in Figure 4.2, and Figure 4.3 for the first and last iterations of the segmentation method. The first row in each figure shows the region of the object from which the models were constructed. The remaining rows show the log likelihood ratios (LLRs) of the object and the background models:

$$LLR_i = \ln \left( \frac{P(\mathbf{d}_i^n | f = 0)}{P(\mathbf{d}_i^n | f = 1)} \right) \quad (4.3)$$

The LLR indicates whether a given site in the image  $i$  is more likely to have been generated by the object or the background model, corresponding to lighter and darker values in Figure 4.2 and Figure 4.3.

The *pairwise* term in Equation (4.1) acts as a smoothness constraint penalizing a change in labels between sites with similar data. It has the following form:

$$V_2(f_i, f_j, \mathbf{d}_i, \mathbf{d}_j) = \exp \left( -\frac{\text{dist}(\mathbf{d}_i, \mathbf{d}_j)^2}{\nu} \right) \cdot \frac{1}{|q_i - q_j|} \delta(f_i, f_j) \quad (4.4)$$

where  $\text{dist}(\mathbf{d}_i, \mathbf{d}_j)$  is a distance function which measures the similarity between two data vectors  $\mathbf{d}_i$  and  $\mathbf{d}_j$ . The parameter  $\nu$  is introduced to control the shape of the function. At larger values of  $\nu$ , smaller distances have little effect. At smaller values, the cost of separating two sites falls to zero quickly with an increase in distance. The term  $\delta(f_i, f_j)$  is an indicator function, and is one only if  $f_i \neq f_j$ , and zero otherwise. The fraction  $1/|q_i - q_j|$  accounts for the different image distances between horizontally, vertically, and diagonally connected sites, with  $q_i$  and  $q_j$  representing the pixels locations of sites  $i$  and  $j$  respectively. In Section 4.4, a DML method is applied to estimate the distance used in numerator of the exponential function Equation (4.4).

### 4.3 Ellipsoidal Object Model

Depth is not an effective cue to model object regions. When depth is used directly, large parts of the scene that are at the same distance from the camera as the object are incorrectly expected to belong to the object. If other cues are not sufficiently strong, the segmentation could spill into the background.

In the previous chapter, this issue was avoided by not using the depth cue in the region model entirely. Such an approach, however, does not utilize information regarding the potential location of the object in space. As a consequence, when multiple objects of a similar appearance and curvature are present, they are all considered equally likely to be the target object. This was previously counteracted by adding an extra constraint derived from the object detection step to focus the segmentation on the correct object.

An alternate approach involves constructing a model of the object which utilizes the full 3D position of points in space, or 2.5D position of points in depth-image or disparity-image space. This has the benefit of improving the algorithm’s ability to correctly identify points likely to belong to the target object, while reducing the spilling effect when the depth feature is used by itself. When such models are used, the object is commonly approximated as elliptical in shape. The likelihood of points in space is then modelled using a normal distribution. The likelihood of background points is modelled as either a uniform distribution, or based on their distance from the dominant plane in the scene [Bjorkman and Kragic, 2010a,b].

A similar approach is taken in this method. The object is assumed to be approximately elliptical in shape. However, the distribution of object points is modelled as a function of their algebraic distances to the ellipsoid fitted to sample object points. The sample points are initially obtained from the input to the method. Segmentation results from a previous iteration are used in the later stages. The model is constructed by fitting an ellipsoid to the current set of object seed points using a least squares fitting method [Li and Griffiths, 2004], details of which can be found in Appendix C. The 3D positions of

points in the scene are converted to algebraic distances. For an ellipsoid, the algebraic distance is:

$$F(\mathbf{v}, \mathbf{p}) = ax^2 + by^2 + cz^2 + 2fyz + 2gxz + 2hxy + 2px + 2qy + 2rz + d \quad (4.5)$$

where  $\mathbf{p} = [x, y, z]^T$  is a 3D point, and the vector  $\mathbf{v} = [a, b, c, f, g, h, p, q, r, d]^T$  contains the parameters of the ellipsoid obtained using the fitting procedure. The distribution of algebraic distances is then used to generate an object model.

The ellipsoid often expands past the object being segmented when the object is thin, or cylindrical. This can cause the ellipsoidal model to assign low likelihoods to points still inside the ellipsoid, but at some distance from the hull. To address this limitation, the likelihood of points inside the ellipsoid is modelled as a uniform distribution with respect to the range of algebraic distances inside the ellipsoid. A convention is enforced that points inside the ellipsoid should have a negative algebraic distance, and points outside should have a positive distance. Considering that the number of points that can belong to the object is significantly smaller than the number of points in the scene, the convention is enforced by reversing the sign of the algebraic distances if the number of points with a negative distance is greater than the number of points with a positive distance. The likelihood of a point outside of the ellipsoid is then modelled using a zero mean Gaussian distribution, the left half of which is replaced with a uniform distribution over the range of negative algebraic distance:

$$P(F(\mathbf{v}, \mathbf{p}_i) | f_i = 0) = \begin{cases} \frac{1}{\sqrt{2\pi}\sigma_{al}} \exp\left(\frac{-F(\mathbf{v}, \mathbf{p}_i)^2}{2\sigma_{al}^2}\right) & \text{if } F(\mathbf{v}, \mathbf{p}_i) \geq 0 \\ \frac{0.5}{|minF(\mathbf{v}, \mathbf{p})|} & \text{if } F(\mathbf{v}, \mathbf{p}_i) < 0 \end{cases} \quad (4.6)$$

where  $\mathbf{p}_i$  is any 3D point in the scene,  $\sigma_{al}$  is the standard deviation of algebraic distances for points from the object seed region outside of the ellipsoid, and  $minF(\mathbf{v}, \mathbf{p})$  is the largest negative algebraic distance.

The background is modelled as a uniform distribution over the range of algebraic distances considered.

$$P(F(\mathbf{v}, \mathbf{p}_i) | f_i = 1) = \frac{1}{\max F(\mathbf{v}, \mathbf{p}) - \min F(\mathbf{v}, \mathbf{p})} \quad (4.7)$$

where  $\max F(\mathbf{v}, \mathbf{p})$  is the largest positive algebraic distance.

## 4.4 Learning the *Pairwise* Potential

The pairwise term in the energy function (4.1) is used to enforce a smoothness constraint. It is intended to allow for sites to be easily separated where the image or geometric data indicates a likely presence of an object boundary while penalizing placing the boundary in homogeneous regions. When multiple features such as texture, depth, or surface normal are considered, the choice of a proper smoothness term becomes challenging. In different circumstances different segmentation cues may be better suited for localizing the object's boundary. The smoothness term needs to be able to adapt to the scene, choosing which cues will be considered, and which will be disregarded.

### 4.4.1 Distance Metric Learning

Given a training set of points where some are known to be from the same class, while other are from different classes, the goal of distance metric learning is to estimate a Mahalanobis distance function such that the distance between similar points is smaller than the distance between dissimilar points [Xing et al., 2002].

The Mahalanobis distance is parametrized by a positive semi definite matrix  $\mathbf{\Lambda} \succeq 0$ :



$$dist_{\Lambda}(\mathbf{d}_i, \mathbf{d}_j) = \sqrt{(\mathbf{d}_i - \mathbf{d}_j)^T \Lambda (\mathbf{d}_i - \mathbf{d}_j)} \quad (4.8)$$

where  $\mathbf{d}_i$  and  $\mathbf{d}_j$  are data vectors at sites  $i$  and  $j$  respectively.

When performing segmentation with a combination of appearance and geometric data, it is beneficial to incorporate similarity or dissimilarity measures that cannot be calculated by taking a difference of feature vectors at the two sites. For example, the angle between two normal vectors may be more informative than a direct difference of the two normal vectors. To allow for this, the problem is modified to learn a pseudo-distance:

$$dist'_{\Lambda}(\boldsymbol{\rho}_{ij}) = \sqrt{\boldsymbol{\rho}_{ij}^T \Lambda \boldsymbol{\rho}_{ij}} \quad (4.9)$$

where

$$\boldsymbol{\rho}_{ij} = [diff(\mathbf{a}_i, \mathbf{a}_j), diff(\boldsymbol{\tau}_i, \boldsymbol{\tau}_j), |z_i - z_j|, \theta(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j)]^T \quad (4.10)$$

The vector  $\boldsymbol{\rho}_{ij}$  contains a set of difference measures for available appearance and geometric features. The first term  $diff(\mathbf{a}_i, \mathbf{a}_j)$  corresponds to an appearance difference which could be a difference of intensities at sites  $i$  and  $j$ , a difference of colour vectors, or a specific colour difference measure. A texture difference is represented as  $diff(\boldsymbol{\tau}_i, \boldsymbol{\tau}_j)$ . The third term is a difference of depth values at sites  $i$  and  $j$ .

It is not possible to measure the curvature from two points at two different locations. Because of this, a different measure is used:

$$\theta(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j) = 1 - |\hat{\mathbf{n}}_i^T \hat{\mathbf{n}}_j| \quad (4.11)$$

where  $\hat{\mathbf{n}}_i$  and  $\hat{\mathbf{n}}_j$  are normal vectors at sites  $i$  and  $j$  respectively.

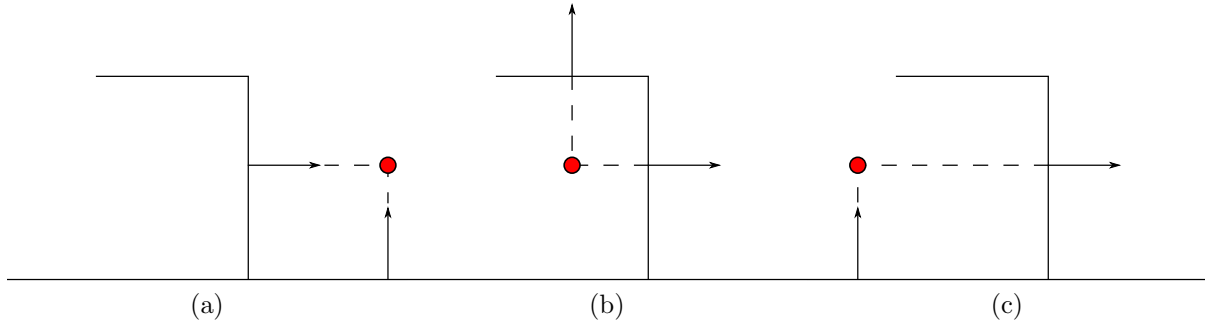


Figure 4.4: Determination of the type of junction between two surfaces. (a) A concave junction likely indicating a boundary between multiple objects, or an object and a surface. (b) A convex junction that is likely part of an internal edge on the object. (c) Undetermined type of junction.

It is important to note that how two surfaces meet is significant. Objects tend to be convex; they bend in onto themselves. A junction between an object and a supporting surface is likely to be concave. For this information to be taken into consideration, a means of determining how surfaces meet is needed. A simple method of determining how surfaces meet is based on inspecting the point of closest approach between the surface normals. A ray is formed for each site under consideration. The ray originates at the 3D point associated with the site, and travels in the direction of the estimated surface normal. If the closest point of approach between the two rays is in front of both 3D points the surfaces are expected to meet at a concave intersection (Figure 4.4(a)). If the point of closest approach is behind both points, a convex intersection is expected (Figure 4.4(b)). Otherwise, no determination about how the surfaces meet can be made (Figure 4.4(c)).

This set of conditions requires the learning of three positive semidefinite matrices  $\mathbf{A}^\alpha = \{\mathbf{A}^+, \mathbf{A}^-, \mathbf{A}^\times\}$ . The superscripts  $+$ ,  $-$ , and  $\times$  correspond to cases where the surfaces meet at a convex, concave, or undetermined intersections, respectively.

To estimate the set of positive semidefinite matrices  $\mathbf{A}^\alpha$ , a maximum margin formulation of the distance metric learning problem is followed [Nguyen and Guo, 2008; Weinberger et al., 2006; Weinberger and Saul, 2008, 2009].

Consider a labelled training set of triplets  $\mathcal{T} = \{(i, j, l) | f_j = f_i, f_l \neq f_i\}$ , where the pair of sites  $i$  and  $j$  are considered similar and share the same label (both object or background). A site  $l$  is selected from a different region. The pairs  $(i, j)^\alpha$  form the similarity constraints, while the pairs  $(i, l)^\alpha$  form the dissimilarity constraints. The superscript  $\alpha$  above each pair indicates how the surfaces associated with the two sites would meet. For each pair, a corresponding matrix  $\mathbf{\Lambda}^\alpha$  would be used to evaluate their distance. For instance, if a pair of sites  $i$  and  $j$  meet at a convex angle, the distance evaluation would be performed as:  $\boldsymbol{\rho}_{ij}^{+T} \mathbf{\Lambda}^+ \boldsymbol{\rho}_{ij}^+$ .

The goal of the learning process is to find a set of matrices  $\mathbf{\Lambda}^\alpha$  such that for each triplet, the squared distance between data vectors at sites  $i$  and  $l$  is greater than the squared distance between data vectors at sites  $i$  and  $j$  plus a unit margin. Formally this can be written as:

$$\forall (i, j, l) \in \mathcal{T} \quad \boldsymbol{\rho}_{il}^{\alpha T} \mathbf{\Lambda}^\alpha \boldsymbol{\rho}_{il}^\alpha \geq \boldsymbol{\rho}_{ij}^{\alpha T} \mathbf{\Lambda}^\alpha \boldsymbol{\rho}_{ij}^\alpha + 1 \quad (4.12)$$

This constraint can be expressed as the following optimization problem [Nguyen and Guo, 2008]:

$$\begin{aligned} \min_{\mathbf{\Lambda}^\alpha, \xi_{ijl}} \quad & \frac{1}{2} \|\mathbf{\Lambda}^\alpha\|_F^2 + \frac{\mu}{|\mathcal{T}|} \sum_{(i,j,l) \in \mathcal{T}} \xi_{ijl} \\ \text{s.t.} \quad & \boldsymbol{\rho}_{il}^{\alpha T} \mathbf{\Lambda}^\alpha \boldsymbol{\rho}_{il}^\alpha - \boldsymbol{\rho}_{ij}^{\alpha T} \mathbf{\Lambda}^\alpha \boldsymbol{\rho}_{ij}^\alpha \geq 1 - \xi_{ijl} \\ & \mathbf{\Lambda}^\alpha \succeq 0 \\ & \xi_{ijl} \geq 0 \end{aligned} \quad (4.13)$$

where  $|\mathcal{T}|$  indicates the number of triplets in  $\mathcal{T}$ . The first term in Equation (4.13) penalizes the size of each matrix in  $\mathbf{\Lambda}^\alpha$  proportional to its squared Frobenius norm,  $\|\cdot\|_F^2$ . The second term penalizes the violations of a margin constraint through the use

of a slack parameter  $\xi_{ijl}$ . The parameter  $\mu \geq 0$  is used to control to what extent the margin violations are penalized.

The optimization problem in Equation (4.13) can be solved as follows [Weinberger and Saul, 2008]. First, the term  $\boldsymbol{\rho}_{ij}^{\alpha T} \boldsymbol{\Lambda}^\alpha \boldsymbol{\rho}_{ij}^\alpha$  is re-written as  $\text{tr}(\mathbf{C}_{ij}^\alpha \boldsymbol{\Lambda}^\alpha)$ , where  $\mathbf{C}_{ij}^\alpha = \boldsymbol{\rho}_{ij}^\alpha \boldsymbol{\rho}_{ij}^{\alpha T}$ , and  $\text{tr}(\cdot)$  is the trace of a matrix. The slack variable  $\xi_{ijl}$  is replaced with a hinge loss, resulting in the following optimization problem:

$$\begin{aligned} \min_{\boldsymbol{\Lambda}^\alpha} \quad & \frac{1}{2} \|\boldsymbol{\Lambda}^\alpha\|_F^2 + \frac{\mu}{|\mathcal{T}|} \sum_{(i,j,l) \in \mathcal{T}} \max(0, [1 + \text{tr}(\mathbf{C}_{ij}^\alpha \boldsymbol{\Lambda}^\alpha) - \text{tr}(\mathbf{C}_{il}^\alpha \boldsymbol{\Lambda}^\alpha)]) \\ \text{s.t.} \quad & \boldsymbol{\Lambda}^\alpha \succeq 0 \end{aligned} \quad (4.14)$$

The solution is obtained iteratively. At each step  $t$ , the gradient of the objective function in Equation (4.14) is calculated with respect to a set of triplets  $(i, j, l) \in \mathcal{K} \subseteq \mathcal{T}$  that violate the margin constraint. A descent step is taken, and  $\boldsymbol{\Lambda}^\alpha$  is projected onto the set of positive semidefinite matrices. The gradient  $\mathbf{G}_t^\alpha$  of Equation (4.14) can be written as:

$$\mathbf{G}_t^\alpha = \boldsymbol{\Lambda}^\alpha + \frac{\mu}{|\mathcal{T}|} \sum_{(i,j,l) \in \mathcal{K}} (\mathbf{C}_{ij}^\alpha - \mathbf{C}_{il}^\alpha) \quad (4.15)$$

Each matrix in  $\boldsymbol{\Lambda}^\alpha$  is then updated by taking a gradient descent step with:

$$\boldsymbol{\Lambda}_{t+\frac{1}{2}}^\alpha = \boldsymbol{\Lambda}_t^\alpha - \eta \mathbf{G}_t^\alpha \quad (4.16)$$

where  $\eta$  is the step size. As a last step in the iteration, each of the  $\boldsymbol{\Lambda}_{t+\frac{1}{2}}^\alpha$  are projected onto the set of positive semidefinite matrices:

$$\boldsymbol{\Lambda}_{t+1}^\alpha = \sum_{k=1}^m \max(0, \lambda_k) \mathbf{e}_k \mathbf{e}_k^T \quad (4.17)$$

where  $\lambda_k$  and  $\mathbf{e}_k$  are the eigenvalues, and eigenvectors of  $\mathbf{\Lambda}_{t+\frac{1}{2}}^\alpha$ .

