5-2013

Sensing Highly Non-Rigid Objects with RGBD Sensors for Robotic Systems

Robert Willimon
Clemson University, rwillim@clemson.edu

Sensing Highly Non-Rigid Objects with RGBD Sensors for Robotic Systems

---

A Dissertation
Presented to
the Graduate School of
Clemson University

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
Computer Engineering

---

by
R. Bryan Willimon
May 2013

Abstract

The goal of this research is to enable a robotic system to manipulate clothing and other highly non-rigid objects using an RGBD sensor. The focus of this thesis is to define and test various algorithms / models that are used to solve parts of the laundry process (i.e. handling, classifying, sorting, unfolding, and folding).

First, a system is presented for automatically extracting and classifying items in a pile of laundry. Using only visual sensors, the robot identifies and extracts items sequentially from the pile. When an item is removed and isolated, a model is captured of the shape and appearance of the object, which is then compared against a dataset of known items. The contributions of this part of the laundry process are a novel method for extracting articles of clothing from a pile of laundry, a novel method of classifying clothing using interactive perception, and a multi-layer approach termed **L-M-H**, more specifically **L-C-S-H** for clothing classification. This thesis describes two different approaches to classify clothing into categories. The first approach relies upon silhouettes, edges, and other low-level image measurements of the articles of clothing. Experiments from the first approach demonstrate the ability of the system to efficiently classify and label into one of six categories (pants, shorts, short-sleeve shirt, long-sleeve shirt, socks, or underwear). These results show that, on average, classification rates using robot interaction are 59% higher than those that do not use interaction. The second approach relies upon color, texture, shape, and edge information from 2D and 3D data within a local and global perspective. The multi-layer approach compartmentalizes the problem into a high (**H**) layer, multiple mid-level (characteristics(**C**), selection masks(**S**)) layers, and a low (**L**) layer. This approach produces "local" solutions to solve the global classification problem. Experiments demonstrate the ability of the system to efficiently classify each article of clothing into one of seven categories (pants, shorts, shirts, socks, dresses, cloths, or jackets). The results presented in this paper show that, on average, the classification rates improve by +27.47% for three categories, +17.90% for four categories, and +10.35% for seven categories over the baseline system, using support vector machines.

Second, an algorithm is presented for automatically unfolding a piece of clothing. A piece of cloth is pulled in different directions at various points of the cloth in order to flatten the cloth. The features of the cloth are extracted and calculated to determine a valid location and orientation in which to interact with it. The features include the peak region, corner locations, and continuity / discontinuity of the cloth. In this thesis, a two-stage algorithm is presented, introducing a novel solution to the unfolding / flattening problem using interactive perception. Simulations using 3D simulation software, and experiments with robot hardware demonstrate the ability of the algorithm to flatten pieces of laundry using different starting configurations. These results show that, at most, the algorithm flattens out a piece of cloth from 11.1% to 95.6% of the canonical configuration.

Third, an energy minimization algorithm is presented that is designed to estimate the configuration of a deformable object. This approach utilizes an RGBD image to calculate feature correspondence (using SURF features), depth values, and boundary locations. Input from a Kinect sensor is used to segment the deformable surface from the background using an alpha-beta swap algorithm. Using this segmentation, the system creates an initial mesh model without prior information of the surface geometry, and it reinitializes the configuration of the mesh model after a loss of input data. This approach is able to handle in-plane rotation, out-of-plane rotation, and varying changes in translation and scale. Results display the proposed algorithm over a dataset consisting of seven shirts, two pairs of shorts, two posters, and a pair of pants. The current approach is compared using a simulated shirt model in order to calculate the mean square error of the distance from the vertices on the mesh model to the ground truth, provided by the simulation model.

Dedication

I would like to dedicate my dissertation to my wife, Kelli, and daughter, Olivia, whose love, support, and inspiration has allowed me to pursue this journey.

Acknowledgments

First I would most like to acknowledge and express my appreciation for the immeasurable support and guidance contributed by Dr. Stan T. Birchfield and Dr. Ian D. Walker, my co-advisors, who guided me through many types of situations, and provided constant support that made my journey a lot easier than it would have been.

Additionally, I also want to express my gratitude to Dr. Adam W. Hoover, and Dr. Damon L. Woodard, not only for their input in the preparation of this dissertation, but also for the many hours of quality instruction they have provided to me in my graduate studies leading up to this point.

I would like to thank all the members, past and present, of the Birchfield group who directly and indirectly provided helpful discussion, and assistance.

Finally, I gratefully acknowledge financial support from the National Science Foundation and DARPA.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

The process of classifying, sorting, and manipulating items is a daily routine for many individuals at home and at work. Daily chores around the house and at work require some type of sorting process. We sort items by which group does it belong to, what space does it occupy, what type of material is it made of, what temperature is it, what shape is it, who it belongs to, what type of color is it, how do you use it, etc.. The items that are classified, sorted, and manipulated range from rigid to semi-rigid to non-rigid to highly non-rigid objects (e.g. recycleables, mail, food, and clothing). Below are a few examples of commercial robotics and residential situations for classifying, sorting, and manipulating.

Industrial robotic systems Recycling bins are placed in commercial settings or public areas for the purpose of reducing the amount of trash that goes to the local landfill. Organizations like Osaka University, IDEC, Mitsubishi Electric Engineering, and ZenRobotics have been developing robotic systems designed to sort and extract recyclable items from construction and residential trash. The United States Postal Service sorts around 563 million pieces of mail a day [98]. Automated mail sorting machines have been in use, by the United States Postal Service, since the late 1950's, e.g. the Transorma. Several companies like National Presort Inc (NPI) and the United States Postal Service (USPS) specialize in automated sorting machines (i.e. Crossfire and AFSM100, respectively) to handle the large amount of mail and parcels on a daily basis. Large food distribution centers handle large quantities of food products a year (e.g. Peperami distributes over 110 million sticks of salami a year [1]). Previously, the food was handled by individuals with a conveyor belt and now robots have been introduced into the process to increase productivity. ABB and Adept designed 3-axis (ABB FlexPicker IRB360) and 4-axis (Adept Quattro s650H) robots for use within a clean room and a meat / poultry processing plant. Companies, like Peperami and Honeytop, have integrated the 3-axis robot into the handling and sorting of food (i.e. salami and pancakes, respectively) onto a pre-designed conveyor belt. Millers textile services, HSHS Shared Services, and UniFirst Laundry utilize large robotic machines that group piles of laundry to wash, dry, and fold. The wash / dry process is streamlined by using conveyor belts to carry washed linens to the dryers. Currently, the folding process is achieved by a human attaching two corners of the linen to a machine and the machine completes the folding process.

Domestic robotic systems Many domestic robotic systems are divided into groups that are categorized into outdoor tasks and indoor chores. Some of the outdoor tasks that robots are being applied to involve lawn mowing (e.g. Friendly Robotics Robo-Mower and Husqvarna Automower), pool cleaning (e.g. Weda B480 robotic commercial pool cleaner), and gutter cleaning (e.g. iRobot Looj). Another task that can be

[1]http://www.peperami.com/

labeled as outdoor or indoor is window cleaning. The Windoro WCR-I001 window cleaning robot is capable of cleaning single and double pane windows. The more popular domestic robots, iRobot's Roomba and Samsung's Navibot, handle vacuuming carpets and cleaning floors. A smaller robotic system used for animal care is the Litter Robot 2 and Catgenie, created by Petnovations, designed to clean litter boxes. Domestic robot assistants developed by Tokyo University's IRT and Toyota's HSR are capable of simple tasks involving picking up dropped objects, retreiving objects from a table, and placing laundry into the washing machine. Korea Institute of Science and Technology developed Mahru-Z and Mahru-M in order to provide help around the house. The Mahru robots are able to place food in a basket and carry it to the owner, also turn on other appliances like a microwave, washing machine, and toasters. Lastly, contributions towards laundry handling [41, 114, 87, 53, 54, 56, 35, 57], laundry classifying [56, 104, 111, 48, 71, 55, 17], folding clothes [70, 67, 66], and unfolding clothes [105] by robotic systems have been researched for over a decade. These robotic solutions to laundry applications will be discussed in detail later in the thesis. Handling, classifying, folding, and unfolding research is conducted by several dual arm manipulators by Willow Garage's PR2 and Kawada Industries' HRP-2 / HRP-4.

1.2 Focus of thesis

One of the areas in robotics that is gaining momentum is focused on classifying, sorting, and manipulating highly non-rigid objects, e.g. clothing / laundry. The process of "doing the laundry" consists of several steps: handling, washing, drying, separating / isolating, classifying, unfolding / flattening, folding, and putting clothing away into a predetermined drawer or storage unit. Figure 1.1 illustrates a high level overview of the laundry process. The focus of this thesis is to explore three parts of the laundry process: classifying, unfolding / flattening, and folding. The classifying part of the laundry process involves taking features and characteristics from the separated / isolated article of clothing and categorizing the item as a shirt, pair of pants, shorts, etc, discussed in section 1.3. The unfolding / flattening part of the process involves removing any creases, folds, and crinkles so that the known item can be properly folded and stored, discussed in section 1.4. The folding part of the laundry process involves determining possible grasp points to manipulate it and fold it into a desired configuration / pose, discussed in section 1.5. The following sections describe the challenges of classifying, unfolding / flattening, and estimating the pose / configuration of an article of clothing for folding.

1.3 Challenge of classifying laundry

The general clothing classification problem is for a robotic system to properly and accurately sort a pile of laundry into specified, predefined groups (e.g., for loading the laundry into a washer, or after the drying process has completed). Laundry can be classified / sorted by category, age, gender, color (i.e. whites, colors, darks), season of use, or by individual. The specific problem addressed in this thesis is classifying each article of clothing into a specified category (e.g. shirts, pants, shorts, jackets, cloths, socks, dresses) using physical characteristics and selection masks to assist the process.

Other areas, along with classification, make up the process of "doing the laundry", mentioned in section 1.2. Figure 1.1 gives a flow chart of the various areas like classifying [56, 104, 48, 71, 55, 17, 35, 113, 89, 88, 64, 102, 110, 111], matching socks [66], and separating / isolating [57, 104, 47, 48, 71].

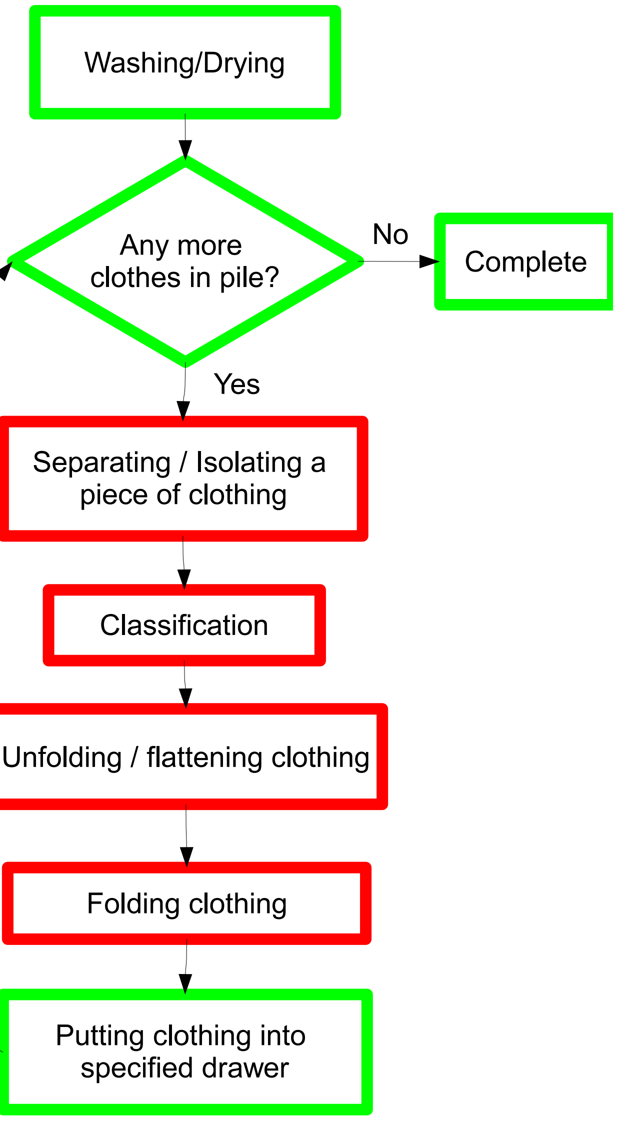

Washing/Drying
Any more clothes in pile?
No
Complete
Yes
Separating / Isolating a piece of clothing
Classification
Unfolding / flattening clothing
Folding clothing
Putting clothing into specified drawer

Figure 1.1: Overview of the laundry process, adapted from [47]. GREEN areas represent parts of the process that have already been explored in previous work, while the RED area represents the part of the process that is the focus of this thesis. The boxes labeled separating / isolating a piece of clothing and classification are discussed in section 1.3 and Chapter 3, the box labeled unfolding / flattening clothing is discussed in section 1.4 and Chapter 4, and the box labeled folding clothing is discussed in section 1.5 and Chapter 5.

In contrast to previous work, the method in this thesis operates on an unorganized, unflattened piece of laundry for the purpose of sorting each individual article. The dataset of clothing that is used in this thesis consists of individual articles placed on a table by a single robotic manipulator. Previous clothing classification papers [56, 48, 71, 55, 17] used the ability of dual manipulators to freely hang each piece of laundry at two calculated grasp points to fully reveal the overall shape and outline of each article of clothing.

In this thesis, a new approach is presented based on characteristics (which was inspired by [58]), a histogram for each article, and low-level image calculations for automatically classifying / sorting laundry into groups. The focus of this thesis is to demonstrate that the framework of a multi-layer classification strategy will improve overall results. Each layer of the approach can be modified to include any type of learning strategy (e.g. SVM, k-NN, neural networks, Bayesian classifier, etc). In this thesis, this approach is limited to one learning strategy since this is not the focus. This new approach is comprised of a multi-layer classification strategy consisting of SVM [12] and bag-of-words model [101], similar to [80], that utilizes characteristics and selection masks to bridge together each calculated low level feature to the correct category for each article of clothing.

The proposed method can be seen as a particular application of the paradigm of *interactive perception*, also known as *manipulation-guided sensing*, in which the manipulation is used to guide the sensing in order to gather information not obtainable through passive sensing alone [50, 51, 30, 103, 112, 106]. The clothing that is used in this paper is actively interacted with in order to gain more information about each article. In other words, deliberate actions change the state of the world in a way that simplifies perception and consequently future interactions.

1.4 Challenge of unfolding laundry

Unfolding laundry is a common household chore that is a difficult problem to automate. Several researchers have worked in the past on unfolding / flattening [105, 17], and folding [66] clothes, which is a later step in the laundry process (see Figure 1.1).

Folding, in particular, is receiving significant attention as of late [64, 70, 67, 66]. Various researchers are developing folding algorithms and machines at the lower end of the laundry process (i.e. folding clothing). Keio University's "Foldy" mobile robot has demonstrated the ability to fold a shirt based on high-level user input.[2] Similar research at other universities aimed at folding manipulation is making progress on folding T-shirts [70, 67] [3], towels [64], pants / shirts [66], and origami [2]. Others have developed cardboard machines to fold T-shirts.[4] These existing systems are invariably dependent on a starting point of having the items identified and laid out flat in a standard configuration prior to manipulation. Cusumano-Towner et al. [17] aimed at solving the unfolding/flattening problem, that is, to flatten a piece of crumpled clothing by implementing a disambiguation phase and a reconfiguration phase. In this thesis, algorithms, simulations, and experiments with robot hardware are presented, introducing a novel solution to the unfolding/flattening problem. In this thesis, a model / algorithm is designed to unfold / flatten clothes after they are labeled as a shirt, shorts, etc. through isolation and classification as in [104]. Figure 1.2 shows the robot system that was used in testing this approach.

[2]http://inventorspot.com/articles/laundryfolding_robot_learns_job_34327
[3]http://www.cs.dartmouth.edu/~robotics/movies/mpb-movie-09-shirt-folding.mov
[4]http://www.metacafe.com/watch/1165247/clothes_folding_machine

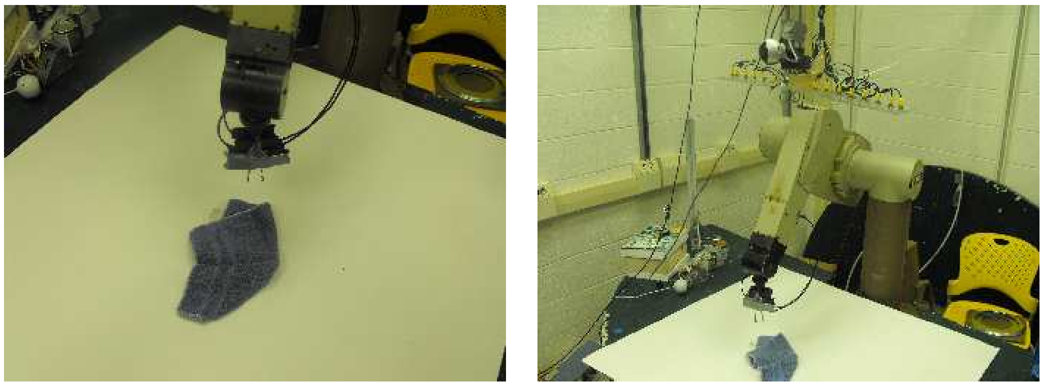


Figure 1.2: The robot system used for flattening a piece of clothing, in this case a washcloth: One PUMA manipulator and one Logitech Quickcam 4000.

1.5 Challenge of estimating the pose of laundry

Pose estimation of non-rigid objects is designed to provide a 3D representation of the article to be applied to object manipulation by calculating the location of grasp points. Laundry manipulation can be applied to washing, drying, folding, and storing clothing. Figure 1.1 gives a flow chart of the various areas like handling [41, 114, 87, 53, 54, 56, 35, 57] and folding a T-shirt [70, 67]. Generally, modeling and handling of deformable objects is applied to either augmented reality or cloth manipulation. One way to determine grasp locations on clothing is to first estimate the current configuration of the article of clothing. The estimated model can provide location and orientation information for particular areas of clothing. For instance, the waist line on a pair of pants, or the collar on a shirt, or the corners of a towel are common locations for humans to grasp laundry. Previous researchers use monocular image sequences [99, 91, 100] to estimate the configuration of non-rigid objects or to calculate the non-rigid structure from motion [24, 97].

In this thesis, a new approach is proposed to provide a model that maps to a deformable object with a low mean squared error using an RGBD sensor. The RGBD sensor provides a 3D view of the object for each frame without the need for 3D estimation using image sequences. Portions of this work appear in [107, 109, 108]. Various implementations are run on a simulated shirt, with ground truth data, to calculate quantitative results by measuring the distance error between the proposed model and the simulated model. The combined approach utilizes feature correspondences using SURF feature descriptors, 3D depth information, and boundary information that can model textured and textureless objects. This thesis focuses on the perception of the system instead of the manipulation of deformable objects. The results shown in the following chapters will demonstrate the effectiveness of the new approaches described in this thesis.

Chapter 2

Previous work

2.1 Clothing classification

There has been previous work in robot laundry which relates to the research in this thesis. Kaneko et al. [48] focused on a strategy to isolate and classify clothing. The work in [48] is an extension of the authors pioneering work in [47]. In these papers, the authors categorize what steps and tasks are needed for laundry (isolate, unfold, classify, fold, and put away clothes), as in Figure 1.1. The authors classify each object into one of three groups (shirt, pants, or towel) by deforming the article of clothing. The process of deforming the object aims to find hemlines as grasp points and the grasp points can not be closer than a predefined threshold. Once the object is held at two grasp points, then the shape of the object is broken up into different regions and the values of the different regions are used to calculate feature values. Each of the feature values are compared against threshold values. If the features are above or below the thresholds, then the object is classified as a shirt, pant, or towel based on a decision tree. The results achieved a 90% classification rate.

Osawa et al. [71] uses two manipulators to pick up / separate a piece of cloth, and classify it. The process began by picking up a piece of cloth from a side table and isolating it by holding it with one manipulator. A single camera is used to pick up / isolate the item and another single camera to locate grasp points while it is being held. While the item is currently being held by the first manipulator, the second manipulator targets the lowest point of the cloth as a second grasp point. The approach assumes that the lowest point is a corner of the article of clothing. This process of grabbing the lowest point of the object occurs two more times. The idea of grasping the lowest point tries to spread the article of clothing by the longest two corner points. A $2D$ outline model is created for each article of clothing which maps out every corner (concave and convex). Also, templates are created to compare each article of clothing after two grasp iterations. The templates are created by using the $2D$ outline model and determining what corner point is furthest from the current grasp point. The templates are also put through this process two more times to replicate what the $2D$ model will look like for comparison to the actual clothing. The article of clothing that is held by both robots is captured by the 2nd camera and converted into a binary image using background subtraction. This binary image is then compared with the binary template to classify the article of clothing. The approach was tested with eight different categories (male short sleeve shirt, women / child short sleeve shirt, long sleeve shirts, trousers, underwear shorts, bras, towels, and handkerchiefs).

Kita et al. [56] extend their original work [53] of determining the state of clothing. The work in [56] provides a three step process to successfully find a grasp point that the humanoid robot can grab: (1) clothing state estimation (previous work [53]); (2) calculation of the theoretically optimal position and orientation of the hand for the next action; (3) modification of position and orientation, based on the robots limitations, so the action can be executed. The 3D model is used by calculating the orientation of the shirt model at the grasp point. The 3D model gives three vectors around the grasp point and the vectors are used in the calculation of the orientation. The authors tested their results of six scenarios and were successful on five. One scenario failed at predicting the correct clothing state, four scenarios were successful but had to use step (3) for completion, and the last scenario was successful without using step (3).

The results in [55] are another extension to Kita et al.'s previous work [53]. The purpose is to accurately determine the shape / configuration of a piece of clothing based on the observed 3D data and the simulated 3D model. Each of the clothing states are compared against the 3D observed data based on the length of the clothing as a first step. Each of the shape states within a threshold value are selected to use in the next step of the process. Then, a correlation function is applied to the 2D plane of the observed data and simulated data to calculate which shape state model is closest to the observed data. The experimental results show that 22 out of 27 attempts were correctly classified as the correct shape state.

Kaneko et al. [47] focus on isolating the clothing after it has been washed and dried. The entire system consisted of four CCD cameras (front, overhead, right, left view) and one robotic arm, but only one camera was used (overhead view). The image was segmented by using Ohlander's region segmentation method (color segmentation) [69]. Then the region was found by finding the largest area and marking that region to be extracted. Finally, the surface geometry was used to determine the grasp point. In a follow-up paper [48], the authors revisit the same steps and tasks that are needed for laundry (isolate, unfold, classify, fold, and put away clothes) by including the task of classification. Two manipulators are used to deform the object multiple times until the shape is recognizable by the system. Finally, the object is classified into one of three groups (shirt, pants, or towel). The process of deforming the object is to find hemlines as grasp points, making sure not to keep the grasp points close together. If hemlines cannot be found, then the lowest part of the object is considered to be the next grasp point. Once the object is held at two grasp points, then the shape of the object is broken up into different regions and the values of the different regions are used to calculate feature values. Each of the feature values are measured against threshold values. Depending on whether the features are above or below the thresholds, then the object is classified as a shirt, pant, or towel based on a decision tree. Shadows (where parts of the object stick out or hang over another part) and convex areas are used to locate possible hemlines.

Cusumano-Towner et al. [17] present an extension of Osawa et al. [71] and Kita et al. [55] combined. The authors identified a clothing article by matching together silhouettes of the article to pre-computed simulated articles of clothing. The silhouettes of each iteration were placed into a Hidden Markov Model (HMM) to determine how well the silhouettes of the current article match against the simulated models of all categories of clothing. The authors placed the article on a table by sliding the clothing along the edge of the table to maintain the configuration while it was hanging in the air. The authors grasped two locations on the cloth during each iteration. The planning algorithm that determines where the next set of grasp points are located was computed by using a directed grasp-ability graph that locates all possible grasp points that can allow the system to reach. The authors tested their approach on seven articles of clothing of the end-to-end approach and were successful with 20 out of 30 trials.

The work in this thesis differs from previous papers [48] [71] [56] [55] [17] in that a single manipulator is used and the results do not depend on hanging clothing in the air by two grasp points. This approach is independent of the type of clothing or how many manipulators are used to collect the data. This work also differs from that of [56] [55] [17] in that predefined shape models are not used. Instead, novel characteristic features are used that describe each category of clothing. The characteristic features are not predefined, but calculated through training the algorithm with *a priori* collected clothing data.

This section of the thesis, described in Chapter 3, extends the problem of previous work [110, 111] by expanding the mid-level layer with more characteristics and incorporating more categories in the high layer. The database of clothing is increased, from previous work, from 117 articles to over 200 articles of various size, color, and type. The algorithm is compared to how well it improves over a baseline system of SVMs, one per category, to classify a pile of laundry into three, four, and seven groups.

2.2 Unfolding clothing

Gibbons et al. [35] describes a method to visually find grasp locations for clothing in a randomly positioned pile by filtering out folds. Kobayashi et al. [57] use three manipulators, a recognition board, two $3D$ hand vision sensors, and a $3D$ overview vision sensor to place a towel in a folding machine. The system of Osawa et al. [70] defines a way to fold clothes lying on a table starting from a hanging position using two manipulators.

Gibbons et al. [35] describes a method to visually find grasp locations for clothing in a randomly positioned pile. The idea is to look for edges on the clothing and filter out folds. The robot that is used was remotely controlled and the user provided the grasping regions. The grasping region was then studied to determine where and how the object is grasped. A $30x30$ pixel window was placed at the center of the region and rotated every 15 degrees to be used as a filter. Those 13 filters were trained on two scenes and tested on nine scenes. Each filter used four categories to determine what a good grasp point would be and what the orientation would be for the gripper. The categories are intensity, color, texture, and depth. The results showed that depth was the most successful. The nine scenes to be tested provided 117 matches of graspable locations, of which 27% of them were correct.

Kobayashi et al. [57] use two manipulators, a pulling up hand, a recognition board, two $3D$ hand vision sensors, and a $3D$ overview vision sensor. The pulling up hand is used as a 3^{rd} manipulator. The 3^{rd} manipulator is placed to the side of both manipulators with a function of only moving up and down with a single gripper attached. The $3D$ hand vision sensors and $3D$ overview vision sensor consists of a stereo camera pair along with a texture light to project a textured pattern on areas that do not have discernible textures. The hand sensors are rotated inward toward the center of the stereo pair to handle purely close range detection. The overall process consists of six steps: (1) Highest point pick up - extract towel on top of pile located on the recognition board; (2) Pick up the end of towel - robot detects an end point of the towel; (3) Passing the towel to the picking up hand - robot passes towel to pulling up hand so that both of the manipulators are free to look for corners; (4) grasping the 1^{st} corner - locate lowest corner of towel and grasp it; (5) Grasping the 2^{nd} corner - first robot spreads towel out and 2^{nd} manipulator searches for next lowest corner; (6) Setting the towel to the next line - by grasping two corners of the towel, the towel is spread out and placed in the cleaning / folding machine. For the vision aspect, the overhead stereo pair are used solely to locate the towel on top of the pile. After the towel is placed on the pulling up hand, each hand stereo pair uses a texture light to accurately determine the depth of the hanging towel and provide a $2.5D$ depth map of the towel. The depth map is used to isolate the towel and find the lowest hanging corner using corner detection. The corner is found by detecting all contours along the edge of the towel. When points along the edge move in a 90 degree angle, then a corner is detected.

The system of Osawa et al. [70] defines a way to fold clothes lying on a table starting from a hanging position. Each piece of clothing is held by two manipulators (one on each side and at the top of the clothing). The piece of cloth is then placed on the table and folded by using two external tools not connected to the robot. The external tools consist of (1) a rotating table with a third of the table able to fold inward, that folds a piece of cloth onto itself, and (2) a flattening tool which consists of a flat table used as a dead weight to get rid of wrinkles. The two manipulators were directly used by translating and rotating the cloth while on the rotating table. The system has a single overhead camera that monitors the status of the cloth lying on the table. The vision system contains models of each of the clothes. Each model describes where and in what order to fold the clothing. The experiments are tested on 4 different types of clothing; (1) towel, (2) short sleeve shirt, (3) pants, and (4) trunks. The results for each type of clothing are as follows: towel (100%), short sleeve shirt (95%), pants (80%), and trunks (90%). The results are calculated by determining how many clothes, out of 20, were folded correctly within a threshold. The threshold is calculated by comparing the actual folded cloth to a folded model of the cloth.

This thesis differs from the previous papers [35] [57] [70] in that a single manipulator is used to unfold a piece of clothing. This section of the thesis, described in Chapter 4, also describes a novel unfolding algorithm designed to flatten a cloth using a two-phase approach. This algorithm utilizes visual cues in order to calculate grasp points and the motion in which the manipulator pulls or pushes the article of clothing.

2.3 Pose estimation

Emerging research on non-rigid objects includes motion planning algorithms for deformable linear objects (DLOs) like ropes, cables, and sutures [86, 68]; Probabilistic RoadMap (PRM) planners for a flexible surface patch [45] or deformable object [4]; learning approaches to sense and model deformable surfaces [31, 54, 46]; and fabric manipulation for textile applications [6, 32, 75]. In particular, the problem of automating laundry has been receiving attention recently because of the increasing availability of calibrated two-handed robots like the PR2. Researchers have demonstrated systems for grasping clothes [35, 113], and tracing edges [88, 64]. Despite the progress in manipulating non-rigid objects, none of this research addresses the problem of estimating the 3D geometry of a non-rigid object.

Two recent research projects have independently aimed at the goal of non-rigid 3D geometry estimation. Bersch et al. [5] use fiducial markers printed on a T-shirt. An augmented reality toolkit is used to detect markers, and stereo imagery is used to automatically generate a 3D triangular mesh. When the article of clothing is picked up, the grasp point is inferred from the distances from the end effector to the visible markers which then leads to a succeedful grasp point in a two-handed robotic system. Similarly, Elbrechter et al. [22] print specially designed fiducial markers on both sides of a piece of paper. Multi-view stereo vision is used to estimate the 3D positions of the markers which are then sent to a physics-based modeling engine. In addition to using two computers to control the two hands, three computers are used for data acquisition and processing.

Other related works that estimate the pose of a non-rigid object, a person, or both use a triangular mesh to determine the non-rigid structure from motion (NRSfM). Researchers that estimate the pose of a non-rigid object use multiview

3D warps [10], quadratic deformation models [25, 24], global cost function [29, 28], Laplacian formulation [72], local deformation models [99, 91, 100], and affine motion estimation [15]. The approaches used to estimate the pose of person are adaptive metric registration [9], articulated structure [26], non-linear kernel model [38, 39], low-rank shape model [73], trajectory reconstruction [76, 77], and energy based multiple model fitting [83]. The models that are used to estimate the pose of a non-rigid object and person are trajectory space [1], smooth time-trajectories [37], non-linear optimization [60, 61], alternating least-squares [74], and locally-rigid motion [97].

These works utilize the movement of features within a video sequence to provide the data used to create a triangular mesh that is overlaid on top of the object or person. Not all of the authors use a connected mesh to estimate the configuration of the object, instead Taylor et al. [97] provide a triangle mesh soup that is overlaid on top of the non-rigid object (e.g. a newspaper). Fayad et al. [24] capture locally rigid patches of an object and combine them together to provide a global pose estimation. In contrast with NRSfM, the goal of this approach is to register the 3D non-rigid model with the incoming RGBD sequence, similar to the 2D-3D registration of Del Bue and colleagues [10, 9].

