ROBOTIC SYSTEM FOR GARMENT PERCEPTION AND MANIPULATION

DAVID ESTÉVEZ FERNÁNDEZ

A dissertation submitted by in partial fulfillment of the requirements for the degree of Doctor of Philosophy in

Electrical Engineering, Electronics and Automation

Universidad Carlos III de Madrid

Advisors:

CARLOS BALAGUER BERNALDO DE QUIRÓS

JUAN CARLOS GONZÁLEZ VÍCTORES

Tutor:

JUAN CARLOS GONZÁLEZ VÍCTORES

Leganés, Madrid, Octubre 2020

To those who sacrificed their precious time doing laundry.

ACKNOWLEDGMENTS

Although this thesis is written in English, let me use Spanish –my mother tongue– for the most part of this section, as it is also the mother tongue of those I would like to thank. I will switch back to English at the end of this section to thank my international colleagues before resuming the rest of the thesis.

Esta tesis pone fin a un viaje de cuatros años –duro, pero bonito a la vez– a lo largo del cual, y por suerte, me ha acompañado mucha gente que de alguna forma u otra lo ha hecho posible. Gracias a ellos puedo decir no sólo que he llegado al final de este camino, sino que además soy mejor persona que cuando lo comencé. Por ello me gustaría dedicar esta sección a recordar y agradecer a todas esas personas su apoyo.

En primer lugar, me gustaría dar las gracias a mis padres por su apoyo incondicional, no sólo durante estos años de doctorado, sino desde siempre. Desde que era pequeño jamás han puesto límites a mi curiosidad –cualidad imprescindible para ser un buen investigador–, sino que han sabido cultivarla y celebrarla, a pesar de los muchos aparatos y ordenadores rotos por el camino, y les estaré siempre agradecido por ello. Por supuesto, tengo que agradecer también su apoyo al resto mi familia, especialmente a mi hermano y a mis abuelos, tanto a los que están aquí para ver este logro, como a los que, estén donde estén, seguro que están orgullosos de su nieto.

En este viaje por un territorio oscuro y desconocido que es la tesis, a veces es muy necesaria una luz que alumbre el camino a seguir. Por suerte, no sólo he contado con una luz, sino que he tenido el privilegio de tener dos: mis tutores Carlos y Juan. Ha sido un orgullo para mi trabajar codo a codo con ellos y poder contar con su sabiduría durante todo este tiempo. Me habéis enseñado muchísimo y os lo agradezco.

Afortunadamente también, no he recorrido este camino en solitario, sino que durante este tiempo he compartido despacho y laboratorio con gente maravillosa, que ha estado ahí siempre que he tenido algún problema o me ha hecho falta algún favor, y que sin duda han hecho que este largo viaje resulte mucho más ameno. Con ellos y ellas he compartido risas, penas, conferencias, seminarios y madrugones para ir a encuentros doctorales. Por ello me gustaría agradecer a mis compañeros y compañe-

ras, a los que todavía siguen y a los que dejaron el laboratorio en busca de nuevas aventuras, el calor, el buen ambiente de trabajo, y la ayuda prestada, ya sea poniendo a punto los robots, preparando artículos, o salvándome de líos burocráticos (por poner algunos ejemplos).

De entre todas las personas que habitan el laboratorio, no podría olvidarme de dar las gracias en especial a mis queridos Wondermonguers: Raúl, Raúl, Elisabeth, Olaya, Noé, Alice, Marcos y Álvaro. Por todo lo que hemos pasado juntos, todas nuestras aventuras, y porque me habéis hecho sentirme parte de una (particular) familia, gracias.

Durante este largo viaje no todo ha sido trabajo, y también me gustaría agradecer a otras personas el tiempo que han compatido conmigo, haciendo este viaje más llevadero.

A mi gran amigo Jorfru y al resto de mis amigos makers –semaforistas y no semaforistas– y por supuesto a toda la gente maravillosa de ASROB y UC3Music, por estar siempre ahí y haber compartido conmigo durante estos años la pasión por imaginar, crear y construir todo tipo de cachivaches y locuras. Gracias por todas las ferias y viajes que hemos hecho juntos, y por las cervezas de después de las reuniones.

A Sergio, Javi y Javi –también conocidos como Radio X– por dejarme formar parte de esta gran banda. La música es y será siempre para mi una parte muy importante de la vida, que nos permite sentir y expresar todo tipo de emociones y, durante unas horas, poder olvidar todo lo que nos preocupa, simplemente disfrutando de ella. Gracias por haber compartido la música conmigo y por haber hecho de mí un mejor músico.

A Cristina, por sus valiosísimos consejos sobre las figuras de esta tesis, y por su apoyo emocional durante los altibajos del camino, que sin duda han ayudado a que (a empujones) haya conseguido terminarlo. Ha sido una suerte conocerte, y espero que pronto tú también consigas llegar al final de tu camino.

And finally, and this time in English, I would like to thank all the wonderful people that I have met at CTU Prague during my research stay, and specially Václav for inviting me to be part of his team, and Vladimír for allowing me to collaborate with him. Děkujeme! I would also like to thank Mircea, Federica, Josselin and Kate for all the good times we shared during my time in Prague. You made me feel a bit more at home in a foreign land, thank you a lot!

PUBLICATIONS

The following works have been published and/or submitted for publication and are included partly or wholly in this thesis:

[1] David Estevez and Juan G Victores. *FoldNet - Hanging Garments Dataset*. https://doi.org/10.5281/zenodo.3932102. Version 1.0. 2020. DOI: 10.5281/zenodo.3932102.

[2] David Estevez, Juan G Victores, and Carlos Balaguer. "Future Trends in Perception and Manipulation for Unfolding and Folding Garments." In: *RoboCity16: Open Conference on Future Trends in Robotics*. 2016. ISBN: 978-84-608-8452-1.

[3] David Estevez, Juan G Victores, Santiago Morante, and Carlos Balaguer. "Towards robotic garment folding: A vision approach for fold detection." In: *2016 International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2016, pp. 188–192. DOI: 10.1109/ICARSC.2016.65.

[4] David Estevez, Raul Fernandez-Fernandez, Juan G Victores, and Carlos Balaguer. "Improving and evaluating robotic garment unfolding: A garment-agnostic approach." In: *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2017, pp. 210–215. DOI: 10.1109/ICARSC.2017.7964077.

[5] David Estevez, Juan G Victores, Raul Fernandez-Fernandez, and Carlos Balaguer. "Robotic ironing with 3D perception and force/torque feedback in household environments." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 6484–6489. DOI: 10.1109/IROS.2017.8206556.

[6] David Estevez, Raul Fernandez-Fernandez, Juan G Victores, and Carlos Balaguer. "Robotic ironing with a humanoid robot using human tools." In: *2017 IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2017, pp. 134–139. DOI: 10.1109/ICARSC.2017.7964065.

[7] David Estevez, Juan G Victores, Raul Fernandez-Fernandez, and Carlos Balaguer. "Towards Clothes Hanging via Cloth Simulation and Deep Convolutional Networks." In: *10th EUROSIM2019 Congress*. EUROSIM Federation of European Simulation Societies. 2019.

[8] David Estevez, Juan G Victores, Raul Fernandez-Fernandez, and Carlos Balaguer. "Enabling garment-agnostic laundry tasks for a Robot Household Companion." In: *Robotics and Autonomous Systems* 123 (2020), p. 103330. DOI: 10.1016/j.robot.2019.103330.

The material from these sources included in this thesis is not singled out with typographic means and references. The author of this thesis states that his role in all the aforementioned materials was first author, conducting the corresponding research and writing the paper.

Publication [7] is wholly contained in Chapter 3, with the corresponding experiments reported in Chapter 7 and conclusions in Chapter 8.

Publications [2–4, 8] are wholly contained in Chapter 4, with the corresponding experiments reported in Chapter 7 and conclusions in Chapter 8.

Publications [5, 6, 8] are wholly contained in Chapter 5, with the corresponding experiments reported in Chapter 7 and conclusions in Chapter 8.

Publication [1] corresponds to a dataset, whose description and creation methodology are contained in Chapter 3. Whenever material from this source is included in this thesis, it is singled out with typographic means and an explicit reference. The roles of the thesis author in this publication were gathering the dataset training examples, processing the data and curating the dataset.

OTHER RESEARCH MERITS

This section includes a list of research merits other than the published articles or other contributions in the Published Content and Contributions section.

[1] David Estevez, Juan G Victores, and Carlos Balaguer. "A Lightweight Finite State Machine C++ Library aimed at Seamless Integration with Robotic Middlewares." Towards Humanoid Robots Operating Systems Workshop at IEEE/RAS International Conference on Humanoid Robots (Humanoids). 2016.

[2] David Estevez, Juan G Victores, and Carlos Balaguer. "A New Generation of Entertainment Robots Enhanced with Augmented Reality." In: *RoboCity16: Open Conference on Future Trends in Robotics*. 2016. ISBN: 978-84-608-8452-1.

[3] David Estevez, Juan G Victores, and Carlos Balaguer. "HORUS: Inspección robotizada de los trajes de protección del personal sanitario de pacientes en aislamiento de alto nivel, incluido el Ébola." In: *Jornadas Nacionales de Robótica*. Comité Español de Automática. 2017. ISBN: 978-84-697-3742-2.

[4] David Estevez, Juan G Victores, Santiago Morante, and Carlos Balaguer. "Robot devastation: Using DIY low-cost platforms for multiplayer interaction in an augmented reality game." In: *7th International Conference on Intelligent Technologies for Interactive Entertainment (INTETAIN)*. IEEE. 2015, pp. 32–36.

[5] Raul Fernandez-Fernandez, David Estevez, Juan G Victores, and Carlos Balaguer. "Improving CGDA execution through Genetic Algorithms incorporating Spatial and Velocity constraints." In: *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2017, pp. 290–295.

[6] Raul Fernandez-Fernandez, David Estevez, Juan G Victores, and Carlos Balaguer. "Reducing the number of evaluations required for CGDA execution through Particle Swarm Optimization methods." In: *IEEE International Conference on Autonomous Robot Systems and Competitions (ICARSC)*. IEEE. 2017, pp. 284–289.

[7] Raul Fernandez-Fernandez, Juan G Victores, David Estevez, and Carlos Balaguer. "Real Evaluations Tractability using Continuous Goal-Directed Actions in Smart City Applications." In: *Sensors* 18.11 (2018), p. 3818.

[8] Raul Fernandez-Fernandez, Juan G Victores, David Estevez, and Carlos Balaguer. "Robot Imitation through Vision, Kinesthetic and Force Features with Online Adaptation to Changing Environments." In: *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 1–9.

[9] Raul Fernandez-Fernandez, Juan G Victores, David Estevez, and Carlos Balaguer. "Quick, Stat!: A Statistical Analysis of the Quick, Draw! Dataset." In: *10th EUROSIM 2019 Congress*. EUROSIM Federation of European Simulation Societies. 2019.

[10] Alice Stazio, Juan G Victores, David Estevez, and Carlos Balaguer. "A Study on Machine Vision Techniques for the Inspection of Health Personnels' Protective Suits for the Treatment of Patients in Extreme Isolation." In: *Electronics* 8.7 (2019), p. 743.

In addition, the author would like to include in this section the following communications and merits:

- The publication *"Improving and evaluating robotic garment unfolding"* was granted the Best Paper Award at the IEEE International Conference on Autonomous Robot Systems and Competitions 2017.

- The publication *"Robotic ironing with 3D perception and force/-torque feedback in household environments"* attracted interest outside of the scientific community, and was reported in several international news media, including the BBC News[1] and New Scientist magazine[2].

1 https://www.bbc.com/news/av/technology-40435011
2 https://www.newscientist.com/article/2138264

- This research was disseminated to the general public through the talk *"Teo aprende a planchar"* at the Pint of Science 2009 festival in Alcalá de Henares, Madrid, Spain.
- The author achieved the 3rd place in the Ferrovial-Ennomotive Robotics Challenge, Global Robot Expo 2017, Madrid, Spain.

ABSTRACT

Garments are a key element of people's daily lives, as many domestic tasks -such as laundry-, revolve around them. Performing such tasks, generally dull and repetitive, implies devoting many hours of unpaid labor to them, that could be freed through automation. But automation of such tasks has been traditionally hard due to the deformable nature of garments, that creates additional challenges to the already existing when performing object perception and manipulation. This thesis presents a Robotic System for Garment Perception and Manipulation that intends to address these challenges.

The laundry pipeline as defined in this work is composed by four independent -but sequential- tasks: hanging, unfolding, ironing and folding. The aim of this work is the automation of this pipeline through a robotic system able to work on domestic environments as a robot household companion.

Laundry starts by washing the garments, that then need to be dried, frequently by hanging them. As hanging is a complex task requiring bimanipulation skills and dexterity, a simplified approach is followed in this work as a starting point, by using a deep convolutional neural network and a custom synthetic dataset to study if a robot can predict whether a garment will hang or not when dropped over a hanger, as a first step towards a more complex controller.

After the garment is dry, it has to be unfolded to ease recognition of its garment category for the next steps. The presented model-less unfolding method uses only color and depth information from the garment to determine the grasp and release points of an unfolding action, that is repeated iteratively until the garment is fully spread.

Before storage, wrinkles have to be removed from the garment. For that purpose, a novel ironing method is proposed, that uses a custom wrinkle descriptor to locate the most prominent wrinkles and generate a suitable ironing plan. The method does not require a precise control of the light conditions of the scene, and is able to iron using unmodified ironing tools through a force-feedback-based controller.

Finally, the last step is to fold the garment to store it. One key aspect when folding is to perform the folding operation

in a precise manner, as errors will accumulate when several folds are required. A neural folding controller is proposed that uses visual feedback of the current garment shape, extracted through a deep neural network trained with synthetic data, to accurately perform a fold.

All the methods presented to solve each of the laundry pipeline tasks have been validated experimentally on different robotic platforms, including a full-body humanoid robot.

RESUMEN

La ropa es un elemento clave en la vida diaria de las personas, no sólo a la hora de vestir, sino debido también a que muchas de las tareas domésticas que una persona debe realizar diariamente, como hacer la colada, requieren interactuar con ellas. Estas tareas, a menudo tediosas y repetitivas, obligan a invertir una gran cantidad de horas de trabajo no remunerado en su realización, las cuales podrían reducirse a través de su automatización. Sin embargo, automatizar dichas tareas ha sido tradicionalmente un reto, debido a la naturaleza deformable de las prendas, que supone una dificultad añadida a las ya existentes al llevar a cabo percepción y manipulación de objetos a través de robots. Esta tesis presenta un sistema robótico orientado a la percepción y manipulación de prendas, que pretende resolver dichos retos.

La colada es una tarea doméstica compuesta de varias subtareas que se llevan a cabo de manera secuencial. En este trabajo, se definen dichas subtareas como: tender, desdoblar, planchar y doblar. El objetivo de este trabajo es automatizar estas tareas a través de un sistema robótico capaz de trabajar en entornos domésticos, convirtiéndose en un asistente robótico doméstico.

La colada comienza lavando las prendas, las cuales han de ser posteriormente secadas, generalmente tendiéndolas al aire libre, para poder realizar el resto de subtareas con ellas. Tender la ropa es una tarea compleja, que requiere de bimanipulación y una gran destreza al manipular la prenda. Por ello, en este trabajo se ha optado por abordar una versión simplicada de la tarea de tendido, como punto de partida para llevar a cabo investigaciones más avanzadas en el futuro. A través de una red neuronal convolucional profunda y un conjunto de datos de entrenamiento sintéticos, se ha llevado a cabo un estudio sobre la capacidad de predecir el resultado de dejar caer una prenda sobre un tendedero por parte de un robot. Este estudio, que sirve como primer paso hacia un controlador más avanzado, ha resultado en un modelo capaz de predecir si la prenda se quedará tendida o no a partir de una imagen de profundidad de la misma en la posición en la que se dejará caer.

Una vez las prendas están secas, y para facilitar su reconocimiento por parte del robot de cara a realizar las siguientes

tareas, la prenda debe ser desdoblada. El método propuesto en este trabajo para realizar el desdoble no requiere de un modelo previo de la prenda, y utiliza únicamente información de profundidad y color, obtenida mediante un sensor RGB-D, para calcular los puntos de agarre y soltado de una acción de desdoble. Este proceso es iterativo, y se repite hasta que la prenda se encuentra totalmente desdoblada.

Antes de almacenar la prenda, se deben eliminar las posibles arrugas que hayan surgido en el proceso de lavado y secado. Para ello, se propone un nuevo algoritmo de planchado, que utiliza un descriptor de arrugas desarrollado en este trabajo para localizar las arrugas más prominentes y generar un plan de planchado acorde a las condiciones de la prenda. A diferencia de otros métodos existentes, este método puede aplicarse en un entorno doméstico, ya que no requiere de un contol preciso de las condiciones de iluminación. Además, es capaz de usar las mismas herramientas de planchado que usaría una persona sin necesidad de realizar modificaciones a las mismas, a través de un controlador que usa realimentación de fuerza para aplicar una presión constante durante el planchado.

El último paso al hacer la colada es doblar la prenda para almacenarla. Un aspecto importante al doblar prendas es ejecutar cada uno de los dobleces necesarios con precisión, ya que cada error o desfase cometido en un doblez se acumula cuando la secuencia de doblado está formada por varios dobleces consecutivos. Para llevar a cabo estos dobleces con la precisión requerida, se propone un controlador basado en una red neuronal, que utiliza realimentación visual de la forma de la prenda durante cada operación de doblado. Esta realimentación es obtenida a través de una red neuronal profunda entrenada con un conjunto de entrenamiento sintético, que permite estimar la forma en 3D de la parte a doblar a través de una imagen monocular de la misma.

Todos los métodos descritos en esta tesis han sido validados experimentalmente con éxito en diversas plataformas robóticas, incluyendo un robot humanoide.

CONTENTS

I INTRODUCTION

1 INTRODUCTION 3
- 1.1 Motivation . . . 3
- 1.2 Challenges . . . 4
- 1.3 The Laundry Pipeline . . . 8
- 1.4 Objectives . . . 9
- 1.5 Document Structure . . . 11

2 BACKGROUND 13
- 2.1 Hanging . . . 14
- 2.2 Unfolding . . . 16
- 2.3 Ironing . . . 21
- 2.4 Garment State Estimation . . . 25
- 2.5 Classification . . . 28
- 2.6 Folding . . . 31
- 2.7 Chapter Summary . . . 39

II METHODS

3 HANGING 43
- 3.1 Introduction . . . 43
- 3.2 Overview . . . 45
- 3.3 Dataset Generation . . . 47
- 3.4 Hanging Prediction . . . 50
  - 3.4.1 HangNet Regression Model . . . 53
  - 3.4.2 HangNet Classification Model . . . 54
- 3.5 Chapter Summary . . . 55

4 UNFOLDING 57
- 4.1 Introduction . . . 57
- 4.2 Overview . . . 58
- 4.3 Garment Segmentation Stage . . . 59
- 4.4 Garment Clustering Stage . . . 62
- 4.5 Garment Pick and Place Points Stage . . . 63
- 4.6 Manipulation . . . 65
- 4.7 Chapter Summary . . . 70

5 IRONING 73
- 5.1 Introduction . . . 73
- 5.2 Overview . . . 75
- 5.3 Garment Segmentation . . . 76
- 5.4 Wrinkleness Local Descriptor (WiLD) . . . 78

5.5 Ironing Path Extraction 80
5.6 Ironing Path-Following Controller 82
5.7 Chapter Summary 85
6 FOLDING 87
6.1 Introduction 87
6.2 Overview 88
6.3 Neural Folding Controller 91
6.4 Cloth State Estimation 93
6.4.1 Data Synthesis 94
6.4.2 FoldNet model 97
6.5 Chapter Summary 103

III EXPERIMENTAL RESULTS

7 EXPERIMENTS 107
7.1 Hanging 107
7.1.1 Hanging Regression Model 107
7.1.2 Hanging Classification Model 111
7.2 Unfolding 117
7.2.1 Purpose of the Experiment 117
7.2.2 Experimental Setup 117
7.2.3 Experimental Evaluation 121
7.2.4 Experimental Results 121
7.3 Ironing 125
7.3.1 Purpose of the Experiment 125
7.3.2 Experimental Setup 125
7.3.3 Experimental Evaluation 127
7.3.4 Experimental Results 128
7.4 Folding 130
7.4.1 Purpose of the Experiment 130
7.4.2 Experimental Setup 132
7.4.3 Experimental Evaluation 133
7.4.4 Esperimental Results 134
7.5 Chapter Summary 135
8 RESULTS AND CONCLUSIONS 139
8.1 Progress Beyond the State of the Art 139
8.1.1 Hanging 140
8.1.2 Unfolding 140
8.1.3 Ironing 141
8.1.4 Folding 141
8.2 Future Lines of Work 142
8.2.1 Hanging 142
8.2.2 Unfolding 143
8.2.3 Ironing 144

8.2.4 Folding 145

BIBLIOGRAPHY 147

LIST OF FIGURES

Figure 1.1 Some examples of domestic environments that present a challenge for robot locomotion 5
Figure 1.2 Garment deformability as a challenge for perception 6
Figure 1.3 Garment deformability as a challenge for manipulation 7
Figure 1.4 Proposed laundry pipeline 10
Figure 2.1 Isolation task 14
Figure 2.2 System proposed in (Sun, 2017) 16
Figure 2.3 Model-driven strategy of clothes recognition for automatic handling introduced in (Kita, 2011) 18
Figure 2.4 Active Random Forests Inference procedure presented in (Doumanoglou, 2014) . 20
Figure 2.5 Multi-stage pipeline for flattening garments introduced in (Sun, 2015) 22
Figure 2.6 Kinesthetic teaching of the positional profile with the method presented in (Kormushev, 2011) 24
Figure 2.7 Experimental setup of the ironing method presented in (Li, 2016) 25
Figure 2.8 State estimation task 26
Figure 2.9 Method for performing state estimation on a deformable object from a video sequence from (Willimon, 2012) 27
Figure 2.10 Classification task 29
Figure 2.11 Hierarchical system for estimation of garment category and pose proposed in (Mariolis, 2015) 30
Figure 2.12 PR2 robot performing a folding sequence using the method presented in (Cusumano-Towner, 2011) 33
Figure 2.13 Polygonal models from (Stria, 2014) for different garment categories 34
Figure 2.14 Two Shadow Dexterous Hands folding a sheet of paper using the method proposed in (Elbrechter, 2012) 36

Figure 2.15 Deformable object manipulation in a simulated environment using the MVP policy proposed in (Wu, 2019) 38
Figure 3.1 Hanging in the laundry pipeline 43
Figure 3.2 An example of traditional clothesline . . . 44
Figure 3.3 Hanging pipeline 45
Figure 3.4 Virtual setup for the hanging simulation . 49
Figure 3.5 Hanging simulation procedure for training examples 50
Figure 3.6 Random selection of training examples for hanging task 51
Figure 3.7 HangNet Regression Model architecture diagram 54
Figure 3.8 HangNet Classification Model architecture diagram 55
Figure 4.1 Unfolding in the laundry pipeline 57
Figure 4.2 Unfolding pipeline 59
Figure 4.3 Garment Segmentation Stage using RGB-D image as input 60
Figure 4.4 Garment Segmentation Stage using a 3D reconstruction of the scene as input 61
Figure 4.5 Garment Clustering Stage 62
Figure 4.6 Graphical representation of the Watershed algorithm 63
Figure 4.7 Pick and place points computation 64
Figure 4.8 Pick and place points computation explained 66
Figure 4.9 Grippers for the unfolding task 68
Figure 5.1 Ironing in the laundry pipeline 73
Figure 5.2 Soft wrinkles and marked creases 74
Figure 5.3 Ironing pipeline 76
Figure 5.4 Ironing segmentation steps 79
Figure 5.5 WiLD descriptor for flat and curved regions 80
Figure 5.6 Different steps of the path extraction stage 83
Figure 5.7 Reference systems for the ironing task . . 84
Figure 6.1 Folding in the laundry pipeline 87
Figure 6.2 Folding task definition 89
Figure 6.3 Under and over extension examples . . . 90
Figure 6.4 Folding paths supported by the simulator 96
Figure 6.5 Examples randomly sampled from the synthetic folding dataset 98
Figure 6.6 Stride and padding visually explained . . 100
Figure 6.7 Max Pooling visually explained 101

Figure 6.8 FoldNet architecture 101
Figure 7.1 Training example input and expected output 109
Figure 7.2 Mean Squared Error (MSE) and Validation MSE with respect to the time step . . 111
Figure 7.3 Mean Absolute Error (MAE) of each of the 3 coordinates (X, Y, Z) with respect to the time step 112
Figure 7.4 Confusion matrices for the classification model and human baseline 116
Figure 7.5 Robotic platforms used in the unfolding experiments 118
Figure 7.6 Unfolding setup for the humanoid robot TEO 119
Figure 7.7 Unfolding setup for the industrial manipulator robot 120
Figure 7.8 Computed unfolding directions overlaid on top of the corresponding garment height map 123
Figure 7.9 Ironing setup for the ironing task 126
Figure 7.10 Wrinkleness on each experiment trial . . . 130
Figure 7.11 Several frames of an ironing operation executed as part of an experiment 131
Figure 7.12 Proposed folding setup 133
Figure 7.13 Selection of cloth state estimation results . 136
Figure 7.14 Errors classified according to factors . . . 137

LIST OF TABLES

Table 7.1 Classification model, 3000 elements 115
Table 7.2 Classification model vs Human baseline, 200 elements 115
Table 7.3 Results analysis of the Unfolding Algorithm (5 trials, 6 categories), per stage and garment category, expressed as percentage (%) 122
Table 7.4 Results analysis of the Unfolding Algorithm (20 trials, 5 categories), per stage and garment category, expressed in percentage (%) 124
Table 7.5 Results of the Ironing Perception Algorithm 129
Table 7.6 Results of the Ironing Algorithm 130

Part I

INTRODUCTION

1 INTRODUCTION

This thesis presents a novel robotic system for garment perception and manipulation, including the study, development and implementation of diverse methods for solving the different tasks of a laundry pipeline. The first chapter serves as a necessary introduction to the motivation behind this work and the main challenges and objectives that this work will undertake. At the same time, it will establish some important definitions of key concepts, such as the different tasks that compose our laundry pipeline.

1.1 MOTIVATION

People have been coexisting with daily tasks even before the appearance of the written word. Some of the tasks were directly related with survival, such as hunting or gathering food, while others were related to maintenance, such as cleaning or assembling different kinds of tools.

As people became more and more organized, they moved from performing most of the tasks by themselves to focusing and specializing in a single task, causing some of these activities to become paid jobs. The continuous advance of science and technology has allowed the progressive automation of a number of tasks and jobs while, at the same time, has created different kinds of new tasks and jobs.

The ultimate goal of automation has always been to reduce or eliminate the human factor in all these required tasks, resulting in a greater amount of free time available for people to perform either more useful or enjoyable tasks. Although any task is susceptible of being automated, the main priority has been automating tasks that meet the criteria of being dull, repetitive, dangerous, or any combination of the three, as people tend to avoid doing them due to boredom or safety concerns.

While a large proportion of the daily tasks available are remunerated (i.e. jobs), there are tasks people need to perform on a daily basis without being paid for doing them. Paid versions of these tasks do exist, but only when hiring other people to eliminate the need to do them. We can find some examples of

unpaid maintenance tasks in domestic tasks such as cooking, doing laundry, cleaning the house, or taking care of people. Despite some of them being enjoyable for some, such as cooking, most of them lay in the category of dull and repetitive tasks that are prone for automation. This category includes laundry tasks, the focus of this thesis.

A vast amount of time is invested weekly on these dull and repetitive domestic tasks but, contrary to what might be expected, most of the focus of the developments in automation has been placed on industrial automation. The main reason behind this fact is that it is much more simple to automate industrial tasks than domestic tasks, as they are typically performed in a more controlled environment with very specific conditions. In addition, industrial tasks can usually be performed by a device independently, without collaboration with a human being, and therefore safety is less of a concern on these systems, as there is little to no physical interaction between human and machine.

As quality of life and life expectancy increase, populations are aging, and the amount of elder people is increasing accordingly. This is another significant factor behind the demand for automated solutions for this kind of tasks. In this scenario, automation not only reduces the labor costs of having to recruit other people to do their domestic tasks, but it also allows elder people to be and feel more independent. This is an important factor too for younger people that, for any reason, may have limited mobility and therefore require the assistance of other home care workers.

Finally, and despite this work focusing on domestic applications, there is a significant portion of the industry and businesses that work with garments, textiles and other deformable objects, so there is also a demand for automated solutions in such scope. Although the throughput of such a system would probably be lower, the main advantage of installing a robotic system versus a traditional automated system is the increased adaptability to different garments and tasks.

1.2 CHALLENGES

As mentioned in the previous section, even though there is a clear demand for automated systems that work with garments and other deformable objects in domestic settings, many chal-

lenges have to be overcome to deploy such systems in a real-world domestic environment.

The first challenge is related to the working environment in which the robot has to be deployed. As opposed to industrial environments, that can be controlled and set up beforehand to fit the robot performing the objective task, domestic environments lack such predictability. In fact, domestic environments are highly dynamic, as people inhabiting them are constantly changing the location of many objects that serve as potential obstacles from the robot perspective, and unstructured, as there is no standard house design, so no comprehensive *a priori* model of the environment can be constructed before deployment.

Domestic environments are adapted to the humans living in them and their morphology which, in some cases, poses a challenge to some types of robots. For instance, stairways can limit the range of motion of wheeled robots, as they cannot climb them. Figure 1.1 shows some examples of domestic environments that are challenging for robots from a locomotion perspective. Not only the environment is adapted to humans, but also the tools they use, which are typically not suited for the most common robot types, as they lack the dexterity to hold and handle them.

Figure 1.1: Some examples of domestic environment that present a challenge for robot locomotion: a cluttered room (left) and a steep stairway (right). *Images by Hans Braxmeier from Pixabay and nikitabuida from Freepik, reproduced with permission.*

The few robots currently commercially available for domestic tasks suffer from all the aforementioned limitations, and are mainly relegated to function as automated vacuum cleaners. To address all these challenges, the author proposes the use of hu-

manoid robots, as their human-like morphology prevents them from suffering from these issues.

Perception of deformable objects poses another important challenge. Computer vision techniques have improved significantly over the last years, mainly due to advances in machine learning techniques such as Deep Learning. Still, most techniques are limited to rigid objects, and can be easily fooled by factors related to real-world operation, such as the presence of noise or varying light conditions. Such factors often cause them to not be reliable enough for robotic systems requiring continuous unsupervised operation in real-world settings.

In addition to all the well-known limitations of state of the art computer vision algorithms applied to rigid objects, garments are deformable, which affects their perception creating a whole new set of challenges.

For rigid objects, their apparent shape on an image depends only on the viewpoint or perspective from which the image is obtained. For instance, a soda can seen from the top is observed as a circle, but when seen from the side it is perceived as a rectangle. In the case of garments, the apparent or perceived shape does not only depend on the viewpoint, but also on the configuration or state of the garment that can easily change due to its deformability (Figure 1.2).

Figure 1.2: Garment deformability poses a challenge for its perception: although the garments on the left and right are the same, their visual appearance is not, being much easier to be recognized in their unfolded state (right).

On the other hand, for a given deformable object hundreds of thousands of different states are possible. While some of them only bring subtle changes to the object's appearance, others can dramatically change the observed shape of the object, hinder-

ing its recognition even for humans. In addition, deformability creates the possibility of self occlusions, as not only external objects or other clothing articles can occlude the garment, but parts of same garment can prevent the camera to obtain a view of certain parts of itself.

Garment deformability not only poses a challenge for its perception, but it also deeply affects garment manipulation. As garments are composed of thin layers of fabric, they require actuators with high dexterity to be grasped correctly.

Typical operations with garments involve separating them from other clothes or moving specific parts of the garment, such as overlapping folds. Without a fine control of the amount of fabric grasped, a manipulation operation can result in the movement of the whole garment or even several garments instead of the desired part. Therefore, successful garment manipulation ideally requires either robotic hands equipped with sensors and fine manipulation skills, or specialized tools specifically designed to handle them correctly.

Additionally, garment deformability is also the reason garments are highly dynamic objects, causing a distinctive chaotic behavior that results extremely difficult to predict (Figure 1.3). In consequence, when manipulating deformable objects, it becomes critical to continuously track the current garment configuration and adapt or recompute the manipulation trajectories in real time based on the current garment state. Precomputed trajectories are not sufficient in the majority of the cases, as it is highly unlikely that the simulated (or estimated) behaviors will match completely the actual real behavior of the garment.

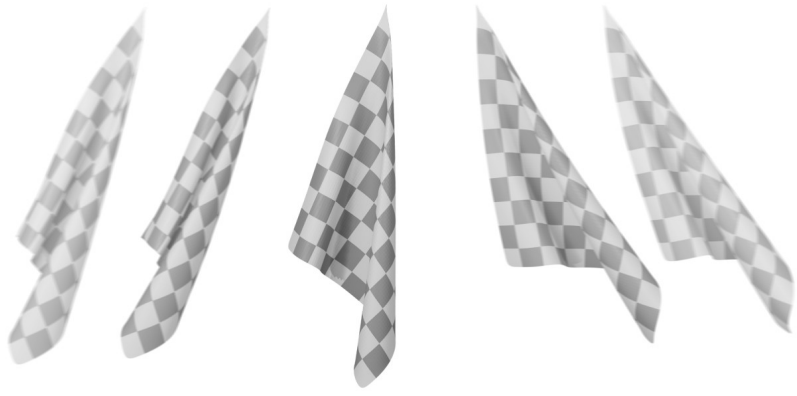

Figure 1.3: Garment deformability poses a challenge for its manipulation, as the shape and behavior of the garment changes dynamically as it is being manipulated.

To place a rigid object in a certain desired pose, knowing the target pose and interpolating between the current and target poses is enough to compute the movement required to reach that pose. On the other hand, the large garment configuration space vastly increases the difficulty of matching a garment pose with a target pose automatically, without providing a sequence of operations to achieve that goal.

Finally, the high variability of garment configurations and different operations that can be applied to them results in a scarcity of quality garment datasets that can be used in a wide range of garment-related tasks. Adding into consideration all the aforementioned challenges related to garments increases even more the already high costs of building a new garment dataset with real-world data. To reduce the incurred costs -in terms of time, labor and money- researchers typically resort to simulation to obtain garment datasets or training data, encountering other challenges such as bridging the simulation-reality gap.

1.3 THE LAUNDRY PIPELINE

Throughout the literature several definitions of a laundry pipeline, featuring different tasks, can be found, depending on the author. Since there is not an agreement on a standard laundry pipeline, it is therefore important to define which are the tasks that compose the laundry pipeline to be used along this thesis before it can be further discussed. This section will describe each of the tasks composing the author's laundry pipeline: hanging, unfolding, ironing, and folding, which are defined as follows:

1. **Hanging**. The hanging task is typically performed right after garments have been washed. Clothes need to be dry before storage, as mold and odors may appear if clothes are stored wet. The task is defined as taking a wet clothing article and placing it on a rope or similar arrangement so that it hangs and dries more easily due to the increased surface area exposed to air and sunlight. Depending on the initial conditions of the task, the hanging task might require to collect the wet clothes from the washing machine, or grasp them from a pile of already-extracted garments.

2. **Unfolding**. After garments are dry, they will be either ironed, or folded to be stored. As an intermediate step, unfolding of the garment is performed to ease recognition of the garment category from a spread state rather than from a randomly folded or crumbled pose, resulting in a better performance of both later tasks.

3. **Ironing**. In most cases, wrinkles will appear in garments due to internal stresses created in the garment fibers during the washing or drying process. To remove them before storage, the garment is typically ironed, that is, heat and pressure are applied to the spread garment with a device called "iron" to permanently remove the existing wrinkles.

4. **Folding**. Finally, the garment is folded for storage. Folding, putting the garment in a compact state, allows it to be stored reducing the space required and the risk of permanent wrinkles appearing in the garment. To facilitate the folding process, typically a predefined folding sequence, dependent on the garment category, is applied.

Figure 1.4 illustrates the author's laundry pipeline. Other tasks, such as washing the clothes, are not considered as part of the pipeline in this thesis, as other home devices such as washing machines already perform the task in an almost automated way. Nevertheless, this is a design decision, an other definitions of a laundry pipeline may exist according to other authors' criteria.

1.4 OBJECTIVES

With the challenges described in Section 1.2 in mind, this thesis aims to address the following objectives:

- The main objective of this thesis is to advance the state of the art in garment perception and manipulation through methods that have a direct application on concrete tasks belonging to the laundry pipeline introduced in Section 1.3, such as hanging, unfolding, ironing, and folding.

- The thesis will focus on developing methods that can be implemented in environments as close as possible to real-world domestic environments, with few to none modifications.

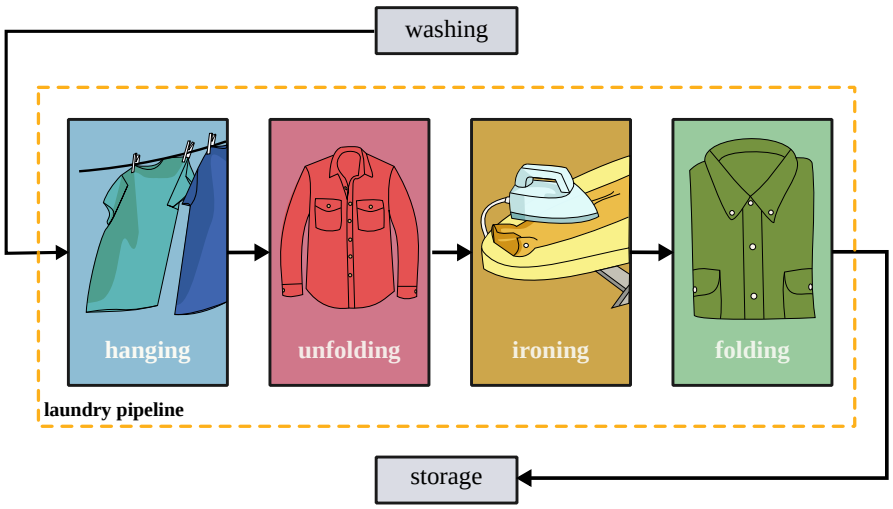

Figure 1.4: The proposed laundry pipeline is composed of four tasks that have to be performed between washing the clothes and storing them: hanging, unfolding, ironing and folding.

- The proposed methods and techniques developed will be subjected to experimental validation and critical review of the results obtained to evaluate the degree of success.

Each of the tasks in the laundry pipeline, due to its dissimilarities with respect to the other tasks, has been associated with its own objectives, that will be discussed in further detail in their corresponding chapters. Still, a brief per-task overview of these objectives is offered in the following paragraphs.

For *hanging*, the main objective is to achieve a first approximation to the hanging task, through a novel formulation of the task. To achieve this objective, a synthetic dataset will be created to study how a robotic system can predict the hangability of a given garment when dropped over a hanger.

For *unfolding*, the main objective is to develop a method to grant a robot the ability to unfold any given garment, independently from its category or shape. For this purpose, a modelless approach will be proposed to locate overlapping regions from garment RGB-D data, and then determine the location of the fold axis to remove the fold.

For *ironing*, the main objective is to obtain a method that allows a robot to remove permanent wrinkles from a garment in a domestic environment. To achieve this objective, a distinction between different types of wrinkle is proposed, that serves to define a human-inspired ironing algorithm applicable to a real-world domestic environment. On the one hand, the algorithm

has to be able to perform the ironing task without a precise control of the illumination or special tools. Additionally, the robot should be able to iron the garment by exerting a precise amount of pressure while following the computed ironing trajectory.

For *folding*, the main objective is to achieve a perception system able to estimate the state of the garment during the folding task. For this purpose, and through a research stay at CTU Prague, a knowledge transfer between both research groups will take place. A theoretical study of existing methods, including the destination lab's methods will be performed. Finally, a new method, based on an experimental simulation framework, will be developed in collaboration with the destination lab.

