CENTER FOR
MACHINE PERCEPTION

CZECH TECHNICAL
UNIVERSITY IN PRAGUE

MASTER'S THESIS

World Representation of a Dual-Arm Robot Manipulating with Clothes

Jan Šindler

sindljan@fel.cvut.cz

May 10, 2013

Thesis Advisor: Václav Hlaváč

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

World Representation of a Dual-Arm Robot Manipulating with Clothes

Jan Šindler

May 10, 2013

Acknowledgment

I would like to thank all the people who help me in my diploma thesis project. The biggest thanks belongs to my supervisor prof. Václav Hlaváč for keeping me on the track and also for constantly reviewing and correcting my texts.

A substantial piece of my work was performed at KTH in Stockholm. Several people from KTH helped me. Christian Smith answered many my questions about robotic manipulation, robotics in general and guided me while I was at Computer Vision and Active Perception Lab of KTH (CVAP) from January to May 2013. Mikael Vejdemo-Johansson helped me with mathematical background of manifolds and suggested me useful books on the topic. I thank Prof. Danica Kragic, the head CVAP, for accepting me warmly and for valuable advices. The big thanks to professor emeritus Jan-Olof Eklundh for helping me with arrangement of the stay at KTH. I am indebted to the Erasmus program that allowed me to enjoy a fantastic international experience.

I thank to Kateřina Hejná for help with English language. Thanks also belongs to Libor Wagner and Vladimír Petrík for their illustration images of a robotic device used at CTU and suggestions connected with a robotic device used at CTU, to Aram Simonian for reading this thesis and his essential notes, to Stephen Miller from the University of California, Berkeley who helped me with running his implementation of the clothing model.

Finally, I would like to thank to my family for supporting my education.

Prohlášení autora práce

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

Declaration

I hereby declare that I have written the diploma thesis myself, using only cited sources according to Methodological guideline of ethical principals connected with thesis writing.

V Praze dne 9.5.2013 .... Jan Gindl

Podpis autora práce

iii

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: Bc. Jan Šindler

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Diploma Thesis: World Representation of a Dual-Arm Robot Manipulating with Clothes

Guidelines:

1. Make yourself familiar with robotized soft material manipulation, e.g. clothes, from the following points of view: robot world representation, planning in it, its connection with sensorial and actuator subsystem, and implementation of practical scenarios.
2. Propose a robot world representation of a dual-arm robot for above mentioned tasks, planning methods and its connection to sensorial and actuator subsystem.
3. Explore existing third party modules. Aim at the implementation independent on a specific robot, because you will work on the diploma project under the Erasmus scholarship umbrella both at CTU in Prague and at KTH in Stockholm.
4. Propose two simple scenarios and verify the implementation on them practically.
5. Describe implemented modules in a design and implementation documentation. Describe experiments.

Bibliography/Sources:

[1] Kita, Y.; Neo, E. S.; Ueshiba, T.; Kita, N.: Clothes handling using visual recognition in cooperation with actions. In IEEE International Conference on Intelligent Robots and Systems, pp. 2710 - 2715, October 2010.

[2] Rubio, O. J.; Huebner, K.; Kragic, D.: Representations for object grasping and learning from experience. In IEEE International Conference on Intelligent Robots and Systems, October 2010.

[3] Dahiya, R. S.; Valle, M.: Robotic Tactile Sensing, Technologies and System, Springer, Berlin 2013.

[4] Additional items will be suggested by the supervisor.

Diploma Thesis Supervisor: prof. Ing. Václav Hlaváč, CSc.

Valid until: the end of the summer semester of academic year 2013/2014

L.S.

prof. Ing. Vladimír Mařík, DrSc.
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Czech Technical University in Prague
Faculty of Electrical Engineering

Department of Cybernetics

DIPLOMA THESIS ASSIGNMENT

Student: Bc. Jan Šindler

Study programme: Cybernetics and Robotics

Specialisation: Robotics

Title of Diploma Thesis: World Representation of a Dual-Arm Robot Manipulating with Clothes

Guidelines:

1. Make yourself familiar with robotized soft material manipulation, e.g. clothes, from the following points of view: robot world representation, planning in it, its connection with sensorial and actuator subsystem, and implementation of practical scenarios.
2. Propose a robot world representation of a dual-arm robot for above mentioned tasks, planning methods and its connection to sensorial and actuator subsystem.
3. Explore existing third party modules. Aim at the implementation independent on a specific robot, because you will work on the diploma project under the Erasmus scholarship umbrella both at CTU in Prague and at KTH in Stockholm.
4. Propose two simple scenarios and verify the implementation on them practically.
5. Describe implemented modules in a design and implementation documentation. Describe experiments.

Bibliography/Sources:

[1] Kita, Y.; Neo, E. S.; Ueshiba, T.; Kita, N.: Clothes handling using visual recognition in cooperation with actions. In IEEE International Conference on Intelligent Robots and Systems, pp. 2710 - 2715, October 2010.

[2] Rubio, O. J.; Huebner, K.; Kragic, D.: Representations for object grasping and learning from experience. In IEEE International Conference on Intelligent Robots and Systems, October 2010.

[3] Dahiya, R. S.; Valle, M.: Robotic Tactile Sensing, Technologies and System, Springer, Berlin 2013.

[4] Additional items will be suggested by the supervisor.

Diploma Thesis Supervisor: prof. Ing. Václav Hlaváč, CSc.

Valid until: the end of the summer semester of academic year 2013/2014

L.S.

prof. Ing. Vladimír Mařík, DrSc.
Head of Department

prof. Ing. Pavel Ripka, CSc.
Dean

Abstract

This master thesis contributes to tasks aiming at perceiving soft objects as textiles or pieces of clothing and manipulating with them using robots. We used a towel and a T-shirt in our experiments.

We started from Miller's (University of California, Berkeley) parameterized model, which represents the piece of clothing. It creates a skeletal model for a given class clothing manually, e.g. for a T-shirt. The approach is suited for spread piece of clothing only. We reimplemented this model based on the original code. We improved Miller's model. Our version copes with no so well spread piece of clothing. Nevertheless, the improved version does not cope with folded pieces of clothing.

The next contribution of this diploma project is the proposal how to represent a piece of clothing using manifold. This extended model should cope with folded pieces of clothing. We did not implement this model, though.

We prepared several software interfaces enabling coexistence between different representations of clothing and a particular robotic device. The interface is designed in such a way that one robotic device can be easily replace by another one or augmented by other sensors.

Abstrakt

Tato diplomová práce přispívá k úlohám vnímání a robotické manipulace s měkkými objekty jako jsou textilie či kusy oblečení. Pro experimenty jsme používali ručník a tričko.

Vyšli jsme z Millerova (University of California, Berkeley) parametrizovaného modelu, který reprezentuje vybraný kus oblečení. Pro daný typ oblečení, např. tričko, jsme ručně z obrysu vytvořili skeletální model. Tento přístup se hodí pouze pro rozprostřené textilie. Model jsme reimplementovali. Využili jsme kód autorů. Millerův model jsme vylepšili tak, že se vyrovná i s malými shrnutími na textilii. Vylepšený model přesto nezvládá situace, kdy je textilie přeložená.

Dalším přínosem diplomového projektu je návrh rozšíření modelu, které textilii reprezentuje pomocí variety. Takto rozšířený model zvládá i přeložené textilie. Tento model jsme ale neimplementovali.

Připravili jsme také rozhraní pro spolupráci modelu textilie a robotického zařízení. Rozhraní je navrženo tak, aby bylo možné robotické zařízení snadno zaměnit za jiné, případně ho doplnit novými senzory.

Contents

Abbreviations

CTU	Czech Technical University in Prague
CVAP	Computer Vision and Active Perception Lab at KTH
DOF	Degree of Freedom
HDD	Hard Disc Drive
IR	Infra Red
KTH	Kungliga Tekniska Högskolan (Royal Institute of Technology)
LLE	Locally Linear Embedding
OS	Operating System
$\mathbb{R}^n$	n-dimensional Euclidean space
RML	Riemannian Manifold Learning
ROS	Robot Operating System, a robotic software middleware
(IC)	a link to the source, used in images taken from others (substitute citation)

1 Introduction

The first reference to intelligent robotics for daily use was probably an autonomous vacuum cleaner. This kind of devices are rather dedicated and their intelligence is replaced by a reactive behavior.

There has been attempts both in academia and industry in recent years to develop and sell a more sophisticated robotic devices, which would be able to help people, e.g. in their homes, as an assistants or housekeepers. These robots have to operate in a rather complicated environments. The desire is that robots can also manipulate with various objects, even those they have never seen before.

Manipulation with rigid objects has been studied extensively. This diploma thesis deals with soft objects manipulation, more specially with clothes, soft fabrics, garments. Let us briefly talk about the terminology used. We will use the term clothing, which represents a piece of garment made from a textile material (cloth). The examples can be a T-shirt, shirt, sock, pants, jacket, blouse, etc. We will use simpler pieces made from cloth, which are not piece of garment, as towel, dish towel, handkerchief. We will call them clothing in this work too. We will also use the word clothing in both singular and plural.

We propose a new representation of clothing, which is based on manifolds and features detected on it. We have demonstrated usefulness of the representation in clothing folding and unfolding domain.

1.1 Motivation

This diploma thesis has been motivated by research questions generated in the European Commission funded project Clothes Perception and Manipulation (CloPeMa, 2012-2015) [39]. The project is focused on a garments manipulation by a two armed industrial manipulator. This diploma project should propose a representation of a robot world model enabling perception/manipulation tasks with clothes. The ability of a system to self-learn from gained experience is expected too. The proposed model should be independent of used sensors and a particular robot. The model independence will be tested on two robots, the first one at my domestic university, the Czech Technical University in Prague; the second one at KTH in Stockholm during my four months long Erasmus stay, which have started in the second half of January 2013. The functionality of the proposed robot world model will be demonstrated on an example of a T-shirt and towel folding, which was performed by others before.

1.2 Problem formulation

Humans surpass other living species in abstracting things they meets every day. Specifically, in the clothes perception and manipulation domain, we humans are able to fold various types of clothing without a priori knowledge about a specific piece of garment. We are able to classify observed object into classes learned mainly by experience and use this classification to handle the object properly.

We recognize a basic types of garment as, e.g. pants, sweater, shirt with long sleeves. This knowledge enables us to fold any instance of the basic model. We are able to apply pre-learned actions, which lead us a folded object, e.g. a sweater. The unfolding is performed in a similar fashion.

The primary goal of this diploma project is to propose a clothing model, which should be able to cover above mentioned basic types of garments. The example of a garment is shown in Figure 1.1. The robot should be able to recognize basic types of garment. The robot should be endowed by a collection of actions, which are appropriate for the garment type and which were pre-learned on garment basic type instances. These actions should be applied to an a priori unknown instance of a garment.

Figure 1.1 An example of a subset of garments used in CloPeMa project.

The secondary goal of this thesis is to verify the proposed clothing model experimentally. This goal induces that a test platform should be used. In particular, the CloPeMa testbed in Prague and a robot at KTH Stockholm are at hand. Experiments will allow to compare various clothing models. The test platform should provide connectivity between a robotic device, its sensors, actuators on one side and the clothing model on the other side.

Results of this diploma thesis are demonstrated on a T-shirt and a towel folding.

1.3 Contributions

The diploma project results described in this thesis bring several contributions.

First, we learned the state-of-the-art approach to piece of clothing representation and its application folding by the University of California, Berkeley group. Their work was published in Miller et al. [16]. We reimplemented their approach using the publicly available code, which used ROS environment for interplay between different components. We call it Berkeley implementation.

Second, we created our own experimental mockup consisting from the Kinect sensor and the background of a uniform green color. The manipulation with piece of clothing was performed manually by a human to replace the robot. We incorporated our Kinect sensor into ROS.

Third, we created intermediate software layers (Robot Specification Layer, Robot Interface Layer and Clothing Manipulation Layer), which enable to abstract the future clothing perception and manipulation tasks from a specific sensing and robotics hardware.

Fourth, we improved methods used in the Berkeley implementation. This allows us to cope with a piece of clothing, which is not fully spread. These improvements were

implemented and tested experimentally.

Fifth, we suggested a potentially more powerful representation of a piece of clothing by a manifold. This approach has the ambition to cope with a scrabbled or folded piece of clothing. We did not implement these ideas, though.

1.4 Thesis structure

The thesis is divided into eleven chapters. Chapter 2 introduces the state-of-the art in clothing representation and robotic manipulation. Chapter 3 describes our approach from the methodological point of view. Three brief Chapters 4 to 6 introduce the design of three software layers, which separate the tasks from the actual sensing and robotic hardware. Chapter 7 describes the methods behind the Berkeley implementation and our improvements to it. Chapter 8 provides our theoretical proposal how to represent a piece of clothing by manifolds. Chapter 9 describes our implementation. Chapter 10 summarizes our experimental results. It also deals with our speed improvements and robustness experiments. The last Chapter 11 summarizes the work performed and suggests a potential future work.

2 State of the art

The topic we work on can be looked at from several view points. The description of the views is divided into the following sections. Section 2.1 informs about existing surveys that are connected with our work. Section 2.2 lists projects that are similar to ours. Section 2.3 describes existing clothing models. Section 2.4 is focused to used hardware such as manipulators, grippers and sensors. Section 2.5 summarizes techniques, which other researchers used for a clothing detection and manipulation.

2.1 A global point of view

A summary about what was done in a dual arm manipulation can be found in Smith et al. "Dual arm manipulation survey" [29]. Hu et al. "Review of Cloth Modeling" [14] contains useful links about a clothing 3D approximation from patterns, a clothing approximation by its profile and a parameterized model of clothing.

The above mentioned articles teach that the clothing modeling can be divided into three areas: (a) geometrical models, (b) continuous models and (c) discrete models.

The geometrical models appeared first and were developed by Weil [35]. This representation is suitable for generating static shape of clothing. The basic idea is to represent clothing by basic geometrical objects (lines, squares, etc.), connect them together and bring some geometrical restrictions into it. The most relevant articles are Chiricota [6]-[5] about the fast 3D approximation of clothing from pattern pictures based on parametrization of 2D polygonal curves, and Turquin [32]-[33] that proposes to represent a piece of garment by its silhouette.

Continuous models are trying to cope with physical properties of the clothing such as banding, stretching and shearing. Discrete models were developed because computers are able to work with discrete signals only. One of the most popular models is the Particle model and its extension the mass-spring model.

All of above mentioned models were mostly used for 3D animation. Some more information about these models can be found in the section 2.3.

The article Smith et al. [29] brings a historical background to dual arm manipulation, its usage and developments. One of the mentioned domestic application is use of manipulators in laundry. It brings basic review about used manipulators in this branch and also describes used sensory systems. Another usage of dual arm robots are works in a kitchen (cleaning up a dishwasher for example) and health care (manipulating with paralyzed people). Also a good description about modeling of an robotic system, motion planning and movement controlling can be found in the article.

2.2 Relevant projects

There is a research group at the University of California, Berkeley, Department of Electrical Engineering and Computer Science which has became famous for towel folding videos since 2010. Relevant articles are Shepard et al. [27] and Towner et al. [30]. More information, some datasets and additional articles can be found on web pages [38]. Web pages [36] provide also some information about clothes modeling and manipulating.