The work in this thesis differs from NRSfM in that the depth of the scene is calculated using a single frame while other authors calculate the depth of the scene using a video sequence. This approach uses mesh configurations from previous frames to estimate the current pose of the object. Unlike [5] [22], this approach does not use fiducial markers to track the current pose of the object and it is capable of handling non-rigid objects without texture.

In an effort to develop a system capable of handling real, unmodified objects, this work aims to remove the need for fiducial markers. This section of the thesis, described in Chapter 5, is inspired by, and based upon, the research of Pilet et al.

[78] which formulates the problem of 2D mesh estimation as energy minimization. In related work, Salzmann et al. [93, 92] describe an approach that operates in 3D but requires the restricted assumption of rigid triangles in order to reduce the dimensionality. Salzmann et al. [94] present a method for learning local deformation models for textured and textureless objects. The approach in this thesis extends this research by finding locally deformable 3D triangular meshes without fiducial markers whether intensity edges are present around the boundary of the object or not.

Chapter 3

Classifying clothing

3.1 Classification overview

The laundry classifying system involves two parts: isolating items from a pile of laundry and classifying isolated items. These two parts are described in detail in this chapter. The classification part of this chapter is broken up into two scenarios: classify in a hanging position and classify in a lying position. Each scenario used two different techniques / approaches on two different sets of data. The “hanging position” approach uses a small dataset, while the “lying position” approach uses a large dataset.

3.2 Isolation of clothing

To isolate an item from a pile of laundry, an overhead image is first segmented, and the closest foreground segment (measured using stereo disparity) is selected. Chamfering is used to determine the grasp point which is then used to extract the item from the pile in an fully automatic way using interactive perception. An overview of the process is shown in Figure 3.1.

3.2.1 Graph-based segmentation

The first step is to segment the overhead image into different regions. Felzenswalb and Huttenlocher's graph-based segmentation algorithm [27] is used because of its straightforward implementation, effective results, and efficient computation. This algorithm uses a variation of Kruskal's minimum-spanning-tree algorithm to iteratively cluster pixels in decreasing order of their similarity in appearance. An adaptive estimate of the internal similarity of the clusters is used to determine whether to continue clustering. Figure 3.2 shows the results of graph-based segmentation on an example 320×240 RGB color image taken in our lab using the default value for the scale parameter ($k = 500$). As can be seen, the segmentation provides a reasonable representation of the layout of the items in the pile. From this result, the foreground regions are determined to be those that do not touch the boundary of the image.

3.2.2 Stereo matching

Since the goal is to remove a single piece of clothing without disturbing the remaining items in the pile, the next step in the process is to determine which item is on top of the pile. While there are many monocular image cues that can provide a hint as to which object is on top, such as the size of the object, its concavity, T-junctions, and so forth, I rely upon stereo matching due to its efficiency, ease of implementation, and robustness. Stereo matching is the process in visual perception leading to the sensation of depth from two slightly different projections of an environment [96]. With rectified cameras, the difference in image coordinates between two corresponding points (the horizontal disparity) is inversely proportional to the

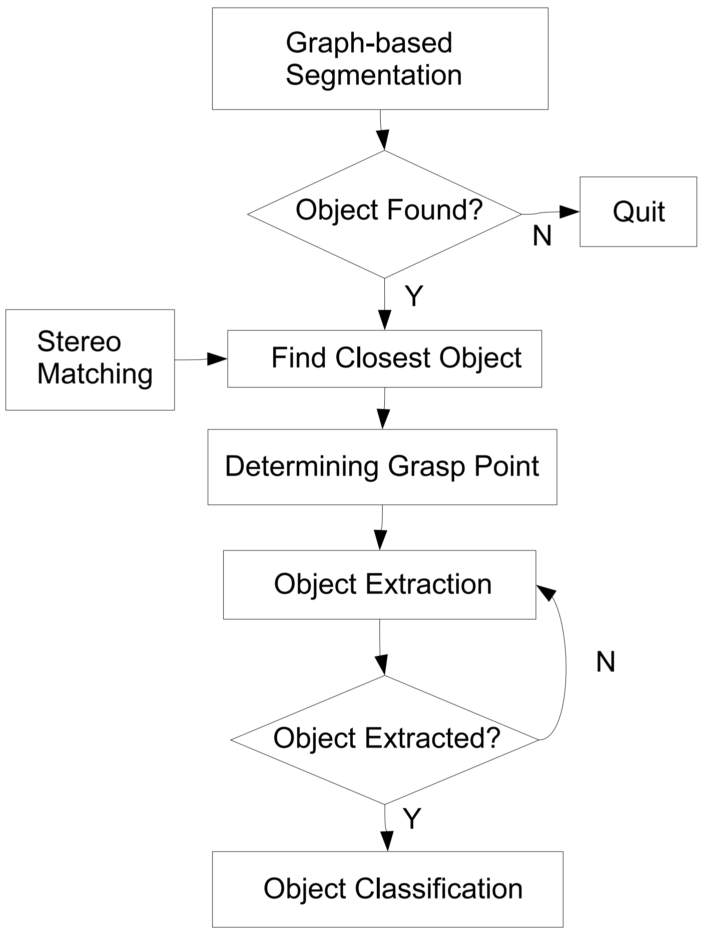


Figure 3.1: Overview of my system for vision-guided extraction of items from a pile of laundry.

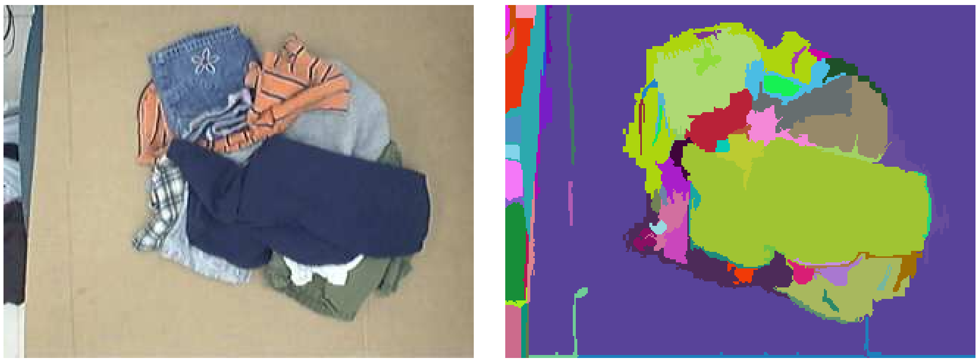


Figure 3.2: LEFT: An image taken by one of the overhead cameras in our setup. RIGHT: The results of applying the graph-based segmentation algorithm. Despite the over-segmentation, the results provide a sufficient representation for grasping an article of clothing.

distance from the camera to the point. An 11×11 window-based sum-of-absolute differences (SAD) stereo algorithm is implemented for its computational efficiency, utilizing MMX/SSE2 SIMD operations and a running sum sliding window to increase the speed of computation.

Due to misalignment of the cameras, reflections in the scene (non-Lambertian surfaces), and occlusion, the resulting disparity image is noisy. To reduce the effects of this noise, a left-right consistency check [33] is employed to retain only those disparities that are consistent in both directions, see Figure 3.3. Photometric inconsistency between the cameras is handled by converting to grayscale, followed by adjusting the gain of one image to match the other. After computing the disparities in this manner, the relative height of each segmented region is determined by the average disparity of all the pixels in the region. Among the foreground regions exceeding a minimum size (0.04% of the image), the one with the largest average disparity is then estimated as the item on top of the pile.

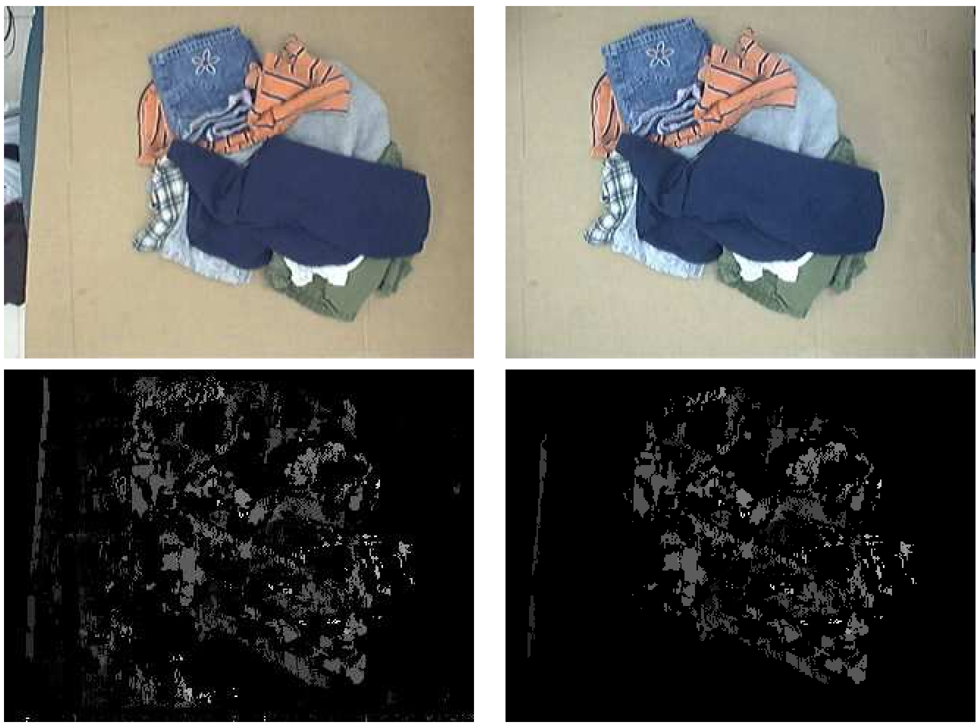


Figure 3.3: Top: A stereo pair of images taken by the overhead cameras in our setup, showing the large amount of photometric inconsistency. Bottom: The disparity image obtained by SAD matching with the left-right disparity check (left), and the result after masking with the foreground and removing small regions (right).

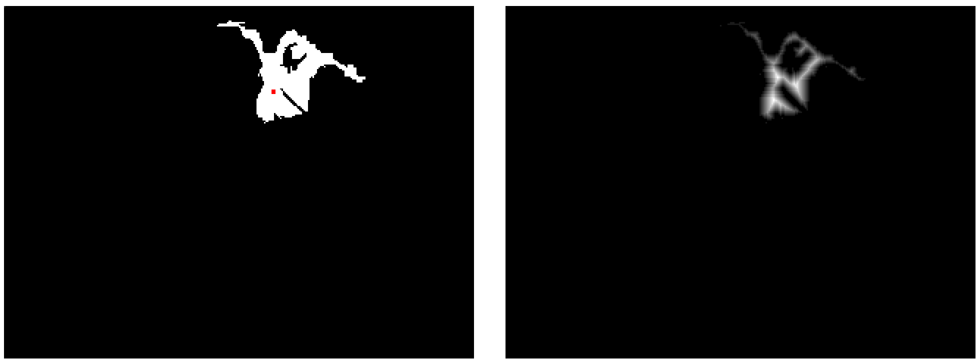


Figure 3.4: LEFT: The binary region associated with an article of clothing (orange shirt), with the grasp point (red dot) computed as the location that maximizes the chamfer distance. RIGHT: The chamfer distance of each interior point to the clothing boundary.

3.2.3 Determining the grasp point

Once the top item is determined, the next step is to determine the grasp point of the item. The approach of Saxena et al. [95] is not used due to the use of non-rigid and irregularly shaped objects. Instead, the 2D grasp point is calculated as the geometric center of the object, defined as the location whose distance to the region boundary is maximum. This point, which can be computed efficiently using chamfering [7], is much more reliable than the centroid of the region, particularly when the region contains concavities which is not uncommon in the scenarios considered herein. Figure 3.4 shows an example of the grasp point found by the maximum chamfer distance for an article of clothing. Note that only one grasp point is found due to the limitation of using a single robotic manipulator.

3.2.4 Extracting the item

Once the grasp point is found, the robot arm is moved over the pile of laundry so that the end effector is positioned above the grasp point. The arm then engages in a procedure that we refer to as *bobbing*. The arm is lowered to just above the estimated height of the item, then end effector is closed, the arm is raised and moved to the side. During this process the presence of the arm occludes the scene, making the images of the overhead cameras uninformative. Therefore, after completion of the process, the images before and after are compared to determine whether the desired item is extracted from the pile. Simple frame differencing with a threshold is used to make this decision. If it is determined that no item is extracted, then the procedure is repeated, this time with the end effector reaching down further. The process is repeated successively at increasing distances until either the item is removed or the end effector reaches the minimum height above the table (to avoid collision). If all attempts are unsuccessful, then the robot arm is returned to the "home" position, and the entire process begins again; in my experiments the robot is always successful using, for the most part, no more than two bobbing procedures. The entire procedure for extracting a single item is repeated until no more objects remain in the pile, which is determined via a threshold on the minimum size of the foreground regions in the segmented image (0.5% of the image).

Note that this *bobbing* procedure is a form of interactive perception. Because our sensors are not accurate enough to precisely compute the distance to the object, and because our gripper is not guaranteed to be oriented in the correct direction, it is virtually impossible to ensure success on the very first try. Moreover, the particular robot I used is not equipped with a force sensor, thereby increasing the difficulty of sensing the environment. To overcome this limitation in sensing, an interactive sensing (yet fully automatic) approach is adopted in which repeated interactions with the environment are used to simplify the problem of sensing.

3.3 Classification of clothing

3.3.1 Classification in a hanging position

The dataset that is used, in this section, consists of five short-sleeved shirts, five pants, five shorts, five long-sleeved shirts, five socks, and five underwear. Once the article is grasped and ready for classification, the arm lifts and swings to the side so that the article hangs freely without touching the table or ground. This open area is monitored by a third side-facing camera that captures both a "frontal view" and a "side view" image of the article, using the robot to rotate 90 degrees about the vertical axis between images. These two views are subtracted from a background image to obtain two binary silhouettes of the clothing. Figure 3.5 illustrates an example of the two views of an isolated article of clothing along with its binary silhouettes. Note that the terms "frontal" and "side" are arbitrary designations, since the robot grasps each object somewhat at random. What is important is that these two silhouttes represent the object's shape from two orthogonal directions (and, by symmetry, from the other two orthogonal directions by a mirror flip).

Features are extracted from the frontal and side images of the article of clothing in order to compare with other previously labeled images of clothing. Let I_Q be one of these two query images (either frontal or side), and let I_D be an image in the dataset. These images are compared using four different features to yield a match score:

$$\Phi(I_Q, I_D) = \sum_{i=1}^{N} w_i \cdot \frac{1}{m_i} f_i(I_Q, I_D), \tag{3.1}$$

where $N = 4$ is the number of features, $w_i \in \{0, 1\}$, and the features are given by

- $f_1(I_Q, I_D) = \mid a_Q - a_D \mid$, the absolute difference in area between the two silhouettes, where a_Q is the number of pixels in the query binary silhouette, and

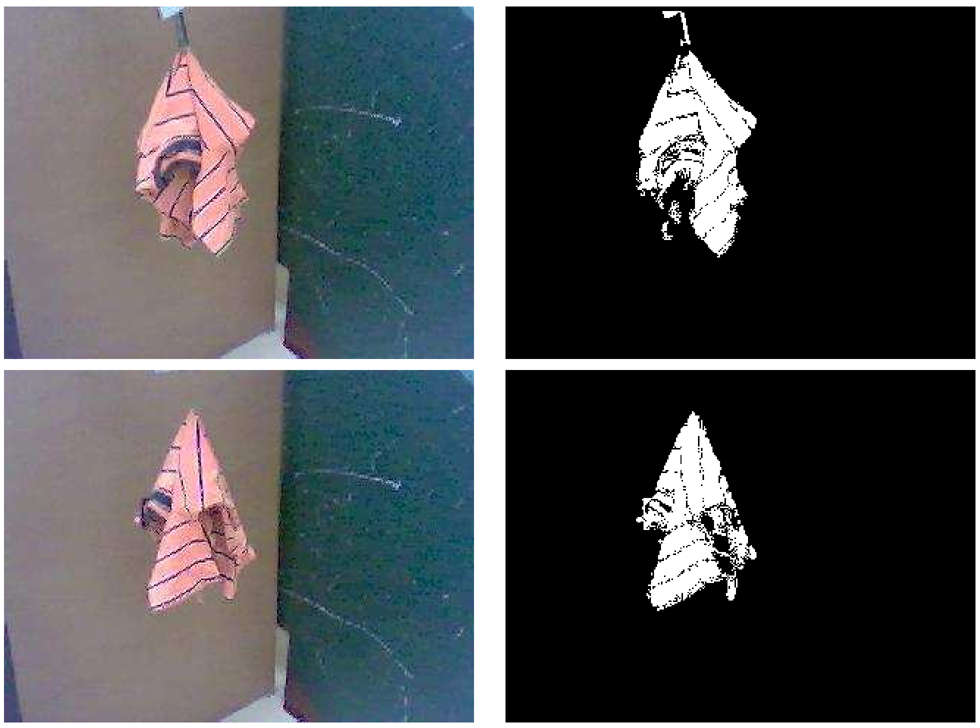


Figure 3.5: The front (top) and side (bottom) views of an isolated article of clothing to be classified (orange shirt). In each case, the original image is shown on the left, while the binary silhouette is shown on the right.

similarly for a_D;

- $f_2(I_Q, I_D) = \mid e_Q - e_D \mid$, the absolute difference in eccentricity between the two silhouettes, where $0 \leq e_Q \leq 1$ is the eccentricity of the query binary silhouette, and similarly for e_D;

- $f_3(I_Q, I_D) = H(BE_Q, BE_D)$, the Hausdorff distance between the Canny edges [11] of the two binary silhouettes;

- $f_4(I_Q, I_D) = H(CE_Q, CE_D)$, the Hausdorff distance between the Canny edges [11] of the original grayscale images.

To ensure proper weighting, each value m_i is the 95th percentile of f_i among all the images in the dataset (robust maximum).

Although color information would be a helpful cue for identifying particular items of clothing, it is not used in this work because color varies so widely within clothing categories. After calculating the match scores, the nearest neighbor algorithm (1-NN) is used to assign a category to the article of clothing. The 1-NN algorithm finds the category associated with the silhouette in the dataset whose match score to the test silhouette is minimum.

As will be seen in the experimental results, the accuracy of the above procedure is rather poor. This is due to the impoverished sensing that occurs when an article of clothing hangs freely from a single grasp point. To overcome this limitation, interactive perception is used. The process of capturing front and side views of an item is repeated ten times. In each iteration, the robot drops the item on the table, and the item is extracted again in a manner similar to that described above. The randomness of the dropping results in a new grasp point that, in general, bears little relationship to previous grasp points. These multiple grasp points provide the system with multiple front and side views of the article of clothing, thereby greatly increasing the chance of accurately classification.

3.3.2 Classification in a lying position

The proposed algorithm is called **L-M-H**, more specifically **L-C-S-H**, for **L**ow Level - **C**haracteristics - **S**election Mask - **H**igh Level. Low level refers to the features that are calculated from the images of each article using a Kinect sensor (e.g. 2D shape, 2D color, 2D texture, 2D edges, and 3D shape). Characteristics refers to the attributes that are found on generic articles of clothing (e.g. pockets, collars, hems, etc). Selection mask refers to a vector that describes what characteristics are most useful for each category (e.g. collars and buttons are most useful for classifying shirts). Characteristics and selection masks are considered to be mid-level layers that connect the low level features to high level categories. High level refers to the categories that are to be classified (e.g. shirts, socks, dresses). Figure 3.6 illustrates the path of the **L-C-S-H** process. The **L-C-S-H** algorithm is described in detail in later sections.

3.3.2.1 Background on algorithmic models with a mid-level layer

Latent semantic analysis The latent semantic analysis (LSA) [18] is another model similar to Bag of Words (BoW) [90], in the sense that it attempts to find the relationship between two variables, terms within a document and the type of document, by producing a "concept" space that relates the two variables. The "concept" space is comprised of latent semantic variables, hence the name, that relate observable data, like documents and terms, to each other. LSA uses matrix and vector operations along with singular value decomposition (SVD) to calculate the "concept" space. After producing the concept space, terms and documents can be compared

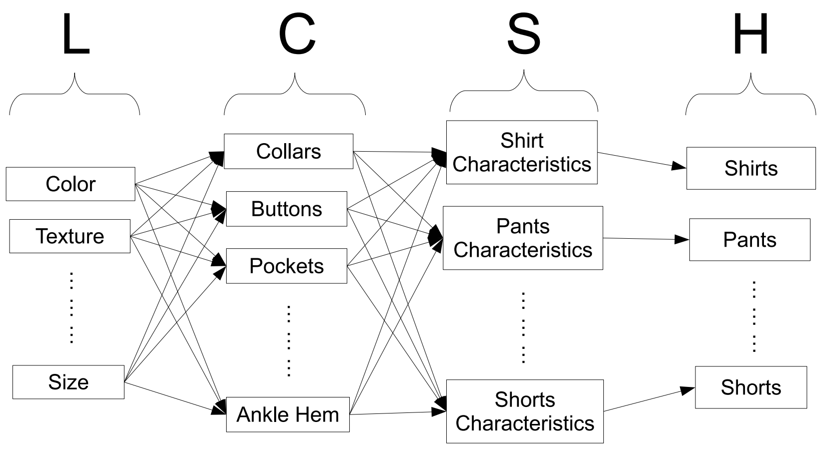


Figure 3.6: The **L-C-S-H** hierarchy with categories at the High level, binary vectors at the Selection levels, attributes at the Characteristic level, and features at the Low level.

with the same dimensionality and vector space using cosine similarity, see equation 3.2. If the cosine value is equal to zero, then the document (d) and the term (t) have no relationship. Otherwise, the higher the cosine value is to 1, the higher probability that the term is contained within the document.

$$\cos\theta = \frac{d \cdot t}{\|d\| \, \|t\|} \tag{3.2}$$

The first step of LSA is the same of BoW. The image is broken-up into patches and we convert each image patch into a feature vector with meaningful data. Each image patch is then considered to be a term, or word, that is counted within the image, or document. Next, all of the patches are placed within a dataset and k-means [63] is computed on the dataset to determine the vector locations of the unique patches. The unique patches are then considered to be the codewords within the codebook and all of the patches within the image are grouped according to the closest relationship to one of the unique patches. The difference in algorithms between BoW and LSA begins after this step.

The occurrences of each unique patch within each image, in the training set, are calculated and placed within a matrix, X, see equation (3.3). The element in location (i, j) within X is number of times patch i occurs in image j. Equations (3.4-3.5) represent a row vector and column vector of matrix X. A row vector consists of the relationship of patch i to all N images. The column vector consists of the relationship of the image j to all M patches. Any representation can be used for the element value within the co-occurrence matrix, but, most commonly, the frequency of occurrences is used. A connection, in vector space, between LSA and BoW would be the j^{th} column vector in the X matrix is the exact same vector as the total sum of the w_i vectors for the j^{th} document.

$$X = \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix} \tag{3.3}$$

$$t_i^T = \begin{bmatrix} x_{i,1} & \cdots & x_{i,n} \end{bmatrix} \tag{3.4}$$

$$d_j = \begin{bmatrix} x_{1,j} \\ \vdots \\ x_{m,j} \end{bmatrix} \tag{3.5}$$

Next, the decomposition of matrix X is needed to find the relationships of each patch with each image using L concepts. For this step, singular value decomposition (SVD) is used to calculate the singular values and left and right singular vectors of the matrix, see equation 3.6. Matrix U contains the left singular vectors that are used to create the transformed t_i^T, named $\hat{t_i^T}$, which gives the occurrence of patch

i in each of the l concepts. Matrix V^T contains the right singular vectors that are used to create the transformed d_j, named $\hat{d_j}$, which gives the relation of image j in each of the l concepts.

$$\begin{array}{ccccccc} X & & U & & \Sigma & & V_T \\ \begin{bmatrix} x_{1,1} & \cdots & x_{1,n} \\ \vdots & \ddots & \vdots \\ x_{m,1} & \cdots & x_{m,n} \end{bmatrix} & = & \begin{bmatrix} \begin{bmatrix} \\ u_1 \\ \\ \end{bmatrix} \cdots \begin{bmatrix} \\ u_l \\ \\ \end{bmatrix} \end{bmatrix} & \cdot & \begin{bmatrix} \sigma_1 & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \sigma_l \end{bmatrix} & \cdot & \begin{bmatrix} [\quad v_1 \quad] \\ \vdots \\ [\quad v_l \quad] \end{bmatrix} \end{array} \tag{3.6}$$

One of the important parts of this process is to decide how many concepts, singular values, will be used in transformation matrices. The k largest singular values are chosen to represent the relation between each patch to each image. The final matrix will be $X_k{=}U_k\Sigma_k V_k^T$. In order to compare patches or images in concept space, they must first be transformed by vector operations and then compared using cosine similarity. The transformation equations are as follows: $\hat{t_i^T}{=}t_i^T V_k\Sigma_k^{-1}$ or $\hat{t_i}{=}\Sigma_k^{-1} V_k t_i$ to transform a query patch into a $k \times 1$ vector and $\hat{d_j}{=}\Sigma_k^{-1} U_k^T d_j$ to transform a query image into a $k \times 1$ vector.

Probabilistic latent semantic analysis The probabilistic latent semantic analysis (pLSA) [42] [43], also called aspect model [44], is an extension of LSA that utilizes a joint probability model to map the relationship of words to documents using latent variables ("concepts"), $z \in Z = \{z_1, \ldots, z_k\}$. The two models use the conditional probability between the words, "concepts", and documents as well as the independent probability of "concepts" and documents, depending on the symmetry of the model.

Equations (3.7-3.8) display the probabilities that are used in each model along with the formulas used for each model. One shortcoming from using the asymmetric model is the cardinality of “concepts” could become a bottleneck between the words and documents. If the number of words and the number of documents are greater the number of “concepts”, then the mapping between the two observable variables will be underfitted, a common problem when determining the number of nodes in the hidden layer for artificial neural networks [14] [23] [34] [59].

$$
\begin{aligned}
P(d,w) &= P(d)P(w \mid d), P(w \mid d) = \sum_{z \in Z} P(w \mid z)P(z \mid d) && (3.7)\\
P(d,w) &= \sum_{z \in Z} P(z)P(d \mid z)P(w \mid z) && (3.8)
\end{aligned}
$$

Model fitting with EM In order to calculate the maximum likelihood estimator for the “concepts” model, the Expectation Maximization (EM) algorithm [19] is used. Expectation maximization is an iterative method for finding estimates where the model depends on latent variables, “concepts”. The E step is calculated and shown in equation 3.9. The M step is calculated and shown in equations 3.10-3.12.

$$
\begin{aligned}
P(z \mid d,w) &= \frac{P(z)P(d \mid z)P(w \mid z)}{\sum_{z' \in Z} P(z')P(d \mid z')P(w \mid z')} && (3.9)\\
P(w \mid z) &\propto \sum_{d \in D} n(d,w)P(z \mid d,w) && (3.10)\\
P(d \mid z) &\propto \sum_{w \in W} n(d,w)P(z \mid d,w) && (3.11)\\
P(z) &\propto \sum_{d \in D} \sum_{w \in W} n(d,w)P(z \mid d,w) && (3.12)
\end{aligned}
$$

The EM algorithm iterates through the E and M step until the multivariate probabilities come to a convergence. The initial conditional probabilites that connect each word to each “concept” and each document to each “concept” are randomly chosen. The M step is highly dependent on the occurrence matrix described in the previous section. The $n(d, w)$ variable is the number of instances that word w and document d occur over the training set. This algorithm is commonly chosen for categorizations when the latent variables (“concepts”) are unobserved.

3.3.2.2 Clothing dataset in a lying position

The dataset that is used in this section, consists of over 200 articles of clothing in over 1000 total configurations. The entire dataset is separated into seven categories and more than three dozen subcategories. Each article of clothing is collected from real laundry hampers to better capture actual items encountered in the real world. Each one is laid flat on a table in a canonical position and dropped on the table in a crumpled position from four random grasp points for a total of five instances per article. Figure 3.7 illustrates seven instances of clothing for each of the seven categories that are used.

The dataset contains 2D color images, depth images, and 3D point clouds that are captured using a Kinect sensor. For this thesis, the different types of items, such as shirts, cloths, pants, shorts, dresses, socks, and jackets that will be utilized might be encountered by a domestic robot involved in automating household laundry tasks. The articles of laundry are captured in this manner to test the possibility of developing a system capable to classifying clothes lying on the floor. Figure 3.8 illustrates an example of a single color image, a single depth image, and a single point cloud for one article.

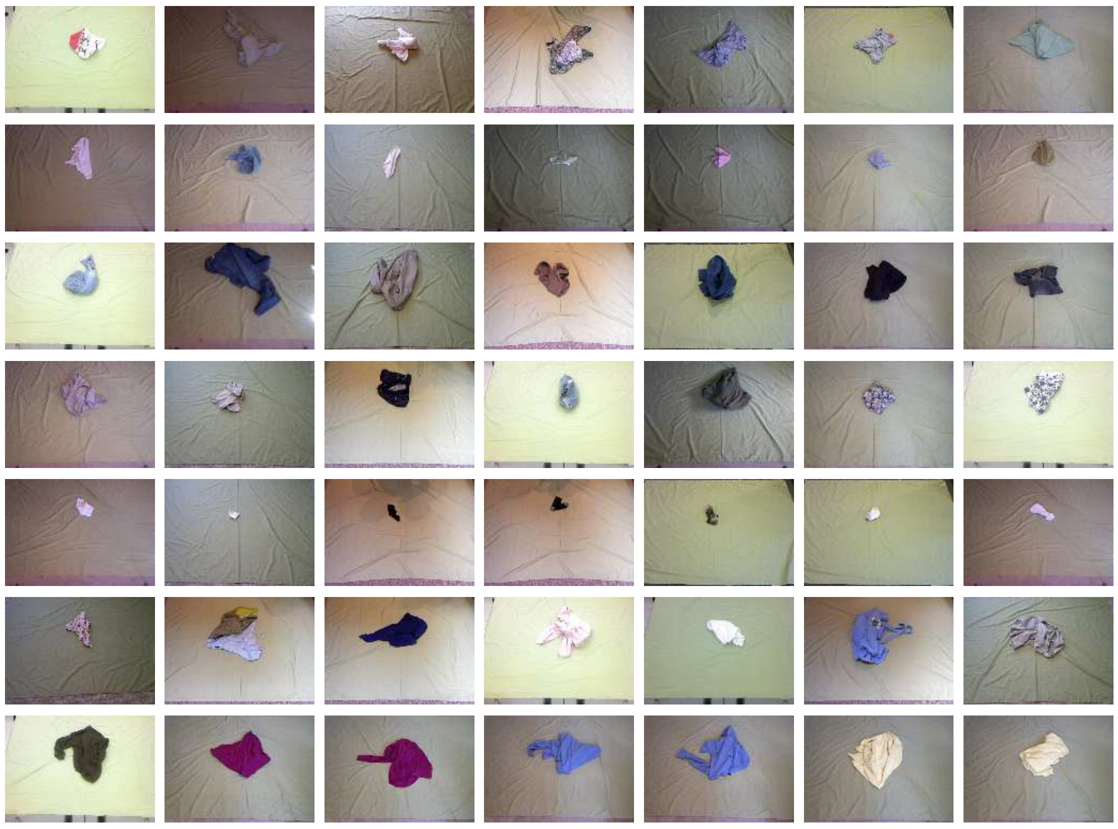


Figure 3.7: From Left to Right: Seven examples of clothing for each of the seven categories. From top to bottom: shirts, cloths, pants, shorts, dresses, socks, and jackets.