In addition, this thesis will serve as *a posteriori* analysis of the author's previous publications on this line of work, including a critical review in retrospective of strong points and weaknesses of past individual works from a global perspective, offering guidance and future lines of work for future research on this topic.

Finally, an important objective of every scientific work is replicability. Allowing other researchers to replicate the proposed algorithms and methods presented in this work ensures the scientific rigor of the results, fosters the study and improvement of the proposed methods by other research groups, and enables the comparison or benchmarking of the different methods that can be used to solve a given task.

For this purpose, the author of the thesis has released the code for all the published methods, which is available as a public repository[1]. Complementary to the code, and as will be introduced in the section explaining the methodology behind it, a dataset [18] for training and validating the hanging model has been released and is available online[2].

1.5 DOCUMENT STRUCTURE

This thesis is structured in a sequential manner, with an initial introductory part followed by a description of the methods presented, and concluding with an analysis of the experimental validation of such methods.

Although it is designed to be read sequentially, a single section of the thesis, such a the state of the art or a specific laundry task might be of special interest for the reader. In that case, this

1 https://github.com/roboticslab-uc3m/textiles
2 https://doi.org/10.5281/zenodo.3932102

section details the structure of the document and the contents of each of the chapters, intending to serve as a guide for the interested reader to guide him/her to find the parts most relevant to his/her particular interests.

Chapter 2 describes the state of the art in garment perception and manipulation, describing existing techniques and methods for working with garments while performing different tasks.

Chapter 3 includes an study of the hanging task, the first task on the laundry pipeline, from the point of view of predicting the behavior of a garment that is dropped over a hanger by a robot with the purpose of hanging it.

Chapter 4 introduces a method for unfolding garments, including the three stages that compose the method: the Garment Segmentation Stage, the Garment Clustering Stage and the Garment Pick and Place Stage.

Chapter 5 presents a method for ironing garments in a real-world environment with a humanoid robot, using RGB-D data as input and force/torque feedback to perform the ironing operation.

Chapter 6 proposes a garment state estimation method applied to folding garments, through the use of a neural folding controller developed in collaboration with the Czech Technical University of Prague.

The last part of the thesis is devoted to the experimental validation of the work presented. Chapter 7 serves as an description of the experiments performed for each of the tasks addressed on this thesis, as well as the results obtained.

A critical analysis of the results obtained is offered in Chapter 8, along with some future lines of work based on the conclusions of the analysis.

2

BACKGROUND

The aim of this chapter is to frame the work presented in this thesis in its context, by offering an extensive review of the scientific literature on garment perception and manipulation for different garment-related tasks. In general, the vast majority of the literature is focused on laundry-related tasks, although some works exist on other tasks such as bed-making [76] or dressing assistance [104][32][17].

In addition to the tasks composing the laundry pipeline defined in Section 1.3: hanging, unfolding, ironing and folding, other additional tasks have been included in this review. As other laundry-related tasks share certain challenges with the tasks in our pipeline, the author believes that it is therefore useful to include them in this section, as they have indirectly influenced this work. The additional tasks not included in our pipeline that are reviewed in this chapter are: isolation (included as part of the hanging task review in Section 2.1), as it can be considered a prerequisite of the hanging task; and state estimation (Section 2.4) and classification (Section 2.5), as both are indirectly required for other tasks of the pipeline such as the folding task.

Due to the task-oriented nature of this thesis, the different techniques used to tackle the tasks that compose our laundry pipeline have been developed sequentially over a span of several years. For that reason, some works from other labs have been published after the development of some of the methods presented in this thesis, while the author was working on a different task of the pipeline.

Even though those publications are chronologically posterior to the work presented in this thesis, the author believes that it is relevant to include them in the literature review of this chapter, as they offer additional context to analyze what garment-related tasks are currently being developed in other labs, and how the work presented on this thesis affected the development of later work published by other authors.

2.1 HANGING

After garments have been washed, the first step in the laundry pipeline is hanging, a necessary step for drying the garments before performing other tasks. Before the actual hanging can take place, it is necessary to retrieve a single garment from its initial location that, depending on the initial conditions, can be either a washing machine, or a pile of garments. The task of separating one clothing item from the rest of them is known as isolation, and is depicted in Figure 2.1.

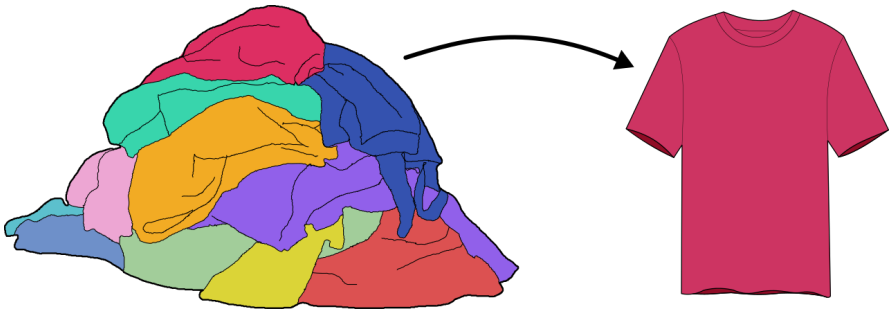

Figure 2.1: The isolation task is defined as selecting one clothing article from a pile of garments.

When working with garments in a laundry context, it is frequent to find different garments mixed and tangled together as they are stored before washing, or after they have been extracted from the washing machine or dryer. Since they are yet to be ironed and folded no ordered storage is required, as wrinkling is not an issue at this stage. But for the robotic system to be able to work with them in an ordered way, a single clothing article has to be isolated from the rest of the garments. Hamajima et al. [23] is one of the first works in the literature to note the need for robotic systems with the ability to work with non-rigid objects such as garments to assist people with their daily tasks, and to propose a solution for the isolation task. The proposed method uses color information to isolate garments from a washed mass prior to unfolding.

This work is extended in a later publication [34], where in addition to the isolation task, they offer the first approaches to garment state modeling and unfolding. The isolation task is solved by applying an algorithm based on simple color clustering of an input RGB image. For the unfolding task, an extensive analysis of the garment dynamics and a deformation state space taxonomy and classification is performed, resulting in a

set of hand-crafted features and a series of steps to be applied to the input image to successfully unfold the garment.

Although isolation is typically performed to a washed mass of garments, it can also be applied to a pile of dry garments, either before washing or after hanging or an electric drier has been used to dry the garments. In this case, classification of the garment is typically performed at the same time, as in [95], where Willimon et al. propose a method to isolate a garment from a pile of garments and then classify it according to its garment category based on its appearance. Their method uses graph-based segmentation to distinguish between the different garments in the pile and, once a garment has been picked up, low level image features such as silhouettes or edges to find it in a database of known items.

Sun et al. [87] present an approach for isolating and recognizing the category of a clothing article from 2.5D features while being robust to deformable shapes. The approach is composed of three phases: a local feature extraction with global coding, a global feature extraction and the actual classification. Local feature extraction is performed by obtaining the B-Spline Patch (BSP) and using a Locality-constrained Linear Coding (LLC) to learn a codebook representation. Global features are a combination of Shape Index (SI) histograms, Local Binary Patterns (LBP) histograms and Topology Spatial Distances (TSD). For the classification, a Support Vector Machine (SVM) with Radial Basis Function (RBF) kernels and a One-Versus-All strategy for multiclass classification are used. The approach is evaluated on a real robot, that is able to autonomously recognize a clothing item from an unconstrained pile of garments. Figure 2.2 shows the proposed three-phase system.

Isolation can be applied not only to piles of clothes, but also to already folded garments. For instance, in [10] the task is to pick up the topmost towel of a pile of towels, by decomposing the task into different procedures to solve, including grasping position detection using YOLO [72], as well as Q-learning for learning a grasping motion and subsequent manipulation of the garment.

Once the garment has been isolated, the hanging task can be performed. To the best of the author's knowledge, very few works exist in the garment-related literature focused on the hanging task itself. Matas et al. [59] propose an end-to-end solution based on state-of-the-art Deep Reinforcement Learning algorithms such as Deep Deterministic Policy Gradient (DDPG)

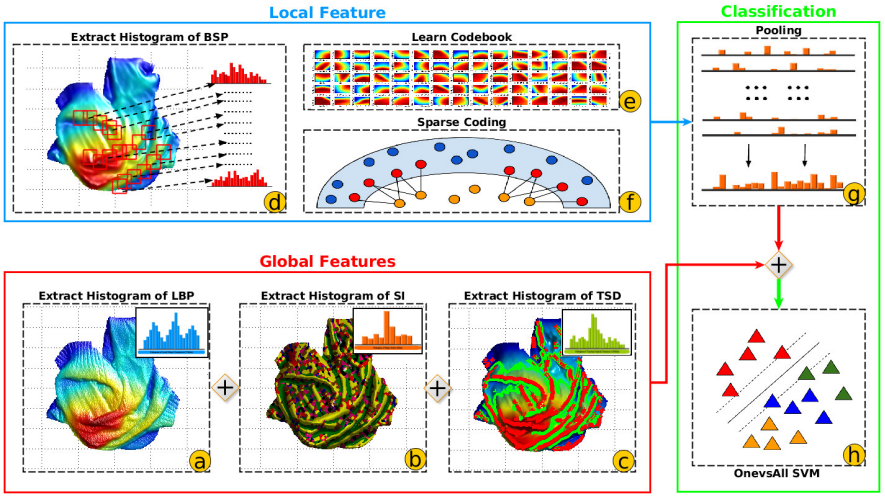

Figure 2.2: Three-phase system for isolation and recognition of garments proposed in [87]. On the top left, the different local features used are shown. The features are input to the classification block, depicted on the right, where they are combined with the global features, on the bottom left, to obtain the final estimation. *The figure was originally published as Figure 2 of "Single-shot clothing category recognition in free-configurations with application to autonomous clothes sorting." by Sun et al., Proc. IEEE IROS 2017 © IEEE, 2017.*

and Deep Deterministic Policy Gradient from Demonstration (DDPGfD). The control policy is learned in simulation, and then transferred to the real world to perform simple folding and hanging tasks with a small towel.

A related task, inserting a commercial clothes hanger, typically used to hang garments in a wardrobe, into a t-shirt is solved by Koishihara et al. through a method based on changes in wrinkle features and overlapping relationships between the garment and the hanger [44].

2.2 UNFOLDING

Garment unfolding, the second task in the laundry pipeline, serves as an intermediate task that allows the robot to extend the now dry clothing article to a fully-spread state to recognize its garment category. By knowing the garment category, the robotic system is able to later perform other tasks, such as ironing or folding the garment properly for storage, in a simpler manner.

Osawa et al. [64] presents one of the first works in garment unfolding, where they introduce a method to unfold and recognize the garment category of a clothing article before proceeding to fold it using two robot manipulators and visual feedback. The clothing item is unfolded by applying an iterative process of repeatedly grasping the hanging garment at the lowest point, to increase the distance from the grasping point to the lowest point of the next hung pose. Once the distance is maximized, the process is completed by fully spreading the garment over a flat surface.

Kobori et al. [42] postulate that clothes handling skills, such as unfolding, can be described as a combination of two different elements: clothes basic information and motion primitives. Clothes basic information defines features or locations of the garment, such as center, bottom, crease, etc, which are obtained using color and depth information so that they are robust to the variability in colors and shapes of garments. Motion primitives are basic actions described in terms of clothes basic information, such as grasp, slip, slide-on or drag. By defining the unfolding task as the combination of a "pick up in the air" action, an "unfold in the air" action and a "spread on a table" action they are able to successfully unfold different types of garments.

A two-stage method for estimating the current state of a garment is introduced by Kita et al. [39]. In the first stage, a set of candidate 3D shapes are obtained via deformable object simulation of a certain garment category being held by different points of the garment outline, and recorded. Then, in a second stage, an observation of the garment obtained by a trinocular stereo camera system is compared with the recorded shapes, which are further deformed to try to match the observation. The shape with the closest match represents the observed garment's current state.

Based on the previous deformable model, they developed a three-stage strategy to grasp the garment from a desired point while holding it with the other hand using a dual-arm humanoid robot [38]. First, the previous model is used to determine the garment state. Then, based on the estimated state, the optimal position and orientation of the robot gripper is computed through the use of a set of heuristics. Finally, due to the limited motion range of the dual arms, a correction stage is performed to modify the position and orientation of the gripper to be in the working range of the robot.

Another work published by the same authors [40] proposes a method to aid visual recognition of the garment category by actively handling clothing articles so that the current observable shape of the article is more informative of the garment category. Actions, including rotating and spreading the clothing item are automatically planned based on the current observed shape using geometrical methods, and performed by a dual-arm manipulator robot in mid-air.

The aforementioned methods are later integrated into a procedure to recognize the garment category and shape by means of strategic observation of the garment while being actively manipulated to enhance recognition [41]. Figure 2.3 shows an overview of the system. The previous work is extented in a recent work by integrating 3D data from multiple views of the hanging garment, to solve problems arising from self-occlusion when using a single-view [37].

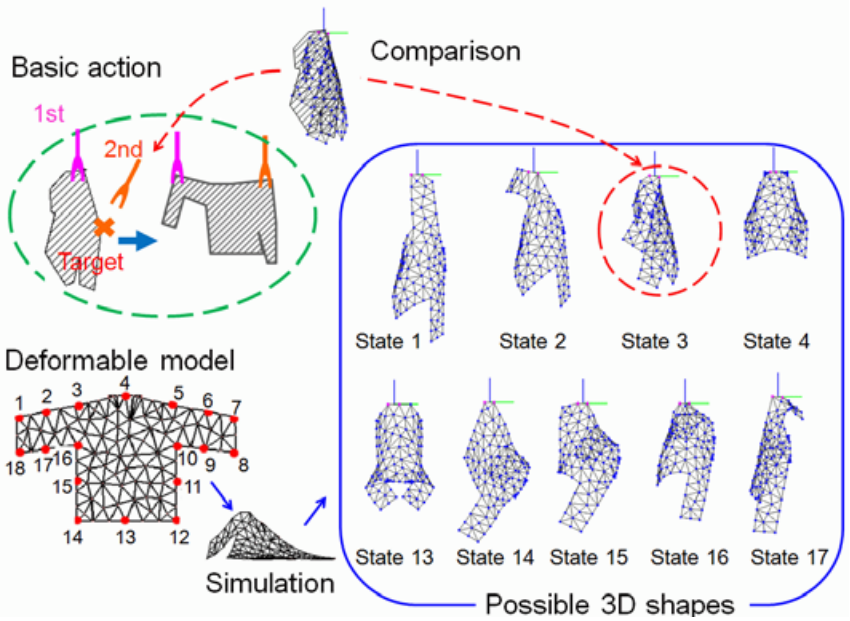

Figure 2.3: Model-driven strategy of clothes recognition for automatic handling introduced in [41]. A basic grasping action is performed, and the garment reconstruction obtained is matched with different candidate shapes generated from a simulated deformable model. *The figure was originally published as Figure 3a of "Clothes handling based on recognition by strategic observation" by Kita et al., Proc. IEEE Humanoids 2011 © IEEE, 2011.*

Willimon et al. [96] propose an algorithm to unfold a piece of cloth of simple geometry (e.g. a towel) using interactive per-

ception, by pulling the garment in different directions from different points until the cloth is fully spread. Their algorithm is performed in two phases: first, wrinkles and folds are initially removed without depth information, by pulling individual corners of the cloth counter-clockwise in 45 degrees increments. In the second phase, depth information is used to locate folds and to find candidate grasp points and directions to remove them.

Yamakazi [101] presents a method to find the most suitable grasping points on the hem of the garment that allow a dual-arm robot to directly extend it from an initial random crumpled state to a fully-spread state, based on depth data obtained from a fixed camera. In a later work [102] the same result is achieved, but using a Convolutional Neural Network to extract a feature vector that is employed both for classification and to obtain the most suitable grasping points as image coordinates.

Li et al. [51] introduces a method to perform garment unfolding and classification from an initial unknown state by performing iterative regrasping of the garment. Simulated thin shell models are used to create a database of different garment poses, that is later utilized to determine the current garment pose from a 3D reconstruction of the real garment. To find the regrasping point, a two-stage energy-based deformable object registration algorithm that aims to minimize energy differences between the source (simulated) and target mesh models is used. This process is repeated iteratively until there is an agreement between the predicted thin shell model and the mesh reconstructed from the garment observations.

Unfolding was one on the main tasks address by the European FP7 research project CloPeMA (Clothes Perception and Manipulation). As part of the CloPeMa project, Doumanoglou et al. [13] introduced Active Random Forests, a method that uses Random Forests to aid the selection of actions that allow the robot to obtain new views of an object to disambiguate its state, and apply it to the problem of garment category classification, key point detection and pose estimation for 4 types of garments. Figure 2.4 graphically depicts the Active Random Forest inference procedure.

This work is later extended to unfold garments in mid-air in [14], where they propose a method for both clothes unfolding and classification using a dual-arm robot based on industrial manipulators and a dual DSLR stereo system. Garment unfolding is performed in mid-air, starting from a random grasping point, and using Hough Forest for estimating the grap point

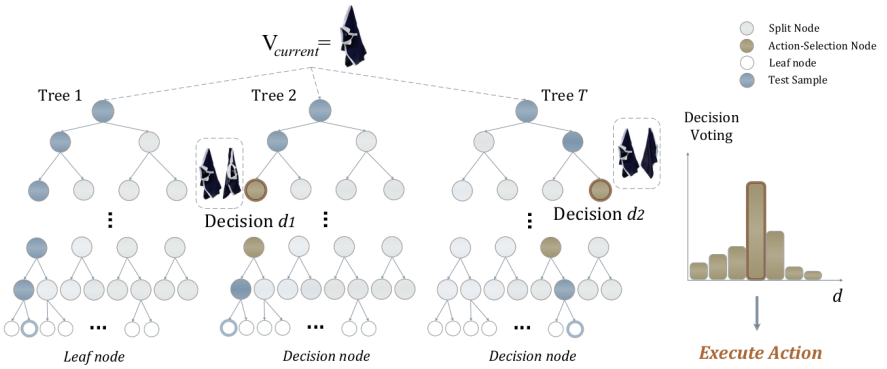

Figure 2.4: Active Random Forests (ARF) Inference procedure presented in [13]. From an arbitrary initial viewpoint $V_{current}$ the ARF can determine either the gament state and category or the next action needed to disamibiguate its state. *The figure was originally published as Figure 5 of "Active Random Forests: An Application to Autonomous Unfolding of Clothes" by Doumanoglou et al., Proc. ECCV 2014 © Doumanoglou et al., 2014.*

locations. Garment classification is based on a data-driven Random Decision Forest built from depth images. Both methods are implemented into a Partially-Observable Markov Decision Process (POMDP) framework that allows the robot to interact with the garments in an optimal way, by taking into account the uncertainties of the recognition and point estimation processes.

Another method developed in the context of the CloPeMa project is presented by Triantafyllou et al. [90], proposing an approach to unfold garments in mid-air with a dual-arm industrial manipulator robot, based on the observation that most garments can be brought to an unfolded configuration when held from two points belonging to the garment outline. To perform this gravity-based unfolding, they propose a geometric approach based on an analysis of the different types of edges detected in the hanging garment from RGB-D images.

Later work from the same authors [89] focuses on upper layer extraction of a garment after it has been laid in a planar half-folded state over a working surface, as part of a garment unfolding pipeline. The upper layer is detected from a depth image, through a depth first algorithm and a simple perceptron applied on a simplified action space based on the edges detected on the garment input image. The algorithm does not require a previous model or template of the garment, and works with irregular shapes as well.

In a similar work, but posterior to the presented in this thesis, Stria et al. [80] propose a model-free approach to unfolding garments based on the use of depth data to detect the overlapping upper layer of a clothing item. The garment depth data is segmented from the background using a color-based Gaussian Mixture Model (GMM) and then a labeling algorithm is applied to the segmented depth data based on an energy minimization framework that assigns different pixels to different layers with different associated potentials and then performs energy minimization based on those potentials. Finally, the fold axis is estimated by a virtual reflection of the folded patch over the different candidates, selecting the one resulting in a shortest unfolded contour.

In a recent work, Jia et al. [31] approach the unfolding task too, by selecting the optimal control action for the unfolding manipulation operation using a Random-forest-based controller, using the observed visual features as input. The random forest is constructed using imitation learning with two substeps: sampling the online dataset and optimizing the controller. The online dataset is constructed by extracting the HOW (Histogram of Oriented Wrinkles) features of the aligned raw RGB image, which is then paired with the action(s) that resulted in that observation. The method is applied to different cloth-related tasks, such as flattening, folding and twisting.

2.3 IRONING

Before the garments can be folded and stored, possible wrinkles created during the washing and drying processes have to be removed. Otherwise, storing an already wrinkled garment will cause the existing wrinkles more be prominent. Different kinds of wrinkles require different techniques to remove them. Large soft wrinkles can be removed by pulling the garment, in a process called flattening.

Sun et al. [83] present a method to flatten wrinkled garments by iteratively pulling the garment, determining the most suitable action point, and force amount and direction to perform the pulling operation. A heuristic-based strategy is applied based on the wrinkles detected from the robot vision system, and a simulator is used to close the interaction loop between the robot end-effector, the garment and the visual feedback obtained by the robot.

In a later work, they propose a system [84] to flatten a piece of clothing based on an dual-arm industrial manipulator robot able to capture high-resolution RGB-D images though a DSLR-camera based stereo vision sensor. The high resolution depth image obtained from the robot is B-spline smoothed to act as a low-pass filter and remove high frequency noise on the cloth reconstruction. Then, a shape and topology analysis based on the shape index [43] is used to locate and characterize the wrinkles. Wrinkle candidates are then analyzed using different quantification metrics to select the most significant ones, which are then used to plan a dual-arm manipulation sequence to flatten the cloth. The complete multi-stage pipeline is depicted in Figure 2.5.

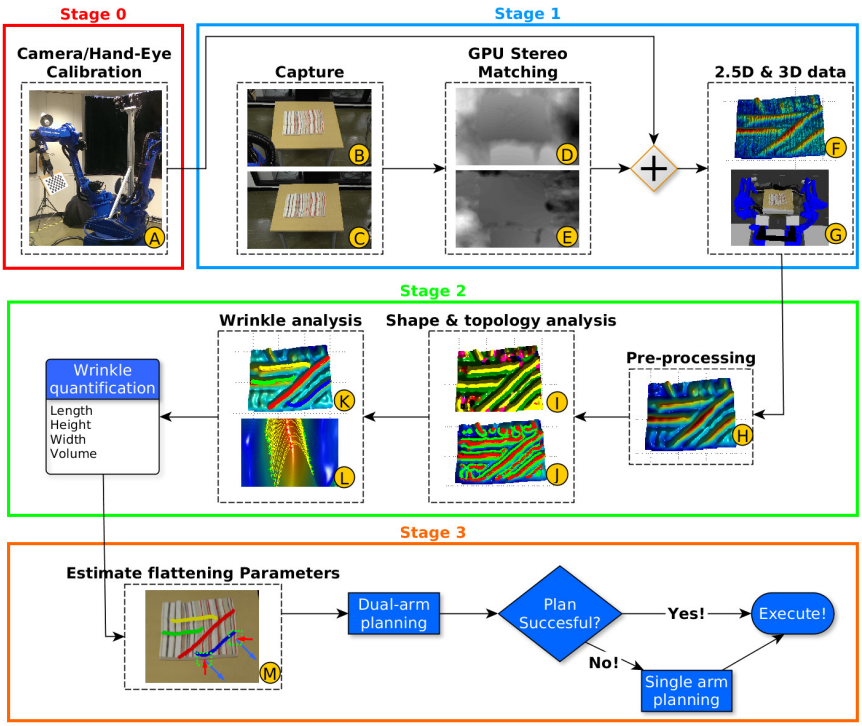

Figure 2.5: Multi-stage pipeline for flattening garments introduced in [84]. Stages 0 and 1 obtain the 2.5D data used by state 2 to locate the wrinkles, which are then flattened in stage 3. *The figure was originally published as Figure 2 of "Accurate Garment Surface Analysis using an Active Stereo Robot Head with Application to Dual-Arm Flattening" by Sun et al., Proc. IEEE ICRA 2015 © IEEE, 2015.*

The previous system is extended in a later work [85], in which the wrinkle information obtained is used to compute the forces required to flatten the cloth, as well as their directions

and gripper contact locations through a heuristic-based strategy and a greedy algorithm that iteratively selects the largest detected wrinkle.

On the other hand, as it will be further discussed in the corresponding ironing chapter (Chapter 5), permanent marked creases cannot be eliminated through flattening, and they must be ironed to be removed. In order to be ironed, the garment is usually placed (inserted) on an ironing board.

Twardon and Ritter introduce a method [92] to identify boundary components using interactive perception to be able to put a hat on a hat-stand, which could be modified to be applied to garment placement on the ironing board prior to ironing. The approach is based on the extraction of closed contours from a depth image, from which boundary components are detected by a graph-based search. After the boundary components have been detected, a heuristic energy function is minimized to find a suitable grasp pose to hold the garment by the boundary component.

Once the garment has been set on the ironing board, the actual ironing can be performed. One of the first works on robot ironing is one published by Khatib et al. [35], where they propose the use of a mobile manipulator for different types of domestic tasks - including ironing. A few years later, the EPSRC grant "A Feasibility Study into Robotic Ironing" resulted in a number of theoretical analysis, such as a study on algorithms for folding and unfolding garments as well as contemporary robotic gripper solutions that could be of aid during the ironing task [8] and a mathematical model of an ironing path derived from a protractor-and-tracing-paper tracking of a human demonstration [9]. Kormushev et al. [45] incorporated an actual physical robot with force feedback to the robotic ironing process. Ironing paths were learned from user demonstrations by kinesthetic teaching (Figure 2.6), and force profiles were extracted from demonstrations via a haptic device.

The most similar work to the one presented in this thesis is the method introduced by Li et al. [52] in which they propose a method to iron a piece of cloth based on the fusion of two modalities of data. Low resolution depth data, which they call "curvature scan", is obtained through a 3D reconstruction obtained from a standard RGB-D sensor. This kind of data is used to identity large soft wrinkles, named "height bumps" in their nomenclature, which are not marked and can therefore be re-

Figure 2.6: Kinesthetic teaching of the positional profile for the ironing task, introduced in [45]. *The figure was originally published as Figure 8a of "Imitation Learning of Positional and Force Skills Demonstrated via Kinesthetic Teaching and Haptic Input" by Kormushev et al., Advanced Robotics © Taylor & Francis, 2011.*

moved by pulling the garment, and it is obtained through a volumetric analysis of the 3D reconstruction.

High resolution data, called "discontinuity scan", is extracted from a pair of RGB images captured under different illumination conditions. The high resolution data represent marked creases, named "permanent wrinkles" in their nomenclature, is obtained by using a Lambertian reflectance model [63] on a pair of images captured under controlled illumination conditions. More precisely, the indoor lights have to be turned off, and two external light sources, calibrated with the current cloth in a completely ironed state, have to be alternatively activated to light the cloth from different directions, which is one of the main limitations preventing this work to be used in a real-world scenario outside the lab. Once located, marked creases are removed by static or dynamic ironing using a combination of position control in the robot and the use of a foam under the cloth as a source of passive compliance. Figure 2.7 shows the experimental setup required for this method.

Finally, we note that, for a machine learning-driven approach to ironing, wrinkle simulation might be useful to obtain synthetic data to speed up data gathering. A novel method for wrinkle simulation is prosposed in recent work by Chen et al.

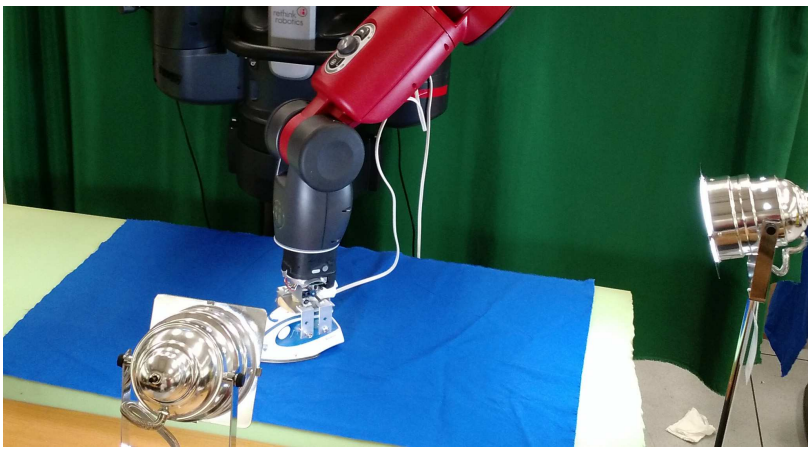

Figure 2.7: Experimental setup of the ironing method presented in [52], including the set of external lamps required for the curvature scan and the green foam required for passive compliance. *The figure was originally published as Figure 8a of "Multi-Sensor Surface Analysis for Robotic Ironing" by Li et al., Proc. IEEE ICRA 2016 © IEEE, 2016.*

[5]. Their method, called mesh superresolution, enhances a low-resolution cloth mesh with high resolution wrinkles though a Convolutional Neural Network, resulting in a 24x speedup compared with a traditional simulation of a mesh of the same size. To be able to use traditional (2D) Convolutional Neural Networks, the cloth mesh is converted to an image by uniformly sampling the mesh and recording the baryocentric coordinates of the triangle where each sample is located as the R, G, and B components of an image.

2.4 GARMENT STATE ESTIMATION

For a large proportion of the garment-related tasks, having information about the full garment state increases the probabilities of success. Many tasks can benefit from state estimation (Figure 2.8), ranging from classification and folding, to sock pairing [94]. For that reason, many of the literature methods perform some kind of garment state estimation either before or during the performance of the main task. As garments are deformable objects, the location of the garment (over a flat surface, hanging on a rope, etc) greatly influences the current garment state.

Li et al. [48] present an approach to obtain the garment category and current pose of a deformable object for later classi-

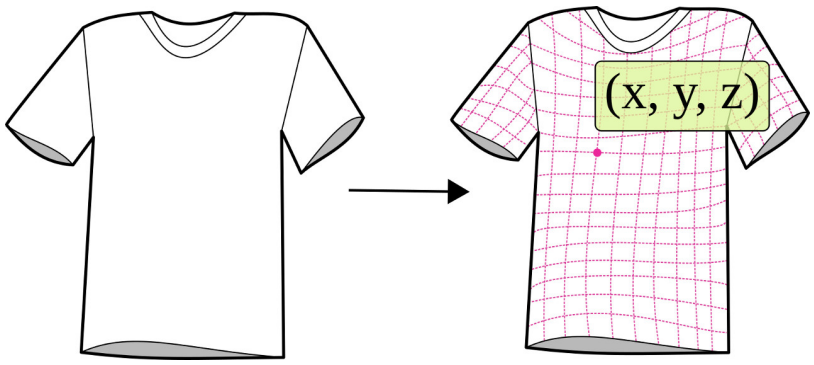

Figure 2.8: The aim of the state estimation task is to obtain an internal representation of the current shape and/or pose of a given garment.

fication from a set of depth images. The core of the approach is the use of sparse coding and dictionary learning to build a codebook from a set of offline simulated garments in different poses. The system is built from two layers: it first determines the garment category of a given garment, and then estimates the current pose from a set of predefined poses for that garment category.

An alternative approach is offered by the same authors in a later work [49] where they present another method to perform joint garment state estimation and classification. Binary 3D features inspired by 3D Shape Context [20], but replacing the spherical coordinate system with a cylindrical coordinate system, are used to build a database with features obtained from simulated garments hanging from different grasp points. The same features can be computed from a 3D reconstruction of the real garment obtained through volumetric fusion of several RGB-D images, and then used to search the most similar category and pose in the database.

Garment state estimation techniques can also be applied to obtain a non-rigid 3D scan of a deformable object. Willimon et al. [98] estimate the current garment state from an RGB-D image using an energy minimization-based method, by computing SURF features, depth, and boundary information, and then applying a 3D nonlinear energy minimization framework.

This work is improved in a later work, where they present a method to perform state estimation on a deformable object from a video sequence of the object being manipulated, using energy minimization and graph cuts [97]. As a consequence, feature correspondence or texture information is no longer necessary to be present in the input video sequence. The method

Figure 2.9: Performing state estimation on a deformable object from a video sequence [98]. On the left, the estimated garment state is superimposed to the source video frame. On the right, the same estimation is show from a 45° angle. *The figure was originally published as Figure 6d of "An Energy Minimization Approach to 3D Non-Rigid Deformable Surface Estimation Using RGBD Data" by Willimon et al., Proc. IEEE IROS 2012 © IEEE, 2012.*

is able to generate the corresponding mesh automatically and handle in-plane rotation by re-initializing the mesh after data has been lost in the image sequence.

Garments are not the only kind of deformable objects where state estimation techniques can be applied, as show by Petit et al. in a work focused on pizza manipulation [66]. Their method tracks a 3D textureless object which undergoes elastic deformations, using the point cloud data provided by an RGB-D sensor and in real-time. The point cloud obtained via visual perception is first fit in a rigid manner and then non-rigidly by modeling elasticity though a FEM simulation.

Recent work shows that non-rigid 3D scanning is possible even from a single image: Pumarola et al. present a method to predict a mesh representing the 3D shape of a patch of a deformable object surface from a single view [69]. Their Convolutional Neural Network-based approach uses different branches that perform 2D detection of the mesh and estimation of a 3D shape consistent with image, respectively. They train the architecture in an end-to-end manner using synthetic data simulated under different conditions in terms of light, deformation, materials, etc.

Visual shape servoing of the garment to be manipulated is another possible application of garment state estimation. Han et al. [24] introduce a method to estimate the shape of a deformable object from a RGB-D image sequence using Signed

Distance Functions (SDFs), to use it in the feedback loop of a manipulation controller.

Hu et al. [28] presents a data-driven controller to perform visual servoing tasks with deformable objects, which allows a robotic system to manipulate them to change their position and shape. The controller is based on a fully-connected deep neural network that is used to map the end-effector's movement and the object's deformation, and it is trained using an online learning strategy. As input, a novel feature is extracted from an RGB-D image using Fast Point Feature Histograms (FPFH) [74] extended with a Principal Component Analysis (PCA). In addition, this work also introduces an occlusion recovery algorithm able to reconstruct the whole deformable object from an incomplete input point cloud.

The same authors present a similar approach for visual servoing of deformable objects with unknown deformation parameters, but using Gaussian Processes (GPs), in a different work [27]. The alternative method uses a modified Gaussian Process Regression (GPR) to learn a mapping between the movement of the robot's end-effector and the deformation measured in the object. To reduce the high computational cost of GPR when working with long manipulation sequences, they select and remove uninformative observation data from the regression process.

2.5 CLASSIFICATION

Before a garment can be folded, it is typically required to know what garment category the clothing article belongs to, as different types of garments have different folding sequences. This task is known as classification (Figure 2.10) and it is closely related to state estimation, as the garment state space depends on the garment category and exact shape.

For classification, some approaches can be applied directly to the crumpled clothing item, such as [103], where they introduce a method to classify garments directly from an image of the garment in a crumpled or wrinkled state. A set of features invariant to translation, rotation, and scale are engineered by hand from a set of Gabor filters applied to the input image, and then used to match the input images with images in a garment database.

Other approaches, such as Kita et al. [36] propose a system to recognize the garment category directly from a hanging gar-

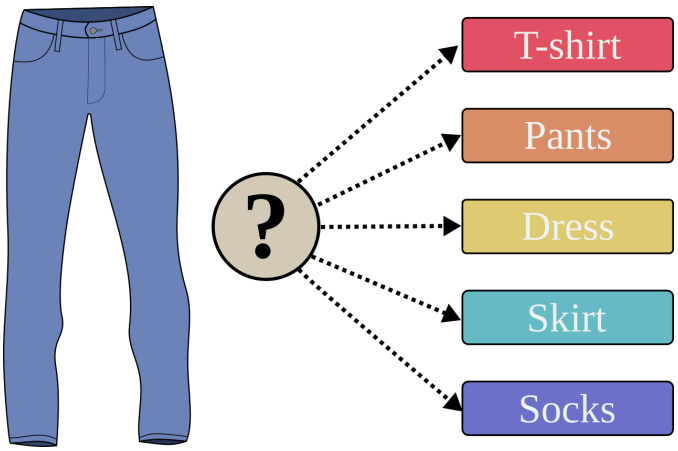

Figure 2.10: The classification task is defined as obtaining the garment category of a given piece of clothing.

ment by virtually flattening the garment using the geodesic distance over the surface of a 3D reconstruction of the hanging garment. Geodesic lines are computed from the 3D point cloud using the boundary points of the observed region as start and end points, by an interpolation of the depth and normal direction at any point of the surface using the element-free Galerkin method and a numerical solution based on a zero-length spring analogy. A later work [37] extends the previous method by integrating 3D data from multiple views of the hanging garment, to solve problems arising from self-occlusion when using a single-view.

Some methods require the garment to be placed over a flat surface for classification, such as GarmNet [22], a two-level model based on Deep Neural Networks. The global level, based on a retrained 50-layer Resnet [25] originally trained on ImageNet [11], is used to locate the garment whereas the low level, based on Convolutional Neural Networks, finds landmarks to locate future grasping candidate regions.

An alternative approach is to pick up the garment to perform classification in mid-air. For instance, Mariolis et al. [54] perform recognition and pose estimation of a garment that has been grasped and is hanging from a single point. The camera system acquires a depth image of the garment from a point of view, although if needed it can integrate several views with a majority voting scheme. The method is based on a 2-layer hierarchy of Deep Convolutional Neural Networks (CNN), where the first one classifies the garment from the raw depth images into one of the three predefined categories: shirt, pants or towel.

For each of the categories, a second CNN estimates the current garment pose. The method has been validated on a custom two-arm industrial manipulator. Figure 2.11 shows the proposed hierarchical system.

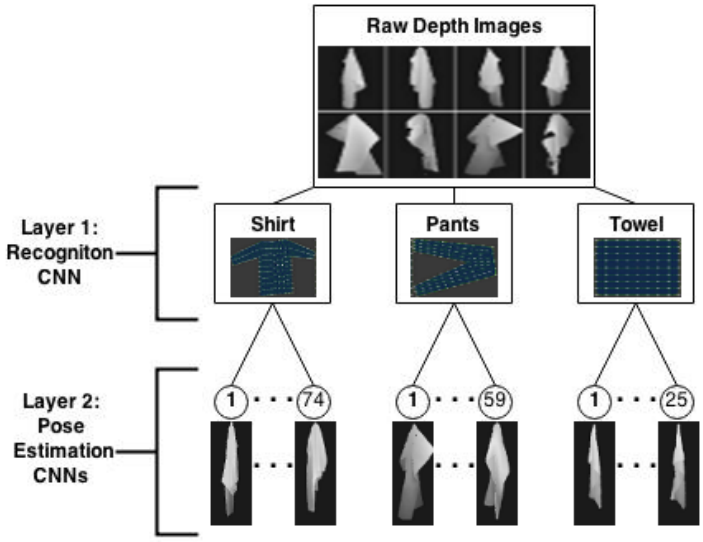

Figure 2.11: Hierarchical system for estimation of garment category and pose proposed in [54]. The garment category is estimated in the first layer, and then the corresponding CNN of layer 2 is selected for pose estimation. *"Pose and Category Recognition of Highly Deformable Objects Using Deep Learning" by Mariolis et al., Proc. IEEE ICAR 2015 © IEEE, 2015.*

Another approach where Deep Learning is used is [21], where they present an approach to garment category classification from a hanging grasped state based on the use of a Convolutional Neural Network. The input of the network is a single view of the garment as a 240x240 pixels depth image, and the system is able to determine the garment category from 5 different classes, including shirt, trousers, polo, towel or no garment. If the current view is not enough to recognize the garment category, the robot can rotate the clothing article to obtain a different view with larger prediction confidence.

