Body Posture Analysis in the Context of Shopping

Thesis,
submitted in partial fulfillment of the requirements for the degree of
Master of Science
in
Media & Knowledge Engineering

Alper Kemal Koç
Born on December 13, 1984, in Balıkesir, Turkey

Man-Machine Interaction Group
Faculty of Electrical Engineering, Mathematics and Computer Science
Mekelweg 4, 2628 CD Delft, The Netherlands
http://ewi.tudelft.nl

Video & Image Processing, Philips Research
High Tech Campus 36, 5656 AE Eindhoven, The Netherlands
http://www.research.philips.com

Summary

Author: Alper Kemal Koc

Student id: 4039947

Email: alperkemal@gmail.com

Section: Masters Media and Knowledge Engineering, Faculty EEMCS, TU Delft

Abstract

Shopping is a daily common activity for all individuals. In that context, there are needs related to security, efficiency and satisfaction. Store owners, customers and producer brands are three parties sharing these needs and there are intelligent systems that offer solutions. Intelligent systems could use the video information from the store and they can offer basic information about what the people in the store are doing by interpreting body language.

In this research to address those questions we aim to design a system which can automatically detect the basic actions in a store that are performed by people. We define the basic actions that are most commonly observed in a store and by considering sequences of these actions, higher level information about customers' behavior can be extracted.

We set up a shopping environment for experiments and made recordings in which people are doing shopping and performing the defined basic actions. We analyze the obtained data to examine the common properties and patterns of shopping related actions.

Next step is to extract the discriminative features from those recordings that can reveal the actions they belong to. For that part we use two tools, ETH Human Pose Detection framework and Kinect sensor developed by Microsoft. ETH Tool detects and returns the limbs of a person in the scene and we use this information to extract the angles between the limbs automatically. Kinect is capable of returning the depth information, people's silhouettes and if configured properly also the body skeleton coordinates. Furthermore the information obtained from silhouettes and body skeleton coordinates are used to extract different types of features. Next we evaluate the two tools and the sets of features with different classifiers by employing the developed automatic action detection software module.

To conclude we examine the shopping store data, evaluate ETH and Kinect tools with different sets of features and yield to conclusions about those actions and the problem itself. The action detection performance is not very high yet that leads us to a lot of interpretations and deeper knowledge about those actions and possible solutions for addressing the challenges of the analyzed problem.

Thesis Committee:

Supervisor	Prof. Dr. Drs. L.J.M. Rothkrantz, Faculty EEMCS, TU Delft
Committee Member	Dr. Ir. P. Wiggers, Faculty EEMCS, TU Delft
Committee Member	Ir. H.J.A.M. Geers, Faculty EEMCS, TU Delft
Committee Member	PDEng. M. C. Popa, Faculty EEMCS, TU Delft
Committee Member	Dr. Ir. Caifeng Shan, Philips Research

Preface

This master thesis in your hand is the final report of my graduation project as a Master's student at TU Delft. Those 2 years of studies was a tough challenge and it is a pleasure to be at the final step of it. Challenges are never easy without people around who significantly assist in any terms. With this opportunity here, I would like to thank to those who have been factors of me making it so far.

I would like to thank Caifeng Shan for supervising me during my internship at Philips Research working on this very project for several months and encouraging me to look for the right questions that should be addressed.

Special thanks to Mirela Popa who supervised me during all these months, always kind and nice and willing to help. Besides providing any technical help without a doubt, she was also always encouraging and caring; reminding me every now and then that motivation and me are the most important parameters of this challenge, encouraging me to stay calm, focused, motivated and relaxed.

I can't thank enough to Prof. L.J.M. Rothkrantz for everything since the day we met 2 years ago when I came to TU Delft for a one week course as a bachelor's student. Back then following his lectures for a week persuaded me to continue my studies at a Master's degree at TU Delft. Throughout this project, he always provided everything I needed in terms of technical advice, encouragement, motivation and all. The decision I made 2 years ago with the inspiration I got from him is proved to be the right one here at this moment again with his inspiration and help throughout the whole process.

Family, without a doubt, is a key in one's life, and also the people that you consider family. I would like to thank all those that are and have been very close to me and substituting for a family for all those 2 years and during this project as well. I thank you for making me feel always supported, never alone and for being there anytime I needed.

In my culture, they say if there is one person that would really cry for you when you are in trouble; that is your mother. The word 'thanks' can no way express what I feel towards my parents for providing everything they have for my wishes and goals. They are the ones who made this all come true. If where I am now is an achievement, that is something that belongs to my parents, and I would be honored to dedicate this to them.

Alper Kemal Koç

Delft – Eindhoven, The Netherlands

June 22, 2011

Table of Contents

Index of Tables

Index of Figures

1 Introduction

Every day in daily life, every individual person absolutely goes to shopping stores. In almost all of those stores there is some kind of surveillance that is usually used for the aim of security, assistance etc. It is also known to common knowledge that every shop owner would like to increase their profits by various methods. This is all about the economy and growing of the shop. There could be many ways of growing and they can be classified in two general sections; increasing the profit and/or decreasing the costs.

Figure 1.1: Shopping and electronics store where people interact a lot with products.

A good way of applying a new method of growing is using the already existing sources someone has. This way the new implementation has lower costs and small adjustments can be beneficial in the desired way. As mentioned above, a high percentage of stores already have surveillance systems and definitely also have some assistant people in the stores. Those assistants are the people that work for the shop and help people to find the product they are looking for or give information about the products. Those assistants are responsible for taking care of the customers and reply to their demands. They watch over the customers and see if anyone needs help with something or if they would like to get more information or they wait to be called by customers. Not only information but the assistants are also usually responsible to increase the sales. However since there is a lot of 'human' factor in this setting, the efficiency might not be good enough, and the costs might be high. It is not always easy to watch all the customers in a shop and provide assistance. In addition to that not all customers ask for assistance, although they have the potential to purchase something but are in need of encouragement. Therefore intelligent observation of customers can yield to less number of assistants, so that the costs can be decreased, and the customers can have better relations with the store. This process increases the sales and the profit as well.

In stores, particularly for the bigger ones, in addition to the sales assistance, it is also important to observe customers for security reasons. In many stores it is quite common and well known that a many theft cases occur every day. Not only theft, but the safety of the store and the customers is also important.

Assisting, customer relations and safety issues, they all require watching over the people in the store about what they are doing. Crucial information is what action people are doing in the store. Therefore an intelligent surveillance system which can detect the actions of the people in a store could yield to the desired system that would satisfy the requirements mentioned above.

This research focuses on automatic video analysis for detecting actions of individual people in a store setting using the surveillance systems and by using image processing methods. The domain of research, shopping context, is one thing that makes this research problem an individual one. In the sections below, we explain this concept in more detail.

1.1 Application Areas and Need of Action Detections

As also explained above, the surveillance and automatic detection of human actions could be used for certain purposes in general: for watching human behaviors in the store, how they walk around and their body language, and then generating semantic interpretation from those observations regarding their interactions with products.

Currently used systems in the market include customer analysis about the purchasing behavior of people. Those systems extract statistical information about customer behaviors related to the products in a store. This information addresses the brands and the store itself for optimizing the product arrangements as well as the setting of the store. The aim is to increase the sales by finding the optimum setting and product presentation.

Watching customer movements in a store could help the assistants in a store and lead them to the point and/or person where there is indeed a need. In such a case, even less number of assistants could do the same amount of work in such a setting where there is no such system and several assistants are just around the shop and waiting to be requested a query by a customer. That actually improves the customer satisfaction as well. Any individual person in a store would have lot better opinion about the store if they feel like they get help/assistance at the exact moment they need it and actually their needs are considered seriously by the store owners.

Customer interaction with the products is actually critical information, particularly today when there is actually a lot of research going on and a lot of implementations are proposed. This is specifically asked by the brands that would like to collect information from real customers about their products. The traditional method for obtaining the measurements starts with choosing a number of people as a test group. Then they are presented with the product and their opinions are recorded and analyzed. However this method might not be always accurate and realistic since the setting is not realistic at all. On the other hand in a setting where the people are actually naturally interacting with the products and giving subconscious, instant and spontaneous reactions are a lot more beneficial and meaningful.

There are face recognition systems evaluating the emotions of a person at the moment of interaction with a product. There are systems measuring the positions of people in a store and also the time periods they spent at specific locations. More examples can be given. Those examples show that, this interaction problem/information is something the brands and producers really need and are interested in.

There are also implemented systems that address the security issues. Those systems mostly focus on human detection and tracking, to check and prevent people from going to certain destinations, or to check if people leave or pick an object somewhere. The aim is to prevent people from stealing things or to prevent them from performing harm-aimed actions.

In case of big shopping malls those systems could prevent terrorist attacks by detecting if someone leaves an object in an unusual way. This way those objects could be detected and they can be checked by the security personnel.

1.2 Problem Definition

Following the information given above, in our research we address one of the basic requirements of such intelligence surveillance systems; the problem is to detect the human actions in a store/shopping context. What to do with that information (as explained above) could be a further topic of research. Yet we focus our research on this problem only and definitely this problem includes its sub-problems.

Although there are many methods offered for human action recognition, as will be stated in the related work section of this report, most of these methods are tested or designed for contexts in which the human actions are quite exaggerated and there is a huge volume of movement. Yet to our knowledge, there is not much research on human action recognition in a shopping context.

In such a system, there could be a lot of modules; object detection, object tracking, voice and facial expression detection. Yet we focus this research problem to the action recognition domain, meaning the actions that are performed by people individually with their bodies. The extracted information about the actions can be combined with other modules and further information can be revealed, yet this belongs to further investigation of the problem. The information we aim to detect could be considered to be low level information; we detect the action yet don't process it further.

In short, the goal of our research is to design automatic tools for the assessment of customers' interaction with products. We will focus on the body language and the associated behavior which can be assessed by detection of people pointing to products, touching products, grabbing products, taking products and putting them in the shopping basket, taking products from the shelf, inspecting them and putting them back, or exposing emotional behavior as raising hands or making special gestures.

The focus of our research is on the following aspects:

- Is there an interaction with products? Do people just pass by without even looking to products or the other way around?
- How do people perform interaction with the products?
- What do people do with a certain product, carefully examine it or just have a glance at it or even check how it looks on them?
- Do the people ask for help and do they need assistance?
- What are people's final decisions regarding a product, buy it and take it with them or leave it back on the shelf?

- Is it possible to model human behavior by considering certain basic which can be automatically detected?
- We will define a set of basic actions regarding the interactions with the products. All further interpretation of product-interaction can be generated from those basic actions.

To solve the problems mentioned above, we will follow several steps in our research. First, we look for already existing methods that work on modeling human movements and if possible find solutions that are suitable in the shopping context. After defining the basic set of actions we are going to develop software which automatically detects those behaviors. For analyzing that system we are going to collect data and construct a dataset for this context and test the software. The next step will be analyzing the recorded data to obtain the common properties and patterns.

Our expectation is to find the optimum feature set that is discriminative for the actions we mentioned above so that by employing an action recognition module we can detect them.

1.3 Research Challenges

In our research, there are some challenges that we face and aim at solving. In a problem like the one we address, there are challenges related to background of the scene, occlusion etc. Below the challenges that are in specific related to shopping context are explained.

1.3.1 Pose Estimation

For automatic detection of action, information about people’s poses in the frames is needed. Therefore the plain frames that are obtained from the scene are needed to be processed and the poses of people in terms of silhouettes, body coordinates should be extracted. This information can lead the method further to action recognition. This is the most important part of our research.

1.3.2 Cluttered Background

As expected in a store, there are many products around, shelves, hangers, human models and all those things are in quite varied colors. Such a setting makes the problem even more complicated because cluttered background is a commonly known problem for image processing. Most of the methods that will be introduced in the related works focus on simple plain backgrounds or much simpler backgrounds comparing to a shopping environment. The cluttered background of a store definitely introduces complexity to the problem and eventually errors.

1.3.3 Occlusion

Occlusion is also another commonly known problem in image processing. In our case, since we will need lot information about the human pose, occlusion is going to add a lot of complexity to the problem because in a shopping environment, people don't usually keep their orientation constants and they keep walking and turning around. That means that a lot of the frames are usually missing some part of the body. More than that, the clutter environment of a store can also create occlusion with the products or other people standing between the target person and the sensor. Interaction with a product makes it even worse. In such a case where the person might be examining some kind of clothing, they usually hold it up and wide as possible, and if that is done towards the sensor/camera, it means that almost all parts of the body are occluded.

1.3.4 Interaction with Products

As interaction with products introduces errors and increases the difficulty of action recognition and all the steps of those methods, it also brings up a new aspect of the problem; the product itself. Then the problem 'tracking the object' also rises. Types of products might need to be known for the definition of the type of the interaction. Particularly in a case where information for a brand or producer is collected, the object itself should be known such as the information is sent to the correspondent party.

1.3.5 Small Volume of Movements

In human tracking and action recognition, most of the actions are visually quite visible with high volume of movement. An action from a sports context can be given, like ballet, where the arms and legs all make huge movements and makes it easier to detect the action. However in a shop, there are actions where small movements are made, like examining a small product, checking how the product looks on the person in the mirror. Those actions are more difficult to be detected with a limited amount of movement. State-of-art methods do not address this problem very often.

1.4 Thesis Outline

Table 1.1: What we do, how we do and where we introduce it.

Action	Goal	Chapter
Getting information in related field	Examining other works that are done in the field to give inspiration about how to approach the problem in hand, about features, classification methods, tools to use.	2
Building a system model and using necessary tools	Designing a whole system that gets the video as input and the action labels as the output, with all the necessary steps in between; an automatic action recognition software.	3
Designing experiments for collecting data to work on	Building up a dataset to observe and actions. Evaluating the performance of the system.	4
Analyzing actions for patterns	Finding the main properties and patterns that appear in certain actions.	4-5
Extracting the discriminative features and analyzing them	For the classification finding the features that have low cost and high performance on detection.	5
Recognition of the action	Correctly detecting the action from a sequence of video.	6
Analysis and comparison of tools and our system	Evaluating the performance of the system we create. Comparing the tools we use like ETH and Kinect, or the features we define.	7
Yielding to conclusion	Answers to the research questions and outcome of our research.	8
Inspiration for further work	Outcomes of our research yielding and requiring further research. Inspiration for further work on certain steps of our system for improving it.	8

In our research, we use two main methods to address the problem. The software tool we call ETH Tool and Kinect that is designed by Microsoft for the gaming console Xbox are used.

Those 2 tools generate data and we extract several sets of features and test their performance on different types of classifications methods for action recognition. ETH Tool is upper-body and limb detection software. From the output of this tool, using the limb information, we generate action detections. Kinect is capable of generating multiple types of information about the scene and makes to solve the problem easier. It is efficient and more accurate. We then compare two approaches.

In the table above we define the actions we take and the goals we aim at. Chapters that include those actions we take are stated in that table as well. Layout of the thesis is as follows. In the next chapter, state-of-art methods and research done is introduced. Most relevant works that constitute the available knowledge in the field are mentioned. In Chapter 3, the model of the system we design for the aim of automatic detection of actions is introduced. In that chapter we also introduce the tools we use in our system. This chapter is the one that also introduce the first step of our system, capturing data. Chapter 4 is where we introduce the dataset we create, how we do it and what it involves. The features we extract from the actions are introduced in Chapter 5. Chapter 6 introduces our method of classification method we use and in Chapter 7 we exhibit our results and analysis on the performance of tools and features. In Chapter 8 we have the conclusions of the research as well as the future work that could be add on top of this work.

2 Related Work

In a shopping environment, people might be in different poses for different purposes. They might be browsing through the products, grab one and put it back, ask for assistance, try a cloth on, buy one and even stealing a product etc. Our aim is to analyze poses of the people in the environment and analyze those poses in order to detect the associated actions.

For the aim of annotating actions in a video sequence, poses will give us an inference. Accuracy of those detections is an important issue so that the actions can be detected. In the current state-of-art, there are several methods of representing people's poses. There are for sure methods that use those representations for action recognition. There is indeed a huge research invested in this area.

Most of the methods that are developed are triggered with different problems. Detecting poses for action recognition is actually a difficult problem, as an example human is a varying articulated object, and applications for different environments might also rise up different application specific problems. Some applications focus on specific environments like in [2] for TV shows, some claim to come up with generic approaches as in [1].

At this point it is beneficial to explain a shopping environment so that problems and limitations are clearer. First of all a shopping environment is definitely a much cluttered place. Most of the previous studies in the field focus on static and simple backgrounds. However our application has to deal also with the cluttered background problem. As it is also expected there might be more than one person in a scene or even much more, the system has to be able to detect multiple people and take the interactions and relations between those people into account, and this situation most probably might cause occlusion problem, both self-occlusion and occlusion between people. Illumination is indeed a common problem for almost all of the applications and this is also an important for a shopping environment. When the environment information is examined and the problems encountered are evaluated, we came up with the points below as the primary capabilities that the system should have,

- Sufficient representation of pose

- Accuracy in cluttered backgrounds
- Robustness to occlusion, both self and interpersonal
- Awareness of objects, clothes for instance, to prevent false negatives
- Low computational costs for a real-time system
- Ability for both frontal and profile view

Below those requirements are explained in detail and proposed solutions are introduced.

2.1 Sufficient Representation of Poses

In the current state-of-art there are various representation methods of a human in an image. Those can be silhouettes [3], bounding boxes or sticks representing the limbs of a person [1] [2]. Depending on the needs of the system, any of those configurations might be representing the whole body, upper body etc. In some systems, there are even some more compact representations like just the head and the hands. Those might vary depending on the requirements and problems.

In the case of action recognition in the context of shopping environment, most of the actions are expressed with hands. Regarding the actions like browsing through products, picking an item, and any other relation with a product, all those are explained with the arms of the person. In our case we do not observe the facial expressions since the problem itself is already a complicated problem and bringing facial expression in front would make the system more complicated rather than helping. Therefore in our situation upper body, and arms in specific, carries the salient information we are indeed looking for.

2.2 Accuracy of Detections

For action recognition, we indeed do not need perfect detections of the limbs. What we need is information which is enough to explain the action in a video sequence. Even some low level recognitions of arms, like resting besides the torso or rise up, or rise to the level of the chest and to the front, in case of browsing through the products or picking one up, might give us valuable information.

As will be explained later, we actually do not look for perfect detection in single frames. There is actually a trade-off between the accuracy of the detections and the computational costs, and real-time of the system is one of our goals, not perfect but rather accurate detections are our current goals in the system.

Even though we are looking for information in a video sequence, for sure we will be working in single images, as frames of the video. Current state-of-art indeed includes two types of detections, single image and video pose estimations. From a different point of view however, video sequence task might be seen as an extension to the first one. Since we do not expect the perfect limb detections, and we are also looking for video sequences, we can use the improvements done in this field. Below those are explained. However since video sequence processing could be seen as an extension to single frame processing, still image detection is our first stop for the procedure.

2.2.1 Single Image Detections

Single image detections might be seen as the ground of further extensions. Almost all methods first rely on single detections, then pass that information on the video sequence information and update the poses with regard to relations between the frames.

There are indeed different approaches to single image human pose estimation problem. Those approaches might differ according to later goals because they might be preprocessing of some higher level problem, like action recognition. Different methods can be classified according to their usage of human models. Some approaches use human models, some not, and also how those models are integrated in the system leads to different types of methods.

Human model used in those approaches are usually tree maps and they configure the joints or limbs of a body. The model after they come up with focus on the relations between the limbs, e.g. for stick representations relation between sticks.

One of the most popular approaches is using the pictorial structures Fischler and Elschager. In [5] Felzenszwalb and Huttenlocher extend this method for object detection. Most of the methods rely on this approach, extending it according to their needs. This method uses color information of an image and builds a model of parts of an object. In case of human detection, this method works like this. Pictorial structure model stands as collection of body parts of a human. There are connections between those parts to yield to the final representation of a person in an image. Pictorial structures are used by many authors like [1] and [4]. Even though this method uses a color-based model, some authors come up with different proposals depending on their constraints, in case color information is not a valuable feature or not discriminative enough.

As stated in [6] as well, one important aspect of human pose estimation is that human body is a very articulated object. Many authors offer methods that have many constraints since it is a very difficult problem. Most of them focus on certain acts of people, like only walking etc. In that case, the limits of possible body configurations are limited and it’s more likely to have a more accurate detection. However in our case and in many cases indeed, it is also important to detect unexpected moves of people as well. Therefore the need is a method that is not limited to certain acts but more like a generic solution that can detect as many poses as possible.

Use of human model also differs between methods. So called indirect human model, is a model that is built at the beginning of the processing and it is used as a reference. Another approach called direct model use, there is a human model and it is updated throughout the procedure with new incoming data.

To give an example, we can have a look at the model Ferrari and his colleagues presented in [2]. In the figure below, the cells represent the limbs of upper body, meaning head, torso, upper and lower arms. A limb is represented with their coordinates, scale and orientations.

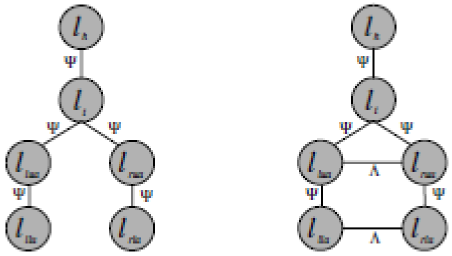


Figure 2.1: Pictorial structure model for limb detection of Ferrari.

Each stick in a human model, with the mentioned attributes, is represented as a vector l_i. What is important about the model is that every stick is defined based on the information given by other sticks. To put it more simply and as an example, upper right arm’s position is both depending on the features from the image itself and also on the vector l of other limbs. This relation is given in the equation below.

$$\mathrm{P}(\mathrm{L}|\mathrm{I}) \propto \exp\left(\sum_{(\mathrm{i},\mathrm{j})\in\mathrm{R}} \psi(\mathrm{l}_\mathrm{i}, \mathrm{l}_\mathrm{j}) + \sum_{\mathrm{i}} \phi(\mathrm{l}_\mathrm{i})\right)$$

Eq. 2.1

In this equation, the probability of stick configurations *L* given image *I* depends on the pair wise potentials of sticks *Ψ* and also the potential from the image *Φ*. This procedure is called parsing. That procedure leads to possibility distributions. An example can be seen in the image below.

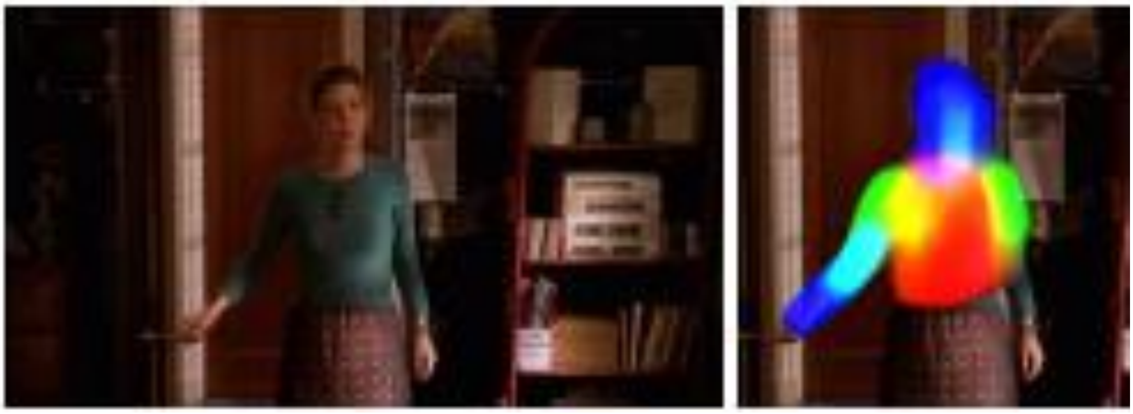


Figure 2.2: Left: Original Image. Right: Possibility map of limbs.

Ramanan comes up in [9] with a solution that is called image parsing and this method also has a wide use among other authors. Pose estimation is done by iterative parsing of the image. Different methods use different features yet Ramanan tries to benefit from those different approaches. In the first iteration, a model called edge-based deformable model is used. This part uses edge properties of the image and the result is a rather weak estimation of the limb coordinates. Using those possible coordinates possible color-models of limbs are learned and in the second iteration this information is processed with region-based deformable model which yields to a more confident information about limbs. The problem in this approach is the selection of the starting point which is quite important since the following steps are based on the initial results. If those soft estimates are terribly wrong, then errors rise cumulatively. One superior property of this method is that the features used are low level image features so the computational costs are lowered and also there aren’t very complicated features to be calculated.

Ronfard in [10] also uses pictorial structures with some advancement. In pictorial structures, the object is modeled as parts of a whole as mentioned before and there are simple part detectors in the processing. In [10] this simple part detectors are replaced with more dedicated part detectors and they are further processed using SVM to detect if it is really a body part or not. Later on using graphical models, those parts are connected to each other with known kinematics information of a body. Methods, which first detects body parts and then examines the connection between those parts are called bottom-up and the other way around is called top-bottom approaches.

2.2.2 Video Sequence Detections

In the case of further processing of poses, it is usually a fact that sequence of poses might lead to an action. Therefore processing the poses using previous or even further frames is also a method that can be used.

Single image detection might include errors for sure. When those poses are dynamically detecting, meaning that a group of frames are taken into account, those errors might be corrected with the information from neighboring frames. In our case, that might be important information to use since there might be pretty wrong detections in between the frames and those might lead to wrong labeling of actions. Using a few frames back, an expected pose can be concluded and can be used as a ground truth for the current frame. That might really improve the accuracy of the detection.

As mentioned in the previous section, most approaches use human models for pose detection. In case of action detection in a video sequence, there is strong evidence which can help improving the detection because there are other detections from previous frames which are more or less quite similar to the current frame. Therefore updating the previously mentioned human model now focuses the relations and potentials between the frames, so that means other relations are added to the already existing limbs pair wise potentials. In this case the model becomes something like the one depicted below.

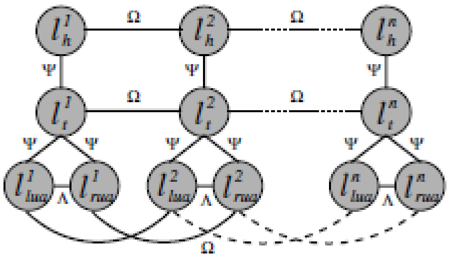


Figure 2.3: Temporal models.

What is different from the presented approach here is, other frames are also taken into account and the potential relations are given with Ω. The equations above improves the features from the image with the pair wise potentials, and this model even puts it further with adding the given information from previous frames. In this case, Ferrari updates the model like below,

$$P(\{L^t\}|\{I^t\}) \propto exp\left(\sum_{t,i}\left(\sum_{j|(i,j)\epsilon R} \psi\left(l_i^t, l_j^t\right) + \Phi(l_i^t) + \Omega(l_i^t, l_i^{t+1}) + \Lambda(l_{lua}^t, l_{rua}^t) + \Lambda(l_{lla}^t, l_{rla}^t)\right)\right)$$

Eq. 2.2

Here we see that, for the current frame probability distribution of limb configurations depends on relations in between limbs, features from the image and also correspondence to the previous frames. There are two other factors seen in the model that are shown as ᐱ. This is just a penalty value not to detect one of the arms twice, to prevent the double detection of a limb.

2.3 Limb Detections

For the sake of examining methods in more detail, we can see how the approach in [1] works. As mentioned before this method also uses articulated body model. This model takes the relation between the body parts into consideration. Every part knowledge affects another part. It is a probabilistic approach and every limb is detected based on the knowledge of other limbs.

In their approach the body configuration is modeled like below,

$$p(L|D) \propto p(D|L)p(L)$$

Eq. 2.3

D representing the image evidence and *L* is the part configuration, including body parts. *p(D|L)* is the likelihood of the image evidence given a particular body part configuration. *p(L)* stands for kinematic tree prior. Obviously *p(L|D)* is the goal that is the body configuration given the image evidence.

Those two terms are both learned from the training data and it is important what those two terms include. Kinematic tree prior *p(L)* includes the probabilistic constraints of part configurations and their influence between each other. Here by probabilistic constraints a few things are meant but kinematic properties of human body is the fatal one. The problem here is indeed this method is not really robust against occlusion because those models mentioned above do not include this information or possibility. Although this methods claims that it overcomes three other prior methods, it still needs adjustments for occlusion etc.

As [1] suffers from occlusion, other methods also have low performance against occlusion. In our case, this is a very important problem as well since the environment is expected to have many people and most of actions we are looking for might include self-occlusion. There are for sure some offered methods aiming at occlusion.

2.4 Occlusion

There are actually two types of occlusions that could be approached differently. First of them is self-occlusion, in which more than one limb of a person might be covered in the same region of the image. It is important to note that, as also mentioned in [6], pictorial structures which is widely used is very weak against self-occlusions. Since the research is still investing on limb detection in a general view, occlusion robust methods are not that strong yet. Pose estimation itself is a very difficult problem already. However there are still works done on occlusion.

In [6], Sigal and Black offer a pose estimation method that is claimed to be occlusion-sensitive. As mentioned above, with the awareness of pictorial structure's weakness against occlusion, they come up with a model that also considers occlusion. They define a variable called occlusion variables and it is simply a binary vector to be applied to the image as a mask. The information includes if the current region is occluded by another limb or not. If that region is occluded, there is a penalty that this mentioned variable reveals so that the limb occluded should be represented as a 0 here so that the algorithm is not confused with the occlusion and yielding to bad models. In the whole flow of the method, first the whole detection is done, and then the message about occlusion are processed and the pose estimation is updated and the final result is obtained. The information used in this method consists of silhouettes and color information of the image.

Besides the self-occlusion, interpersonal occlusion is also quite important since there might be more than one person in a scene and they might be occluding each other. The common problem is actually assuming that there is only one person in the scene or people are not occluding each other at all. The solution indeed is here and methods should also include that possibility as well and they should be addressing many people to be detected in a scene.

Eichner and Ferrari propose a new model in [4] to overcome the interpersonal occlusion. As mentioned several times before, almost all methods come up with body models that help them yield to the final pose estimation. In their method authors include the occlusion in their pictorial structure model by extending the traditional pictorial structures method. For a review and to make sense, below is the traditional pictorial structures model for single person indeed,

$$P(L|I,\Theta) \propto exp\,(\sum_{(i,j)\in E} \psi(l_i,l_j) + \sum_{i} \phi(l_i|I,\Theta))$$

Eq. 2.4

Authors extend this model and add a few more variables for the occlusion information, and the final result is,

$$P(L|I,\Theta,\mathrm{Z}) \propto \exp\,(\sum_{p\in P}\sum_{(i,j)\in E} \psi(l_i^p,l_j^p,) + \sum_{p\in P}\sum_{i} \Phi(l_i^p|I,\Theta,Z) + \sum_{(p,q)\in\chi}\sum_{i}\sum_{j} a_{ij}\omega(l_i^p,l_j^q))$$

Eq.2.5

In this model, the last term in the right side of the equation, there is information about the locations of people in the scene. Their method again first starts with the upper-body detection and the coordinate information of those upper bodies help them construct the mentioned term in this model. They also update the first term, which is kinematic constraints and they distribute it among the people in the scene. Z here itself represents the given occlusion probabilities using the bounding boxes information. They calculate the probability of the occlusion state for every person in the scene and every body part as well, and this value depends on upper body detections in the scene. It is important to note that the authors also consider the border occlusion, which means that a whole body of a person might not be in the scene but some.

An important thing to be mentioned about occlusion is that most of the methods come up with models that assume that all parts of a body are visible. However this might not be the case and when the method works, it assumes there are all the body parts existing and it yields to a false negative. It is critical to include also occlusion information in the body models.