Shirts (85)	Cloth (30)	Pants (25)	Shorts (25)
Dress (10)	Socks (22)	Jacket (5)	

Table 3.1: List of seven unique categories and the number of articles for each category. The number of articles contained in each category is indicated in parantheses.

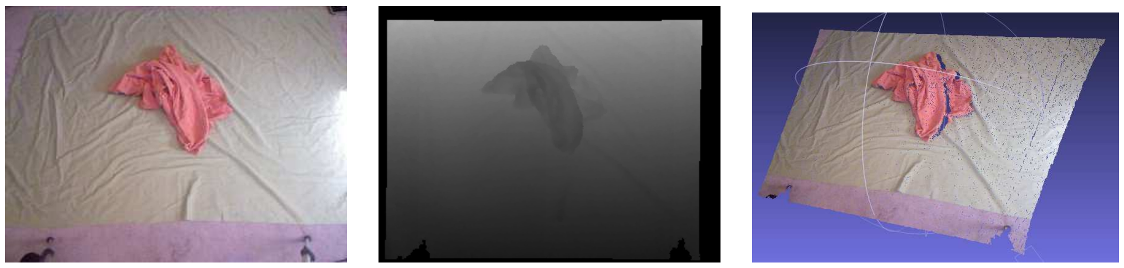


Figure 3.8: Data to be used in my approach. From left to right: Color image, depth image, and a point cloud of a t-shirt.

3.3.2.3 Low level features

The **L** component of the **L-C-S-H** approach uses the low level features to estimate if the article does or does not have a particular characteristic. The low level features that are used in this approach consist of Color Histogram (CH), Histogram of Line Lengths (HLL), Table Point Feature Histogram (TPFH), boundary, Scale-Invariant Feature Transform (SIFT), and Fast Point Feature Histogram (FPFH). Each low level feature will be described in detail in further subsections. The low level features that are chosen for each article of clothing are used to determine / classify characteristics, shown in Table 3.2. Nine out of the 27 characteristics can be calculated from shape information, seven out of 27 can be calculated from color information, five out of 27 can be calculated from texture information, and ten out of 27 can be calculated from edge information. The dataset that is used consisted of five instances of each article, see section 3.3.2.2. In order to combine the low level features of all five instances into a single value or histogram, each value is calculated by averaging each individual value along with its neighbors, in the case of the histogram. Equation (3.13) is the computation used to combine all five instances of each article.

$$C_j = \frac{1}{N}\sum_{j=1}^{N} v_j + \frac{1}{2}(v_{j-1} + v_{j+1}) \tag{3.13}$$

where C_j is the averaged score of the histograms, in position j, N is the number of instances of each article ($N = 5$ is used in this thesis), v_j is the current value within the histogram along with v_{j-1} and v_{j+1} to be the immediate neighbors to the left and right of the current value, respectively.

For the part of the algorithm that converts from low level to characteristics, the low level features, described in the following subsections, are compared to the various characteristics. Since the characteristics are binary values, libSVM [12] is used to solve the two-class problem. Each low level feature is used to determine if the characteristic is in class 1 or 2. Class 1 contains positive instances and class 2 contains negative instances.

For an article of clothing, an RGB image and a raw depth map is captured, from which a 3D point cloud is derived. Background subtraction is performed on the RGB image to yield an image of only the object. The background subtraction is performed using graph-based segmentation [27], see section 3.2.1 for a description and see Figure 3.9 for an example of the segmentation applied to the current dataset. Once the object is isolated within the image, multiple features are calculated from the RGB image and the 3D point cloud. These features, which are discussed in later sections, capture 2D shape, 2D color, 2D texture, 2D edges, and 3D shape for both global and local regions of the object.

3.3.3 Global features

3.3.3.1 Color Histogram (CH)

A CH is a representation of the distribution of the colors in a region of an image, derived by counting the number of pixels with a given set of color values. CH are chosen in this work because they are invariant to translation and rotation

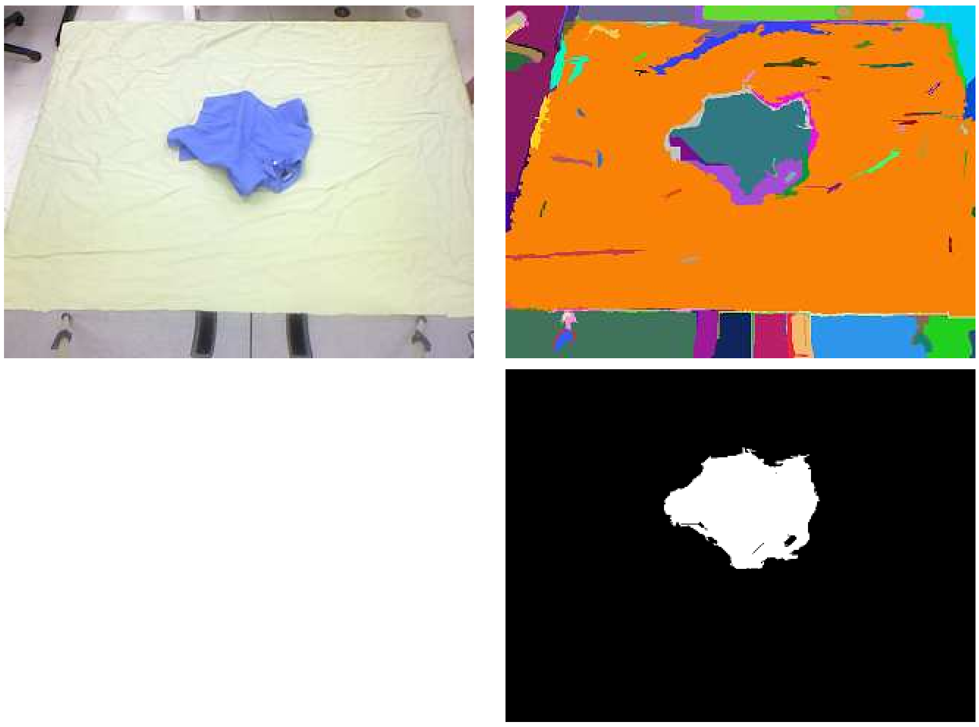


Figure 3.9: LEFT: An image taken by the overhead camera in my setup. MIDDLE: The results of applying the graph-based segmentation algorithm to locate the foreground. RIGHT: The binary image represents the location of the foreground object.

about the viewing axis, and for most objects they remain stable despite changes in viewing direction, scale, and 3D rotation. CH is used to distinguish, for example, between lights and darks, as well as denim. The CH of each object is computed, in HSV color space. The HSV color space is computed by converting the RGB space to hue, saturation, and value. Equations (3.14)-(3.18) display the equations needed to compute the hue of a pixel. In the case of C=0, the first bin in the hue color channel is incremented to keep track of all undefined instances. Equations (3.19)-(3.20) display the equations needed to compute the value and saturations of a pixel, respectively. 15 bins are used for hue color channel and 10 bins for the saturation and value color channels, leading to 35 total bins. The hue ranges from 0 $\rightarrow$ 360 degrees, the saturation and the value both range from 0.0 $\rightarrow$ 1.0. Figure 3.10 illustrates the difference in an article of clothing with light colored and dark colored fabric.

$$M = max(R, G, B) \tag{3.14}$$

$$m = min(R, G, B) \tag{3.15}$$

$$C = M - m \tag{3.16}$$

$$H' = \begin{cases} \text{undefined,} & \text{if C=0} \\ \frac{G-B}{C} \text{ mod } 6, & \text{if M=R} \\ \frac{B-R}{C} + 2, & \text{if M=G} \\ \frac{R-G}{C} + 4, & \text{if M=B} \end{cases} \tag{3.17}$$

$$H = 60^o x H' \tag{3.18}$$

$$V = max(R, G, B) \tag{3.19}$$

$$S = \frac{C}{V} \tag{3.20}$$

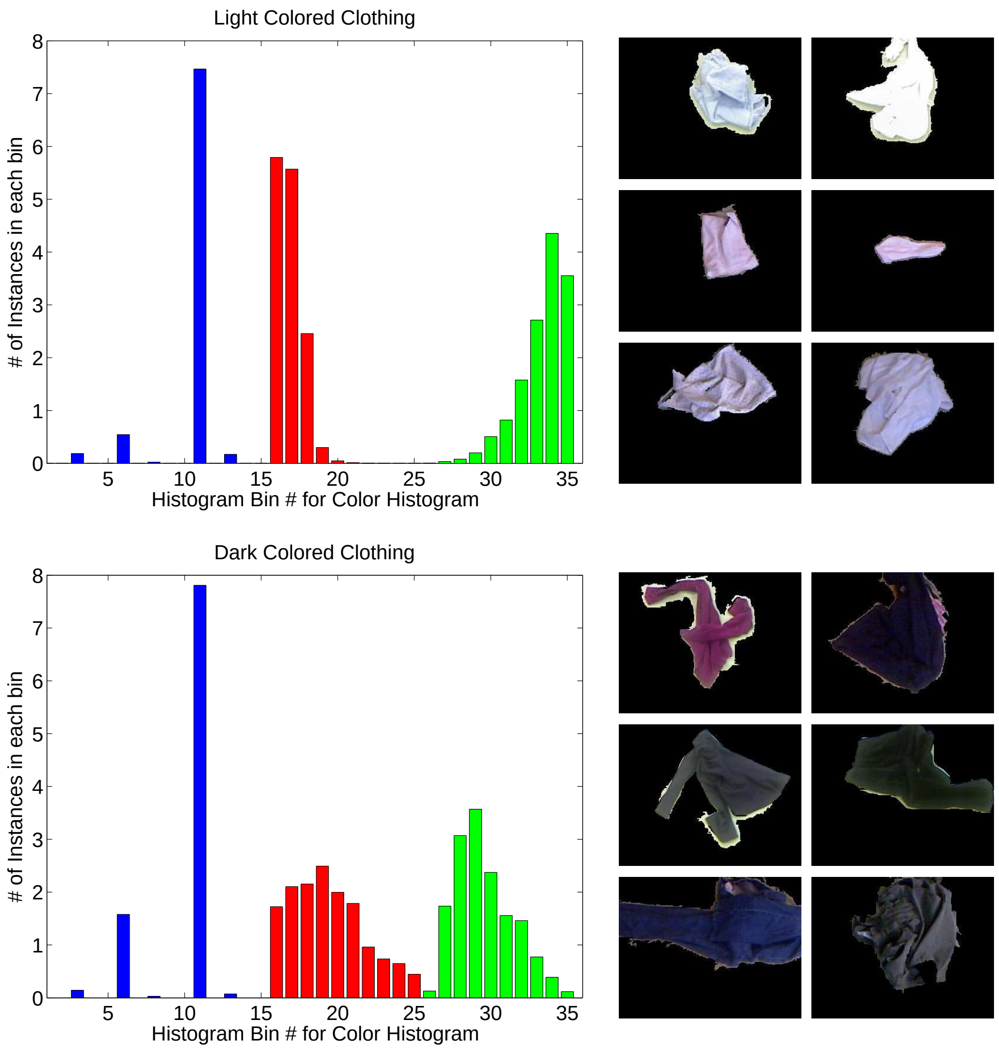


Figure 3.10: Resulting histograms of light (top) and dark (bottom) colored clothing. Each plot (left) shows the histogram of the HSV color space, which consists of three parts: hue (blue), saturation (red), and value (green). For each characteristic, the 6 examples (right) used to compute the histogram are shown.

3.3.3.2 Histogram of line lengths (HLL)

The histogram of line lengths (HLL) is a novel feature that is introduced to help distinquish between stripes, patterns, plaid, etc. For this, the object image is used as before (after background subtraction) to compute the Canny edges [11], then eroded with a 3x3 structuring element of ones to remove effects of the object boundary. Then the length of each Canny edge is computed using Kimura et al. [52] method. The Kimura et al. method uses diagonal movement (N_d) and isothetic movement (N_o) to accurately calculate the length of a pixelated edge within an image, see equation (3.21). Diagonal movements (N_d) refer to moving from the center pixel to one of the four corners within a 3x3 window and isothetic movements (N_o) refer to moving directly left, right, up, or down, see Figure 3.11. These pixel lengths are used to compute a histogram of 20 bins that range from 0 pixels to 1000 pixels, so each bin captures lengths within 50 pixels. Lengths greater than 1000 get mapped down to the last bin. Figure 3.12 illustrates the difference in an article of clothing with no texture, with stripes, and with plaid.

$$L = [N_d^2 + (N_d + N_o/2)^2]^{1/2} + N_o/2 \tag{3.21}$$

3.3.3.3 Table point feature histogram (TPFH)

The TPFH consists of a 263 dimension array of float values that result from three 45 value subdivisions, that are calculated from extended Fast Point Feature Histograms (eFPFH), and 128 value subdivision for table angle information. This feature is a variant on the Viewpoint Feature Histogram (VFH) [85]. The eFPFH values are calculated by taking the difference of the estimated normals of each point within the objects pointcloud and the estimated normal of the objects pointcloud

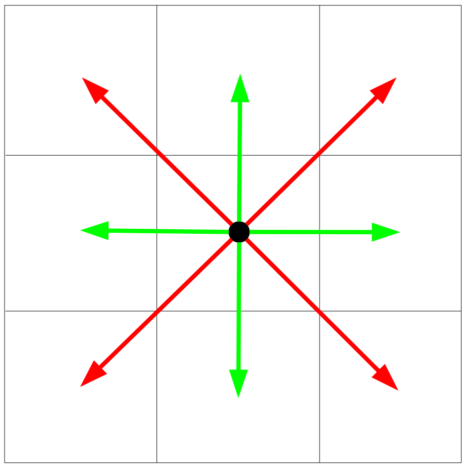


Figure 3.11: Example illustration of diagonal movements and isothetic movements within a $3x3$ window. The red arrows refer to diagonal movements and green arrows refer to isothetic movements.

center point. The estimated normals of each point and the center point are calculated by projecting them on the XY, YZ, and XZ plane. The differences between the two normals are labeled β, θ, and ϕ. These three values, ranging from $-90 \rightarrow +90$ degrees (4 degrees each bin), are placed within bins in the three 45 value histograms of the eFPFH. Figure 3.13 illustrates the graphical representation of the normals and how they are calculated. The 128 value table histogram is computed by finding the angle, α, between each normal vector and the translated central table vector for each point. The central table vector is translated to the same point as the normal that is currently being computed. Figure 3.13 illustrates the difference in direction for each vector. In TPFH, the eFPFH component is a translation, rotation, and scale invariant, while the table component is only a scale invariant, 3D representation of the local shape. Figure 3.14 illustrates an example of how the TPFH values are visually different in two separate categories.

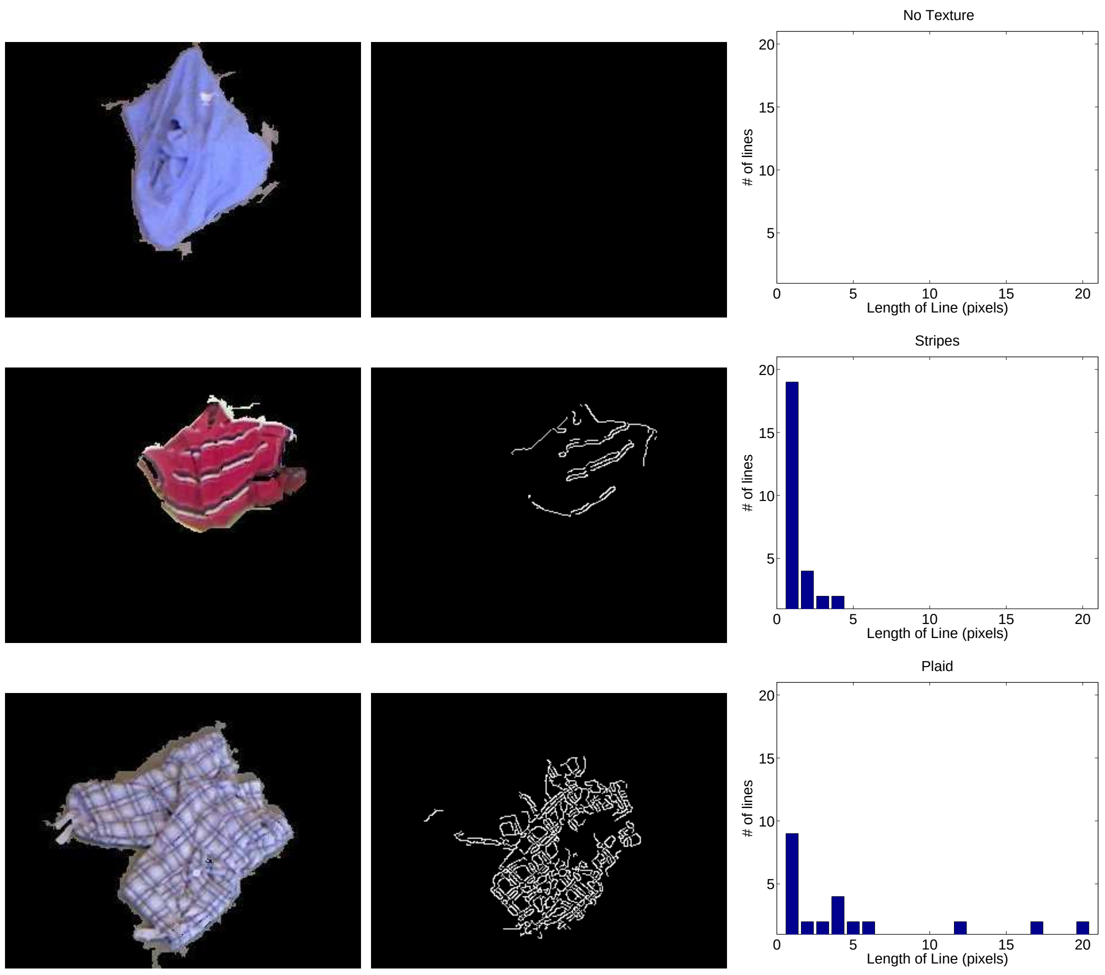


Figure 3.12: Resulting histograms of an article of clothing with no texture (top), with stripes (middle), and with plaid (bottom). Each row contains the original image (left), the image after using Canny edge detector (middle), and the histogram of line lengths (right).

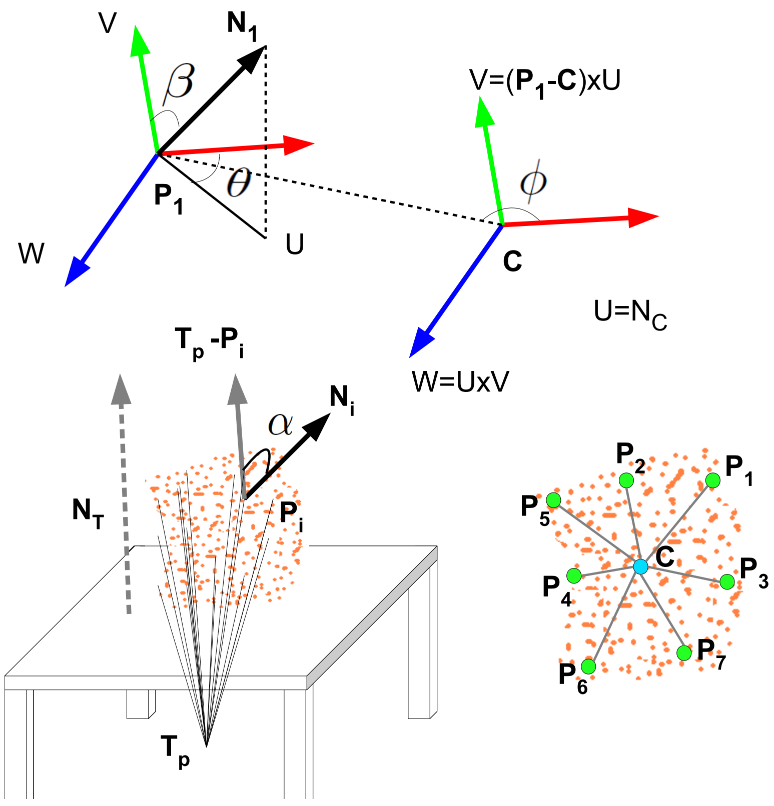


Figure 3.13: Graphical representation of the two components of TPFH. Top: eFPFH component with β, θ, and ϕ shown. V, W, and U represent the coordinates of the point C (centroid), with respect to the normal of C and the position of C and P_1. N_1 is the normal of point P_1. Bottom: Table component with α shown. T_P is the centroid of the table, while N_T is the normal of the table. N_i is the normal of point P_i and $P_1 \rightarrow P_7$ are arbitrary points in the cloud surrounding point C.

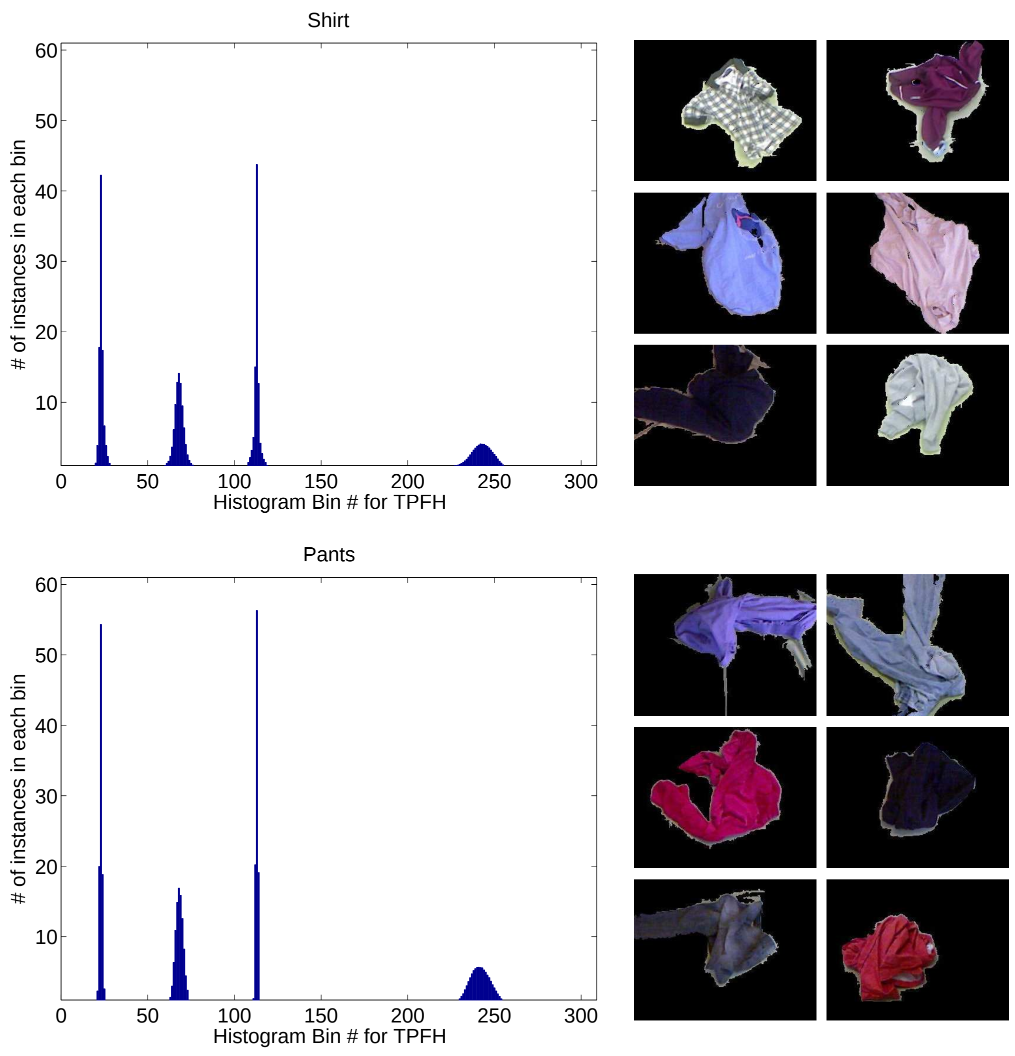


Figure 3.14: Example of the visual differences in global shape information between two different categories, namely shirts (top) and pants (bottom). Each row contains the average histogram of the TPFH (left) and 6 examples of each category used to calculate the average histogram (right). Shirts have a relatively lower # of instances in the first and third peak than pants. This example shows why the TPFH feature is useful in classifying clothing.

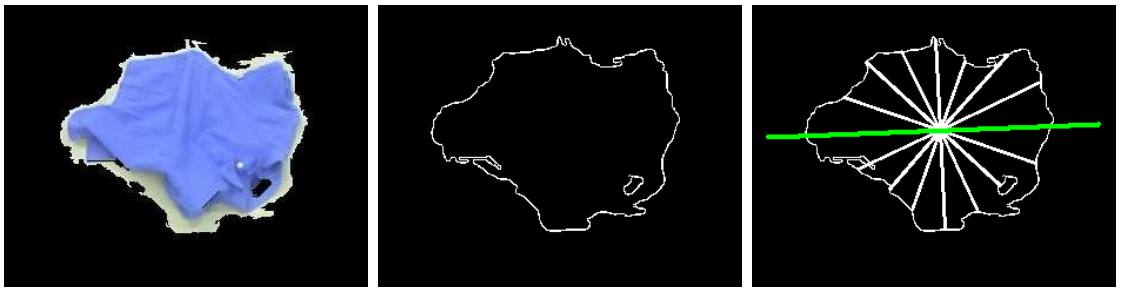


Figure 3.15: LEFT: Article segmented from background. MIDDLE: The binary boundary of the article. RIGHT: The boundary of the article along with the lines connecting the centroid to the boundary, whose distances are used in the feature vector. The green line represents the major axis found by PCA.

3.3.3.4 Boundary

The boundary feature captures 2D shape information by storing the Euclidean distances from the centroid of each article to the boundary. First, the centroid of each binary image is calculated containing the object (after foreground / background segmentation). Then, starting at the angle of the major axis found by principle components analysis, 16 angles that range from 0 to 360 (i.e., $0, 22.5, 45, 67.5, \ldots, 315, 337.5$) are calculated around the object. For each angle, the distance from the centroid to the furthest boundary pixel is measured, see Figure 3.15.

3.3.4 Local features

3.3.4.1 Scale Invariant Feature Transform (SIFT)

The Scale Invariant Feature Transform, SIFT [62], descriptor is used to gather useful 2D local texture information. The SIFT descriptor locates points on the article (after foreground / background segmentation) that provide local extremum when convolved with a Gaussian function. These points are then used to calculate a histogram of gradients (HoG) from the neighboring pixels. The descriptor consists of a

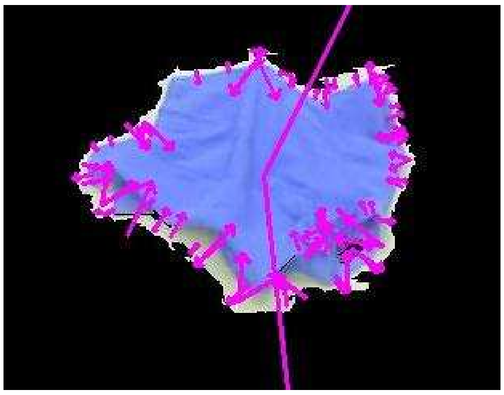


Figure 3.16: Resulting SIFT features overlayed on the image. The arrows represent the orientation and scale (magnitude) of each SIFT descriptor.

128 value feature vector that is scale and rotation invariant. Figure 3.16 illustrates the SIFT feature points found on an article. After all of the feature points are found on an article, the SIFT descriptors are placed within a Bag-of-Words model [101], described in detail in section 3.3.4.3, to calculate 100 codewords. These codewords provide each article with a 100 element feature vector that represents the local texture information.

3.3.4.2 Fast Point Feature Histogram (FPFH)

The Fast Point Feature Histogram, FPFH [84], descriptor is used to gather local 3D shape information. The FPFH descriptor utilizes the 3D point cloud and foreground / background segmentation for each article and segments the article points from the background points of the point cloud. For each 3D point, a simple point feature histogram (SPFH) is calculated by taking the difference of the normals between the current point and its k neighboring points with a radius r. Figure 3.17 illustrates an example of a 3D point along with its neighbors. The radius is precomputed for each point cloud to best capture local shape information. Once all of the SPFHs are

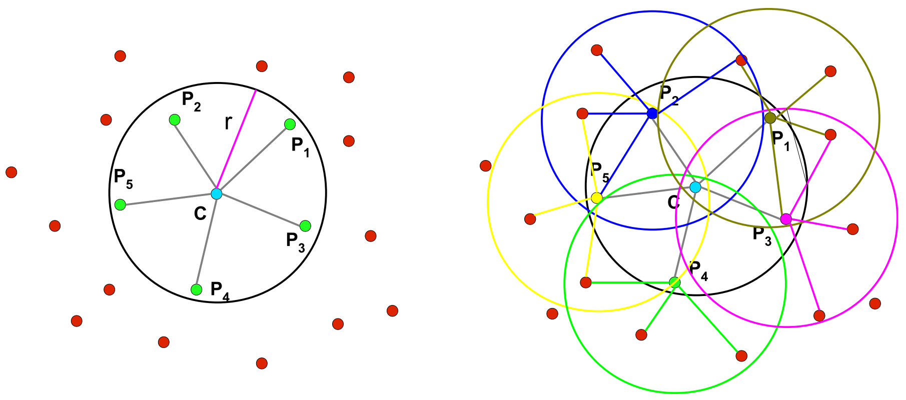


Figure 3.17: LEFT: Example of the SPFH of a single point. An area of radius r encircles 5 neighboring points for this example. RIGHT: An example of the FPFH of the current point along with surrounding SPFHs used in the reweighting scheme. Each neighboring SPFH is color coded to represent the surrounding connections within the various regions.

computed, the FPFH descriptor of each point is found by adding the SPFH of that point along with a weighted sum of the k neighbors, see equation (3.22). FPFH is a histogram of 33 values, i.e. three sets of 11 bins for the three different orthogonal planes of XY, YZ, andXZ.

$$FPFH(p) = SPFH(p) + \frac{1}{k}\sum_{i=1}^{k} w_i SPFH(i) \tag{3.22}$$

3.3.4.3 Bag of words model

The bag of words (BoW) [90] model is originally implemented for natural language processing (NLP) [13] using a vector space model, but more commonly used in computer vision applications [79]. The goal of this model is to identify documents by categorizing them based on the words within the document, while the ordering of the words is ignored. Each document is broken up based on which words are used in the document and the number of occurrences that each word appeared in the document.

This model is extended to the computer vision field for object categorization. The model uses the same underlying process for categorization, but the words are now patches within an image. These patches are chosen based on the type of categorization that is being implemented. For example, a 200×200 image can be broken into 100 equally sized 20×20 patches across the image. A patch within an image is similar to a word within a document.

