Universidade Federal Fluminense

ANDRÉ LUIZ BRANDÃO

M5AIE: A Method for Body Part Detection, Tracking and Pose Classification using RGB-D Images

> NITERÓI 2013

UNIVERSIDADE FEDERAL FLUMINENSE

### ANDRÉ LUIZ BRANDÃO

## M5AIE: A Method for Body Part Detection, Tracking and Pose Classification using RGB-D Images

Thesis presented to the Computing Graduate Program of the Universidade Federal Fluminense in partial fulfillment of the requirements for the degree of Doctor of Science. Topic Area: Visual Computing and Interfaces.

Advisor: Esteban Walter Gonzalez Clua

Co-advisor: Leandro Augusto Frata Fernandes

> NITERÓI 2013

#### Ficha Catalográfica elaborada pela Biblioteca da Escola de Engenharia e Instituto de Computação da UFF

B817 Brandão, André Luiz M5AIE : a method for body part detection, tracking and pose classification using RGB–D images / André Luiz Brandão. – Niterói, RJ : [s.n.], 2013. 85 f.
Tese (Doutorado em Computação) - Universidade Federal Fluminense, 2013. Orientadores: Esteban Walter Gonzalez Clua, Leandro Augusto Frata Fernandes.
1. Algoritmo de classificação. 2. Interação homem-máquina. 3. Processamento de imagem. 4. Subtração de fundo. 5. Classificação de pose humana. I. Título. CDD 005.133

### M5AIE: A Method for Body Part Detection, Tracking and Pose Classification using RGB-D Images

André Luiz Brandão

Thesis submitted to the Computing Graduate Program of the Universidade Federal Fluminense as a partial requisite for obtaining the degree of Doctor of Science. Topic Area: Visual Computing and Interfaces.

Aprooved by:

Prof. D.Sc. Esteban Walter Gonzalez Clua / IC-UFF (Advisor)

Prof. D.Sc. Leandro Augusto Frata Fernandes / IC-UFF

(Co-advisor)

Prof. D.Sc. Didier Stricker / TU-Kaiserslautern

Rox Ukrue & Ulda ( (th

Prof. D.Sc. Rosa Maria E. M. Costa / UERJ

Prof. D.Sc. Anselmo Antunes Montenegro / IC-UFF

Prof. D.Sc. Aura Conci / IC-UFF

Niterói, September 3<sup>rd</sup>, 2013.

To my parents, my sisters, friends and Flávia.

"I've missed more than 9000 shots in my career. I've lost almost 300 games. 26 times, I've been trusted to take the game winning shot and missed. I've failed over and over and over again in my life. And that is why I succeed."

Michael Jordan

## Acknowledgments

I would like to express my gratitude to my supervisor, Professor Esteban Clua, for all his patience and all the opportunities he gives me every day to grow as a researcher and as a person. I also want to thank my co-advisor Professor Leandro Augusto Frata Fernandes for all his technical knowledge that he shared with me since the beginning of our cooperation. I also want to express my gratitude to the committee members for the important contributions and suggestions that helped to improve the quality of this thesis. I would like to express, in particular, my special gratitude to Professor Didier Stricker for accepting me as an external researcher at DFKI for six months in Kaiserslautern, Germany. That period was the best period of my professional live until the end of my thesis.

A great number of people helped me in several ways. I would like to thank:

- From DFKI, Leivy Michelly Kaul (for all the "miracles" and frienship), Klaus Greff, Yan Cui, Bernd Krolla, Daniel Steffen, Attila Reiss, Markus Weber, Frank Michel, Jan Hirzel, Nils Petersen, Gabriele Bleser, Gerd Reis, Vladimir Hasko, José Manuel Amaral Henriques, Miguel Fernando Portela Guimaraes de Sousa, Stephan Krauß, Johannes Köhler, Uwe Mayer, Markus Miezal, Tobias Nöll, Alain Pagani, Sarvenaz Salehi, Shuyan Liu, Navid Mostofi, Hiroshi Yonekubo, Pablo Abad, Jean-Marc Hengen, and mainly Christiano Gava (for everything you have done for me as a colleague and as a friend).
- From Kaiserslautern, but not from DFKI: Dasha Pupashenko, Misha Pupashenko, Ramy Nazier, Ilham Kurnia, Viktor Duke, Olga Bondari, Bina Dav, Leman Incirliler, Burcu Akın, Tung Vo Dinh, Antonio Gutierrez, Ayşe Dilara Aydın, Seher Balkanlı, Franck Gonzalez, Ludovic Delval, Injy Khairy, Mohamed Mohsen Khamis, Nermeen Abdel-Aziz, Peter Ezzat, Mohamed Selim, Caroline Nabil Sabty, Omar Naji, Nazly Sabbour, Ahmed Khattab, Kalliopi Katsika, Pratip Chakraborty, Arvind Reddy, Himanshu Saxena, Neeraj Saini, Raja Rajesh Sattiraju and Robert Marxen.

- From MediaLab-UFF: Bruno Moreira, Daniel Madeira, Tiago Bonini, Erick Passos, Heron Souza Marques, Jose Ricardo da Silva, Luiz Guilherme, Marcelo Zamith, Mark Joselli, Micheli Knechtel, Thales Luis Sabino, Diego Barboza, Felipe Pedrosa Martinez, João Bentes, Luis Valente, Marco Aurelio, and mainly to Giancarlo Taveira and Christian Ruff (for their friendship and sharing the apartment).
- From the Jecripe team: Bruno Queiroz, Fernando Ribeiro, Caroll Bulcão and specially Pedro Thiago Mourão (for friendship and turning things more interesting than they really are).
- From UFF: Cristina Nader Vasconcelos, Daniela Trevisan, Simone Martins, Diego Brandão, Sergio Carvalho, Teresa Cancela, Viviane Aceti, Anand Subramanian, and mainly to Juliana Mendes (for tolerating me, even in those days I took her patience away) and Marister Outão (another person who made some miracles for me).
- From UFSM: Professor Cesar Pozzer (for encouraging me every time we met each other).
- To all my friends that were my undergraduate colleagues, specially Alex Marin, Caco Beck, Cássio Ehlers, Daniel Biasoli, Fabricio Sperandio and Wagner de Morais.
- To my childhood friends: Daniel Gressler, Mateus Marques and his wife Aline Siqueira, and Ricardo Marques.
- To Therezinha Barbosa, her son Igor, her daughter Ticiana Senra and all her family for helping me since the first day I arrived at Niterói and including me as a family member. I will never forget you!
- To my family members: Deodoro (my father), Silvia (my mother), Letícia (my sister) and Rodrigo (my brother-in-law), Lenisa (my sister) and Virgilio (my brother-in-law) for the unconditional love and everyday support.
- To my beloved Flávia Feitosa (my girlfriend) for supporting me, for the example, discipline, advises and affection. Without you, my path would be much harder than it already was. I want to also express my gratitude to your family, specially your parents Emilson and Vera.
- To CAPES and DAAD for financial support.

To all of you, thank you very, very, much!!!

## Resumo

A detecção e rastreamento automáticos de partes do corpo em imagens coloridas é altamente sensível à características de aparência, como iluminação, cor de pele e roupas. Como resultado, o uso de informação de profundidade tem sido mostrado como uma alternativa atrativa a imagens coloridas, ao variar as condições de iluminação. Também, detecção e rastreamento de partes do corpo ainda é um problema que trás desafios, principalmente porque o feitio e a profundidade dos corpos em imagem podem mudar, dependendo da perspectiva. Esta tese apresenta um método de detecção e rastreamento, chamado M5AIE (acrônimo para: *M*edial Axis transformation, Adapted AGEX, ASIFT, Aligned *I*mages, e *E*stimation), que utiliza cores e informação de profundidade.

Contudo, para filtrar parte do montante inútil de informação em imagens, primeiro é necessário tratar a tarefa de subtração de fundo. Câmeras de profundidade devem oferecer uma alternativa para abordagens nessa tarefa, pois informações de profundidade parecem ser mais adequadas para subtração de fundo. O M5AIE preenche este espaço ao examinar alguns dos mais conhecidos algoritmos de subtração de fundo para serem utilizados com informações de profundidade.

O método M5AIE aplica a abordagem de distância acumulativa extrema (AGEX) para detecção de pontos candidatos a partes do corpo. O M5AIE faz uso do algoritmo ASIFT para extração de características e adapta o método convencional de correspondências para fazer o rastreamento e rotulação das partes do corpo em sequências de imagens que contêm informações de cores e profundidade.

Depois, são endereçados os algoritmos de previsão para classificar poses humanas. Algoritmos de classificação tornaram-se importantes em jogos, principalmente por causa da introdução de novos paradigmas de interação como a Interface Natural de Usuário (*Natural User Interface – NUI*). É possível de encontrar vários trabalhos que fazem uso de algoritmos de classificação, na literatura, mas eles ainda não apresentam estudos que comparam diferentes algoritmos no contexto de classificação de poses humanas.

Também, é proposto uma análise de algoritmos de classificação, utilizando nosso método M5AIE para detecção e rastreamento de partes do corpo, com diferentes algoritmos: C4.5 Árvore de Decisão de Ganho de razão, Classificador Naïve Bayes e algoritmo de vizinhança KNN. Como consequência do nosso estudo, são apresentados os resultados que podem auxiliar pesquisadores na escolha entre os algoritmos selecionados a serem utilizados na classificação de poses humanas no contexto de jogos digitais.

## Abstract

The automatic detection and tracking of human body parts in color images is highly sensitive to appearance features such as illumination, skin color and clothes. As a result, the use of depth information has been shown to be an attractive alternative to color images due to its invariance to lighting conditions. Also, body part detection and tracking is still a challenging problem, mainly because the shape and depth of the imaged body can change depending on the perspective. This thesis presents a body part detection and tracking method, called M5AIE (acronym for *M*edial Axis transformation, *A*dapted AGEX, ASIFT, Aligned Images, and Estimation), that uses both color and depth information.

However, to filter part of the useless amount of information in images, the M5AIE first addresses how it handles the background subtraction task. Depth cameras might offer a compelling alternative to background subtraction approaches, because depth information seems to be better suited for the task. The M5AIE fills this gap by examining some well known background subtraction algorithms for the use with depth images.

The M5AIE method has applied a modified Accumulative Geodesic Extrema (AGEX) approach for detecting body part candidates. It also has used the Affine-SIFT (ASIFT) algorithm for feature extraction, and has an adapted conventional matching method to perform tracking and labeling of body parts in a sequence of images that has color and depth information.

Afterwards, the prediction algorithms are addressed to classify human poses. Classification algorithms became an important subject in games, mainly because of the introduction of new interaction paradigms such as the Natural User Interfaces (NUI). It is possible to find many works that make use of classification algorithms in the literature, but they still do not provide any study that compares different classification algorithms in the context of human pose classification.

Also a detailed classification algorithms analysis is proposed, using the M5AIE method with different algorithms: C4.5 Gain Ratio Decision Tree, Naïve Bayes Classifier and K-Nearest Neighbor (KNN) Algorithm. As a consequence of this study, results are provided to help researchers to choose among the selected algorithms to use in human pose classification in digital games context.

## Palavras-chave

- 1. Detecção de partes do corpo
- 2. Rastreamento de partes do corpo
- 3. Reconhecimento de pose
- 4. Classificação de poses
- 5. Subtração de fundo
- 6. Algoritmos de classificação

## Keywords

- 1. Body part detection
- 2. Body part tracking
- 3. Pose recognition
- 4. Pose classification
- 5. Background subtraction
- 6. Classification algorithms

# Glossary

AGEX	:	Accumulative Geodesic EXtrema Points
ASIFT	:	Affine Scale Invariant Feature Transform
AUC	:	Area Under the Curve
KNN	:	K-Nearest Neighbor
NUI	:	Natural User Interface
RGB-D	:	RGB and Depth Information
$\operatorname{SIFT}$	:	Scale Invariant Feature Transform
SLAM	:	Simultaneously Localization And Mapping
SVM	:	Support Vector Machine

## Contents

Lis	List of Figures xiii					
Lis	st of '	Tables x	٢v			
1	Intro	troduction				
	1.1	Motivation	2			
	1.2	Thesis Statement	3			
		1.2.1 Background Subtraction	4			
		1.2.2 Body Part Detection and Tracking	5			
		1.2.3 Human Pose Classification	6			
	1.3	Thesis Outline	6			
2	Back	ground Subtraction: Information Reduction for Human Body Part Detection	8			
	2.1	Kinect Depth Image Characteristics	10			
	2.2	Background Subtraction Challenges	11			
	2.3	Basic Methods	12			
	2.4	Adaptations	14			
	2.5	Experiments and Results	15			
	2.6	Discussion	18			
3	M5A	IE: The Body Part Detection and Tracking Method	19			
	3.1	Related Work	20			
	3.2	The M5AIE Method	22			

		291	Digenete	Madial Aria Through Distance Transformation	იი		
		3.2.1	2.1 Discrete Medial Axis Through Distance Transformation				
		3.2.2	Graph (	Construction Based on RGB-D Images	25		
		3.2.3	Accumu	lative Geodesic Extrema Points	25		
		3.2.4	Body Pa	art Labeling	26		
		3.2.5	ASIFT-	Based Tracking of AGEX Points	26		
			3.2.5.1	Diffusion in the background of sub-images	28		
			3.2.5.2	Searching in a region instead of searching for coordinates only	28		
			3.2.5.3	Use of tiny images instead of complete frames	29		
			3.2.5.4	Body-parts position estimation	30		
	3.3	Exper	iments an	d Results	32		
	3.4	Discus	sion		36		
4	Ana	lysis of	the Class	sification Algorithms Using M5AIE-Extracted Human Poses	37		
	4.1	Relate	ed Work		38		
	4.2	Pose (	Classification				
		4.2.1	C4.5 Ga	in Ratio Decision Tree	40		
		4.2.2	Naïve B	ayes Classifier	41		
		4.2.3	K-Neare	est Neighbor Classifier	42		
	4.3	Bound	ling Box a	and Grid	43		
	4.4	Exper	iments an	d Results	45		
		4.4.1	Testing	Categorical Attributes	45		
		4.4.2	Testing	Numerical Attributes	47		
	4.5	Discus	sion		55		
5	Con	clusions	s and Fut	ure Directions	57		
л,					00		

60

# List of Figures

1.1	Music's House: Betinho is dancing to music inside the house	3
2.1	Foreground objects (right) are detected in depth images (left) taken by a static depth camera	9
2.2	Examples of training images that have pure background	15
2.3	Sample images from the segmentation for all of the methods and all of the sequences. Every image is presented with and without filtering (see Post-Processing in Section 2.4).	17
3.1	Flowchart of the proposed M5AIE approach applied to RGB-D images to identify the pose of the imaged subject. See Section 3.2 for details. The question mark after the AGEX Point Detection stage verifies whether the Labeling stage of the algorithm can be performed	23
3.2	A pair of images related to the same frame of a sequence: (a) RGB infor- mation is used to color the foreground pixels and in the tracking stages of our algorithm. (b) Depth information (displayed with false-color) is used to distinguish the background and foreground pixels	24
3.3	A discrete medial axis transformation (b) is applied in the segmented RGB- D image (a) to reduce the number of pixels to be considered during the AGEX-graph construction. The red pixels in (a) and (b) are the AGEX points	27
3.4	Filtering the matching result.	29
3.5	Sub-images of the detected five main body parts.	31
3.6	Background-diffused version of the sub-images presented in Figure 3.5. $$ .	31
3.7	Kinect performs asynchronous acquisition of RGB and depth images. As a result, the quality of the RGB-D alignment procedure performed by Kinect's API can be affected by rapid movements of the user, which leads to inconsistent RGB-D image formation	32
		04

3.8	Illustration of $T$ -pose and $dancing$ human poses in a game developed by	
	our research group that inspired our experiments	33
3.9	Illustration of <i>Play guitar</i> and <i>Play drums</i> human poses	33
3.10	Illustration of the <i>Play drums</i> human pose	34
4.1	A bounding box limits the human body, and it is divided into $N \times N$ cells.	
	In this example, $N = 8$	43
4.2	The aim of the training stage is to build a classification model	44
4.3	The classification model is used to predict the class of the tuples that are	
	being tested	45

# List of Tables

2.1	The results for the algorithms run on the test sequences. The rows with $e_+$					
	and $e_{-}$ represent false positives and false negatives respectively. The values					
	are specified with respect to the total number of background pixels in the					
	ground truth data. The lowest positive and negative errors are highlighted					
	for each test sequence.	16				
3.1	Image sequence evaluation	35				
4.1	Confusion Matrix for Naïve Classifier	46				
4.2	Confusion Matrix for C4.5 Gain Ratio Decision Tree.	46				
4.3	Image sequence evaluation for Volunteer A	49				
4.4	Image sequence evaluation for Volunteer B	50				
4.5	Image sequence evaluation for Volunteer C	51				
4.6	Results for $N = 8$	52				
4.7	Results for $N = 16. \ldots \ldots$	53				
4.8	Results for $N = 32$	54				
4.9	Results for $N = 64$ .	55				

## Chapter 1

## Introduction

Over the past few years, important advances have been achieved in Computer Vision research, especially in gesture recognition. Those advances have created many new possibilities of applications of Human-Computer Interaction, health-care and digital games [1].

