Czech Technical University in Prague
Faculty of Electrical Engineering
Department of Cybernetics

Visual Perception of Garments for their Robotic Manipulation

Doctoral Thesis

Jan Stria

Advisor: prof. Ing. Václav Hlaváč, CSc.
Advisor specialist: doc. RNDr. Daniel Průša, Ph.D.

Study programme: Electrical Engineering and Information Technology
Field of study: Artificial Intelligence and Biocybernetics

Prague, September 2019

Abstract

The presented work addresses the visual perception of garments applied for their robotic manipulation. Various types of garments are considered in the typical perception and manipulation tasks, including their classification, folding or unfolding. Our work is motivated by the possibility of having humanoid household robots performing these tasks for us in the future, as well as by the industrial applications. The main challenge is the high deformability of garments, which can be posed in infinitely many configurations with a significantly varying appearance.

The thesis deals with several perception sub-tasks included in the robotic folding scenario, where the goal is bringing a heap of unknown crumpled garments to a folded stack. A custom built dual-arm robot, equipped with consumer grade sensors, is employed, mimicking reasoning and actions of humans. The thesis addresses namely category classification of a hanging garment, visual detection and analysis of folds needed for their removal and pose estimation of a spread garment for its folding. An integration of the sub-tasks into a state-of-the-art pipeline for the folding scenario is also described.

The pose estimation of a spread garment utilizes a single image. A novel two-stage segmentation algorithm is proposed. A probabilistic model of the background color is learned and used for an automated initialization of the existing segmentation method, which would require a user input otherwise. The extracted garment contour is matched to a newly developed polygonal model. Its vertices are defined manually, while their mutual positions are learned from data. Two matching procedures based on a dynamic programming are introduced, minimizing specific cost functions. Vertices of the fitted model correspond to landmark points on the garment contour. They are used for folds planning. The garment pose is checked after each fold by fitting a folded model.

The inverse task is the unfolding of a partially folded garment. The folded parts of the garment are detected by analyzing its visible surface and assigning the pixels either to the bottom or top folded layer. The optimum assignment is found by the graph based minimization of a specific energy function. It combines image and depth information. Its parameters are determined automatically from data. The folding axis is estimated by analyzing the shape of a virtually unfolded garment.

The category classification of a hanging garment utilizes 3D point cloud, obtained by fusing depth maps taken from multiple viewpoints. The cloud is processed with a novel convolutional neural network. It utilizes a generalized convolution operation defined over the spatially local neighborhood of each point. The point cloud is repeatedly subsampled by the network, while size of the neighborhood grows in deeper layers. The obtained local descriptors are aggregated to a single global feature vector. The network is trained on a dataset of common 3D objects and transfered to the domain of garments, in which the convolutional layers extract features for a linear classifier.

Abstrakt

Tématem předložené práce je strojové vnímání textilií založené na obrazové informaci a využité pro jejich robotickou manipulaci. Práce studuje několik reprezentativních textilií v běžných kognitivně-manipulačních úlohách, jako je například třídění neznámých oděvů podle typu nebo jejich skládání. Některé z těchto činností by v budoucnu mohly být vykonávány domácími robotickými pomocníky. Strojová manipulace s textiliemi je poptávaná také v průmyslu. Hlavní výzvou řešeného problému je měkkost a s tím související vysoká deformovatelnost textilií, které se tak mohou nacházet v bezpočtu vizuálně velmi odlišných stavů.

Práce se zabývá několika dílčími úkoly strojového vnímání vycházejícími ze scénáře robotického skládání haldy zmuchlaných kusů oděvu, jejichž typ není předem znám. V experimentech využíváme pro tento účel postaveného dvourukého robotu, vybaveného běžně dostupnými senzory. Robot s prádlem manipuluje podobným způsobem jako lidé. V práci se zabýváme především klasifikací typu visícího oděvu, detekcí skladů na částečně složeném kusu oděvu nutnou pro jeho rozložení a odhadem konfigurace rozloženého oděvu pro jeho skládání. Popisujeme také integraci těchto dílčích úloh do unikátního funčního celku.

Vstupem metody pro odhad konfigurace rozprostřeného kusu oděvu je jediný snímek, který je segmentován ve dvou krocích. Využíváme naučeného pravděpodobnostního modelu barvy pozadí sloužího k automatické inicializaci jinak manuální segmentační metody. Takto nalezený vnější okraj je poté slícován se speciálně vyvinutým polygonálním modelem tvaru oděvu. Zatímco vrcholy modelu definujeme ručně, jejich vzájemná pozice je naučena z dat. V práci představujeme dvě různé metody pro lícování modelu na obrys oděvu. Obě jsou založeny na minimalizaci určité cenové funkce za pomoci dynamického programování. Vrcholy modelu, odpovídající zvoleným význačným bodům na vnějším obrysu daného typu oděvu, jsou následně využity pro plánování skladů. Konfigurace oděvu je kontrolována po každém jednotlivém skladu na základě lícování částečně složeného modelu.

V jistém smyslu opačnou úlohou je rozkládání částečně složeného oděvu. Detekce jednotlivých skladů je založena na anayze viditelné části povrchu oděvu. Jednotlivé obrazové body jsou přiřazeny buďto do spodní, nebo do svrchní přeložené vrstvy. Optimální zařazení obrazových bodů do vrstev je formulováno jako grafová úloha minimalizace určité cenové funkce, která zohledňuje obrazovou i hloubkovou informaci. Parametry funkce odhadujeme automaticky z pozorovaných dat. Oděv je následně několikrát virtuálně rozložen s cílem identifikovat skutečnou osu skladu.

Vstupem metody pro rozpoznání typu visícího oděvu je množina 3D bodů. Body jsou zrekonstruovány z hloubkových dat zachycených z mnoha různých pohledů. Množina bodů je poté zpracována konvoluční neuronovou sítí. Ta využívá nově vyvinuté operace konvoluce definované na okolí jednotlivých bodů. Velikost okolí v hlubších vrstvách roste, zatímco počet bodů postupně snižujeme pomocí náhodného vzorkování. Lokální příznaky popisující okolí bodů jsou následně zkombinovány do globálního vektoru. Parametry sítě učíme na velké databázi obecných 3D objektů. Při rozpoznávání oděvů konvoluční vrstvy slouží k extrakci příznaků vstupujících do lineárního klasifikátoru.

Authorship

I hereby certify that the results presented in this thesis were achieved during my own research in cooperation with my thesis advisor prof. Ing. Václav Hlaváč, CSc. and my advisor specialist doc. RNDr. Daniel Průša, Ph.D. The robotic experiments presented in the thesis were performed with the help of the CloPeMa project team, namely my colleagues Ing. Vladimír Petrík, Ph.D. and Ing. Libor Wagner.

Acknowledgement

Foremost, I would like to thank my thesis advisor prof. Ing. Václav Hlaváč, CSc. for his support during my doctoral studies, for sharing his rich experience with me, and mainly for his patience. I would like to thank my advisor specialist doc. RNDr. Daniel Průša, Ph.D for introducing me to research and helping me to achieve and publish my first own scientific results.

I would like to thank to the members of the CloPeMa project team, who dared to fight the complex task of autonomous robotic manipulation of soft materials, providing an interesting research topic for my thesis. I would namely like to thank to my colleagues Ing. Vladimír Petrík, Ph.D and Ing. Libor Wagner, who were responsible for planning of the manipulation tasks and who made it possible to perform the robotic experiments.

I gratefully acknowledge financial support provided by the project FP7-ICT-288553 CloPeMa (Clothes Perception and Manipulation) funded by European Commission, project CZ.02.1.01/0.0/0.0/15_003/0000468 IMPACT (Intelligent Machine Perception) funded by European Regional Development Fund, project TE01020197 CAK (Center for Applied Cybernetics) funded by Technology Agency of the Czech Republic, and projects SGS13/205/OHK3/3T/13, SGS15/204/OHK3/3T/13, SGS18/205/OHK3/3T/37 that were funded by Grant Agency of the Czech Technical University in Prague.

Last but not least, I would like to thank my family. Above all, I express my gratitude to my wife Eva, who supported me during my seemingly never-ending studies and work on this thesis by all means.

Contents

1. Introduction

The proposed work deals with the visual perception of garments using conventional cameras and depth sensors. The perception is an important component of the robotic systems which are able to manipulate garments autonomously. It is crucial for the reasoning, decision making or motion planning. It comprehends various tasks, including the localization of a garment in the working area of a robot, classification of its category, pose estimation or detection of possible grasping points.

1.1. Challenges

Both the visual perception and robotic manipulation of garments are challenging tasks. The challenges are due to the softness of materials used, which makes the garments highly deformable. The visual appearance of a garment is not fully determined by the viewpoint selection and illumination, as for rigid objects, but also by its current configuration (Fig. 1.1). Moreover, the space of possible configurations is infinite. The variability of appearance is even significantly higher than for the articulated objects consisting of several movable rigid parts. The garments are typically heavily self-occluded, especially when posed in a crumpled state (Fig. 1.1). It may be therefore impossible to observe some parts important for their understanding.

The robotic manipulation of garments is a challenging task as well. Due to folds and wrinkles, it is problematic to approach and grasp the garment reliably. Since the garment is deforming while being grasped, it is difficult to adjust the grasping force. Once the garment is held and manipulated, it keeps deforming. It is not only because of the manipulation, but also due to the gravity, stretching or friction. It is therefore very challenging to track individual parts of the garment under manipulation. Moreover, care must be taken to avoid tearing or ripping it.

Existing methods, including those described in the proposed work, employ a pragmatic approach to deal with the aforementioned challenges. The manipulation task is usually split into several simpler steps. Each step consists of perception, followed by reasoning and actual manipulation. This repeated perception-action loop is called an active perception. The existing perception algorithms are usually rather reactive, using only a very limited amount of information from the previous steps.

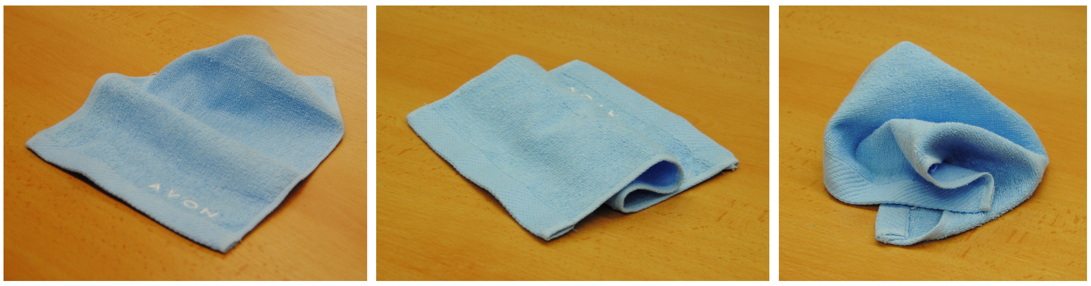


Figure 1.1. Various configurations of the soft towel caused by its deformation. Notice the heavy self-occlusions present in the folded and crumpled states.

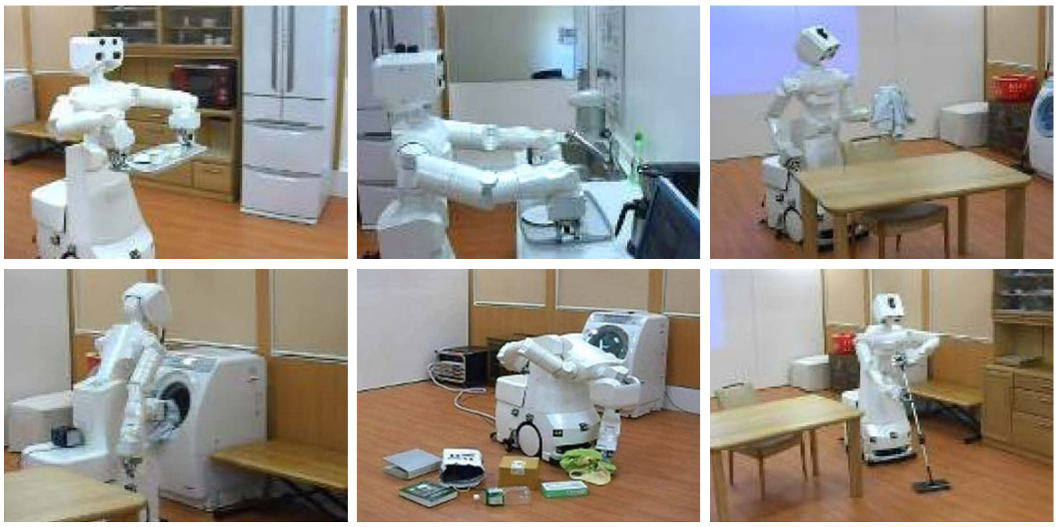


Figure 1.2. [Yamazaki et al., 2012] believe that the various daily housekeeping tasks could be performed by humanoid robots in future. © 2012 IEEE

No advanced model of the garment is used that would be tracked during the manipulation and updated as more information is gathered. We intended to build such a complex model originally that would combine rather certain information about already seen parts of the garment with hypotheses about the unseen. However, we learned that it would be very difficult to build and maintain it. It is also probably not necessary for accomplishing the standard manipulation tasks described in the thesis. The advanced qualitative modeling thus remains an open challenge.

1.2. Motivation

There is a serious problem with the aging society [Yamazaki et al., 2012]. It is caused by an increasing life expectancy and decreasing birthrate. [Yamazaki et al., 2012] claim that 20% of Japanese population was older than 65 years in 2005. This ratio is expected to double to 40% by 2055. Some of these senior people may suffer by reduced mobility. They may not be able to perform usual household activities such as cooking, washing dishes, laundering, ironing, clothes folding, vacuuming, sweeping etc. Some of them may be unable to get dressed or to have a bath. Quality of their lives can be significantly improved by personal assistance.

[Yamazaki et al., 2012] believe that the assistance could be provided by robots in the future (Fig. 1.2). They will probably use general manipulators resembling human hands to perform various tasks and operate common house facilities. They will be required to deal with garments while laundering, ironing or assisting with dressing up. They will act mostly autonomously, but they might also collaborate with humans on some tasks. These household robots will be also used by people who do not require personal assistance, but do not want to spent their free time by housekeeping.

There is a demand for robots manipulating highly deformable objects in industry as well. In contrast to general purpose household robots, the devices are constructed specifically for one particular task. The industrial manipulation tasks are performed repetitively and in a well-defined environment, which brings a significant simplification. On the other hand, the industrial robots are expected to operate with nearly a perfect reliability and usually very quickly.

An example of the industrial manipulation with garments are laundry companies, which supply hotels and hospitals with clean towels or bedding. Many of them use partially automated lines[1], however, there are still tasks that need to be performed by humans. They include inputting dirty garments to washing machines or setting already washed garments to mangling machines. Since working conditions are often unhealthy because of dust, heat and humidity, there is a demand for fully automated lines, as proposed by [Hata et al., 2008]. Another possible application are fully automated industrial sewing machines[2].

The aforementioned applications have attracted the attention of scientific community in robotics during the last decade. The related publications are described in Chap. 2. Robotic manipulation of garments has recently been a topic of several international research projects, including I-DRESS[3] and Clothes Perception and Manipulation (CloPeMa)[4]. The author of the proposed work was involved in the latter.

CloPeMa was an international project funded by European Commission in years 2012–2015. Four academic institutions and one industrial partner were involved: Centre for Research and Technology Hellas (coordinator), Czech Technical University in Prague, University of Glasgow, Universita Degli Studi Di Genova and Neovision s.r.o. The goal of the project was to advance the state-of-the-art in autonomous perception and manipulation of various kinds of fabrics and garments. A dual-arm robotic platform was developed by the project consortium that we use extensively for the experimental evaluation in this thesis.

The research presented in this thesis was started during the project CloPeMa. The author was responsible for the visual perception utilized in the folding task, which is described in Chap. 3. Later it was integrated into the unfolding and folding pipeline described in Chap. 4. The author's research continued after the successful finish of the project CloPeMa. Chap. 5 describes a method for detecting layers of a folded garment. Chap. 6 deals with the classification of hanging garments.

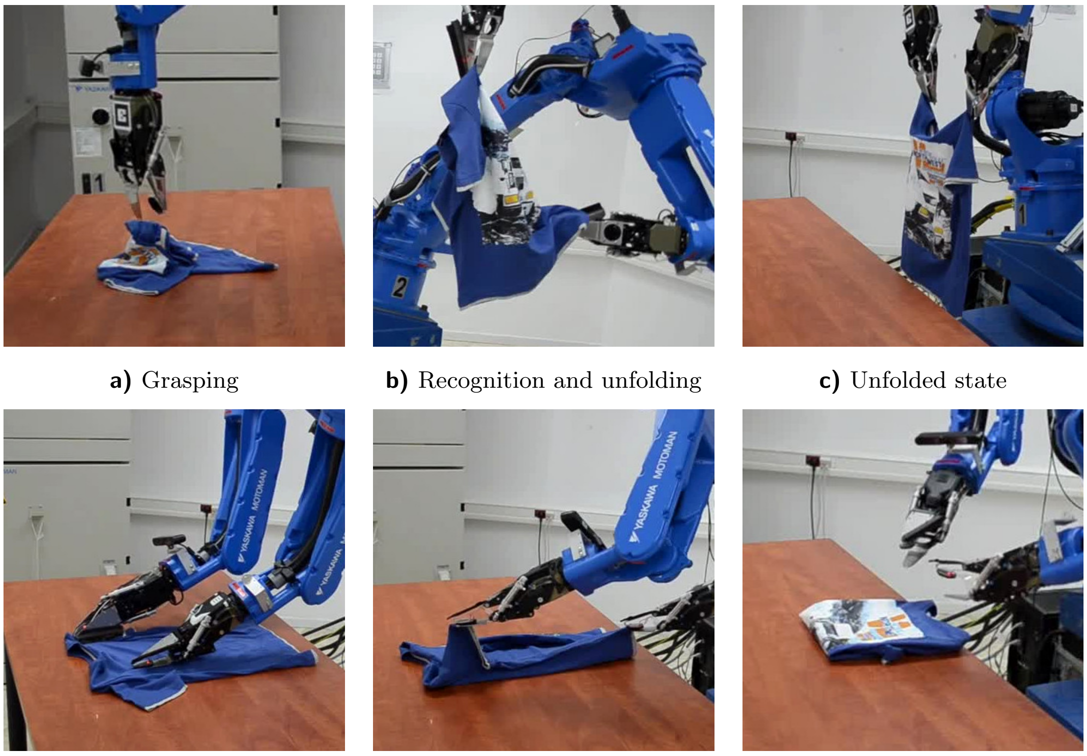
1.3. Task formulation

The vast majority of the existing scientific works, dealing with the robotic manipulation of garments, focus on one or more sub-tasks in processing a heap of unknown garments. There are basically two possible scenarios. The first scenario is sorting of the garments based on their color or material properties to several groups that need to be laundered separately. The second scenario is folding the heap of crumpled garments (Fig. 1.3). This is typically needed after laundering when the garments are taken out of the dryer.

Dual-arm robots are usually used for the manipulation of garments, resembling human actions. The arms are mounted grippers that are able to grasp a garment. Two arms make it possible to hold the garment at two different points at the time. The garment can be stretched by moving the arms away. Most importantly, it is possible to release and regrasp the garment with one of the grippers while holding it with the other. This maneuver is utilized mainly during the unfolding.

Both sorting and folding scenarios share similar sub-tasks. Single garment needs to be isolated from the pile at first to be processed (Fig. 1.3a). A suitable grasping point must be chosen so that only one garment will be held reliably. Since there is not enough

[1]Jensen-Group: http://www.jensen-group.com
[2]SoftWear Automation: http://www.softwearautomation.com
[3]I-DRESS http://www.i-dress-project.eu
[4]CloPeMa: http://www.clopema.eu



a) Grasping **b)** Recognition and unfolding **c)** Unfolded state

d) Flattening **e)** Folding **f)** Final folded state

Figure 1.3. The folding scenario consists of several sub-tasks. a) The crumpled garment is grasped and lifted up. b) Its category is recognized, pose estimated, the garment is unfolded and c) placed on a table. d) The remaining wrinkles are flattened. e) The garment pose is estimated and it is folded. f) The scenario ends with the fully folded garment.

information about the garments included in the pile at that moment, the pile is usually segmented based on texture. The grasping point is chosen based on specific geometrical properties, most often detected wrinkles or folds that are suitable for grasping.

The isolated garment is perceived with a camera, depth sensor or stereo rig to be classified prior to the further manipulation (Fig. 1.3b). The usual sub-task is classification of its category, e.g. pants, shorts, T-shirt or towel. Other classification tasks comprehend identification of a particular garment known in advance or recognition of various properties, including a color, texture and material. Additional feature domains may be used, including force-torque and tactile sensors. The garment can be manipulated prior to and during the classification to gain more information. This approach is known as an active perception.

The sorting scenario usually ends up with placing the sorted garments into separate buckets. The folding scenario continues with unfolding of the hanging garment, utilizing its known category. The garment can be unfolded either completely or only partially in this stage. In the former case, the goal is to hold the garment at two predefined locations, e.g. corners of a towel sharing its shorter side or shoulders of a T-shirt (Fig. 1.3c). In the latter, the goal is to disentangle the garment and bring it to a simpler configuration. The perception is needed to detect the desired grasping points, sometimes based on the complete pose estimation.

The fully or partially unfolded garment is placed on a table. The next sub-task is removal of wrinkles on the garment surface, usually achieved by pulling it sideways while holding it with the other arm (Fig. 1.3d). The related sub-task is detection and unfolding of the folds remaining from the previous step.

The last sub-task of the folding scenario is folding of the spread garment laying on a table (Fig. 1.3e). Its category and pose must be estimated if not known from the previous stages. The garment is folded iteratively, imitating the approach used by humans, e.g. a towel is folded over its middle repeatedly or sleeves of a T-shirt are folded independently before folding the whole T-shirt over (Fig. 1.3f).

There are also tasks not involved directly in any of the considered scenarios. They include robotic ironing or detection of a garment tossed in a room, which is useful for collecting dirty garments to be laundered. The related research topic is also the robotic dressing assistance, where the safety of the involved human and the potential human-robot collaboration bring additional challenges.

The proposed work deals mainly with three sub-tasks related to the folding scenario: folding of a spread garment (Chap. 3), unfolding of a partially folded garment (Chap. 5) and category classification of a hanging garment (Chap. 6).

1.4. Contribution

The thesis deals with the perception of garments applied in various robotic manipulation tasks. All the author's original work relates to the perception itself, not to the actual robotic manipulation. The main contributions can be summarized as follows:

- **Segmentation method for objects with known background.** We propose a fully automated method for segmenting an unknown object having a background with known statistical color properties [Stria et al., 2014a]. The background color is modeled with Gaussian mixture model (GMM) learned from data. The model is used for the initial segmentation, assuming that the background pixels are assigned a high likelihood, whereas object pixels are less likely in GMM. The obtained segmentation is used for initializing GrabCut algorithm instead of requiring a user input.
- **Method for pose estimation of a polygonal object.** We introduce an algorithm for fitting a model to an object which has approximately polygonal shape. Slight deformations are acceptable. The object contour is approximated by a polygon at first. The simplified contour is matched a polygonal model. The model vertices are defined manually, whilst their mutual positions are learned from data. The matching is formulated as a minimization of a specific cost function comprehending terms of two [Stria et al., 2014a] or three [Stria et al., 2014b] different types. The minimization is solved by dynamic programming.
- **Integration of the folding method into CloPeMa pipeline.** The method for estimating pose of a spread garment and its folding was integrated into CloPeMa pipeline for the folding scenario [Doumanoglou, Stria et al., 2016], enabling to bring a heap of unknown crumpled garmens to a folded state. To the best of our knowledge, it is the first fully working pipeline for the folding scenario that is able to deal with various types of garments.
- **Method for detection of individual stacked layers of folded garment.** The problem of detecting folded layers is formulated as pixels labeling, with the labels corresponding to individual layers [Stria et al., 2017]. The method combines image and depth information and encodes it into a specific energy function. It is assumed that the top layer is closer to the camera than the bottom one and that a boundary of the layers is visible in the image. The optimum labeling minimizing the energy is found using graph cuts.

- **Generalization of convolution for unstructured point clouds.** A standard discrete convolution is defined for vectors or grids with a regular structure, which determines a local neighborhood of each point. We generalize the convolution for sets of unordered 3D points [Stria and Hlaváč, 2018]. The local neighborhood of each point is formed by its spatially closest neighbors ordered by distance.
- **Neural network architecture for point clouds classification.** The aforementioned generalized convolution is used as a building block in the convolutional neural network (CNN) for classification of 3D point clouds [Stria and Hlaváč, 2018]. Our architecture utilizes techniques analogous to those known from CNNs for image classification. Size of the local neighborhood grows in the deeper layers, increasing sizes of the receptive fields. The point cloud is repeatedly subsampled by the network, decreasing the output size in the deeper layers. The network is trained on common 3D objects and transferred to the domain of hanging garments.

1.5. Outline

The presented thesis is divided into seven chapters. The original contributions of the author are described mainly in Chap. 3, 5 and 6. Their contents are partially independent because they deal with various sub-tasks related to the folding scenario, described in Chap. 4. However, the later chapters reuse some methods from the previous chapters. They are therefore ordered chronologically, as the described methods have been developed, instead of ordering them based on the folding scenario. We humbly suggest the reader to follow the intended order.

- Chap. 2 describes the state-of-the-art in machine perception and robotic manipulation of garments. It concerns all major sub-tasks related to the folding scenario, including those not covered by our own work. Chap. 2 also summarizes methods for applying CNNs on various types of 3D data, focusing mainly on 3D point clouds, which provides the context for Chap. 6.
- Chap. 3 deals with pose estimation of a garment with the application in its robotic folding. It describes the method for the garment segmentation, its contour extraction and simplification. The polygonal model of the garment contour is defined and two dynamic programming based algorithms are proposed for its matching to the perceived contour. The method is applied both for the initially spread garment and for checking its pose after individual folds. The performance is evaluated experimentally on the dataset containing garments of various categories.
- Chap. 4 describes integration of the folding method into the CloPeMa pipeline for the folding scenario. The other modules of the pipeline developed by the CloPeMa consortium are summarized, including methods used for grasping, classification, hanging pose estimation and flattening of garments. The experimental results are provided, including performance evaluation of the robotic folding.
- Chap. 5 focuses on automated folds detection and their robotic unfolding. The garment is segmented from its background using the method proposed in Chap. 3. Its surface is segmented into regions corresponding to the individual folded layers. Once the layers are detected, the possible folding axes are generated and the garment is unfolded virtually to select the correct axis. The folds are removed in a greedy manner by a cooperative manipulation of both robotic arms. The performance is evaluated on a dataset of folded garments and in real robotic experiments.
- Chap. 6 proposes an alternative strategy to the category classification of hanging garments used in the CloPeMa pipeline. Depth maps capturing the garment from

different viewpoints are fused into a 3D point cloud. A single global feature vector is extracted from the point cloud with a novel CNN architecture. The feature vector is classified with a support vector machine (SVM). The classification accuracy is evaluated on publicly available datasets of hanging garments.
- Chap. 7 concludes the thesis. The achieved results are summarized and possible future extensions are proposed.

2. Related work

The majority of existing works, which deal with machine perception and robotic manipulation of garments, focus on one or more sub-tasks from the folding or sorting scenario. The scenarios and sub-tasks are defined in Sec. 1.3. This chapter summarizes the related works. Contributions of others are organized according to the mentioned sub-tasks into Sec. 2.1–2.6. Sec. 2.7 reviews the available methods for classification and semantic segmentation of 3D data using convolutional neural networks (CNNs).

2.1. Grasping point estimation and grasping

The first sub-task is grasping of a garment tossed on a flat surface, usually a table. There can be only a single garment or whole heap of garments from which a single garment needs to be isolated. The garment category is usually neither known in advance, nor is attempted to be recognized prior to grasping because of heavy deformations. The grasping is therefore uninformed, usually relying on geometric features extracted from the visible garment surface.

Robotic manipulation of garments was pioneered by [Hamajima and Kakikura, 1998], who describe, on a rather conceptual level, the whole pipeline for clothes folding scenario, as defined in Sec. 1.3. The later work [Hamajima and Kakikura, 2000] focuses on grasping a single garment from a heap perceived from above with a camera. An image region covered with the heap is provided manually. The heap is segmented recursively into sufficiently large regions of uniform color or texture. Centers of the regions form candidate locations for grasping.

[Willimon et al., 2011a] also segment the heap into regions of uniform texture using a graph-based method [Felzenszwalb and Huttenlocher, 2004]. Surface of the heap is reconstructed from stereo images and used to compute the average height of each region above the table. The geometric center of the highest region is grasped by the robot (Fig. 2.1). The fulfillment of grasping is checked by subtracting two images of the arm, before and after the operation. If not successful, it is repeated closer to the table, which is an example of simple but robust approach.

[Ramisa et al., 2012] assume that there is only a single garment on the work table. They combine a color image and depth map from a range sensor to detect highly wrinkled parts suitable for grasping. The wrinkledness is measured for each individual input patch based on the scale-invariant feature transform (SIFT) [Lowe, 2004] and depth descriptors. The patches are classified by the logistic regression and support vector machine (SVM) [Cortes and Vapnik, 1995] learned from graspable and non-graspable examples. [Ramisa et al., 2013, Ramisa et al., 2016] introduce the Fast Integral Normal 3D (FINDDD) descriptor developed specifically for textiles, improving the performance over the generic descriptors. FINDDD is based on histograms of quantized normals estimated from a depth map.

[Yamazaki, 2014] detects the potential hemlines and corners of a tossed towel. They search for the discontinuities in the input depth map and examine local geometry of such discontinuous parts. The goal is to grasp the towel with both arms near neighboring corners sharing an edge and unfold it directly.

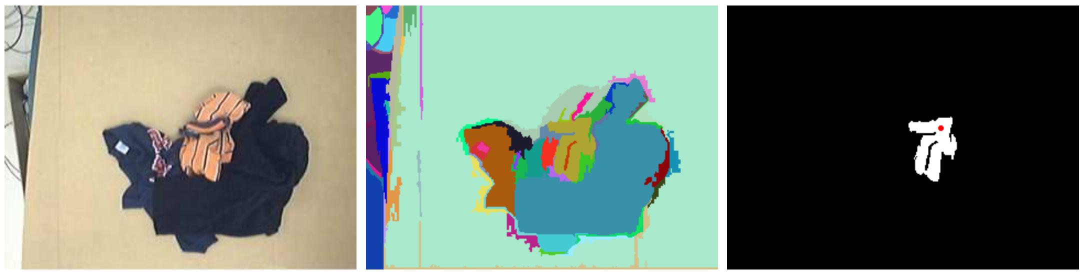


Figure 2.1. [Willimon et al., 2011a] segment the heap of garments based on texture information, compute the average height of each region above the table and grasp the geometric center of the highest region (red dot). © 2011 IEEE

[Cusumano-Towner et al., 2011] find the outer boundary of a crumpled garment by segmenting it from a background and grasp it from a side. [Foresti and Pellegrino, 2004] address the detection of grasping points on furs. After the fur regions are segmented, their skeleton is extracted and used to locate narrow branches, which are good candidates for stable grasping. [Hata et al., 2008] and [Bersch et al., 2011] choose the highest point of the heap as the grasping point, while [Maitin-Shepard et al., 2010] choose the central point. [Kita et al., 2011] grasp the vertically oriented parts of a garment that are located high enough above a table. [Alenyà et al., 2012] benchmark various grasping strategies. They also describe common grasping concepts and identify possible issues.

2.2. Classification and pose estimation of hanging garments

Once the garment has been grasped and lifted up, its category must be classified and pose estimated prior to the manipulation. The possible categories comprehend e.g. pants, shorts, T-shirts or towels, which need to be unfolded and folded in different ways. The garment is usually held at a single point and hanging down because of gravity, which reduces the space of its possible configurations. The second arm can be used for its active manipulation, e.g. to regrasp it at different point or spread it. The existing methods can be split into the model-based ones, which fit a garment model to perceived data, and learning-based ones, which extract features from perceived data and learn a classifier on top of them.

The aforementioned model-based approach is employed by [Kita and Kita, 2002]. They use a planar mass-spring model [Lander, 1999] of a pullover, whose approximate size is known in advance. The virtual model is spread, grasped for each of its 20 vertices and lifted up in a physics simulator, obtaining 20 different hanging poses. Silhouette of the observed pullover is then overlaid over all simulated poses to find the best matching one. [Kita et al., 2004a, Kita et al., 2004b] use a virtual model of pants in addition to the pullover. This enables category recognition by finding the best match over both models and all simulated poses. The matching is improved by observing the garment from two different angles and checking the consistency of individual pose estimations.

[Kita et al., 2009] reconstruct a 3D point cloud of the observed garment using a trinocular camera system. The original mass-spring models are replaced with 3D models simulated in the professional animation software Maya[1]. The shape of the virtual hanging model is adjusted with two types of forces before it is matched to the point cloud. The internal forces preserve the model shape by keeping the distance between

[1]Autodesk Maya: http://www.autodesk.com/products/maya

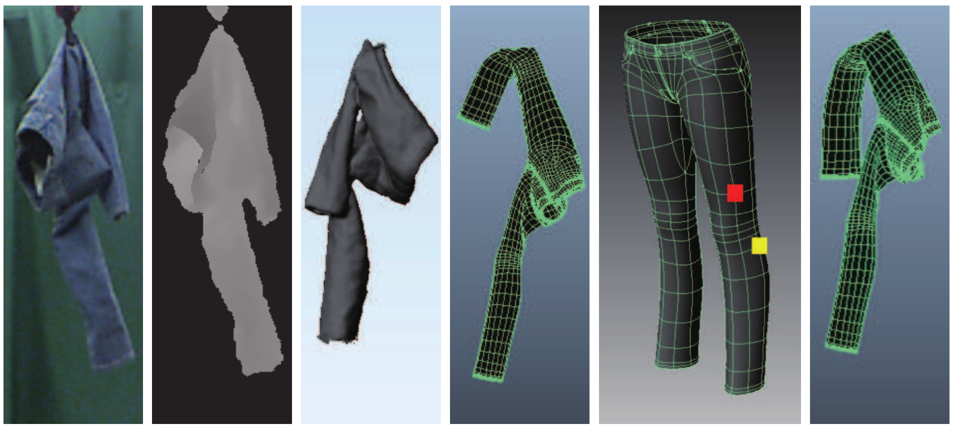


Figure 2.2. [Li et al., 2014b] reconstruct 3D model of a perceived garment by fusing depth maps from multiple viewpoints. It is matched to virtual models, which are simulated hanging from different grasping points. © 2014 IEEE

vertices, which simulates the elasticity and flexural rigidity. The external forces comprehend the gravitation force and attraction to the closest point from the cloud. Once the model is adjusted, its similarity with the point cloud is computed as the overlap ratio of their 2D projections. The perception was further improved by an active manipulation of the hanging garment [Kita et al., 2010], including its rotation and spreading.

[Li et al., 2014a] combine the model and learning-based approach. They build SVM classifiers of category and pose, using bag-of-words (BoW) [Lazebnik et al., 2006] model on top of quantized SIFT features extracted from depth maps. The classifiers are trained on an artificial dataset of hanging garments simulated in Maya. An unknown real garment is rotated around its vertical axis and perceived from 150–200 viewpoints, each of them classified independently. The final classification is obtained by a majority voting. [Li et al., 2014b] speed up the method by merging the depth maps from individual viewpoints to a volumetric representation (Fig. 2.2). The reconstructed unknown garment is classified to the closest simulated virtual model with respect to the weighted Hamming distance. [Li et al., 2015a] further register the real garment to the best matching model. The registration consists of a rigid transformation followed by a non-rigid deformation of the virtual model. Estimation of the non-rigid deformation is formulated as an energy minimization problem.

[Willimon et al., 2011a] employ the active perception approach to classify a garment category. The unknown garment is repeatedly grasped and lifted up, two images are taken from different viewpoints and the garment is dropped again. The procedure is repeated 10 times, obtaining 20 images of the hanging garment held at random points. Each image is matched to a dataset of annotated templates to find its nearest neighbor. The similarity measure combines global features extracted from silhouettes with information about Canny detected edges [Canny, 1986].

[Willimon et al., 2012] register the deformable mesh model of a garment to the sequence of RGBD images capturing a garment being held and manipulated by a human. The registration is based on minimizing the weighted sum of several terms, using a semi-implicit iterative algorithm [Kass et al., 1988]. The sum comprehends the smoothness term capturing the internal energy of the mesh, or the terms responsible for keeping the correspondences between the mesh and RGBD data. [Willimon et al., 2013a] improve the method by building the initial 3D mesh automatically from the first RGBD image in the sequence. They also modify the energy function.