Image feature extraction In order to transform each patch into a single entity, a feature descriptor is chosen to represent each patch within the image. The feature descriptor can range from color values, texture gradients, edge locations, or a histogram. A well-known robust feature descriptor histogram is Scale-invariant feature transform (SIFT) [62]. SIFT converts an image patch into a 128-dimensional vector. This vector can be considered to be a word representation within a document.

Clustering unique patches into "codewords" The next step is to determine how many unique patches, "codewords", are in the image. Several of the vectors, representing each patch, may have similar aspects or values and can be considered to be the same patch. "Codewords" are analogous to topics within a latent variable model, described in later sections. The goal is to group like-patches together to calculate unique patch locations within the vector space. Clustering is used to find groups of patches and calculate vector means to represent unique patch locations. A common clustering method that is used for this calculation is k-means clustering [63]. The value chosen for k determines the number of "codewords" that map image patches to image categories.

Figure 3.18 illustrates a simple example of an image broken-up into various patches. V unique patch groups are extracted from the various image patches to

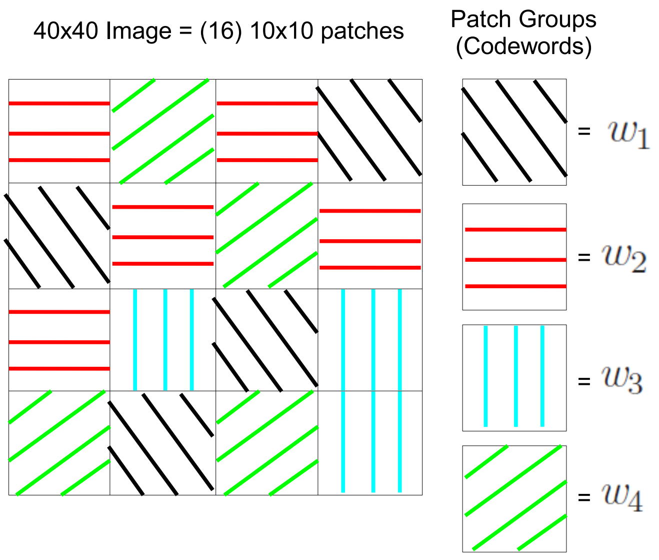


Figure 3.18: Bag-of-words model example to illustrate the grouping of patches within an image.

create V "codewords", w_i as $i = 1 \ldots V$, ($V = 4$ in the case of Figure 3.18). These V "codewords" make up the "codebook", which is a collection of all "codewords" within the training set $\{w_1, w_2, \cdots, w_V\}$. The "codebook" consists of V "codewords" that are $Vx1$ vectors. The i^{th} vector is represented by (V-1) 0's and a 1 in the i^{th} position, $w_i = 1$ and $w_j = 0$ for $i \neq j$.

Converting "codeword" information into vector space After the "codebook" is completed, each image in the dataset, both training and testing, is converted to a matrix W, which consists of N vectors to represent the N patches with the image. In the case of Figure 3.18, the "codebook" will comprise of four "codewords", $w_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$, $w_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$, $w_3 = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$, and $w_4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}$ and the resulting representation of the image will be $W = \{w_2, w_4, w_2, w_1, w_1, w_2, w_4, w_2, w_2, w_3, w_1, w_3, w_4, w_1, w_4, w_3\}$, as the rows of the image are concatenated together.

Finally, a classifier is chosen to test and compare the model. Simple classifiers, like naïve Bayes [82] and support vector machines (SVM) [12], are commonly used to group an image with N patches into one of several categories. In the case of a naïve Bayes classifier, a categorization decision would consist of the maximum total product of the probability of each category, $p(c)$, and the probabilities of each "codeword" (w_i) appearing within each particular category, $p(w_i \mid c)$, see equation 3.23.

$$c^* = \arg\max_c p(c) \prod_{i=1}^{N} p(w_i \mid c) \tag{3.23}$$

3.3.4.4 Final input feature vector

Once the features are computed, the global features are concatenated to create a histogram of $35 + 20 + 263 + 16 = 334$ values. For local features, SIFT and FPFH are calculated separately through the bag-of-words model to get two 100-element histograms of codewords. Being concatenated, this yields 200 values for the local features. Then being concatenated with global features yields 534 values, which are then fed to the multiple one-versus-all SVMs. Each of the local and global level features are scaled from $0 \rightarrow 100$ before concatenation to allow for equal weighting among each type of feature.

3.3.5 Mid-level layers

3.3.5.1 Characteristics

The **C** component of the **L-C-S-H** approach uses the characteristics found on everyday clothing that is best suited for each category. A binary vector of 27 values are used that correspond to each characteristic used to learn what attributes are needed to separate the differences between shirts, pants, dresses, etc. The 27 characteristics that are chosen are shown in Table 3.2, along with the number of instances and definition of each characteristic in the dataset. The training procedure used for classifying characteristics is the 5-fold cross validation (CV) [20], which is described in section 3.6. Figure 3.19 illustrates the percentage of how much each characteristic is used per category.

3.3.5.2 Selection Mask

The **S** component of the **L-C-S-H** approach uses the characteristics to determine which subset is best suited for each category. The selection masks are stored as a binary value; 0 or *FALSE*, means that the category does not need that particular characteristic, 1 or *TRUE*, means that the category does need that particular characteristic.

Then, the process determines which characteristics have a higher importance and which have a lower importance for each category. Therefore, one characteristic is zeroed out at a time and viewed the resulting percentage over the rest of the characteristics. The percentage is determined by comparing the characteristic vector for each article against mean vectors for each category, which are calculated in section 3.3.6. If the current percentage is higher or equal to the previously calculated percentage, then that particular characteristic is zeroed out for the following iteration(s).

The next iteration repeated the same process of zeroing out one characteristic at a time along with having the previously chosen characteristics zeroed out. After permanently zeroing out several features, the resulting classification percentage began to increase to a peak. Then the peak began to decrease after the remaining characteristics are zeroed out. Next, a binary vector is created for each category that

Collar (21)	the part of a garment that fastens around or frames the neck
Front Pockets (39)	pockets on the front of any garment
Back Pockets (26)	pockets on the back of any garment
Side Pockets (8)	pockets on the side of any garment
Elastic Band (32)	any sort of stretchy band, mainly around the waist
Front Zipper (28)	zippers on the front of any garment
Top buttons (30)	button on the top of any garment, like polo shirts or pants
Belt loops (25)	loops around the waist for a belt
Top brackets (10)	brackets on the top of any garment, like pants
Graphic pictures (47)	any types of pictures or logos on a garment
Graphic texts (7)	any type of text or word on a garment
Dark Colored (70)	clothing that is darker in color, close to black
Colored (94)	any clothing with color that is not whites or darks
White Colored (38)	any clothing that is totally white
V-neck (7)	neckline protrudes down the chest and to a point
Denim (13)	a rugged cotton twill textile
Plaid (5)	a pattern consisting of crossed horizontal and vertical bands
Striped (11)	a pattern consisting of horizontal or vertical bands
Patterns (18)	any repetitious design that doesn't fall under plaid or striped
Round neck (73)	neckline that encircle the neck in a curved shape
Ankle hem (30)	a hem located in the ankle region
Thigh hem (28)	a hem located in the thigh region
Inseam (52)	the seam that binds the length of the inner trouser leg
Shin hem (15)	a hem located in the shin region
Wrist hem (26)	a hem located in the wrist region
Shoulder hem (15)	a hem located in the shoulder region
Bicep hem (59)	a hem located in the bicep region

Table 3.2: List of 27 unique characteristics, their definitions, and the number of instances for each characteristic that are chosen to differentiate categories of clothing. The number of articles containing each characteristic is indicated in parantheses.

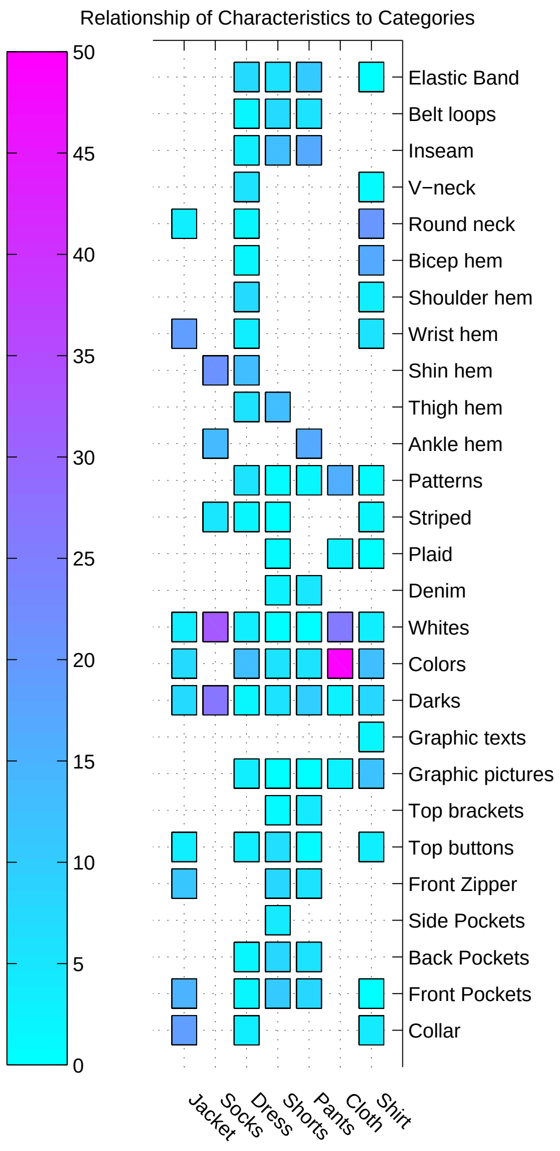


Figure 3.19: A graphical representation of the percentage of characteristics that are contained within each category. The color scheme changes from red to purple as the percentage increases. Light red colors represent small percentages. Dark purple colors represent high precentages. Blank areas represent a percentage of zero.

contained a 1, or *TRUE*, if the characteristic yielded a higher classification percentage and a 0, or *FALSE*, if the characteristic yielded a lower classification percentage.

3.3.6 High level categories

The **H** component of the **L-C-S-H** approach uses the characteristic vectors that correspond to the articles within the training set to average the binary vectors of each category together to create a mean vector, as a descriptor, for each category. In other words, all shirts have averaged their characteristic vectors together, all dresses have averaged their characteristic vectors, and so on to create seven unique mean vectors, one for each category. Then the selection mask is multiplied by the characteristic vector to zero out any characteristics with a low classification percentage.

3.4 Extraction / isolation experiment

Figure 3.20 shows the results of the system after different steps of the extraction and isolation procedure. The system removes items from a pile of laundry one at a time by identifying a grasp point in the closest region, then deploying the robotic arm to that location, *bobbing* for the item, and moving the item to another part of the workspace. Once the item is extracted and isolated, the system interacts with it by rotating about a vertical axis in order to gain both front and side views of the item using the side-facing camera. Two 2D silhouettes of the item are created for each grasp point; by repeating this process ten times, 20 silhouettes are obtained for each item (ten front views and ten side views). The procedure successfully removed all items from the pile one at a time.

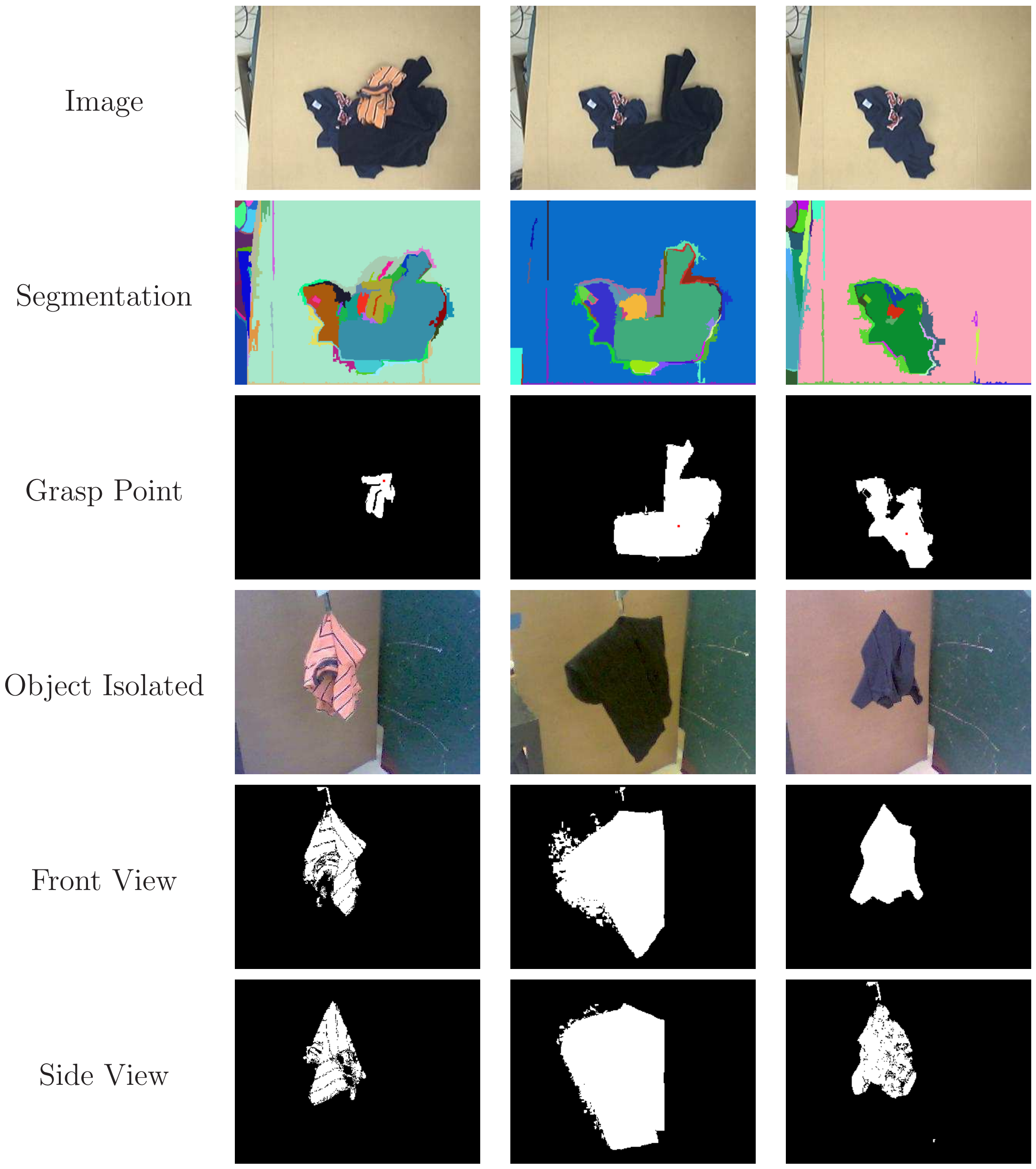


Figure 3.20: Three example experiments of the extraction and isolation process. From top to bottom: The image taken by one of the downward-facing stereo cameras, the result of graph-based segmentation, the object found along with its grasp point (red dot), the image taken by the side-facing camera, and the binary silhouettes of the front and side views of the isolated object for one of the 10 grasping instances. Time flows from left to right, showing the progress as each individual article of clothing is removed from the pile and examined for classification.

Test Name	w_1	w_2	w_3	w_4
Area	1	0	0	0
Eccentricity	0	1	0	0
Binary Edges	0	0	1	0
Canny Edges	0	0	0	1
Combo A	1	1	0	0
Combo B	1	1	0	1
Combo C	0	1	0	1
Combo D	1	0	0	1

Table 3.3: The weights used for eight different tests conducted for the "lying approach".

3.5 Classification experiment in a hanging position

With six categories, five items per category, and 20 images per item, the dataset collected by the extraction / isolation procedure consists of 600 images. This dataset is labeled in a supervised learning manner so that the corresponding category of each image is known. Eight tests are used to compare the training and test images, see Table 3.3.

I conducted two experiments: leave-one-out classification and train-and-test classification. In leave-one-out classification, each of the 600 images is compared with the remaining 599 images in the dataset. If the nearest neighbor among these 599 is in the same category as the image, then the classification is considered a success, otherwise a failure. Results for all 100 images for each category are combined to yield a classification rate for that category, and the procedure is repeated for all eight tests in Table 3.3. The results, shown in Figure 3.21, reveal that most categories are either classified, on average, well (near or above 50%) or poorly (near or below 30%), with the best results achieved by the Combo A and Combo B tests.

In train-and-test classification, three articles of clothing from each category are selected for the training set and the remaining two articles are used for the test

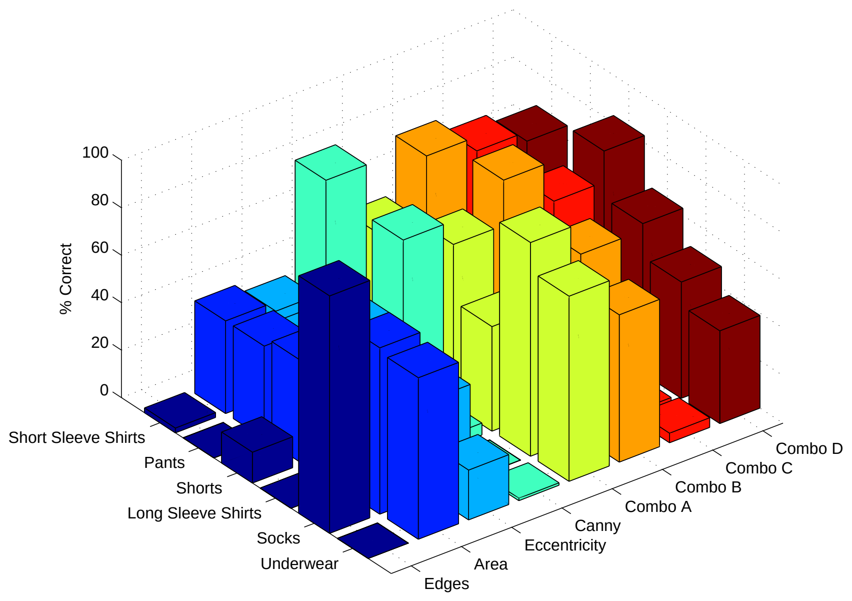


Figure 3.21: Leave-one-out classification results for all six categories using eight different tests.

set. Therefore, each image is compared with the 360 images in the training set, and the category of the nearest neighbor among these images is compared with the category of the image to determine success or failure. Results for all 40 test images for each category are combined to yield a classification rate for that category, and the procedure is repeated for all eight tests in Table 3.3. The results, shown in Figure 3.22, show results that are significantly worse than those of the leave-one-out experiment, due to the reduced amount of data used for training. We should emphasize that classifying an unknown article of clothing (hanging from a single, arbitrary grasp point) from a single image is extremely difficult even for a human viewer.

3.5.1 Interaction vs. Non-interaction

One of the goals of this work is to determine the usefulness of interactive perception in a clothing classification context. The process of interacting with each

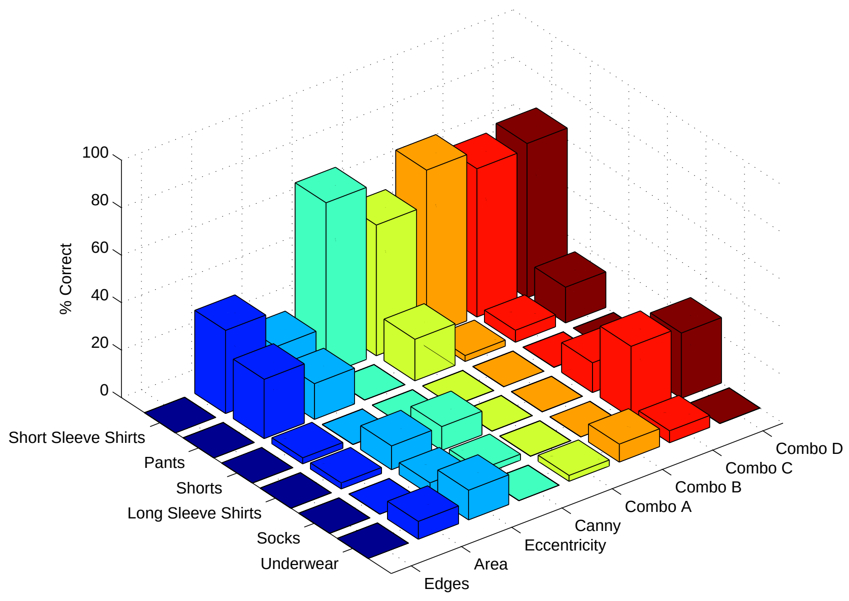


Figure 3.22: Train-and-test classification results for all six categories using eight different tests.

article of clothing provided the system with multiple views using various grasp locations, allowing the system to collect 20 total images of each object. Therefore, in the next experiment I compared features from all 20 images of each object with the remaining images in the dataset. The procedure is as follows:

1. The query article of clothing is compared with all articles in the dataset, and the category of the closest matching article is considered to be the category of the query article.

2. Two articles are compared by examining the 400 match scores between their pairs of images (20 images per article).

3. For each of the 20 images of the query article, the 1-NN among the other 20 images is found; these 20 match scores are then added to yield the distance between the two articles.

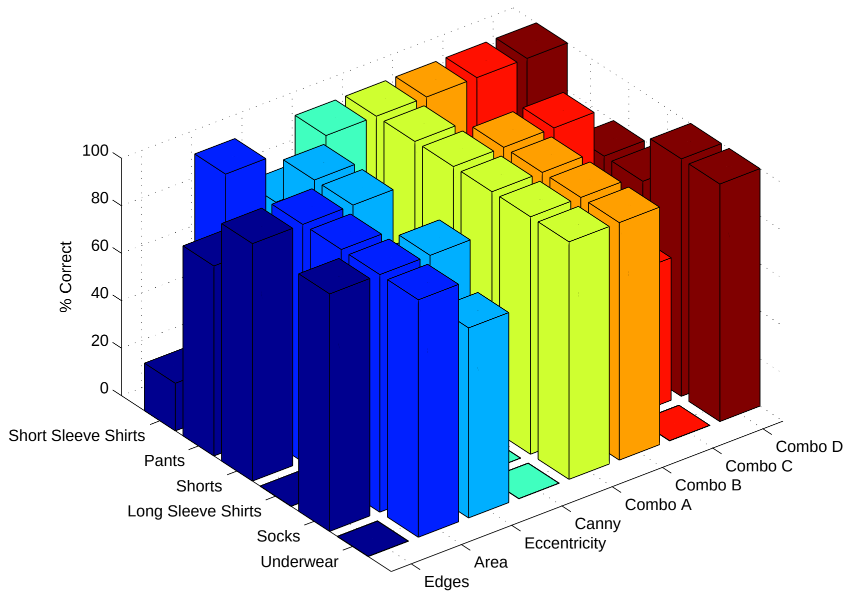
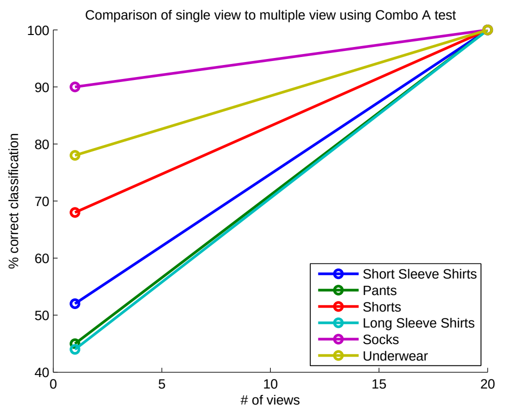


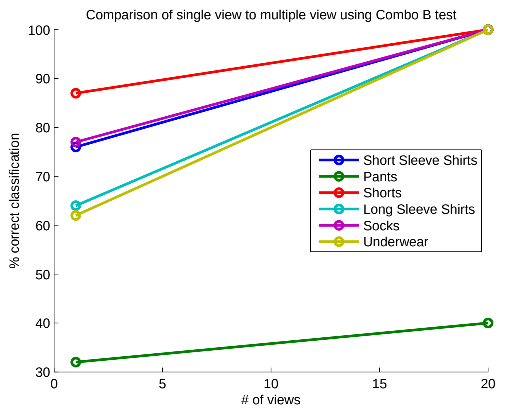
Figure 3.23: Classification results for all 30 objects using all 20 images for comparison.

This procedure corresponds to a manipulator picking up and dropping an object multiple times to get multiple views of it in order to better classify it. The results, shown in Figure 3.23, are indeed significantly higher than those obtained without interactive perception. In fact, Combo A achieved 100% classification on all categories using this method.

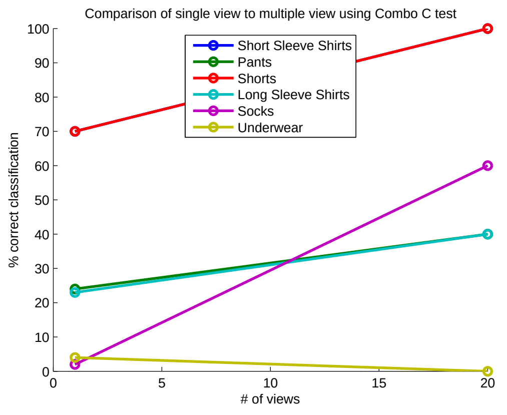
For comparison, the classification rates for all categories for the four different combination tests are shown in Figure 3.24, illustrating the difference between using a single view versus using all 20 views of an object. For Combo A, the average classification rate using a single image is 62.83%, while the average classification rate using all 20 images is 100%. These results show that, on average, classification rates using robot interaction are 59% higher than those that do not use interaction.



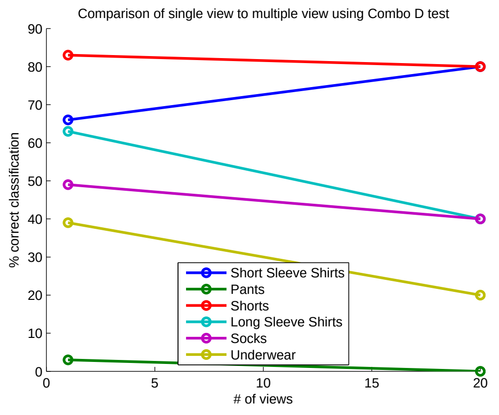
Combo A



Combo B



Combo C



Combo D

Figure 3.24: Classification results for all 30 objects using all 20 images for comparison using the combination tests.

3.6 Classification experiment in a lying position

The proposed **L-C-S-H** approach is applied to a laundry scenario to test its ability to perform practical classification. In the following experiments, the entire dataset consisted of 85 shirts, 30 cloths, 25 pants, 25 shorts, 10 dresses, 22 socks, and five jackets. The dataset is labeled using a supervised learning approach so that the corresponding category of each image is known. An approach is provided that illustrates the use of having mid-level characteristics (attributes) in grouping clothing categories that are difficult to classify. First, the baseline approach that uses SVMs, with a radial basis function kernel, is considered to directly classify each category without the use of mid-level characteristics. Then, the characteristics are introduced as an alternate mapping from low level features to high level categories.

3.6.1 Cross Validation for Clothing Classification

Each experiment is conducted using the 5-fold cross validation (CV) [20] to completely test each stage of the algorithm. 80% of each characteristic / category is used for training and 20% for testing, with each train and test set being mutually exclusive for each validation. To better describe the separation, 80% of the dataset is used that had a 1, or *TRUE*, for that characteristic / category and 80% of the dataset that had a 0, or *FALSE*, for that characteristic / category to make the training set, the rest of the dataset went to the test set. The dataset is organized into five equally-sized groups that share the same amount of characteristics and same amount of categories. The resulting values from each SVM ranges from $-1 \rightarrow +1$, with a deciding threshold of 0. For these experiments, five instances of each article are used to describe the feature vector. The approach with four, three, two, and one instance for each article is also tested. The results find that using five instances provides a higher overall true positive rate.

3.6.2 Baseline approach without mid-level layers

This experiment demonstrates the baseline approach using the low level features to classify each article of clothing correctly with support vector machines (SVM). Various aspects of the SVM approach are considered to better understand how the articles are categorized. This experiment uses local, global, and a combined set of low level features as input for each SVM that is used to classify for a particular article. Since there are seven categories to choose from, then seven SVMs are trained in order to provide a probability value for each input vector from the test set. For this type of multi-class classification, I chose the One-Versus-All (OVA) approach for each SVM, which uses a single SVM to compare each class with all other classes combined. Previous researchers [36, 65, 81, 21] suggest that for sparse data sets, the OVA approach provides better results over the Max-Wins-Voting (MWV) approach, which uses a single SVM for each pair of classes resulting in a voting scheme. Results are shown in Table 3.4 and 3.5.

3.6.3 Testing approach with characteristics (L-C-H)

For this experiment with **L-C-H**, 5-fold CV is used to test this stage of the algorithm, as in the previous experiments. The goal of this experiment is to determine the increase of overall performance with the addition of characteristics. Results are shown in Table 3.6 and 3.7. The training set still consists of the ground truth characteristics to compute the mean vectors for each category. The intermediary results of classifying the characteristics using local and global features range from 16% $\rightarrow$ 96%, with an average of 63%.

Local Features			
	Shirt	Dress	Socks
Shirt	**98.82**	1.18	0.00
Dress	100.00	**0.00**	0.00
Socks	0.00	13.64	**86.36**
Average TPR = **61.73%**			

Global Features			
	Shirt	Dress	Socks
Shirt	**57.65**	2.35	40.00
Dress	60.00	**0.00**	40.00
Socks	36.36	18.18	**45.45**
Average TPR = **34.37%**			

Local and Global Features			
	Shirt	Dress	Socks
Shirt	**40.00**	0.00	60.00
Dress	40.00	**0.00**	60.00
Socks	18.18	27.27	**54.55**
Average TPR = **31.52%**			

Table 3.4: Results of baseline system using SVM for three categories. TPR stands for True Positive Rate.

Local Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**98.82**	0.00	1.18	0.00	0.00	0.00	0.00
Cloth	63.33	**0.00**	20.00	10.00	0.00	6.67	0.00
Pants	100.00	0.00	**0.00**	0.00	0.00	0.00	0.00
Shorts	96.00	0.00	4.00	**0.00**	0.00	0.00	0.00
Dress	100.00	0.00	0.00	0.00	**0.00**	0.00	0.00
Socks	0.00	0.00	0.00	13.64	0.00	**86.36**	0.00
Jacket	100.00	0.00	0.00	0.00	0.00	0.00	**0.00**
Average TPR = **26.46%**							

Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**48.24**	0.00	1.18	10.59	0.00	40.00	0.00
Cloth	13.33	**50.00**	3.33	23.33	0.00	10.00	0.00
Pants	28.00	0.00	**0.00**	16.00	0.00	56.00	0.00
Shorts	44.00	0.00	4.00	**16.00**	0.00	36.00	0.00
Dress	40.00	0.00	0.00	20.00	**0.00**	40.00	0.00
Socks	18.18	9.09	0.00	27.27	0.00	**45.45**	0.00
Jacket	60.00	0.00	0.00	20.00	0.00	20.00	**0.00**
Average TPR = **22.81%**							

Local and Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**1.18**	0.00	4.71	34.12	0.00	60.00	0.00
Cloth	0.00	**0.00**	13.33	43.33	0.00	43.33	0.00
Pants	0.00	0.00	**8.00**	32.00	0.00	60.00	0.00
Shorts	0.00	0.00	8.00	**32.00**	0.00	60.00	0.00
Dress	0.00	0.00	0.00	40.00	**0.00**	60.00	0.00
Socks	0.00	0.00	4.55	45.45	0.00	**50.00**	0.00
Jacket	0.00	0.00	0.00	40.00	0.00	60.00	**0.00**
Average TPR = **13.03%**							

Table 3.5: Results of baseline system using SVM to classify objects within seven categories.