The list of other relevant projects I found is:

- STIFF [52] - Its aim is to understand and mimic a variable stiffness paradigm that is used by a human nervous system. The goal is to equip an agile, robust and versatile robot hand-arm.
- VIACTORS [54] - This project aims at developing and exploiting actuators for a new generation of robots that would be able to co-exist and co-operate with people.
- LEAPFROG [44] - A module B of this project named "Automated Garment Assembly" is closely relevant to CloPeMa. It was focused on better handling of limp materials (fabric).
- THE [53] - Its aim is to study a human hand and its sensors, which humans use for fundamental cognitive functions.
- DEXMART [42] - The project is focused to areas where dexterous and autonomous dual-hand manipulation are required.

Articles written about clothing manipulation are rather frequent and can be found in Section 2.5.

2.3 Clothing modeling

Most works about clothing/garment modeling appear in computer graphics. These models usually give up realistic physical modeling due to its complexity. The precise physical models are based on partial differential equations. Their solution using finite elements method is usually too slow for required fast rendering of computer graphics models. Computer graphics thus uses various gross simplifications, which are sufficient for rendering but inappropriate for clothing manipulation tasks. Models based on clothing physical parameters such as elasticity were used in 3D screen rendering. Examples of its usage can be seen in Au et al. [1] or Chen [4]. Almost every work in this area uses Particle-System model, Finite Elements method, Spring-Mass model or alike.

Let us briefly summarize three dominant approaches:

- The *Particle-System model* represents the cloth by a grid of particles. Particles exhibit attraction forces between each other. The strength of attraction is given by the type of used material. The model can easily deal with external forces such as gravity and collisions. It is often used for simulating object dynamics. The example of this modeling system use can be found many articles, e.g. [20], [19] and [7]. The particle models are suitable for modeling of other objects like a fire, explosions and others.
- The *Finite Elements method* attempt to break complex object described physically by a system of partial differential equations into a mesh of finite elements, on which a computationally tractable system of ordinary differential equations can be defined. This approach model has been used in [8] that combines CAD clothing patterns into one object and animates the clothing in dynamic scenes.
- The *Spring-Mass model* was proposed by Provot [21] for animating clothing objects. A piece of clothing is modeled by a group of points placed in a mesh grid connected by three types of strings. The strings simulate physical bindings inside a clothing and are structural (resist stretching), sheer (resist sheering) and flexion (resist bending). The example of model is depicted in Figure 2.1. The animation is generated by forces applied to the model and by taking into account fundamental dynamic laws. The model is used in modern 3D modeling tools like Maya and Blender.

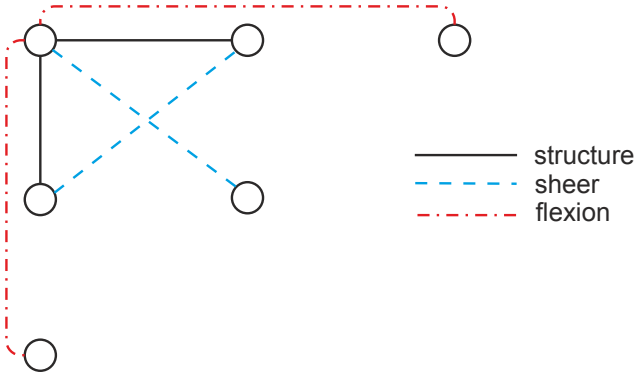


Figure 2.1 The example of a mass-string model.

Clothing modeling for manipulating purposes is covered in the literature only a little to our best knowledge. A point cloud clothing model has been used in [27] and [3]. "Graph of connected hems" have been used in [12] as a clothing model.

The clothing description/model, which is closer to our idea, was introduced in Miller [16]. Their model is based on a parameterized shape description of a clothing. The authors used landmarks detected on the piece of garment and determine constraints between a position of it. They proposed a separate model for each garment type/class (shirt, pants, ...). The model contains a minimal set of landmarks, each augmented with its position. These landmarks should distinguish between the classes. Wang [34] published an extension of the model, which added local features (such as a clothing texture) into it. A more detailed description of both approaches can be found in Section 2.5.

2.4 Hardware used by others

2.4.1 Manipulators

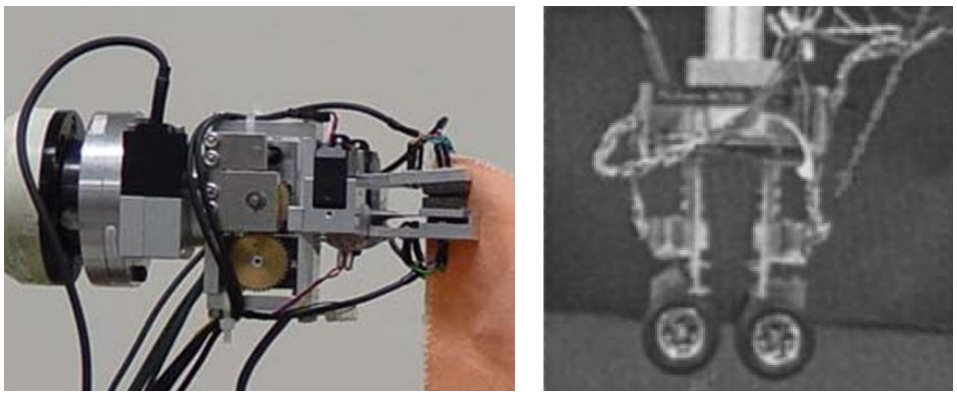
A PR2 robot [49] or its modifications has been the most favorite device used for experiments with garments. The PR2 robot has been used in works [27], [2], [34], [16] and [30]. Kawasaki JS2 and Yamaha Rch-44 were applied in the Salleh [25] work. Asimo [37] in the Gienger [11] work was the only one other robot we have found.

2.4.2 Grippers

Almost every researcher, the research topics of which is not connected with a "How a garment should be held" use standard type of gripper (Figure 2.3) with:

- 2 rotation fingers [3], [27], [30], [34], [16] (Figure 2.3a);
- 3 rotation fingers [23] (Figure 2.3b);
- 2 translation fingers [55] (Figure 2.3c);
- Articulated hand [11] (Figure 2.3d);

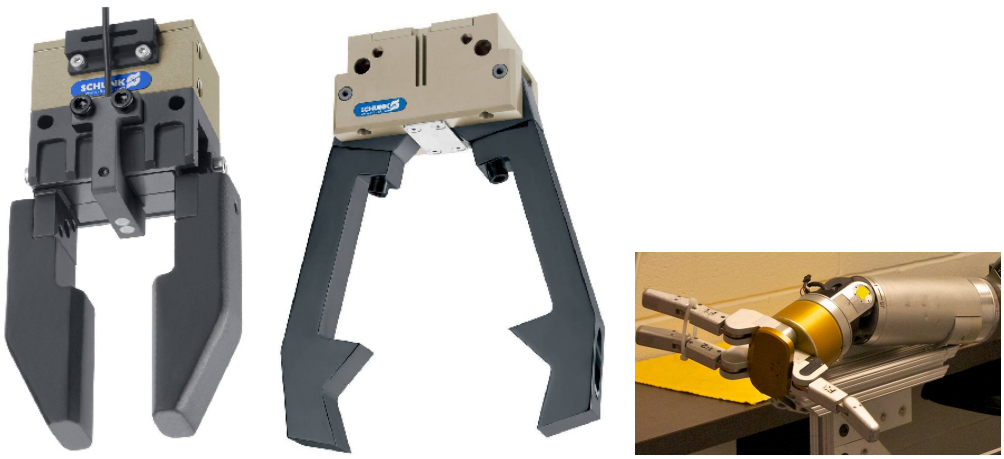
Two types of the grippers, which were not so usual and were designed specially for clothing manipulation were found. These are the Inchworm gripper [25] and the wheeled gripper [12]. The inchworm gripper (Figure 2.2a) is able to move along a garment hem



a) Inchworm gripper. (IC) **b)** Wheeled gripper. (IC)

Figure 2.2 Specialized grippers developed for manipulation with clothing.

easily. The wheeled gripper (Figure 2.2b) can pick up a clothing easily, which lies flat on a table and furthermore it does not need so precise detection of a grasping points.



a) Two translational fingers. **b)** Two rotational fingers. **c)** Three rotational fingers.

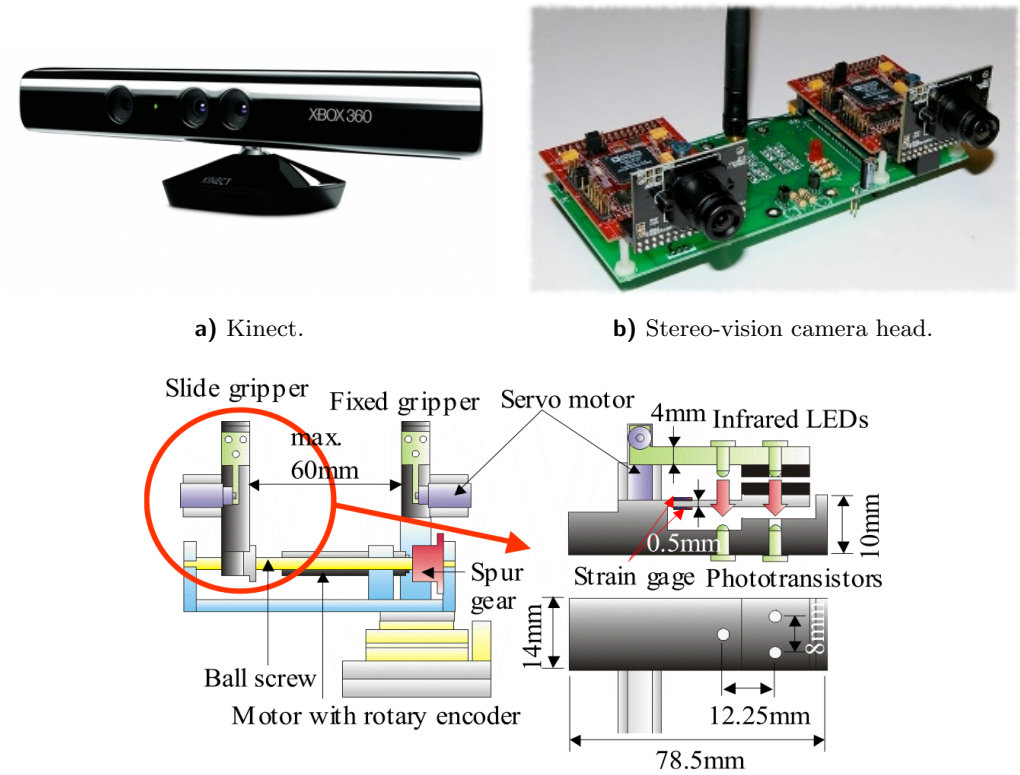
d) Articulated hand.

Figure 2.3 Commonly used grippers (IC).

2.4.3 Sensors

A very wide range of used sensors can be found:

- A Kinect or something that works on the same principle is described in Ramisa [23] (Figure 2.4a) - They used Kinect for a depth map acquisition.
- A stereo-vision camera head [27](figure 2.4b) - It is also used for a depth map acquisition.
- A stereo-vision with a pattern projection [13] - This works on the same principal as Kinect but not in the compact body.
- A single (vision) camera is used as an image source for corner detection in Salleh [25]. The camera was also used as input source of data for features classification and decision making in Wang [34].
- A pressure sensor [3](figure 2.5) - Pressure sensors in this example are used for detection that clothing has been grasped correctly.
- An infrared sensors [25](figure 2.4c) - The IR sensors are used for a clothing corner detection inside a gripper.



a) Kinect.

b) Stereo-vision camera head.

c) IR corner detector in a gripper.

Figure 2.4 Used optical sensors (IC).

2.5 How others manipulate clothing?

We list approaches how others dealt with the clothing perception and manipulation process.

- Maitin-Shepard [27], which has got a lot of popularity in the press, proposed a new technique for a clothing corner detection. The input were images from a stereo-

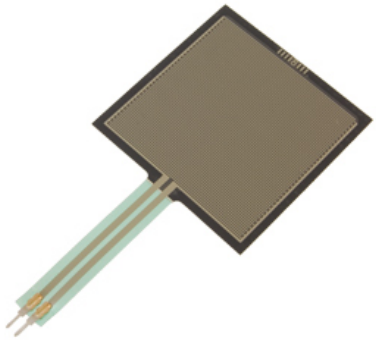


Figure 2.5 Pressure sensor (IC).

vision camera head. Depth-discontinuity points and a depth-discontinuity map were computed from input images. Authors used a sharpness of a curvature to distinguish between folds and edges of a clothing. The process they have used was: picking up a clothing from a pile → shaking the clothing to put it into the low-energy configuration → putting clothing into the basic configuration (corner detection, re-grasping and twisting) → folding towel in an open loop sequence of previously defined moves.

- Bersch [3] used a SVM classifier to learn which points are the best for grasping. That is to say, the article is also focused on the grasp point detection. Features for each of the points are measured. It is learned which combination of the features was the best for grasping. The situation was simplified by fiducial markers stuck on a shirt surface. The following actions were used: picking up a shirt from a table (highest point detection) → creating a model (rotating the shirt and creating a point cloud) → estimating current grasp point (uses a position of the markers) → selecting next grasp point (the learned information was used) → computing grasp pose → executing grasp → performing grasp verification → folding (open loop).
- Ramisa [23] tried to find the best grasping point on a shirt as well. The main motivation of this article was to extract as much as possible information about a shirt that lies on the table before a robot picks it up. A "Bag of Features (BoF)" and SVM classifier were used to create a regression model of a general representation of a shirt. The process was as follows: capturing an image of the shirt → dividing it into subregions → computing geodesic-depth histograms (GDH) for each of the subregion → computing BoF on each GDH and stack it into histograms → comparing the histograms with learned regression model to tell which of the regions may contain a collar → grasping the collar (the highest point in the proper subregion).
- Hamajima [12] tried to deal with one piece of clothing separation from a pile task and the piece of clothing classification. A color segmentation was used. If the color of the pile was solid, shadows that appeared on the clothing were used. A model based on connected hems was used. The decision about which kind of a clothing a robot was manipulating was made by using the information about detected hems and their connections. The model was used for planning clothing unfolding. Hems detection algorithm is given in the article as well.
- The stereo vision with projected markers was used in Hata [13]. Two densities of markers were used. One sparse markers density were used for quicker orientation in the image. Points of the second higher density were used for creating a clothing 3D

model. A towel was used for demonstration. The used process is known from others: highest point detection → picking up → the lowest point detection → re-grasping → ...

- Cusumano-Towner [30] built their algorithm of two basic phases. The first phase is called "Disambiguation Phase". A probabilistic model (Hidden Markov model [HMM]) was used to determine the identity of a piece of clothing and its configuration. The second phase is called "Reconfiguration Phase". In the second phase, authors tried to convert the model from a known configuration to a desired configuration. The HMM was used in this phase to reduce the uncertainty in the model state quickly. Next, a clothing simulator based on a strain-limited model was used. Finally their own planning algorithm generated a plan in a reconfiguration phase. The planning algorithm generated a sequence of re-grasp moves to get the clothing article into a desired configuration.
- Miller [16] proposed a new parameterized shape model for clothing. Its main idea is that a piece of clothing can be parameterized by detected landmarks (such as collar, sleeve, ...). A polygon fitted on detected landmarks defines a shape of a piece of clothing. Constraints between the landmarks allow to distinguish between garment classes (shirt, pants, ...) and enable planning of clothes folding. The polygon can be used to perform an intelligent manipulation of the clothing article. The constraints in the model are determined by skeletal model of a clothing. The authors create a separate model and a folding sequence for each of the garment classes.
  The model has a restriction. It expects that a piece of clothing can be crudely spread out on a flat surface. It assumes that there is a procedure able to spread the clothing. The article describes the used procedure as follows. For each known class model → detect expected landmarks (expectation is given by the selected model) → create a polygonal contour encapsulating landmarks → compute a function (called energy function), which describes quantitatively the correspondence between the model and a particular article → optimize parameters such as rotation, translation, scale and deformation to minimize the energy function. After computation of the energy function for each of the classes, select the one with the minimal energy. A selected energy function determines the class model that is the most probably similar to the manipulated object. The folding procedure, which was prepared for each class in advance, follows.
- Wang [34] dealt with sock manipulation and its paring. The main goal was to convert an initial pose of a sock into to predefined configuration. The pairs between socks were found next. The authors started with a garment model that was based on the model of garment shape proposed in Miller [16]. The model was extended to embody local features detected on the clothing such as the cloth micro-texture. The authors described the following key issues:
  - A minimal set of manipulation primitives needed for sock manipulation.
  - Algorithms used for features computation from garment texture and shape.
  - The way how to use and train the SVM classifier for feature classification such as location of toes and heel areas.
  - The global model that consists from three parts: model itself, model cost and a way how to fit parameters.
  - Two possible ways how to do a sock configuration detection - via an appearance features, via a global model.
  - A sock pairing algorithm.