Some approaches use photosensitive tags to decode optical signals and can capture the location of each point, its orientation, incident illumination and reflectance [2]. The photosensitive tags are markers that are imperceptible. However, the need to use tags can complicate their application in a digital game context. Beyond data-driven pose estimation approaches, there are some different classifications of movement recognition approaches: marker-based motion-capture (MoCap) that require retro-reflective markers or LEDs [3], bare-hand tracking [4, 5], hand-tracking with instrumented gloves (for example, Cyberglave) and, finally, color markers [6]. The retro-reflective markers are obtrusive, and another problem is the difficulties that are related to the financial cost of using instrumented gloves. Some work that is related to bare-hand tracking [4, 5] requires computationally expensive inference algorithms that search the high-dimensional pose space of the hand. Last, color markers [6] have demonstrated applications in limited domains or for short motion sequences [7].

An inexpensive approach is described in [7], in which the authors propose an easyto-use color glove. In that case, however, the color gloves are inexpensive, but it was necessary to use instrumented gloves in the construction of the database, and its application performs hand virtual reconstruction, which is unnecessary in our proposal.

There are some approaches that apply a two-dimensional image as an input and estimate the three-dimension coordinates of the selected keypoints [8, 9]. Moreover, the definition of Poselets is presented as a specific part of the human pose under a given viewpoint. It is defined with a set of examples that are close in 3D configuration space. The main contribution of [8, 9] is a notion of a part, a "poselet" and an algorithm for selecting good poselets. The authors constructed a dataset of humans in 3D (H3D), which provides the annotation of 15 types of regions of a person and 19 types of keypoint annotations, including joints, eyes, nose, and self-occlusion treatment. In our work, we do not use Poselets. Instead, we combine depth and RGB information to generate the RGB-D images.

### 1.1 Motivation

The use of games with movement recognition systems is an active research area. Among the many applications of these techniques, we can remark on the rehabilitation of patients using serious games. Applications can focus on rehabilitation while dividing the treatment into parts. Some examples of body-part rehabilitation can be found for balance rehabilitation [10], upper limb rehabilitation [11] and wrist rehabilitation [12].

The work of Schönauer et al. [13] describes a full body input for rehabilitation. The contribution of their work includes the implementation of a serious game that targets the rehabilitation of patients who have chronic pain in the lower back and neck. Schönauer et al. argue that the tracker used for their motion capture system, which is an io-tracker [14], is a passive marker that is based on an infrared optical motion-tracking system.

Another related study that addresses movements is the interactive dancing game presented by Tang et al. [15]. This game follows the design principle of a good game should be easy to play but difficult to master [16]. There are two modes in the game: the training mode and the freestyle mode. The dance moves are composed of various difficulty levels that are suitable for both novices and skilled players.

The related work presented good arguments to use movement recognition in games. Based on these statements, we decided to develop a game that is designed for children with Down syndrome, called Jecripe [17]. The Jecripe game is a previously developed system that inspired this thesis. This game has a main character, called Betinho, and it consists of a set of different activities that stimulate different cognitive abilities. The Jecripe game stimulates the movements of children with Down syndrome through imitating Betinho (see Figure 1.1).

The imitation ability is stimulated in tasks created for Jecripe in a virtual place called The Music House. In this house, the game demands the imitation of simple movements



Figure 1.1: Music's House: Betinho is dancing to music inside the house.

of the body, which are accompanied by singing interactive popular songs presented by the software.

We first demonstrated our intention to include movement recognition in the Jecripe game in 2009 [18]. This game was officially presented for the first time in 2010 [17]. A further study about communicability was also conducted [19] using the Semiotic Inspection method [20]. However, until the end of this thesis, it was not possible to integrate movement recognition in the Jecripe game. However, this game inspired all of the research that was performed for this work. This study presents each of the challenges that were addressed for pose recognition in image sequences. The following section provides a summary of the stages of this study.

#### 1.2 Thesis Statement

The launch of low-cost capture devices of depth images promoted facilities in gesture recognition research. In 2010, Microsoft Kinect was launched, and it is described in [1]. This paper presents how the device works and its applicability to digital games. The authors selected the possible user's playing movements (driving, kicking, running, and navigating menus), which are applied in Xbox 360 console games. The selected movements characterize a paradigm for interaction, which is the Natural User Interface (NUI).

In our work, we used Kinect to collect sequences of images. However, this study is not completely tied to the Microsoft sensor. Any other device that can combine color and depth information can be used in our work. This study is limited to indoor environments, static backgrounds, the static position and orientation of the sensor and single-user segmentation.

The context of body part detection and tracking requires information filtering to address only the necessary information. To filter the information, it is necessary to accomplish some tasks. The first task is to remove the most basic useless information, which is the background. Without the background, we can then handle the human body pixels. However, body part detection does not need all of the human body pixels. We decided to apply the Medial Axis transformation to filter an even larger amount of pixels. The Medial Axis provides the number of pixels that enable detection of the five main body parts. Then, a tracking method must be developed. We used a feature extraction and matching method to track each of the body parts from one image to the next image in a sequence.

Body part detection and tracking in image sequences is challenging because this task requires information filtering to bring about the use of less information. The resulting information must be structured because it will provide the detection of the body parts. The body parts are tracked with an algorithm, frame by frame, to store time sequence information. It is possible to use a feature extraction and matching algorithm as part of a tracking method because it compares two input images. Once there are human poses to be identified, it is possible to use the position of each body part in each image in a sequence to define the human poses that can be applied to classification algorithms for prediction purposes.

This thesis is composed of three stages, which we implemented to perform background subtraction, body part detection, and tracking and human pose classification. The following sections provide a description of each stage.

#### 1.2.1 Background Subtraction

Depth sensors have been used for a wide variety of problems, including skeleton tracking [21], gesture recognition [22], activity monitoring, collision detection [23], 3D reconstruction [24] and robotics. The task of background subtraction appears to be facilitated with the depth information that is available. It is, therefore, quite surprising to see that only a few studies on this subject can be found that use depth information for a background subtraction task. Early publications address background subtraction using stereoscopic cameras [25, 26]. There are papers that address Kinect and that mention the use of background subtraction [21, 27, 22, 28]; however, there were only small contributions that address the topic directly. In this task, we describe how we filled this gap, and we provide a starting point for further research.

#### **1.2.2** Body Part Detection and Tracking

The task of human body part detection and tracking is not trivial. Addressing the human body is challenging because its shape can be very different from one person to another. Additionally, humans have different skin colors, and clothes can vary in both their colors and shapes. These reasons, among others, make body part detection a complex task. Because body part detection can be used for body part tracking, certain aspects must be considered, such as the human skeleton and medial axis. Even knowing the human skeleton's profile, we must assume that there are many degrees of freedom [29].

Reliable results on body part detection and tracking tasks have been achieved by using depth information. Depth information outperforms intensity images in the sense that they intrinsically remove appearance features, such as the color of the skin, the color of the clothes and different background appearances, which can vary for different objects and colors [29]. Additionally, depth images provide extra information about the imaged objects, i.e., their actual geometry.

The object's geometry is given with the point distances between these objects and the sensor that forms a point cloud. The points of the point cloud can be used for body part detection, as vertices of a graph, and they can be connected with weighted edges. The weight of each edge is the Euclidian distance between the connected points. The generated graph can be used to detect body parts in the extremes of a graph. This approach is used in a method that is described by Plageman et al [29], for which the interesting points are called the Accumulative Geodesic EXtrema Points (AGEX).

The proposed solution is based on the key observation that once the body parts are detected in one frame, the same body parts can be used by matching methods for tracking each of them in the next frame. For this task, we describe the M5AIE Method, which detects and tracks the body parts. The name M5AIE is an acronym for each of the used concepts in our approach: Medial Axis transformation, for data filtering; Adapted

AGEX, for the body part detection; ASIFT, for the body parts tracking, Aligned Images (RGB-D), and Estimation, also for tracking.

The main contributions on this topic include the following:

- The combination of the AGEX and ASIFT methods using aligned RGB and depth images for labeling five major defined body parts (hands, feet and head); and
- Track each of the body parts by using an adapted ASIFT matching algorithm.

#### 1.2.3 Human Pose Classification

The application of movement recognition in games has many different approaches. Many of the movement recognition technologies apply video as an input to a recognition system. The literature provides a few studies that compare classification algorithms in the context of games. Huang et al. [30] compared Naïve Bayes, Decision Trees, and Support Vector Machines (SVMs) to evaluate which is the best measure to use when classification algorithms are compared. In [30], the authors used binary datasets and compared the use of two different measures: accuracy and Area Under the Curve (AUC). We use the accuracy measure in the experiments to evaluate the selected classification algorithms. Amor et al. [31] used intrusion detection system datasets to compare Naïve Bayes and Decision Trees. Amor et al. described how a Naïve Bayes classification algorithm can provide competitive results. The authors of the two studies [30, 31] did not consider datasets on human pose classification in their experiments.

To the best of our knowledge, no work in the literature has made a comparison among the classification algorithms in human pose recognition in the context of games. In this task, we propose an analysis of classification algorithms that use the M5AIE method: the C4.5 Gain Ratio Decision Tree [32], Naïve Bayes Classifier [33] and K-Nearest Neighbor (KNN) Classifier [34]. As a consequence of this study, the results can help researchers to choose among the selected algorithms for use in human pose classification in a digital games context. The main contribution on this topic is a comparative analysis of three classification algorithms in human pose classification.

### 1.3 Thesis Outline

This thesis is organized as follows:

- Chapter 2 presents a study of background subtraction algorithms. In this study, algorithms that use only RGB information were adapted to also use depth information. The experiments and results from our selected background subtraction algorithm are used in the following stages in this research.
- Chapter 3 presents our M5AIE Method. This method is used as the main method for the detection and tracking of body parts. The M5AIE method filters the information on images in sequences using the selected background subtraction for human body detection, which is the foreground. More information is filtered using the medial axis to generate a graph. This graph is used to detect five main body parts: head, hands and feet. The ASIFT method is used for tracking the objectives of the detected body parts. The relative positions of each of the body parts define human poses.
- Chapter 4 describes an analysis of three different classification algorithms in the context of human pose prediction. The use of the M5AIE method provides the relative positions of the five main body parts. These positions generate tuples that are used to evaluate the classification algorithms.
- Chapter 5 presents the concluding remarks and suggested future work.

### Chapter 2

# Background Subtraction: Information Reduction for Human Body Part Detection

The release of Microsoft's Kinect had an important impact on computer vision. This device changed the face of many problems, such as gesture recognition [22], activity monitoring, 3D reconstruction [24] and Simultaneously Localization and Mapping (SLAM) [35, 36].

The fusion of different sensors combined with a very low price makes Kinect an excellent choice for many applications. Certainly, its most interesting sensor is the depth camera, which Microsoft used to make a quality gesture control product for the Xbox 360. Since then, Kinect has been used for a wide variety of problems, including skeleton tracking [21], gesture recognition [22], activity monitoring, collision detection [23], 3D reconstruction [24] and robotics.

Many applications, especially those from the field of human computer interaction, utilize a static camera to track moving persons or objects. Those applications greatly benefit from background subtraction algorithms, which separate the foreground (objects of interest) from the potentially disturbing background (see Figure 2.1). This preprocessing step is well known in computer vision (for an overview, see [37]) and helps to reduce the complexity of further analysis; it can increase the quality of the overall result.

Additionally, the task of background subtraction is easier with a depth image at hand. It is, therefore, quite surprising to see that only a few studies on the subject can be found. Early publications address background subtraction based on the use of Kalman Filters [38]. Cheung and Kamath [39] provided a classical approach in urban traffic



Figure 2.1: Foreground objects (right) are detected in depth images (left) taken by a static depth camera

videos for performing background subtraction and moving object tracking. Stereoscopic cameras were also used for background subtraction [25, 26]. There are studies that use a depth sensor, which mention the use of background subtraction [21, 40, 22, 28]; however, until the end of 2011, there was no publication that directly addressed this topic. This chapter describes a study that was presented in [41], and it is part of a previous stage for human body part detection. One of the goals of the study described in this chapter is to compare four different background subtraction algorithms. We provide results that can help to understand why the use of depth information can facilitate the task of background subtraction. To prove this premise, in this chapter, the used images were produced with depth information only. The background subtraction is an image information reduction stage in the M5AIE method described in Chapter 3. Background subtraction includes how we make human body detection because the foreground is the human body in our context.

This chapter describes how we addressed the background subtraction task. Section 2.1 presents a characteristics analysis of the Kinect depth sensor. The impact of Kinect's depth sensor on the background subtraction problem is described in Section 2.2. Section 2.3 describes the selected four background subtraction algorithms. Adaptations on the algorithms to the domain of the depth images are presented in Section 2.4. Section 2.5 describes the experiments and results, which were made only for the background

subtraction problem.

### 2.1 Kinect Depth Image Characteristics

We start this section with an overview of the distinct characteristics of depth images provided by Kinect. These characteristics will provide the basis for analyzing the problems that are associated with the task of background subtraction. The functional principles of the Kinect will not be discussed in this work (Refer to [42] instead).

Although the Kinect's depth image resolution is  $640 \times 480$  pixels, the effective resolution is much lower because the depth calculation depends on small pixel clusters. The detection range is between 50 centimeters and approximately 5 meters, with a field of view of approximately 58°. Depth information is encoded by using 11 bits for the depth information and 1 bit to indicate an undefined value.

The most important property is the use of distance information and the possibility of combining it with color intensities. The use of the distance, or depth, information provides the possibility of making the image independent of the illumination, texture and color. However, direct sunlight can outshine the projected pattern, turning many pixels to undefined. Certain types of material properties can also hinder obtaining stable depth recognition, including high reflectiveness and transparency or dark colors.

An image that is generated using depth information only is, in this work, called a depth image. The depth image contains different types of disturbances and noise. We characterize the pixels according to those errors as follows:

- Stable: A fixed depth value with only a small variance; the value increases quadratically with the range (see [42]).
- Undefined: A special value that means that no depth information is available. This circumstance is typical for object shadows, direct sunlight, and objects that are below the minimum range of 50 centimeters. Shadowing occurs when the depth information of the background cannot be captured by the sensor because, for example, an object or a person blocks the capturing of this information. Each captured noise pixel and each pixel of the shadow receive a zero value.
- Uncertain: Switching in a random manner between the undefined and stable state. This circumstance is often the case for boundaries of undefined regions, reflections, transparencies, very dark objects, and fine-structured objects (e.g., hair).

• Alternating: Switching between two different stable values.

Occasionally, there are pixels that have "uncertain" and "alternating" characteristics, i.e., they switch between two different stable values and the undefined state. It is also important to note that alternation and uncertainty do not usually occur pixel-wise but instead tend to occur cluster-wise; therefore, contours can differ substantially from frame to frame.

### 2.2 Background Subtraction Challenges

Next, we give a summary of challenges faced by background subtraction algorithms (also referred to as foreground detection) that work on depth images. The list is based on a more detailed summary of [43]. We recited only the challenges that are related to depth images, and we also modified the descriptions to better reflect the characteristics of depth images as provided by the Kinect sensor.

- Moved Objects: The method should be able to adapt to changes in the background, such as a moved chair or a closed door.
- **Time of Day:** Direct sunlight can outshine the infrared patterns that are used for depth estimation, resulting in undefined pixels in the corresponding regions. If the illumination changes, then the state of the pixels in the affected regions might also change (to stable or undefined), which results in the pixel classification of "uncertain" (see Section 2.1). This circumstance is similar to the moved object problem.
- **Dynamic Background** This problem, originally referred to as *waving trees* in [43], can be caused by any constantly moving background object, e.g., slowly pivoting fans.
- **Bootstrapping:** In some environments, it is necessary to learn a background model in the presence of objects.
- **Foreground Aperture:** When a homogeneous background object moves, changes in the inner part might not be detected by a frame-to-frame difference algorithm. This scenario is especially true for depth images because there is no color and texture.
- Shadows and Uncertainty: The system is required to handle undefined and uncertain pixels (see Section 2.1) both in the foreground and in the background. Additionally,

foreground objects often cast shadows, which should not be considered to be foreground. This problem behaves differently with the Kinect because only the inherent shadow casting of the sensor is relevant. Additionally, these shadows always result in an undefined value, which makes it easy to rule them out as foreground.

There are more challenges listed by Toyama et al. in [43] and omitted in our work because they are out of our scope. The omitted challenges are: "Light Switch", "Sleeping Person", "Waking Person" and "Camouflage". We omitted the point "Light Switch" as artificial lighting because it does not affect the Kinect system. Furthermore, the challenges "Sleeping Person" and "Waking Person" were dropped because we believe that this task is better solved at a higher level that includes semantic knowledge<sup>1</sup>. Finally, the "Camouflage" problem was also omitted because the depth images lack both color and texture.