Local Features			
	Shirt	Dress	Socks
Shirt	**41.18**	58.82	0.00
Dress	40.00	**60.00**	0.00
Socks	0.00	90.91	**9.09**
Average TPR = **36.76%**			

Global Features			
	Shirt	Dress	Socks
Shirt	**15.29**	84.71	0.00
Dress	40.00	**60.00**	0.00
Socks	13.64	86.36	**0.00**
Average TPR = **25.10%**			

Local and Global Features			
	Shirt	Dress	Socks
Shirt	**21.18**	78.82	0.00
Dress	20.00	**80.00**	0.00
Socks	36.36	63.64	**0.00**
Average TPR = **33.73%**			

Table 3.6: Results of **L-C-H** proposed system for three categories.

Local Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**41.18**	0.00	0.00	0.00	58.82	0.00	0.00
Cloth	23.33	**0.00**	6.67	0.00	70.00	0.00	0.00
Pants	44.00	0.00	**0.00**	0.00	56.00	0.00	0.00
Shorts	36.00	0.00	0.00	**0.00**	64.00	0.00	0.00
Dress	40.00	0.00	0.00	0.00	**60.00**	0.00	0.00
Socks	0.00	0.00	54.55	13.64	31.82	**0.00**	0.00
Jacket	20.00	0.00	0.00	0.00	80.00	0.00	**0.00**
Average TPR = **14.45%**							

Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**14.12**	0.00	4.71	0.00	80.00	0.00	1.18
Cloth	16.67	**0.00**	6.67	13.33	63.33	0.00	0.00
Pants	20.00	0.00	**4.00**	0.00	76.00	0.00	0.00
Shorts	28.00	0.00	4.00	**4.00**	64.00	0.00	0.00
Dress	40.00	0.00	0.00	0.00	**60.00**	0.00	0.00
Socks	13.64	0.00	0.00	0.00	86.36	**0.00**	0.00
Jacket	20.00	0.00	0.00	0.00	80.00	0.00	**0.00**
Average TPR = **11.73%**							

Local and Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**21.18**	0.00	0.00	0.00	78.82	0.00	0.00
Cloth	10.00	**0.00**	0.00	0.00	90.00	0.00	0.00
Pants	24.00	0.00	**0.00**	0.00	76.00	0.00	0.00
Shorts	28.00	0.00	0.00	**0.00**	72.00	0.00	0.00
Dress	20.00	0.00	0.00	0.00	**80.00**	0.00	0.00
Socks	36.36	0.00	0.00	4.55	59.09	**0.00**	0.00
Jacket	20.00	0.00	0.00	0.00	80.00	0.00	**0.00**
Average TPR = **14.45%**							

Table 3.7: Results of **L-C-H** proposed system to classify clothing using seven categories.

3.6.4 L-C-H vs. L-C-S-H

For this experiment with **L-C-S-H**, 5-fold CV is used to test this stage of the algorithm, as in the previous experiments. The goal of this experiment is to determine the increase of overall performance with the addition of a selection mask between the characteristics and the high level categories. The training set still consists of the ground truth characteristics to compute the mean vectors for each category. Each category demonstrates an increase in percentage until a threshold is reached. Each threshold varies based on the low level features used and what category is being tested. At that time, the percentage then eventually decreases to 0% after reaching the threshold.

Figure 3.25 illustrates the resulting TPR (True Positive Rate) as the characteristic with the lowest percentage is removed at each iteration, using local and global features. Each category demonstrates an increase in percentage until a threshold is reached. This threshold is decided to coincide with the least amount of remaining characteristics that produce the highest TPR. At that time, the percentage then eventually decreases to 0% when the rest of the characteristics are zeroed out. So, zeroing out a subset of negatively important characteristics improve the overall TPR. Characteristic values for the mean category vector in the training set are the only values that are zeroed out. The training vectors go through this process to better describe how each category is characterized, while the testing vectors are not changed and compared to how close they are to each mean vector.

Local Features			
	Shirt	Dress	Socks
Shirt	**95.80**	4.20	0.00
Dress	13.32	**86.65**	0.02
Socks	0.60	11.67	**87.72**
Average TPR = **90.06%**			

Global Features			
	Shirt	Dress	Socks
Shirt	**90.41**	9.51	0.08
Dress	2.80	**96.95**	0.25
Socks	30.45	69.55	**0.00**
Average TPR = **62.45%**			

Local and Global Features			
	Shirt	Dress	Socks
Shirt	**58.64**	40.64	0.72
Dress	4.32	**95.67**	0.02
Socks	37.10	44.66	**18.24**
Average TPR = **57.52%**			

Table 3.8: Results of **L-C-S-H** proposed system for three categories.

Local Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**95.80**	0.00	0.00	0.00	4.20	0.00	0.00
Cloth	28.08	**0.00**	0.00	0.00	69.45	2.48	0.00
Pants	53.33	0.00	**0.00**	0.00	46.67	0.00	0.00
Shorts	47.53	0.00	0.00	**0.00**	52.47	0.00	0.00
Dress	13.32	0.00	0.00	0.00	**86.65**	0.02	0.00
Socks	0.60	0.01	0.00	0.00	11.67	**87.72**	0.00
Jacket	33.81	0.00	0.00	0.00	66.19	0.00	**0.00**
Average TPR = **38.60%**							

Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**90.41**	0.00	0.00	0.00	9.51	0.08	0.00
Cloth	23.04	**0.00**	0.00	0.00	76.90	0.06	0.00
Pants	37.04	0.00	**0.00**	0.00	62.94	0.02	0.00
Shorts	36.93	0.00	0.00	**0.00**	63.00	0.07	0.00
Dress	2.80	0.00	0.00	0.00	**96.95**	0.25	0.00
Socks	30.45	0.00	0.00	0.00	69.55	**0.00**	0.00
Jacket	23.36	0.00	0.00	0.00	76.64	0.00	**0.00**
Average TPR = **26.77%**							

Local and Global Features							
	Shirt	Cloth	Pants	Shorts	Dress	Socks	Jacket
Shirt	**65.79**	0.00	0.00	0.89	31.49	0.04	1.79
Cloth	18.67	**0.00**	3.72	4.43	67.42	0.01	5.75
Pants	36.00	0.00	**1.93**	0.52	45.53	0.00	16.02
Shorts	32.94	0.00	1.72	**2.24**	45.62	0.05	17.43
Dress	5.48	0.00	0.00	0.07	**93.80**	0.00	0.65
Socks	37.54	0.00	1.26	3.17	50.06	**0.00**	7.97
Jacket	17.39	0.00	0.00	0.03	50.37	0.02	**32.20**
Average TPR = **27.99%**							

Table 3.9: Results of **L-C-S-H** proposed system for seven categories.

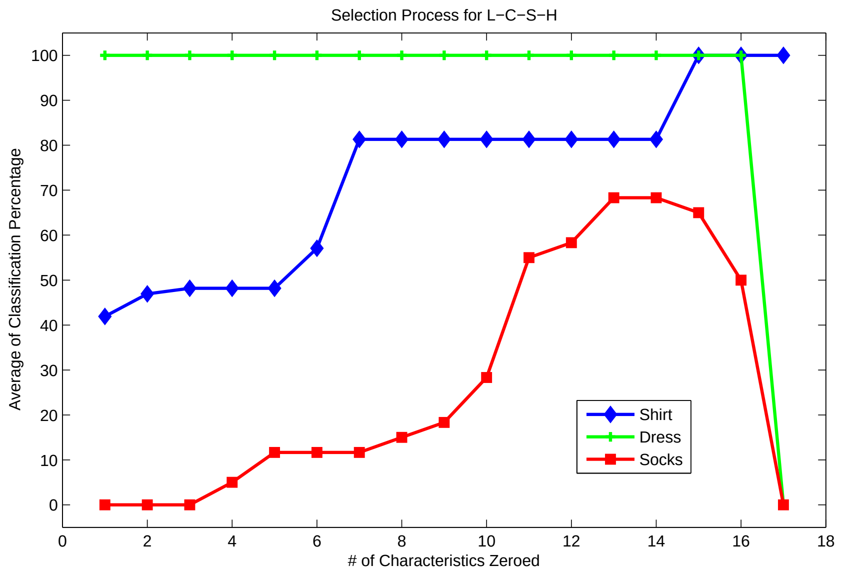


Figure 3.25: Characteristic Selection Process for **L-C-S-H** within three categories: Results of removing a single redundant characteristic at each iteration. Starting from removing no characteristics and finishing with removing all characteristics.

Chapter 4

Unfolding clothing

4.1 Unfolding overview

The overall goal of this chapter is to define a two-phase algorithm that describes how to unfold laundry into a flat canonical position. In the first phase, initial wrinkles and/or folds are removed without using any depth information (and hence can be accomplished with a single overhead camera). The second phase implements the proposed model, explained below, to remove more difficult folds using depth information. Each component of the model will be explained in further detail below. The corners of the article are used as initial locations for possible grasp points. Corners are used based on an assumption that if the corners are lying flat on the table in opposite orientation and/or are evenly spaced apart, then the article is closer to the canonical (desired) position. The idea behind obtaining a goal orientation/form is to remove all peaks (i.e. topologically high areas) and decrease the vertical size iteratively into a uniform layout. Several factors are considered to characterize a method in flattening an article of clothing (e.g. peak ridges, continuity of a surface, and corner locations).

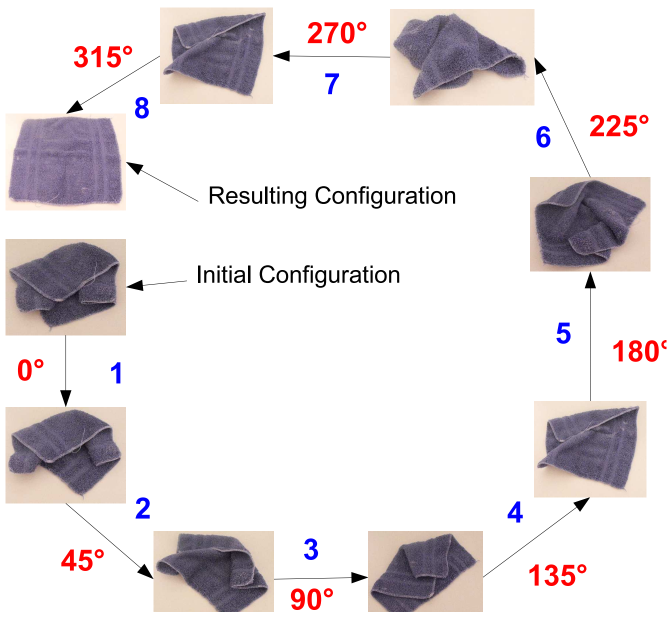


Figure 4.1: Process of the first phase to unfolding / flattening laundry. Each step of the process is numbered along with the orientation that is used to transform one configuration into another. In each step, the outer edge of the piece of clothing is grasped and pulled away from the center of the object. This is an illustrative example only.

4.2 First phase of the unfolding algorithm

The purpose of the first phase of flattening a piece of clothing is to remove any minor wrinkles and/or folds. This phase provides a better configuration for the second phase to start with instead of the initial configuration. In the first phase, the robot moves around the cloth counter-clockwise, pulling at individual corners every d degrees (d is set to be 45). The cloth is grasped at the edge of the clothing (determined by foreground / background segmentation) and pulled away from the centroid. Figure 4.1 illustrates the process of the first phase, which consists of the first eight steps of the algorithm.

4.3 Second phase of the unfolding algorithm

The second phase of flattening a piece of clothing uses depth information to locate possible folds in the cloth, and to find grasp points and directions to enable the folds to be removed. Each iteration of this phase involves six steps, which are described in the following subsections.

4.3.1 Step One: Peak ridge

The peak ridge is a binary map computed from the depth image as follows:

$$P_R(x, y) = \begin{cases} 1, \text{ if } E(x, y) \geq 0.9 \max_{x',y'} E(x', y') \\ 0, \text{ otherwise} \end{cases}, \tag{4.1}$$

where $E(x, y)$ is the value of the depth image for pixel (x, y), and $\max_{x',y'} E(x', y')$ is the maximum value in the depth image, which is called the peak. In the depth image, larger (brighter) points are farther from the table, so the peak is the highest point above the table. This equation locates the area(s) containing pixels whose depth is within 10% of the peak. The area that matches this criterion and also includes the maximum value is the peak ridge. The function computes an (x_C, y_C) point (centroid) and the orientation $\theta_{maj}/\theta_{min}$(major/minor vectors) of the peak ridge. Figure 4.2 illustrates the original depth image of the object and the binary mask of the peak ridge of the object.

4.3.2 Step Two: Corner locations

To detect corners along the edge of the cloth, the Harris corner detector [40] is applied to the binary image that results from thresholding the depth image so that points on the table are zero while points on the cloth are one. The corner locations

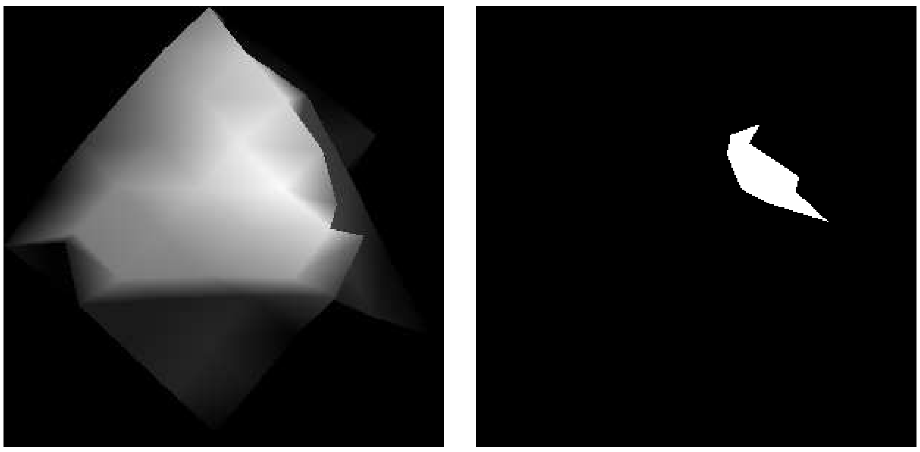


Figure 4.2: Depth image (left) and peak ridge (right) of an unfolded washcloth. In the depth image, brighter points are closer to the sensor (higher above the table). The peak ridge contains points within 10% of, and contiguous with, the peak.

are then the locations of these Harris corners, so that $C_L(x, y) = 1$ if a Harris corner is found at location (x, y), and 0 otherwise. The procedure finds locations of all detected corners and returns the locations in terms of position (x_N, y_N) and orientation θ_N of the corner. Some corners will be located on the peak ridge, while others will be located in the other non-peak regions, see Figure 4.3.

4.3.3 Step Three: Discontinuity function

Discontinuities in the depth image are stored in a binary array computed as follows:

$$D_C(x, y) = \begin{cases} 1, \text{ if } B_{3\times3}(x, y) \bigcap B_{5\times5} = 1 \\ 0, \text{ otherwise} \end{cases} , \tag{4.2}$$

where $B_{3\times3}(x, y)$ tests for sharp increases / decreases in values of the depth image, $B_{5\times5}(x, y)$ tests for sharp increases / decreases in the slope of the depth image, and $\bigcap$ is the logical AND operator. More specifically, $B_{3\times3}(x, y) = 1$ if $\max(\mid E(x+1, y) - E(x-1, y) \mid, \mid E(x, y+1) - E(x, y-1) \mid) > th$, where $th = 5$ is a threshold, and 0

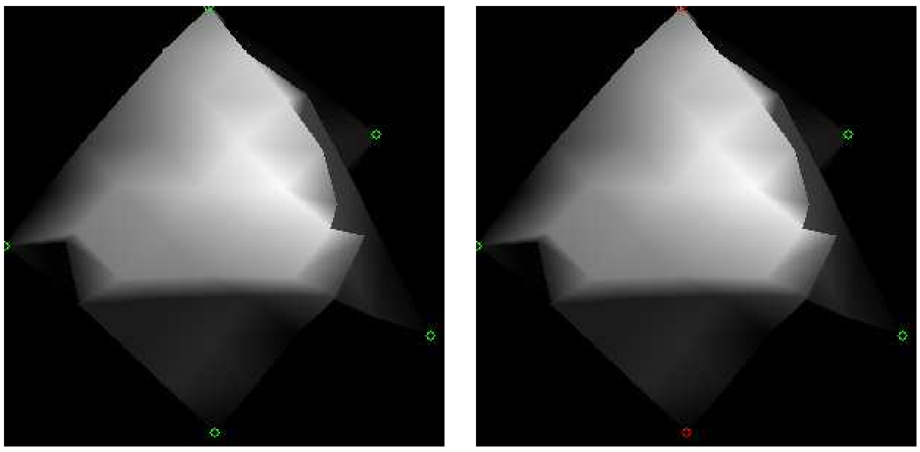


Figure 4.3: LEFT: Corner locations of the object, indicated by green circles, found by applying the Harris detector to the binary image that segments the cloth from the background table. In this case there are five corners. Note that, where the cloth is flat, the depth image blends into the background in the figure. RIGHT: The same corners distinguished by whether they are connected with the peak region (green), or outside the peak region (red). In this case three of the five corners are connected with the peak region.

otherwise. $B_{3\times3}(x, y)$ looks for large changes in the depth image using the 4-neighbors of the pixel in the surrounding 3×3 window. Similarly, $B_{5\times5}(x, y) = 1$ if, in either the up / down or left / right direction, the slopes of the depth image along successive columns / rows are either in different directions or vary by an amount more than th. Therefore, $B_{5\times5}(x, y)$ looks for large changes in the slope of the depth image using the 5×5 neighborhood of the pixel. Figure 4.4 illustrates the results of equation (4.2).

4.3.4 Step Four: Continuity check of the peak ridge

The continuity check combines the peak ridge with the discontinuity function. The value $C_P(x, y) = 1$ if (x, y) is contiguous with the peak ridge as determined by the discontinuity function, and 0 otherwise. To compute this function, a floodfill procedure is applied to the peak ridge image, successively incorporating adjacent

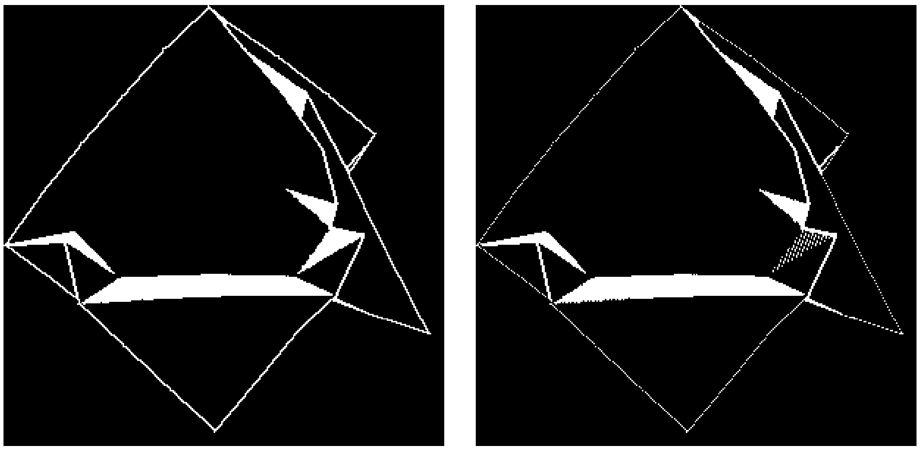


Figure 4.4: Locations of discontinuity (white points) on the object before (left) and after (right) removing steep slopes. The left image displays $B_{3\times3}$, while the right image displays $D_C(x, y)$.

pixels whose value in the discontinuity function is 0. When a discontinuity pixel is found whose value is 1, it is not included in the region. Figure 4.5 shows the result of this procedure.

4.3.5 Step Five: Continuity check for all peak corner locations

The continuity check on all peak corner locations determines which corners are connected to the peak region:

$$C_C(x, y) = C_L(x, y) \bigcap C_P(x, y). \tag{4.3}$$

This equation returns a subset of $C_L(x, y)$ that contains the locations of corners. The remaining subset, $F(x, y) = C_L(x, y) \bigcap \overline{C_C}(x, y)$, where the overline indicates binary complement, contains the locations of corners known to be located on a different region than the one containing the peak ridge. Such corners are likely on a different

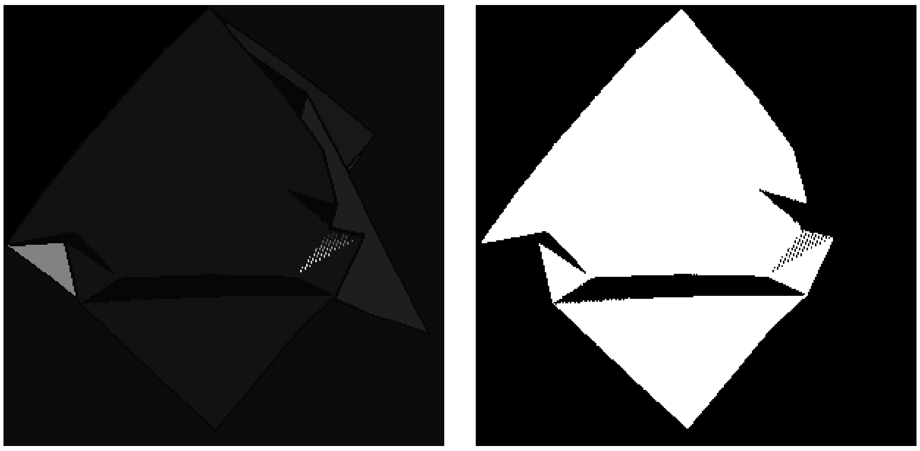


Figure 4.5: LEFT: Different regions of the object found by applying connected components after finding discontinuities. RIGHT: The region connected to the peak ridge.

fold of the cloth. Figure 4.3 shows the former corners (for which $C_C(x, y) = 1$) as red circles, while the latter corners (which are located on a different fold from the peak ridge) are shown as green circles.

4.3.6 Step Six: Cloth Model Grasp Point and Direction

The former computations are used to determine a grasp point for the cloth as follows. All of the points for which $C_C(x, y) = 1$ are candidate grasp points, and the final grasp point is selected arbitrarily from among these candidates. For the direction in which to pull the cloth, there are two possibilities. First, if $| \{(x, y) : C_C(x, y) = 1\} | = | \{(x, y) : C_L(x, y) = 1\} |$, that is, all of the detected corners are on the peak region surface, then the cloth is pulled away from the centroid of the cloth. Otherwise, the cloth is assumed to contain a fold, and therefore it is pulled toward the centroid in an attempt to flatten the fold. The entire six-step procedure is repeated until the cloth does not change shape.

4.4 Unfolding experiments on pieces of clothing

4.4.1 Differences Between Pulling in Different Directions

Figure 4.6 illustrates the initial cloth configuration, I_C, along with the eight different configurations that result from pulling the cloth from a single (x_G, y_G) coordinate in eight different orientations, θ_G. The eight different orientations proceed from 0 to 315 degrees in a counter-clockwise direction at 45-degrees intervals, i.e., the sequence of angles is $0, 45, 90, 135, 180, 225, 270, 315$, in degrees. The convention that 0 degrees points toward the bottom of the image is adopted.

Figure 4.7 displays the difference values from the initial configuration to the eight different configurations, in terms of pixel values. The lower the difference value, the more in common the two configurations share in terms of shape. As can be seen from Figure 4.7, the 4^{th} and 5^{th} configurations are significantly different from the initial configuration, in terms of shape. This plot illustrates how much the cloth configurations can change from pulling on a single point.

In Figure 4.7, the reason the low and high orientations are correlated to the initial configuration is because those orientations are pulling the point away from the centroid of the cloth. The middle orientations (i.e., 135 and 180) have a very different shape due to the fact that they are pulling the point over the centroid and therefore completely change the configuration / topology of the cloth.

4.4.2 Experimental Test for First Phase of Unfolding Algorithm

This experiment tested the first phase of the proposed algorithm and monitored the process from eight iterations of pulling the cloth from point (x_G, y_G) in

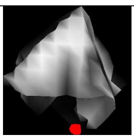
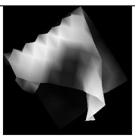
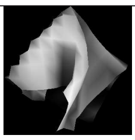
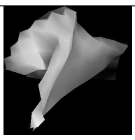
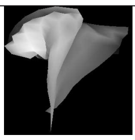
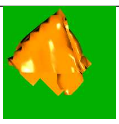
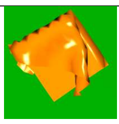
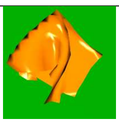
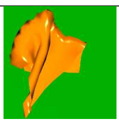
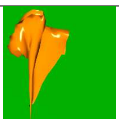
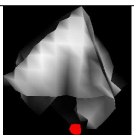
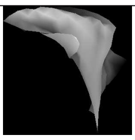
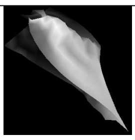
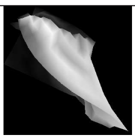
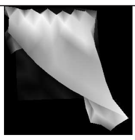
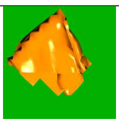
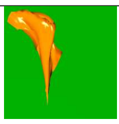
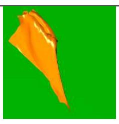
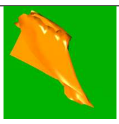
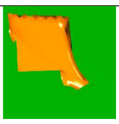
Depth Image					
Cloth Model					
Orientation θ_G	I_C	0	45	90	135
Depth Image					
Cloth Model					
Orientation θ_G	I_C	180	225	270	315

Figure 4.6: From Left to Right, Top to Bottom: The depth image of the initial configuration, I_C, with the grasp point (x_G, y_G) coordinate marked with a red dot; and the eight different configurations resulting from pulling the cloth from the initial configuration in eight different orientations, θ_G, using the same (x_G, y_G) coordinate. Pulling in the direction opposite the centroid of the object tends to improve flattening (e.g., $\theta_G = 0$).

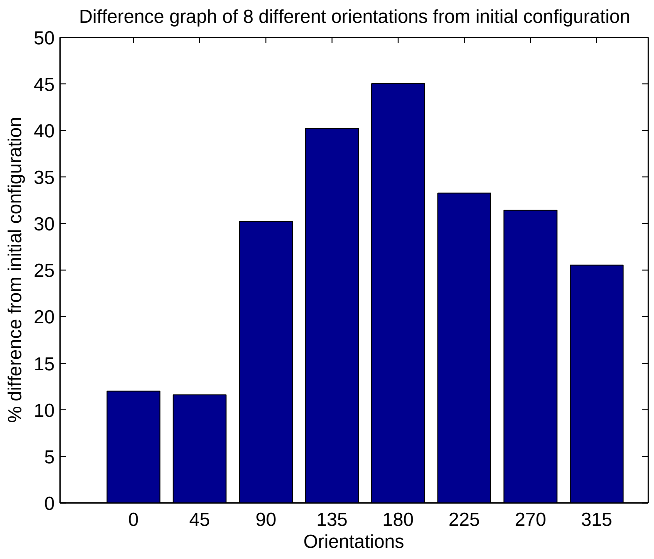


Figure 4.7: Difference graph of the eight different configurations of Figure 4.6 to the initial configuration in the same figure, measured as a difference between binary images (with 1 indicating foreground and 0 indicating background).

orientation θ_G starting with the same initial configuration, I_C, as in section 4.4.1. Figure 4.8 illustrates the configurations throughout the entire process. As can be seen from Figure 4.8, the models continually change the configuration in a manner that flattens and unfolds larger areas of the cloth as the iterations increase. Eventually, the cloth is mostly flattened into a more recognizable shape in the final iteration. The following equation describes how the percentage of flatness is calculated:

$$PC_{\text{Flat}} = \frac{\sum_{(x,y)} E(x,y) < \mu}{\sum_{(x,y)} E(x,y)}, \tag{4.4}$$

where $E(x,y)$ is the value of the depth image, and $\mu(=20)$ is a threshold indicating the maximum depth image value for which the cloth is considered to be lying flat on the table. The overall goal of the next step in the unfolding process is to increase the

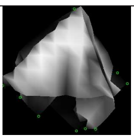
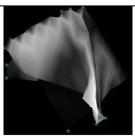
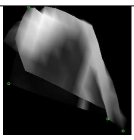
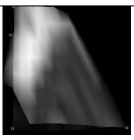
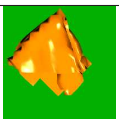
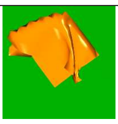
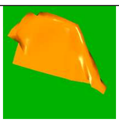
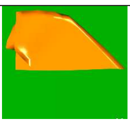
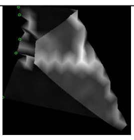
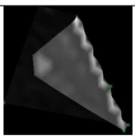
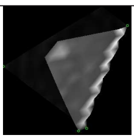
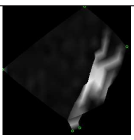
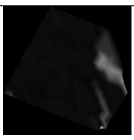
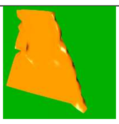
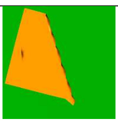
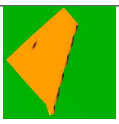
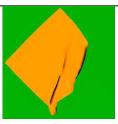
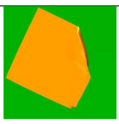
Depth Image					
Cloth Model					
$PC_{\text{Flat}}(\%)$	11.08	17.60	16.30	8.16	
Depth Image					
Cloth Model					
$PC_{\text{Flat}}(\%)$	33.78	44.12	41.47	68.31	83.51

Figure 4.8: Experimental test for first phase of unfolding algorithm for one initial configuration (folded cloth, see section 4.4.3). From Left to Right: Eight successive configurations resulting from pulling the initial configuration into eight successive orientations, θ_G, from eight different grasp points, (x_G, y_G), using the first phase of the proposed algorithm. Pulling in directions opposite the centroid causes the object to be nearly flattened.

flatness toward 100%.

4.4.3 Taxonomy of Possible Starting Configurations

Figure 4.9 displays the initial and final configurations of three different starting configurations after the eight steps of the first phase of the proposed algorithm. The dropped cloth is created by dropping the cloth onto the table from a predefined height, the folded cloth is created by sliding the article across the corner of the table and allowing it to fold on top of itself, and the placed cloth is slowly placed on the table from the same position as the dropped cloth. For the most part, all of the final configurations contain a large amount of the cloth to be unfolded and/or

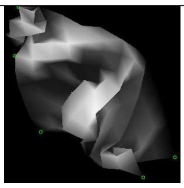
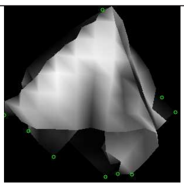
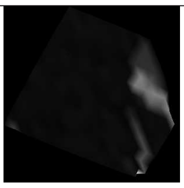
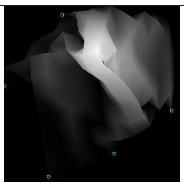
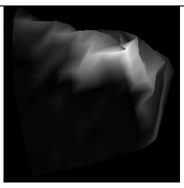
	Configuration		$PC_{\text{Flat}}(\%)$	
Cloth	Initial	Final	Initial	Final
Dropped			4.8	18.1
Folded			11.1	83.5
Placed			30.6	56.2

Figure 4.9: Various initial and final configurations along with the percentage of cloth that is unfolded / flattened using the first phase of the proposed algorithm. The initial configuration is obtained by dropping the cloth (top), folding the cloth across a table edge (middle), and placing the cloth by lowering it to the table (bottom). In all cases the algorithm increases the flattening percentage of the object. The final configurations are a result of the first phase only.

flattened. Figure 4.9 also displays the percentage of cloth that is unfolded/flattened in the initial and final configurations. As observed in Figure 4.9, the difference in percentage between the initial and final position is always higher.