## 4.5 Experimental Results

To examine the performance of the improved segmentation method, it was tested on the set of images presented in Chapter 3. In each case, a point on a target object was provided to initialize the segmentation algorithm. The segmentation is then performed autonomously.

The scenes were captured with a Basler 601f camera with a resolution  $656 \times 491$  pixels, using a 16mm focal length lens. Dense depth estimation was performed using optical flow [Zach et al., 2007], with a random pattern projected onto the scene to improve the quality of the 3D reconstruction [Rusu et al., 2009a]. The parameters of the energy function were set to  $T = 1.4 \times 10^{-2}$  and  $\nu = 0.003$ .

All steps of the algorithm were executed on a computer with an Intel Q9550 processor, with 8 GB of memory. With the exception of the optical flow algorithm, and the graph cuts algorithm, all other the steps of the segmentation method were implemented in Matlab 8.1 (release R2013a). A summary of the computational performance of the method and it's constituent parts is shown in Table 4.1.

The use of normal vectors to estimate edge costs requires that sharp features are preserved. For this reason a voting based approach proposed by Boulch and Marlet [2012] is used to calculate normal vectors. In this method triangles are formed between a point of interest, and pairs of randomly selected points within a small sphere around the target point. Each triangle is used to calculate a normal vector. The calculated normals are converted into spherical coordinates, and are binned on a spherical grid. The bin with the largest number of votes is selected, and the normal vector is reconstructed from the average values of angles within that bin. Figure 4.5 shows the maximum normal vector difference as defined by  $\theta(\hat{\mathbf{n}}_i, \hat{\mathbf{n}}_j)$  calculated from normals estimated using

| Algorithm Step               | Time Required (s) |          |          |
|------------------------------|-------------------|----------|----------|
|                              | Mean              | $\sigma$ | Maximum  |
| Optical flow for 16 images   | 307.788           | 3.494    | 318.604  |
| Depth map calculation        | 523.302           | 5.452    | 534.497  |
| Point cloud construction     | 36.036            | 9.042    | 50.141   |
| Curvature/normal calculation | 330.490           | 9.658    | 357.569  |
| Texture features calculation | 27.568            | 2.131    | 31.128   |
| Intensity model construction | 0.655             | 0.018    | 0.703    |
| Texture model construction   | 3.246             | 0.327    | 3.614    |
| Curvature model construction | 0.341             | 0.008    | 0.355    |
| Ellipsoid model construction | 1.953             | 0.976    | 4.308    |
| Graph construction           | 1.135             | 0.003    | 1.138    |
| Distance metric learning     | 30.649            | 3.841    | 38.842   |
| Edge cost evaluation         | 60.475            | 3.703    | 67.415   |
| Graph cuts                   | 0.269             | 0.020    | 0.304    |
| Combined algorithm           | 1320.668          | 25.443   | 1382.211 |

Table 4.1: Computational performance of the improved segmentation algorithm.

the voting method (Figure 4.5(a)), and a method based on least squares plane fitting (Figure 4.5(b)). It can be seen that most sharp features are preserved in Figure 4.5(a) where the normals were calculated using the voting method [Boulch and Marlet, 2012]. When normal vectors are calculated using a least squares plane fit, sharp features are lost (Figure 4.5(b)).

Sets of similar and dissimilar points used to learn the distance are generated from the object and the background regions. The set of similar points is chosen as the set of neighbours in the image graph for which both vertices appear in the object seed region. For each pair of similar points, a dissimilar point is chosen from the background region. This point is selected radially outwards from the centroid of the object region. Figure 4.6

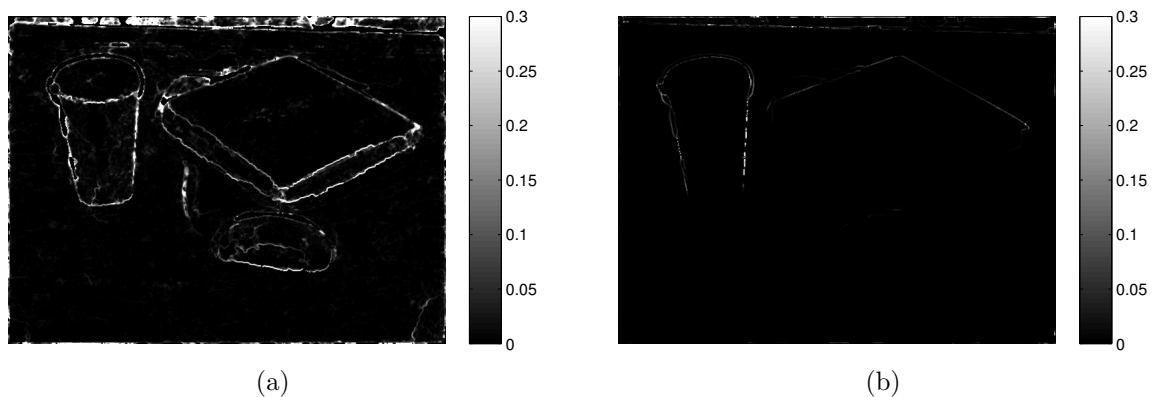


Figure 4.5: Normal vector differences calculated for vectors estimated using a voting algorithm (a), and a least squares plane fitting method (b).

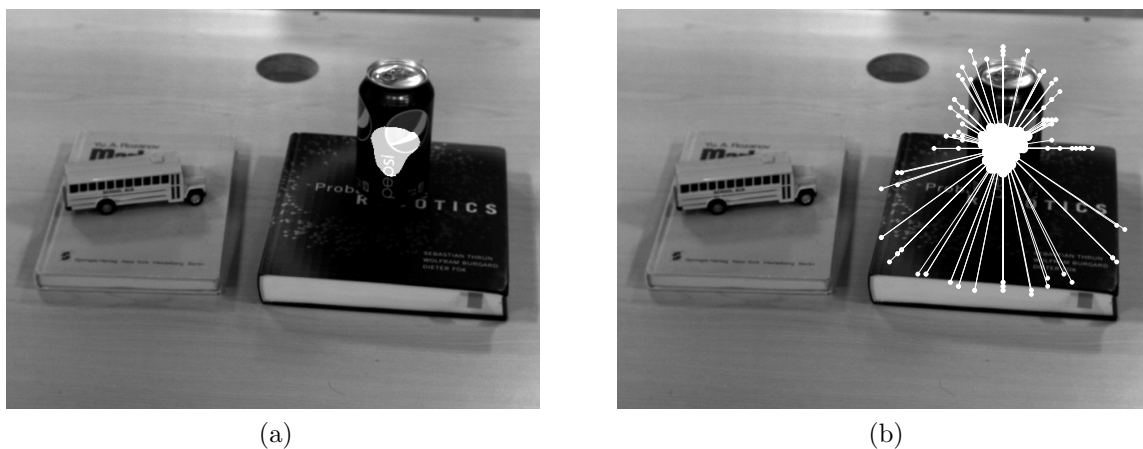


Figure 4.6: Selected pairs of similar and dissimilar points. Similar points are taken from the neighbours among the highlighted region (a). These points are paired with points in the background region to generate dissimilar points (b).

shows a subset of generated similar and dissimilar points. Similar pairs of points are taken from the region highlighted in Figure 4.6(a). A sample of dissimilar pairs of points is shown in Figure 4.6(b). Prior to the distance learning step, the feature differences are normalized such that the maximum difference of each feature in the training set is one.

Segmentation results for all of the tested scenes are shown in Figure 4.15. The figure shows the tested scene, along with the ground truth segmentations for each object, and

the results obtained using the improved segmentation method. Segmentation results for objects that were not detected with the geometric object detection method in the previous chapter are shown separately, in Figure 4.7.

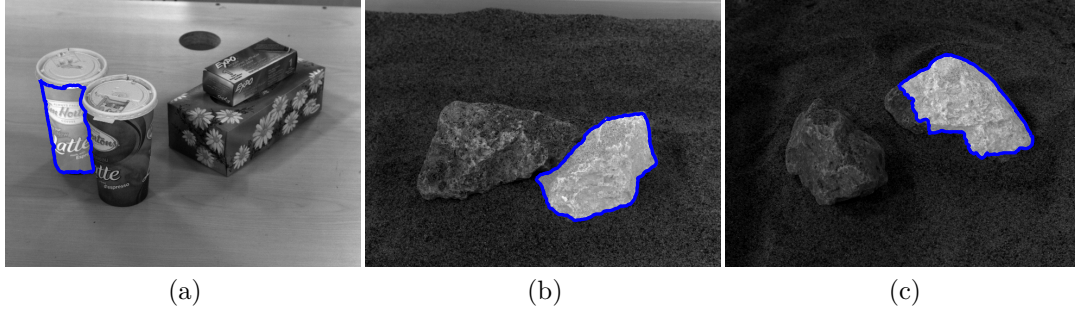


Figure 4.7: Segmentation results for objects which could not be detected using the geometric object detection method.

Figure 4.8 and Figure 4.9 present quantitative evaluation of the segmentation performance. The same metrics are used as in Chapter 3. They include the  $F_1$  score, precision, and recall measures (Figure 4.8). Mean and maximum distances from the detected boundary to the true object boundary are also shown (Figure 4.9). For comparison, the performance of the segmentation algorithm developed in the previous chapter is also presented.

Table 4.2 and Table 4.3 present summaries of segmentation performance in comparison to the algorithm developed in the previous chapter, as well as the benchmark segmentation method proposed by Bjorkman and Kragic [2010a,b]. Table 4.2 shows the means and standard deviations of the  $F_1$ , precision, and recall scores for the three tested algorithms. Table 4.3 shows the mean and standard deviation of the average and maximum distances between the detected and true object boundaries.

The results show that on average the improved segmentation algorithm exhibits slightly higher  $F_1$  score, and higher recall score than the method presented in Chapter 3. However, due to several instances where the method was not able to segment the object, the improved method performs worse in terms of mean and maximum distance from the



#### 4.5. Experimental Results

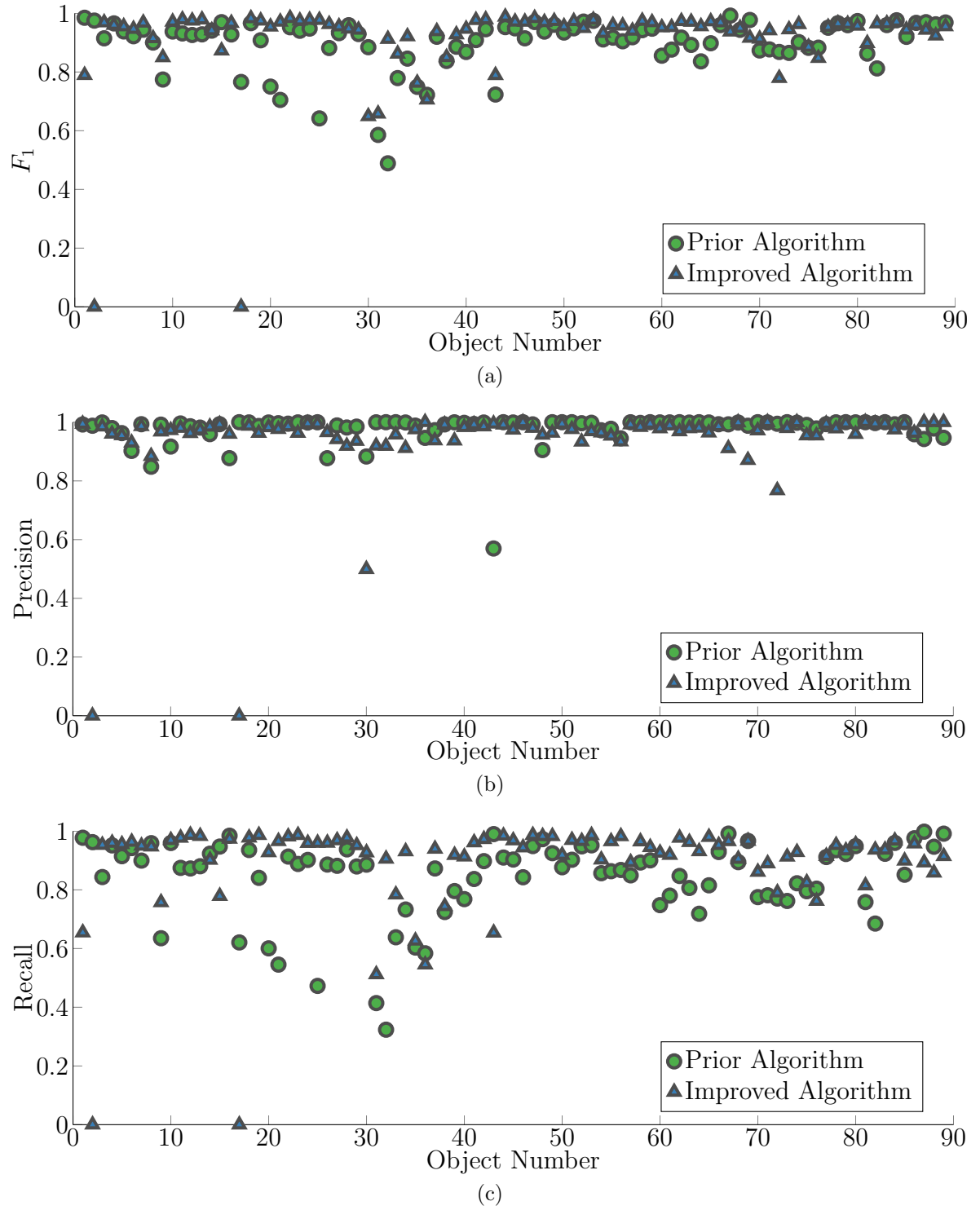


Figure 4.8: Results for the improved segmentation algorithm, showing the  $F_1$  (a), precision (b), and recall (c) scores. The same performance metrics for the segmentation algorithm developed in the previous chapter are shown.