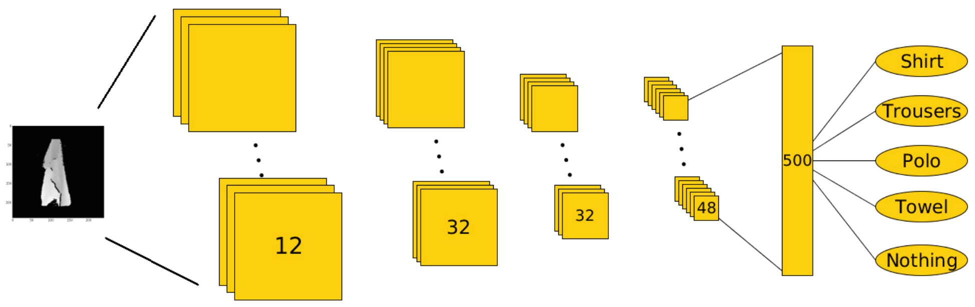


Figure 2.3. [Gabas et al., 2016] classify garment categories with CNN, whose input is a depth map. The classifications are aggregated over multiple viewpoints. © 2016 Springer

[Bersch et al., 2011] deal with the pose estimation of a hanging T-shirt whose surface is covered with fiducial markers. The surface is reconstructed from multiple images and represented with a triangulated mesh, which enables to measure geodesic distances of the markers. It is assumed that the garment pose is fully determined by the currently grasped mesh vertex. The grasped vertex is estimated with naive Bayes classifier, using the geodesic and measured Euclidean distances of the markers as features.

[Doumanoglou et al., 2014a] train a Random Forest (RF) [Breiman, 2001] on simple features extracted from a depth map, including depth difference of two points or curvature estimated in a selected point. The RFs serve both for category classification of a hanging garment and for predicting the next grasping point on its surface during unfolding. The classifier is used in the partially observable Markov decision process (POMDP) [Sondik, 1978] framework that decides whether the hanging garment should be rotated and perceived from another viewpoint for more informed decision. [Doumanoglou et al., 2014b] omit the POMDP framework and incorporate the next best view selection directly to the decision forest, called Active Random Forest (ARF).

Instead of using handcrafted features, there have been attempts recently to learn the whole classification pipeline in the form of a CNN, whose input is a depth map. [Mariolis et al., 2015] use networks comprising 3 convolutional and 3 fully connected layers, hyperbolic tangent activations and L_2 norm pooling. There is a single network for the category classification and additional category specific networks, which classify the currently grasped point from the predefined set of points on the garment surface. The classifications are aggregated over multiple viewpoints by majority voting. The networks are trained on 643k synthetic depth maps simulated in Blender[2] and 4.8k real depth images. [Kampouris et al., 2016] combine the proposed CNN with the aforementioned ARF [Doumanoglou et al., 2014a] to improve the category classification accuracy.

[Gabas et al., 2016] classify garment categories with a network based on AlexNet [Krizhevsky et al., 2012], comprising 4 convolutional and 2 fully connected layers, max-pooling, rectified linear unit (ReLU) activations and dropout (Fig. 2.3). The network is trained on 4.3k real depth maps, formed by sequences of 12 maps depicting the same hanging garment from different viewpoints. The classification accuracy is 83% for single view and 92% while voting over all 12 viewpoints. [Corona et al., 2018] use a hierarchy of these networks to detect the grasping points for unfolding. The detection is posed as the regression of a point coordinates and its visibility, optimizing a loss function independent on the grasps ordering. Training data were extended by 60k artificial samples, which improved the classification accuracy to 97%.

[2]Blender: http://www.blender.org

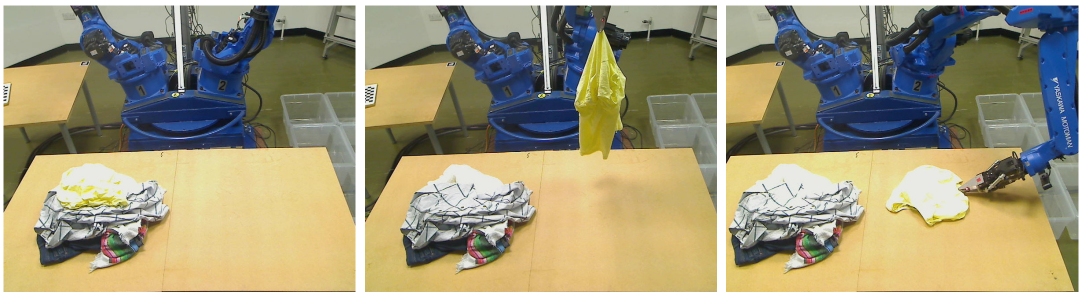


Figure 2.4. [Sun et al., 2016] manipulate the garment actively in order to classify its category from multiple configurations. © 2016 IEEE

2.3. Classification and pose estimation of tossed garments

The category classification of a randomly tossed garment, found in a crumpled state, is even a more challenging problem than classification of a hanging garment. The majority of the existing methods are learning-based, usually using SVM classifier on top of the features extracted from images and depths. Matching a model to a crumpled garment would be very difficult. The methods for pose estimation are rather rare, compared to the case of hanging garments described in Sec. 2.2.

[Willimon et al., 2013b] use a multi-level classification architecture. The low-level component combines the local SIFT image features, FPFH [Rusu et al., 2009] depth features and multiple global histogram based features. The feature vectors are classified with the binary SVM classifiers to decide about 27 mid-level characteristics, including presence of a collar or round neck, stripedness, material etc. These characteristics, together with learned masks weighting their importance for various categories of garments, are then used for the high-level classification.

[Sun et al., 2016] employ the active perception approach (Fig. 2.4). The perception utilizes features extracted from a depth map produced by a stereo head, which provides a better resolution than standard range sensors. The garment is grasped, shaken or flipped, and tossed again after each perception stage, so that its configuration changes before the next perception. The classification confidences are modeled, tracked and updated with a Gaussian process. On contrary, [Sun et al., 2017] classify the garment from a single depth map. They combine many local and global features, including shape index [Koenderink and Van Doorn, 1992], local binary patterns [Ojala et al., 1996] and features designed specifically for this task. The original Gaussian process based classifier is compared to RF and SVM. The latter provides a superior performance if the radial basis function (RBF) [Friedman et al., 2001] kernel is used.

[Ramisa et al., 2016] use their own FINDDD descriptor. It is based on quantized normals estimated from a depth map, as described in Sec. 2.1. They build a BoW model on top of FINDDD and classify it with an SVM model using the linear and RBF kernels. The same pipeline is used for the identification of wrinkles and collars.

[Yamazaki and Inaba, 2013] deal with the identification of a particular crumpled garment from a set of garments varying in category and material. A grayscale image of the garment is filtered with multi-scale and multi-orientation Gabor filters (Fig. 2.5). It is assumed that the low frequency responses correspond to wrinkles, whereas the high frequencies correspond to contours and overlaps. Their geometric properties are measured, discretized and accumulated into histograms. The concatenated histogram forms a global descriptor that is classified with SVM using the RBF kernel. [Yamazaki, 2017] extend the method by segmenting the garment into superpixels at first, whose features

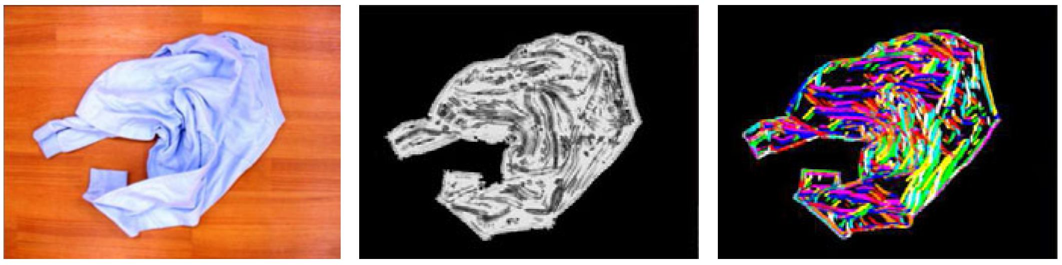


Figure 2.5. [Yamazaki and Inaba, 2013] filter the input image with multi-scale and multi-orientation Gabor filters to detect low-frequency wrinkles and high-frequency contours and overlaps, as well as their orientations. © 2013 IEEE

are described locally. Each superpixel is classified separately. It is therefore possible to segment a whole heap containing multiple garments into individual items.

[Wang et al., 2011] deal with configuration recognition and pairing of socks. The local texture of a sock is described with the local binary patterns (LBP) [Ojala et al., 1996] and MR8 filter responses [Varma and Zisserman, 2005]. The local shape is described with the histogram of oriented gradients (HOG) [Dalal and Triggs, 2005]. The local descriptors are classified with SVM to detect specific parts of the sock, e.g. a heel or toe. The detected parts are combined in a global shape model [Miller et al., 2011], which determines the overall configuration. The local texture and shape descriptors are also utilized to match pairs of socks together. Two pairing strategies are compared: greedy matching and globally optimal pairwise matching.

2.4. Unfolding of hanging garments

The unfolding of a garment, which is held hanging under the gravity, is the next step in the pipeline. The majority of the methods assume that the garment category and pose are already known from the previous step, as described in Sec. 2.2, which is needed for selection of the manipulation strategy. The goal of the unfolding is usually to hold the spread and hanging garment at predefined locations, e.g. shoulders for a T-shirt or neighboring corners for a towel. However, there are also methods approaching the unfolding without knowing the garment category and in pure geometric manner, based on the detected folds and hemlines.

[Hamajima and Kakikura, 1998, Hamajima and Kakikura, 2000] propose a method for regrasping a lifted garment by its hemlines. Detection of the hemlines is based on the observed shadows and outer shape of the hanging garment. The hemlines should arguably form a convex outline. The desired goal is to hold the garment at two different hemlines or at two endpoints of the same hemline. Then the garment can be at least partially unfolded by stretching it out. [Kaneko and Kakikura, 2001] match the outline of a stretched garment to three basic template shapes. Each template is accompanied with a decision tree for recognizing the category and pose of the garment.

[Maitin-Shepard et al., 2010] focus on unfolding of a hanging towel (Fig. 2.6). The hemlines are assumed to appear as the outer contours that have significant depth discontinuities in depth maps taken from various viewpoints. The hemlines are fitted variously angled corners with the random sample consesus (RANSAC) [Fischler and Bolles, 1981]. The corners detected in images are verified by fitting a 3D plane to the reconstructed point cloud. The towel is regrasped several times until two neighboring corners are

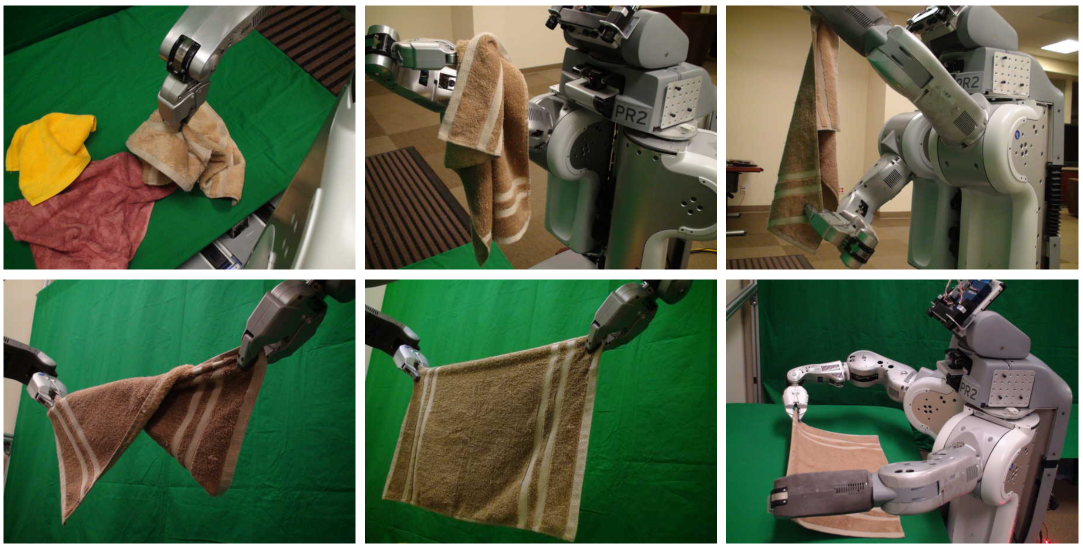


Figure 2.6. [Maitin-Shepard et al., 2010] unfold a towel in a series of manipulations, including untwisting, utilizing information about the detected hemlines and corners. © 2010 IEEE

held. It is then untwisted by stretching it out and rotating the grippers, pulled over the edge of a table and folded.

[Cusumano-Towner et al., 2011] improve the method and extend it for various types of garments. The garment is manipulated to an arbitrary recognizable configuration at first by regrasping its lowest point repeatedly. The sequence of manipulations is modeled with the hidden Markov model (HMM). The garment is modeled as a 3D triangulated mesh. The currently grasped mesh vertices and the garment category form the hidden states of HMM. Once the garment category and partially unfolded pose are known, the garment is laid on a table and unfolded in another series of manipulations.

[Triantafyllou et al., 2016] unfold a hanging garment partially by grasping it at two different outline points, without classifying its category. The garment is then laid down on a table and fully unfolded [Mariolis and Malassiotis, 2015], as described in Sec. 2.5. First, discontinuities in the input depth map are detected as Canny edges [Canny, 1986]. Junctions of the edges form candidates for folds appearing on the garment outline. They are pruned with several heuristic rules. The most convenient point on the outline near the fold is then grasped, optimizing a position and rotation of the gripper.

[Gabas and Kita, 2017] focus on localizing hemlines of a hanging towel, which are suitable grasping locations for its unfolding. The input depth map is processed with the Canny detector [Canny, 1986] at first. It provides the desired edges aligned with the hemlines, but also edges due to the folds, wrinkles and noise. The hemline edges are distinguished with a CNN classifier, which is fed with square patches centered at the detected edges. The most horizontal hemline is grasped and pulled sideways to open the towel, bringing it to a more unfolded state.

2.5. Unfolding and flattening of laying garments

Several methods deal with unfolding of garments placed roughly flat on a table surface. It is assumed that the garment is placed in a configuration that could be reached by folding a fully unfolded garment once or more times over arbitrarily located and oriented folding axes. It is further assumed that the folding axes form straight line segments

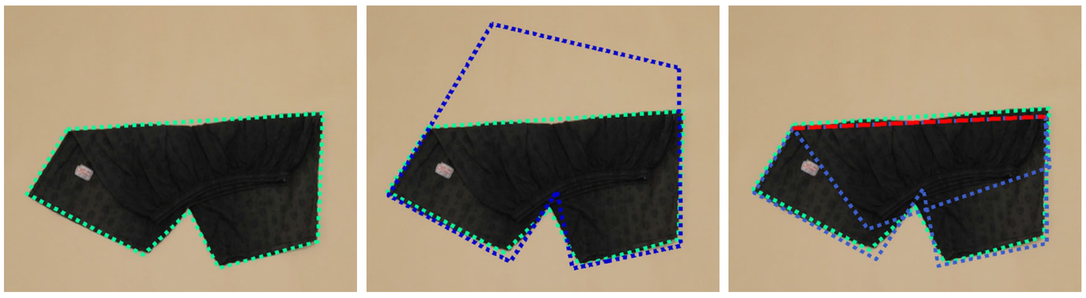


Figure 2.7. [Mariolis and Malassiotis, 2015] extract the polygonal contour of a folded garment (green) and match it partially to the template of an unfolded garment (blue). The template is folded virtually over the estimated folding axis (red). © 2015 Springer

on the outer boundary of the garment. In practice, such a folded configuration can be reached either by robotic folding, as described in Sec. 2.6, or by unfolding the hanging garment only partially at first, e.g. using the approach by [Triantafyllou et al., 2016] described in Sec. 2.4, and then placing it on a table.

[Mariolis and Malassiotis, 2013] match templates of unfolded garments to images of folded garments (Fig. 2.7). The garment contour is extracted and approximated with a polygon, sides of which form candidate folding axes. Remaining part of the contour is matched partially to all unfolded templates, providing hypothesis on the location of the folding axis for each template. The template is then folded virtually over the estimated axis and matched to the observed contour using the inner distance shape contexts [Ling and Jacobs, 2007]. The best matching template determines both the configuration and category. In addition, [Mariolis and Malassiotis, 2015] also unfold the contour virtually and register it to the unfolded template, while allowing some deformations of the template.

[Estevez et al., 2016] propose a garment agnostic approach to unfolding from an RGBD input. The background of a known color is segmented with a simple thresholding. Part of the depth map corresponding to the garment is segmented with the watershed algorithm [Šonka et al., 2014] into several regions, which correspond to individual folded layers. The highest region is selected for unfolding. The unfolding direction is determined by examining height changes over various paths from that region to the outer contour. [Estevez et al., 2017] improve the method by integrating depth maps from many viewpoints with Kinect Fusion [Newcombe et al., 2011]. The background segmentation is based on fitting a plane to the underlying table.

Once the garment is fully unfolded, its surface is usually still wrinkled, which makes the following folding process, described in Sec. 2.6, more difficult. The flattening deals with wrinkles detection, estimation of their sizes and orientations. The garment is then flattened by being pulled in the appropriate direction.

[Willimon et al., 2011b] proposed a method that combines flattening in the first phase with unfolding in the second phase. In the first phase, the robotic arm moves around the cloth, pulling each corner away from the garment center multiple times every 45 degrees. Dozens of pulls are usually performed. The second phase utilizes a stereo vision to segment the garment surface to the regions of continuous depths. The corners of the highest region are pulled either towards or away from the garment center, depending on its particular configuration.

[Sun et al., 2013] detect wrinkles in a depth map obtained from a virtual towel simulated with a mass-spring model [Lander, 1999]. The wrinkledness measure is computed

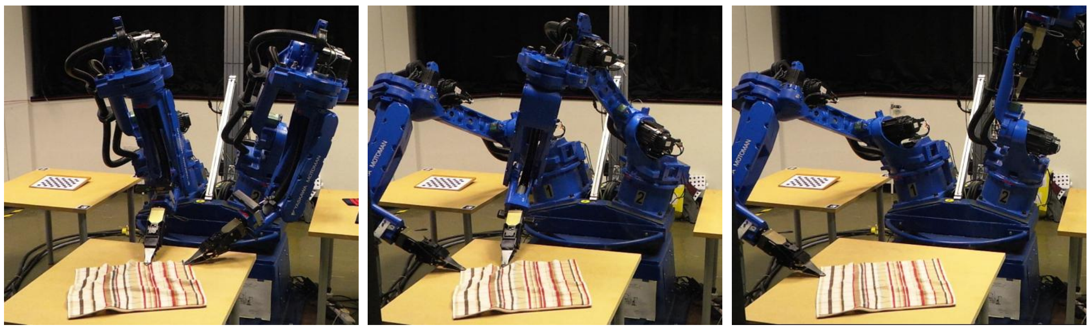


Figure 2.8. [Sun et al., 2015] detect position and orientation of wrinkles on the garment surface. The wrinkles are removed by pulling the garment in the horizontal direction. © 2015 IEEE

for each surface point at first, which is the mean absolute deviation of depths in its neighborhood. The highly wrinkled points are selected with thresholding, clustered with k-means [Friedman et al., 2001] algorithm and aggregated into larger clusters, which are believed to correspond to wrinkles. Position and orientation of the most salient wrinkle is used to plan a flattening move, which reposes in pulling the a corner or edge of the towel.

[Sun et al., 2015] modify the method for real garments of multiple categories, using precise depth maps reconstructed from stereo. The garment surface is piecewise approximated with B-splines and segmented to several classes based on curvature. The classes are e.g. a cup, saddle or ridge. The segmentation determines the geometry and topology of wrinkles. The flattening utilizes two robotic arms pulling the garment simultaneously, which reduces the number of required iterations (Fig. 2.8).

2.6. Garments folding

The last stage of the folding scenario is the actual folding of the garment. The existing works deal mainly with two sub-tasks. The first is pose estimation of the spread garment, which is usually implemented by model fitting. The approximate pose might be known from the previous manipulation, however, it needs to be refined for a more precise folding and checked after performing individual folds. The second sub-task is planning of the folding trajectory that would minimize the garment slipping on a table.

[Berg et al., 2011] introduce a theoretical approach to folding any spread garment. It is based purely on its shape, which is assumed to be polygonal. It is further assumed that the garment is infinitely flexible and that there is an infinite friction between the garment and a table. The grasping points on the contour are defined so that the hanging part of the garment is immobilized by gravity during folding. The grippers holding the garment follow triangular trajectories called g-folds.

[Petrík et al., 2015] revise the triangular trajectory and compare it to a circular trajectory. They show that for idealized rigid materials, flexible in the folding axis only, the circular trajectory prevents the garment from slipping, as it minimizes the horizontal pulling force. For real materials, the optimal trajectory is bound partially by the triangular and circular ones.

[Petrík et al., 2016] generate different folding trajectories for materials of various stiffnesses, again minimizing the horizontal pulling force to prevent slipping. They consider only narrow soft strips that can be approximated and modeled with 1D Euler-Bernoulli

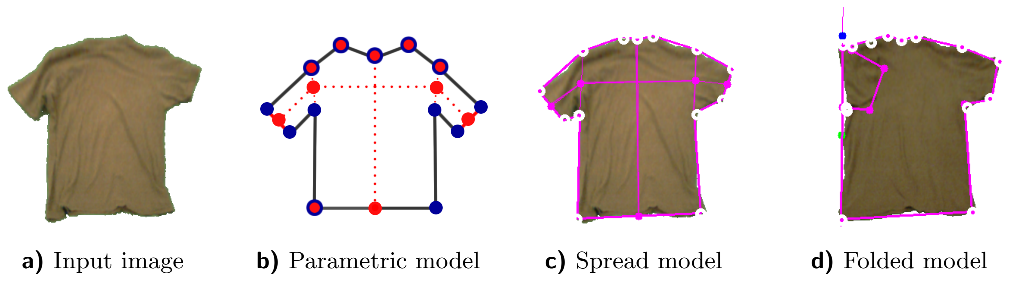


Figure 2.9. [Miller et al., 2011] estimate a) the garment pose by b) matching it the parametric polygonal model, which is formed by skeleton (red dots) and derived landmarks (blue dots). The method is used for c) spread and d) folded garments. © 2011 IEEE

beams [Timoshenko, 1983]. The final trajectory is obtained by the means of physics simulation. It resembles the shape of the letter R and consists of four phases determined by boundary conditions. [Petrík et al., 2017] extend the method for rectangular pieces of soft materials that can be modeled with Kirchoff-Love shell theory for 2D elastic shapes [Simo and Fox, 1989]. A single weight-to-stiffness parameter must be known to generate the trajectory, which is measured manually. [Petrík et al., 2018] estimate the parameter automatically by measuring the cloth shape with a laser range finder.

[Miller et al., 2011, Miller et al., 2012] focus on estimating the category and pose of a spread garment from a single image (Fig. 2.9). The background of a known color is subtracted. The garment contour is matched to a parametric polygonal model, vertices of which correspond to landmark points, e.g. corners or armpits. The matching procedure minimizes distances between the model outline and the contour points, using a coordinate-wise descent over the model parameters. The optimization comprises three phases, each one minimizing over a larger subset of parameters. The garment category is classified by selecting the best matching model over multiple categories. The garment is folded using the previously described g-folds [Berg et al., 2011]. Its configuration is checked after each fold by fitting a folded model derived from the original one.

Similarly, [Li et al., 2015b] estimate the garment pose by matching its contour to a deformable polygonal template. The matching procedure minimizes the distances of the contour points to the template edges, as well as stretching of the model edges and bending of its angles. The folding trajectory is again optimized with respect to the empirically learned material properties and friction, preventing the garment from slipping. The space of possible trajectories is formed by Bézier curves [Farin, 2014].

2.7. Convolutional networks for 3D data

Chap. 6 of the thesis deals with the category classification of hanging garments. Sec. 2.2 summarized the existing methods. Our approach is based on applying a convolutional neural network (CNN) to a 3D point cloud representing the garment in order to extract a single global feature vector. The proposed network architecture is novel and applicable in a wider setting of classifying unstructured 3D point clouds. Therefore, this section summarizes the existing CNN architectures for 3D objects processing. There are basically three possible representations of 3D objects and three corresponding families of architectures.

First, a standard CNN architecture for an image classification, which is based on 2D convolutions, can be applied on a depth map containing the object [Gupta et al., 2014]. This approach is used by all the existing methods dealing with garments classification

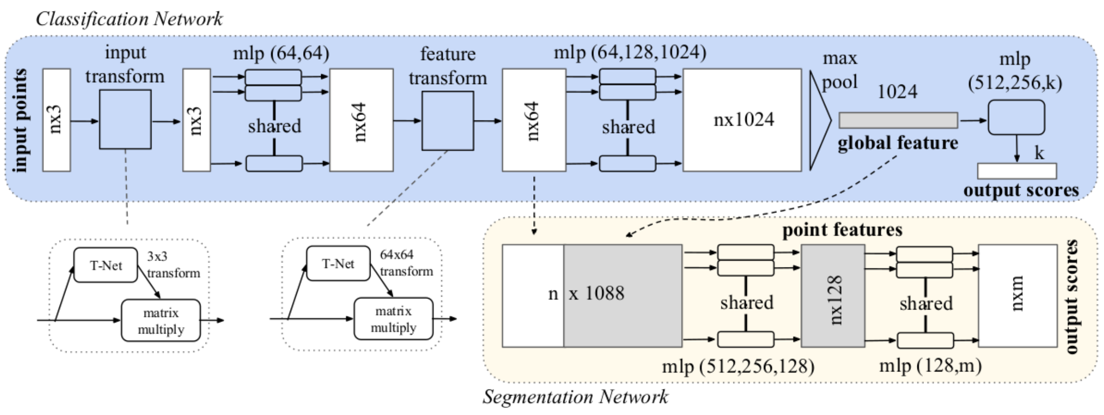


Figure 2.10. [Qi et al., 2017a] propose PointNet architecture for the classification and semantic segmentation of 3D points clouds. The convolutions are applied separately and gradually to each of n input points, transforming its dimension from 3 to 1024. The single 1024-D global feature vector is obtained by max-pooling over all local features. © 2017 IEEE

[Mariolis et al., 2015, Kampouris et al., 2016, Gabas et al., 2016, Corona et al., 2018], as described in Sec. 2.2. Some networks are even pretrained on annotated images, which are better available than depth maps, and fine tuned on depth data later. The classification accuracy can be improved by voting over multiple views of the same object. The network can eventually work with multiple views inherently [Su et al., 2015], building its own internal 3D representation of the object.

Second, there are networks working with the volumetric representation of an object [Maturana and Scherer, 2015, Wu et al., 2015]. Their architecture is very similar to the standard image networks. The relation of the neighboring pixels is analogous to the neighboring cells in the volumetric grid. The discrete 2D convolutions are replaced with 3D convolutions. Padding, striding or pooling are extended to 3D naturally. The volumetric representation of an object is very sparse and therefore rather ineffective, as the cells contain only binary occupancy values. It also depends heavily on the object pose and discretization of the grid.

Third, it is possible to apply the convolutions on 3D points clouds directly. This representation of an object is very compact, yet powerful, as individual parts of the object can be sampled with different densities. The main challenge reposes in generalizing the convolution operation over the unstructured set of 3D points. The related challenge is in making the network invariant to any particular ordering of the points.

The representation based on 3D point clouds was pioneered by [Qi et al., 2017a] in their PointNet architecture (Fig. 2.10). They apply convolution kernels separately and independently on each individual point, which makes the output invariant to any ordering of the points in the cloud. The convolutions therefore serve as simple feature transformers, whose weights are learned. The input point cloud is transformed to a canonical 6D pose at first by applying an affine transformation, whose parameters are determined by a small spatial transformer sub-network [Jaderberg et al., 2015]. The feature transforming convolutions are stacked into multiple layers. The number of convolution kernels grows from 64 in the first layer to 1024 in the last one, transforming each input 3D point to 1024-D feature vector. The local feature vectors are max-pooled into a single global 1024-D feature vector. This keeps the architecture invariant to any ordering of the points in the cloud, as the pooling operation is symmetric in its arguments. The global feature vector is classified with a stack of fully connected layers.

The improved PointNet++ architecture [Qi et al., 2017b] applies the main block from the original PointNet iteratively and in a hierarchical manner. The convolutions are again applied on each point separately to transform its 3D coordinates to a higher dimensional feature space. The max-pooling is only applied to the subsets of spatially close points. Each subset is therefore represented by a single feature vector, together with one selected representative member point. In the next iteration, the convolutions are applied on this partitioned representation of the original point cloud.

The closest to our own KnnNet architecture is PointCNN [Li et al., 2018], which was developed independently approximately at the same time. The so called X-convolution operation is introduced. It is defined for groups of spatially close points. Each point is transformed independently to a higher dimensional feature space at first, similarly to [Qi et al., 2017a]. The feature vectors of all neighboring points are then transformed jointly to a latent feature space. Parameters of the transformations are learned from data. It is assumed that the latent representation of the neighboring points is invariant to their ordering. A convolution is applied on the latent feature vectors.

Similarly to our KnnNet, [Wang et al., 2018] build the graph connecting each point with its k-nearest neighbors. The convolution operation is defined over individual edges of the graph. The extracted edge features from all edges incident to a particular point are aggregated with a function symmetric in its arguments. The related research field are the CNN architectures working with general graphs. [Veličković et al., 2017] employ the self-attention mechanism to select only the relevant neighbors, while aggregating the features over a group of graph vertices.

3. Folding spread garments

We propose a novel method to recognize the configuration of a spread out garment, followed by its robotic folding. The perception comprises several steps. It starts with a single RGB image taken from above. The garment is segmented from the underlying surface, whose color properties are known and learned in advance. The garment contour is extracted and approximated by a polygon.

The simplified polygonal contour is then fitted a polygonal model (Fig. 3.1), which is specific for the particular category of garments. There are five categories used in the experimental evaluation: towels, shorts, pants, short-sleeved T-shirts and long-sleeved shirts. The vertices forming the polygonal model are defined manually. They correspond to the key landmark points on the garment contour, e.g. corners, armpits or crotch. The mutual positions of the vertices are learned automatically from data.

The fitting procedure is based on the minimization of a specified cost function, which expresses dissimilarities between observed and expected data. Two different cost functions [Stria et al., 2014a] and [Stria et al., 2014b] are presented and compared, comprehending different sets of cost terms. The overall costs are optimized with the dynamic programming approach. The fitted model provides reliable estimation of the garment landmark points, which are used to plan the folding manipulation (Fig. 3.1). The configuration of the garment is checked after each fold by fitting a folded polygonal model. It is derived automatically from the original model based on the planned fold.

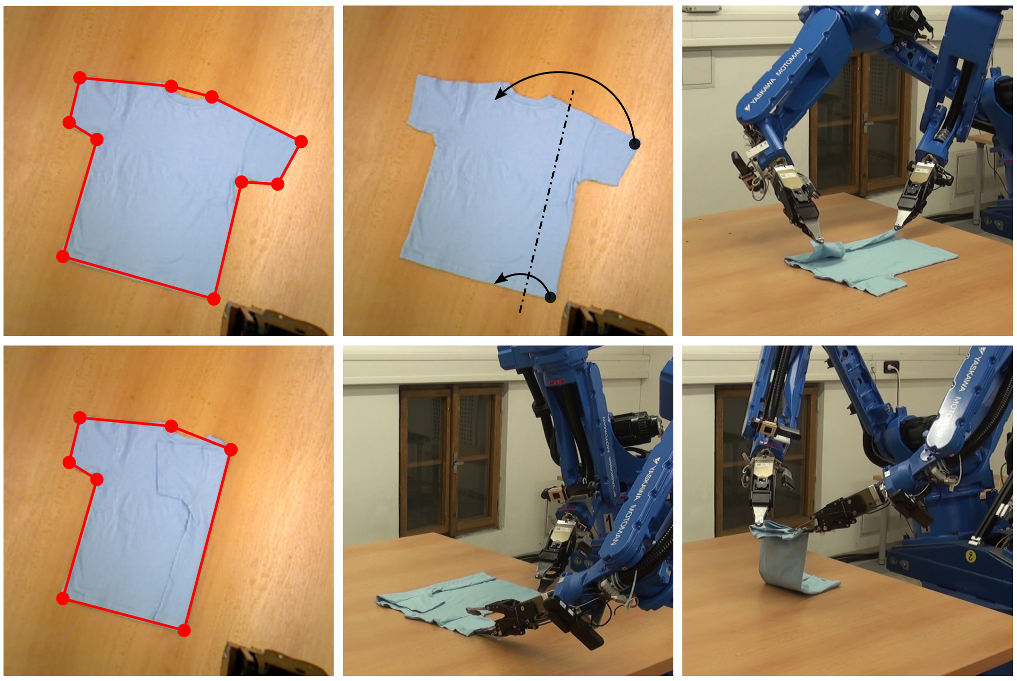


Figure 3.1. Overview of the robotic folding. The contour of a spread garment is fitted a polygonal model to estimate its pose. The vertices of the matched model determine the folding axis. The garment pose is checked after each fold.

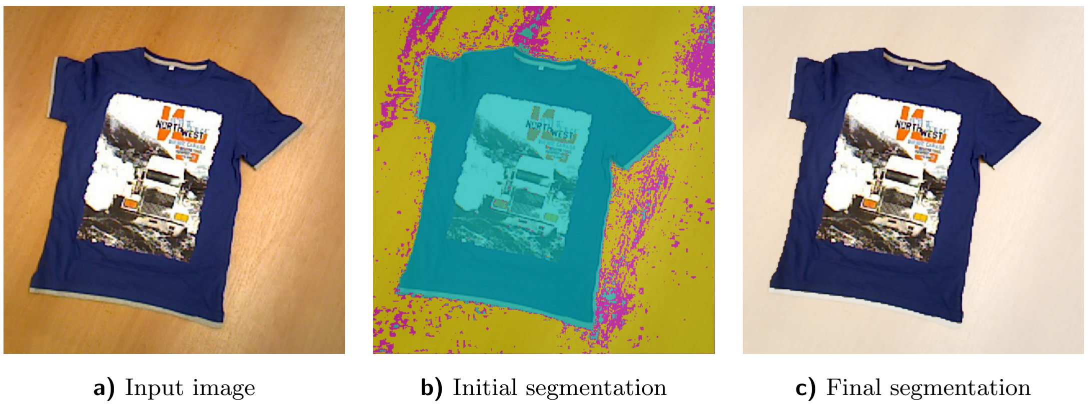


Figure 3.2. Segmentation pipeline. a) Input RGB image taken from top view. b) Initial segmentation is obtained by classifying each pixel as foreground (cyan), background (yellow) or unknown (magenta). c) The final segmentation mask is provided by the GrabCut algorithm.

3.1. Background segmentation

The input of our pipeline for pose estimation of a spread garment is a single RGB image taken with one RGBD sensors attached to the wrists of both robotic arms. It shows the top view of the garment spread out on a flat surface, which is a wooden table in our experiments. The image may also contain a partial view of a gripper, as the RGBD sensor is attached to the wrist, and the scene around the table. Since the sensor, robotic arms and table are calibrated properly and their relative poses are therefore known, everything except the table and garment laying on it can be masked out.

The segmentation of the garment and underlying surface of the table (Fig. 3.2a) is based on two assumptions, which reduce the task complexity:

- The garment and table have statistically dissimilar colors. It is therefore possible to distinguish them in the input image.
- The color properties of the table do not vary significantly in time and therefore they can be learned from data. In practice, it means that the same RGBD sensor is used for perception under approximately constant lighting conditions.

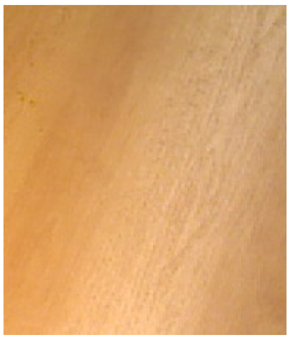
The background color is modeled probabilistically with the Gaussian mixture model (GMM) comprising K components. The likelihood of RGB color $x = (x^R, x^G, x^B)$ is:

$$p(x) = \sum_{k=1}^{K} \pi_k \, \mathcal{N}(x; \mu_k, \Sigma_k), \tag{3.1}$$

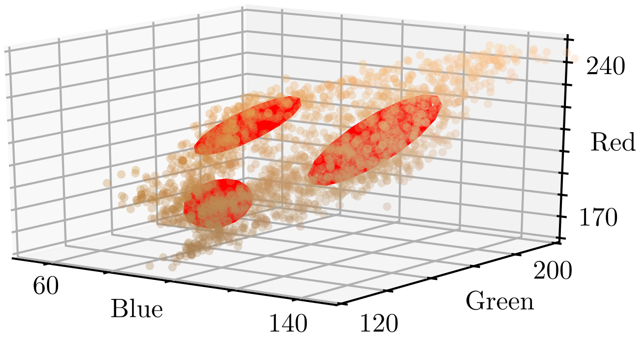
where π_k is a prior probability of the k-th component and $\mathcal{N}(z; \mu_k, \Sigma_k)$ denotes a 3D normal distribution having a mean vector μ_k and covariance matrix Σ_k.