4.4.4 Experiment using Both Phases of Unfolding Algorithm

This experiment tested the proposed algorithm in determining if this approach would completely flatten a piece of clothing. The test used the first and second phase of the algorithm to grasp the cloth at various locations, (x_G, y_G), and move the cloth at various orientations, θ_G, until the cloth obtained a flattened percentage greater than 95%. Figure 4.10 illustrates the configurations at selected iterations of the

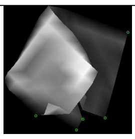
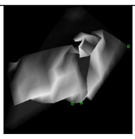
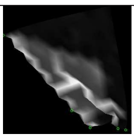
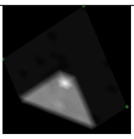
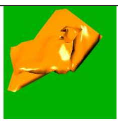
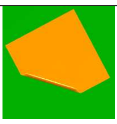
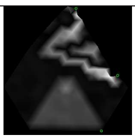
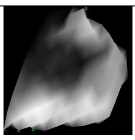
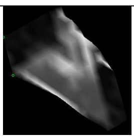
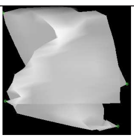
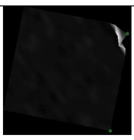
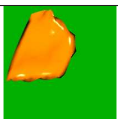
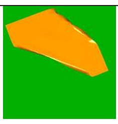
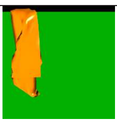
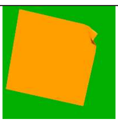
Iteration	1	7	13	19	
Depth Image					
Cloth Model					
$PC_{\text{Flat}}(\%)$	3.21	22.61	45.27	72.32	
Iteration	25	31	37	43	49
Depth Image					
Cloth Model					
$PC_{\text{Flat}}(\%)$	46.36	4.54	23.93	0.01	95.57

Figure 4.10: Flattened cloth test for one initial configuration (folded cloth, see section 4.4.3). From Left to Right, Top to Bottom: Nine different configurations resulting from pulling the initial cloth 49 times in successive orientations and grasp points, using the second phase of the proposed algorithm. The nine configurations shown are selected by hand to be representative of the 49. The final iteration resulted in flattening the cloth with a PC_{Flat} value over 95%.

entire algorithm. The percentages of flatness range from $0.01\% \rightarrow 95.6\%$. Figure 4.11 shows the percentage of flatness against all iterations of the algorithm.

4.4.5 Unfolding Experiment using PUMA 500

The goal of this experiment is to test the performance of my algorithm in a real world environment using a PUMA 500 manipulator. Figure 4.12 displays the results of using the peak region on an actual cloth to determine which corner position (x_G, y_G) to select and in which orientation θ_G to pull the object. I used a Logitech QuickCam 4000 for an overhead view to capture the configuration of the cloth. After running

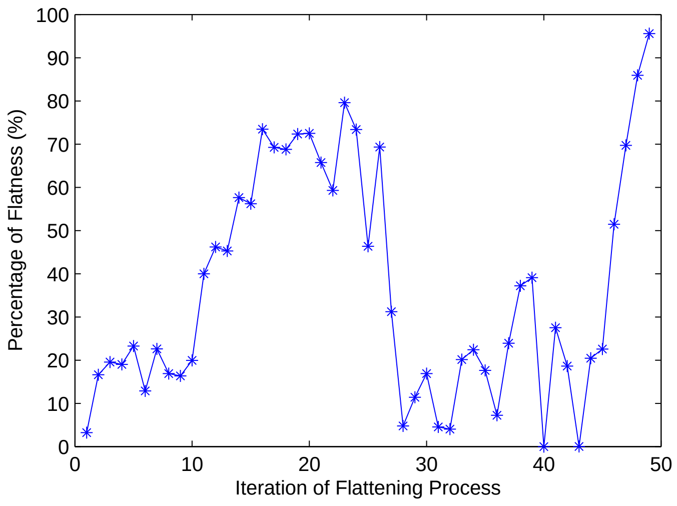


Figure 4.11: Plot of the percentage of flatness against all iterations of the algorithm.

the image through my approach, the system calculated an output that is transmitted to the robot for extraction. Figure 4.13 shows an example of the movements of the robot after the location and orientation are found.

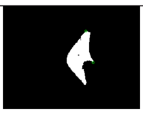
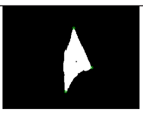
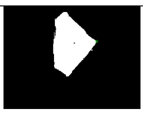
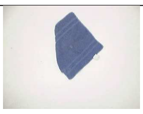
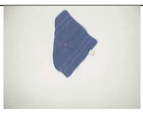
Peak Region					
Actual Cloth					
Orientation θ_G	I_C	0	45	90	135
Peak Region					
Actual Cloth					
Orientation θ_G	I_C	180	225	270	315

Figure 4.12: Actual cloth test for one initial configuration. From Left to Right, Top to Bottom: The initial configuration and eight different configurations resulting from pulling the initial cloth into 8 different orientations, θ_G, using 8 different grasp points, (x_G, y_G) successively, using the first phase of the proposed algorithm (since depth information is not available).

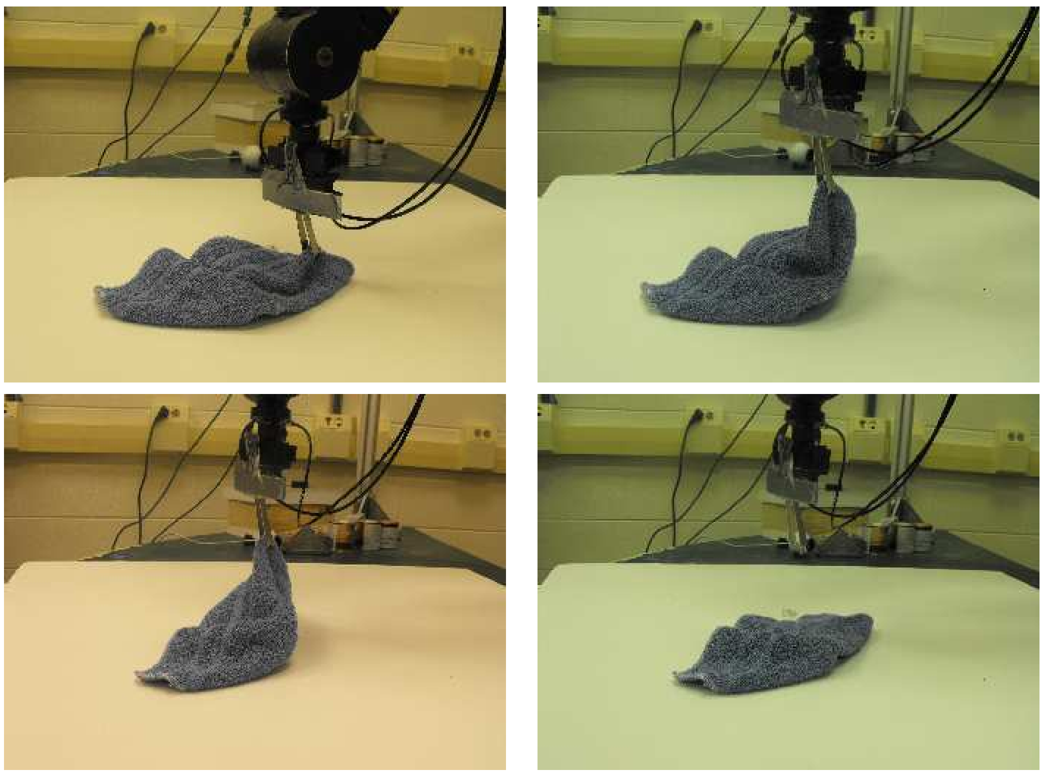


Figure 4.13: The various steps of the PUMA manipulator pulling the cloth in a specified orientation, θ_G. From Left to Right, Top to Bottom: the manipulator grasps the cloth on the table, picks the cloth up to a predefined height, pulls the cloth in a precalculated orientation, and drops the cloth back onto the table.

Chapter 5

Pose estimation

5.1 Pose estimation overview

For this chapter, an article of clothing is modeled as a one-sided 3D triangulated mesh. Let $V = (v_1, \ldots, v_n)$ be the n vertices of a 3D triangulated mesh, where $v_i = (x_i, y_i, z_i)$ contains the 3D coordinates of the ith vertex. The state vector V captures the shape of the mesh at any given time, where we have omitted the time index to simplify the notation.

The goal of this research of this thesis is to find the most likely mesh V^* in each frame, by finding the shape that minimizes the energy of the system:

$$V^* = \arg\min \Psi(V). \tag{5.1}$$

5.2 Energy function

The energy functional is defined as the sum of four terms:

$$\Psi(V) = \Psi_S(V) + \lambda_C\, \Psi_C(V) + \lambda_D\, \Psi_D(V) + \lambda_B\, \Psi_B(V), \tag{5.2}$$

where $\Psi_S(V)$ is a smoothness term that captures the internal energy of the mesh, $\Psi_C(V)$ is a data term that seeks to ensure that corresponding points are located near each other, $\Psi_D(V)$ measures the difference in depth between the mesh vertices and the input, and $\Psi_B(V)$ is the boundary term that regulates the mesh vertices located on the boundary of the object. The weighting parameters λ_C, λ_D, and λ_B govern the relative importance of the terms.

Each of the energy terms is useful in different scenarios. Figure 5.1 illustrates the contribution of each energy term for a toy example of a horizontal slice of vertices from the mesh.

5.2.1 Energy term - Smoothness

Let $\hat{V} = (\hat{v}_1, \ldots, \hat{v}_n)$, where $\hat{v}_i = (\hat{x}_i, \hat{y}_i, \hat{z}_i)$, be a hexagonal grid of equilateral triangles that is created from the first image of the video sequence. We will refer to $\hat{V}$ as the *canonical* mesh. Except for the boundaries, each vertex in the canonical mesh has, when projected orthogonally onto the $z = 0$ plane, three pairs of collinear adjacent points passing through it, similar to [78]. Let E be the set of all vertex index triplets such that $(i, j, k) \in E$ means that $\hat{v}_i$, $\hat{v}_j$, and $\hat{v}_k$ form two connected, equidistant, and collinear edges in the projected canonical mesh, see Figure 5.2.

Since the projected canonical mesh is formed from equilateral triangles (i.e.

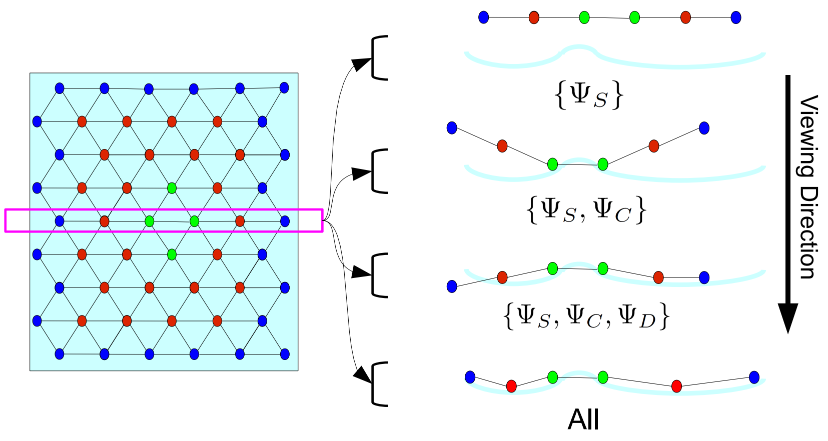


Figure 5.1: Each of the energy terms contributes to improving the fit between the mesh and the object. Left: Front view of an example mesh, containing regular vertices (red dots), boundary vertices (blue dots), and vertices near image texture (green dots). Right: Top view of mesh. From top to bottom, the smoothness term ensures low derivatives in the mesh, the data term pulls the mesh toward the correspondences (green dots), the depth term moves vertices along the sensor viewing direction, and the boundary term introduces lateral forces to increase the fit at the boundaries (blue dots).

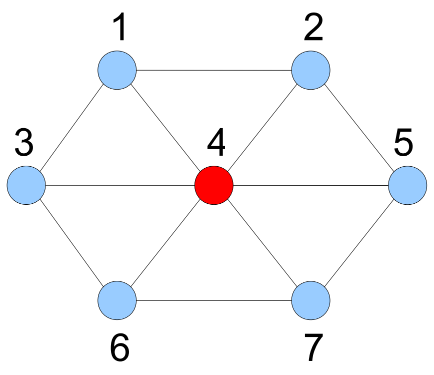


Figure 5.2: This figure illustrates an example of a hexagonally connected vertex, where vertices $(1, 2, 3, 4, 5, 6, 7) \in V$. Vertex 4 has three pairs of collinear edges: $\{(1, 4, 7), (2, 4, 6), (3, 4, 5)\} \in E$.

equidistant edges), the following is derived:

$$\hat{v}_i - \hat{v}_j = \hat{v}_j - \hat{v}_k \qquad \forall (i,j,k) \in E, \tag{5.3}$$

assuming that the initial configuration is fronto parallel. Therefore, in the deformed mesh V, this will approximately hold:

$$v_i - 2v_j + v_k \approx 0, \qquad \forall (i,j,k) \in E \tag{5.4}$$

which leads to the following smoothness term:

$$\begin{aligned}\Psi_S(V) = \frac{1}{2} \sum_{(i,j,k)\in E} & (x_i - 2x_j + x_k)^2 \\ & +(y_i - 2y_j + y_k)^2 \\ & +(z_i - 2z_j + z_k)^2. \end{aligned} \tag{5.5}$$

Let

$$X = \begin{bmatrix} x_1 & x_2 & \cdots & x_n \end{bmatrix}^T \tag{5.6}$$

$$Y = \begin{bmatrix} y_1 & y_2 & \cdots & y_n \end{bmatrix}^T \tag{5.7}$$

$$Z = \begin{bmatrix} z_1 & z_2 & \cdots & z_n \end{bmatrix}^T \tag{5.8}$$

be vectors containing the x, y, and z coordinates, respectively, of the deformed mesh. Extending the work in [78] to 3D, the above equation can be rewritten in matrix form as

$$\Psi_S(V) = \frac{1}{2}(K_{col}X)^T K_{col}X$$

$$+\frac{1}{2}(K_{col}Y)^T K_{col} Y$$
$$+\frac{1}{2}(K_{col}Z)^T K_{col} Z, \tag{5.9}$$

where K_{col} is an $n_{col} \times n$ matrix, where n_{col} is the number of collinear triplets, and $n_{col} \approx 3n$, where the boundaries cause the approximation. The elements of the matrix K_{col} contain the value 0, 1, or -2 depending upon the relationships within each triplet $(i, j, k) \in E$. Each row of K_{col} corresponds to a triplet within the triangle mesh, and each element in the row has a value of zero except for the three locations of i, j, k which contain 1, -2, and 1, respectively. Each column of K_{col} corresponds to a vertex within the mesh and has nonzero elements for the triplets containing the vertex.

The above expression can be simplified as follows:

$$\Psi_S(V) = \frac{1}{2}(X^T KX + Y^T KY + Z^T KZ), \tag{5.10}$$

where $K = K_{col}^T K_{col}$ is an $n \times n$ constant matrix capturing the collinear and adjacent vertices in the canonical mesh using E.

5.2.2 Energy term - Correspondence

The correspondence term uses the Speeded Up Robust Feature (SURF) descriptor, discussed in section 5.2.2.2, from the input data of the current RGBD image to compare a possibly deformed mesh V with the canonical mesh $\hat{V}$ using barycentric coordinates.

5.2.2.1 Barycentric coordinates

Suppose a point $p = (x, y, z)$ in the canonical RGBD image that happens to lie within the triangle defined by vertices $\hat{v}_i$, $\hat{v}_j$, and $\hat{v}_k$, as illustrated in Figure 5.3.

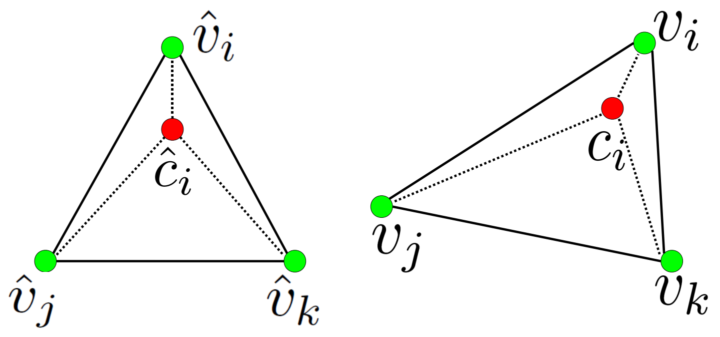


Figure 5.3: Example of barycentric coordinates. The point $\hat{c}_i$ is in one of the equilateral triangles in the canonical mesh (left) that retains its relative position within the triangle as the triangle is deformed (right).

The barycentric coordinates of p are defined as the triplet $(\beta_i, \beta_j, \beta_k)$ such that $p = \beta_i \hat{v}_i + \beta_j \hat{v}_j + \beta_k \hat{v}_k$ and $\beta_i + \beta_j + \beta_k = 1$. Given p, the barycentric coordinates are computed by solving the system of equations

$$\begin{bmatrix} \hat{x}_i - \hat{x}_k & \hat{x}_j - \hat{x}_k \\ \hat{y}_i - \hat{y}_k & \hat{y}_j - \hat{y}_k \end{bmatrix} \begin{bmatrix} \beta_i \\ \beta_j \end{bmatrix} = \begin{bmatrix} x - \hat{x}_k \\ y - \hat{x}_k \end{bmatrix} \tag{5.11}$$

for β_i and β_j, then setting $\beta_k = 1 - \beta_i - \beta_j$. If p lies within the triangle, then the barycentric coordinates will lie within 0 and 1, inclusive, $0 \leq \beta_i, \beta_j, \beta_k \leq 1$; and the z equation will also be satisfied: $(\hat{z}_i - \hat{z}_k)\beta_i + (\hat{z}_j - \hat{z}_k)\beta_j = z - \hat{z}_k$.

Now suppose that the mesh has deformed from $\hat{V}$ to V. If the relative position of p in the triangle remains fixed, then the transformation can be defined as:

$$T_V(p) = \beta_i v_i + \beta_j v_j + \beta_k v_k, \tag{5.12}$$

where v_i, v_j, and v_k are the 3D coordinates of the deformed mesh vertices that make up the triangle in which p lies. $T_V(p)$ yields the 3D coordinates of p when the mesh is described by V.

5.2.2.2 SURF descriptors and putative matching

In order to capture correspondences between consecutive frames, the Speeded Up Robust Feature (SURF) detector and descriptor [3] is used. SURF is a fast and robust feature selector that is invariant to scale and rotation, providing interest locations throughout the image. SURF detects features throughout the image, but a simple foreground / background segmentation procedure using depth values, see section 5.4.1, is used to remove SURF features on the background, leaving only features on the triangular mesh.

Putative matching of the SURF descriptors is used to establish sparse correspondence between successive images in the video. For each pair of descriptors, the Euclidean distance is computed, and the minimum is selected as the proper feature match. Only the top fifty percent are chosen based on the minimum Euclidean distances of the feature descriptors. To improve robustness, feature matching is constrained to lie within a given threshold of a location of the feature in the previous frame of the sequence, assuming that the motion is small in consecutive frames.

5.2.2.3 Correspondences

Let $\mathcal{C} = \{(\hat{c}_i, c_i)\}_{i=1}^{m}$ be the set of m correspondences between the canonical image and the input image. A specific correspondence $(\hat{c}_i, c_i)$ indicates that the 3D point $\hat{c}_i$ in the canonical image matches the 3D point c_i in the input image, where $\hat{c}_i = (\hat{x}_{ci}, \hat{y}_{ci}, \hat{z}_{ci})$ and $c_i = (x_{ci}, y_{ci}, z_{ci})$ for $i = 1, \ldots, m$. The data term is the sum of the squared Euclidean distances between the input point coordinates (SURF features) and their corresponding canonical coordinates (barycentric coordinates):

$$\Psi_C(V) = \frac{1}{2} \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \| \ c_i - T_V(\hat{c}_i) \ \|^2 \ . \tag{5.13}$$

5.2.3 Energy term - Depth

The depth term measures the difference between the z coordinate of each 3D vertex in the current mesh and the measured depth value. That is, it measures the distance along the ray passing through the 3D vertex to the current depth image obtained by the RGBD sensor:

$$\Psi_D(V) = \frac{1}{2} \sum_{i=1}^{n} |d(x_i, y_i) - z_i|^2 \, , \tag{5.14}$$

where $d(x_i, y_i)$ is the value of the depth image evaluated at the position (x_i, y_i), while z_i is the depth of the vertex at that same position. This term is designed to ensure that the mesh fits the data along with z axis even in textureless areas where no correspondences can be found.

5.2.4 Energy term - Boundary

Every boundary vertex is expected to remain near the boundary of the object even as it undergoes non-rigid deformations. Figure 5.4 illustrates the need for the boundary term in textureless areas. To deal with this situation, the boundary term minimizes the distances between boundary vertices and the 3D boundary points within the image, as determined by the foreground / background segmentation procedure (described in section 5.4.1):

$$\Psi_B(V) = \frac{1}{2}\sum_{v_i \in \mathcal{B}} (g_d(v_i) - v_i)^2 + \frac{1}{2}\sum_{g_i \in \mathcal{G}} (g_i - v_d(g_i))^2, \tag{5.15}$$

where v_i is the boundary vertex, $g_d(v_i)$ is the nearest boundary point to the vertex v_i in the current image d, g_i is the boundary point, $v_d(g_i)$ is the nearest vertex point to the boundary g_i in the current image d, $\mathcal{B}$ is the set of boundary vertices, and $\mathcal{G}$ is the set of boundary points. This term introduces two lateral forces on the vertices to ensure the mesh fits the data when the object's motion has components parallel to the image plane, as illustrated in Figure 5.5. Figure 5.6 displays an example of a situation when the extended boundary term provides a better mesh representation of the non-rigid object.

5.3 Energy minimization process

The goal is to locate the mesh that best explains the data while adhering to the smoothness constraint, described in section 5.2.1. To find the configuration of minimum energy, the partial derivative of the energy function is computed with respect to the vectors X, Y, and Z, and set the result to zero:

$$\begin{aligned}
\frac{\partial \Psi(V)}{\partial X} &= \frac{\partial \Psi_S(V)}{\partial X} + \lambda_C \frac{\partial \Psi_C(V)}{\partial X} \\
&\quad + \lambda_D \frac{\partial \Psi_D(V)}{\partial X} + \lambda_B \frac{\partial \Psi_B(V)}{\partial X} && (5.16) \\
\frac{\partial \Psi(V)}{\partial Y} &= \frac{\partial \Psi_S(V)}{\partial Y} + \lambda_C \frac{\partial \Psi_C(V)}{\partial Y} \\
&\quad + \lambda_D \frac{\partial \Psi_D(V)}{\partial Y} + \lambda_B \frac{\partial \Psi_B(V)}{\partial Y} && (5.17) \\
\frac{\partial \Psi(V)}{\partial Z} &= \frac{\partial \Psi_S(V)}{\partial Z} + \lambda_C \frac{\partial \Psi_C(V)}{\partial Z}
\end{aligned}$$

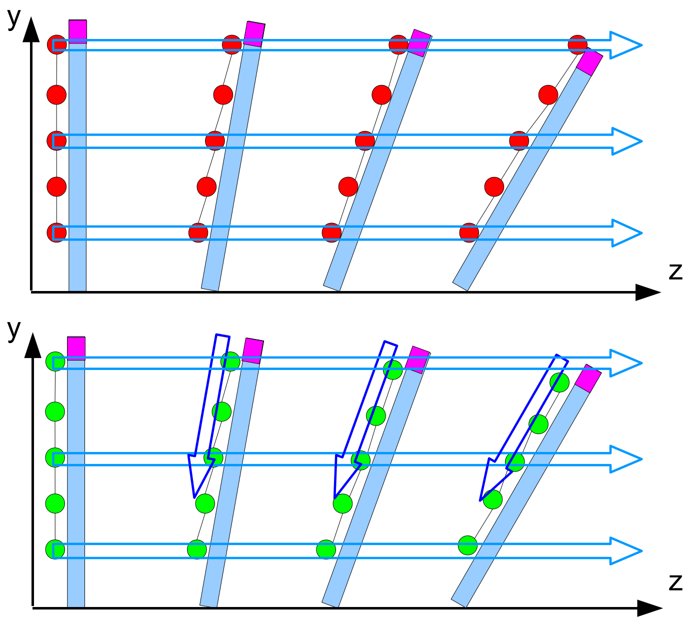


Figure 5.4: An example illustrating the need for the boundary term. Top: As a vertical object is increasingly slanted over time (from left to right), the vertices deviate from their true location, due to the limitation of the depth term to induce only forces parallel to the viewing direction (indicated by horizontal light blue arrows). Bottom: With the boundary term imposing lateral forces, the mesh vertices remain in their proper locations.

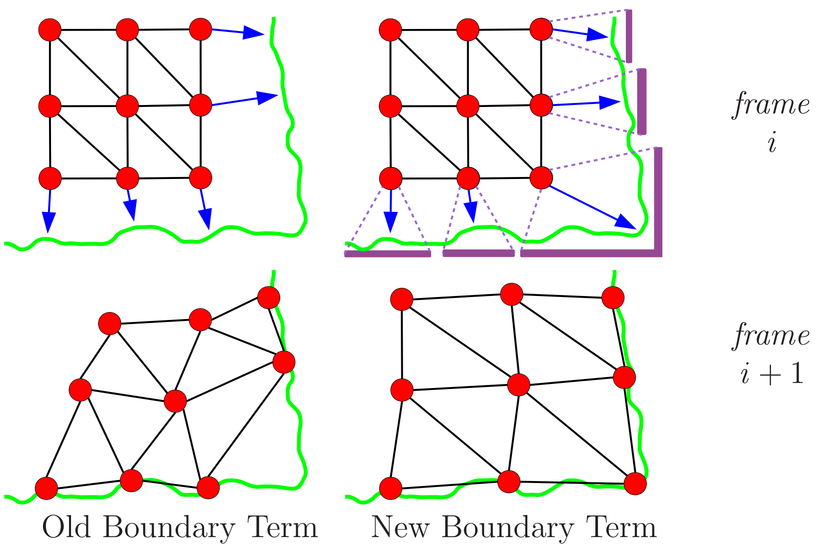


Figure 5.5: Left: Original implementation of boundary term from [107]. Red dots represent mesh vertices, green curve represents contour of object, and blue arrows represent the closest contour pixel for each mesh vertex. Right: Extended implementation of boundary term. Blue arrows represent closest contour centroid to each mesh vertex. The centroid of contour pixels is calculated from each purple group.

$$+\lambda_D \frac{\partial \Psi_D(V)}{\partial Z} + \lambda_B \frac{\partial \Psi_B(V)}{\partial Z} \tag{5.18}$$

The partial derivatives for smoothness are straightforward:

$$\frac{\partial \Psi_S(V)}{\partial X} = KX \tag{5.19}$$

$$\frac{\partial \Psi_S(V)}{\partial Y} = KY \tag{5.20}$$

$$\frac{\partial \Psi_S(V)}{\partial Z} = KZ. \tag{5.21}$$

Rewriting the barycentric transformation using an $n \times 1$ vector B whose elements are β_i, β_j, and β_k in the appropriate slots but zeros everywhere else:

$$T_V(p) = VB, \tag{5.22}$$

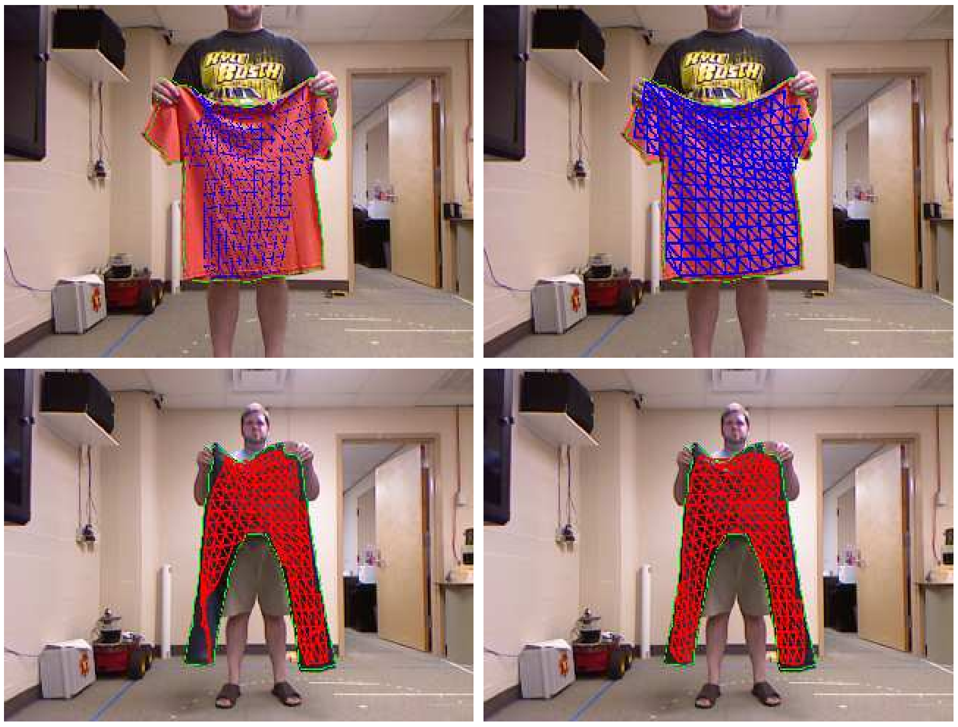


Figure 5.6: Left: Resulting mesh estimation with boundary term in [107]. Right: Resulting mesh estimation with extended boundary term.

the partial derivatives for the correspondence term are as follows:

$$\frac{\partial \Psi_C(V)}{\partial X} = -\sum_{(\hat{c}_i, c_i)\in\mathcal{C}} \left(x_{ci} - X^T B\right) B \tag{5.23}$$

$$\frac{\partial \Psi_C(V)}{\partial Y} = -\sum_{(\hat{c}_i, c_i)\in\mathcal{C}} \left(y_{ci} - Y^T B\right) B \tag{5.24}$$

$$\frac{\partial \Psi_C(V)}{\partial Z} = -\sum_{(\hat{c}_i, c_i)\in\mathcal{C}} \left(z_{ci} - Z^T B\right) B \tag{5.25}$$

The partial derivative of the depth term is similar to that of the correspondence term. Rewriting z_i using an $n \times 1$ vector F_i containing a one in the ith slot but zeros everywhere else:

$$z_i = Z^T F_i, \tag{5.26}$$

yields

$$\frac{\partial \Psi_D(V)}{\partial Z} = \sum_{i=1}^{n} (d(x_i, y_i) - Z^T F_i) F_i, \tag{5.27}$$

Similarly, the partial derivatives of the boundary term require rewriting v_i using the vector F_i:

$$v_i = V^T F_i, \tag{5.28}$$

leading to

$$\begin{aligned}\frac{\partial \Psi_B(V)}{\partial X} &= -\sum_{v_i\in\mathcal{B}} (g_d(v_i) - X^T F_i) F_i \\ &\quad -\sum_{g_i\in\mathcal{G}} (g_i - X^T F_i) F_i \end{aligned} \tag{5.29}$$

$$\begin{aligned}\frac{\partial \Psi_B(V)}{\partial Y} &= -\sum_{v_i\in\mathcal{B}} (g_d(v_i) - Y^T F_i) F_i \\ &\quad -\sum_{g_i\in\mathcal{G}} (g_i - Y^T F_i) F_i \end{aligned} \tag{5.30}$$

$$\begin{aligned}\frac{\partial \Psi_B(V)}{\partial Z} &= -\sum_{v_i \in \mathcal{B}} (g_d(v_i) - Z^T F_i) F_i \\ &\quad -\sum_{g_i \in \mathcal{G}} (g_i - Z^T F_i) F_i,\end{aligned} \tag{5.31}$$

In order to calculate the minimal solution, the semi-implicit scheme used by Kass et al. [49] is employed. The smoothness term is treated implicitly, while the correspondence, depth, and boundary terms are treated explicitly. Letting V_t represent the mesh at iteration t and V_{t-1} represent the mesh at the previous iteration $t-1$, the resulting equations are:

$$\begin{aligned}\alpha(X_t - X_{t-1}) &+ KX_t \\ &- \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(x_{ci} - X_{t-1}^T B\right) B \\ &- \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - X^T F_i) F_i \\ &- \lambda_B \sum_{g_i \in \mathcal{G}} (g_i - X^T F_i) F_i = 0\end{aligned} \tag{5.32}$$

$$\begin{aligned}\alpha(Y_t - Y_{t-1}) &+ KY_t \\ &- \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(y_{ci} - Y_{t-1}^T B\right) B \\ &- \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - Y^T F_i) F_i \\ &- \lambda_B \sum_{g_i \in \mathcal{G}} (g_i - Y^T F_i) F_i = 0\end{aligned} \tag{5.33}$$

$$\begin{aligned}\alpha(Z_t - Z_{t-1}) &+ KZ_t \\ &- \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(z_{ci} - Z_{t-1}^T B\right) B \\ &- \lambda_D \sum_{i=1}^{n} (d(x_i, y_i) - Z^T F_i) F_i \\ &- \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - Z^T F_i) F_i\end{aligned} \tag{5.34}$$

$$- \quad \lambda_B \sum_{g_i \in \mathcal{G}} (g_i - Z^T F_i) F_i = 0,$$

where α (> 0) is the adaptation rate that will move the triangle mesh with every iteration of the energy minimization process. This will allow the mesh to slowly position itself over the object until completion. The amount chosen for the movement rate depends on how fast the mesh model converges and how accurate the model is to the tracked locations before a minimum is found.

