Faculdade de Engenharia da Universidade do Porto

FEUP

Sit-to-Stand Movement Analysis using the Kinect Platform

Francisco José Macedo Fernandes

Master in Bioengineering
(Biomedical Engineering field)

Supervisor: Jorge Alves da Silva

July, 2013

Sit-to-Stand Movement Analysis using the Kinect Platform

Francisco José Macedo Fernandes

Master in Bioengineering
(Biomedical Engineering field)

Approved in oral examination by the committee:

(Supervisor)

23th July, 2013

Abstract

In the last decades, several research projects have been devoted to the understanding and analysis the sit-to-stand (STS) movement, its characteristics and impact in our daily lives. Despite the efforts, there isn't currently a standard method to analyse its normality. Most of the developed methods use markers in order to collect data from points of interest. The Kinect is a new technology which enables the acquisition of three dimensional depth images in real time and body-joint tracking without markers. In this dissertation a new STS movement analysis system using the Kinect platform is presented.

The STS movement was divided into 5 phases: "Sitting", "Phase 1", "Phase 2", "Phase 3" and "Standing". An initial segmentation of the acquired movements was performed, obtaining time windows of interest. From this initial segmentation a manual evaluation of the data was performed, creating an initial dataset. Each sample of the dataset corresponds to a 13-dimensional feature vector, collected from a single frame of the movement. In order to balance the classes the Synthetic Minority Over-sampling Technique (SMOTE) was used, obtaining a new dataset.

Hidden Markov Models (HMMs) classifiers, trained with the datasets were employed to classify the samples. In the training phase two different training algorithms - Baum-Welch algorithm and Segmental K-means algorithm - were used for training. A precision of 83% and recall of 87% were obtained for classifier used in the final application. Angles and angular velocities of the trunk, knees and ankles were extracted and analysed. An interface to display real time information of the movement was developed in order to give feedback to the user.

Resumo

Nas últimas décadas, diversos projetos de pesquisa têm-se dedicado a compreender e analisar o movimento Sentar-Levantar, as suas características e impacto em nossas vidas diárias. Apesar dos esforços, não existe atualmente nenhum método padrão para analisar a sua normalidade. A maioria dos métodos desenvolvidos requer o uso de marcadores para recolher dados de pontos de interesse. A Kinect é uma nova tecnologia que permite a aquisição de imagens de profundidade tridimensionais em tempo real e seguimento do corpo humano sem marcadores. Nesta dissertação é apresentado um novo sistema de análise de movimento Sentar-Levantar usando a plataforma Kinect.

O movimento Sentar-Levantar foi dividido em cinco fases: "*Sitting*", "*Phase 1*", "*Phase 2*", "*Phase 3*" e "*Standing*". Foi efetuada uma segmentação inicial dos movimentos, obtendo-se janelas de tempo de interesse. Depois da segmentação inicial foi realizada uma avaliação manual dos dados adquiridos, tendo sido criado um conjunto de dados inicial. As amostras recolhidas são vetores de 13 dimensões, recolhidos a partir de uma única *frame* do movimento. Com objetivo de equilibrar as classes a técnica *Synthetic Minority Over-sampling* (SMOTE) foi usada, obtendo-se um novo conjunto de dados.

Classificadores baseados em *Hidden Markov Models* (HMMs) foram treinados com os conjuntos de dados. Na fase de treino foram usados dois algoritmos – *Baum-Welch algorithm* e *Segmental K-means algorithm* – para treinar os classificadores. Obteve-se precisão de 83% e *recall* de 87% para o classificador utilizado na aplicação final. Ângulos e velocidades angulares do tronco, joelhos e tornozelos foram extraídos e analisados. Um interface com a informação do movimento em tempo real foi desenvolvido de forma a fornecer feedback ao utilizador.

Acknowlegments

I would like to thank Prof. Jorge Alves da Silva for the several suggestions given during the development of this project and for putting up with my personality.

I would also like to thank INEB (*Instituto de Engenharia Biomédica*) for the availability of the facilities and material.