The mixture is learned from the training set containing colors of pixels from the background. The number of GMM components K is determined empirically, based on the number of color clusters visible in training data. We use three components for the wooden table in our experiments.

The training data (Fig. 3.3a) are split to K clusters $C_1, \ldots, C_K$ at first, employing the binary tree algorithm for the palette design [Orchard and Bouman, 1991]. It starts with assigning all RGB vectors to a single cluster and then iteratively constructs a binary tree like hierarchy of clusters in the top-bottom manner. The cluster having the largest variance is split to two new clusters in each iteration. The separating plane passes



a) Background sample



b) GMM representation of background RGB colors

Figure 3.3. Pixel colors from a) sample background image are used to train b) probabilistic model using GMM. The training data are visualized as points in RGB space assigned the corresponding color. GMM components are visualized as red ellipsoids. They are centered at means of the Gaussians and their shape is determined by covariance matrices.

through the center of the cluster and its normal vector is parallel with the principal eigenvector of the cluster.

The prior probability π_k, mean vector μ_k and covariance matrix Σ_k for the k-th GMM component is estimated using the maximum likelihood principle [Duda et al., 2000] from RGB vectors contained in the corresponding cluster C_k:

$$\pi_k = \frac{|C_k|}{\sum_{l=1}^{K} |C_l|}, \tag{3.2}$$

$$\mu_k = \frac{1}{|C_k|} \sum_{x \in C_k} x, \tag{3.3}$$

$$\Sigma_k = \frac{1}{|C_k|} \sum_{x \in C_k} (x - \mu_k)(x - \mu_k)^\top. \tag{3.4}$$

The learned GMM (Fig. 3.3b) of the background color provides an initial partial segmentation (Fig. 3.2b) of the input image. A pixel having the color x is assigned the label z based on the likelihood (3.1) of being part of the background:

$$z = \begin{cases} \text{foreground}, & p(x) < P_F, \\ \text{unknown}, & P_F \leq p(x) \leq P_B, \\ \text{background}, & P_B < p(x). \end{cases} \tag{3.5}$$

The probability thresholds P_F and P_B are set so that 3 % training pixels, which are background samples (Fig. 3.3a), have likelihood lower than P_F and 80 % greater than P_B. It means that 80 % training data would be classified correctly, 3 % misclassified as foreground and 17 % would remain undecided.

The final segmentation (Fig. 3.2c) is found using the GrabCut [Rother et al., 2004] segmentation algorithm. The standard GrabCut requires the user to provide samples of the foreground and background pixels. In our case, the samples are provided automatically. They are formed by the pixels labeled as foreground and background (3.5) in the partial segmentation (Fig. 3.2b).

Based on the provided samples, the GrabCut builds the initial probabilistic models of the foreground and background color, again using the GMM (3.1). The GrabCut

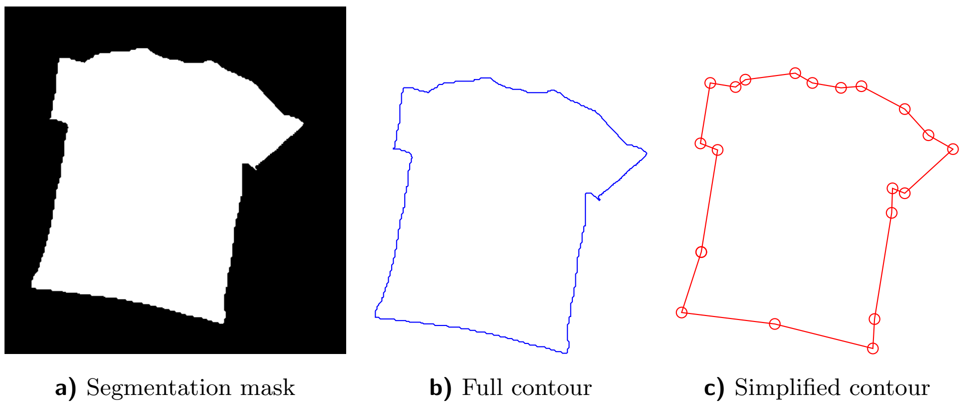


Figure 3.4. Contour processing pipeline. a) The input is a binary segmentation mask. b) The garment contour is extracted from the mask. c) The contour is approximated by a polygon.

is an iterative optimization algorithm. In each iteration, the GMMs of foreground and background colors are re-estimated from the actual pixel labeling. The pixels are relabeled, minimizing an energy function enforcing smoothness of the segmentation. The optimization is interrupted after three iterations in our implementation instead of iterating until convergence, as the later iterations bring only minor refinements. The result is the garment segmented from its background (Fig. 3.2c).

3.2. Contour processing

The garment contour is extracted from the obtained segmentation mask (Fig. 3.4a) using the Moore's algorithm [Gonzalez et al., 2009] for boundary tracking. The contour consists of hundreds points corresponding to the pixels at the garment outline (Fig. 3.4b). Their exact number depends mainly on the garment size, its distance from the camera and the image resolution. However, the contour shape is rather simple and therefore can be approximated sufficiently by a polygon, which has dozens vertices at most (Fig. 3.4c).

The simplification method is based on the algorithm for the optimal approximation of an open curve by a polyline [Perez and Vidal, 1994]. The algorithm utilizes a dynamic programming approach to minimize the overall distance of the original contour points to the segments of the approximating polyline. The number of segments is a parameter of the method. The algorithm can be also utilized to approximate a closed curve by a polygon. One point on the curve is chosen and considered to form both ends of an open curve, therefore breaking the cycle. To find the optimal polygonal approximation, the algorithm would have to be run for each point of the original closed curve. In practice, we stop it after several iterations to obtain a sufficient approximation.

3.3. Polygonal models of garment contour

The possible shapes of the outer contour for various garments are described with polygonal models. There is a separate model defined for each considered category of garments, which are towels, shorts, pants, short-sleeved T-shirts and long-sleeved shirts in our case (Fig. 3.5). The polygonal model is determined by its vertices and their mutual positions. The vertices were defined manually. They correspond to the important landmark points on the garment contour, e.g. corners or armpits.

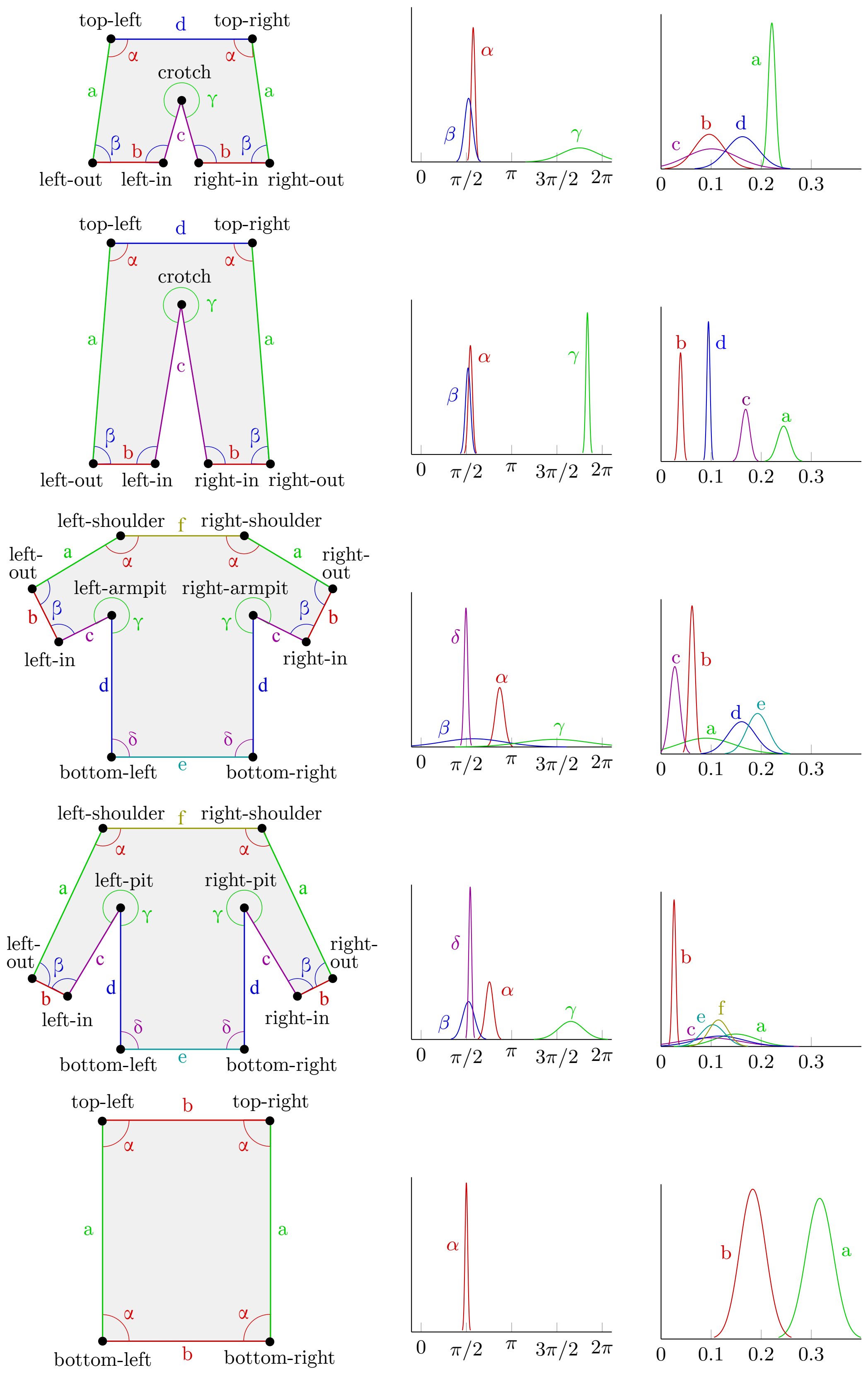


Figure 3.5. Polygonal models for various categories of garment. Learned distributions of inner angles in radians (middle) and relative edge lengths with respect to the contour length (right). The names and colors of the angles and edges are consistent over all three columns.

The mutual positions of the vertices are learned from data. They are described and represented probabilistically (Fig. 3.5). We learn a probability distribution of the inner angles adjacent to the particular vertex. There is also a probability distribution of the relative length for each edge, given the total circumference of the polygonal model. Some distributions are shared among more vertices or edges of the particular model, because of the obvious left-right and top-bottom symmetries.

The inner angles and relative edge lenghts are modeled with 1D normal distributions (Fig. 3.5). Their means and variances were estimated according to the maximum likelihood principle [Duda et al., 2000], using the training images of garments that were annotated manually with the polygonal models. The usage of normal distributions was decided empirically based on the distributions observed in training data.

Our approach can be extended easily for new categories of garments by defining a new model and learning its parameters. We use the following models in our experiments:

- The towel is determined by its 4 corner vertices. All incident inner angles share the identical probability distribution. It is obvious that the angles should be approximately right. The top edge shares the length distribution with the bottom edge and their length is the towel width. The left and right edge also share the distribution and equal to the towel height. We define the height to be longer than the width in order to deal with the ambiguity.
- The shorts are determined by 7 vertices, including 2 symmetric top corners, 4 symmetric corners at the inner and outer ends of legs and a single crotch. The symmetric vertices share the distributions of angles. The edges forming both legs share distributions of lengths due to the left-right symmetries.
- The pants are determined by the same 7 vertices as the shorts. The most important difference is that the relative lengths of legs are obviously longer than for the shorts.
- The short-sleeved T-shirt is determined by 10 vertices, including 2 symmetric shoulders, 4 symmetric inner and outer ends of sleeves, 2 symmetric armpits and 2 symmetric bottom corners. The symmetric vertices share the distributions of angles. The edges on the left and right side share the distributions of the edge lengths.
- The long-sleeved shirt is analogous to the short-sleeved model. The learned relative lengths of the sleeves are obviously longer. Despite its name, the model is also used for jumpers, sweaters or jackets, which all share a similar outer shape.

3.4. Model matching

The polygonal model is matched to the simplified contour in order to estimate the garment pose. The simplified contour consists of N planar points, which are the vertices of the approximating polygon (Fig. 3.4c). The polygonal model is determined by the vertices $v_1, \ldots, v_M$, where M is specific for the particular model, e.g. it equals 4 for a towel having 4 corners. We choose N so that it holds $N > M$, i.e. the simplified contour comprehends more points than is the number of the model vertices.

The matching of the simplified contour points to the polygonal model can be described by the mapping function $f\colon \{p_1, \ldots, p_N\} \to \{v_1, \ldots, v_M, e\}$ (Fig. 3.6). It holds:

$$f(p_i) = \begin{cases} v_m, & \text{point } p_i \text{ is mapped to vertex } v_m, \\ e, & \text{point } p_i \text{ is aligned to an edge.} \end{cases} \tag{3.6}$$

The mapping f needs to satisfy the following constraints in order to describe a valid matching of the model to the simplified contour:

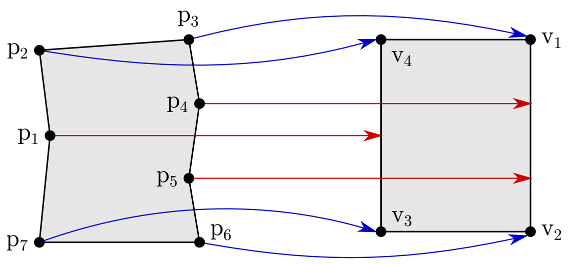


Figure 3.6. Visualization of the function f mapping the points $p_1, \ldots, p_N$ to the vertices $v_1, \ldots, v_M$ (blue arrows). Some points are mapped to the model edges instead (red arrows). The mapping preserves the clockwise ordering of points and vertices.

- For each vertex v_m there exists a point p_i mapped to it: $\forall v_m \exists p_i \colon f(p_i) = v_m$.
- No two points p_i and p_j can be mapped to the same vertex v_m. However, some points remain unmapped to any vertex, being mapped to the model edge represented by the special symbol e instead: $\forall p_i \neq p_j \colon f(p_i) = f(p_j) \Rightarrow f(p_i) = e = f(p_j)$.
- The mapping preserves the ordering of the contour points and the model vertices in the clockwise direction.

The number of all possible mappings f satisfying the aforementioned constraints can be enumerated easily. First, one of N points is selected to be mapped to the vertex v_1. Then an arbitrary subset of $M-1$ points from the remaining $N-1$ points is mapped to the vertices $v_2, \ldots, v_M$. Thus the number of all possible mappings is:

$$N\binom{N-1}{M-1} \geq N\left(\frac{N-1}{M-1}\right)^{M-1}. \tag{3.7}$$

Each possible mapping function f is associated a cost $C(f)$. The total cost is given by the summation of various local costs, which express the local qualities of the particular mapping. Their definition utilizes special cyclic increment and decrement operators, ensuring that the updated point index i and vertex index m stay in their valid ranges. The cyclic operators are defined as:

$$i \oplus 1 = \begin{cases} i+1, & i < N, \\ 1, & i = N, \end{cases} \tag{3.8}$$

$$i \ominus 1 = \begin{cases} i-1, & i > 1, \\ N, & i = 1, \end{cases} \tag{3.9}$$

$$m \boxplus 1 = \begin{cases} m+1, & m < M, \\ 1, & m = M, \end{cases} \tag{3.10}$$

$$m \boxminus 1 = \begin{cases} m-1, & m > 1, \\ M, & m = 1. \end{cases} \tag{3.11}$$

The first of the local costs is the vertex matching cost $V^m_{i,j,k}$, which is defined for each triple of points p_i, p_j, p_k from the simplified contour and each model vertex v_m:

$$V^m_{i,j,k} = -\lambda_V \log \mathcal{N}(|\angle p_i p_j p_k|; \mu_m, \sigma^2_m). \tag{3.12}$$

It expresses how the size of the oriented angle $|\angle p_i p_j p_k|$ fits the normal distribution $\mathcal{N}(\ \cdot\ ; \mu_m, \sigma^2_m)$ of the inner angles adjacent to the vertex v_m. The distribution mean μ_m and variance σ^2_m are learned from data (Fig. 3.5). The cost is weighted by λ_V.

The edge matching cost $E^m_{j,k}$ is defined for each pair of points p_j, p_k and each polygonal model vertex v_m:

$$E^m_{j,k} = -\lambda_E \log \mathcal{N}\left(\frac{\|\overline{p_j p_k}\|}{\sum_{i=1}^n \|\overline{p_i p_{i\oplus 1}}\|}; \nu_m, \tau_m^2\right). \tag{3.13}$$

It expresses how the relative length of the line segment $\overline{p_j p_k}$ (with respect to overall length of the contour) fits the distribution of relative lengths of the model edge $\overline{v_m v_{m\boxplus 1}}$. The distribution mean ν_m and variance τ_m^2 are learned from data (Fig. 3.5). The cost is weighted by λ_E.

The segment matching cost is defined for each point p_i from the simplified contour in the following way:

$$S_i = -\lambda_S \log \mathcal{N}(|\angle p_{i\ominus 1} p_i p_{i\oplus 1}|; \xi, \phi^2). \tag{3.14}$$

For simplicity, we also define the aggregated segment matching cost $S_{j,k}$ for each pair of points p_j, p_k from the simplified contour as follows:

$$S_{j,k} = \sum_{i \in I_{j,k}} S_i = -\lambda_S \sum_{i \in I_{j,k}} \log \mathcal{N}(|\angle p_{i\ominus 1} p_i p_{i\oplus 1}|; \xi, \phi^2). \tag{3.15}$$

The range of indices $I_{j,k}$ passed through by the index i is defined as:

$$I_{j,k} = \begin{cases} \{j+1, \ldots, k-1\}, & j \leq k, \\ \{j+1, \ldots, N, 1, \ldots, k-1\}, & j > k. \end{cases} \tag{3.16}$$

The segment costs represent the penalty paid for points mapped to the model edge instead of vertex. Such points together with their neighboring segments should resemble straight lines. Thus each angle $\angle p_{i\ominus 1} p_i p_{i\oplus 1}$ should resemble a straight angle. This is why the mean and the variance of the normal distribution are set empirically to π and $\pi^2/16$. The costs are weighted by λ_S.

Weights of the matching costs were set empirically as $\lambda_V = 1$, $\lambda_E = 1/3$ and $\lambda_S = 1$ to balance the typical values of the costs. Note that both vertex and segment matching cost evaluate angles and therefore their weights are equal, whereas the edge matching cost evaluate relative lengths.

The matching algorithm aims at finding such mapping f^* of the simplified contour points to the model vertices, that would minimize the associated cost $C(f)$:

$$f^* = \arg\min_f C(f). \tag{3.17}$$

The number of possible mappings is exponential in the number of model vertices M, as shown in (3.7). It would be therefore inefficient to evaluate the costs exhaustively. Sec. 3.5 and Sec. 3.6 propose efficient polynomial algorithms for exploring the space of possible mappings and finding the globally optimum mapping. The local matching algorithm [Stria et al., 2014a], described in Sec. 3.5, finds the optimum matching with respect to more local version of the vertex matching cost and the segment matching cost. The global matching algorithm [Stria et al., 2014b], introduced in Sec. 3.6, optimizes with respect to the combination of vertex, edge and segment matching cost.

3.5. Local matching algorithm

The local matching algorithm optimizes the cost with respect to the local neighborhoods of the contour points that are mapped to the model. The cost has the following form:

$$C_{\mathrm{loc}}(f) = \sum_{f(p_i)=v_m} V_i^m + \sum_{f(p_i)=e} S_i. \tag{3.18}$$

It is a sum of the costs for mapping individual points (Fig. 3.7). Each point p_i mapped to the edge is penalized with the segment cost S_i (3.14). The cost for mapping it to the vertex v_m is a local version of the vertex cost (3.12):

$$V_i^m = V_{i\ominus 1,i,i\oplus 1}^m. \tag{3.19}$$

The local cost $C_{\mathrm{loc}}(f)$ is minimized with the dynamic programming based approach. Alg. 1 lists the main part of the optimization procedure. It finds the optimum mapping with respect to the additional constraint:

$$1 \leq i_1 < i_2 < \ldots < i_M \leq N \text{ for } f(p_{i_m}) = v_m \text{ and } m \in \{1, \ldots, M\}. \tag{3.20}$$

It says that both sequences of the point indices and vertex indices, to which the points were mapped, are ordered in ascending. Alg. 1 basically chooses one of the leading points and maps it to the vertex v_1, some of its successors along the contour is mapped to v_2, some successor of the already mapped successor to v_3 and so forth.

Alg. 1 fills the cost matrix $T \in \mathbb{R}^{N\times M}$ incrementally, where T_i^m is the cost paid for matching the sub-contour $(p_1, \ldots, p_i)$ to the model vertices $(v_1, \ldots, v_m)$. To find the cost T_i^m, Alg. 1 tries to map various points p_j, $j \in \{m, \ldots, i\}$, to the vertex v_m, using the previously computed costs for matching the sub-contours $(p_1, \ldots, p_{j-1})$ to the model vertices $(v_1, \ldots, v_{m-1})$. Fig. 3.7 shows an example of a single iteration of the mininimization. The result T_N^M is the cost of mapping all N contour points to all M model vertices.

Algorithm 1 Local matching algorithm

```
**Input:**
$V \in \mathbb{R}^{N\times M}$, $V_j^m$ = cost of matching angle $\angle p_{j\ominus 1} p_j p_{j\oplus 1}$ to vertex $v_m$
$S \in \mathbb{R}^{N}$, $S_k$ = cost of mapping point $p_k$ to edge

**Output:**
$T \in \mathbb{R}^{N\times M}$, $T_i^m$ = cost of matching sub-contour $(p_1, \ldots, p_i)$ to vertices $(v_1, \ldots, v_m)$

for all $i \in \{1, \ldots, N\}$ do
    $T_i^1 \leftarrow \min_{j\in\{1,\ldots,i\}} \left( \sum_{k=1}^{j-1} S_k + V_j^1 + \sum_{k=j+1}^{i} S_k \right)$
end for

for all $m \in \{2, \ldots, M\}$ do
    for all $i \in \{m, \ldots, N\}$ do
        $T_i^m \leftarrow \min_{j\in\{m,\ldots,i\}} \left( T_{j-1}^{m-1} + V_j^m + \sum_{k=j+1}^{i} S_k \right)$
    end for
end for

return $T_N^M$
```

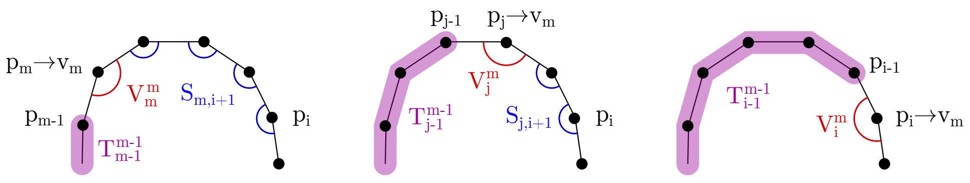


Figure 3.7. Minimization of the total matching cost T_i^m goes over $j \in \{m, \ldots, i\}$, using the previously computed costs T_{j-1}^{m-1}, vertex matching cost V_j^m and segment matching costs S_k. The figure shows three example candidates for the optimum matching for matching sub-contour $(p_1, \ldots, p_i)$ to vertices $(v_1, \ldots, v_m)$. See Alg. 1 for details.

In order to find the optimum solution with respect to (3.18) and not constrained by (3.20), Alg. 1 is called repeatedly for every simplified contour shifted by n positions, where $n \in \{0, \ldots, N-1\}$:

$$\left(p_1^{(n)}, \ldots, p_N^{(n)}\right) = (p_{n+1}, \ldots, p_N, p_1, \ldots, p_n). \tag{3.21}$$

Rows of the matrix $V \in \mathbb{R}^{N \times M}$, containing the precomputed vertex matching costs, and elements of the vector $S \in \mathbb{R}^N$, containing the precomputed segment matching costs, are also shifted by n positions to match the shifted contour. The shifted cost matrix $V^{(n)} \in \mathbb{R}^{N \times M}$ and vector $S^{(n)} \in \mathbb{R}^N$ are:

$$\left[V^{(n)}\right]_{(1,\ldots,N)}^m = V_{(n+1,\ldots,N,1,\ldots,n)}^m, \tag{3.22}$$

$$\left[S^{(n)}\right]_{(1,\ldots,N)} = S_{(n+1,\ldots,N,1,\ldots,n)}. \tag{3.23}$$

The costs are precomputed in $\mathcal{O}(NM)$ time. Both minimizations in Alg. 1 for computing T_i^1 and T_i^m can be performed incrementally in $\mathcal{O}(N)$ time by remembering the summation values for previous j's. The minimization is called $\mathcal{O}(NM)$ times to fill all elements in $T \in \mathbb{R}^{N \times M}$. Thus Alg. 1 runs in $\mathcal{O}(N^2M)$ time. It is called N times for all possible shifts of the contour, as defined in (3.21). Thus the overall time complexity of the local matching algorithm is $\mathcal{O}(N^3M)$. It is therefore polynomial both in the number of simplified contour points and model vertices. Despite the relatively high polynomial degree, the algorithm runs effectively, as we show in the experimental evaluation (Sec. 3.9).

3.6. Global matching algorithm

The global matching algorithm optimizes the cost with respect to more global neighborhoods of the contour points. It has the following form:

$$C_{\text{glob}}(f) = \sum_{\substack{m \in \{1,\ldots,M\} \\ i,j,k \in \{1,\ldots,N\}}} \left\{ V_{i,j,k}^m + E_{j,k}^m + S_{j,k} \mid f(p_i) = v_{m \boxminus 1}, f(p_j) = v_m, f(p_k) = v_{m \boxplus 1} \right\} \tag{3.24}$$

Each point p_j mapped to the vertex v_m is penalized with the vertex cost $V_{i,j,k}^m$ (3.12), given that two selected points p_i and p_k are mapped to the neighboring vertices $v_{m \boxminus 1}$ and $v_{m \boxplus 1}$ (Fig. 3.8a). The segment $\overline{p_j p_k}$ is penalized with the edge cost $E_{j,k}^m$ (3.13) for matching the model edge $\overline{v_m v_{m+1}}$. The segment cost $S_{j,k}$ (3.15) is paid for approximating the sub-contour $(p_l | l \in I_{j,k})$ with the segment $\overline{p_j p_k}$.

Algorithm 2 Global matching algorithm

```
Input:
$r$ = index of point mapped to vertex $v_M$, i.e. $f(p_r) = v_M$
$V \in \mathbb{R}^{N\times N\times N\times M}$, $V^m_{i,j,k}$ = cost of matching angle $\angle p_i p_j p_k$ to vertex $v_m$
$E \in \mathbb{R}^{N\times N\times M}$, $E^m_{j,k}$ = cost of mapping segment $\overline{p_j p_k}$ to edge $\overline{v_m v_{m+1}}$
$S \in \mathbb{R}^{N\times N}$, $S_{j,k}$ = cost of approximating sub-contour $(p_{j+1}, \dots, p_{k-1})$ by $\overline{p_j p_k}$
Output:
$T \in \mathbb{R}^{N\times N\times M}$, $T^m_{j,k}$ = cost of mapping sub-contour $(p_1, \dots, p_{k-1})$ to vertices
                    $(v_1, \dots, v_m)$ s.t. $f(p_j) = v_m, f(p_k) = v_{m+1}$
for all $j \in \{2, \dots, r-M+2\}$ do
    for all $k \in \{j+1, \dots, r-M+3\}$ do
        $T^2_{j,k} \leftarrow \left(V^1_{r,1,j} + V^2_{1,j,k}\right) + \left(E^M_{r,1} + E^1_{1,j} + E^2_{j,k}\right) + (S_{r,1} + S_{1,j} + S_{j,k})$
    end for
end for
for all $m \in \{3, \dots, M-1\}$ do
    for all $j \in \{m, \dots, r-M+m\}$ do
        for all $k \in \{j+1, \dots, r-M+m+1\}$ do
            $T^m_{j,k} \leftarrow E^m_{j,k} + S_{j,k} + \min_{i\in\{m-1,\dots,j-1\}} \left(T^{m-1}_{i,j} + V^m_{i,j,k}\right)$
        end for
    end for
end for
return $T^M_{r,1} \leftarrow \min_{i\in\{M-1,\dots,r-1\}} \left(T^{m-1}_{i,r} + V^m_{i,r,1}\right)$
```

The global cost $C_{\text{glob}}(f)$ is minimized with the approach based on dynamic programming, similarly to the local cost in Sec. 3.5. Alg. 2 lists the main part of the optimization procedure. It finds the optimum mapping with respect to the additional constraints:

$$f(p_1) = v_1, \tag{3.25}$$

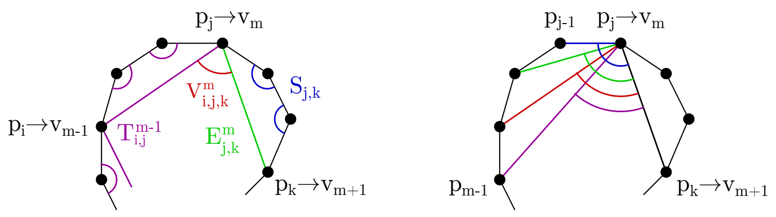
$$f(p_r) = v_M, \text{ where } r \in \{M, \dots, N\}, \tag{3.26}$$

$$1 < i_2 < \ldots < i_{M-1} < r \text{ for } f(p_{i_m}) = v_m \text{ and } m \in \{2, \dots, M-1\}. \tag{3.27}$$

The first two constraints fix the points mapped to the vertices v_1 and v_M. The last constraint is analogous to (3.20) for the local matching, saying that the indices of points and vertices, which the points are mapped to, proceed in ascending order. Alg. 2 finds the optimal mapping of the sub-contour $(p_2, \dots, p_{r-1})$ to the vertices $(v_2, \dots, v_{M-1})$.

Alg. 2 fills the cost tensor $T \in \mathbb{R}^{N\times N\times M}$ incrementally. The value $T^m_{j,k}$ is the cost paid for matching the sub-contour $(p_1, \dots, p_{k-1})$ to vertices $(v_1, \dots, v_m)$, so that $f(p_j) = v_m$ and $f(p_k) = v_{m+1}$ (Fig. 3.8a), i.e. the points p_j and p_k mapped to vertices forming the model edge $\overline{v_m v_{m+1}}$ are fixed. Note that the vertex cost paid for $f(p_k) = v_{m+1}$ is not included in $T^m_{j,k}$ (Fig. 3.8a). In each iteration, Alg. 2 tries to map various candidate points p_i, where $i \in \{m-1, \dots, j-1\}$, to the vertex v_{m-1} (Fig. 3.8b). Each such mapping determines the vertex cost $V^m_{i,j,k}$ for matching the angle $\angle p_i p_j p_k$ to the vertex v_m. The result $T^M_{r,1}$ is the cost of mapping all N contour points to all M model vertices, satisfying the constraints (3.25–3.27).

In order to find the optimum solution with respect to (3.24) and not constrained by (3.25–3.27), Alg. 2 is called repeatedly for each possible combination of shifting



a) Various matching costs **b)** Minimization of cost $T^m_{j,k}$

Figure 3.8. a) The total matching cost $T^m_{j,k}$ combines previously computed total cost $T^{m-1}_{i,j}$ with the vertex cost $V^m_{i,j,k}$, edge cost $E^m_{j,k}$ and segment cost $S_{j,k}$. b) Minimization of $T^m_{j,k}$ involves exploration of various candidate points p_i mapped to v_{m-1}, where $i \in \{m-1, \dots, j-1\}$.

the simplified contour by n positions, as in (3.21), with selection of the point mapped to v_M. Elements of the tensor $V \in \mathbb{R}^{N \times N \times N \times M}$, containing the precomputed vertex costs, elements of the tensor $E \in \mathbb{R}^{N \times N \times M}$, containing the precomputed edge costs, and elements of the matrix $S \in \mathbb{R}^{N \times N}$, containing the precomputed segment matching costs, are shifted to match the shifted contour, analogously to (3.22) and (3.23).

The costs are precomputed in $\mathcal{O}(N^3M + N^2M + N^2) = \mathcal{O}(N^3M)$ time. The minimization of $T^m_{j,k}$ in Alg. 1 can be performed incrementally in $\mathcal{O}(N)$ time by remembering the summation values for previous i's. The minimization is called $\mathcal{O}(N^2M)$ times to fill all elements in $T \in \mathbb{R}^{N \times N \times M}$. Thus Alg. 2 runs in $\mathcal{O}(N^3M)$ time. It is called for each possible combination of N shifts of the contour with $N - M + 1$ options of selecting $f(p_r) = v_M$, i.e. $\mathcal{O}(N^2)$ times in total. The overall time complexity of the global matching algorithm is therefore $\mathcal{O}(N^5M)$. Although that the degree of the polynomial is rather high, the experiments (Sec. 3.9) show that it runs nearly in real-time. It is mainly because the number of simplified contour points N is chosen between 10 and 20, depending on the complexity of the matched model, which is enough for a precise approximation of the original contour.

3.7. Folded models

The proposed algorithms for matching the simplified contour of a garment to the model are not used for the spread garments only. After recognizing the initial spread configuration of the garment, the robot performs a single fold, which means e.g. folding one sleeve of a shirt or folding a towel in half. The garment configuration needs to be checked afterwards because of its possible unintended translation or rotation caused by manipulation. The most common issue is slipping of the garment on a table. It is caused by the horizontal pulling force generated by the gripper that holds the garment. The pulling force can be minimized by following a specific folding trajectory, which considers material properties of the garment [Petrík et al., 2016, Petrík et al., 2017].

To check a partially folded configuration of the garment, an image is taken, segmented and the extracted contour is simplified, as described in Sec. 3.1–3.2. The simplified contour is matched to the polygonal model of a folded garment, which is derived from the original model. The process is fully automated. The parameters of the original model are adjusted to that concrete garment at first. The adjusted model is folded virtually, based on the planned fold.

Fig. 3.9 illustrates the incremental creation of the folded models. The original vertices are being replaced by the vertices denoting endpoints of individual folds. The s-th fold

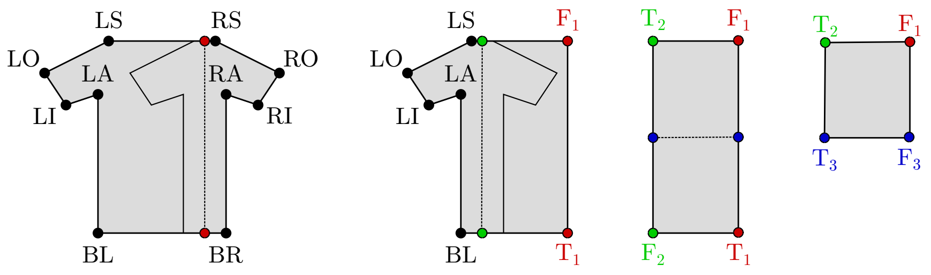


Figure 3.9. Incremental creation of the folded models for a short-sleeved T-shirt. Some original vertices are being replaced by new vertices, denoting endpoints of the individual folds (plotted in various colors).

is performed over the folding axis denoted by two endpoints F_s and T_s. They are ordered in the clockwise direction along the outline. The endpoints form two new vertices F_s and T_s that are connected by an edge in the folded model. All the original vertices positioned between F_s and T_s are removed. The vertices in the proximity of F_s and T_s are removed as well to keep the resulting folded model simple. In Fig. 3.9, the right shoulder (RS) and bottom-right corner (BR) are removed during the first fold, while the left shoulder (LS) and bottom-left corner (BL) are removed during the second fold.

The distributions of inner angles and relative edge lengths, which are used to evaluate the vertex and edge costs, are adjusted to correspond to the actually perceived garment and planned fold. The mean μ_m, parameterizing the vertex cost $V^m_{i,j,k}$ (3.12), is updated to the angle $\angle v_{m\boxminus 1} v_m v_{m\boxplus 1}$ measured in the fitted model. The mean ν_m, parameterizing the edge cost $E^m_{j,k}$ (3.13), is updated to the relative edge length $\overline{v_m v_{m\boxplus 1}}$ measured in the fitted model. All variances σ^2_m (3.12) and τ^2_m (3.13) are updated to the least variance learned for the original model. The variances for the newly added vertices and edges are set to twice that value because of the uncertainty in the planned fold. By updating the means and variances, prior information about a shape of the concrete piece of garment is encoded directly into the model.