Rearranging terms leads to

$$\begin{aligned}
(K + \alpha I) X_t &= \alpha X_{t-1} \\
&+ \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(x_{ci} - X_{t-1}^T B \right) B \\
&+ \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - X_{t-1}^T F_i) F_i \\
&+ \lambda_B \sum_{g_i \in \mathcal{G}} (g_i - X_{t-1}^T F_i) F_i
\end{aligned} \tag{5.35}$$

$$\begin{aligned}
(K + \alpha I) Y_t &= \alpha Y_{t-1} \\
&+ \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(y_{ci} - Y_{t-1}^T B \right) B \\
&+ \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - Y_{t-1}^T F_i) F_i \\
&+ \lambda_B \sum_{g_i \in \mathcal{G}} (g_i - Y_{t-1}^T F_i) F_i
\end{aligned} \tag{5.36}$$

$$\begin{aligned}
(K + \alpha I) Z_t &= \alpha Z_{t-1} \\
&+ \lambda_C \sum_{(\hat{c}_i, c_i) \in \mathcal{C}} \left(z_{ci} - Z_{t-1}^T B \right) B \\
&+ \lambda_D \sum_{i=1}^{n} (d(x_i, y_i) - Z_{t-1}^T F_i) F_i \\
&+ \lambda_B \sum_{v_i \in \mathcal{B}} (g_d(v_i) - Z_{t-1}^T F_i) F_i
\end{aligned} \tag{5.37}$$

$$+\lambda_B \sum_{g_i \in \mathcal{G}} (g_i - Z_{t-1}^T F_i) F_i$$

Solving this equation for the unknowns X_t, Y_t, and Z_t yields the desired result as an iterative sparse linear system.

5.4 Segmentation and initialization

When creating an initial mesh, the segmented object from the first input image is used. If data is lost during the minimization process and the configuration of the object cannot be calculated, then the 3D mesh is reinitialized and continue the energy minimization. The segmentation, initialization, and reinitialization is now described in detail.

5.4.1 Segmentation and contour extraction using graph cuts

The approach to contour extraction is formulated as a min-cut / max-flow (graph cut) problem [8]. This problem is solved by using the alpha-beta swap. The depth and color information is used to provide the foreground / background segmentation. The graph cuts framework comprises of a set of nodes combined by weighted links (*n*-links). Each node is also connected to a source and a sink with weighted links (*t*-links). The graph cuts algorithm calculates the binary segmentation within the framework so that a subset of the nodes are connected to the source and the remaining nodes are connected to the sink. The binary segmentation requires that a subset of the *n*-links and *t*-links will need to be removed / cut. Since the input images use humans to interact with the objects, a skin detector [16] is needed to segment out a human's arms and hands from the scene. The skin detector used in this thesis is trained on 1000 images with hand-labeled skin and non-skin pixels.

The following equations describe the weights placed on the *t*-links to the source, (5.38), and the sink, (5.39).

$$w_{source} = \begin{cases} e^{\left(\frac{-(d_i-\mu_d)^2}{\sigma_t^2}\right)}, & \text{if not skin} \\ e^{\left(\frac{-(d_\infty-\mu_d)^2}{\sigma_t^2}\right)}, & \text{if skin} \end{cases} \tag{5.38}$$

$$w_{sink} = 1 - w_{source} \tag{5.39}$$

where d_i is the current depth pixel, μ_d is the average depth value in the image, σ_t is the deviation at which the object moves within the depth image, and d_∞ is the highest depth value captured by the sensor.

The next equation describes the weight placed on the k neighboring *n*-links.

$$w_k = e^{\left(\frac{-(d_i-d_{i+k})^2}{\sigma_n^2}+\frac{-(I_i-I_{i+k})^2}{\sigma_c^2}\right)} \tag{5.40}$$

where I_i is the current color pixel, d_{i+k} and I_{i+k} are the k-th neighboring pixels, σ_n and σ_c are the deviations of neighboring pixels in the depth image and color image, respectively. Figure 5.7 illustrates an example of a shirt segmentation using the graph cuts algorithm and the skin detector.

5.4.2 Mesh Model Generator

The initial mesh is created using the binary mask (described in section 5.4.1) of the first image in the video sequence using a mesh model generator (MMG). The MMG assumes that the initial configuration of the object is approximately fronto-parallel. The MMG automatically creates a triangular mesh by overlaying a grid of triangles on top of the initial configuration. All triangles that do not contain more than 10% of the object are omitted from the mesh model. The triangles that are not

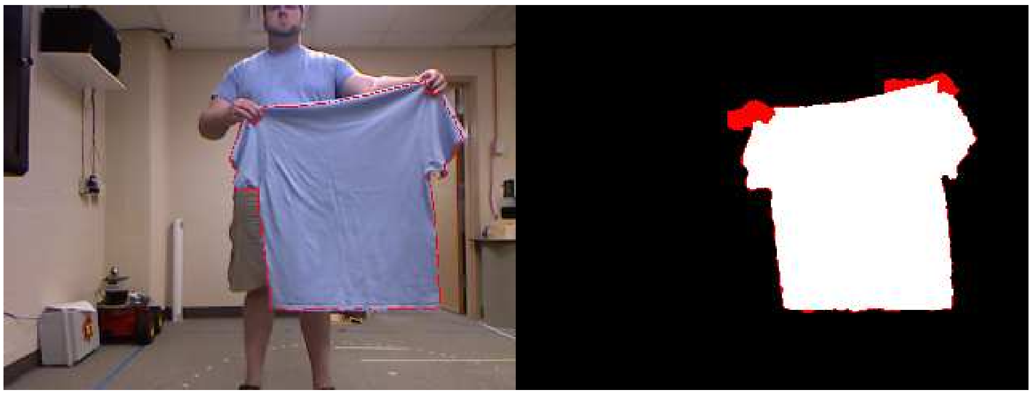


Figure 5.7: Left: Color image within a shirt sequence. Right: Segmentation of shirt from background while using the skin detector. White pixels represent foreground (shirt), black pixels represent background, and red pixels represent skin.

omitted make up the initial mesh, see Figure 5.8.

5.4.3 Reinitialization

When a large of amount of input data is lost, the algorithm halts momentarily and begins the reinitialization process. This process is designed to recover the configuration of the object after the transition of having one hand release the garment (after which gravity collapses it) then grasp it again and pull it so that the face garment is again visible. The reinitialization process attempts to calculate the current orientation of the object by matching a 3D model to the current input data. The 3D model is found by calculating the average canonical configuration of each object similar to the current object. A 3D plane is then calculated, using a least squares approach, to fit the 3D input data. The equation used to describe the plane is then used to determine the amount of rotation needed about the X, Y, and Z axis. The 3D model is then mapped to the input data and is used to create a new triangular mesh by applying the MMG, see section 5.4.2, to the model.

The reason behind using a known model, in this procedure, is to predefine

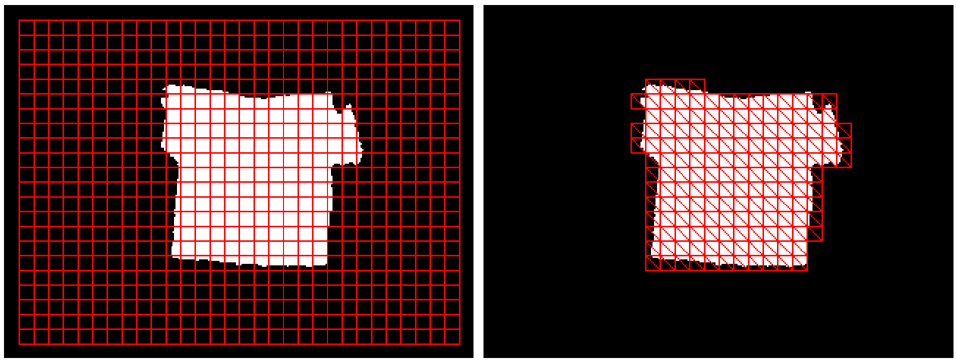


Figure 5.8: Left: Mesh model overlayed on top of object's mask before omitting any triangles. Right: Resulting mesh model after omitting triangles with less than 10% of object.

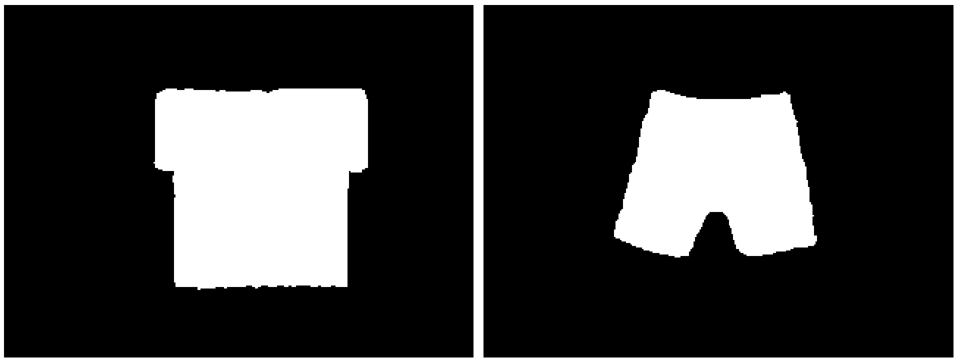


Figure 5.9: Left: 3D model for reinitializing a shirt sequence. Right: 3D model for reinitializing a shorts sequence.

areas of each object (e.g. collars, sleeves, belt loop, etc) that are lost. This process will allow for the mesh to be recreated, using the 3D model, and known areas relabeled before rotating the mesh about the X, Y, and Z axis. For example, if a particular grasp point or area of concentration is lost due to a loss in input data, then this process will allow for the algorithm to relocate that point or area. Once the new mesh model is created, the energy minimization algorithm continues again to estimate the configuration of the object. Figure 5.9 displays two examples of the 3D models used for reinitialization.

5.5 Pose estimation experiments of non-rigid objects

RGBD video sequences are captured of seven shirts, one pair of pants, two shorts and two posters to test the proposed method's ability to handle different non-rigid objects in a variety of scenarios (e.g. change in translation / scale; in-plane and out-of-plane rotation). The contributions made by the novel depth and boundary energy terms are verified to the accuracy of the estimated object configuration. For the experiments, the weights are set according to $\lambda_C = 1.3$, $\lambda_B = 0.8$, and $\lambda_D = 0.6$. Further results of the system can be seen in the online video. [1]

5.5.1 Illustrating the contribution of the depth term

Figure 5.10 compares the algorithm with and without the depth term, but with the smoothness and correspondence term in both cases. The depth term causes the triangular mesh model to adhere to the object along the z-axis perpendicular to the image plane. As shown in the figure, the resulting 3D meshes, more accurately, capture the current configuration of the object when the depth term is included. The improvement is especially noticeable in untextured areas.

5.5.2 Illustrating the contribution of the boundary term

Figure 5.11 compares the algorithm with and without the boundary term, using the smoothness, correspondence and depth term in both cases. The boundary term introduces lateral forces to cause the mesh to adhere to the object in areas near the contour of the object. The improvement resulting from the boundary term is

[1] `http://www.youtube.com/watch?v=1Y4EdZu1pwo`

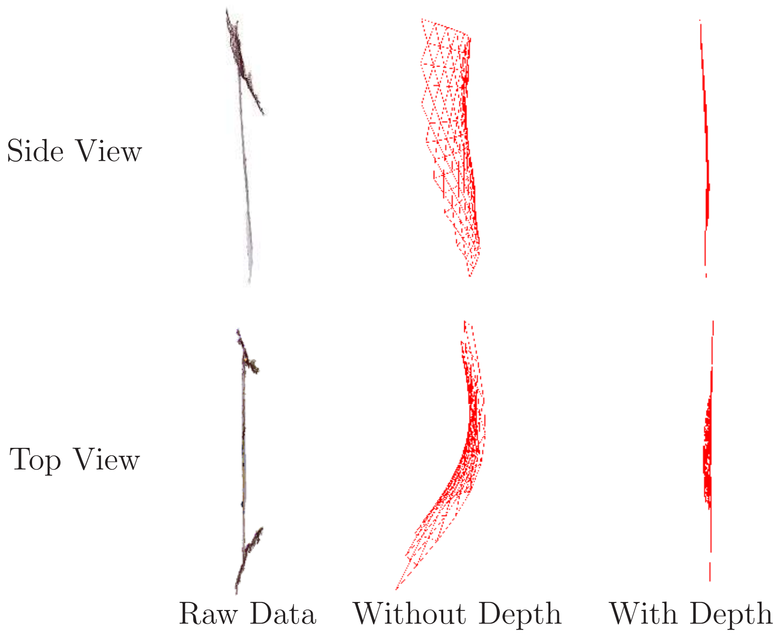


Figure 5.10: The depth term improves results significantly. From left to right: the input point cloud, the estimated mesh without using the depth term, and the estimated mesh using the depth term.

clear from the figure. Note also the subtle error along the top edge of the object, where the depth term alone raises the top corners (making it appear as though the top middle is sagging), as described in Figure 5.4.

5.5.3 Estimating the pose of different articles of clothing

In this experiment, the algorithm is tested on various image sequences involving seven shirts, one pair of pants, and two pairs of shorts. I create various changes in translation, scale, and rotation on each article of clothing to show how well the algorithm performs. Figures 5.12 and 5.13 display each image sequence with the resulting mesh output. Some of the meshes, in Figures 5.12 and 5.13, are blue and some are red, the reason being that the color of each mesh is selected to be of an opposing color, so that the color of the object is easily viewed.

5.5.4 Estimating the pose of posters

In this experiment, the algorithm is tested on another group of non-rigid objects, e.g. posters. This experiment tests my approach on semi-rigid poster board with out-of-plane oscillations. Figure 5.14 displays the results of two poster sequences.

5.5.5 Reinitializing a mesh after in-plane rotation

When the object is rotated in-plane and is momentarily held by just one arm, most of the information is lost in the image because the object is crumpled together. This loss of information requires a reinitialization phase whenever the object is regrasped by a second arm. Figure 5.15 illustrates three examples of in-plane rotation and how they result with and without the reinitialization process.

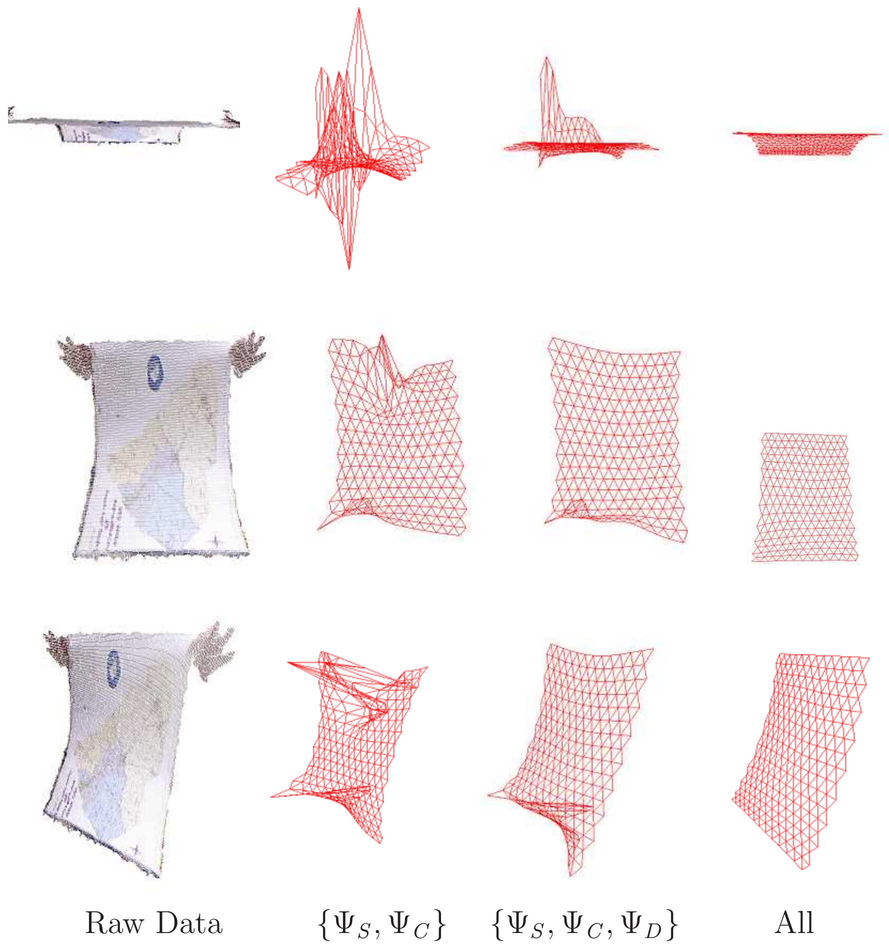


Figure 5.11: The improvement resulting from the various terms. From left to right: texture mapped point cloud of RGBD input data, the mesh resulting from only the smoothness and correspondence terms, the mesh from all but the boundary term, and the mesh resulting from all the terms. From top to bottom: top view, front view, and side view with 45^o pan.

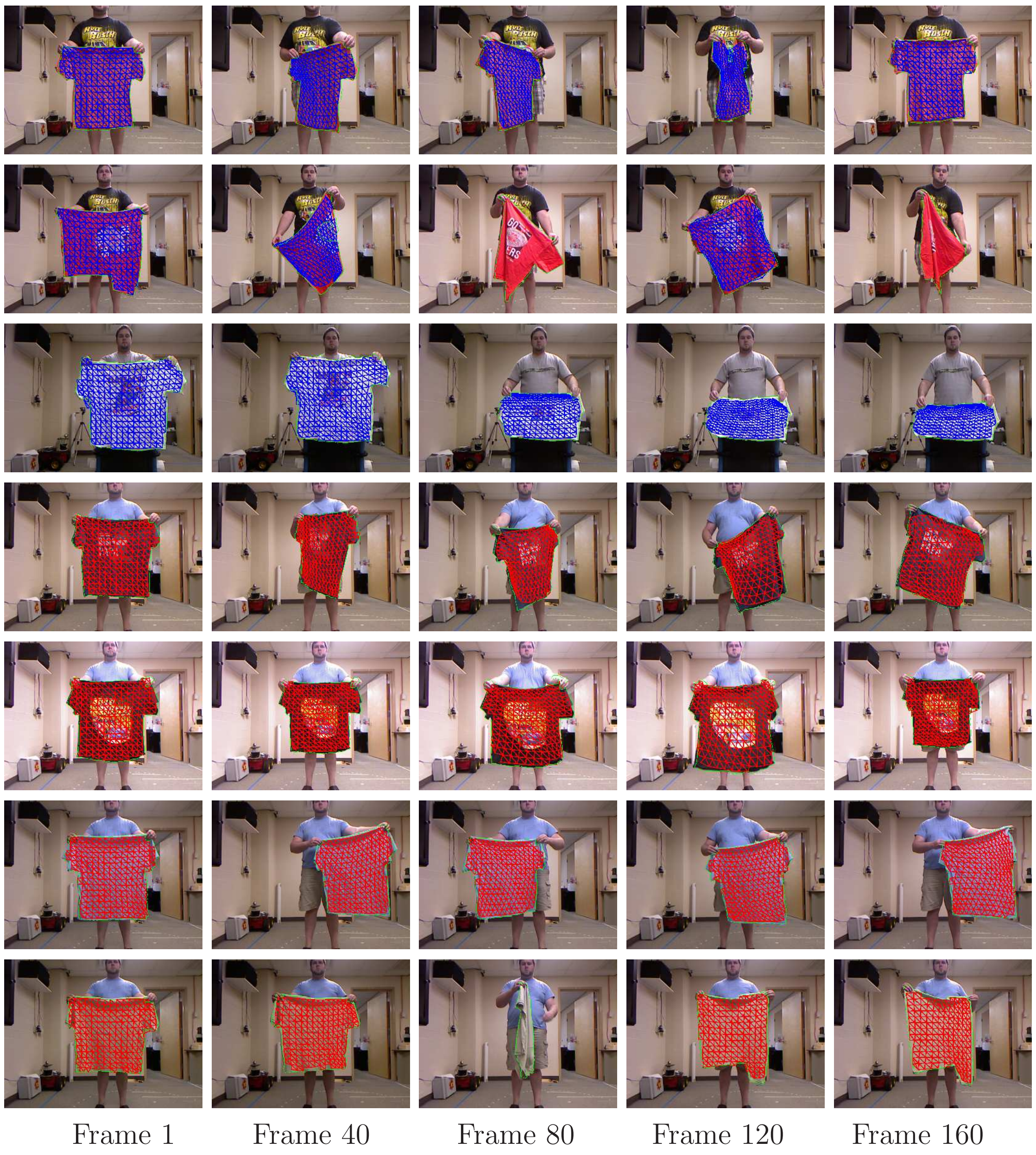


Figure 5.12: From left to right: Five images selected from image sequences involving clothing. From top to bottom: Orange shirt moving out-of-plane, orange shirt moving in-plane, white shirt used in [107], blue shirt moving in-plane and out-of-plane, black shirt changing scale, blue shirt translating from side to side, and green shirt moving in-plane. Some instances of each sequence do not have meshes due to a loss of data.

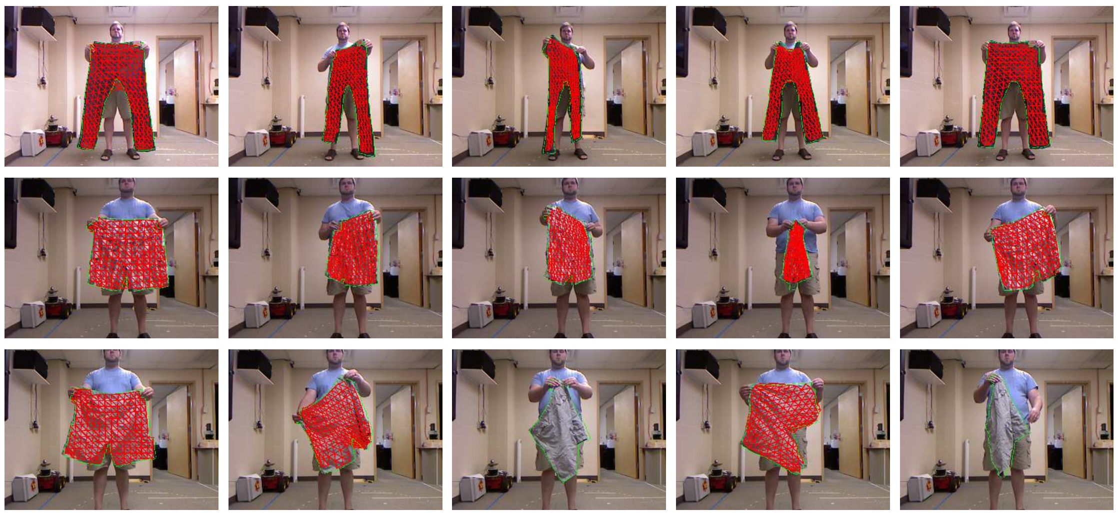


Figure 5.13: From left to right: Five images selected from image sequences involving clothing. From top to bottom: Pair of pants moving out-of-plane, checkered shorts moving out-of-plane, and gray shorts moving in-plane. Some instances of each sequence do not have meshes due to a loss of data.

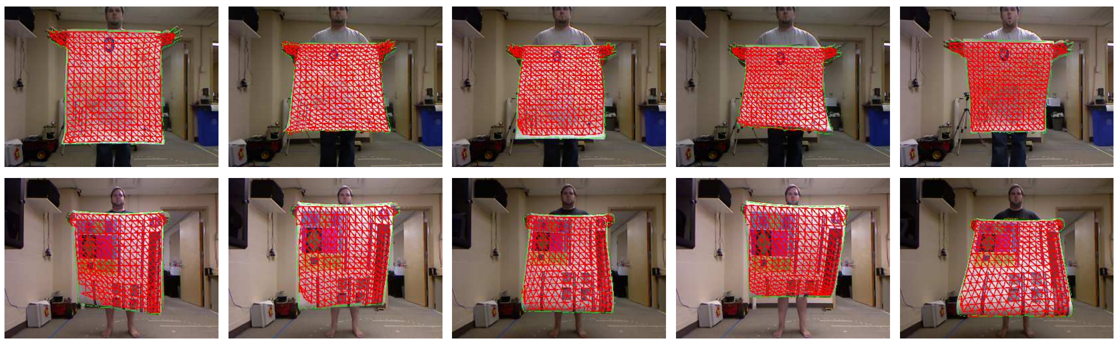


Figure 5.14: From left to right: Five images selected from image sequences involving posters. From top to bottom: White poster used in [107] and poster with various colored regions. Each poster moves out-of-plane of the camera.

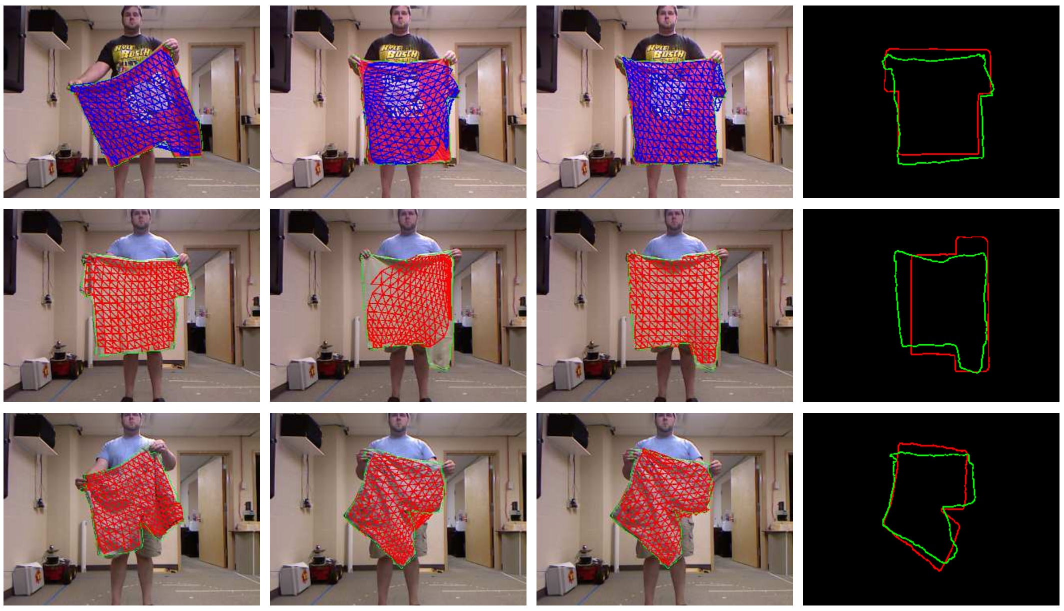

Frame i
Frame $i+20$ (w/o reinit)
Frame $i+20$ (w/ reinit)
Frame $i+20$ Model Contour vs. Actual Contour

Figure 5.15: From left to right: Image before in-plane rotation, image after in-plane rotation without reinitializing, image after in-plane rotation with reinitializing, and 3D model contour rotated to reinitialize (red outline is 3D model contour, green outline is actual contour). From top to bottom: Orange shirt ($i = 98$), green shirt ($i = 128$), and gray shorts ($i = 114$).

MSE ($pixels^2$)	$\Psi_S + \Psi_D +$ Ψ_C	$\Psi_S + \Psi_D +$ Ψ_B	$\Psi_S + \Psi_D +$ $\Psi_C + \Psi_B$
All Vertices	6.46	6.25	6.23
Boundary Vertices	6.75	6.19	5.93
Percentage Viewed	87.85	87.85	87.85

Table 5.1: Calculated mean squared error (MSE in $pixels^2$) by measuring the distance from actual points in the simulated point cloud to the mesh vertices. The results compare various implementations that are designed to isolate particular energy terms. While qualitatively it is difficult to see the difference in approaches (see Figure 5.16), we can quantitatively measure the increase in performance by lowering the MSE.

5.5.6 Quantitative results using a simulated model

This experiment is designed to compare the pose estimation mesh against a simulated shirt model with ground truth data. Various implementations of the algorithm are compared to determine which techniques improve the results of estimating the pose of an article of clothing. The different implementations involve the current approach using smoothness and depth terms, along with isolating the (1) correspondence term, the (2) boundary term, and (3) combining the correspondence and boundary term. It is not possible to compare our approach with other datasets since our algorithm depends on 3D image data instead of monocular video sequences [99, 91, 100]. Table 5.1 displays the mean squared error (MSE in $pixels^2$) that is calculated between the mesh vertices and the ground truth data points. Figure 5.16 illustrates six images from the current implementation, since the results of the other implementations look quite similar.