Contents

List of Figures

List of Tables

Abbreviations and Symbols

List of abbreviations

3D	Three-dimensional
BWa	Baum-Welch algorithm
COM	Centre of mass
FN	False Negative
FP	False Positive
HMM	Hidden Markov Model
LED	Light-Emitting Diode
SDK	Software Development Kit
SKMa	Segmental K-means algorithm
SMOTE	Synthetic Minority Oversampling Technique
STS	Sit-to-Stand
TN	True Negative
TP	True Positive
WPF	Windows Presentation Foundation

List of symbols

γ	Ankle angle with the ground

Chapter 1

Introduction

1.1 - Context, Motivation and Objectives

Rising from a chair is one of the basic daily functions required for independent living. This is even more noticeable when a good mobility is required to perform daily tasks, such as using the bathroom, cooking or even going to the working site. The sit-to-stand (STS) movement is a particularly difficult task for elderly individuals, especially if any musculoskeletal or neurological disorders are present [1, 2]. Although it is just one of the many daily activities, it is performed in average 60 times a day by community-dwelling adults and young individuals [3]. The correct assessment of the STS movement is helpful in the determination of the functional level of a person [1-8].

Depending on the scope of the study, several evaluation methods can be used to classify and analyse aspects of the STS movement. These methods can vary from motion analysis systems [8, 9], force plates and goniometry [7] to electromyography analysis [10-12] and accelerometry [13]. Due to the high variety of factors that influence the STS movement, there isn't currently a standard method to characterise its normality and performance. Also, these methods are usually expensive, requiring specialised equipment and the help of health professionals in order to be performed, being highly time consuming activities.

The Kinect from Microsoft is a new accessible, affordable and programmable technology which enables real-time three-dimensional (3D) body-joint tracking [14, 15], along with localization and tracking of objects with good accuracy and resolution [14, 16]. Although the initial applications of the Kinect were mainly for videogames, several works are in development in other fields [17-22]. Some of the most promising fields of application are the human gesture recognition [20-22] and the rehabilitation [17-19] fields, where the Kinect can bring an interactive and dynamic environment to our homes, which used to be inaccessible.

The aim of this dissertation is to develop an approach for a computer aided analysis of the STS movement using the 3D body-joint tracking data acquired with the Kinect platform. The developed system should be as automatic as possible, giving feedback to the user about the movement and enabling the acquisition, and posterior consultation and analysis of the data.

Since no specific equipment besides the Kinect platform is required, this system shall be accessible to everyone with a Kinect, leaving open the possibility of its adaptation to the rehabilitation field as a home system to help with the patients' physiotherapy.

1.2 - System Overview

In order to analyse the STS movement, a system using the Kinect potentialities was idealized. The system starts by obtaining the 20 body-joint coordinates generated by the Kinect sensor in real time, during the performance of the STS movement for each captured frame. These movements were performed in a specific and controlled room setup.

An initial filtering was performed in order to remove noise and jittering from the samples. The movements were then manually segmented and analysed, frame-by-frame, labelling each frame accordingly. The objective of this manual segmentation and labelling is to have a ground truth and enable the training of a Hidden Markov Model (HMM) sequence classifier to recognize different classes. The main objective of the classification step is to divide the movement into different phases, in order to better analyse the characteristics of each phase.

A set of features was used to train the HMM sequence classifier. These features were obtained by processing the 3D body-joint data previously collected. This data consisted of successive space coordinates (x, y, z) of the 20 body-joints collected while performing the STS movement. A good set of features is important to reach good results in the classification step. While processing the acquired data, information about the movement can be obtained. This information can range from trajectory of the joints and centre of mass (COM), joint angle variations, angle variations between body segment to the duration of the movement and phases. The last step of the system is to give visual feedback to the user, showing this information and being able to save it for later usage and analysis.

1.3 - Contributions

In the following list are summarized the main contributions of this dissertation:

- It is presented a new system for the analysis of the STS movement. This system uses the Kinect platform to acquire data, providing a low-cost and accessible tool for home rehabilitation purposes.