Manipulation of the clothing article can be employed to interact with the garment in a similar manner a human would do to find good views of the garment and recognize it. In addition to the Active Random Forests [13][14] mentioned in the unfolding section, other works [86] propose an interactive approach to clothes classification for highly wrinkled clothes using multi-class Gaussian Processes (GP), where the robot can use actions

such as shake or flip to improve the confidence of the recognition. They use marginal likelihood maximization for GP hyperparameter optimization, as well as the Laplace approximation for posterior inference.

Corona et al. [6] propose a method to simultaneously recognize the garment category and bring it to a known pose though interaction. They train a hierarchy made of three levels of Convolutional Neural Networks. First, a CNN classifier predicts the garment category and then the input data is passed to a second layer where, depending on the category, it is passed to a different CNN that predicts the first grasping point. After the first grasp has been executed, the resulting pose is passed to the third layer, which predicts the second grasping point required to bring the garment to the desired pose.

Instead of a depth image, [79] propose the use of a unstructured point cloud of the hanging garment[1] to classify the garment using a Convolutional Neural Network based on PointNet [70], using local features based on k-nearest neighbors of each point, which are later integrated into a global feature vector. The network is trained with ShapeNet [4], a large dataset containing general 3D objects.

In recent work, Martinez et al. [56] investigate how perceiving the dynamic behavior of a garment in a continuous way as it is being picked up improves the quality of the final classification of the garment. For that purpose, they continuously extract visual features from an RGB-D video stream, which are then fused using the Locality Constrained Group Sparse Representation (LGSR) algorithm. For training, they prepared a database with annotated videos of 150 picking actions.

2.6 FOLDING

The last stage of the laundry pipeline is folding, a task required to store garments in a compact way that reduces the chances of producing additional wrinkles during storage. Folding is a very active topic within the deformable object research community, and there is a vast collection of works in the literature approaching the folding task.

Some of the methods take advantage of state estimation techniques to model the garment pose and the manipulation sequence required to successfully fold a clothing article. The UC

1 Obtained using a modified version of Kinfu: github.com/Nerei/kinfu_remake

Berkeley Robot Learning Lab published some of the early works on robot folding, starting with a method for folding where a grasp point detection algorithm is used to detect corners of a piece of cloth [53]. The method is robust against changes in patterns or textures present in the cloth, as it uses only geometric cues.

The previous work was later extended by Miller et al. [60], incorporating the use of a parameterized shape model to recognize the configuration of garments already spread out on a flat surface. For each clothing category, a parameterized shape model is created, and the intra-class variation in each garment category is achieved by modifying the values of the different parameters in the model. The different models are fit from an image of the garment using an energy-based optimization process to match the contour of the observed garment with the contour of the parameterized shape model, that acts as a template. The method is able to track the state of the garment even when folds are introduced, and therefore serves to close the perception loop during the folding sequence.

A different approach by the same lab utilizes a Hidden Markov Model (HMM) [7] to enable a PR2 robot to bring a clothing article into a target configuration from any given state, by performing a sequence of manipulations and observations. At the end of the sequence the article is in a known state, although it might not be the target configuration. This sequence is depicted in Figure 2.12. After the first sequence, a second one is planned to bring the article into the desired configuration. A simulation with a simplified FEM model is used to train both the HMM transition and observation models.

The aforementioned work was combined with work from Van der Berg et al. [93] presenting the concept of gravity folding, a cloth model that allows a robot to work with garments laying on a flat table based purely on geometry rather than taking into account the dynamics of the garment. The concept of g-fold is then introduced, as a fold helped by gravity and achievable while in a certain region of the garment geometric state space. Using the gravity folding cloth model, an algorithm is presented than, given a certain cloth geometry (i.e. category), computes the amount of grippers needed and the movement of the grippers to achieve a given final configuration, specified as a sequence of g-folds. The complete framework for robotic laundry folding, combining all the described techniques are tested and evalatuated in a later work [61].

Figure 2.12: PR2 robot performing a folding sequence using the method presented in [7]. On the left, the initial crumpled state is shown. Through a sequence of manipulations the robot is able to unfold the pants to a known state (shown on the right), and the robot can then proceed to fold them. *The figure was originally published as Figures 1 and 2 of "Bringing Clothing into Desired Configurations with Limited Perception" by Cusumano-Towner et al., Proc. IEEE ICRA 2011 © IEEE, 2011.*

Bersch et al. [3] present a similar approach using a PR2 robot to fold a clothing article starting from an initial crumpled state. To perceive the clothing article, it combines 3D information from a 360° stereo scan with 2D information from several views, using fiducial markers on the clothing article to determine the garment state. Using the information extracted with the perception system, a greedy policy selects the suitable grasp pose to grab cloth folds and bring the garment to the target folded state.

Yanakawa et al. [100] use a high-speed dual robot hand system with a high-speed visual feedback loop to achieve dynamic folding of a sheet-like flexible object such as a piece of cloth. Planning is done in simulation and then applied to the real robot by adding a high-speed visual feedback control loop in order to correctly grasp the cloth in real time.

This work is later improved by Bergstom et al. [2] by learning a temporal object model for deformable objects from observations. For this, shapeme histograms are extracted from the image obtained by the robot camera and used as features to both train and test the model. The model is based on a Hidden Markov Model (HMM), whose states are obtained from the training set by building a Gaussian Mixture Model over some of the PCA components of the set, and is able to successfully fold the cloth in both static and dynamic conditions.

Stria et al. [81] propose a polygonal model that describes a garment based on the angles and relative lengths of the different segments that define the garment contour, as shown for different garments in Figure 2.13. The garment is segmented from the background using a Gaussian Mixture Model (GMM) based on the assumption of a uniform color background. The contour is then extracted using Moore's algorithm for tracing 8-connected boundary of a region, and simplified by approximating it to a polyline with dynamic programming. Once extracted, the contour is fitted to the template for that particular garment category through the minimization of an energy function that encodes the dissimilarities between the model and the observed data. This work is extended in later work [82], and integrated in the complete folding pipeline of the CloPeMa project [15].

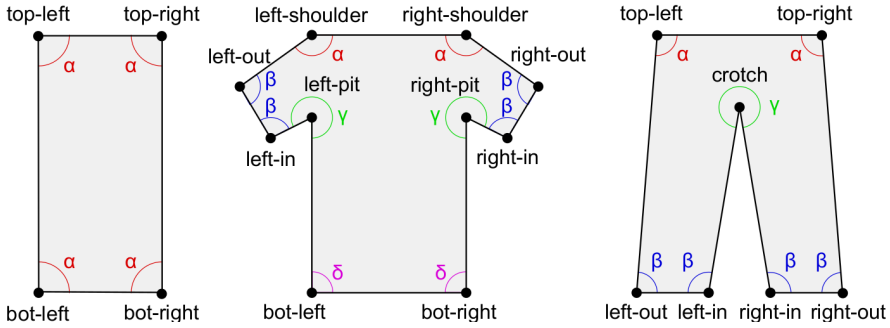

Figure 2.13: Polygonal models from [81] for different garment categories: towel (left), T-shirt (middle), and pants (right). *The figure was originally published as Figure 4 of "Polygonal Models for Clothing" by Stria et al., Proc. TAROS 2014* © *Springer, 2014.*

Hou et al. [26] [107] introduce a method to obtain the corresponding garment category from a piece of clothing already spread out over a flat surface prior to folding. The method utilizes a particle-based polygonal model which is fitted to an image of the observed garment automatically, without the need for a manual template. The garment is aligned by hand, and there must be contrast between the garment and the background surface. The model is restricted to the following garment categories: towel, shirt, and trousers, each one with its own model and parameters.

The model is further tested in a later work [106], where they approach the folding task by fitting the particle-based polygonal model to a garment laying flat on a working surface, and then plan and execute the folding trajectory according to a predefined folding sequence.

Jia et al. [30] present a method for deformable object manipulation based on a novel feature called Histogram of Oriented Wrinkles (HOW), obtained from an RGB data stream using Gabor filters. The feature serves as high-level representation of a clothing article and its current state. A visual feedback dictionary is then built from the HOW features to correlate the features to the control instructions required, mapping the velocity in the high-dimensional feature space to the velocity of the robot end-effectors.

The previous method is improved in a later work [31] by selecting the optimal control action for the manipulation operation through a Random-forest-based controller, using the observed visual features as input. The random forest is constructed using imitation learning with two substeps: it first samples the online dataset, and then optimizes the controller. The online dataset is constructed by extracting the HOW-features of the aligned raw RGB image, which is then paired with the actions that resulted in that observation. The method is tested by applying it to different cloth-related tasks, such as flattening, folding and twisting.

In recent work, Petrik et al. focus on understanding the physical properties and state of a cloth during folding. In [67], they focus on accurate folding of a single cloth strip by taking into account the dynamics and behavior of the cloth though a vision feedback-based controller. The controller is trained using reinforcement learning in a calibrated simulation matching the real cloth strip properties. In a later work [68] they study the effect of the static instability present in the folding process, how it affects the folding process, and how it is required to take it into consideration to perform a successful trajectory planning, by analyzing different existing trajectories for folding fabrics.

State estimation oriented to folding is not limited to garments, and it can be applied to other kinds of deformable objects. For instance, Elbrechter et al. [16] present a method for paper manipulation with anthropomorphic hands. In this work, real-time modeling of a sheet of paper is performed with the combination of a visual tracking of the surface of the paper though BCH-code fiducial markers with a physical model, including elastic bending of the sheet plus non-elastic bending corresponding to permanent folds or crease lines. The model is tested on a set of Shadow Dexterous Hands to perform actual manipulation via a closed loop controller based on tactile and visual information (Figure 2.14).

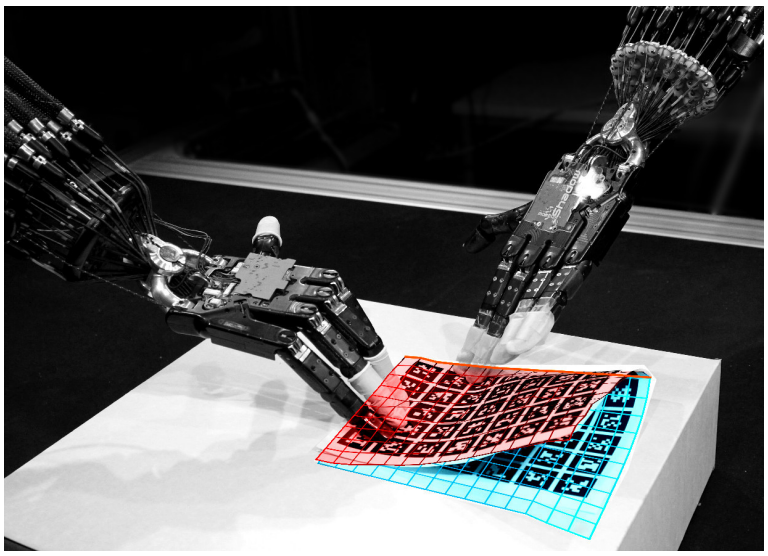

Figure 2.14: Two Shadow Dexterous Hands folding a sheet of paper using the method proposed in [16]. *The figure was originally published as Figure 1 of "Folding Paper with Anthropomorphic Robot Hands using Real-Time Physics-Based Modeling" by Elbrechter et al., Proc. Humanoids 2012 © IEEE, 2012.*

While most approaches to garment folding leverage the use of a model of the garment state to obtain extra information that simplifies the folding task, there are other approaches to folding that aim to perform folding directly without an explicit modeling of the garment. Kruse et al. [46] developed a vision-force hybrid controller for a dual-arm Baxter robot to manipulate cloths collaboratively with a human. The visual feedback helps the controller to overcome the deficiencies present on a pure force-feedback controller when the cloth is slack between the two agents performing the manipulation task, and the system is able to apply the required tension correctly.

Li et al. [50] propose a method to obtain an optimal folding trajectory by minimizing a quadratic objective function measuring similarity between the shape of the garment in a simulation and the shape specified by the user, which is then transferred to the robot for execution.

Yang et al. [105] study the possible use of a robot worker in an automated production line of garment-related manufacturing by studying the folding of a deformable object as main task. For that purpose the robot acquires some demonstrations from a remote operator, that controls the robot via a real-time teleoperation setup including a first-person view of the robot via

a head-mounted display for the operator. A two-phase deep learning model including a Deep Convolutional Autoencoder and a Deep Time Delay Neural Network is used to learn from the human demonstrations.

Tanaka et al. [88] propose a system for manipulation planning based on a Deep Neural Network derived from an Autoencoder architecture with an extra manipulation network in between the encoder and decoder networks that represent the transformation on the cloth state due to a manipulation action. This Encoder-Manipulation-Decoder network is able to predict, given a low-resolution depth input image of the cloth initial state, and a manipulation input, the corresponding aspect of the cloth after the manipulation operation.

Other approaches make use of reinforcement learning or similar methods to solve the folding task, generally by learning in a simulated environment and then transferring that acquired knowledge to the real task. Balaguer et al. [1] combine imitation and reinforcement learning to solve the folding task, using human demonstrations to reduce the search space of the reinforcement learning algorithm so that convergence is faster while a deformable object model is not required. Folding is perform through a momentum fold, a swinging motion takes advantage of the cloth dynamics.

Tsurumine et al. [91] present a folding method based on two sample-efficient Deep Reinforcement Learning (DRL) algorithms: Deep P-Network (DPN) and Dueling Deep P-Networks (DDPN). DPN exploits the advantages of both DRL for high-dimensional state space and Dynamic Policy Programming for smooth policy update, resulting in a method that combines the nature of smooth policy update with the capability of automatic feature extraction of Deep Neural Networks to enhance the sample efficiency and learning stability with fewer samples. Policy update smoothness is promoted by limiting the relative entropy of the Kullback-Leibler divergence between the current and new policies.

Wu et al. [99] propose to solve manipulation of deformable objects through model-free visual reinforcement learning. To accelerate learning, as the place point depends heavily on the grasp point, they propose a conditional action space, where the output of the picking policy is used as input for the placing policy, while learning both policies independently. For training the placing policy, the picking policy is random. For testing, the Maximum Value of Placing (MVP) picking policy is used. The

proposed method is trained in simulation and transferred to a real PR2 robot to perform the actual manipulation operations. Figure 2.15 shows the corresponding manipulation sequences obtained with two different deformable objects by following the learned MVP policy.

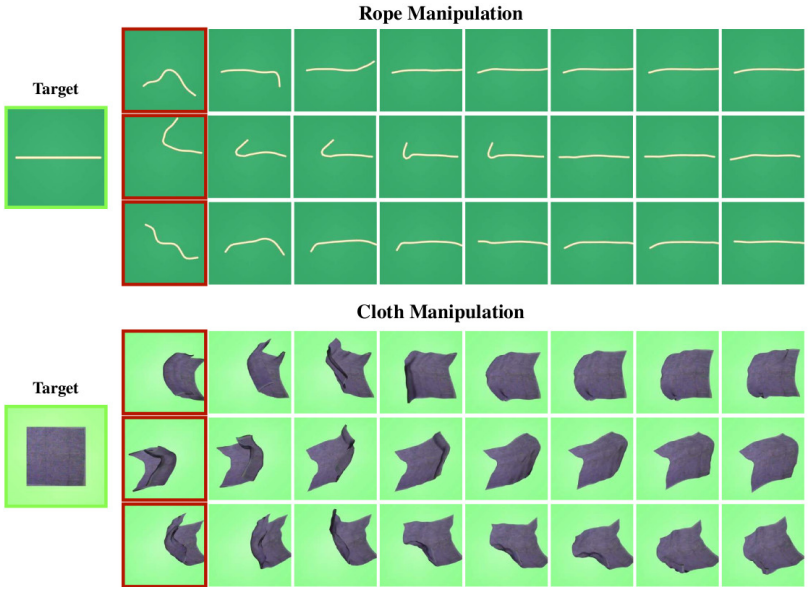

Figure 2.15: Deformable object manipulation in a simulated environment using the MVP policy [99]. Each picture represents the state of the rope after 5 manipulation steps (top) and the garment after 10 manipulation steps (bottom), from the state of the previous image. *The figure was originally published as Figure 5 of "Learning to Manipulate Deformable Objects without Demonstrations" by Wu et al., ArXiv © Wu et al., 2019.*

Jangir et al. [29] introduce a Deep Reinforcement Learning approach to solve cloth manipulation tasks in simulated cloths, by focusing on reaching a goal position with non-grasped points of the cloth by learning adequate trajectories of grasped points. Few demonstrations are required to improve control policy learning, and sparse rewards are utilized to avoid engineering complex reward functions. Observations are obtained from the simulator using different state spaces depending on the task to solve, that are then performed within the simulator.

2.7 CHAPTER SUMMARY

In this chapter, the literature related to garment perception and manipulation with robotic systems has been reviewed, and the most significant methods have been listed and described.

In addition to the tasks related to our laundry pipeline, composed by the hanging, unfolding, ironing, and folding tasks, additional tasks have been included in this review. Since these tasks, that include isolation, state estimation and classification, are also applied to garments, they share some challenges and techniques with the tasks on our pipeline, and therefore they might be of interest for the reader.

Part II

METHODS

3

HANGING

This chapter describes our advances in hanging with robots, the first task of the laundry pipeline (Figure 3.1).

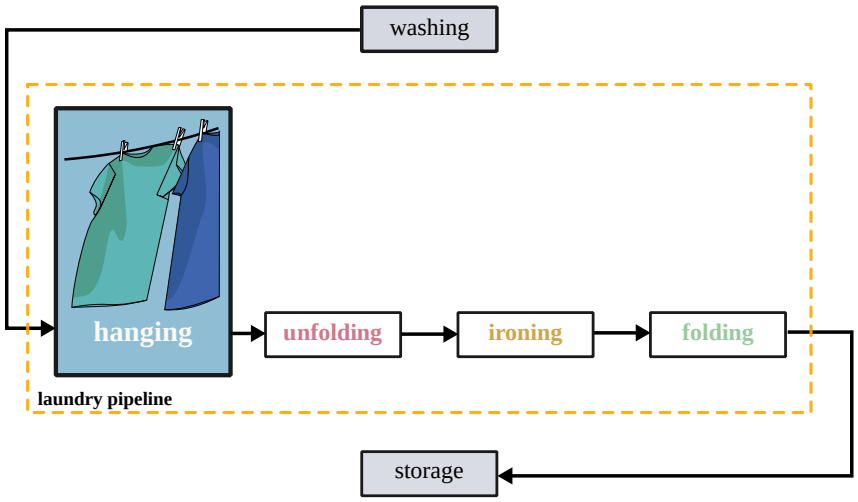

Figure 3.1: Hanging is the first task of the laundry pipeline.

3.1 INTRODUCTION

The laundry pipeline presented in this thesis starts immediately after a washing cycle, when garments are extracted from the washing machine. Although the water content present in the washed garments can be reduced through the use of a final spin cycle, most garments will remain wet after the washing process. When clothing articles are kept wet for an extended period of time, odors and bacteria may develop, and therefore there is a need for garments to be completely dry before they can be further processed in the laundry pipeline, for instance, to store them.

Different methods exist for drying clothes. Traditionally, clothing articles have been hanged from a clothesline to let them dry naturally, using clothespins to hold the garments to the rope, as seen in Figure 3.2. Hanging them increases the garment surface in contact with the air, causing a faster evaporation of the water present in the garment, and therefore a faster drying. Although the process can be further improved if sunlight is cast upon the

clothes, this drying method works even in cold environments, where water freezes and then sublimates into water vapor. Figure 3.2 depicts a traditional clothesline.

Figure 3.2: An example of a traditional clothesline, where wet garments are hanged from a rope using clothespins to let them dry naturally.

Washing and drying clothes are the only two laundry-related tasks that have been successfully automated with a widely available commercial machine: the washing machine and electric dryer, respectively. Even though in some cases using a dryer machine is useful and convenient, hanging clothes in a traditional way has some advantages over using an electric dryer:

- Exposing garments to open air and direct sunlight enhances freshness and removes strong odors from them, while disinfecting and bleaching white clothes due to the UV radiation present in sunlight.
- Hanging garments is more gentle on clothing than the electric dryer, reducing the stress put on cloth fibers and avoiding some side effects of heat such as shrinking.
- As clotheslines are usually located outside the house, in balconies and patios, they take advantage of unused space, while freeing useful space within the house by reducing the number of dedicated machines to be stored in the household.

In addition to all the aforementioned advantages, traditional clothes hanging takes advantage of natural sources of light, resulting in a more environment-friendly method compared to using a drying machine[1], specially in countries with more sunlight hours available. All the previous reasons motivate the use of a robotic system to hang clothes in a traditional way.

3.2 OVERVIEW

The hanging task can be defined as part of a more complete hanging pipeline in which the garment is first isolated from the rest of the wet clothing articles, then hanged on a clothesline to let it dry, and finally retrieved to be processed in the next task of the laundry pipeline. Depending on the initial conditions of the task, the garment might need to be retrieved from the washing machine, after the washing cycle has finished, or from a pile of wet garments that have already been collected from the washing machine, with different challenges associated to each of these scenarios. Figure 3.3 depicts this hanging pipeline.

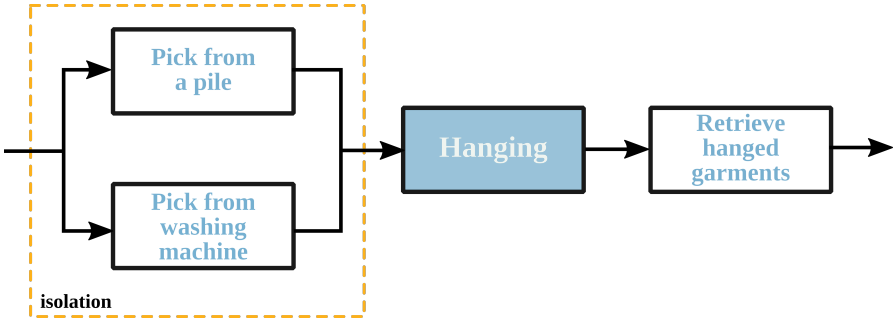

Figure 3.3: Hanging pipeline. Before garments can be hanged, they have to be isolated from the rest of the wet garments. Once they are hanged and dry, the garments have to be retrieved.

Even if we only focus on the actual hanging task, hanging garments from a clothesline is a complex and challenging task involving several motor and perceptual skills. Through a careful observation of how people perform this task, several subtasks can be noticed, each of them requiring different skills:

- **Selecting and picking a garment from a pile.** This subtask requires the ability to differentiate between different

1 In this case, the author assumes that the robot introduced to do the hanging performs other tasks from the laundry pipeline.

garments in a pile based on visual information, and to successfully grasp the desired garment exclusively, without grabbing any other clothing item from the pile. The target clothing article must be on the surface of the pile to be reachable by the robot. As the pile arrangement will change after each interaction, due to the dynamic nature of garments, for grasping subsequent clothing articles the pile analysis and grasp computation have to be repeated iteratively.

- **Placing the garment on the clothesline.** This subtask potentially requires bimanipulation to keep the clothing article spread on the clothesline, as well as the ability to estimate the category and current pose of the garment to be able to bring it from the initial grasped pose to the target spread pose.

- **Holding the garment with clothespins while keeping the garment on the clothesline.** Grasping and placing clothespins requires fine precision and dexterity in the robot hands, and holding the garment during the clothespin placement requires bimanipulation as well as a high degree of coordination between the different actions to perform.

Each of these subtasks is, on its own, complex enough that a whole thesis could be devoted to each of them, a fact endorsed by the lack of scientific literature on the hanging task (see Section 2.1). Therefore, to be able to achieve a first approximation to solve the problem, a simplification of the previously described hanging task is required.

In this work, the hanging task will be defined as *"dropping a randomly grasped garment over a clothesline or similar structure in such a way that the garment will remain hanged from that structure"*. The main aim of the new formulation is not just simplifying the problem, but also focusing the learning process of the robot on the understanding of the basic dynamics of clothing articles during the hanging task.

Under this formulation, a key ability for the robot is to predict whether a garment will remain hanged when dropped over the hanger, or will fall. The objective of this chapter is then to design and train a model that can be used by a robot to predict the result of a given hanging action prior to releasing the garment over the hanger. As input, the model will use a depth

image of the scene, including both the garment and the environment. Using the predictions from this model, the robot can then search for a valid position to release the garment, achieving a successful hanging result.

To fulfill the proposed objective, two HangNet models have been developed using Deep Learning techniques, each of them focused on predicting the outcome of a hanging operation with two different precision levels: a simple binary classification (will hang / won't hang), and a more precise estimation of the final garment position in the euclidean space, both described in Section 3.4. To train these models, a new dataset has been generated using the process described in Section 3.3.

3.3 DATASET GENERATION

To gain the ability to predict the outcome of dropping a garment over a hanger, the model has to successfully learn the garment behavior and dynamics through several training examples of different hanging actions. Deep Neural Networks with several hidden layers, such as the ones used in the HangNet models, have a large amount of parameters to fit during training, providing this kind of architectures with a large capacity to express the variety and richness of a training set.

Nonetheless, the same large capacity can also lead to a tendency to overfit the training data in the presence of a small dataset, by memorizing the training examples so that the network underperforms when it encounters novel inputs. As a consequence, a large amount of training data is required for this kind of model to learn the patterns present in the training data and generalize correctly to unseen input data.

Nowadays, training Deep Neural Networks for general tasks such as object classification is relatively simple, as high quality public datasets exist with thousands of already preprocessed and labeled images. On the other hand, there is a lack of similar datasets for robotic tasks, specially when working in niche applications such as garment-related tasks. Therefore, new datasets need to be created, curated and tailored for these particular applications.

However, recording real robot performances to obtain training data is expensive from multiple perspectives, including economic requirements, human labor and time spent in both supervising the robot performances and processing the recorded data. For instance, for the hanging task described in this chap-

ter, a garment has to be grasped by the robot and taken to a random point for each of the training examples. Then it has to be released and the system has to be reset to the initial state before the grasp and release sequence for the next example can take place.

In addition, if the garment position has to be recorded during the freefall, as in this case, an additional tracking system needs to be installed, possibly requiring some kind of fiducial marker. Due to the large amount of time that has to be devoted to each individual example, compiling a dataset with tens of thousands of training examples is unattainable unless several robots are setup to work in parallel, which is unfeasible for the budget of most research groups.

One possible solution to this challenge is to augment or replace the real data with data obtained through simulation, much more cost-effective and faster to obtain. Simulated data has the additional advantage of being simpler to process and label, as the simulation conditions can be controlled, and the state of the simulation is known and can be recorded for every simulation step.

The main disadvantage is that, since the characteristics of the simulated domain are very different from those of the real world domain (no noise, no lens distortion, different illumination conditions, uniform colors...), a Deep Neural Network trained on simulated data may not necessarily have the same performance on the real data, a fact that has to be taken into account with techniques such as domain randomization.

As the robotics research community is focused mostly on working with rigid objects, robotic simulators are generally designed to work only with rigid bodies, offering very limited support for deformable objects.

The computer graphics community, on the other hand, is a very active community in the use of cloth and soft body simulations, as realistic simulations of deformable objects are often required to create compelling computer generated pictures, movies and video-games. Motivated by a desire to depict garments in all these digital media with increasing realism has lead to the development of very precise cloth simulation tools in both 3D graphics editors and game engines. Therefore it makes sense to borrow these tools from the computer graphics community for our simulation purposes.

To generate the simulated dataset, a rough digital replica of our laboratory environment is built in a 3D editor, including the

garment, floor and hanger, as depicted in Figure 3.4. For this application a hanger made with aluminum extrusion has been built instead of a clothesline, as it is more stable and simpler to relocate than a fixture with a rope. To simulate the garment, a simple square cloth is used as a starting point before moving on to more complex garment geometries.

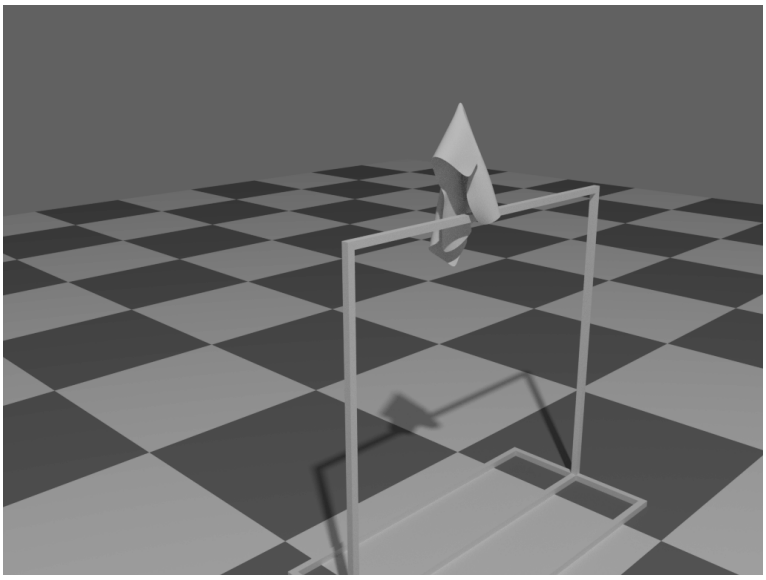

Figure 3.4: Virtual setup for the hanging simulation.

For each of the simulated trials, the garment is placed randomly in the vicinity of the topmost region of the hanger, laid flat and fixed by a random grasp point that is selected from all the vertices that compose the cloth mesh. The initial position of the garment is determined for each trial by sampling its X, Y and Z coordinates from a normal distribution with mean $\vec{\mu}_{init}$ and standard deviation $\vec{\sigma}_{init}$. This distribution is kept centered around the hanger, and close to it, to increase the chances of having instances where the garment is randomly hanged.

As mentioned previously, the cloth starts in a flat configuration, and then the physics simulation is started, causing the garment to hang in the air due to gravity, while being held only at the randomly selected grasping point. Once the simulation has reached a stable enough state, a depth image of the scene is captured, and the garment is then released, letting it fall towards the floor according to the physics simulation. The center of mass of the garment is tracked and recorded for a fixed amount of simulation steps. After the fixed amount of steps, it is assumed that the garment has reached a stable state, that can

be either hanging from the hanger, or laying on the floor. The simulation procedure is shown in Figure 3.5.

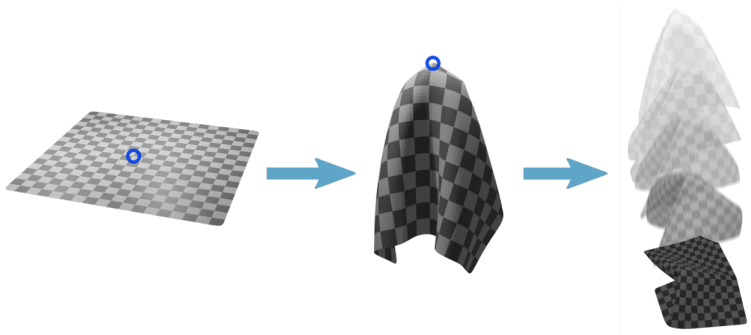

Figure 3.5: Hanging simulation procedure for training examples. A vertex of the cloth mesh is randomly selected (left) and the cloth simulation is run while the cloth is held by that vertex. Once the simulation has reached an steady state (middle), the garment is dropped on to the hanger (right), and is run for a predetermined amount of time.

For each of the trials, the depth image of the initial state of the working environment is recorded as input, and the trajectory of the center of mass of the garment once it has been dropped is recorded as the expected prediction for a single training example. A total of 15 000 examples were simulated and recorded. Figure 3.6 shows a random selection of training examples from the dataset.

The training dataset [18] is publicly available, and can be downloaded from Zenodo[2]. The aim of releasing the dataset publicly is twofold. On the one hand, it serves as a training set for the models that will be introduced in the next sections, and therefore can be used as a means to ensure the reproducibility of this work. By training the proposed models with the original data and hyperparameters, its possible to replicate the results presented in the experimental results section of this work. Additionally, the publicly available dataset can be used as a benchmark to compare the performance of this method as a baseline with other future works, fostering a healthy scientific competition that can generate better and more efficient models.

3.4 HANGING PREDICTION

To study the problem of predicting whether a garment will fall or hang when dropped over a hanger, two different tasks (re-

2 https://doi.org/10.5281/zenodo.3932102

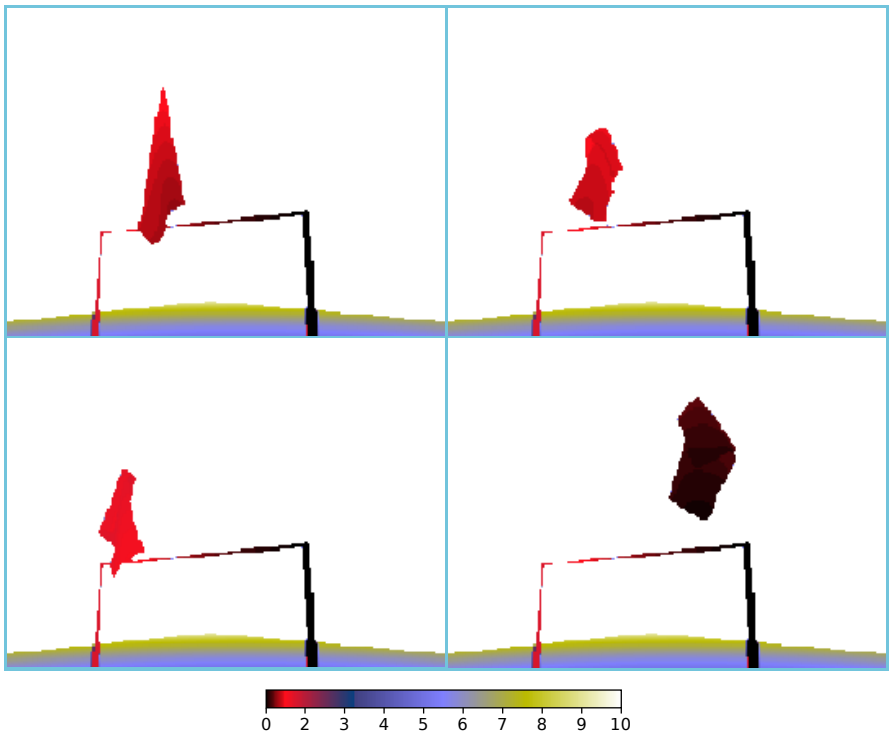

Figure 3.6: Random selection of training examples for hanging task. Each of the four input examples is shown as a depth map, whose color represents the distance from the sensor, in the scale indicated in the figure.

gression and classification) are defined to represent the problem from two different perspectives:

- The regression problem consists in predicting the final position of the garment at a given time step after it has been dropped over the hanger. The position of the garment, as in the simulation, it is tracked using the X, Y and Z coordinates of the center of mass of the garment.
- The classification problem, on the other hand, aims to predict just the final state of the garment once it has been dropped over the hanger, and classify it as hanged or not hanged.

For each of the two problems, a specific *HangNet* model has been developed. Both *HangNet* models are based on Deep Convolutional Neural Networks (CNNs). Whereas fully-connected neural networks have all the neurons of the previous layer connected to the next layer, this kind of architecture uses neurons connections to form filter banks that are then applied to the input image by convolution, obtaining several feature maps. The filter bank of the next layer then uses the previous layer feature maps to obtain their own feature maps containing increasingly abstract features as depth is increased.

A 2D convolutional layer can be described by four parameters: the size of the input image (N rows by M columns), the filter size (F) and the number of filters (D), and can be expressed as $conv2d(NxM, F, D)$. The activation map a of a given layer l and filter f for the pixel (x, y) can be computed as:

$$a_{lf}^{xy} = g(b_{lf} + \sum_{d}^{D} \sum_{i=0}^{F-1} \sum_{j=0}^{F-1} w_{lfd}^{ij} \cdot a_{(l-1)d}^{(x+i)(y+j)}) \tag{3.1}$$

Where b_{lf} is the bias for the current layer and filter, d is each of the D filters in the $(l-1)$-th layer, w_{lfd}^{ij} is the weight value at position (i, j) of filter d for the current layer and filter, and g is the activation function. For both models a stride of 1 pixel is fixed for all filters and layers. Each of the weights and biases of the model is a trainable parameter that has to be optimized to fit a training set, as opposed to the activations a_{lf}^{xy}, which are intermediate variables required to compute the final output of the network.

Several choices exist for the activation function $g(z)$. In both of the *HangNet* models, the Exponential Linear Unit (ELU) function is used due to its ability to yield negative output values, as opposed to other activation functions such as the Rectified Linear Unit (ReLU). The activation function ELU can be defined as:

$$g(z) = \begin{Bmatrix} z & z > 0 \\ \alpha \cdot (e^z - 1) & z <= 0 \end{Bmatrix} \tag{3.2}$$

Where α is a hyperparameter with a default value of 1.

Due to the large dimensionality of the input image, feature maps from early layers are typically down-sampled progressively to reduce their size, while allowing for larger filter banks at deeper layers with more abstract features. This operation is performed in a special type of layer called max-pooling layers. A max-pooling layer performs a sample-based discretization by applying a max filter to subregions of the feature maps.

As with the convolutional layer, this layer can also be described in terms of its parameters as $\mathtt{MaxPooling}(N, S)$, where N is the size of the subregion (an NxN patch) and S is the stride, which is typically selected to avoid subregion overlapping ($S = N$).

The architecture of the *HangNet* models for both regression and classification are described in the following sections.

3.4.1 *HangNet Regression Model*

For the regression problem, the corresponding *HangNet* model is a CNN composed of 4 sets of convolutional layers, and 2 fully connected layers to compute the output. Sets 1 and 3 are built from 2 convolutional layers with 16 filters of size (3x3) and an Exponential Linear Unit (ELU) as the activation function, followed by a max-pooling layer of size (2x2) and stride 2. Sets 2 and 4 are composed of a single convolutional layer with 32 filters of size (3x3) and ELU as the activation function, followed by a max-pooling layer of size (2x2) and stride 2. The fully connected layers have 300 and 3 neurons, respectively. For the regression problem, we use ELU as activation function for the last layer, representing the predicted 3D coordinates (X, Y and Z) of the garment at a given time step. The total number of learnable parameters of this model is 2 100 707. Figure 3.7 shows the *HangNet* model architecture for regression.

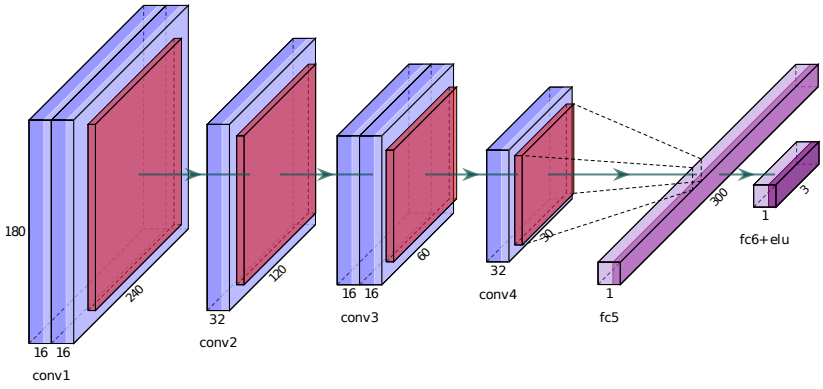

Figure 3.7: HangNet Regression Model architecture diagram.

The parameters are trained by optimizing a custom loss using an Adam stochastic optimizer. As the aim of the prediction is to hang garments, the most interesting information obtained from the prediction is the value of the Z coordinate. To emphasize in the importance of the Z coordinate, a custom weighted loss function was applied:

$$\text{loss} = \omega_x \cdot (X_{gt} - X_{pr})^2 + \omega_y \cdot (Y_{gt} - Y_{pr})^2 + \omega_z \cdot (Z_{gt} - Z_{pr})^2 \tag{3.3}$$

Where $\vec{X}_{gt} = (X_{gt}, Y_{gt}, Z_{gt})$ are the ground truth coordinates of the point in the training example, $\vec{X}_{pr} = (X_{pr}, Y_{pr}, Z_{pr})$ are the predicted coordinates, and ω_x, ω_y and ω_z are hyperparameters expressing the relative importance of each of the coordinate components.