These results show that the approach derived in this thesis provides a small MSE when comparing the distances from all vertices and boundary vertices to the ground truth. While the first variation of the approach, in Table 5.1, provides a unique energy term of feature correspondence and the second variation of the ap-

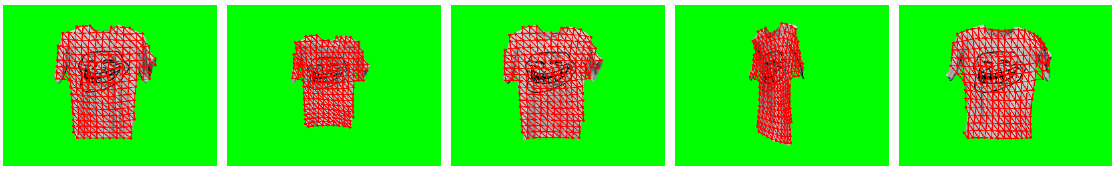


Figure 5.16: From left to right: Six images selected from an image sequence involving a simulated shirt. Only one sequence is shown because the output mesh from each of the implementations are very similar when overlaid on top of the object.

proach provides a unique energy term of extended boundary, the approach derived in this thesis combines these energy terms to improve the pose estimation of highly deformable objects (e.g. shirts, pants, shorts, posters).

Chapter 6

Conclusion

6.1 Clothing classification

Two novel methods to classifying clothing are proposed: (1) **L-M-H**, more specifically **L-S-C-H**, in which mid-level layers (i.e. physical characteristics and selection masks) are used to classify categories of clothing and (2) multiple interactions. The proposed systems use datasets of 2D color images and 3D point clouds of clothing captured from a Kinect sensor and a Logitech webcam. The thesis is focused on determining how well these new approaches improve classification over a baseline system. The datasets of clothing that are used in these experiments contain articles of clothing whose configurations are difficult to determine their corresponding category.

The overall improvement of these new approaches illustrate the critical importance of middle level information and physical interation. The addition of middle level features, termed characteristics and selection masks for this problem, enabled the classification process to improve, on average, by +27.47% for three categories, +17.90% for four categories, and +10.35% for seven categories, see Table 6.1. Increasing the number of categories, increases the number of characteristics for the algorithm to classify. With the increase in complexity, the average true positive rate decreases due to the difficulty of the problem. The addition of multiple interactions enabled the classification process to improve, on average, by 59% for six different categories.

Using these approaches could improve other types of classification / identification problems such as object recognition, object identification, person recognition, etc. The use of middle level features could range from one or more layers. This approach does not have to be limited to using only two layers of features (e.g. characteristics and selection masks). The notion of adding more features in between the low and high level may increase the percentage rate. Layers of filters, that could be used to distinquish between categories, could include separating adult clothing from child clothing or male clothing from female clothing. The use of multiple interactions could also apply to each article of clothing being held by more than one manipulator to provide different types of feature information. If articles of clothing are held by dual manipulators, then a more detailed shape descriptor could be used for classification.

Future extensions of these approaches include classification of sub-categories (e.g. types of shirts, types of dresses), age, gender, and the season of each article of clothing. The **L-C-S-H** experiments are used on a subset of characteristics that are useful for each category. Probably the amount of characteristics that are used in the mid-level layer correspond to the resulting classification percentages. The interactive perception experiments are used on a small dataset and can be studied further by applying this approach to a larger dataset of hanging articles of clothing. These approaches could be applied to separating clothing into three groups of darks, colors, and whites before the laundry is placed into the washing machine. These novel approaches can apply to grouping and labeling other rigid and non-rigid items beyond

Baseline			
# of categories	Local	Global	Local + Global
3	61.73	34.37	31.52
4	46.3	38.28	19.22
7	26.46	22.81	13.03

L-C-S-H			
# of categories	Local	Global	Local + Global
3	90.06	62.45	57.52
4	67.54	46.84	43.13
7	38.60	26.77	27.99

Increase from Baseline to L-C-S-H			
# of categories	Local	Global	Local + Global
3	+28.33	+28.08	+26.00
4	+21.24	+8.56	+23.91
7	+12.14	+3.96	+14.96

Table 6.1: Overview of results for varying numbers of categories: (A) 3, (B) 4, and (C) 7. Group (A) consists of shirts, socks, and dresses. Group (B) consists of group (A) plus cloths. Group (C) consists of group (B) plus pants, shorts, and jackets.

clothing.

6.2 Unfolding clothing

An approach to interactive perception is proposed in which a piece of cloth is flattened into a canonical position by pulling at various locations of the cloth. The algorithm is shown to provide an initial step in the process of unfolding / flattening a piece of laundry by using features of the cloth. The features used in this thesis are a handful of possible cues that could be used in the future to flatten a piece of laundry in fewer iterations. Other features that are considered, but not used, are a prior physical model of the cloth, the relationship between each corner and edge, and the physical features of the texture and material of the cloth.

Though this is a first step, future research in this novel approach of interactive perception would be directed towards using other types of laundry (i.e. shirts, pants, etc). Another direction could be to handle parts of clothing that are folded inside out, like the arm of a shirt or a leg on a pair of pants. I believe that these areas are fruitful extensions for future research.

6.3 Pose estimation

A new and novel algorithm that estimates the 3D configuration of a deformable object through an RGBD video sequence using feature point correspondence, depth, and boundary information is presented and demonstrated. This information is computed into energy terms that are used in a nonlinear energy function using a semi-implicit scheme. The proposed approach is quantitatively compared against various implementations of an energy function using a 3D simulation of a t-shirt. Combining energy terms from various implementations are shown to lower the mean squared error of the distance from mesh model points to ground truth points.

This approach is applied to various types of textured and textureless deformable objects to demonstrate the robustness of the algorithm. This approach is shown that it can be applied to a number of scenarios. For instance, service robots manipulate clothing when sorting, folding, and storing clothing in a laundry scenario. Also, an industrial robotic platform (e.g. textile environment or paper mill) handles different types of clothing and paper material on a day to day basis.

In the future, this research can be extended to handle a two-sided 3D triangular mesh that covers both the front and back of the object. This two-sided mesh would be applied to folding clothing in a service robot environment. Another step is to integrate this algorithm into a robotic system that can grasp and handle textured or textureless non-rigid objects in an unstructured environment.

Appendices

Appendix A

Vision Algorithms in this Thesis

The approaches discussed in this thesis are shown below in the form of algorithms. Each section will contain one or more algorithms that represent one of the previous chapters.

A.1 Classifying Clothing

For each approach used during the clothing classification part of this thesis, articles of clothing were isolated and visual data was captured for each one. Two separate approaches were used on clothing that was hanging and clothing that lying on the floor. The two algorithms in Figures A.1, A.2 and A.3 explain how each approach was performed.

A.2 Unfolding Clothing

For the approach used during the cloth unfolding part of this thesis, a simulated cloth was visually captured with an overhead camera. Each image is analyzed to

Algorithm: Classification for Hanging Clothes

Input: Front and side image of segmented clothing
Output: Estimated category of clothing

1. *Feature values:* For each segmented image
   (a) Calculate area of binary silhoutte for each image
   (b) Calculate Canny edges for each colored image and each binary silhoutte
2. *Match score:* For each combination of query image and dataset image
   (a) Calculate the match score between the front image feature values and j^{th} dataset image feature values
   (b) Calculate the match score between the side image feature values and j^{th} dataset image feature values
3. *Locate minimum score:* For one article of clothing
   (a) Search through the match score matrix of size $n \times 2$ for the smallest value
   (b) Define query category the same as the dataset category with the smallest value

Figure A.1: Clothing classification algorithm for the hanging position described in detail in Chapter 3

Algorithm: Classification for Lying Clothes - Training

Input: 2D color image and 3D point cloud of all segmented clothing in training set
Output: SVM model; Selection masks and mean vectors for each category

1. *Create feature vector:* For each segmented image
   (a) Calculate color histogram of clothing within HSV space (CH)
   (b) Calculate histogram of line lengths (HLL)
   (c) Calculate table point feature histogram (TPFH)
   (d) Calculate boundary information
   (e) Calculate fast point feature histogram (FPFH)
   (f) Calculate scale invariant feature transform (SIFT)
   (g) Combine FPFH and SIFT into a Bag-of-Words model
   (h) Create a final feature vector by combining CH, HLL, TPFH, boundary and Bag-of-Words model
2. *Train SVM for characteristics:* For each segmented image in the training set
   (a) Insert final feature vector into SVM of a particular characteristic
   (b) Assign 0 or 1 based on if article of clothing contains that particular characteristic
   (c) Repeat first two steps for all characteristics
   (d) Run SVM for all characteristics to obtain SVM model information
3. *Train K-means for categories:* For each segmented image in the training set
   (a) Calculate a binary vector for article of clothing consisting of 0's and 1's
   (b) Average all binary characteristic vectors for each category
4. *Calculate selection mask for categories:* For each segmented image in the training set
   (a) See description in detail at Algorithm 1

Figure A.2: Clothing classification algorithm for the lying position, training stage, described in detail in Chapter 3

```
Input: Characteristic vector for $w^{th}$ category ($CC_w$)
Output: Selection mask for $w^{th}$ category ($SM_w$)

$SM_w$=$2^{26} - 1$;
$RM_w$=$2^{27} - 1$;

foreach $Article_i \in Category_w$ do
    $NC_w = (CC_w \text{ x } SM_w) \cap RM$;
    if $Result(NC_w) \geq Result(CC_w)$ then
        $RM_w = RM_w \cap 2^{27} - 2^i - 1$;
    end
    $SM_w = SM_w >> 1$;
end
```

Algorithm 1: The process of **C-S** to determine the final selection mask for each category. CC_w, NC_w, SM_w, and RM_w are binary vectors that represent if a characteristic is used (1) or removed (0), out of the 27 listed in Table 3.2, for each category. >> stands for a bit shift in the binary vector. *Result*(.) calculates the resulting classification percentage after running the new selection mask on the training set.

Algorithm: Classification for Lying Clothes - Testing

Input: 2D color image and 3D point cloud of all segmented clothing in test set; SVM model; Selection masks and mean vectors for each category
Output: Estimated category of clothing for all articles of clothing in test set

1. *Test SVM for characteristics:* For each segmented image in the testing set
   (a) Insert final feature vector into SVM model of each characteristic
   (b) Take results of each characteristic SVM model and create output vector of binary values
2. *Test K-means for categories:* For each segmented image in the testing set
   (a) Take output vector of binary values (from previous step) and calculate nearest neighbor to each category mean vector
   (b) Category of nearest neighbor is determined to be category of segmented image

Figure A.3: Clothing classification algorithm for the lying position, testing stage, described in detail in Chapter 3

accurately locate a grasp point and move the cloth around on a table. The algorithm in Figure A.4 explains how the approach was performed.

A.3 Pose Estimation

For the approach used during the pose estimation part of this thesis, a non-rigid object was visually captured with a RGB-D sensor from a front-parallel position. Each image and point cloud is analyzed to accurately model the current configuration of the object as it moves through a video sequence. The algorithm in Figure A.5 explains how the approach was performed.

```
Algorithm: Unfolding a piece of cloth on a table
Input: Depth image of segmented cloth
Output: Grasp point and movement orientation

  1. First phase of unfolding: For each segmented image
      (a) Locate all corner positions on cloth
      (b) Set $d$ from 0 to 315 degrees, increasing by steps of 45 degrees:
            i. Find point on cloth that is $d$ degrees from centroid
           ii. Calculate closest corner to previously found point
          iii. Grasp corner and move object outward from the centroid at $d$ degrees
  2. Second phase of unfolding: For each segmented image
      (a) Capture featured regions of the cloth: For each segmented image
            i. Calculate peak ridge of object using depth image
           ii. Calculate corner locations along border
          iii. Find all continuous regions along the object
           iv. Find intersection of continous regions and peak region
            v. Find all corner locations on continuous peak region ($C_C() = 1$)
      (b) Determine grasp point and orientation: For each segmented image
            i. Randomly choose a single corner location from $C_C$
           ii. Grasp corner
          iii. Movement and orientation is decided on whether all of the corner
               locations are located on the peak region
           iv. If so, then pull away from centroid
            v. If not, then pull over the centroid
```

Figure A.4: Cloth unfolding algorithm described in detail in Chapter 4

Algorithm: Estimating the pose of a non-rigid object

Input: 2D color image and 3D point cloud of segmented non-rigid object
Output: Estimated model of 3D pose of non-rigid object

1. *Initialize mesh model for each segmented non-rigid object*
2. *Capture feature information:* For each segmented non-rigid object
   (a) Find SURF feature correspondence from previous image in sequence
   (b) Calculate boundary location of segmented object
   (c) Calculate depth information from point cloud
3. *Calculate energy equation using feature information for each segmented non-rigid object*
4. *Minimize energy terms:* For each segmented non-rigid object
   (a) Differentiate energy equation and solve for 0
   (b) Use semi-implicit scheme to calculate the new mesh model coordinates
   (c) Iterate process until converge or *MAX* iterations is reached
5. *Update 3D coordinates of mesh model for each segmented non-rigid object*

Figure A.5: Pose estimation algorithm for non-rigid objects described in detail in Chapter 5

Bibliography

[1] I. Akhter, Y. Sheikh, S. Khan, and T. Kanade. Trajectory space: A dual representation for nonrigid structure from motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(7):1442–1456, July 2011.

[2] D. J. Balkcom and M. T. Mason. Robotic origami folding. *Inter. J of Robotics Research*, 27(5):613–627, May 2008.

[3] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool. SURF: Speeded up robust features. *Computer Vision and Image Understanding*, 110(3):346–359, June 2008.

[4] O. B. Bayazit, J.-M. Lien, and N. M. Amato. Probabilistic roadmap motion planning for deformable objects. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 2126– 2133, May 2002.

[5] C. Bersch, B. Pitzer, and S. Kammel. Bimanual robotic cloth manipulation for laundry folding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1413–1419, Sept. 2011.

[6] M. Bordegoni, G. Frugoli, and C. Rizzi. Direct interaction with flexible material models. In *Proceedings of the Seventh International Conference on Human-Computer Interaction (HCI)*, pages 387–390, Aug. 1997.

[7] G. Borgefors. Distance transformations in digital images. *Computer Vision, Graphics, and Image Processing*, 34(3):344–371, 1986.

[8] Y. Boykov and V. Kolmogorov. 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.

[9] A. D. Bue. Adaptive metric registration of 3D models to non-rigid image trajectories. In *Proceedings of the European Conference on Computer Vision*, pages 87–100, 2010.

[10] A. D. Bue and A. Bartoli. Multiview 3D warps. In *Proceedings of the International Conference on Computer Vision*, pages 87–100, Nov. 2011.

[11] J. F. Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(6):679–698, 1986.

[12] C. C. Chang and C. J. Lin. LIBSVM: a library for support vector machines, 2001.

[13] E. Charniak and D. McDermott. *Introduction to Artificial Intelligence.* Addison-Wesley, 1985.

[14] D. L. Chester. Why two hidden layers are better than one? In *Proceedings of the International Joint Conference on Neural Networks*, pages 1265–1268, 1990.

[15] T. Collins and A. Bartoli. Locally affine and planar deformable surface reconstruction from video. In *Proceedings of the International Workshop on Vision, Modeling and Visualization*, 2010.

[16] C. Conaire, N. O'Connor, and A. Smeaton. Detector adaptation by maximising agreement between independent data sources. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–6, June 2007.

[17] M. Cusumano-Towner, A. Singh, S. Miller, J. F. O'Brien, and P. Abbeel. Bringing clothing into desired configurations with limited perception. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, May 2011.

[18] S. Deerwester, S. Dumais, G. Furnas, T. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41(6):391–407, 1990.

[19] A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, Series B (Methodological)*, 39(1):1–22, 1977.

[20] P. A. Devijver and J. Kittler. *Pattern Recognition: A Statistical Approach.* Prentice Hall, London, 1982.

[21] K. Duan and S. S. Keerthi. Which is the best multiclass SVM method? an empirical study. In *Proceedings of the Sixth International Workshop on Multiple Classifier Systems*, pages 278–285, 2005.

[22] C. Elbrechter, R. Haschke, and H. Ritter. Bi-manual robotic paper manipulation based on real-time marker tracking and physical modelling. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1427–1432, Sept. 2011.

[23] A. Elisseeff and H. Paugam-Moisy. Size of multilayer networks for exact learning: analytic approach. *Advances in Neural Information Processing Systems*, 9:162–168, 1996.

[24] J. Fayad, L. Agapito, and A. D. Bue. Piecewise quadratic reconstruction of non-rigid surfaces from monocular sequences. In *Proceedings of the European Conference on Computer Vision*, 2010.

[25] J. Fayad, A. D. Bue, L. Agapito, and P. Aguiar. Non-rigid structure from motion using quadratic deformation models. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

[26] J. Fayad, C. Russell, and L. Agapito. Automated articulated structure and 3D shape recovery from point correspondences. In *Proceedings of the International Conference on Computer Vision*, pages 431–438, Nov. 2011.

[27] P. Felzenszwalb and D. Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[28] R. Ferreira, J. Xavier, and J. Costeira. Reconstruction of isometrically deformable flat surfaces in 3D from multiple camera images. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, 2009.

[29] R. Ferreira, J. Xavier, and J. Costeira. Shape from motion of nonrigid objects: The case of isometrically deformable flat surfaces. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

[30] P. Fitzpatrick. First contact: An active vision approach to segmentation. In *International Conference on Intelligent Robots and Systems (IROS)*, pages 2161–2166, 2003.

[31] P. Fong. Sensing, acquisition, and interactive playback of data-based models for elastic deformable objects. *International Journal of Robotics Research*, 28(5):622–629, May 2009.

[32] M. Fontana, C. Rizzi, and U. Cugini. 3D virtual apparel design for industrial applications. *Computer-Aided Design*, 37(6):609–622, 2005.

[33] P. Fua. Combining stereo and monocular information to compute dense depth maps that preserve depth discontinuities. In *Proceedings of the 12th International Joint Conference on Artificial Intelligence*, pages 1292–1298, 1991.

[34] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, Jan. 1992.

[35] P. Gibbons, P. Culverhouse, and G. Bugmann. Visual identification of grasp locations on clothing for a personal robot. In *Conf. Towards Autonomous Robotic Systems (TAROS)*, pages 78–81, Aug. 2009.

[36] A. Gidudu, G. Hulley, and T. Marwala. Image classification using SVMs: One-against-One vs One-against-All. In *Proccedings of the 28th Asian Conference on Remote Sensing*, 2007.

[37] P. Gotardo and A. Martinez. Computing smooth time-trajectories for camera and deformable shape in structure from motion with occlusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(10):2051–2065, Oct. 2011.

[38] P. Gotardo and A. Martinez. Kernel non-rigid structure from motion. In *Proceedings of the International Conference on Computer Vision*, pages 802–809, Nov. 2011.

[39] P. Gotardo and A. Martinez. Non-rigid structure from motion with complementary rank-3 spaces. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3065–3072, June 2011.

[40] C. G. Harris and M. Stephens. A combined corner and edge detector. In *Proceedings of the 4th Alvey Vision Conference*, pages 147–151, 1988.

[41] S. Hata, T. Hiroyasu, J. Hayash, H. Hojoh, and T. Hamada. Flexible handling robot system for cloth. In *International Conf. of Mechatronics and Automation*, pages 49–54, 2009.

[42] T. Hofmann. Probabilistic latent semantic analysis. In *Proc. of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, pages 289–296, 1999.

[43] T. Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1999.

[44] T. Hofmann, J. Puzicha, and M. I. Jordan. *Unsupervised learning from dyadic data*, volume 11. 1999.

[45] C. Holleman, L. E. Kavraki, and J. Warren. Planning paths for a flexible surface patch. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 21–26, May 1998.

[46] A. Howard and G. Bekey. Recursive learning for deformable object manipulation. In *Proceedings of the 8th International Conference on Advanced Robotics (ICAR)*, pages 939–944, July 1997.

[47] M. Kaneko and M. Kakikura. Planning strategy for unfolding task of clothes - isolation of clothes from washed mass -. In *13th Annual Conference of RSJ*, number 1, pages 455–456, 1996.

[48] M. Kaneko and M. Kakikura. Planning strategy for putting away laundry - isolating and unfolding task. In *Proc. of the 4th IEEE I.S. Assembly and Task Planning*, pages 429–434, 2001.

[49] M. Kass, A. Witkin, and D. Terzopoulos. Snakes: Active contour models. *International Journal of Computer Vision*, 1(4):321–331, 1988.

[50] D. Katz and O. Brock. Manipulating articulated objects with interactive perception. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 272–277, May 2008.

[51] J. Kenney, T. Buckley, and O. Brock. Interactive segmentation for manipulation in unstructured environments. In *International Conference on Robotics and Automation (ICRA)*, pages 1377–1382, 2009.

[52] K. Kimura, S. Kikuchi, and S. Yamasaki. Accurate root length measurement by image analysis. *Plant and Soil*, 216(1):117–127, 1999.

[53] Y. Kita and N. Kita. A model-driven method of estimating the state of clothes for manipulating it. In *Proc. of 6th Workshop on Applications of Computer Vision*, pages 63–69, 2002.

[54] Y. Kita, F. Saito, and N. Kita. A deformable model driven visual method for handling clothes. In *Inter. Conf. on Robotics and Automation*, pages 3889–3895, 2004.

[55] Y. Kita, T. Ueshiba, E. Neo, and N. Kita. Clothes state recognition using 3d observed data. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 1220–1225, May 2009.

[56] Y. Kita, T. Ueshiba, E. Neo, and N. Kita. A method for handling a specific part of clothing by dual arms. In *Conf. on Intelligent Robots and Systems (IROS)*, pages 3403–3408, 2009.

[57] H. Kobayashi, S. Hata, H. Hojoh, T. Hamada, and H. Kawai. A study on handling system for cloth using 3-d vision sensor. In *Proceedings of IEEE IECON*, pages 4180–4185, 2008.

[58] N. Kumar, A. C. Berg, P. N. Belhumeur, and S. K. Nayar. Attribute and simile classifiers for face verification. In *Proceedings of the International Conference on Computer Vision*, pages 365–372, 2009.

[59] S. Lawrence, C. Giles, and A. Tsoi. Lessons in neural network training: Overfitting may be harder than expected. In *Proceedings of the Fourteenth National Conference on Artificial Intelligence*, pages 540–545, 1997.

[60] X. Lladó, A. D. Bue, and L. Agapito. Non-rigid metric reconstruction from perspective cameras. *Image and Vision Computing*, 28(9):1339–1353, Sept. 2010.

[61] X. Lladó, A. D. Bue, A. Oliver, J. Salvi, and L. Agapito. Reconstruction of non-rigid 3D shapes from stereo-motion. *Pattern Recognition Letters*, 32(7):1020–1028, May 2011.

[62] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[63] J. MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[64] J. Maitin-Shepard, M. Cusumano-Towner, J. Lei, and P. Abbeel. Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *International Conference on Robotics and Automation (ICRA)*, 2010.

[65] J. Milgram, M. Cheriet, and R. Sabourin. One Against One or One Against All: Which one is better for handwriting recognition with SVMs? In *Tenth International Workshop on Frontiers in Handwriting Recognition*, Oct. 2006.

[66] S. Miller, M. Fritz, T. Darrell, and P. Abbeel. Parametrized shape models for clothing. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 4861–4868, May 2011.

[67] S. Miller, J. van den Berg, M. Fritz, T. Darrell, K. Goldberg, and P. Abbeel. A geometric approach to robotic laundry folding. *International Journal of Robotics Research (IJRR)*, 31(2):249–267, Feb. 2012.

[68] M. Moll and L. E. Kavraki. Path planning for deformable linear objects. *IEEE Transactions on Robotics*, 22(4):625–636, Aug. 2006.

[69] R. Ohlander, K. E. Price, and D. R. Reddy. Picture segmentation using a recursive region splitting method. *Computer Graphics and Image Processing*, 8(3):313–333, Dec. 1978.

[70] F. Osawa, H. Seki, and Y. Kamiya. Clothes folding task by tool-using robot. *Journal of Robotics and Mechatronics*, 18(5):618–625, 2006.

[71] F. Osawa, H. Seki, and Y. Kamiya. Unfolding of massive laundry and classification types by dual manipulator. *J of Advanced Computational Intelligence and Intelligent Informatics*, 11(5):457–463, 2007.

[72] J. Ostlund, A. Varol, and P. Fua. Laplacian meshes for monocular 3D shape recovery. In *Proceedings of the European Conference on Computer Vision*, 2012.

[73] M. Paladini, A. Bartoli, and L. Agapito. Sequential non-rigid structure-from-motion with the 3D-implicit low-rank shape model. In *Proceedings of the European Conference on Computer Vision*, 2010.

[74] M. Paladini, A. D. Bue, M. Stošić, M. Dodig, J. Xavier, and L. Agapito. Factorization for non-rigid and articulated structure using metric projections. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2898–2905, June 2009.

[75] K. Paraschidis, N. Fahantidis, V. Petridis, Z. Doulgeri, L. Petrou, and G. Hasapis. A robotic system for handling textile and non rigid flat materials. *Computers in Industry*, 26:303–313, 1995.

[76] H. Park, T. Shiratori, I. Matthews, and Y. Sheikh. 3d reconstruction of a moving point from a series of 2d projections. In *Proceedings of the European Conference on Computer Vision*, 2010.

[77] H. S. Park and Y. Sheikh. 3D reconstruction of a smooth articulated trajectory from a monocular image sequence. In *Proceedings of the International Conference on Computer Vision*, pages 201–208, Nov. 2011.

[78] J. Pilet, V. Lepetit, and P. Fua. Fast non-rigid surface detection, registration and realistic augmentation. *International Journal of Computer Vision*, 76(2):109–122, Feb. 2008.

[79] G. Qui. Indexing chromatic and achromatic patterns for content-based colour image retrieval. *Pattern Recognition*, 35:1675–1686, 2002.

[80] A. Ramisa, G. Alenyá, F. Moreno-Noguer, and C. Torras. Using depth and appearance features for informed robot grasping of highly wrinkled clothes. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 1703–1708, May 2012.

[81] R. Rifkin and A. Klautau. In defense of One-vs-All classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[82] I. Rish. An empirical study of the naive bayes classifier. In *Proceedings of IJCAI Workshop on Empirical Methods in Artificial Intelligence*, 2001.

[83] C. Russell, J. Fayad, and L. Agapito. Energy based multiple model fitting for non-rigid structure from motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3009–3016, June 2011.

[84] R. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (FPFH) for 3D registration. *Intelligent Autonomous Systems IAS*, 2009.

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

[86] M. Saha and P. Isto. Motion planning for robotic manipulation of deformable linear objects. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, pages 2478–2484, May 2006.

[87] K. Salleh, H. Seki, Y. Kamiya, and M. Hikizu. Tracing manipulation in clothes spreading by robot arms. *Journal of Robotics and Mechatronics*, 18(5):564–571, 2006.

[88] K. Salleh, H. Seki, Y. Kamiya, and M. Hikizu. Tracing manipulation of deformable objects using robot grippers with roller fingertips. In *International Joint Conference on SICE-ICASE*, pages 5882–5887, Oct. 2006.

[89] K. Salleh, H. Seki, Y. Kamiya, and M. Hikizu. Inchworm robot grippers for clothes manipulation. *Artificial Life and Robotics*, 12(1–2):142–147, 2008.

[90] G. Salton and M. J. McGill. *Introduction to Modern Information Retrieval.* McGraw-Hill, 1983.

[91] M. Salzmann and P. Fua. Linear local models for monocular reconstruction of deformable surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(5):931–944, May 2011.

[92] M. Salzmann, F. Moreno-Noguer, V. Lepetit, and P. Fua. Closed-form solution to non-rigid 3D surface registration. In *Proceedings of the European Conference on Computer Vision*, Oct. 2008.

[93] M. Salzmann, J. Pilet, S. Ilic, and P. Fua. Surface deformation models for non-rigid 3-D shape recovery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(8):1481–1487, Aug. 2007.

[94] M. Salzmann, R. Urtasun, and P. Fua. Local deformation models for monocular 3D shape recovery. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2008.

[95] A. Saxena, J. Driemeyer, and A. Y. Ng. Robotic grasping of novel objects using vision. *International Journal of Robotics Research*, 27:157–173, Feb. 2008.

[96] D. Scharstein and R. Szeliski. A taxonomy and evaluation of dense two-frame stereo correspondence algorithms. *International Journal of Computer Vision*, 47(1):7–42, 2002.

[97] J. Taylor, A. Jepson, and K. Kutulakos. Non-rigid structure from locally-rigid motion. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2761–2768, June 2010.

[98] United States Postal Service. Facts and figures about your postal service (Postal Facts 2011). http://about.usps.com/who-we-are/postal-facts/welcome.htm.

[99] A. Varol, M. Salzmann, E. Tola, and P. Fua. Template-free monocular reconstruction of deformable surfaces. In *Proceedings of the International Conference on Computer Vision*, pages 1811–1818, Oct. 2009.

[100] A. Varol, A. Shaji, M. Salzmann, and P. Fua. Monocular 3D reconstruction of locally textured surfaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(6):1118–1130, June 2012.

[101] H. M. Wallach. Topic modeling: Beyond bag-of-words. In *In Proceedings of the 23rd International Conference on Machine Learning*, pages 977–984, 2006.

[102] P. C. Wang, S. Miller, M. Fritz, T. Darrell, and P. Abbeel. Perception for the manipulation of socks. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Sept. 2011.

[103] B. Willimon, S. Birchfield, and I. Walker. Rigid and non-rigid classification using interactive perception. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1728–1733, 2010.

[104] B. Willimon, S. Birchfield, and I. Walker. Classification of clothing using interactive perception. In *International Conf. on Robotics and Automation (ICRA)*, pages 1862–1868, 2011.

[105] B. Willimon, S. Birchfield, and I. Walker. Model for unfolding laundry using interactive perception. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2011.

[106] B. Willimon, S. Birchfield, and I. Walker. Interactive perception of rigid and non-rigid objects. *International Journal of Advanced Robotic Systems (IJARS)*, 2012.

[107] B. Willimon, S. Hickson, I. Walker, and S. Birchfield. An energy minimization approach to 3D non-rigid deformable surface estimation using RGBD data. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

[108] B. Willimon, I. Walker, and S. Birchfield. 3D non-rigid deformable surface estimation. *Submitted to Autonomous Robotics (AURO).*

[109] B. Willimon, I. Walker, and S. Birchfield. 3D non-rigid deformable surface estimation without feature correspondence. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, 2013.

[110] B. Willimon, I. Walker, and S. Birchfield. A new approach to clothing classification using mid-level layers. In *Proceedings of the International Conference on Robotics and Automation(ICRA)*, 2013.

[111] B. Willimon, I. Walker, and S. Birchfield. Classification of clothing using midlevel layers. *ISRN Robotics*, vol. 2013, Article ID 630579, 17 pages, 2013. doi:10.5402/2013/630579.

[112] R. B. Willimon. Interactive perception for cluttered environments. Master's thesis, Clemson University, Dec. 2009.

[113] K. Yamakazi and M. Inaba. A cloth detection method based on image wrinkle feature for daily assistive robots. In *IAPR Conference on Machine Vision Applications*, pages 366–369, 2009.

[114] Y. Yoshida, J. Hayashi, S. Hata, H. Hojoh, and T. Hamada. Status estimation of cloth handling robot using force sensor. In *International Symp. on Industrial Electronics (ISIE)*, pages 339–343, 2009.