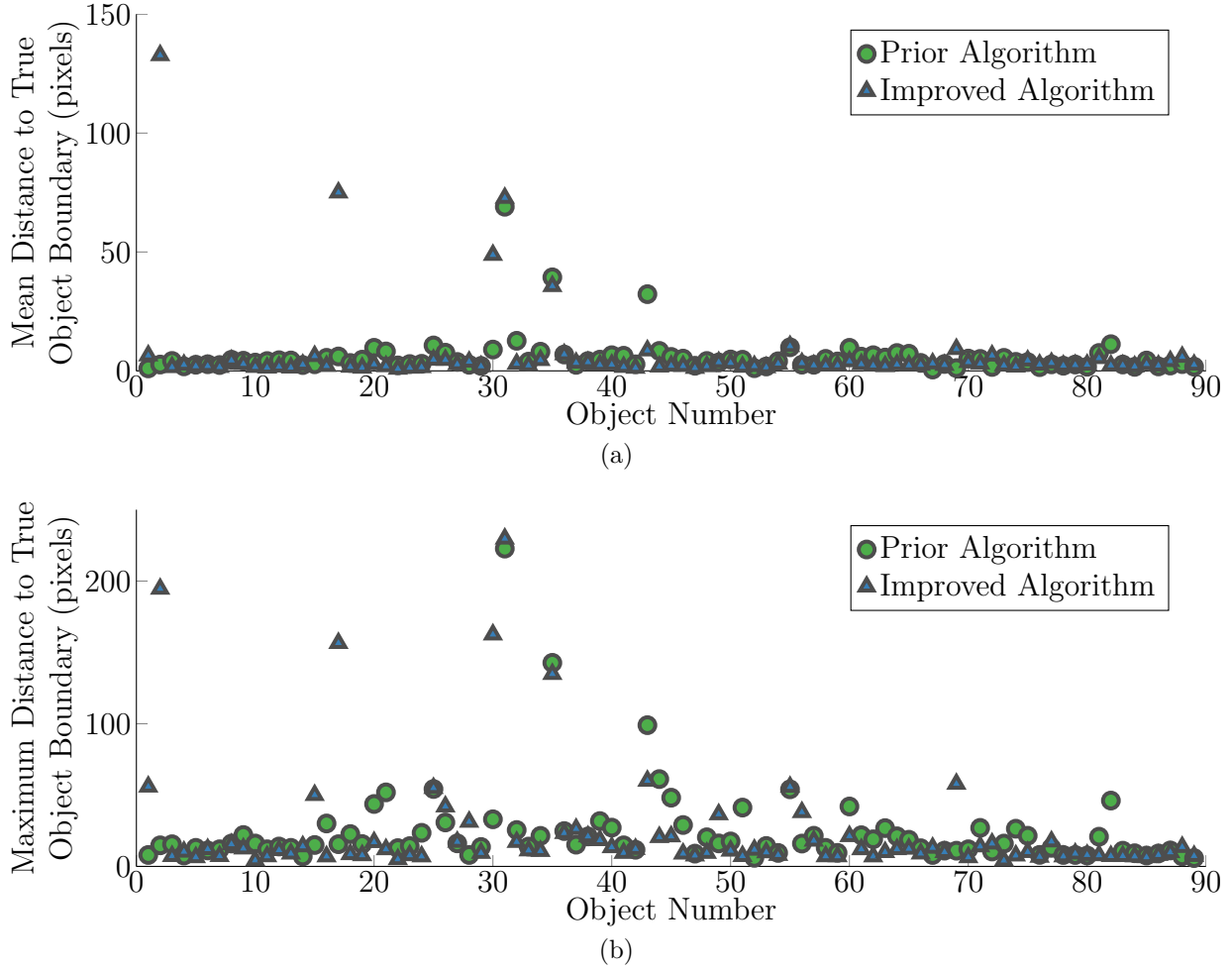


Figure 4.9: Results for the improved segmentation algorithm, showing the mean (a) and maximum (b) distance from the detected boundary to the true object boundary. The same metrics for the segmentation algorithm developed in the previous chapter are shown.

true object boundary. For the same reasons, the standard deviation of the performance metrics of the improved method is higher. In two instances the segmentation collapsed to zero area (Figure 4.11). If these two extreme cases are excluded from the performance evaluation, the improved method shows better performance on all metrics when compared against the method presented in Chapter 3; with the exception of precision. Similarly, when these two cases are excluded, the improved method presents with similar

#### 4.5. Experimental Results

variations of the performance scores. In comparison to the method of Bjorkman and Kragic [2010a,b], the improved algorithm exhibits overall better performance.

| Method                        | F1     |          | Precision |          | Recall |          |
|-------------------------------|--------|----------|-----------|----------|--------|----------|
|                               | Mean   | $\sigma$ | Mean      | $\sigma$ | Mean   | $\sigma$ |
| Improved method               | 0.9100 | 0.1549   | 0.9380    | 0.1576   | 0.8920 | 0.1667   |
| Improved method*              | 0.9309 | 0.0696   | 0.9596    | 0.0669   | 0.9125 | 0.0974   |
| Method presented in Chapter 3 | 0.8979 | 0.0896   | 0.9780    | 0.0540   | 0.8441 | 0.1331   |
| Bjorkman and Kragic [2010a,b] | 0.8367 | 0.1671   | 0.8564    | 0.2341   | 0.8753 | 0.1302   |

Table 4.2: Mean and standard deviation ( $\sigma$ ) of the  $F_1$ , precision, and recall scores for the improved segmentation algorithm, as well as the method proposed in Chapter 3 and the method of Bjorkman and Kragic [2010a,b]. Performance metrics obtained by excluding the two failed segmentations (Figure 4.11) from the analysis are indicated by \*.

When segmenting object with a thin structure, such as the tea cups (Figure 4.15, scene 6, object 20; scene 7, object 21 and scene 8, object 25), the improved method shows better performance when compared to the method presented in Chapter 3. Due to the use of the ellipsoidal object model, the method is able to accurately predict that the back surface of the cups belongs to the object region. This is in contrast to the previously presented method, which is unable to expand the object region to include the back surface of the cup.

When presented with multi-part, non-convex objects, such as the Stewie doll (Figure 4.15, scene 9, object 35), the improved method is still unable to segment the entire object. The parts composing the complex objects are individually convex. Additionally, the objects are placed in visually cluttered scenes, making it difficult for the method to use intensity or texture. In these cases, the method converges to a locally convex component of the larger object.

| Method                        | Mean Distance to True Object Boundary |          | Maximum Distance to True Object Boundary |          |
|-------------------------------|---------------------------------------|----------|--|----------|
|                               | Mean                                  | $\sigma$ | Mean                                     | $\sigma$ |
| Improved method               | 7.0818                                | 18.1381  | 25.2667                                  | 40.5549  |
| Improved method*              | 4.8571                                | 9.7187   | 21.8116                                  | 33.7214  |
| Method presented in Chapter 3 | 5.8074                                | 8.4696   | 23.7206                                  | 28.6687  |
| Bjorkman and Kragic [2010a,b] | 13.9709                               | 19.5428  | 46.7471                                  | 51.1294  |

Table 4.3: Mean and standard deviation ( $\sigma$ ) of the average and maximum distances to the true object boundary for the improved segmentation algorithm, as well as the method proposed in Chapter 3 and the method of Bjorkman and Kragic [2010a,b]. Performance metrics obtained by excluding the two failed segmentations (Figure 4.11) from the analysis are indicated by \*.

The method was also not able to fully segment the coffee cups (Figure 4.15, scene 1, object 1, and scene 11), and the large book (Figure 4.15, scene 5, object 15). Due to the intensity difference between the cups and the background table, and because the mean intensity of the top surface of the cups is much closer to that of the background, the method is not able to correctly segment the objects. In the case of the book in scene 5 (Figure 4.15, object 15) the segmentation was not able to expand past the high intensity edges provided by the white lettering on the book.

The method was not able to segment the books scene 1 and 5 (Figure 4.15, scene 1 object 2 and scene 5 object 17). These scenes are also shown in Figure 4.11, with the initial object seed regions, and background seed rings shown in Figure 4.11(a) and Figure 4.11(b). The regions were removed because the cost of separating the edges at the perimeter of the object seed region was higher than the cost of removing the region entirely. Following the learning step, no significant edges were detected near the object seed regions. Combined with the similarity in appearance between the regions from which the object and background are modelled, the cost of separating the edges surrounding

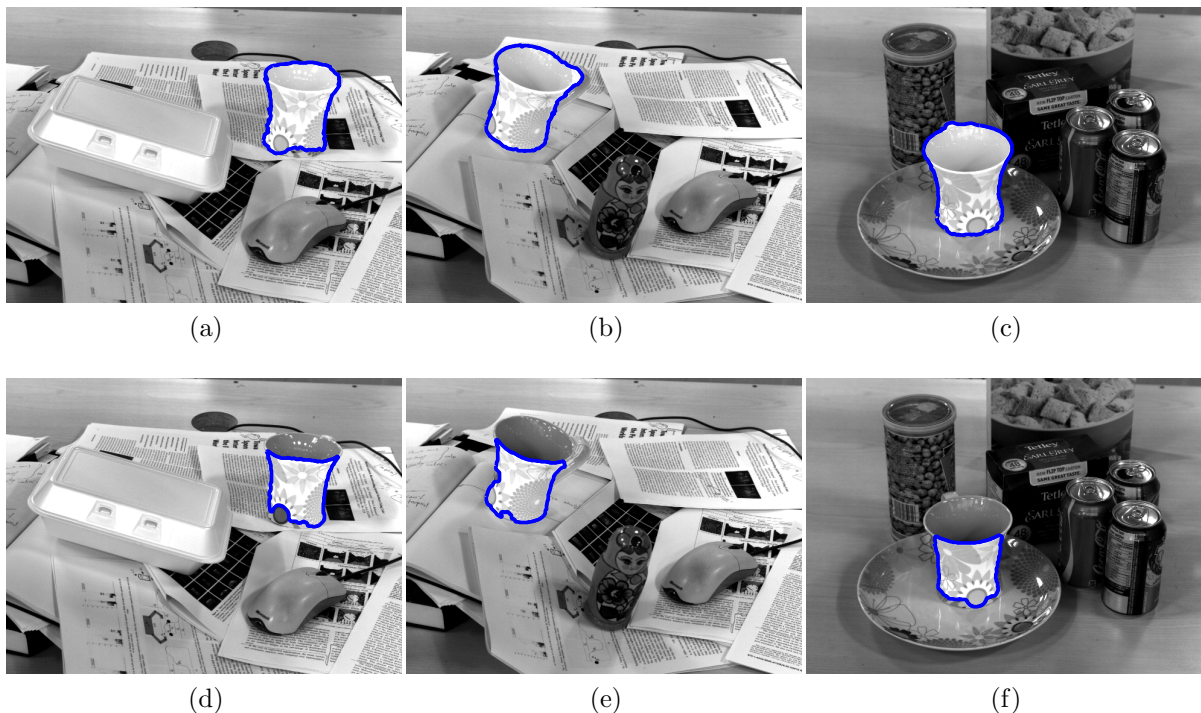


Figure 4.10: Comparison of segmentation results on objects with thin structure between the improved segmentation algorithm (a)–(c), and the method proposed in the previous chapter (d)–(f).

object region was higher than the cost of removing it.

Segmentation spilling into the background region was observed for a round plate (Figure 4.15, scene 8, object 30) and for a tissue box (Figure 4.15, scene 11, object 41). The results of segmentations for these scenes are additionally shown in Figure 4.12(a), and Figure 4.12(b) respectively.

The segment corresponding to the plate is observed to include the cup resting on top of the plate, as well as a part of the background. This occurs because the plate curves upwards near its edges. Over a number of iterations the ellipsoid fitted to the plate increases in volume, and expanded upwards. Combined with the similar appearance between the cup and the plate, this causes the cup to be assigned to the object region. The segmentation was not observed to converge, and continued to expand into the background as the

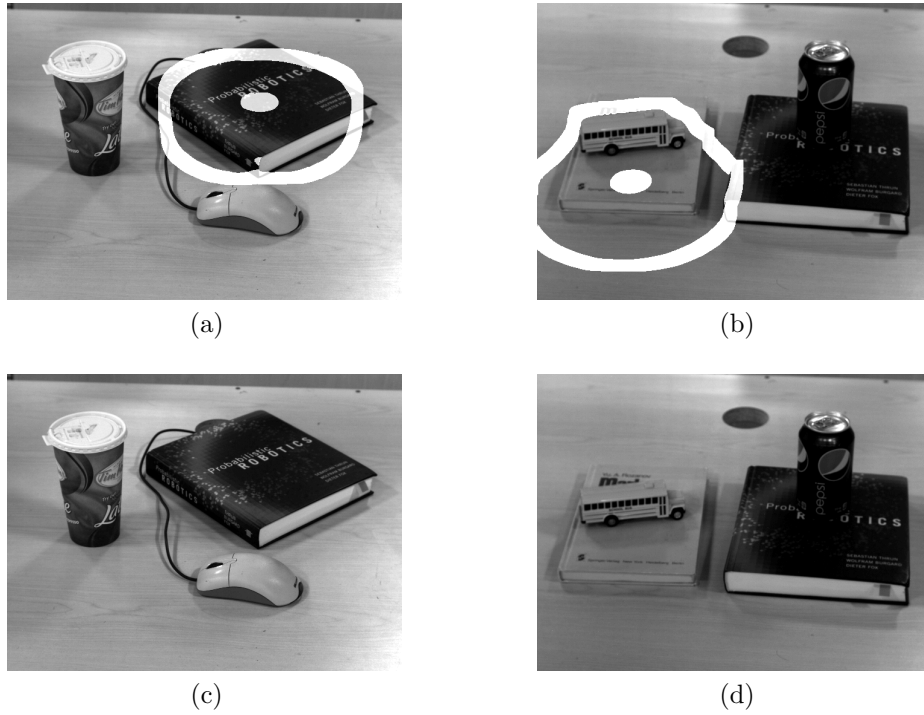


Figure 4.11: Examples of segmentation error due to shrinking of the object region. Initial object seed regions and background seed rings are shown in (a) and (b). No segmentation is obtained, which is shown with unlabelled images (c) and (d).

ellipsoid expanded further.

The object region corresponding to the tissue box in Figure 4.11(b) is observed to contain the eraser box resting on top of it. Due to the shape of the tissue box, the ellipsoid extends past the boundaries of the object, causing a part of the smaller box to be contained within the ellipsoid. The appearance of the two objects is very similar, and over several iterations parts of the smaller top object are assigned to the tissue box. The segmentation converges when both objects are segmented together. It should be noted that the method was able to successfully segment the smaller top object from the tissue box.

Examples of learned feature weights are shown in Figure 4.13, and Figure 4.14 for the pop can (Figure 4.15, scene 5, object 16) and the doll (Figure 4.15, scene 7, object 22) respectively. The top row shows feature weights estimated for the convex surfaces. The



Figure 4.12: Examples of segmentation error due to the object region expanding onto the background.

middle row shows feature weights associated with a concave surfaces, while the bottom row presents feature weights which are used when the surface type cannot be determined. The first column shows the feature weights for the first iteration of the algorithm, and the second column shows the weights for the last iteration. The first bar in each graph corresponds to the overall significance assigned to that feature type.

In both cases, for convex surfaces, depth is considered as a significant feature, in combination with intensity and texture. This is due to the fact that most dissimilar points of this type result from samples on the top of the object being paired with a distant portion of the background surface, while similar pairs come from internal object regions where depth does not change significantly.

Features for the unknown surface type are most often the result of pairings of points on the side of the object with distant background features. This results in a combination of surface normal and depth, combined with other features when they are informative as being considered most significant.

For concave surface junction types the normal vector difference feature is dominant. In the case of the doll, the learned distance favours a combination of normal difference and intensity. This is a desired result as the darker object is separated from the lighter table by both intensity and a difference in surface normal direction. For the pop can object, the normal difference is detected as the most significant feature. This is again desirable,

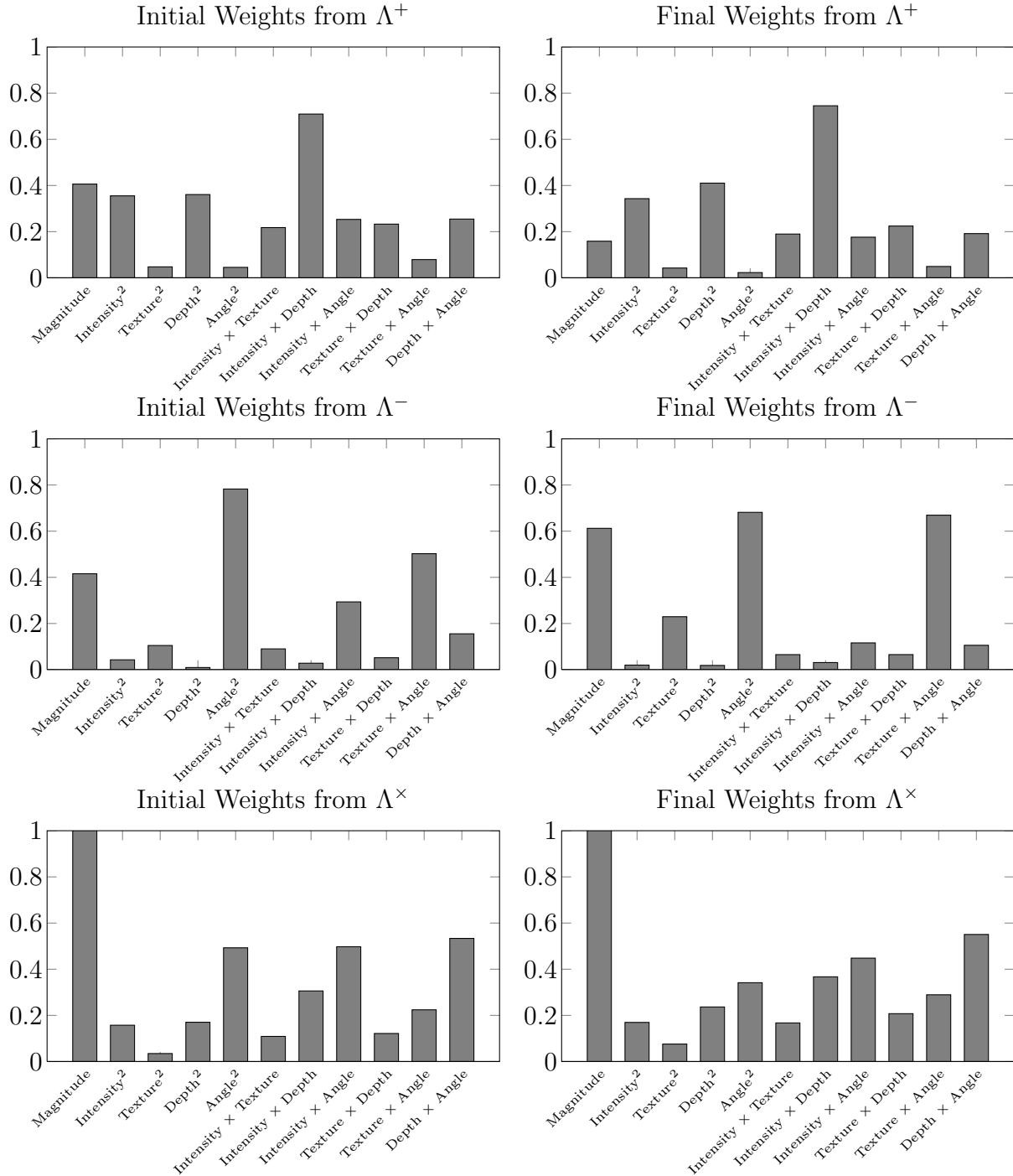


Figure 4.13: Learned feature weights from the first and the last iteration of the segmentation algorithm for object 16 in Figure 4.15, scene 5.