2.5 Search Space and Costs

In human pose estimation problems, the computational costs are quite high since it is a difficult problem because of the high articulation of human. There are many possible location and orientations for arms in particular and to detect those most of the methods are processing most of those possibilities to yield to the best detection.

Although like in [1], there are no proposals made for search space reduction, still reducing it make a huge contribution for computational costs and more accurate detections and pose estimations.

There are indeed different types of search spaces in those approaches. One of the problems is detecting the person in an image and looking for limbs at the correct positions. There is a difference between the person, object in other words, and the background. Searching the background has for sure no benefit in pose estimation. For these problems, some approaches offer background subtractions yet to do this, the camera should be static and also the background should be static as well. The mentioned pictorial structures [5] also use background subtraction. In some situations like video processing as presented in [2], background subtraction is not possible because it is not stable.

If background is not static, then there are those other methods offered. Following the background subtraction method and improving it, Stauffer in [12] offers a method in which the background is modeled and in case there is a change, the known background image is updated and adapted to the new condition.

In cases where background subtraction is not possible, there are different methods offered. In [2] the offered method reduces the search space in quite 2 efficient steps. In the first step, an upper body in the image is detected and the whole further processing is done in this bounding box. The second step is called foreground highlighting and although it does not produce the best results, still it contributes to search space reduction. Foreground subtractions works like this.

From the upper body box detected in the previous step, a few points are selected as possible background, possible torso and possible head. This are just estimates following the coordinates of the bounding box, expecting the head to be somewhere in the upper middle of the box and torso below that. This is not the best approach but the aim is not that anyway.

Learning the color models from those points, every pixel in the bounding box and a few more pixels around the box as a frame are labeled as foreground or background. The overall result is a mask for the image parser to work on including the possible object, human in this case. It for sure includes some of the background but still the search space is reduced.

In our case however, the camera is expected to be static, and the background as well with a possible few exceptions. Those exceptions might be caused by illumination differences in different times and possible location changes of objects in the scene. A possible, and to be tested for sure, method for search space reduction in our case might be using the adaptive background subtraction.

2.6 Action Recognition

As stated in [8] action recognition has been a known field of research however results are still not satisfactory yet. Works that are made in this field usually are for “surveillance, medical studies and rehabilitation, robotics, video indexing, and animation for film and games” [8].

For action recognition, we can review the approaches considering the features that are used. There are quite a lot of methods that use the silhouettes or silhouette contours. This representation and the features that are extracted from those are proved to be good features for action recognition task. In [28], authors extract silhouettes and find the distances to the local maxima points of the silhouette contour. The feature set representing a pose in a frame is the vector of distance between the point on the contour and the center of mass. Later they also find the local maxima points in this vector. Those points are assumed to be the end of limbs, which is the case in most poses. In [15] edge features are used for discrimination. The edges meant here are also originated from the contour of silhouette image. The method relies on the human pose and they match the pose sequence to one of those in the database. As mentioned, edge features are used for that matching process. For each action there is a key frame(s), a key pose in other words, and frames of the test sequence are matched to that key frame. For a kneeling action for example, they pick the apex poses of the action and try to find the matches with the least cost. However in this action recognition method, rather than all of the frames being included, the limited numbers of frames that match the model are searched for. Therefore they make the detection with just a few frames of the sequence. In [16] rather low level features are used but still using the silhouette information. They extract features from the sequence of videos, put them in a matrix and get the covariance matrix of that matrix. This way a feature vector of constant dimension for every sample is created, regardless of the number of frames in the sample. Features are extracted from the silhouettes, naming silhouette tunnel when all sequence is put together. The features that are extracted from the silhouettes are the distances to the extreme points, in 8 directions, of the silhouette from the center of mass. In [18] an approach using silhouettes, silhouettes tunnels and silhouette volumes is presented. From those silhouette volumes, they extract a couple of features.

Silhouettes can also be used for other purposes as well. In [21] a method is introduced and it can detect the interaction of people with the object using the silhouettes. They can detect if someone has something in hand, or just left it somewhere.

There are methods that use other features rather than the silhouette information, [24] using 3D data and computes history volumes. They extract features from those motion history volumes. They claim those features are discriminative for action recognition task. [23] detects and tracks hands and using HMM action recognition is done. [27] detects part of the body and tracks them. The movement pattern explains the action.

Action complexity approaches are also a well-known perspective for solving the problem. There are approaches which say that actions are a combination of more simple ‘sub-actions’. In [20] they allow simultaneously performed actions to be detected. One can ‘jump’ while ‘running’.

It is also important which method is used for the classification task. One of the most popular methods in this field is the Hidden Markov Models (HMMs). [19, 22, 23] uses Hidden Markov Model for classification. [26] uses time-sequential features and HMMs for classification. Another popular method is Support Vector Machines as it is used in [17] and [27]. Different than those [18] uses nearest neighbors classifier. [24] uses linear discriminant analysis. Those examples show that there could be different choices for the classification method and they are all proved to succeed with a good feature set.

In [17] the problem that is addressed is the number of frames that is required to perform an action or to represent an action. They offer to work on sequences of actions with number of frames from 1 to 10. They prove that this yields to the same performance as processing the whole video sequence. In [15] detecting the action depends on the match of frames with the key poses. Therefore there is no strict statement of the number of frames an action should have.

2.7 Common Feature Definitions

In the methods mentioned above and the methods we are going the use and explain in detail in later chapters, there are specific terms for information/features that are quite often used. Those are quite important and it is also beneficial for the reader to explain here which they are. The terms and features that will show up quite often are 'silhouette', 'limb sticks' and 'depth' information. Below those terms are explained in details.

2.7.1 Silhouette

The *silhouette* of an image is the labels of the people that are in the frame. In terms of features, a silhouette image of an image with m rows and n columns is a binary matrix with m rows and n columns that has the value 1 for the pixels where the person is. In Figure 2.4 an example can be seen, with the original image, that is obtained from one of the samples of our recordings and the relevant silhouette binary image for the specific person that we track actions of.

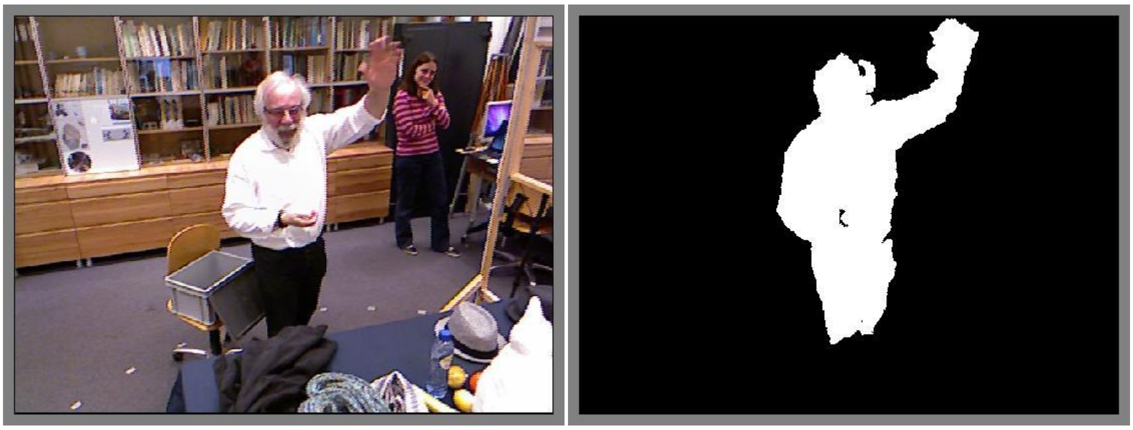


Figure 2.4: Silhouette sample. Left: Original image. Right: Silhouette image for the significant person.

2.7.2 Limb Sticks

Limb stick coordinates are the locations of the limbs of a person that is being tracked. Since we focus on upper body, we are usually going to focus on upper body limbs like, head, torso, upper arms, and lower arms. The value of those features is simply the end points of the mentioned sticks in terms of row and column in the image matrix.

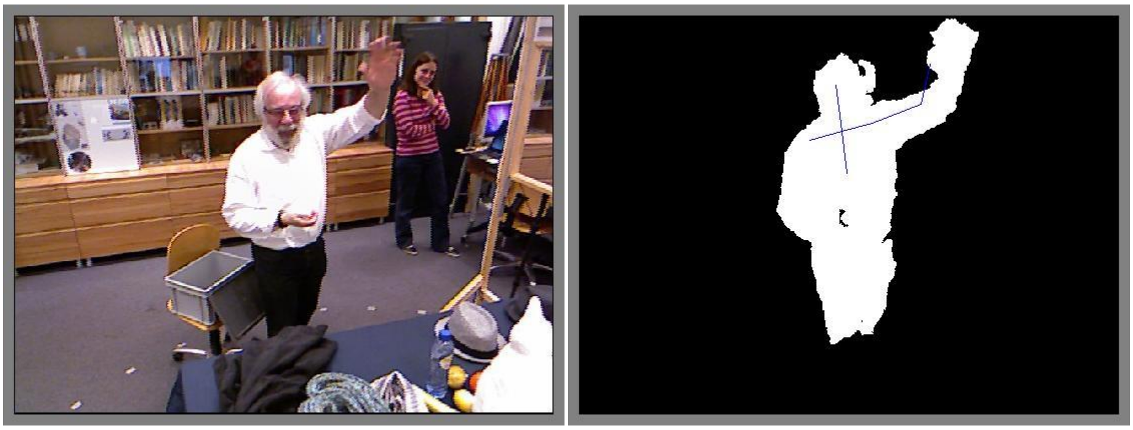


Figure 2.5: Stick sample. Left: Original image. Right: Silhouette image with the stick configurations.

In Figure 2.5 we again have the same sample from the previous section. On the left the original image frame from the sequence can be seen and on the right, there is the silhouette image with the sticks drawn in blue. In this case, the sensor is capable of generating the position of head, neck, torso, right and left shoulders, left elbow and left hand. So the head, torso, shoulder, upper left and lower left arms sticks can be determined. As can visually be seen from the sample, stick configurations are not as accurate as the silhouette images, yet it is good enough to interpret the movement of the arm from that position information. As explained in later chapters, that lack of accuracy is considered and relevant methods are designed accordingly.

2.7.3 Depth

Sensors available are capable of measuring the depth information in a scene. This helps in a lot of fields, like extracting silhouette information as well. As will be explained in detail later, depth information in a scene might be crucial in a shopping context where arms are quite likely to be in front of the body and can't be explained with silhouette information. In our experiments we extract depth information and process it for further use.

The depth information that is generated by the sensor, naturally measures all the points of the field of view. However we usually need the depth information that is only correspondent with the person we keep track of. Apparently all other depth information is redundant.

The information that is initially generated by the sensor, we call depth, and the image that has only depth information about the person and nothing else for remaining pixels, we call depth silhouette. This information is actually easily and simply generated with applying the silhouette information as masking and clearing all other data in the scene.

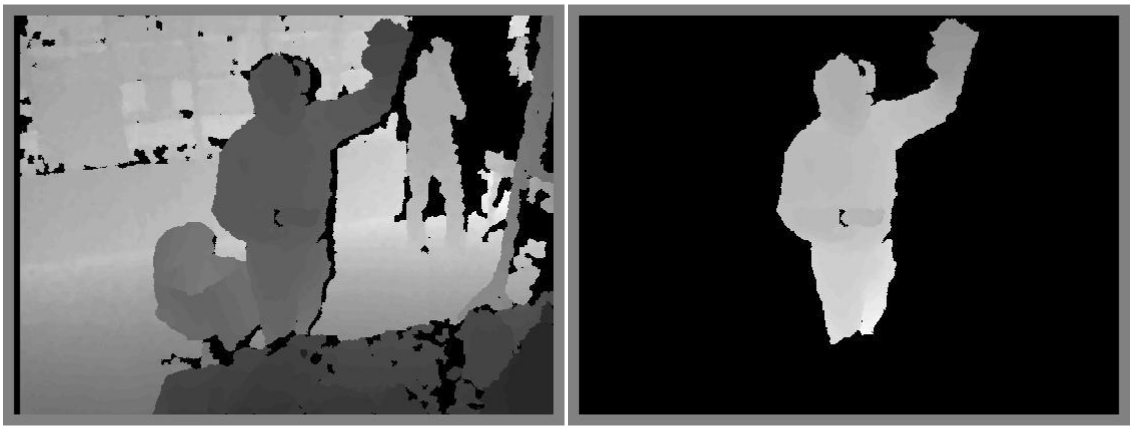


Figure 2.6: Depth information of a video frame. Left: Depth image with whole scene. Right: Depth silhouette.

As can be seen in Figure 2.6, on the left there is the depth information of the whole scene. Since we have the silhouette information of the scene as well, we simply apply it as a mask and get the image on the right, which only includes the depth of the person we track.

2.8 Conclusions on Related Work

Examining the mentioned works on the problem field gave us some ideas about how to approach the problem. It is seen that although there are many methods published in the field, there aren't many works for certain contexts, in particular shopping context. Therefore there is definitely a need of methods and the approaches that focus on that context might yield to advancements in the research field because of the new problems introduced by this context.

It is seen that for feature extraction, there are a couple of methods that are popular and proved to be successful for discrimination. It is seen that silhouettes and contours are good features and they have been used a lot. Therefore we decide to include this concept and develop our approach on this feature sets. It is also seen that although there are methods which use body part tracking and then yielding to action detection, there aren't much works that really focus on the relation between the body parts of the body. Our assumption is that arms in an action do most of the 'meaningful' movements, especially in shopping context, therefore we decide to focus on upper body and the positions of arms relative to each other and the rest of the body.

In terms of recognition step, there is first one thing that needs to be decided; feature set representing the sequence as time-series or another constant dimension feature set is extracted. We’ve seen that Hidden Markov Models are quite popular for the recognition task and the feature sets are usually extracted per frame and the sequence then is represented as a time series of those feature vectors. It is a popular and proved method; therefore we decided to include that in our approach as well. However there are also methods which extract another feature vector from those time-series features and represent the sample regardless the number of frames in it. One common method of doing is getting the covariance matrix of the time-series feature set. Therefore we decide to use this approach as well. This inspires us to use the standard deviation of the series, because the main diagonal of the covariance matrix are the standard deviations. We decide to test those approaches in our research.

Those inspirations we get from the related works in the field set up our first step of approaching this problem and using those assumptions, feature sets and recognition methods, we build up our system that will be explained in detail below.

3 System Architecture

3.1 Model

For the aim of finding a solution to the problem in hand we need develop a system. As in another system that are a few steps in our model. First we start with capturing the data. Yet obviously there are a few things we need to do to make sure the data is suitable for the next steps. This step is followed by extracting the features and then the classification is done for recognition. In this chapter we introduce the software we developed and the tools we used.

Whole research includes many steps and our design has several parts to it. The aim is to have an automated action recognition system therefore the software framework we build is automated. However there are also some parts of the research where we manually design, like experiments as would be expected.

In the figure below, we show all steps of the research. The steps are shown as rectangular boxes and the data produced are shown as data collection. Red boxes are the automated parts of our software framework. Those constitute the software framework we develop with inclusion of other tools we use. Those other tools are Kinect as the hardware and ETH, PRTools and HTK as software. Those parts that are done by those additional tools are shown in green boxes. As an example, Kinect return a video file. We process that file, segment it into frames and generate the data we are going to use in the later steps.

Blue boxes are part of our work as well yet they are the manual steps. Those are not done by the software framework we make but are done with manual work. In the coming chapters, all those steps will be explained.

In this chapter we first introduce the tools we use. After that we explain how we extend and use those tools. Following chapters will also show the other steps of the automated software framework flow as shown in Figure 3.2.

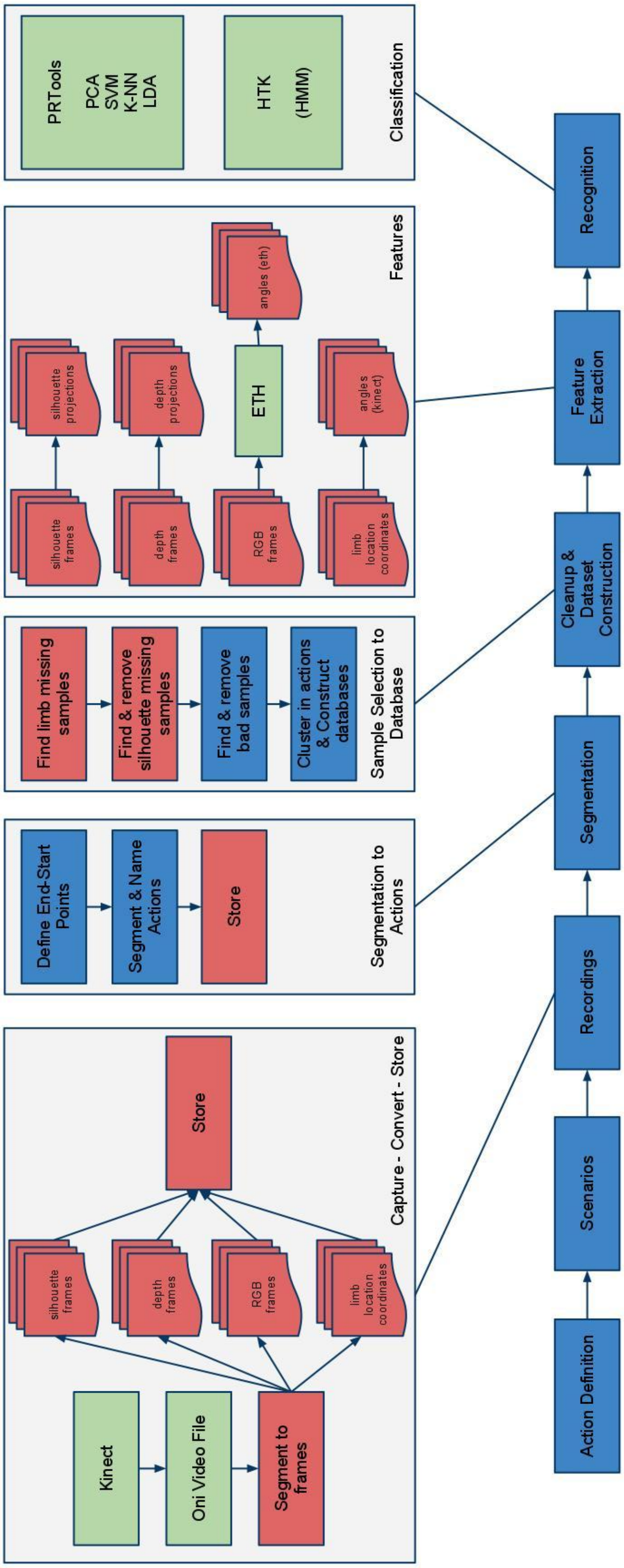


Figure 3.1: Overall workflow of this research. Steps in red indicate the automated parts of our framework. Steps in blue indicate manual parts of our framework. Steps in green are the additional tools we used.

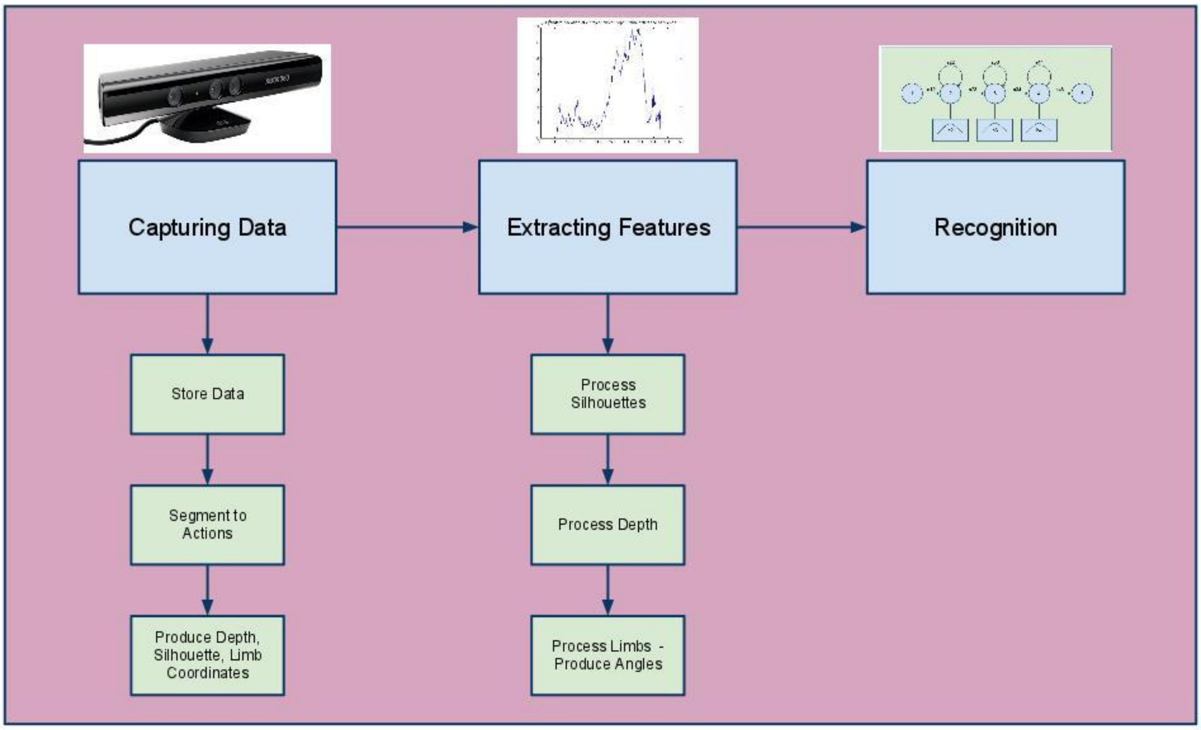


Figure 3.2: Work flow of the overall developed system.

3.2 Tools

There are mainly 4 tools we use in this experiment, ETH, Kinect, HTK and PRTools. We use Kinect for capturing data. Kinect and ETH are used to extract the information and lead to the features we use. HTK and PRTools we use at the classification step of the system.

3.2.1 ETH Tool

The software that we will refer to as ETH Tool, is a software toolbox developed at ETH Zurich by Ferrari et.al [2]. The software can process single images, or frames from a video, to detect humans in the scene and their body stick positions. The tool works only with upper bodies. In this chapter, this tool will be introduced in more detail.

There are three main parts of the system. The upper-body detection framework within the software detects the people in the scene and passes it to the later steps. Then a process called ‘foreground highlighting’ takes over the job and by using the upper body frame it reduces the search space by roughly labeling foreground and background areas in the image. Later on the pose estimation step is done consisting of the parsing method of Ramanan.

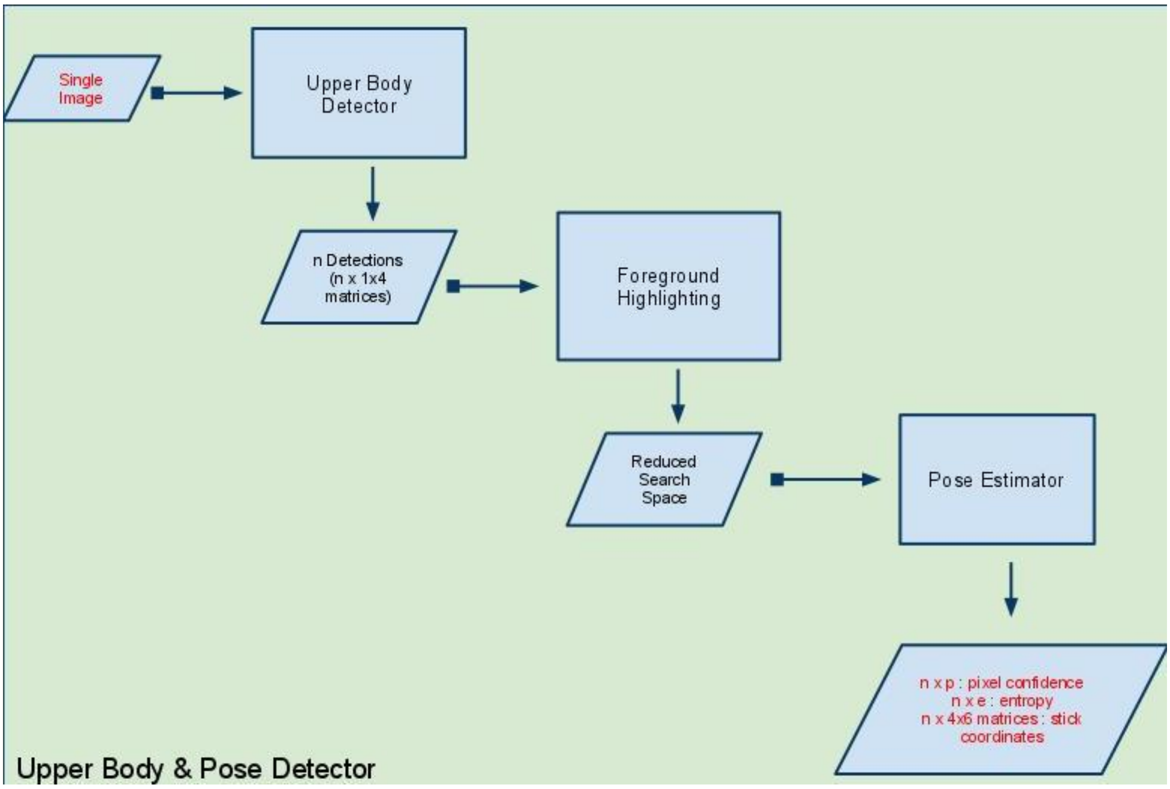


Figure 3.3: Work flow of ETH Tool.

3.2.1.1 Approach in Detail

3.2.1.2 Upper-Body Detections

For upper body detection, the system uses Histograms of Oriented Gradients. The idea is that the image is divided into small blocks, called ‘cells’ and each of them are described as histogram of oriented gradients. Later on a sliding window system searches for objects and localizes them. Every time an object is localized, a linear SVM classifies the object as ‘object’ or ‘non-object’.

During the process, some side-tools are used to improve the performance. Photometric normalization is used in every cell to overcome the illumination difference problem between tiles. This makes the algorithm robust to different lighting properties. Upper body detection has its limitations though. A person’s orientations regarding the camera location might be different in many situations. The system is now not successful with profile views. It is aimed at frontal views and it is accurate as long as the viewpoint is between 30 degrees range from exact frontal view. Besides that, it can also detect back view; the person does not need to be facing the camera.

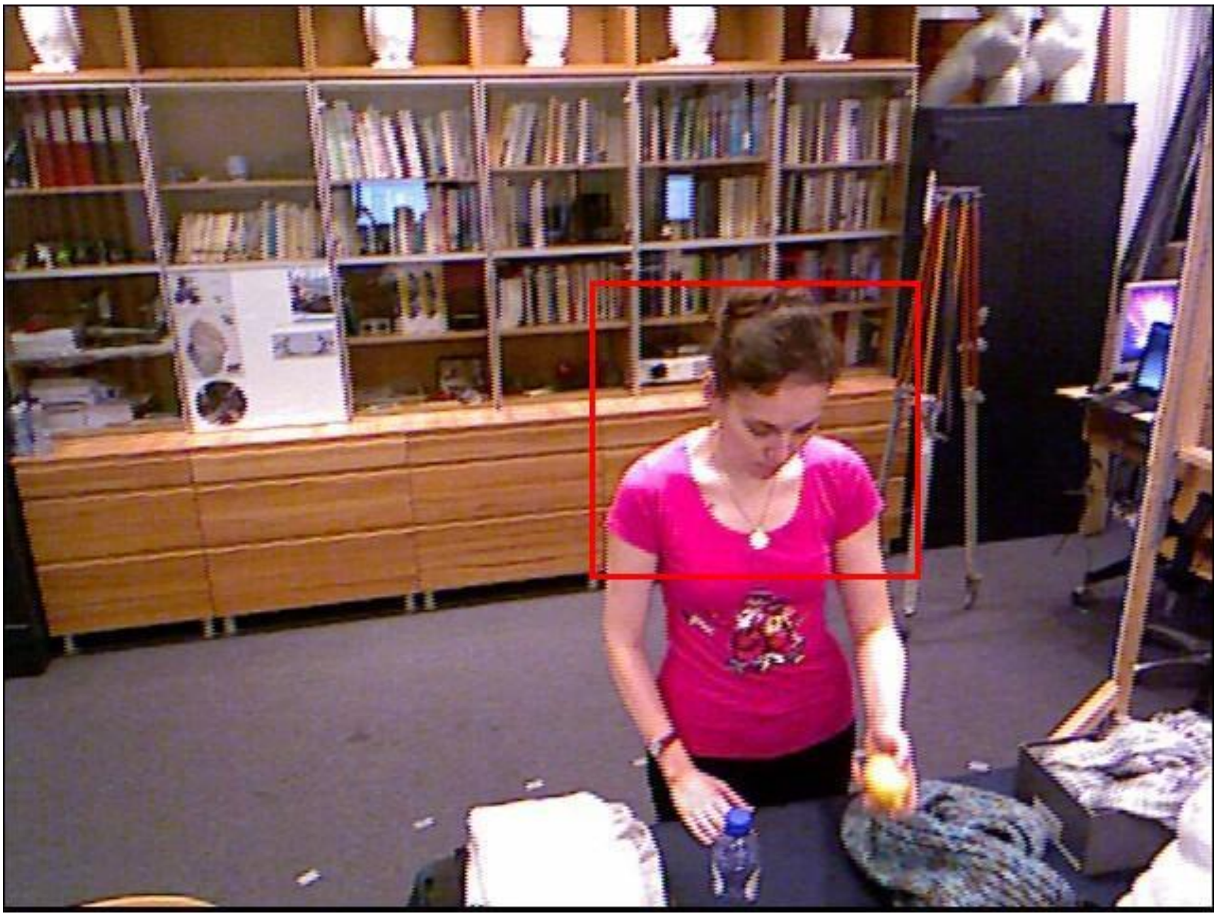


Figure 3.4: A frame with upper body detection with the first step of ETH Tool.

This part of the system simply returns a bounding box where it detects an object, an upper body in this case. This output is passed onto the pose estimation part of the system. Upper body detection is successful enough, as it efficiently detects the upper bodies with precise locations.

3.2.1.3 Foreground Highlighting

The aim of the foreground highlighting is to reduce the search space that will be used in the next step, image parsing. Image parsing is a method that has a high computational cost; therefore it is important to reduce to search space as much as possible. However it is also important not to introduce more complexity while trying to get rid of another one.

In this step of ETH tool, 6 points in the upper body detected region of interest are selected. Those are points where the head, torso, background on sides and background that is above the head. Those points are selected because those are the parts of the body which have expected certain locations. The bounding box returned from the upper-body detector gives an idea.

Since the way the upper-body detected is trained is know, it can also be predicted where the head in that box is and the torso. The challenge is that arms could be anywhere. From those points 4 regions are selected, foreground, background etc. Then Grabcut method is used and from those points, the tool tries to model the color map properties of foreground and background in an image. Starting from the area where it is definitely foreground, Grabcut keeps expanding that area until it meets pixels which match the background color model it defined at the learning phase.