### 2.3 Basic Methods

Many background subtraction (and foreground detection) algorithms have been proposed. Cannons [44] provides an overview of the subject. Most of those algorithms were created with color images in mind. In this work, we chose four standard methods and adapted them to the segmentation of depth images, which allowed us to achieve three suitable and high quality possible solutions. The four methods were compared, and the method that had the best results was identified. The winner algorithm is used as the background subtraction algorithm, which is the previous stage of body part detection and tracking (Chapter 3).

The experiments and results were made specifically to know which of the four background subtraction algorithms have the best results for use as a previous stage in body part detection and tracking. The experiments did not aim to evaluate the used gestures or occlusions in any of the sequences. These experiments were used to choose the best background subtraction algorithm. Chapter 3 describes how the selected method is used in the M5AIE method, which uses RGB-D images. The experiments made at this stage were performed by using images that were generated only from depth information because this information is used for RGB-D image generation.

The selected methods are described in the following:

<sup>&</sup>lt;sup>1</sup>For an in-depth discussion please refer to [43]

- **First Frame Subtraction:** In this method, the first frame of the sequence is subtracted from every other frame. Absolute values that exceed a threshold are marked as foreground.
- Single Gaussian: In this method, Wren et al. [45] represent the 2D regions by clusters of points that are represented by spatial means and covariance matrices. In Wren et al.'s work, the environment is similar to ours: a single user and a static background. The authors represent the user and scene, with each having a different model. The scene is modeled as a texture, and each pixel of this model is associated with a mean color value and a distribution about that mean. Each pixel has a color distribution that is modeled with a Gaussian that models the scene. Because a human occludes the scene, the color distribution will vary frame by frame. Then, the human pixels will affect the scene model. The human's model and his/her pixels with the scene model make it possible to distinguish which pixels are part of the human and which pixels are part of the background. Because the background is considered to be class zero, only human pixels are considered in an image for the purpose of identifying a moving body. More details of the Single Gaussian method are in [45].
- **Codebook Model:** This more elaborate model was presented by Kim et al. [46]. It aggregates the sample values for each pixel into codewords. The Codebook model considers background values over a long image sequence. The background is modeled according to the clustering method proposed by Kohonen [47]. The number of Codebooks is the number of pixels in each image in the sequence. Thus, each pixel constructs a codebook that is based on codewords. The codewords are composed by color and brightness values at each instant. The model is constructed without making parametric assumptions such as a Gaussian distribution. After the construction of the background model, each pixel of an image is compared to the model. A pixel is classified as background if: "(1) the color distortion to some codeword is less than the detection threshold, and (2) its brightness lies within the brightness range of that codeword" [46].
- Minimum Background: The minimum background is one of the first models that was developed solely for depth images. The Minimum Background Subtraction algorithm is composed of training and subtraction stages. During the training stage, this approach limits the background values while considering the following assumptions: indoor environment, static background, and static position and orientation of the sensor. At this stage, a lookup matrix with the same size as the depth image is

created to store the minimum depth values that are assumed by each pixel during a frame sequence that captures only the background elements. The subtraction stage is applied to every subsequence frame. By comparing the stored minimum values with the current depth values, the approach is capable of distinguishing background and foreground pixels [40].

### 2.4 Adaptations

The presented methods must be adapted to work for depth images only. As a result, we developed and included different improvements: Uncertainty Treatment, Filling the Gaps and Post-Processing.

Uncertainty Treatment: Treating the undefined value (zero) as normal depth information leads to problems with almost every model (e.g., turning most shadows into foreground). Thus, the question arises: how do we treat the undefined values? Shadow pixels are certainly not foreground, but sometimes the shadow of an object falls onto the foreground. For example, a hand in front of the torso yields a shadow on the torso that is part of the foreground. The foreground can also contain undefined regions that are caused by reflective objects (glass, for example). These problems illustrate that, on a pixel level, it is impossible to decide whether an undefined value belongs to the foreground or not. This decision clearly requires additional knowledge (other sensory input, the region around the pixel). However, it is not the task of a background subtraction algorithm to perform complex reasoning. Background subtraction should merely be a preprocessing step (see Principle 1 in [43]). Thus, we decided to treat all of the undefined values as background for all of the presented methods.

Filling in the Gaps: Undefined pixel values can lead to gaps within the background model that is learned by each presented algorithm. Those gaps can lead to errors because every "defined" pixel value differs from an undefined background. Thus, depending on the chosen policy, they will either lead to false positives or to false negatives. To close the gaps, an image reconstruction algorithm (as in [48]) that attempts to estimate the correct values for the undefined regions can be used. This approach can obviously only reduce the errors that are induced by those gaps and does not completely eliminate them. According to the experiments, the method from [48] works quite well in practice.

**Post-Processing:** As discussed earlier, the depth images as generated by Kinect contain a substantial amount of noise. This consideration leads to a large number of false

positives in the form of very small blobs and thin edges around the objects. The desired foreground (i.e., humans), on the other hand, appears always quite large because of the range of constraints of the Kinect sensor. Therefore, morphological filters are an easy way of improving the final result. We experimented with the erode-dilate-operation and the median filter, but both of them change the contour of the desired foreground. A connected component analysis, on the other hand, combined with an area threshold is suitable to remove the false positive regions while keeping the foreground intact. This threshold can be quite high for most applications (1000 pixels in our case). This filtering is applied as a post-processing step to all of the presented methods. However, we also evaluate each of them without any filtering.

#### 2.5 Experiments and Results

To evaluate the different approaches, we recorded a set of three typical sequences for the application of human body tracking. All of them were recorded indoors at 30 fps and with a resolution of  $640 \times 480$ . Every sequence contains at least 100 training frames of pure background. Figure 2.2 illustrates an example training image with pure background.

- **Gesturing 1** : The camera shows a wall at a distance of approximately 3 meters for a few seconds. Then, a person enters and stands in front of the sensor performing some gestures (641 frames).
- **Gesturing 2** : The same as in the first sequence, but the background contains a large number of edges (643 frames).



(a) RGB image

(b) Depth image

Figure 2.2: Examples of training images that have pure background.

		Gesturing 1		Gesturing 2		Occlusion	
		$N_{\rm BG} = 179,896,685$		$N_{\rm BG} = 160,009,777$		$N_{\rm BG} = 164, 507, 583$	
		$N_{\rm FG} = 17,018,515$		$N_{\rm FG} = 37,519,823$		$N_{\rm FG} = 9,674,817$	
		plain,	filtered,	plain,	filtered,	plain,	filtered,
		in~%	in~%	in ~%	in~%	in ~%	in~%
First Frame	$e_+$	8.83	2.67	0.71	0.02	1.75	0.09
Subtraction	$e_{-}$	0.00	0.05	0.00	0.37	0.91	1.58
Single	$e_+$	0.52	0.19	1.91	0.07	2.40	0.85
Gaussian	$e_{-}$	8.33	9.15	9.98	13.55	6.42	9.02
Codebook	$e_+$	0.06	0.01	0.04	0.01	0.24	0.07
Model	$e_{-}$	0.00	0.03	0.00	0.30	1.32	1.84
Minimum	$e_+$	0.06	0.00	0.04	0.00	0.19	0.07
Background	$e_{-}$	0.00	0.02	0.00	0.37	1.20	1.94

Table 2.1: The results for the algorithms run on the test sequences. The rows with  $e_+$  and  $e_-$  represent false positives and false negatives respectively. The values are specified with respect to the total number of background pixels in the ground truth data. The lowest positive and negative errors are highlighted for each test sequence.

**Occlusion** : This sequence shows an office that has some chairs; then, a person enters and walks between the chairs. The ideal foreground for this sequence is marked manually in every frame (567 frames).

The Kinect depth sensor produces data that has a large amount of noise at the edges of the objects. For this type of noise, a single frame evaluation would not be representative. Therefore, we created ground truth videos that contain the ideal foreground segmentation for each sequence. The first two sequences were recorded in such a way that simple distance truncation cleanly separates the foreground. In the third sequence, the foreground was marked manually in each frame.

We measured the error of every algorithm using the absolute number of false positives  $N_{e_+}$  (background that was marked as foreground) and false negatives  $N_{e_-}$  (foreground that was marked as background). To establish some comparability we also measured an error ratio for every sequence, which is

$$e_{+} = \frac{N_{e_{+}}}{N_{BG}} \text{ and } e_{-} = \frac{N_{e_{-}}}{N_{FG}}$$
 (2.1)

respectively, where  $N_{\rm BG}$  and  $N_{\rm FG}$  are the total number of background and foreground pixels in the ground truth sequence. The results can be found in Table 2.1, and some selected frames for every video and method are shown in Figure 2.3.



Figure 2.3: Sample images from the segmentation for all of the methods and all of the sequences. Every image is presented with and without filtering (see Post-Processing in Section 2.4).

The *First Frame Subtraction* performs surprisingly well. Unfiltered, it produces the smallest false negative ratio among all of the considered algorithms. However, it is sensitive to all types of noise; as a result, depending on the background, there can be many false positives.

The statistical approach used by the *Single Gaussian* method is affected by the high variances of alternating pixels on the one hand and the low variance of stable pixels on the other hand. If the constant multiplied by the standard deviation is high, then there will be false negatives when a foreground object occludes a high variance region. If the constant is small, then stable pixels will emit a large amount of noise. Consequently, we concluded that the depth values provided by Kinect cannot be modeled effectively by a single Gaussian distribution.

The best overall results are achieved by the *Codebook Model* and the simple *Minimum Background* method. Both methods manage to eliminate the errors of uncertain and alternating regions without missing the desired foreground. Because the Minimum Background method is faster and simpler, we found it to be the best choice among the algorithms that we have considered.

### 2.6 Discussion

In this chapter, we described how to apply adaptations of four different approaches of background subtraction to depth images. These approaches were evaluated on three different test sequences using ground truth data. We have identified a simple and fast algorithm, the *Minimum Background* algorithm, which gave close to perfect results in our experiments. Thus, for the scenario of a static Kinect and a static background, the problem of background subtraction could be considered to be solved in the context of the present work. This finding clearly shows that the task of background subtraction is much easier when using only depth information compared with color images.

Nevertheless, there are still some open questions for future work in the context of background subtraction. Scenarios that have a moving Kinect or a dynamic background require more sophisticated algorithms. For the problem of bootstrapping, we suggest testing more complex background subtraction techniques, e.g., optical-flow, which attempts to handle foreground clutter during the training phase. The described results in this chapter were presented in [41]. Finally, the color camera of the Kinect could complement the depth camera for the task of background subtraction. The combination of depth and color information yields images that are referred to as RGB-D images. The next chapter describes how RGB-D image sequences can be used for body part detection and tracking.
# Chapter 3

# M5AIE: The Body Part Detection and Tracking Method

Human body part detection, tracking and pose classification are challenging tasks because a human's shape varies from one person to another. However, depth information outperforms intensity images in the sense that it intrinsically removes appearance features such as the color of the imaged objects [29]. Additionally, depth provides extra information about the scene, such as the actual geometry of the objects. RGB information is also used for detecting and tracking human body parts. The combination of RGB and depth information can be a powerful tool in this context.

RGB and depth images captured from the real world contain a large amount of information. Much of this information is irrelevant to human body part detection and pose recognition. Therefore, filtering the data is an important task to reduce the computational load of the body part detection. Another issue in this context is how to track the body parts in a video sequence. Knowledge about each body part position yields information on pose recognition.

Several methods exist for body part detection; most of these methods are for intensity cameras. Moeslund et al. [49] describe 350 related studies in the Computer Vision field. However, the present study takes advantage of the use of depth information, the Accumulative Geodesic Extrema, or AGEX, which is described by Plagemann et al. [29]. The AGEX points are detected through a graph construction that uses the human body's pixels. The Dijkstra algorithm [50] is applied to finding the extreme points of the generated graph. These extremes are considered to be the main body parts: head, hands and feet. The authors of AGEX use the results as an input for a motion capture system in a later study [51]. Baak et al. [52] also use the geodesic extrema points as one of the stages for full body pose reconstruction.

After the human body part detection and labeling, it is possible to track each of the parts. We use the ASIFT algorithm, which takes advantage of the SIFT method. The SIFT method was proposed by Lowe [53], and it provides matching between different images by extracting features that are invariant to rotation, illumination and isotropic scale. Yu and Morel [54, 55] made improvements to the SIFT method and proposed the Affine-Invariant SIFT (ASIFT) method. Both methods were originally applied to grayscale images.

This chapter presents a method for performing data filtering and body part detection and tracking. In this work (see Figure 3.1), independent RGB pixels and depth information of a frame are taken and produce an RGB-D image. Next, the background subtraction algorithm (selected from the study described in Chapter 2) reduces the image to foreground information only. In this case, the foreground is a person. To reduce even more the amount of data that is used in the pose estimation, a medial axis transformation is applied. The output of this stage will then be used to detect and label human body parts. With the labeled body parts in hand, it is possible to track each of them in the video sequence and to estimate the positions of the body parts according to their velocity.

The name M5AIE is an acronym for each of the used concepts in our approach: Medial Axis transformation, for data filtering; Adapted AGEX, for the body part detection; ASIFT, for the body parts tracking, Aligned Images (RGB-D), and Estimation, also for tracking.

This chapter is organized as follows: Section 3.1 presents some of the related studies. Section 3.2 describes the M5AIE method. The results are presented in Section 3.3. Section 4.5 concludes the chapter.

# 3.1 Related Work

Several Computer Vision studies have solved movement recognition problems using either RGB or depth images. If both RGB and depth values captured by a specific sensor are combined, the possibility of correctly handling pose-recognition issues increases. In this work, we use depth information for background subtraction. The RGB information is used to produce RGB-D images. Because the ASIFT method requires grayscale images, the RGB-D images are converted to grayscale, when the ASIFT tracking method is called by the M5AIE method.

Regarding human pose recognition, Shotton et al. [1] have described an approach that is based on single depth images captured with a Microsoft Kinect sensor. The main contribution of Shotton's work is to treat pose estimation as object recognition, using an intermediate body parts representation to find the joints with high accuracy.

An accurate pose estimator from single-depth images was described by Ye et al. [56]. The authors used a dataset as input to make the pose estimation and presented a pose refinement scheme that can handle pose and body size differences. In their work, they also proposed a pose detection algorithm that is view independent.

Shotton et al. [1] and Ye et al. [56] do not make use of RGB with depth information. A combination of RGB and depth images (RGB-D) has been used for different purposes. Henry et al. [57] presented how the RGB-D images can be used to build 3D maps of indoor environments. Lai et al. [58] also used RGB-D data to recognize instances of a previously trained object. Endres et al. [59] used feature descriptors to provide simultaneously the localization and mapping (SLAM) of RGB-D cameras. Their approach was evaluated using SIFT [53], SURF [60], and ORB [61] descriptors. We use depth data in background subtraction and RGB-D images in body part detection. The pixel intensities computed from the RGB-D values are also used for tracking.

The use of geodesic distances in human body part detection was proposed by Plageman et al. [29] as part of the AGEX points. Ganapathi et al. [51] used AGEX for performing real-time motion capture from depth images. Both studies used depth sensors that were based on a time-of-flight camera to capture the depth data. AGEX was also used by Baak et al. [52] for full body pose reconstruction. Although these studies are very accurate when detecting major body parts (head, hands and feet), the detection of joints in [29, 51, 52] is performed as a naïve estimation of their position with respect to the five main body regions, i.e., head, hands and feet. Such an approach might fail when the imaged person is holding objects such as rackets, balls or other video game gadgets. To avoid estimation problems, we use only the positions of each major body part, which are first mapped to exactly where the parts are, without any estimation. Considering the challenges of our motivational game (the Jecripe game), which is the context of our technique, this study shows that the five main body parts are sufficient for pose classification.

AGEX point detections are usually performed while considering the whole imaged body. In contrast to the conventional approach, the M5AIE method estimates the AGEX points from the pixels of a person's discrete medial axis. It also performs tracking by extracting ASIFT features and matching them between frames. Silberman and Fergus [62] used the SIFT algorithm on depth images for indoor scene segmentation. The main goal of Silberman and Fergus was to label objects (bed, bookshelf, floor, sofa, and table) in a scene, while combining the depth and color images to obtain satisfactory results. Another study that uses SIFT and depth images was presented by May et al. [63]. The main goal in [63] is to perform environment mapping.