#### 4.5. Experimental Results

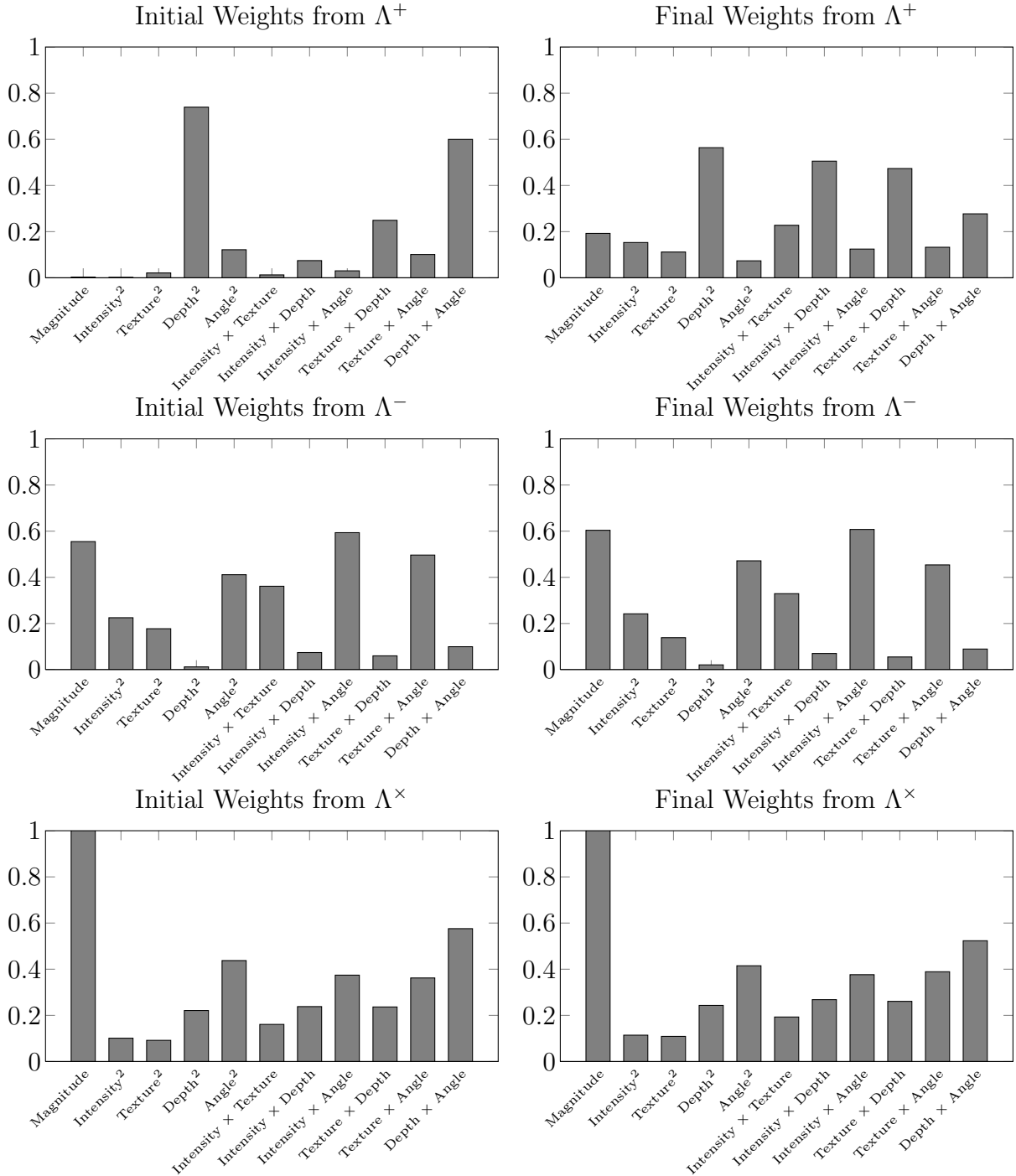


Figure 4.14: Learned feature weights from the first and the last iteration of the segmentation algorithm for object 22 in Figure 4.15, scene 7.

since the can and the book which supports it, share the same intensity but are separated by a change in surface normal orientation. Note that since both objects are convex, the magnitude of the convex surface is considered less significant. This indicates that the method assigns higher importance to feature differences across the unknown and concave surface junction types.

## 4.6 Conclusions and Discussion

This chapter presents a graph-based segmentation algorithm that incorporates a distance metric learning step used to estimate the relative significance of multiple appearance and 3D geometric cues. The learned distance is used in the smoothness term of the segmentation algorithm to improve its ability to accurately detect object boundaries, and increase the method's ability to adapt to each segmentation task. The segmentation is performed utilizing the graph cuts energy minimization method, while the energy function is derived based on the Conditional Random Fields framework. The effectiveness of the proposed segmentation method, and the effect of distance metric learning on the segmentation accuracy have been examined experimentally. When contrasted against the method developed in the previous chapter, mean improvement is observed in the recall score and  $F_1$  score, with a slight decline in other performance metrics. In two cases the method was unable to obtain a segmentation, and the object region was observed to collapse to zero area. These cases are shown in Figure 4.11. If the two extreme cases are removed from the performance analysis, the improved method is observed to perform better in all metrics except precision. The removal of these cases from evaluation also results in the standard deviation of all performance metrics in the improved method to appear similar to what was observed with the method presented in Chapter 3.

With a few exceptions discussed in the previous section, the method was able to accurately segment 82 of the 89 objects it was tested on, with a majority of segmentations being performed in cluttered and visually complex environments. Additionally, the

method does not rely on a sophisticated initialization or object detection scheme, and only requires a single point on the object as initialization.

In the improved method, an ellipsoidal object model is used. This allows the method to predict which parts of the scene should correspond to the object, and allows for a distinction between multiple similar objects in the scene. However, the model does not allow for segmentation of complex object that may consist of multiple non-convex parts. Additionally, the model can expand past the boundaries of the object and in certain circumstances can cause the segmentation to spill onto the background or other adjacent objects. An alternative model that could allow for the region belonging to the object to be identified, while avoiding the errors observed when using the ellipsoidal model would be desired.

Similarly to the method presented in Chapter 3, the improved algorithm is limited by its reliance on accurate dense 3D reconstruction of the scene. While it is not dependent on 3D information for object detection, the method does require accurate 3D reconstruction to perform segmentation in scenes that are visually complex. Without the means to deal with missing 3D data, or data of poor quality, the performance of the segmentation algorithm can degrade.

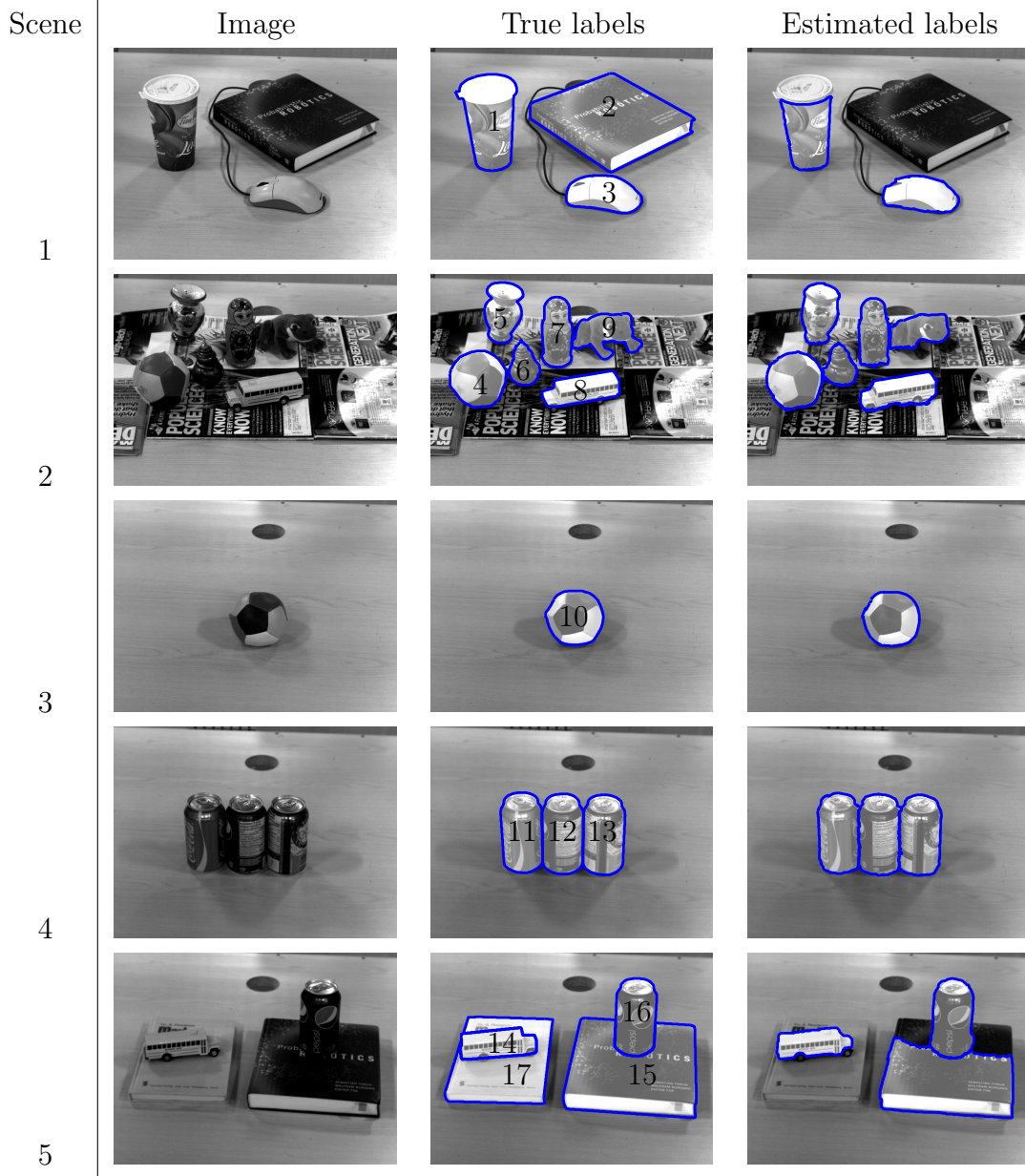


Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.

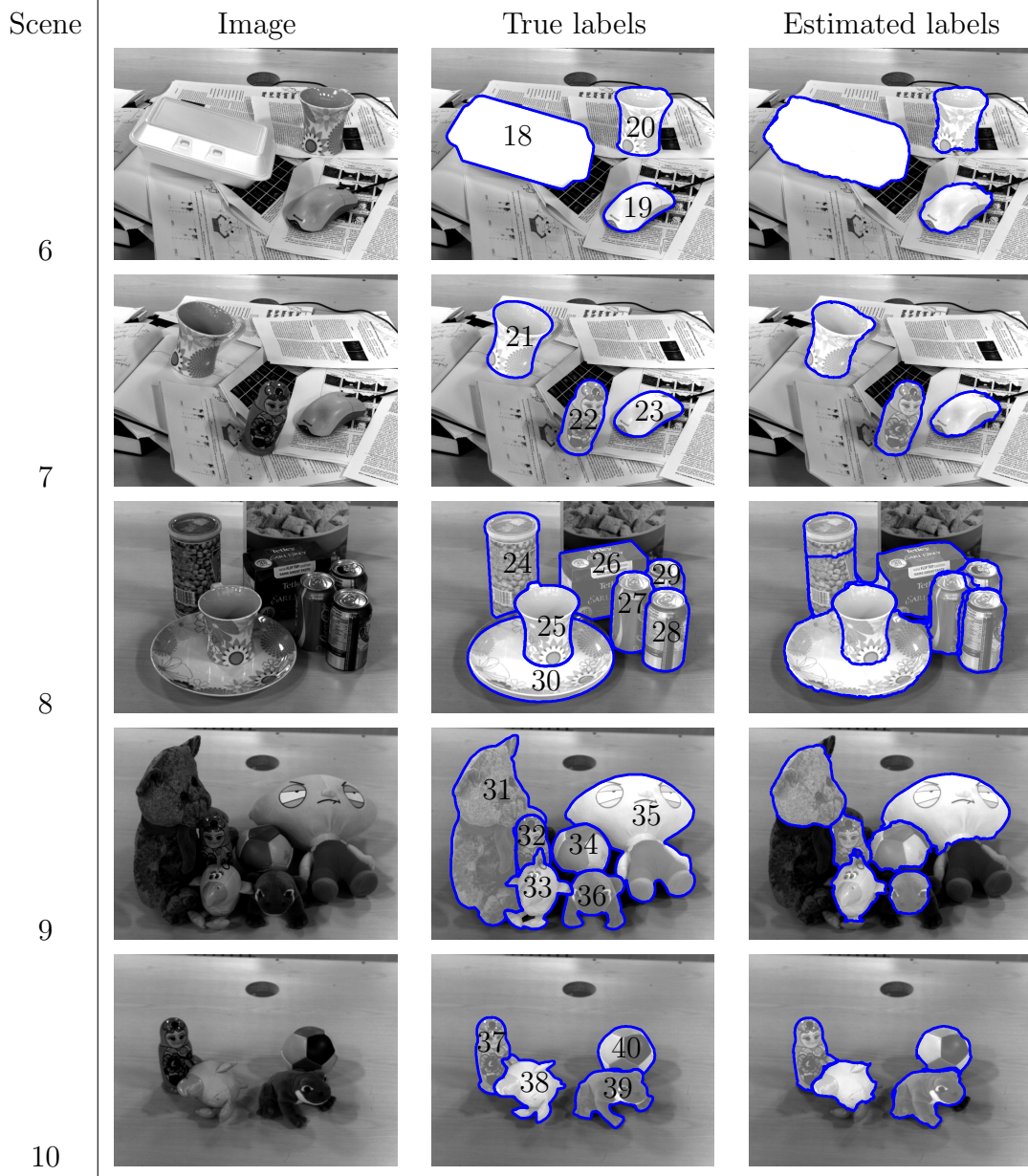


Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.