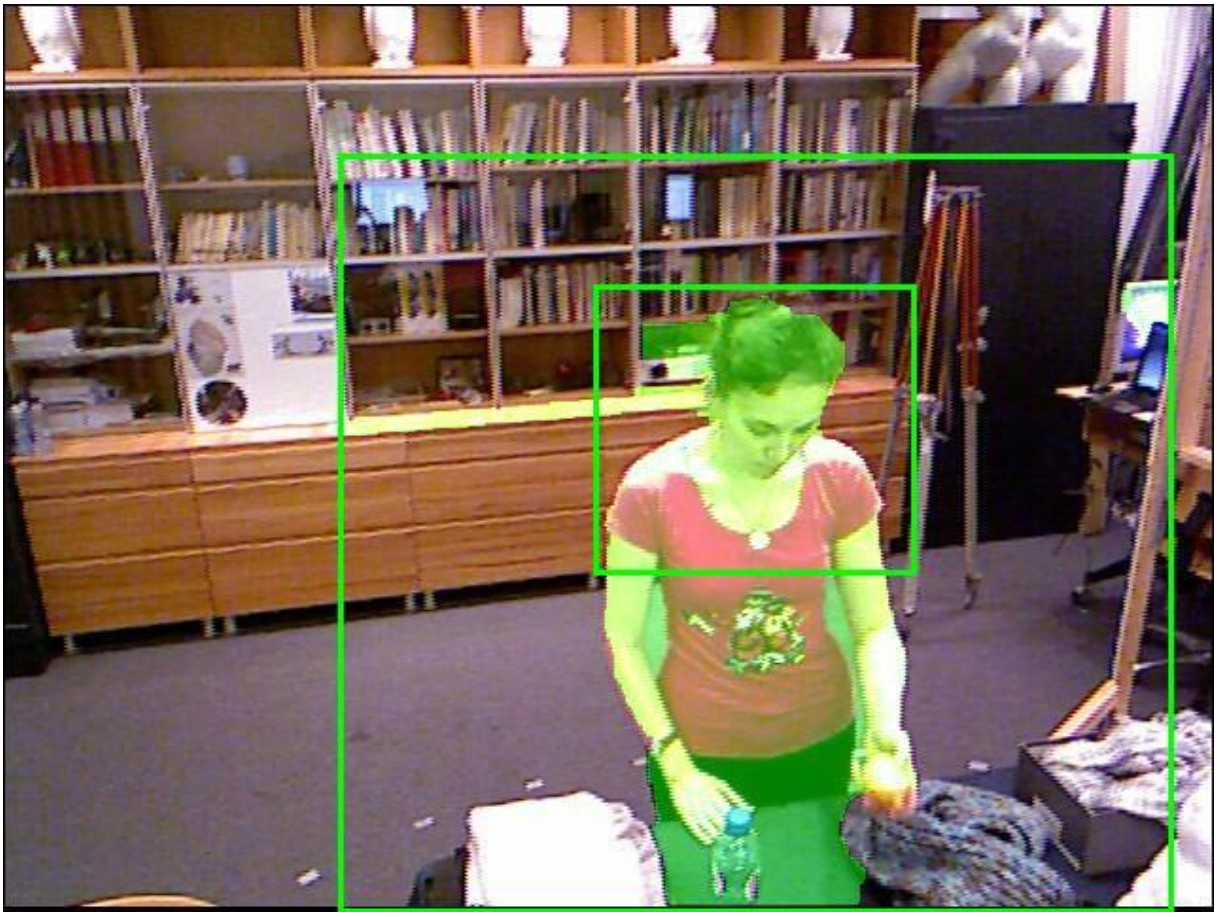


Figure 3.5: Foreground highlighted sample produced at the second step of the ETH Tool.

The obtained result from this step is an image mask where the background is roughly canceled out. However it is crucially important to note that not all background areas are taken out, but rather most of it. Therefore this step has relatively smaller computational cost however reduces one of the next steps.

3.2.1.4 Image Parsing

This step has in image input where the foreground including the person is labeled and the bounding box. The bounding box is no more needed since the foreground includes the target person anyways.

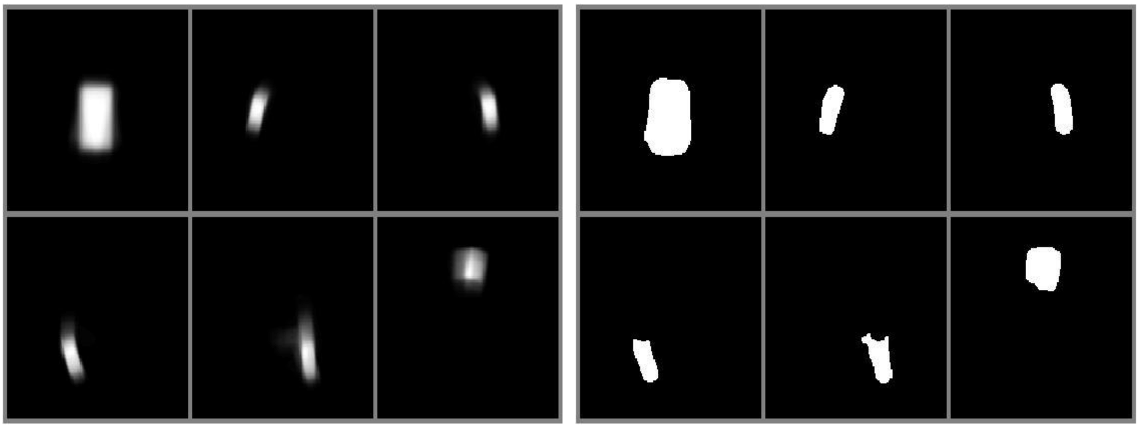


Figure 3.6: ETH Tool: segmentation priors for head, torso, upper arms and lower arms.

There are two parameters used here; the color models and edge map. The color models that were created in the previous step to find the background and foreground are passed to this step and also edges are used to find the objects, limbs in this case. Using segmentation priors from training phase, edge and color maps, and appearance models of limbs all together, system ends up with the stick locations probabilities and the sticks as well. In Figure 3.8Figure 3.8 on the left we can see the sticks with segmentation probability as well. On the right, there is the final phase where the stick locations decision is finalized.

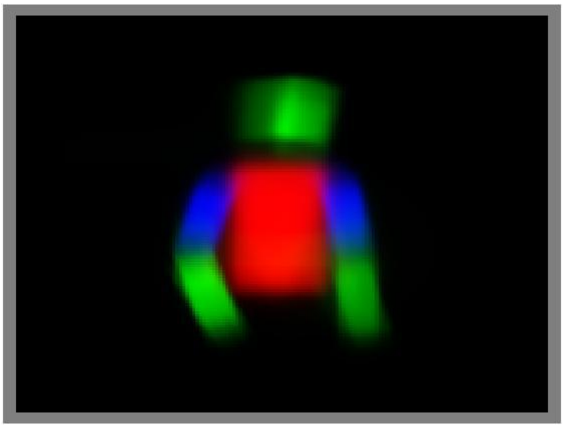


Figure 3.7: Segmentation priors put together considering the kinematic limitations.

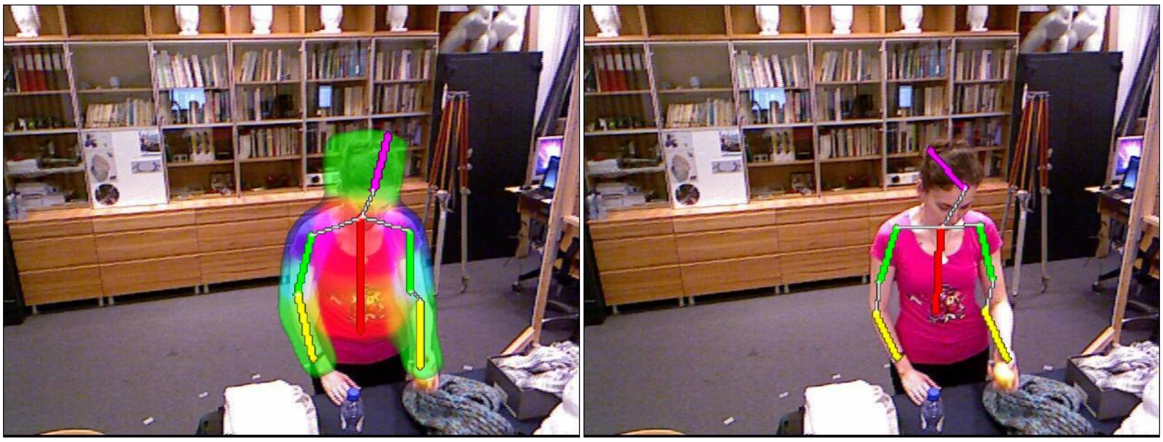


Figure 3.8: Image parsing and limb stick locations produced by last step of ETH Tool.

3.2.1.5 Capabilities

ETH tool is capable of detecting people in a scene and return the stick coordinates. There are a few other data it returns and it is capable of working in some situations where the problem is indeed considered difficult. The data it produces:

- ETH Tool returns the bounding boxes for the upper bodies in an image. It is highly successful for this step.
- Foreground area mask is produced. However this data cannot be really used as an individual representation of the image because it is rough estimation of the foreground/background area in the image. It is not really a silhouette.
- Stick location probability masks are returned. The areas where the limbs are most likely returned in a map. Limb sticks indeed are generated from that information.
- Limb stick coordinates are returned. Those are locations of the head, torso, upper arms and lower arms.

3.2.1.6 Limitations

There are a few limitation of the tool and also some problems that it sometimes faces and can’t overcome.

- The tool is designed for upper bodies and frontal view. In previous versions of the tool, it used to fail drastically when the person had a profile view. This is improved in the newer version yet still it is most powerful when there is a frontal view. It can work with almost frontal views as well but the performance decreases.

- The tool is not very strong in case when one or more limbs are not visible. In those cases it still detects some parts as those limbs. Besides that when the person is occluded with another object, it again fails because the color models are then learned from that object.

3.2.1.7 Requirements

The tool requires a few things for the optimum detection. When those requirements are met, detection performance increases and it is best to work with.

- The person needs to be away from the camera at least a couple of meters so that the whole upper body can be seen. If the person is too close and just the upper body is seen and the arms hardly fit in the frame, the tool fails.
- Person needs to be facing the camera or the back should be seen. Complete profile views should be avoided.

3.2.2 Kinect

3.2.2.1 Introduction to Kinect

Kinect is a device developed by Microsoft for the gaming console Xbox. In a very general sense, it is able to detect people in a scene, get their pose, track them and if calibrated, extract a skeleton representation of those people.

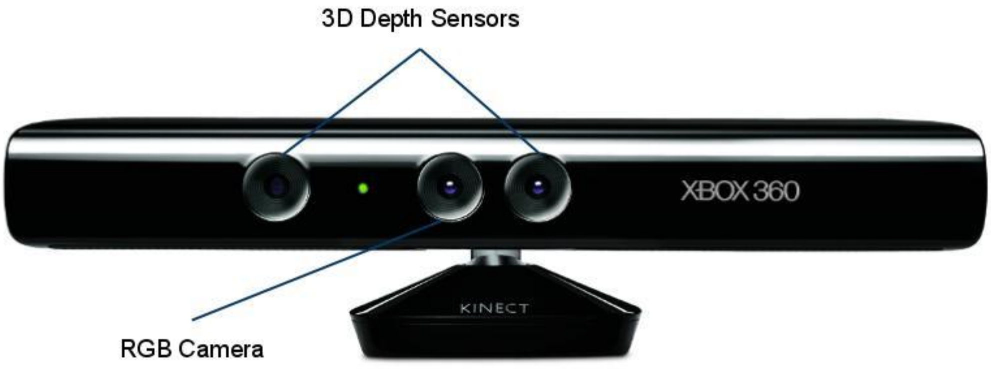


Figure 3.9: Kinect with sensors.

It has two IR cameras for depth sensing as well as an RGB camera. With the help of the IR cameras, it is able to extract the depth map of the scene, which improves the detection rate. It is quite robust against occlusion, frontal and profile view etc. Those are explained in details below.

In our case, Kinect can be used as a sensor, both for recording and for the action recognition task. There are abilities of Kinect that are quite beneficial to our task, as well as some limitations.

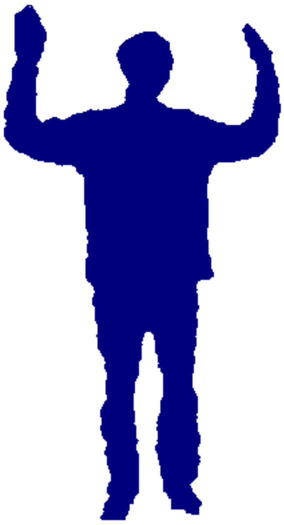


Figure 3.10: Pose required for calibration Kinect for skeleton tracking.

3.2.2.2 Recording properties

- Kinect can record 640x480 32-bit colors at 30 frames per second. Depth information of the scene is streamed with 320x240 16-bit depths at 30 frames per second as well.
- It can detect and track up to 6 people simultaneously.
- Observed data can be saved in a format called Oni. This is a file format created by the framework called OpenNI that enables to work with Kinect.
- When the calibration pose that is shown in Figure 3.10, is done, Kinect is also capable of detecting body parts and tracking them. This can be adjusted to whole body or only upper-body. Because we work with upper-body, we use upper-body configuration. Kinect is not capable of saving stick coordinates yet we develop our own software for this.

3.2.2.3 Generated Data

The data that is produced by Kinect and that can be retrieved have a few parts to it. Kinect is capable of generating depth, texture, user information and skeleton information in a scene and below we explain those in detail.

- **Depth Map**: Using the input from the IR cameras, Kinect is able to observe the environment in 3D and can extract the depth map of the scene. This way, occlusion problem becomes easier to solve.
- **Texture Map**: Kinect also gives the RGB color map of the scene as an output. That can be recorded just like any RGB camera would do.
- **User Map**: Kinect gives an output of binary images that would include the detected people in the scene. They are separately hold in different images per person and can be reached as soon as a person is in the scene. This is the way to get the silhouettes in the scene.
- **Limb (skeleton) sticks**: If the configuration is done, Kinect is able to detect the limbs of a person and extract them as information of the scene. However to be able to do that, the person has to stand in front of the Kinect, with whole body visible, and has to stand for a few seconds with both hands in the air. This pose is shown in Figure 3.10. Since our context is stores and it is not really possible to ask customers to do that calibration in real life, this data can be used in tests but a method for action recognition has to be produced without the limb detection. For calibration the following steps have to be followed,
    - Once the person is on front of the Kinect sensor and whole or upper body is visible, depending on which mode it works on, Kinect should detect the person and the silhouette image should be visible on the person.
    - If Kinect is following the person and the silhouette image is visible, then both arms are put up as shown in Figure 3.10, and wait till Kinect notifies that calibration is done. This process might take from 10 seconds or a little bit more depending on the positions of the Kinect sensor.
    - Once the calibration is done, Kinect tracks the limbs. If the person goes out of the frame but comes in really quick, the tracking continues. However if the

person stays out of the frame for too long, once she/he is back, Kinect recognizes that person as a new user and the calibration needs to be done again.

3.2.2.4 Limitations

Although Kinect is highly talented, there are still some limitations as well. Below we explain those.

- Kinect is not capable of detecting the skeleton of the person without doing the calibration.
- People in the scene need to be in the field of view of the camera, and that will be explained below.
- The Kinect sensor cannot save all the data it produces. It can save depth and texture information however it cannot save skeleton information.

3.2.2.5 Requirements

Working efficiently with Kinect requires certain things. Those are mostly related to the position of Kinect and the distance of the person from the sensor.

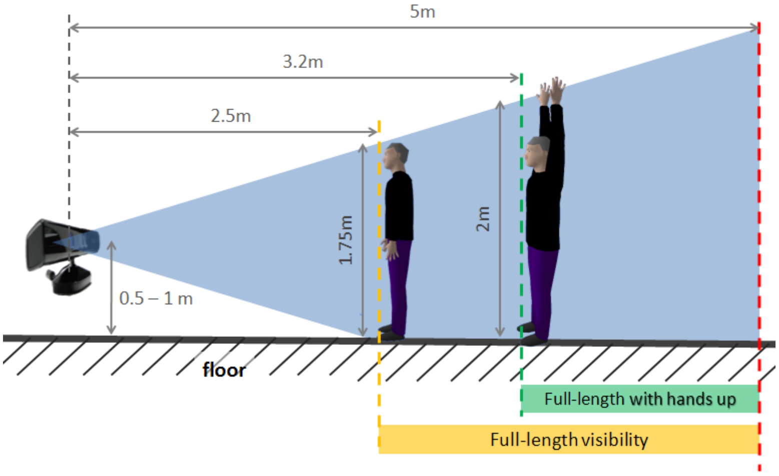


Figure 3.11: Kinect capture area: side view.

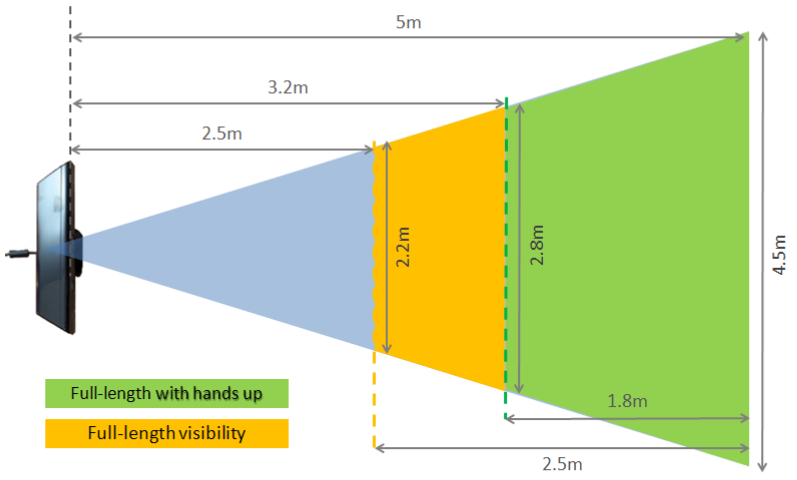


Figure 3.12: Kinect capture area: top view.

Kinect does not require to see the whole body if the tracking is configured as the upper-body only. However whatever that is chosen to be tracked, needs to be in the field of view of Kinect. In Figure 3.12 and Figure 3.12 we can see the field of view of Kinect. People are required to be away from the Kinect about 2 meters but in our case since the arms might go out of the frames, they might need to be even further away. Horizontal field of view of Kinect is 57 degrees while the vertical is 43. The depth sensor can work within a range of 1.2 to 3.5 meters.

Kinect is however not indeed introduced for developers by Microsoft but for the Xbox gaming console. Therefore it is indeed impossible to get the mentioned data from Kinect. However third part companies have developed software to make Kinect work with a PC and develop applications on it as well. Microsoft approves those open source frameworks. In our experiments we use the OpenNI framework which is designed for Kinect and similar sensor that can make it work when connected to a PC and get the data it generates. On top of that, we produce our own software for saving the data and processing it.

3.2.3 HTK Tool

HTK, Hidden Markov Model Toolkit, is software that is mainly designed to implement HMM applications for speech recognition. It has indeed much functionality for data preparation, processing and classification, as stated in the manual [30]. Yet those parts are done by our software so we use the HTK for recognition and cross validation.

The tool HVite of HTK uses the Viterbi algorithm for recognition. We use the Viterbi algorithm recognition, with left-to-right topology, using several numbers of states and Gaussian mixtures. There are also other tools for analyzing the data, yet we also do that with our own software.

3.2.4 PRTools

PRTools stands for the Matlab Toolbox for Pattern Recognition. Toolbox includes many algorithms and functions for several pattern recognition tasks. The toolbox is also used in [31] and this book serves like a manual besides the actual manual as well.

Toolbox supplies the user with feature extraction, analysis, classification and evaluation tools. We use this tool for the classification implementation with Support Vector Machines, Linear Discriminant Analysis and Nearest-Neighbor Classification steps.

3.3 Developed Software

Implementation of our system is a software toolbox. The tools we mention above are additional packages that we build our software on. Figure 3.2 shows the work flow of our software and here we explain that and how we use the mentioned tools in a bit more detail.

Our capturing device as mentioned before is the Kinect. OpenNI framework enables individuals to use Kinect on a PC as well. Kinect is originally designed for the Microsoft gaming console Xbox. OpenNI is able to pull some of the information that Kinect actually produces and we process that data. As mentioned above, Kinect generates depth, silhouette and if calibrated limb coordinates. However OpenNI framework is not capable of saving all the data. Here we introduce the capturing and saving step of our software.

The outcomes of our capturing console are shown in the figure above. We store all those outcomes in separate related files per frame.

In the later stage, our software extracts the features from those captured and stored files. At this step we use the stored files mentioned above and the ETH tool as well. RGB frames are processed with the ETH tool to give the limb stick coordinates of the frame. Our software processes those and produces the features we need.

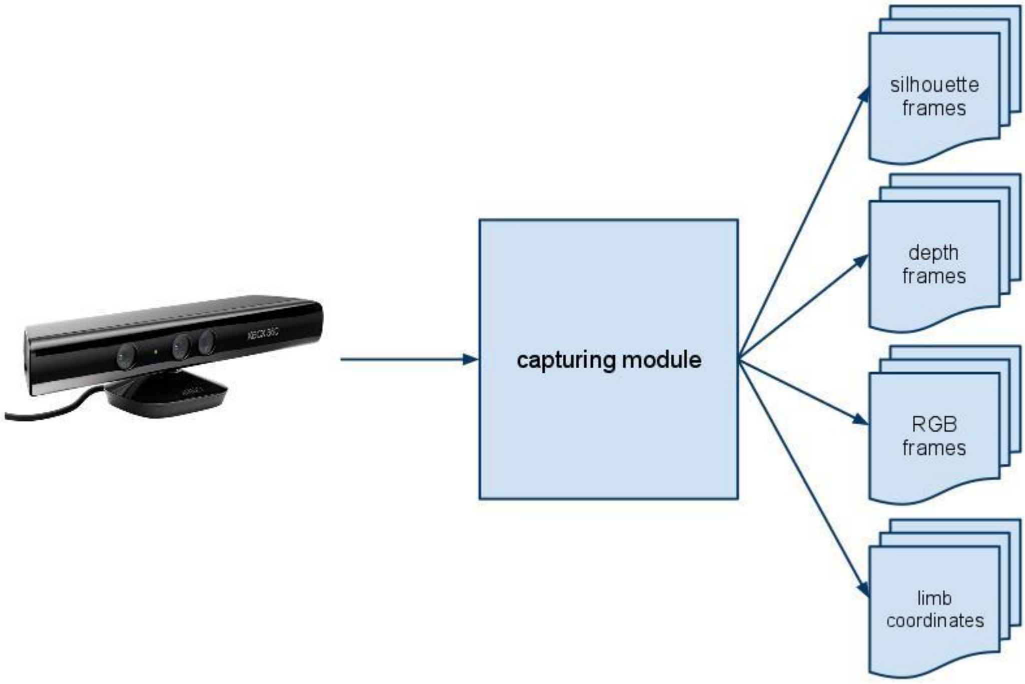


Figure 3.13: Capturing Module of our software framework; Processing Kinect outcome.

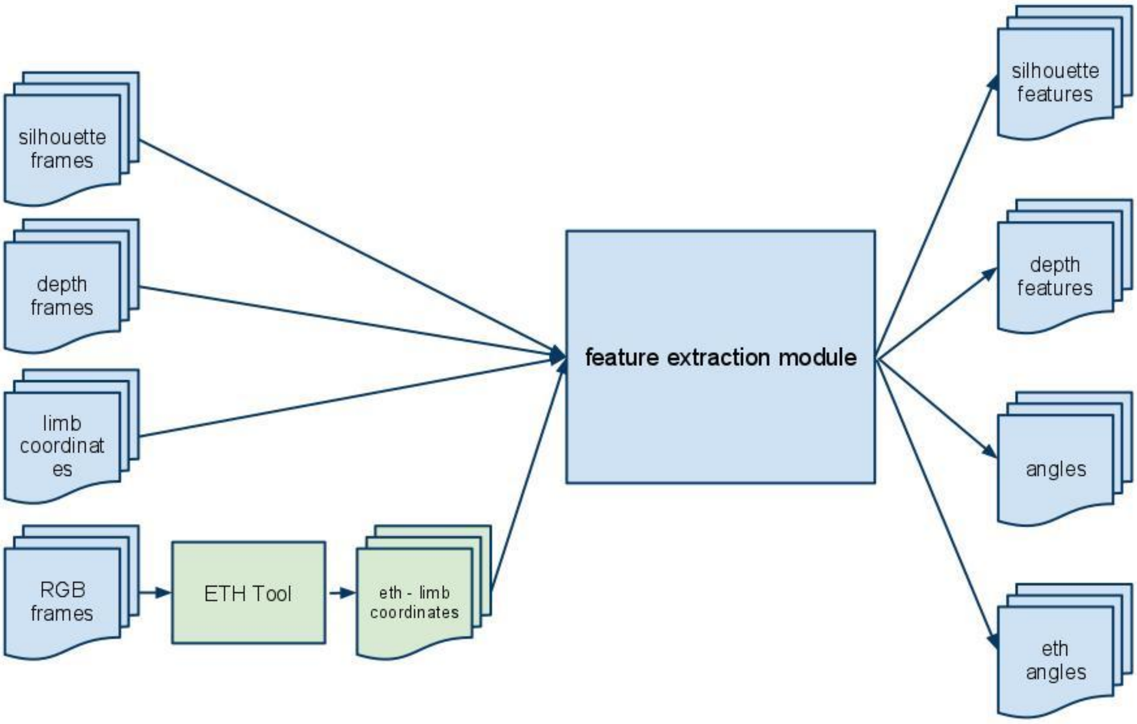


Figure 3.14: Feature extraction module of developed software.

Our feature extraction module extract silhouette and depth related features and angles. ETH tool produces new set of limb coordinates and we also convert those to angles.

Next step is classification and we have a few main approaches for that step. All those steps of our software will be now explained in detail below in each chapter. However for the experiments and implementation of our system, we definitely need to capture data and establish a dataset with which we can also evaluate the performance of our system. In the next step we explain how we construct a database.

4 Data Acquisition

4.1 Pre-Processing

First step of the implementation requires capturing the data and processing it. In the previous section we explained that step. Next step is collecting the data for experiments but there are things need to be done before starting that step for clarity.

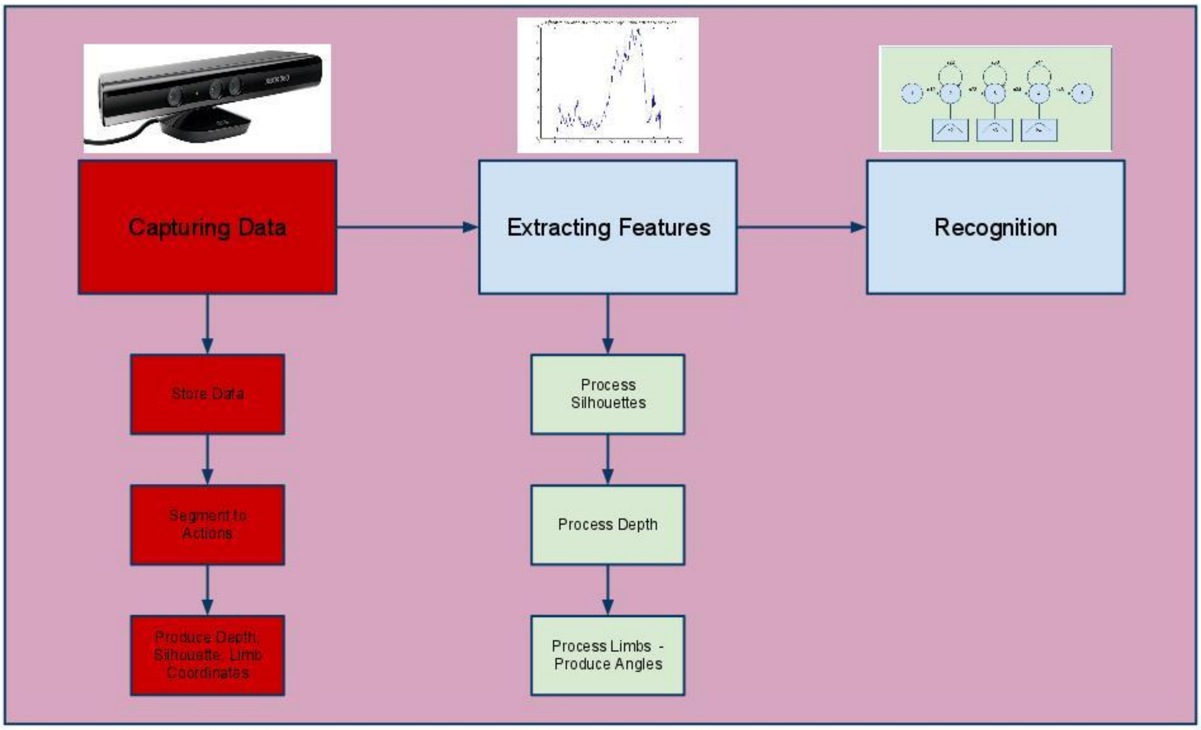


Figure 4.1: Capturing step of our system.

Our aim is to detect actions in a video recording. We use our software to extract the needed information from the capturing sensors but this can be considered raw data. To make the recording more meaningful, here we should explain what an action is and what the actions we aim to detect mean in detail. Below we go into detail of our target actions.

4.1.1 Action Definitions

Although we here mention many names of actions, it is crucially important to clearly define what those actions are. It is inevitable that just a name of an action, could mean different things to different people and interpretations. More than that, it is also known that not every action is done exactly by everyone. Every individual person has their own way of doing 'things'. For example, if the question “What is browsing?” is asked to a couple of people, different responses might be obtained. Therefore for the clarity and consistency of this database, we defined those actions as clear as possible also letting them free enough to be person specific and 'characteristic'. In the following sections, we define what we mean by the actions that exist in the recordings and what their general outlines are. In those actions, what can be different in between samples is also explained.

There is one important principle about the actions, when we do the sampling and decide the set of frames that include an action. Every action starts with the neutral pose. Later in the sequence there is usually an apex pose and the final frame is again the neutral pose. Neutral pose could change from case to case. A person could be holding a bag in his/her hand, so we can call this a neutral pose. So the action sequence of frames includes the last frame that has that pose as the starting frame of the action sequence, the following frames where there is the action itself and the first frame that includes that neutral pose as the last frame of the action sequence of frames.

4.1.1.1 Browsing

When somebody is browsing is in our experiment setting, the person is interested in the products on the table however they do not really focus on one product. Basically they go through the products on the table and see if there is any in them that would really attract their attention. This action is expected to occur more often when in a store there is a table, or in more general sense a 'stage', with a lot of products on it. This could also be a deep basket where a lot of products are stuffed. The person randomly, or in order, goes through the products on the table, or in the basket.

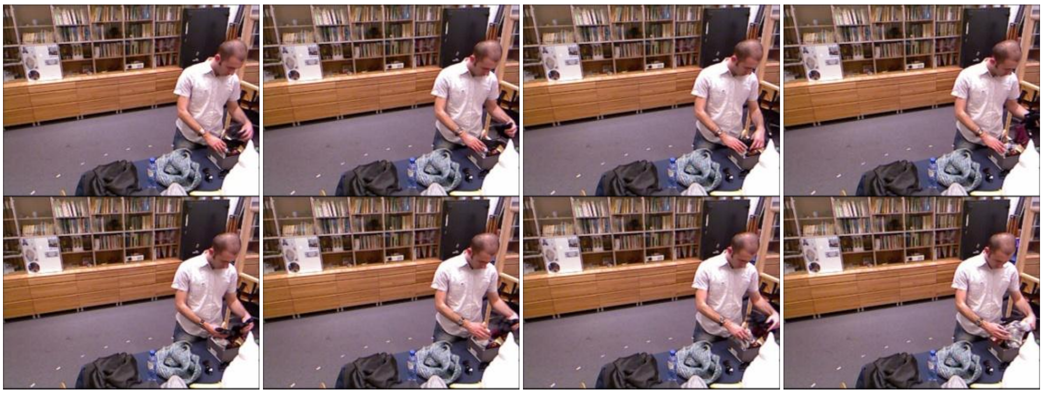


Figure 4.2: Characteristic frames from a browsing action sequence.