  The authors performed the task as follows. Measure a feature dataset used for training classifiers → perform the texture classification → recognize a sock configuration

→ change a sock configuration to avoid sock bunching → repeat previous steps for all socks → pair socks.

3 Our approach

3.1 Methodological thoughts

After analyzing the state of the art provided in Chapter 2 and having in mind that this diploma project should potentially consider more than one model of a clothing, we realized an interface between a particular robot testbed and a chosen clothing model is needed. We name it the "Robot Interface Layer". The high-level view at our proposed system is in Figure 3.1. More detailed description of the Robot Interface Layer is in Chapter 5.

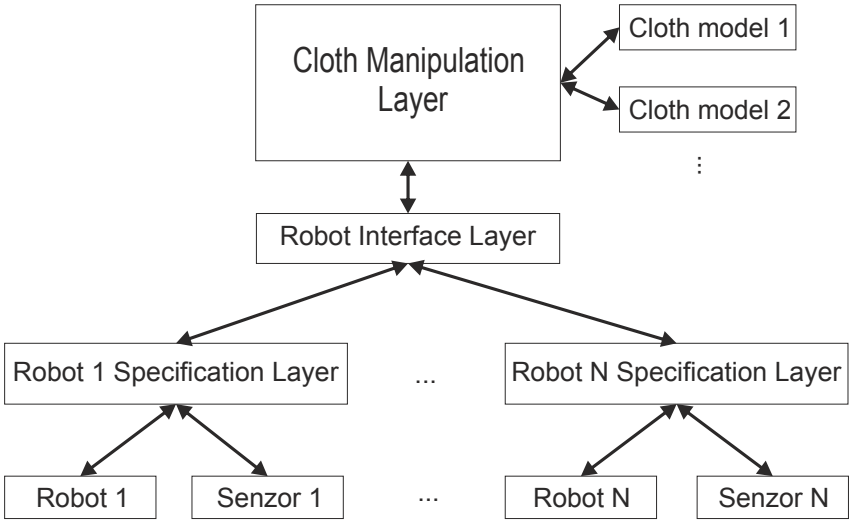


Figure 3.1 A block diagram of a clothing folding system.

There is a caveat, the causality ("chicken or the egg") dilemma. A particular clothing model is needed to create the interface layer. What was first? The clothing model or the Robot Interface Layer? The clothing model cannot be tested without connection to robot sensors; on other hand, the design of the Robot Interface Layer is difficult without a clothing model.

The model proposed in the article Miller et al. [30] is the closest to our approach. We decided to implement it as a base model together with related algorithms allowing to fit the model to the observed piece of garment. We used the implementation provided in the repository [41].

After experiments and and some modification, we chose this clothing model to create the Robot Interface Layer. We realized in the debugging phase that the model did not work as we expected. We decided to change the Miller et al. model slightly and to improve it. The improvement of the clothing model and its implementation is described in Chapter 7.

Our assignment is creating the system for clothing manipulation. The design of the system comes from requirements described in Section 3.2. The system consists from three layers. This approach reduces system complexity, provides encapsulation on both hardware and software levels. We name the system "Test platform" in this thesis. The Test platform is a generic concept. We worked with two instances of it. The first instance is the Yaskawa robots-based dual arm system implemented at CTU in Prague. The second instance is the generic robot system used at KTH Stockholm. Both robot systems were planned to use in our experiments.

Let us describe the architecture in more detail.

- The first layer drawn in Figure 3.1 is the "Robot Specific Layer". The Robot Specific Layer encapsulates a robotic device that is able to perform manipulation tasks and its sensors like tactile sensors or cameras placed on robot body into one object. The other and not less important function of this layer is to reduce functionality of a robot and define the robot affordances. We don't want to be able to do everything with a robot just grasp and move with clothing. The reduction help us to with future planning. In terms of computer science this functionality just implement abstract interface functions defined by Robot Interface Layer. This layer is specific for every robotic device and its sensors. More information can be found in the chapter 4.
- The second layer is the Robot Interface Layer, which can create the homogeneous interface for all of the devices. It reflects demands of Clothing Manipulation Layer for folding and manipulating with clothing as well as demands for sensory data to build clothing model properly. More information can be found in the chapter 5.
- The top layer is the "Clothing Manipulation Layer", which is the main building block of the system. This block takes sensory data one one hand and clothing model on other and combine them together. In other words at the beginning it takes not initialized model of clothing. Let a type of a model be known as a priory information (but it can also be decided what type of clothing is observing). Then the layer combine sensory data and the model to initialize it. After initialization a folding process might begins. This layer control the folding mechanism. During a folding the layer updates the model from sensory data and also creates folds on the model. In other words the main function of this layer is to update the model based on observations of a real object and to decide about actions needed to bring the model into the folded state. More details are given in Chapter 6.

We implemented the parameterized model [30] and conducted experiments with it. Based on the gathered experience, we decided to enhance it. Our new contribution is the use of a "manifold model" of a piece of clothing. Informally, a manifold is a mathematical construct allowing to express entities, which create a distinct subspace in a multidimensional space. In our particular case, the robot working space can be treated as a 3-dimensional (3D) Euclidean space. The cloth in it is a "thin surface", which can be represented locally as a 2D surface. This approach opens the door to more realistic configurations of a garment used in folding/unfolding experiments.

The new manifold-based model was implemented and tested. The gathered experience is described in Chapter 8.

3.2 Requirements

We aim at providing a connection between the robot, sensors and the clothing model. Two pieces of clothing (a towel, a T-shirt) were chosen for testing and debugging. We also used three different clothing models,

1. Miller et al.
2. Our extension to Miller et al.
3. Our new manifold-based model.

In addition, we used two different robot platforms, one at KTH in Stockholm and the another one at CTU in Prague.

This facts induced the following requirements to the whole system:

- Be ready for switching between different models.
- Provide enough information to the user of the system for easier comparison of a chosen clothing model.
- Be applicable to different robotic platforms.
- Be extensible to another sensors or actuators

4 Robot Specification Layer

The main functionality of the "Robot Specification Layer" is to implement abstract functionalities prescribed by the "Robot Interface Layer" and to connect the manipulator actuators, sensors.

Section 4.1 describes used devices and some implementation details related to them. Section 4.2 tells which tools were used.

4.1 Hardware used in our experiments

We planned that our clothing representation method/software will be used on two robotic systems, the first at KTH in Stockholm and the second at CTU in Prague. Before the test with a real robot, a simple manipulator mockup was created and used for software development and initial experimentation.

A simple manipulator mockup

The mockup was created for method design and code development. The mockup is depicted in Figure 4.1. Kinect is used as a sensor and the mockup manipulation abilities are performed by a human. This approach allowed thinking about the method without bothering about a real robot.

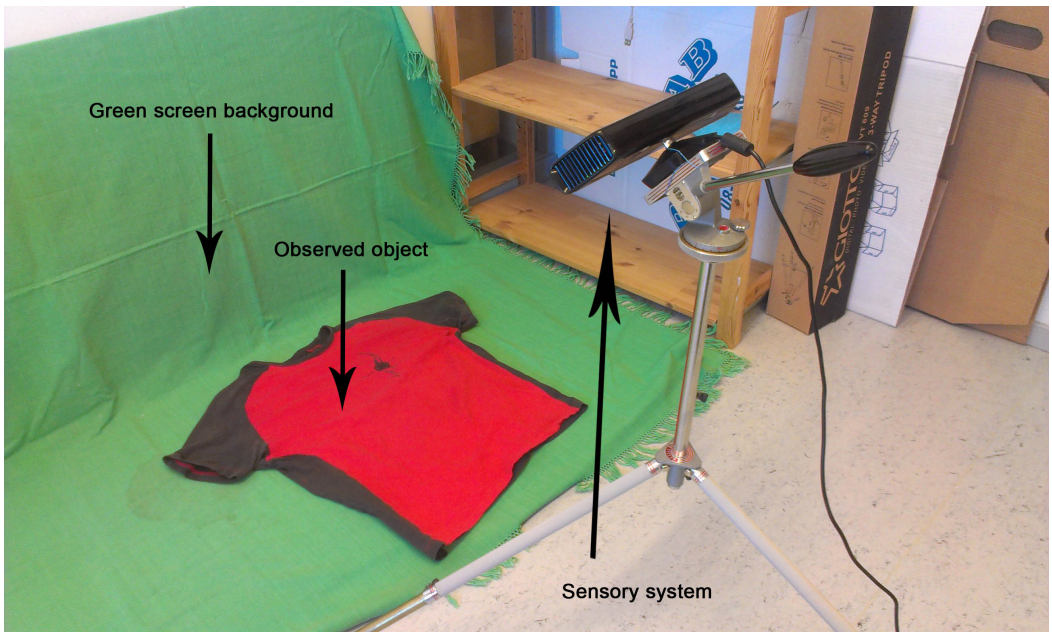


Figure 4.1 The mockup.

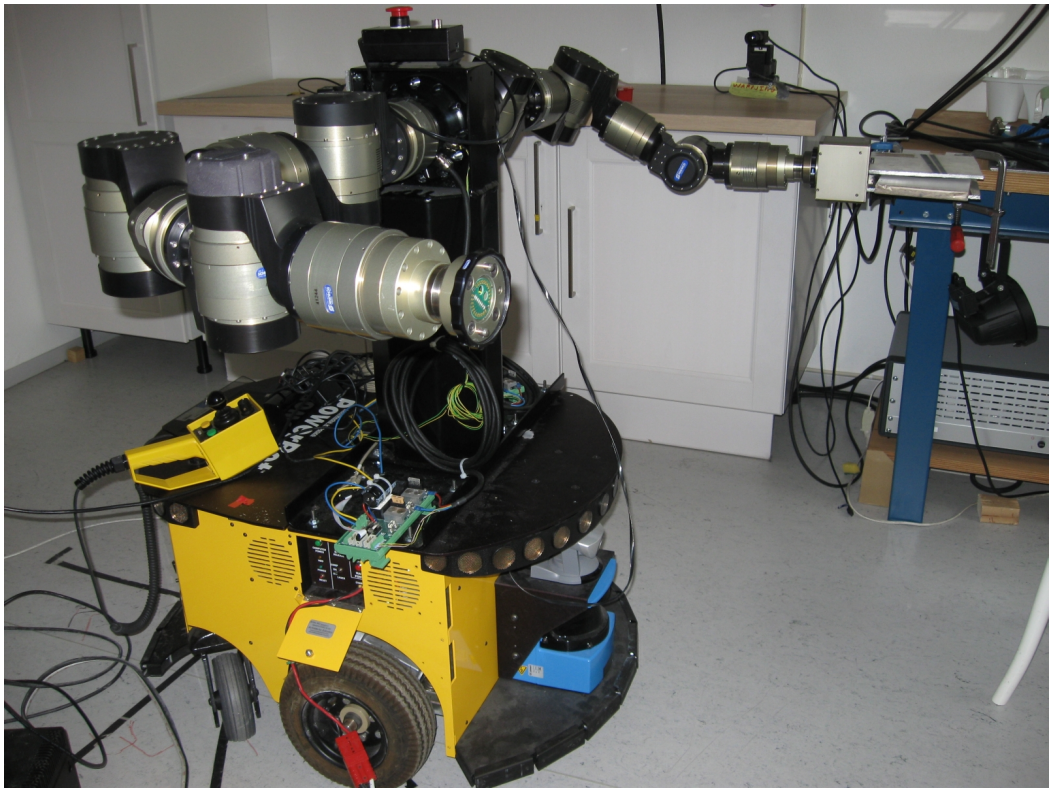


Figure 4.2 Robotic device used at KTH.

Robotic device used at KTH

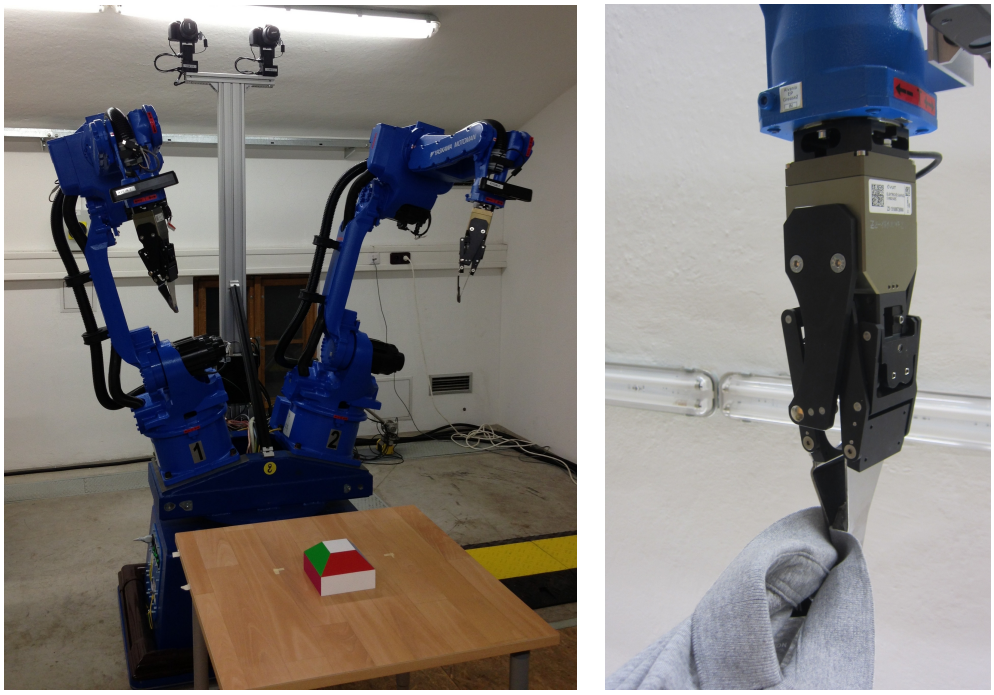
The robotic device used at KTH is the two-armed Schunk manipulator placed on a mobile platform, see Figure 4.2. Each arm has 7 degrees of freedom. A standard 2-finger parallel gripper is used as the end effector. There is a Kinect placed on the mobile platform. There is a force sensor in both wrists of manipulators.

Robotic device used at CTU

The robotic device used at CTU is called the CloPeMa testbed because it has been developed in the CloPeMa project. It consists of two industrial welding manipulators MA1400 from the Japanese manufacturer Yaskawa-Motoman. Each arm has 6 degrees of freedom. Both arms are arranged on a rotation platform r750 providing the 13th degree of freedom. There has not been force sensing in CloPeMa testbed so far. However, the six degrees of freedom torque sensors are planned between each arm and its gripper. CloPeMa testbed is shown in Figure 4.3a.