3.8. Robotic folding

Once the garment configuration is known, it can be folded by a dual-arm robot. Sec. 4.1 describes the robotic testbed used in the experiments. The individual folding operations and their order are defined manually. Single operation corresponds e.g. to folding one sleeve or folding a towel in half. The manipulation is adjusted to the shape, size and configuration of the particular piece of garment.

Each folding operation is parameterized by the points on the garment contour, which need to be grasped, and by the axis, over which the grasped garment should be folded. Positions of the grasping points, as well as position and orientation of the folding axis are specified relative to the vertices of the matched polygonal model. E.g. in order to fold a spread towel in half, two corners sharing the shorter hemline need to be grasped and folded over the axis passing through the centers of the longer hemlines. The folding axis is therefore approximately perpendicular to the longer hemlines.

We adopt the gravity based folding approach proposed by [Berg et al., 2011]. The method assumes an infinite flexibility of the garment and an infinite friction between the garment and table. Despite these unrealistic assumptions, the gravity based folding performs well on real garments in practice. It is also less computationally expensive than

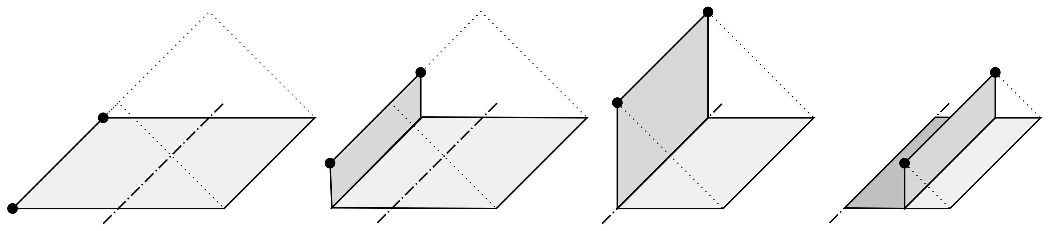


Figure 3.10. Gravity based folding [Berg et al., 2011] is based on grasping the garment contour at defined positions and moving them along triangular paths (dotted lines). The paths are symmetric with respect to the folding axis (dash-dotted line) and culminate above it.

the simulation based methods [Li et al., 2015b, Petrík et al., 2016, Petrík et al., 2017], which consider the material properties.

During the gravity based manipulation [Berg et al., 2011], one part of the garment lays flat on a table, while the other part hangs down (Fig. 3.10). The grippers holding the garment outline follow the triangular paths, ensuring that the hanging part is kept immobilized all the time. The paths are contained in planes perpendicular to the folding axis, are symmetric with respect to the axis and culminate above it (Fig. 3.10).

[Berg et al., 2011] further define theoretical locations of the grasping points that would keep the hanging part of the garment immobilized. E.g. a towel needs to be held for its two neighboring corners (Fig. 3.10), in contrast to holding it with a single gripper only for the center of its hemline, which would cause the hemline to collapse down. The required grasping points depend on the garment shape, mainly on its convexity and concavity. Since the method may require more that two grippers for more complex shapes, we use only a smaller set of the grasping points, minimizing the possibility of deformations. E.g. the wrist of a sleeve is grasped in the middle instead of grasping its corners, which is possible due to its narrowness.

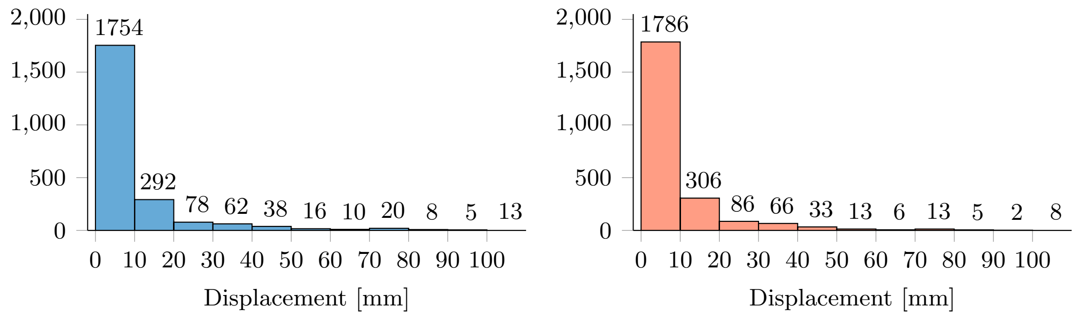
Our dual-arm robot is equipped with the specialized two-finger grippers, designed specifically for the manipulation of soft materials (Sec. 4.1). The lower gripper slides under the garment from side and grasps its outline. Then it follows the previously described triangular trajectory (Fig. 3.10). We use two approaches to generate trajectories for the arms: motion planning and interpolation. The former is used to approach the initial folding position. The Open Motion Planning Library (OMPL) [Şucan et al., 2012] is used to schedule collision free trajectories between multiple states of the joint. After testing several planning algorithms from OMPL, we found out that RRT-Connect [Kuffner and LaValle, 2000] suits best our needs, since it successfully finds a plan in most of the cases and in a reasonable time. The interpolation method generates points distributed uniformly on a curve in Cartesian coordinates and computes the inverse kinematics for each point to produce the final trajectory. It is used to compute the triangular folding trajectory and for the trajectory in the vicinity of the grasping point. The interpolation method provides a full control over the trajectory, which ensures no damage to the garment.

3.9. Experiments

The proposed computer vision pipeline for pose estimation of a garment was evaluated on garments of 5 categories, for which the models were defined and learned from data. Fig. 3.5 shows the parameters of the normal distributions, which specify the vertex

Garment	Local matching			Global matching		
	Success	Displac. [mm]		Success	Displac. [mm]	
		Mean	Stdev.		Mean	Stdev.
Towels	53 / 53	3.0	2.3	53 / 53	3.4	2.3
Shorts	29 / 32	9.9	14.6	32 / 32	8.4	6.4
Pants	53 / 54	6.0	8.6	54 / 54	5.6	6.2
T-shirts	61 / 61	11.7	21.6	61 / 61	10.8	17.5
Shirts	90 / 90	11.7	16.1	90 / 90	10.6	14.3
Total	286 / 290	9.8	16.3	290 / 290	9.0	13.3

Table 3.1. Comparison of the local and global matching procedure evaluated on 290 testing samples. The number of successfully fitted models is shown for each procedure and each category of garments. The mean displacements of the fitted model vertices from the annotated landmark points, as well as their standard deviations were computed from the successfully fitted models only.



a) Local matching **b)** Global matching

Figure 3.11. Displacements of the fitted model vertices from the annotated landmark points. Histograms are computed over all vertices and all testing data, excluding 4 failure cases for the local matching procedure. More than 92 % vertices are displaced less than 30 mm.

(3.12) and edge (3.12) matching costs. The parameters of each model were learned from 20 annotated images of garments posed in various spread configurations. The testing dataset contains 290 images, including 53 images of towels, 32 shorts, 54 pants, 61 short-sleeved T-shirts and 90 long-sleeved shirts.

Two different matching methods were evaluated: the simpler local matching (Sec. 3.5) and more robust global matching (Sec. 3.6). The previous steps of the pipeline, including the segmentation (Sec. 3.1) and contour processing (Sec. 3.2), are shared by both methods. The number of points forming the simplified contour is proportional to the number of vertices for the particular model. We choose $N = M + 10$, i.e. there are 10 more points than the number of vertices for each model (3.7).

The results are summarized in Tab. 3.1. The global matching procedure succeeded in all 290 test cases, while the local matching failed completely in 4 cases. Moreover, the local matching provides slightly less accurate localization of the landmark points. Tab. 3.1 shows the mean displacements of the matched vertices from the manually annotated points, as well as their standard deviations. The aforementioned 4 failures of the local matching procedure were excluded from further quantitative evaluation.

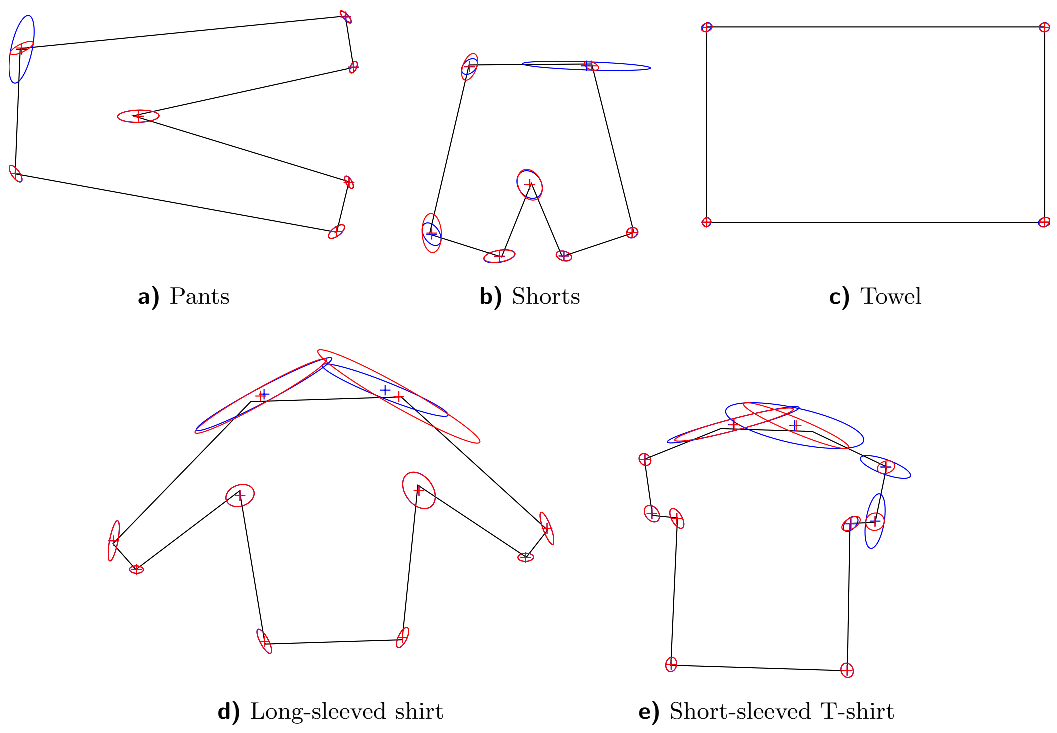


Figure 3.12. Summary of displacements between the vertices of the matched models and manually annotated landmarks. The outlines correspond to the learned models (means of normal distributions of inner angles and relative lengths were used to plot the outlines). The crosses denote the mean displacements for each vertex, and the ellipses correspond to the covariance matrices of the displacements (the ellipses were plotted sized up five times).

The distributions of displacements are not symmetric around the means. They are skewed positively, however, not heavy tailed. Fig. 3.11 shows the histograms. Regarding the global matching, 93.7 % displacements, over all testing samples and all vertices, are below 30 mm, 99.7 % below 100 mm and the largest observed displacement was 135 mm. The displacements of the local matching, excluding 4 complete failures, are below 30 mm in 92.5 % cases and below 100 mm in 99.4 % cases. The displacement of 130 mm was exceeded in a single case only.

Fig. 3.12 visualizes the matching errors for individual vertices of all polygonal models and compares the local and global matching approach. The individual displacement vectors were collected from all testing images, normalized based on the orientation and size of the particular garment and averaged. The covariance matrix of the normalized displacements for each vertex was computed and plotted as an ellipse sized up 5 times.

The largest matching errors were observed for the vertices representing the shoulders of short-sleeved T-shirts and long-sleeved shirts, where the usual displacement ranges from 20 to 40 millimeters. The reason being that the shoulders do not form an easily distinctive shape on the contour and may be confused with a neckline. This makes not only the matching challenging, but also introduces noise to the manual annotations, further increasing the errors. The locations of shoulders are therefore not used for planning of folds. The position and orientation of the folding axis is based on the mutual positions of bottom corners and armpits.

An another source of errors is the imperfect segmentation, which produces various artifacts on the extracted contour. The artifacts confuse primarily the local matching

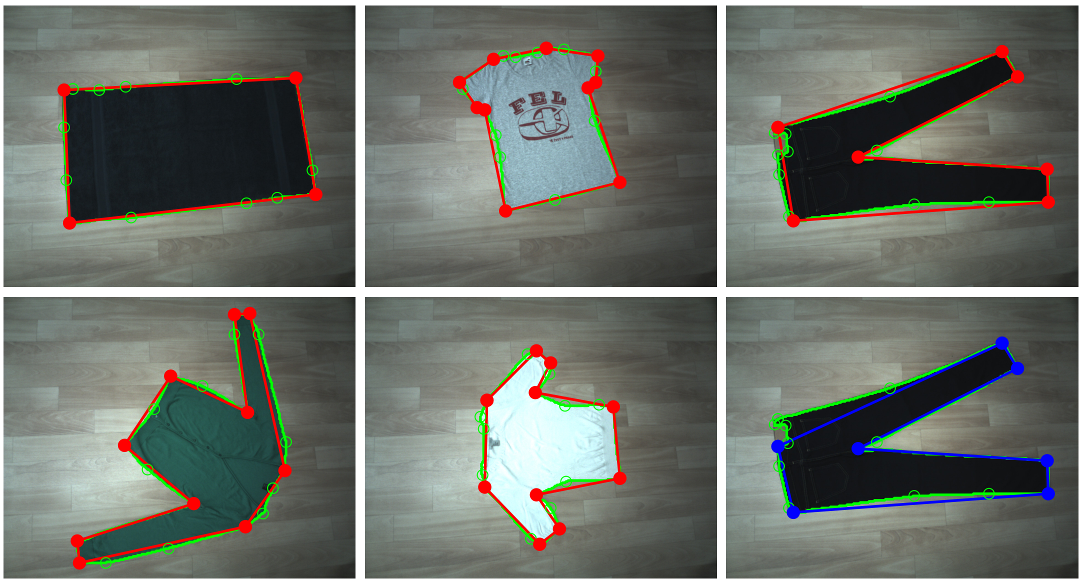


Figure 3.13. Selected examples of the polygonal models matched to the simplified garment contour (shown as green circles connected with segments). The last column compares the accuracy of the global matching (red polygon in the first row) with the local matching (blue polygon in the second row), which got confused by the incorrectly segmented waist label.

procedure, whereas the global algorithm can cope with them. We observed these errors mainly in case of pants and shorts (Fig. 3.12), where the leather waist labels are sometimes segmented as part of the similarly colored background. See Fig. 3.13 for the qualitative comparison of both matching procedures in such case.

The matching errors achieved by our method are sufficiently low for a reliable grasping and planning of the folding operations, considering the size of the gripper and usual size of the garment. The displacements are slightly lower than those reported by [Miller et al., 2011], whose approach is also based on matching the polygonal models to the extracted garment contour. Our visual analysis method is significantly faster, however, enabling nearly a real time execution, compared to 30–150 seconds reported by [Miller et al., 2011] just for the model matching.

The proposed vision pipeline was implemented in MATLAB and C++. The time performance was evaluated on a notebook with Intel i7-3740GM 2.7 GHz processor and 16 GB memory. The segmentation takes 1.2 seconds on average for 640 $\times$ 480 input images. The contour simplification algorithm is the most time consuming operation. It takes between 0.5 and 3.5 seconds almost always, depending mainly on the contour complexity. The subsequent model matching procedures work with the already simplified contours. Their runtime is negligible, compared to the previous steps. It is 0.14 seconds on average for the global matching and less than 0.01 for the local matching. The complete vision pipeline therefore runs in 2–5 seconds almost always.

The robotic experiments[1] were conducted on a dual-arm robotic testbed (Sec. 4.1). The model matched to the garment contour determines the next fold. Positions of the initial grasping points and final releasing points are computed from locations of the model vertices. Several approaching directions of the gripper are generated for each grasping and each releasing point to increase the chance of planning a collision free trajectory successfully.

[1]Folding videos: `http://cmp.felk.cvut.cz/~striajan/phd`

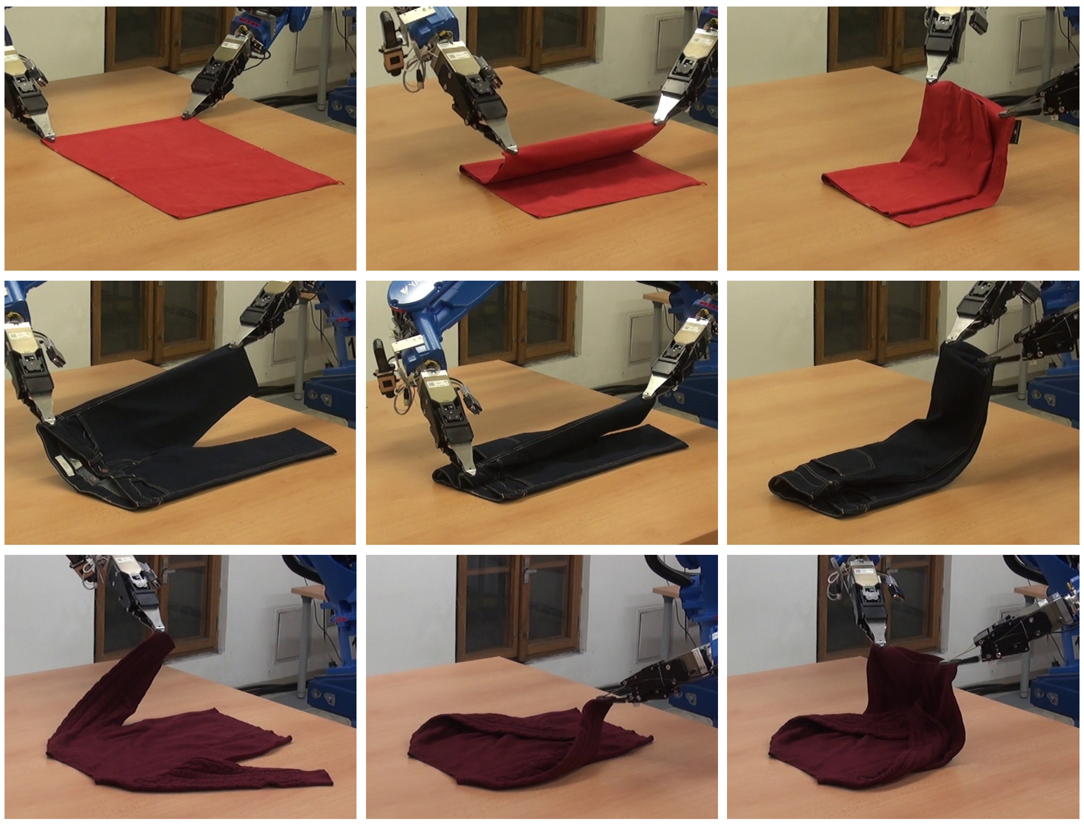


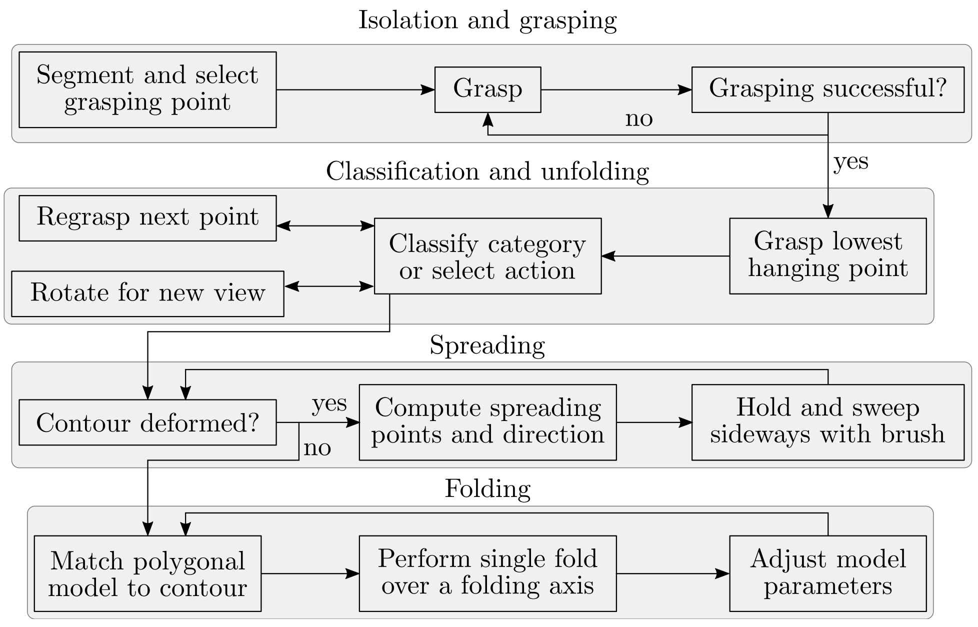
Figure 3.14. A detailed view of the robot folding successfully garments of different categories.

The overall success rate of the proposed folding method, including vision, planning and manipulation, is approximately 90 %. The detailed experiments are provided in Sec. 4.6, where our method was integrated into the pipeline for the folding scenario. Fig. 3.14 shows several successful folding sequences. The most frequent cause of failures in an unsuccessful computation of the inverse kinematics. The second is an unsuccessful motion planning, either due to a limited manipulation range, or due to a possible collision of the robotic arms. The third factor is a slippage of the garment during its folding, resulting in an incorrectly placed fold.

4. Pipeline for folding scenario

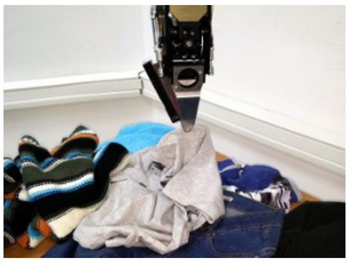
Our method for the pose estimation and folding of unknown spread garments, proposed in Chap. 3, was integrated into the complex pipeline developed by the Clothes Perception and Manipulation (CloPeMa) project [Doumanoglou, Stria et al., 2016]. The pipeline attempts to solve the folding scenario, introduced in Sec. 1.3. This chapter summarizes briefly the components of the pipeline developed by other members of the CloPeMa consortium, to provide a broader context for our folding method. The dual-arm robotic testbed is introduced that was developed by the CloPeMa team and used in all robotic experiments. The performance of the complete pipeline is evaluated.

The presented pipeline comprehends several steps (Fig. 4.1 and 4.2). There is a pile of unknown garments at the beginning, which is perceived with an RGBD sensor. The pile is segmented into separate garments using color and texture information. The ideal grasping point is selected based on the features computed from a depth map, preferring high and narrow wrinkles. The unknown garment is grasped, lifted up and then grasped again for its lowest hanging point to reduce the space of its possible configurations. The recognition and unfolding of the hanging garment are performed in the active manner, utilizing the framework of Active Random Forest (ARF) for the detection of grasping points, while optimizing actions of the robot. The unfolded garment is placed on a table and optionally spread by pulling it horizontally into various directions. Pose of the spread garment is recognized by matching its contour to polygonal models, as described in Chap. 3, and the garment is folded step by step.

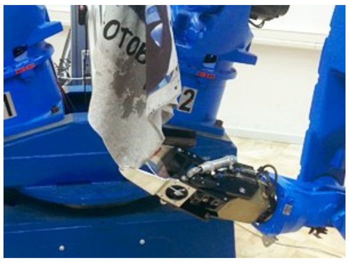


Isolation and grasping
Segment and select grasping point
Grasp
Grasping successful?
no
yes
Classification and unfolding
Regrasp next point
Rotate for new view
Classify category or select action
Grasp lowest hanging point
Spreading
Contour deformed?
yes
no
Compute spreading points and direction
Hold and sweep sideways with brush
Folding
Match polygonal model to contour
Perform single fold over a folding axis
Adjust model parameters

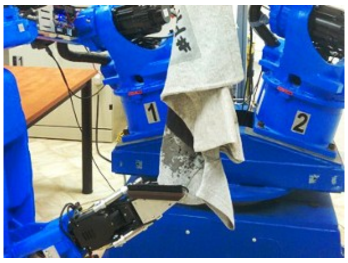
Figure 4.1. Main steps of the proposed pipeline for a folding scenario, which comprehends four perception and manipulation sub-tasks.



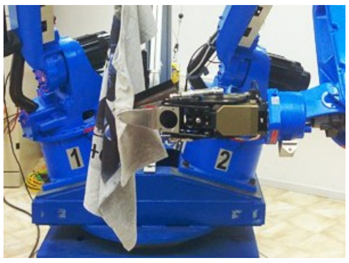
a) Isolate from heap and lift



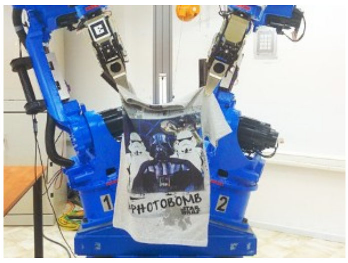
b) Grasp lowest point



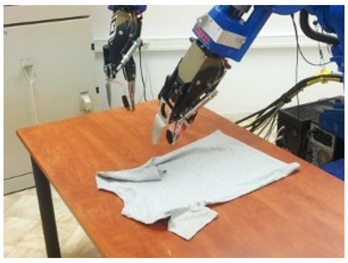
c) Grasp first unfolding point



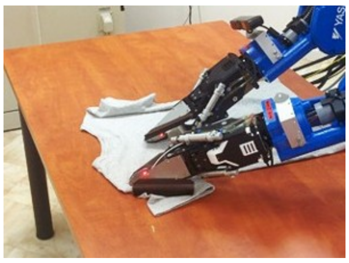
d) Grasp other unfolding point



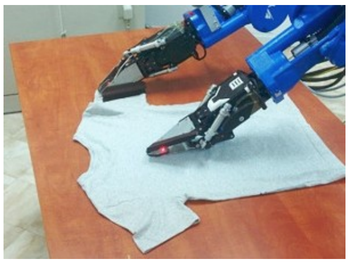
e) Hanging unfolded T-shirt



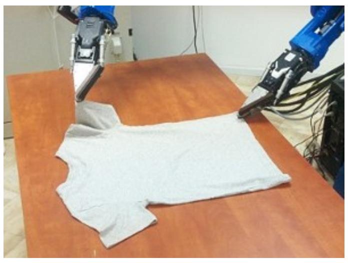
f) Lay down on table



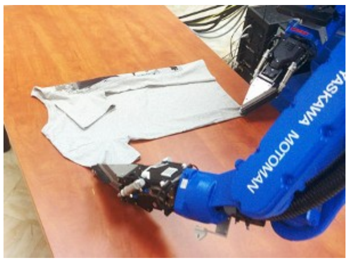
g) Spread left sleeve



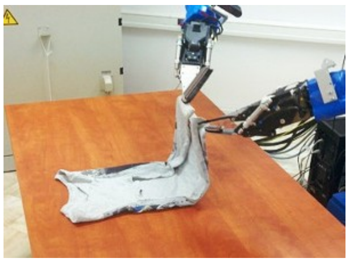
h) Spread right sleeve



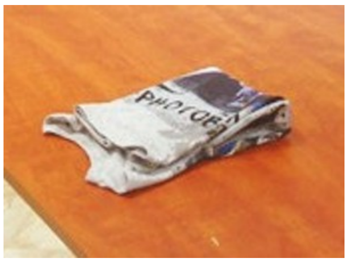
i) Fold right side



j) Fold left side

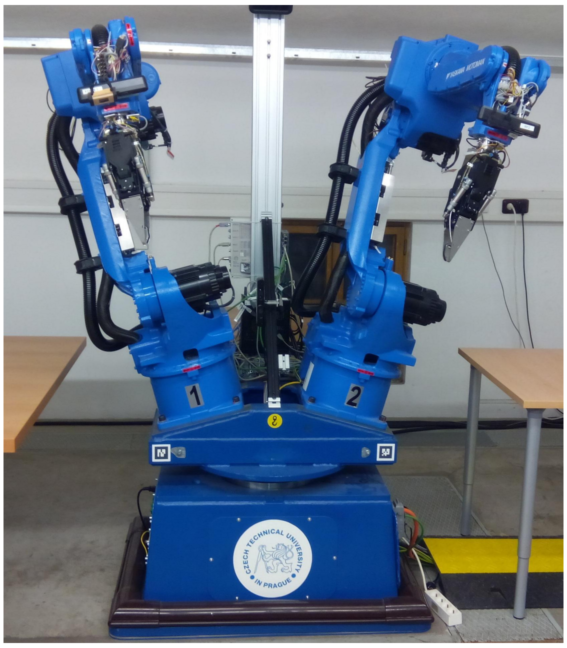
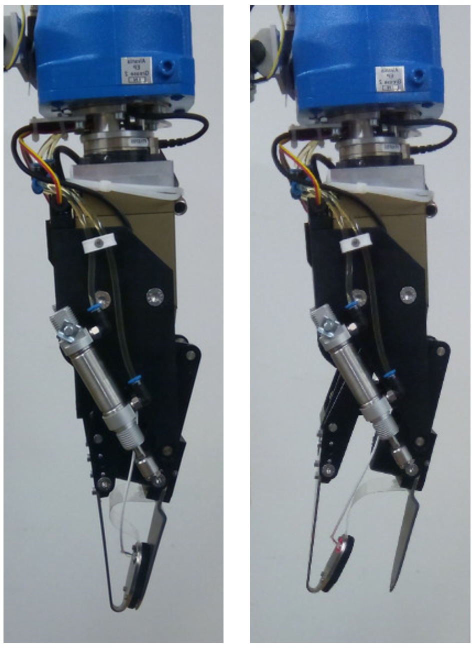


k) Final fold in half



l) Fully folded T-shirt

Figure 4.2. An example run of the pipeline applied on a T-shirt. a) The optimal grasping point is selected, the T-shirt is isolated from a heap and lifted up. b) Its lowest point is regrasped to reduce the number of possible configurations. c) The first unfolding point is grasped with the free arm, the previously lowest point is released and d) the second unfolding point is grasped. e) The T-shirt is unfolded by stretching the grasped points out. f) The unfolded T-shirt is laid on the empty table. g) Both left and h) right sleeve are spread using the brush tool if necessary. i) The folding procedure comprehends folding the right and j) left side of the T-shirt before k) performing the final fold. l) The result is the fully folded T-shirt.

a) Dual-arm CloPeMa robot **b)** Closed and opened gripper

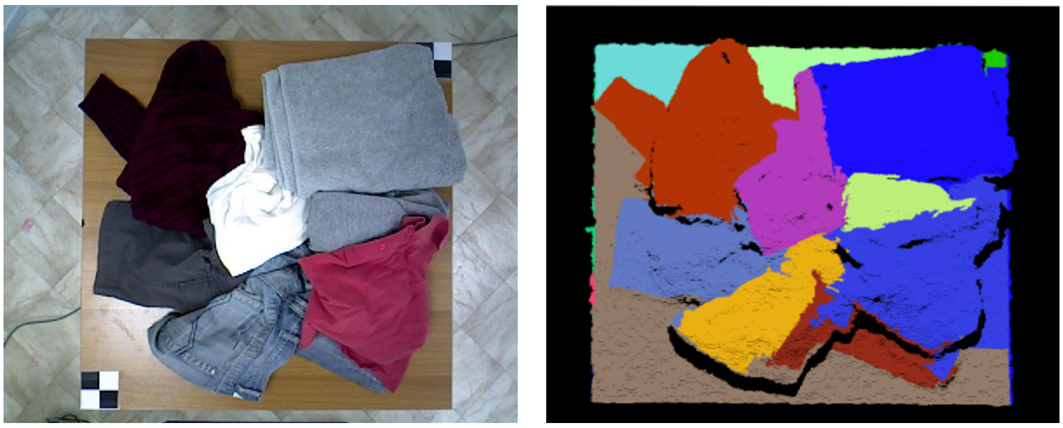
Figure 4.3. CloPeMa testbed is composed mainly of two industrial arms that are attached custom jaw-like grippers and consumer grade RGBD sensors.

4.1. CloPeMa testbed

All robotic experiments described in the presented thesis were performed using the CloPeMa robotic testbed. The robot was designed by the project consortium members. Three similar copies of the robot were built and used by the member institutions. The experimental evaluation of the folding pipeline described in this chapter was performed at the Center for Research of Technology Hellas. All other experiments included in this thesis were performed at the Czech Technical University in Prague.

The robot is composed mainly of standard industrial components. Its body consists of two Motoman MA1400 hollow wrist robotic arms mounted to R750 turn-table (Fig. 4.3a). The components are controlled by two DX100 controllers working as master and slave. The arms are attached jaw-like grippers developed by the CloPeMa consortium and designed specifically for the manipulation of soft materials [Le et al., 2013]. The thin finger is intended for sliding easily under the garment placed on a table, while the other finger moves towards it to close the grip (Fig. 4.3b).

The robot is equipped with several sensors. There are three combined RGBD sensors ASUS Xtion PRO attached to the robot, two at the wrists and one at the base. The provided RGB images have the resolution 1280 × 1024, while the depth maps are 640 × 480 at 30 Hz. The depth sensing range, limited by the structured light technology, is approximately from 80 cm to 3.5 m. The error of depth measurements increases quadratically with the distance and ranges from several millimeters to centimeters [Smíšek et al., 2013]. A head comprising of two Nikon D5100 cameras for stereo vision [Schmidt et al., 2016] and corresponding pan/tilt units is mounted on the robot base. The experiments described in this thesis use only the RGBD sensors attached to the wrists. The grippers are equipped with tactile sensors and photometric stereo for material sensing. The wrists comprehend force and torque sensors.



a) Input RGB image of pile **b)** Segmentation projected to point cloud

Figure 4.4. The input RGB image is segmented based on the color and texture information. The resulting segments form candidates for graspable regions.

The perception and control system of the robot are built on top of the Robotic Operating System (ROS) [Quigley et al., 2009]. The basic functionality of moving the arms and reading positions from their joints is provided by the MotoROS package, which is delivered by the manufacturer. We also utilize the MoveIt package and Open Motion Planning Library (OMPL) [Şucan et al., 2012] for motion planning.

4.2. Isolation and grasping

The pipeline starts with a pile of crumpled garments of various categories, colors and materials placed on a table. The pile is perceived with the RGBD sensor from above. The first step is the isolation and grasping of a single garment from the pile.

The depth map is rectified by fitting a plane to the table surface with the random sample consesus (RANSAC) [Fischler and Bolles, 1981] and subtracting it from the depth values. The rectified depths are filtered with the multi-scale and steerable kernels based on the difference of Gaussians (DoG) to detect bell shaped profiles [Koller et al., 1995], which correspond to wrinkles of various orientations and widths. The ridges of the wrinkles are detected with the non-maximum suppression of the filter responses. The high, narrow and isolated wrinkles are preferred for grasping.

The RGB image of the pile (Fig. 4.4a) is convolved with Gabor filters of multiple frequencies and orientations to extract the local feature vectors. The differences of their magnitudes are used as a dissimilarity measure in the graph-based segmentation algorithm [Felzenszwalb and Huttenlocher, 2004]. The resulting regions correspond either to individual garments or their parts due to the over-segmentation (Fig. 4.4b). The region highest above the table is selected for picking up. The best graspable ridge contained inside the region and not located close to its boundary is selected to avoid grasping multiple garments at once.

The gripper approaches the selected ridge from above, having its orientation aligned. The success of the grasping operation is checked by comparing two depth maps of the heap, taken before and after the picking up. If the average absolute difference is lower than a predefined threshold, the attempt is repeated with the second best grasping candidate. Neither possible slippage of the garment from the gripper during its lifting, nor grasping of multiple garments, as they have not been observed in the experiments.

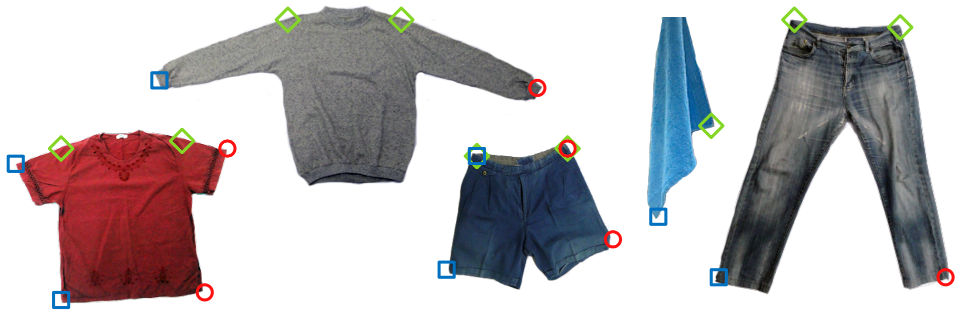


Figure 4.5. Blue squares denote the possible lowest hanging points for randomly grasped garment of a particular category. Red circles denote the symmetric lowest points. Green diamonds denote the desired points to be held after unfolding.