# 3.2 The M5AIE Method

The M5AIE method aggregates different concepts; some of them were not originally developed for detecting and tracking. The computational flow of the M5AIE algorithm is illustrated in Figure 3.1. After the alignment of the RGB (Figure 3.2a) and depth (Figure 3.2b) images of a given frame, the M5AIE uses the Minimum Background Subtraction algorithm to address most of the unnecessary information in the frame (Chapter 2). In turn, the area of the person facing the sensor (Figure 3.3a) is replaced by the few pixels that define its discrete medial axis (Figure 3.3b) transformation (Subsection 3.2.1). The detection of body part candidates begins by building a graph in which each pixel of the medial axis is seen as a vertex that is connected to its neighbors by weighted edges; each weight is given by the Euclidean distance between a pair of pixels (Subsection 3.2.2). Using this graph, body part candidates are detected through AGEX point detections (red points in Figure 3.3, Subsection 3.2.3). A labeling step is performed to relate the AGEX points to their respective body parts (Subsection 3.2.4). When the labeling fails, the information computed in the previous frame is used in combination with the ASIFT method for tracking the body parts into the current frame (Subsection 3.2.5).

### 3.2.1 Discrete Medial Axis Through Distance Transformation

The 2D medial axis transform constitutes finding the centers of the maximum disks that can fit inside of an object [64]. A disk is maximal if it is not contained by any other such disk. The set of all centers is called the *medial axis*. When working with digital images, the discrete medial axis of a shape can be computed from the ridge of the discrete distance transformation [65]. Because the discrete medial axis of a discrete object is a connected structure that is composed of a small number of pixels inside that object, the M5AIE method uses such a structure to reduce the number of pixels that are to be considered as vertices in the graph computation (Section 3.2.2).

The M5AIE has performed discrete medial axis extraction by computing the discrete



Figure 3.1: Flowchart of the proposed M5AIE approach applied to RGB-D images to identify the pose of the imaged subject. See Section 3.2 for details. The question mark after the AGEX Point Detection stage verifies whether the Labeling stage of the algorithm can be performed.

distance transform of the binary image that results from the Minimum Background Subtraction (Chapter 2). In turn, we have applied mean-C adaptive local thresholding [65] to identify a superset of the pixels that represent the ridge of the distance transform. From the superset, one could extract the ridge pixels. However, in practice, the exact ridge pixels are not necessary in subsequent steps of our algorithm because the cardinality of the superset is already much smaller than the cardinality of the original set of pixels that represent the user's body. A segmented image and the result of the discrete medial axis (superset) extraction through a distance transformation are presented by Figure 3.3. The following code illustrates a simplified example of how the Medial Axis can be implemented using OpenCV.

def MedialAxis( img ):

grayImg = convertToGrayScale(img)



(a) RGB image

(b) False-color depth information

Figure 3.2: A pair of images related to the same frame of a sequence: (a) RGB information is used to color the foreground pixels and in the tracking stages of our algorithm. (b) Depth information (displayed with false-color) is used to distinguish the background and foreground pixels.

```
cv.Threshold(grayImg, grayImg, 0, 255, cv.CV_THRESH_BINARY)
cv.Erode(grayImg,grayImg, None, 2) #3
imgToTransform = cv.CreateImage( cv.GetSize(grayImg),
cv.IPL_DEPTH_32F, 1);
imgToScale = cv.CreateImage( cv.GetSize(grayImg),
cv.IPL_DEPTH_8U, 1);
cv.DistTransform(grayImg, imgToTransform,
distance_type=cv.CV_DIST_L2, mask_size=cv.CV_DIST_MASK_PRECISE,
mask=0.0, labels=None)
cv.ConvertScale( imgToTransform, imgToScale, 1.0, 0);
cloneScale = cv.CloneImage (imgToScale);
cv.AdaptiveThreshold (imgToScale, cloneScale, 255,
cv.CV_ADAPTIVE_THRESH_MEAN_C, cv.CV_THRESH_BINARY, 9, 0);
cv.Dilate(cloneScale, cloneScale, None, 1)
cv.Erode(cloneScale, cloneScale, None, 2) #3
return cloneScale
```

## 3.2.2 Graph Construction Based on RGB-D Images

Image pixel coordinates were used to build a graph in linear time, as implemented in Schwarz et al. [66]. In such a case, two vertices are considered to be neighbors if the corresponding pixels are separated by a maximum distance threshold  $\delta$ . The graph is represented as  $G_t = (V_t, E_t)$ , where  $V_t$  are the vertices that are related to pixels in the image plane, and  $E_t$  are the edges that connect the vertices. We follow Plagemann et al.'s strategy [29] to connect two vertices and Schwarz et al.'s scheme to weight the edges with the Euclidean distance of the imaged surface points related to the vertices. Formally, Schwarz et al. define the edges as:

$$E_t = \{(x_{ij}, y_{kl}) \in V_t \times V_t \mid ||x_{ij} - x_{kl}||_2 < \delta \land ||(i, j)^T - (k, l)^T||_{\infty} \le 1\}, \quad (3.1)$$

where  $\|\cdot\|_2$  is the Euclidean distance,  $\|\cdot\|_{\infty}$  is the maximum norm, and  $(i, j)^T$  and  $(k, l)^T$  are the 2D coordinates of the points  $x_{ij}$  and  $x_{kl}$  in the depth image. As a consequence of computing the medial axis, the original body can be represented by patches of unconnected pixels. To solve this problem, we used a  $\delta$  value that connects the disconnected parts. The value of  $\delta$  was obtained from experiments in which the distance between the unconnected pixels was measured.

#### 3.2.3 Accumulative Geodesic Extrema Points

Figure 3.3 illustrates a segmented RGB-D image (Figure 3.3a), which is used as input to a discrete medial axis transform. Figure 3.3b illustrates the image from the preprocessing stage that is used to generate the graph. The Accumulative Geodesic Extrema Points (AGEX) are selected while considering the distances of the points according to the edges that connect the vertices in the graph  $G_t$  [29]. This method maximizes the distances of the points using the Dijkstra algorithm [50]. To accomplish this goal, the first AGEX point ( $AGEX_1$ ) is chosen to be the closest point to the centroid ( $c^t$ ) of the human body. The shortest distance between  $c^t$  and all of the other vertices that belong to graph  $G_t$  are calculated with Dijkstra's algorithm, and the vertex with the longest distance among all of the shortest distances is selected as  $AGEX_2$ .

Once the second AGEX point is selected, a zero cost edge between  $AGEX_1$  and  $AGEX_2$  is added to graph  $G_t$ . The aim of adding this edge is to not allow the selection of the same point in a subsequent call of the Dijkstra algorithm. The steps of finding the vertex that has the longest distance in all of the shortest distances that are calculated and

adding a zero edge between the two points are repeated considering  $AGEX_2$  instead of  $AGEX_1$ , and so on until  $AGEX_6$  can be found. The red points in Figure 3.3 correspond to the AGEX points of the imaged subject.

Schwarz et al. [66] approximate the geodesic distances of  $AGEX_{k-1}$  and  $AGEX_k$  by

$$d_G = \sum_{e \in SP(x,y)} w(e), \tag{3.2}$$

where SP(x, y) contains all of the edges along the shortest path between the vertices x and y.

#### 3.2.4 Body Part Labeling

The initialization step for body part labeling comprises a person facing the camera for a few seconds and taking a snapshot on a T-pose (the T-pose can be seen in Figures 3.2 and 3.3). The first six AGEX points correspond to the centroid, head, hands and feet, not necessarily selected in that order. They are labeled according to the position relative to the centroid  $(AGEX_1)$ . Until this stage of the process, the hands, feet and head have not been labeled.

Because  $AGEX_1$  is the centroid  $(c^t)$ , the lower and upper parts of the body can be defined and labeled (from  $AGEX_2$  to  $AGEX_6$ ) according to their coordinate values. Assuming the T-pose, the point that has the highest upper value compared with the centroid is considered to be the head. The two points below the centroid are the right and left feet. Finally, the other two points are the right and left hands. These labeled points are considered in the initialization step, and they are detected at the beginning of the image sequence. As long as this configuration remains unchanged, the AGEX method is used to detect and label each of the body parts. However, when the described configuration changes, then the M5AIE starts to use the ASIFT method, using time sequence information that is based on point estimations to track labels from one frame to another, as described in the next subsection.

#### 3.2.5 ASIFT-Based Tracking of AGEX Points

The ASIFT algorithm was proposed by Morel and Yu [54] for affine-invariant imagefeature extraction. The ASIFT method expects grayscale images as input. The technique transforms the input image by applying tilts and rotations for a small number of latitude



(a) Segmented RGB-D image

(b) Input image for graph generation

Figure 3.3: A discrete medial axis transformation (b) is applied in the segmented RGB-D image (a) to reduce the number of pixels to be considered during the AGEX-graph construction. The red pixels in (a) and (b) are the AGEX points.

angles. Those transformations make ASIFT features affine invariant. Each transformed image is submitted to feature extraction using the SIFT algorithm.

The ASIFT simulates many angles of an object in a given image. The simulation is described as the Affine Camera Model and Tilts. Morel and Yu describe that digital image acquisition of a flat object can be described as:

$$u = S_1 G_1 A \tau u_0 \tag{3.3}$$

According to Morel and Yu, u is a digital image and  $u_0$  is an infinite resolution frontal view of the flat object.  $\tau$  is a plane translation and A is a planar projective map due to the camera motion.  $G_1$  is a Gaussian convolution modeling the optical blur, and  $S_1$  is the standard sampling operator on a regular grid. A major difficulty of the recognition problem is that the Gaussian convolution  $G_1$ , which becomes a broad convolution kernel when the image is zoomed out, does not commute with the planar projective map A.

The ASIFT method simplifies the cited model, by reducing A to an affine map. The Affine Camera Model makes the reduction of A which considers the deformation of objects when a real camera moves its position to capture an image of the same object, however, in a different point of view. *Transition tilts* is a concept that illustrates transition tilts between a frontal view and a *slanted* view of the same object in different images. These tilts are also considered to extract features and match points between two images. Then, the SIFT method [53] is used. See [54] for more details about the ASIFT method.

In the tracking strategy, ASIFT is used to identify the features in the frame t that are related to the AGEX points identified in frame t - 1. However, ASIFT cannot be used directly in tracking due to some practical issues: (i) in the case of background segmented images, ASIFT detects too many features in the border of the foreground region; (ii) there is not necessarily a matching feature for every pixel from one image to another; (iii) the time execution increases as the input images become larger; and (iv) ASIFT can match two features whose positions are far away from an expected conservative maximum distance. These problems are addressed using the following heuristics.

#### 3.2.5.1 Diffusion in the background of sub-images

With the background pixels colored with black and the RGB color of the body pixels converted to a grayscale, the ASIFT method usually detects features only at the frontier between the foreground and the background regions. To solve this issue, the background pixels of the sub-images are filled with the diffusion of the RGB values that are computed according to their colored neighbor pixels. The diffusion process makes the sub-images have smoother transitions in their intensities among the foreground and background pixels. As a result, the contrast inside the portions of the image that are related to the person's body become more significant, which improves the detection of ASIFT features inside the foreground region. Examples of background-diffused sub-images are shown in Figure 3.6. These examples were computed based on the sub-images in Figure 3.5. The backgrounddiffused images are the input for the ASIFT, after they are converted to grayscale. The diffusion method is performed as follows:

- Step 1: Create a list of background pixels and keep it sorted in descending order with regard to the number of foreground (black) 8-connected neighbor pixels of each entry.
- Step 2: Replace the RGB value of the first pixel in the list by the mean RGB values of its neighboring foreground pixels. Remove such a pixel from the list, treat it as a foreground pixel and update the order of the remaining pixels.
- Step 3: Go to the first step or stop when there are no more background pixels to be processed.

#### 3.2.5.2 Searching in a region instead of searching for coordinates only

This heuristic is related to the problem that there is not necessarily a matching feature for every pixel from one image to another. As a consequence of this assumption, a body part position can be lost if only its coordinates are considered. This problem was handled in the following way: if there is no body part matching feature from the sub-image at t-1 with the sub-image at t, then the method searches for the point P, which is the nearest body feature in t-1 that has a match in t. The M5AIE method filters the matching result, considering P to be the body part and its matching feature in t as the final result. As shown in Figure 3.1, the ASIFT method is composed by two stages: the ASIFT features extraction and the Features Matching. We filter the Features Matching results to have only one matching feature for each pair of images. Figure 3.4 illustrates all the matching features (Figure 3.4 (a)) and the filtered result (Figure 3.4 (b)). The white lines in Figure 3.4 (a) are the matching features and the only white line in Figure 3.4 (b) is the filtered result.



(a) All matching features

Figure 3.4: Filtering the matching result.

Considering the person's movement, the body part in frame t-1 can be located anywhere in a region of the frame t. The region is delimited according to a distance from the point in t-1 and the frame at t. This adaptation is not in the ASIFT method, but it is in the matching point output. The reference implementation provided by Morel and Yu [54] returns all of the matching ASIFT features from an image in t-1 and t. The specialized matching scheme, on the other hand, returns only a single feature in a region in t-1 that is related to a feature in t.

#### 3.2.5.3Use of tiny images instead of complete frames

To avoid the heavy computational load of ASIFT applied to the whole image, the M5AIE method applies ASIFT on five tiny images that contain the body parts in frame t-1and the sub-images of the regions in which the same body parts can possibly be found in

sult

frame t. It is important to note that the location and labeling of the body parts in frame t-1 is always known. In the case of the first frame, the T-pose will guarantee the success of the labeling process. In subsequent frames, the body parts will be found by labeling or tracking processes that are performed in the functions of frame t-2. Figure 3.5 illustrates the five sub-images at frame t-1. We assume that each body part does not move too much from one frame to the next frame. Each of the tiny images has a different body part in it, which allows the matching features provided by ASIFT to be in approximately the same region from one image to another.

#### 3.2.5.4 Body-parts position estimation

To assert the consistency of the matching of ASIFT features in the sub-images of consecutive frames, the M5AIE estimates the expected location of the feature in frame t using the uniform linear motion equation considering its location in frames t - 1 and t - 2. The estimation is made using the following equation:

$$s_t = s_{t-1} + v t, (3.4)$$

where  $s_{t-1}$  is a coordinate value (x or y) of the feature in the previous frame, v is the velocity value calculated from (3.5), and t is a constant related to time. The velocity value is computed as:

$$v = \frac{\Delta S}{\Delta T},\tag{3.5}$$

where  $\Delta T$  is constant for this case.

The uniform linear motion displacement of each coordinate is computed using:

$$\Delta S = s_{t-2} - s_{t-1}, \tag{3.6}$$

In the M5AIE method, the acquisition of the color and depth images is performed by the same apparatus (a Kinect), and the RGB-D image alignment is performed by Kinect's SDK. However, because of the asynchronous nature of the image sensors, the final aligned RGB-D image can be ill-formed. As a result, background color pixels can be incorrectly mapped to foreground regions. Figure 3.7 illustrates an extreme case in which many depth pixels of the human body were painted with color information from the background.

To make the proposed matching procedure suitable for tracking, four major situations were identified to be handled: (i) the matched ASIFT feature and the point estimated with equation (3.4) correspond to well-mapped background pixels; (ii) the matched ASIFT



Figure 3.5: Sub-images of the detected five main body parts.



(a) head (b) left hand (c) right hand (d) left foot (e) right foot



feature resides in the well-mapped background while the estimated point is part of the user's body; (iii) the matched ASIFT feature belongs to the human body, and the estimated point is part of the background; and (iv) both the matched ASIFT feature and the estimated point correspond to the actual body.

In the first case, the method searches the body part pixel that is closest to the ASIFT feature found. The second case is when the resulting ASIFT feature is part of the background and the estimated point is part of the body, which generates two sub-cases: (a) if the distance between the two points is smaller than a threshold, then the estimated point will be the final result; and (b) if the distance between the two points is larger than a threshold, then the nearest point from the ASIFT feature that belongs to the human body will be the result.

The third case occurs when the ASIFT feature belongs to the human body and the estimated point belongs to the background. In this case, there are two sub-cases, which are similar to the previous case: (i) if the distance between the two points is smaller than the threshold, then the ASIFT matching feature will be the final result; and (ii) if the distance between the two points is larger than the threshold, then the nearest point from the ASIFT feature that belongs to the human body will be the result. Finally, the fourth case is when the ASIFT matching feature and the estimated point belong to the human body and, again, there are two sub-cases: (i) if the distance between the two points is smaller than the threshold, then the ASIFT feature is the final result; and (ii) if the distance between the two points is larger than the threshold, then a point whose



Figure 3.7: Kinect performs asynchronous acquisition of RGB and depth images. As a result, the quality of the RGB-D alignment procedure performed by Kinect's API can be affected by rapid movements of the user, which leads to inconsistent RGB-D image formation.

coordinates are the average of the two points is generated, and this middle point will be the final result.

The four defined heuristics are necessary because the alignment of the RGB and the depth information could consider images that are acquired at different times. This alignment is provided by the apparatus. If there was no interval for aligning RGB and depth information, the presented heuristics would be unnecessary.

# 3.3 Experiments and Results

The described approach was implemented in Python and was evaluated on real image sequences. The ASIFT algorithm was implemented in C++. We used the reference implementation provided by Yu and Morel at [55]. We used OpenCV to perform the distance transform, adaptive thresholding and other basic image processing procedures. The image sequences were collected using a Kinect sensor, which provides both depth and color images with a  $640 \times 480$  pixel resolution. The resolution of the tiny images was set to  $80 \times 80$ . The goal of this experimental evaluation is to demonstrate the following:

• The modified AGEX can be used for body part detection and labeling in all of the frames of the sequence that have the expected AGEX point configuration described in Section 3.2;



Figure 3.8: Illustration of T-pose and dancing human poses in a game developed by our research group that inspired our experiments.