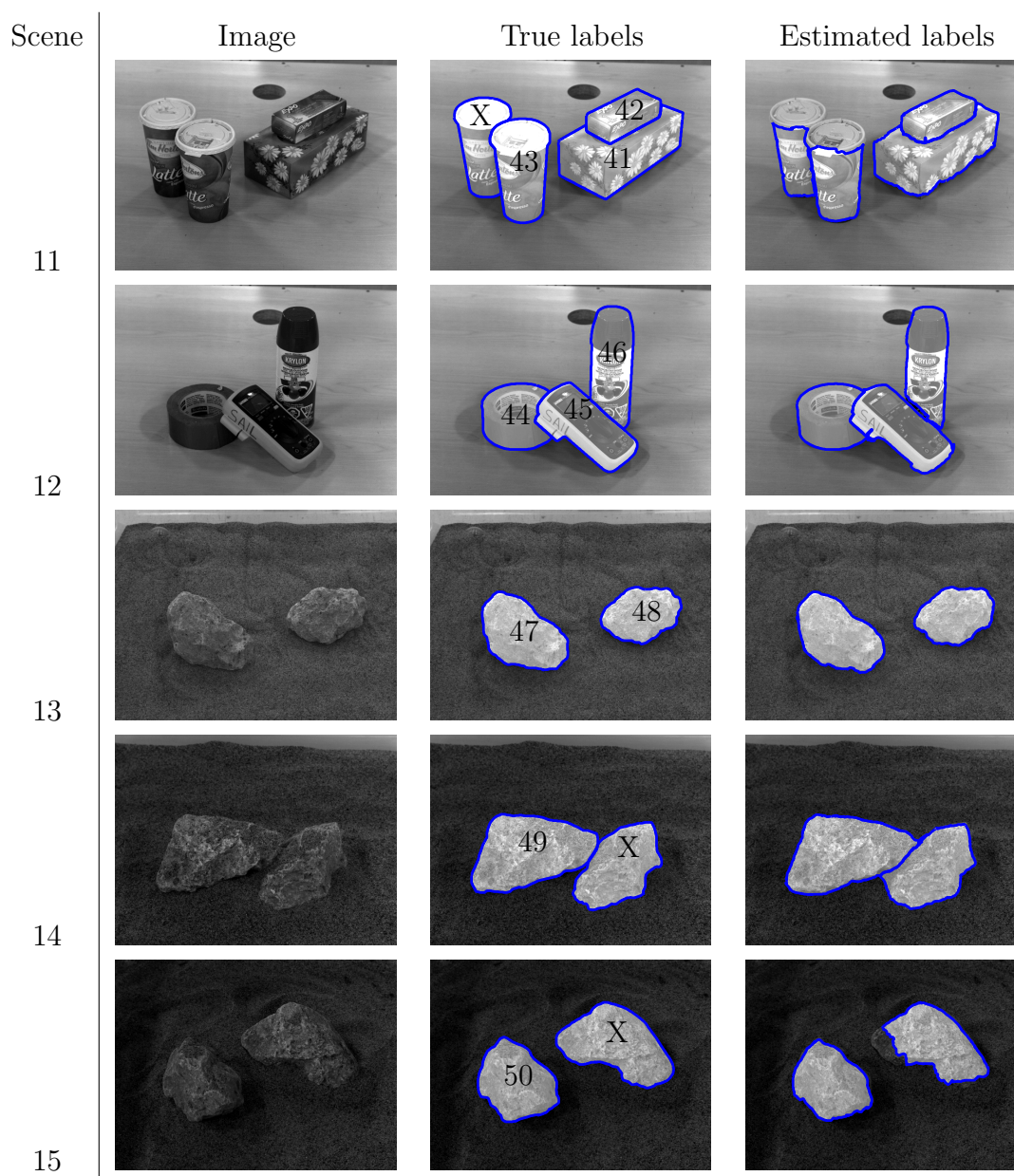


Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.








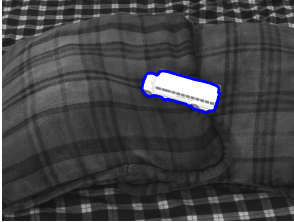


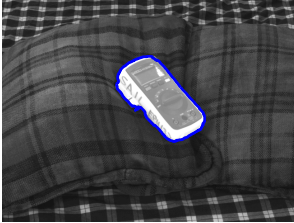




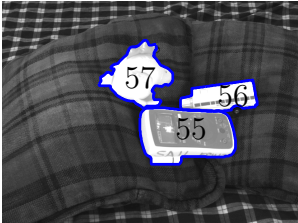

| Scene | Image   | True labels  | Estimated labels  |
|-------|---|--|---|
| 16    |    |    |    |
|       |    |    |    |
| 18    |   |   |   |
|       |  |  |  |
| 20    |  |  |  |

Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.

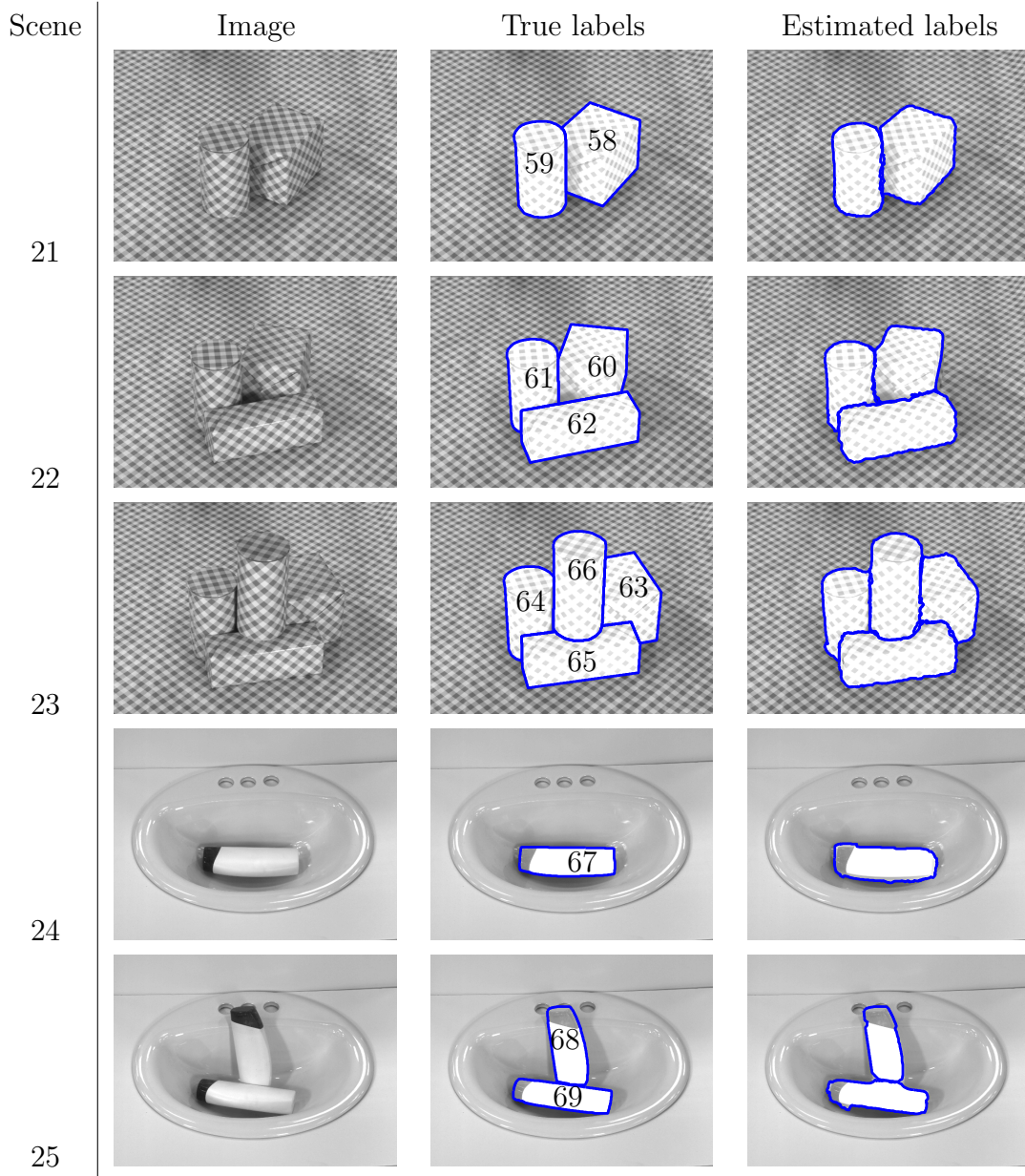


Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.




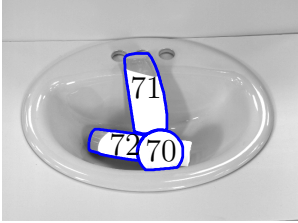
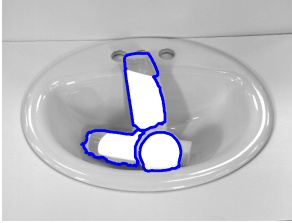

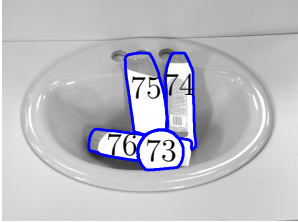
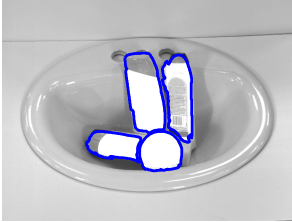

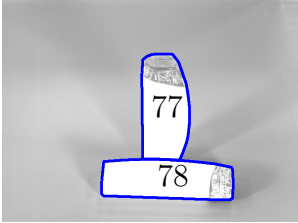
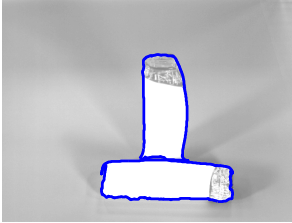

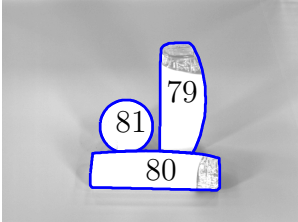
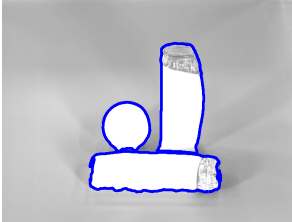
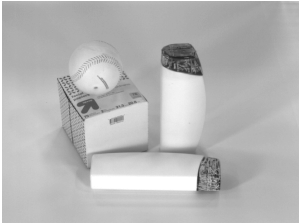
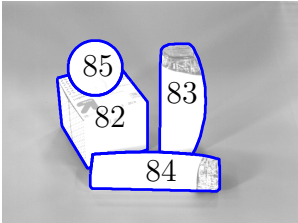
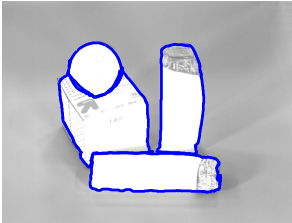
| Scene | Image   | True labels  | Estimated labels  |
|-------|---|--|---|
| 26    |    |    |    |
|       |    |    |    |
| 28    |   |   |   |
|       |  |  |  |
| 30    |  |  |  |

Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.







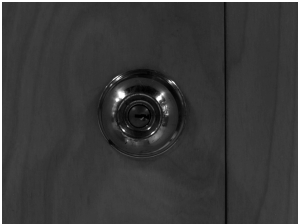
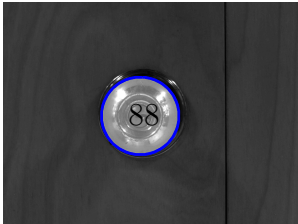
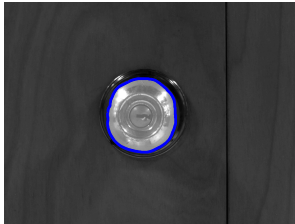

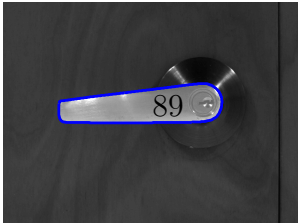
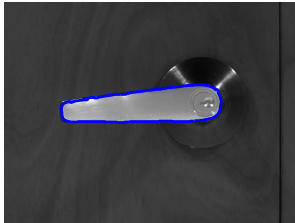
| Scene | Image   | True labels   | Estimated labels  |
|-------|---|---|---|
| 31    |    |    |    |
| 32    |    |    |    |
| 33    |   |   |   |
| 34    |  |  |  |

Figure 4.15: Scenes used to evaluate the performance of the improved segmentation method. Scene number is shown separately on the left. Column 1 shows an image of the test scene. Column 2 shows the ground truth labelling and object numbers, with an X indicating that the object was not detected by the method presented in Chapter 3. Column 3 shows results of applying the improved segmentation method.

# Conclusions

---

This final chapter presents a summary of the contributions of this thesis, and discusses possible directions for future research.

## 5.1 Summary of Contributions

### 3D Segmentation for Door Handle Localization

Based on the analysis of the geometric structure of door and handle environment, a 3D segmentation method has been proposed to allow for the segmentation and localization of a door handle of unknown geometry and appearance. The method functions by first detecting and segmenting the background elements of the scene such as the door and any adjacent walls or structures, leaving only the handle. The pose and geometry of the handle are estimated by fitting a bounding box to its principal components when projected onto the dominant plane in the scene. The bounding box can then be used to guide robotic grasping and manipulation of the handle. This method can be performed with a single, end effector mounted camera. By focusing on the removing the background elements in the scene, the proposed method does not require any knowledge of the handle's geometry, and by utilizing 3D data no appearance information of either the door or the handle is required.

The proposed algorithm was tested experimentally by performing door handle localization in scenes composed of a combination of six handles and 3 doors. Experimental results have demonstrated the effectiveness of the proposed method.

## Object Segmentation for Online Model Acquisition

Two object segmentation methods are developed to allow online partial object model acquisition. The object models are intended to be used to guide robotic grasping or manipulation tasks.

The segmentation problem is formulated as the Maximum a Posteriori estimation of pixel labels conditioned jointly on appearance and geometric data. The problem is formulated using the Conditional Random Fields framework, and an exact solution is obtained efficiently using the graph cuts energy minimization technique.

The first segmentation method utilizes a combination of intensity, texture, depth, and curvature. The use of depth information in the region term is avoided to eliminate segmentation errors in areas where the object contacts its supporting surface, or other objects. Initial object seeds are detected based on closed contours of depth edges and high curvature edges, allowing the segmentation algorithm to initialize appearance and geometric models without relying on user input. By utilizing appearance and geometric data jointly, the method is able to function effectively in situations where individual cues alone cannot accurately determine the regions or boundaries of the object. The effectiveness of the proposed method was verified by performing segmentations on a number of scenes of varying complexities.

A second, improved, segmentation algorithm is developed which incorporates a distance metric learning step to estimate the relative significance of available segmentation features. Using distance metric learning from pairs of similar and dissimilar points sampled from object and background seed points, the method learns the distance used in the smoothness term of the segmentation energy function. The segmentation is performed iteratively. The method alternates between learning the object and background models, learning a distance metric, and performing segmentation. Additionally, the improved segmentation algorithm incorporates an ellipsoidal object model to improve the methods ability to localize object regions. The likelihood of 3D points in space is modelled as a

function of a point’s algebraic distance from an ellipsoid fitted to the object seed points. This allows for better localization of the object, while avoiding the draw backs of modelling the distribution of points based on their distance from the camera. The method’s performance is tested experimentally on the same dataset as the method described above.

## 5.2 Future Work

### 3D Segmentation for Door Handle Localization

The presented method for door handle segmentation and localization was demonstrated to perform well in most circumstances. However, the performance of the algorithm is limited when presented with featureless, reflective or transparent surfaces. While dense depth estimation of reflective or transparent surfaces is very challenging, the algorithm’s performance when dealing with textureless surfaces, or reflective handles can be improved by incorporating sparse features such as corners, lines or conics in both the door plane detection and door handle localization phases of the algorithm. Incorporating the robot’s other available sensors in the detection of background elements could also assist in improving the performance of the method.

### Object Segmentation for Online Model Acquisition

The proposed segmentation algorithms were shown to be able to accurately segment objects in a variety of visually and geometrically complex environments. Despite this, a number of limitation are noted, which provide several avenues for future work. One limitation of the presented segmentation methods is their computational complexity. The presented methods require a dense 3D reconstruction of the observed scene, as well as the calculation of a number of secondary features such as texture and curvature.

Before segmentation is performed, both region models and distance functions have to be estimated, which leads to large amount of computation. Improving the computational performance of the presented methods is a recommended direction for future research.

Both of the presented segmentation methods rely on the availability of dense 3D data for segmentation. The first segmentation algorithm also relies on 3D information for object detection. The methods currently cannot function with missing 3D information. One direction for future work would involve enabling the segmentation algorithm to function with missing, or very noisy 3D data.

It is important to note that the problems of image filtering and restoration, and segmentation are related. In image restoration, it is common to enforce a smoothness constraints at all points of the image except where a discontinuity is present. As a consequence, the regions where this smoothness constraint is broken can be considered as a segmentation of the image. It should be possible to combine filtering and restoration (in particular of noisy or missing 3D data) and segmentation into a uniform framework.

It was emphasized by Rasolzadeh et al. [2010] that, in contrast to traditional image processing systems, a robot is an “*active*” observer. A robotic system is able to interact with the environment to obtain more information, and to utilize this information to learn and improve its performance in future tasks. With this in mind it may be a desired direction for future research to utilize the results of the manipulation process to provide information regarding the accuracy of the segmentation. This could allow the segmentation algorithm to improving its performance over time, and reduce the need for offline training with manually labelled images.