4.3. Classification and unfolding

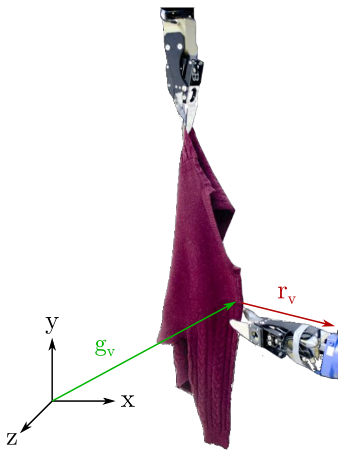
The picked up garment is classified according to its category at first. For each category, two points are defined on its surface for which the garment should be held in order to be unfolded (Fig. 4.5). The whole unfolding procedure is performed in the air, with the help of gravity. The garment is being regrasped repeatedly, grasping either the desired unfolding point directly in each step, or grasping such point that would facilitate the unfolding in future.

Prior to classification, the garment is regrasped for its lowest hanging point to reduce the space of its possible configurations significantly. The number of possible lowest points for each category of garments is very limited. Omitting the symmetries, there are only two such points for shorts and short-sleeved T-shirts, and one lowest point for towels, pants and long-sleeved shirts (Fig. 4.5).

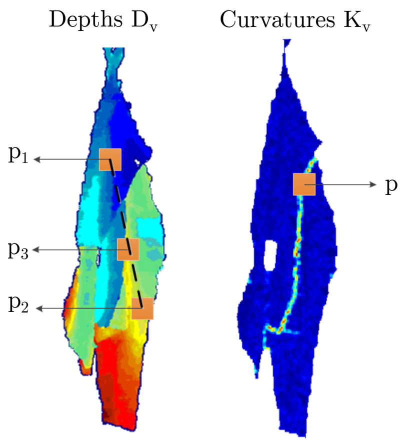
The category of the regrasped garment is recognized jointly with the currently held point, which was the lowest hanging point before regrasping. Thus the set of possible classes is the Cartesian product of garment categories and possible lowest points, without considering the symmetries (Fig. 4.5). The related task is the regression of the next grasping point and the direction it should be approached from by the gripper. They are represented as real 3D vectors. The classification and regression are solved in the novel framework of Active Random Forest (ARF) [Doumanoglou et al., 2014b]. It is also responsible for deciding whether and how much the garment should be rotated around its vertical axis at a certain time, which brings information from a new viewpoint.

The ARF framework is based on the classic Random Forest (RF) [Breiman, 2001], which is an ensemble of multiple decision trees. Each tree is trained on a randomly generated subset of the training data, which is known as bagging. The sampled data are being split recursively. A random subset of the features is generated for each node. The best split is selected among these candidates, optimizing a specified objective. When a defined criteria is met, the node becomes a leaf. The leaves predict the outcome variable based on the assigned training data. During inference, all trees are evaluated in parallel and the output is given by voting or averaging over the reached leaves. The trees in ARF contain action-selection nodes in addition to split and leaf nodes (Fig. 4.7). They serve for bringing more information into the process of samples splitting. They decide about rotating the garment and observing it from a new viewpoint.

Each training sample for ARF represents one particular instance of a hanging garment perceived from the particular viewpoint $v \in V$. The viewpoints from V are distributed in uniform discrete steps on a horizontal circle around the garment. They originate in



a) Grasping vectors



b) Pixel tests

Figure 4.6. a) Two 3D vectors are regressed by ARF: the position of the grasping point g_v and direction of the gripper r_v. b) The split nodes of ARF utilize simple thresholding tests based on depths and curvatures of individual pixels.

rotating the robotic wrist holding the hanging garment around the vertical axis, while perceiving it from a fixed location. The training sample is a tuple:

$$(D_v, c, g_v, r_v) \text{ for } v \in V, \tag{4.1}$$

where D_v denotes a depth map acquired from the viewpoint v. The class to be recognized c comprehends both the garment category and the originally lowest hanging point, now being held by the gripper. The next grasping point is represented by the 3D vector g_v, while r_v is a 3D vector aligned with the ideal direction to approach g_v with the gripper (Fig. 4.6a). It is oriented away from the garment surface. If the desired grasping point is not visible from the viewpoint v, then g_v and r_v are undefined.

Fig. 4.7 provides an overview of the ARF training. The input samples (4.1) are split to the training set S^{T} and validation set S^{D}. Each split node stores the set V_{seen} of already seen viewpoints, which is propagated to its children. The garment is initially perceived from a single viewpoint only, i.e. $V_{\text{seen}} \leftarrow \{v_{\text{current}}\}$.

The split nodes utilize very simple tests to control a flow of the training samples to their left and right child nodes. There are three types of functions (Fig. 4.6b):

$$f_1(v, p_1, p_2) = D_v(p_1) - D_v(p_2), \tag{4.2}$$
$$f_2(v, p_1, p_2, p_3) = (D_v(p_1) - D_v(p_3)) - (D_v(p_3) - D_v(p_2)), \tag{4.3}$$
$$f_3(v, p) = |K_v(p)|, \tag{4.4}$$

where $D_v(p)$ is a depth value in the pixel p seen from the viewpoint v, and $K_v(p)$ is a curvature in the pixel p estimated from the neighbors of p in the depth map D_v. A random set of functions is generated, each parametrized with a randomly selected viewpoint $v \in V$ and random pixel coordinates. The value of the function is compared to a randomly generated threshold.

The best test, which is used for splitting, is selected from the described random set with respect to the objective Q. It is equal to the classification objective Q_{class} in the upper part of the tree, where the class is unknown, and switches to the regression

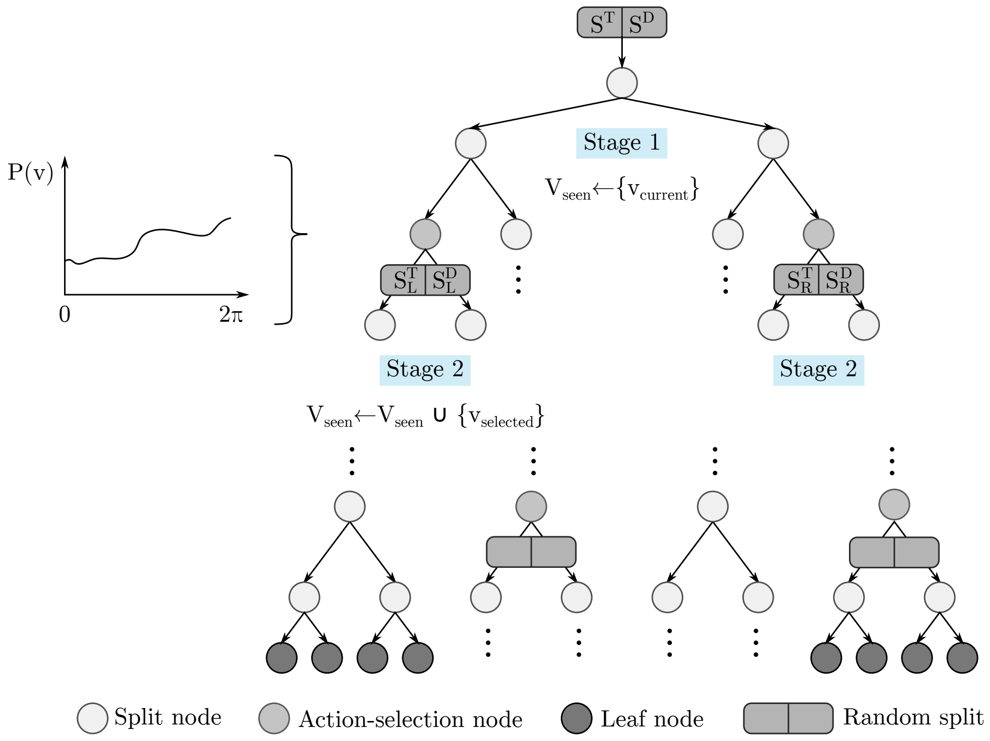


Figure 4.7. ARF training procedure. The tree contains three types of nodes: split, leaf and action selection. The training S^{T} and validation S^{D} data are being split recursively in the split nodes. The leaf nodes provide information about the garment category and next grasping point. The action selection nodes select the next best viewpoint from the distribution $P(v)$.

objective Q_{reg} deeper in the tree, where the class is already decided:

$$Q = \begin{cases} Q_{\text{class}} = -\sum_{n \in \{\mathrm{L},\mathrm{R}\}} \frac{|S^{\mathrm{T}}_n|}{|S^{\mathrm{T}}|} \sum_c P_n(c) \log_2 P_n(c), & \text{if } \max_c P(c) \le t, \\ Q_{\text{reg}} = -\sum_{n \in \{\mathrm{L},\mathrm{R}\}} \frac{|S^{\mathrm{T}}_n|}{|S^{\mathrm{T}}|} \log |\Sigma_n(q)|, & \text{otherwise.} \end{cases} \tag{4.5}$$

The probability distribution over the classes in the particular node is denoted P and thresholded with $t = 0.9$. The classification objective Q_{class} is a weighted sum of the categorical entropies from both child nodes $n \in \{\mathrm{L}, \mathrm{R}\}$. The regression objective Q_{reg} is based on the entropy of the regressed continuous variable q, which is either the grasping point g or the direction r. The regressed variable is selected randomly. Since g, r are assumed to be distributed normally, the entropy of q is proportional to the determinant of the covariance matrix $|\Sigma_n(q)|$ in the child node n.

The next best view selection is integrated to the tree. It improves the classification and regression accuracy, and enables detection of the grasping points hidden in the already seen views. The split functions (4.2–4.4) and objectives (4.5) use only the training samples S^{T} from the already seen viewpoints V_{seen}. This set becomes uninformative at certain depth and the tree starts to overfit. This is detected by evaluating the splits on the independent validation samples S^{D} and comparing the training and validation distributions using the Jeffrey divergence [Doumanoglou et al., 2014b]. If it exceeds a predefined threshold, the action-selection node is added. It extends the set of the already seen viewpoints V_{seen} with a new viewpoint v_{selected} (Fig. 4.7).

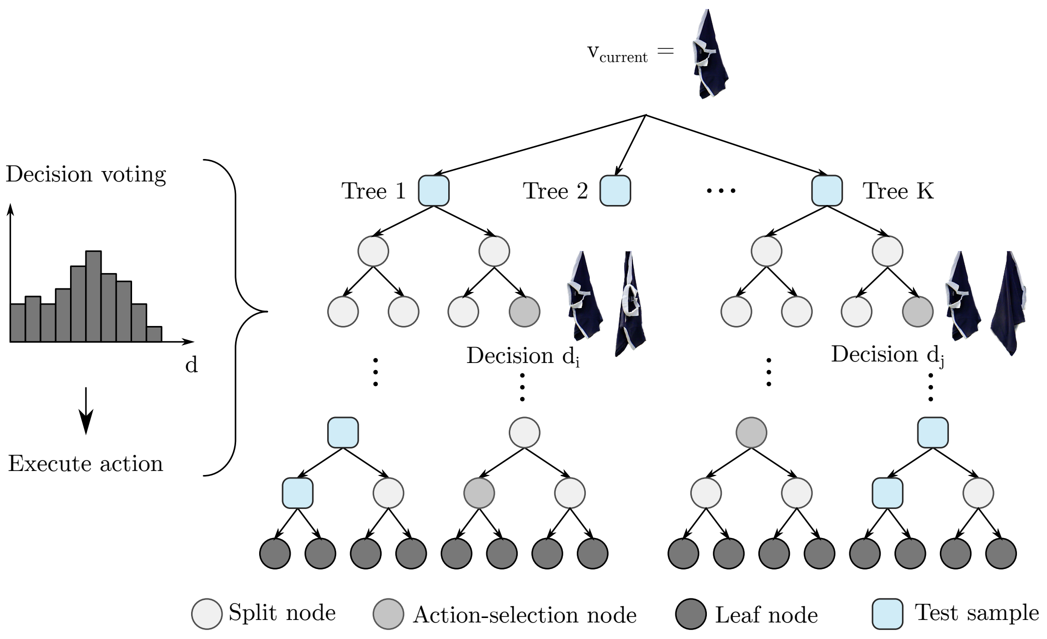


Figure 4.8. ARF inference procedure. The leaf nodes provide information about the garment category and next grasping point. A new viewpoint, found during the training procedure, is selected in each action-selection node, guiding the robot to rotate the garment.

The viewpoint $v \in V$ to be used in the pixel tests (4.2–4.4) is sampled with the probability $P(v)$. The distribution $P(v)$ is constructed to prefer already seen viewpoints. The unseen viewpoints are sampled with the probability decreasing exponentially with the increasing rotation angle needed to reach them. In addition, the distribution prefers such viewpoints, from which the grasping point is visible, i.e. located on the side facing the sensor and not being occluded by wrinkles.

Fig. 4.8 provides an overview of the ARF inference. It starts with a single view of the hanging garment. The trees are traversed from their roots. The inference procedure reaches leaf nodes in some trees, whereas it ends up in the action-selection nodes in other trees, where the next view is required to continue the traversal. The action-selection nodes from different trees vote about the next viewpoint. The trees that voted for the selected viewpoint can be further traversed using the new viewpoint, while the remaining trees keep their votes for the next voting. The parallel traversal stops once a predefined number of leaves is reached. The class distributions from the reached leaves are averaged and the most probable class is selected. The grasping point g and grasping direction r are decided by Hough voting [Doumanoglou et al., 2014b].

4.4. Spreading

The unfolded garment is held hanging at two predefined locations, which are specific for the particular category of garments (Fig. 4.5). The garment is then pulled over the edge of a table and laid down on it, which usually results in a partially deformed configuration (Fig. 4.9). This can be improved by spreading, which is applied specifically for the short-sleeved T-shirts in our pipeline. One arm is pressing the T-shirt towards the table to prevent it from sliding, while the other is sweeping it repeatedly with a small brush attached to its gripper (Fig. 4.9).

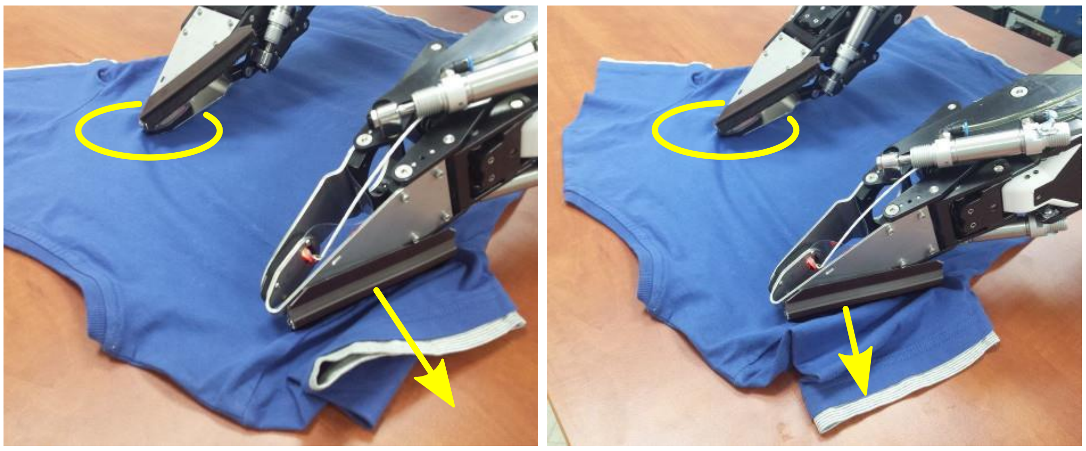
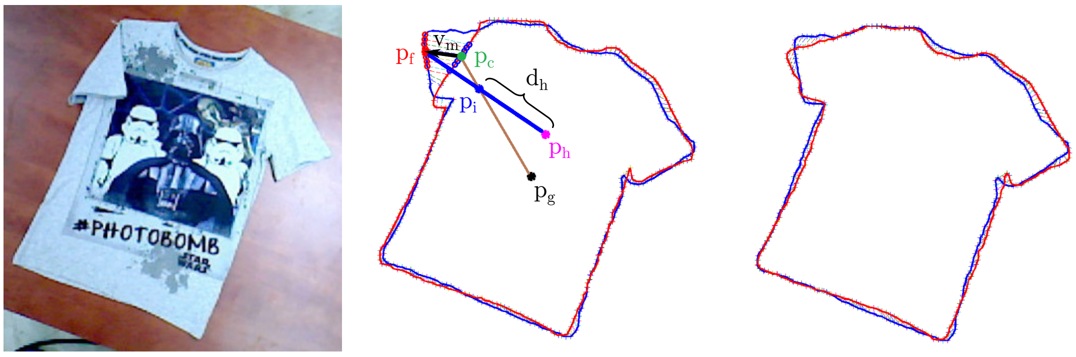


Figure 4.9. The brush tool attached to the gripper is moved in the direction shown by the arrow. The other arm is holding the T-shirt to prevent it from sliding.



a) Deformed T-shirt **b)** Deformed contour **c)** Spread contour

Figure 4.10. a) The unfolded T-shirt is deformed. b) The deformed contour (red) is matched to the spread template (blue). The spreading gripper moves from p_i to p_f, while the other gripper is holding the T-shirt at p_h. c) No deformation is present after the spreading.

The contour of the deformed T-shirt (Fig. 4.10a) is extracted and matched to the spread template using the inner distance shape contexts [Ling and Jacobs, 2007]. The pairs of matching points determine a global similarity transformation, comprehending translation, rotation and scaling. The transformation is used for registering the deformed contour to the template (Fig. 4.10b). The deformations are defined as such continuous subsets of the contour, for which the distances to the registered template exceed a predefined threshold. The deformation with the largest total distance is selected for spreading. The central point of the largest deformation is denoted p_c. The displacement vectors between the deformation and registered template are averaged into the vector v_m, which is used to compute the spreading direction.

The spreading operation is defined by tree points (Fig. 4.10b): p_i and p_f, denoting the initial and final position of the brush, and p_h, denoting the position of the holding gripper, which is pressing the T-shirt towards the table to prevent it from sliding. All three points are located on a single line in the image plane. Locations of p_i, p_f and p_h are computed from the described mean displacement vector v_m, central point of the deformation p_c and the center of gravity p_g of the garment as follows:

$$p_\mathrm{i} = p_\mathrm{c} + \|v_\mathrm{m}\| \frac{p_\mathrm{g} - p_\mathrm{c}}{\|p_\mathrm{g} - p_\mathrm{c}\|}, \tag{4.6}$$

$$p_\mathrm{f} = v_\mathrm{m} + p_\mathrm{c}, \tag{4.7}$$

$$p_\mathrm{h} = p_\mathrm{i} + d_\mathrm{h} \frac{p_\mathrm{i} - p_\mathrm{f}}{\|p_\mathrm{i} - p_\mathrm{f}\|}. \tag{4.8}$$

The initial distance d_h of the spreading and holding gripper is chosen large enough to prevent the robotic arms from colliding.

The estimated points p_i, p_f and p_h in the image plane determine the positions of the robotic arms in the world coordinates. The orientation of the brush is approximately perpendicular to the spreading direction $\overrightarrow{p_\mathrm{i}p_\mathrm{f}}$. A few degrees deviation is allowed if the inverse kinematics has no solution for the strictly perpendicular orientation.

4.5. Folding

The unfolded and optionally spread garment is folded with our method introduced in Chap. 3. The extracted and simplified contour of the spread garment is matched the polygonal model for the corresponding category, which is known from the classification and unfolding phase (Sec. 4.3). The final pipeline utilizes specifically the global matching method to fit the polygonal model (Sec. 3.6).

Despite that the approximate position, orientation and configuration of the spread garment are known from the previous step, they are not used to simplify the matching. The model is fitted from scratch. The main reason is that the individual methods included in the pipeline were developed independently and integrated later. In principle, it would be possible to map each model vertex only to such contour points that are close to the approximately known location of the corresponding landmark. This restricted matching could be also used for checking the pose of a partially folded garment.

4.6. Experiments

This section provides the experimental evaluation of the modules included in the pipeline (Sec. 4.2–4.4), as well as the end-to-end robotic experiments of the whole pipeline. The presented experiments were performed at the Center for Research of Technology Hellas. The pipeline was also deployed to the testbed at the Czech Technical University in Prague and used as a robotic manipulation demo occasionally.

The grasping module (Sec. 4.2) was evaluated in the real robotic experiment. The robot was isolating a single item from a heap repeatedly. There were 80 runs performed. The garments in the heap were changed and shuffled manually every 10 runs. The run was considered successful if only a single garment was grasped and lifted up without affecting the heap significantly. The maximum number of 3 retries was allowed. The overall success rate was 95 %. The failures were caused by the incorrect localization of the grasping point and due to slippage of the garment during the gripper closing.

The classification of the hanging garment category, regression of the next grasping point and regression of the corresponding gripper orientation (Sec. 4.2) were evaluated on the dataset of 30 garments. It includes 6 unique garments for each of 5 categories: towels, shorts, pants, short-sleeved T-shirts and long-sleeved shirts. There are 72,000 training and 600 testing samples. Each sample is the annotated depth map seen from a

Task	Towels	Shorts	Pants	T-shirts	Shirts	Overall
Classification [%]	85	100	100	91	98	95
Grasping point [%]	93	100	100	86	96	95
Orientation [%]	83	81	94	71	67	79

Table 4.1. Percentual success rates for the classification of the garment category and regression of the next grasping point and corresponding gripper orientation vectors. The allowed error of the estimate is 10 cm for the grasping point and 18° for the gripper orientation.

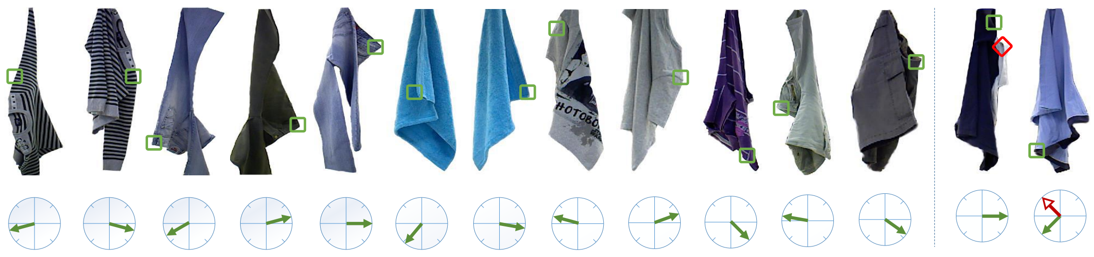


Figure 4.11. Qualitative results for the regression of the grasping point and corresponding gripper orientation. Two failures shown on the right are due to wrong grasping point detection (red diamond) and wrong orientation estimation (hollow red arrow).

particular viewpoint, as defined in (4.1). The detection of the grasping point is considered successful, if the distance to the ground truth is lower than 10 cm. The divergence of 18° is allowed for the estimated orientation of the grasping gripper. Tab. 4.1 summarizes the success rates for the individual garment categories. Fig. 4.11 shows the selected qualitative results for the regression tasks, including two failures.

The spreading module (Sec. 4.2) was evaluated in the real robotic experiment, using two T-shirts. Each T-shirt was placed on the table 15 times by pulling it over the edge, while the other T-shirt was used as a spread template for the registration and detection of deformations. The spreading was considered successful if all landmark points were recognized correctly by the polygonal model matching (Sec. 4.5). No configuration satisfied the above criterion before spreading, whereas 25 out of 30 configurations did afterwards, which gives 83 % success rate. The maximum distance of the deformed contour from the template decreased from 14.3 cm to 3.7 cm on average by the spreading. There were 2.8 spreading actions needed on average in each run.

The performance of the complete pipeline[1] was evaluated on 3 categories of garments: towels, shorts and short-sleeved T-shirts. Pants and long-sleeved shirts were not used because the limited workspace of the robot does not allow to manipulate large garments. Each category was represented by 4 unique items (Fig. 4.12). Each item was evaluated 8 times, starting from randomly tossed configurations. Fig. 4.2 shows an example run.

The complete pipeline was successful in 72 out of 96 runs. The overall success rate is therefore 79 %. It can be understood as a product of the success rates for the individual steps, since a failure in any step causes the whole pipeline to fail. The experiment was overlooked by a human operator, who interrupted the execution in case of a failure. In some cases, it would be also possible to restart the whole pipeline or at least some steps (Fig. 4.1) to possibly improve the overall success rate.

The following failures were observed in the experiment: a towel was misclassified as a T-shirt 7 times, wrong grasping points were detected for shorts 2 times and a T-shirt

[1]Videos of the pipeline: `http://cmp.felk.cvut.cz/~striajan/phd`

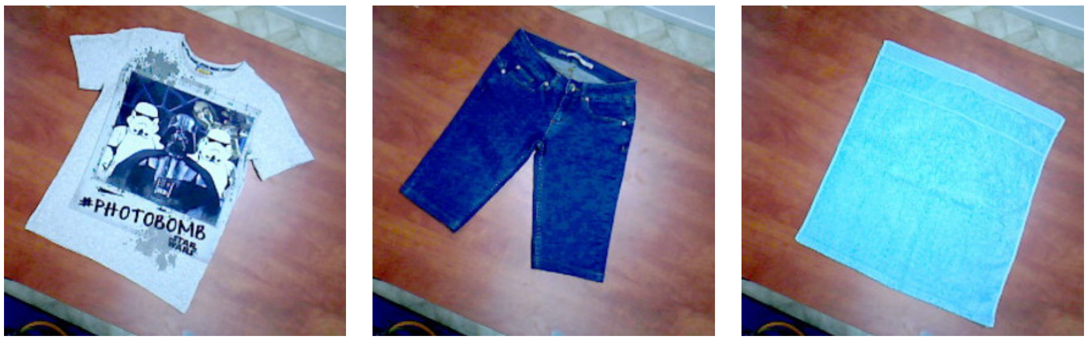


Figure 4.12. Examples of garments used for the evaluation of the complete pipeline.

	Shorts	T-shirts	Towels	Overall
Successful / all trials count	30 / 32	21 / 32	25 / 32	76 / 96
Success rate [%]	94	66	78	79

Table 4.2. Overall results of experiments testing the complete pipeline including grasping, category recognition, unfolding, spreading and folding.

was not spread 2 times. The folding module failed 9 times, out of which 2 failures were caused by the pose estimation method and 7 failures by the unsuccessful planning of the folding movement, mainly due to the limited workspace of the robot. Tab. 4.2 provides a summary of results for individual garment categories. The most failures were observed for T-shirts, which have the most complex shape and are usually manufactured from the most deformable materials (Fig. 4.12).

The overall success rate of 79 % sets a baseline for the folding scenario. To the best of our knowledge, it is the first and up to this day the only end-to-end pipeline that can deal with various categories of garments. For comparison, [Maitin-Shepard et al., 2010] report a nearly perfect success rate, but with a solution developed specifically and exclusively for towels.

The execution of the pipeline takes 8 minutes on average, depending on the garment category. Most of the time is spent by the robotic manipulation, while all perception and reasoning takes under 30 seconds. For comparison, [Maitin-Shepard et al., 2010] report approximately 20 minutes for processing a single towel. The relative slowness of our pipeline, with respect to the usual expectations of the general public and industry, is caused by the robot operating in moderate speed due to the safety reasons.

The detection of the initial grasping point (Sec. 4.2) takes approximately 1 second, whereas the actual grasping and lifting takes 20 seconds. A single depth map of the hanging garment is analyzed in 30–40 milliseconds by ARF (Sec. 4.3). Depth maps from 5 different viewpoints are required on average. The whole process of unfolding the garment by grasping two predefined points on its surface takes slightly over 2 minutes. The analysis of the deformed contour (Sec. 4.4) takes approximately 10 seconds, while the spreading using a brush lasts around 50 seconds. The spreading procedure is executed at most 3 times. The pose estimation of the spread garment (Sec. 4.5) takes 2–5 seconds, depending on the garment type. A single folding operation is performed approximately in 30 seconds. There are two folds needed for towels and shorts, and 3 folds for T-shirts.

5. Unfolding folded garments

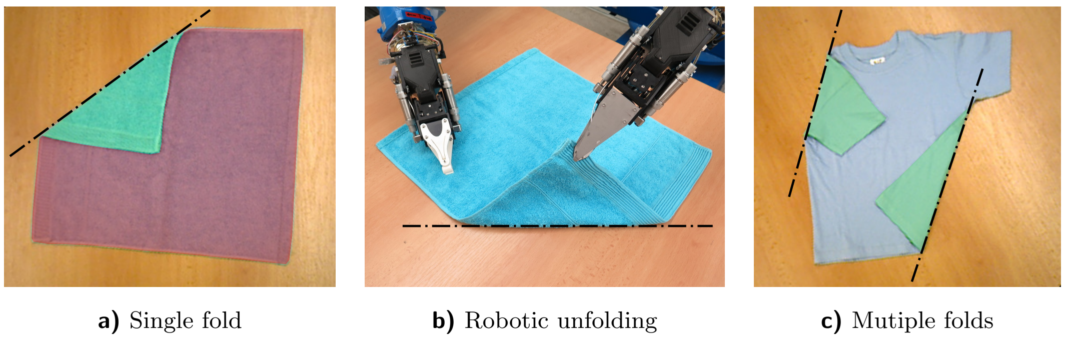
We propose a method for the robotic unfolding of an unknown garment that was placed flat on a table and folded over an unknown axis [Stria et al., 2017]. The algorithm combines image and depth data to detect the bottom and top layer of the garment, where the top layer denotes the folded part of the garment. The layers detection is formulated as a labeling of the garment surface and solved in an energy minimization framework. Parameters of the energy function are estimated automatically from perceived data.

Once the garment pose is known, several candidate folding axes are generated and used to unfold the garment virtually. The shapes of the virtually unfolded garment are analyzed to select the true folding axis among the candidates. The method does not set any constraints on the garment shape. It is able to deal with various categories of garments, including jackets, pants, shorts, skirts or T-shirts of any sleeve lengths, as shown in the experimental evaluation.

The garment is unfolded by our dual-arm robot, utilizing a cooperated manipulation of both arms. One arm grasps the boundary of the detected top folded layer and brings it over the estimated folding axis. Meanwhile, the second arm is pressing the bottom layer towards the table to prevent the garment from slipping.

5.1. Task formulation and motivation

The presented method deals with the visual analysis and robotic unfolding of a garment laid on a table. The garment category is unknown and not utilized by our method. It is assumed that the garment is posed in such a configuration that can be achieved by grasping the boundary of the fully spread garment and folding it over a selected folding axis (Fig. 5.1a). The location and orientation of the axis is unknown and must be estimated by our method. We use the fact that the axis forms a straight line segment on the outer boundary of the garment.



a) Single fold **b)** Robotic unfolding **c)** Mutiple folds

Figure 5.1. a) The initial configuration of a garment folded over the axis (dashed line), which forms a straight line segment on the boundary. The folded garment consists of the top (green) and bottom (red) layer. b) The top layer is grasped and unfolded over the axis, while the bottom layer is pressed towards the table to prevent the garment from slipping. c) Multiple folds are allowed, if the top layers (green) do not overlap.

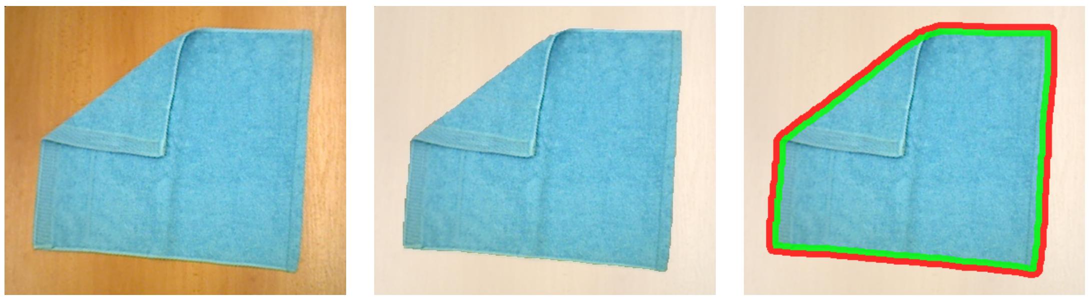


Figure 5.2. a) The input image is b) segmented. c) The segmentation mask P is eroded to a subset mask P_e (removed green strip) and dilated to a superset mask P_d (added red strip).

Let us denote the folded part a top layer and the remaining one a bottom layer. The bottom layer lies directly on a table, while the top part is placed on top of it (Fig. 5.1a). Our method assumes that the top layer does not cover the bottom one completely. Note that the majority of garments consist of a front and back side sewn together, i.e. two layers of material, but we consider them a single layer for our purposes. The garment is allowed to be folded multiple times, if no two top folded layers overlap (Fig. 5.1c).

The goal is to detect the top layer, estimate the folding axis and unfold the garment by bringing the top layer back over the axis (Fig. 5.1b). If there are more top folded layers, they are being detected and unfolded one by one.

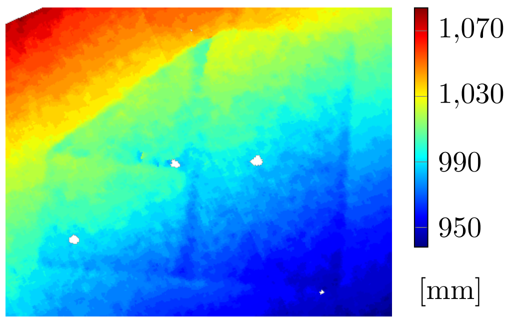
Although the task definition may seem rather artificial, it is motivated by two possible scenarios. First, it can replace grasping, classification and unfolding of a hanging garment in the folding pipeline, if its initial configuration satisfies the assumptions. This was first proposed by [Mariolis and Malassiotis, 2015] and [Estevez et al., 2016] (Sec. 2.5). Another motivation is the two-stage unfolding procedure for completely crumpled garments, proposed by [Triantafyllou et al., 2016] (Sec. 2.4). The hanging garment is untangled and stretched out in the air, laid down on a table and the remaining folds are removed, using the procedure analogous to ours.

5.2. Image and depth preprocessing

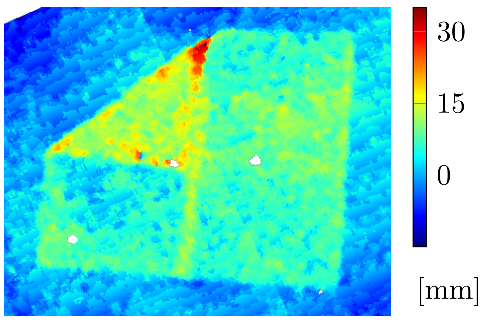
The input of our method for folded layers detection is formed by the RGB image I and depth map D. They are acquired by the RGBD sensor (Sec. 4.1). The image and depth data are equally sized and calibrated. The color $I(p)$ and depth $D(p)$ of the pixel p therefore refer to the same location in the real world, enabling to combine information from both modalities effectively.

The image I is used to segment the observed garment from its background, which is a wooden table in our experiments (Fig. 5.2a). The two-phase segmentation method introduced in Sec. 3.1 is used. It is assumed that the background differs from the garment and does not change in the experiments. A probabilistic model of the background color is learned, which provides the coarse and incomplete segmentation. It is used to initialize the GrabCut [Rother et al., 2004] algorithm automatically, which provides the final smooth segmentation mask (Fig. 5.2b).

Let us denote P the set of pixels from the final mask that correspond to the garment. The mask is morphologically eroded and dilated with a disc structuring element [Šonka et al., 2014], radius of which corresponds approximately to 2 cm. Let us denote



a) Depths acquired by RGBD sensor



b) Normalized heights above table

Figure 5.3. a) Depths acquired by the range sensor are subtracted from the plane approximating the table surface to obtain b) heights above the table. The depth and height values are missing in the white areas.

the eroded and dilated pixels P_e and P_d, respectively (Fig. 5.2c). It holds $P_\mathrm{e} \subset P \subset P_\mathrm{d}$. The eroded mask enables to avoid noisy image and depth values near the garment boundary. The dilated mask enables selection of the surrounding table pixels.

The depth map D is acquired by the RGBD sensor attached to the wrist of the robot (Sec. 4.1). Although the arm points approximately downwards during the acquisition, the view direction is not perfectly perpendicular to the table surface, which appears skewed in D. Another challenge are the missing depth values, caused mainly by the occlusions and limitations of the structured light technology on glossy surfaces.

The contour of the dilated segmentation mask P_d is extracted. If the segmentation was successful, the contour pixels belong to the table, as they are located approximately 2 cm outwards from the garment boundary. The depth values from the contour pixels are used to estimate a plane approximating the table surface. Its parameters are fit with the random sample consesus (RANSAC) [Fischler and Bolles, 1981]. The input depth map D (Fig. 5.3a) is then subtracted from the estimated table plane. This results in the height map H, where $H(p)$ is a height of the pixel p above the table (Fig. 5.3b).