- A new automatic segmentation of the STS movement based on HMMs is presented. A total of 5 main phases detected are “Sitting”, “Phase 1”, “Phase 2”, “Phase 3” and “Standing”.

- An assessment and analysis of the acquired angles and angular velocities is performed, providing an idea of the applicability of the system.

- A final application that allows the users to analyse the STS movement, review them and save the acquired for other applications.

1.4 - Document Outline

Chapter 2 (Literature Review) addresses several topics found in the literature that are relevant for presented work: different ways to analyse and subdivide the STS movement over the years, main characteristics that can be extracted from the STS movement, an overview of the Kinect platform main characteristics, along with the viability of the acquired data, some considerations about the features acquired with the Kinect and their limitations due to the factors that affect the STS movement, and finally classification methodologies with focus on the HMMs.

In *Chapter 3* we explore the methodologies, the decisions taken and reasoning behind these decisions. In this chapter a detailed description of the proposed system is presented

The next chapter (*Chapter 4*) describes the developed interface for the system. The libraries and software used in this work are also mentioned in this chapter. The experimental results are reported, analysed and discussed in C*hapter 5*. Lastly, conclusions are drawn and future work is proposed in *Chapter 6*.

Chapter 2

Literature Review

In the last decades, many researchers have developed and proposed several methods to analyse and evaluate the STS movement. Depending on the aim of the study the STS movement can be defined and analysed from different scopes. Kinematics, kinetics, muscle contraction and patients' functional evaluation are the most usual analysis methods [1, 2].

The Kinect brought to everybody's home an interactive and programmable system, which enables the collection of 3D depth images and body-joint tracking. Several researches of its applicability in the rehabilitation field are on development, showing promising results [17-19, 23-26].

In this chapter, we will organise the literature review by identifying the main contributions in the main phases of the STS movement analysis, namely for phases definition, movement segmentation, feature measurement and classification.

2.1 - STS Movement

To begin with, it is important to understand the basics of the STS movement, and how it has been described over the years, along with the major factors that impact its analysis.

Rising from a chair is a basic daily function required for independent living. The inability to perform such a task, depending on the degree of limitation, may lead to institutionalization, impaired functioning and reduced mobility in daily living activities. In the worst case scenario it may lead to death [1].

In order to successfully perform the STS movement, a shifting of weight from the buttocks and posterior thighs to the feet is required. This process requires an anterior followed by a vertical movement of the body's centre of mass (COM) [2]. This is executed primarily by a flexion of the hips and anterior movement of the head-arms-trunk segment, immediately followed by the extension of the hips, knees and ankles [2, 8, 27].

Depending on the scope of study, the way that the STS movement is defined varies and different definitions of phases are possible.

2.1.1 - STS movement subdivision into phases

Back in 1986, Nuzik et al. [7] developed a visual model of the STS movement pattern from film data collected of 38 women and 17 men. Using body landmarks as data points, angles of interest were defined and angle variations were recorded during the STS movement. In order to compare the movement between subjects, the movement time was divided into 5% increments, providing points of comparison. For each interval the mean and standard deviation of each angle were calculated across all subjects, along with the mean horizontal and vertical coordinates of the data points, creating a schematic of the entire movement cycle, as shown in Figure 2.1.