3.4.2 *HangNet Classification Model*

For the classification problem, the corresponding *HangNet* model is composed of a total of 4 sets of convolutional layers, and 2 fully connected layers to compute the output. Sets 1 and 3 include 2 convolutional layers with 16 filters of size (3x3) and ELU as the activation function, followed by a max-pooling layer of size (2x2) and stride 2. Sets 2 and 4 are composed of a single convolutional layer with 32 filters of size (3x3) and ELU as the activation function, followed by a max-pooling layer of size (2x2) and stride 2. The fully connected layers have 300 and 1 neuron, respectively.

As only one class is to be predicted, instead of a conventional Softmax function, a Sigmoid activation function is applied to

the output neuron to obtain the probability of the garment falling to the floor given a certain input depth image. When the output is above a threshold of 0.5, the prediction is that the garment will fall, otherwise the prediction is that it will hang. The total number of trainable parameters of this model is 2 099 705. Figure 3.8 depicts the architecture of the classification network.

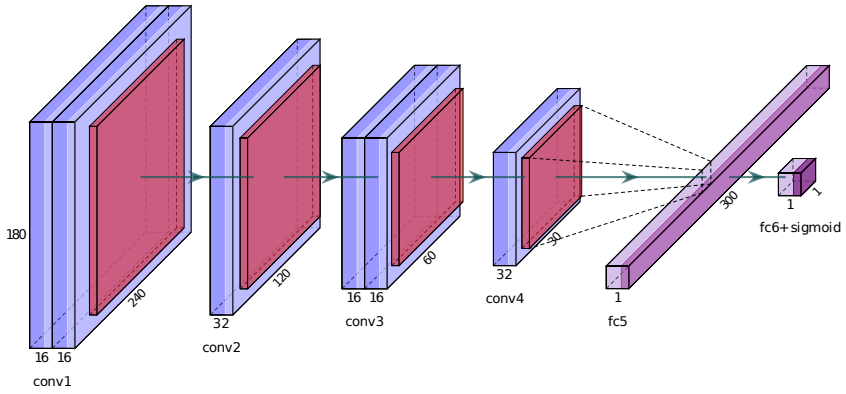

Figure 3.8: HangNet Classification Model architecture diagram.

During the simulation only the waypoints of the trajectory of the garment's center of mass while falling were recorded for each time step, so the binary labels for each training sample of the classification model were obtained from the recorded waypoints, based on the Z coordinate of the last point of the trajectory (Z_{end}). Depending on the final location of the center of mass, and considering a threshold T_{floor}, the binary label $\mathtt{floor}$ is computed as:

$$\mathtt{floor} = \begin{cases} 1, & \text{if } Z_{end} < T_{floor} \\ 0, & \text{otherwise} \end{cases} \tag{3.4}$$

The resulting labels are used as the ground truth to train the classification model. For training, a Binary Cross Entropy loss is optimized by an Adam stochastic optimizer. This loss is weighted accordingly to account for the imbalance of the hanged/floor classes in the synthetic dataset.

3.5 CHAPTER SUMMARY

It is important to dry garments before they can be further processed or stored after washing, to prevent odors and bacteria.

Hanging is a traditional method for drying clothes that has several advantages over using a drying machine. As the typical method for hanging clothes requires complex sensorimotor skills, a proposed simplified formulation of the hanging task is proposed: "dropping a garment over a clothesline or similar device so that it will stay hanged". Under this formulation, predicting whether a garment dropped over the hanger will fall or not becomes a key ability for the assistive robot.

To develop such ability two *HangNet* models are proposed, based on Deep Convolutional Neural Networks, for two different hanging prediction tasks: regression and classification. The regression task consists in estimating the final position of the garment from a depth image of the initial conditions before dropping the garment. In a similar way, the classification task aims to predict whether the garment will hang or fall based on the same initial information.

In the absence of existing datasets for this task, and due to the cost, time, and labor required to build a dataset with real world data to train both models, a synthetic dataset was built. The dataset was generated through cloth simulation in a 3D editor, from which the initial depth image of the grasped garment, and the trajectory described by the garment during its fall after it is released were recorded for several trials.

4

UNFOLDING

This chapter describes our advances in unfolding with robots, the second task of the laundry pipeline (Figure 4.1).

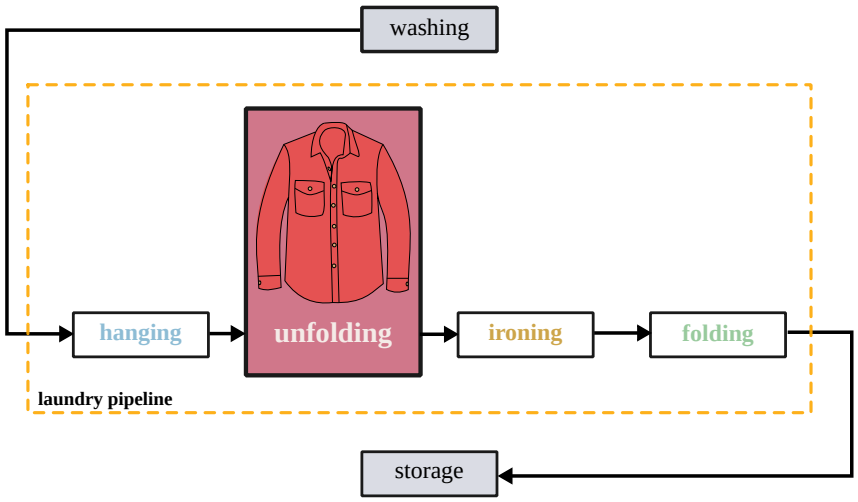

Figure 4.1: Unfolding is the second task of the laundry pipeline.

4.1 INTRODUCTION

Over the decades, people have developed various techniques to perform the different tasks in the laundry pipeline as efficiently as possible, sometimes resorting to specific techniques tailored for specific garment categories. For instance, the step sequence followed to fold a shirt is different than the one a person would execute to fold some pants or a skirt.

As a consequence, to apply the correct technique, it is highly useful to possess certain knowledge about the garment category to which the technique will be applied. Highly wrinkled and entangled garments are challenging to work with because it is extremely hard to recognize the garment category they belong to, and thus, for most techniques a previous unfolding step is required to place the garment in a fully spread configuration, noticeably increasing the chances of success of posterior steps.

Due to the complexity of bringing a clothing article to a concrete folded configuration stochastically, when performing cer-

tain tasks such as folding people usually follow a sequence of predefined steps that are known to bring the garment to a given folded configuration. Therefore, for this kind of tasks, the garment category has to be first recognized to be able to select the correct sequence.

On the other hand, the unfolding task only requires to bring the garment to a fully spread configuration, which is a simpler state to reach than folded, and thus no predefined sequence is required. For that reason, unfolding largely benefits from a model-less approach, in which the robot performing the task does not require any prior knowledge of the garment category to be able to successfully unfold it.

4.2 OVERVIEW

The method presented in this chapter is a model-less approach to unfolding that can be applied to any type of cloth or garment category to obtain a fully spread configuration. After it is fully spread, its category can be detected and the corresponding folding algorithm or ironing placement procedure can be applied.

The method assumes that a piece of clothing has already been picked up from a pile of garments, ideally dry, and placed flat over some working surface. For this, the most common method is to grasp the garment by one random point, pick it up, and grab the lowest point with the other arm. Once these two points are grasped, the garment can be laid flat onto the working surface, with some parts potentially overlapping the garment.

The aim of the method presented in this chapter is to remove the resulting overlapped parts to achieve a fully spread configuration. As input, the method takes an RGB-D image obtained by the robot using a stock RGB-D camera located in its head. The output obtained is the location of the 3D points where the robot should pick and place some garment part to put the clothing article in a configuration closer to the fully spread one.

The method can be divided into three consecutive stages:

- First, the **Garment Segmentation Stage** separates the garment from the background and extracts a simplified version of the garment contour.
- The next stage, the **Garment Clustering Stage**, uses the segmentation results from the previous stage and the depth

channel of the RGB-D image to locate the different overlapping regions in the garment.

- Finally, the regions are passed to the **Garment Pick and Place Points Stage**, that uses them in conjunction with the simplified garment contour to locate the fold axis and compute the most suitable pick and place points to remove the overlap.

This method can be applied recursively until all the overlapping regions have been spread out. Figure 4.2 depicts the complete unfolding pipeline.

Figure 4.2: Unfolding pipeline. The input of the system is the RGB-D observation of the garment, and the output are the pick and place points for the unfolding manipulation operation. As the pipeline is iterative, the sequence of three stages can be repeated until the garment is fully spread.

4.3 GARMENT SEGMENTATION STAGE

The objective of the Garment Segmentation Stage is to detect what elements of the input data belong to the garment, and which ones belong to the background, as well as to obtain a simplified representation of the garment outer shape or contour.

This objective can be achieved in different ways depending on the input data the robot has access to and the environmental conditions in which the robot is going to operate. It is assumed that only one garment is present in the input image, as the preprocessing operation described in Section 4.2 has already been performed.

If a single RGB-D image is to be used as input for this stage (Figure 4.3), the robot has to operate under the assumption that garments are typically more colorful than the background surrounding them, which is a reasonable assumption for most tables and other working surfaces that can be found in real domestic environments.

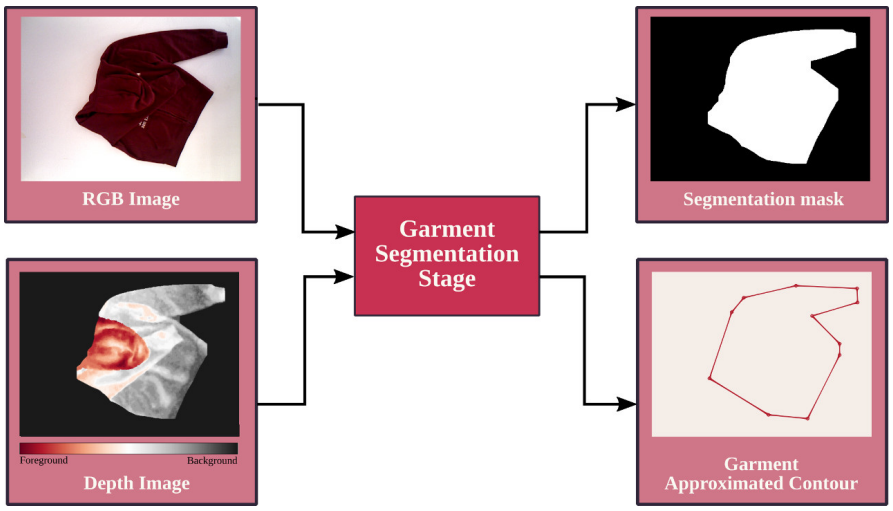

Figure 4.3: Garment Segmentation Stage using an RGB-D image, shown as an RGB and a depth image, as input.

In this case, the RGB components of the input image are converted to the HSV color space, and then the Saturation (S) and Value (V) channels are used to select colorful pixels (high amount of Saturation and intermediate amount of Value). To remove possible noise present in the input image, a Gaussian kernel is applied to the S and V channels as a noise filter, and then the optimal values for the threshold operations are found using Otsu's threshold selection method [65].

Once each of the channels has been thresholded, the binary images are combined with a bitwise AND operation. To remove existing holes in the resulting mask, a closing morphological operation is applied, followed by an opening operation.

The previous method is simple and can be computed in a fast manner, but as it is based only on color, its application is limited to certain kinds of garments and backgrounds. More robust results can be obtained if the working environment is scanned and reconstructed as a 3D point cloud, and that point cloud is used instead of a single RGB-D image (Figure 4.4).

This method has the additional benefit of achieving a garment reconstruction with a finer resolution, as several depth images are integrated into a single point cloud. To obtain such reconstruction, an existing algorithm such as KinectFusion [62] can be used.

Once the reconstructed point cloud is obtained, the next step is to correctly label the points belonging to the garment and the ones belonging to the flat surface supporting the garment. For such purpose RANSAC [19] is used to fit a plane represent-

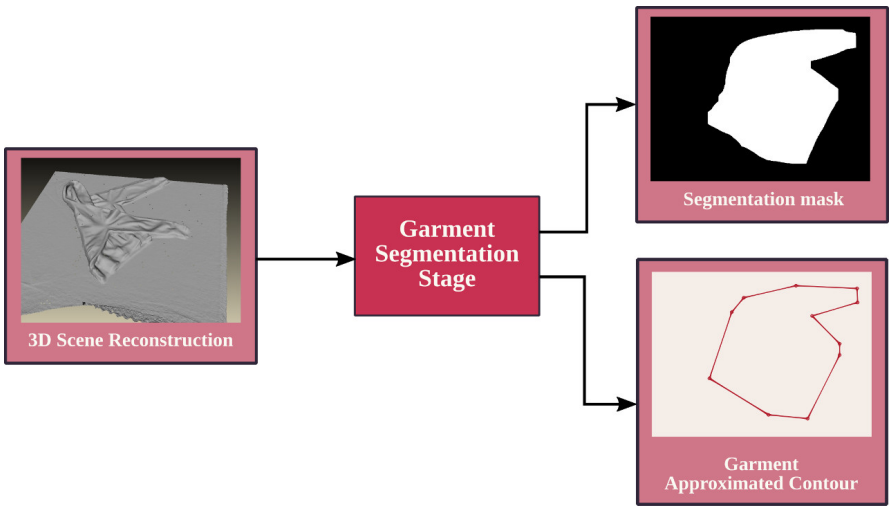

Figure 4.4: Garment Segmentation Stage using a 3D reconstruction of the scene as input.

ing the working surface to the point cloud, where the points matching the model are considered background and removed from the point cloud.

Then, Euclidean Clustering is used to locate and label the points belonging to the garment, as it is assumed to be the largest cluster found by the clustering algorithm. The moments of inertia of the garment cluster are computed and used to generate a 3D bounding box that encloses the garment. Based on the height and spacing of the points inside the 3D bounding box a depth image representing the garment from a top view can be obtained by discretizing the bounding box in different regions, that correspond to pixels of the generated depth image. For each of the regions the highest point is found, and its height, measured with respect to the base of the bounding box, is stored in a 2D image.

Once this process has been completed for all the regions, a 2D image is obtained that, after being normalized and converted to a 8-bit representation, can be used as depth map in later stages. To obtain the segmentation mask a similar process is followed, but considering the occupancy of each region as the criteria to create the 2D binary image representing the segmentation mask.

Using any of the two different methods proposed for garment segmentation, a binary mask representing the pixels that belong to the clothing article can be obtained. A blob labeling algorithm is then applied to the mask to obtain the 2D projection of the garment contour, that is further simplified using the

Ramer-Douglas-Peucker [71][12] algorithm to obtain the simplified garment contour. Both the binary mask and the simplified garment contour will be used in later stages to determine the most suitable points for the unfolding operation.

4.4 GARMENT CLUSTERING STAGE

Once the garment has been successfully segmented from the background the next stage, the Garment Clustering Stage, is applied. This stage employs the depth information from the 3D reconstruction of the garment to both group together pixels belonging to the same garment region, and to find the overlap relationship between them, that is, which garment region is overlapping or being overlapped by another region.

The clustering stage (Figure 4.5) operates under two main assumptions: that pixels with similar height values belong to the same garment region, and that regions with a higher height value overlap regions with a lower height value. Ideally, a single cluster per region would yield the best performance in the next stage, but our method also supports smaller, superpixel-like regions that are more likely to appear in presence of noise in the input data or wrinkled regions.

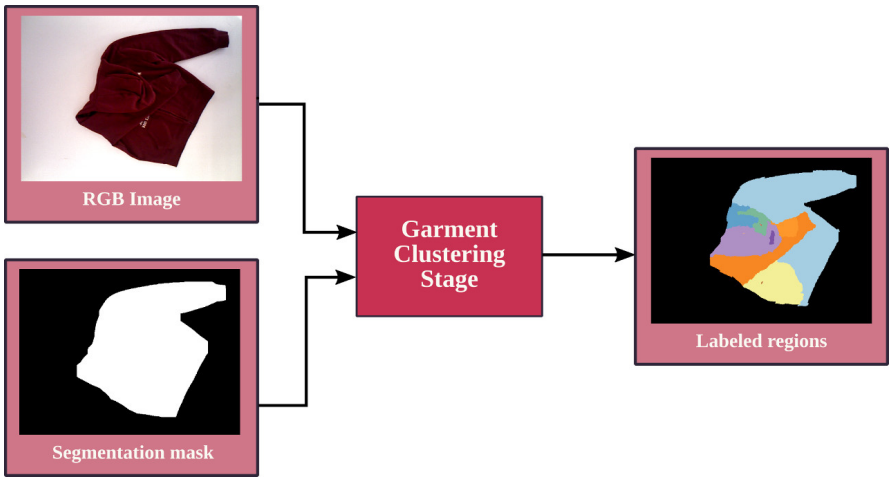

Figure 4.5: Garment Clustering Stage.

Using the segmentation mask obtained in the previous stage, only depth pixels belonging to the garment are selected and passed to the next step that performs height clustering.

To find the garment region clusters, the Watershed algorithm [77] is used. The Watershed algorithm interprets the values of the pixels in a grayscale image as heights, in a similar way a to-

pographic map represents relief in a terrain. The algorithm then starts flooding the "terrain" starting from the lower regions until the basin from different points meet at the watershed lines. The watershed lines represent the boundaries of the different regions or clusters. Figure 4.6 illustrates the Watershed algorithm.

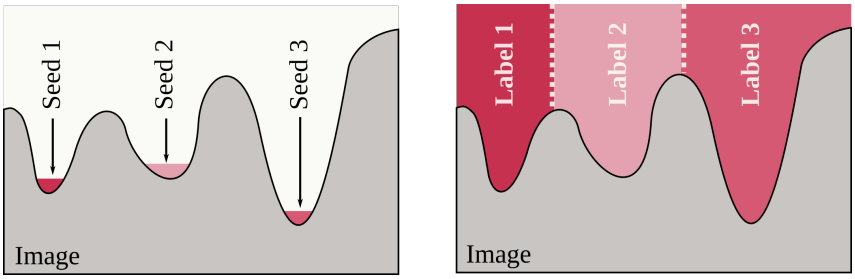

Figure 4.6: Graphical representation of the Watershed algorithm. The image represents a topografic relief that is filled with water from the different seed points. The watershed lines, shown as dotted lines in the right figure, indicate the boundaries of the different labeled clusters.

Once the different clusters have been identified, the values of the pixels in each region are substituted with the mean or median value for all the pixels of that region, to obtain homogeneous regions and filter out the noise present in the input depth image. The clustered image, as well as the highest cluster and location of the highest point within that cluster are recorded and passed to the next stage to compute the most suitable pick and place points for the unfolding operation.

4.5 GARMENT PICK AND PLACE POINTS STAGE

A simple way to describe the unfolding operation is to specify two points: a point to pick the garment or garment part, and a point to release it. The aim of the last stage of the unfolding algorithm, the Garment Pick and Place Points Stage, is to find the most suitable pick and place points to successfully perform the unfolding operation. This stage requires as input both the simplified garment contour obtained in the Garment Segmentation Stage and the clustered depth image from the Garment Clustering Stage (Figure 4.7).

First, a candidate set of unfolding paths is generated under the assumption that the fold has at least one contour edge that also belongs to the observed contour of the garment. To generate each of the paths, a straight line is constructed by joining

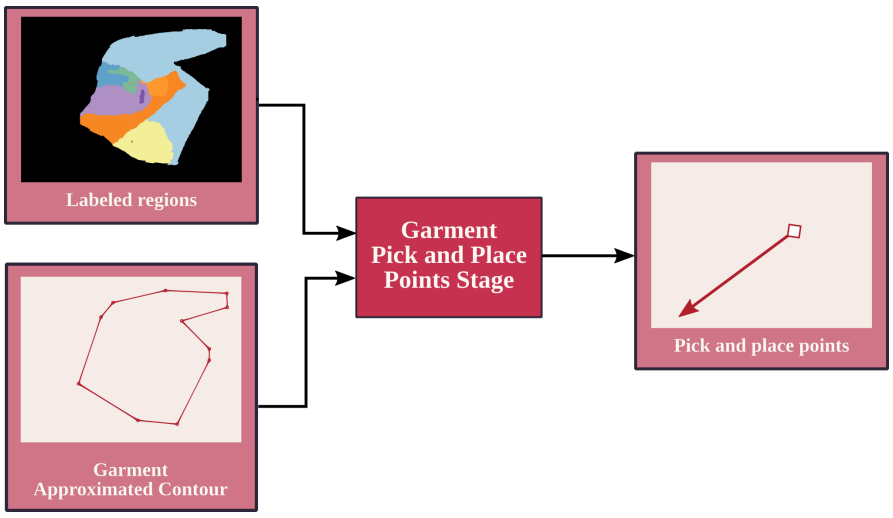

Figure 4.7: Pick and place points computation.

the midpoint of each of the simplified garment contour edges with the highest point of the highest cluster region. If any of the candidate paths intersects the simplified garment contour at any point other than the two points used to generate the path, that path is discarded, so that no paths that go outside of the garment are used as candidates for the next step.

For each of these candidate paths, a metric representing the changes and discontinuities in height of the different garment regions along the path is computed. This metric is called *bumpiness* (B), and the procedure to compute the *bumpiness* value is the following.

Each of the candidate paths, $\mathtt{path}$, is a 2D parametric line ($\mathbb{R} \to \mathbb{R}^2$) described by a single parameter r corresponding to the radial distance from the highest point to a given point of the line:

$$\mathtt{path}(r) = [u(r), v(r)] \tag{4.1}$$

Each of the paths, of length L, is divided in segments of constant length l, obtaining a total of m sampled height values:

$$m = \left\lfloor \frac{L}{l} \right\rfloor \tag{4.2}$$

The clustered depth image $\mathcal{D}(u, v)$ is sampled at each of the m discrete points for each of the n candidate paths, generating

n ordered sets $S = \{s_1, ..., s_m\}$, in which each element s_i of the set is computed as follows:

$$s_i = \mathcal{D}(path(i \cdot l)), \quad i = 0, 1, 2, ..., m\ , \tag{4.3}$$

To obtain the *bumpiness* value B_{path} of a given path, the accumulated relative difference of each set of m sampled height values s_i is computed for the set corresponding to that path:

$$B_{path} = \sum_{i=2}^{m} |s_i - s_{i-1}|, \tag{4.4}$$

Once the *bumpiness* value B_{path} is computed for all candidate paths, the path with the lowest value is selected and the corresponding edge of the simplified garment contour is labeled as fold edge. A large *bumpiness* value indicates that there is one or more large discontinuities along the path, which means that the path goes through regions belonging to two different overlap levels. A path with low *bumpiness*, on the other hand, indicates that the regions it crosses possess an approximately continuous height value and therefore belong to the same overlapping region.

Once the fold edge has been located, the pick and place points are selected. The pick point is computed by extending the selected candidate path until it intersects with the boundary of the highest cluster. If more than one intersection points exist, the one with the largest distance to the fold edge is selected.

To obtain the place point, the fold edge is used as symmetry axis to compute the mirror image of the pick point. The complete unfolding algorithm (Segmentation, Clustering and Pick and Place Points Stages) can be repeated after the unfolding operation has been performed until the garment is completely unfolded.

The described process to obtain the pick and place points is visually summarized in Figure 4.8.

4.6 MANIPULATION

Once the pick and place points have been determined, the corresponding manipulation operation is computed from them and commanded to the robot. The typical manipulation operation

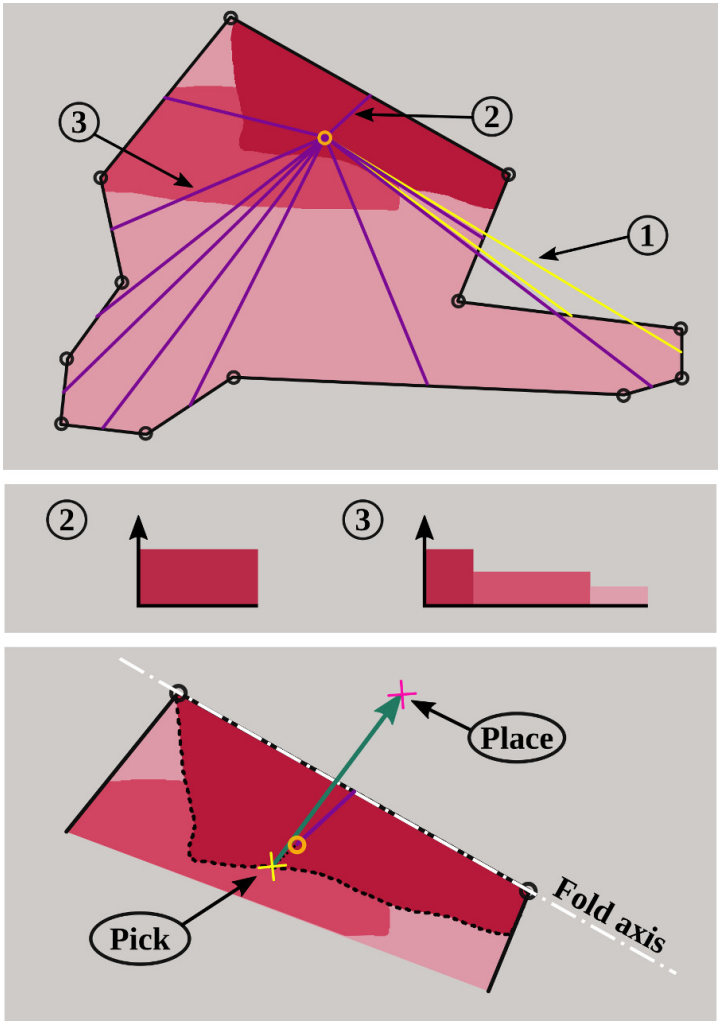

Figure 4.8: The top figure shows the different watershed clusters of a given garment, with the corresponding garment contour and the candidate paths. (1) is one of the two candidates discarded (yellow lines), as they intersect with the garment outline more than once. (2) and (3) are two paths for which the height profile is shown on the middle figure. (3) has a higher bumpiness value than (2), as there are several changes in height along the path. The bottom figure shows the computation of the pick and place points as described in this subsection, including the highest region outline used to compute the pick point and the fold axis used to locate the place point.

for unfolding is composed of three actions: grasping the garment, following a trajectory computed from the pick and place points, and releasing the garment.

A critical element for the grasping and releasing actions is the end-effector or gripper installed on the robot. Depending

on the task, different choices exist for grippers, that range from claw-like grippers to anthropomorphic hands. For cloth-related manipulation, some designs exist in the literature that allow to grab very thin garments by scooping [47] or using rolling elements [33]. Since the experiments performed in this thesis[1] were executed by two different platforms, that were designed with general purposes in mind, custom grippers were developed for each specific platform. The equipment and infrastructure already present on each of the platforms were taken into account for the design of each of the grippers, resulting in two different gripper models.

For the humanoid robot, a 3D printed gripper was designed, featuring two single-link fingers actuated by two hobby servomotors. The angular position of the fingers is controlled by an Arduino UNO board[2] which serves as an interface with the robot. It receives open and close commands and translates them into angular positions for the gripper fingers. The gripper is small and lightweight, as it is aimed to be installed in a humanoid robot arm, but at the same time the servomotors provide enough force to hold most common garments such as T-shirts and pants.

For the industrial robot, a different 3D printed gripper was developed and integrated with the existing pneumatic system, which is controlled through standard general purpose digital signals of the robot. This gripper is designed to be opened and closed with a linear action, while obtaining a certain amount of compliance for grasping garments from the pneumatic system that actuates the gripper. The tip of the gripper is coated with EVA foam to increase the friction between the cloth and the gripper, as the surface of the 3D printed plastic used for the gripper is too smooth to provide enough friction. Figure 4.9 depicts the two different gripper models for the two robot platforms to be used in the experiments.

When working with garments laying over a flat surface, the design of both grippers ensures a successful grasp in most situations without any computation of the grasping orientation. By approaching the garment using an orientation perpendicular to the flat surface, the gripper is able to correctly grasp the garment.

Additionally, by controlling the height of the grasping point, the amount of garment layers grasped can be controlled. This

1 The experimental evaluation will be described in detail in Section 7.2.

2 https://store.arduino.cc/arduino-uno-rev3, last visited: 04-06-2020

Figure 4.9: Grippers for the unfolding task. On the left, the gripper to be installed in the humanoid robot is depicted performing a grasping test. On the right, the custom gripper to be installed in the industrial robot is shown.

is useful, as when the garment presents an overlapping fold, at least two layers are involved. But, even in the case of manipulating a single garment, it may involve dealing with many cloth layers. For instance, a fully extended T-shirt has two cloth layers: the front and the back of the garment.

For the trajectory, a path has to be selected to move the gripper from the pick point to the place point, lifting the corresponding garment part to move it to the new location. Several paths may be used for this purpose.

The simplest path is the rectangular one, consisting on three straight segments. The first one, the lift segment, is composed by a source point equal to the pick point and a destination point with the same x and y coordinates than the source, but its z coordinate is incremented by a small offset. A second segment connects the ending of the previous segment with the start of the next one, keeping a constant z coordinate. The last segment lowers the garment part before its release, starting with a point with the same x and y coordinates than the place point, but with the same z coordinate as the ending of the previous segment, and ending in the place point. For a given pick point $P_{pick} = (x_{pick}, y_{pick}, z_{pick})$ and place point

$P_{place} = (x_{place}, y_{place}, z_{place})$, the rectangular path is defined as:

$$path_{rect}(t) = \begin{cases} (x_{pick}, y_{pick}, z_{pick} + h \cdot t) & \forall t \in [0,1) \\ (x_{pick} + (x_{place} - x_{pick})(t-1), & \\ y_{pick} + (y_{place} - y_{pick})(t-1), & \\ z_{pick} + h) & \forall t \in [1,2) \\ (x_{place}, y_{place}, & \\ z_{pick} + h + (z_{place} - z_{pick} - h)(t-2)) & \forall t \in [2,3] \end{cases} \tag{4.5}$$

Where h is the vertical offset, measured with respect to the pick point, and $t \in [0,3]$ is a parameter describing the advance of the end-effector along the path.

The rectangular path has the advantage of being simple to compute and implement, but it can result in a movement excessively abrupt for some pick and place locations. If the lift and lower segments are converted from vertical lines to oblique lines, with angles α and β respectively for each segment, a smoother trapezoidal path can be achieved. The trapezoidal path can be defined in terms of the 4 points defining the three segments of the path:

$$path_{trap} = \begin{cases} P_1 = P_{pick} = (x_{pick}, y_{pick}, z_{pick}) & \\ P_2 = (x_{pick} + \frac{h}{\tan(\alpha)} \cos(\arctan(\frac{y_{place} - y_{pick}}{x_{place} - x_{pick}})), & \\ y_{pick} + \frac{h}{\tan(\alpha)} \sin(\arctan(\frac{y_{place} - y_{pick}}{x_{place} - x_{pick}})), & \\ z_{pick} + h) & \\ P_3 = (x_{place} + \frac{h - (z_{place} - z_{pick})}{\tan(\beta)} \cos(\arctan(\frac{y_{place} - y_{pick}}{x_{place} - x_{pick}})), & \\ y_{place} + \frac{h - (z_{place} - z_{pick})}{\tan(\beta)} \sin(\arctan(\frac{y_{place} - y_{pick}}{x_{place} - x_{pick}})), & \\ z_{pick} + h) & \\ P_4 = P_{place} = (x_{place}, y_{place}, z_{place}) & \end{cases} \tag{4.6}$$

A fold can be described as a part of the garment that has been rotated about 180° around the fold axis. Based on this de-

scription, each of the points belonging to the folded part of the garment can be considered to have followed a circular path. If the fold has to be performed or removed with precision, the most suitable path is therefore a circular path, that can be described in terms of the pick point $P_{pick} = (x_{pick}, y_{pick}, z_{pick})$ and place point $P_{place} = (x_{place}, y_{place}, z_{place})$ as:

$$path_{circ}(t) = \begin{cases} \frac{x_{pick}+x_{place}}{2} + R \cdot \cos(t \cdot \pi) \cdot \cos(\arctan(\frac{y_{place}-y_{pick}}{x_{place}-x_{pick}})) \\ \frac{y_{pick}+y_{place}}{2} + R \cdot \cos(t \cdot \pi) \cdot \sin(\arctan(\frac{y_{place}-y_{pick}}{x_{place}-x_{pick}})) \\ \frac{z_{pick}+z_{place}}{2} + R \cdot \sin(t \cdot \pi) \end{cases} \tag{4.7}$$

Where $t \in [0, 1]$ is a parameter describing the advance of the end-effector along the path, and R is the radius of the circular path, computed as:

$$R = \frac{1}{2} \cdot \left\| \vec{P}_{pick} + \vec{P}_{place} \right\| \tag{4.8}$$

More advanced choices of folding paths exist, such as Bézier curves [50], that can be selected depending on the garment characteristics and requirements of the movement to be performed. For folding, the expected result is a garment part being placed over a specific location. Therefore, the choice of a manipulation path is more critical, and circular or Bézier curve-based paths are more appropiate choices. In this case, choosing an incorrect path, as could be one that is too abrupt, might cause slipping of the cloth during the movement, resulting in a displacement of the garment that will prevent it from reaching the desired target pose.

On the other hand, the requirements for unfolding are much less strict. Even if the garment moves slightly as a result of pulling it while following the selected path, the result will not be affected, as the garment will be unfolded nonetheless. Therefore, for the unfolding experiments, rectangular or trapezoidal paths were typically selected for both robots.

4.7 CHAPTER SUMMARY

Many tasks in the laundry pipeline benefit from having some prior knowledge about the garment category with which the

robot is working. As recognizing the garment category from a crumpled or entangled piece of clothing is a highly difficult task, an unfolding operation has to be performed to spread the garment in preparation for the recognition algorithm.

This chapter describes the garment unfolding algorithm, that is composed of three stages: the Garment Segmentation Stage, the Garment Clustering Stage, and the Garment Pick and Place Points Stage.

The Garment Segmentation stage uses RGB-D information from the robot perception system to obtain a segmentation mask and simplified garment contour. Then, the Garment Clustering Stage groups pixels belonging to the different overlapping regions present in the depth image and locates the highest cluster and highest point within that cluster. With all the information computed in previous stages, the Garment Pick and Place Points Stage finds the most suitable pick and place points for a successful unfolding operation based on a custom metric called *bumpiness*.

For the final manipulation operation, two different grippers were designed for each of the robots to be used. Even though several choices of folding paths exist, for the unfolding experiments rectangular and trapezoidal paths were selected as a good compromise between complexity of the movement and a successful unfold.

5

IRONING

This chapter describes our advances in ironing with robots, the third task of the laundry pipeline (Figure 5.1).

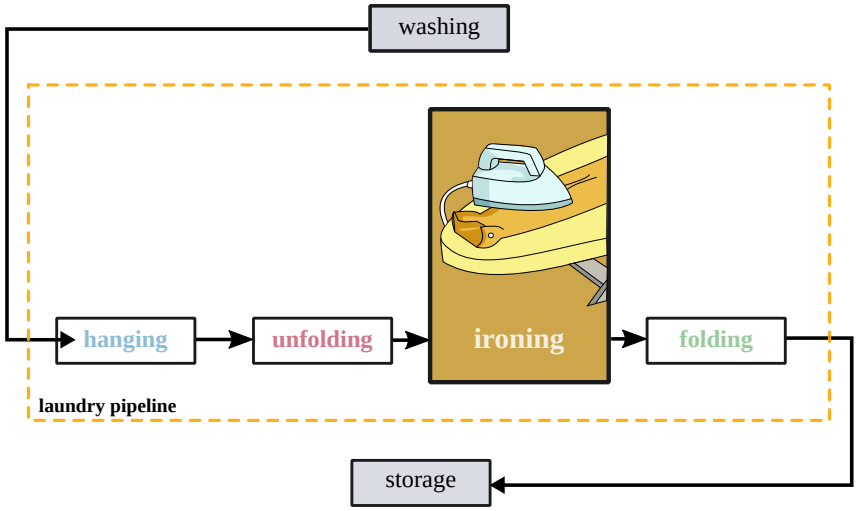

Figure 5.1: Ironing is the third task of the laundry pipeline.

5.1 INTRODUCTION

Washing clothes responds to a necessity in terms of both hygiene and garment maintenance, as garments become increasingly dirty the longer they are being worn, due to their continuous contact with the environment and the human body. On the other hand, wrinkle removal is performed purely for aesthetic purposes, as people find ironed clothes more aesthetically pleasing. In some cases, winkled clothes can even be associated culturally with a lower status or a not careful enough person, similar to the association made in the case of people wearing dirty clothes.

As a consequence, people tend to remove wrinkles from their garments before wearing them. Even if garments are not going to be worn right after being washed, wrinkle removal is performed before storage, to prevent wrinkles from appearing or becoming harder to remove.

Two different kinds of wrinkles can be distinguished in clothing articles: *soft wrinkles* and *marked creases*, both depicted in

Figure 5.2. The first ones, *soft wrinkles*, are large undulations present in cloths due to the deformable nature of the textile materials of which they are made. The more flexible the material is, the smaller the *soft wrinkles* that tend to be formed. *Soft wrinkles* appear due to the interaction of the garment with objects and elements of the environment or other parts of itself, and they can be removed by flattening the garment, either by pulling the outer rim of the garment outwards or by passing a flat object on top of the garment surface.

Figure 5.2: Soft wrinkles (left) and marked creases (right) on the same garment.

Marked creases, on the other hand, are sharper and smaller than *soft wrinkles* and appear due to changes in the bonding forces of fibers that occur during the washing process. Due to the water and heat applied to the clothing article while washing it, fiber bonds in the garment are allowed to shift to new positions, creating permanent *marked creases*, commonly known as "wrinkles"[1]. As *marked creases* occur at fiber bond level, they will persist any attempt to remove them by flattening them out through manipulation alone.

In order to successfully remove *marked creases*, heat and pressure have to be applied to the garment to reset the fiber bonds to their flattened positions. This process is called ironing, and it is typically performed with the help of an electric iron, although other devices and methods based on steam also exist.

Despite the existence of devices such as washing machines, that can easily wash any type of garment with very little assistance from a human, there is not a similar machine able to perform the ironing task for any kind of garment in an automated fashion. The most recent innovations in that direction are still bulky and expensive, and are only capable to process

1 Although both *soft wrinkles* and *marked creases* are commonly referred by people as "wrinkles", the author will keep a different naming throughout the thesis to be able to differentiate them.

certain garment categories, requiring extensive assistance from a person to set or manipulate the clothing article.

Having to own a specific machine for each laundry task is both expensive and requires a large amount of dedicated space for these devices in the household. To solve this issue, the author proposes to use a humanoid robot to be able to perform all these tasks, as a Robot Household Companion.

5.2 OVERVIEW

As discussed in the related work section (Section 2.3), existing approaches to ironing focus on static ironing of individual *marked creases* by locating and targeting them one by one. Due to the small size of *marked creases*, this method requires a working environment with controllable illumination, which is easily achievable in an industrial setting, but almost impossible to obtain in a real domestic setup.

As one of the objectives of this thesis is the application of the developed methods in an unmodified domestic environment, an alternative approach to the ironing task is required. For this reason, the approach proposed in this thesis is based on observations of how people perform the ironing task. Instead of looking for individual *marked creases* to apply the iron to them one by one, people iron the whole garment surface dynamically, focusing on avoiding the creation of additional *marked creases*.

This can occur if the heat and pressure from the iron is applied to a *soft wrinkle* incorrectly, altering the fiber bonds in such a way that, instead of removing an existing *marked crease*, a new one is created. This process of iteratively ironing the whole surface is repeated until the whole surface is free of *marked creases*, focusing on regions with a larger amount of remaining *marked creases*.

The main advantage of this method is that the critical element is shifted towards how to iron *soft wrinkles*, which are larger in size, and therefore easier to locate with the current technology in RGB-D sensors. This approach also removes the necessity of detecting individual *marked creases*, allowing its application in environments in which illumination cannot be controlled by the robot.