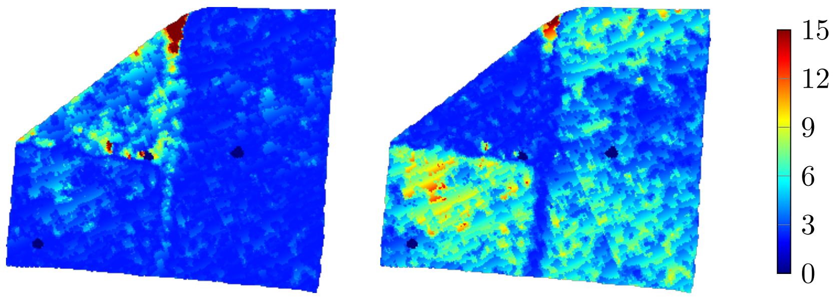
5.3. Layers detection

We formulate the task of detecting the top and bottom layer as a task of finding the optimum labeling for a visible surface of the garment. Each garment pixel $p \in P$ needs to be assigned a label $z_p \in \{T, B\}$. The label T represents the top layer and B the bottom layer. Construction of the optimum labeling is based on two assumptions:

- Boundaries between the top and bottom layer appear as edges in the image I.
- Pixels from the top folded layer appear higher above the table in the height map H than pixels from the bottom layer.

Since the outer contour of the garment is a source of undesirable edges in the input image I, only the inner eroded pixels P_e (Fig. 5.2c) are labeled at first. The labeling is then extrapolated to the boundary pixels of the original full segmentation mask P. The task of labeling the pixels $p \in P_\mathrm{e}$ optimally with the labels $z_p \in \{T, B\}$ is formulated as the following energy minimization problem, which is a standard and common formulation in many computer vision applications [Boykov and Kolmogorov, 2004]:

$$Z^* = \operatorname*{arg\,min}_{Z \in \{T,B\}^{|P_\mathrm{e}|}} \sum_{p \in P_\mathrm{e}} U_p(z_p) + \sum_{\{p,q\} \in N_\mathrm{e}} V_{p,q}(z_p, z_q). \tag{5.1}$$



a) Bottom potentials $U_p(B)$ **b)** Top potentials $U_p(T)$

Figure 5.4. Unary potentials of pixels being assigned to the bottom and top layer of the folded garment, i.e. the potentials $U_p(B)$ and $U_p(T)$ for each $p \in P_e$.

The functions $U_p\colon \{T, B\} \to \mathbb{R}$ are called unary potentials. They express a cost of assigning each pixel $p \in P_\mathrm{e}$ to the particular layer. The functions $V_{p,q}\colon \{T, B\}^2 \to \mathbb{R}$ are pairwise potentials. They are defined for all pairs of neighboring pixels $\{p, q\} \in N_\mathrm{e}$, where $p, q \in P_\mathrm{e}$. The pairwise costs align the boundary between the labeled layers with the edges observed in the image. They also make the boundary smooth.

The unary potential $U_p(z)$ for the pixel $p \in P_\mathrm{e}$ and label $z \in \{T, B\}$ is defined as:

$$U_p(z) = \begin{cases} -\log \mathcal{N}\big(H(p); \mu_z, \sigma^2\big), & H(p) \text{ is known,} \\ 0, & \text{otherwise.} \end{cases} \tag{5.2}$$

The definition is based on the assumption of normally distributed heights of the pixels belonging to a particular layer. The potentials are the negative log-likelihoods of two normal distributions $\mathcal{N}\big(\mu_T, \sigma^2\big)$ and $\mathcal{N}\big(\mu_B, \sigma^2\big)$. The mean height of the top and bottom layer is μ_T and μ_B, respectively. The variance σ^2, shared by both distributions, is caused by the wrinkled surface of a garment and by the noise present in the input depths. Sec. 5.4 explains how μ_T, μ_B and σ^2 are estimated from the observed heights.

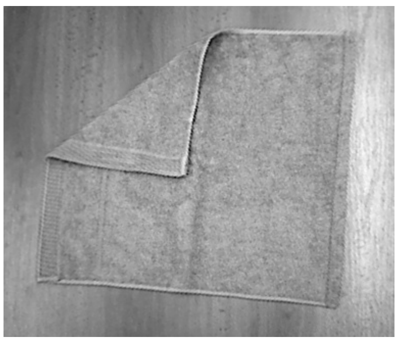
Fig. 5.4 shows an example of the unary potentials computed from Fig. 5.3b. The definition of unary potentials was inspired by [Rother et al., 2004], who use distributions of foreground and background colors, instead of heights, for the color-based segmentation of RGB images.

The pairwise potentials $V_{p,q}(z_p, z_q)$ are defined for all pairs of neighboring pixels $\{p, q\} \in N_\mathrm{e}$, where $p, q \in P_\mathrm{e}$. We use 8-connected neighborhood, i.e. all pairs of vertically, horizontally or diagonally adjacent pixels. The potentials are defined similarly to [Rother et al., 2004]:

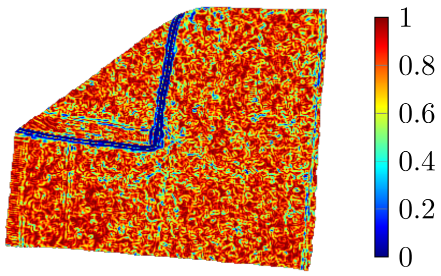
$$V_{p,q}(z_p, z_q) = \begin{cases} \gamma_1 + \frac{\gamma_2}{d(p,q)} \exp\left(-\frac{g(I,p,q)}{2E[g]}\right), & z_p \neq z_q, \\ 0, & z_p = z_q. \end{cases} \tag{5.3}$$

The term $d(p, q)$ denotes the spatial distance of the pixels p and q in the image grid. It is equal to 1 for the horizontally and vertically adjacent pixels. It equals $\sqrt{2}$ for the diagonal neighbors. The function g evaluates the visual difference of the pixels p and q in the image I. The term $E[g]$ denotes the mean value of the function g over all pairs of the neighboring pixels:

$$E[g] = \frac{1}{|N_\mathrm{e}|} \sum_{\{p',q'\} \in N_\mathrm{e}} g(I, p', q'). \tag{5.4}$$



a) Grayscale image



b) Normalized transformed gradients

Figure 5.5. a) The gray-scale image I_{gray} used for computing pairwise potentials. b) The gray-scale image is smoothed and the gradient magnitudes I_{grad} are estimated. The figure shows the normalized and exponentiated gradients $\exp\left(-I_{\text{grad}}/E[I_{\text{grad}}]\right)$, similarly to (5.3).

We set $\gamma_2 = 50$ as in [Rother et al., 2004] to balance the typical values of the unary and pairwise potentials. In addition to [Rother et al., 2004], the potentials are extended with the Ising prior term, represented by the constant $\gamma_1 = 1$. It prefers labelings with a shorter boundary between the top and bottom layer.

The labeling procedure is not sensitive to the exact choice of the function g. It can be almost an arbitrary function that responds to the presence of edges between the pairs of pixels. The input RGB image I is transformed to a gray-scale image I_{gray} at first (Fig. 5.5a), which is then smoothed with the bilateral filter [Gonzalez et al., 2009] to reduce a noise, while preserving edges:

$$I_{\text{smooth}}(p) = \frac{\sum_q w(p,q) I_{\text{gray}}(q)}{\sum_q w(p,q)}, \tag{5.5}$$

$$w(p,q) = \begin{cases} \exp\left(-\frac{||p-q||_2^2}{2\sigma_{\text{d}}^2} - \frac{||I_{\text{gray}}(p)-I_{\text{gray}}(q)||_2^2}{2\sigma_{\text{r}}^2}\right), & ||p-q||_\infty \leq d_{\text{w}}, \\ 0, & \text{otherwise}. \end{cases} \tag{5.6}$$

It computes a weighted average of the neighboring pixel values for each pixel $p \in P_{\text{e}}$ over the window of size $(2d_{\text{w}}+1) \times (2d_{\text{w}}+1)$ centered at the pixel p, where $d_{\text{w}} = 2$. The weights combine differences of neighboring pixel coordinates and values. The constants were set empirically as $\sigma_{\text{d}} = 2$ and $\sigma_{\text{r}} = 1$ for gray-scale values from $[0, 1]$.

The smoothed image I_{smooth} is convolved with the horizontal and vertical Sobel filters G_{x} and G_{y} [Gonzalez et al., 2009] to estimate the partial derivatives along the horizontal and vertical axis. The gradient image I_{grad} contains squared norms of the gradient:

$$I_{\text{grad}} = (I_{\text{smooth}} * G_{\text{x}})^2 + (I_{\text{smooth}} * G_{\text{y}})^2. \tag{5.7}$$

The magnitudes are averaged for pairs of neighboring pixels to construct g (Fig. 5.5b):

$$g(I,p,q) = \frac{I_{\text{grad}}(p) + I_{\text{grad}}(q)}{2}. \tag{5.8}$$

The pairwise potentials, as defined in (5.3), are regular by the definition introduced by [Kolmogorov and Zabin, 2004], because for each $\{p,q\} \in N_{\text{e}}$ it holds:

$$0 = V_{p,q}(T,T) + V_{p,q}(B,B) \leq V_{p,q}(T,B) + V_{p,q}(B,T). \tag{5.9}$$

Therefore, the globally optimum labeling Z^* with respect to (5.1) can be found effectively by constructing a special weighted graph and constructing the maximum flow

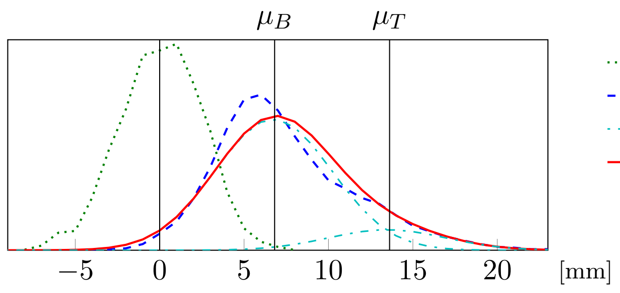


Figure 5.6. Normalized histograms of table heights and garment heights. The garment heights are fitted GMM with two components by EM algorithm to estimate μ_T, μ_B and σ^2.

from a source to a sink vertex [Boykov and Kolmogorov, 2004]. The source and sink vertices represent the labels $\{T, B\}$. They are connected to a 2D grid of 8-connected vertices representing individual pixels. The maximum flow in the graph corresponds to the minimum cut, i.e. to the subset of edges separating the source from the sink, and it determines the optimum labeling.

5.4. EM algorithm for heights estimation

We show how to estimate the expected height of the top μ_T and bottom μ_B layer above the table as well as their variance σ^2. They are used to compute the unary potentials (5.2). We model the heights of pixels from P_e using the Gaussian mixture model (GMM) with two components, each of them corresponding to a single layer. The components are weighted by priors π_T, π_B that are also unknown. Let us denote $\theta = (\mu_T, \mu_B, \sigma^2, \pi_T, \pi_B)$ all unknown parameters of the mixture. The likelihood of the pixel p having the height $H(p)$ is given by:

$$\mathcal{L}(\theta; H(p)) = \sum_{z\in\{T,B\}} \pi_z\, \mathcal{N}\big(H(p); \mu_z, \sigma^2\big). \tag{5.10}$$

The unknown parameters θ are estimated by the expectation-maximization algorithm (EM) [Dempster et al., 1977]. It is based on the incremental refinement $\theta^{(t)}$ of their initial estimate $\theta^{(0)}$ for $t = 1, 2, \ldots$ The EM algorithm alters between the expectation and maximization steps, while increasing the lower bound on the likelihood (5.10). Our algorithm is similar to estimation of the standard GMM with independent components [Bilmes et al., 1998]. However, the components are not independent in our case. They share the variance σ^2. Moreover, denoting Δ_μ the unknown thickness of one layer of the garment, the mean heights are linked as follows:

$$\mu_B = \Delta_\mu, \tag{5.11}$$

$$\mu_T = 2\Delta_\mu. \tag{5.12}$$

The initial values of the parameters $\theta^{(0)}$ are estimated by applying the k-means algorithm [Duda et al., 2000] on the heights of eroded pixels P_e. Each layer is represented by a single mean, i.e. $k = 2$. The k-means result in a cluster of pixels from the bottom and a cluster of top pixels. The thickness $\Delta_\mu^{(0)}$ is computed based on (5.11–5.12) from the heights corresponding to the centers of clusters. The priors $\pi_T^{(0)}$ and $\pi_B^{(0)}$ are initialized to the relative sizes of clusters and $\sigma^{(0)}$ is the sample standard deviation.

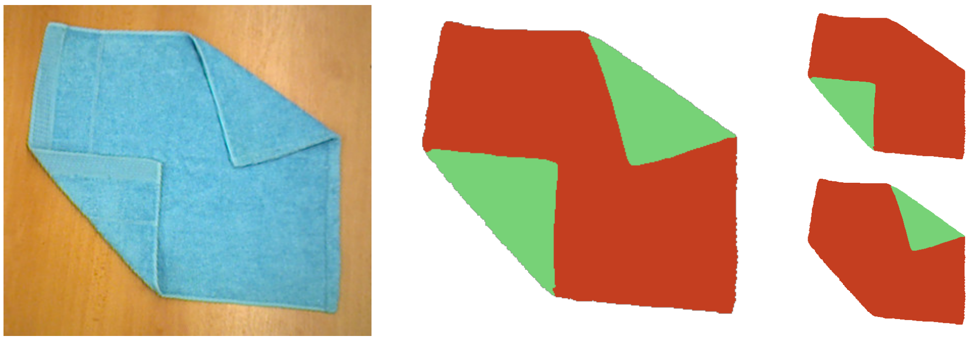


Figure 5.7. a) Twice folded garment with top layers not overlapping. b) Result of layers detection. Both top folded layers (green) were detected. c) Two modified labelings where only a single top layer is forced.

In the estimation (E) step, a variational distribution $Q_p^{(t)}(z)$ is built for each pixel $p \in P_\mathrm{e}$ using current values of the parameters $\theta^{(t)} = (\mu_T^{(t)}, \mu_B^{(t)}, (\sigma^{(t)})^2, \pi_T^{(t)}, \pi_B^{(t)})$:

$$Q_p^{(t)}(z) = \frac{\pi_z^{(t)} \mathcal{N}\Big(H(p); \mu_z^{(t)}, (\sigma^{(t)})^2\Big)}{\sum_{z' \in \{T,B\}} \pi_{z'}^{(t)} \mathcal{N}\Big(H(p); \mu_{z'}^{(t)}, (\sigma^{(t)})^2\Big)}. \tag{5.13}$$

In the maximization (M) step, the variational distributions $Q_p^{(t)}(z)$ are used to estimate the new values of parameters $\theta^{(t+1)}$. The thickness of the layer $\Delta_\mu^{(t+1)}$ is estimated at first:

$$\Delta_\mu^{(t+1)} = \frac{\sum_{p \in P_\mathrm{e}} \Big(2Q_p^{(t)}(T) + Q_p^{(t)}(B)\Big) H(p)}{\sum_{p \in P_\mathrm{e}} 4Q_p^{(t)}(T) + Q_p^{(t)}(B)}. \tag{5.14}$$

Using (5.11–5.12) to compute $\mu_T^{(t+1)}$, $\mu_B^{(t+1)}$ from (5.14), the variance and priors are estimated as in the standard EM for GMM [Bilmes et al., 1998]:

$$(\sigma^{(t+1)})^2 = \frac{1}{|P_\mathrm{e}|} \sum_{p \in P_\mathrm{e}} \sum_{z \in \{T,B\}} Q_p^{(t)}(z) \Big(H(p) - \mu_z^{(t+1)}\Big)^2, \tag{5.15}$$

$$\pi_z^{(t+1)} = \frac{1}{|P_\mathrm{e}|} \sum_{p \in P_\mathrm{e}} Q_p^{(t)}(z). \tag{5.16}$$

The EM algorithm converges to a local optimum [Dempster et al., 1977], which takes 30–50 iterations in our case. Fig. 5.6 shows an example of the final GMM estimated from the height map H shown in Fig. 5.3b.

5.5. Folding axis detection

Once the layers detection is finished, each pixel $p \in P$ is assigned either to the top or bottom layer. As stated in Sec. 5.1, the garment is allowed to be folded multiple times, as long as the top layers do not overlap (Fig. 5.7a). Each top folded layer then forms a separate connected component in the computed labeling (Fig. 5.7b). They are unfolded one by one and in a greedy manner.

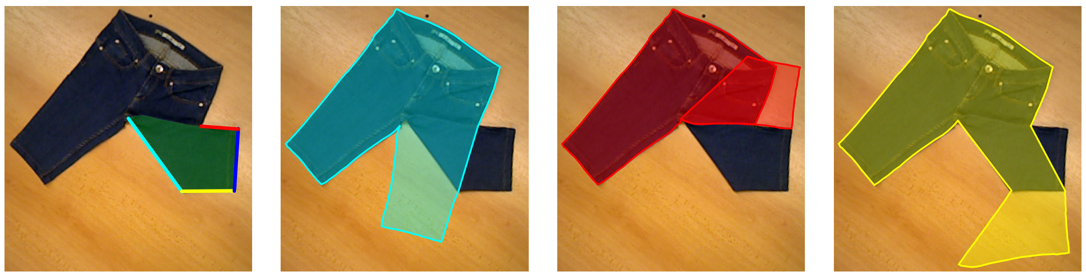


Figure 5.8. a) Outer contour of the top layer (green) approximated by 4 candidates folding axes (various colors). b) Unfolding over the true axis. c) Unfolding over the false candidate axis leading to an self overlapping unfolded contour. d) Unfolding over the false candidate axis that leads to a complex unfolded contour.

A separate modified labeling is constructed for each connected component, where only the pixels from that component belong to the top layer, while all other pixels are hard assigned to the bottom layer (Fig. 5.7c). The cost of each modified labeling is evaluated with respect to the energy function (5.1). The component with the minimum cost is selected for unfolding.

As stated in Sec. 5.1, the folding axis forms an approximate segment on the outer contour of the top layer (Fig. 5.1a). In order to estimate the folding axis, the outer contour of the top layer is approximated by a polyline [Perez and Vidal, 1994]. The algorithm approximates the outer contour with a single segment at first and then splits it recursively at the furthest point on the contour. The resulting polyline can consist of one or more segments. Each segment forms a candidate for the axis (Fig. 5.8a), which is estimated by fitting a line to the contour points adjacent to the segment with the robust M-estimator [Huber, 1973].

The detected top layer is unfolded virtually by reflecting it over each candidate axis (Fig. 5.8a). The candidate axes, for which the reflected top layer overlaps the bottom layer, are rejected from further processing (Fig. 5.8c). The unfolded contours for the remaining candidates are analyzed and compared. Since the true shape of the garment below the top layer is not known, the contour of the bottom layer is simply connected to the contour of the unfolded top layer with two straight segments. The candidate axis giving the shortest unfolded contour is chosen as the true folding axis (Fig. 5.8b). We use this simple heuristic successfully to reject very complex invalid contours (Fig. 5.8d).

5.6. Robotic unfolding

The robotic unfolding uses two cooperated arms. The first arm unfolds the selected top layer, while the other one holds the bottom layer to prevent the garment from slipping. The first arm grasps the inner boundary of the top layer and pulls it back over the estimated folding axis. The gripper follows a triangular unfolding path, which is analogous to the triangular folding path introduced in Sec. 3.8 and shown in Fig. 3.10, but it is oriented reversely. The triangular shape of the path helps to immobilize the hanging part of the garment due to the gravity. Similarly to Sec. 3.8, we do not grasp the folded layer at the theoretically justified locations [Berg et al., 2011] that would ensure immobility of the manipulated garment during unfolding, as more than one grasping point would be needed usually.

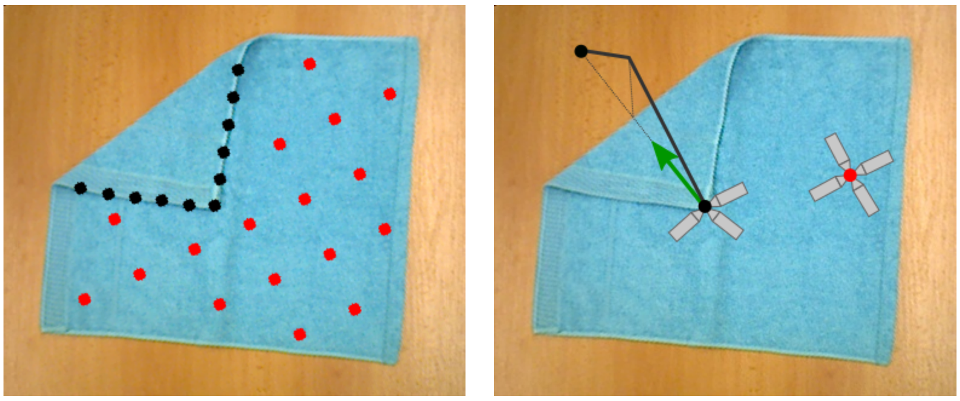


Figure 5.9. a) The grasping candidates on the inner contour of the top layer (black dots) and holding candidates on the surface of the bottom layer (red dots). b) The unfolding path (black polyline) for the most prioritized grasping point and for different orientations of the grippers. The sliding direction of the gripper (green arrow) is same for all orientations.

The second arm is pressing the bottom layer towards the table to prevent the garment from slipping during unfolding, which would occur otherwise due to the horizontal pulling force generated by the unfolding arm. This approach is used instead of generating such unfolding trajectory that would minimize the horizontal force, as described in Sec. 2.4, and that would enable to employ both arms for the unfolding. However, the trajectory optimization is computationally demanding and relies on known material properties of the garment.

Several grasping and holding position candidates are sampled and tested to increase the robustness of the method, because the kinematic restrictions of the robot do not allow to plan trajectories for each combination of the positions. The candidates for the grasping position are sampled uniformly on the inner boundary of the top layer (Fig. 5.9a). Several orientations of the gripper, limited by the shape of the top layer and by the gripper mechanics, are sampled for each candidate grasping point to find such configuration that allows the gripper to slide under the top layer while grasping it. The holding position candidates are sampled uniformly on the bottom layer (Fig. 5.9a), with the orientations allowed by the gripper mechanics.

The planning algorithm repeatedly selects a pair of the grasping and holding candidates and tries to follow the triangular unfolding path virtually (Fig. 5.9b). If the path is feasible for the robot, it is used for the real unfolding. Otherwise, the next grasping and holding pair is selected and tested. The grasping positions more distant from the folding axis are tested first, as they usually prevent the top layer from deforming during unfolding (Fig. 5.9a). The grasping orientations and holding positions are tested in an arbitrary order for each grasping position.

5.7. Experiments

The proposed method for the detection of folds was evaluated on the dataset, which we acquired for this purpose. It contains manually annotated images and depth maps of garments posed in various folded configurations. The dataset contains 13 garments of 8 categories (Fig. 5.10). They are made of different materials, e.g. cotton, polyester, denim or leather. Therefore they vary significantly in thickness, stiffness or friction.

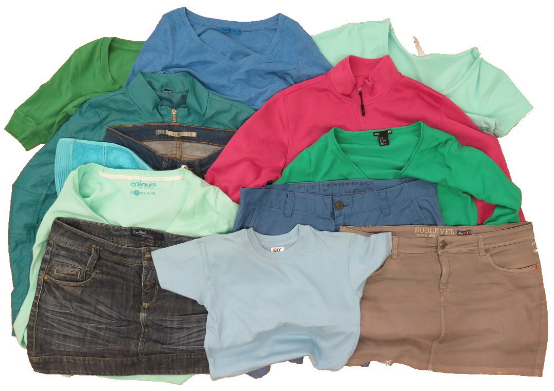


Figure 5.10. Garments used for the experimental evaluation, including a jacket, jeans, shorts, 2 skirts, 2 sweaters, a sweatshirt, a towel and 4 T-shirts of various sleeve lengths. The garments are made of various materials and having various colors.

Garment	Items	Success	Failure cause		Displac. [mm]	
			Layers	Axis	Mean	Stdev.
Jacket	1	14 / 15	1	0	6.4	8.6
Jeans	1	12 / 15	3	0	3.4	4.5
Shorts	1	14 / 15	1	0	3.5	3.4
Skirt	2	25 / 30	5	0	4.1	8.9
Sweater	2	26 / 30	2	2	4.2	4.8
Sweatshirt	1	14 / 15	0	1	2.7	3.2
Towel	1	14 / 15	1	0	3.4	2.4
T-shirt	4	51 / 60	9	0	4.2	3.8
Total	13	170 / 195	22	3	4.0	5.2

Table 5.1. Performance evaluation of layers detection and folding axis estimation on various garments included in the dataset. The displacement of the detected boundary between the top and bottom layer is provided for the successfully detected layers.

They are also variously colored. Each garment was placed on the table and posed into 15 different folded configurations, which results into 195 testing data samples in total.

The images and depth maps were acquired by the calibrated RGBD sensor (Sec. 4.1). The images and depths are registered and scaled to the same resolution 640×480 pixels. The acquired data were annotated manually to have a ground truth for the evaluation. The boundary between the top and bottom layer was denoted with a polyline drawn carefully over the image. The folding axis is denoted with an oriented line segment, whose right side is aligned with the outer side of the garment boundary.

The dataset was used to evaluate the methods for layers detection (Sec. 5.3) and folding axis estimation (Sec. 5.5). The results are summarized in Tab. 5.1. The first column states the garment type, the second one the number of unique items of that particular type. Each item was posed into 15 different configurations. The third column presents the ratios of the correctly recognized folded configurations. The next two columns analyze the counts of failures caused either by the layers detection or the folding axis estimation. The overall success rate is 87 %. It does not vary significantly

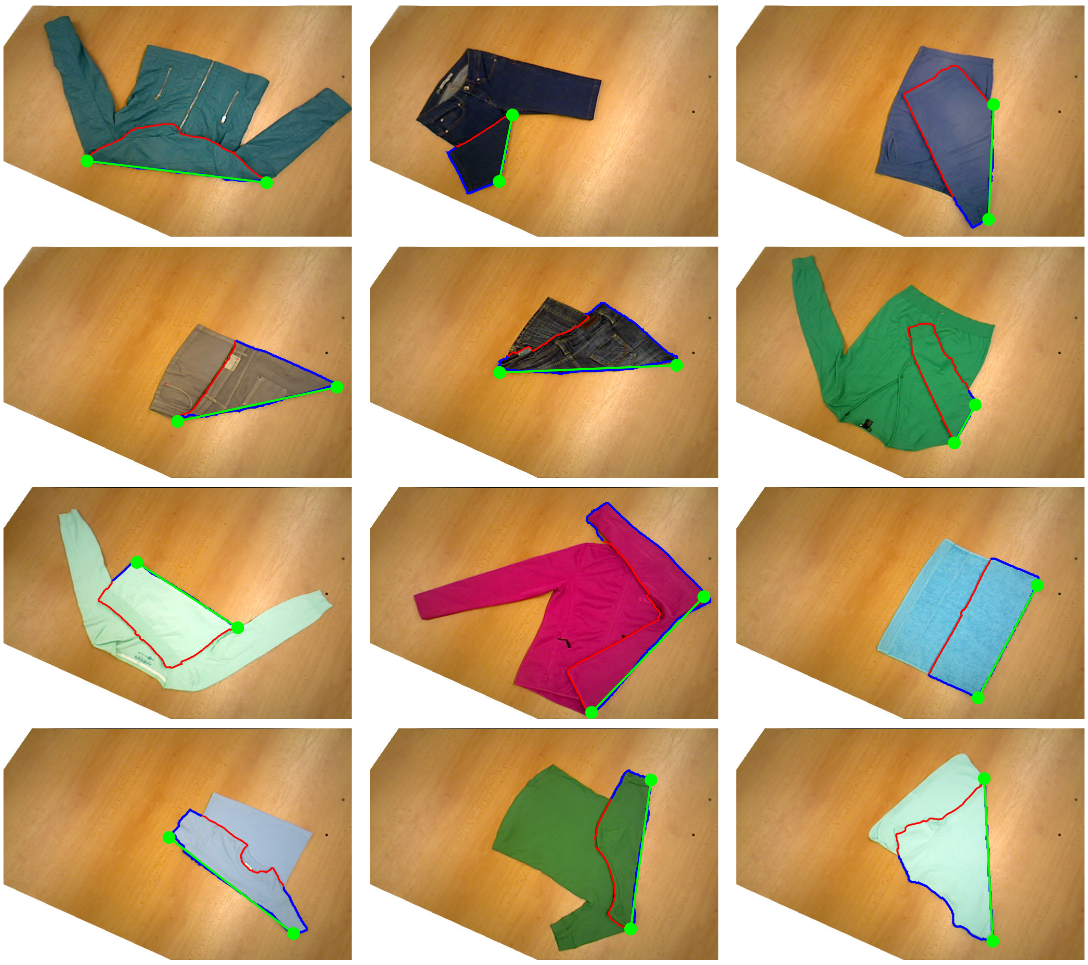
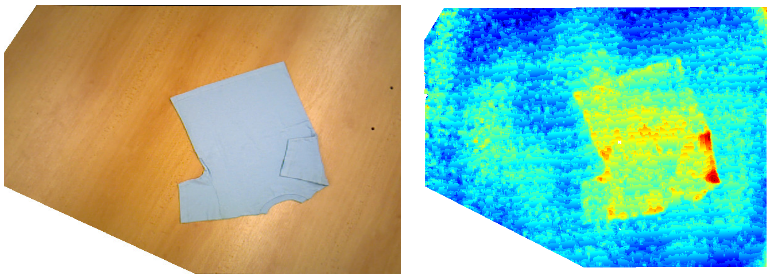


Figure 5.11. Selected examples of the successfully detected folds and folding axes. Each image shows the detected boundary between the layers (red), outer boundary of the top layer (blue) and estimated folding axis (green segment). The background outside the table is masked out.

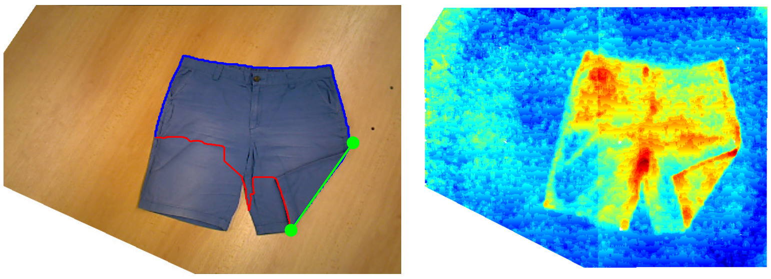
for various garments, which proves the generality of the proposed method. The layers detection mechanism tends to be more error-prone, having 11 % failure rate, compared to 2 % failure rate of the folding axis detection.

The last two columns of Tab. 5.1 provide a quantitative evaluation of the layers detection. For each of 173 configurations, in which the layers were detected correctly, the displacement of the detected and annotated boundary between the layers is computed. Namely, for each point of the correctly detected boundary, we find its closest point on the manually annotated boundary. The usual displacement is several millimeters, which is more than sufficient for reliable grasping, considering the size of the gripper. Fig. 5.11 shows selected examples of the successfully detected layers and folding axes.

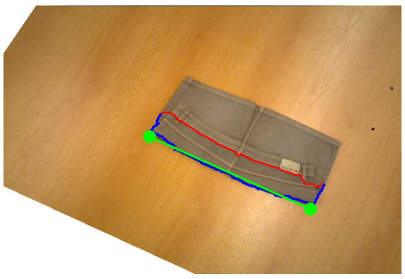
The observed failures can be split into three categories. The first category are failures in the detection of layers when all pixels happen to be assigned to the single layer only. It occurs when heights of the layers differ insignificantly and their boundary is not clearly visible in the image (Fig. 5.12a). The second category of failures corresponds to a wrong labeling of the layers. It can be caused either by misleading depth information (Fig. 5.12b), or by the presence of strong image edges, which are not adjacent to the boundary between the layers (Fig. 5.12c). The last and rarest type of failures is a wrongly chosen folding axis. It is caused by the employed heuristic on choosing such



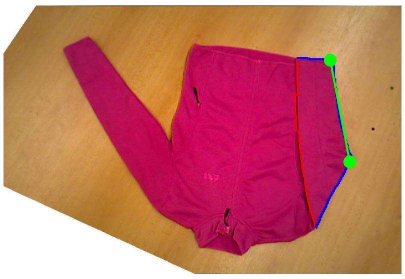
a) Undetected top layer



b) Wrong detection due to depth



c) Wrong detection due to edges



d) Wrong folding axis

Figure 5.12. Analysis of the observed failures. a) All pixels were assigned to the bottom layer. b) Wrongly detected layers caused by misleading depth information and c) by misleading edges in the image. d) Wrongly selected folding axis caused by the employed heuristics.

axis, which has the shortest unfolded contour of all possible candidates. The heuristic can fail for some configurations of a garment (Fig. 5.12d).

The proposed perception method was implemented in Python and C++. The time performance was evaluated on a notebook with Intel i7-3740QM 2.7 GHz processor and 16 GB memory. The preprocessing stage (Sec. 5.2) takes 3.1 seconds on the average, spent mostly by the segmentation. Time spent by the detection of layers (Sec. 5.3) varies from 0.4 to 1.8 seconds. It is directly proportional to the size of the observed garment, as it is solved by labeling each pixel. Estimation of the folding axis (Sec. 5.5) takes 0.1 seconds at most.

The dual-arm robotic testbed (Sec. 4.1) was used for evaluation of the manipulation procedure (Sec. 5.6). The performance is affected mainly by two factors. First, the space of positions and orientations reachable by the robot is limited due to its construction properties. The robot is able to unfold rather small garments placed close to its base. Second, only the left arm was equipped with the griper suitable for grasping of garments (Fig. 4.3b) during the experimental evaluation. Therefore, the robot was able to unfold only such configurations, where the folding axis is located on the left side (Fig. 5.9a). If two grippers were available, it would be possible attempting to plan

Garment	Success	Failure cause		
		Detection	Planning	Execution
Shorts	3 / 5	1	1	0
Sweatshirt	4 / 5	1	0	0
Towel	5 / 5	0	0	0
T-shirt	5 / 5	0	0	0
Total	17 / 20	2	1	0

Table 5.2. Performance evaluation and failure analysis of unfolding.

the manipulation for both combinations of the grasping and holding arm. Since both limiting factors are rather technical than methodological, we tend to place the garments into the configurations reachable and graspable by the robot in our experiments.

Tab. 5.2 summarizes the experimental results. Each garment was placed into 5 different folded configurations[1]. The garments were unfolded successfully in 17 attempts out of 20. Two failures were caused by the wrong detection of folds. In one case, the garment was placed in such configuration that it was impossible to plan its unfolding. The robotic manipulation itself was always successful in the experiment.

Selection of the grasping and holding position and planning of the folding trajectory takes 1–15 seconds, depending strongly on the ranking of the successfully planned candidate in the priority queue (Fig. 5.9a). The unfolding manipulation takes usually 40–50 seconds, with the robot starting and finishing in the initial position and moving rather slowly due to the safety reasons.

[1]Unfolding videos: `http://cmp.felk.cvut.cz/~striajan/phd`

6. Classifying category of hanging garments

A novel method [Stria and Hlaváč, 2018] is proposed for classifying the category of an unknown garment, which was grasped by the robot, lifted up and is being held in a hanging state. The input is a sequence of depth maps taken from different viewpoints around the garment. The depth maps are acquired by a stationary RGBD sensor, while rotating the robot wrist holding the garment, i.e. rotating the hanging garment around its vertical axis. The depth maps are fused into a single 3D point cloud.

The point cloud is fed into a convolutional neural network (CNN) that transforms it into a single global feature vector of a fixed size. The network works directly with the unordered set of 3D points. It utilizes a novel type of convolution defined over a local neighborhood of a point. The convolution is invariant to a rotation, translation and scaling of the point cloud. The whole network is invariant to a permutation of the points. Sizes of the local neighborhoods grow and the point cloud is repeatedly subsampled in deeper layers. This causes the receptive fields of the neurons to grow.

The network was trained on a dataset of common 3D objects, since there are no large datasets of hanging garments available. The features extracted by the convolutional layers are general enough to be transfered to our domain. The fully connected layers are not adapted to a new task, however. The local features for individual points are pooled to a single global feature vector instead, which is classified with a support vector machine (SVM) [Cortes and Vapnik, 1995] trained on smaller datasets of garments.

6.1. Fusion of depth maps

The proposed method deals with the category classification of an unknown garment for the purpose of the folding scenario (Sec. 1.3). The garment is randomly tossed initially. It may be even in a heap of crumpled garments. Our method assumes that a single garment was already grasped by the robot, lifted up and it is now being held in a hanging state under the gravity, e.g. using the approach described in Sec. 4.2. The hanging garment can be optionally regrasped for its lowest hanging point to reduce the space of its possible configurations (Sec. 4.3).

The garment is perceived from many viewpoints distributed around it. The gripper holding the garment is oriented downwards, while rotating the wrist, i.e. the garment is rotated around the vertical axis. The RGBD sensor is attached to the other arm, which remains stationary. The alternative would be mounting the sensor onto a tripod. This results in a sequence of viewpoints distributed on a horizontal circle around the garment, whose vertical axis intersects the centre. It is assumed that the wrist is rotated at slow pace, at least several seconds per a single rotation, so that the configuration of the hanging garment does not change by fluttering.