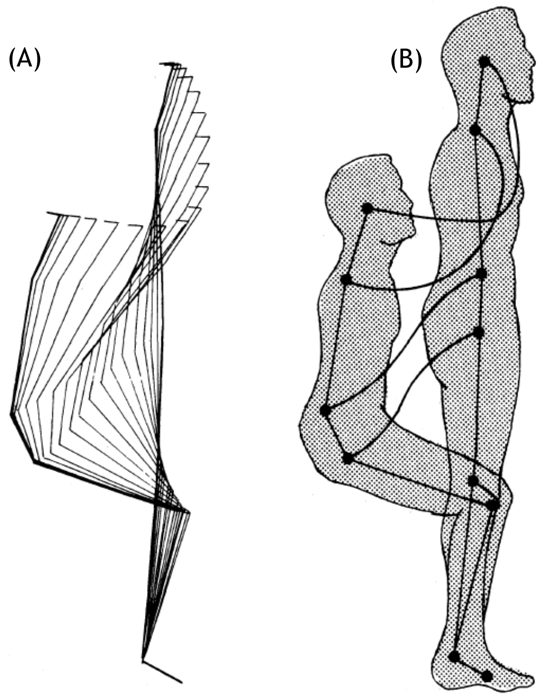


Figure 2.1 - (A) Diagram of a representative movement pattern; the data points were joined by lines to form stick lines; (B) Diagram of the trajectories of the data points at tragus, acromion, midiliac crest, hip and knee (image from [7]).

In the study the authors concluded that the STS movement could be subdivided into two main phases, the flexion phase which occurred during the first 35% of the movement cycle, and the extension phase, denoted by a reversal movement of the head and rapid extension of the knee. As seen on Figure 2.1, in this kinematic study the body landmarks used are the ankle, knee, hip, pelvis, trunk and the head. These points coincide with the some of the body-joint detected by the Kinect [28, 29]. In this work they were used in order to obtain a representative diagram of their trajectories during the STS movement, characterising the variation of the angles between body segments.

Later, Schenkman et al. [8] described the STS movement using kinematic and kinetic variables, defining 4 main phases for the movement. The first phase is the flexion-momentum phase. It starts with the initiation of the movement and ends just before the buttocks are lifted from the seat. The second phase is the momentum-transfer phase. This phase begins as the buttocks are lifted and ends when maximal ankle dorsiflexion is achieved. The third phase is the extension phase which is initiated just after maximum dorsiflexion and ends when the hips first cease to extend (including leg and trunk extension). The last one is the stabilization phase. It starts after hip extension is reached and ends when all motion associated with stabilization is completed [8].

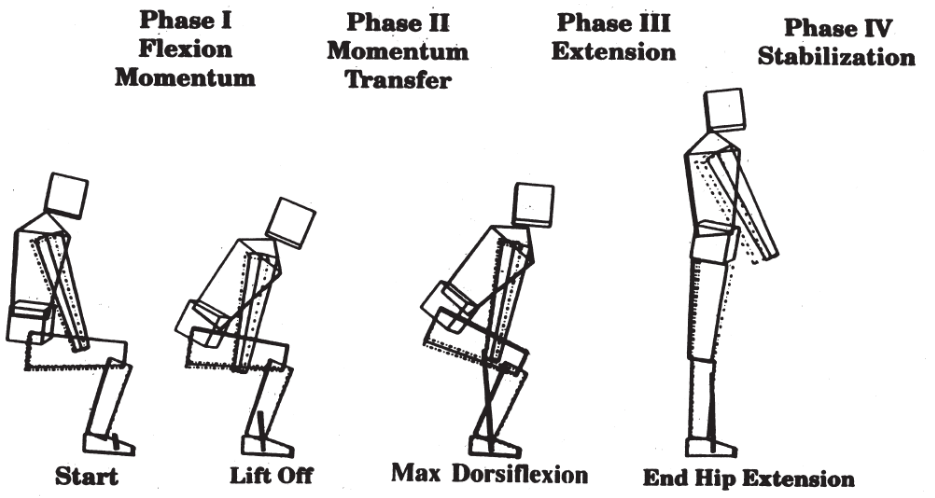


Figure 2.2 - Four phases of the STS movement marked by four key events (image from [8]).

The variables analysed in this study were joint angles, velocities and torques of specific upper and lower segments of the body, examining the maximum values achieved and the timing of this events. In order to be able to analyse these variables, multiple LEDs were embedded in fixed arrays and anchored to 11 body segments. These worked as markers in order to obtain the desired data. In this study, instead of marking and analysing specific body landmarks of the human body (like in [7]), the body segments that connect these landmarks were tracked and analysed.

In the previously described studies, markers were used in order to be able to obtain the data. These methods are not suitable to be used but in laboratory controlled environments, being expensive and in some cases uncomfortable, as it is possible to see from the setup of these studies ([7, 8, 30]).