(a) Play guitar (b) Play flute

Figure 3.9: Illustration of *Play guitar* and *Play drums* human poses.

• The ASIFT algorithm can be used for tracking goals.

We previously collected 14 sequences with human poses that were inspired in the Jecripe game [17, 19]. The poses are: *T-pose, dancing* (left hand on hip and right hand on head), *playing guitar, playing flute* and *playing drums* (see Figure 3.8, Figure 3.9 and Figure 3.10). Two other movements, which were not related to the game, were also included: *punching* and *kicking*.

In all of the sequences, the person moved from the initial T-pose to one of the other four poses. Table 3.1 illustrates the results of detection and tracking. The first three sequences were made for the *dancing* pose; they have 140, 116 and 140 frames, respectively. The *dancing* movement had the best results for the detecting and tracking tasks; every body part was tracked successfully until the end of the sequence.



Figure 3.10: Illustration of the *Play drums* human pose.

The *playing guitar* sequences were tested in the next two sequences of the table (sequences 4 and 5). Both of them had 140 frames. In sequence 4, one of the body parts was lost during the tracking. This loss occurred when one body part was detected in the background in one image and the matching point was lost. As was mentioned in Subsection 3.2.5, this type of situation is possible because color and depth information could be acquired at different times by the apparatus. Sometimes, this interval causes an error in the coloring of the foreground pixels with background color information. In sequence 5, no problem was observed while tracking body parts until the moment that a self-occlusion occurred. The self-occlusion occurred when one of the hands was placed in front of the torso, and all of the matching points were detected at the torso instead of at the moving hand.

The sequences that range from 6 to 9 in Table 3.1, in which we tested *playing drums* and *playing flute*, had exactly the same problem as sequence 5. All of the body parts were correctly tracked until a self-occlusion occurred (a hand in front of the torso). Sequence 10 had the same problem as sequence 4, in which there was a problem in tracking one of the body parts, which was caused by the error of coloring foreground pixels with background color information. As seen, the *dancing movement* (sequences 1, 2 and 3) did not have any type of self-occlusion and, consequently, had the best results on the tracking.

Other types of tests were made with *punching* and *kicking* sequences, where the person moved much faster from the original T-pose to the other tested positions. The velocity

Sequence	Movement	Number	Tracking		
Number		of Images	Until The End		
Sequence 1	dancing	140	yes		
Sequence 2	dancing	116	yes		
Sequence 3	dancing	100	yes		
Sequence 4	playing guitar	140	no		
Sequence 5	playing guitar	140	$yes^*$		
Sequence 6	playing drums	190	$yes^*$		
Sequence 7	playing drums	130	$yes^*$		
Sequence 8	playing drums	130	$yes^*$		
Sequence 9	playing flute	140	$yes^*$		
Sequence 10	playing flute	190	no		
Sequence 11	punching	84	no		
Sequence 12	punching	81	$yes^{**}$		
Sequence 13	kicking	53	no		
Sequence 14	kicking	66	yes		
*Tracked until the end of the sequence; however there was a problem					

Table 3.1: Image sequence evaluation.

\*\*Problem caused by movement velocity.

in the presence of self-occlusion.

of the movements influenced the number of frames, which was less than 100. Sequences 11 and 12 tested the *punching* pose. Sequence 11 had the same problem as sequences 4 and 10, which was similar to Figure 3.7. Sequence 12 had a problem with tracking the punching hand, and the tracked points moved from the hand to the elbow. This problem might be caused by the velocity of the movement, which causes larger changes from one image to the following image and could lead to this problem.

Other types of tests were made with *punching* and *kicking* sequences, where the person moved much faster from the original T-pose to the other tested positions. The velocity of the movements influenced the number of frames, which was less than 100. Sequences 11 and 12 tested the *punching* pose. Sequence 11 had the same problem as sequences 4 and 10, which was similar to Figure 3.7. Sequence 12 had a problem with tracking the punching hand, and the tracked points moved from the hand to the elbow. This problem may be caused due the velocity of the movement, that caused larger changes from one image to the following image, leading to this problem.

The kicking movements are represented by sequences 13 and 14. Sequence 13 had exactly the same problem as sequence 11. However, the body parts were correctly tracked in sequence 14. After we had identified that the M5AIE method had problems with self-occlusions, we made more experiments with no self-occlusions at all. The mentioned experiments are presented in Section 4.4.

# 3.4 Discussion

This chapter presented the M5AIE method for detecting and tracking the five main parts of the human body (head, hands and feet) in sequences of RGB-D images. The method generates tuples with each body part position. The proposed approach combines an effective background subtraction method, the discrete medial axis transformation in the construction of simpler graphs to be used in the detection of AGEX points, heuristics for labeling, and ASIFT-based tracking of labeled structures.

We investigated how to adapt the ASIFT method for tracking objectives and showed that it is possible to achieve good results with the tested movements. The key insights of this investigation are the following:

- ASIFT and estimation can be combined and used for tracking the objectives of movements without self-occlusions;
- It is necessary to make improvements in the tracking method to use it with movements in which there is a body part occlusion; and
- The RGB-D aligning procedure caused the loss of one of the body parts during tracking. This type of problem might not occur in the future through the synchronous acquisition of color and depth information.

The proposed M5AIE algorithm was implemented in proof-of-concept programs. We did not consider the computational load of this specific implementation to be a fundamental requirement because the main goal in the experiments is to assert the possibility of using a hybrid technique for body part detection, tracking and pose classification. The bottlenecks for real-time results are the Medial Axis and the ASIFT algorithm. Pinto and Freitas [67] showed real-time results for the Medial Axis Transform. In 2013, Chiu et al. [68] presented a fast SIFT design for real-time visual feature extraction. Because ASIFT depends on the SIFT results, we believe that the M5AIE can be efficiently implemented and used as part of real-time tracking solutions that are applied to games.

The M5AIE method is limited to an indoor environment, static background, static position and orientation of the sensor and to single-user segmentation. Experiments showed that, to be correctly tracked, the sequences must not have body part occlusions. The next chapter introduces a study that has different classification algorithms: the C4.5 Gain Ratio Decision Tree, Naïve Bayes Classifier and KNN Classifier.

# Chapter 4

# Analysis of the Classification Algorithms Using M5AIE-Extracted Human Poses

This chapter describes the context of human pose recognition research and also presents an analysis of different classifiers in pose prediction. As was mentioned previously, depth sensor devices provide better options for the development of interaction paradigms such as a Natural User Interface (NUI). Shotton et al. presented how the device works and its applicability in digital games. Recently, Shotton et al. [69] described new approaches to human pose estimation and used tree structures.

Tang et al. [15] describe a study that uses an interactive dancing game with real-time recognition of continuous dance moves from 3D human motion capture. The authors describe positive feedback from the users' experiences. Tang et al. also describe the development of their own motion recognition algorithm for dance moves.

Rogez et al. [70] address human pose recognition as a classification problem. In their work, the authors describe a pose detection algorithm that is based on tree structures. Shotton et al. [1, 69], Tang et al. [15], Rogez et al. [70] and many other related studies mention the used classification algorithm. However, the authors do not explain why they chose each of their options, and they do not compare their algorithms with traditional classification algorithms.

To the best of our knowledge, there is no work in the literature that performs a comparison among classification algorithms in human pose recognition in the context of games. Huang et al. [30] compared Naïve Bayes, Decision Trees, and Support Vector Machines (SVMs), to evaluate which is the best measure to use when classification algorithms are compared. In [30], the authors used datasets that had only two classes (binary datasets) and compare the use of two different measures: accuracy and Area Under the Curve (AUC). In this thesis, the accuracy measure was used in the experiments (Section 3.3) to evaluate the selected classification algorithms. Amor et al. [31] used intrusion detection system datasets to compare Naïve Bayes and Decision Trees. Amor et al. described how a Naïve Bayes classification algorithm can provide competitive results. The authors of the two studies [30, 31] did not consider datasets in human pose classification in their experiments.

This chapter describes a detailed analysis, using the M5AIE method (Chapter 3) with different algorithms: C4.5 Gain Ratio Decision Tree [32], Naïve Bayes Classifier [33] and K-Nearest Neighbor (KNN) Classifier [34]. The results could help researchers to choose among the selected algorithms to use in human pose classification in the context of digital games. The experiments and results were published in [71].

This chapter is organized as follows: Section 4.1 presents some of the related studies. Section 4.2 describes the selected classification algorithms for this study. Section 4.3 describes how we adress the position of each of the body parts. The experiments and results are presented in Section 4.4. Section 4.5 concludes the work with a discussion and future directions for the research.

# 4.1 Related Work

Human action recognition is a related area of Computer Vision that addresses motion in videos. Mota et al. [72] introduced a video motion indexing scheme that was based on modeling optical flow. In their work, the authors proposed a global motion tensor descriptor for video sequences, and optical flow was described with a polynomial representation. In contrast to Mota et al.'s work, this work is concerned with the detection and tracking of body parts in RGB-D image sequences and with pose identification in single frames of the sequence.

Movement recognition in games has many different approaches. Many of the movement recognition technologies apply video or frames in real time as an input of a recognition system. A low-cost approach is described by Wang and Popović [7]; these authors propose an easy-to-use color glove. In that case, even the color gloves are inexpensive; it was necessary to use instrumented gloves in the construction of the database. The database is used to recognize letters in sign language. Nevertheless, Wang and Popović's work performs pose virtual reconstruction, which is unnecessary in the context of this work. Wang and Popović applied their work on sign language finger spelling. Figueiredo et al. [73] also make use of gloves, and their study made use of a yellow gloves, which were applied to interaction only, without pose or movement recognition. Almeida [74], as Wang and Popović, worked with sign language and they made use of a depth sensor to capture the movements. Wang and Popović and Almeida use classifiers but they did not detail the used classifiers. Figueiredo et al.'s work do not use classifier because their focus is on interaction.

Bourdev and Malik [8] and Bourdev et al. [9] apply a two dimensional image as input and estimates the three dimension coordinates of the selected keypoints. Moreover, it is presented the definition of Poselets as a particular part of the human pose under a given viewpoint. This approach is defined with a set of examples that are close in 3D configuration space. The main contribution of [8, 9] is the notion of a part of a pose, a "poselet", and an algorithm for selecting good poselets. Each poselet provides examples for training a linear SVM classifier. However, none of these studies detailed why they chose the selected classifier.

Bleiweiss et al. [75] describe a real-time framework that blends the player's actual movements, which are tracked using a depth sensor, with pre-defined animation sequences. According to the authors, their depth-based framework enables an enhanced visual feedback mechanism by understanding the player's full body motion and seamlessly blending it with pre-animated content. Bleiweiss et al. made use of a full body tracking algorithm proposed by Ganapathi et al. [51]. In [75], the authors presented a system whose skeleton mimics the player's movements.

The work of Schönauer et al. [13] describes a full body input for rehabilitation. The contribution of their work includes the implementation of serious game targeting rehabilitation of patients with chronic pain of the lower back and neck. Schönauer et al. argue that the tracker used for their motion capture system, which is an io-tracker [14], is marker based and uses an infrared optical motion tracking system. In [13], Schönauer et al. used a classification tree for posture estimation and did not compare the classifier with any other algorithm.

An interactive dancing game was presented by Tang et al. [15]. The motion recognition algorithm was developed based on a finite state machine representation and a Block Matching Cost. The game uses motion templates, and the Block Matching Cost calculates the cost for matching a frame of the player's move with a frame of a template move.

Tian et al. [76] presented a semantic feature to represent characteristics of different human motion classes. Cimen et al. [77] describe how they classify human motion with descriptors that are related to emotion classification. Sun et al. [78] show how conditional regression forests can be used in human pose estimation.

In this chapter, we do not consider action recognition or virtual reconstruction, and we also do not consider their interaction. We focus on a comparison of different algorithms for human pose classification in the game context. The Kinect system was chosen to collect the image sequences of a single player who makes game movements. There are available technologies to integrate Kinect with the Unity3D engine, which was used in the development of the Jecripe game [17]. This game inspired the selected movements in the experiments.

# 4.2 Pose Classification

Classification techniques were used in this study to identify categorical labels such as "Pose A" and "Pose B" for the current subject, according to the position of each of the body parts that are detected or tracked in a given image of the sequence.

The human pose classification was performed using three different algorithms: the C4.5 Gain Ratio Decision Tree [32], the Naïve Bayes classifier [33] and the K-Nearest Neighbor (KNN) classifier [34]. These algorithms were selected due to their low computational load and simplicity, which makes them suitable for real-time applications.

# 4.2.1 C4.5 Gain Ratio Decision Tree

Decision trees follow the "divide and conquer" approach. According to Amor et al. [31], the decision tree structure is composed of the following elements: (i) decision node, which specifies a test attribute that is responsible for the comparison of an attribute value with a constant; (ii) an edge that is one of the possible attribute values (the test attribute is placed here); and (iii) leaf nodes that give the classification to which the object belongs.

Decision trees have two stages: building the tree and the classification itself. Building the tree constitutes selecting the test value for each decision node and the classification labels of each leaf. Decision trees are built based on a given training set. The classification stage is made starting from the root of the decision tree. To go down the tree, tests are made to achieve one of the leaf nodes.

Many algorithms were developed for the construction of decision trees for the classification task. In the experiments, the considered decision tree was the C4.5 Gain Ratio Decision Tree algorithm developed by Quinlan [32]. The C4.5 Gain Ratio Decision Tree selects the attribute that has the largest number of possible values to be assumed as the current node in the tree construction. This criterion is an extension to information gain. The gain ratio is defined as:

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$
(4.1)

However, the Gain Ratio applies a normalization to information gain "using a split information", defined as:.

$$SplitInfo_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right),\tag{4.2}$$

where D is a set of tuples and |D| is the number of tuples in D. Additionally,  $|D_j|$  is the number of times that tuple  $D_j$  appears in D. In equation 4.1, we have Gain(A) which is the information gain. Gain(A) defines how much is gained if the attribute A is used as the branching node in the decision tree structure. The information gain is defined as:

$$Gain(A) = Info(D) - Info_A(D) , \qquad (4.3)$$

where  $Info_A$  is the information value needed to have a testing tuple arriving at an exact classification using a value of attribute A. A classification value is the value of the attribute which defines the class of each tuple. Info(D) is the information gain value computed for each value of the attribute class. The information gain is defined as:

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j).$$

$$(4.4)$$

Then, the selection of an attribute to be the current node in the tree is defined on the maximum gain ratio value as the splitting attribute. For more details about decision tree structures, see [79].

### 4.2.2 Naïve Bayes Classifier

Naïve Bayes makes a strong independence relation in which the features are independent in the context of a session class [31, 33]. The Naïve Bayesian classifier works, basically, as follows: (i) the training set is composed of tuples, and these tuples are attribute values in a predefined order; (ii) for each class, a conditional probability can be calculated based on the used training set; and (iii) the likelihood of a testing tuple is defined based on the calculated conditional probabilities of each class. The following working flow of the Naïve Bayes Classifier is described in [79].

- Step 1: Let *D* be a training set of tuples and each tuple with its class label. Each tuple is represented by a predefined sequence of *n* attribute values  $(A_1, A_2, ..., A_n)$ , such as  $X = (x_1, x_2, ..., x_n)$ .
- Step 2: Consider *m* classes  $(C_1, C_2, ..., C_m)$ . For each tuple, *X*, the classifier will predict each class of tuple  $X_i$  belongs to the class having the highest probability, conditioned on *X*.
- Step 3: As P(X) is the same for all classes, only  $P(X|C_i)P(C_i)$  need be maximized. If the class a priori probabilities,  $P(C_i)$ , are not known, then it is assumed that the classes are equally likely, and we maximize  $P(X|C_i)$ . Otherwise we maximize  $P(X|C_i)P(C_i)$ .
- Step 4: The values of the attributes are conditionally independent of one to another, given the class label of the sample. Then an evaluation is done considering if the attribute is categorical or continuous valued.
- Step 5: The classifier predicts that the class label of X is  $C_i$  if it is the class that maximizes  $P(X|C_i)P(C_i)$ .

The main difference between the two mentioned classifiers is that while C4.5 is a decision tree classifier, the Naïve Bayes is based on the Bayes rule of conditional probabilities. In decision trees, the attributes are tested, and the final classifications are at the leaves. In this approach, the attributes have a high level of dependency on each other. However, the Naïve Bayes classifier evaluates each attribute individually, considering them to be independent.