In a general sense, the expected movement is one or both hands somewhere around the surface of the table, holding products or moving them side to side to see other products.

4.1.1.2 Examining

In a shopping environment, people mostly pick up a product and examine it in a closer view. They look at the product in more detail and focus on that product. This doesn't have to be a single product though. People sometimes pick up two products that are similar and compare them. They examine both in a closer look and put them next to each other and try to see which one is more likely for them to buy.

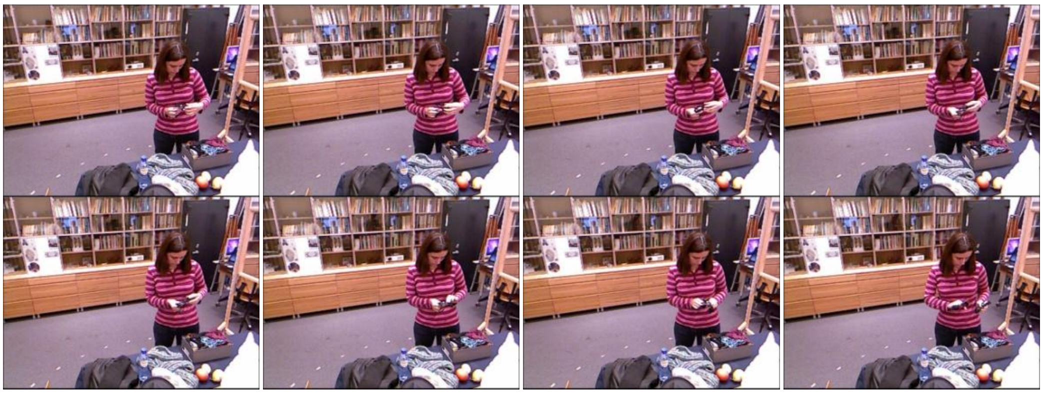


Figure 4.3: Frames from an examining action sequence.

For an examining action, the person is holding a product in his/her hands, around their belly or chest, but in more general in front of them and is looking at the product. Just like in browsing action, again there isn't huge volume of movements yet more like small movements. Those movements are expected to be especially with hands or arms since they will be holding and moving the object to see it from different angles. In a sense of silhouette information for example, the expected movement is especially focused on arms; arms will be making small movements so that the width of the silhouette around torso will be changing. Whole body is not likely to move, yet that does not mean the person is absolutely standing still either.

4.1.1.3 Trying on

Especially in a clothing store, people put on the products on themselves to see if it would fit or if it would look nice on them or not. For this action though, the object that the person is interacting with could make a lot of change. People could be trying glasses, a jacket, a scarf etc. For those mentioned 3 products for example, the movements of the body is quite different. However one unique property for all those products with the trying on action, it is certain that there will be a relatively high volume of movement. With glasses for example, the person will only be putting his/her hands up to his/her head and then put them down. Yet with a scarf or a jacket, there will be huge movement around the upper body, more like around the chest. This is the characteristic property the trying on action has.

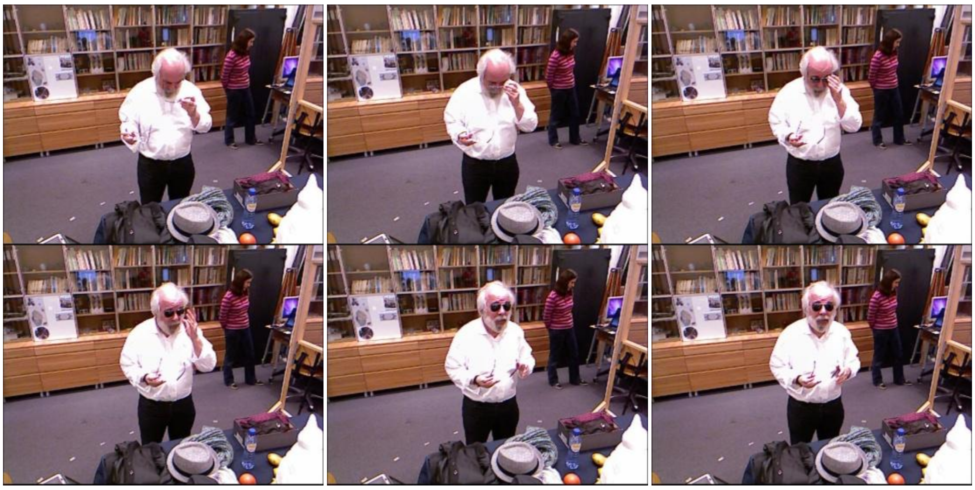


Figure 4.4: Frames from a trying on action sample sequence.

4.1.1.4 Picking

Picking up a product is a very common action for all kinds of stores. In any kind of store, either to examine closer or to buy it, people pick up products from the shelves or tables. This action is expected to be pretty much similar for different people too. There might still be individual person dependant properties, yet the general outline of the action is quite clear. Again the action starts with the neutral pose, then the person leans to a table or a shelf, holds a product and then goes back to neutral pose or the examining action pose. The critical action is therefore carried with arms, one or both of them.

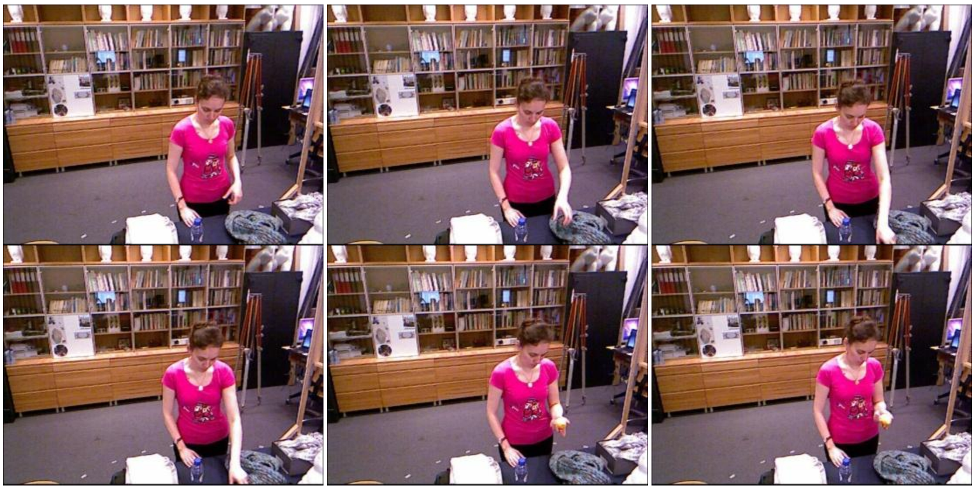


Figure 4.5: Frames from a picking action sequence.

The term we use here as picking has a general meaning though. A person might be picking up a product from a table, putting it back, putting the product in a shopping basket or could be picking up a product from the shopping basket to put it back etc. Those could be different visually, yet the characteristic movement is common for all of them; one or both of the arms leaning to a point and then coming back.

4.1.1.5 Waving

In a store, people might be asking for help from the assistant, calling to their shopping partners etc. In those cases, people usually wave to someone. This action is observed to be quite common and characteristic as well. Although again this action could be characteristic for every individual, the critical and common property of it is pretty much constant. Person starts with the neutral pose again, raise one of the arms, hold it up and move around a little bit, then put their arm back. This is the critical information we are looking for and it again is also carried with the arms.

4.1.1.6 Driving Shopping Cart

In all shopping stores, there is some kind of a shopping cart. We call it a shopping cart but it could be a shopping basket too. People walk around with those shopping cars, put stuff in them, take out stuff from them etc. Interaction with the shopping car is significant because it usually is the final phase of the interaction with the product. If the person likes the product and decides to buy it, they put it in the basket. If that action is detected, the interaction with the product could also be finalized in terms of automated information generation. That information as well could be beneficial for a lot of things; for information gathering for brands, log information for the store itself, security checks etc.

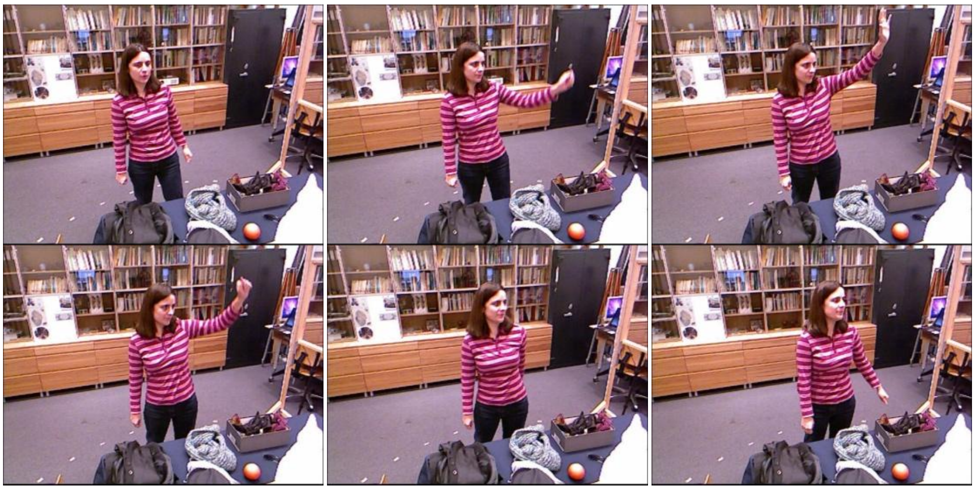


Figure 4.6: Waving frame from an action sample sequence.

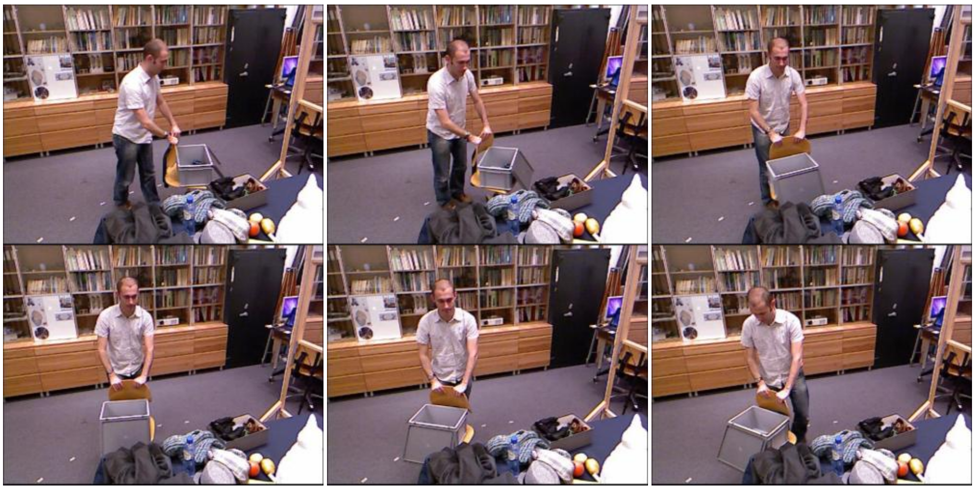


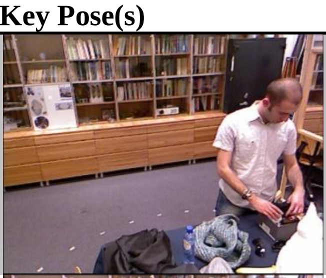
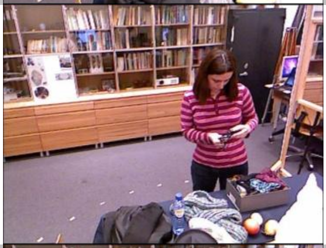
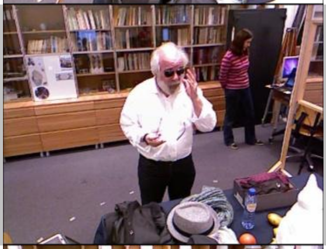
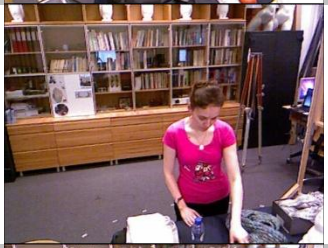
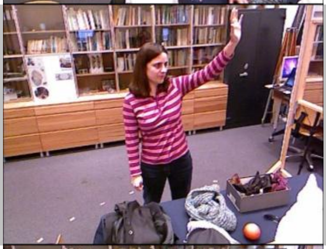
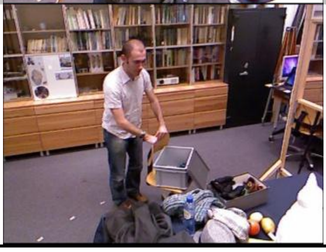
Figure 4.7: Sample frames from a driving shopping cart action sequence.

With ‘driving shopping car’ action, we simply mean people holding a shopping cart in our case, and walking around with it. There is not much movement expected with the arms, but rather with the whole body moving around, hands usually leaning forward and holding the shopping car and pushing it.

4.1.2 Actions Review and Relations

We define the basic actions as above. It is also important how those actions are linked to each other. Defining basic actions is indeed important yet is also important to know how those actions are linked to each other. First we give a short review of the basic actions in the table below.

Table 4.1: Actions with short descriptions a frame with a characteristic pose.

Actions	Description	Key Pose(s)
Browsing	Person goes through the products on the table with hands.	
Examining	Person holds one or more products in hand, looking from closer.	
Trying on	Person putting on a product, like a jacket, scarf, vest etc.	
Picking	Person leans to the table, picks a product, pulls hand back.	
Waving	One or two arms in the air, person waves at some direction.	
Driving Shopping Cart	Person holding the cart handle and pushes or pull.	

By relations between those actions, we mean how they might be followed by another. We usually don’t strict that order of actions happening however in certain scenarios they indeed might be followed by each other. In the table below, we present a table with actions as columns stating if or not they could follow each other. Actions in the rows show the first and actions in the columns show the following, in other terms second, action. If that order of actions is possible, then it is stated as ‘likely’. Combinations that are stated as ‘very likely’ are the most common combinations. Those combinations are actually that order of actions would normally be expected. Combinations that are not really possible, and that actually require another action in between, are stated as ‘unlikely’.

Table 4.2: Combinations of actions following each other.

	Browsing	**Examining**	**Trying on**	**Picking**	**Waving**	**Driving Shopping Cart**
Browsing	likely	unlikely	unlikely	very likely	unlikely	likely
Examining	unlikely	likely	likely	likely	likely	unlikely
Trying on	likely	likely	likely	likely	likely	unlikely
Picking	likely	very likely	very likely	likely	likely	unlikely
Waving	likely	likely	likely	likely	likely	unlikely
Driving Shopping Cart	very likely	unlikely	unlikely	likely	likely	likely

For better understanding of the table above, we can focus 3 samples. Picking followed by examining is showed as ‘very likely’. It is usually possible that people pick a product from the shelf or table to indeed look at it closer. This is a sample of choosing a product and examining, to see if the person would like to purchase that or not. Examining followed by trying on is shown as ‘likely’. It is not always expected from people to put a product on after they examine it, yet it is also possible that people might put something on after a closer look. Unlikely combination of driving a shopping cart followed by trying on is quite an obvious example. To put something on someone needs to have something in hand. Driving a shopping cart needs to be followed by picking action at least to have something in hand to try on.

4.2 Experiment Design

ETH Tool works with the upper body and our assumption about actions is that most of the information about the action is explained by the upper body, particularly in this case of shopping context. Therefore our experiments are designed to record the upper bodies of the participants and their interactions with the products as well. We set up a “shopping” environment, which contains a table and products on it. This setting is pretty common for most types of shops, where there is a “stage”, a table, a hanger etc., where there are products and people in the shop interact with those products. The choice of a table is to emphasize and make the interaction with products more visible. Even more than that when a table with products on is used, participants and their upper body movements are more obvious to the sensor.

On the table, we have a bunch of products, like scarves, vests, gloves, glasses etc., pretty much products which one would find in a textile store. To make it as general as possible, the products are carefully chosen. Even if the products might differ in different types of stores, the interaction would be as similar as possible.

The aim of this setting is to let people interact with those products, go through them and see what they would like, even try them on, examine them closer in more detail etc. Those actions are what one would be expected to do in a textile shop. If any random clothing shop is observed, it can clearly be seen that those actions mentioned above are what people most commonly do.

To make sure that we have a reasonable number of samples for all actions, we designed a few scenarios and kindly asked people to follow those. However, it is also important to have data as realistic as possible. Therefore the scenarios are not so strict that people would become “unnatural” while doing them. They are pretty much general outlines of what they are expected to do, and once they stay in those outlines, they are free to do whatever they want to do. In most of the scenarios, even those general outlines are quite weak not to influence people, yet some scenarios have some instructions to make sure we have enough samples for each action. Yet those scenarios where there are relatively more strict instructions were recorded later than the free scenarios.

There are a few common movements in a store, and we obviously expected a number of basic actions like the following ones:

- Picking a product
- Examining a product closer
- Browsing and going through the products on a table or on a shelf
- Putting a product back on the table
- Trying on a product, usually a cloth, hat or glasses etc.

- Compare several similar items
- Putting a product in the shopping basket or shopping cart
- Taking something back from the basket or shopping cart and putting it back on the table
- Ask for help/assistance

Those actions are expected to be recorded, and further investigations about those actions are made and for the sake of simplicity, number of samples and frequency of occurrence of those actions, some actions are selected from those and are put in the database. That procedure is also explained in later sections of this chapter.

Kinect is the main sensor used in the recording. As mentioned in previous chapters, Kinect is capable of producing the depth and RGB information of the scene and if the configuration is done, it can also produce the limb stick coordinates of the participant. Every participant was asked to do the calibration at the beginning of the recording so that we could also have that information as well. The results and the observations about the experiments are explained in the last section of this chapter.

4.2.1 Recording Environment Setup

In this section we explain the setting of the recordings environment. In Figure 4.8 and Figure 4.9, the general setting of the recording room can be seen. The configuration shown in the figure is restricted to the requirements of the Kinect sensor. As seen in the figure and mentioned earlier, we have a table that has several products on it. We have the participants and the Kinect sensor is placed right in front of the table so that we can have a frontal view of the person. We also have a mirror in the room in case the person would like to see how a product looks on them, which is absolutely a common event in a clothing store.

4.2.2 Equipment

The equipments we use in this experiment are listed in the list below.

- 2 Kinect as sensors
- Table
- Products: Vests, scarves, jackets, glasses, gloves etc.

- Shopping cart
- Laptops for recording the data Kinect produces.

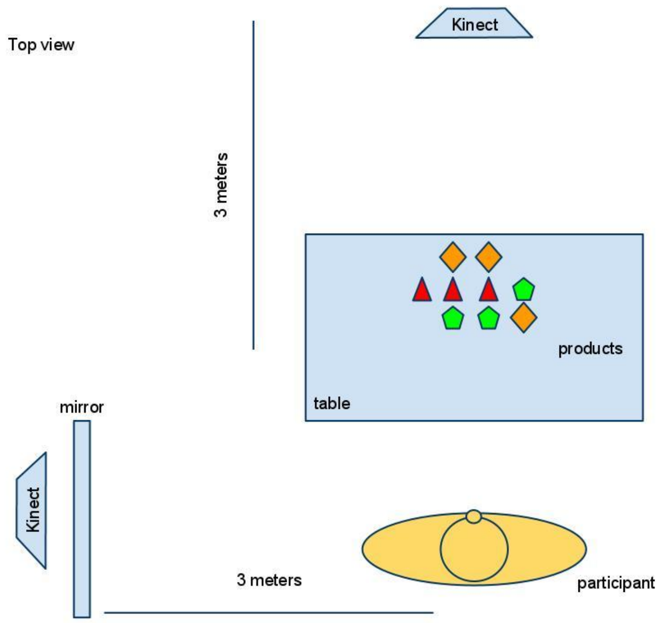


Figure 4.8: Top view of the environment setting of the recordings with the Kinect sensor.

4.3 Recordings

For recordings we do include a few scenarios which we introduce to the participants so that they can follow them. Below in this section, we introduce those scenarios.

4.3.1 Scenarios

As explained before, our scenarios do not consist of strict instructions that actually wouldn't be beneficial for the sake of spontaneity of actions. Rather than giving strict instructions we give participants general outlines about what we expect them to do.

When a shopping store is examined it can be seen that there are very common events going on. An expected event for example in a store would follow like this: person enters the shopping store and they look around the shop. Their first aim is to observe what the store has and if there is any match with what they want and what the store offers. They walk around and at a table or a shelf they stop to have a closer look at the options. They browse through the product on the table and try to see if any of them would be in their interest. When they find something interesting enough, they pick it up for a closer look. If they are still interested after that closer examination, and if it especially is a cloth, they put it on to see how the thing looks on him/her. Later on they even decide to buy the product or put it back.

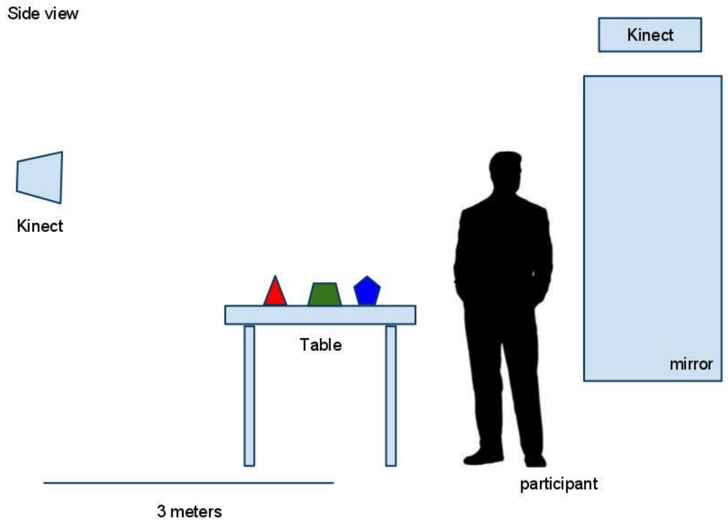


Figure 4.9: Side view of the environmental setting of the recordings with the Kinect sensor.

In the common event explained above, the actions browsing, picking, examining, trying and putting back or putting in the basket can be seen. It is quite obvious that there are quite a lot of common scenarios like that and they include the actions we defined in the previous chapter.

We aimed at recreated those very common scenarios in a shopping context so that we could observe and collect data on the actions we defined. We defined 6 scenarios that either have many of the basic actions or just a few of them. In either case they are all very common and expected flow of actions. Below we explain those scenarios and the basic actions that are expected from those scenarios.

4.3.1.1 Scenario 1: Browsing

The task given to the participant is to check what is on the shelf and look through them if there is anything specific attractive to them and they would consider buying it.

In this scenario we expect people to go through the products on the table that are mostly in a cluttered setting. This way people need to touch the product, move them side to see another one etc. The expected actions from people to perform are browsing and maybe picking.

4.3.1.2 Scenario 2: Pick favorite item

In this scenario people look at the products and pick one that they would seriously consider buying. Therefore people need to see the product on the table and pick one or more of them and examine them more closely to check if they would purchase it. We also ask them to put the products that they would like to buy in the shopping cart.

Expected actions in this scenario is maybe browsing, picking up a product, examining, maybe put back on the table or put in the basket if the person decides she/he would buy that object.

4.3.1.3 Scenario 4: Evaluate products

This scenario requires people to compare products and this is something in stores people mostly too. People usually pick up two or more similar objects and examine and compare them closer to see which one of those they would prefer.

We expect people to browse the products a little bit, pick up products and examine but in a comparing manner. Those are the expected actions to be obtained in the recordings.

4.3.1.4 Scenario 5: Ask for assistance

Mentioned earlier in this report, human assistants are a fact in stores and people want to get their help. Participants have picked their favorite products, examined them closely so far and they might have questions. Therefore we ask participants to ask for help from the shop assistant if they have any questions or are in need of help.

Expected outcome of samples from that recording would be the waving or a similar action that people perform to ask for help.

4.3.1.5 Scenario 6: Purchased item in the shopping cart

In the previous scenarios people have examined products, asked their questions and decided to buy a couple of product and put those in their shopping carts. Here in this scenario, we ask people to go around with their shopping cart and also evaluate the products they have in their cart. We ask them to double check the item they purchased if they really want those or not.

In this scenario we expect people to drive the shopping cart, browse the items in their cart and maybe pick them from the cart and put product back on the table.

4.4 Post-processing

4.4.1 Annotation and Sample Selection

When the recordings were made and the raw database is obtained, it is important to segment those recordings and get the actual frames that will be needed. Those frames should be including the moments where the above mentioned actions occur. Yet for this segmentation process, there should be some principles decided so that the samples are all equivalent to each other. That is what a consistent database should be. That way the methods that we are going to use will be working without initially introduced errors originated from the frames not including the actual action, double including an action, or any other similar problem.

4.4.1.1 Beginning and End of an Action Segment

As also mentioned in previous sections, the definitions of the actions are made quite clear. All actions start with the neutral pose, include the following frames where the action occurs until the first frame where the action ends and the neutral pose is observed.

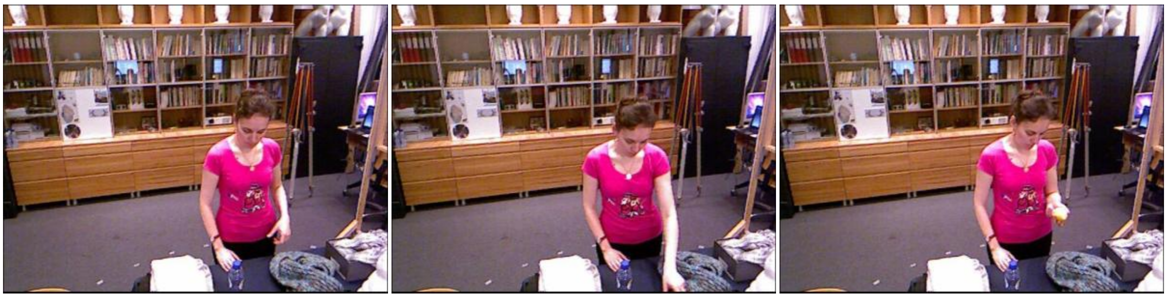


Figure 4.10: Sample of starting and end frames of a picking action. First frame pose neutral, next frame is the apex pose; last frame is the neutral or the beginning of next action.

Neutral pose we mention here could be a couple of different poses. One of them is that the frame where the person is really standing still with two arms hanging from both sides of the body and pretty much doing nothing. If this frame is followed by the frames where the action is happening, it is taken as the starting frame of the action. Taking picking action as an example here, the person is standing still at frame *i* and in frame *i+1*, at least one of the arms start leaning forward. This move keeps happening in the following frames, until the frame where the arm is at the apex of reaching, and probably holding the product. However the sample is not ended here. Following that climax pose, there is a move for the arms going back. As we mention at the beginning of this paragraph, the neutral pose doesn't have to be the standing still, but could be the first frame of the next action as well. In most of the cases, picking action is followed by examining action, for example. In that case, the last frame for the picking sample would be the first frame where the arms are again back close to the body, and the examining action is just about to start.

There are also actions in which actually there is none of those 'apex of peak poses, like in the case of the picking action. For browsing action for example, there is no climax of a move, location of arms, but rather a continuous repetition of patterns of the body. In browsing action, the arms are around the table going through the products. In those cases the decision of the starting and ending frames of the sample doesn't really depend on the neutral pose. In those cases, we make sure that there are no frames in the sample where the poses in those frames could be linked to another action. In none of the browsing actions, we make sure that there is no 'leaning forward or back' move of the arms, which is expected in the picking action. In those frames, the starting and ending frames are also pretty much the poses that would be expected in the browsing category.

4.4.1.2 Eliminating and Filtering Bad Samples

Obviously in the recordings produced, there are many samples and instances of those actions we are looking for. However it is important to make sure the samples selected to be put in the database are 'clear' and good samples to construct a good database. This does not necessarily mean all the samples are really perfect and includes frames exactly expected poses though. This is not good because the system will be trained and tested with those samples and the system should also be sensitive to different implementations of actions by different and also to common problems and errors that could happen in a recording. Otherwise the system would really rely on perfect samples that would fail even with a sample that has an error in it that could actually occur quite often and could indeed be solved. So overall, we really need to have 'good' samples for a good database.

As mentioned above, there should be principles to select samples to put them in the database. There are a few principles followed in the sample selection phase. Below in the list, those principles are explained.

- False silhouette area: This principle is especially made for Kinect because the methods that we will explain later rely on silhouette information. Although it is not known for sure, Kinect apparently uses depth information to extract the silhouettes. There are a lot of methods in the state-of-art approaches that follow that procedure. Yet in cases where the person is interacting with a product or the table itself, the hands or arms are so close to each other that some parts of the table are also labeled as the person and are added to the silhouette. Although this could be one of those commonly occurring errors that actually should be taken into account a solution should be found, comparing to the number of samples in total, the rate of those samples are not that high. Therefore those samples are decided to be taken out not to introduce initial errors to methods relying on silhouette information.
- Out of scene: A camera definitely has a field of view and sometimes it is possible that the person might be out of it. This could be solved with carefully placing the sensors and some other ways, so it really isn't a problem that should be solved with an algorithm. Therefore those examples are also filtered.
- Limb detection failure: As will be explained later, Kinect failed to detect the limbs in a reasonable number of cases because of reasons that will be analyzed and explained in later chapters. Those samples are not permanently taken out, yet they are not used for the methods that strictly rely on the limb stick locations.
- Plain bad sample: As normally would be expected, there are those samples where simply the recording, person tracking etc. fails. Those samples are also taken out since person tracking is really not the main part of our research but rather the action detection itself.

4.4.2 Conclusions on Experiments and Data

Recordings are made, samples are chosen and the database is constructed .Yet during this procedure and after the database itself is in the latest version, there are some observations about the recordings themselves, actions and different implementation of actions by different people. Here in this section we explain those visually observable properties of the actions and the recordings. Besides that the database is explained and introduced in numbers as well.

4.4.2.1 General Observations

Different people could do the same action in a different way. Therefore there are definitely person dependant properties in the samples and recordings. Those are to be solved with the method yet there are those problems in the recordings that are visually observable that could influence some errors on the results. Here in this section we mention those commonly seen problems. Those are to be addressed in the method created in the next chapter.

Most of the problems that we explain here are really related to the type of recordings and context we have. Our research focuses on shopping context, so obviously there are specific problems that should be addressed. Those problems are not that common in Weizmann database [18] for example.

4.4.2.2 Occlusion

Occlusion is a very common problem that could be seen in any kind of dataset and recording. It is basically the blocking of an object by another. In our case it is important since part of the body could be occluded and it could actually be carrying the critical information about the action. For picking action for example, if the arm involved in the action is not visible to the recording sensor, it is not possible to detect the action. In Weizmann and other similar datasets there are also occlusion problems and there are a lot of methods used to solve this problem. In our case though, we have a huge amount of occlusion and specific types of it.

In a shopping store, in a real case, people never face only at one direction. They walk around the store, go to different locations etc. Yet even if we take a table in a store and place the sensor behind it, still people might turn around. In Weizmann dataset for example, all the recordings are done consciously, making sure the participants face the camera. Yet in a real life shopping store setting, that is definitely not the case. Therefore in some samples, there are a lot of occlusions.