A standard two-finger parallel gripper is mounted on each arm. The special design extension, called CloPeMa hand, is mounted on the gripper. It resembles a bigger pair of tweezers. CloPeMa gripper was designed and manufactured by the CloPeMa project partner from Italian University of Genova. CloPeMa hand is specialized for clothing/garment manipulation. It also purposively extends the opening gap with respect to the parallel gripper. In a later stage, CloPeMa hand will be equipped with tactile sensors working on capacitance principle. CloPeMa hand serving as an end effector is shown in Figure 4.3b.

CloPeMa testbed is equipped by Kinect-family sensors, one on each arm. The actual



a) Manipulator. **b)** End effector.

Figure 4.3 CloPeMa testbed used at CTU.

type is Xtion. There is also a stereo-head mounted on a precise pan-tilt unit and mounted on a rotating platform. The stereo head provides high resolution color images.

4.2 ROS as a middleware, software related thoughts

ROS

ROS (Robot Operating System [51]) is used by both robotic systems at KTH in Stockholm and CTU in Prague. It serves as a unifying middleware connecting particular hardware components and allowing to write the same software running on both robotic systems.

The interface is a standard TCP/IP protocol implemented in ROS. It allows us to run independent modules controlling robot arms as well as reading from sensors and controlling them. Each ROS node can be implemented in a different programming language. The nodes (modules) can run on different computers and communicate via network. This features brings a great potential of easy extensibility to the system.

Our software tools

Python v2.7.3 [50] was chosen as our principal implementation language. There is abundance of computer vision and artificial intelligence algorithms was implemented in it. These modules are freely available. In addition, the Miller et al. model, see Chapter 7, was implemented in it. We use it as a base model in our development.

Our code was implemented and tested on the Linux, version Ubuntu Precise 12.04. It seems that almost everybody in robotics is using this version of the operating system

including KTH and CTU teams, the robots of which are used for experimenting in this diploma project..

We have chosen the "Git" for software version control connected with account at "GitHub" pages. This solution provides easy accessibility of the code to other people and also works as a backup medium. An implementation of testing platform is available at https://github.com/sindljan/clothFolding-berk.

Third party components

We have used third party libraries or components in this diploma project. The list of used libraries and their brief descriptions follows:

- opencv [46] - OpenCV is a library implemented by many researchers, originally maintained by Intel, recently taken over by Willow garage. We use the following components/functions:
  - CloneImage, CreateImage, SaveImage, LoadImage, CreateMemStorage ... for manipulation with image data.
  - NamedWindow, ShowImage, WaitKey, DestroyWindow ... these functions provide basic manipulation with the image window.
  - PolyLine, CreateMat, Set, Circle, Line ... to create a new object in the image.
  - CvBridge ... to convert images between various formats.
  - CVButton ... active objects on image window.
  - FindHomography, GetSize, FindContours, ContourArea, PointPolygonTest ... to get more information about the image.
  - WarpPerspective, GetSubRect ... for modifications of image data.
- Vector2D - the Python script containing set of function for computation and manipulation with vectors, points and segments in a 2D space. It was implemented by a group that worked on article [16] too.
- freenect [43] - The library and drivers for the Kinect and Xtion. It provides communication and integration to the ROS.

4.3 Connecting Kinect to a virtualized system

Used hardware and software in experimental part of this work (Section 10.1, Table 10.1) requires a little more effort while installating of a Kinect camera than usual. It can be found that "OpenNi" drivers are mostly recommended for Kinect in ROS. A problem was that this drivers did not work with a virtual machine. That was the reason why a software tool "freenect" was chosen. The description how to install the driver can be found on ROS wiki pages (http://www.ros.org/wiki/freenect_stack). The trouble of this driver is that it does not support the "1473" type of Kinect (the newest version of Kinect).

5 Robot Interface Layer

The main goal of this layer is to create the interface between various instances of the specific robotic device, its sensors and the "Clothing Manipulation Layer". In other words, the Clothing Manipulation Layer requires some sensory data for updating a model of the clothing. In addition, it requires tools for manipulating with clothing and mobile sensors (for example a camera placed on a robot arm). To reduce the state space we do not want to leave the Clothing Manipulation Layer do whatever it wants with the robot and sensors. Therefore we define robot affordances by limiting robot actions to a predefined set.

Here is the list of abstract functions provided by the "Robot Interface Layer":

- How to get data from sensors:
  - SetPositionOfObsvObject(position) ... setup position where observed object should be placed on the scene
  - GetImageOfObsvObject() ... return RGB image of an observed object
  - GetPointCloud() ... return point cloud that represent observed scene
- How to change a scene:
  - Push(startPoints,endPoints) ... push with robotic end effector from "startPoints" to "endPoints"
  - Grasp(graspPoint) ... provide grasp on defined "graspPoint"
  - Place(arm,goalPosition) ... move "arm" from local position to the "goalPosition" and release object that hold in its jaws
  - Fold(graspPoints,goalPoints) ... provide fold on clothing. Where "graspPoints" are grasping points and "goalPoints" are goal points.
  - UnFold(graspPoints,goalPoints) ... remove fold on clothing. Where "graspPoints" are grasping points and "goalPoints" are goal points.
  - LiftUp(points) ... lifts clothing up by defined points.

This approach is flexible. New functionalities can be added into the interface layer. They can be implemented on a real device later.

The robot keeps a list of affordable functionalities in order to bring more control to available functionalitis. The list is subset of all abstract functions defined in "Robotic Interface Layer". It means that "Clothing Manipulation Layer" would be able to use only allowed functionalities.

6 Clothing Manipulation Layer

The "Clothing Manipulation Layer" serves as a connection point between a model and a robotic device and its sensors. The principal functionality of the layer is to drive the whole process of folding, unfolding, model updating (according to the actual state of an observed object), data recording and so on. In other words, this layer drives whole process of manipulation with clothing.

Implementation

We have implemented the folding process that starts with capturing the depth map of the manipulated piece of garment by Kinect. The creation of a fold line is followed by the folded model creation and finished by actual folding. The folding process is depicted on the diagram in Figure 6.1. The "Clothing Manipulation Layer" uses "Robot Interface Layer" to get all input data for model tuning and for manipulation with clothing as depicted in Figure 3.1.

The list below describes what each box does:

- *Initialization* ... Prepare initial clothing model for the folding process.
  - "Get initial image of clothing" ... Call "Robot Interface Layer" that provides the last image taken by Kinect. There is a possibility to use the image from a file instead an image from Kinect device.
  - "Make a top view from image" ... Removes the perspective distortion caused seeing the scene from an oblique angle. Homography is explored to undo unwanted distortions.
  - "Get the initial clothing model" ... Load a clothing model from the model storage.
  - "Fit the model to the captured image" ... Take the initial model and the image from Kinect. Use the fitting algorithm [16] to update the model in order to minimize the difference between the model contour and the contour of an observed object.
- *Folding process* ... Proceed with the preprogrammed folding on the clothing.
  - "Propose a fold line" ... Propose a fold line on an actual state of the model. A description of how folds are defined can be found in Section 6.
  - "Create folded model" ... This part takes the actual state of the model, which should be the initial fitted model or the fitted folded model (from the previous step), and creates a folded model according to proposed folding line from it.
  - "Execute the fold line on the real clothing" ... This function creates a set of actions for the robotic device that transforms clothing to a folded state according to the proposed fold line. A more detailed description is Section 6.
  - "Fit the folded model to the captured image" ... Take the folded model and fit it onto the observed object contour. It uses the updated fitting algorithm from the article [16]. Our improvements of the algorithm are described in section 7.2.

An implementation of a modified basic clothing model and testing platform can be downloaded from [28]

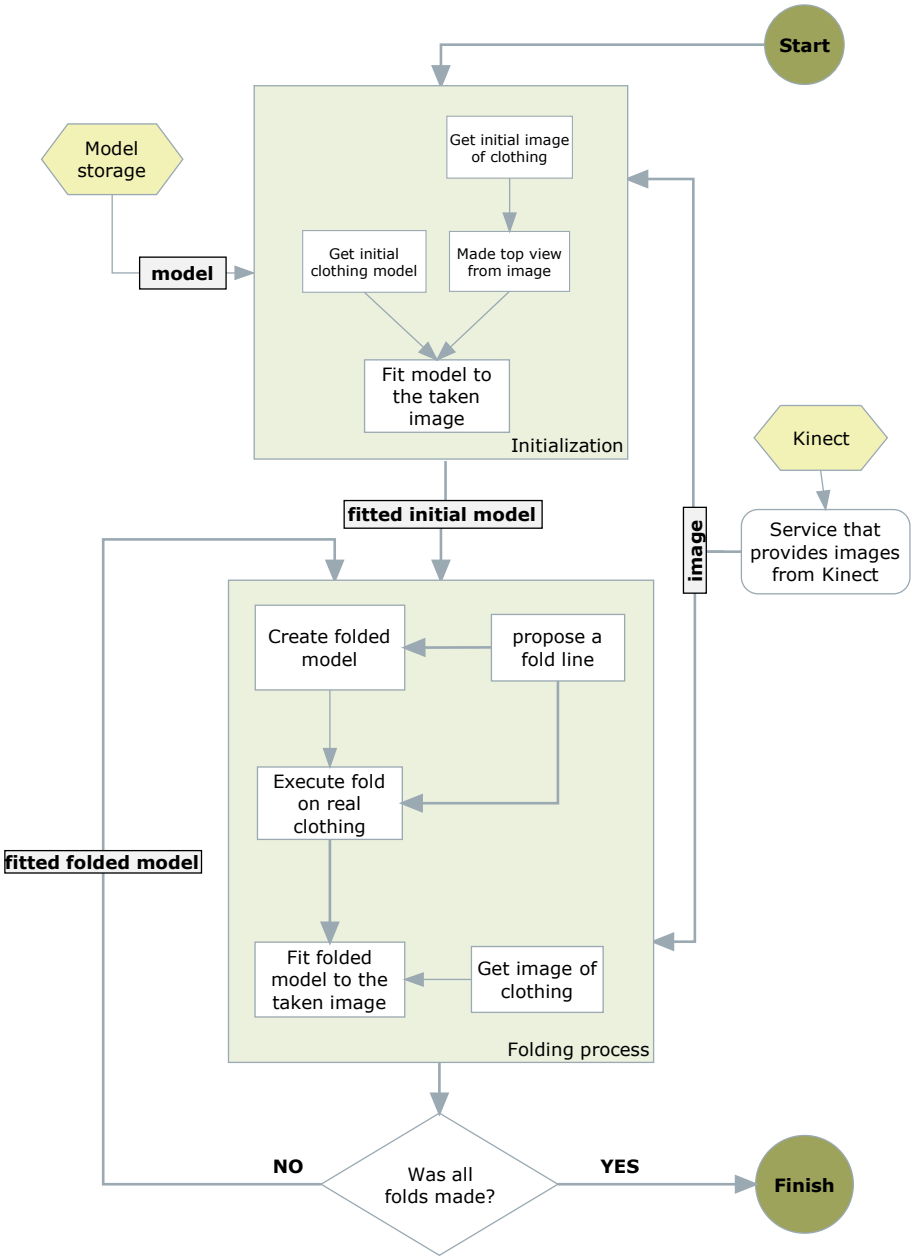


Figure 6.1 Architecture of the folding system.

Folding execution

This part of a script realizes the fold on a specific robotic device. The script takes model with the fold and the model without the fold. It creates a subset of model vertices.

It selects vertices that change their position after folding. From the subset, it selects two vertices which are the farthest from the folding line. It uses the simple Euclidean distance between a point and a line in 2D. These vertices are marked as grasp points. Their positions should be sent to the real robot or to the human who simulates robot moves. After the robot grasps these points, the script will sent the robot new point positions by simply mirroring previous points around the folding line. It should be also possible to make a little bit more complex movements and to insert some curve between the grasping points and their final position.

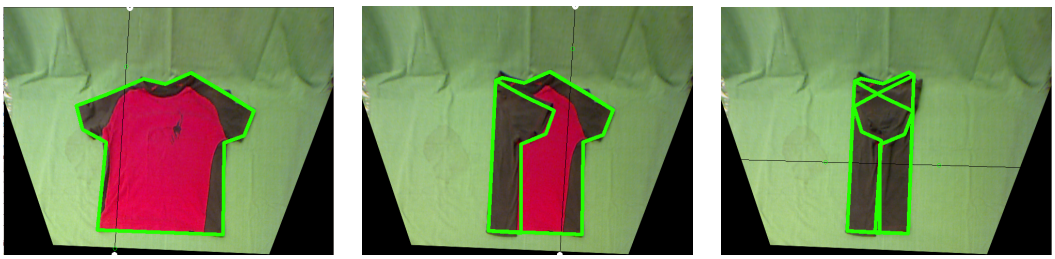
The fold line definition

The main advantage of having a model is that we can ask more abstract questions about an observed object. We explore this ability for creating folding lines. The folding lines are hard programmed.

The explanation of how it was made was provided for the T-shirt folding example. The script creates three folding lines on an unfolded T-shirt as follows:

- The first fold is defined by two points. The first point is the point between the collar and the left shoulder. The second point is defined as the left corner of the lower hem shifted by a vector defined by the first point and the left shoulder point.
- The second fold is defined in the same way as the first one but it was applied on the right side of the T-shirt.
- The last fold is also defined by two points. The points are the middle points between the left shoulder and the left lower hem and the right shoulder and the right lower hem.

Examples of fold lines are depicted in Figure 6.2.



a) 1st fold line. **b)** 2nd fold line. **c)** 3rd fold line.

Figure 6.2 An example of fold lines.

7 Parametrized shape model for pieces of clothing

7.1 Polygonal clothing model and the related fitting process

This section describes the clothing model presented in the article Miller et al. [16] very shortly. The clothing model was implemented by the author of the article and we have used their implementation as the starting point of our work.

The clothing model consists of two main parts. The first part is the clothing skeletal model. The skeletal model represents the shape model of a particular clothing category (shirt, pans, etc.). The example of such a skeletal model for the T-shirt is depicted in Figure 7.1. The skeletal model is created manually by the human user.

Figure 7.1 An example of a T-shirt skeletal model.

The second part of the model is given by the silhouette of the observed object, of a piece of garment, which is more or less stretched on a planar surface. The silhouette is an outer boundary of the object, which is provided by the object vs. background segmentation. We simplified the segmentation task in our experiments by having an uniform green background, recall Figure 4.1.

The silhouette is approximated by straight line segments, which is common in computer graphics under the name polygonal representation. It is obtained from the segmented object boundary by transforming the raster image into a 2-dimensional vector representation. This transformation is called vectorization or conversion from raster to vector representation in literature usually. The outcome is a polygonal representation of a 2D region given representing the object boundary, called shortly a polygon. We

call the second part of the model the "polygonal model" and use it also as the name for clothing modeling as described in the Miller et al. article [16].

The polygonal model, allows defining semantic restrictions applied to geometrical parts of the skeletal model. The restrictions constrain modification of the model geometry during the phase, in which we are trying fit the model to the observed object. The restrictions are specific for each category of the piece of garment. As an example of such a restrictions, we can say that the width of a T-shirt sleeves cannot be bigger than the length of the bottom hem of the same T-shirt.

The example of a polygonal model for a T-shirt is shown in Figure 10.2c.