# Appendices





# Random Sample Consensus (RANSAC)

---

The Random Sample Consensus (RANSAC) algorithm is a method for fitting models to data containing a significant number of (potentially gross) outliers. The algorithm iterates over two main steps. First a model is fitted to a minimal subset of points randomly selected from the data. The subset of points is minimal in the sense that it is the smallest set of points needed to estimate the model’s parameters. An evaluation step then determines the “quality” of the model. In more detail, the steps of the RANSAC algorithm can be described as follows [Fischler and Bolles, 1981]:

1. Randomly select a minimal subset of data points.
2. Estimate the model using the randomly selected set of points.
3. Determine the number of inliers as points whose distance from the model is less than a threshold  $t$ .
4. If the number of inliers is greater than for any previous model, keep the current model, else disregard.
5. Repeat 1–4 for  $k$  iterations (optionally, terminate if the number of inliers is greater than some threshold  $d$ ).
6. Using the largest set of inlier points calculate the model using any desired model fitting method.

The threshold  $t$  for determining inliers will depend on both the model being thought, and the noise properties of the data. While it can be set analytically, it is also common to

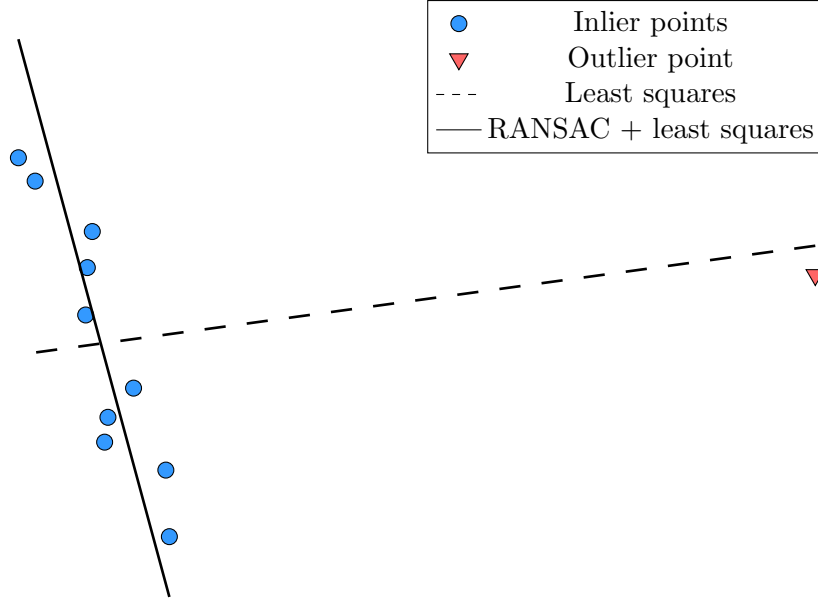


Figure A.1: Demonstration of the RANSAC algorithm.

set the threshold experimentally or heuristically based on knowledge about the problem at hand.

The number of iterations  $k$  can be set heuristically, or based on an expected number of inliers in the data and the desired precision. Assuming that the ratio of inliers  $w$  in the data is known (or can be estimated), and given a desired probability  $z$  that at least one iteration randomly sampled a set of points containing no outliers, the number of required iterations can be determined as [Fischler and Bolles, 1981]:

$$k = \frac{\ln(1 - z)}{\ln(1 - w^n)} \quad (\text{A.1})$$

A simple line fitting example demonstrating the RANSAC algorithm is shown in Figure A.1. A set of points forming a line with one gross outlier point is used as input to the RANSAC and least squares algorithms. Because RANSAC is able to ignore the

---

outlier point, it is able to correctly estimate the desired model parameters, whereas the least squares method fails to produce a useful result.



# Kernel Density Estimation and Mean Shift

---

Kernel density estimation is a nonparametric method for estimating the probability density function of a  $d$ -dimensional random variable. Consider a set of  $n$  discretely sampled points  $\{x_i\}_{i=1}^n$  from some probability density function in  $d$ -dimensions. The kernel density estimate of the underlying density function can be calculated as:

$$\hat{f}_K(x) = \frac{c_K}{nh^d} \sum_i^n k\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \quad (\text{B.1})$$

where  $K(x) = c_K k(\|x\|^2)$  is the kernel with a bandwidth  $h$ , and  $c_K$  is a constant ensuring that the kernel integrates to 1.

The mean shift algorithm is a non-parametric iterative mode seeking method. It is used to efficiently find peaks in high dimensional data and assign points to the corresponding clusters. The mean shift algorithm is derived from kernel density estimation methods [Comaniciu and Meer, 2002].

Taking the gradient of Equation (B.1) results in:

$$\nabla \hat{f}(x) = \frac{2c_K}{nh^{d+2}} \left[ \sum_i^n g\left(\left\|\frac{x - x_i}{h}\right\|^2\right) \right] \left[ \frac{\sum_i^n x_i g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)}{\sum_i^n g\left(\left\|\frac{x - x_i}{h}\right\|^2\right)} - x \right] \quad (\text{B.2})$$

where the first term on the right hand side of Equation (B.2) can be considered as a kernel density estimate using the kernel  $G(x) = c_G g(x)$ , with a kernel profile  $g(x) = -k'(x)$ . The second term of Equation (B.2) is called the *mean shift*:

$$m_{G,h}(x) = \frac{\sum_i^n x_i g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)}{\sum_i^n g\left(\left\|\frac{x-x_i}{h}\right\|^2\right)} - x \propto \frac{\nabla \hat{f}_K(x)}{\hat{f}_G(x)} \quad (\text{B.3})$$

where

$$\hat{f}_G(x) = \frac{c_G}{nh^d} \sum_i^n g\left(\left\|\frac{x-x_i}{h}\right\|^2\right) \quad (\text{B.4})$$

From Equation (B.3), it can be seen that the mean shift vector points in the direction of maximum ascent of the kernel density estimate  $\hat{f}_K(x)$ , and is modulated by the density estimate  $\hat{f}_G(x)$ . The mean shift vector, therefore, takes large steps in the direction of the gradient of  $\hat{f}_K(x)$  when the density estimate  $\hat{f}_G(x)$  is low, and takes smaller steps near the peaks of  $\hat{f}_G(x)$  [Comaniciu and Meer, 2002].

The mean shift procedure can now be summarized as follows:

1. Using a kernel  $G(x)$  with a bandwidth  $h$  calculate the mean shift vector  $m_{G,h}(x)$  at location  $x$ .
2. Set  $x = x + m_{G,h}(x)$ .
3. Repeat steps 1. and 2. until the mean shift vector is zero or is sufficiently small.

As an additional benefit, the mean shift algorithm can be used to assign each point to a cluster by tracking which local maximum a point is attracted to. The local maxima then become clusters, with points in their attraction basin being assigned to them.

---

The mean shift procedure is demonstrated in Figure B.1 using a set of points sampled from a 2-dimensional distribution with one mode. The mean shift algorithm easily converges to the mode of the distribution.

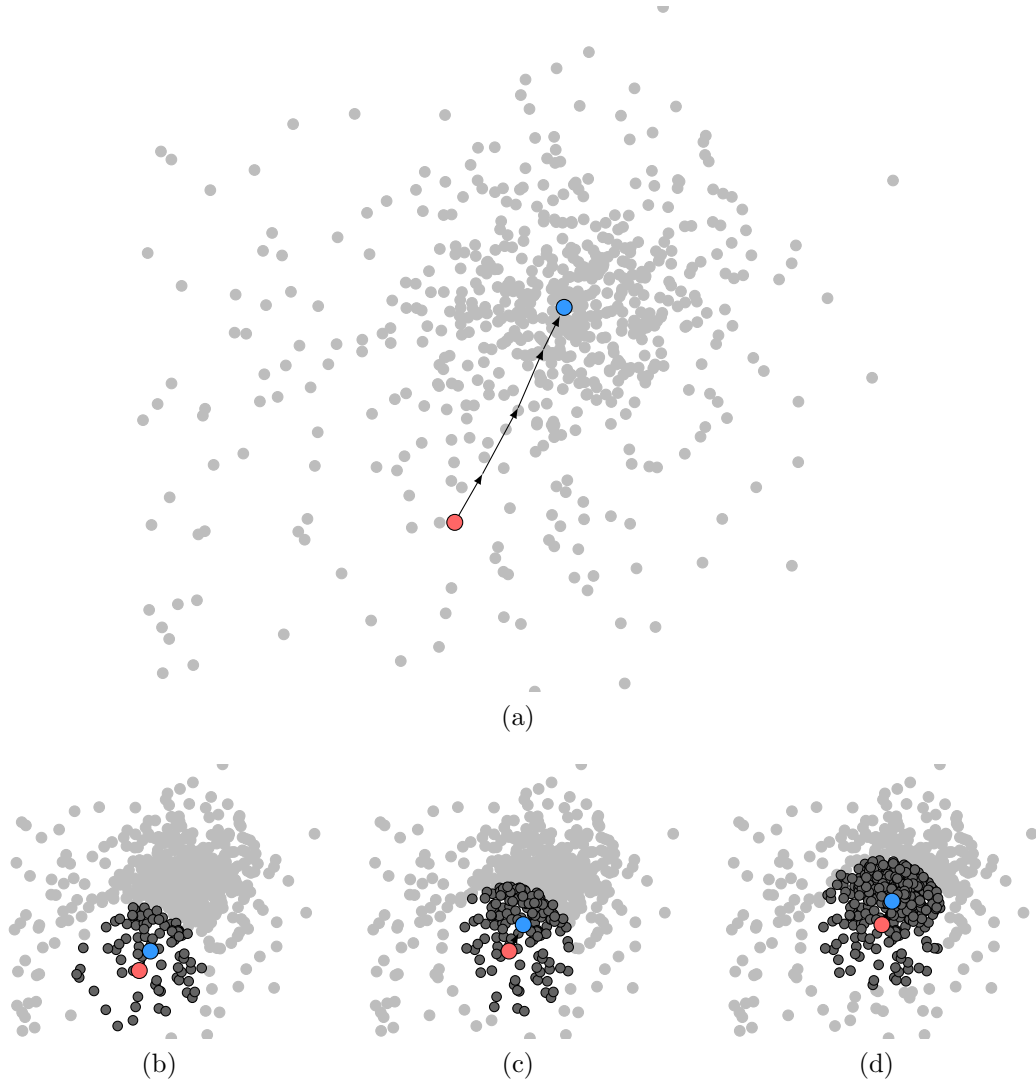


Figure B.1: Demonstration of the mean shift algorithm. (a) the path a point takes to the local maximum. (b)–(d) three iterations of the mean shift algorithm using a uniform kernel, showing the initial location (red) and the final locations (blue) at each iteration. The bandwidth of the kernel is indicated by the black circle. Dark gray points are used to calculate the mean shift vector at each iteration.





# Least Squares Ellipsoid Fitting

---

Let  $\mathbf{p} = [x, y, z]^T$  be any point in 3D space. An ellipsoid is a quadric surface which can be described by a second degree polynomial:

$$F(\mathbf{v}, \mathbf{p}) = ax^2 + by^2 + cz^2 + 2fyz + 2gxz + 2hxy + 2px + 2qy + 2rz + d = 0 \quad (\text{C.1})$$

where  $\mathbf{v} = [a, b, c, f, g, h, p, q, r, d]^T$  is a vector of parameters, and  $F(\mathbf{v}, \mathbf{p})$  is called an algebraic distance. Equation (C.1) can be written in matrix form as:

$$\mathbf{X}^T \mathbf{v} = 0 \quad (\text{C.2})$$

where  $\mathbf{X} = [x^2, y^2, z^2, 2yz, 2xz, 2xy, 2x, 2y, 2z, 1]^T$ .

To ensure that the quadratic surface represents an ellipsoid, constraints are introduced. Let:

$$I = a + b + c \quad (\text{C.3})$$

$$J = ab + bc + ac - f^2 - g^2 - h^2 \quad (\text{C.4})$$

$$K = \begin{vmatrix} a & h & g \\ h & b & f \\ g & f & c \end{vmatrix} \quad (\text{C.5})$$

A quadratic surface is an ellipsoid when:

$$J > 0, \quad I \times K > 0 \quad (\text{C.6})$$

It was shown by Li and Griffiths [2004], that a sufficient condition for the quadric surface to represent an ellipsoid is:

$$4J - I^2 > 0 \quad (\text{C.7})$$

Because the scale of the parameter vector can be chosen arbitrarily, the inequality can be re-written as:

$$4J - I^2 = 1 \quad (\text{C.8})$$

Ellipsoids satisfying the condition of Equation (C.8) are a subset of the ellipsoid family. For this reason Li and Griffiths [2004] propose a search based method. The constraint in Equation C.8 is replaced with:

$$kJ - I^2 = 1 \quad (\text{C.9})$$

where  $k \geq 4$ . A range of  $k \in [a, b]$ ,  $a \geq 4$ , is selected. The range  $[a, b]$  is searched for the solution to Equation (C.1) with largest value of  $k$ , that satisfies the constraints of Equation C.9 and Equation (C.6). Since the search is over a single variable, a good value for  $k$  can be found quickly. Alternatively,  $k$  can be set to  $k = 4$ , removing the need to perform a search, but resulting in a poor fit to flat, or elongated data [Li and Griffiths, 2004].

For a given value of  $k$  the ellipsoid fitting procedure is as follows. Let  $\{\mathbf{p}_i\}_{i=1}^m$  be the set of  $m$  points in 3D to which the ellipsoid is to be fitted. For each 3D point  $k$ , we can write:

$$\mathbf{X}_i = [x_i^2, y_i^2, z_i^2, 2y_iz_i, 2x_iz_i, 2x_iy_i, 2x_i, 2y_i, 2z_i, 1]^T \quad (\text{C.10})$$

A solution to Equation C.1 under the constraint of Equation (C.9) can then be obtained by minimizing the sum of squared algebraic distances between the quadric surface, and the set of data points:

$$\min \|\mathbf{D}\mathbf{v}\|^2 \quad s.t. \quad kJ - I^2 = 1 \quad (\text{C.11})$$

where  $\mathbf{D} = [\mathbf{X}_1^T, \mathbf{X}_2^T, \dots, \mathbf{X}_m^T]^T$ . To enforce the constraint in Equation (C.9), an additional matrix  $\mathbf{C}$  is defined as:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_1 & 0_{6 \times 4} \\ 0_{4 \times 6} & 0_{4 \times 4} \end{bmatrix} \quad (\text{C.12})$$

---

where

$$\mathbf{C}_1 = \begin{bmatrix} -1 & \frac{k}{2} - 1 & \frac{k}{2} - 1 & 0 & 0 & 0 \\ \frac{k}{2} - 1 & -1 & \frac{k}{2} - 1 & 0 & 0 & 0 \\ \frac{k}{2} - 1 & \frac{k}{2} - 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -k & 0 & 0 \\ 0 & 0 & 0 & 0 & -k & 0 \\ 0 & 0 & 0 & 0 & 0 & -k \end{bmatrix} \quad (\text{C.13})$$

The solution to the constrained linear least squares optimization can be obtained as the eigenvector vector associated with the unique positive eigenvalue of the system [Li and Griffiths, 2004]:

$$\mathbf{D}\mathbf{D}^T \mathbf{v} = \lambda \mathbf{C} \mathbf{v} \quad (\text{C.14})$$

Once obtained, the solution is checked against the constraints of Equation (C.6). If the constraint is not satisfied, the value of  $k$  is reduced by a step  $\delta k$ , and the procedure is repeated.



# Optical Flow

---

Optical flow can be considered as an approximation of the projection of 3D motion of points or surfaces onto the image plane of the camera [Horn 1986]. It is estimated from the apparent motion of intensity patterns in the observed images of the scene.

Optical flow is most often estimated under the assumption that the intensity of a point does not change when it is imaged from slightly different positions, this is called the brightness constancy constraint [Baker et al., 2011; Truccp and Verri, 1998]. Following the notation in chapter 2, let  $\mathbf{q} = (u, v)$  be a point on the image, and let  $\dot{\mathbf{q}} = (\dot{u}(u, v), \dot{v}(u, v))$  be the motion field at that point; the brightness constancy can then be expressed as:

$$I_0(\mathbf{q}) = I_1(\mathbf{q} + \dot{\mathbf{q}}) \quad (\text{D.1})$$

where  $I_0$  and  $I_1$  are images taken of the same scene at time  $t = 0$  and  $t = 1$  respectively.