Besides the orientation of the person in the scene, occlusion is originated also from the products. There is a lot of interaction with products in a store, people pick up those, hold them up, turn them around etc. In most of those cases, the product would occlude most parts of the body and even the limbs. Especially for the trying on and examining actions, and if the product is huge like a jacket, then there definitely is occlusion. For the browsing and examining actions, hands are in front of the body and in the silhouette information arms are not visible yet arms are quite critical because they carry the action information.

Following those observations about the occlusion, it should be noted that a method that is aiming for a shopping context should definitely consider the occlusion of the arms and also them being in front of the body.

4.4.2.3 Depth and Interaction with Products

As also stated in the previous section, Kinect uses depth information to extract silhouettes in a scene. Most methods also use depth information to extract several features. In our recordings and samples, it is seen that there is a serious problem often seen.

In a scene when a person is interacting with a product, or a big object like a table, the hands of the person and that object have the same depth at that moment. In those cases if the method does not use color information or the contours, it is quite likely that the object could also be labeled as the person. In our recordings, there is a table that would normally be the case in almost any kind of shopping store, and people interact with it. This mentioned problem occurs in several frames of recordings, where the table is also labeled as the person. When those frames are examined in more detail, it could be seen that the depth value of the table and the person’s arm are equal or relevantly very close. In this case, a lot of methods fail if particularly the method relies on silhouettes.

Kinect uses silhouette information to detect limbs as well, but not in all frames. Once the limbs are detected, although Kinect is a black box and how it works is not known, it is highly possible that there is limb tracking after that point on. So it probably uses the color map of the limbs as well. In those cases where the table is also labeled as the person and is included in the silhouette, it is observed that in some of the cases the limb detection fails but in some it is good enough. Although the limb detections that are considered good enough here are enough to roughly track limbs, they still have errors comparing to those frames where there is no silhouette error.

4.4.2.4 Common Properties of Actions

In our recordings there are 4 different people performing the actions. This is done to make sure that the system includes the possible differences of implementation of an action by different people. With that information common properties of actions like in general and in between different people can be observed.

It is observed that the period of time it takes to practice an action differs but has a certain mean value. Even if different people do them differently, they still take pretty much the same amount of time. Picking action for example takes about 10 to 15 frames. A browsing action could be different if the person finds the product she/he is looking for but it definitely still is longer than a picking action with 20 to 40 frames. Trying on action indeed depends on the products that are being tried. If the person is trying on glasses, the action doesn't take long, since there isn't much that could be varied between people and it is a short action with around 20 frames. However if the product is like a jacket or a scarf, it takes longer and it could be something around 25 and could go up to 50 frames, and even more is possible.

4.4.3 Statistical Information about the Database

As mentioned earlier in this thesis, there are basically six actions often observed in our recordings. Those are picking, browsing, examining, trying on, driving shopping car and waving. The samples obtained are filtered and eliminated as mentioned earlier as well, and the final dataset is constructed.

Table 4.3: All the samples obtained from the segmentation of recordings.

Action Name	Number of Samples
Browsing	11
Examining	37
Picking	139
Trying on	22
Waving	7
Driving Shopping Car	5

In Table 4.3 the results of the first round of segmentation can be seen. There are 6 actions selected, browsing, examining, picking, trying on, waving and driving shopping car. The number of samples per action is varied though. This is not a common property for similar datasets, however in our case that is shopping context, number of samples also show the frequency of those actions. It is quite obvious that picking action is the most common action in such a context with more than a hundred samples. We don't have many samples with driving the shopping car because that really depends on the setting and field of view of the camera. In our experiment we focus on the interaction of people with products on the table and in that case the shopping car is usually next to people and they rarely drive it because they already focus on a small field of interest.

Table 4.4: Subclasses of picking action.

Subclasses of Picking Action	Number of Samples
Picking from table	66
Picking from basket	6
Putting in basket	26
Putting on table	41

An important note about two actions, trying on and picking has to be mentioned. For those actions, there might be subclasses defined. For trying on action, as mentioned earlier, the product that is being used could make some difference. Those samples could be classified as different actions, yet they all belong to the trying on action. In Table 4.5 we can see the number of samples for trying on action per object. Similar to that, picking action could also have subclasses. A person might be picking an object from the table, putting it back, put it in the basket or pick an object from the basket. However it is a difficult problem to tell the difference between picking a product from the table and putting a product on the table. Movement is pretty much exactly the same with reverse order of frames. One possible solution to that could be detecting and tracking the objects that are being held by the person in the scene, yet this could be future work for this thesis. Number of samples per subclasses of picking action can be seen in Table 4.3 the results of the first round of segmentation can be seen. There are 6 actions selected, browsing, examining, picking, trying on, waving and driving shopping car. The number of samples per action is varied though. This is not a common property for similar datasets, however in our case that is shopping context, number of samples also show the frequency of those actions. It is quite obvious that picking action is the most common action in such a context with more than a hundred samples. We don’t have many samples with driving the shopping car because that really depends on the setting and field of view of the camera. In our experiment we focus on the interaction of people with products on the table and in that case the shopping car is usually next to people and they rarely drive it because they already focus on a small field of interest

Table 4.5: Subclasses of Trying on Action.

Subject used for trying on	Number of Samples
Hat	9
Scarf	3
Vest/Jacket	4
Glasses	6

After deciding on the actions and filtering them, the final dataset that is being used in the experiments is constructed. For picking action only samples of the ‘picking from table’ subclass are used to avoid the problem of similarity between those subclasses of picking. In Table 4.6 final dataset and number of samples can be seen. For some of the experiments, waving and driving the shopping car actions are not involved and only four actions are used because of the lack of number of samples.

Table 4.6: Actual database used in experiments.

Action Name	Number of Samples
Browsing	9
Examining	22
Picking	39
Trying on	12
Waving	7
Driving Shopping Car	4

5 Feature Extraction and Analysis

The system for this action recognition tasks includes a number of steps. First we capture data using the Kinect tool which can generate several types of data. With the ETH Tool and using the RGB frames of the data we generate the limb locations. After those steps we have sets of data representing the scene. After that, comes the feature extraction part and we try to find the most discriminative features that would clearly define the action. Using those we use the classification task where we detect the actions.

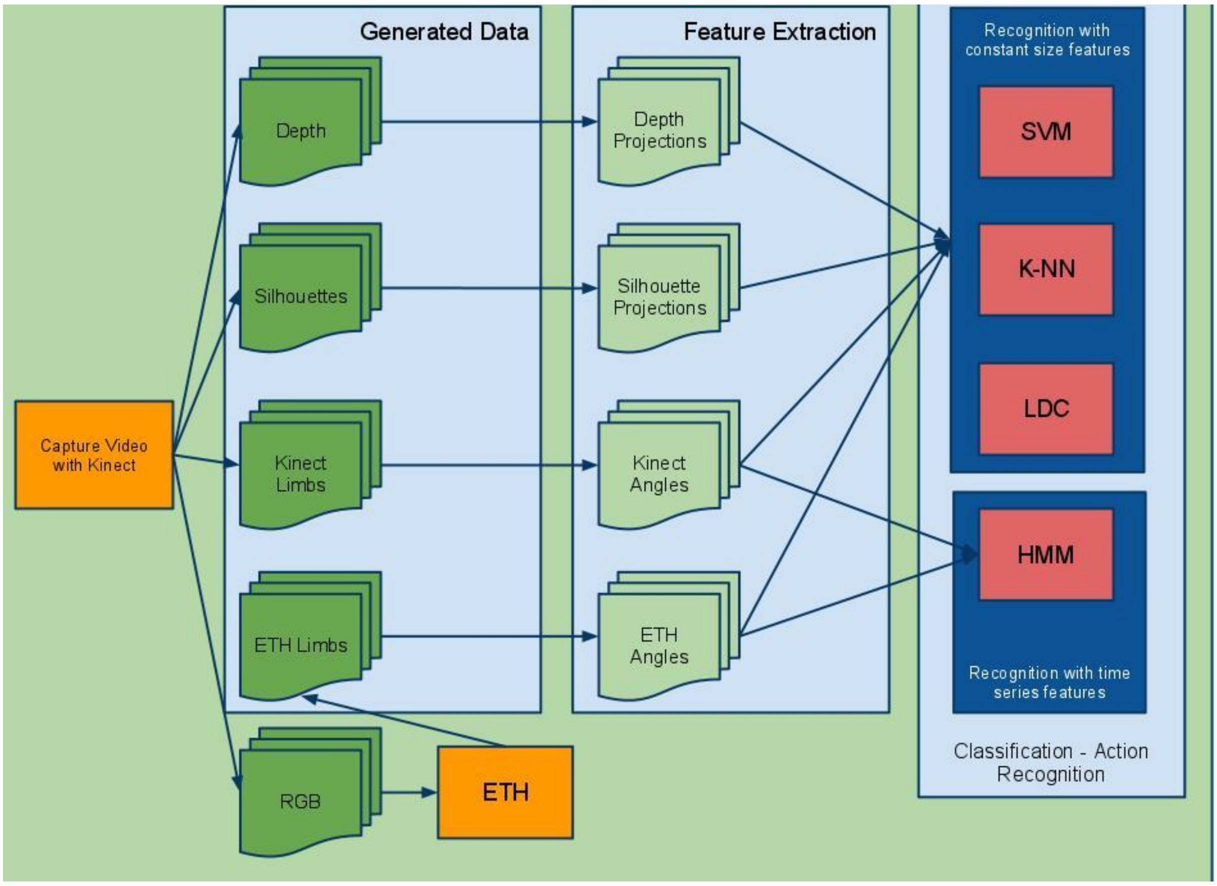


Figure 5.1: General work flow of the system architecture.

Two tools ETH and Kinect and the data they generate were introduced in the previous chapters. Here in the chapter the focus is about the features. Here we define what types of features we produce and how we produce them. Rest of the chapter is divided into types of features and their method of extraction.

In our experiments two tools are used; ETH Tool and Kinect. For those two tools, different approaches can be considered. The information they produce have strengths and weaknesses, so different methods can be designed considering those weaknesses. It is also important to compare and evaluate those two tools so we also used the same method for both approaches to see the performances.

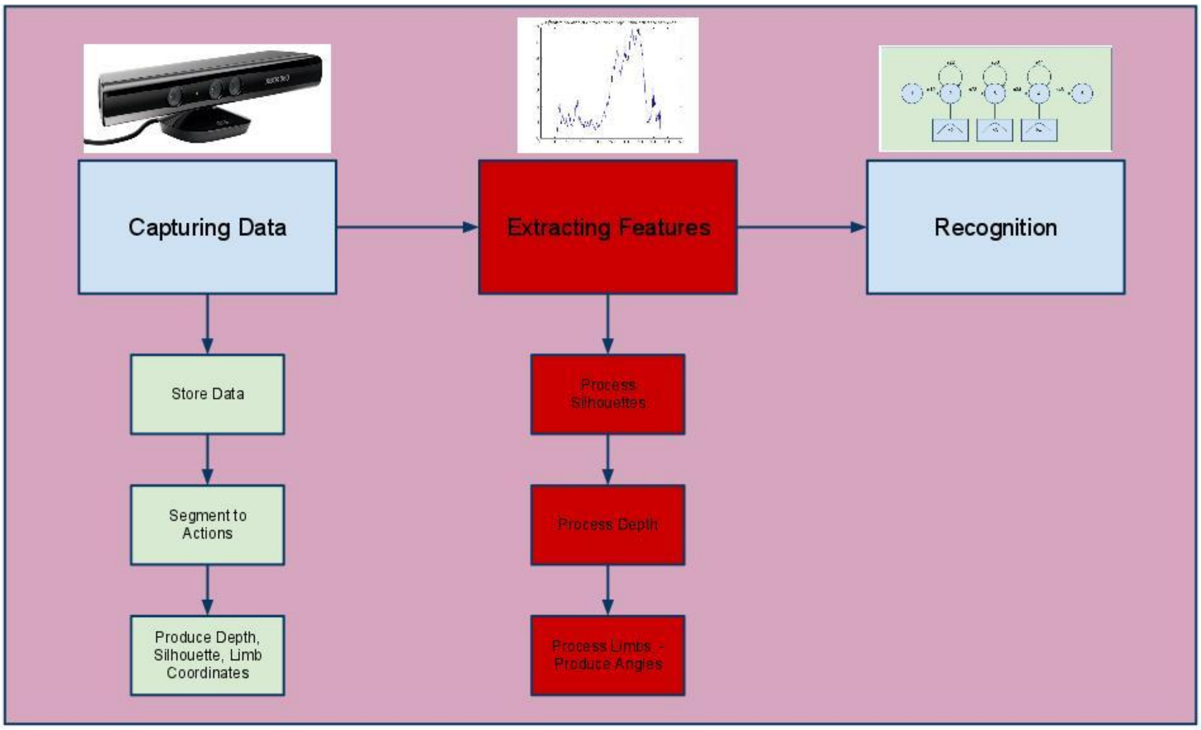


Figure 5.2: Feature extraction step of our system.

As mentioned earlier in the thesis in Chapter 2.7, there are 3 sets of features that could be extracted from the recordings; silhouettes, depth and stick configurations. These ‘features’ are more like information about the scene though; we extract further features from those information sets. In this section, we describe how those features are extracted and how the feature sets are constructed.

It is important to mention that in temporal problems the sample needs to be represented with temporal features. In our case, number of frames is different per sample. Features that are extracted only from one frame lead to another temporal feature set for a temporal sample. In our approaches we extracted features from frames and then lead to another feature vector that describes that array of feature vectors. Below this is explained in detail.

5.1 Stick Coordinates and Angles

Stick coordinates is a feature set that can be used with both tools, ETH Tool and Kinect. The main goal of ETH is to detect the limbs of a person in a scene and return the coordinates of the limbs of the upper body. Kinect is also capable of originating certain locations of a person if it is configured and Kinect is tracking the person at that moment.

One idea for representing the action with stick coordinates is about the movement of the arms when a certain action is performed. With most of the action a person can perform it can be expected that the arms would do a lot of significant moves that could define that action. In our case the actions we have like browsing, examining, picking etc; the arms are the lead actor for performing the action. Those actions are done by the arms, or if not, the arms have a certain position. For picking action for example, although the person might also move his/her body and lean forward a little bit, still the main characteristic movement is done by the arm. The arm leans forward and picks a product. In our definition of actions in Section 4.1.1 that is actually quite clear that we define the set of actions by the movements of the arms. However that might not always be the case. As an example to that, driving the shopping cart action is not like picking. In that case the arms are not really moving around and making certain poses, but they have a characteristic pose. Although relative to the torso arm locations are constant, that constant pose also tells a lot about the action. They both, or at least one of the arms, lean forward, hold the handle of the shopping cart and stay like that for most of the frames of the action sample.

Table 5.1: Angles used for angle feature vector.

Joint Locations for Angle Features	Feature Representation
Torso – Upper Right Arm	*tur*
Torso – Upper Left Arm	*tul*
Upper Right Arm – Lower Right Arm	*ulr*
Upper Left Arm – Lower Left Arm	*ull*
Torso – Y Axis	*ty*

Considering the idea explained above, the movement of the arms explaining the action, it can be yielded that the angles between the limbs could be a feature set defining an action. For that reason, 5 joint locations are picked. Those are shown in the table above.

$$af_n = [tur_n\ tul_n\ ulr_n\ ull_n\ ty_n]$$

Eq. 5.1

We extract those features per frame and call that feature vector af_n for nth frame in the sample.

af_n feature is extracted per frame and then we end up with N observations for a sample with N frames. The constructed matrix for a sample is:

$$af = \begin{bmatrix} af_1 \\ \vdots \\ af_n \end{bmatrix}$$

Eq. 5.2

This matrix has N rows and 5 columns. Obviously the dimensions of this matrix depend on the number of frames for an individual sample. Yet a constant size of feature size needs to be constructed and for that the relative change of those features between frames needs to be extracted. A covariance matrix can well handle this job. The covariance of this *af* matrix returns a matrix that has 5 rows and 5 columns. That doesn't depend on the number of frames per sample therefore it is useful.

An important property of covariance matrix is that it is a symmetric matrix. Therefore the values that are below the main diagonal are actually already included in the upper part of the main diagonal. Overall the angular feature vector *saf* per sample is constructed by getting the main diagonal and the values above that from the covariance matrix of the *af* matrix.

$$saf = [af_{11} \quad \dots \quad af_{15} \quad af_{22} \quad \dots \quad af_{25} \quad af_{33} \quad \dots \quad af_{35} \quad af_{44} \quad \dots]$$

Eq. 5.3

saf feature vector has 15 dimensions and it represents a whole action sample, independent of the number of frames it has.

5.2 Silhouettes

As defined earlier, a silhouette image is the binary image that has a value 1 for the pixel values that are labeled as the person in the scene. For a sample with *K* frames, we have *K* binary images. From those silhouettes, several features can be extracted. In our methods, we extracted x and y projections, extreme points, frame differencing and feature differencing using the projections. In this section we explain those.

5.2.1 Projections

A sample that has *K* frames has *K* binary images. Every binary image has *M* rows and *N* columns. We extract two vectors of features from every image, X projection and Y projection. X projection is a vector with *N* dimensions, as well as Y projection has *M* dimensions. We will refer to X projection as *xproj* and *yproj* will stand for Y projection of an image. *ImS* stands for a binary silhouette image. *xproj* and *yproj* are projections of the silhouette image in x and y axis, respectively.

$$xproj_n(i) = \sum_{j=0}^{M} ImS_n(j,i)$$

Eq. 5.4

$$yproj_n(i) = \sum_{j=0}^{N} ImS_n(i,j)$$

Eq. 5.5

In Eq. 5.4 and Eq. 5.5, formulas for the *xproj* and *yproj* for the n^{th} frame of a sample are shown. Those projections make sense when they are visually observed. In Figure 5.3 an example of projection extraction can be seen with the original image, silhouette image and the projections. In the Y projection plot, it can be seen that the values start with a relatively small mean value and then rises and not varied much. First part is related to the head of the person and the rest is the representation of the torso and the arms together. It is not possible to visually distinguish the arms from the torso from that projection though. X projection on the other hand carries significant information as well. We can see the peak of the curve which is the central part of the body and the skewed parts of the curve represent the arms of the person.

In our experiments *xproj* and *yproj* are used as the feature sets yet as can be seen the dimension of the feature vector is quite high. The overall feature vector per frame is then the $proj_n$, which is basically $xproj_n$ and $yproj_n$ put together.

$$proj_n = [xproj_n \; yproj_n]$$

Eq. 5.6

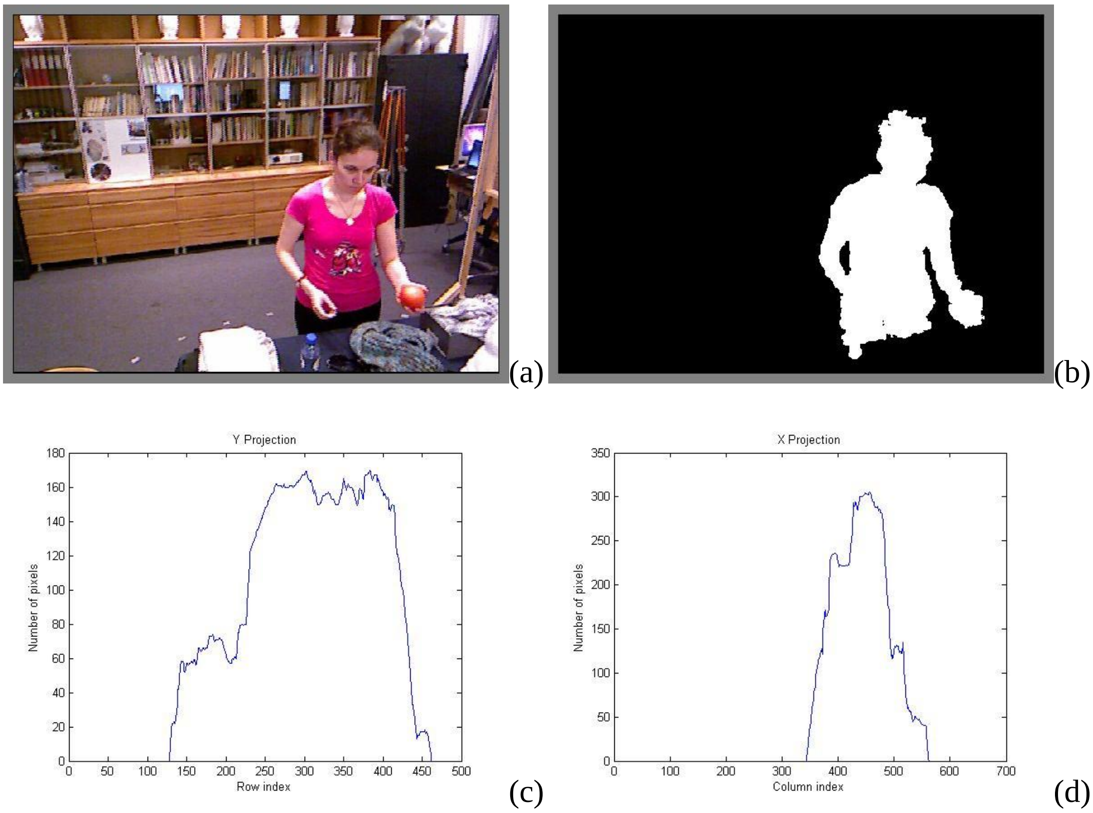


Figure 5.3: X and Y projections sample. (a) Original image. (b) Silhouette image. (c) Related Y projection. (d) Related X projection.

As seen in Eq. 5.6 when the *xproj* and *yproj* are put together the dimension of the *proj* feature vector per frame increases to M + N, number of rows and columns of the image, dimensions. In our recordings the resolution is 640 by 480; therefore *proj* vector has 1120 elements. This is already quite a high dimension for a feature vector but it is only per frame. In a sample with *N* frames, the feature dimension even increases more. The overall projections feature matrix of a sample with *N* frames would be,

$$sproj = \begin{bmatrix} proj_1 \\ \vdots \\ proj_n \end{bmatrix}$$

Eq. 5.7

This *proj* matrix has 1120 columns but the number of rows differs because of the different number of frames per sample. However we need to have a feature vector per sample which has a constant dimension. More than that, what we are really focusing here is to emphasize the specific movement in the silhouette. If the body is pretty much still at some parts, for the projections those parts will all have very close values and not vary much; therefore what we expect is a high variance in parts of the projections related to the parts of the body where the movement is. Very simply the standard deviation of the *proj* matrix will give us the emphasized variance in the projections in a sample. In the mentioned methods, there are approaches where the covariance matrix is used to represent a whole sample, yet the diagonal of the covariance matrix is already the standard deviation of the matrix. Therefore the feature vector *s*, standing for standard deviation, for a sample with *N* frames is,

$$sp(i) = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}\left(proj(k,i) - \overline{proj(:,i)}\right)^2}$$

Eq. 5.8

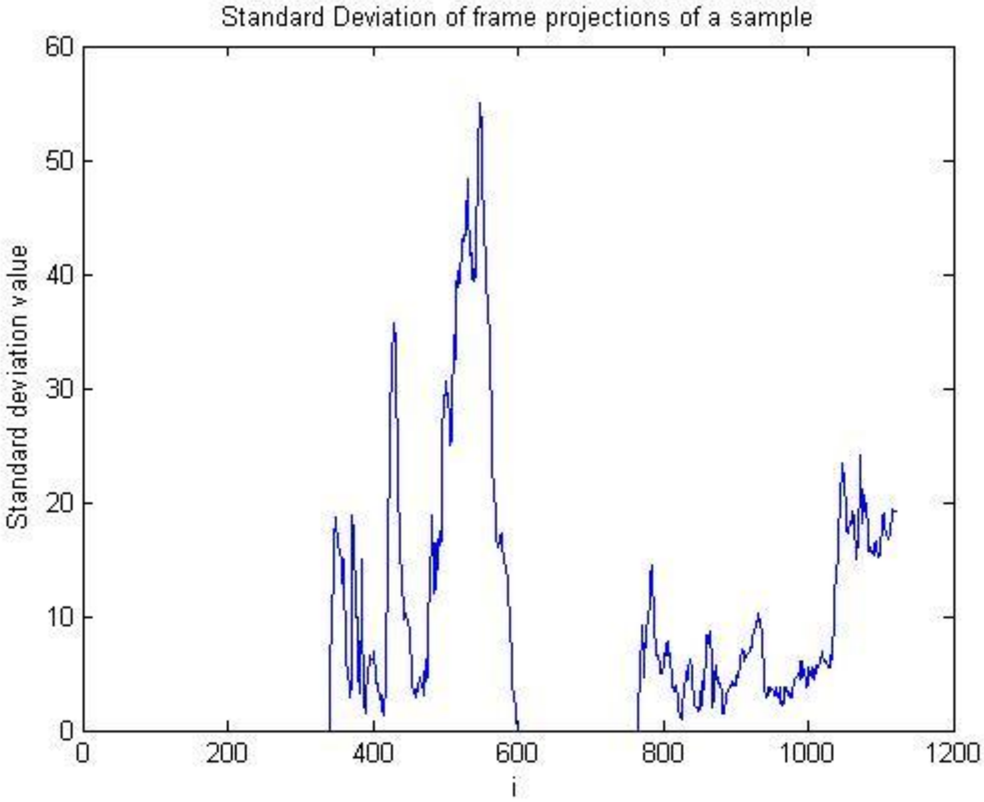


Figure 5.4: Projection standard deviation feature for a sample.

From Figure 5.4 some reasoning can be made. This feature is from a sample of picking action. What we can see is that the standard deviation values make a significant peak around the indexes between 450 and 600. This part represents the y projection of the silhouette and therefore this peak belongs to lower part of the body. Just like that, the x projection has relatively higher values around the right hand side of the body. When the sample is observed it can be seen that the person in the scene is leaning out with the right hand, grabbing a product and taking the hand back. Therefore we expect movement and variance in the silhouette on the lower part of the body and right hand side of the torso. This feature set and sample prove that this principle can really show the action information.

5.2.2 Features Originated from Projections

Although projections themselves can be used as the features for a classifier or detection system, they have relatively high dimensions. If a feature set that is slower and can still explain the action information is extracted, we can assume that the action can still be represented.

As seen from the previous section silhouette image already carries a lot of information and the projections can be significantly beneficial for that goal. What is explained in the previous section is mostly that the peaks of the variance can show the action area. Therefore it can be assumed that some set of features that can be extracted from those projections can have a smaller dimension, so that computational costs can be decreased, and the action information can still be represented. For that aim, the projections can be considered as distributions and distribution properties like mean, standard deviation, skewness can also explain a lot. In Figure 5.4 is a good example and the skewness values of that projections can also lead us to the same reasoning; y projection has a peak value on the right hand side and so the x projection. Skewness would also represent this reasoning, and comparing to other samples of actions it can even show the action information in a scale.

The distributional features used in this feature extraction are; center of mass, variance of the distribution, skewness, index of median. We extract those features for *xproj* and *yproj*, put them together and have a feature set of 8 dimensions.

A very important thing in extracting those features is that if we take the *xproj* and *yproj* without any manipulation, then the feature set becomes sensitive to the distance to the camera and the person location in the room. As seen in Figure 5.3 c and d, there is huge information that belongs to the pixels not labeled for the person. What we are interested in this vector is the values that only correspond to the projection and have non-zero values. Therefore we need to shift and scale the projection of the silhouette of the person, in other words the region of interest.

The idea behind this shifting is enlarging the person to the size of the screen, therefore the *xproj* should have 640 dimensions and *yproj* should have 480. We first simply select the non-zero values of the projections. Then the problem is to make the vector have a dimension of 640, for *xproj* for example. In this step, the projection vectors we have definitely have less than that many dimensions, so it becomes a problem of interpolation and filling in the values in between.

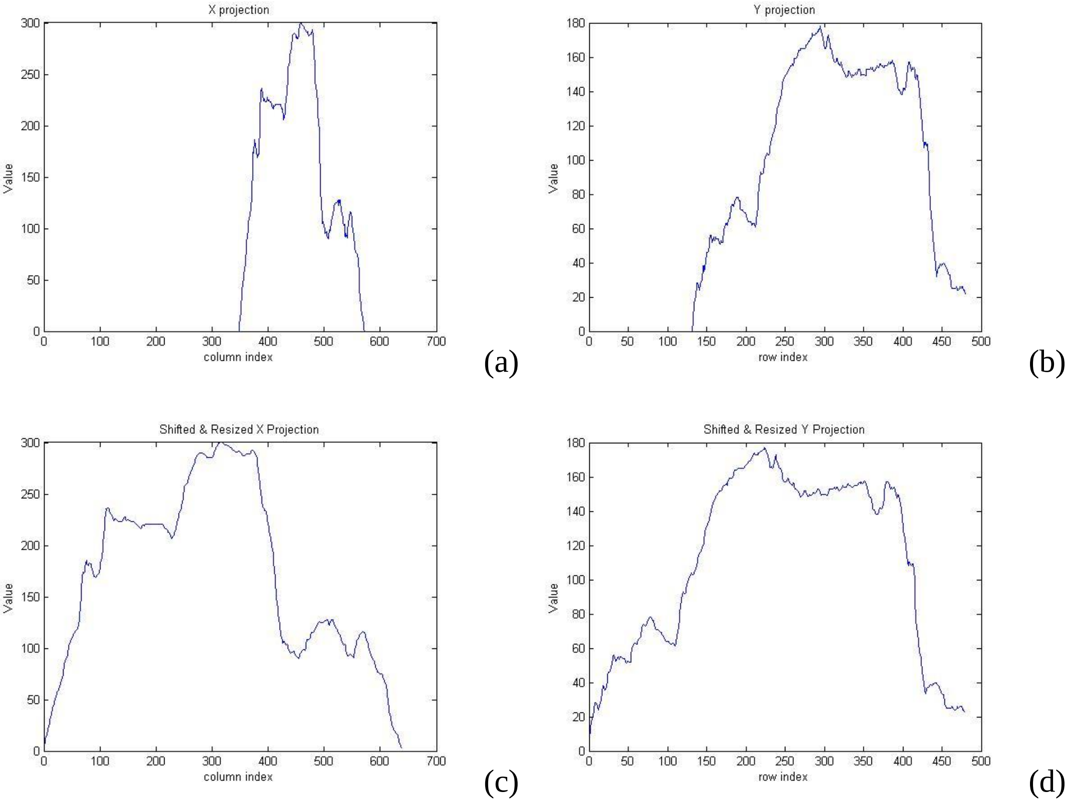


Figure 5.5: Shifting and resizing example. (a) Original x projection. (b) Original y projection. (c) New x projection. (d) New y projection.

In Figure 5.5 this operation can be seen with an example. In (a) and (b) original x and y projections can be seen respectively. In (c) and (d) we see the projections after the interpolation. They have the general outlines and properties of the original ones yet they have now 640 and 480 dimensions as can be seen, as if the person exactly fit in the frame.

Now that our projections are independent of the person's location and distance, we can extract the distribution of features from those projections, considering them as if they were distributions. As it was done in the previous section, we extract those features per frame. Then we have a matrix of *N* observations of those features for a sample with *N* frames.

$$vol_x = \sum_{k=0}^{640} xproj(k)$$

Eq. 5.9

$$com_x = \sum_{k=0}^{640} (xproj(k) * k)/vol_x$$

Eq. 5.10

$$comsd_x = \sqrt{\sum_{k=0}^{640} \frac{xproj(k)}{vol_x} (k - com_x)^2}$$

Eq. 5.11

$$skew_x = \frac{\frac{1}{vol_x} \sum_{k=0}^{640} xproj(k) * (k - com_x)^3}{(\frac{1}{vol_x} \sum_{k=0}^{640} xproj(k) * (k - com_x)^2)^{3/2}}$$

Eq. 5.12

$$maxi_x = arg \max_i(xproj(i))$$

Eq. 5.13

In Eq. 5.9, Eq. 5.10, Eq. 5.11, Eq. 5.12 and Eq. 5.13 formulas for volume, center of mass index, mass standard deviation, skewness and index for peak value can be seen. For features of the y projection, value of 640 becomes 480.