The ironing approach proposed in this thesis is as follows. First, a 3D reconstruction of the working environment is built, and the garment data is segmented from the rest of the back-

ground. Then, a wrinkle descriptor is computed to locate the regions of the garment in which *soft wrinkles* are present.

Once the location of these regions is known, an ironing path is computed in such a way that the *soft wrinkles* can be removed with the next ironing operation, to avoid creating new *marked creases* when ironing. Finally, an ironing path-following controller allows the robot to apply the iron to the garment with the correct amount of pressure, while following the previously computed path. Figure 5.3 depicts the complete ironing pipeline.

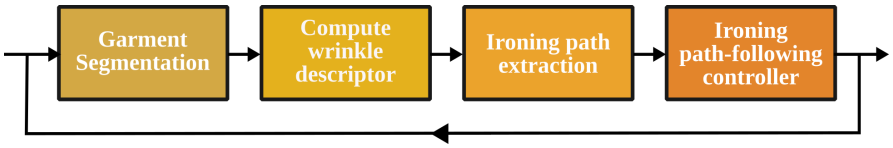

Figure 5.3: Ironing pipeline. The input of the algorithm is a 3D reconstruction of the working environment, which is used to compute and perform the most suitable ironing path. The sequence is repeated iteratively until all the *marked creases* have been removed.

The proposed approach offers the following contributions:

- It applies the iron in a dynamic way which, as opposed to other static ironing methods, results in a faster ironing process, while reducing the chances of burning the garment, as it applies heat uniformly over a wider area.
- The proposed approach does not require a light-controlled environment, as there is no need to locate individual *marked creases* to apply the iron specifically to each of them, and thus it can be applied to a real domestic environment.
- The ironing operation uses a force/torque-based controller, removing the need for any external mechanism providing passive compliance to cope for the lack of control over the force exerted to the garment on the working surface.

5.3 GARMENT SEGMENTATION

The first stage of the ironing algorithm consists in obtaining a 3D reconstruction of the garment surface. For this purpose, a 3D reconstruction of the whole working environment is built by integrating several views obtained with an RGB-D sensor using the KinectFusion [62] algorithm.

In addition to the garment data, the resulting reconstruction includes elements from the background and ironing board that need to be removed. To achieve this, the RANSAC [19] algorithm is used to locate different planes in the image, and is combined with a set of heuristics to locate the most significant elements of the reconstruction:

- The ground plane will be the horizontal plane with the lowest z coordinate from all the horizontal planes detected.
- The ironing board legs and the part of the garment that hangs from the board typically form a vertical plane close to the robot.
- The top of the ironing board, which contains the region of the garment we are interested in, is the remaining horizontal plane.
- Any other elements of the background can be filtered out based on their distance to the robot, as they will be further away from the robot than the rest of the objects of interest.

Once the top of the ironing board is located and extracted, the resulting point cloud will contain data from both the garment and the ironing board surface. It is assumed that points from both distributions are linearly separable based on color and location, in such a way that two sets of points X_0 and X_1 can be obtained such that:

$$\sum_{i=1}^{n} \omega_i x_i > k \qquad \forall x \in X_0 \tag{5.1a}$$

$$\sum_{i=1}^{n} \omega_i x_i < k \qquad \forall x \in X_1 \tag{5.1b}$$

Where $\omega_1, \omega_2, \ldots, \omega_n$ are weights, x_i is a point in the joint initial distribution, and k is a real number.

Based on the previous assumption, the set of n points contained in the ironing board plane can be partitioned into $k = k_G + k_B$ different clusters, with k_G containing all the points that belong to the garment distribution, and k_B containing the points that belong to the ironing board distribution.

To limit the number of possible clusters to consider, uniform color and a Lambertian reflectance model are assumed for the garment surface and the ironing board, so that the number

of clusters to be considered is $k = 1 + 1 = 2$. In addition, to improve the separability of the clusters, the color space of the point cloud is converted from the RGB space to the HSV space.

After the color space conversion, the feature vector $\vec{x}$ encodes both the spatial location of the point and the HSV color components:

$$\vec{x} = (x, y, z, h, s, v) \tag{5.2}$$

Encoded using the feature vector $\vec{x}$, the points in the region of interest extracted in the previous step can now be partitioned into the garment and ironing board clusters (k_G and k_B, respectively) by iteratively minimizing the within-cluster sum of squares with the following general expression:

$$\arg\min_{S} \sum_{i=1}^{k} \sum_{\vec{x} \in S_i} \|\vec{x} - \mu_i\|^2 \tag{5.3}$$

Where k is the number of clusters (in our case $k = 2$), S_i is the i-th cluster in $S = S_1, S_2, \ldots, S_k$, $\vec{x}$ is the feature vector, and μ_i is the mean of all the feature vectors in S_i.

Once both clusters have been obtained, each of them can be labeled as either "garment" (k_G) or "ironing board" (k_B) based on prior knowledge about the location and orientation of the ironing board. Only one of the two extremes of a typical ironing board is prepared to hold a garment, so the garment cluster must lay in that direction; as opposed to the ironing board cluster, that must correspond to the opposite direction. The centroids of both clusters are used to determine their relative position, that is matched against the aforementioned prior.

Once the garment cluster is located and labeled, smaller subclusters that might appear within that cluster are filtered using Euclidean clustering to ensure a single, consistent cluster. Figure 5.4 shows the different steps of the segmentation stage.

5.4 WRINKLENESS LOCAL DESCRIPTOR (WILD)

After the set of points belonging to the garment surface has been extracted from the input point cloud, the garment surface has to be analyzed to determine the location of *soft wrinkles*. *Soft wrinkle* localization is critical to compute a successful ironing

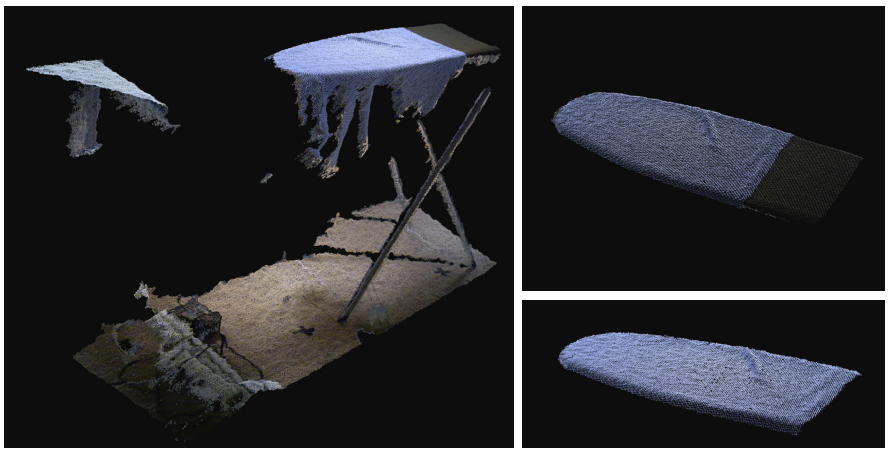

Figure 5.4: Ironing segmentation steps: 3D reconstruction of the working environment (left), segmentation of the top of the ironing board (top right) and garment top segmentation (bottom right).

path, as ironing a *soft wrinkle* incorrectly could lead to the creation of new additional *marked creases*, instead of removing the existing ones. As *soft wrinkles* are regions of the garment with a certain curvature and a high volume footprint, a curvature analysis must be performed to the garment surface to estimate the location of *soft wrinkles*.

A classic Hessian matrix analysis could be used to determine the wrinkle location, as well as some general-purpose 3D descriptors that use curvature to characterize objects, such as the Radial Surface Descriptor (RSD) [58]. Instead, pursuing an increase in performance with respect to existing methods, the author developed a custom descriptor, tailored to suit the needs of this particular problem (i.e. locate wrinkles in garments).

The resulting descriptor, the Wrinkleness Local Descriptor (WiLD), measures how abrupt is the curvature around a point by comparing its normal with the normals of neighboring points. To compute the neighbors of each point (and their normals) in the most efficient way, a kd-tree or other kind of indexed space representation might be constructed first. Then, for each of the points in the garment surface, the WiLD descriptor value can be computed according to the following expression:

$$\mathrm{WiLD}(\vec{i}) = \frac{1}{k} \cdot \sum_{\vec{j} \in K} \vec{n}_{\vec{i}} \cdot \vec{n}_{\vec{j}} \tag{5.4}$$

Where $\vec{i}$ is the point for which the WiLD descriptor is being computed, K is the set of k nearest neighbors of $\vec{i}$, each of them

denoted by $\vec{j}$, and $\vec{n}_{\vec{i}}$ and $\vec{n}_{\vec{j}}$ are the normals of $\vec{i}$ and $\vec{j}$, respectively.

For flat regions, the product of normals will result in a higher value, and therefore the WiLD descriptor for that region will be close to one. On the opposite side, a region with a very abrupt curvature with yield a product of normals close to zero, resulting in a low WiLD value. Figure 5.5 depicts both scenarios.

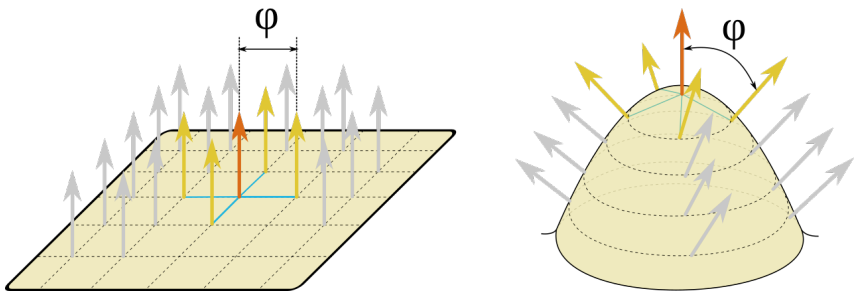

Figure 5.5: On flat regions (left), the angle between the neighboring normals, φ, is 0, and therefore the value of the WiLD descriptor is close to 1. On the other hand, as the surface curvature increases, the angle formed between two neighboring normals gets closer to 90°, and the WiLD descriptor value is closer to 0. Note that some normals were omitted in the figure to improve visualization.

As *soft wrinkles* are regions with a high curvature, but not as abrupt as, for instance, *marked creases*, their values of the WiLD descriptor will lay in the intermediate values, and can be located by thresholding:

$$wrinkle(\vec{x}) = \begin{cases} 1, & \text{if } L_t < \text{WiLD}(\vec{x}) < H_t \\ 0, & \text{otherwise} \end{cases} \tag{5.5}$$

Where L_t and H_t are the lower and higher thresholds, determined empirically.

5.5 IRONING PATH EXTRACTION

Once the regions presenting *soft wrinkles* have been located, a suitable ironing path has to be computed to be able to remove the *soft wrinkles* while avoiding the creation of new additional *marked creases*. Inspired by observations of how people perform the *soft wrinkle* removal, an ironing path extraction method has been developed that uses the location of the wrinkled regions as

well as some heuristics to determine the most suitable ironing path.

The assumption is that the most effective way to remove this kind of wrinkles is to apply the iron perpendicularly to the crest or spine of the *soft wrinkle* towards the edge of the ironing board, so that the air trapped below the *soft wrinkle* can be released and the garment can recover its original flatness. As the air has a pathway to escape, this strategy reduces the chances of creating new *marked creases* by unintentionally folding the *soft wrinkle* regions and ironing over the folded material.

As the garment surface can be considered mostly flat on top of the ironing board, the previously computed raw wrinkle descriptor is projected over the garment / ironing board plane and then discretized to obtain a 2D image. Possible occlusions due to several points of the point cloud belonging to the same image pixel bucket are handled using z-buffering. A similar approach, but based on the occupancy of each of the pixel buckets is used to obtain a 2D segmentation mask that identifies regions with descriptor values as garment and regions without WiLD data as part of the background.

To deal with possible artifacts from empty pixel buckets due to noise or missing points during the discretization process, a binary closing morphological operation is applied to the binary mask. The erosion applied for the closing operation is slightly larger than the dilation to remove the outer border of the garment. As the garment placed in the ironing board is pulled down by effect of gravity, and due to the shape of the ironing board, the outer border of the garment will be always curved and can be ignored to improve the wrinkle detection capabilities of the WiLD descriptor. The contour of the garment is then obtained from the processed mask, as it will be required in a later step.

Equation 5.5 is then applied to the normalized WiLD 2D image to locate the wrinkled regions, that are labeled using a standard blob labeling algorithm. As any of the resulting regions could be a equally suitable region to apply the next ironing operation, the strategy selected is to start ironing the region with the largest area, so the blob with a larger area size (in pixels) is selected to perform the next steps.

To find the ironing path, the selected blob is skeletonized using the Zhang-Suen [108] algorithm, and the resulting pixels are converted into a graph based on their connectivity. Depending on the shape of the wrinkle blob, it is possible for the graph

obtained to have more than one branch. To determine the starting and final points of the ironing path, the following criteria is used:

- The starting point corresponds to the leaf node of the graph closest to the garment contour extracted from the processed segmentation mask.
- The final point is the leaf node of the graph that is furthest from the garment contour.

By applying these rules, the ironing operation will always be performed from the interior of the garment towards the outer rim, allowing the release of the air trapped below the *soft wrinkle*. To recover the full path from the start and ending points, a depth-first search is run on the skeleton graph using the start point as the root node and the final point as the target.

After the method has been applied, the obtained waypoints are then converted back to 3D space before the robot executes the corresponding ironing action. Figure 5.6 depicts the different steps of the path extraction stage.

5.6 IRONING PATH-FOLLOWING CONTROLLER

Once the most suitable ironing path has been computed, the robot performs the ironing operation through a path-following ironing hybrid controller. The objective of this controller is to exert a certain force with the iron over the garment, while following the waypoints computed in the previous stage as reference.

Let $w = \{w^{(0)}, ..., w^{(N)}\}$ be the set of waypoints obtained by the perception system in the previous stages of this method. Each of the waypoints is described by its Cartesian components as $w^{(i)} = (w_x^{(i)}, w_y^{(i)}, w_z^{(i)})$, where the components for each waypoint are expressed in the robot base reference system. To obtain the waypoints, the ironing path computed in the WiLD image frame of reference in the previous stage is first converted back to the RGB-D sensor frame of reference, and then to the robot root frame of reference. Figure 5.7 shows the different reference systems for this task.

Starting from an initial resting position, the robot has to reach the starting point of the trajectory, while avoiding any collisions with the ironing board. For such purpose, two movements are performed.

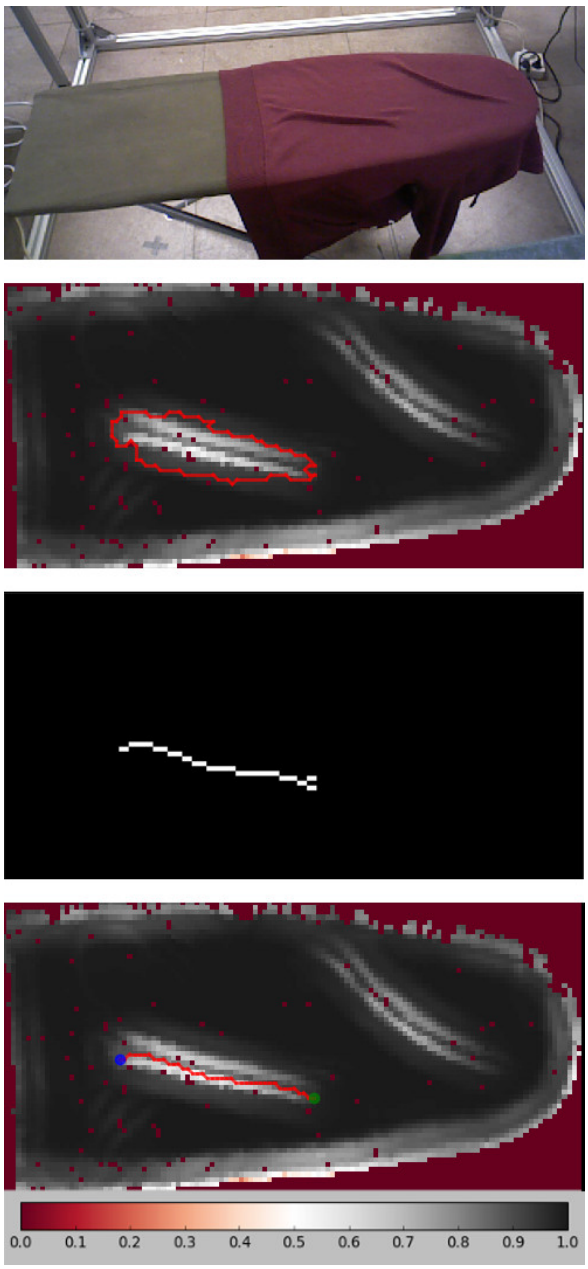

Figure 5.6: Different steps of the path extraction stage. The original setup is shown in the topmost subfigure as reference. Then, from top to bottom: the largest wrinkle is selected (shown in red) from the WiLD projected image. Then, the selected binary blob is skeletonized. Finally, the ironing path (red) is extracted from the skeletonized wrinkle, along with the starting (blue) and final (green) points (bottom).

First, the robot performs an initial trajectory with a set of waypoints $w_0 = \{w_0^{(0)}, ..., w_0^{(N)}\}$, which is used to move the iron

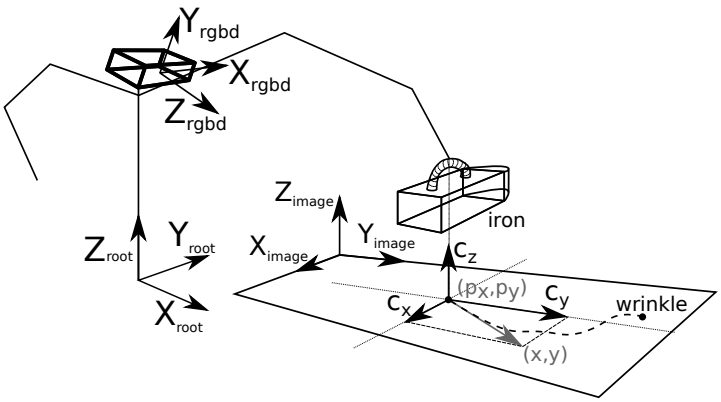

Figure 5.7: Reference systems for the ironing task, including the robot root frame, the RGB-D sensor frame and the image frame, along with the control vector $\vec{c}$.

from the initial pose to a position at a predefined height over the ironing board, in such a way that the projection of the iron over the ironing board plane corresponds to the starting point of the ironing trajectory. The final position is computed by replacing the components $w_{0x}^{(N)}$ and $w_{0y}^{(N)}$ of the predefined initial trajectory w_0 by the components $w_x^{(0)}$ and $w_y^{(0)}$ of the visually-obtained ironing trajectory w.

When the initial trajectory has been performed successfully, a second movement to approximate the iron to the ironing board and make a controlled contact is executed. The iron is lowered vertically while monitoring the force/torque sensor to detect contact. A threshold $F_d = ((F_{dx}, F_{dy}, F_{dz}), (T_{dx}, T_{dy}, T_{dz}))$ is set, where (F_{dx}, F_{dy}, F_{dz}) correspond to the Cartesian components of the force measured by the sensor, and $(T_{dx}, T_{dy}, T_{dz}))$ to the Cartesian components of the torque. Once the measurements of the sensor exceed this threshold, the downward movement is concluded, and the ironing path-following hybrid controller starts working.

The lowering movement is performed in the joint space, either in position control mode or velocity control mode. If position control is used, a small joint space increment Δq_{cmd} is computed for a given desired Cartesian increment vector $\Delta x_d = (0, 0, -x_{dz})$ using the Levenberg-Marquardt algorithm [55] to perform inverse kinematics, which is then feed to the position control system to perform the movement.

If velocity control is used instead, a velocity control loop $\dot{q}_{cmd} = J_A^{\dagger}(q) \cdot \dot{x}_d$ is implemented, where $\dot{q}_{cmd}$ is the joint velocity vector that is commanded, $J_A^{\dagger}(q)$ is a Moore-Penrose pseu-

doinverse that uses singular value decomposition (SVD) based on householders rotations with updated joint positions q at each iteration, and $\dot{x}_d = (0, 0, -\dot{x}_{dz})$ is the desired Cartesian velocity vector.

Once contact with the ironing board is detected, the ironing path-following hybrid controller starts the actual ironing operation using the ironing path waypoints $w_0 = \{w_0^{(0)}, ..., w_0^{(N)}\}$ received from the perception system and converted to the robot base reference system. The path-following hybrid control is governed by the control vector $\vec{c}^{(\tau)}$:

$$\vec{c}^{(\tau)} = \begin{cases} c_x^{(\tau)} = v_t \cdot \cos(\operatorname{atan2}((w_y^{(i)} - y^{(\tau)}), (w_x^{(i)} - x^{(\tau)}))) \\ c_y^{(\tau)} = v_t \cdot \sin(\operatorname{atan2}((w_y^{(i)} - y^{(\tau)}), (w_x^{(i)} - x^{(\tau)}))) \\ c_z^{(\tau)} = c_z^{(\tau-1)} + K_f \cdot (F_{dz} - F_z^{(\tau)}) \end{cases}, \tag{5.6}$$

Where the current time step is τ, the Cartesian components of $\vec{c}^{(\tau)}$ are $(c_x^{(\tau)}, c_y^{(\tau)}, c_z^{(\tau)})$, v_t is a hand-crafted constant that corresponds to the desired tangential component of $\vec{c}$, $x^{(\tau)}$ and $y^{(\tau)}$ are the Cartesian components of the instantaneous position, and K_f is the user-tuned proportional gain of the force control on the vertical axis. Figure 5.7 shows the control vector $\vec{c}$ in the context of the wrinkle-following trajectory.

To perform the control in this stage position or velocity mode in the joint space can be used, as in the contact trajectory. In position mode, the control vector $\vec{c}$ may be used as an increment in the cartesian space Δx_d to compute a small commanded joint increment Δq_{cmd} through inverse kinematics, as previously. In velocity mode, the differential form $\dot{q}_{cmd}$ may be computed through the premultiplication of $J_A^{\dagger}(q)$ on $\vec{c}$ used as $\dot{x}_d$.

Once the ironing operation has been completed, a vertical ascent stage similar to the contact trajectory is performed, using $\Delta x_d = (0, 0, +x_{dz})$ for joint space position control through inverse kinematics, or $\dot{x}_d = (0, 0, +\dot{x}_{dz})$ for joint space velocity control via differential inverse kinematics. These commands are performed until a safe height is reached.

5.7 CHAPTER SUMMARY

Garments present two types of wrinkles: *soft wrinkles*, large wrinkles that can be removed mechanically, and *marked creases*,

smaller wrinkles that require heat and pressure to be removed (i.e. ironing). Some existing approaches focus on the detection and removal of individual *marked creases*, requiring a controlled environment that is very difficult to obtain in real world domestic scenarios. The proposed ironing algorithm, instead, is inspired on the way people perform the ironing task, removing the necessity of detecting *marked creases*, and requiring only the detection of the larger, easier to locate *soft wrinkles*.

This chapter describes the human-inspired ironing algorithm, built from two main components: a perception system and a path-following ironing controller for the actual iron manipulation. The perception system computes the most suitable ironing trajectory in three stages.

First, a Garment Segmentation stage builds a 3D reconstruction of the working environment and separates the points that belong to the garment from the background. In the next stage, a custom descriptor, WiLD, is used to characterize the curvature of the different garment regions. Finally, a path extraction stage uses the descriptor computed in the previous stage to locate the *soft wrinkle* regions and compute an ironing path to remove them. The path is then used as input for the path-following ironing controller to perform the ironing operation, following the desired trajectory while maintaining a constant force over the garment.

6 FOLDING

This chapter describes our advances in folding with robots, the last task of the laundry pipeline (Figure 6.1).

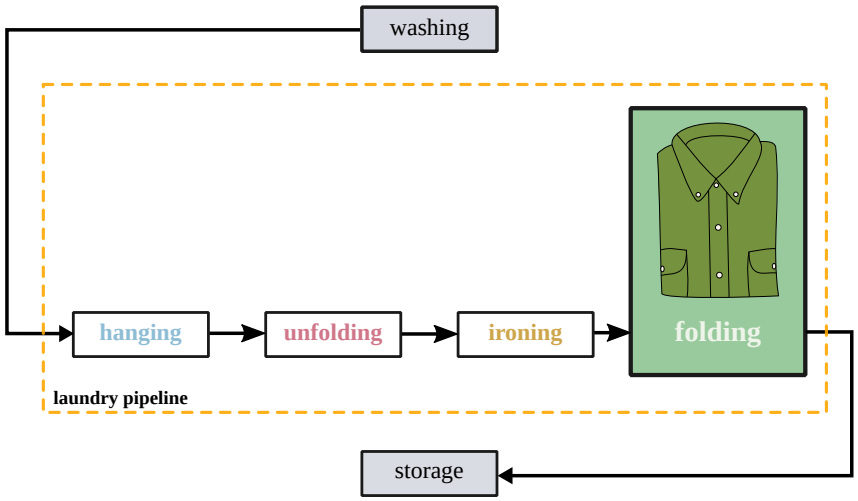

Figure 6.1: Folding is the last task of the laundry pipeline.

6.1 INTRODUCTION

After a garment has been completely unfolded and, if needed, ironed, the last step of the laundry pipeline is to fold it. Folding garments serves several purposes. On the one hand, a folded garment is easier to store, as it is kept more compact and flat, so it is therefore easier to stack it and fit it in a wardrobe or drawer. Folding garments also eases recognizing and fetching stored garments, and allows transporting or handling them in a more convenient way. In addition, folding garments can be used to prevent wrinkles from appearing during storage, by keeping the garment as flat as possible.

Folding is typically performed through a predefined sequence of folding steps, that are dependent on the garment category. By following a predefined sequence, a homogeneous result can be achieved, obtaining folded garments with a similar final shape, that can be stored more easily. Additionally, having a predefined sequence makes it faster to achieve the task, as the person

does not need to figure out a good method to fold the garment for each new clothing article that he or she encounters.

As introduced in the corresponding State of the Art section (Section 2.6), several approaches already exist that, once the garment category has been correctly identified from a flattened garment, are able to apply the folding sequence that matches that particular garment category. The sequences are composed of a concatenation of simple folding operations, that progressively transform the garment shape from the fully spread state to the folded state. As errors happening along the consecutive steps of the sequence will accumulate and affect negatively to the end result, a key aspect when folding is keeping the accuracy of the folds as high as possible.

But achieving the necessary fold precision is difficult. As opposed to rigid bodies, where the shape and dynamics of the object are kept constant during their manipulation, garments can be deformed and interact with other parts of themselves while being folded. Furthermore, the deformation depends on garment characteristics (e.g. material stiffness, weight, etc) that are not directly observable from static images. In some cases, these characteristics can be estimated through physical interaction with the garment, but it is generally a slow and complex process.

The aim of the folding method proposed in this chapter is to focus in the precision aspect, through a controller that is able to guide the folding action in real time using visual feedback as a reference.

6.2 OVERVIEW

As discussed in the previous section, a key aspect when folding a garment is to be able to perform each action of the folding sequence with enough accuracy to obtain the desired folded shape once the sequence is complete. To achieve this objective, and due to the large disparity in garment shapes and types, the work presented in this chapter focuses on a simplified version of the problem. In this version, a single folding operation is performed over a cloth strip, so that both ends match at the end of the folding action (Figure 6.2).

From all the possible folding paths to follow when performing the folding operation, the most frequently used in the literature are the triangular and circular paths. These paths are computed statically based only on the initial garment shape,

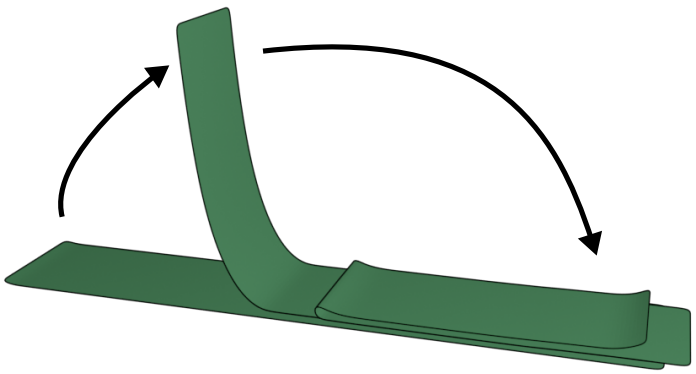

Figure 6.2: The folding task is defined as manipulating a cloth strip so that both ends match at the end of the folding action.

without taking into account the material properties of the cloth and, therefore, they suffer respectively from under and over extension issues, as depicted in Figure 6.3. If accuracy is to be achieved, the folding path has to adapt in real time to the behavior of the garment, measured through some kind of feedback.

When correcting the folding trajectory, one aspect to keep in mind is that the friction coefficient between the cloth and itself is typically larger than the friction coefficient between the cloth and the working surface. As a consequence, once the garment has contacted itself during the fold, it is very difficult to correct the folding path by pulling the grasping point, without dragging the whole garment due to the lower friction between the cloth and the working surface. Therefore, it is important to make a good first contact in the correct location to avoid the need for lifting the cloth and reattempting the folding operation.

To dynamically control the cloth during the folding operation, so that it can reach the correct contact location and proceed with the rest of the folding operation, a controller is proposed based on prior work by Vladimír Petrík et al. [67]. The proposed controller is currently being developed in collaboration with the original authors, a collaboration materialized through a research stay at the Czech Technical University (CTU) of Prague. Although this is an ongoing work, with a pending publication to report it, the author believes that the preliminary results are interesting enough to be reported in this thesis in the context of the complete laundry pipeline. In addition, in later sections, the necessary improvements and evaluation to be performed in the future will be listed.

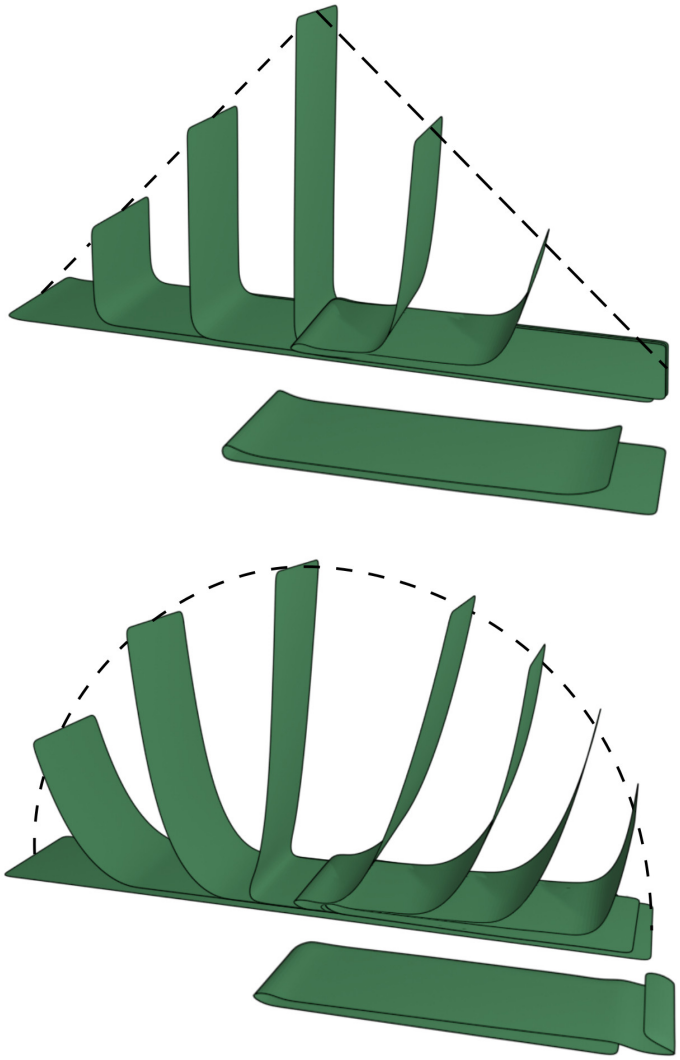

Figure 6.3: Under and over extension examples. Under extension (top) happens in triangular folding paths, as the garment flexibility does not allow the extension required to follow exactly the path, pulling the garment as it is folded. Over extension (bottom) occurs in circular paths, as the garment is not only flexible on the folding axis, but it will curve due to gravity, displacing the folding axis along the movement.

The prior work implemented a neural controller that used visual feedback to estimate the state of the cloth strip during folding, and to guide the cloth to the correct contact point while being folded. To obtain the state of the cloth, classical computer vision techniques were used to extract a 3-dimensional state vector from the cloth strip profile and the location of two key-points: the grasping point and the contact point. The neural

controller was trained in the context of a reinforcement learning problem, augmented with domain randomization.

The work proposed in this thesis, developed in collaboration with CTU Prague, improves their original garment state estimation [67] by using a Deep Neural Network to obtain a state vector that describes the cloth strip state with a higher detail (723 dimensions vs the original 3 dimensions). The main objective of this new state estimation method is to improve the accuracy of the neural controller policy by increasing the amount of information available about the garment state, extracted visually from RGB images during the cloth strip manipulation.

The following sections will describe in detail both components of the folding method: the neural controller, and the new improved state estimation model.

6.3 NEURAL FOLDING CONTROLLER

To accurately control the cloth position during the folding action, a Neural Folding Controller has been developed to compute the next robot action given a certain garment state. The neural controller is a contribution from the author's collaborator Vladimír Petrík, but it is included in this chapter as it is crucial to understand the context of the state estimation algorithm. The Neural Folding Controller can be expressed as:

$$a_\varphi = \pi_\theta(x) \tag{6.1}$$

Where a_φ is the next robot action expressed as the angle of the gripper motion relative to the horizontal axis φ, x is the garment state, and π_θ is the policy. In the original work, the garment state was a 3-dimensional vector extracted from visual feedback, where the 3 dimensions represented the vertical and horizontal positions of the grasping point, and the horizontal position of the contact point. In the improved version proposed in this thesis, the state is a 723-dimensional vector, obtained from an RGB image of the cloth strip by a deep neural network-based model.

The policy π_θ is built from a function-approximating neural network with a single hidden layer featuring a total of 20 hidden units. The neural controller is trained using reinforcement learning by finding the set of parameters θ that maximizes the obtained rewards:

$$\theta = \arg\max_\theta R(\phi) \tag{6.2}$$

Where $R(\phi)$ is the reward obtained by following the policy π_θ, and ϕ encodes the cloth strip properties.

As the cloth strip properties are hard and time-consuming to estimate from visual or tactile perceptions of the garment, one possible strategy to obtain θ is to use domain randomization to obtain the set of parameters θ as a function of some strip properties sampled randomly from a prior cloth strip property distribution:

$$\theta = \arg\max_{\theta} \mathbb{E}_{\phi \sim p(\phi)} [R(\phi)] \tag{6.3}$$

Where $\mathbb{E}$ is the expectation computed over the parameters ϕ sampled from the prior cloth strip property distribution $p(\phi)$.

To obtain the reward $R(\phi)$, a simulation of the folding operation following the folding policy π_θ is computed. The cloth strip is modeled in simulation as a set of balls of equal weight placed at a constant distance from each other, joined by rotational joints with stiffness and damping parameters that model the cloth properties. The elasticity of the garment is considered negligible, and the effect of friction is modeled indirectly, using a method that will be described later on this section.

The simulation is run for a certain amount of simulation timesteps, and finishes once the cloth has made contact with itself. If contact has not occurred after N_{max} timesteps have been reached, the simulation times out and halts. The reward is then computed as the sum of two terms; an intermediate reward computed every i-th timestep and a final reward, expressing the expected final result:

$$R(\phi) = \alpha \cdot \sum_{i=0}^{N} r_i(\phi) + r_f(\phi) \tag{6.4}$$

Where N is the total amount of timesteps that the simulation has been run, r_i and r_f are the intermediate a final rewards, respectively, and α is a scaling factor to prioritize the effect of one reward over the other.

The objective of the intermediate reward r_i is to indirectly model the garment friction with the table. Modeling the exact friction of the real garment would require to estimate the friction coefficient for different cloths and working surfaces, which would be costly and time consuming. Instead, the simulated strip is fixed on one end to the working surface, and the force required to keep it fixed, f_x, is tracked. The intermediate reward

term teaches the controller to keep this force as low as possible while performing the fold, to avoid slipping:

$$r_i(\phi) = -\frac{|f_x|}{N} \tag{6.5}$$

Where f_x is the force required to keep the simulated cloth fixed to the working surface and N is the number of simulation steps. Note that, as f_x is only used for training to model friction, there is no need to measure it in the real world when running the controller.

The final reward r_f evaluates the suitability of the contact point location reached by the cloth under the policy π_θ. Depending on whether the contact was achieved or not, the value of the final reward is:

$$r_f(\phi) = \begin{cases} -|d|, & \text{if } N < N_{max} \\ -\infty, & \text{otherwise} \end{cases} \tag{6.6}$$

Where d is represents the layer misalignment, computed as the difference in length between the two cloth branches spanning from the contact point, N is the number of timesteps elapsed at the end of the simulation, and N_{max} is the maximum amount of timesteps the simulation will run without a contact before timeout. If contact occurred, N will be less than the timeout threshold N_{max}.

Once the correct contact point has been reached using the neural controller, the cloth can be fully folded by a linear interpolation in the Cartesian space between the gripper location at the timestep when contact occurred and the location of the target cloth endpoint.

6.4 CLOTH STATE ESTIMATION

To improve the performance of Petrík's method, a new deep learning-based model, called FoldNet, is proposed in this work as an improvement over the original state estimation algorithm.

The FoldNet model allows the Neural Controller to access a more detailed visual feedback extracted from an RGB image of the cloth, improving the accuracy of the estimation and, therefore, of the controller. Instead of the 3-dimensional state vector used in Equation 6.1, FoldNet estimates a state vector with 723 dimensions, representing the 241 points that define the cloth strip shape as is being folded. This feedback is then used by the controller to estimate the next robot action a_φ.

To train the model, a custom dataset obtained through cloth simulation is crafted. The dataset includes a total of 29 120 samples with a large visual variability in terms of textures, illumination, background, poses, etc, that help the model to generalize and be applicable to real data.

This section will introduce both the data synthesis process as well as the architecture of the FoldNet model.

6.4.1 *Data Synthesis*

Deep learning models, such as the FoldNet model presented in this section, typically require a large amount of data for training, as they contain a significant number of layers, with a considerable quantity of parameters to tune in each of the layers. If the amount of training examples is not sufficiently large, it can lead to a failure to extract the patterns present in the training data, resulting in models with issues such as high bias or high variance.

To perform the state estimation, the FoldNet model utilizes a single RGB image as input, and estimates a state vector encoding the predicted shape of the cloth strip in 3D. Consequently, to train this model a series of labeled pairs of input RGB images and the corresponding 3D shape of the cloth strip for that image are required.

As previously discussed in Section 3.3, to obtain such a large amount of training examples, the most suitable option would be to resort to an existing dataset. The advantages of using such a dataset, in addition to obtaining high quality labeled data with low-to-none effort, are that it serves as a benchmark of the algorithm performance, enabling a comparison with other methods, and fostering the reproducibility of this work.

Unfortunately, as in the case of the hanging task, there is no specific dataset for this particular task publicly available. For that reason, a new custom dataset for the cloth strip folding task was generated synthetically from a cloth simulation.

Theoretically, to obtain the best performance, the model should be trained with data from a domain as close as possible to the data that will be used in the final application. In the case of the FoldNet model, that would mean obtaining the dataset from real world examples.

However, obtaining such a large dataset in a practical way is very complicated. To name a few issues that arise with this approach, there is no simple way of tracking the shape of the gar-

ment while performing the folding operation in the real world, so it would possibly need to be labeled by hand. In addition, obtaining enough variety of cloths, working surfaces and backgrounds so that the resulting model is general enough would be impractical in terms of cost and time required to record and label all the examples.