How is the similarity between the polygonal model and the observed object assured? The example of the polygon stemming from the object silhouette is shown in Figure 7.3. The end points of the model skeletal are related to the polygonal model.

If both polygonally approximated boundaries of the object and the model exist then we define the error function E following the article [16] as:

$$E(P) = \alpha\, \overline{d}\,(M_c \to O_c) + (1-\alpha)\, \overline{d}(O_c \to M_c)\,, \tag{7.1}$$

where M_c are model contours, O_c are object contours, parameter α adjusts how the model fits to the contour. Parameter α was set to 0.5 experimentally. The term $\overline{d}(A \to B)$ is the average nearest-neighbor distance from set A to set B defined as:

$$\overline{d}(A \to B) = \frac{1}{|A|} \sum_{a \in A} (\arg\max_{b \in B} ||b - a||)^2\,. \tag{7.2}$$

Authors of the article [16] are using a three-step numerical optimization that creates variation of a skeletal model to minimize the energy function. The optimization process computes a new position of one point from the skeletal model as $p_{new} = p + \delta$ and then compute an error function again. If the error function is smaller than the previous one then the optimization algorithm keeps the new point and updates δ as $\delta_{new} = \epsilon\,\delta$, where ϵ is the exploration factor. Otherwise it computes the new delta as $\delta_{new} = 0.5\,\delta$ for this parameter and keeps the old point position. This procedure is performed for each point of the skeletal model. This "round" is called the "iteration".

The optimization phases follow-up to each other phases and are as follows:

- The model orientation phase – allows the variation in rotation, translation and scale of the whole model;
- The model symmetric phase – allows the variation in rotation, translation, scale of each skeletal model point or segment but takes care about the model symmetry;
- The model asymmetric phase – allows variation in rotation, translation, scale and deformation of each of the skeletal model point or segments.

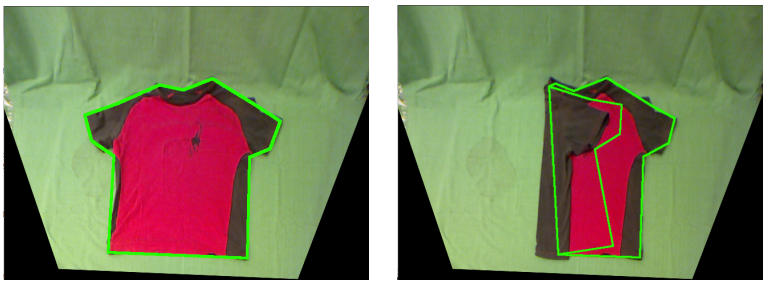
Each of the phases can be bypassed. The solution with the smallest error was chosen at the end of the optimization because the actual model state corresponds best to the current state of the observed object. It is worth mentioning that the optimization of the whole model is performed only before the folding process begins. Only the folding line is optimized using same optimization process after folding starts.

The use of the observed object silhouette unfortunately restricts the model applicability. The model can be used only when a clothing is crudely spread on a flat surface.

The source code implementing the model can be found at the GitHub web page [41]. A more detailed description of the model and other related processes can be found in the mentioned article.

7.2 Our modification of the basic fitting process

The fitting process works fine for the initial fitting as illustrated in Figure 10.2c. Some problems occurred when we tried to perform fitting of a folded model. An example of such a case is shown in Figure 10.2d. The possible reason why this happens can be:



a) Initial fitting.

b) Fitting of a model with a single fold.

Figure 7.2 The problem with fitting of the folded model.

- An incorrect use of the fitting process and connected clothing models. The cause might be a poor documentation of the used code related to the article [16].
- A missing part of the observed object silhouette caused by a fold. When a piece of clothing is folded, the script is not able to extract the silhouette that appears inside of the clothing, see Figure 7.3, in which the blue color represents the silhouette. Nevertheless, the polygon induced by a folded skeletal model of an object model provides the missing part of the silhouette. This causes serious troubles during the fitting process.

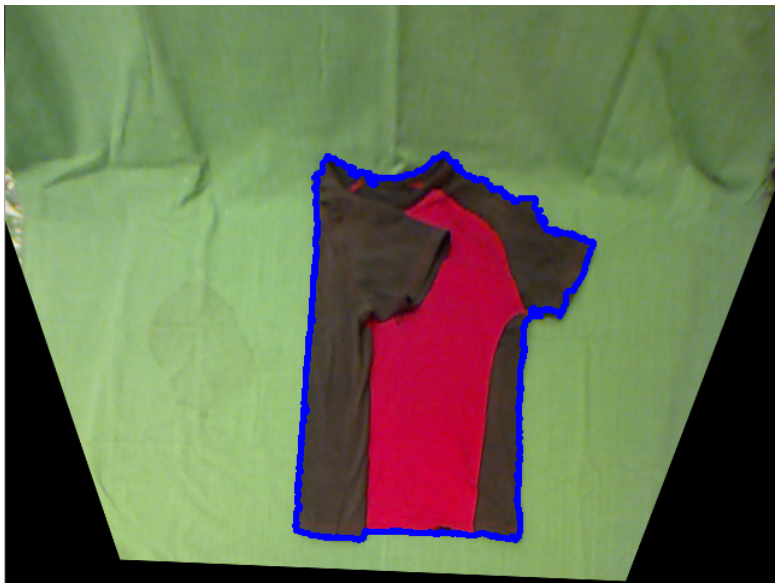


Figure 7.3 Contour of a T-shirt with a single fold.

We decided to solve this problem the following way. The whole fitting process is driven by the error function given by Equation (7.1), which is computed from a difference between the object contour and the model contour. We changed the way how the error function is computed for the folded models. Creation of the "silhouette" of a folding

line allows that the error function is computed in the same manner as before. The only one difference is that the model contour is supplied by the folding line now. In other words, we simplified the fitting of the object contour to the whole folded model as the fitting of the object contour to the folding line. This approach is faster and works better and it does not affect the initial fitting, which is the most complicated and the most important.

We changed the metric used in distance computation, Equation (7.2) from $\sum x^2$ to $\sum x^4$ to make the fitting process more sensitive to the difference between the model contour and the object contour.

8 Manifold extension of shape models

We realized that the polygonal model described in Chapter 7 was not able to cope with situations, in which a piece of clothing is not nicely spread on a surface. The example of such a clothing is in Figure 8.1. The wrong performance of the model stems from the fact that the boundary (silhouette) of an observed object is explored for model fitting. On the other hand, the parameterized shape model has some advantages, which we wanted to keep. This was the main reason why we were looking for a solution that would extend the polygonal model by Miller et al. [16] to be able to manipulate with more general clothing configurations. The proposal described in the following Sections 8.1 is one possible solution.

Figure 8.1 Clothing on a pile.

8.1 Fundamental idea

The main idea comes from the fact that the clothing in the ideally spread configuration is a smooth planar object. When it is crumpled up, it creates a surface, an $\mathbb{R}^2$ subspace embedded in a $\mathbb{R}^3$ space. The clothing material properties induce that crumpled clothing is locally smooth. This distinguishes a clothing from, e.g. a paper or a thin metal sheet. The latter two would create a 2D subspace as well. However, they would not necessarily be smooth locally.

This local smoothness property allows us to use a parameterized shape model introduced in Chapter 7 even for a crumpled clothing. If we change the layout, and it is not important how many folds we create at this moment, bumps or other deformations occur. Under the assumption that the clothing is not transparent, the visible part of the examined clothing remains a smooth and thin $\mathbb{R}^2$ subspace in a $\mathbb{R}^3$ space in many

possible configurations. This fundamental idea brought us to the notion of a manifold, which is a known concept in mathematics. The needed concepts from the manifold theory are given in the following section.

8.2 Manifold, theory

Manifold is a powerful mathematical concept that helps describing objects that can be expressed in a less complex and useful way by embedding into less dimensional space. A good illustration of this phenomenon is one of the most basic manifolds – lines embedded in $\mathbb{R}^2$. It can be shown that we need only one variable to position a point on a straight line. We would need n variables to express line in $\mathbb{R}^n$ space. Manifold shows that other variables are dependent on the controlling variable, or said differently, expressed as functions performing mapping from $\mathbb{R}$ space to $\mathbb{R}^n$ space. This function is called the mapping function (in our text denoted ϕ). In our our special case of a straight line in $\mathbb{R}^2$, the mapping function $\phi = (x, k\,x + q)$. The same idea can be used for the circle that, from a topological point of view, is the same object as a line only the ϕ function is different, $\phi(x) = (x, \sqrt{1 - x^2})$ while considering the mapping of the unit circle to $\mathbb{R}^2$ space.

In a general case the mapping function $\phi(x)$ writes as

$$\phi(x) = (x, a_1 x, a_2 x, \ldots, a_n x)\,. \tag{8.1}$$

One example, which is used in our common lives and we do not think of it usually in manifold terminology, is the mapping of the Earth ($\approx$ sphere) to a plane (usual paper map).

A manifold concept brings a complexity reduction. It provides a transfer of n-dimensional object representation (usually non-Euclidean) to m-dimensional one, where $m \ll n$. More than one definition of a manifold can be found in literature [31], [18],[45]. We provide one definition, which suits our purposes:

Definition 1 *A topological space M is **locally Euclidean** of dimension n if every point p in M has a neighborhood U such that there is a homeomorphism ϕ from U onto an open subset of $\mathbb{R}^n$.*

Definition 2 *A **topological manifold** is a Hausdorff, second countable, locally Euclidean space. It is said to be of dimension n if it is locally Euclidean of dimension n.*

We will need two more concepts – the chart and the atlas. They are defined as follows.

Definition 3 *A pair $(U, \phi\colon U \to \mathbb{R}^n)$ is called **chart**.*

Definition 4 *An **atlas** on a locally Euclidean space M is a collection $\mathfrak{U} = (U_\alpha, \phi_\alpha)$ of pairwise C^∞ – compatible charts that cover M, i.e., such that $M = \bigcup_\alpha U_\alpha$.*

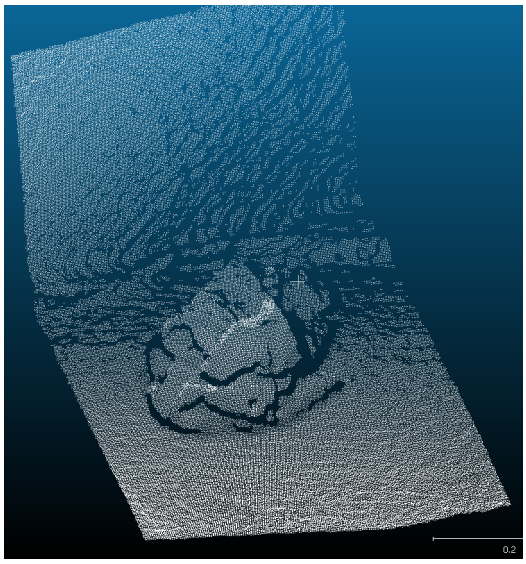
All of the definitions in this subsection are citations of the Manifold chapter from [31]. A really nice example of manifold usage in computer vision (face recognition), that explains a lot, can be found in the paper [26].

8.3 Manifold used for clothing modeling

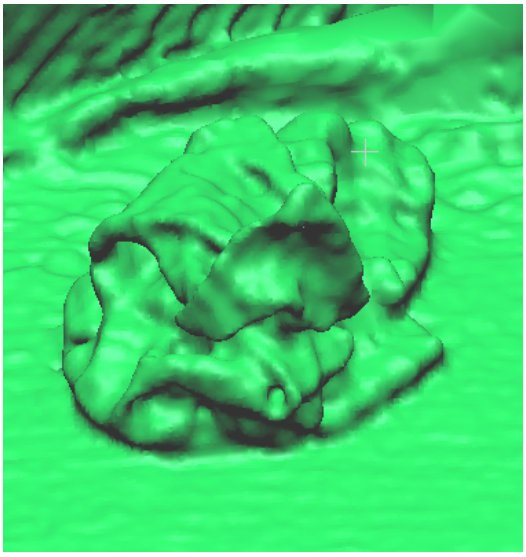
Let us apply the manifold concept to our clothing modeling task. The extension of the basic clothing model will be demonstrated on the towel example. However, it can be used for other pieces of garment too. The word "object" will be used to express briefly the term observed object and the word "model" as an abbreviation for the polygonal $\mathbb{R}^2$ model presented in Chapter 7.

Let manifold M be a visible surface, a $\mathbb{R}^2$ subspace embedded in a $\mathbb{R}^3$ space of 3D coordinates provided by the range finder. The observable part of M is a nontransparent surface visible by vision sensors. The example of such a manifold is depicted in Figure 8.2b. How is it obtained? Simply, it is enough to observe the object by Kinect that gives us RGBD data. RGB stands for red, green, blue of the color image and D represents the distance from the sensor. The point cloud is created from RGBD data, see Figure 8.2a. We used pcl library [47] for doing so. The point cloud provides a semi-sparse representation. Each point in it gives the distance from the sensor and RGB inform about the visual appearance of the point.

Having the point cloud, a triangular mesh is created by an instance of Delunay triangulation. We used the tool CloudCompare [40] for this purpose. The towel represented as a mesh is illustrated in Figure 8.2b. The mesh representaion of the towel was created by the CloudCompare tool with the 3D mesh Poisson reconstruction addon. The tool is part of a standard 3rd party CloudCompare tool set. More about 3D mesh Poisson reconstruction can be found in [48]. The whole process is illustrated in Figure 8.3. The top left image shows the color image of the scene with a crumpled towel. The bottom image visualizes the point cloud captured by the Kinect sensor. The top right image shows the mesh representation of the scene surface, in which the towel surface is higher than the plane representing the background. Pink arrows show the processing order.



a) Point cloud representation of the towel.



b) The mesh representation of the towel.

Figure 8.2 Example of a manifold created from data captured by the Kinect sensor.

We will use atlas A and chart (U_i, ϕ_i) concepts for clothing modeling. We know from Section 8.2 that the atlas is just a union of charts, Figure 8.5b. The chart is an Euclidean representation of manifolds, Figure 8.5a. If we thing about our polygonal models, e.g. a towel polygonal model in Figure 8.4 then it is enough to choose a proper

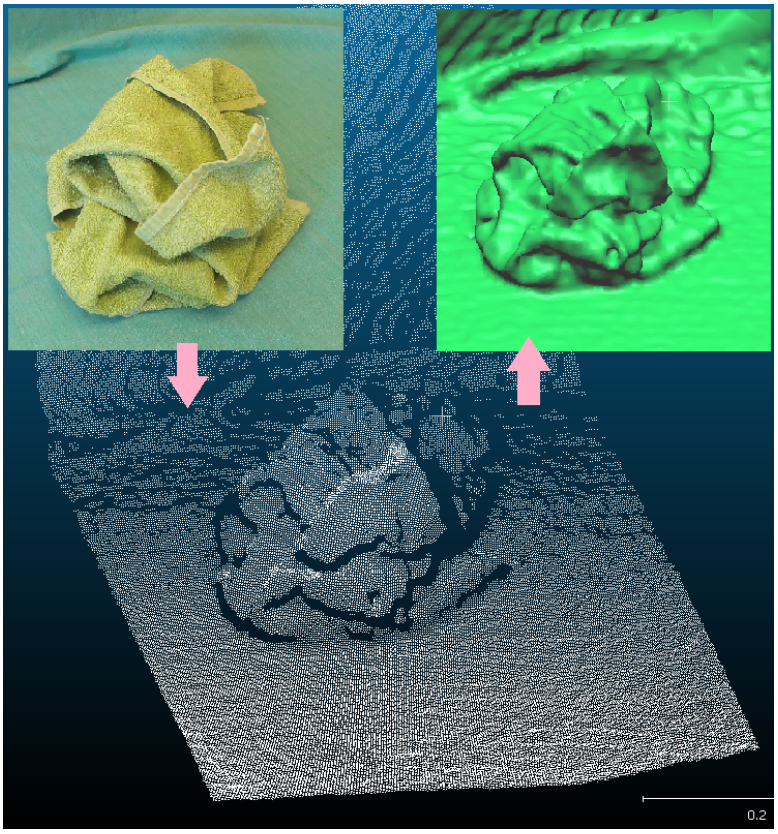


Figure 8.3 Retrieval of a polygonal model.

coverage of the model. A description of the coverage will be given in Section 8.4. This is the representation we have been using in our work.