As mentioned before, those values are calculated for both *xproj* and *yproj* vectors. Per frame the feature vector is a fusion of those two vectors. Let *disfeat* stand for the feature vector per frame, then the feature vector consists of,

$$disfeat_n = \left[com_{x_n}\ comsd_{x_n}\ skew_{x_n}\ maxi_{x_n}\ com_{y_n}\ comsd_{y_n}\ skew_{y_n}\ maxi_{y_n}\right]$$

Eq. 5.14

Eq. 5.14 stands for feature vector per frame and for the sample feature matrix; we put those observations in a feature matrix. In Eq. 5.15, *disfeat* stands for the feature matrix for the sample *S*.

$$disfeat = \begin{bmatrix} disfeat_1 \\ \vdots \\ disfeat_n \end{bmatrix}$$

Eq. 5.15

The problem again here is that the dimensions of that matrix depend on the number of frames in the sample. Therefore we need to yield to the feature vector that will define the whole sample and will have the same dimension for all samples. The method we used in the previous section, getting the standard deviation of this matrix, again explains a lot about the sample; therefore we use the same implementation here.

$$sdf(i) = \sqrt{\frac{1}{N-1}\sum_{k=1}^{N}\left(disfeat(k,i) - \overline{disfeat(:,i)}\right)^2}$$

Eq. 5.16

Table 5.2: Mean and standard deviation of distributional features of a waving action sample (leon2_wave2).

sdf	**mean$_x$**	**std$_x$**	**skewness$_x$**	**mode$_x$**	**mean$_y$**	**std$_y$**	**skewness$_y$**	**mode$_y$**
mean	272,41	136,19	0,42	266,5	238,48	117,25	0,12	175,29
std	23,35	7,81	0,25	83,90	12,19	2,92	0,10	50,51

Eq. 5.16 shows the distributional features vector s*df* for a sample. In Table 5.2 and Table 5.3 we see the mean and standard deviation values of distributional features of a waving and picking action samples respectively. When those two samples are examined it can be seen that for the waving action, the person raises his right hand and waves his hand. For the picking action the person leans with her right hand to the table, picks a product and takes her hand back. In both cases it is the right arm and in both cases the arms lean forward a certain direction. Yet the difference is that in one of them arm leans down, in the other one arm leans up. The differences cause the mean values of most features in the *df* vectors to be same, however the y projection shows the difference. In specific, the skewness and mode values of the y projections emphasize the differences between actions. In one of the projections skewness is negative and it is positive in the other one. We can also see that the mode, index of peak values are quite far from each other.

Table 5.3: Mean and standard deviations of distributional feature of a picking action sample (marina2_pick1).

sdf	**mean$_x$**	**std$_x$**	**skewness$_x$**	**mode$_x$**	**mean$_y$**	**std$_y$**	**skewness$_y$**	**mode$_y$**
mean	320,88	149,78	0,02	359,1	243,25	112,38	-0,05	273,10
Std	18,49	5,17	0,13	102,48	17,03	4,70	0,15	56,64

5.2.3 Frame Differencing

For action recognition the aim is to find and extract features that emphasize the movement in the scenes. Those movements, volume, location and direction of them define the action. The features we used above depend on the projections without any manipulations on the silhouette images. However when the silhouette and movement ideas are brought together, we can assume that the difference between the silhouette of two successive frames might tell the location and volume of movement between those frames. Therefore before extracting those features, frame differencing might reveal the movement in the sample.

For frame differencing we use the silhouette differences between two frames and end up with a new image matrix that has values 0, 1 and -1. Pixels with value 0 mean no movement in the area. Pixels with value -1 are the fields where the person moved from. Pixels with value 1 are the areas where the person moved to.

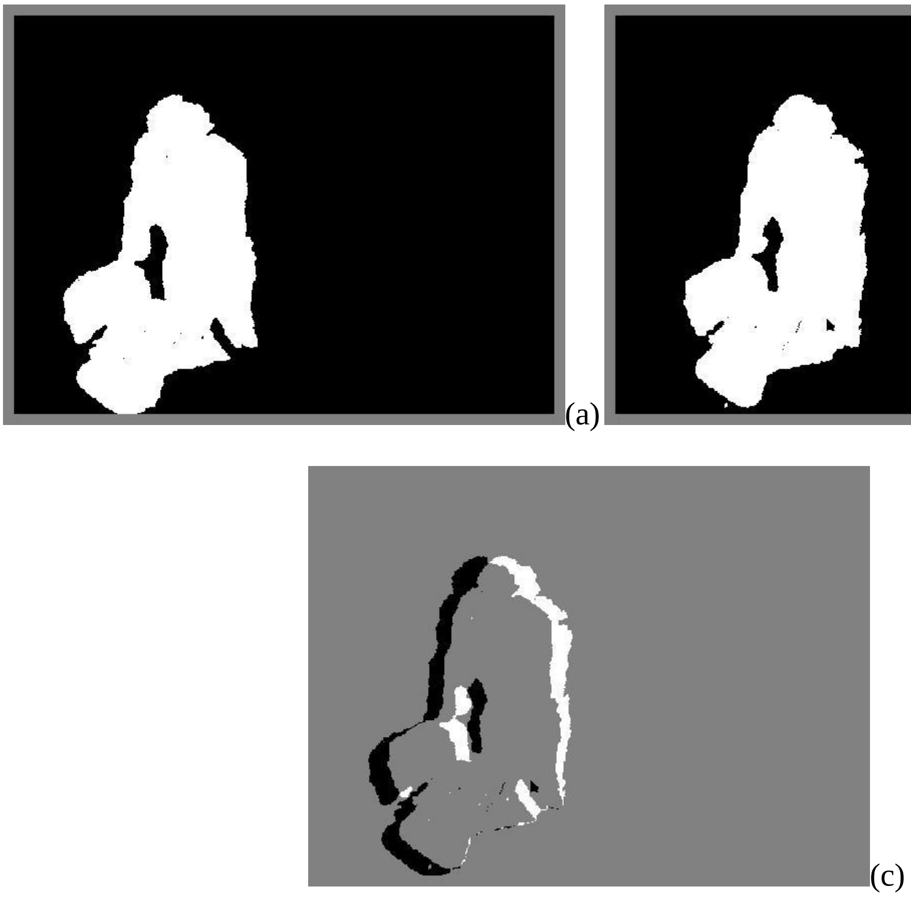


Figure 5.6: Frame differencing sample. (a) Silhouette image i. (b) Silhouette image i+1. (c) Frame difference between (a) and (b). (leon2_wsc1)

As we can see in Figure 5.6, the difference between the two frames gives us information about the movement. The black pixels are where the person moved from, and white pixels are where the person moved to. From that information we can extract pretty much the contours of the person and the direction of the movement.

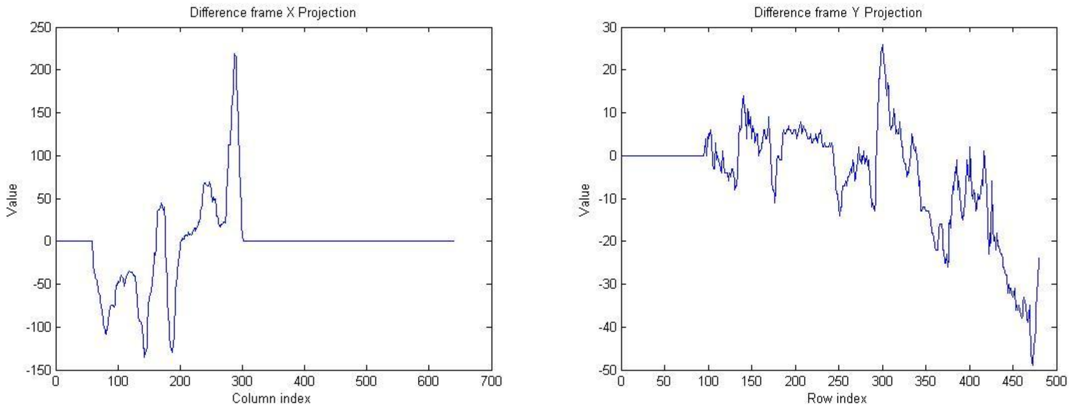


Figure 5.7: A difference frame’s x and y projections, respectively.

Figure 5.7 shows the projections of the difference of frame. From that information, we can see that x projection shows that the person is moving to the right, and from the y projection we can see that around the lower parts of the body there is a huge volume of ‘moving from’.

In Figure 5.8 we can see the difference between two samples from two different actions. Driving shopping cart includes a large volume of movement while examining action relatively less on the other hand. Especially the y projection mean vector show that, first action has a lot of variance in the values, yet the examining action has smooth values. In both cases there is a little bit of moving to the right, yet it can also be said the range and volume of movement is relatively small for examining action.

After all, frames can be differences and as shown in the examples above, they can carry a lot of information about the action. Right after that step, the methods that are explained in section 5.2.1 and 5.2.2 can be used to extract several features from those difference frame sequences.

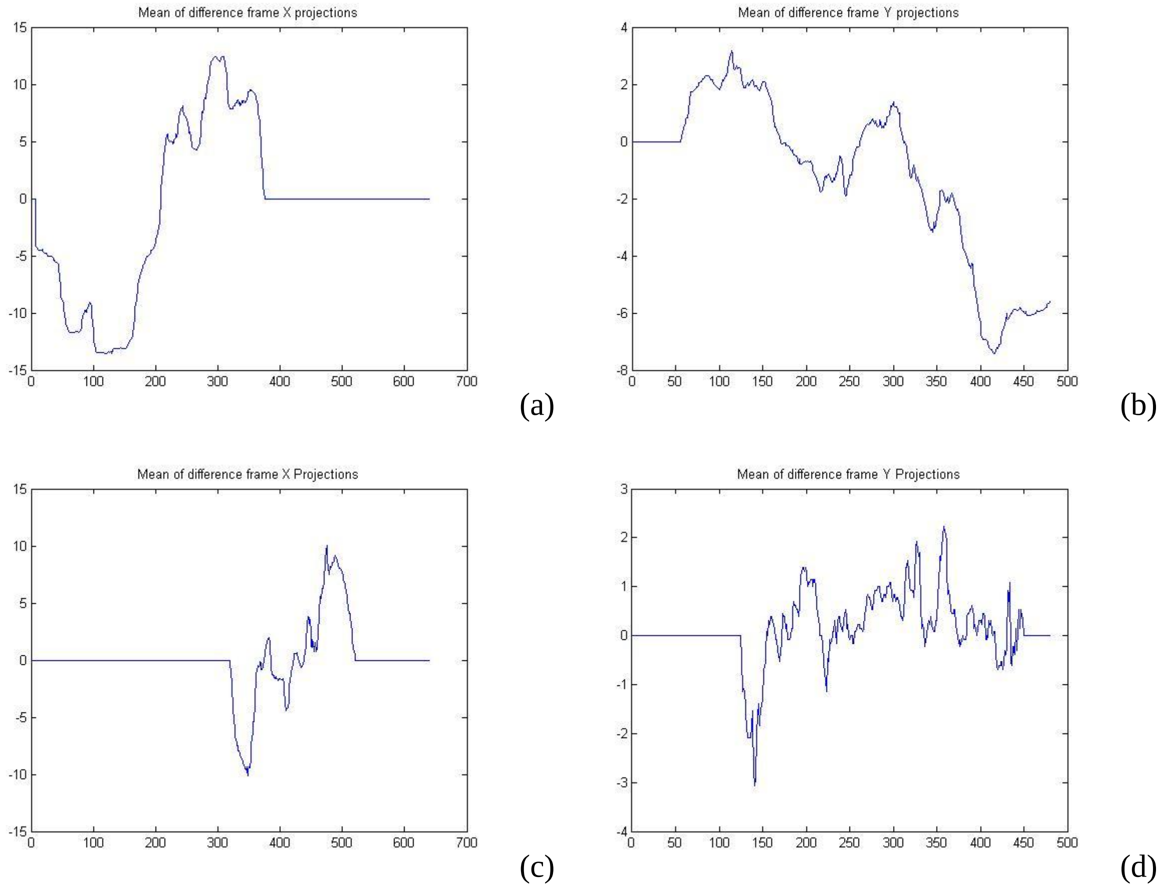


Figure 5.8: Mean x and y projections of difference frames from two samples. (a) Mean x projection of driving with shopping cart sample. (b) Mean y projection of driving with shopping cart sample. (c) Mean x projection of examining sample. (d) Mean y projection of examining sample.

5.2.4 Feature Differencing

Besides frame differencing, feature differencing can also lead to good representation of actions. Especially for projections, the difference tells about the action. For picking sample for example, the difference of projections would show the movement that is expected to be leaning of one of the arms.

As we can see in Figure 5.9 feature differencing also tells about the action. In the figure we can see two successive frames, their x and y projection differences. Then we see the standard deviations of x and y projection differences of the whole sample. From those two graphs we can see the high variance around the side of the body, in x projection, and we see the clear high variance around the lower part of the body, which is exactly what we would expect from a picking action.

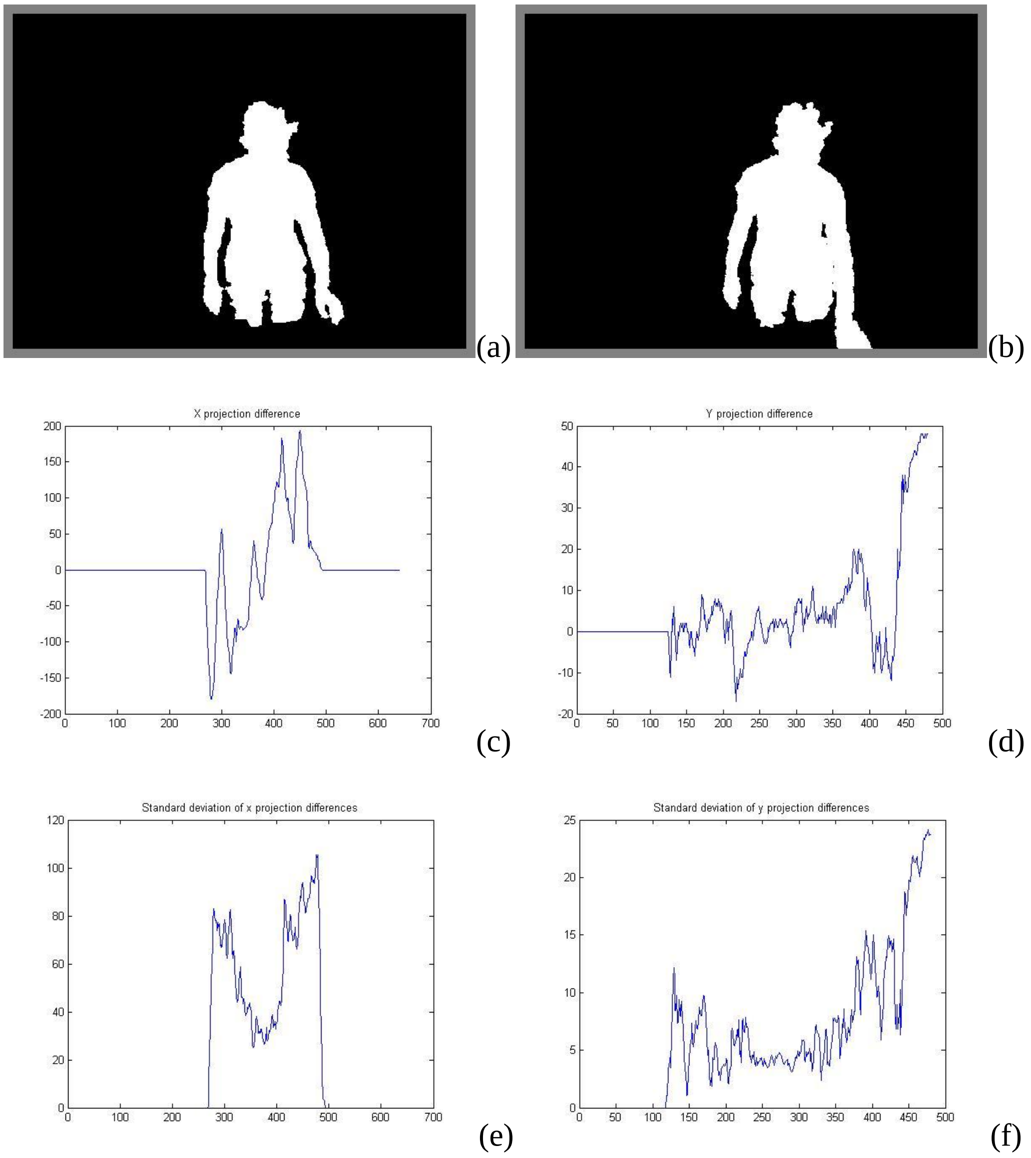


Figure 5.9: Feature projection difference sample. (a) Silhouette image frame i. (b) Silhouette image frame i+1. (c) X projection difference between those two frames. (d) Y projection difference between those two frames. (e). Standard deviation of x projection difference of that sample. (f) Standard deviation of y projection difference of that sample.

5.3 Depth Silhouettes

What a depth silhouette was explained in section 2.7.3. Kinect is capable of generating the depth information of the scene and using the silhouette mask we can yield to the depth silhouette of the person with the depth information.

Silhouettes can explain the information in a sample or a frame, so that the depth. Indeed depth silhouettes are expected to tell even more. With silhouettes the information that is obtained is only 2D and we cannot extract information about the distance of the body and/or limbs.

The features that we extract from the depth silhouettes are not that different from the features extracted from silhouette images, yet they might be carrying more or different information comparing to those. In this section we explain the extracted features from depth silhouettes.

5.3.1 Projections

The depth information returned by the Kinect has large values. For a proper representation and comparison with the silhouette images, this needs to be normalized. What we basically first do is then we shift the values in a scale from 0 to 1. Silhouette images are binary images with values 0 or 1. This case the depth silhouette with all values between 0 and 1 has more sensitive information about the scene.

In Figure 5.10 we can see a comparison of depth silhouette and silhouette image projections. Thick lines in those graphs are projections originated from the depth silhouettes. What can be seen here is that the signal produced here still has pretty much the same structure, however it is more sensitive to the depth of different parts of the body.

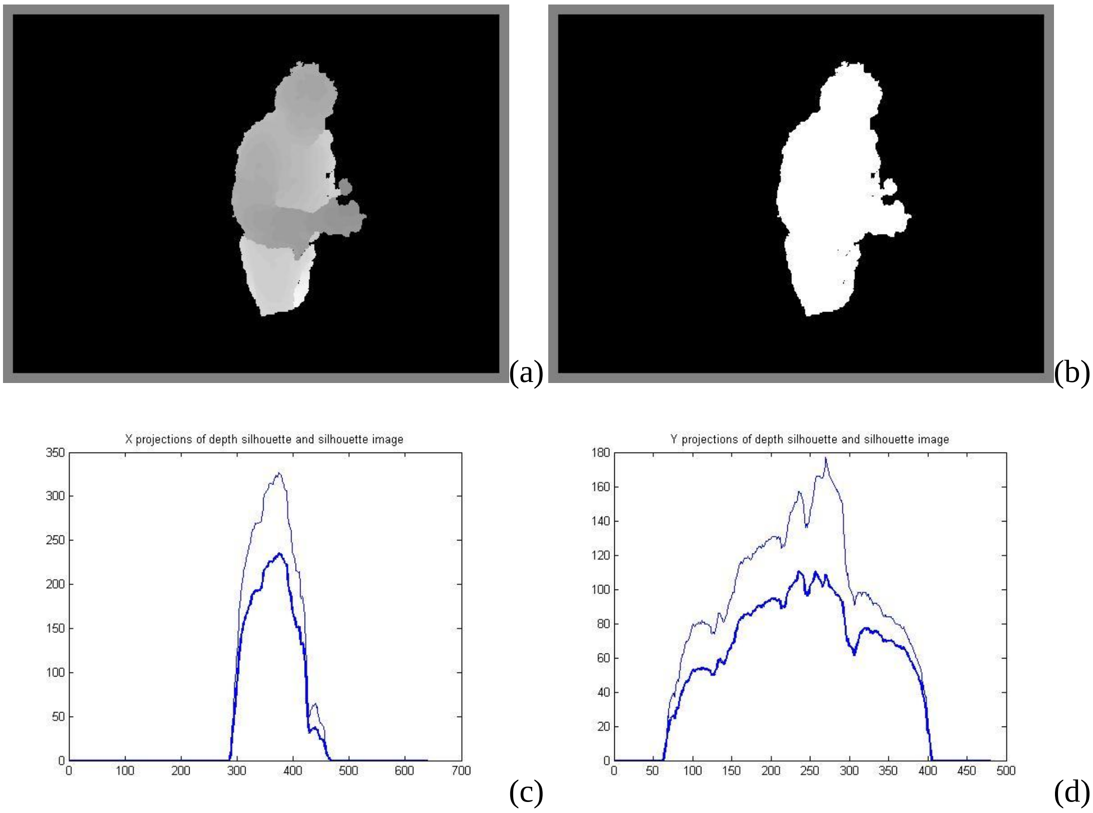


Figure 5.10: Depth silhouette and silhouette image difference sample. (a) Depth silhouette of frame. (b) Silhouette image of frame. (c) Comparison of X projections, thick line showing the depth silhouette. (d) Comparison of Y projections, thick line showing the depth silhouette.

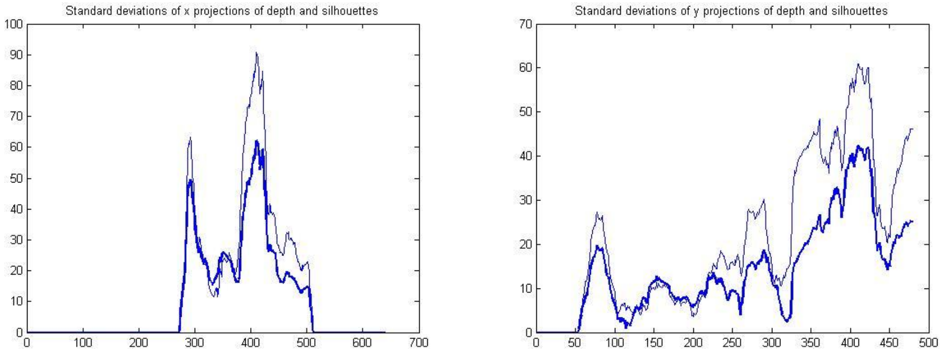


Figure 5.11: Sample standard deviations of projections, comparison between depth silhouettes and silhouette images. Thick lines represent data originated from depth silhouettes. Left: X projections. Right: Y Projections.

In Figure 5.11 we can see the projection feature vector per sample. As it was done before, the feature vector per sample is the standard deviation of projections of all frames in the sample. Thick lines refer to depth silhouettes again. As we can see, there is definitely information difference between those two approaches. Depth silhouettes produce different information than the silhouette images, considering the fact that they include more information about the frame and sample than the silhouette images.

5.3.2 Features Originated from Projections

Further feature representation can be extracted from the projections produced. For this section we extract the same features we used in section 5.2.2 and the same equations from Eq. 5.9 to Eq. 5.16 are implemented here as well. Just like we did with the silhouette images, we shift and resize the projections as well.

When the same procedure is applied, we get the results that are shown in Table 5.4 and Table 5.5. Those two samples are from two different action samples, picking and examining respectively, also showing the comparison between extracting features from silhouette images and depth silhouettes.

Table 5.4: Mean and standard deviation of depth and silhouette distributional features of a picking action.

sdf &ddf	**mean$_x$**	**std$_x$**	**skewness$_x$**	**mode$_x$**	**mean$_y$**	**std$_y$**	**skewness$_y$**	**mode$_y$**
mean – depth	361,17	37,66	0,17	376,33	250,91	88,10	-0,003	278,50
mean - silhouettes	362,04	38,04	0,18	373,17	248,48	88,46	0,02	285,17
std - depth	10,60	4,80	0,21	8,91	22,97	6,13	0,05	72,24
Std - silhouettes	11,70	4,94	0,21	11,44	25,81	8,10	0,08	68,56

Table 5.5: Mean and standard deviation of distributional features of depth and silhouette images of an examining action (leon1_examine1).

sdf &ddf	**mean$_x$**	**std$_x$**	**skewness$_x$**	**mode$_x$**	**mean$_y$**	**std$_y$**	**skewness$_y$**	**mode$_y$**
mean – depth	325,42	44,07	-0,06	342,56	250,98	81,98	0,03	235,11
mean - silhouettes	325,20	43,97	-0,07	341,06	245,56	81,68	0,07	235,17
std - depth	6,26	0,51	0,03	10,94	1,21	0,37	0,01	3,72
Std - silhouettes	6,31	0,48	0,03	8,09	1,17	0,36	0,02	3,81

5.3.3 Frame Differencing

Frame differencing as explained before can extract and explain a lot of information about the actions. When the depth silhouettes are used, we can observe the volume of movement in between two successive frames. Different than silhouette images, the values of frame differences of depth silhouette can have values between -1 and 1 and can also show the limb movements by showing the depth change.

Projections and other features can be produced from those frame differences. A sample with *N* frames can be turned to a sequence of difference frames with *N-1* number of frames. Every new difference frame matrix might have values in a scale of -255 to 255. That is caused that the depth silhouettes have a scale from 0 to 255.

As we can see in Figure 5.12 frame difference of two successive depth silhouettes can yield to the action area in those frames. In the third image, we can see the parts in white where the person moves to and parts in black where the person just moved from.

X and Y projections of those frame differences can give us information about the move. In Figure 5.13 we can see those projections for one depth difference frame. There are features that can be extracted from those features, besides that they can also be used as feature sets. Yet those all yield to frame based features, so for us it is important to obtain sample based features and see if depth frame differencing can explain an action. Therefore from the same sample, a picking action sample, we can observe that standard deviation of those projections of the all frame in the depth difference sequence.

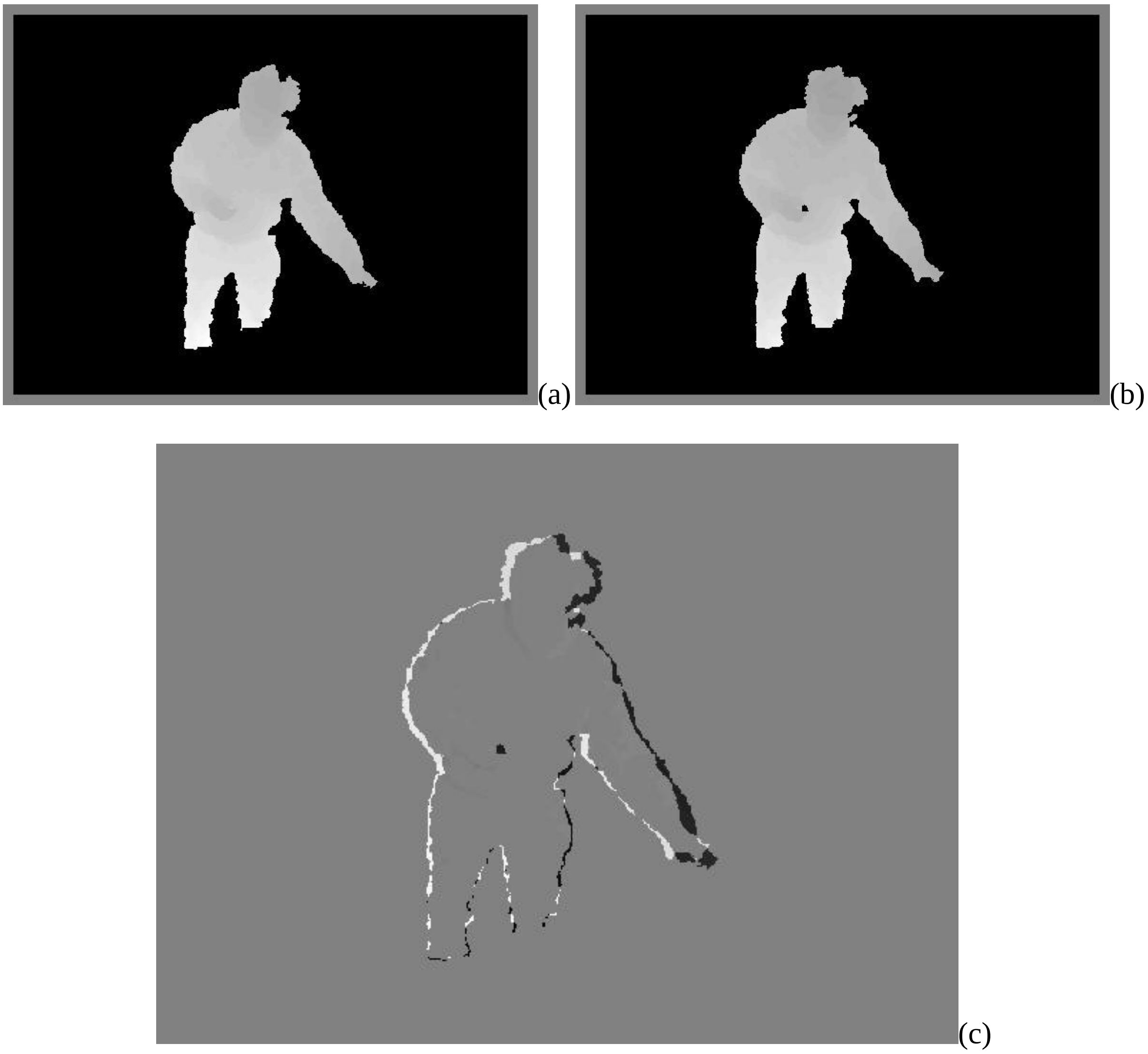


Figure 5.12: Depth frame difference sample. (a) Depth silhouette frame i. (b) Depth silhouette frame i+1. (c) Depth frame difference. (leon3_pick_1)

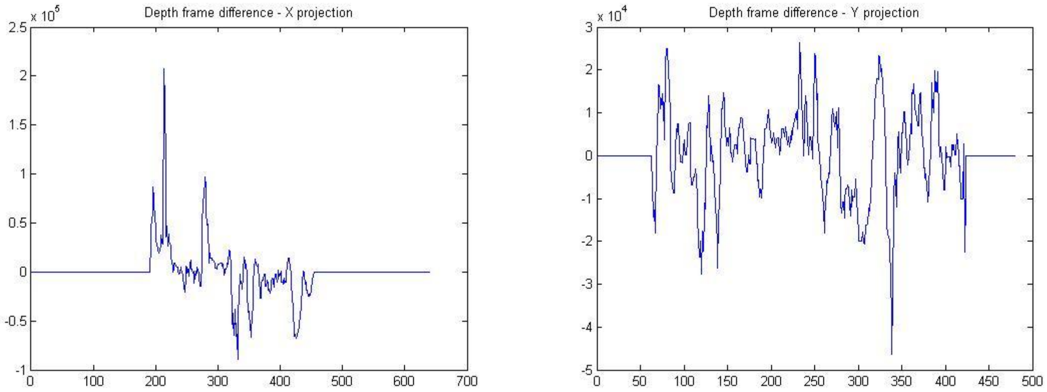


Figure 5.13: X and y projections of depth frame difference image in Figure 5.12, respectively.