An alternative method would be to obtain synthetic data through a cloth simulation. However, training the model only on this data will cause it to fail to generalize from the simulated domain to the real world, due to the lack of variety in the visual cues than can be obtained from a standard simulation. The reason is that the main focus of a simulator is to reproduce a physical or mechanical behavior as closely as possible, and thus images obtained from it are usually not photorealistic, and feature colorful and plain objects, with a visual aspect distant from the aspect of real objects.

To solve this problem, one can exploit the high amount of parameter customization in a simulation to vastly modify the visual aspect of the simulated cloth and environment for each trial or training example. This way, the model is trained from images with very disparate aspects, learning how to distinguish which visual features are important for the task, and which ones are just stylistic. This technique of modifying the simulated data to obtain variability is called domain randomization.

Using this technique, and similarly to the hanging task, a synthetic dataset was created in an open source 3D modeling software, which provides the following benefits over recording real data:

- Since the shape of the cloth and its deformation can be obtained in real time from the simulation, it provides automatic labeling of the 3D data to be used as ground truth for each example.

- As the movement of the cloth can be defined manually by the user, it provides a great variety of folding paths that can be set up and simulated in a precise and repeatable way.

- It offers a great diversity of materials, textures, shapes and illumination choices, which contribute to obtain enough variability in the input images to avoid overfitting the model when training.

The simulation is setup as follows: a flat piece of cloth, with an elongated shape, is placed over a flat surface in the simulated environment, to simulate the cloth strip on a table. The cloth properties are selected to be in a range similar to the real cloths used in the experiments to validate the old state estimation method.

The cloth is fixed at one end, to avoid slipping, and the other end is moved programmatically following a predefined path. Support for simulating four different folding paths has been added to the 3D modeling editor: triangular, circular, trapezoidal and polygonal (Figure 6.4). The first three paths have already been defined in the corresponding section of the unfolding chapter[1] (Section 4.6). The polygonal path consists of a series of equally-spaced waypoints that the cloth endpoint crosses while performing a linear interpolation in the cartesian space between two consecutive points.

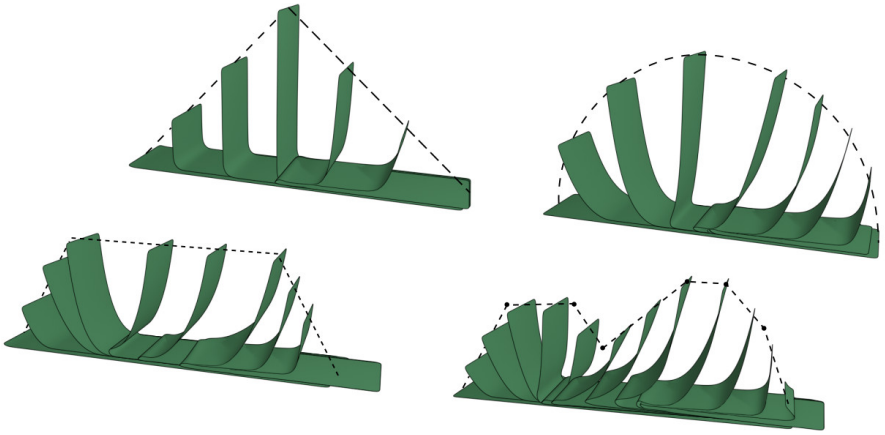

Figure 6.4: Folding paths supported by the simulator: triangular (top left), circular (top right), trapezoidal (bottom left) and polygonal (bottom right). Note that for some of the paths, due to slippage, the cloth is displaced during folding, resulting in under or over extension (as described in Section 6.2).

To increase the visual variability of the data, the aspect of the cloth and the working surface was randomized for each training example. Two possible textures are supported: a uniform color for the whole object, or a checkerboard pattern of two colors, with a random scale factor for the size of the squares. The illumination is achieved through three simulated point lights,

1 The triangular path can be considered a particular case of the trapezoidal path where $P_2 = P_3$.

that are randomly placed in the scene, with a random value of intensity for each of them.

For the background of the scene, and to increase the variability, a random image sampled from a picture dataset is used. Additionally, for some of the samples, randomly, the working surface is hidden for the final render, so the resulting image only shows the cloth strip and the random background. Hiding the working surface for some examples allows the model to recognize the garment independently of the presence of a table underneath it.

The point of view from which the render is generated is also randomly selected by pointing the virtual camera towards the cloth strip at a certain distance and modifying the location of the camera randomly so that it keeps pointing at the garment.

A total of 29 120 examples have been generated, by rendering the virtual scene while simulating the cloth behavior during 251 timesteps for each of the 4 types of folding paths. The aspect of each of the scene elements (cloth strip, working surface, and background) was randomized, along with the location of the lights and camera. All the random variables of the scene are sampled from normal distributions that control the boundaries of the values and locations that are assigned to them.

Figure 6.5 depicts a random selection of training examples sampled from the synthetic folding dataset.

For each of the training examples, the final rendered RGB image is recorded as input for the model, along with a state vector of 241 3D points that define the shape of the deformed cloth strip. This state vector represents the ground truth or label for that particular example, and is obtained by extracting the location of the vertices that form the spine or midline of the cloth strip.

6.4.2 *FoldNet model*

To perform the state estimation of the cloth strip during the folding operation, a deep convolutional neural network-based model called FoldNet was developed. As mentioned previously, the input of the FoldNet model is a single RGB image of the cloth strip being folded, and the output of the model is an estimation of the current shape of the cloth strip, encoded as a 723-dimensional state vector representing a total of 241 3D points that describe the 3D shape of the cloth.

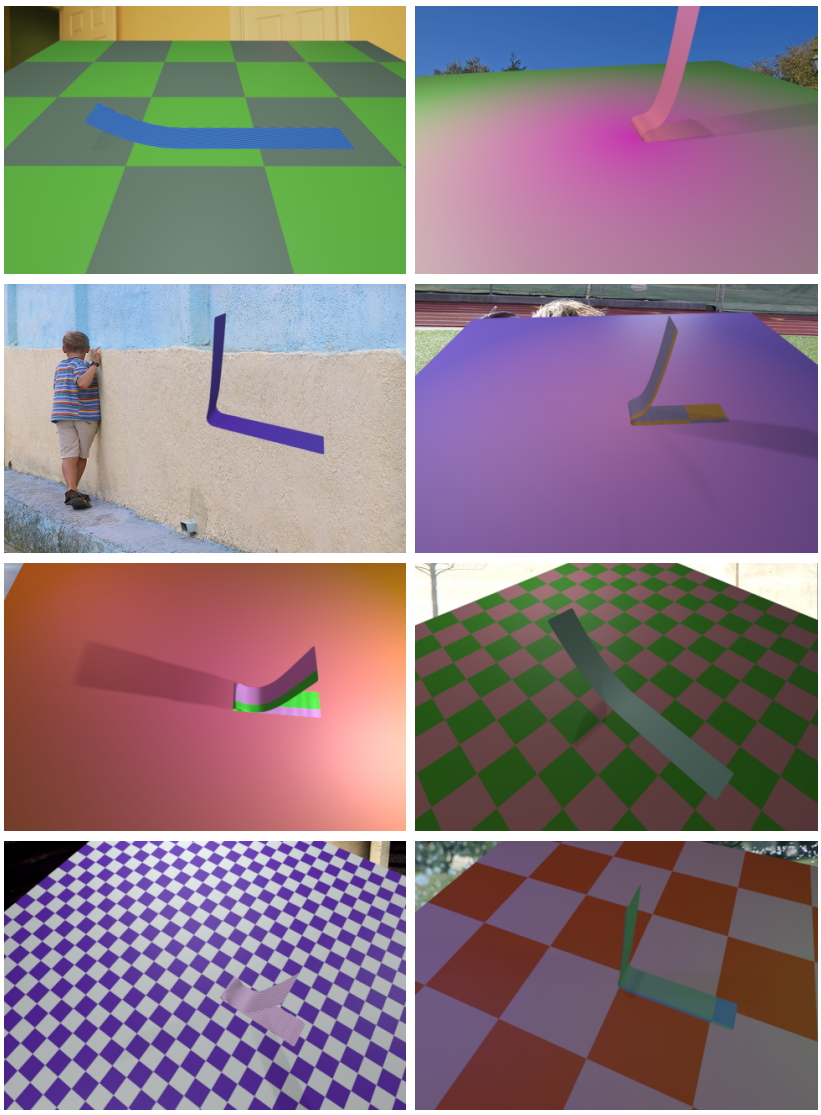

Figure 6.5: Some examples randomly sampled from the synthetic folding dataset with domain randomization enabled. Domain randomization adds visual variability to the images by changing the objects' textures, illumination, background and camera pose.

To exploit the 2-dimensional structure of the input data, the FoldNet model is composed of several stacked 2D convolutional layers, that use trainable filter banks applied through convolutions to extract features from the input data. The advantage of this type of layer is that it not only uses the values of each pixel in the image, but it also takes into account their spatial distribution, being able to extract information from images in a more efficient way than a fully-connected layer.

A 2D convolutional layer can be described, using its four main parameters, as $conv2d(F, D, S, P)$; where F is the filter size, D is the number of filters, S is the stride, and P is the padding. The stride S represents the movement of the sliding window used to apply the convolution operation for each of the filters in the filter bank. A value of $S = 1$, for instance, signifies that the window moves 1 pixel each time the filter bank is applied to the input of the layer (Figure 6.6). The padding P represents how many rows and columns of a certain value are added to the borders of the input data of a given layer. A value of 1 indicates that 1 row and column are added to every border of the input image, so that for instance an image of 5x5 pixels becomes a 7x7 image (Figure 6.6). The purpose of padding is preserving the spatial dimension of the input after a convolution so that the input and output are identical in size.

The second type of layer used in the FoldNet model is the Max Pooling layer, that can be described in terms of its window size W, and stride S, and can be written as $maxpooling(W, S)$. The Max Pooling layer extracts as output the maximum value among all the values found in a WxW window of the input data, as depicted in Figure 6.7. Max Pooling is a form of downsampling where the spatial dimension of the output is effectively reduced to:

$$N_{out} = (N_{in} - W)/S + 1 \tag{6.7}$$

With W being the window size, S being the stride, and N_{out} and N_{in} being the size of the output and input data, respectively. The purpose of Max Pooling is to reduce the computational cost by reducing the size of the output activation data for each layer, which needs to be stored to compute the activations of the next layer and the gradients required for training. Additionally, it increases the robustness of the system by removing some of the information available to the next layers, reducing the chances of overfitting the training data.

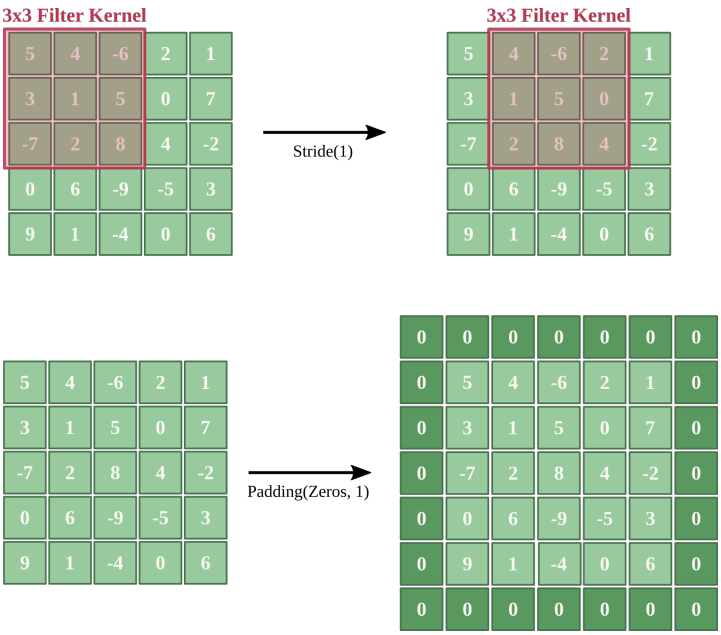

Figure 6.6: In the top figure, a 3x3 convolutional filter kernel is applied to a given matrix, by applying the kernel to a window of size 3x3, represented with a red box. With a stride value of 1, the window is shifted one position for each convolution, until all the possible vertical and horizontal combinations are covered. In the bottom figure, a padding of 1 is applied to the input matrix, using zeros as the padding value.

FoldNet combines both types of layer to perform the cloth strip shape estimation. The model has 5 blocks of stacked 2D convolutional layers of homogeneous filter size, with Max Pooling layers in between the blocks. By stacking convolutional layers with a small receptive field, the effective receptive field of the block is increased, while keeping the number of parameters to train low. For instance, a 5x5 convolution filter has the same receptive field as two 3x3 filter size stacked layers, and stacking three 3x3 filter size convolutional layers amounts to an effective receptive field of 7x7. In both cases, the number of required parameters is lower as the number of layers is higher. An additional advantage of stacking several layers instead of using a larger filter size, is that it adds more non-linearities to the discriminative function of the resulting features.

Figure 6.8 shows the architecture of the FoldNet model. The first block is composed by two 2D convolutional layers of filter size 3x3, 64 filters, stride 1 and padding 1. Before the next block, a Max Pooling layer with a 2x2 window and stride 2 is placed.

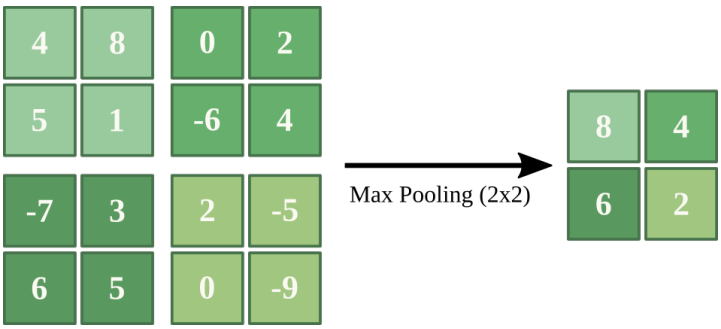

Figure 6.7: Max Pooling is a downsampling technique that substitutes a window of a given size by the maximum value in that window. In this figure, the window size is 2x2, with a stride of 2.

The next block has two 2D convolutional layers of filter size 3x3, 128 filters, stride 1 and padding 1, with a Max Pooling layer of window size 2x2 and stride 2. The third block is composed by three 2D convolutional layers of filter size 3x3, with 256 filters, stride 1 and padding 1, followed by a Max Pooling layer of 2x2 window size and stride 2. The next two blocks are identical, and feature three stacked 2D convolutional layers with filter size 3x3, a total of 512 filters each, stride 1 and padding 1. Each of this 2 blocks has its respective Max Pooling layer with stride 2 and window size 2x2. For all the previous convolutional layers, due to the filter size 3x3 and the padding of 1 px, the spatial dimension of the output is preserved after each convolutional layer, and the downsampling only occurs at the Max Pooling layers.

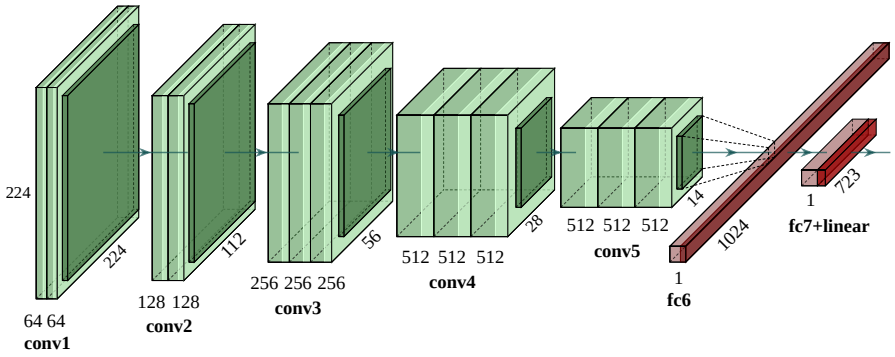

Figure 6.8: FoldNet architecture

After the last block, a fully-connected layer with a total of 1024 neurons combines the features in the last Max Pooling layer to estimate the cloth shape. For both the convolutional and the fully-connected layers, the activation function used is

ReLU. To generate the output state vector, an additional fully-connected layer of 723 neurons is added, with a linear activation function.

This architecture is inspired by the VGG-16 network [78], a popular network for object classification. As the size and amount of filters of the first layers of the FoldNet model is equal to the corresponding filters of a VGG-16 model, the weights of an already trained VGG-16 network can be used as the initial state for our model. This technique is called transfer learning, and allows our model to harness all the knowledge already stored in the weights of the VGG-16 network, and use it to construct more powerful features that are able to obtain an accurate 3D representation of the cloth strip shape.

As the first layers of the VGG-16 network contain low-level features from which more abstract concepts are built, they are general enough to be applicable to different tasks and detect general patterns. For that reason, transfer learning can be used to obtain a good initial set of weights to extract good features from the RGB image. After the initial layers have computed the features, a pair of fully connected layers learn to utilize the aforementioned features to estimate the shape of the cloth strip.

The network is then initialized with the VGG-16 weights, and then further trained with a new objective function to fine-tune the weights for the cloth state estimation task. In this case, the objective function to minimize is the Mean Squared Error (MSE) of the predicted state vector and the ground truth vector obtained from the simulation.

Transfer learning is an important technique to reduce the costs of training in terms of time required, economic costs, and energy consumption. The original VGG-16 network has 138 million parameters and was trained on a very large dataset, ImageNet [73], that includes over 14 million images from 1000 different object categories. According to the original paper, the training of VGG-16 took between 2 and 3 weeks on 4 Titan Black GPUs. Accordingly, to be able to train such a network from scratch requires an immense amount of resources that only the laboratories with the highest budgets can afford to spend, so transfer learning allows other research groups to reuse and benefit from the original VGG-16 training.

To make the input image compatible with the VGG-16 weights used as initialization, the input image has to be preprocessed before being fed to the network. The image is cropped and isotropically scaled to 224x224 px to match the VGG-16 origi-

nal input size, and then the RGB mean values of the ImageNet training set are subtracted from the image to normalize it. The image can then be fed into the network to obtain the estimated 723-dimensional output vector that can be unrolled into a state vector of 241 3D points encoding the cloth shape.

The preliminary results obtained with the FoldNet model will be reported in Section 7.4.

6.5 CHAPTER SUMMARY

Folding is the last task in the laundry pipeline, and offers several advantages such as a more compact storage of garments or the prevention of wrinkles during storage. Typically, folding is performed by applying a series of consecutive folding operations to a fully spread garment to reach the desired folded shape. One key aspect when performing such folds is to be as accurate as possible, as folding errors will accumulate through the folding sequence, affecting the end result.

This chapter presents the improvement of an existing method proposed by CTU Prague, being developed as an ongoing collaboration with the original authors that started from a research stay. Although only preliminary results have been obtained from this method, the author considers them interesting enough to be included in this thesis.

The original neural controller used for folding utilizes low-dimensional visual feedback to perform a folding operation on a cloth strip. Reinforcement learning is used in conjunction with domain randomization to train the controller from a cloth simulation, using a two-term reward that takes into account the final result of the folding action and helps modeling the friction between the garment and the working surface.

In the improved version, the low-dimensional state vector is replaced by a higher dimensional one, to achieve a higher precision. The cloth state estimation model, FoldNet, is able to estimate the cloth shape as it is being folded from an RGB image of the cloth. The model is trained using a custom synthetic dataset obtained though cloth simulation on a 3D editor, and counts with a total of 29 120 training examples.

Part III

EXPERIMENTAL RESULTS

7 EXPERIMENTS

This chapter describes the experimental evaluation of the different methods described in this thesis. To prove the validity of all the algorithms and methods proposed in this thesis, a set of experiments were designed and performed.

The aim of this chapter is to describe the purpose of each of these experiments, along with the experimental setup for each of them and the metrics used for the evaluation, reporting the different results obtained from the experiments. As the methods proposed for the different tasks on the laundry pipeline are very diverse, each of the sections of this chapter will be devoted to an individual task of the pipeline and its corresponding method.

To foster the reproducibility of the results presented in this chapter, the code for all the proposed methods that have been already published is available as a public repository[1].

7.1 HANGING

This section describes the experimental setup and results of the hanging methods introduced in Chapter 3. Two HangNet models were proposed for two different tasks: regression and classification. Each of the models is reported in a different subsection, including a detailed description of the experimental setup and obtained results.

7.1.1 *Hanging Regression Model*

7.1.1.1 *Purpose of the Experiment*

For the regression model, the main focus was to study the degradation of the performance of the model as time advances after the garment has been dropped. The proposed hypothesis to validate is that, due to the chaotic behavior of deformable objects and the interaction of the cloth with the hanger, as time advances, the uncertainty of the model's prediction would increase, resulting in a decay in the model's accuracy.

1 https://github.com/roboticslab-uc3m/textiles

For this purpose, the predictions of the regression model for the trajectory of a cloth that has been dropped over a hanger will be compared with the expected trajectory obtained through deformable object simulation.

7.1.1.2 *Experimental Setup*

The experimental evaluation of the regression model requires both a trained regression model, as well as some test data to measure the performance of the model. This subsection describes the training process to obtain the trained model, as well as how the test data was selected.

To train the regression model, a synthetic dataset [18] composed by a total of 15 000 training examples was generated following the simulation procedure described in Section 3.3, using the Open Source 3D editor Blender[2]. The virtual camera was placed in a fixed location, from which both the garment and the hanger were visible at every step of the simulation. The initial position of the garment is generated by sampling a point from a normal distribution with $\mu_{init} = (0, 0, 1.5)$ m and $\sigma_{init} = (0.01, 0.4, 0.2)$ m. The dataset is publicly available online and can be downloaded from Zenodo[3].

For each of the training examples, once the garment reached a stable pose in the simulation, a depth image of the virtual setup was recorded as input for the HangNet model. Each pixel of the depth image represents the distance from the camera to the corresponding object, in meters. The time instant in which the depth image is recorded is considered the initial time step of the training example.

The simulation is then resumed and the garment is dropped from its initial location, while tracking and recording the trajectory of the center of mass of the garment, which will be used as the expected output for training the HangNet model. The simulation is run for a total duration of 51 simulation steps, a number selected as a good compromise between computational cost and achieving a stable final state (either hanged or fallen to the floor) for the simulated garment.

Before the training examples can be fed to the deep neural network model for training, each of the depth images is cropped at a distance of 2 m to remove the empty background,

2 https://www.blender.org, last accessed: 18-05-2020
3 https://doi.org/10.5281/zenodo.3932102

and then normalized in the 0 m to 2 m range. Figure 7.1 shows a random training example gathered from the training set.

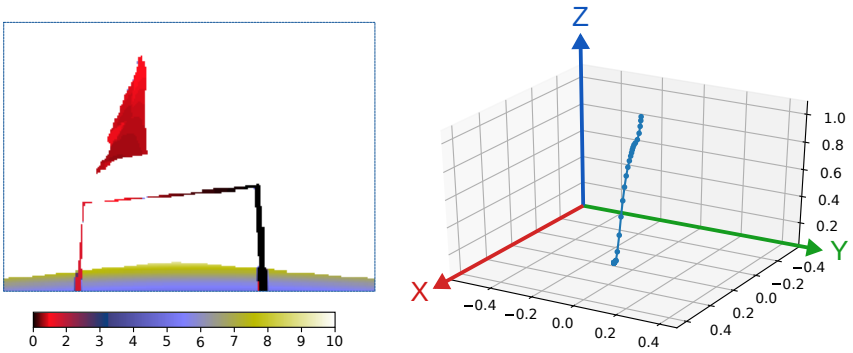

Figure 7.1: Training example input (left) and expected output (right). The input is a depth map, represented here in color to improve the depth perception. The expected output is the trajectory of the falling garment. In this particular example, it can be observed that the garment hits the hanger, but then falls to the floor.

As the training examples were obtained by randomly sampling a different initial position for each of them, and due to the chaotic nature of the garment physics, there is a variability in the amount of training examples in which the garment reaches a hanged state. More precisely, the ratio of garments being hanged vs not hanged is near 1 to 3, resulting in an imbalanced dataset.

To cope with the imbalance present in the dataset, stratification was used as the technique selected to perform the split of the dataset into the training/validation/test sets, in such a way that each of the sets has a similar proportion of examples of each of the two classes. The amount of training examples after the stratified split was the following: 20% of the examples (3000) were used for testing and, from the 80% remaining (12 000), 20% (2400) were used for validation and 80% (9600) for training.

To train the HangNet regression model, an Adam stochastic optimizer was used, with a learning rate of 10^{-4}. The custom loss introduced in section Section 3.4 (Equation 3.3) was used as training loss, with weights $\omega_x = 0.033$, $\omega_y = 0.033$ and $\omega_z = 0.33$. The model was trained for 10 epochs, with a batch size of 32.

7.1.1.3 *Experimental Evaluation*

To evaluate the overall performance of the regression model, the Mean Squared Error (MSE) of the predicted location of the center of mass of all the points in the test set is used as the metric, which can be computed as:

$$\mathrm{MSE} = \frac{1}{N} \sum_{i}^{N} \sum_{j}^{3} (Y_{ij} - \hat{Y}_{ij})^2 \tag{7.1}$$

Where i represents each of the N examples in the test set, j is each of the 3 Cartesian coordinates (x, y and z), Y_{ij} is the expected output for a given example and coordinate of the test set (i.e. the ground thruth), and $\hat{Y}_{ij}$ is the corresponding prediction value of the regression model for that example and coordinate.

To study the influence of the uncertainty on each of the individual coordinates of the prediction with respect to the time step, the L1-norm is computed individually as the error for each of the coordinates and examples, and from it, the Mean Absolute Error (MAE) and standard deviation of each coordinate is calculated for each time step.

The MAE can be defined in a compact form as:

$$\mathrm{MAE} = \frac{1}{N} \sum_{i}^{N} |Y_i^d - \hat{Y}_i^d| \tag{7.2}$$

Where i represents each of the N examples in the test set, d is each of the 3 Cartesian coordinates (x, y and z), Y_i^d is the expected output for a given example and coordinate of the test set (i.e. the ground truth), and $\hat{Y}_i^d$ is the corresponding prediction value of the regression model for that example and coordinate.

7.1.1.4 *Experimental Results*

To check the hypothesis proposed for this model, characterizing the effect of time on the uncertainty of the prediction and, as a consequence, on the accuracy of the model, the regression model was trained to predict the location of the center of mass of the garment at different time steps. Figure 7.2 depicts the results of this experiment, in terms of the Mean Squared Error (MSE) of the predicted location of the center of mass of the garment as simulation time advances.

To analyze how much each of the spatial coordinates contributes to the total uncertainty of the prediction, the MAE was

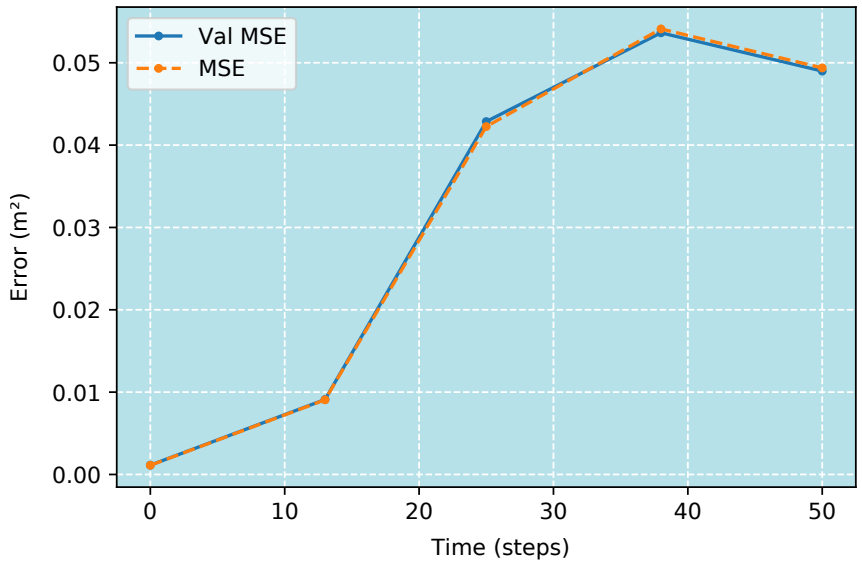

Figure 7.2: Mean Squared Error (MSE) and Validation MSE with respect to the time step.

computed individually for each of the 3D components of the prediction location (X, Y and Z). Figure 7.3 shows the value of the MAE for each of the X, Y and Z components, as well as the variation in error for each of them. As it can be observed in the graph, the variation in the location error ranges form a few millimeters right after the simulation has started to several centimeters as the simulation advances, becoming more dramatic in the case of the Z coordinate, which is an expected result, due to the interaction of the garment with the hanger, and the fact that the garment can remain randomly hanged or not hanged, affecting mainly to the expected location in the Z axis.

7.1.2 *Hanging Classification Model*

7.1.2.1 *Purpose of the Experiment*

For the classification model, the main interest was to determine its accuracy when predicting the outcome of dropping a given garment with certain initial conditions over a hanger. As the prediction has two possible outcomes, the garment remaining hanged or otherwise falling to the floor, each outcome can be considered one class of a binary classification problem.

With the purpose of determining such accuracy, and considering the difficulty of the task due to the uncertainty associated with the chaotic behavior of garments, the performance of the

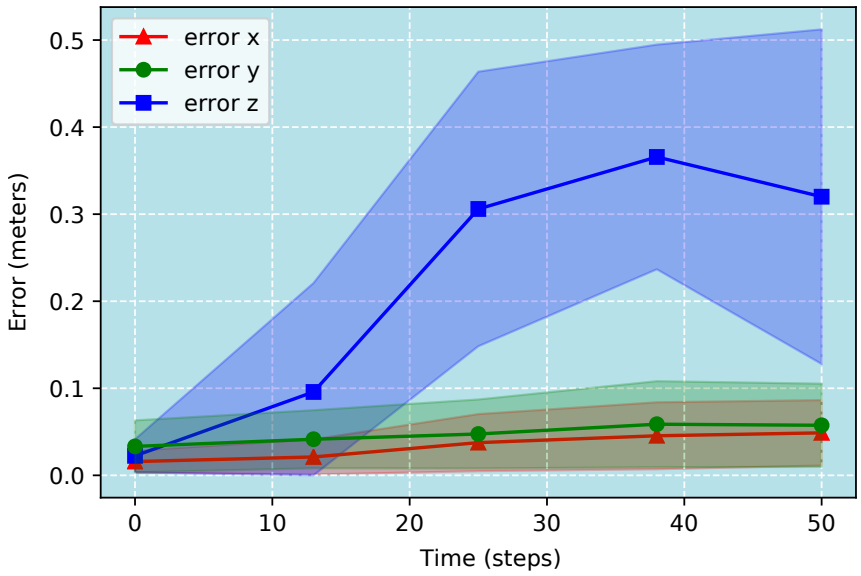

Figure 7.3: Mean Absolute Error (MAE) of each of the 3 coordinates (X, Y, Z) with respect to the time step, along with the standard deviation for each coordinate.

classification model was compared against the performance of a human expert, used as a baseline.

7.1.2.2 *Experimental Setup*

As with the regression model, the experimental evaluation of the classification model requires a trained classification model, as well as some test data to evaluate the performance of the model. Additionally, a baseline performance has to be obtained from predictions from a human expert. This subsection describes the training process to obtain the trained model, as well as the test data selection process and acquisition of the human baseline.

To train the HangNet classification model, the same dataset as the one described in Section 7.1.1 was used, and the same stratification techniques were use to deal with the imbalance of the dataset when performing the training/validation/test splits.

The trajectory previously recorded as expected output for the regression experiment is processed to obtain a single binary label, since the output of the classification model is a single binary value representing the prediction of the network: 0 when the garment will stay hanged after dropping it, or 1 if the garment will fall once dropped. To perform the conversion, Equa-

tion 3.4 is used, with a threshold value empirically determined to be $T_{floor} = 0.81$ m.

To train the classification model, an Adam stochastic optimizer was used, with a learning rate of 10^{-4}. In addition, L2 regularization was added to the model, with a regularization strength of 0.01, to improve the generalization capabilities of the deep neural network.

An additional subset of 200 examples were randomly selected from the test set, and presented to a human expert to obtain a performance baseline. The human expert, using the same input data as the classification model -a depth image of the initial conditions- had to predict which of the two possible outcomes will happen once the garment is dropped.

The human visual perceptual range is more limited than a computer to distinguish slight changes in shades of grey and, therefore, depth values in the depth image. For that reason the input image, originally a 8-bit greyscale depth image, was transformed using a GIST stern[4] colormap for display. Training examples depicted in Figure 7.1 have been rendered using the GIST stern colormap and can serve as a reference for the interested reader.

7.1.2.3 *Experimental Evaluation*

The evaluation of the classification model is performed in two different steps. First, the accuracy of the model is evaluated using the test set alone, through a set of standard metrics that include precision, recall and F1 score. Additionally, to evaluate the relevance of the results achieved, the same set of metrics is computed from a subset of 200 elements extracted from the test set. The results from this subset are compared with a baseline obtained from a human expert, by computing the same metrics from predictions performed using as input the same data as the HangNet model, a depth image of the initial conditions.

One of the metrics used for the model evaluation is the precision metric, which measures the capability of a model to distinguish true positives from false positives, and is computed as follows:

$$\text{Precision} = \frac{\text{True positives}}{\text{True positives} + \text{False positives}} \tag{7.3}$$

4 https://www.ncl.ucar.edu/Document/Graphics/ColorTables/MPL_gist_stern.shtml, last accessed: 18-05-2020

Precision is an important metric for tasks in which the penalty or cost of misclassifying a negative sample as a positive one is high. An example of such task is email spam detection, where classifying a legitimate email as spam causes some loss of information to the user. In the case of the hanging task, precision is an important metric too, as classifying a garment that will fall to the floor as if it will remain hanged might cause the robot to drop the garment in an incorrect position, expecting the garment to remain hanged.

Similarly, recall is a metric that measures the ability of a model to distinguish and avoid false negatives, and can be computed as:

$$\text{Recall} = \frac{\text{True positives}}{\text{True positives} + \text{False Negatives}} \tag{7.4}$$

Recall is an important metric for tasks where there is a high penalty or cost associated to predicting a false negative. Some examples of such tasks include fraud detection or sick patient detection, tasks in which there is a high penalty if a true positive case is not detected correctly. For the hanging task, the recall metric is less important than precision, as if a dropped garment is predicted to fall to the floor, though it would have remained hanged, it only delays the detection of a good drop location for hanging, as the robot will keep looking for a suitable drop location without releasing the garment.

When both metrics are important, the F_1 score can be used to obtain an overall performance by combining both metrics. The F_1 score is the harmonic mean of the precision and recall metrics, and can be computed as:

$$F_1 \text{ score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{7.5}$$

As the F_1 score represents a balance between precision and recall, it is a good metric to be used in the case of unbalanced datasets, like the synthetic dataset used in the experiments of the hanging task.

7.1.2.4 *Experimental Results*

Once the model was trained, precision, recall and F_1 score metrics were computed using a test set composed of 3000 training examples, obtaining the results reported in Table 7.1.

The performance of the classification model was compared with a baseline obtained by computing the same metrics as before -precision, recall and F_1 score- from the performance of a

Table 7.1: Classification model, 3000 elements

Class	Precision	Recall	F1-score	# elements
Hanged	0.51	0.32	0.39	819
Floor	0.78	0.88	0.83	2181

human expert in the subset of 200 randomly selected training examples. Table 7.2 shows the results of the analysis. Although the model has a lower recall for the hanged class than the human expert, it improves the performance of the human in all remaining metrics being considered. As mentioned in the Experimental Evaluation section, for this application the recall is not a critical factor, as a false negative (i.e. predicting that a garment will fall when it would remain hanged) only delays finding a good location for dropping the garment over the hanger.

Table 7.2: Classification model vs Human baseline, 200 elements

Class	Precision	Recall	F1-score	# elements
Classification Model				
Hanged	0.60	0.39	0.47	54
Floor	0.80	0.90	0.85	146
Human Baseline				
Hanged	0.38	0.56	0.45	54
Floor	0.80	0.66	0.72	146

Additionally, and for a more detailed comparison of the performance of the model and the human expert in terms of false positives and false negatives, confusion matrices for both results were computed. As depicted in Figure 7.4, the confusion matrices show that the classification model outperforms the human expert when predicting garments that fell to the floor, while having a similar performance when predicting garments that remained hanged.

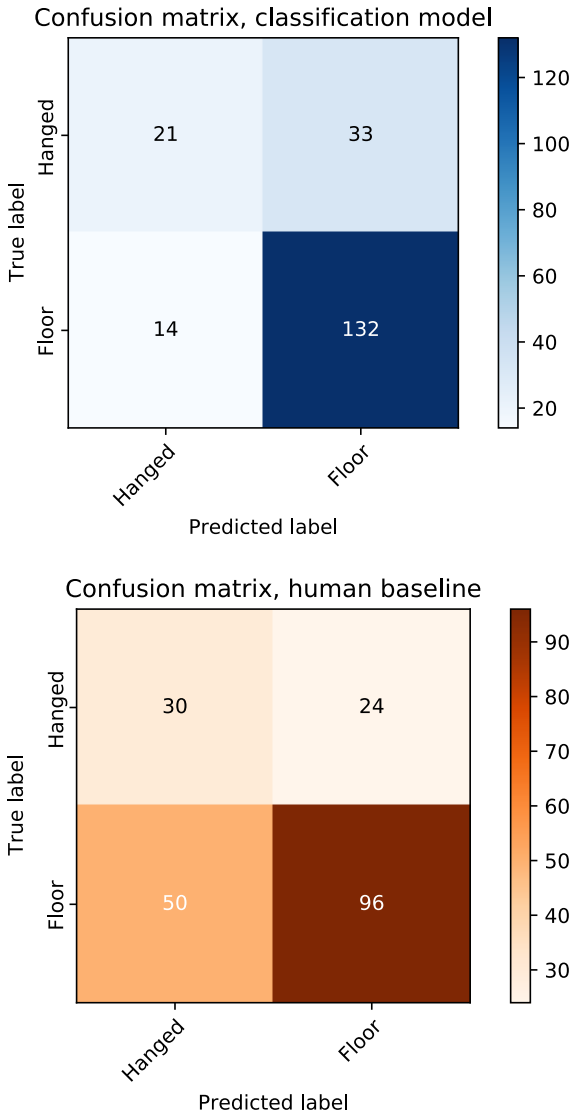

Figure 7.4: Confusion matrices for the classification model and human baseline.

7.2 UNFOLDING

This section describes the experimental setup and results of the unfolding algorithm presented in Chapter 4. Two different versions of the algorithm were developed, a initial version using the color-based Garment Segmentation Stage, as well as an improved version using the 3D reconstruction-based Garment Segmentation Stage. Both methods were described in detail in the corresponding section (Section 4.3). Each of them has been experimentally tested and validated using several different robotic platforms through the experimental procedures explained in this section.

7.2.1 *Purpose of the Experiment*

The aim of the unfolding experiments is to measure the ability of the presented algorithm to unfold clothing items belonging to different garment categories. For this purpose, a set of garments will be assembled, from which they will be extracted and placed over a flat surface with a series of hand-made folds. The robot will then have to undo the folds autonomously by applying the proposed unfolding algorithm. The performance of the method will be evaluated by checking if the outcome of the unfolding action matches the expected outcome (i.e. the garment is correctly unfolded). Additionally, the experiments will be performed with several robotic platforms to check the validity of the proposed approach for different types of robots.

As the unfolding algorithm has three consecutive stages, the performance of each of them will be tested independently, to study which of the stages contributes more prominently to the success or failure of each unfolding action.

7.2.2 *Experimental Setup*

Two different robotic platforms were used to evaluate the unfolding algorithm: a humanoid robot and an industrial manipulator, both depicted in Figure 7.5. The main motivation behind the use of two different platforms is to take advantage of their complementary features. The humanoid robot can offer validation of the approach in a real domestic environment, but as a highly experimental platform, its reliability, repeatability and setup times are inferior compared to an industrial solution. The industrial manipulator, on the other hand, can only be used in a

more controlled, industrial environment, but provides a higher experimental throughput as it can be set up and operated at a faster rate.