The equation provides only one constraint for two unknowns at each pixel. To address this, a second constraint is introduced, often called the prior or smoothness constraint penalizing non smooth motion fields [Baker et al., 2011; Truccp and Verri, 1998]. Most existing methods combine the data and the smoothness terms in an optimization problem with a two term energy or function:

$$E = \int_{\Omega} \lambda E_{data} + E_{smooth} \quad (\text{D.2})$$

Where  $\Omega$  represents the set of image coordinates,  $E_{data}$  is the data term used to evaluate how well the solution agrees with the brightness constancy constraint, and  $E_{smooth}$  is used

to regularize the solutions. The parameter  $\lambda$  is used to control the relative significance of the data and the smoothness terms.

Together the data and the prior terms fully define the energy function that is minimized to obtain the optimal solution. For a more detailed summary and taxonomy of optical flow algorithms please see [Baker et al., 2011], and the references therein.

The optical flow algorithm used in this work is based on the method presented by Zach et al. [2007]. The method defines the data and smoothness terms which penalize the  $L1$  norm of the brightness constancy constraint violation, and the gradient of the estimated motion field over the image:

$$E = \int_{\Omega} \{ \lambda |I_0(\mathbf{q}) - I_1(\mathbf{q} + \dot{\mathbf{q}})| + | \nabla \dot{\mathbf{q}} | \} d\mathbf{q} \quad (\text{D.3})$$

where  $\nabla \dot{\mathbf{q}}$  is the gradient of the motion field  $\dot{\mathbf{q}}$ .

A coarse to fine approach for solving Equation (D.3) is described by Zach et al. [2007]. Additionally, a method for implementing the algorithm on a graphics card is presented. For details of implementation see [Zach et al., 2007].

To demonstrate the quality of 3D reconstruction obtained using the chosen optical flow algorithm, 3D reconstruction was performed on a set of three planar surfaces with a varying amount of visible texture. The surfaces include a plane with a checker pattern, a plane with a wooden texture, and a uniform white surface with no visible texture. To improve the quality of 3D reconstruction, a pattern was projected into the scene when acquiring images in Chapter 3 and Chapter 4. To demonstrate the effect this has on the quality of depth estimation, images with a projected pattern were also captured. Each surface was placed in two orientation, with either the left or the right side tilted away from the camera. Images were captured at 1cm increments, with the surfaces positioned approximately 1m in front of the camera. In total 17 images are captured for each scene. Optical flow is calculated between the central image, and each of the 16 displaced images.

---

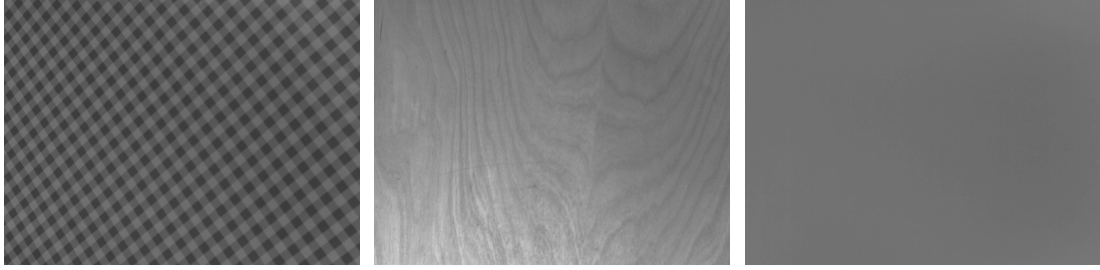
Reconstruction of a point cloud from the estimated optical flow follows the procedure described in Section 2.2 of Chapter 2. The depth maps are stacked into a single matrix, and combined by using a median filter with a  $5 \times 5 \times 16$  window size.

Sample images for the tested surfaces are shown in Figure D.1(a), with corresponding filtered depth maps, and reconstructed point cloud shown in Figure D.1(b) and Figure D.1(c), respectively.

Sample results for surfaces imaged with a projected pattern are shown in Figure D.2. Images of the surfaces with a projected pattern are shown in Figure D.2(a), while reconstructed depth maps, and point clouds are shown in Figure D.2(b) and Figure D.2(c), respectively.

A simple quantitative evaluation of the quality of 3D reconstruction is obtained by examining the deviation of the 3D points from a plane fitted to the point cloud using the RANSAC algorithm. For a more comprehensive examination of many available optical flow algorithms and their performance analysis, refer to the surveys provided in [Baker et al., 2011, 2009, 2007].

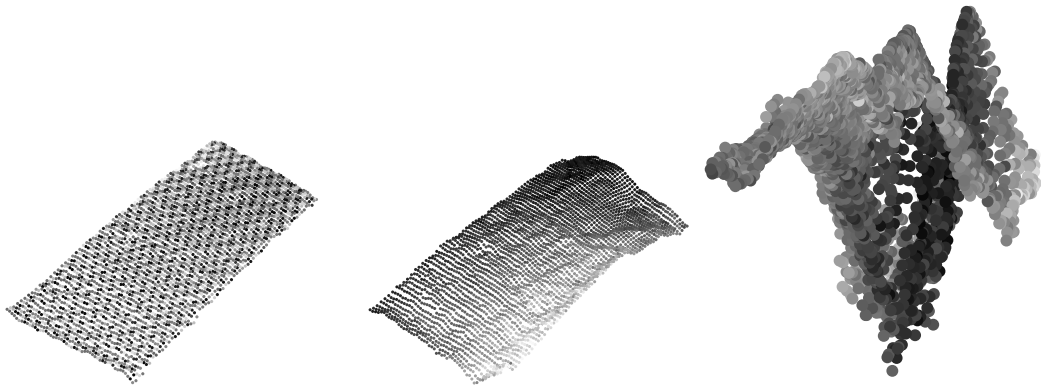
Table D.1 provides a summary of the quality of 3D reconstruction. It presents the standard deviation ( $\sigma$ ) of the points from the fitted plane for each surface type, with and without the projected pattern. Results are shown for reconstruction using individual depth maps for each pair of images, as well as the single filtered depth maps obtain using the median filter procedure described above. When the surface has sufficient visible texture, the chosen optical flow algorithm allows for accurate 3D reconstruction. The quality of the reconstruction decreases when the surface has a less pronounced texture. In cases where no texture and no features can be observed, recovered 3D structure exhibits significant error. The projected pattern adds artificial texture to surfaces. This allows for accurate 3D reconstruction of the surface, even when it does not posses visible texture naturally.



(a)



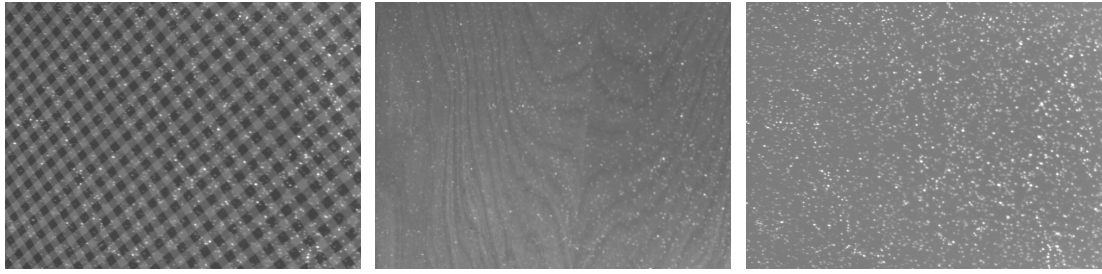
(b)



(c)

Figure D.1: Results of 3D reconstruction for three types of surfaces, without a projected pattern (a). Reconstructed depth maps are shown in (b), with the corresponding point clouds shown in (c).

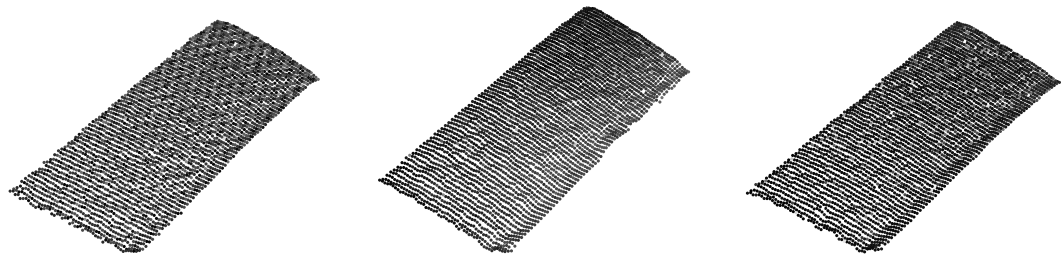




(a)



(b)



(c)

Figure D.2: Results of 3D reconstruction for three types of surfaces, using a projected pattern (a). Reconstructed depth maps are shown in (b), with the corresponding point clouds shown in (c).

| Surface Type       | No Pattern            |                     | Pattern               |                     |
|--------------------|-----------------------|---------------------|-----------------------|---------------------|
|                    | $\sigma$ (unfiltered) | $\sigma$ (filtered) | $\sigma$ (unfiltered) | $\sigma$ (filtered) |
| Checker pattern    | 0.0046                | 0.0032              | 0.0047                | 0.0026              |
| Wooden texture     | 0.0505                | 0.0350              | 0.0038                | 0.0024              |
| Textureless, white | N/A                   | 0.6028              | 0.0039                | 0.0023              |

Table D.1: Standard deviation of reconstructed points from a fitted planar model for different surface types. All values are presented in meters.

# References

---

- Andreopoulos, A. and Tsotsos, J. Active vision for door localization and door opening using playbot: A computer controlled wheelchair for people with mobility impairments. In *Proceedings of the 5th Canadian Conference on Computer and Robot Vision*, pages 3–10. 2008.
- Asfour, T., Azad, P., Vahrenkamp, N., Regenstein, K., Bierbaum, A., Welke, K., Schroder, J., and Dillmann, R. Toward humanoid manipulation in human-centred environments. *Robotics and Autonomous Systems*, 56:54–65, 2008.
- Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M.J., and Szeliski, R. A database and evaluation methodology for optical flow. In *Proceedings of the 11th IEEE International Conference on Computer Vision*, pages 1–8. 2007.
- Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M.J., and Szeliski, R. A Database and Evaluation Methodology for Optical Flow. Technical report, Middlebury College, Microsoft Research, 2009.
- Baker, S., Scharstein, D., Lewis, J., Roth, S., Black, M., and Szeliski, R. A database and evaluation methodology for optical flow. *International Journal of Computer Vision*, 92(1):1–31, 2011.
- Batra, D., Kowdle, A., Parikh, D., Luo, J., and Chen, T. iCoseg: Interactive co-segmentation with intelligent scribble guidance. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3169 –3176. June 2010.
- Batra, D., Kowdle, A., Parikh, D., Luo, J., and Chen, T. Interactively co-segmentating topically related images with intelligent scribble guidance. *International Journal of Computer Vision*, 93:273–292, 2011.

- Beucher, S. The watershed transformation applied to image segmentation. *Scanning Microscopy Supplement*, 6:299–314, 1992.
- Bjorkman, M. and Eklundh, J. Vision in the real world: Finding, attending and recognizing objects. *International Journal of Imaging Systems and Technology*, 16(5):189–208, 2006.
- Bjorkman, M. and Kragic, D. Active 3d scene segmentation and detection of unknown objects. In *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, pages 3114–3120. May 2010a.
- Bjorkman, M. and Kragic, D. Active 3d segmentation through fixation of previously unseen objects. In *Proceedings of the British Machine Vision Conference*, pages 119.1–119.11. BMVA Press, 2010b.
- Blake, A., Kohli, P., and Rother, C., editors. *Markov Random Fields for Vision and Image Processing*. MIT Press, 2011.
- Bleiweiss, A. and Werman, M. Fusing time-of-flight depth and color for real-time segmentation and tracking. In *Dynamic 3D Imaging*, editors A. Kolb and R. Koch, volume 5742 of *Lecture Notes in Computer Science*, pages 58–69. Springer Berlin, 2009.
- Bone, G., Lambert, A., and Edwards, M. Automated modeling and robotic grasping of unknown three-dimensional objects. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 292–298. 2008.
- Boulch, A. and Marlet, R. Fast and robust normal estimation for point clouds with sharp features. In *Computer Graphics Forum*, volume 31, pages 1765–1774. Wiley Online Library, 2012.
- Boykov, Y. and Funka-Lea, G. Graph cuts and efficient N-D image segmentation. *International Journal of Computer Vision*, 70(2):109–131, 2006.

- Boykov, Y. and Jolly, M. Interactive graph cuts for optimal boundary & region segmentation of objects in N-D images. In *Proceedings of the 8th IEEE International Conference on Computer Vision*, volume 1, pages 105–112. 2001.
- Boykov, Y. and Kolmogorov, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.
- Boykov, Y., Veksler, O., and Zabih, R. Markov random fields with efficient approximations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 648–655. June 1998.
- Busin, L., Vandenbroucke, N., and Macaire, L. Color spaces and image segmentation. volume 151 of *Advances in Imaging and Electron Physics*, pages 65–168. Elsevier, 2008.
- Caselles, V., Kimmel, R., and Sapiro, G. Geodesic active contours. *International Journal of Computer Vision*, 22(1):61–79, 1997.
- Cheng, Y. Mean shift, mode seeking, and clustering. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(8):790–799, 1995.
- Cho, S., Haralick, R.M., and Yi, S. Improvement of kittler and illingworth’s minimum error thresholding. *Pattern Recognition*, 22:609–617, 1989.
- Chung, W., Kim, G., and Kim, M. Development of the multi-functional indoor service robot psr systems. *Autonomous Robots*, 22(1):1–17, 2007.
- Chung, W., Rhee, C., Shim, Y., Lee, H., and Park, S. Door-opening control of a service robot using the multifingered robot hand. *IEEE Transactions on Industrial Electronics*, 56(10):3975–3984, October 2009.
- Cigla, C. and Alatan, A. Object segmentation in multi-view video via color, depth and motion cues. In *Proceedings of the 17th IEEE Signal Processing and Communications Applications Conference*, pages 668–671. Antalya, Turkey, April 2009.

- Cohen, L.D. and Cohen, I. Finite-element methods for active contour models and balloons for 2-d and 3-d images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15:1131–1147, 1993.
- Comaniciu, D. and Meer, P. Mean shift: A robust approach toward feature space analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(5):603–619, 2002.
- Cootes, T., Taylor, C., Cooper, D., and Graham, J. Active shape models - Their training and application. *Computer Vision and Image Understanding*, 61(1):38–59, 1995.
- Cremers, D., Rousson, M., and Deriche, R. A review of statistical approaches to level set segmentation: integrating color, texture, motion and shape. *International Journal of Computer Vision*, 72(2):195–215, 2007.
- Cyganek, B. and Siebert, J. *An introduction to 3D computer vision techniques and algorithms*. John Wiley and Sons, 2009.
- Felzenszwalb, P. and Huttenlocher, D. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.
- Filliat, D. and Meyer, J. Map-based navigation in mobile robots:: I. A review of localization strategies. *Cognitive Systems Research*, 4(4):243–282, 2003.
- Fischler, M.A. and Bolles, R.C. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 6:381–395, June 1981.
- Forsyth, D. and Ponce, J. *Computer vision: a modern approach*. Prentice Hall, 2002.
- Franke, M. Color image segmentation based on an iterative graph cut algorithm using time-of-flight cameras. *Pattern Recognition*, 6835:462–467, 2011.

- Fukunaga, K. and Hostetler, L. The estimation of the gradient of a density function, with applications in pattern recognition. *IEEE Transactions on Information Theory*, 21(1):32–40, January 1975.
- Geman, S. and Geman, D. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (6):721–741, 1984.
- Goddard, J.S. *Pose and Motion Estimation from Vision using Dual Quaternion-Based Extended Kalman Filtering*. Ph.D. thesis, University of Tennessee, Knoxville, 1997.
- Greig, D., Porteous, B., and Seheult, A. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society. Series B*, pages 271–279, 1989.
- Gumhold, S., Wang, X., and MacLeod, R. Feature extraction from point clouds. In *Proceedings of the 10th International Meshing Roundtable*, pages 293–305. 2001.
- Haralick, R. and Shapiro, L. Image segmentation techniques. *Computer Vision, Graphics, and Image Processing*, 29(1):100–132, 1985.
- Harville, M., Gordon, G., and Woodfill, J. Foreground segmentation using adaptive mixture models in color and depth. In *Proceedings of the IEEE Workshop on Detection and Recognition of Events in Video*, pages 3–12. 2001.
- Hirano, Y., Kitahama, K., and Yoshizawa, S. Image-based object recognition and dexterous hand/arm motion planning using RRTS for grasping in cluttered scene. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2041–2046. 2005.
- Huebner, K. and Kragic, D. Selection of robot pre-grasps using box-based shape approximation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1765–1770. 2008.