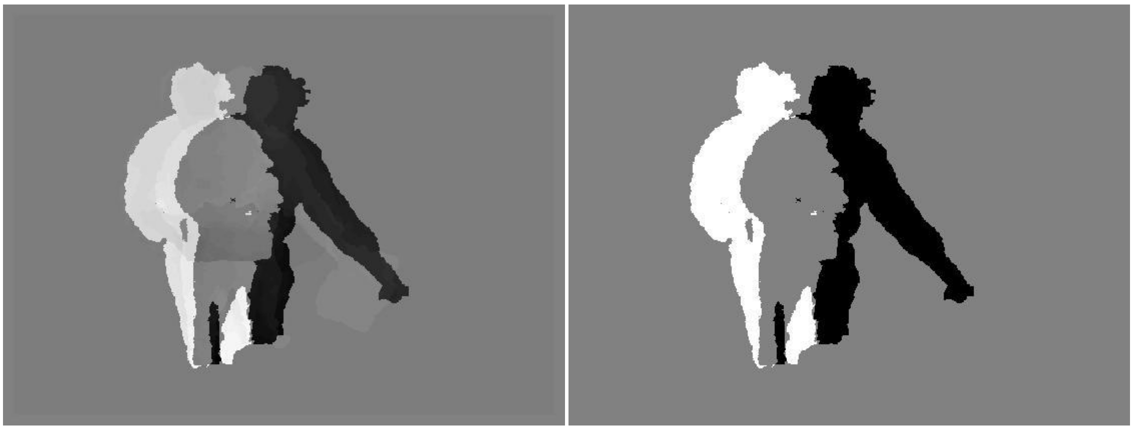


Figure 5.14: All depth silhouette and silhouette image frame differences of sample put together.

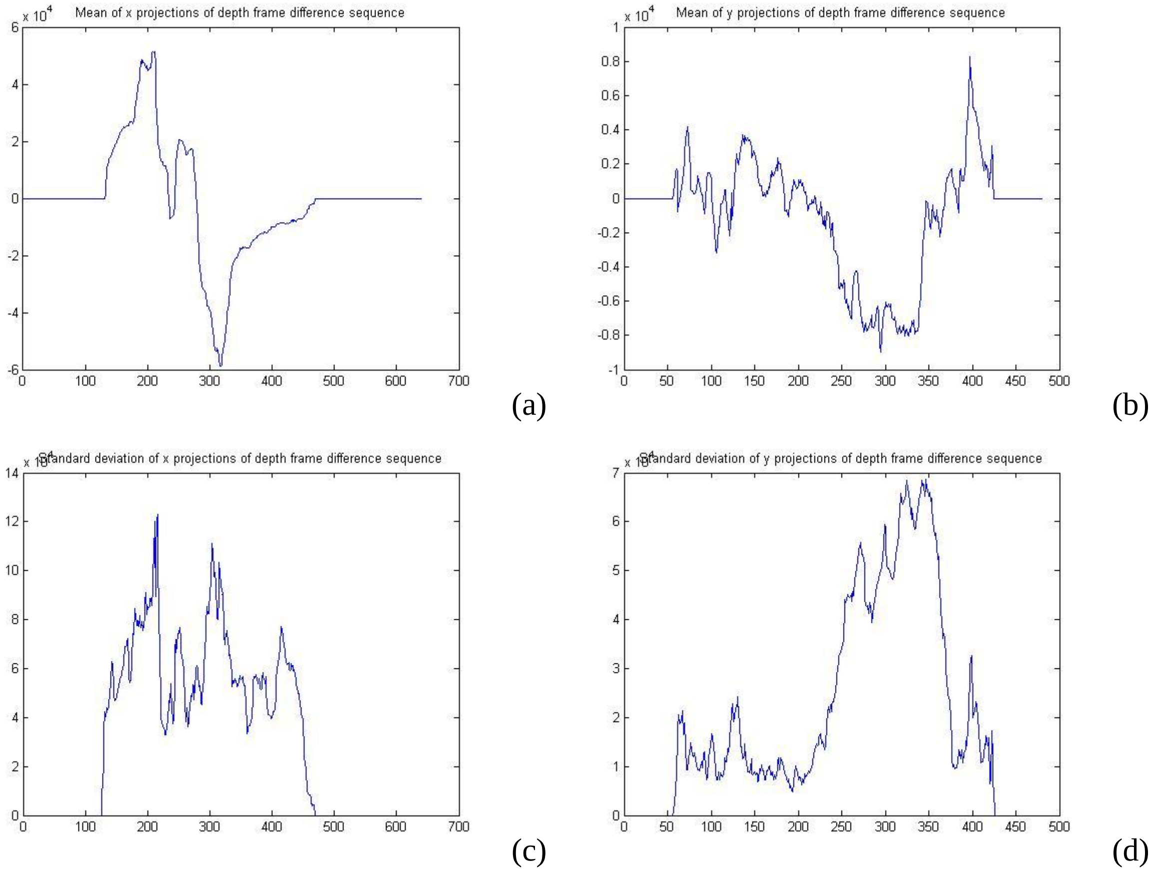


Figure 5.15: Sample standard deviation and mean of x and y projections. (a) Mean of X projections. (b) Mean of Y projections. (c) Standard deviation of x projections. (d) Standard deviation of y projections.

In Figure 5.14 we can see a comparison between the depth silhouette and silhouette image frame differencing. In the image on the left, we can see all the frame differences from a depth silhouette sequence put together. There we can see the huge movements and rather smaller movements and even the object can be detected. In the image on the right, we see the silhouette image frame differences all put together from the same sample. As can visually be seen, depth images can explain more about the action and frame differencing could be a good way for this.

Figure 5.15 shows the properties of x and y projections of a sample that includes depth frame differencing. From those figures, the direction of movement, amount of movement, variety of movement can be detected.

6 Recognition: Classification

6.1 Classification and Action Detection

Next step of the system after extracting the features is obviously how to use those features and detect the actions automatically. This is a problem of classification of those features. In our case there can be a couple of different choices for that part due to two reasons; we have a couple of different feature sets and our case has a temporal side too, therefore classification methods that rely on time could be also helpful.

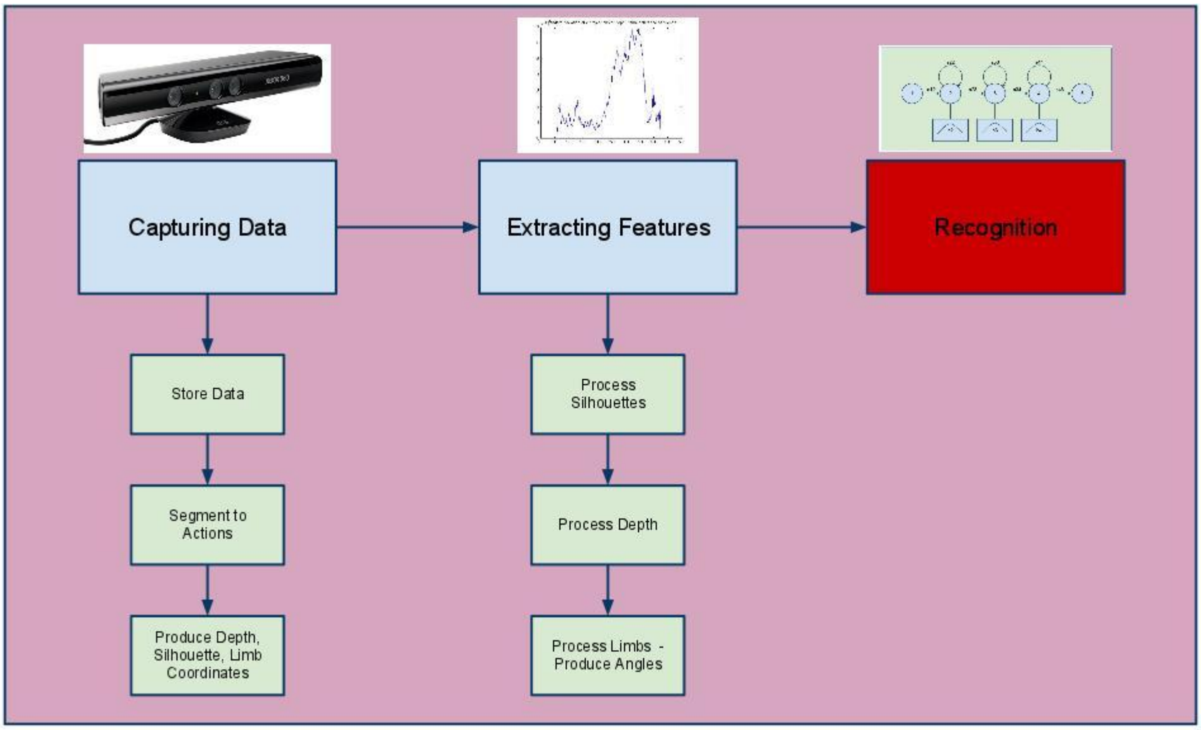


Figure 6.1: Recognition step of our system.

In our approach besides using classification methods using feature sets that are sample based like support vector machines, k-nn methods, for feature sets that depend on frames we use hidden markov models that are widely used for such feature sets.

Classification methods can be clustered in two groups depending on the feature sets we use, angles approaches where we use upper body limb angles and silhouettes approaches where we use the features that are extracted from silhouette images or depth silhouettes. In the following section we explain those approaches.

6.1.1 Angles Approach

Feature sets that include the upper body limb angles are explained in Section 5.1. As also explained above, features can also be used as both frame based and sample based. In that sense we can also divide this section in two sections accordingly as below.

6.1.1.1 Time Series – Detection with Hidden Markov Models

When features that are frame based are used it is important to note that in that case the length of the feature set depends on the number of frames the sample has. That is a problem for most of classification methods because they require a constant size of feature set per observation. More than that frame based features have temporal properties that should be taken into account. When those notes are put together, it can be said that Hidden Markov Models [19] could be a good choice for detecting those actions.

Hidden Markov Models are a strong mathematical model for predicting the temporal features. For this approach, we have feature matrices per sample with 5 columns, belonging to every feature, and N many rows for a sample with N frames. Every row is an observation. The feature matrix dimensions per sample differ yet Hidden Markov Model is capable with work such data because it is capable of processing continuous observations and it makes predictions.

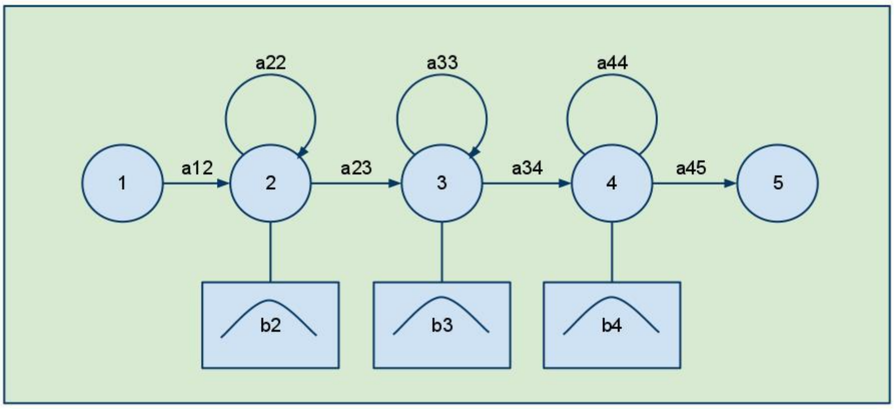


Figure 6.2: A standard 3 state Hidden Markov Model [13].

Hidden Markov Model is a tool which can be used to model time series [14]. In our case, frames constitute a time series so we try to model those different actions. Hidden Markov Models use the Markov chains that include transition probabilities from one state to another. In Figure 6.2 we can see a sample Hidden Markov Model with 3 states.

It is also possible to design different topologies for the states. The transition probability matrix that states the possibilities of changing from one state to the other has a different structure then. Yet in our experiments we use left-to-right topology that can be considered a simple standard one.

We use different number of states in the experiments to see the performance and yield to the optimum design. We give the observations to the hidden Markov Model and the action that has the highest likelihood is chosen to be the label for the sample.

6.1.1.2 Detection with Spatial Features

Features can be extracted per sample and that way they can be applied to classifier that do not depend on temporal data either. In our methods of feature extraction, we also extracted feature vectors of a constant size that would represent a whole sample. In this section, we explain the method we used for this method of action recognition.

As explained in Section 5.1 we extracted a feature vector af_n per frame and then put all frame observations in a matrix *af*. For the aim of getting a constant feature size of feature, we get the covariance matrix of the frame observations and this yields us to the vector *saf* that has a constant size independent of the number of frames in a sample. Those feature sets are explained in Eq. 5.1, Eq. 5.2 and Eq. 5.3.

Mentioned equations lead us to the feature sets for samples and the next step is the classification itself. For that part and for the sake of evaluating different classifiers, we use a number of different classifiers. The classifiers we use are Support Vector Classifier, K-Nearest Neighbors Classifier, and Linear Bayes Normal Classifier. Besides those mentioned classifiers, we also use classifier combination methods and use various combinations of those classifiers.

In Figure 6.3 the work flow of action recognition can be seen. The generated feature sets are divided into train and test sets. 70% of the data is selected as the training set and 30% is used as the test set. Mentioned classifiers are trained using the training set and the test set is given to those trained classifiers in step 2 to detect the actual actions.

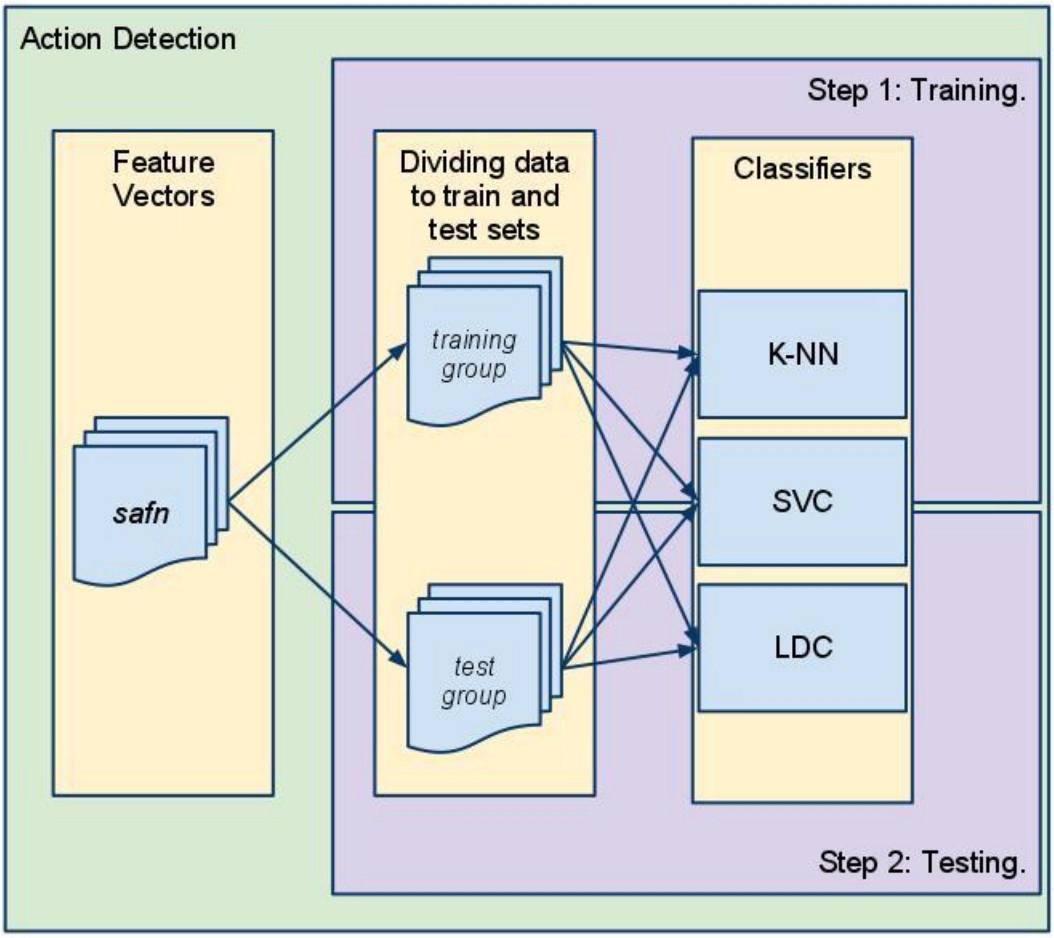


Figure 6.3: Action detection module: Processing generated features for sample based angles approach.

Leading to a rational detection result is crucial and therefore doing this operation only once might not lead to reasonable results. Therefore this procedure is repeated 5 times, generating different train and test sets every time and randomly so that the same operation can be applied to different samples as well. After all we get the mean error rate as the classification performance.

6.1.2 Silhouettes Approach

The silhouettes approach has many variations to be applied. As we explained in the relevant section, we generate several feature sets from the silhouettes images. More than that, this section is only processed with Kinect data because ETH Tool is used with the angles approach. The aim for that is to see if another approach rather than the angles approach could exceed the performance of angles approach. The reason why we don’t use ETH Tool for that method is because as explained in ETH section, ETH doesn’t really extract silhouettes information. It rather extracts ‘foreground’ section which aims at reducing the search space and accurate silhouettes extraction is not expected anyways. The aim is to roughly get rid of the background. However Kinect is pretty accurate when it comes to silhouettes information extraction.

For the sake of reminding the feature sets we extract from silhouettes, we mention them here again. Table 6.1 shows the 7 different feature sets we use for this approach. The general outline of the method is that all those feature sets are tested separately for the same classifier groups.

Table 6.1: Feature sets extracted for the Silhouettes Approach from the 2D silhouettes images and 3D depth silhouettes.

Extracted Features from Silhouettes Approach		
	Per frame	**Per sample**
X and Y projections from 2D Silhouettes	*sproj*	*sp*
Distributional features from 2D Silhouettes projections	*disfeat*	*sdf*
X and Y projections from 2D Silhouettes frame differencing	*diffr*	*sfrd*
X and Y projections from 2D Silhouettes feature differencing	*difffe*	*sfef*
X and Y projections from Depth Silhouettes	*dproj*	*dp*
Distributional features from Depth Silhouettes	*ddisfeat*	*ddf*
X and Y Projections from Depth Silhouettes frame differencing	*ddiffr*	*dfrd*

In Table 6.1 we can see the abbreviations of feature sets for both silhouette images and depth silhouettes. For the classification task we use the sets that are sample based, as mentioned before.

Figure 6.3 shows the work flow for this approach. We repeat the process 7 times for 7 different feature sets stated in Table 6.1. 70% of the data is randomly selected as the training set and rest is for testing. The feature sets we use for this approach include relatively higher dimensions. As an example, the x and y projections have dimensions more than 1000. Therefore it is important and simpler to map those features to smaller dimensions for the accuracy of classifiers. It can be said that much less number of features out of those feature vectors can indeed explain the variation in the dataset. The experiments show that indeed around 20 principal components of x and y projections explain more than 95% of the variation in the data.

7 Results and Analysis

7.1 Results

Features are extracted and they are used with the classifiers that are introduced in work flows in section 6.1. Our experiments can roughly be clustered in two groups; angles approach and silhouette based approach and the relatively different databases used for each.

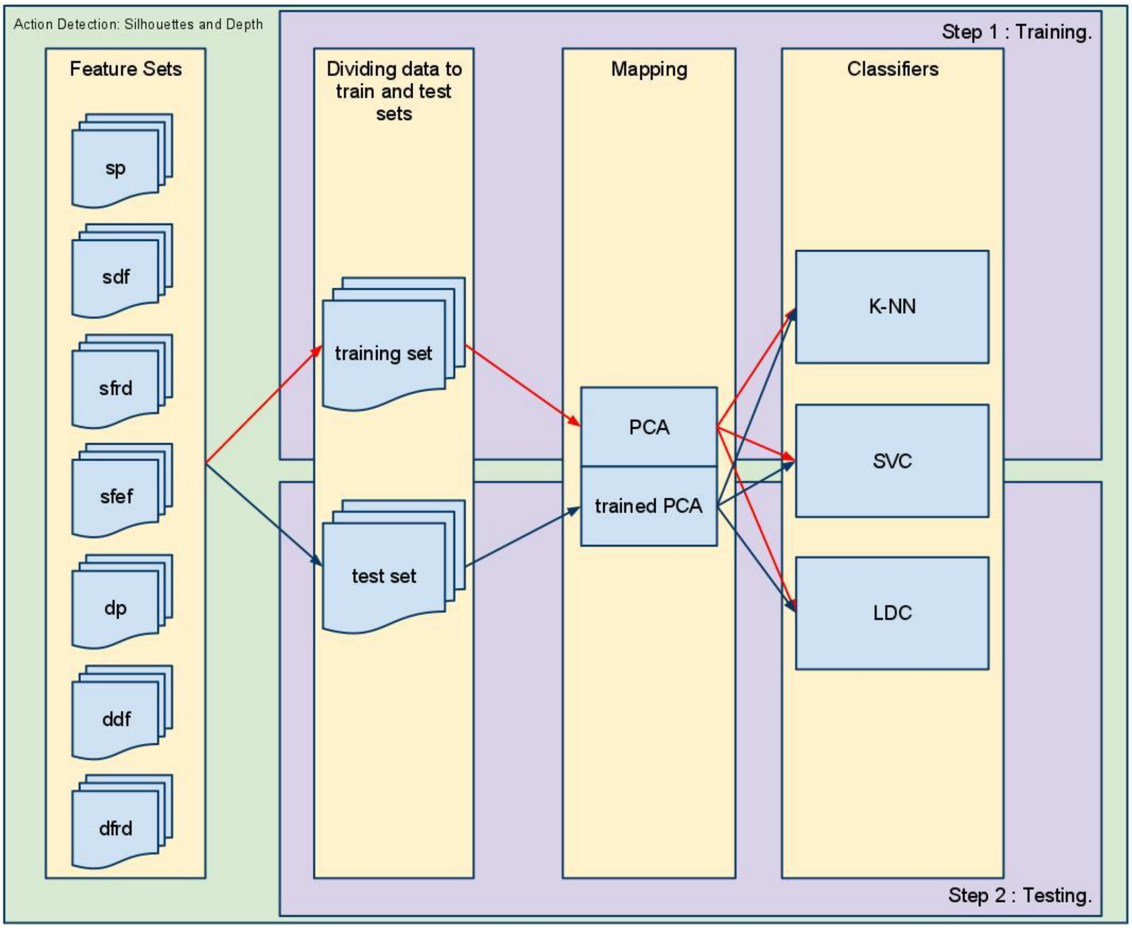


Figure 7.1: Action detection work flow using the feature sets extracted from silhouette and depth information.

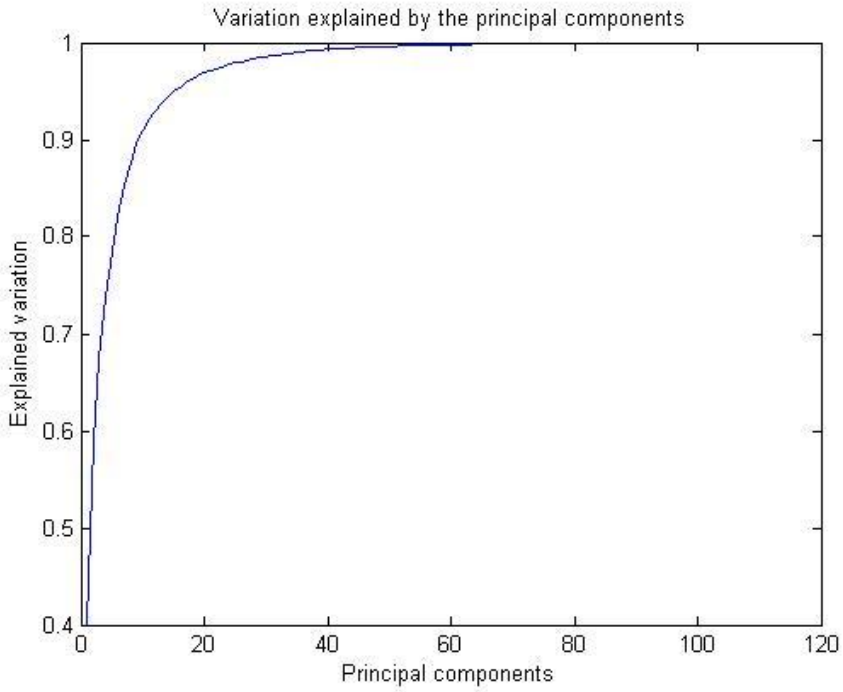


Figure 7.2: Principal component analysis on dataset with 4 actions. Plot shows the relation between the number of principal components and explained variation.

7.1.1 Angles Based Results

Those results are obtained by using two different approaches; varied dimension-temporal and sample based feature sets. Frame based approach is tested with HMM and sample based approach is tested with other classifiers.

When the temporal features are observed, it is seen that both ETH and Kinect tools indeed are missing some detections. ETH tool, as later will be explained in more detail, in some samples and some frames totally missed the person in the scene. Therefore for those frames, it returns a null result. For samples where there are no detections at all, there is nothing we can do because there are no features at all. Therefore we excluded those samples from the dataset before we run the tests. Besides that for those samples where there are undetected frames, we just exclude those frames from the sequence and construct a new feature matrix with the detections from the frames where the detections are made. This is done due to the reason that in an action sample, sometimes not all the frames, but even just a few of them indeed can explain the action. In ETH case, it is observed that the tool can indeed start detecting the person in case there is a move from the location where she/was not detected. Overall this process makes the number of rows of the feature matrix of a sample actually less than the number of frames there are in it. Yet this is a risk to take, because ETH or any other tool might miss detection at some frame, yet rest of the detections have to go with the flow.

It is also observed that actually Kinect miss some detection as well. However those are not totally missed people, yet Kinect decides that one or more of the limb sticks are indeed not visible. However our feature definition from the previous section requires having 5 angles per frame, yet Kinect might return null values for some of those. For those feature vectors they are usually arms that are not detected. Therefore the obtained feature sets are updated for that purpose. Undetected arms are assumed to be just hanging down the side of the body just like it would look in the neutral pose. This way it is stated to the classifier that the arm has no action and has no contribution in performing the action. In those cases it is seen that undetected arms are actually the ones that are not performing the action.

The problem of missing just one frame in between the sequence is not a problem for Kinect. It is observed in the data that once a limb is detected, it is not missed for a long while. This means that there is no one single frame missing detections in between, but rather a couple of frames in the beginning of the sequence or end.

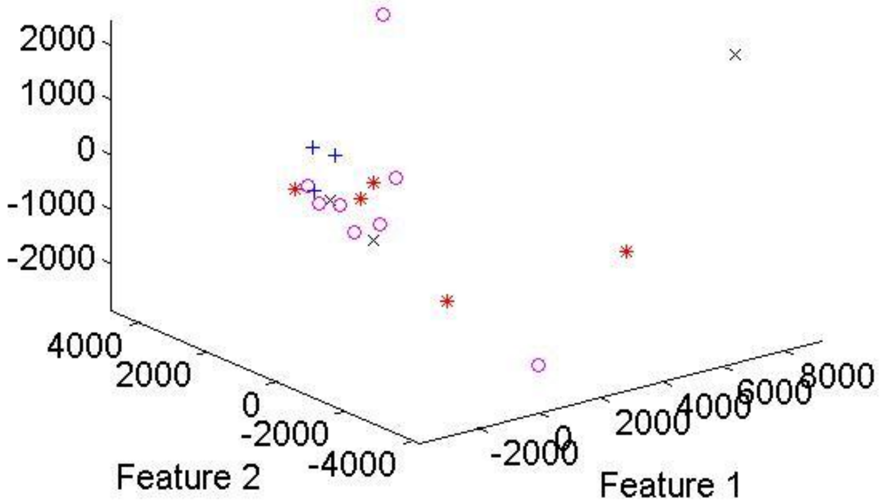


Figure 7.3: ETH Sample based angle features mapped to 3 dimensions.

In the figure above the visualization of sample based angle features extracted with the ETH tool can be seen in 3 dimensions. The table below shows the results of this part of the experiment.

Table 7.1: Error rates of different classifiers using the sample based angle features extracted by both ETH and Kinect tools.

	SVC	**LDC**	**K-NN**	**NMC**
ETH Tool	63,16	68,42	**57,89**	63,16
Kinect	69,57	73,91	69,57	**52,17**

The results above show the detection rates for feature sets where the feature vector represents the whole sequence without revealing the number of frames. The experiments done with the Hidden Markov Models where features are spatio-temporal are represented below.

Table 7.2: Error rates for different number of state with HMM Tool.

States	**ETH**	**Kinect**
2-states-HMM	42,50	**49,70**
4-states-HMM	**36,67**	55,60
6-states-HMM	54,17	52,03
8-states-HMM	49,17	53,82
10-states-HMM	62,50	53,17

From the results in Table 7.2 it can be seen that increasing the number of states actually decreases the performance. For ETH tool the lowest error rate is obtained for a 4 states model and it is 36.67%. For Kinect 2 states are enough to get a good performance and the resulting error rate is 49.70%. Furthermore it can be observed that considering the temporal relation between frames, by employing the HMM classifier is beneficial for the action recognition task, as the error rate decreases in both cases.

7.1.2 Silhouette Based Results

In this section we run 7 tests using 7 different feature sets that were described before in Chapter 5. For the sake of simplicity, we will observe the results in subsections belonging to each feature set.

For the feature set *sp*, experiments showed that the shifting and normalization that is done for extracting the distributional features from projections indeed improved the performance for this feature set as well. Therefore we use this approach here as well. The work flow can be seen in Figure 7.4.

For examining data and analyzing the discrimination power of the feature set, we used PCA method and plotted the first 3 principal components of the whole feature set. In Figure 7.5 that plot can be seen. Yet it is important to see the performance of classifiers. We used Leave one out validation method, in which one sample is left out as test set and all other samples are used for training. This process is repeated for all the samples in the dataset so that all samples can be tested.

In Figure 7.4 the work flow for this process can be seen. Figure 7.3 shows the mentioned visualization of samples with features mapped to 3 dimensions.

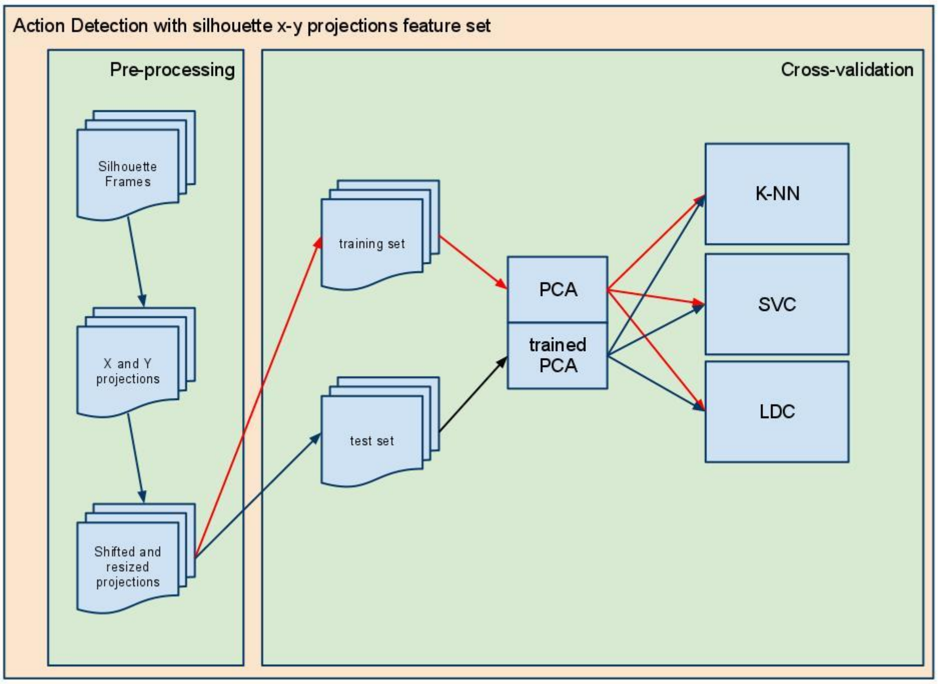


Figure 7.4: Detection work flow for *sp* feature set.