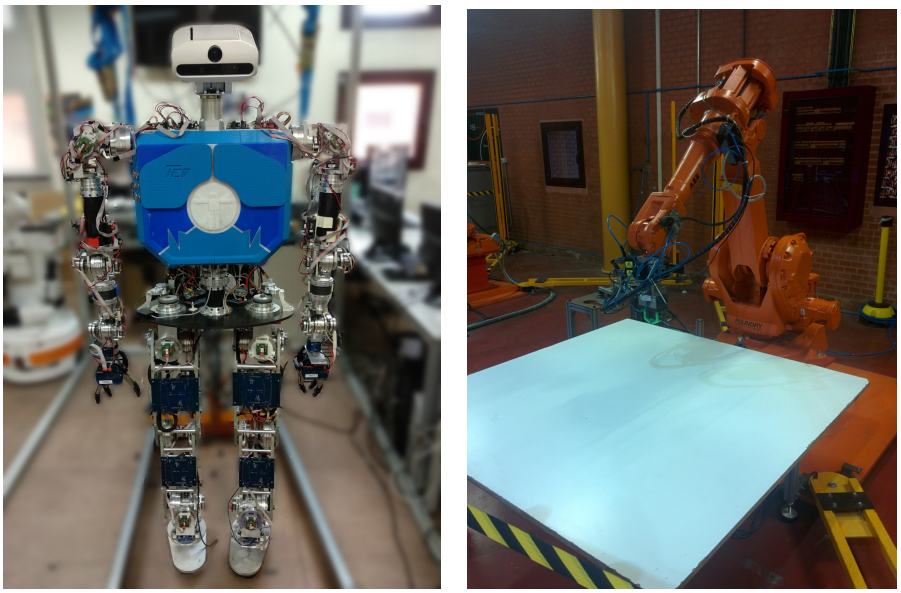

Figure 7.5: Robotic platforms used in the unfolding experiments: TEO, the humanoid robot (left), and an ABB IRB 2400 industrial manipulator robot (right).

The humanoid robot is called TEO [57] and it is an ongoing research project being developed in the Robotics Lab of the Carlos III University of Madrid. It has a height and weight similar to those of a adult male (around 1.8 m and 80 kg), a total of 28 DOF, and is equipped with an ASUS Xtion PRO LIVE RGB-D sensor for visual perception, and a custom 3D printed gripper (previously described in Section 4.6). The humanoid robot offers experimental proof of the validity of our approach in a hypothetical real domestic environment, as it is a type of robot very likely to be found in people's houses in the future.

For the experiments with this robot, in addition to the integrated RBG-D, an extra RGB-D sensor of the same model and characteristics is placed in the upper part of the working environment, to provide a birds-eye view of the garments the robot is working with. The setup is completed with a flat white surface on top of which the garments are placed. The unfolding setup for the humanoid robot is shown in Figure 7.6.

Once validated on the humanoid robot, and to increase the throughput of the experiments for a more thorough evaluation of the algorithm, an industrial manipulator was used to perform a series of trials. The industrial manipulator selected to

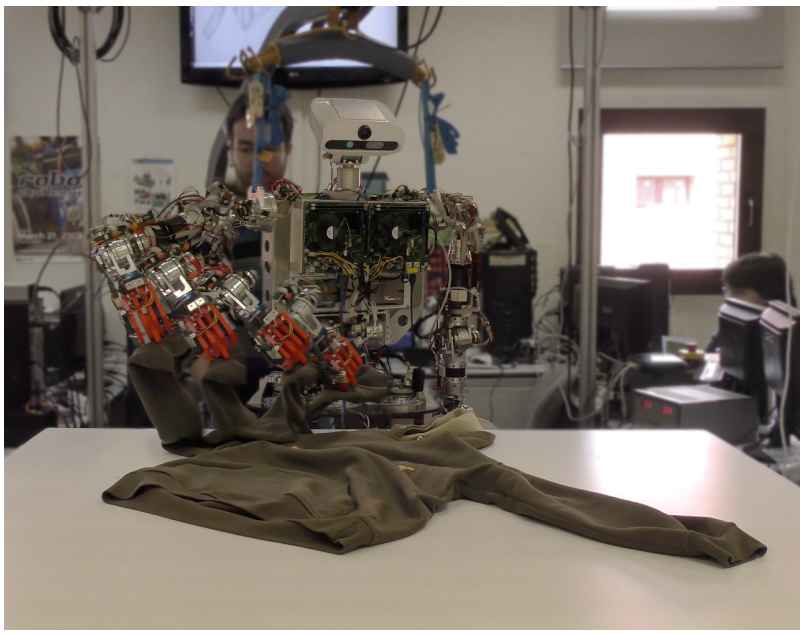

Figure 7.6: Unfolding setup for the humanoid robot TEO. Garments are placed and folded on a white surface in front of the robot to perform the unfolding action.

execute the exhaustive evaluation was an ABB IRB 2400 industrial manipulator robot, due to its relatively large reachability.

A custom 3D printed gripper, with EVA foam added on the inner part of the gripper claws to reduce slippage, was fixed to the robot to grab the garments. Additionally, this custom tool has an ASUS Xtion PRO LIVE RGB-D sensor attached as a way to obtain a 3D reconstruction of the working environment. This tool was described in detail in Section 4.6. The industrial manipulator controller is running a server modified from the open_abb[5] controller that receives the target points and orientations from our main PC and executes them accordingly. A flat white surface is used as the working surface on which the garments are placed, as depicted in Figure 7.7.

For the evaluation of the approach that uses the color-based Garment Segmentation Stage, a pair of depth and RGB images of size 640x480 px were recorded per garment sample, all captured from a bird's eye perspective using the top ASUS Xtion sensor as input for the algorithm. For the approach using a 3D reconstruction-based Garment Segmentation Stage, several RGB-D images were obtained from different perspectives and combined to obtain a 3D reconstruction of the working environment. In the trials belonging to the evaluation with the hu-

5 https://github.com/robotics/open_abb, last accessed: 18-05-2020

Figure 7.7: Unfolding setup for the industrial manipulator robot. The robot is equipped with a custom tool composed of a 3D printed gripper and an RGB-D camera. Garments are placed and folded on a white surface located in front of the robot.

manoid robot the different perspective were achieved by using the 2 DOF of the robot's neck to gather RGB-D data from different points of view.

In the trials where the industrial manipulator was used to evaluate the algorithm, the industrial manipulator was used to obtain different views of the garment by rotating the ASUS Xtion sensor over the table and around the garment. For each garment sample, the manipulator performs two circular trajectories: a higher one with an average camera inclination of 60° with respect to the table normal, and a lower one, with an average inclination of 30°.

Depth data obtained from different viewpoints along the circular trajectories is integrated using the Kinect Fusion algorithm to obtain a single 3D reconstruction of the garment and working environment. By integrating different views, the total resolution achieved increases, providing more resolution than a single depth frame. For these experiments, PCL's [75] implementation of Kinect Fusion was used[6].

6 https://pcl.readthedocs.io/projects/tutorials/en/latest/using_kinfu_large_scale.html, last accessed: 24-07-2020

Once the garment data, either RGB-D or a 3D reconstruction depending on the method, has been recorded, it is then used as input for the unfolding algorithm. All the required stages of the algorithm will be then executed: computing the segmentation mask, clustering of the different overlapped regions, and the most suitable pick and place points. The obtained points are then converted from the sensor frame of reference to the robot's root frame of reference, so that the robot can perform the corresponding unfolding pick and place sequence.

For each individual trial of each experiment, a single garment is placed over the flat white surface. Each garment is initially laid fully extended over the working surface manually, and then folded to obtain either one or two folds, depending on the experiment. To have enough variation in color, shape, and cloth thickness, a garment is selected from the following garment categories: skirt, jacket, pants, polo, robe, hoodie.

7.2.3 *Experimental Evaluation*

To measure the performance of the unfolding algorithm, the evaluation of the corresponding experiments will be performed qualitatively, by classifying the outcome of each trial as a success or failure based on whether the robot was able to unfold the garment or not.

As the unfolding algorithm is composed by three sequential stages, the performance of previous stages might affect the outcome of later stages. For instance, if the Segmentation Stage passes an incorrect segmentation mask to the next stage, the results of the Clustering Stage might be affected. For that reason, and to minimize the effect of previous stages in the performance of later stages, the performance of each of the stages is classified as a success or failure independently of the previous stage results. This means that, for instance, even if some part of the garment is missing on the masked image, it is still passed to later stages and evaluated, as it might nevertheless lead to a successful clustering and unfolding.

7.2.4 *Experimental Results*

Two experiments were performed with the industrial manipulator to compare the performance of the color-based Garment Segmentation Stage approach versus the 3D reconstruction-based Garment Segmentation Stage approach. For both experiments,

a total of 6 garment categories were evaluated: Skirt, Jacket, Pants, Polo, Robe and Hoodie. For each of them, 5 trials were performed, 3 of them with different single folds, and 2 of them with different double folds. As mentioned in the previous section, these initial folds were performed by hand to ensure consistent initial conditions for all the trials.

The obtained results are reported in Table 7.3. A stage by stage analysis is presented, demonstrating an increase of performance of the 3D reconstruction-based Garment Segmentation Stage version of 20 % with respect to the color-based Garment Segmentation Stage approach, yielding an overall success rate of 60 %. As some failures in previous stages of the algorithm (e.g. Garment Segmentation Stage) could still lead to a successful clustering and unfolding, the output of each stage of each trial was independently classified as a success or failure to obtain a more accurate description of the performance of the algorithm.

Table 7.3: Results analysis of the Unfolding Algorithm (5 trials, 6 categories), per stage and garment category, expressed as percentage (%)

Stage / Category	Skirt	Jacket	Pants	Polo	Robe	Hoodie	All
Color-based approach							
Segmentation	100	100	80	100	100	20	**83.3**
Clustering	80	60	60	80	60	0	**56.7**
Pick & Place Points	60	40	20	80	40	0	**40**
3D reconstruction-based approach							
Segmentation	100	100	100	60	60	100	**86.7**
Clustering	80	80	80	80	80	80	**80**
Pick & Place Points	60	60	60	40	80	60	**60**

Some examples of the final output and computed unfolding directions by the 3D reconstruction-based unfolding algorithm for 2 of the trials of each of the 6 categories are shown in Figure 7.8.

To further validate the results of the 3D reconstruction-based approach, an additional experiment with 20 trials per garment category for 5 garment categories was performed, for a total of 100 trials. For each category, 10 of the 20 trials were performed with a single fold, and the remaining ones with 2 folds. For each of the unfolding operations, the robot required an average

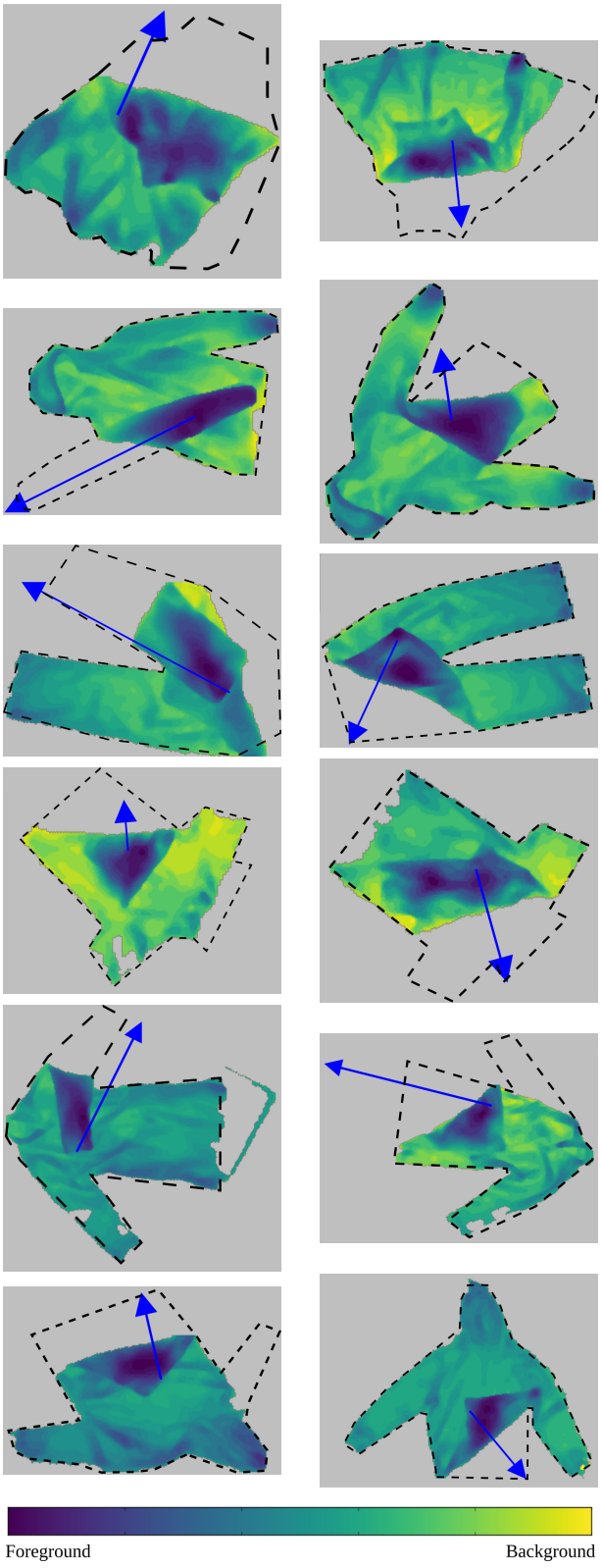

Figure 7.8: Computed unfolding directions overlaid on top of the corresponding garment height map. Each row includes the output corresponding to 2 of the 5 algorithm runs for each of the 6 garment categories considered: Skirt, Jacket, Pants, Polo, Robe, and Hoodie. The unfolded garment outline has been added by hand to each figure for illustrative purposes.

of 63 ± 1 s for the 3D garment reconstruction and an average of 28.5 ± 6.8 s to perform the actual pick and place operation. Table 7.4 provides a stage by stage analysis regarding the extended experiments with the additional 20 trials for each of the 5 garment categories.

Table 7.4: Results analysis of the Unfolding Algorithm (20 trials, 5 categories), per stage and garment category, expressed in percentage (%)

Stage / Category	Skirt	Jacket	Pants	Polo	Robe	All
3D reconstruction-based approach (extended trials)						
Segmentation	100	100	95	100	100	**99**
Clustering	85	60	63.2	85	75	**73.6**
Pick & Place Points	70.6	75	33.3	94.1	78.6	**70.3**

Table 7.3 and Table 7.4 show that both experiments present a consistent overall performance, with a slight decrease in performance in the clustering stage and increasing in the remaining stages. In the case of the color-based approach, the method presents some sensitivity to the color of the garment as expected, with a decrease in performance occurring with darker garments such as the hoodie.

On the other hand, with the 3D reconstruction-based method some failure modes can be observed, for instance, in the case of very thin garments such as the robe or the polo, where the RANSAC algorithm misinterprets part of the garment as part of the background table. In contrast, previous failure modes in the presence of darker garments no longer occur, as the 3D reconstruction-based approach is color-independent.

In the case of the clustering stage, the most common failure mode is caused by single overlapped regions being divided into several clusters, causing unstable bumpiness values. Split clusters can also cause the selected cluster edge to not correspond to the edge of the whole overlapped region, but to a smaller region within it.

If this inner edge is used to compute a pick point, it is impossible for the robot to fully unfold the region, as the point would lay below the unfolded region. Therefore a folded region of size equal to the distance between the pick point and the region edge would be left after the manipulation operation. To address these issues and improve the overall performance of the algorithm, some discussion is provided in Chapter 8.

7.3 IRONING

This section describes the experimental setup and results of the ironing algorithm presented in Chapter 5. For the evaluation of the ironing algorithm, two sets of experiments were designed, one for the evaluation of the perception algorithm, and other for the evaluation of the complete ironing algorithm.

7.3.1 *Purpose of the Experiment*

The objective of the ironing experiments is to validate the performance of the proposed ironing algorithm as a whole, as well as to compare the effectiveness of the wrinkle detection algorithm against other general purpose 3D descriptors.

More precisely, in the case of the complete ironing pipeline, the aim of the experiment is to determine the effectiveness of the approach in a real world setting. This is achieved by manually setting a clothing item on an ironing board and letting the robot iron it autonomously, tracking the amount of wrinkles removed, as well as how long it takes to iron it in terms of iterations and time.

To validate the performance of the wrinkle detection algorithm individually, it will be applied to a 3D reconstruction of a garment in the working setup, and the results will be compared to the performance of a general purpose 3D descriptor, RSD, to demonstrate the effectiveness of the perception algorithm finding wrinkles.

7.3.2 *Experimental Setup*

For the ironing algorithm, the experimental setup is intended to emulate a typical domestic environment. For this purpose, a real ironing board was set up in our laboratory and fit with real garments, and a standard unmodified electric iron was used to iron the garments (Figure 7.9).

The robotic platform selected for the experiments was our humanoid robot TEO [57], previously introduced in the unfolding experiments section. As the main focus of this set of experiments was validating the ironing algorithm, and not grasping or dexterous manipulation, one of the hands of the robot was removed, and temporarily replaced by the iron, fixed to the robot's wrist using custom 3D printed parts.

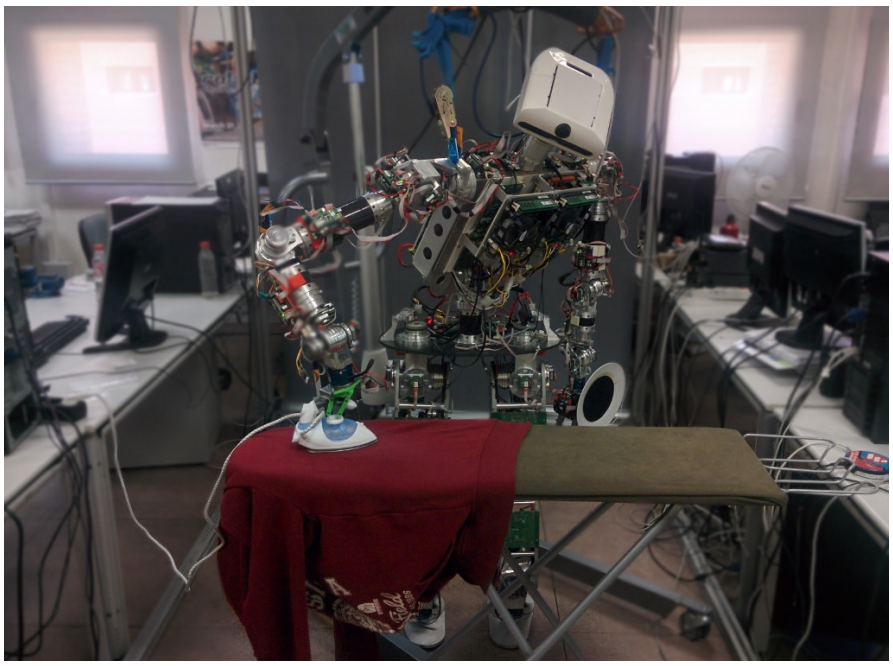

Figure 7.9: Ironing setup for the ironing task. The humanoid robot TEO is fitted with a functional iron, and the garment to be ironed is manually placed in an ironing board in front of the robot.

To obtain the force/torque feedback required for the ironing path-following hybrid controller described in Section 5.6, the humanoid robot is equipped with a JR3 6D force/torque sensor on each wrist. For visual feedback of the environment and the garment to be ironed, an ASUS Xtion PRO LIVE RGB-D sensor is present on the robot's head.

Similarly to the unfolding experiments reported in Section 7.2, the RGB-D sensor obtains a 3D reconstruction of the working scene using PCL's implementation of the Kinect Fusion algorithm. Both the pan and tilt degrees of freedom of the robot head are required to move the sensor and achieve sufficient disparity to generate a correct 3D reconstruction of the working environment.

Once the scene is scanned, the 3D reconstruction obtained is then fed as input for the ironing algorithm, that uses the garment data to generate the ironing path waypoints that the iron has to follow to perform the ironing operation. The waypoints are computed by the algorithm in the 3D reconstruction frame of reference, so they must be converted back to the robot's root frame of reference before any manipulation operation can be performed.

For the evaluation of the ironing algorithm, two sets of experiments are proposed to evaluate, respectively, the perception algorithm (wrinkle detection) and the algorithm as a whole (including the actual ironing operation). The values for the parameters of the ironing algorithm used in both experiments were the following: 0.02 for the RANSAC threshold, 0.02 for the normal estimation radius, 0.03 for the WiLD neighborhood radius, 0.95 for the WiLD upper threshold, 0.4 for the WiLD lower threshold, 11 for the erosion structuring element size.

The first set of experiments was designed to evaluate the accuracy of the perception algorithm to detect wrinkled regions. For this purpose, a garment was set up on the ironing board by hand, and the scene was then scanned and reconstructed by the humanoid robot. The garment was segmented from the background using the algorithm described in Section 5.3, and then the wrinkled regions were extracted using the algorithm from Section 5.4.

The second set of experiments evaluates the performance of the whole ironing pipeline by completely ironing a garment with the humanoid robot. For each of the trials, the garment to be ironed is placed manually and mostly flat over the ironing board. Then, two *soft wrinkles* are created randomly by hand over the garment, ensuring similar initial conditions for all the trials, both in terms of garment state and pose, as well as presenting the same number of initial *soft wrinkles* along all the trials.

7.3.3 *Experimental Evaluation*

For the first set of experiments, where only the perception component is being evaluated, only the wrinkle extraction is required to compare its performance to other methods. In this case, the performance of the WiLD descriptor was compared against the performance of the Radius-based Surface Descriptor (RSD), using the Jaccard similarity index (JSI) as the metric:

$$JSI(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{7.6}$$

Where A and B are the sets of detections to be compared. To compute the JSI metric, the results from the WiLD and RSD evaluations were compared with hand-labeled ground truth of the 3D reconstructions.

On the other hand, for the second set of experiments, that evaluates the performance of the complete ironing algorithm, a *wrinkleness* metric, representing how wrinkled the garment surface is, is computed from the 3D reconstruction of the garment after each ironing operation. The *wrinkleness* metric can be obtained for a given wrinkle-extracting function as follows:

$$\text{wrinkleness} = \frac{\sum_{\vec{x} \in G} wrinkle(\vec{x})}{|G|} \tag{7.7}$$

Where G is the set of points belonging to the garment and $|G|$ represents the cardinality of that set. This *wrinkleness* measures the progress towards a completely ironed garment, so that the ironing algorithm can be executed iteratively until the *wrinkleness* value obtained is sufficiently close to 0 to be negligible. The number of iterations required to reach that state, as well as the value of the *wrinkleness* metric for each iteration are recorded for each trial, along with the time elapsed in each iteration, to offer additional insight about the performance of the method.

7.3.4 *Experimental Results*

The first set of experiments, to evaluate the ability of the perception algorithm to detect wrinkles, were composed of 10 trials: 5 with garments with 1 *soft wrinkle* and 5 with garments presenting 2 *soft wrinkles*. Results of this set of experiments are reported in Table 7.5, along with the time required for computing both descriptors, using a machine with an Intel(R) Core(TM) i7-4790 CPU @ 3.60GHz processor and NVidia GeForce GTX 960 graphics card with PCL-1.7 over an Ubuntu GNU/Linux distribution.

On average, WiLD presents a 25% better JSI compared to the RSD, while performing over 40% faster. In terms of the precision measured with the JSI metric, the hypothesis is that WiLD reaches a higher value as it has been developed specifically for wrinkles and does not attempt to be a general descriptor as is the case of RSD.

Regarding computation times, the complexity of both algorithms depends non-linearly on the size of the input to process, which is in the order of millions of points. For that reason the measured times for both methods lie in a 30 s to 60 s interval, depending on the method. Still, WiLD enables much faster processing through a simple and multi-threaded implementation,

Table 7.5: Results of the Ironing Perception Algorithm

	RSD		WiLD	
Experiment	JSI (%)	Time (s)	JSI (%)	Time (s)
#1	36.67	52.12	68.41	36.07
#2	32.51	50.23	60.98	34.51
#3	44.90	52.46	67.39	36.69
#4	30.75	52.37	59.65	36.62
#5	41.12	53.24	61.94	38.06
#6	42.27	50.30	68.64	34.88
#7	48.84	49.72	64.02	34.88
#8	38.30	50.42	65.56	35.39
#9	31.07	49.68	67.03	34.23
#10	34.60	51.60	60.64	35.39
Mean	38.10	51.22	64.43	35.67

which could be further improved by using GPU for the computations instead of CPU multithreading.

To evaluate the performance of the full ironing pipeline, including the actual manipulation of the iron to perform the computed ironing paths, a second set of experiments was performed. A total of 5 experiments were performed where the robot had to autonomously and iteratively remove all wrinkles present in a garment setup in the ironing board by hand. For the manipulation control, the values of the F_d parameter used are ((0,0,-60), (0,25,0)) sensor Internal Units (IU).

For each iteration performed, the values of *wrinkleness* were recorded along with the average times for each iteration, and are reported in Table 7.6. Note that the average times shown in Table 7.6 only measure the elapsed time per ironing manipulation operation. While intermediate *wrinkleness* values vary for each trial, all of them converge to zero wrinkles after only 2 ironing iterations, which can be considered a very satisfactory result.

The values of *wrinkleness* tracked for each iteration and trial are can be also seen as a graph in Figure 7.10.

Finally, Figure 7.11 shows a sequence of frames which depicts one of the ironing operations that was performed during these experiments.

Table 7.6: Results of the Ironing Algorithm

Experiment	#1	#2	#3	#4	#5
Number of iterations	2	2	2	2	2
Avg. time (s) / iteration	71.827	67.803	72.561	75.684	63.684

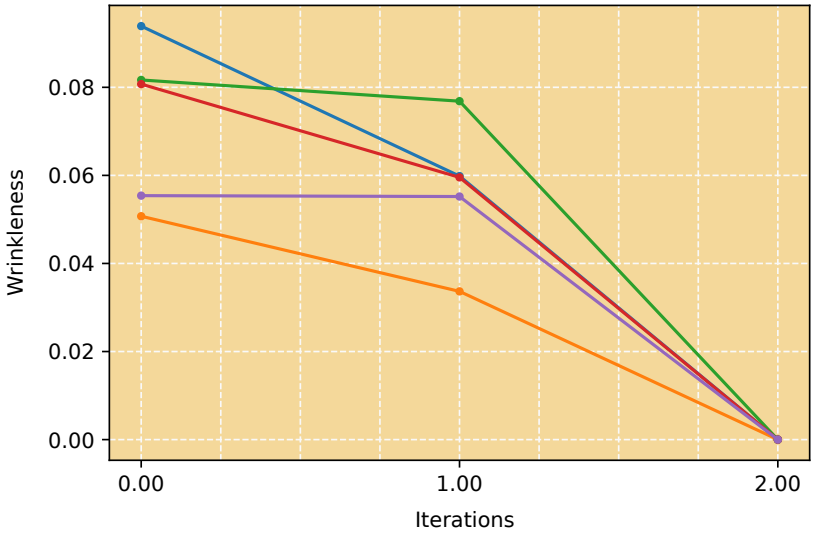

Figure 7.10: Wrinkleness on each experiment trial, from iteration 0 (initial wrinkleness) to iteration 2. No trial required more than 2 iterations to achieve zero wrinkleness (i.e. no wrinkles between the WiLD thresholds could be found).

7.4 FOLDING

This section describes the experiments for the folding algorithm presented in Chapter 6. As the folding neural controller and cloth state estimation model are an ongoing work, only partial validation of the performance has been executed and reported in this section. Still, and for the sake of completeness, the author will include in this section his proposal of the required future experiments for the validation of this approach, as well as a thorough justification of the necessity and usefulness of each proposed experiment.

7.4.1 *Purpose of the Experiment*

The folding experiments have two main objectives: to validate the usefulness and accuracy of the FoldNet model for cloth state estimation during folding, and to check if the improved

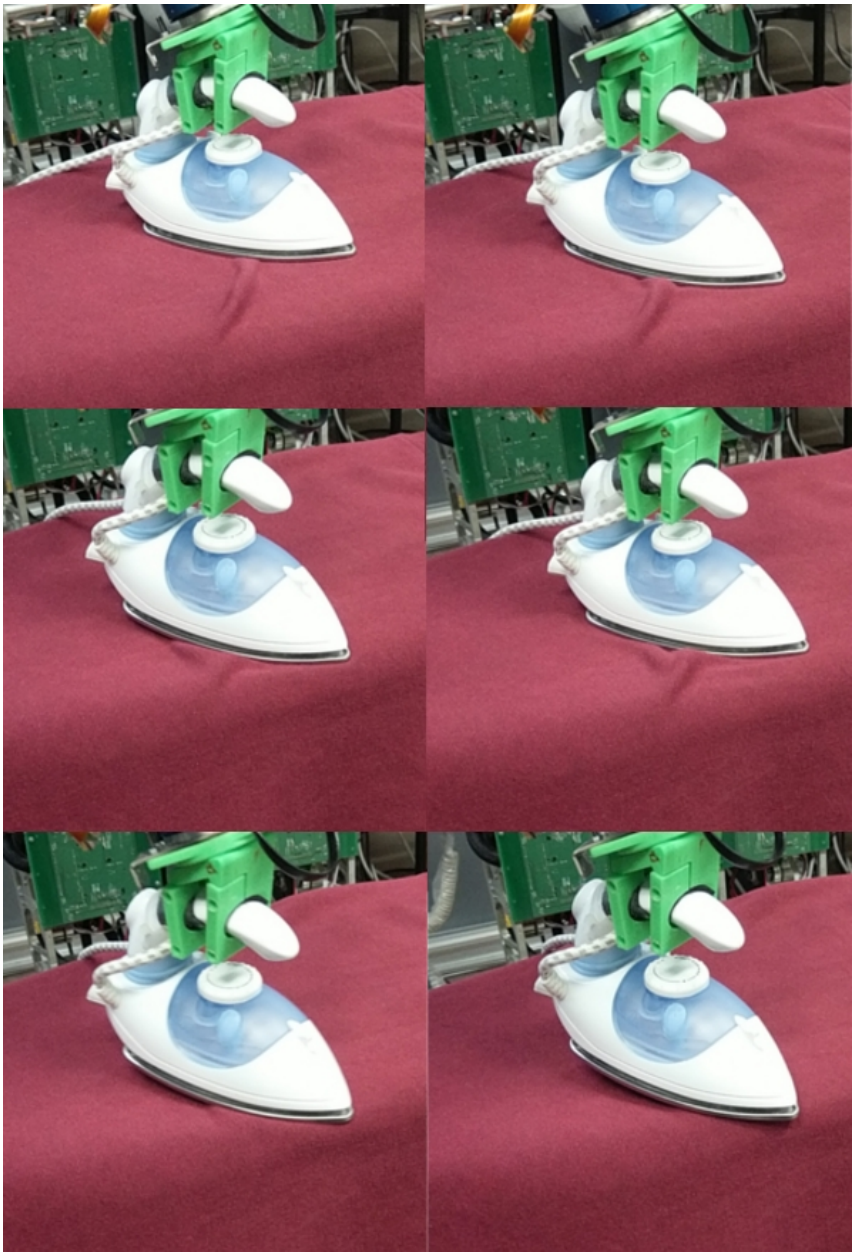

Figure 7.11: Several frames of an ironing operation executed as part of an experiment.

state estimation surpasses the performance of both the original neural controller as well as other state-of-the-art methods for folding.

More precisely, the accuracy of the cloth state estimation has to be measured and compared with existing methods to determine how precise is the estimation and what kind of estimation errors are incurred by the method. The evaluation will be performed through both simulated and real world images, to assess the generalization capabilities of the method.

Additionally, the neural controller has to be evaluated experimentally with the original and the improved state estimation methods to determine if there is an improvement in performance with the new approach, as expected. Correspondingly, the improved version of the neural controller will be compared with two state-of-the-art methods for folding: the triangular and circular folding paths, to determine if there is an improvement in accuracy with respect to them too.

7.4.2 *Experimental Setup*

For the evaluation of the cloth state estimation model, as well as testing the results in simulation using the test set, real data is required to validate the generalization capabilities of the model once applied to real data. For that purpose, real data needs to be captured and labeled for the validation of the method.

The proposed setup to perform the data capture would be a setup in which several cloths of different properties are attached, one at a time, to a flat working surface. A side camera will be placed to capture a side view of the cloth strip being folded, from which the actual shape of the cloth can be extracted while being folded, and measured. Another camera is placed to capture an RGB image of the cloth from a different point of view, calibrating the system to determine the relative locations of the cloth strip and both cameras. Once the system is calibrated and set up, a robot arm can proceed to fold the cloth strip using different paths, in a similar way as how the synthetic data was captured. The state estimation obtained from the FoldNet model can be matched with the labeled data to check the accuracy of the estimation. Figure 7.12 depicts the proposed experimental setup.

For the neural controller evaluation, the ideal setup would be one that replicates as close as possible the setup employed to test the original controller, to be able to perform a comparison

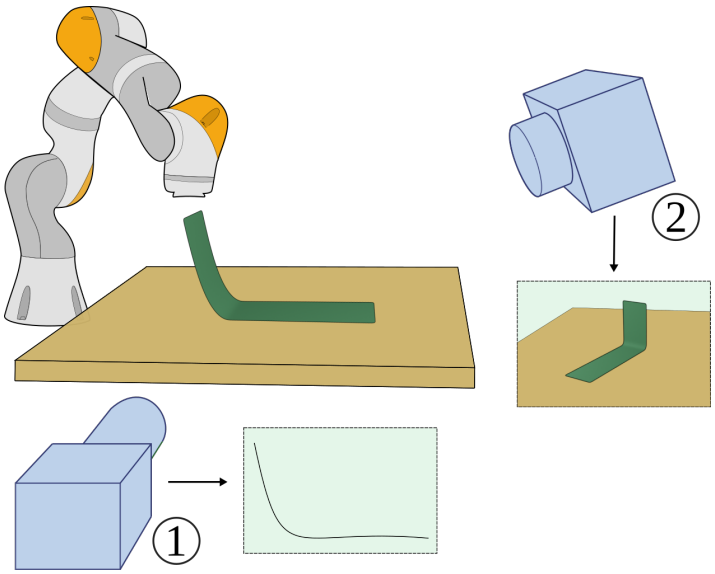

Figure 7.12: Proposed folding setup. A cloth strip is fixed to a flat surface, ready to be manipulated by a robot arm, while a pair of cameras gather different views of the cloth strip. Camera 1 records a side view of the cloth that allows to estimate the cloth shape while folding. Camera 2 obtains an RGB image of the cloth that is used as input for the estimation algorithm.

of both systems under similar conditions. Therefore, the setup would feature a cloth strip attached to a white flat surface and a external calibrated camera recording the scene to obtain the visual feedback. The original robot used for the experiments was a Franka Emika Panda robot, although other similar robot arms such as the KUKA IIWA available in the CTU Prague labs can be used instead.

Additionally, and to validate the usefulness of the approach for a robot that could be deployed in a domestic environment, one of the aims of this thesis, the performance of the neural controller should be tested and evaluated in our humanoid platform TEO as well.

7.4.3 *Experimental Evaluation*

To measure the performance of the cloth state estimation model using the synthetic data test set, the estimated cloth shape is matched with the actual shape deformation obtained from the

simulation. The average error of all the 241 3D points that define the shape is computesp as:

$$\text{Error} = \frac{1}{N} \cdot \sum_{i}^{N} \sqrt{\sum_{j}^{3} (Y_{ij} - \hat{Y}_{ij})^2} \tag{7.8}$$

Where $N = 271$ is the total number of 3D points that define the cloth shape, Y_{ij} is the current coordinate (X, Y, or Z) of the current point of the ground truth cloth shape, and $\hat{Y}_{ij}$ is the corresponding coordinate of the estimated cloth shape.

For the real data, the metrics to be used would be identical to the ones used with the synthetic data, with the only difference being that the labels of the ground truth (the 241 3D points that define the cloth shape) would have to be extracted from an RGB image, either using classical computer vision techniques or by hand. An alternative approach would be measuring the error of each of the 241 estimated points with respect to the measured shape, defined with either a higher resolution polygonal chain or a continuous curve.

As for the neural controller, the same metric as the original paper would be used: layer displacement measured from the first contact point to both extremes after a folding operation. Additionally, other metrics such as time required for the folding operation or for computing the required visual feedback could be incorporated.

7.4.4 *Esperimental Results*

As mentioned along the thesis, the proposed folding method is an ongoing work, and therefore only partial validation through some of the proposed experiments has been performed and is reported in this section, leaving the remaining experiments as future work.

The FoldNet cloth state estimation model has been tested with the synthetic dataset generated from simulation using a test set of 5712 elements with promising results. The results for that preliminary testing are reported in this section, along with a study of how the different parameters of the synthetic data affect the accuracy of the results.

Using Equation 7.8, the FoldNet model obtained an average error of 0.008 m ,computed throughout the test set examples. The standard deviation of the error was 0.003 69 m, with

the more accurate examples having an error of about 0.002 m-0.003 m, and the less accurate having an error ranging between 0.03 m and 0.04 m.

Figure 7.13 depicts some of the examples showing all the accuracy range. As can be observed, in the case of the examples with a higher error, the shape of the cloth is still captured accurately, but shifted from the actual shape location. This could be due to the model not perceiving correctly the depth or scale of the image, as having only a flat ground surface might not provide enough visual cues to estimate depth from a monocular image.

In addition to the study of the accuracy of the model, average error values were computed based on different parameters of the synthetic dataset, to study the influence of different factors on the accuracy of the estimation. The factors selected were the folding path being used, the camera point of view and the frame of the simulation. Figure 7.14 shows the different results obtained.

In the case of the simulation frame, as expected, early frames have less errors than later frames, as the cloth strip in a flat state is simpler to estimate than the folded strip. In the case of the different trajectories, the difference in errors between them is not that significant, being only about 0.0006 m between the best and the worst trajectory errors. The factor that most influences the accuracy of the estimation is the camera point of view, with a difference of about 0.005 m between the best and the worst performances.

In the case of the camera point of view, the best results (left side of the graph) correspond to camera poses closer to the cloth and with a lower elevation angle from the ground, while the worst results (right side of the graph) correspond to points of view closer to a bird's eye perspective of the cloth strip.

7.5 CHAPTER SUMMARY

To test the validity of the algorithms presented in this thesis, a thorough experimental evaluation was performed for all the tasks of the laundry pipeline addressed in this work: hanging, unfolding, ironing and folding.