- Huebner, K., Welke, K., Przybylski, M., Vahrenkamp, N., Asfour, T., Kragic, D., and Dillmann, R. Grasping known objects with humanoid robots: A box-based approach. In *Proceedings of the International Conference on Advanced Robotics*, pages 1–6. 2009.
- Ihrke, I., Kutulakos, K., Lensch, H., Magnor, M., and Heidrich, W. Transparent and specular object reconstruction. *Computer Graphics Forum*, 29:2400–2426, 2010.
- Ilea, D. and Whelan, P. Image segmentation based on the integration of colour-texture descriptors—a review. *Pattern Recognition*, 44:2479–2501, 2011.
- Jain, A.K. and Farrokhnia, F. Unsupervised texture segmentation using gabor filters. *Pattern Recognition*, 24(12):1167 – 1186, 1991.
- Jain, A. and Kemp, C. EL-E: An assistive mobile manipulator that autonomously fetches objects from flat surfaces. *Autonomous Robots*, 28(1):45–64, 2010.
- Jia, Y. and Zhang, C. Learning distance metric for semi-supervised image segmentation. In *Proceedings of the 15th IEEE International Conference on Image Processing*, pages 3204 –3207. October 2008.
- Johnson-Roberson, M., Bohg, J., Bjorkman, M., and Kragic, D. Attention-based active 3d point cloud segmentation. In *Proceedings of the 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1165–1170. October 2010.
- Kapur, J., Sahoo, P., and Wong, A. A new method for gray-level picture thresholding using the entropy of the histogram. *Computer Vision, Graphics, and Image Processing*, 29(3):273–285, 1985.
- Kass, M., Witkin, A., and Terzopoulos, D. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.
- Kim, J. and Hong, K. Color-texture segmentation using unsupervised graph cuts. *Pattern Recognition*, 42(5):735–750, 2009.



- Kittler, J. and Illingworth, J. Minimum error thresholding. *Pattern Recognition*, 19:41–47, 1986a.
- Kittler, J. and Illingworth, J. Minimum error thresholding. *Pattern Recognition*, 19(1):41–47, 1986b.
- Klingbeil, E., Saxena, A., and Ng, A. Learning to open new doors. In *proceedings of the Robotics Science and Systems workshop on Robot Manipulation*. 2008.
- Klingbeil, E., Saxena, A., and Ng, A. Learning to open new doors. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2751–2757. October 2010.
- Kolda, T.G., Lewis, R.M., and Torczon, V. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45:385–482, 2003.
- Kosov, S., Thormahlen, T., and Seidel, H.P. Using active illumination for accurate variational space-time stereo. In *Image Analysis*, editors A. Heyden and F. Kahl, volume 6688 of *Lecture Notes in Computer Science*, pages 752–763. Berlin: Springer, 2011.
- Kragic, D., Petersson, L., and Christensen, H. Visually guided manipulation tasks. *Robotics and Autonomous Systems*, 40(2-3):193–203, 2002.
- Kuehnle, J., Xue, Z., Stotz, M., Zoellner, J., Verl, A., and Dillmann, R. Grasping in depth maps of time-of-flight cameras. In *Proceedings of the International Workshop on Robotic and Sensors Environments*, pages 132–137. Ottawa, Canada, October 2008.
- Kunchev, V., Jain, L., Ivancevic, V., and Finn, A. Path planning and obstacle avoidance for autonomous mobile robots: A review. In *Knowledge-Based Intelligent Information and Engineering Systems*, pages 537–544. Springer, 2006.

- Lafferty, J., McCallum, A., and Pereira, F. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, pages 282–289. 2001.
- LeCun, Y., Chopra, S., Hadsell, R., Huang, F.J., Bakir, G., Hofman, T., Scholkopf, B., Smola, A., and Taskar, B. A tutorial on energy-based learning. In *Predicting Structured Data*, pages 191–241. MIT Press, 2006.
- Leung, T. and Malik, J. Representing and recognizing the visual appearance of materials using three-dimensional textons. *International Journal of Computer Vision*, 43(1):29–44, 2001.
- Li, S.Z. *Markov Random Field Modeling in Image Analysis*. Advances in Pattern Recognition. Springer London, 3rd edition, 2009.
- Li, Q. and Griffiths, J. Least squares ellipsoid specific fitting. In *Proceedings of Geometric Modeling and Processing*, pages 335 – 340. 2004.
- Liu, G., Ahmad, S., and Ren, L. Hybrid control of door-opening by modular reconfigurable robots. In *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 954–959. St. Louis, MO, October 2009.
- Liu, G., He, X., Yuan, J., Abdul, S., and Goldenberg, A. Development of modular and reconfigurable robot with multiple working modes. In *Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, pages 3502–3507. Pasadena, CA, May 2008.
- Lowe, D.G. Three-dimensional object recognition from single two-dimensional images. *Artificial Intelligence*, 31(3):355–395, 1987.
- Lowe, D.G. Fitting parameterized three-dimensional models to images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(5):441–450, 1991.

- Malik, J. and Perona, P. Preattentive texture discrimination with early vision mechanisms. *Journal of the Optical Society of America A*, 7(5):923–932, May 1990.
- Malladi, R., Sethian, J., and Vemuri, B. Shape modeling with front propagation: A level set approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(2):158–175, 1995.
- Marton, Z.C., Goron, L., Rusu, R.B., and Beetz, M. Reconstruction and verification of 3d object models for grasping. In *Proceedings of the 14th International Symposium on Robotics Research*, pages 315–328. Lucerne, Switzerland, 2009.
- McKenna, S., Raja, Y., and Gong, S. Tracking colour objects using adaptive mixture models. *Image and Vision Computing*, 17(3):225–231, 1999.
- Meyer, J. and Filliat, D. Map-based navigation in mobile robots:: II. A review of map-learning and path-planning strategies. *Cognitive Systems Research*, 4(4):283–317, 2003.
- Miller, A. and Allen, P. Graspit! A versatile simulator for robotic grasping. *Robotics and Automation Magazine*, 11(4):110–122, 2004.
- Mudigonda, P.K. *Combinatorial and Convex Optimization for Probabilistic Models in Computer Vision*. Ph.D. thesis, Oxford Brookes University, 2008.
- Nakagawa, Y. and Rosenfeld, A. Some experiments on variable thresholding. *Pattern Recognition*, 11:191–204, 1979.
- Nguyen, N. and Guo, Y. Metric learning: A support vector approach. In *Machine Learning and Knowledge Discovery in Databases*, editors W. Daelemans, B. Goethals, and K. Morik, volume 5212 of *Lecture Notes in Computer Science*, pages 125–136. Springer Berlin Heidelberg, 2008.
- Ojala, T., Pietikainen, M., and Harwood, D. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996.

- Ojala, T., Pietikainen, M., and Maenpaa, T. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, 2002.
- Otsu, N. A threshold selection method from gray-level histograms. *Automatica*, 11(285-296):23–27, 1975.
- Permuter, H., Francos, J., and Jermyn, I. A study of gaussian mixture models of color and texture features for image classification and segmentation. *Pattern Recognition*, 39(4):695–706, 2006.
- Petrovskaya, A. and Ng, A. Probabilistic mobile manipulation in dynamic environments, with application to opening doors. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 2178–2184. Hyderabad, India, 2007.
- Prats, M., Sanz, P., and del Pobil, A. Reliable non-prehensile door opening through the combination of vision, tactile and force feedback. *Autonomous Robots*, 29(2):201–218, 2010.
- Protiere, A. and Sapiro, G. Interactive image segmentation via adaptive weighted distances. *IEEE Transactions on Image Processing*, 16(4):1046–1057, 2007.
- Rao, D., Le, Q., Phoka, T., Quigley, M., Sudsang, A., and Ng, A. Grasping novel objects with depth segmentation. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2578–2585. October 2010.
- Rasolzadeh, B., Bjorkman, M., Huebner, K., and Kragic, D. An active vision system for detecting, fixating and manipulating objects in the real world. *International Journal of Robotics Research*, 29(2-3):133–154, 2010.
- Reiser, U., Connette, C., Fischer, J., Kubacki, J., Bubeck, A., Weisshardt, F., Jacobs, T., Parlitz, C., Hagele, M., and Verl, A. Care-O-Bot 3 - Creating a product vision for service robot applications by integrating design and technology. In *Proceedings*

- of the *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1992–1998. IEEE, 2009.
- Rhee, C., Chung, W., Kim, M., Shim, Y., and Lee, H. Door opening control using the multi-fingered robotic hand for the indoor service robot. In *Proceedings of the 2004 IEEE International Conference on Robotics and Automation*, pages 4011–4016. 2004.
- Ritter, Gerhard, X. and Wilson, Joseoh, N. *Handbook of Computer Vision Algorithms in Image Algebra*. CRC Press, 2001.
- Rosenfeld, A. and Torre, P.D.L. Histogram concavity analysis as an aid in threshold selection. *IEEE Transactions on Systems, Man, and Cybernetics*, 1983.
- Rosenhahn, B. *Pose Estimation Revisited*. Ph.D. thesis, Universitat Kiel, 2003.
- Rotenstein, A., Andreopoulos, A., Fazl, E., Jacob, D., Robinson, M., Shubina, K., Zhu, Y., and Tsotsos, J. Towards the dream of an intelligent, visually-guided wheelchair. In *Proceedings of the 2nd International Conference on Technology and Aging*. 2007.
- Rother, C., Kolmogorov, V., and Blake, A. Grabcut: Interactive foreground extraction using iterated graph cuts. *ACM Transactions on Graphics*, 23(3):309–314, 2004.
- Rusu, R.B., Bradski, G., Thibaux, R., and Hsu, J. Fast 3d recognition and pose using the viewpoint feature histogram. In *Proceedings of the 23rd IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2155–2162. Taipei, Taiwan, 2010.
- Rusu, R.B., Holzbach, A., Diankov, R., Bradski, G., and Beetz, M. Perception for mobile manipulation and grasping using active stereo. In *Proceedings of the 9th IEEE-RAS International Conference on Humanoid Robots*, pages 632–638. Paris, France, December 2009a.
- Rusu, R., Meeussen, W., Chitta, S., and Beetz, M. Laser-based perception for door and handle identification. In *Proceedings of the International Conference on Advanced Robotics*, pages 1–8. Munich, June 2009b.

- Sauvola, J.J. and Pietikainen, M. Adaptive document image binarization. *Pattern Recognition*, 33:225–236, 2000.
- Scharstein, D. and Szeliski, R. High-accuracy stereo depth maps using structured light. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages I–195 – I–202. June 2003.
- Shapiro, L. and Stockman, G. *Computer Vision*. Prentice-Hall, 2001.
- Shi, J. and Malik, J. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.
- Sobieranski, A., Abdala, D., Comunello, E., and Wangenheim, A. Learning a color distance metric for region-based image segmentation. *Pattern Recognition Letters*, 30(16):1496–1506, 2009.
- Sobieranski, A., Comunello, E., and von Wangenheim, A. Learning a nonlinear distance metric for supervised region-merging image segmentation. *Computer Vision and Image Understanding*, 115(2):127–139, 2011.
- Srinivasa, S., Berenson, D., Cakmak, M., Collet, A., Dogar, M., Dragan, A., Knepper, R., Niemuehler, T., Strabala, K., Vande Weghe, M., et al. Herb 2.0: Lessons learned from developing a mobile manipulator for the home. *Proceedings of the IEEE*, 100(8):2410–2428, 2012.
- Srinivasa, S., Ferguson, D., Helfrich, C., Berenson, D., Collet, A., Diankov, R., Gallagher, G., Hollinger, G., Kuffner, J., and Weghe, M. Herb: A home exploring robotic butler. *Autonomous Robots*, 28(1):1–17, 2010.
- Strom, J., Richardson, A., and Olson, E. Graph-based segmentation for colored 3d laser point clouds. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2131–2136. October 2010.

- Szeliski, R. *Computer Vision: Algorithms and Applications*. Springer, 2010.
- Szumner, M., Kohli, P., and Hoiem, D. Learning CRFs using graph cuts. In *Computer Vision – ECCV 2008*, editors D. Forsyth, P. Torr, and A. Zisserman, volume 5303 of *Lecture Notes in Computer Science*, pages 582–595. Springer Berlin / Heidelberg, 2008.
- Tkalcic, M. and Tasic, J. Colour spaces: Perceptual, historical and applicational background. In *Proceedings of the IEEE Region 8 EUROCON. Computer as a Tool.*, volume 1, pages 304–308. September 2003.
- Truccp, E. and Verri, A. *Introductory Techniques for 3-D Computer Vision*. Prentice Hall, 1998.
- Tsai, D.M. A fast thresholding selection procedure for multimodal and unimodal histograms. *Pattern Recognition Letters*, 16:653–666, 1995.
- Tsotsos, J., Verghese, G., Dickinson, S., Jenkin, M., Jepson, A., Milios, E., Nuflo, F., Stevenson, S., Black, M., Metaxas, D., Culhane, S., Ye, Y., and Mann, R. Playbot a visually-guided robot for physically disabled children. *Image and Vision Computing*, 16(4):275–292, 1998.
- Varma, M. and Zisserman, A. A statistical approach to texture classification from single images. *International Journal of Computer Vision*, 62(1-2):61–81, 2005.
- Vincent, L. and Soille, P. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13:583–598, 1991.
- Wang, B., Jiang, J.W., Cai, H.G., and Liu, H. Grasping unknown objects based on 3d model reconstruction. In *Proceedings of the International Conference on Advanced intelligent Mechatronics*, pages 461–466. Monterey, California, USA, 2005.

- Wang, S. and Siskind, J.M. Image segmentation with minimum mean cut. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 517–524. 2001.
- Wang, L., Yang, R., and Davis, J. BRDF invariant stereo using light transport constancy. *IEEE transactions on pattern analysis and machine intelligence*, 29(9):1616–1626, 2007.
- Wang, F., Zhang, C., Shen, H., and Wang, J. Semi-supervised classification using linear neighborhood propagation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, pages 160–167. IEEE, 2006.
- Weinberger, K., Blitzer, J., and Saul, L. Distance metric learning for large margin nearest neighbor classification. In *Advances in Neural Information Processing Systems 18*, editors Y. Weiss, B. Scholkopf, and J. Platt, pages 1473–1480. MIT Press, Cambridge, MA, 2006.
- Weinberger, K.Q. and Saul, L.K. Fast solvers and efficient implementations for distance metric learning. In *Proceedings of the 25th International Conference on Machine Learning*, pages 1160–1167. ACM, 2008.
- Weinberger, K. and Saul, L. Distance metric learning for large margin nearest neighbor classification. *The Journal of Machine Learning Research*, 10:207–244, 2009.
- Weszka, J.S. A survey of threshold selection techniques. *Computer Graphics and Image Processing*, 7:259–265, 1978.
- Wu, Z. and Leahy, R. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113, 1993.
- Xiang, S., Nie, F., and Zhang, C. Learning a Mahalanobis distance metric for data clustering and classification. *Pattern Recognition*, 41(12):3600 – 3612, 2008.



- Xing, E.P., Ng, A.Y., Jordan, M.I., and Russell, S.J. Distance metric learning with application to clustering with side-information. In *Proceedings of the Neural Information Processing Systems Conference*, pages 505–512. 2002.
- Xu, C. and Prince, J.L. Snakes, shapes, and gradient vector flow. *IEEE Transactions on Image Processing*, 7:359–369, 1998.
- Yamazaki, K., Tomono, M., and Tsubouchi, T. Picking up an unknown object through autonomous modeling and grasp planning by a mobile manipulator. *Field and Service Robotics*, 42:563–571, 2008.
- Yamazaki, K., Tomono, M., Tsubouchi, T., and Yuta, S. A grasp planning for picking up an unknown object for a mobile manipulator. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 2143–2149. 2006.
- Yang, L. and Jin, R. Distance metric learning: A comprehensive survey. Technical report, Michigan State University, 2006.
- Zach, C., Pock, T., and Bischof, H. A duality based approach for realtime TV-L1 optical flow. In *Pattern Recognition*, editors F. Hamprecht, C. Schnrr, and B. Jahne, volume 4713 of *Lecture Notes in Computer Science*, pages 214–223. Springer Berlin Heidelberg, 2007.
- Zhang, L., Curless, B., and Seitz, S.M. Spacetime stereo: Shape recovery for dynamic scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 367–374. June 2003.
- Zickler, T., Ho, J., Kriegman, D., Ponce, J., and Belhumeur, P. Binocular helmholtz stereopsis. In *Proceedings of the 9th IEEE International Conference on Computer Vision*, volume 2, pages 1411 –1417. October 2003.