The RGBD sensor provides images and depth maps of the garment (Fig. 6.1a). The proposed method utilizes only depth data. It is therefore invariant to various colors and textures. The first step is a segmentation of the garment from its background in the depth map. If the sensor and robot are hand-eye calibrated, it is possible to keep only

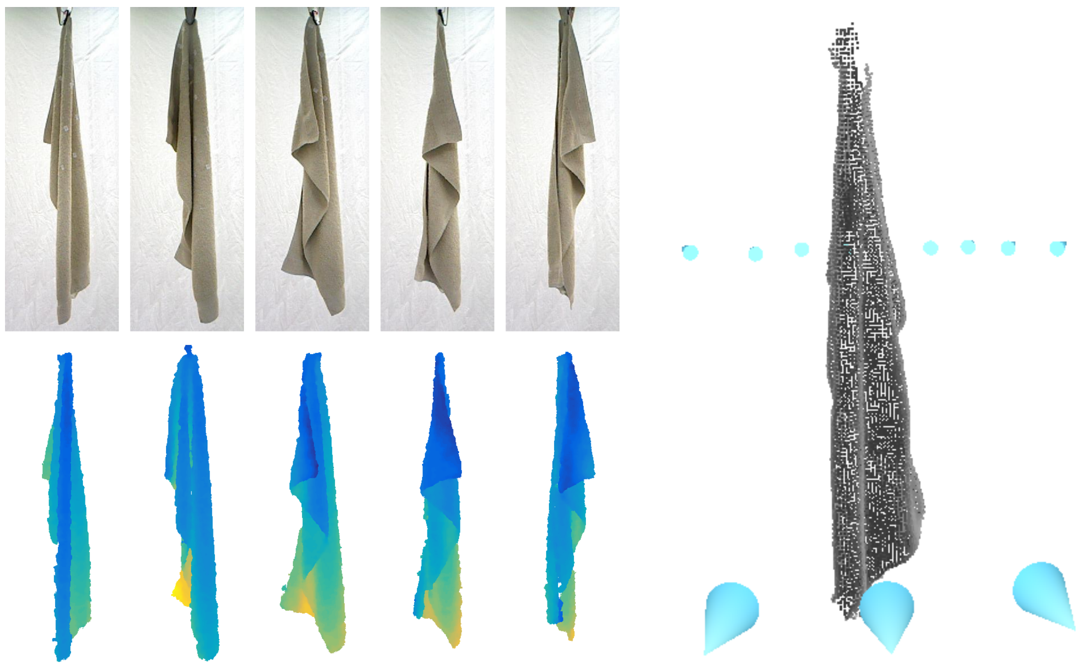

a) Input images and corresponding depth maps
b) Reconstructed 3D point cloud

Figure 6.1. Reconstruction of 3D point cloud. a) The input depth maps acquired from multiple viewpoints around the garment are segmented and fused. b) The reconstructed 3D point cloud with the estimated sensor poses visualized as cones.

data from a properly sized cuboid or cylinder below the gripper holding the garment. The cuboid is defined manually in our experiments on the existing datasets.

The depth maps from multiple viewpoints (Fig. 6.1a) are fused into a single global dense surface model by the Kinect Fusion algorithm [Newcombe et al., 2011], while tracking 6DOF pose of the sensor over time (Fig. 6.1b). Depth measurements from each viewpoint are converted to 3D vertices and normals in the camera coordinate system at first, using the intrinsic calibration matrix. The sets of 3D vertices from two consecutive viewpoints are registered by the Iterative Closest Point (ICP) [Besl and McKay, 1992] algorithm, which also estimates the relative rigid 6DOF transformation of the sensor, i.e. its relative translation and rotation with respect to the garment. It would be also possible to define a prior on the transformation, as the relative movement is controlled by rotating the robot wrist. However, the default method is used for simplicity.

The relative transformations are combined incrementally. The oriented 3D vertices from all viewpoints are brought to a single global coordinate system and fused into a volumetric representation. Each voxel stores a running average of the truncated signed distance function (TSDF). Its value is negative for the points inside the object and positive for those outside. The reconstructed 3D point cloud contains points corresponding to those voxels, whose neighbors have an opposite sign of TSDF.

We use the KinFu[1] implementation of the fusion algorithm in CUDA. It delivers real-time 3D reconstructions, even if running on the low-level GPU Nvidia GT 730M. We use 256^3 volumetric grid to represent the reconstructed cubical space of the edge length 200 cm. Density of the reconstructed 3D point cloud is therefore 7.8 mm, which corresponds approximately to the depth quantization step used by commonly available

[1]KinFu library: `http://github.com/Nerei/kinfu_remake`

RGBD sensors. We have found out that one full rotation of the garment with $10°$ step between the individual viewpoints suffices to obtain a precise enough reconstruction.

The reconstructed 3D point cloud usually contains thousands to tens of thousands points in our case. It is downsampled by selecting $n = 1024$ points randomly, which is enough for a suitable representation of its original shape. All points have an equal probability of being selected, which results in an approximately uniform coverage of the original point cloud in practice. The sampled point cloud is translated to zero mean and scaled to span a unit ball.

6.2. Local neighborhood of points

The classical CNNs [Krizhevsky et al., 2012] utilize 2D convolution kernels that are applied locally over the neighborhood of a pixel in the image grid. The ordering of the neighboring pixels is well defined by the grid. There is no such a neighborhood and ordering induced implicitly by a 3D point cloud. The pioneering PointNet architecture [Qi et al., 2017a] thus applies the convolution on each point separately, transforming its features to a higher dimension.

We propose a novel KnnConv operation defined on a local neighborhood of a point, which is given by its k-nearest neighbors (k-NN) from the point cloud. The standard Euclidean distance of the points in 3D space is used as a distance metric. An alternative is the geodesic distance [Spivak, 1970], which is a length of the shortest path between two points on the object surface. The neighborhood induced by the Euclidean and geodesic distance differ especially in highly curved parts of the object, e.g. on a wrinkle.

We experimented with reconstructing the object surface from the input point cloud, but rejected it from several reasons. First, the reconstruction is very time consuming, compared to the computation performed by the network. Second, the highly curved parts of the object, where we would benefit the most from knowing the actual surface, are the main source of reconstruction errors. Last, it is possible in principle for the network to learn estimating the local surface even from the Euclidean neighborhood.

The neighboring points are found by an exhaustive search over all n input points. An alternative are the approximate search methods, e.g. a k-D tree [Muja and Lowe, 2014]. However, from our experience, they do not bring any performance improvement over the exhaustive search implemented on GPU for $n = 1024$.

The input of our method is the 3D point cloud $\{p_1, \ldots, p_n\}$, where $p_i \in \mathbb{R}^3$. Since the PyTorch framework [Paszke et al., 2017], in which our network is implemented, does not support computation of the pairwise distance matrix $D \in \mathbb{R}^{n\times n}$, the Euclidean distance $D_{i,j}$ of two points p_i and p_j is computed based on the following formula:

$$D_{i,j} = ||p_i - p_j||^2 = ||p_i||^2 - 2p_i^\top p_j + ||p_j||^2. \tag{6.1}$$

The input points are organized into the design matrix P. The 3D coordinates of each point form an individual row:

$$P = \begin{bmatrix} p_1^\top \\ \vdots \\ p_n^\top \end{bmatrix} \in \mathbb{R}^{n\times 3}. \tag{6.2}$$

The squared Euclidean norms $||p_i||^2$ from (6.1) are computed in parallel by summing the columns of the element-wise Hadamard product $P \circ P$. The dot products

$p_i^\top p_j$ from (6.1) are computed for all pairs of points in parallel as the following matrix multiplication:

$$p_i^\top p_j = \left(PP^\top\right)_{i,j}. \tag{6.3}$$

The rows of the distance matrix D are sorted in ascending order. Each row is sorted independently on GPU in $\mathcal{O}(n \log n)$ time using the PyTorch framework, while remembering the sorted indices. This results in the matrix $K \in \{1, \dots, n\}^{n \times n}$, containing in the i-th row indices of all n points ordered in ascending distance from the point p_i. The matrix K is computed only once at the beginning and reused later to obtain k-NN for different values of k. When the point cloud is subsampled in deeper layers by keeping only a fraction of the points, the matrix K is subsampled accordingly by keeping only the corresponding rows and filtering values in their columns.

6.3. Convolution on point sets

The local neighborhoods of points are specified by taking the first k columns of the index matrix K. Each point is thus considered to be its closest neighbor. The remaining neighbors are ordered in ascending Euclidean distance from the reference point. The relation of the neighborhood and ordering of the neighbors are therefore invariant to any permutation of the input points $\{p_1, \dots, p_n\}$. They are also invariant to a rotation and translation of the point cloud, which do not affect the distances between points, and to its scaling, which multiplies all distances by a constant factor. The k-NN relation therefore enables defining a convolution over the local neighborhoods of points, similarly to a convolution over the neighboring pixels in an image grid.

In addition to its 3D spatial coordinates $p_i \in \mathbb{R}^3$, each point is also assigned a d-dimensional feature vector $x_i \in \mathbb{R}^d$. We set $x_i = p_i$ in our experiments for the input point cloud, i.e. the initial feature vectors are just world coordinates of the points. In general, the initial feature vector x_i can contain an RGB color of the point, its surface normal estimated from the local neighborhood, or an arbitrary local point cloud descriptor [Hana et al., 2018] computed in advance.

The convolution combines the feature vectors from a local neighborhood of a point and transforms them to a new feature vector describing the reference point. The feature vectors of k neighboring points, specified by the matrix K, are concatenated at first to obtain a feature vector z_i describing the neighborhood of the i-th point:

$$z_i = \begin{bmatrix} x_{K_{i,1}} \\ \vdots \\ x_{K_{i,k}} \end{bmatrix} \in \mathbb{R}^{kd}. \tag{6.4}$$

A single convolution kernel is specified by a vector of weights $w_j \in \mathbb{R}^{kd}$. The convolution is applied on the neighborhood of the i-th point by computing a dot product with the neighborhood descriptor z_i. The result is optionally added a scalar bias $b_j \in \mathbb{R}$:

$$z_i^\top w_j + b_j. \tag{6.5}$$

A new transformed feature vector x_i' for the i-th point is formed by applying m different convolution kernels and corresponding biases:

$$x_i' = \begin{bmatrix} z_i^\top w_1 + b_1 \\ \vdots \\ z_i^\top w_m + b_m \end{bmatrix} \in \mathbb{R}^m. \tag{6.6}$$

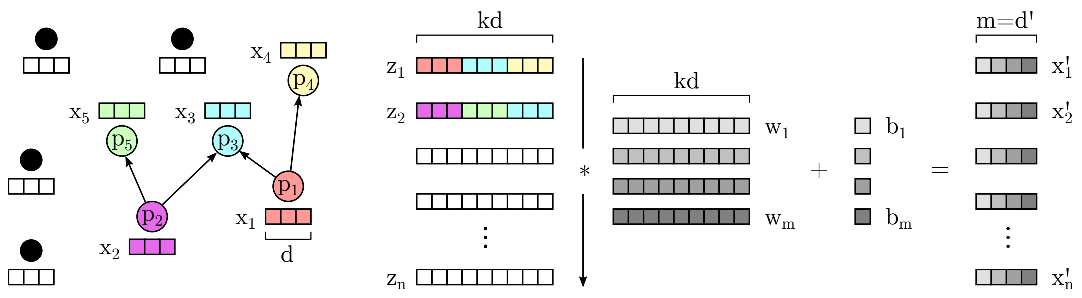


Figure 6.2. Visualization of KnnConv($k = 3, d = 3, m = 4$) operation. The point cloud contains n points. The i-th point has spatial 3D coordinates $p_i \in \mathbb{R}^3$ (shown as a dot) and is assigned a feature vector $x_i \in \mathbb{R}^d$ (shown as a vector). The feature vectors of k spatially closest neighbors of each point (denoted with arrows) are concatenated to a neighborhood descriptor $z_i \in \mathbb{R}^{kd}$. Each descriptor z_i is dot multiplied with m kernels $w_j \in \mathbb{R}^{kd}$ and added biases b_j to form a new transformed feature vector $x'_i \in \mathbb{R}^m$ for the i-th point.

Let us denote a single layer of convolutions as KnnConv(k, d, m), where d is the dimensionality of the input feature vectors, k is the size of the local neighborhood and m is the number of convolution kernels. The layer has $m(kd + 1)$ learnable parameters in total, which are the weight vectors $w_1, \dots, w_m$ and corresponding biases $b_1, \dots, b_m$. The neighborhood size k and number of kernels m are hyperparameters of the layer, which need to be specified in advance and validated.

A special case is KnnConv(k, d, m) for $k = 1$. Since each point is its closest neighbor, it holds $z_i = x_{K_{i,1}} = x_i$ in this case. Each feature vector x_i is therefore transformed separately by such convolution layer. This restricted transformation is useful for combining current point features to more complex higher features. It is also a transformation used solely by the original PointNet [Qi et al., 2017a] architecture.

We implemented the KnnConv operation in PyTorch framework [Paszke et al., 2017], using the available implementation of 1D convolution. The local neighborhood descriptors (6.4) for all points are organized to a single $n \times kd$ tensor and convolved with $1 \times kd$ tensor of weights (Fig. 6.2). This can be seen as a discrete multi-channel 1D convolution of a vector, which has a length n and kd channels, with a convolution kernel, which has a unit size and kd channels. It requires $\mathcal{O}(nkd)$ multiplications and additions.

In practice, KnnConv is computed for a batch of r point clouds and all m convolution kernels in parallel. The descriptors (6.4) are therefore organized into a single $r \times n \times kd$ tensor and convolved with $m \times 1 \times kd$ tensor of weights over the second dimension. The result is a $r \times n \times m$ tensor of new m-dimensional feature vectors for all points. Whole computation therefore requires $\mathcal{O}(rnmkd)$ multiplications and additions. The implementation of convolutions in PyTorch is highly optimized and massively parallelized, utilizing capabilities of modern Nvidia GPUs through CUDA and cuDNN libraries.

6.4. Subsampling of points

The local neighborhood descriptor z_i is formed by concatenating the feature vectors $x_{K_{i,1}}, \dots, x_{K_{i,k}}$ assigned to k-NN of a point, as defined in (6.4). Therefore, the local neighborhood descriptors z_i belonging to the neighboring points share common subvectors and also the transformed feature vectors x'_i, defined in (6.6), of the neighboring points encode partially redundant information. To reduce the redundancy and computational costs, we subsample the point cloud after several KnnConv layers, similarly to

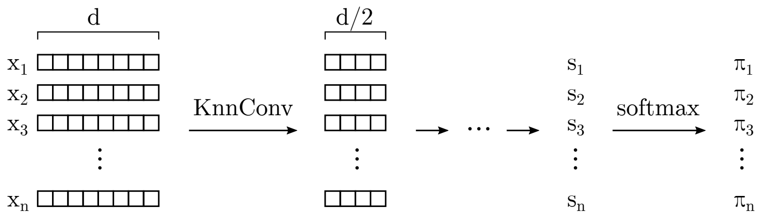


Figure 6.3. Example of the sub-network for estimating the sampling probabilities from the feature points $f(x_i) = \pi_i$. It is formed by a stack of KnnConv(k, d, m) for $k = 1$ and d halving gradually. The result scores $s_i \in \mathbb{R}$ are normalized to probabilities π_i with softmax.

[Qi et al., 2017b]. Only a subset of points is selected randomly to represent the object, whereas the remaining points are dropped out from the point cloud, together with their feature vectors. If the sampled points cover approximately all parts of the original point cloud, almost no information should be lost.

Prior to subsampling, the i-th point is assigned a probability π_i of being selected. This can be considered a categorical distribution over all n points:

$$\pi_i \geq 0, \quad \sum_{i=1}^{n} \pi_i = 1. \tag{6.7}$$

We sample n' points from this distribution without a replacement, where usually $n' = n/2$, i.e. a half of the points is discarded. During the backward pass, the gradients are propagated only through the sampled points. However, as the local neighborhood descriptors consist of feature vectors from k-NN points, as defined in (6.4), the gradients get propagated gradually to almost all discarded points.

We experimented with two approaches of selecting the subset of points. The first method assigns an equal probability of being sampled $\pi_i = 1/n$ to each of n points. As n is in the order of hundreds, it is likely that the sampled points cover all parts of the 3D object due to the law of large numbers.

The second approach computes the probability as a function of the feature vector at a particular layer $\pi_i = f(x_i)$. The motivation is that the feature vector $x_i \in \mathbb{R}^d$ could hold information about the importance of the i-th point, which is extracted by f. The function f is realized by a small sub-network (Fig. 6.3) of several KnnConv layers, which predict a sampling score $s_i \in \mathbb{R}$ for each point, followed by the softmax function [Goodfellow et al., 2016], which normalizes the scores to the categorical probability distribution (6.7). The function f is learned in parallel with the main network.

The experiments showed, however, that the sampling utilizing the computed probabilities does not bring any improvement over the uniform probabilities. The possible reason is that the sub-network f is not able to learn a correct mapping from a high dimensional space $\mathbb{R}^d$ to a scalar probability. Moreover, propagation of the gradients through samples from the categorical distribution (6.7) is not straightforward, requiring a specific reparametrization [Jang et al., 2017]. The final architecture thus assigns uniform sampling probabilities $\pi_i = 1/n$ to all points.

The subsampling can be considered a form of pooling, which is used in CNNs for image processing [Goodfellow et al., 2016]. In the standard pooling, the feature vectors from several neighboring pixels, usually 2×2 or 3×3 window, are replaced with a single aggregated vector, usually containing maximums over the corresponding values. A spatial resolution of the input is reduced in consequence, which also reduces a number of computations in deeper layers. This is also true for the proposed subsampling.

Since the points are sampled randomly, the subsampling improves robustness of the network and prevents overfitting. The same object is represented by different subsets of points in the individual epochs during training. As only a random subset points is kept and used to update the weights in deeper layers, the subsampling can be seen as a dropout [Srivastava et al., 2014] on the data side.

6.5. Network architecture

The input of the network is a 3D point cloud $\{p_1, \dots, p_n\}$. The matrix K is built by the KnnSearch module (Sec. 6.2), containing in the i-th row indices of all points ordered in ascending distance from the point p_i. The matrix K is used by KnnConv layers.

The point cloud is processed by a stack of KnnConv layers (Sec. 6.3), extracting a set of local features. The l-th convolutional layer $\text{KnnConv}(k^{(l)}, d^{(l)}, m^{(l)})$ is specified by three parameters: the number of neighbors $k^{(l)}$, dimensionality of inputs $d^{(l)}$ and number of convolution kernels $m^{(l)}$. The input feature vector $x_i^{(l)} \in \mathbb{R}^{d^{(l)}}$ is transformed to a new vector $x_i^{(l+1)} \in \mathbb{R}^{m^{(l)}}$ by the l-th layer, i.e. it holds $m^{(l)} = d^{(l+1)}$. We set $x_i^{(1)} = p_i \in \mathbb{R}^3$, i.e. the initial feature vectors are Cartesian coordinates of the points.

The neighborhood size $k^{(l)}$ increases in deeper layers, causing the point features to describe higher level concepts related to larger parts of the object. An exception are the feature transformation convolutions (Sec. 6.3), where $k^{(l)} = 1$. The number of kernels $m^{(l)}$ increases in deeper layers. Outputs of the convolutional layers are batch normalized [Ioffe and Szegedy, 2015]. The rectified linear unit (ReLU) [Goodfellow et al., 2016] activation function is applied. The point cloud is subsampled (Sec. 6.4) after a block of convolutional layers, followed by an another block.

The output of the convolutional part of the network with l layers is a set of point feature vectors $\{x_1^{(l+1)}, \dots, x_{n'}^{(l+1)}\}$, where $x_i^{(l+1)} \in \mathbb{R}^{m^{(l)}}$ and $n' \leq n$ because of involved subsamplings. As proposed by [Qi et al., 2017a], a function g symmetric in its arguments is applied to obtain a global feature vector independent on ordering of the points:

$$g\left(x_1^{(l+1)}, \dots, x_{n'}^{(l+1)}\right) \in \mathbb{R}^{m^{(l)}}. \tag{6.8}$$

The recommended form of g is a pooling function that aggregates the corresponding elements of all n' vectors to a single value. We use specifically the max-pooling in our KnnNet. The global feature vector is therefore the element-wise maximum over all local features. We experimented with implementing the function g with a recurrent neural network (RNN) [Goodfellow et al., 2016], namely the gated recurrent unit (GRU) [Cho et al., 2014]. The set of features $\{x_1^{(l+1)}, \dots, x_{n'}^{(l+1)}\}$ is processed as a sequence by the network. Since RNNs are not invariant to the ordering of inputs, the feature vectors were permuted randomly during the training. However, RNNs did not bring any significant improvement over the pooling.

The global feature vector (6.8) is classified by a stack of fully connected layers. Let us denote a single layer as $\text{FC}(c^{(l)})$, where $c^{(l)}$ is the number of neurons. All layers except the last one are batch-normalized and use ReLU activation function. The output layer uses the logarithm of softmax function to predict logarithms of a categorical distribution over c output classes.

Outputs of all fully connected layers except the last one are regularized by dropout [Srivastava et al., 2014]. Let us denote the dropout layer as $\text{Dropout}(p)$. It discards the output of each individual neuron from the previous layer with the probability $1 - p$

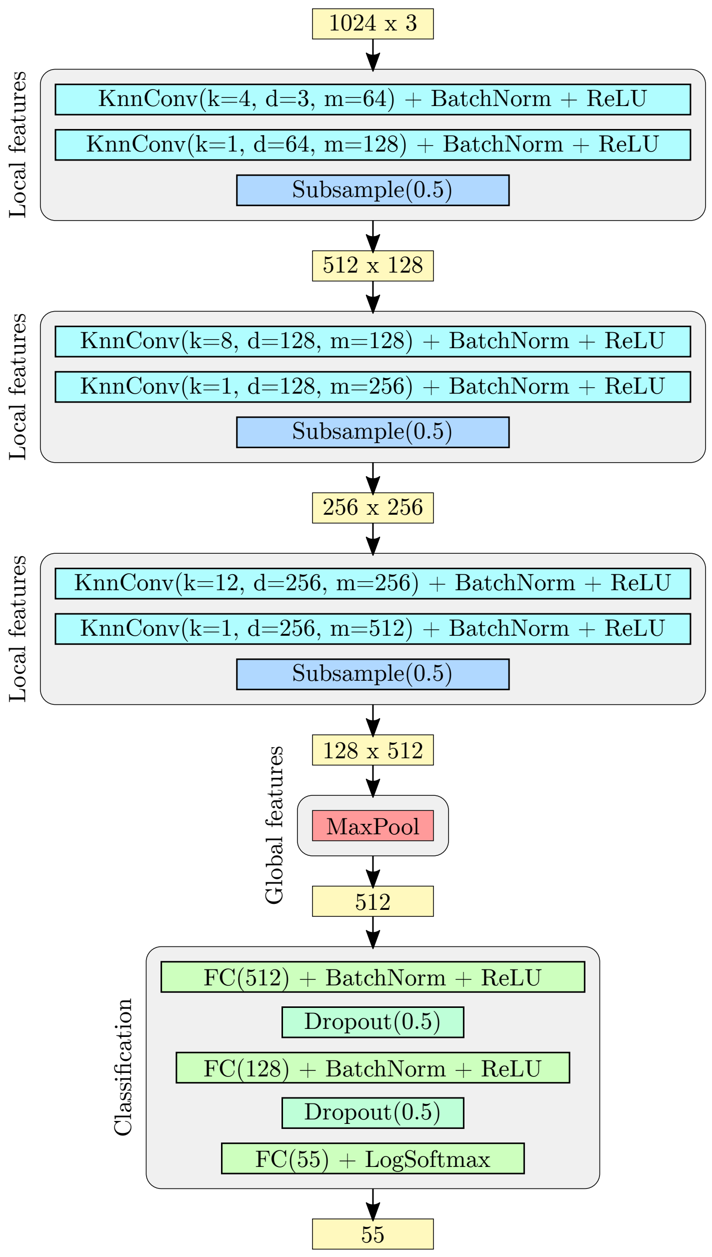

1024 x 3
Local features
KnnConv(k=4, d=3, m=64) + BatchNorm + ReLU
KnnConv(k=1, d=64, m=128) + BatchNorm + ReLU
Subsample(0.5)
512 x 128
Local features
KnnConv(k=8, d=128, m=128) + BatchNorm + ReLU
KnnConv(k=1, d=128, m=256) + BatchNorm + ReLU
Subsample(0.5)
256 x 256
Local features
KnnConv(k=12, d=256, m=256) + BatchNorm + ReLU
KnnConv(k=1, d=256, m=512) + BatchNorm + ReLU
Subsample(0.5)
128 x 512
Global features
MaxPool
512
Classification
FC(512) + BatchNorm + ReLU
Dropout(0.5)
FC(128) + BatchNorm + ReLU
Dropout(0.5)
FC(55) + LogSoftmax
55

Figure 6.4. Proposed KnnNet architecture. Yellow boxes show sizes of data in individual stages for a single point cloud. The input is formed by 1024 points of the dimension 3. The points are reduced to a half repeatedly by a random sampling, while the dimension of their feature vectors grows up to 512 because of KnnConv layers. The local feature vectors of all 128 points are max-pooled to a single global feature vector of the dimension 512. The dimension of the output vector 55 correspond to a number of classes.

Figure 6.5. Selected 3D models from ShapeNet dataset [Chang et al., 2015], which contains approximately 51 thousands samples over 55 categories of common objects.

by setting it to zero. The output is unchanged with the probability p. Gradients are not propagated through the zeroed nodes during the backward pass.

We experimented with different architectures of the network, namely with various numbers of convolutional blocks and layers inside these blocks, numbers of fully connected layers and sizes of layers. We also tested various sizes of local neighborhoods, numbers of sampled points or settings of dropout. Many different architectures were trained, as described in Sec. 6.6. The architecture with the best validation accuracy was used in the final experiments (Fig. 6.8).

Fig. 6.4 shows the final architecture of our network. It comprehends three blocks of convolutional layers for the extraction of local features. Each block consists of a single KnnConv layer over the local neighborhoods of points, one feature transforming KnnConv layer and a random subsampling of the point cloud to a half. Size of the local neighborhood grows from 4 in the first block to 12 in the last one. Number of the convolution kernels grows from 64 to 512 after the last block, which is therefore a dimension of the final local feature vectors for 128 sampled points. The local features are max-pooled to a single global feature vector of dimension 512 that is classified by a stack of 3 fully connected layers.

6.6. Network training

The existing CNNs for garments classification use depth maps on their input (Sec. 2.2). They are trained from scratch on datasets of hanging garments, without any pre-training on general RGBD datasets. The shortage of real world data is partially overcome by generating synthetic samples with the cloth simulation engines included in Blender[2] [Mariolis et al., 2015] or Maya[3] [Gabas et al., 2016]. It is, however, difficult to ensure sufficient diversity of synthetic data to avoid overfitting.

Our network is trained on ShapeNet [Chang et al., 2015], which is arguably the largest publicly available dataset of annotated 3D models (Fig. 6.5). It contains more than 51 thousands mesh models over $c = 55$ categories of common objects, including vehicles, household equipment, electronics etc. We believe that by training our network on such a variety of objects, the convolutional layers learn to extract general enough features that can be transfered to a new domain of garments. The dataset is split to 90 % training and 10 % validation subset, which was used to develop the network architecture and optimize its hyperparameters.

Each 3D mesh model from ShapeNet is converted to a 3D point cloud by sampling 2048 uniformly distributed points on its surface. The point clouds are normalized to

[2]Blender: `http://www.blender.org`

[3]Autodesk Maya: `http://www.autodesk.com/products/maya`

be zero meaned and located inside a unit ball, as in Sec. 6.1. Following augmentations are employed in each training epoch to avoid overfitting. A subset of $n = 1024$ points is sampled randomly, which ensures that the same cloud is almost never seen again in different epochs. The sampled point cloud is rotated around the vertical axis by an angle selected randomly from the range $[0, 2\pi)$ radians.

The network is trained with Adam [Kingma and Ba, 2014] algorithm, using the recommended values of hyperparameters $\beta_1 = 0.9$ and $\beta_2 = 0.999$. The initial learning rate $\alpha = 0.001$ is decreased every 20 steps by multiplying it with the factor 0.7. The categorical cross entropy loss is minimized. No weight decay is used. The network is trained for 100 epochs in batches of 32. The classification accuracy on the validation subset of the ShapeNet dataset is 81%.

The training takes approximately 6 hours on a single GPU Nvidia GeForce GTX 1080 Ti, which has 3584 CUDA cores and 11 GB memory. Since the whole network together with the ShapeNet dataset take approximately 3.3 GB memory, no data transfers between the main and GPU memory are needed during training.

6.7. Classification

Despite that the network is trained on a dataset of common 3D objects, it is intended to be used for classifying the categories of hanging garments. A standard approach would be the adaptation of the weights to a new domain [Yosinski et al., 2014]. It is achieved by freezing the weights up to a certain layer, usually all convolutional layers, and adapting the remaining weights on a smaller dataset from that particular domain. The convolutional weights can be eventually fine tuned with a small learning rate.

We experimented with freezing different parts of the network, up to the penultimate fully connected layer. However, as the available datasets of hanging garments contain only several hundreds of training sequences, it is difficult to avoid overfitting. Therefore, only the convolutional part of the network followed by the pooling (6.8) is used to extract a single global feature vector representing the point cloud, which is classified by SVM.

Given the set of training data $\{(g_j, y_j) \mid j = 1, \dots, N\}$, where $g_j \in \mathbb{R}^d$ and $y_j \in \{\pm 1\}$, the linear SVM is a linear classifier [Friedman et al., 2001] that maximizes the distance between the decision boundary and the closest samples. This should improve generalization on unseen data. SVM training can be formulated as the following constrained optimization task:

$$\begin{aligned} &\min_{\substack{\beta \in \mathbb{R}^d \\ \beta_0 \in \mathbb{R} \\ \xi \in \mathbb{R}^N}} \frac{1}{2} ||\beta||^2 + C \sum_{j=1}^{N} \xi_j \\ &\text{subject to } y_j(\beta^\top g_j + \beta_0) \geq 1 - \xi_j, \\ &\qquad \xi_j \geq 0. \end{aligned} \tag{6.9}$$

The linear decision boundary is specified by the parameters β, β_0. The slack variable $\xi_j \in (0, 1]$ penalizes a correctly classified g_j, which is close to the decision boundary, while $\xi_j > 1$ is a penalty for a misclassified sample. The sample g_j is a global feature vector obtained by pooling over local features. It is an output of the function g defined in (6.8), i.e. there is a single g_j computed for each of N training point clouds. Since SVM is a binary classifier, i.e. $y_j \in \{\pm 1\}$, we train c different classifiers for c output classes using one-against-all strategy. Each of them classifies samples from the specified class as positive and all remaining samples as negative.

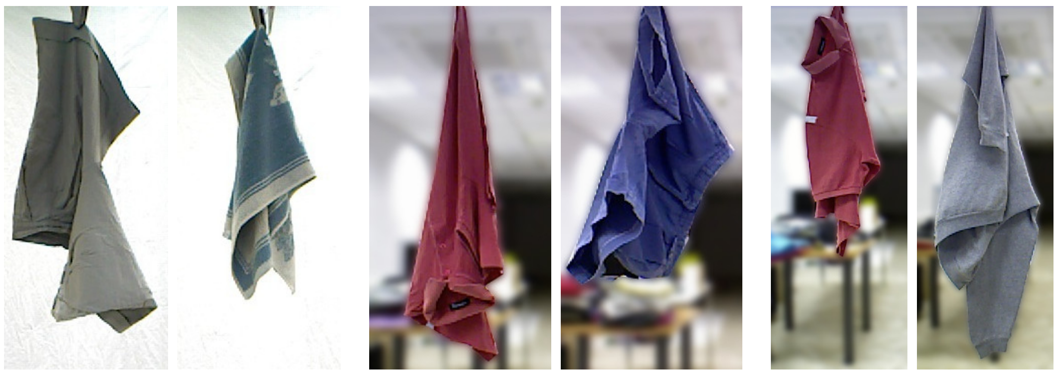


Figure 6.6. Samples from datasets of hanging garments used in the evaluation: a) randomly grasped garments [Kampouris et al., 2016]; b) garments regrasped for the lowest point followed by c) the first unfolding point [Doumanoglou et al., 2014b].

The regularization hyperparameter C, which weights the per-sample classification penalties, is found by cross-validation over the training set. We use specifically the LIBSVM implementation [Chang and Lin, 2011] of SVM.

6.8. Experiments

The proposed method for the classification of hanging garments was evaluated on two publicly available datasets [Kampouris et al., 2016] and [Doumanoglou et al., 2014b]. The former contains randomly grasped garments, while garments in the latter one were regrasped for the lowest hanging point to reduce the space of possible configurations. Both datasets contain images and depth maps taken from multiple viewpoints around the garment, enabling the reconstruction of 3D point clouds (Sec. 6.1).

The referred datasets were used only for training in the corresponding original works. The evaluation was performed on additional testing data, which are either not available anymore [Doumanoglou et al., 2014b] or do not contain full sequences of viewpoints needed for fusing depth maps into point clouds [Kampouris et al., 2016]. We therefore use the original training datasets both for training and out-of-sample evaluation of our method. Its performance is compared to the reported performance of the original methods evaluated on unavailable or unsuitable testing data.

Our method is evaluated in leave-one-out manner [Friedman et al., 2001]. One clothing item per each category is put into the testing set and the SVM classifier (Sec. 6.7) is trained on the remaining items in each iteration. The process is repeated as many times as is the number of unique garment items per category. The classification results are averaged over all runs. Cross-validation of SVM hyperparameters is performed also in leave-one-out manner on each training set independently.

The dataset of randomly grasped garments [Kampouris et al., 2016] contains 16 unique items of 5 categories (Fig. 6.6a). Each garment was grasped by a robot at various points on its surface and hung up. The dataset comprises 3 pants $\times$ 59 grasping points, 3×74 shirts, 3×20 shorts, 4×56 T-shirts and 3×25 towels. That is 758 combinations in total, each perceived from 180 consecutive viewpoints rotated approximately by $2°$. Only 90 views are used to reconstruct 3D point clouds.

[Mariolis et al., 2015] applies CNN directly on the depth maps. To improve the classification accuracy, CNN was later combined [Kampouris et al., 2016] with the approach

Method	Towels	Shorts	Pants	T-shirts	Shirts	Overall
RF [Doumanoglou et al., 2014a]	72	**73**	47	80	82	75
dCNN [Mariolis et al., 2015]	48	53	60	76	84	70
RF+dCNN [Kampouris et al., 2016]	**78**	67	56	**90**	88	82
Ours	48	72	**90**	84	**90**	**83**

Table 6.1. Comparison of garment category classification accuracies (in percents) evaluated on the dataset of randomly grasped hanging garments [Kampouris et al., 2016].

True\Pred.	Towels	Shorts	Pants	T-shirts	Shirts
Towels	48	1	3	28	20
Shorts	2	72	3	23	0
Pants	1	1	90	1	7
T-shirts	2	4	3	84	7
Shirts	1	0	3	6	90

Table 6.2. Relative confusion matrix (in percents) for the proposed classifier evaluated on the dataset of randomly grasped hanging garments [Kampouris et al., 2016]. Rows correspond to true garment categories, columns to predictions.

based on RF [Doumanoglou et al., 2014a]. Predictions from multiple viewpoints are aggregated, which can be considered an alternative to our fusion of depths.

Tab. 6.1 shows the comparison of the classification results. Overall accuracy of our method is 83 % which is approximately on par with the current state of the art approach combining CNN and Random Forest (RF) [Kampouris et al., 2016]. Tab. 6.2 shows a relative confusion matrix for our classification results. Its rows correspond to true garment categories, columns to our predictions.

The main source of our failures are towels, for which only 48 % samples are classified correctly, while 48 % samples are classified incorrectly as short-sleeved T-shirts and long-sleeved shirts. We believe that the low performance on towels is caused by the absence of distinguishable parts like collars or sleeves.

The dataset collected by [Doumanoglou et al., 2014b] contains 4 clothing categories: pants, shirts, shorts and T-shirts. Each category is represented by 6 unique items. Each garment was grasped by a robot, lifted up and regrasped for its lowest point (Fig. 6.6b). There is 1 possible lowest point for pants and shirts and 2 lowest points for shorts and T-shirts. Each combination of a garment item and the possible lowest point occurs 20 times. This gives 720 sequences in total, each comprehending data from 40 viewpoints distributed approximately equidistantly on a circle around the garment.