## 4.2.3 K-Nearest Neighbor Classifier

In 1967, Cover and Hart [34] introduced the k-Nearest Neighbor as a pattern classifier. A training set is built by tuples and a tuple X, whose class is unknown, is then tested. The tuple X is compared with each of the training tuples. The k closest tuples to Xare considered to predict its class. "Closeness" is considered a distance metric, and it can be calculated, for example, with the Manhattan, Chebyshev or Euclidean distance. The three distances were selected because they use the vertical and horizontal coordinates system, which is used by the M5AIE method to generate tuples. The unknown class of X is assigned to the most common class among its k nearest neighbors.

# 4.3 Bounding Box and Grid

In this work, the algorithms receive as input the labels and the locations of the body parts according to an  $N \times N$  grid that is defined inside the bounding box that contains the whole body of the imaged subject. Figure 4.1 shows the grid squares with N = 8. A bounding box was used to identify the cell number of the body parts. The bounding box provides the relative positions according to the detected human body. This approach makes it possible to identify the cell number of the body parts, independently of their occupied positions in the whole segmented image.



Figure 4.1: A bounding box limits the human body, and it is divided into  $N \times N$  cells. In this example, N = 8.

The considered classification algorithms (C4.5, Naïve Bayes and KNN) require the execution of a training stage to build a model to be used during the classification of the poses (see Figure 4.2). In our work, the dataset is used both for training and testing. The class of each tuple comprises the cell-coordinates in the grid that body parts assume

at each image in a sequence. The pose classification of each training tuple was made manually in each frame. In the classification procedure, a tuple constitutes a sequence in which the cell position of every individual body part is described in the same order as the order that appears in the attributes definition. The following code illustrates an example of how the attributes are declared and a few training tuples:

Orelation poses

@attribute headx NUMERIC @attribute heady NUMERIC @attribute rightFootx NUMERIC @attribute rightFooty NUMERIC @attribute leftFooty NUMERIC @attribute leftHandx NUMERIC @attribute leftHandy NUMERIC @attribute rightHandx NUMERIC @attribute rightHandy NUMERIC @attribute rightHandy NUMERIC @attribute rightHandy NUMERIC @attribute rightHandy NUMERIC

#### @data

0, 3, 7, 5, 7, 3, 1, 0, 1, 7, tpose 0, 4, 7, 5, 7, 3, 1, 0, 1, 7, tpose 0, 4, 7, 6, 7, 1, 0, 0, 3, 7, dancing 0, 3, 7, 5, 7, 2, 0, 0, 3, 7, dancing 0, 5, 7, 6, 7, 3, 1, 0, 3, 7, guitar 0, 3, 7, 5, 7, 3, 3, 0, 3, 7, drum ...



Figure 4.2: The aim of the training stage is to build a classification model.



Figure 4.3: The classification model is used to predict the class of the tuples that are being tested.

The testing tuples are the input for the classification model (see Figure 4.3). This classification model predicts the class of each testing tuple based on the training data set. In the classification procedure, a tuple constitutes a sequence in which the cell position of every individual body part is described in the same order that appears in the attributes definition.

# 4.4 Experiments and Results

The classification algorithms were evaluated using the data mining tool WEKA 3.6.8 (Waikato Environment for Knowledge Analysis) [80]. To adopt the traditional classifiers C4.5 Gain Ratio Decision Tree, Naïve Bayes and KNN, we used the J48, Naïve Bayes and Ibk implementations that are available in the WEKA tool, respectively.

We used k-fold cross-validation in our test. In this approach, the dataset is randomly partitioned into k subsets. Only one subset is used as validation data for testing the model. The other k - 1 subsets are used for training the classification model. The cross-validation process is repeated k times. Each of the k subsets is used only once for validation. The final result is the average of the results obtained at each round. In our experiments, we used k = 10.

### 4.4.1 Testing Categorical Attributes

We first describe how the first experiments were made as a proof of concept. We tested human pose classification for each frame using the C4.5 Gain Ratio Decision Tree [32] and the Naïve Bayes Classifier [33]. In these first tests, the input information for the classification stage is a set of five categorical attributes, which have the cells' numbers ("Cell 1", "Cell 2", ..., "Cell N") that contain the tracked body parts of the subject. The main goal of using categorical labels for the body parts attributes was to prove that it is

Tabl	ю <b>т.т.</b>	001	i upi	on m	TIT	TOLT	
a	b	с	d	е	f	g	Classified As
100	0	0	0	0	0	0	a = tpose
1	223	0	0	0	0	0	b = dancing
0	0	61	0	0	0	0	c = guitar
0	0	0	0	0	0	0	d = flute
2	6	0	0	135	0	0	e = drum
2	0	2	0	0	11	0	f = kick
3	0	0	0	1	0	41	g = punch

Table 4.1: Confusion Matrix for Naïve Classifier.

possible to use different classification algorithms in the M5AIE method.

The used sequences were exactly the same as shown in Table 3.1. Here, a bounding box that defines the position of the person's full body was made for each of the segmented input images. To organize the dataset, this bounding box is divided into eight columns and eight rows  $(8 \times 8)$ , which results in 64 cells. Each of the detected body parts is in one of the cells. Sequences of cells that stand in a pre-defined order are the dataset tuples.

In these experiments, we used six categorical attributes: *head*, *right foot*, *left foot*, *left hand*, *right hand*, and *pose*. The first five attributes are the body parts, and the values to be assumed are the cell indexes (which range from 0 to 63) where the body parts are located. All of the body part attributes were defined as categorical values. The pose attribute corresponds to the relation and can assume 7 categorical values: *tpose*, *dancing*, *guitar*, *flute*, *drum*, *kick*, and *punch*.

Table 3.1 presents errors in sequences 4, 9, 10, 11, and 13. Because of these errors, we decided to not consider data from these sequences. Sequences 5, 6, 7, 8, 9 and 12 had problems with tracking when there was self-occlusion. However, we considered the frames when the person was already in a pose, and the error did not occur. In sequences 1, 2, 3 and 14, the tracking was made correctly. We also did not make any tuple for *playing flute* movements because of the large number of images that had self-occlusions. With

	a	b	с	d	е	f	g	Classified As
-	99	0	0	0	1	0	0	a = tpose
	0	224	0	0	0	0	0	b = dancing
	0	0	61	0	0	0	0	c = guitar
	0	0	0	0	0	0	0	d = flute
	2	0	0	0	140	0	1	e = drum
	0	0	1	0	0	14	0	f = kick
	0	0	0	0	1	0	44	g = punch

Table 4.2: Confusion Matrix for C4.5 Gain Ratio Decision Tree.

the mentioned restrictions, the dataset for pose classification training and testing was composed of 588 tuples altogether.

The output of the testing procedure is a confusion matrix for each classification algorithm. Tables 4.1 and 4.2 illustrate the confusion matrix for Naïve Bayes and the C4.5 Gain Ratio Decision Tree, respectively. The Naïve Bayes Classifier correctly classified 97.1088% of the instances (571 altogether) and incorrectly classified only 2.8912% of them (17 tuples).

The C4.5 Gain Ratio Decision Tree Classifier correctly classified 98.9796% of the instances (582 tuples), and only 1.0204% of the instances (6 of them) were incorrectly classified. These results show that the dataset that was used produces a very high number of correctly classified instances with both classifiers. As a result, it is possible to conclude that the cell division, when assuming a grid of  $8 \times 8$  cells, is adequate for the poses that were used and that the bounding box that was used for localization purposes was adequate as well.

#### 4.4.2 Testing Numerical Attributes

After we showed convincing results that it is possible to use different classifiers while using the M5AIE method, we then used a set of ten numeric attributes. Each of these attributes is related to a coordinate of the considered body parts. The *pose* attribute is the only attribute that is categorical. The training data begins with the label @data. Each attribute is separated by a comma. The last attribute is the testing class.

The goal of these experiments is to answer the following questions:

- 1. Can the classifiers make correct predictions with different poses and the same class?
- 2. Is there any difference in the results when using different users to build the data set?
- 3. Which is the best value for N in the grid  $N \times N$ ?
- 4. Which of the three considered classification algorithms is the best for human pose prediction in the game context?

As in Section 3.3, we previously collected sequences with human poses that were inspired by the Jecripe game [17, 19]. The poses define the classes, and they can assume the following values: T-pose, dancing, play guitar, and play drums. Nevertheless, in

these experiments, three other movements, which were not related to the game, were also included: punch, kick and kick + punch.

We characterize the classes as the following: The *T*-pose constitutes a person with both arms and hands at the same level as the shoulders. In the *dancing* class, one of the hands is on the head; the other hand is on the hip, and one or both feet are on the ground. As a consequence, we have six combinations of poses for the class *dancing*: (i) left hand on the head and feet on the ground; (ii) left hand on the head and moving left foot; (iii) left hand on the head and moving right foot; (iv) right hand on the head and feet on the ground; (v) right hand on the head and moving right foot; and (vi) right hand on the head and moving left foot. All of the six poses have the same class, which is *dancing*.

In the *playing guitar* class, the user imitates the moves of playing an instrument, shaking the right hand while the left hand stays at the same level as his/her shoulders. The *playing drums* class is when the user shakes his/her hands up and down alternately. There are two possible poses for the *punch* class, both of which have feet on the ground: (I) right hand and (II) left hand. Similar to the *punch*, the *kick* class can be made with: (a) right foot and (b) left foot, with both hands below the centroid. The *kick* + *punch* class can be made in four different poses: (A) kick with left foot and punch with left hand; (B) kick with left foot and punch with right hand; (C) kick with right foot and punch with right hand.

We used three different volunteers in our experiments: A, B and C. For each user, we collected a different number of sequences. Volunteer A is male, 1.76 meters tall, and has dark hair. Table 4.3 shows the collected sequences with Volunteer A. We collected 17 sequences with all of the classes.

Volunteer B is male, 1.90 meters tall and has blond hair. Volunteer B made 14 different sequences in four classes, all of them without self-occlusion. All of the possible poses for each of the four classes were collected. Table 4.4 details each of the collected poses from Volunteer B.

Volunteer C is female, 1.66 meters tall and has dark hair. Similar to Volunteer B, we collected sequences of four different classes with Volunteer C. Additionally, no problem was detected during the collection of the poses, which shows that the M5AIE method works well in sequences that do not have self-occlusions. We collected 13 sequences with Volunteer C because we wanted to test fewer training tuples with the pose kick + punch (A).

Table 4.5: Image sequence evaluation for volumeer A.						
Sequence	Movement	Number	Tracking			
Number		of Images	Until the End			
Sequence A1	dancing (i)	140	yes			
Sequence A2	dancing (i)	116	yes			
Sequence A3	dancing (ii)	100	yes			
Sequence A4	playing guitar	140	$yes^*$			
Sequence A5	playing drums	190	$yes^*$			
Sequence A6	playing drums	130	$yes^*$			
Sequence A7	playing drums	130	$yes^*$			
Sequence A8	punch $(I)$	84	yes			
Sequence A9	punch $(I)$	81	$yes^{**}$			
Sequence A10	kick $(a)$	66	yes			
Sequence A11	dancing (iii)	58	yes			
Sequence A12	dancing (ii)	68	yes			
Sequence A13	kick + punch (A)	57	yes			
Sequence A14	dancing (iv)	104	yes			
Sequence A15	dancing $(v)$	152	yes			
Sequence A16	dancing (vi)	98	yes			
Sequence A17	kick $+$ punch (D)	55	yes			
*Tracked until the end of the sequence, but there was a problem						

Table 4.2. Income as an inclustion for Valuetoen A

in the presence of self-occlusion.

\*\*Problem caused by movement velocity.

We observed that the M5AIE method had problems with poses that had self-occlusions. The problems were detected in the *playing quitar* and *playing drums* poses. This problem detection was crucial for the collection of the other users' sequences; as a result, we avoided collecting these poses. However, we kept the results to make the tuples and test the classification algorithms. In only one sequence, the tracking method had problems that were caused by the movement velocity, but the pose classification was not affected.

The dataset that was used for both the training and testing comprises the gridcoordinates that body parts assume at each frame of a set of image sequences that were produced for this work and the manual classification of the pose in each frame. We varied the number of cells of the grid in each frame, as follows:  $8 \times 8$  (Table 4.6),  $16 \times 16$ (Table 4.7),  $32 \times 32$  (Table 4.8) and  $64 \times 64$  (Table 4.9).

The set of k values for the KNN algorithm is  $\{1, 3, 5, 7, 9, 11\}$ , and different distances were used in our experiments. We combined the set of k values with the Manhattan, Chebyshev and Euclidean distances. For each of the N values of the grids  $N \times N$ , we made a data set that had all of the tuples from the three different users that made the described poses and 2128 tuples.

Table 4.4. Image sequence evaluation for volumeer D.						
Movement	Number	Tracking				
	of Images	Until the End				
dancing (i)	99	yes				
dancing (iv)	84	yes				
dancing (iii)	84	yes				
dancing (ii)	62	yes				
dancing $(v)$	72	yes				
dancing (vi)	79	yes				
punch (I)	65	yes				
punch (II)	75	yes				
kick $(b)$	70	yes				
kick (a)	79	yes				
kick + punch (C)	73	yes				
kick $+$ punch (D)	74	yes				
kick + punch (B)	99	yes				
kick + punch (A)	97	yes				
	dancing (i) dancing (iv) dancing (iii) dancing (iii) dancing (v) dancing (v) dancing (v) punch (I) punch (II) kick (b) kick (a) kick + punch (C) kick + punch (D) kick + punch (A)	MovementNumber of Imagesdancing (i)99dancing (iv)84dancing (iii)62dancing (iii)62dancing (v)72dancing (vi)79punch (I)65punch (II)75kick (b)70kick (a)79kick + punch (C)73kick + punch (B)99kick + punch (A)97				

Table 4.4: Image sequence evaluation for Volunteer B.

Table 4.6, where N = 8, shows that the Naïve Bayes Classifier gave the highest number of incorrectly classified instances (21.22%). For all of the other classifiers, the percentage of instances that were correctly classified were above 93%. The C4.5 Gain Ratio Decision Tree had similar results as the KNN algorithm when  $k \ge 3$ . As the k value increased, the percentage of correctly classified instances decreased. Nevertheless, the Manhattan distance had the best results for every k value. The best of all of the results in Table 4.6 were with K = 1, primarily from using the Manhattan distance, with a 98.24% correct. Most of the errors made by the classifier were from confusing *dancing* with *punch* and *kick* + *punch* classes.

Considering N = 16 (Table 4.7), once more, the Naïve Bayes Classifier gave the highest percentage of incorrectly classified instances, with 28.74%. For all the other classifications, the incorrectly classified instances were less than 8%. The C4.5 Gain Ratio Decision Tree had only similar results with  $k \ge 7$  considering the Manhattan and Euclidean distances. If we consider only the values with the same value k, the Chebyshev distance gave the worst results. On the other hand, the Manhattan distance gave the best results. We could observe that the best results were obtained again with k = 1 and the Manhattan distance. Again, increasing the value of k, the results become worse for all of the used distances. The best percentage of correctness with N = 16 (98.84%) was slightly better than with N = 8 (98.24%), when both used k = 1 and the Manhattan distance. The dancing class was confused with the *playing guitar*, *punch* and *kick* + *punch* classes.

With N = 32, similar to with N = 8 and N = 16, the Naïve Bayes classifier gave the

Sequence	Movement	Number	Tracking
Number		of Images	Until the End
Sequence C1	dancing (i)	48	yes
Sequence C2	dancing (iv)	69	yes
Sequence C3	dancing (iii)	45	yes
Sequence C4	dancing (ii)	54	yes
Sequence C5	dancing $(v)$	54	yes
Sequence C6	dancing (vi)	45	yes
Sequence C7	punch (I)	90	yes
Sequence C8	punch (II)	88	yes
Sequence C9	kick $(b)$	49	yes
Sequence C10	kick (a)	54	yes
Sequence C11	kick + punch (C)	90	yes
Sequence C12	kick + punch (D)	100	yes
Sequence C13	kick + punch (B)	85	yes

Table 4.5: Image sequence evaluation for Volunteer C.

smallest percentage of correctly classified instances (76.17%). All of the other results had more than 95% correctness on instances of classification. If we compare the C4.5 algorithm with KNN (without the Chebyshev distance), we obtain similar results to when  $k \ge 9$ . The best results were with K = 1 but with the Euclidean distance (99.77%), which was followed very closely by the Manhattan distance (99.72%). This result is even better than the best result in Table 4.7. Most of the incorrectly classified instances occurred with instances of *dancing*, *punch* and *kick* + *punch*. Table 4.8 shows the results for N = 32.

Table 4.9 shows the results with N = 64. As was expected, the Naïve Bayes had 22.98% incorrectly classified instances, followed by KNN with k = 11 and the Chebyshev distance, which had 5.45% incorrect. All of the others gave more than 94% correctly classified instances. The C4.5 algorithm had similar results with only KNN when k = 11. Similar to the other best results, in Table 4.9, KNN with k = 1 gave the best results with both distances, Manhattan and Euclidean, with exactly the same value, 99.81%. Because the results are very close to 100% using N = 64, we could observe a relatively high number of errors using Naïve Bayes, which gave errors in the classes *dancing*, *punch* and *kick* + *punch*.