The introduction of a manifold representation enables using the manifold as a basis for a more general model expressing a piece of garment. This would need some other way how to generalize the concept of the skeletal model, expressing semantics of the piece of garment and a procedure for updating/learning the model from observations. We have not developed these ideas further in this diploma thesis.

A global view of whole problem is depicted in Figure 8.6. The atlas, its coverage and manifold are known. We have to find a mapping function (homeomorphism) ϕ from the manifold to the polygonal model. This is the same problem as to find the graphs. A good approach to solve this problem seems to be given by Freedman article [10]. The article provides technique how to reconstruct manifold from unorganized points. We can use this technique for finding manifold in each of our subregion.

The remaining task is to find correspondences between manifold subspaces U_i and their proper positions in the model. To solve this task we explore object features that are detectable by our system. In the towel example, the detectable features are edges, corners and a towel hanger illustrated in Figure 8.4. LUC stands for left upper corner, RUC for the right upper corner, LLC for the left lower corner, RLC for the right lower corner and H denotes the towel hanger. Having edges, corners and the towel hanger, relations between features are sought. Some features are linked to another features, e.g. one corner is linked to another corner by an edge. In our case, LUC and LLC are linked by the edge LE (left edge), LLC and RLC is linked by BE (bottom edge), RLC is linked to RUC by RE (right edge), RUC is linked to LUC by TE (top edge). The

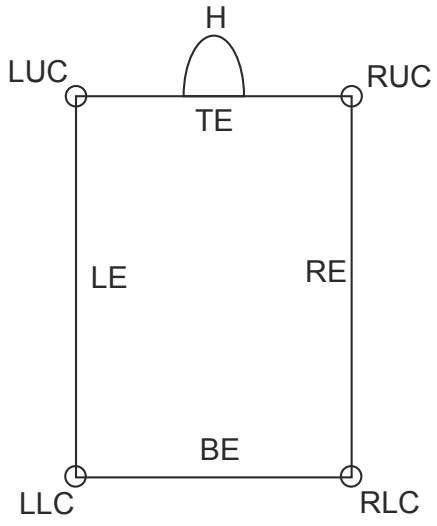


Figure 8.4 The polygonal model of a towel.

towel hanger H is linked to the edge TE. The information about the position of the particular feature on the represented object constrains a possible interpretation, i.e. a choice of the correct chart from the model, see Figure 8.5b.

If all of above mentioned has been done it is easy to describe extended $\mathbb{R}^2$ polygonal model of a piece of garment. Let new extended model be an atlas created from coverage of previous $\mathbb{R}^2$ polygonal model and proper ϕ_i^{-1} functions. Actually it is enough to have one basic polygonal model and set of mapping functions ϕ_i^{-1}.

The knowledge of graphs (U_i, ϕ_i) helps us to perform a back projection of a detected feature on the object (p) to the model (np) even the object is in the $\mathbb{R}^3$ space,

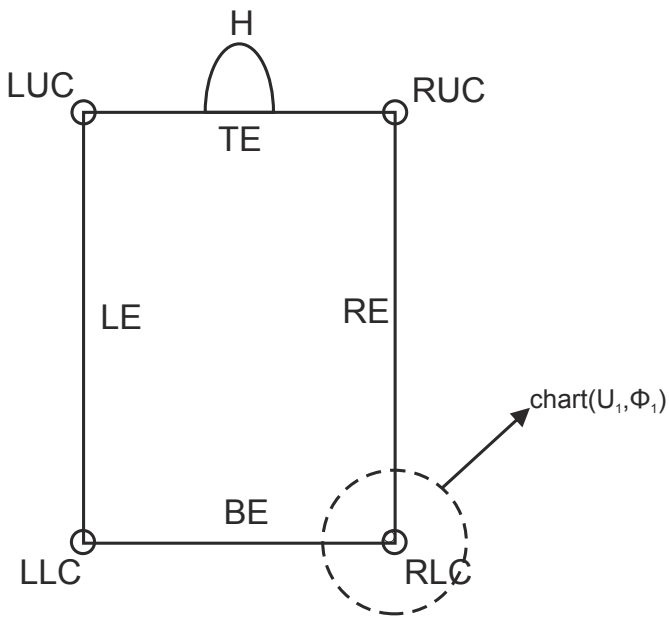
$$np = \phi(p) \tag{8.2}$$

The set of extended models is knowledge, which is to be provided by a learning process. The concept is illustrated in Figure 8.7. The root of the tree is the nicely spread state, which is equivalent to the basic polygonal model. Children nodes contain extended models emerging form some action that help us to get to these states. It means that leaves of the tree contains starting configurations. The arrows depict the order of involved processes.

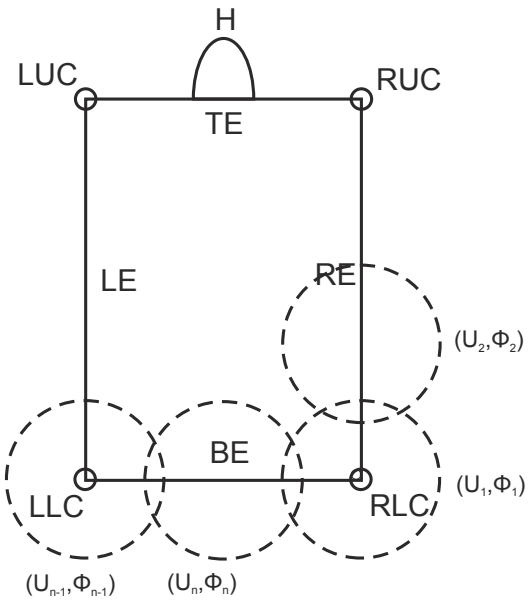
An approach mentioned above expects that the atlas exists and it is known. It comes from the fact that we wanted to extend the original 2D polygonal model. Another approach is that we do not know an atlas and we wanted to create it by some algorithm. The LLE algorithm presented in the article [24] might be helpful. The LLE is a scheme that is trying to find a low-dimensionless global coordinates when data lies on a manifold.

8.4 Basic clothing model coverage – the atlas

Two approaches how to cover the basic clothing model will be discussed. The first approach covers only features, which we are able to distinguish easily, i.e. those which



a) Chart.



b) Atlas.

Figure 8.5 Manifold usage in the towel example.

are on the clothing border. An example of such a covering is depicted in Figure 8.8a. The principal advantage of this approach is its simplicity. We are usually not interested about things we do not know or we can not do anything about them such as wrinkles in the middle of a clothing.

The second approach deals with other features the first one does not cope with. The second approach covers uniformly the basic model as depicted on Figure 8.8. The advantage of this coverage is that it cares about features that are "inside" of the clothing borders. The main disadvantage according to our opinion is that the uncertainty of

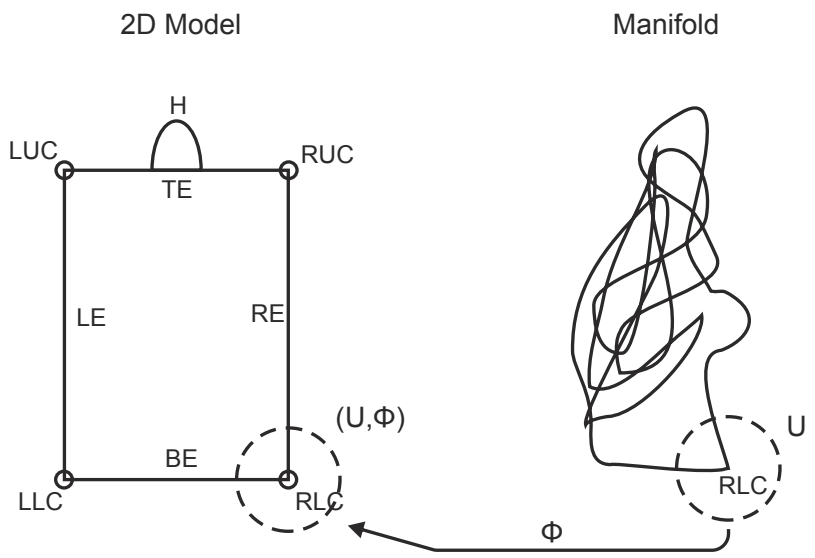


Figure 8.6 A global view.

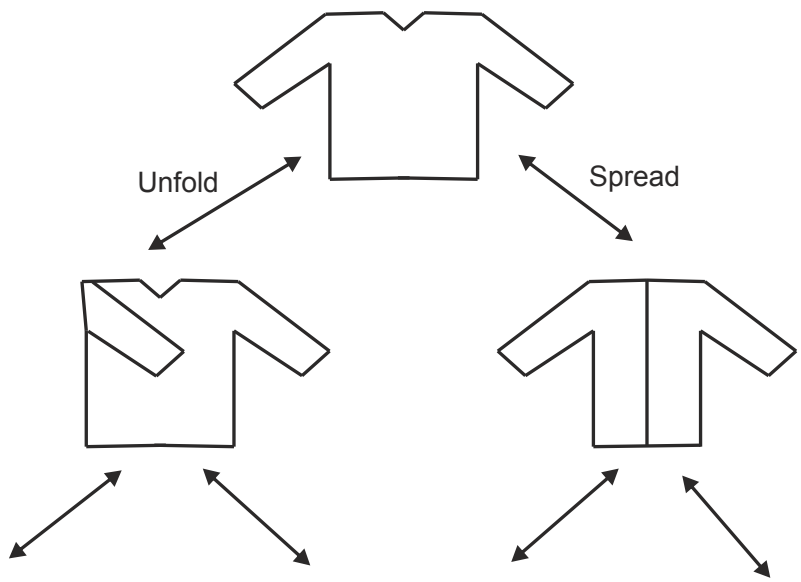


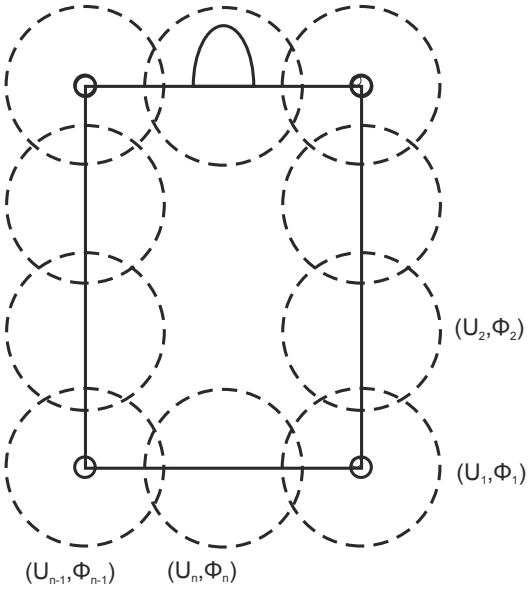
Figure 8.7 Graph for the decision making.

mapping inside of clothing is big. In other words we do not have enough known or distinguishable point that works like an key points inside of a clothing.

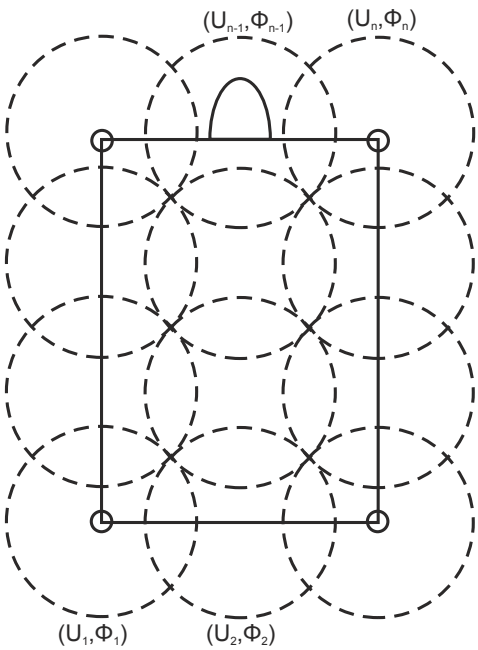
In both approaches, the space is covered by circle elements generated by function:

$$[x - (Sp_x + tu_x)]^2 + [y - (Sp_y + tu_y)]^2 - \frac{|\overrightarrow{u}|}{d} = 0\,, \tag{8.3}$$

where $\overrightarrow{u} = Sp - Ep$, the parametrization variable $t = 0 : \frac{1}{d} : 1$, where d determines the space division, Sp and Ep are starting and ending point of one polygonal element that determines the basic model. It is a normal parametrization of a circle in $\mathbb{R}^2$ space.



a) Feature coverage.



b) Uniform coverage.

Figure 8.8 Atlas coverage.

There is a small issue in all read articles (Hong et al. [22], Dong et al. [15]), which are using manifolds. We did not find if they have a priori information about atlases, which we have. They were usually using algorithms that create mapping functions and atlases automatically such as LLE, ISOMAPS, RME. Maybe this should be called in some another way.

8.5 Similarity metric

We need to tackle the issue how to measure the similarity between data from observations and a chosen clothing model. The idea is that we create a pre-learned set of atlases A^i and measured manifolds M. Let C be a coverage of the manifold described in Section 8.4 and C_j be one element from the coverage. We expect that we know correspondence between graphs of an atlas and the element. Then the score s_i of one element C_j is

$$s_i = \sum_{j=1}^{n} A_j^i - \phi_i(C_j) \,, \tag{8.4}$$

where $\phi_i \colon M \to A_i$ is the mapping function from the manifold to the graph. Next, we are looking for mapping that gives us best score

$$s_{i_{best}} = \arg\min_{i \in I}(s_i) \,, \tag{8.5}$$

where I is a set of all known atlases. Finally, we have to create a threshold that decides if the obtained score A^i_{best} is good enough or if it is necessary to create a new element A^i.

9 Implementation

9.1 Introductory notes to implementation

We started our implementation/experimental work by duplicating University of California, Berkeley results as described in Miller et al. [16]. Their code was available too [41]. The core of the implementation is written in Python v2.7.3 [50]. Our description of their implementation is provided in the next Section 9.2. We will refer to the original implementation as to Berkeley implementation (of the polygonal model).

The Berkeley implementation uses several third party libraries written in different languages, mainly in C_{++}. Functions from third party libraries are called from Python code. It can be done easily as described in `http://docs.python.org/2/extending/extending.html`. We listed the third party components with their short characterization already in Section 4.2.

After reimplementing and testing the Berkeley code, we observed that obtained results were worse than those reported in the article [16]. The first problem was much slower speed of fitting process. Their reported was roughly 30 seconds. Our best fitting took 2.5 minutes. The second and more serious problem was that the code failed in fitting observation to the folded model.

We chose Python as our principal implementation language too. Our extension to the Berkeley implementation followed the structure of the original code. Both problems were solved. The description of our solution can be found in coming Section 9.3.