[Doumanoglou et al., 2014a] apply the RF classifier on simple features computed from a depth map. Information from more views is aggregated by the partially observable Markov decision process (POMDP) [Sondik, 1978]. [Doumanoglou et al., 2014b] aggregates directly in the Active Random Forest (ARF), which also selects the next best viewpoint, as described in Sec. 4.3. Only several views are needed for a decision, compared to tens views distributed uniformly around the garment required by our method. On the other hand, time spent by rotating the wrist holding the garment is negligible compared to the other manipulation steps (Sec. 4.6).

We achieve 91 % overall classification accuracy. The results are approximately consistent over all 4 categories (Tab. 6.3). The original works [Doumanoglou et al., 2014a, Doumanoglou et al., 2014b] claim perfect 100 % accuracy. The RF approach, how-

True\Pred.	Shorts	Pants	T-shirts	Shirts
Shorts	88	1	11	0
Pants	4	92	0	4
T-shirts	1	0	97	2
Shirts	2	2	10	86

Table 6.3. Relative confusion matrix (in percents) for the proposed classifier evaluated on the dataset of garments regrasped for the lowest point [Doumanoglou et al., 2014b]. Rows correspond to true garment categories, columns to predictions.

True\Pred.	Shorts	Pants	T-shirts	Shirts
Shorts	83	2	14	1
Pants	2	91	2	5
T-shirts	3	2	88	7
Shirts	0	3	16	81

Table 6.4. Relative confusion matrix (in percents) for the proposed classifier evaluated on the dataset of garments being held for the lowest or unfolding point [Doumanoglou et al., 2014b]. Rows correspond to true garment categories, columns to predictions.

ever, has inferior performance on the randomly grasped garments (Tab. 6.1). Since our method normalizes the input point cloud to a unit ball, it distinguishes the garment category purely on its shape. On contrary, the RF can learn to classify the garments based on their size, since e.g. shorts usually occupy smaller portion of the depth map than pants. Such size based classification may not be a desired behavior. Moreover, thanks to the normalization, our method is arguably easier transferable to different robotic setups, in which a sensor with a different resolution may be used or its distance to the garment may vary over time due to the movement of a robot.

The dataset [Doumanoglou et al., 2014b] contains additional 360 sequences of garments that have been already unfolded partially (Fig. 6.6c) by identifying and grasping a particular point. These data are not used for classification in the original work [Doumanoglou et al., 2014b], because the garment category must be already known before unfolding in practice. We mixed these 360 sequences with 720 sequences for the lowest point. The resulting dataset should therefore be more complex than the lowest point one (Fig. 6.6b), but still simpler than the one with random grasping points (Fig. 6.6a), as the possible states of a garment are limited to a certain extent.

The achieved overall recognition accuracy 86 % proves this hypothesis, compared to 83 % for the random and 91 % for the lowest points. Tab. 6.4 shows a confusion matrix for individual categories.

7. Conclusion

In the thesis, several perception tasks were studied related to a dual-arm robotic manipulation of garments. The proposed work was motivated mainly by the scientific challenges emerging from handling soft objects and materials, reposing in their high deformability. The other motivation are possible future applications in robotic household assistants, which should be capable of laundering, as well as industrial applications in robotic production lines. It should be said, however, that we do not believe in such applications being common in the upcoming decade.

The thesis dealt mainly with three perception tasks originating from garments folding, unfolding and classification. All three tasks are related to the robotic folding scenario. The main contributions of the thesis are summarized in Sec. 7.1. The possible extensions and yet unexplored research topics are outlined in Sec. 7.2.

7.1. Summary

The contents and contributions of the proposed thesis can be summarized as follows:

- Chap. 3 dealt with the pose estimation of a garment for its robotic folding. The garment is segmented from its background automatically with a novel two-stage segmentation method. It is applicable if the statistical properties of the background are known and different from the foreground object. The extracted and simplified contour of the garment is matched a polygonal model, whose vertices are defined manually, but their mutual positions are learned from data. The model is matched with a dynamic programming approach, minimizing a predefined cost. Two cost functions of various complexity were developed and compared. The model matching procedure should be applicable to any polygonal shaped objects.
- Chap. 4 described the integration of the folding method into the CloPeMa pipeline. It is the first such end-to-end pipeline for bringing a heap of crumpled garments of various categories to a fully folded state. Despite the achieved experimental results are promising for such a complex and multistage task, the potential commercial applications would require a significant improvement in terms of robustness and speed.
- Chap. 5 addressed mainly the task of detecting the stacked layers of a folded garment, formulated as its pixels labeling. The novelty reposes in combining image and depth information and encoding them into a specific energy function. Its parameters are determined by a specific version of the EM algorithm derived for our settings. The position and orientation of the folding axis is estimated by generating several possible candidates and examining shapes of the virtually unfolded garment.
- Chap. 6 dealt with the classification of hanging garments represented as 3D point clouds. It employs a CNN based classifier of general objects, utilizing a novel convolution operation defined on local spatial neighborhoods of points. The other techniques involve sub-sampling of the point cloud and increasing sizes of the receptive fields in deeper layers. The network is trained on common 3D objects and used as a global feature extractor in case of hanging garments, followed by the SVM classifier.

7.2. Possible extensions

We identified the following possible future research directions related to the presented methods and to the topic of garments perception and manipulation in general:

- Robustness and speed of the proposed methods would need to be improved for their potential commercial applications. This is also true for the vast majority of other state-of-the-art methods for clothes handling. A possible solution is implementation of various fail-safe strategies for failures detection and recovery. Information coming from force, torque and tactile sensors might be incorporated as well.
- The method for unfolding of folded garments, proposed in Chap. 5, utilizes no model of the garment shape. It is thus generally applicable to various categories of garments. However, incorporating such a model could improve the detection robustness and avoid the heuristics used for a folding axis estimation. Another possible improvement is a generalization to more than two overlapping folded layers.
- The CNN for classification of 3D point clouds, described in Chap. 6, may be modified to perform additional tasks. The final classification layer can be replaced by a regression of the next grasping point for the garment unfolding. Eventually, all fully connected layers could be replaced by the transposed convolutions to obtain a semantic segmentation of the garment parts, e.g. sleeves collars or pockets.
- An interesting task is a highly precise reconstruction of the deformed garment surface. The difficulties repose in the lack of texture, repeating texture patterns, thinness of the material or its possible transparency. The most problematic step from our experience is the surface reconstruction from a dense 3D point cloud, during which discontinuities due to folds and wrinkles must be identified reliably.
- A related research topic is an advanced representation of the garment. Observed deformations of the reconstructed surface might enable to estimate the material properties or to hypothesize about the occluded and covered parts of the garment. Even more ambitious task is tracking of the garment state under manipulation, resulting in proper updates and refinements of the advanced representation. It might be also helpful to model dynamics of the fabric.

Bibliography

[Alenyà et al., 2012] Alenyà, G., Ramisa, A., Moreno-Noguer, F., and Torras, C. (2012). Characterization of textile grasping experiments. In *Proc. ICRA Workshop on Conditions for Replicable Experiments and Performance Comparison in Robotics Research*, pages 1–6. 10

[Berg et al., 2011] Berg, J. v. d., Miller, S., Goldberg, K., and Abbeel, P. (2011). Gravity-based robotic cloth folding. In *Proc. Int. Workshop on Algorithmic Foundations of Robotics (WAFR)*, pages 409–424. 17, 18, 33, 34, 58

[Bersch et al., 2011] Bersch, C., Pitzer, B., and Kammel, S. (2011). Bimanual robotic cloth manipulation for laundry folding. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1413–1419. 10, 12

[Besl and McKay, 1992] Besl, P. J. and McKay, N. D. (1992). Method for registration of 3-D shapes. In *Sensor Fusion IV: Control Paradigms and Data Structures*, volume 1611, pages 586–607. 66, 91

[Bilmes et al., 1998] Bilmes, J. A. et al. (1998). A gentle tutorial of the EM algorithm and its application to parameter estimation for Gaussian mixture and hidden Markov models. *Int. Computer Science Institute*, 4(510):126. 56, 57

[Boykov and Kolmogorov, 2004] Boykov, Y. and Kolmogorov, V. (2004). An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137. 53, 56

[Breiman, 2001] Breiman, L. (2001). Random forests. *Machine Learning*, 45(1):5–32. 12, 43, 91

[Canny, 1986] Canny, J. (1986). A computational approach to edge detection. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 8(6):679–698. 11, 15

[Chang et al., 2015] Chang, A. X., Funkhouser, T., Guibas, L., Hanrahan, P., Huang, Q., Li, Z., Savarese, S., Savva, M., Song, S., Su, H., et al. (2015). ShapeNet: An information-rich 3D model repository. *arXiv:1512.03012*. 73

[Chang and Lin, 2011] Chang, C.-C. and Lin, C.-J. (2011). LIBSVM: A library for support vector machines. *ACM Trans. on Intelligent Systems and Technology*, 2(3):27. 75

[Cho et al., 2014] Cho, K., Van Merriënboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proc. Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734. 71, 91

[Corona et al., 2018] Corona, E., Alenyà, G., Gabas, A., and Torras, C. (2018). Active garment recognition and target grasping point detection using deep learning. *Pattern Recognition*, 74:629–641. 12, 19

[Cortes and Vapnik, 1995] Cortes, C. and Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3):273–297. 9, 65, 91

[Cusumano-Towner et al., 2011] Cusumano-Towner, M., Singh, A., Miller, S., O'Brien, J. F., and Abbeel, P. (2011). Bringing clothing into desired configurations with limited perception. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3893–3900. 10, 15

[Dalal and Triggs, 2005] Dalal, N. and Triggs, B. (2005). Histograms of oriented gradients for human detection. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, volume 1, pages 886–893. 14, 91

[Dempster et al., 1977] Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *J. of the Royal Statistical Society*, pages 1–38. 56, 57, 91

[Doumanoglou et al., 2014a] Doumanoglou, A., Kargakos, A., Kim, T.-K., and Malassiotis, S. (2014a). Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 987–993. 12, 76

[Doumanoglou et al., 2014b] Doumanoglou, A., Kim, T.-K., Zhao, X., and Malassiotis, S. (2014b). Active random forests: An application to autonomous unfolding of clothes. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 644–658. 12, 43, 45, 46, 75, 76, 77, 91

[Doumanoglou, Stria et al., 2016] Doumanoglou, A., Stria, J., Peleka, G., Mariolis, I., Petrík, V., Kargakos, A., Wagner, L., Hlaváč, V., Kim, T.-K., and Malassiotis, S. (2016). Folding clothes autonomously: A complete pipeline. *IEEE Trans. on Robotics*, 32(6):1461–1478. 5, 39

[Duda et al., 2000] Duda, R. O., Hart, P. E., and Stork, D. G. (2000). *Pattern Classification, 2nd ed.* Wiley. 23, 26, 56

[Estevez et al., 2017] Estevez, D., Fernandez-Fernandez, R., Victores, J. G., and Balaguer, C. (2017). Improving and evaluating robotic garment unfolding: A garment-agnostic approach. In *Proc. Int. Conf. on Autonomous Robot Systems and Competitions (ICARSC)*, pages 210–215. 16

[Estevez et al., 2016] Estevez, D., Victores, J. G., Morante, S., and Balaguer, C. (2016). Towards robotic garment folding: A vision approach for fold detection. In *Proc. Int. Conf. on Autonomous Robot Systems and Competitions (ICARSC)*, pages 188–192. 16, 52

[Farin, 2014] Farin, G. (2014). *Curves and surfaces for computer-aided geometric design: A practical guide.* Elsevier. 18

[Felzenszwalb and Huttenlocher, 2004] Felzenszwalb, P. F. and Huttenlocher, D. P. (2004). Efficient graph-based image segmentation. *Int. J. of Computer Vision*, 59(2):167–181. 9, 42

[Fischler and Bolles, 1981] Fischler, M. A. and Bolles, R. C. (1981). Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. of the ACM*, 24(6):381–395. 14, 42, 53, 91

[Foresti and Pellegrino, 2004] Foresti, G. L. and Pellegrino, F. A. (2004). Automatic visual recognition of deformable objects for grasping and manipulation. *IEEE Trans. on Systems, Man, and Cybernetics*, 34(3):325–333. 10

[Friedman et al., 2001] Friedman, J., Hastie, T., and Tibshirani, R. (2001). *The elements of statistical learning*. Springer. 13, 17, 74, 75, 91

[Gabas et al., 2016] Gabas, A., Corona, E., Alenyà, G., and Torras, C. (2016). Robot-aided cloth classification using depth information and CNNs. In *Proc. Articulated Motion and Deformable Objects (AMDO)*, pages 16–23. 12, 19, 73

[Gabas and Kita, 2017] Gabas, A. and Kita, Y. (2017). Physical edge detection in clothing items for robotic manipulation. In *Proc. Int. Conf. on Advanced Robotics (ICAR)*, pages 524–529. 15

[Gonzalez et al., 2009] Gonzalez, R. C., Woods, R. E., and Eddins, S. L. (2009). *Digital Image Processing Using MATLAB, 2nd ed.* Gatesmark. 24, 55, 91

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT Press. 70, 71, 91

[Gupta et al., 2014] Gupta, S., Girshick, R., Arbeláez, P., and Malik, J. (2014). Learning rich features from RGB-D images for object detection and segmentation. In *Proc. European Conf. on Computer Vision (ECCV)*, pages 345–360. 18

[Hamajima and Kakikura, 1998] Hamajima, K. and Kakikura, M. (1998). Planning strategy for task untangling laundry — isolating clothes from a washed mass. *J. of Robotics and Mechatronics*, 10(3):244–251. 9, 14

[Hamajima and Kakikura, 2000] Hamajima, K. and Kakikura, M. (2000). Planning strategy for task of unfolding clothes. *Robotics and Autonomous Systems*, 32(2):145–152. 9, 14

[Hana et al., 2018] Hana, X.-F., Jin, J. S., Xie, J., Wang, M.-J., and Jiang, W. (2018). A comprehensive review of 3D point cloud descriptors. *arXiv:1802.02297*. 68

[Hata et al., 2008] Hata, S., Hiroyasu, T., Hayashi, J., Hojoh, H., and Hamada, T. (2008). Robot system for cloth handling. In *Proc. Annual Conf. of IEEE Industrial Electronics Society (IECON)*, pages 3449–3454. 3, 10

[Huber, 1973] Huber, P. J. (1973). Robust regression: Asymptotics, conjectures and Monte Carlo. *The Annals of Statistics*, pages 799–821. 58

[Ioffe and Szegedy, 2015] Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proc. Int. Conf. on Machine Learning (ICML)*, pages 448–456. 71

[Jaderberg et al., 2015] Jaderberg, M., Simonyan, K., Zisserman, A., et al. (2015). Spatial transformer networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2017–2025. 19

[Jang et al., 2017] Jang, E., Gu, S., and Poole, B. (2017). Categorical reparameterization with Gumbel-softmax. In *Proc. Int. Conf. on Learning Representations (ICLR)*. 70

[Kampouris et al., 2016] Kampouris, C., Mariolis, I., Peleka, G., Skartados, E., Kargakos, A., Triantafyllou, D., and Malassiotis, S. (2016). Multi-sensorial and explorative recognition of garments and their material properties in unconstrained environment. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1656–1663. 12, 19, 75, 76

[Kaneko and Kakikura, 2001] Kaneko, M. and Kakikura, M. (2001). Planning strategy for putting away laundry — isolating and unfolding task. In *Proc. IEEE Int. Symp. on Assembly and Task Planning*, pages 429–434. 14

[Kass et al., 1988] Kass, M., Witkin, A., and Terzopoulos, D. (1988). Snakes: Active contour models. *Int. J. of Computer Vision*, 1(4):321–331. 11

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv:1412.6980*. 74

[Kita et al., 2011] Kita, Y., Kanehiro, F., Ueshiba, T., and Kita, N. (2011). Clothes handling based on recognition by strategic observation. In *Proc. IEEE/RAS Int. Conf. on Humanoid Robots (Humanoids)*, pages 53–58. 10

[Kita and Kita, 2002] Kita, Y. and Kita, N. (2002). A model-driven method of estimating the state of clothes for manipulating it. In *Proc. IEEE Workshop on Applications of Computer Vision (WACV)*, pages 63–69. 10

[Kita et al., 2010] Kita, Y., Neo, E. S., Ueshiba, T., and Kita, N. (2010). Clothes handling using visual recognition in cooperation with actions. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2710–2715. 11

[Kita et al., 2004a] Kita, Y., Saito, F., and Kita, N. (2004a). A deformable model driven method for handling clothes. In *Proc. IEEE Int. Conf. on Pattern Recognition (ICPR)*, pages 243–247. 10

[Kita et al., 2004b] Kita, Y., Saito, F., and Kita, N. (2004b). A deformable model driven visual method for handling clothes. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3889–3895. 10

[Kita et al., 2009] Kita, Y., Ueshiba, T., Neo, E. S., and Kita, N. (2009). Clothes state recognition using 3D observed data. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1220–1225. 10

[Koenderink and Van Doorn, 1992] Koenderink, J. J. and Van Doorn, A. J. (1992). Surface shape and curvature scales. *Image and Vision Computing*, 10(8):557–564. 13

[Koller et al., 1995] Koller, T. M., Gerig, G., Szekely, G., and Dettwiler, D. (1995). Multiscale detection of curvilinear structures in 2-D and 3-D image data. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, pages 864–869. 42

[Kolmogorov and Zabin, 2004] Kolmogorov, V. and Zabin, R. (2004). What energy functions can be minimized via graph cuts? *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(2):147–159. 55

[Krizhevsky et al., 2012] Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). ImageNet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1097–1105. 12, 67

[Kuffner and LaValle, 2000] Kuffner, J. J. and LaValle, S. M. (2000). RRT-connect: An efficient approach to single-query path planning. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 995–1001. 34

[Lander, 1999] Lander, J. (1999). Devil in the blue-faceted dress: Real-time cloth animation. *Game Developer Magazine*, 21. 10, 16

[Lazebnik et al., 2006] Lazebnik, S., Schmid, C., and Ponce, J. (2006). Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2169–2178. 11, 91

[Le et al., 2013] Le, T.-H.-L., Jilich, M., Landini, A., Zoppi, M., Zlatanov, D., and Molfino, R. (2013). On the development of a specialized flexible gripper for garment handling. *Int. J. of Automation and Control Engineering*, 1(2):255–259. 41

[Li et al., 2018] Li, Y., Bu, R., Sun, M., Wu, W., Di, X., and Chen, B. (2018). PointCNN: Convolution on X-transformed points. In *Advances in Neural Information Processing Systems (NIPS)*, pages 820–830. 20

[Li et al., 2014a] Li, Y., Chen, C.-F., and Allen, P. K. (2014a). Recognition of deformable object category and pose. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 5558–5564. 11

[Li et al., 2014b] Li, Y., Wang, Y., Case, M., Chang, S.-F., and Allen, P. K. (2014b). Real-time pose estimation of deformable objects using a volumetric approach. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 1046–1052. 11

[Li et al., 2015a] Li, Y., Xu, D., Yue, Y., Wang, Y., Chang, S.-F., Grinspun, E., and Allen, P. K. (2015a). Regrasping and unfolding of garments using predictive thin shell modeling. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1382–1388. 11

[Li et al., 2015b] Li, Y., Yue, Y., Xu, D., Grinspun, E., and Allen, P. K. (2015b). Folding deformable objects using predictive simulation and trajectory optimization. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 6000–6006. 18, 34

[Ling and Jacobs, 2007] Ling, H. and Jacobs, D. W. (2007). Shape classification using the inner-distance. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 29(2):286–299. 16, 47

[Lowe, 2004] Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *Int. J. of Computer Vision*, 60(2):91–110. 9, 91

[Maitin-Shepard et al., 2010] Maitin-Shepard, J., Cusumano-Towner, M., Lei, J., and Abbeel, P. (2010). Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2308–2315. 10, 14, 15, 50

[Mariolis and Malassiotis, 2013] Mariolis, I. and Malassiotis, S. (2013). Matching folded garments to unfolded templates using robust shape analysis techniques. In *Proc. Int. Conf. on Computer Analysis of Images and Patterns*, pages 193–200. 16

[Mariolis and Malassiotis, 2015] Mariolis, I. and Malassiotis, S. (2015). Modelling folded garments by fitting foldable templates. *Machine Vision Applications*, 26(4):549–560. 15, 16, 52

[Mariolis et al., 2015] Mariolis, I., Peleka, G., Kargakos, A., and Malassiotis, S. (2015). Pose and category recognition of highly deformable objects using deep learning. In *Proc. Int. Conf. on Advanced Robotics (ICAR)*, pages 655–662. 12, 19, 73, 75, 76

[Maturana and Scherer, 2015] Maturana, D. and Scherer, S. (2015). VoxNet: A 3D convolutional neural network for real-time object recognition. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 922–928. 19

[Miller et al., 2011] Miller, S., Fritz, M., Darrell, T., and Abbeel, P. (2011). Parametrized shape models for clothing. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 4861–4868. 14, 18, 37

[Miller et al., 2012] Miller, S., Van Den Berg, J., Fritz, M., Darrell, T., Goldberg, K., and Abbeel, P. (2012). A geometric approach to robotic laundry folding. *Int. J. of Robotics Research*, 31(2):249–267. 18

[Muja and Lowe, 2014] Muja, M. and Lowe, D. G. (2014). Scalable nearest neighbor algorithms for high dimensional data. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 36(11):2227–2240. 67

[Newcombe et al., 2011] Newcombe, R. A., Izadi, S., Hilliges, O., Molyneaux, D., Kim, D., Davison, A. J., Kohi, P., Shotton, J., Hodges, S., and Fitzgibbon, A. (2011). Kinectfusion: Real-time dense surface mapping and tracking. In *Proc. IEEE Int. Symp. on Mixed and Augmented Reality (ISMAR)*, pages 127–136. 16, 66, 91

[Ojala et al., 1996] Ojala, T., Pietikäinen, M., and Harwood, D. (1996). A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59. 13, 14, 91

[Orchard and Bouman, 1991] Orchard, M. and Bouman, C. (1991). Color quantization of images. *IEEE Trans. on Signal Processing*, 39(12):2677–2690. 22

[Paszke et al., 2017] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in PyTorch. In *Advances in Neural Information Processing Systems (NIPS)*. 67, 69

[Perez and Vidal, 1994] Perez, J.-C. and Vidal, E. (1994). Optimum polygonal approximation of digitized curves. *Pattern Recognition Letters*, 15(8):743–750. 24, 58

[Petrík et al., 2018] Petrík, V., Cmíral, J., Smutný, V., Krsek, P., and Hlaváč, V. (2018). Automatic material properties estimation for the physics-based robotic garment folding. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 7449–7454. 18

[Petrík et al., 2015] Petrík, V., Smutný, V., Krsek, P., and Hlaváč, V. (2015). Robotic garment folding: Precision improvement and workspace enlargement. In *Towards Autonomous Robotic Systems (TAROS)*, pages 204–215. 17

[Petrík et al., 2016] Petrík, V., Smutný, V., Krsek, P., and Hlaváč, V. (2016). Physics-based model of a rectangular garment for robotic folding. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 951–956. 17, 32, 34

[Petrík et al., 2017] Petrík, V., Smutný, V., Krsek, P., and Hlaváč, V. (2017). Single arm robotic garment folding path generation. *Advanced Robotics*, 31(23-24):1325–1337. 18, 32, 34

[Qi et al., 2017a] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. (2017a). PointNet: Deep learning on point sets for 3D classification and segmentation. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 652–660. 19, 20, 67, 69, 71

[Qi et al., 2017b] Qi, C. R., Yi, L., Su, H., and Guibas, L. J. (2017b). PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *Advances in Neural Information Processing Systems (NIPS)*, pages 5105–5114. 20, 70

[Quigley et al., 2009] Quigley, M., Conley, K., Gerkey, B. P., Faust, J., Foote, T., Leibs, J., Wheeler, R., and Ng, A. Y. (2009). ROS: An open-source robot operating system. In *Proc. ICRA Workshop on Open Source Software*. 42, 91

[Ramisa et al., 2012] Ramisa, A., Alenyà, G., Moreno-Noguer, F., and Torras, C. (2012). Using depth and appearance features for informed robot grasping of highly wrinkled clothes. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1703–1708. 9

[Ramisa et al., 2013] Ramisa, A., Alenyà, G., Moreno-Noguer, F., and Torras, C. (2013). FINDDD: A fast 3D descriptor to characterize textiles for robot manipulation. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 824–830. 9, 91

[Ramisa et al., 2016] Ramisa, A., Alenyà, G., Moreno-Noguer, F., and Torras, C. (2016). A 3D descriptor to detect task-oriented grasping points in clothing. *Pattern Recognition*, 60:936–948. 9, 13

[Rother et al., 2004] Rother, C., Kolmogorov, V., and Blake, A. (2004). GrabCut – interactive foreground extraction using iterated graph cuts. *ACM Trans. on Graphics*, 23(3):309–314. 23, 52, 54, 55

[Rusu et al., 2009] Rusu, R. B., Blodow, N., and Beetz, M. (2009). Fast point feature histograms (FPFH) for 3D registration. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 3212–3217. 13

[Schmidt et al., 2016] Schmidt, A., Sun, L., Aragon-Camarasa, G., and Siebert, J. P. (2016). The calibration of the pan-tilt units for the active stereo head. In *Image Processing and Communications Challenges 7*, volume 389 of *Advances in Intelligent Systems and Computing*, pages 213–221. Springer. 41

[Simo and Fox, 1989] Simo, J. and Fox, D. (1989). On a stress resultant geometrically exact shell model. Part I: Formulation and optimal parametrization. *Computer Methods in Applied Mechanics and Engineering*, 72(3):267–304. 18

[Smíšek et al., 2013] Smíšek, J., Jančošek, M., and Pajdla, T. (2013). 3D with Kinect. In *Consumer Depth Cameras for Computer Vision*, pages 3–25. Springer. 41

[Sondik, 1978] Sondik, E. J. (1978). The optimal control of partially observable Markov processes over the infinite horizon: Discounted costs. *Operations Research*, 26(2):282–304. 12, 76, 91

[Šonka et al., 2014] Šonka, M., Hlaváč, V., and Boyle, R. (2014). *Image processing, analysis, and machine vision, 4rd ed.* Cengage Learning. 16, 52

[Spivak, 1970] Spivak, M. (1970). *A comprehensive introduction to differential geometry.* Brandeis University. 67

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. *J. of Machine Learning Research*, 15(1):1929–1958. 71

[Stria and Hlaváč, 2018] Stria, J. and Hlaváč, V. (2018). Classification of hanging garments using learned features extracted from 3D point clouds. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 5307–5312. 6, 65

[Stria et al., 2017] Stria, J., Petrík, V., and Hlaváč, V. (2017). Model-free approach to garments unfolding based on detection of folded layers. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3274–3280. 5, 51

[Stria et al., 2014a] Stria, J., Průša, D., and Hlaváč, V. (2014a). Polygonal models for clothing. In *Towards Autonomous Robotic Systems (TAROS)*, pages 173–184. 5, 21, 28

[Stria et al., 2014b] Stria, J., Průša, D., Hlaváč, V., Wagner, L., Petrík, V., Krsek, P., and Smutný, V. (2014b). Garment perception and its folding using a dual-arm robot. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 64–67. 5, 21, 28

[Su et al., 2015] Su, H., Maji, S., Kalogerakis, E., and Learned-Miller, E. (2015). Multi-view convolutional neural networks for 3D shape recognition. In *Proc. IEEE Int. Conf. on Computer Vision (ICCV)*, pages 945–953. 19

[Şucan et al., 2012] Şucan, I. A., Moll, M., and Kavraki, L. E. (2012). The open motion planning library. *IEEE Robotics and Automation Mag.*, 19(4):72–82. 34, 42, 91

[Sun et al., 2013] Sun, L., Aragon-Camarasa, G., Cockshott, P., Rogers, S., and Siebert, J. P. (2013). A heuristic-based approach for flattening wrinkled clothes. In *Towards Autonomous Robotic Systems (TAROS)*, pages 148–160. 16

[Sun et al., 2015] Sun, L., Aragon-Camarasa, G., Rogers, S., and Siebert, J. P. (2015). Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 185–192. 17

[Sun et al., 2017] Sun, L., Aragon-Camarasa, G., Rogers, S., Stolkin, R., and Siebert, J. P. (2017). Single-shot clothing category recognition in free-configurations with application to autonomous clothes sorting. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 6699–6706. 13

[Sun et al., 2016] Sun, L., Rogers, S., Aragon-Camarasa, G., and Siebert, J. P. (2016). Recognising the clothing categories from free-configuration using Gaussian-process-based interactive perception. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 2464–2470. 13

[Timoshenko, 1983] Timoshenko, S. (1983). *History of strength of materials: With a brief account of the history of theory of elasticity and theory of structures*. Courier Corporation. 18

[Triantafyllou et al., 2016] Triantafyllou, D., Mariolis, I., Kargakos, A., Malassiotis, S., and Aspragathos, N. (2016). A geometric approach to robotic unfolding of garments. *Robotics and Autonomous Systems*, 75:233–243. 15, 16, 52

[Varma and Zisserman, 2005] Varma, M. and Zisserman, A. (2005). A statistical approach to texture classification from single images. *Int. J. of Computer Vision*, 62(1-2):61–81. 14

[Veličković et al., 2017] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., and Bengio, Y. (2017). Graph attention networks. *arXiv:1710.10903*. 20

[Wang et al., 2011] Wang, P. C., Miller, S., Fritz, M., Darrell, T., and Abbeel, P. (2011). Perception for the manipulation of socks. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 4877–4884. 14

[Wang et al., 2018] Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., and Solomon, J. M. (2018). Dynamic graph CNN for learning on point clouds. *arXiv:1801.07829*. 20

[Willimon et al., 2011a] Willimon, B., Birchfield, S., and Walker, I. (2011a). Classification of clothing using interactive perception. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 1862–1868. 9, 10, 11

[Willimon et al., 2011b] Willimon, B., Birchfield, S., and Walker, I. (2011b). Model for unfolding laundry using interactive perception. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 4871–4876. 16

[Willimon et al., 2012] Willimon, B., Hickson, S., Walker, I., and Birchfield, S. (2012). An energy minimization approach to 3D non-rigid deformable surface estimation using RGBD data. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2711–2717. 11

[Willimon et al., 2013a] Willimon, B., Walker, I., and Birchfield, S. (2013a). 3D non-rigid deformable surface estimation without feature correspondence. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 646–651. 11

[Willimon et al., 2013b] Willimon, B., Walker, I., and Birchfield, S. (2013b). A new approach to clothing classification using mid-level layers. In *Proc. IEEE Int. Conf. on Robotics and Automation (ICRA)*, pages 4271–4278. 13

[Wu et al., 2015] Wu, Z., Song, S., Khosla, A., Yu, F., Zhang, L., Tang, X., and Xiao, J. (2015). 3D ShapeNets: A deep representation for volumetric shapes. In *Proc. IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 1912–1920. 19

[Yamazaki, 2014] Yamazaki, K. (2014). Grasping point selection on an item of crumpled clothing based on relational shape description. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3123–3128. 9

[Yamazaki, 2017] Yamazaki, K. (2017). A method of classifying crumpled clothing based on image features derived from clothing fabrics and wrinkles. *Autonomous Robots*, 41(4):865–879. 13

[Yamazaki and Inaba, 2013] Yamazaki, K. and Inaba, M. (2013). Clothing classification using image features derived from clothing fabrics, wrinkles and cloth overlaps. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 2710–2717. 13, 14

[Yamazaki et al., 2012] Yamazaki, K., Ueda, R., Nozawa, S., Kojima, M., Okada, K., Matsumoto, K., Ishikawa, M., Shimoyama, I., and Inaba, M. (2012). Home-assistant robot for an aging society. *Proc. of the IEEE*, 100(8):2429–2441. 2

[Yosinski et al., 2014] Yosinski, J., Clune, J., Bengio, Y., and Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in Neural Information Processing Systems (NIPS)*, pages 3320–3328. 74

A. Acronyms

6DOF six degrees of freedom
ARF Active Random Forest [Doumanoglou et al., 2014b]
BoW bag-of-words [Lazebnik et al., 2006]
CloPeMa Clothes Perception and Manipulation project[1]
CNN convolutional neural network [Goodfellow et al., 2016]
DoG difference of Gaussians [Gonzalez et al., 2009]
GMM Gaussian mixture model [Friedman et al., 2001]
EM expectation-maximization algorithm [Dempster et al., 1977]
FINDDD Fast Integral Normal 3D descriptor [Ramisa et al., 2013]
GPU graphics processing unit
GRU gated recurrent unit [Cho et al., 2014]
HMM hidden Markov model [Friedman et al., 2001]
HOG histogram of oriented gradients [Dalal and Triggs, 2005]
ICP Iterative Closest Point [Besl and McKay, 1992]
k-**NN** k-nearest neighbors [Friedman et al., 2001]
LBP local binary patterns [Ojala et al., 1996]
OMPL Open Motion Planning Library [Şucan et al., 2012]
POMDP partially observable Markov decision process [Sondik, 1978]
RANSAC random sample consesus [Fischler and Bolles, 1981]
RBF radial basis function [Friedman et al., 2001]
RNN recurrent neural network [Goodfellow et al., 2016]
ReLU rectified linear unit [Goodfellow et al., 2016]
RF Random Forest [Breiman, 2001]
RGB 3-channel (red, green and blue) image
RGBD 4-channel (red, green, blue and depth) data
ROS Robotic Operating System [Quigley et al., 2009]
SIFT scale-invariant feature transform [Lowe, 2004]
SVM support vector machine [Cortes and Vapnik, 1995]
TSDF truncated signed distance function [Newcombe et al., 2011]

[1]CloPeMa: `www.clopema.eu`

B. Author's publications

We list all publications of the thesis author. The publications are split to those related to garments perception and those related to previous author's work on recognition of handwritten mathematical formulas. Each publication is accompanied by the number of citations in the Web of Science (WoS) and Google Scholar, excluding auto-citations, as captured in September 2019.

Publications related to the thesis

Impacted journal articles

- Doumanoglou, A., **Stria, J.**, Peleka, G., Mariolis, I., Petrík, V., Kargakos, A., Wagner, L., Hlaváč, V., Kim, T.-K., and Malassiotis, S. (2016). Folding clothes autonomously: A complete pipeline. *IEEE Trans. on Robotics*, 32(6):1461–1478.
  Authorship: [20%–**20%**–10%–10%–10%–9%–9%–4%–4%–4%] (*A. Doumanoglou and J. Stria are main authors of the article with an equal contribution. Their surnames are ordered alphabetically in the list of authors.*)
  Citations: 8 (WoS), 12 (Scholar)

Publications excerpted by WoS

- **Stria, J.** and Hlaváč, V. (2018). Classification of hanging garments using learned features extracted from 3D point clouds. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 5307–5312.
  Authorship: [**80%**–20%]
  Citations: 1 (WoS), 1 (Scholar)
- **Stria, J.**, Petrík, V., and Hlaváč, V. (2017). Model-free approach to garments unfolding based on detection of folded layers. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 3274–3280.
  Authorship: [**60%**–20%–20%]
  Citations: 0
- **Stria, J.**, Průša, D., Hlaváč, V., Wagner, L., Petrík, V., Krsek, P., and Smutný, V. (2014). Garment perception and its folding using a dual-arm robot. In *Proc. IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, pages 64–67.
  Authorship: [**50%**–10%–10%–10%–10%–5%–5%]
  Citations: 26 (WoS), 37 (Scholar)

Other conference publications

- **Stria, J.**, Průša, D., and Hlaváč, V. (2014). Polygonal models for clothing. In *Towards Autonomous Robotic Systems (TAROS)*, pages 173–184.
  Authorship: [**50%**–25%–25%]
  Citations: 8 (WoS), 15 (Scholar)

Additional publications

Publications excerpted by WoS

- **Stria, J.**, Bresler, M., Průša, D., and Hlaváč, V. (2012). MfrDB: Database of annotated on-line mathematical formulae. In *Proc. International Conference on Frontiers in Handwriting Recognition (ICFHR)*, pages 540-545.
  Authorship: [**60%**–20%–10%–10%]
  Citations: 4 (WoS), 8 (Scholar)

Other conference publications

- **Stria, J.**, Průša, D., and Hlaváč, V. (2014). Combining structural and statistical approach to online recognition of handwritten mathematical formulas. In *Proc. Computer Vision Winter Workshop (CVWW)*, pages 103-109.
  Authorship: [**50%**–40%–10%]
  Citations: 1 (WoS), 2 (Scholar)
- **Stria, J.** and Průša, D. (2011). Web application for recognition of mathematical formulas. In *Proc. Conference on Theory and Practice of Information Technologies (ITAT)*, pages 47-54.
  Authorship: [**50%**–50%]
  Citations: 2 (WoS), 3 (Scholar)