Until this point, we exposed the results, showing each table in an isolated way. However, we can observe additional results by comparing the tables with one another. All of the algorithms had similar results while considering the same algorithm with different Nvalues. In all of the cases, the worst results came from the Naïve Bayes Classifier. The C4.5 had similar results with KNN depending on the k value of each Table. Although the results are very similar from one table to another, we can see that the results of the C4.5

Classification with Grid $8 \times 8$	Correct*	Incorrect**
C4.5 Gain Ratio Decision Tree	97.26%	2.74%
Naïve Bayes	78.78%	21.22%
KNN with K=1 and Manhattan Distance	98.24%	1.76%
KNN with K=1 and Chebyshev Distance	98.07%	1.93%
KNN with K=1 and Euclidean Distance	98.24%	1.76%
KNN with K=3 and Manhattan Distance	97.79%	2.21%
KNN with K=3 and Chebyshev Distance	97.01%	2.99%
KNN with K=3 and Euclidean Distance	97.66%	2.34%
KNN with K=5 and Manhattan Distance	96.89%	3.11%
KNN with $K=5$ and Chebyshev Distance	95.94%	4.06%
KNN with $K=5$ and Euclidean Distance	96.60%	3.40%
KNN with K=7 and Manhattan Distance	96.23%	3.77%
KNN with K=7 and Chebyshev Distance	94.22%	5.78%
KNN with $K=7$ and Euclidean Distance	95.99%	4.01%
KNN with K=9 and Manhattan Distance	95.94%	4.06%
KNN with $K=9$ and Chebyshev Distance	93.32%	6.68%
KNN with $K=9$ and Euclidean Distance	95.86%	4.14%
KNN with K=11 and Manhattan Distance	96.31%	3.69%
KNN with K=11 and Chebyshev Distance	93.20%	6.80%
KNN with K=11 and Euclidean Distance	96.15%	3.85%
*Correctly Classified Instances		·

Table 4.6: Results for N = 8

\*\*Incorrectly Classified Instances

algorithm and KNN become better when N becomes higher. Considering the distances, in general, the Manhattan gave the best results if we compare the same k value in every Table. The Euclidean distance gave very similar results to the Manhattan, and only once the results from the Euclidean distance were better than the Manhattan distance. In all of the KNN experiments, the Chebyshev distance gave a percentage of incorrectly classified instances that was higher than for the other two considered distances.

Returning to the main goals of the experiments, we can conclude that:

- 1. With the exception of the Naïve Bayes, all of the other classifiers had at least 92% of the instances classified correctly. With this result, we consider that the tested classifiers can make correct predictions with different poses of the same class.
- 2. Even using three different users to build our data set, we had a high number of instances correctly classified. Thus, we consider that the usage of different volunteers in our experiments did not affect the results. Moreover, these results showed that the use of the M5AIE method for body part detection and tracking works properly in movements without self-occlusions. Further experiments should be performed

Classification with Grid $16 \times 16$	Correct*	Incorrect**
C4.5 Gain Ratio Decision Tree	97.39%	2.61%
Naïve Bayes	71.26%	28.74%
KNN with K=1 and Manhattan Distance	98.84%	1.16%
KNN with K=1 and Chebyshev Distance	98.31%	1.69%
KNN with K=1 and Euclidean Distance	98.79%	1.21%
KNN with K=3 and Manhattan Distance	98.36%	1.64%
KNN with K=3 and Chebyshev Distance	96.33%	3.67%
KNN with K=3 and Euclidean Distance	97.97%	2.03%
KNN with K=5 and Manhattan Distance	98.02%	1.98%
KNN with K=5 and Chebyshev Distance	95.60%	4.40%
KNN with $K=5$ and Euclidean Distance	97.58%	2.42%
KNN with K=7 and Manhattan Distance	97.39%	2.61%
KNN with $K=7$ and Chebyshev Distance	95.22%	4.78%
KNN with $K=7$ and Euclidean Distance	97.29%	2.71%
KNN with K=9 and Manhattan Distance	97.20%	2.80%
KNN with K=9 and Chebyshev Distance	94.30%	5.70%
KNN with K=9 and Euclidean Distance	96.86%	3.14%
KNN with K=11 and Manhattan Distance	96.47%	3.53%
KNN with K=11 and Chebyshev Distance	92.90%	7.10%
KNN with $K=11$ and Euclidean Distance	96.18%	3.82%
*Correctly Classified Instances	•	·

Table 4.7: Results for N = 16

\*\*Incorrectly Classified Instances

using dozens (or maybe hundreds) of volunteers to check whether the classification models would be affected. If this results are similar with a much larger number of volunteers, then the classification models are good for additional volunteers.

- 3. Because the results were improved with increasing values of N, the best results were given with N = 64. However, if we continue to increase the value of N, the results could be improved until a certain value. However, there is a possibility that, from a certain value on, the results might not become any better or they even might start to become worse. The last assumption is justified because the number of grids can become so large that each pixel could occupy more than one cell in the grid. Again, further experiments should be performed to find the exact value of N for which the results obtain the best percentage of instances that are correctly classified.
- 4. The classification algorithm with the best results was the KNN algorithm with k = 1 while using the Manhattan distance.

As mentioned in 3, we believe that if we continue to increase the value of N, it could improve the results even more until a certain limit value is obtained. From that limit

Classification with Grid $32 \times 32$	Correct*	Incorrect**
C4.5 Gain Ratio Decision Tree	98.59%	1.41%
Naïve Bayes	76.17%	23.83%
KNN with K=1 and Manhattan Distance	99.72%	0.28%
KNN with K=1 and Chebyshev Distance	99.58%	0.42%
KNN with K=1 and Euclidean Distance	99.77%	0.24%
KNN with K=3 and Manhattan Distance	99.39%	0.61%
KNN with K=3 and Chebyshev Distance	98.45%	1.55%
KNN with K=3 and Euclidean Distance	99.34%	0.66%
KNN with K=5 and Manhattan Distance	99.34%	0.66%
KNN with K=5 and Chebyshev Distance	97.93%	2.07%
KNN with $K=5$ and Euclidean Distance	98.83%	1.17%
KNN with K=7 and Manhattan Distance	99.15%	0.85%
KNN with $K=7$ and Chebyshev Distance	97.32%	2.68%
KNN with $K=7$ and Euclidean Distance	98.64%	1.36%
KNN with K=9 and Manhattan Distance	98.73%	1.27%
KNN with K=9 and Chebyshev Distance	95.82%	4.18%
KNN with K=9 and Euclidean Distance	98.03%	1.97%
KNN with K=11 and Manhattan Distance	98.26%	1.74%
KNN with K=11 and Chebyshev Distance	95.21%	4.79%
KNN with K=11 and Euclidean Distance	97.37%	2.63%
*Correctly Classified Instances		·

Table 4.8: Results for N = 32

\*\*Incorrectly Classified Instances

value for N onward, the results could start to become worse. Perhaps if we normalized the coordinates according to the bounding box instead of a grid divided into cells, we could obtain the best results. We observed that, for all experiments with the KNN Classifier, as the k value increased, the percentage of correctly classified instances decreased. This happens because if we consider a high number of nearest points, we start to observe points that are not so near to the considered point and the final result is affected of an amount of very different points from the current point. Then, the best results came with k = 1, in all the experiments of the KNN Classifier. We consider the KNN with k = 1 and the Manhattan distance as the winning algorithm in our experiments.

According to the concept of each distance measure, our inference for why we obtained the worst results using the Chebyshev distance is that this distance undervalues the distance between the body parts in each frame and the classifier makes mistakes when making its predictions. The Chebyshev distance gives the longest distance considering all of the axis distances from point A to another point B. Then, the body parts can be closer than they actually are to each other. However, the Manhattan and Euclidean distances can be more realistic for human movements. This last assumption should be the reason
Classification with Grid $64 \times 64$	Correct*	Incorrect**
C4.5 Gain Ratio Decision Tree	98.54%	1.46%
Naïve Bayes	77.02%	22.98%
KNN with K=1 and Manhattan Distance	99.81%	0.19%
KNN with K=1 and Chebyshev Distance	99.62%	0.38%
KNN with K=1 and Euclidean Distance	99.81%	0.19%
KNN with K=3 and Manhattan Distance	99.62%	0.38%
KNN with K=3 and Chebyshev Distance	98.26%	1.74%
KNN with K=3 and Euclidean Distance	99.34%	0.66%
KNN with K=5 and Manhattan Distance	99.44%	0.56%
KNN with K=5 and Chebyshev Distance	97.23%	2.77%
KNN with K=5 and Euclidean Distance	99.20%	0.80%
KNN with K=7 and Manhattan Distance	99.25%	0.75%
KNN with K=7 and Chebyshev Distance	96.76%	3.24%
KNN with K=7 and Euclidean Distance	98.92%	1.08%
KNN with K=9 and Manhattan Distance	99.01%	0.99%
KNN with K=9 and Chebyshev Distance	95.39%	4.61%
KNN with K=9 and Euclidean Distance	98.50%	1.50%
KNN with K=11 and Manhattan Distance	98.50%	1.50%
KNN with K=11 and Chebyshev Distance	94.55%	5.45%
KNN with K=11 and Euclidean Distance	97.84%	2.16%
*Correctly Classified Instances		

Table 4.9: Results for N = 64

\*\*Incorrectly Classified Instances

for the best results for the Manhattan distance, and the Euclidean distance gives very similar results in comparison to the Manhattan distance.

#### 4.5 Discussion

We made a comparison among different classification algorithms in human pose recognition and the game context. In this work, we proposed and developed a detailed analysis using our own pose detection and tracking method, called M5AIE, while using different algorithms: the C4.5 Gain Ratio Decision Tree [32], Naïve Bayes Classifier [33] and K-Nearest Neighbor (KNN) Classifier [34]. The selected pose classes were inspired by the Jecripe game [17] and we added three more poses in our experiments.

In the experiments with categorical attributes, the C4.5 and Naïve Bayes worked well, judging by the fact that they correctly classified more than 97% of the instances. This result shows that the output of the tracking and labeling stages produces qualified tuples that can be used with the adopted classification techniques. This result also provided the decision to continue to make more experiments with classifiers. Additionally, we are

convinced that the used classifiers are suitable for pose classification purposes.

In the experiments with numerical attributes, we used three volunteers that had very different biotypes to collect the pose sequences with variations in the numbers of images and poses. In addition to the different classification algorithms, we tested three types of distances: Manhattan, Chebyshev and Euclidean. In all the experiments with the KNN Classifier, as the k value increased, the percentage of correctly classified instances decreased. This happens because if we consider a high number of nearest points, we start to observe very different points that could be far away from the considered point and they affect the final result. We consider KNN with k = 1 and the Manhattan distance as the winner because it provided the best results in all of the experiments. We believe that the coordinates of the five main body parts can be normalized in the bounding box because, in our experiments, as long as we increased the division of the used grid (8, 16, 32 and 64), the results became better. However, we also believe that there is a limit when dividing the grid. Further experiments should be performed to find the value for which the division does not make sense anymore. Additionally, further experiments should be performed to prove that normalized coordinates could be a good choice in the usage of a bounding box for cell definition. We presented the results of this chapter in [71].

## Chapter 5

## **Conclusions and Future Directions**

The focus of this thesis is on Computer Vision and Digital Games research. In fact, the primary motivation for this work is to contribute research that makes it easier to implement different concepts in Natural User Interfaces. Since the beginning of the development of the Jecripe game [17], there was the intention to stimulate the movements of children with Down syndrome throught NUI [18]. However, it was not possible to integrate the presented method for human body part detection, tracking and human pose classification with the aforementioned game. We presented several studies that are related to each of the challenges that we had to address. The first challenge that we addressed was the background subtraction task. We presented a study in which we described a comparison among four different background subtraction algorithms. This study included adaptations on three methods that were not developed to address depth information. This study motivated us to use the *Minimum Background* algorithm and we published this work in [41].

This thesis describes the M5AIE method for detecting and tracking five main parts of the human body (head, hands and feet) in sequences of RGB-D images. The proposed approach combines an effective background subtraction method, the discrete medial axis transformation, in the construction of simpler graphs to be used in the detection of AGEX points, heuristics for labeling, and ASIFT-based tracking of labeled structures.

The proposed M5AIE algorithm was implemented in proof-of-concept programs. We did not consider the computational load of this specific implementation to be a fundamental requirement because the main goal in the experiments was to assert the possibility of using a hybrid technique for body part detection, tracking and pose classification. The bottleneck for real-time results computing the Medial Axis and the ASIFT algorithm. Pinto and Freitas [67] showed real-time results for the Medial Axis Transform. In 2013, Chiu et al. [68] presented a fast SIFT design for real-time visual feature extraction. Be-

cause ASIFT depends on the SIFT results, we believe that the M5AIE can be efficiently implemented and used as part of real-time tracking solutions that are applied to games. To prove that the M5AIE method is effective, including on the human pose prediction task, we made a comparison among the classification algorithms when applied to human pose recognition in the game context. In this study, we proposed and developed a detailed analysis using the M5AIE with different algorithms: the C4.5 Gain Ratio Decision Tree [32], Naïve Bayes Classifier [33] and K-Nearest Neighbor (KNN) Classifier [34]. We used three volunteers with very different biotypes to collect the pose sequences, using variations in the numbers of images and poses. In addition to the different classification algorithms, we tested three types of distances: Manhattan, Chebyshev and Euclidean. We consider KNN with k = 1 and the Manhattan distance to be the winner because it provided the best results in all of the experiments. We believe that the coordinates of the five main body parts can be normalized in the bounding box because, in our experiments, as long as we increased the division of the used grid (8, 16, 32 and 64), the results improved. However, we also believe that there is a limit on dividing the grid. We presented

We believe that the contributions of this work are as follows:

the study of the classification algorithms in [71].

- A comparison among different background subtraction algorithms;
- The combination of the AGEX and ASIFT methods using aligned RGB and depth images for labeling five major defined body parts (hands, feet and head);
- Tracking each of the body parts using an adapted ASIFT matching algorithm;
- Description of how different classification algorithms can be used in human pose classification in the digital games context; and
- A comparative analysis of three classification algorithms in human pose classification.

#### 5.1 Future Work

Concerning the background subtraction task, we identified some open questions for future work in this area. Scenarios with a moving Kinect or a dynamic background require more sophisticated algorithms. Examples of dynamic background include moving trees and moving background (the window of a train, for instance). Future work will also include the application of the M5AIE method with a partial occlusion treatment between two users and the use and comparison of more classifiers for pose recognition with multiple subjects. The occlusion treatment is a very large challenge in Computer Vision. The solution of self-occlusions and occlusions between two or more people could change research in human-computer interaction, digital games and several other research areas.

With regards to the classification study, further experiments should be performed to determine the N value (which divides a bounding box), whose division is no longer logical. Additionally, further experiments should be performed to demonstrate that normalized coordinates could be a good choice in using a bounding box for cell definition. Moreover, further experiments should be performed using dozens (or maybe hundreds) of volunteers to check if the classification models would be affected. If the results using considerably more volunteers are similar to our experiments, then the classification models are good and are independent of the number of volunteers.

Returning to the considerations of games research, an important contribution involves the integration of the M5AIE method into digital games. This integration could provide more studies with disabled people, including rehabilitation and health care contributions. More contributions are possible on e-learning and human-computer interaction. This study is the first study of several projects that will include NUI to help people with different profiles.

# Bibliography

- J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, and A. Blake, "Real-time human pose recognition in parts from single depth images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (CVPR), Colorado Springs, CO, USA, 2011, pp. 1297–1304.
- [2] R. Raskar, H. Nii, B. deDecker, Y. Hashimoto, J. Summet, D. Moore, Y. Zhao, J. Westhues, P. Dietz, J. Barnwell, S. Nayar, M. Inami, P. Bekaert, M. Noland, V. Branzoi, and E. Bruns, "Prakash: lighting aware motion capture using photosensing markers and multiplexed illuminators," *ACM Trans. Graph.*, vol. 26, July 2007.
- [3] J. Park and Y.-L. Yoon, "Led-glove based interactions in multi-modal displays for teleconferencing," in *Proceedings of the 16th International Conference on Artificial Reality and Telexistence–Workshops*, ser. ICAT '06. Washington, DC, USA: IEEE Computer Society, 2006, pp. 395–399.
- [4] B. Stenger, A. Thayananthan, P. H. S. Torr, and R. Cipolla, "Model-based hand tracking using a hierarchical bayesian filter," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, pp. 1372–1384, September 2006.
- [5] G. Dewaele, F. Devernay, R. Horaud, and F. Forbes, "The alignment between 3d data and articulated shapes with bending surfaces," in *In Proceedings of the 9th European Conference on Computer Vision*, Springer, Ed., vol. Part III. Springer, 2006, pp. 578–591.
- [6] C. Theobalt, I. Albrecht, J. Haber, M. Magnor, and H.-P. Seidel, "Pitching a baseball: tracking high-speed motion with multi-exposure images," ACM Trans. Graph., vol. 23, pp. 540–547, August 2004.
- [7] R. Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," ACM Trans. Graph., vol. 28, pp. 63:1–63:8, July 2009.