In Table 7.3 we can see the error rates of 3 different classifiers for this feature set. Obviously Linear Bayes Normal Classifier outperforms the other classifiers. For the aim of seeing the differences, we place here the confusion matrices of all 3 classifications.

Table 7.3: Error rate of classifiers after cross validation using the silhouettes x and y projections as features.

feature set	**SVC**	**LDC**	**K-NN**
Silhouette projections	32,26	26,88	38,71
Silhouette frame differencing	33,33	24,73	33,33
Silhouette projection differencing	35,48	29,03	36,56
Depth projections	37,63	27,96	37,63
Depth frame differen.	37,63	26,88	38,71

Table 7.4: Confusion matrix of LDC classifier with silhouette projections.

%	Browsing	Examining	Picking	Trying on	Waving	Shopping cart
Browsing	**44,44**	22,22	33,33	0	0	0
Examining	4,55	**81,82**	13,64	0	0	0
Picking	5,13	12,82	**79,49**	0	2,56	0
Trying on	0	16,67	16,67	**66,67**	0	0
Waving	0	0	14,29	14,29	**71,43**	0
Shopping cart	0	0	50,00	0	0	**50,00**

Table 7.5: Confusion matrix of SVC classifier with silhouette projections.

%	Browsing	Examining	Picking	Trying on	Waving	Shopping cart
Browsing	**55,56**	22,22	22,22	0	0	0
Examining	0	**54,55**	36,36	0	4,55	4,55
Picking	2,56	15,38	**82,05**	0	0	0
Trying on	8,33	8,33	25,00	**58,33**	0	0
Waving	0	0	14,29	14,29	**71,43**	0
Shopping cart	25,00	0	25,00	0	0	**50,00**

Table 7.6: Confusion matrix of K-NN classifier with silhouette projections.

%	Browsing	Examining	Picking	Trying on	Waving	Shopping cart
Browsing	**11,11**	22,22	55,56	11,11	0	0
Examining	9,09	**68,18**	13,64	4,55	4,55	0
Picking	5,13	5,13	**87,18**	2,56	0	0
Trying on	0	25,00	25,00	**50,00**	0	0
Waving	0	0	42,86	0	**57,14**	0
Shopping cart	25,00	25,00	25,00	0	0	**25,00**

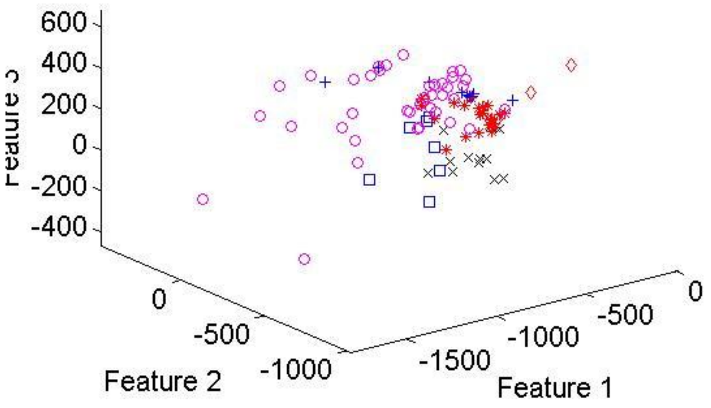


Figure 7.5: Silhouette frame difference projections mapped to 3D.

Table 7.7: Confusion matrix of LDC with silhouette frame differencing features.

%	**Browsing**	**Examining**	**Picking**	**Trying on**	**Waving**	**Shopping cart**
Browsing	**11,11**	22,22	55,56	0	0	11,11
Examining	0	**95,45**	4,55	0	0	0
Picking	5,13	10,26	**76,92**	2,56	0	5,13
Trying on	0	16,67	0	**83,33**	0	0
Waving	0	0	0	14,29	**85,71**	0
Shopping cart	25,00	0	25,00	0	0	**50,00**

Table 7.8: Confusion matrix of LDC with depth frame differencing features.

%	**Browsing**	**Examining**	**Picking**	**Trying on**	**Waving**	**Shopping cart**
Browsing	**22,22**	22,22	55,56	0	0	0
Examining	0	**90,91**	9,09	0	0	0
Picking	10,26	12,82	**74,36**	0	2,56	0
Trying on	0	25,00	0	**66,67**	8,33	0
Waving	0	0	0	14,29	**85,71**	0
Shopping cart	25,00	0	0	0	0	**75,00**

7.2 Analysis

In the previous section, the results of action detections are presented. In this section, based on obtained outcomes the different methods can be compared and evaluated. We will compare the ETH and Kinect tool, both by angles and silhouettes approach.

7.2.1 ETH vs. Kinect

The common method that was used both for ETH tool and Kinect is the angles approach. That is because ETH tool does not generate silhouette images. Therefore we rely on the angles performance to compare these two tools. Angles were also used with two different approaches, frame based features using Hidden Markov Models and sample based features using SVC, K-NN and LDC.

The database that was used for angles approach is relatively a smaller database and the reason for that is that both tools actually are not accurate enough for limb detections. Kinect is presented to be very successful with limb detection and person tracking; however the angle of the Kinect sensor is crucially important for that. In our experiments for the aim of viewing the table and the products as well Kinect sensor was placed at a relatively higher point. Yet in that position Kinect failed to track limbs. Kinect also requires doing the configuration for limb detection and tracking however even after a few seconds after the configuration, depending on the orientation of the person and the location of the sensor, it fails to track some or all of the limbs.

Another important thing about Kinect's limb tracking that should be noted is that Kinect is indeed aware of the occlusion and for example if one of the arms is missing, the stick configuration it shows just lacks that limb. This problem is indeed not a problem of Kinect but a general problem as an obstacle for this research.

There are a couple of problems observed with the ETH tool as well. As mentioned earlier, ETH tool relies on the upper body detection and then on the foreground highlighting. If any of those steps fails, then the rest also fails. If the upper body detection module fails to detect the person in the scene, the whole system just assumes that there is no body in the frame. On the other side, if the upper body detects a person and the foreground highlighting step returns a search space, limb detection module acts like there is definitely a person in the scene with all limbs absolutely visible. The problem of occlusion is not integrated in ETH tool, therefore if for example an arm of the person is indeed on the other side of the person and not visible to the sensor, ETH tool still return limb stick configurations for that limb too, and as could be expected it is false detection.

Occlusion problem is mostly avoided in our experiment. The occlusion that is faced most is caused by the occlusion by the product. That big part of an object that could occlude a huge part of the body can be seen in the 'tryon' action samples in which people are holding big objects like jackets etc.

In short two main problems that are faced with Kinect and ETH tools are; undetected limbs and frames with no detection at all, respectively. Here we explain those in numbers.

Table 7.9: Kinect: Rate of frames without detected limb relative to all the frames in the dataset.

	Upper right arm	Upper left arm	Lower right arm	Lower left arm	Torso	Total num of frames
Browsing	12,07	74,14	20,11	74,14	0	174
Examining	10,27	2,16	12,43	2,16	0	185
Picking	2,38	0	2,38	0	0	84
Trying on	12,07	6,90	12,07	6,90	0	58
Waving	15,79	75,44	21,05	75,44	0	57
Driving shopping cart	17,19	20,31	17,19	20,31	0	64

Table above shows the undetected limbs by Kinect tool. Those are the rational to the total number of frames in the action. Those numbers also give some information about the actions. It can be said that for Kinect it is easier to detect and track limbs, arms in particular in this case, with the picking action in which the arms are indeed leaning to outer boundaries of the body. Frame numbers of undetected arms are pretty low for this action. The actions that are most difficult for Kinect seem to be browsing and waving actions. Although waving action is expected to be easier, still the extreme position of the arms prevents Kinect from performing still well.

One important thing that was mentioned earlier as one of the potential problems was with the browsing action in which the person touches another object, or even the table, and Kinect labels that object as the person as well. This really damages the limb detection and tracking. Although in both browsing and examining action, arms are expected to be in front of the body and difficult to be detected, it is proved that that is not really a problem for Kinect because it is seen that it can perform well with examining action. However touching another object in the browsing action decreases the limb detection performance.

Table 7.10: Totally undetected rate of samples per action for the ETH Tool.

	Undetected sample rate (%)	**Total num of Samples**
Browsing	40,00	5
Examining	16,67	6
Picking	0,00	8
Trying on	25,00	4
Waving	66,67	3
Driving Shopping cart	33,33	3

It is indeed quite obvious that the limb detection for ETH tool is indeed quite low. There are samples where the tool completely missed the detection and in the other samples where there are detections, there are still some frames where the tool failed. With further investigation and research missing frames in between detected frames can be solved, however the samples that have no detection at all have no solution for the ETH tool. As can be seen in Table 7.10 this rate is also quite high. That actually explains why the detection rate with the angles approach is quite low. There are a lot of missing detections.

In Figure 7.6 we can see a frame where both ETH and Kinect tools successfully detects the limbs. The problem however is that in that sample, Kinect is capable of detecting all the limbs in all the frames. However the ETH tool fails to detect the person in 3 frames out of 7 frames in the sample in total. ETH is good for these specific frames, yet in terms of the samples Kinect outperforms ETH

Table 7.11: Undetected frame rate for the eth tool.

	Undetected Frame Rate (%)	Total num of Frames
Browsing	33,33	174
Examining	80,00	185
Picking	84,52	84
Trying on	62,07	58
Waving	54,39	57
Driving shopping cart	40,63	64

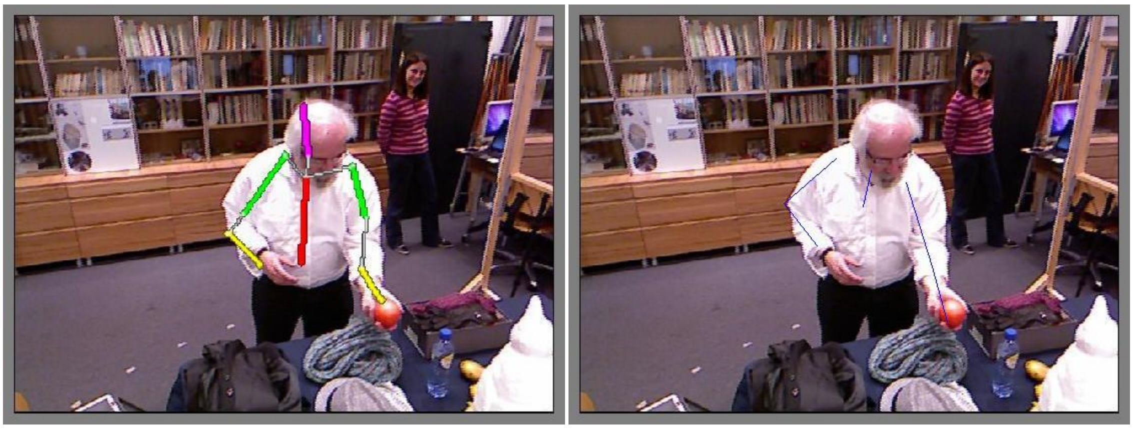


Figure 7.6: Limb detection comparison. Left: ETH tool. Right: Kinect.

In Figure 7.7 we can see a sample where both tools fail. The failure that the Kinect has is the common problem we mentioned above. Kinect is missing the left arm, yet it could still accurately detect the right arm. Browsing problem can yet be seen here though; Kinect labels some part of the table or the object as the person arm as well. In this example though, ETH fails hard. ETH is accurate with torso and upper arms yet lower arms are totally out of scope.

In this sample Kinect completely misses the left arm in all 26 frames. In two frames, it fails to detect right lower arm as well. ETH on the other hand detects 15 frames out of the 26 frames in this sample.

Above we have compared the detection capabilities of two tools ETH and Kinect. Those numbers show that with the current approach and performance they are actually not qualified enough to be used for further stages of an action detection system. Yet however to see the performance we still tested those tools and mentioned the results in the previous section. Table 7.1 shows the sample based feature detection performances of those two tools. It shows that angles features originated from ETH tool best performs with a K-NN classifier with an error rate of 57.89%. Obviously this is not qualified enough to be implemented in any recognition system. Kinect on the other hand best performs with Nearest Mean Classifier with an error rate of 52.17%. Although it seems to be a little better than the ETH results, none of them can be acceptable to be used in an action recognition system.

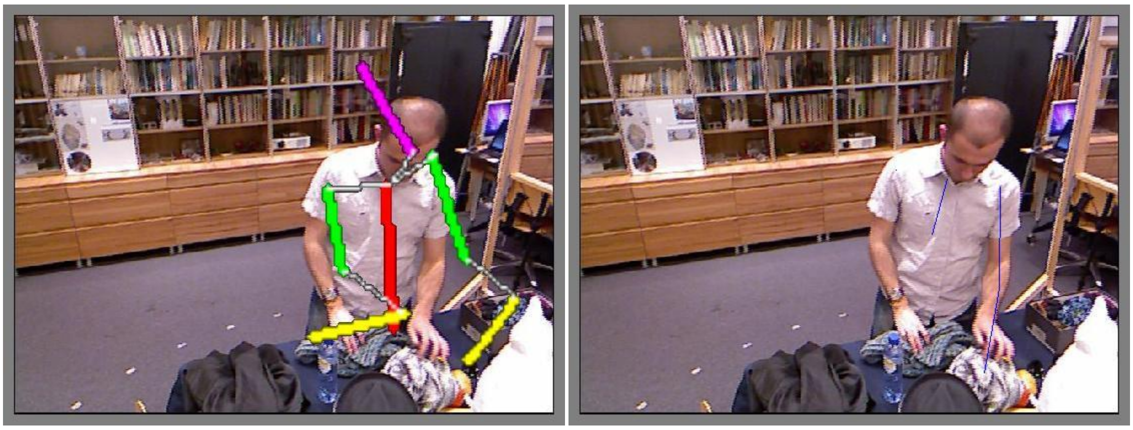


Figure 7.7: Limb detection failure sample from a browsing sample frame.

When it comes to evaluating the results of the angles approach the results of the HMM classification needs to be mentioned. The results for the procedure are shown in Table 7.2 and there are quite interesting results.

Comparing only ETH and Kinect tools on that method, it shows that for both tools that approach is not a good way for building an action recognition system because even the lowest error rate of both of them is 36.67%. Yet still to compare both, it is proved that ETH seems to performing better than Kinect with an error rate of 49.70%. However it should be considered that when the ETH tool is tested with that approach, there are samples where detections are missed in all frames therefore they are not even considered in the classification method. Besides that, because of those missing samples, there is a lack of samples for waving and driving with the shopping cart actions. Only samples from browsing, examining, picking and trying on are used for ‘ETH Angles with HMM’ approach.

One of the outcomes is that, although missing a whole sample seems to be a bad performance, ETH can still perform well and yield to better results with HMM. Kinect has detections on all samples but it is missing some of the limbs. This shows that, rather than missing a few limbs it is better to miss the whole sample. In other words, detecting a few limbs does not really yield to a good result. If the problem with the detection itself is solved for ETH, it appears to perform much better than Kinect and HMM combination. Therefore the problem that should be addressed in the future is Upper Body detection module of the ETH tool, especially for profile and difficult angle views.

The results with spatial features really are a failure. However ETH – HMM combination with angles and temporal representation of features is promising. Comparing those different sets of features for only ETH shows that ETH can actually yield to good results. It is promising. Yet Kinect is really not good for both sets of angle representation. As we see in the silhouettes results, Kinect should be used with silhouette features rather than limb sticks it produces.

HMM and number of states should also be evaluated. As the number of states is increased, the complexity increases. However in our case in this problem, it shows that it actually doesn't help to solve the problem but even makes it worse. The best performance of both tools is with small number of states. In both cases increasing the number of states leads to a higher error rate, especially with ETH tool; with 10 states, error rate goes up to 62.50%.

7.2.2 Angles vs. Silhouettes

When it comes to 2 different feature extraction approaches we used, we have Kinect results in hand and can evaluate the performance of those features sets because we don't generate silhouette information with the ETH tool. In this section, we explain the evaluation and comparison of those 2 different feature sets and also the different silhouette based feature sets as well.

Since we already know that angles approach is not performing well, we can start with comparing the different silhouette based features. All feature sets have detection rates more or less around 75%. It is known that there are methods in the stat-of-art that are much better than those. Yet it is also important to analyze and observe why those feature sets have such low detection rates.

Construction of those feature sets follow a basic work flow; plain x and y projections from the silhouettes are thought to be the first descriptive set and later on other feature sets are designed to improve the performance. The result of this can actually be seen. When only the silhouettes projections are used the performance is relatively a little bit lower than for other sets. Silhouette projection differencing although looks like it is descriptive method, when the data is observed, is not good enough and the error rate increases to 29% from 26%. On the other hand for 2D silhouette images, frame differencing appears to be the best choice for silhouette image information because it decreases the error rate to 24.73%. Those results belong to the LDC classifier and the same pattern can be seen for the K-NN classifier as well. However for SVC which is a quite popular and effective classification method, plain x and y projections do the job better than other feature sets.

Depth information is expected to explain more about the actions because it can also reveal the depth of the arms and the vanishing of the arms when they are in front of the body is not an issue. However for the current feature sets it is also proved that depth data does not improve the performance but yields to almost the same results. When plain x and y projections of silhouette, depth information and also the frame differencing are considered it is seen that depth information even decreases the performance.

We can also compare the performance of different classifiers on the constant feature set. In all feature sets it is seen that Linear Bayes Classifier is quite better than other approaches. LDC is particularly better with examining, picking and waving actions. One thing that should be noted is that K-NN classifier significantly fails with browsing and examining actions and mostly labels them as the picking action.

Results also show that some feature sets perform better with certain actions. Table 7.4, Table 7.7 and Table 7.8 show that with a good example. In all those 3 tables the confusion matrices of LDC classifier with 3 different feature sets can be seen. Plain silhouette projections are very good with examining action and good enough for picking action; however it fails more than half of the time for browsing. Silhouette frame differencing, which has a better performance than other feature sets prove to be an expert on examining, trying on and waving actions. However it shows that for browsing action it fails even more, with detection rate of 11.11%. Depth frame differencing features also fail in browsing action yet they do better than any other feature set on driving the shopping car action. All those observations put together show that for almost all features sets with all different classifiers, browsing appears to be one of the most difficult actions to be detected while picking is relatively an easier target for those approaches.

Angles approach significantly fails when it is compared to silhouettes approach. Few reasons are mentioned in the previous section about the data itself being not good enough. Yet when only those results are observed it is seen that silhouette based approaches significantly outperforms angles based approaches.

8 Conclusions and Future Work

8.1 Conclusions

For the aim of finding the answer of the research questions that were in hand at the beginning of this research, we designed experiments, made the recordings and came up with a dataset of video recordings of people performing shopping related actions in a shopping context. Two tools were used; ETH and Kinect to extract the information from those recordings. Angles and silhouette based features were extracted and they were tested with several classification methods.

The results that were obtained in this research are below the state-of-art action recognition method performances; however this research here addresses the shopping context which includes more complex problems. Therefore the results obtained are still promising and they can be improved with further work and research.

Our main aim in this research was to create a software framework that is capable of automated action recognition. As a starting point we searched the state-of-art methods and looked for inspiration. From this research we found a few tools and a few ideas to use. Kinect and ETH tools are the outcomes of our search in the field and we decided to use and evaluate if they could be used for a problem like this. They were not designed for such an environment yet we added our software on them and extended them to use them for our purposes. They gave promising results for this context. About defining features we saw that silhouettes and contours are quite a common approach so using the silhouette information we extracted our own features as well as the angles approach, which was indeed not quite common. We evaluated and compared those all and saw that silhouette approach is quite encouraging. Although angles approach was not as good as silhouettes, it was still promising to work on.

Our next goal was to examine the shopping context, come with the properties and action definitions. We also established that with the scenarios and basic action definitions. Our basic actions definition and scenarios combination showed that defining those actions is a good step for working in a shopping context; those actions were able to define a shopping scenario in more detail. In our recordings, our scenarios definitely lead to the basic actions we were looking for. We also analyzed and examined those basic actions and came to the conclusion that they each have characteristic and unique properties so a system that can do automatic action recognition is feasible.

Our research aim was to detect and recognize actions. Our extracted features and classification approach also achieved that with a reasonable acceptable rate. Our software of automated action recognition is the outcome of this aim and the proof that solving this problem is possible.

The tools we found were indeed not designed for this problem, in particular Kinect tool. Our aim was to evaluate and compare those tools and try to see if they could be adapted in such a system. From the very beginning of Kinect, we developed a system on top of it and completely adjusted it to the problem; it was originally designed for playing games and work only with the Xbox. We could adjust both ETH and Kinect tools to this problem and could evaluate their performances and compare them.

We wanted an environment and a new database of actions of this context. To our knowledge there is no database of actions in a shopping context. We established a recording environment, recreated the shopping context and constructed a database for this context. Different people perform the same actions but move in different ways. The database that is created involves 4 different people; however the methods that are applied do not depend on the people. Therefore it is also proved that actions have certain properties that are practiced by different people as well and features sets that capture these abstract properties of actions can enable independent action detection.

We wanted to work with pose estimation problem in this context and we established this with ETH and Kinect tools. It is also proved that none of them are the perfect solutions but with additional modules, like we created, they can be useful for other stages of the system. Weaknesses of ETH tool about detecting the upper bodies are spotted. Because of that problem, pose detection fails with ETH. However this problem can be overcome with Kinect’s silhouette information. Silhouettes of Kinect are proved to be quite precise and useful for this problem.

Cluttered background cause complexity to human detection and pose estimation. ETH uses the foreground highlighting step to get rid of the cluttered background which introduces more edges that damages the limb detection part in further stages. With Kinect we overcome this problem using the silhouettes. Silhouettes are generated using the depth sensor outcome and they are quite successful at removing the background. The problem of objects with the same depth as the person, touched by the person and being labeled as the person’s silhouette is proved to be less important because it even explains a lot about the action. Therefore extra labeling of objects as the person even helps the system.

Occlusion was one of the expected problems in this research. Indeed ETH suffers a lot from the occlusion problem. Kinect also suffers from it when the angles approach is wanted to be used. Kinect indeed doesn’t fail but it actually successfully detects unseen limbs but this is not good for the classification module. We overcome this problem with the silhouettes and depth silhouettes feature sets, in which actions are not represented by the arm locations but by considering the volume and shape of the whole body.

One of the aims of this research was to detect the interaction of people with products. The methods we used do not indeed detect the objects and track those objects, however they aim to detect the actions that are related to object interaction, like picking, examining etc. Results show that certain features and classifications methods can indeed distinguish those similar actions that are directly linked with objects. It is promising that the interaction with products can be actually automatically detected. As mentioned above methods tested can actually tell the difference between picking, browsing and examining, for example, and further more complex systems can process those basic actions and with additional modules, more accurate and precise information about this interaction as well.

Asking for help is represented in our dataset as waving to the assistant. In most approaches used here, the detection of this action is not good, yet however it is promising that when the data is carefully examined in more detail, it can be seen that waving action have very unique properties.

Problem of small volumes of movements, especially for browsing and examining actions is also solved with the silhouettes approach. Method using the angles might fail in those actions because of occlusion and missing detection of limbs. However silhouettes describe those small movements easily because they are much different than other actions where there is huge volume of movements.

In our experiments we extracted a couple of different features sets. The confusion matrices produced and introduced in this thesis show that certain feature sets can be more discriminative for certain actions. It is worth noting that this shows every action has definitely very specific and unique properties and some feature set really brings that in front.

8.2 Future Work

Our experiments showed us there can be a lot of future work for improving this system and it is encouraging to go into deeper and in more details in this domain. Below we list some of the items that should be the future work for this research.

- Depth information can be examined more to extract more discriminative features. When visually observed it is seen that indeed depth silhouettes carry a lot of information about the pose in a frame and action in a sample. Therefore extracting the most powerful feature set from the depth information should be investigated further on.

- For detecting the decision regarding a product, we need a sequence of basic actions that were investigated in this research. As mentioned above, object detection and tracking methods could be investigated and integrated with this system here for a more developed, complex and capable system.

- We extracted several feature sets and experienced that indeed individual feature sets can perform better with a certain action class. In a further work for this setting, those relatively more powerful feature set and action relations can be investigated in more detail to yield to even more discriminative feature sets. More than that, there are methods that improve performance with feature fusion. This could be part of the future work.

- We examined the basic actions. From the producer point of view, those information that are produced here and will be produced when sequence of basic actions are also evaluated can be used to extract statistical information about consumer-product relations.

- We detected that certain features or classifiers work better with certain actions. Therefore a topic of further research could be about feature fusion and using a cascaded approach. This way classifier can focus on the actions which they are best at.
- It is known to all that shopping stores are usually crowded places. Therefore future work should be a system that works simultaneously with multiple people. More than that interaction between people should be also in the scope of research.
- For the aim of solving the problems of the ETH tool and using the strength of Kinect, a hybrid system can be implemented. In this system, instead of using the upper-body detection and foreground highlighting steps of the ETH, Kinect can be used. Silhouettes can be used as masked to RGB frames and can be given to the parsing step of the ETH. That way there will be much less search space, no background at all, upper body detection is guaranteed and parsing step can get better color maps and stronger edges.
- Implementing this system in stores require a real-time system. Our experiments rely on segmented data. Future work can be done on this issue. Observations on the produced dataset show that those basic actions have more or less a range of number of frames. Shortest action, which is picking for example, is usually around 10 frames. Longer actions are usually those who don't really have an apex pose. Therefore even if the action takes a lot of frames, less number of frames can also define that action. Therefore a sliding window method that keeps searching sequences of 10 and a bit more frames can realize the real-time system. As stated in [17] such a system that has a constant size of a sliding window can indeed be implemented successfully.

.

9 References

[1] M. Andriluka, S. Roth, B. Schiele. “Pictorial Structures Revisited: People Detection and Articulated Pose Estimation”. *CVPR 2009*.

[2] V. Ferrari, M. Martin-Jimenez, A. Zisserman. “2D Human Pose Estimation in TV Shows. “ *Springer 2009*.

[3] I. Haritaoglu, D. Harwood, L. S. Davis. “Ghost – A Human Body Part Labeling System Using Silhouettes,” *ICPR 1998*.

[4] M. Eichner, V. Ferrari. “We Are Family: Joint Pose Estimation of Multiple Persons.”

[5] P.F. Felzenszwalb, D.P. Huttenlocher. “Pictorial Structures for Object Recognition,” *Springer 2005*.

[6] L. Sigal, M.J. Black. “Measure Locally, Reason Globally: Occlusion-sensitive Articulated Pose Estimation,” *Vision and Pattern Recognition, 2006*.

[7] T.B. Moeslund, E. Granum. “A Survey of Computer Vision-based Human Motion Capture,” *Computer Vision and Image Understanding, 2001*.

[8] T.B. Moeslund, A. Hilton, V. Kruger. “A Survey of Advances in Vision-based Human Motion Capture and Analysis,” *Computer Vision and Image Understanding, 2006.*

[9] D. Ramanan. “Learning to Parse Images of Articulated Bodies,” *Advances in Neural Information Processing Systems, 2007*.

[10] R. Ronfard, C. Schmid, B. Triggs. “Learning to Parse Pictures of People,” *ECCV 2002*.

[11] S. Johnson, M. Everingham. “Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation”

[12] C. Stauffer, W.E.L. Grimson. “Adaptive Background Mixture Models for Real-time Tracking” *CVPR 2009*.

[13] R.C. v. Dalen. “Lexical Stress in Speech Recognition.” Delft University of Technology, 2005.

[14] Karin F. Driel, “Building a Visual Speech Recognizer.” Delft University of Technology, 2009.

[15] H. Jiang, Mark S. Drew, Ze-Nian Li. “Action Detection in Cluttered Video with Successive Convex Matching.” IEEE Transactions on Circuits and Systems for Video Technology, vol. 20, no. 1. 2010.

[16] Kai Guo, P. Ishwar, J. Konrad. “Action Recognition in Video by Sparse Representation on Covariance Manifolds of Silhouette Tunnels.” ICPR, 2010.

[17] Konrad Schindler, Luc Van Gool. “Action Snippets: How Many Frames Does Human Action Recognition Require?” IEEE Computer Society Conference on Computer Vision, 2007.

[18] L. Gorelick, M. Blank, E. Shechtman, M. Irani, R. Basri. “Actions as Space-Time Shapes.” IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 29, no. 12, 2007.

[19] L.R. Rabiner, B.H. Juang. “An Introduction to Hidden Markov Models.” IEEE ASSP Magazine, 1986.

[20] D. Ramanan, D.A. Forsyth. “Automatic Annotation of Everyday Movements.” Technical Report, Tech. Rep. No. UCB//CSD-03-1262, Berkeley, CA. 2003.

[21] I. Haritaoglu, R. Cutler, D. Harwood, L.S. Davis. “Backpack: Detection of People Carrying Objects Using Silhouettes.” International Conference on Computer Vision, Corfu, Greece. 1999.

[22] M. Brand, N. Oliver, A. Pentland. “Coupled Hidden Markov Models for Complex Action Recognition.” Proceedings of IEEE Computer Vision and Pattern Recogniton. 1996.

[23] D.J. Moore, I.A. Essa, M.H. Hayes. "Exploiting Human Actions and Object Context for Recognition Tasks." IEEE International Conference on Computer Vision, Corfu, Greece. 1999.

[24] D. Weinland, R. Ronfard, E. Boyer. "Free Viewpoint Action Recognition using Motion History Volumes." Computer Vis. Image Understand., vol. 104, no. 2, pp. 249-257. 2006.

[25] J.Y. Chang, J.J. Shyu, C.W. Cho. "Fuzzy Rule Inference Based Human Activity Recognition." IEEE International Symposium on Intelligent Control. 2009.

[26] J. Yamato, J. Ohya, K. Ishii. "Recognizing Human Action in Time-Sequential Images using Hidden Markov Model." Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, pp. 379-385. 1992.

[27] C. Shüldt, I. Laptev, B. Caputo. "Recognizing Human Actions: A Local SVM Approach." Proceedings of the 17th International Conference on Pattern Recognition. 2004.

[28] M. Ekinci, E. Gedikli. "Silhouette Based Human Motion Detection and Analysis for Real-Time Automated Video Surveillance." Turkish Journal of Electrical Engineering and Computer Sciences 13(2), pp. 199-230. 2005.

[29] I. Haritaoglu, D. Harwood, L.S. Davis. "W4: Real-Time Surveillance of People and Their Activities" IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 8. 2000.

[30] S. Young, G. Evermann, M. Gales et. al. "The HTK Book (for HTK version 3.4)". Cambridge University Press, 2006.

[31] Ferdi van der Heijden, Robert P.W. Duin, Dick de Ridder, David M.J. Tax. "Classification, Parameter Estimation and State Estimation – An Engineering Approach Using Matlab". John Wiley & Sons. 2004.