The diploma project assignment requires that developed representations of clothing are tested on two robotic manipulators, at KTH Stockholm and at CTU Prague. This was not possible, though. During the Erasmus stay at KTH Stockholm from beginning of January 2013 to end of April 2013, their robot was not available. A few days between the return from Stockholm and the deadline for diploma thesis submission did not allow to perform testing on the CloPeMa project dual-arm manipulator.

Actually, we realized that testing on a real robot is not necessary while designing, implementing and testing models representing a piece of clothing. We created a simple mockup for sensing the scene with piece of clothing using Kinect sensor and replaced robotic manipulations by movements performed by a human. We also simplified the segmentation of a piece of clothing from background. This is done by using uniform green background. This green color does not appear in our objects of interest. Our mockup was already shown in Section 4.1.

Having said that we replaced robots by the mockup, we still prepared the connection to real robots. The connection to the specific hardware is maintained by the Robot Specification Layer and by the Robot Interface Layer. We explored ROS. These interfaces allow an easy extension if a new piece of hardware has to be connected to the system. We tested this functionality by writing the ROS service, which connects Kinect sensor to the system. This is described in Section 9.4.

Finally, the description of the testing platform described in Chapters 4, 5, and 6. The implementation is described in Section 9.5.

All the implemented scripts can be found in Git repository. The link to repository with our implementation can be found in [28]

9.2 Berkeley implementation and its reimplementaion

The Berkley implementation can be downloaded from Git repository, which the Miller et al. group stored on GitHub server. It can be downloaded from https://github.com/rll/visual_feedback. The implementation contains the following tools:

- model maker – the graphical tool for creating basic models (./visual_feedback/clothing_models/scripts/model_maker.py),
- fold maker – graphical tool for creating folded models (./visual_feedback/clothing_models/scripts/fold_maker.py),
- shape fitting – provides fitting of basic (or folded) model to an input image using a black box optimization of the skeletal model (./visual_feedback/visual_feedback_utils/src/visual_feedback_utils/shape_fitting.py),
- thresholding – provides forward/backward segmentation (./visual_feedback/visual_feedback_utils/src/visual_feedback_utils/thresholding.py),
- various basic models (./visual_feedback/visual_feedback_utils/models/),
- third party component FLANN (Fast Library for Approximate Nearest Neighbors) [9], which performs the fast approximate nearest neighbor search [17].

After we downloaded the startup model it did not work. There were two reasons for it. The first reason was the fact that the code was significantly modified because it was newly used for a different task – socks folding. The second reason was that the implementation used a never version of the OpenCV library. Both problems were solved after an email exchange with the author S. Miller from the University of California, Berkeley. He uploaded the original code to the repository. I recommend new users of the Berkeley implementation to start from this working version.

The most important class is the "Model" class that represent the most general model of clothing and can be found in (./visual_feedback/clothing_models/src/clothing_models/-Models.py). Several classes are inherited from the Model class, which are specialized for various types of clothing. Modifications caused by folding can be found in the same file.

The second most important class of the Berkeley implementation is the ShapeFitter class hidden in the shape_fitting.py script. While initializing the class, the user has an opportunity to set up parameters of the Black Box optimizer and of optimization phases. Both items are described in the article [16]. The most important method is "fit", which takes as the input the unfit basic (or folded) model as well as the boundary of an observed object. It returns the updated model that best matches to the provided boundary.

9.3 Our extensions to Berkeley implementation

We described the idea of our modification of the Berkley implementation in Section 7.2. This modifications changed files Models.py, shape_fitting.py and Vector2D.py. There is also a change in the thresholding script. We modified it to work with our background. All changed files can be found in the repository [28] in a folder "modified_components". The modifications are commented in the code. If a user likes to use our modified approach, she/he should override the original files from the Berkeley implementation to make it work.

9.4 ROS service connecting Kinect sensor

The connection between the Kinect sensor and other parts of the system was solved as a standard ROS service. It is implemented in the file /scripts/ImageReaderService.py. This service reads data published by the Kinect node periodically and keeps the last read image in the memory. If another part of the program asks for the image from the Kinect sensor then this service returns the last grabbed image. This approach requires to have the definition of the service message content. The definition can be found in the file /srv/GetImage.srv. We also implemented a small script that allows us to save images from the Kinect sensor to the disk for future use. The script is called ImagePicker.py and can be found in folder ./scripts/.

9.5 Testing platform

Software related to our testing platform is implemented in the file FoldingProcess.py, RobInt.py and HumanManipulator.py. The files can be found in the folder ./scripts/. A FoldingProcess.py script passes through the whole folding process. It starts with the unfolded piece of clothing, does the initial fitting, which provides folds on clothing, and finishes with the folded piece of clothing. This script uses all above mentioned scripts and offers also logging functions. The logged information is used to measure the computational time and other assessments in our experiments. This script offers two possible ways how to get input images. The first way takes images directly from the Kinect sensor. The second one take images stored in a disk. RobInt.py script is an implementation of the Robot Interface Layer. It is an abstract class that works like a parent for classes that specifies concrete robotic device. HumanManipulator.py script contains implementation of a RobInt class for robotic manipulator substitute by human.

10 Experimental results

The experimental part verifies the functionality of the Berkley implementation (Section 10.2) as well as our modification to it (Section 10.3). The chapter also documents experiments with setting of a fitting algorithm.

A significant part of experiments was dedicated to increase the speed of the fitting algorithm of our modified clothing model (Section 10.4) and testing its robustness (Section 10.5). We expected that speed up is possible if the parameters of the fitting algorithm would are properly set up.

We expected testing our results on a robotic manipulator in our experiments but they were not available in the right moment. All tests were performed manually on the experimental mockup, Subsection 4.1. The mockup contains the Kinect sensor, the background, a piece of cloth of a uniform green color to simplify the segmentation of the object (the piece of clothing) from the background.

The human performed the robobotic maniuplation job in our experiments. The human executed folds of piece of clothing, which were the outcome of our folding algorithm. I was the person who conducted the experiments.

We believe that lack of tests with real robots does not undermine results of the diploma project. To enable a human as an actor in the experiments, a waiting routine was added synchronize with human action. The visualization tool was added too. The description of both hardware and software used in our experiments was described in Section 10.1.

10.1 Description experiment settings

Hardware/software settings of the mockup were same in all our experiments. This section describes the setting, which should provide the needed information enabling another person repeating the experiments and comparing outcomes of individual experiments.

The speed improvements of the fitting algorithm and the polygonal model representing the piece of clothing depend very much o the used hardware and software. We provide a list of it in Table 10.1 below.

Table 10.1 This table listed used hardware and software

Used computer	Lenovo x200 Tablet, Intel(R) Core(TM)2 Duo CPU L9400, 1.87GHz, 4 GB RAM
Parent OS (pOS)	Windows 7 Professional, 32 bits
Virtual OS (vOS)	Ubuntu 12.04 Precise
Virtuallization tool	VMware 7.0.0, 1 GB RAM
HW unifying OS	ROS Fuerte 1.8.10
Kinect	XBOX 360 Kinect model 1414
Kinect drivers	pOS: Microsoft KinectSDK-v1.6, vOS: freenect_stack (can be found on ros wiki pages)

Figure 10.1 shows the arrangement of the testing scene, which we call a mockup. Its

simplicity have been an advantage because we could establish it easily in two locations, Stockholm and Praha. It also simply decouples the clothing representation task from a specific robotic hardware. Kinect sensor are cheap and abundant now. They are easily accessible to anyone.

Figure 10.1 Arrangement of an experiment.

10.2 Parametrized skeletal model testing

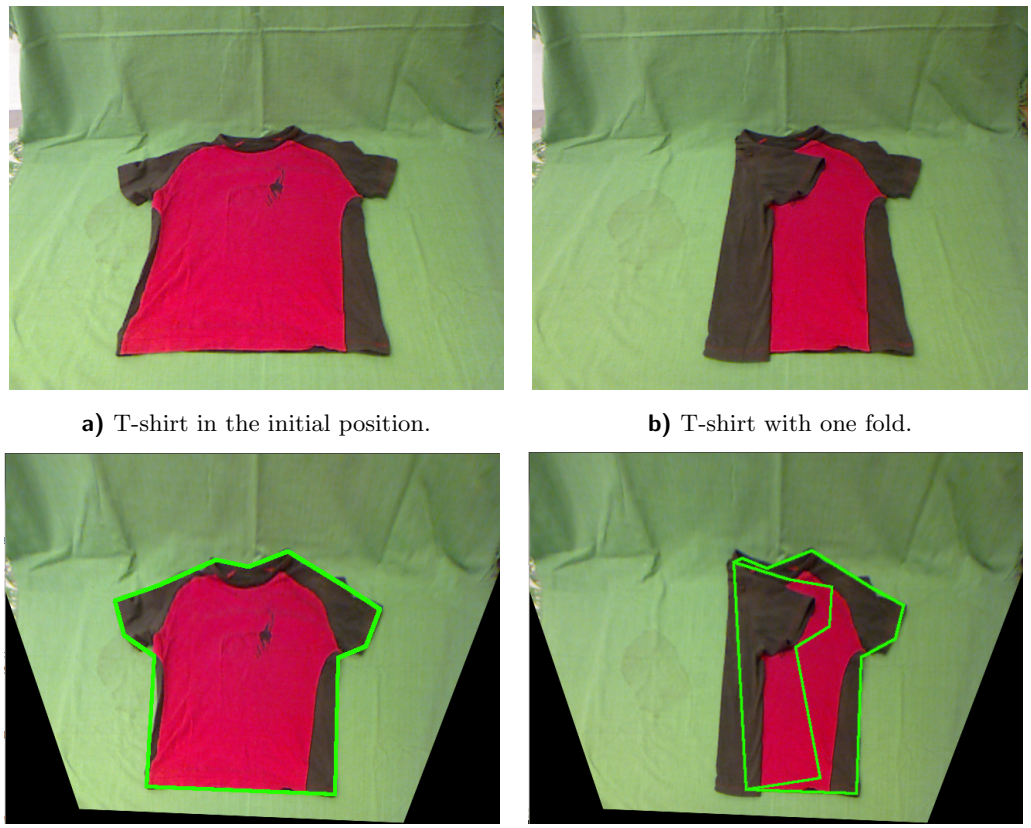
We spent a few weeks on reimplementing of the Berkeley initial polygonal model because their code did not work as expected. Our aim was to test if this state-of-the art method works in the same way as explicated in the article [16]. We used the parameterized model of the T-shirt used by the Berkeley group. We tried to replicate their results. Our expectations were following:

- The fitting algorithm should be able to fit properly the unfolded T-shirt in a similar way it fits a folded piece of cloth.
- The initial fitting should take 2.5 minutes at maximum.
- Fitting of the folded model should be much faster than the initial fitting.

Our expectations follow results provided by the article [16]. The input images used in our experiments are depicted in Figure 10.2.

Figures 10.2c and 10.2d show fitting results. It can be seen from the fitting results that the initial fitting worked correctly. The problem occurs when a fitting method tried to fit a folded model to a folded T-shirt. We created a model of a T-shirt cut out from a sheet of paper to prove that this problem was caused by using the Berkeley polygonal model and Berkeley fitting procedure. The paper model eliminates all errors caused by the fact that folded clothing is not flat. The paper T-shirt in all four states (unfolded, one fold, two folds, folded state) is depicted on Figure 10.3. Our experiments on the paper model demonstrated that problems were not caused by the third dimension of the T-shirt. It popped up that this issue is more expressed in the folded state. The problems with fitting of the folded clothing occurred again.

Another significant issue was related to speed. The fitting procedure took four to six minutes instead of target 2.5 minutes. Fitting of the folded model took almost ten minutes. These results were worse than our expectations. We had in mind that the model of clothing are expected to be used in real time robotic manipulations.



a) T-shirt in the initial position.

b) T-shirt with one fold.

c) Fitting of T-shirt in the initial position.

d) Fitting of T-shirt with one fold.

Figure 10.2 Experiments with the startup parametrized polygonal/skeletal model.

A more detailed explanation of issues related to the Berkeley polygonal model, fitting the observed data to this representation including speed of fitting was described in Section 7.2.

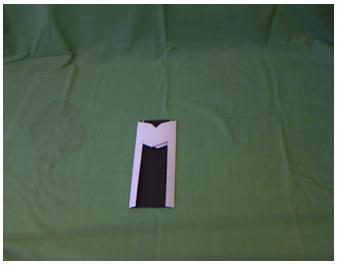
10.3 Our extension of the Berkeley model

We extended the Berkeley representation of a piece of clothing. We tested this extension experimentally too. The input images of our experiments were almost the same as in experiments described in Section 10.2. Small variations in cloth spreading were inevitable. The results of the experiments are depicted on Figure 10.8. We can see that our fitting method works fine in all phases of the folding. The experiments also demonstrated robustness of our solution. We report the experimental validation of the robustness separately in Section 10.5.

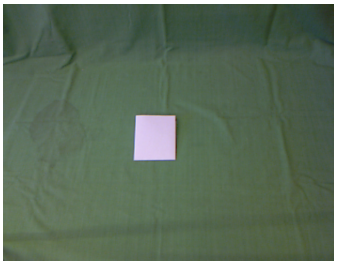
We performed fitting experiments on the towel folding example in order to test that our solution works also for another type of clothing than a T-shirt. The input images of the towel folding experiment are in Figure 10.4. Results of the fitting process can be found in Figure 10.5. Based on the experiments outcomes, we can say that the folding line of a model is properly fit to the folding line of the observed clothing. Small errors are still visible. They are caused by the fact that the homography transferring the image from a side view to a top view was not precisely set up. The other reason is that the twice folded towel is not flat anymore. Nevertheless, we concluded that the fitting results of the towel folding are good enough for our purposes.

a) Without folds.

b) One fold.



c) Two folds.



d) Folded state.

Figure 10.3 A paper model of a T-shirt.

a) Unfolded towel.

b) Towel with one fold.

c) Towel with two folds.

Figure 10.4 Towel folding experiment - Input data.

a) Unfolded towel. **b)** Towel with one fold. **c)** Towel with two folds.

Figure 10.5 Towel folding experiment - Fitting results.

10.4 Polygonal model speed issues

We were also improved the time needed for fitting the folded piece of clothing model to observations. We provide description of related experiments here.

The basic idea reflects the observation that the runtime of the fitting algorithm was very depend on the correct setup of parameters that control the black box optimizer. This optimizer fits the clothing model to the border of the observed and segmented piece of clothing. We were changing parameters influencing the fitting algorithm most and measured the time spent on fitting. We also evaluated quantitative the fitting result by the parameter called score. Table 10.2 shows the parameters and explains their role in the fitting algorithm. It tells which parameters were modified during fitting algorithms speed testing.

Table 10.2 Parameters of the fitting speed tests.

epsilon	Parameter defines threshold when numerical optimization stops.
fold_cont_pts	Number of points that represents folding line.
mode_cont_pts	Number of points that represents model.
num_of_iter	Maximal number of optimization iteration. Its represented by two numbers (A,B). The A number define number of iteration for initial fitting and the B number define number of iteration for folding line fitting.

We used the same input images in each experimental run related to speed improvement. The input dataset is depicted in Figure 10.8. Figure 10.6 shows the time spent on fitting the clothing model to the observed object. Figure 10.7 illustrated the score that represents the "energy function" as explained in the article [16]). The setting of an fitting algorithm is given in Table 10.3. Figure 10.8 depicts results of the fitting process. A green color represents fitted model.

The result of this experiment is shown graphically in Figure 10.7. We can say that it is possible to speed up fitting algorithm just by setting its parameters properly. In the robustness tests in Section 10.5 we used this fastest setting and it proved to be good enough. There might be some space for improvement if the a robotic device and automated testing is used.