For hanging, the experiments aimed to understand the effect of time on the uncertainty of predictions about the behavior of a garment about to be hanged, and to compare the performance

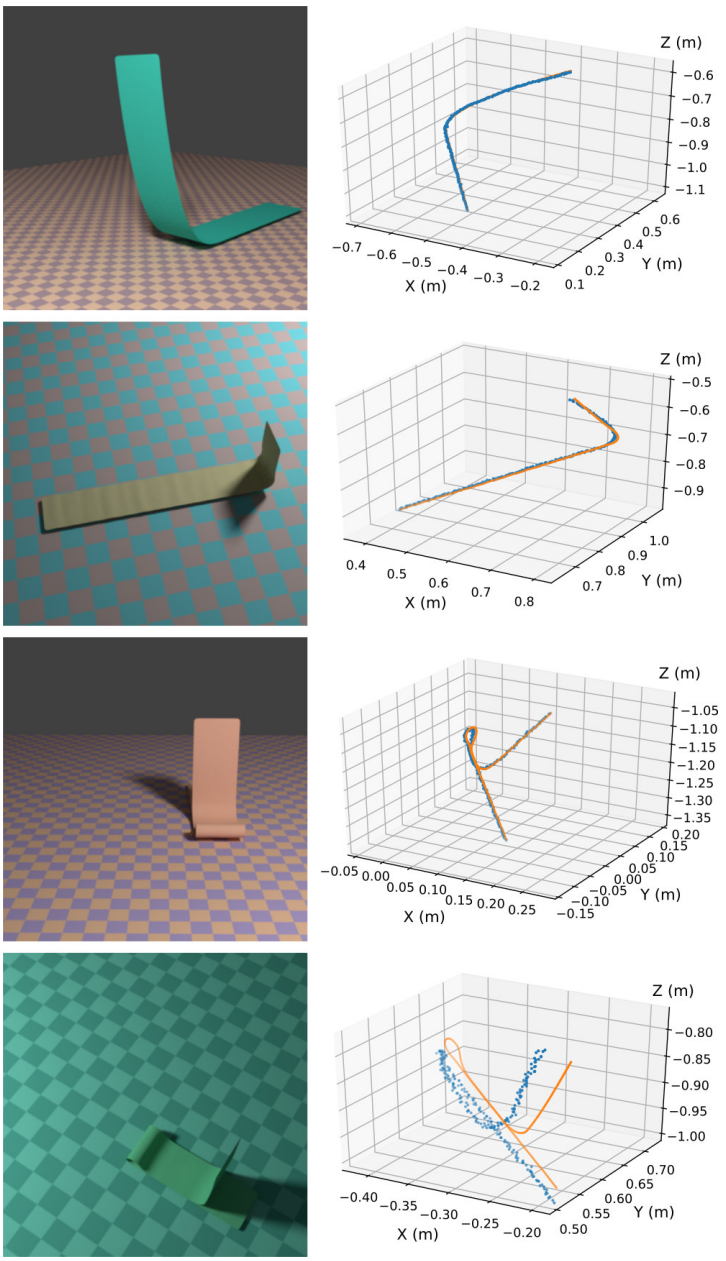

Figure 7.13: Selection of cloth state estimation results. On the left, the input image is shown; on the right, the estimation of the model (blue) is shown along the ground truth (orange). It can be observed that even in the cases with a larger mean error (bottom figure), the model is still able to capture the cloth shape.

of the presented HangNet model with a baseline provided by a human expert.

For unfolding, two different approaches based on color and 3D data were evaluated and compared through extensive testing with garments from 6 categories in three sets of experi-

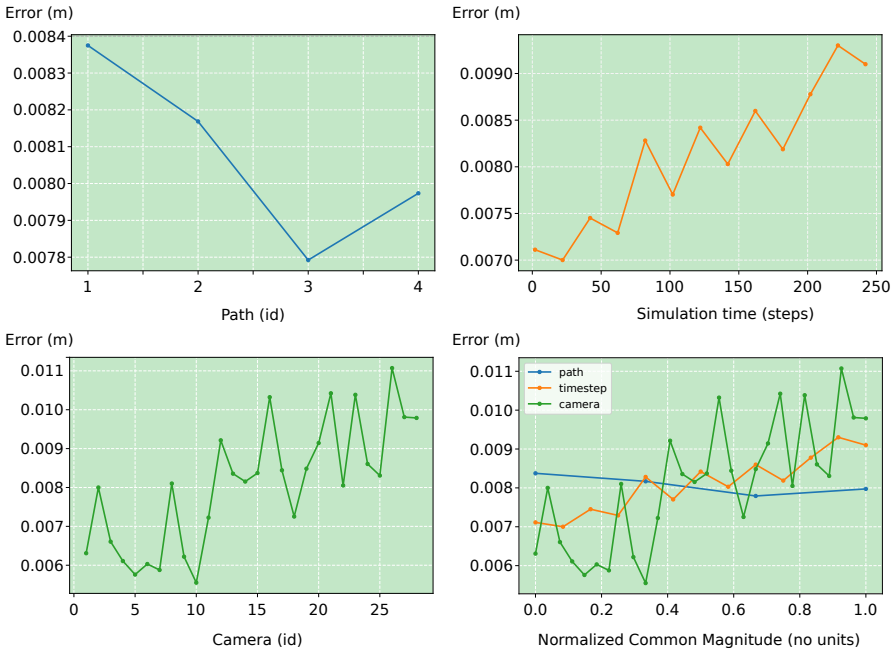

Figure 7.14: This figure shows the different errors classified according to different factors: path used for folding (top left), simulation timestep (top right) and camera point of view (bottom left). The last figure (bottom right) serves as a comparison of the different factors in the same scale.

ments. Both approaches were tested on a humanoid robot and an industrial manipulator.

For ironing, two experiments were designed to independently test the wrinkle detection capabilities of the perception algorithm, as well as the whole ironing pipeline. These experiments were conducted in an emulated domestic environment, with a standard ironing board and iron, and uncontrolled light conditions.

For folding, a study of the preliminary results obtained in simulation was provided, along with directions about how to perform an extensive evaluation of the complete folding algorithm, including a proposed experimental setup and metrics as future work.

8

RESULTS AND CONCLUSIONS

Doing laundry is a complex process involving a pipeline composed of several tasks that are very different in nature. This thesis has presented an automated approach to the different tasks belonging to the laundry pipeline: hanging, unfolding, ironing and folding. Each of these tasks has been analyzed and tackled individually by the author, leading to several methods, one per task, that can be applied with different robotic platforms, including a full-body humanoid robot.

This chapter aims to offer a discussion of the experimental results presented in Chapter 7, examining the main contributions of this work to the state of the art on deformable object perception and manipulation. Additionally, the limitations of the different methods are discussed at the end of the chapter, providing several possible lines of future work to address them.

8.1 PROGRESS BEYOND THE STATE OF THE ART

The aim of this section is to present the contributions of this thesis to the advance of the state of the art on garment perception and manipulation. As the methods proposed to solve each of the tasks in the laundry pipeline are very disparate, they will be analyzed independently, and the contributions of each method will be shown in the context of their corresponding task.

Amongst all the contributions listed below, the author would like to highlight his work on robotic ironing as the most relevant contribution of this thesis to the State of the Art on robotic garment perception and manipulation. The proposed ironing method improves existing state-of-the-art approaches by being able to successfully remove wrinkles in a domestic, uncontrolled environment using unmodified human tools. The contributions of this method will be discussed in detail in the corresponding section, Section 8.1.3. In addition to the scientific contributions, the author would like to highlight the interest attracted by this particular method since its publication in the International Conference on Intelligent Robots and Systems (IROS), not only by

the research community, but also by the general public, as it has been featured in several international press publications[1][2][3][4].

8.1.1 *Hanging*

This thesis presents a detailed study of the feasibility of predicting the dynamic behavior of a garment during the hanging task, defined as dropping a clothing article over a hanger. To predict the outcome of a hanging action, a synthetic dataset composed by a total of 15 000 examples was obtained via deformable object simulation.

Two models based on deep convolutional neural networks were trained using the synthetic data to predict the behavior of a piece of clothing when dropped over a hanger. Each of them was trained for a different task, either determining if the garment will hang or fall (classification) or estimating the future location of the garment after a given amount of time (regression). In addition, the performance of this method was compared with the prediction abilities of a human expert, obtaining very positive results.

8.1.2 *Unfolding*

The unfolding task is defined as fully extending a garment over a flat surface, starting from a garment state in which some parts of the garment are overlapping other parts. This thesis introduces a model-less approach to garment unfolding that requires no prior knowledge of the garment category to determine the most suitable actions to unfold a given garment. The proposed method leverages a custom metric called *bumpiness* in combination with a set of heuristics to determine both the fold axis and the corresponding manipulation actions required to unfold the garment.

The proposed method can be applied directly to depth data gathered by a RGB-D sensor, making it color-independent and immune to the patterns frequently found in garments. Additionally, the presented method does not require bi-manipulation

1 https://www.bbc.com/news/av/technology-40435011/ironing-robot-tackles-creased-clothes, last visited 01-06-2020
2 https://www.newscientist.com/article/2138264, last visited 01-06-2020
3 https://www.digitaltrends.com/cool-tech/teo-robot-ironing-home/, last visited 01-06-2020
4 https://gizmodo.com/1796380533, last visited 01-06-2020

to spread the garment, although it can benefit from it, and has been validated successfully on different robotic platforms. Such platforms include a full-body humanoid robot and an industrial manipulator. An extensive experimental evaluation is provided demonstrating the effectiveness of this approach.

8.1.3 *Ironing*

This thesis presents a method to perform robotic ironing on simple garments. The main contribution of the proposed method, as opposed to other existing methods based on *marked creases* detection, is that it focuses on achieving a feasible implementation in a domestic setting. As a consequence, this work removes the need for an extensive control of the light conditions of the working environment, a requirement of existing methods.

To be able to be deployed in an unmodified domestic setup, this approach leverages the use of 3D perception to detect large *soft wrinkles*, and force/torque feedback to perform the ironing operation in a compliant manner. This way, the robot can iteratively iron a garment using the same unmodified tools that a person would use (i.e. iron and ironing board).

To detect the *soft wrinkle* regions a novel 3D descriptor called WiLD (Wrinkleness Local Descriptor) was developed that, as opposed to other generic 3D descriptors such as RSD, is tailored specifically to find such regions on garments. As the experimental results show, the WiLD descriptor is faster and more precise than other generic alternatives, and allows the robot to completely iron the surface of a garment in a few iterations.

8.1.4 *Folding*

This thesis presents a method for garment state estimation based on a deep neural network applied to garment folding. The proposed method, developed in collaboration with CTU Prague, improves their original neural controller applied to the folding task by estimating the shape of the garment being folded with a higher detail -241 3D points- from purely visual feedback.

Although this method is an ongoing work, preliminary tests show promising results in simulation. Additionally, a complete experimental evaluation of the method on real robotic platforms is proposed and will be performed in the future to fully validate the method.

In addition to the garment estimation method, a synthetic dataset of 29 120 training examples for the cloth strip folding task has been compiled from simulated data, with a large visual variability in terms of textures, illumination, background, poses, etc, that will be released to the public domain accompanying the pending publication of the method.

8.2 FUTURE LINES OF WORK

It is of great importance to reflect and offer an honest discussion of the limitations of this work so that they can be taken into consideration and addressed in the future. For the same purpose, it is equally important to additionally propose possible improvements and future lines of research relative to this work.

One of the main limitations of this work as a whole is the independence between the different tasks of the laundry pipeline. Individually, the presented methods can be applied to solve each of the tasks, but they still require some initial setup to place the garment in the initial working conditions for each of them. An interesting extension to this work would be to focus on the intermediate steps to be performed between tasks, so that the whole laundry pipeline could be executed end-to-end without human intervention.

Regarding each of the individual tasks, their main shortcomings will be analyzed individually in the following subsections.

8.2.1 *Hanging*

As observed in the results obtained during the hanging experiments, one of the main limitations of the proposed deep convolutional neural network-based model is the decrease of accuracy in the prediction as time advances. The cause of the decreasing accuracy is an increase in uncertainty as the model has to predict the future of a chaotic system. This issue could be addressed by explicitly taking into account the temporal component of the garment movement in the regression model, using a different architecture such as a Recurrent Neural Network.

An alternative approach could be to integrate both regression and classification models to increase the accuracy of the predictions by training the model in two different populations: the samples where the garment falls to the floor and the samples in which the garment remains hanged.

The classification model surpasses the performance of a human expert for all metrics computed, except for the recall of the hanged class, which can be seen as a limitation of this model. This might be due to the imbalance of the hanged class of the synthetic dataset used to train the model, and could be improved by obtaining more examples of the hanged class to balance the dataset.

Finally, this study works as a first approach to the hanging task that paves the road for more complex solutions involving experiments with real garments to validate the results obtained with the synthetic data, as well as the addition of a controller that uses this prediction model to successfully hang garments.

8.2.2 *Unfolding*

As mentioned in the corresponding section (Section 7.2.4), the performance of the Segmentation Stage has been improved by the color-based approach with a reconstruction-based approach. However, for very thin garments, the reconstruction approach might result in a bad segmentation, as RANSAC sometimes confuses thin parts of the garment as being part of the table. In these cases, some color information may increase the accuracy of the segmentation.

Regarding the Clustering Stage, the overall performance of the stage might be improved with a fine tuning of the watershed parameters based on the garment material or thickness. Other clustering algorithms can be considered, aimed at increasing robustness against wrinkles, which can result in small clusters within the overlapped regions. Due to the recent success of convolutional neural networks for object segmentation, a possible future line of work might be to replace the Segmentation and Clustering stages with a model able to directly segment the overlapping regions from the RGB-D (or even just RGB) data from the sensor.

As the presented method is focused on visual perception, there is room for more work to be developed in the domain of manipulation. One aspect to improve could be grasping, by performing material studies for increasing gripper-garment friction, and adding tactile feedback to avoid grasping both the overlapped region and the underlying garment. Additionally, the orientation of the gripper, as well as the trajectory for the pick and place operation have been manually selected and fixed for all the experiments, so further research is required to deter-

mine if this strategy is optimal or results could be improved by selecting a trajectory and gripper orientation based on each particular unfolding case.

8.2.3 *Ironing*

Due to the great diversity of garment shapes, textures, materials and decorative elements, ironing is a challenging task to automate. The proposed method, as a first approach to a practical robot ironing process, is focused in ironing simple garments.

This initial work can be extended by the addition of a mechanism to detect and take into account different elements present in garments, such as buttons, zippers or other decorative elements. Ideally, the generated ironing trajectory should avoid these elements. Other garment parts, such as collars, cuffs or pockets require different ironing approaches, that in addition have to be adapted depending on their size and location. Detecting such elements and incorporating their location in the trajectory generator would benefit the proposed method.

Another limitation of the current method is the color uniformity constraint, that limits this approach to garments with a single color. A more relaxed constraint, or the incorporation of other segmentation methods such as convolutional neural networks would allow this algorithm to work with garments with more than one color, or even decorative patterns.

Regarding manipulation, position and velocity control have been tested with force/torque feedback. While velocity control reduces the jerk of the movements, it must be bounded to avoid excessive joint space velocities when the determinant of the Jacobian is near zero. There is obvious room for testing torque control using the transposed Jacobian matrix J^T which may enable enhancements through active compliance.

In addition, due to the complexity of the ironing task, the current approach only tackles the actual ironing action, but this approach could be extended by including the remaining steps of the ironing process to have a fully automated process. One of these steps is the manipulation of the garment for its placement in the ironing board. Once the current garment patch is correctly ironed, this garment has to be removed and, if necessary, placed again in the next configuration.

Another possible step that might improve the performance of the ironing method is to pull the garment once it has been placed on the ironing board, to reduce the amount of *soft wrin-*

kles initially present in the garment before ironing. This step would require a robot able to perform bimanipulation to pull to garment while holding the garment in position.

8.2.4 *Folding*

Even though this thesis has presented some preliminary results of our proposed folding method, it is still an ongoing work that is yet to be refined and published. As such, most of the future lines of work that this section would have to introduce, have already been discussed in previous sections. Therefore, this section will serve as a review and summary of all the proposed lines of action to be continued in the future.

The most important line of work that has be completed in the near future is a full experimental evaluation of the proposed cloth state estimation method, both in simulation and in real world conditions. The experiments proposed in Section 7.4.2 will validate the accuracy of the state estimation model alone, by comparing the shape estimations obtained through the FoldNet model with synthetic data and hand-labeled data, measuring the magnitude of the estimation errors and comparing them with state-of-the-art methods.

Once the validation of the state estimation method has been performed individually, the complete improved neural controller has to be tested with a real robotic manipulator to measure the accuracy of the folding controller on real cloth strips. The experimental setup proposed for such experiments tries to replicate as close as possible the original experiments on Petrík's neural controller, so that the results obtained in the experiments can be comparable. In addition, experiments with our humanoid robot TEO are proposed to validate the usefulness of the neural controller on a real domestic setting.

Along with the experiments proposed in this work, additional extensions can be made to the folding method. For instance, the neural controller could be extended to be able to represent a more complex policy, which might increase the performance of the controller. The current controller works on simple cloth strips that serve as a base to extend the method to more complex cloth geometries, such as the ones found in real garments. Finally, this work focuses on a single folding operation, but it could be extended to a complete folding sequence, validating the approach.

BIBLIOGRAPHY

[1] Benjamin Balaguer and Stefano Carpin. "Combining imitation and reinforcement learning to fold deformable planar objects." In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2011, pp. 1405–1412.

[2] Niklas Bergström, Carl Henrik Ek, Danica Kragic, Yuji Yamakawa, Taku Senoo, and Masatoshi Ishikawa. "Online learning of temporal state models for flexible objects." In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pp. 712–718.

[3] Christian Bersch, Benjamin Pitzer, and Sören Kammel. "Bimanual robotic cloth manipulation for laundry folding." In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2011, pp. 1413–1419.

[4] Angel X Chang, Thomas Funkhouser, Leonidas Guibas, Pat Hanrahan, Qixing Huang, Zimo Li, Silvio Savarese, Manolis Savva, Shuran Song, Hao Su, et al. "Shapenet: An information-rich 3d model repository." In: *arXiv preprint arXiv:1512.03012* (2015).

[5] Lan Chen, Juntao Ye, Liguo Jiang, Chengcheng Ma, Zhanglin Cheng, and Xiaopeng Zhang. "Synthesizing cloth wrinkles by CNN-based geometry image superresolution." In: *Computer Animation and Virtual Worlds* 29.3-4 (2018), e1810.

[6] Enric Corona, Guillem Alenyà, Antonio Gabas, and Carme Torras. "Active garment recognition and target grasping point detection using deep learning." In: *Pattern Recognition* 74 (2018), pp. 629–641.

[7] Marco Cusumano-Towner, Arjun Singh, Stephen Miller, James F O'Brien, and Pieter Abbeel. "Bringing clothing into desired configurations with limited perception." In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2011, pp. 3893–3900.

[8] Jian S Dai, Paul M Taylor, Hong H Liu, and Hua Lin. "Folding algorithms and mechanisms synthesis for robotic ironing." In: *International Journal of Clothing Science and Technology* 16.1/2 (2004), pp. 204–214.

[9] Jian S Dai, Paul M Taylor, P Sanguanpiyapan, and Hua Lin. "Trajectory and orientation analysis of the ironing process for robotic automation." In: *International Journal of Clothing Science and Technology* 16.1/2 (2004), pp. 215–226.

[10] Satonori Demura, Kazuki Sano, Wataru Nakajima, Kotaro Nagahama, Keisuke Takeshita, and Kimitoshi Yamazaki. "Picking up One of the Folded and Stacked Towels by a Single Arm Robot." In: *2018 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE. 2018, pp. 1551–1556.

[11] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. "Imagenet: A large-scale hierarchical image database." In: *2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE. 2009, pp. 248–255.

[12] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature." In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[13] Andreas Doumanoglou, Tae-Kyun Kim, Xiaowei Zhao, and Sotiris Malassiotis. "Active random forests: An application to autonomous unfolding of clothes." In: *European Conference on Computer Vision (ECCV)*. Springer. 2014, pp. 644–658.

[14] Andreas Doumanoglou, Andreas Kargakos, Tae-Kyun Kim, and Sotiris Malassiotis. "Autonomous active recognition and unfolding of clothes using random decision forests and probabilistic planning." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 987–993.

[15] Andreas Doumanoglou, Jan Stria, Georgia Peleka, Ioannis Mariolis, Vladimir Petrik, Andreas Kargakos, Libor Wagner, Václav Hlaváč, Tae-Kyun Kim, and Sotiris Malassiotis. "Folding clothes autonomously: A complete

pipeline." In: *IEEE Transactions on Robotics* 32.6 (2016), pp. 1461–1478.

[16] Christof Elbrechter, Robert Haschke, and Helge Ritter. "Folding paper with anthropomorphic robot hands using real-time physics-based modeling." In: *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE. 2012, pp. 210–215.

[17] Zackory Erickson, Henry M Clever, Greg Turk, C Karen Liu, and Charles C Kemp. "Deep haptic model predictive control for robot-assisted dressing." In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 1–8.

[18] David Estevez and Juan G Victores. *FoldNet - Hanging Garments Dataset*. https://doi.org/10.5281/zenodo.3932102. Version 1.0. 2020. DOI: 10.5281/zenodo.3932102.

[19] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography." In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[20] Andrea Frome, Daniel Huber, Ravi Kolluri, Thomas Bülow, and Jitendra Malik. "Recognizing objects in range data using regional point descriptors." In: *European Conference on Computer Vision (ECCV)*. Springer. 2004, pp. 224–237.

[21] Antonio Gabas, Enric Corona, Guillem Alenyà, and Carme Torras. "Robot-aided cloth classification using depth information and CNNs." In: *International Conference on Articulated Motion and Deformable Objects*. Springer. 2016, pp. 16–23.

[22] Daniel Fernandes Gomes, Shan Luo, and Luis F Teixeira. "GarmNet: Improving Global with Local Perception for Robotic Laundry Folding." In: *Annual Conference Towards Autonomous Robotic Systems*. Springer. 2019, pp. 49–61.

[23] Kyoko Hamajima and Masayoshi Kakikura. "Planning Strategy for Unfolding Task of Clothes-Isolation of clothes from washed mass." In: *Proceedings of the 35th SICE Annual Conference. International Session Papers*. IEEE. 1996, pp. 1237–1242.

[24] Tao Han, Xuan Zhao, Peigen Sun, and Jia Pan. "Robust shape estimation for 3D deformable object manipulation." In: *Communications in Information and Systems* 18.2 (2018), pp. 107–124.

[25] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep residual learning for image recognition." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778.

[26] Yew Cheong Hou, Khairul Salleh Mohamed Sahari, Leong Yeng Weng, Dickson Neoh Tze How, and Hiroaki Seki. "Particle-based perception of garment folding for robotic manipulation purposes." In: *International Journal of Advanced Robotic Systems* 14.6 (2017).

[27] Zhe Hu, Peigen Sun, and Jia Pan. "Three-dimensional deformable object manipulation using fast online Gaussian process regression." In: *IEEE Robotics and Automation Letters* 3.2 (2018), pp. 979–986.

[28] Zhe Hu, Tao Han, Peigen Sun, Jia Pan, and Dinesh Manocha. "3-D Deformable Object Manipulation Using Deep Neural Networks." In: *IEEE Robotics and Automation Letters* 4.4 (2019), pp. 4255–4261.

[29] Rishabh Jangir, Guillem Alenya, and Carme Torras. "Dynamic Cloth Manipulation with Deep Reinforcement Learning." In: *arXiv preprint arXiv:1910.14475* (2019).

[30] Biao Jia, Zhe Hu, Jia Pan, and Dinesh Manocha. "Manipulating highly deformable materials using a visual feedback dictionary." In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2018, pp. 239–246.

[31] Biao Jia, Zherong Pan, Zhe Hu, Jia Pan, and Dinesh Manocha. "Cloth Manipulation Using Random-Forest-Based Imitation Learning." In: *IEEE Robotics and Automation Letters* 4.2 (2019), pp. 2086–2093.

[32] Ravi P Joshi, Nishanth Koganti, and Tomohiro Shibata. "Robotic cloth manipulation for clothing assistance task using dynamic movement primitives." In: *Proceedings of the Advances in Robotics*. 2017, pp. 1–6.

[33] Takashi Kabaya and Masayoshi Kakikura. "Service robot for housekeeping: Clothing handling." In: *Journal of Robotics and Mechatronics* 10 (1998), pp. 252–257.

[34] Manabu Kaneko and Masayoshi Kakikura. "Planning strategy for putting away laundry-isolating and unfolding task." In: *Proceedings of the 2001 IEEE International Symposium on Assembly and Task Planning (ISATP2001). Assembly and Disassembly in the Twenty-first Century.(Cat. No. 01TH8560)*. IEEE. 2001, pp. 429–434.

[35] Oussama Khatib, Kazu Yokoi, Oliver Brock, K Chang, and Arancha Casal. "Robots in human environments: Basic autonomous capabilities." In: *The International Journal of Robotics Research* 18.7 (1999), pp. 684–696.

[36] Yasuyo Kita and Nobuyuki Kita. "Virtual flattening of clothing item held in the air." In: *2016 23rd International Conference on Pattern Recognition (ICPR)*. IEEE. 2016, pp. 2770–2776.

[37] Yasuyo Kita. and Nobuyuki Kita. "Virtual Flattening of a Clothing Surface by Integrating Geodesic Distances from Different Three-dimensional Views." In: *Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 5: VISAPP,* INSTICC. SciTePress, 2019, pp. 541–547.

[38] Yasuyo Kita, Toshio Ueshiba, Ee Sian Neo, and Nobuyuki Kita. "A method for handling a specific part of clothing by dual arms." In: *2009 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2009, pp. 4180–4185.

[39] Yasuyo Kita, Toshio Ueshiba, Ee Sian Neo, and Nobuyuki Kita. "Clothes state recognition using 3d observed data." In: *2009 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2009, pp. 1220–1225.

[40] Yasuyo Kita, Ee Sian Neo, Toshio Ueshiba, and Nobuyuki Kita. "Clothes handling using visual recognition in cooperation with actions." In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2010, pp. 2710–2715.

[41] Yasuyo Kita, Fumio Kanehiro, Toshio Ueshiba, and Nobuyuki Kita. "Clothes handling based on recognition by strategic observation." In: *2011 11th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2011)*. IEEE. 2011, pp. 53–58.

[42] Hiroko Kobori, Youhei Kakiuchi, Kei Okada, and Masayuki Inaba. "Recognition and motion primitives for autonomous clothes unfolding of humanoid robot." In: *Intelligent Autonomous Systems 11, IAS 2010* (Jan. 2010), pp. 57–66.

[43] Jan J Koenderink and Andrea J Van Doorn. "Surface shape and curvature scales." In: *Image and vision computing* 10.8 (1992), pp. 557–564.

[44] Yosuke Koishihara, Solvi Arnold, Kimitoshi Yamazaki, and Takamitsu Matsubara. "Hanging work of T-shirt in consideration of deformability and stretchability." In: *2017 IEEE International Conference on Information and Automation (ICIA)*. IEEE. 2017, pp. 130–135.

[45] Petar Kormushev, Sylvain Calinon, and Darwin G Caldwell. "Imitation learning of positional and force skills demonstrated via kinesthetic teaching and haptic input." In: *Advanced Robotics* 25.5 (2011), pp. 581–603.

[46] Daniel Kruse, Richard J Radke, and John T Wen. "Collaborative human-robot manipulation of highly deformable materials." In: *2015 IEEE international conference on robotics and automation (ICRA)*. IEEE. 2015, pp. 3782–3787.

[47] Loan Le, Matteo Zoppi, Michal Jilich, Han Bo, Dimiter Zlatanov, and Rezia Molfino. "Application of a biphasic actuator in the design of the clopema robot gripper." In: *Journal of Mechanisms and Robotics* 7.1 (2015).

[48] Yinxiao Li, Chih-Fan Chen, and Peter K Allen. "Recognition of deformable object category and pose." In: *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2014, pp. 5558–5564.

[49] Yinxiao Li, Yan Wang, Michael Case, Shih-Fu Chang, and Peter K Allen. "Real-time pose estimation of deformable objects using a volumetric approach." In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 1046–1052.

[50] Yinxiao Li, Yonghao Yue, Danfei Xu, Eitan Grinspun, and Peter K Allen. "Folding deformable objects using predictive simulation and trajectory optimization." In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, pp. 6000–6006.

[51] Yinxiao Li, Danfei Xu, Yonghao Yue, Yan Wang, Shih-Fu Chang, Eitan Grinspun, and Peter K Allen. "Regrasping and unfolding of garments using predictive thin shell modeling." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 1382–1388.

[52] Yinxiao Li, Xiuhan Hu, Danfei Xu, Yonghao Yue, Eitan Grinspun, and Peter K Allen. "Multi-sensor surface analysis for robotic ironing." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 5670–5676.

[53] Jeremy Maitin-Shepard, Marco Cusumano-Towner, Jinna Lei, and Pieter Abbeel. "Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding." In: *2010 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2010, pp. 2308–2315.

[54] Ioannis Mariolis, Georgia Peleka, Andreas Kargakos, and Sotiris Malassiotis. "Pose and category recognition of highly deformable objects using deep learning." In: *2015 International Conference on Advanced Robotics (ICAR)*. IEEE. 2015, pp. 655–662.

[55] Donald W Marquardt. "An algorithm for least-squares estimation of nonlinear parameters." In: *Journal of the society for Industrial and Applied Mathematics* 11.2 (1963), pp. 431–441.

[56] Luz Martínez, Javier Ruiz-del Solar, Li Sun, J Paul Siebert, and Gerardo Aragon-Camarasa. "Continuous perception for deformable objects understanding." In: *Robotics and Autonomous Systems* 118 (2019), pp. 220–230.

[57] Santiago Martínez, Concepción Alicia Monje, Alberto Jardón, Paolo Pierro, Carlos Balaguer, and Delia Munoz. "Teo: Full-size humanoid robot design powered by a fuel cell system." In: *Cybernetics and Systems* 43.3 (2012), pp. 163–180.

[58] Zoltan-Csaba Marton, Dejan Pangercic, Nico Blodow, Jonathan Kleinehellefort, and Michael Beetz. "General 3D modelling of novel objects from a single view." In: *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2010, pp. 3700–3705.

[59] Jan Matas, Stephen James, and Andrew J Davison. "Sim-to-Real Reinforcement Learning for Deformable Object Manipulation." In: *Conference on Robot Learning*. 2018, pp. 734–743.

[60] Stephen Miller, Mario Fritz, Trevor Darrell, and Pieter Abbeel. "Parametrized shape models for clothing." In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2011, pp. 4861–4868.

[61] Stephen Miller, Jur Van Den Berg, Mario Fritz, Trevor Darrell, Ken Goldberg, and Pieter Abbeel. "A geometric approach to robotic laundry folding." In: *The International Journal of Robotics Research* 31.2 (2012), pp. 249–267.

[62] Richard A Newcombe, Shahram Izadi, Otmar Hilliges, David Molyneaux, David Kim, Andrew J Davison, Pushmeet Kohi, Jamie Shotton, Steve Hodges, and Andrew Fitzgibbon. "KinectFusion: Real-time dense surface mapping and tracking." In: *2011 10th IEEE International Symposium on Mixed and Augmented Reality*. IEEE. 2011, pp. 127–136.

[63] Michael Oren and Shree K Nayar. "Generalization of the Lambertian model and implications for machine vision." In: *International Journal of Computer Vision* 14.3 (1995), pp. 227–251.

[64] Fumiaki Osawa, Hiroaki Seki, and Yoshitsugu Kamiya. "Unfolding of massive laundry and classification types by dual manipulator." In: *Journal of Advanced Computational Intelligence and Intelligent Informatics* 11.5 (2007), pp. 457–463.

[65] Nobuyuki Otsu. "A threshold selection method from gray-level histograms." In: *IEEE transactions on systems, man, and cybernetics* 9.1 (1979), pp. 62–66.

[66] Antoine Petit, Vincenzo Lippiello, Giuseppe Andrea Fontanelli, and Bruno Siciliano. "Tracking elastic deformable objects with an RGB-D sensor for a pizza chef robot." In: *Robotics and Autonomous Systems* 88 (2017), pp. 187–201.

[67] Vladimír Petrík and Ville Kyrki. "Feedback-based Fabric Strip Folding." In: *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2019, pp. 773–778.

[68] Vladimír Petrík, Vladimir Smutny, and Ville Kyrki. "Static stability of robotic fabric strip folding." In: *IEEE/ASME Transactions on Mechatronics* (2020).

[69] Albert Pumarola, Antonio Agudo, Lorenzo Porzi, Alberto Sanfeliu, Vincent Lepetit, and Francesc Moreno-Noguer. "Geometry-aware network for non-rigid shape prediction from a single view." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4681–4690.

[70] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. "Pointnet: Deep learning on point sets for 3d classification and segmentation." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017, pp. 652–660.

[71] Urs Ramer. "An iterative procedure for the polygonal approximation of plane curves." In: *Computer graphics and image processing* 1.3 (1972), pp. 244–256.

[72] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. "You only look once: Unified, real-time object detection." In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 779–788.

[73] Olga Russakovsky et al. "Imagenet large scale visual recognition challenge." In: *International journal of computer vision* 115.3 (2015), pp. 211–252.

[74] Radu Bogdan Rusu, Nico Blodow, and Michael Beetz. "Fast point feature histograms (FPFH) for 3D registration." In: *2009 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2009, pp. 3212–3217.

[75] Radu Bogdan Rusu and Steve Cousins. "3D is here: Point Cloud Library (PCL)." In: *IEEE International Conference on Robotics and Automation (ICRA)*. Shanghai, China, 2011.

[76] Daniel Seita, Nawid Jamali, Michael Laskey, Ajay Kumar Tanwani, Ron Berenstein, Prakash Baskaran, Soshi Iba, John Canny, and Ken Goldberg. "Deep Transfer Learning of Pick Points on Fabric for Robot Bed-Making." In: *International Symposium on Robotics Research (ISRR)*. 2019.

[77] Beucher Serge and Christian Lantuéj. "Use of watersheds in contour detection." In: *Workshop on Image Processing, Real-time Edge and Motion Detection, Rennes, France*. 1979.

[78] Karen Simonyan and Andrew Zisserman. "Very deep convolutional networks for large-scale image recognition." In: *arXiv preprint arXiv:1409.1556* (2014).

[79] Jan Stria and Václav Hlavác. "Classification of Hanging Garments Using Learned Features Extracted from 3D Point Clouds." In: *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2018, pp. 5307–5312.

[80] Jan Stria, Vladimír Petrík, and Vaclav Hlaváč. "Model-free approach to garments unfolding based on detection of folded layers." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 3274–3280.

[81] Jan Stria, Daniel Průša, and Václav Hlaváč. "Polygonal models for clothing." In: *Conference Towards Autonomous Robotic Systems*. Springer. 2014, pp. 173–184.

[82] Jan Stria, Daniel Průša, Václav Hlaváč, Libor Wagner, Vladimír Petrík, Pavel Krsek, and Vladimír Smutnỳ. "Garment perception and its folding using a dual-arm robot." In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 61–67.

[83] Li Sun, Gerarado Aragon-Camarasa, Paul Cockshott, Simon Rogers, and J Paul Siebert. "A heuristic-based approach for flattening wrinkled clothes." In: *Conference Towards Autonomous Robotic Systems*. Springer. 2013, pp. 148–160.

[84] Li Sun, Gerardo Aragon-Camarasa, Simon Rogers, and J Paul Siebert. "Accurate garment surface analysis using an active stereo robot head with application to dual-arm flattening." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 185–192.

[85] Li Sun, Gerardo Aragon Camarasa, Aamir Khan, Simon Rogers, and Paul Siebert. "A precise method for cloth configuration parsing applied to single-arm flattening." In: *International Journal of Advanced Robotic Systems* 13.2 (2016), p. 70.

[86] Li Sun, Simon Rogers, Gerardo Aragon-Camarasa, and J Paul Siebert. "Recognising the clothing categories from free-configuration using gaussian-process-based interactive perception." In: *2016 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2016, pp. 2464–2470.

[87] Li Sun, Gerardo Aragon-Camarasa, Simon Rogers, Rustam Stolkin, and J Paul Siebert. "Single-shot clothing category recognition in free-configurations with application to autonomous clothes sorting." In: *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2017, pp. 6699–6706.

[88] Daisuke Tanaka, Solvi Arnold, and Kimitoshi Yamazaki. "EMD Net: An encode–manipulate–decode network for cloth manipulation." In: *IEEE Robotics and Automation Letters* 3.3 (2018), pp. 1771–1778.

[89] Dimitra Triantafyllou and Nikos A Aspragathos. "Upper Layer Extraction of a Folded Garment Towards Unfolding by a Robot." In: *International Conference on Robotics in Alpe-Adria Danube Region*. Springer. 2018, pp. 597–604.

[90] Dimitra Triantafyllou, Ioannis Mariolis, Andreas Kargakos, Sotiris Malassiotis, and Nikos Aspragathos. "A geometric approach to robotic unfolding of garments." In: *Robotics and Autonomous Systems* 75 (2016), pp. 233–243.

[91] Yoshihisa Tsurumine, Yunduan Cui, Eiji Uchibe, and Takamitsu Matsubara. "Deep reinforcement learning with smooth policy update: Application to robotic cloth manipulation." In: *Robotics and Autonomous Systems* 112 (2019), pp. 72–83.

[92] Lukas Twardon and Helge Ritter. "Interaction skills for a coat-check robot: Identifying and handling the boundary components of clothes." In: *2015 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2015, pp. 3682–3688.

[93] Jur Van Den Berg, Stephen Miller, Ken Goldberg, and Pieter Abbeel. "Gravity-based robotic cloth folding." In: *Algorithmic Foundations of Robotics IX*. Springer, 2010, pp. 409–424.

[94] Ping Chuan Wang, Stephen Miller, Mario Fritz, Trevor Darrell, and Pieter Abbeel. "Perception for the manipulation of socks." In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2011, pp. 4877–4884.

[95] Bryan Willimon, Stan Birchfield, and Ian Walker. "Classification of clothing using interactive perception." In: *2011 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2011, pp. 1862–1868.

[96] Bryan Willimon, Stan Birchfield, and Ian Walker. "Model for unfolding laundry using interactive perception." In: *2011 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2011, pp. 4871–4876.

[97] Bryan Willimon, Ian Walker, and Stan Birchfield. "3D non-rigid deformable surface estimation without feature correspondence." In: *2013 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE. 2013, pp. 646–651.

[98] Bryan Willimon, Steven Hickson, Ian Walker, and Stan Birchfield. "An energy minimization approach to 3D non-rigid deformable surface estimation using RGBD data." In: *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2012, pp. 2711–2717.

[99] Yilin Wu, Wilson Yan, Thanard Kurutach, Lerrel Pinto, and Pieter Abbeel. "Learning to Manipulate Deformable Objects without Demonstrations." In: *arXiv preprint arXiv:1910.13439* (2019).

[100] Yuji Yamakawa, Akio Namiki, and Masatoshi Ishikawa. "Dynamic manipulation of a cloth by high-speed robot system using high-speed visual feedback." In: *IFAC Proceedings Volumes* 44.1 (2011), pp. 8076–8081.

[101] Kimitoshi Yamazaki. "Grasping point selection on an item of crumpled clothing based on relational shape description." In: *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2014, pp. 3123–3128.

[102] Kimitoshi Yamazaki. "Selection of grasp points of cloth product on a table based on shape classification feature." In: *2017 IEEE International Conference on Information and Automation (ICIA)*. IEEE. 2017, pp. 136–141.

[103] Kimitoshi Yamazaki and Masayuki Inaba. "Clothing classification using image features derived from clothing fabrics, wrinkles and cloth overlaps." In: *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2013, pp. 2710–2717.

[104] Kimitoshi Yamazaki, Ryosuke Oya, Kotaro Nagahama, and Masayuki Inaba. "A method of state recognition of dressing clothes based on dynamic state matching." In: *Proceedings of the 2013 IEEE/SICE International Symposium on System Integration*. IEEE. 2013, pp. 406–411.

[105] Pin-Chu Yang, Kazuma Sasaki, Kanata Suzuki, Kei Kase, Shigeki Sugano, and Tetsuya Ogata. "Repeatable folding task by humanoid robot worker using deep learning." In: *IEEE Robotics and Automation Letters* 2.2 (2016), pp. 397–403.

[106] Cheong Hou Yew and Khairul Salleh Mohamed Sahari. "Real-time modeling and parameter approximation of dexterous garment folding by robot." In: *Artificial Life and Robotics* 24.1 (2019), pp. 119–126.

[107] Cheong Hou Yew, Khairul Salleh Mohamed Sahari, and Tze How Dickson Neoh. "Parametrized 2D/3D Garment Model For Dexterous Robotic Manipulation." In: *2018 Joint 10th International Conference on Soft Computing and Intelligent Systems (SCIS) and 19th International Symposium on Advanced Intelligent Systems (ISIS)*. IEEE. 2018, pp. 287–291.

[108] Tongjie Y. Zhang and Ching Y. Suen. "A fast parallel algorithm for thinning digital patterns." In: *Communications of the ACM* 27.3 (1984), pp. 236–239.