- [8] L. Bourdev and J. Malik, "Poselets: Body part detectors trained using 3d human pose annotations," in *International Conference on Computer Vision*, sep 2009.
  [Online]. Available: http://www.eecs.berkeley.edu/lbourdev/poselets
- [9] L. Bourdev, S. Maji, T. Brox, and J. Malik, "Detecting people using mutually consistent poselet activations," in *European Conference on Computer Vision*, sep 2010. [Online]. Available: http://www.eecs.berkeley.edu/ lbourdev/poselets
- [10] A. L. Betker, A. Desai, C. Nett, N. Kapadia, and T. Szturm, "Game-based exercises for dynamic short-sitting balance rehabilitation of people with chronic spinal cord and traumatic brain injuries," *Physical Therapy*, vol. 87, no. 10, pp. 1389–1398, Oct. 2007.
- [11] J. W. Burke, M. D. J. McNeill, D. K. Charles, P. J. Morrow, J. H. Crosbie, and S. M. McDonough, "Optimising engagement for stroke rehabilitation using serious games," *Vis. Comput.*, vol. 25, pp. 1085–1099, October 2009. [Online]. Available: http://portal.acm.org/citation.cfm?id=1667552.1667556
- [12] J. Decker, H. Li, D. Losowyj, and V. Prakash, "Wiihabilitation: rehabilitation of wrist flexion and extension using a wiimote-based game system," *Governor's School* of Engineering and Technology Research Journal, 2009.
- [13] C. Schönauer, T. Pintaric, and H. Kaufmann, "Full body interaction for serious games in motor rehabilitation," in *Proceedings of the 2nd Augmented Human International Conference*, ser. AH '11. New York, NY, USA: ACM, 2011, pp. 4:1–4:8.
- [14] T. Pintaric and H. Kaufmann, "Affordable infrared-optical pose-tracking for virtual and augmented reality," in *Proceedings of Trends and Issues in Tracking for Virtual Environments Workshop*, IEEE, Ed., vol. VR. IEEE, 2007.
- [15] J. K. T. Tang, J. C. P. Chan, and H. Leung, "Interactive dancing game with real-time recognition of continuous dance moves from 3d human motion capture," in *Proceedings of the 5th International Conference on Ubiquitous Information Management and Communication*, ser. ICUIMC '11. New York, NY, USA: ACM, 2011, pp. 50:1–50:9.
- [16] M. Moore and J. Sward, Introduction to the Game Industry. Prentice Hall, 2007.
- [17] A. Brandão, L. Brandão, G. Nascimento, B. Moreira, C. N. Vasconcelos, and E. Clua, "Jecripe: stimulating cognitive abilities of children with down syndrome in pre-scholar age using a game approach," in *Proceedings of the 7th International Conference on*

Advances in Computer Entertainment Technology, ser. ACE '10. New York, NY, USA: ACM, 2010, pp. 15–18.

- [18] A. Brandão, E. Passos, C. N. Vasconcelos, A. Conci, E. Clua, S. Brandão, P. Mourão, and M. d'Ornellas, "Stimulating imitation of children with down syndrome using a game approach," in *Proceedings of the VIII Brazilian Symposium on Games and Digital Entertainment*, ser. SBGAMES '09, 2009, pp. 97–100.
- [19] A. Brandão, D. Trevisan, L. Brandão, B. Moreira, G. Nascimento, P. Mourão, C. N. Vasconcelos, and E. Clua, "Semiotic inspection of a game for children with down syndrome," in *Proceedings of the IX Brazilian Symposium on Games and Digital Entertainment*, ser. SBGAMES '10, 2010.
- [20] C. S. de Souza, C. F. Leitao, R. O. Prates, and E. J. da Silva, "The semiotic inspection method," in *IHC '06: Proceedings of VII Brazilian symposium on Human factors in computing systems.* New York, NY, USA: ACM, 2006, pp. 148–157.
- [21] A. Kar, "Skeletal tracking using Microsoft Kinect," *Methodology*, vol. 1, pp. 1–11, 2010, department of Computer Science and Engineering, IIT Kanpur.
- [22] M. Tang, "Recognizing hand gestures with Microsoft's Kinect," Stanford University, Tech. Rep., 2011, department of Electrical Engineering, Stanford University.
- [23] J. Pan, S. Chitta, and D. Manocha, "Probabilistic collision detection between noisy point clouds using robust classification," in *International Symposium on Robotics Research (ISRR)*, 2011.
- [24] Y. Cui and D. Stricker, "3d shape scanning with a kinect," in ACM SIGGRAPH 2011 posters, ser. SIGGRAPH '11. New York, NY, USA: ACM, 2011.
- [25] G. Gordon, T. Darrell, M. Harville, and J. Woodfill, "Background estimation and removal based on range and color," in *Computer Vision and Pattern Recognition*, 1999. IEEE Computer Society Conference on., vol. 2, 1999.
- [26] Y. Ivanov, A. Bobick, and J. Liu, "Fast lighting independent background subtraction," *International Journal of Computer Vision*, vol. 37, no. 2, pp. 199–207, June 2000.
- [27] S. Brutzer, B. Höferlin, and G. Heidemann, "Evaluation of Background Subtraction Techniques for Video Surveillance," in *Computer Vision and Pattern Recognition* (CVPR). IEEE, 2011, pp. 1937–1944.

- [28] L. Xia, C. C. Chen, and J. K. Aggarwal, "Human detection using depth information by Kinect," in International Workshop on Human Activity Understanding from 3D Data in conjunction with CVPR (HAU3D), Colorado Springs, CO, Jun. 2011.
- [29] C. Plagemann, V. Ganapathi, D. Koller, and S. Thrun, "Real-time identification and localization of body parts from depth images," in *Proceedings of the IEEE International Conference on Robotics & Automation (ICRA)*, Anchorage, Alaska, USA, 2010, pp. 3108–3113.
- [30] J. Huang, J. Lu, and C. X. Ling, "Comparing naive bayes, decision trees, and svm with auc and accuracy," in *IEEE International Conference on Data Mining*, 2003.
- [31] N. B. Amor, S. Benferhat, and Z. Elouedi, "Naive bayes vs decision trees in intrusion detection systems," in *Proceedings of the 2004 ACM symposium on Applied computing*, ser. SAC '04. New York, NY, USA: ACM, 2004.
- [32] J. R. Quinlan, C4.5: programs for machine learning. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1993.
- [33] P. Domingos and M. Pazzani, "On the optimality of the simple bayesian classifier under zero-one loss," *Machine learning*, vol. 29, no. 2, pp. 103–130, 1997.
- [34] T. Cover and P. Hart, "Nearest neighbor pattern classification," Information Theory, IEEE Transactions on, vol. 13, no. 1, pp. 21–27, 1967.
- [35] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "RGB-D mapping: Using depth cameras for dense 3D modeling of indoor environments," in *Proc. of the International Symposium on Experimental Robotics (ISER)*, Delhi, India, 2010.
- [36] J. Sturm, S. Magnenat, N. Engelhard, F. Pomerleau, F. Colas, W. Burgard, D. Cremers, and R. Siegwart, "Towards a benchmark for RGB-D SLAM evaluation," in *Proc.* of the RGB-D Workshop on Advanced Reasoning with Depth Cameras at Robotics: Science and Systems Conf. (RSS), Los Angeles, USA, June 2011.
- [37] T. Bouwmans, "Recent advanced statistical background modeling for foreground detection: A systematic survey," *RPCS*, vol. 4, no. 3, pp. 147–176, 2011.
- [38] A. Brandt, K. Karmann, and S. Lanser, "Recursive motion estimation based on a model of the camera dynamics." in *European Signal Processing Conference.*, vol. 2, 1990, pp. 959–962.

- [39] S.-c. S. Cheung and C. Kamath, "Robust techniques for background subtraction in urban traffic video," in *Electronic Imaging*. International Society for Optics and Photonics, 2004, pp. 881–892.
- [40] E. Stone and M. Skubic, "Evaluation of an inexpensive depth camera for passive inhome fall risk assessment," in 5th International Conference on Pervasive Computing Technologies for Healthcare (PervasiveHealth), 2011, 2011, pp. 71–77.
- [41] K. Greff, A. Brandão, S. Krauß, D. Stricker, and E. Clua, "A comparison between background subtraction algorithms using a consumer depth camera," in *Proceedings* of International Conference on Computer Vision Theory and Applications, vol. 1. Rome, Italy: SciTePress, 2012, pp. 431–436.
- [42] K. Khoshelham, "Accuracy analysis of kinect depth data," in International Society for Photogrammetry and Remote Sensing (ISPRS), vol. 38, 2011, pp. 1–6.
- [43] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, "Wallflower: Principles and practice of background maintenance," in *IEEE International Conference on Computer Vision*, vol. 1. Los Alamitos, CA, USA: IEEE Computer Society, 1999, pp. 255–261.
- [44] K. Cannons, "A review of visual tracking," York University, Department of Computer Science and Engineering, Tech. Rep. CSE-2008-07, Sep. 2008.
- [45] C. Wren, A. Azarbayejani, T. Darrell, and A. Pentland, "Pfinder: Real-time tracking of the human body," *Pattern Analysis and Machine Intelligence, IEEE Transactions* on, vol. 19, no. 7, pp. 780–785, Jul. 1997.
- [46] K. Kim, T. H. Chalidabhongse, D. Harwood, and L. Davis, "Realtime foreground-background segmentation using codebook model," *Real-Time Imaging*, vol. 11, no. 3, pp. 172–185, Jun. 2005. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1077201405000057
- [47] T. Kohonen, "Learning vector quantization," Neural Networks, vol. 1, pp. 3 16, 1988.
- [48] A. Telea, "An image inpainting technique based on the fast marching method," Journal of Graphics Tools, vol. 9, no. 1, pp. 25–36, 2004.

- [49] T. B. Moeslund, A. Hilton, and V. Krüger, "A survey of advances in vision-based human motion capture and analysis," *Computer Vision and Image Understanding*, *Elsevier Science Inc.*, vol. 104, pp. 90–126, November 2006.
- [50] E. Dijkstra, "A note on two problems in connexion with graphs," Numerische Mathematik, vol. 1, no. 1, pp. 269–271, 1959.
- [51] V. Ganapathi, C. Plagemann, S. Thrun, and D. Koller, "Real time motion capture using a single time-of-flight camera," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, San Francisco, CA, USA, 2010, pp. 755–762.
- [52] A. Baak, M. Müller, G. Bharaj, H.-P. Seidel, and C. Theobalt, "A data-driven approach for real-time full body pose reconstruction from a depth camera," in *IEEE 13th International Conference on Computer Vision*, Bacelona, Spain, Nov 2011, pp. 1092–1099.
- [53] D. Lowe, "Distinctive image features from scale-invariant keypoints," International Journal of Computer Vision, vol. 60, no. 2, pp. 91–110, January 2004.
- [54] J.-M. Morel and G. Yu, "Asift: A new framework for fully affine invariant image comparison," SIAM J. Img. Sci., vol. 2, no. 2, pp. 438–469, Apr. 2009.
- [55] G. Yu and J.-M. Morel, "Asift: An algorithm for fully affine invariant comparison," *Image Processing On Line*, p. Online, 2011.
- [56] M. Ye, X. Wang, R. Yang, L. Ren, and M. Pollefeys, "Accurate 3d pose estimation from a single depth image," in *Proceedings of International Conference on Computer Vision.* IEEE, 2011, pp. 731–738.
- [57] P. Henry, M. Krainin, E. Herbst, X. Ren, and D. Fox, "Rgb-d mapping: Using kinectstyle depth cameras for dense 3d modeling of indoor environments," *The International Journal of Robotics Research*, vol. 31, no. 5, pp. 647–663, 2012.
- [58] K. Lai, L. Bo, X. Ren, and D. Fox, "Sparse distance learning for object recognition combining rgb and depth information," in *IEEE International Conference on on Robotics and Automation*, 2011.
- [59] F. Endres, J. Hess, N. Engelhard, J. Sturm, D. Cremers, and W. Burgard, "An evaluation of the rgb-d slam system," in *Robotics and Automation (ICRA)*, 2012 *IEEE International Conference on*, 2012, pp. 1691–1696.

- [60] H. Bay, A. Ess, T. Tuytelaars, and L. V. Gool, "Speeded-up robust features (surf)," Computer Vision and Image Understanding, vol. 110, no. 3, pp. 346 – 359, 2008.
- [61] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *Computer Vision (ICCV)*, 2011 IEEE International Conference on, 2011, pp. 2564–2571.
- [62] N. Silberman and R. Fergus, "Indoor scene segmentation using a structured light sensor," in *Proceedings of the International Conference on Computer Vision - Workshop* on 3D Representation and Recognition, 2011, pp. 601–608.
- [63] S. May, D. Droeschel, D. Holz, C. Wiesen, and S. Fuchs, "3d pose estimation and mapping with time-of-flight cameras," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Workshop on 3D Mapping*, Nice, France, October 2008, pp. 1–6.
- [64] H. Blum, "A transformation for extracting new descriptors of shape," Models for the perception of speech and visual form, vol. 19, no. 5, pp. 362–380, 1967.
- [65] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, 3rd ed. Prentice Hall, 2008.
- [66] L. A. Schwarz, A. Mkhitaryan, D. Mateus, and N. Navab, "Human skeleton tracking from depth data using geodesic distances and optical flow," *Image Vision Comput.*, vol. 30, no. 3, pp. 217–226, Mar. 2012.
- [67] F. Pinto and C. Freitas, "Fast medial axis transform for planar domains with general boundaries," in *Computer Graphics and Image Processing (SIBGRAPI)*, 2009 XXII Brazilian Symposium on, 2009, pp. 96–103.
- [68] L.-C. Chiu, T.-S. Chang, J.-Y. Chen, and N.-C. Chang, "Fast sift design for real-time visual feature extraction," *Image Processing*, *IEEE Transactions on*, vol. 22, no. 8, pp. 3158–3167, 2013.
- [69] J. Shotton, R. Girshick, A. Fitzgibbon, T. Sharp, M. Cook, M. Finocchio, R. Moore, P. Kohli, A. Criminisi, A. Kipman, and A. Blake, "Efficient human pose estimation from single depth images," in *Decision Forests for Computer Vision and Medical Image Analysis*, ser. Advances in Computer Vision and Pattern Recognition. Springer London, 2013, pp. 175–192.

- [70] G. Rogez, J. Rihan, S. Ramalingam, C. Orrite, and P. H. Torr, "Randomized trees for human pose detection," *Computer Vision and Pattern Recognition*, *IEEE Computer Society Conference on*, vol. 0, pp. 1–8, 2008.
- [71] A. Brandão, L. A. F. Fernandes, and E. Clua, "A comparative analysis of classification algorithms applied to m5aie-extracted human poses," in *Proceedings of the XII Brazilian Symposium on Games and Digital Entertainment*, ser. SBGAMES '13, 2013.
- [72] V. Mota, E. Perez, M. Vieira, L. Maciel, F. Precioso, and P. Gosselin, "A tensor based on optical flow for global description of motion in videos," in *Graphics, Patterns and Images (SIBGRAPI), 2012 25th SIBGRAPI Conference on*, 2012, pp. 298–301.
- [73] L. Figueiredo, J. Teixeira, A. Cavalcanti, V. Teichrieb, and J. Kelner, "An open-source framework for air guitar games," in *Games and Digital Entertainment (SBGAMES)*, 2009 VIII Brazilian Symposium on, 2009, pp. 74–82.
- [74] R. N. de Almeida, "Portuguese sign language recognition via computer vision and depth sensor," Master's thesis, Lisbon University Institute, Lisbon, October 2011.
- [75] A. Bleiweiss, D. Eshar, G. Kutliroff, A. Lerner, Y. Oshrat, and Y. Yanai, "Enhanced interactive gaming by blending full-body tracking and gesture animation," in ACM SIGGRAPH ASIA 2010 Sketches, ser. SA '10. New York, NY, USA: ACM, 2010, pp. 34:1–34:2.
- [76] T. Qi, Y. Feng, J. Xiao, Y. Zhuang, X. Yang, and J. Zhang, "A semantic feature for human motion retrieval," *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 399–407, 2013.
- [77] G. Cimen, H. Ilhan, T. Capin, and H. Gurcay, "Classification of human motion based on affective state descriptors," *Computer Animation and Virtual Worlds*, vol. 24, no. 3-4, pp. 355–363, 2013.
- [78] M. Sun, P. Kohli, and J. Shotton, "Conditional regression forests for human pose estimation," in *Computer Vision and Pattern Recognition (CVPR)*, 2012 IEEE Conference on, 2012, pp. 3394–3401.
- [79] J. Han, M. Kamber, and J. Pei, *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[80] I. H. Witten and E. Frank, Data Mining: Practical machine learning tools and techniques. Morgan Kaufmann, 2011.