Table 10.3 Experiment setting.

Exp. num	epsilon	fold_cont_pts	model_cont_pts	num_of_iter
0	0.2	30	1000	25,15
1	0.002	30	1000	25,15
2	0.01	30	1000	20,15
3	0.01	30	600	20,15
4	1	30	1000	17,15

a) Initial fitting. **b)** Fitting of 1sr, 2nd and 3rd fold line.

Figure 10.6 Time spent on the fitting of the clothing model to the observed object.

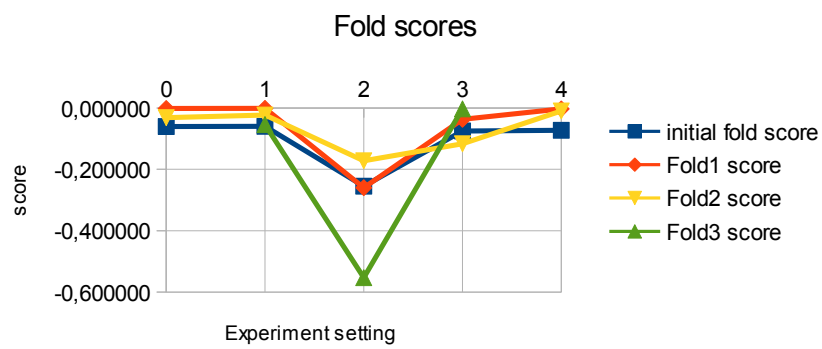


Figure 10.7 Score earned in the fitting process.

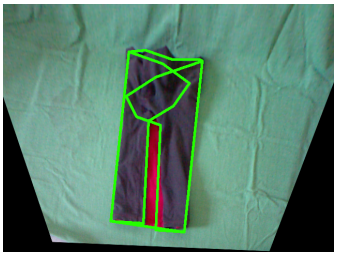
10.5 Polygonal clothing model, robustness issues

We tested how robust was our extension to the Berkeley implementation in the experiment. We tested robustness of the algorithm by creating four different spreadings of the T-shirt ranging from a nicely spread one to a rather crumpled one. The fitting algorithm tried to fit the model to the boundary of the observed object. The correctness of this correspondence was evaluated.

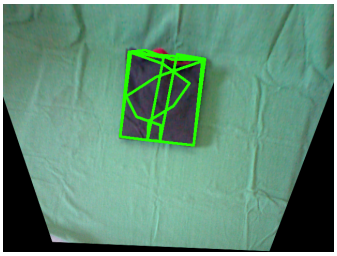
The source images and fitting results are depicted in Figure 10.9. The results of testing show that the extended polygonal model copes with a little undulated piece of clothing. Problems pop up when the clothing is spread really badly, as, e.g., in Figure 10.9d. Here the clothing model was not fit correctly. It is not so clear from Figure 10.9 but the left sleeve has been fit incorrectly to the bottom hem.

a) Initial fitting.

b) Fitting of the 1st fold line.



c) Fitting of the 2nd fold line.



d) Fitting of the 3rd fold line.

Figure 10.8 Example of model fitting results.

a) Nicely spread clothing.

b) Badly spread clothing.

c) Badly spread clothing.

d) Badly spread clothing.

Figure 10.9 Example of fitting robustness testing.

11 Conclusions and ideas for future work

11.1 Conclusions

The diploma project brought us to a new field related to perception and manipulation with pieces of clothing. This topic has been of interest to academia only. Industrial approaches to manipulating pieces of garments are rather different. Only a few research groups worldwide deal with the clothes perception/manipulation topic. In addition, almost every research group has its own approach how to deal with a clothes modeling (representation). We also realized that this field has a rather wide scope ranging from manipulators, computer vision techniques, planning, robot world representation, etc.

Our starting point was the understanding and reimplementation of the state-of-the-art approach to piece of clothing representation and its folding by the University of California, Berkeley group. Their work was published in Miller et al. [16].

This diploma project contributes in several directions.

On the practical side, we created our own experimental mockup consisting from the Kinect sensor and the background of a uniform green color. The manipulation with the piece of clothing was performed manually by a human to replace the robot. We incorporated our Kinect sensor into ROS.

We also proposed intermediate software layers (Robot Specification Layer, Robot Interface Layer and Clothing Manipulation Layer), which enable to abstract the future clothing perception and manipulation tasks from a specific sensing and robotics hardware.

On the research side, we improved methods used in the Berkeley implementation. This allows us to cope with a piece of clothing, which is not fully spread. These improvements were implemented and tested experimentally.

We also suggested a potentially more powerful representation of a piece of clothing by a manifold. This approach has the ambition to cope with a scrabbled or folded piece of clothing. We did not implement these ideas, though.

While designing, implementing and testing the Berkeley clothing model and our extension to it, we found that the original algorithm was rather slow. We did not get the same results as were published in the original article [16]. This was caused partially by a slower computer we used and partially by the inefficiency of the method itself. Our implementation was almost three times slower than the published one. We sped up the method considerably.

The performed experiments with the improved representation show that the new polygonal model is able to represent clothing that is spread on the desk and creates an approximately flat object. On the other hand, the model is not able to work with crumpled or folded clothing.

We also proposes the extension of above mentioned model to enable working with a crumpled or even folded clothing. The extension was based on the idea that the clothing is a $\mathbb{R}^2$ object embedded in $\mathbb{R}^3$ space. The mathematical concept, a manifold, is a tool at hand. We proposed this representation only theoretically, though.

Experiments performed illustrate that our modification of the polygonal model did what we expected. It can be seen that the implemented clothing models and algorithms

that perform association to observed data work in a range of situations.

11.2 Future work ideas

We have some ideas for a possible future work too. There is a space for clothing model extension. The manifold model extension was only outlined in this thesis. It needs to be developed, implemented and tested.

Another potential work is connected with the testing platform. The robotic device was unavailable during the Erasmus student exchange at KTH Stockholm. That is why the whole testing was performed on a mockup with a Kinect sensor and with a human as a substitute for a robotic device. The testing platform should be tested on a real robotic device and also slightly modified or extended to fit perfectly.

The other idea for future work is the design and implementation of learning algorithms to help decisions leading to folded/unfolded state of a piece of clothing. The idea is mentioned in Chapter 8. It needs more substantial attention, though.

Bibliography

[1] C. K. Au, Z. Wu, and M.M.F. Yuen. Effect of fabric properties on cloth draping modeling. In *Geometric Modeling and Processing 2000. Theory and Applications. Proceedings*, pages 69–76, 2000. 8

[2] J. V. D. Berg, S. Miller, K. Goldberg, and P. Abbeel. Gravity-based robotic cloth folding. In *In Proc. WAFR*, 2010. 9

[3] Ch. Bersch, B. Pitzer, and S. Kammel. Bimanual robotic cloth manipulation for laundry folding. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, San Francisco, CA, USA, Sep 2011. 9, 11, 12

[4] Q. Chen and J. Yao. The technical research of 3d garment fitting based on improved collision-check algorithm. In *Audio, Language and Image Processing, 2008. ICALIP 2008. International Conference on*, pages 893–898, July. 8

[5] Y. Chiricota. Three-dimensional garment modelling using attribute mapping. *International Journal of Clothing Science and Technology*, 15(5):346–358, 2003. 7

[6] Y. Chiricota, O. Cochaux, and A. Provost. Geometrical modelling of garments. *International Journal of Clothing Science and Technology*, 13(1):38–55, 2001. 7

[7] E. B. David, H. H. Donald, J. W. Michael, and E. B. David. A particle-based model for simulating the draping behavior of woven cloth, 1994. 8

[8] O. Etzmuss, M. Keckeisen, and W. Strasser. A fast finite element solution for cloth modelling. In *Computer Graphics and Applications, 2003. Proceedings. 11th Pacific Conference on*, pages 244–251, 2003. 8

[9] FLANN – Fast Library for Approximate Nearest Neighbors. `http://docs.opencv.org/modules/flann/doc/flann_fast_approximate_nearest_neighbor_search.html`. 40

[10] D. Freedman. Manifold reconstruction from unorganized points. In *Signals, Systems and Computers, 2000. Conference Record of the Thirty-Fourth Asilomar Conference on*, volume 2, pages 1744–1748 vol.2, 2000. 33

[11] M. Gienger, M. Toussaint, and Ch. Goerick. Task maps in humanoid robot manipulation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2008)*, Nice, France, Sep 2008. 9

[12] K. Hamajima and M. Kakikura. Planning strategy for task of unfolding clothes. *Robotics and Autonomous Systems*, 1999. 9, 12

[13] S. Hata, T. Hiroyasu, J. Hayashi, H. Hojoh, and T. Hamada. Robot system for cloth handling. *IEEE*, 2008. 11, 12

[14] X. Hu, Y. Bai, S. Cui, X. Du, and Z. Deng. Review of cloth modeling. In *Computing, Communication, Control, and Management, 2009. CCCM 2009. ISECS International Colloquium on*, volume 4, pages 338–341, 2009. 7

[15] D. Huang, Z. Yi, and X. Pu. Manifold-based learning and synthesis. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(3):592–606, 2009. 37

[16] S. Miller, M. Fritz, T. Darrell, and P. Abbeel. Parametrized shape models for clothing. In *Proceeding of the International Conference on Robotics and Automation (ICRA 2011)*, Shanghai, China, May 2011. 5, 9, 13, 21, 23, 26, 27, 28, 30, 39, 40, 43, 46, 50

[17] M. Musa and D.G. Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. In *VISAPP - International Conference on Computer Vision Theory and Applications*, page 331340, 2009. 40

[18] L. I. Nicolaescu. *Lectures On The Geometry Of Manifolds*. World Scientific Publishing Company, Incorporated, 2007. 31

[19] M. Oshita and A. Makinouchi. Real-time cloth simulation with sparse particles and curved faces. In *Computer Animation, 2001. The Fourteenth Conference on Computer Animation. Proceedings*, pages 220–227, 2001. 8

[20] J. Plath. Realistic modelling of textiles using interacting particle systems. *Computers & Graphics*, 24(6):897 – 905, 2000. ¡ce:title¿Calligraphic Interfaces: towards a new generation of interactive systems¡/ce:title¿. 8

[21] X. Provot. Deformation constraints in a mass-spring model to describe rigid cloth behavior. In *IN GRAPHICS INTERFACE*, pages 147–154, 1995. 8

[22] H. Qiao, P. Zhang, D. Wang, and B. Zhang. An explicit nonlinear mapping for manifold learning. *Cybernetics, IEEE Transactions on*, 43(1):51–63, 2013. 37

[23] A. Ramisa, G. Alenya, F. N. Moreno, and C. Torras. Using depth and appearance features for informed robot grasping of highly wrinkled clothes. In *International Conference on Robotics and Automation (ICRA 2012)*, Minnesota, USA, May 2012. 9, 11, 12

[24] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000. 34

[25] K. Salleh, H. Seki, Y. Kamiya, and M. Hikizu. Inchworm robot grippers in clothes manipulation - optimizing the tracing algorithm. In *International Conference on Intelligent and Advanced Systems (ICIAS 2007)*, Kuala Lumpur, Malaysia, Nov 2007. 9, 11

[26] H. Seung, Sebastian, and D. D. Lee. The manifold ways of perception. *Science*, 290(5500):2268–2269, 2000. 31

[27] J. M. Shepard, M. C. Towner, J. Lei, and P. Abbeel. Cloth grasp point detection based on multiple-view geometric cues with application to robotic towel folding. In *Proceedings of the International Conference on Robotics and Automation (ICRA 2010)*, 2010. 7, 9, 11

[28] J. Šindler. An updated implementation of a model and fitting algorithms for contour fitting approach and testing platform. `https://github.com/sindljan/clothFolding-berk`. 23, 39, 40

[29] Ch. Smith, Y. Karayiannidis, L. Nalpantidis, X.i Gratal, P. Qi, D. V. Dimarogonas, and D. Kragic. Dual arm manipulation - survey. *Robotics and Autonomous Systems*, 2012. 7

[30] M. C. Towner, A. Singh, S. Miller, J.F. OBrien, and P. Abbeel. Bringing clothing into desired configurations with limited perception. In *Proceeding of the International Conference on Robotics and Automation (ICRA 2011)*, Shanghai, China, May 2011. 7, 9, 13, 15, 16

[31] L.W. Tu. *An Introduction to Manifolds*. Universitext Series. Springer, 2010. 31

[32] E. Turquin. A Sketching Interface for Garment Design. Master's thesis, 2004. 7

[33] E. Turquin, M.P. Cani, and J. Hughes. Sketching garments for virtual characters. In John F. Hughes and Joaquim A. Jorge, editors, *Eurographics Workshop on Sketch-Based Interfaces and Modeling*, Grenoble, France, 2004. Eurographics. 7

[34] P. Ch. Wang, S. Miller, M. Fritz, T. Darrell, and P. Abbeel. Perception for the manipulation of socks. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2011)*, San Francisco, CA, USA, Sep 2011. 9, 11, 13

[35] J. Weil. The synthesis of cloth objects. In David C. Evans and Russell J. Athay, editors, *Proceedings of the 13st Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH 1986*, pages 49–54. ACM, 1986. 7

[36] Personal pages of Pieter Abbeel with articles about personal robotics. http://www.eecs.berkeley.edu/~pabbeel/personal_robotics.html. 7

[37] Honda - Asimo robot. http://asimo.honda.com/. 9

[38] Research group dealing with clothing at the University of California, Berkeley. http://rll.berkeley.edu/ICRA_2011/index.html, http://rll.eecs.berkeley.edu/pr/icra10/. 7

[39] CloPeMa project – clothes perception and manipulation. http://www.clopema.eu/. 4

[40] Cloud compare software. http://www.danielgm.net/cc/. 32

[41] Implementation of a model and fitting algorithms for contour fitting approach, University of California, Berkeley. https://github.com/rll/. 15, 27, 39

[42] DEXMART project – dexterous and autonomous dual-arm/hand robotic manipulation with smart sensory-motor skills. http://www.dexmart.eu/. 8

[43] Openkinect library that contains drivers for Kinect and Xtion. https://github.com/OpenKinect/libfreenect, https://github.com/piyushk/freenect_stack. The second link is more important because it contains nodes for ROS. 21

[44] LEAPFROG project – leadership for european apparel production from research along original guidelines. http://www.leapfrog-eu.org/. 8

[45] Manifold. http://en.wikipedia.org/wiki/Manifold. 31

[46] Open Source Computer Vision. http://opencv.willowgarage.com/wiki/. 21

[47] Point cloud library. http://pointclouds.org/. 32

[48] 3d mesh poisson reconstruction. http://vr.tu-freiberg.de/scivi/?page_id=365, http://research.microsoft.com/en-us/um/people/hoppe/poissonrecon.pdf. 32

[49] Willow Garage - PR2 robot. http://www.willowgarage.com/pages/pr2/overview. 9

[50] Python programming language. http://docs.python.org/2/index.html, http://www.python.org/. 20, 39

[51] Robot Operating System. http://www.ros.org/wiki/. 20

[52] STIFF project – biomorphic variable stiffness. http://www.stiff-project.org/. 8

[53] THE project – the hand embodied. http://www.thehandembodied.eu/. 8

[54] VIACTORS project – variable impedance actuation systems embodying advanced interaction behaviors. http://www.viactors.eu/. 8

[55] Y. F. Zheng and M. Z. Chen. Trajectory planning for two manipulators to deform flexible beams. *Robotics and Autonomous Systems*, 1994. 9