Since the objective of the developed work was to develop a markless system to analyse the STS movement using the Kinect, the approaches described before were not completely suitable for our system.

In 2010, Goffredo et al. [9] explored a markless computer vision technique used to track natural elements on the human body surface. Translation, rotation and scaling were estimated by means of a maximum likelihood approach in the Gauss-Laguerre transform domain [9]. The technique was applied to the analysis of the STS movement in young and elderly people. The movements were subdivided into three phases defined by kinematics, and data, such as duration of the phases, trunk, knees and ankle angles, minimum trunk and ankle angles angle, and maximum trunk and ankle angular velocities, was extracted [9]. The first phase starts with the trunk flexion and ends at the beginning of knee extension. The second phase ends when the trunk reverses to extension. Finally, the third phase corresponds to the extension of the body to the standing position [9]. The representation of the 3 phases can be seen on Figure 2.3.

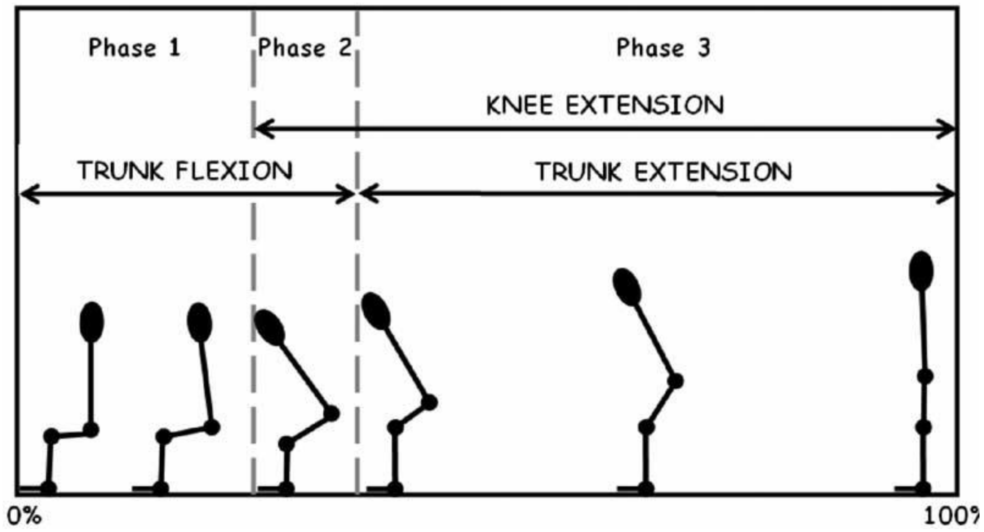


Figure 2.3 - STS motor task with phases defined by kinematic data (image from [9]).

When inspecting this subdivision of the movement, the first phase ends when the ankle angle decreases at 95% of its maximal value. The second phase ends when the trunk angle decreases to its minimum value. The movement ends when the trunk angle returns to 90º (upright stance) [9].

A subdivision into phases resulting from the combination of the aforementioned information was used in this work. The detailed description of this methodology is presented on *Chapter 3*.

2.2 - Kinect characteristics brief overview

The Kinect is a device developed by Microsoft which can either be used with the Xbox 360 gaming console, or with a computer. A detailed description of the Kinect characteristics and respective SDK can be found in [28], [29], [31].

By means of 3D depth images, RGB images and audio devices, the Kinect allows the control of games using the player's body instead of a remote controller. This is possible due to the identification of the user's joints and consequent movement tracking in a three-dimensional space by means of sensor data analysis [28, 29]. Understanding the limitations and errors associated with the Kinect measurements is important when defining the limitations of the platform.

2.2.1 -3D depth data

The 3D depth sensor is composed by an infrared (IR) projector and an IR camera, which together enable the acquisition of the depth images. The IR projector emits a single beam which is split into multiple beams. This is done by a diffraction grating which creates a constant pattern of speckles. The pattern is then captured and correlated against a reference pattern, which was obtained by capturing a plane at a known distance [14]. The system has a limited field of view, as shown on Figure 2.4.

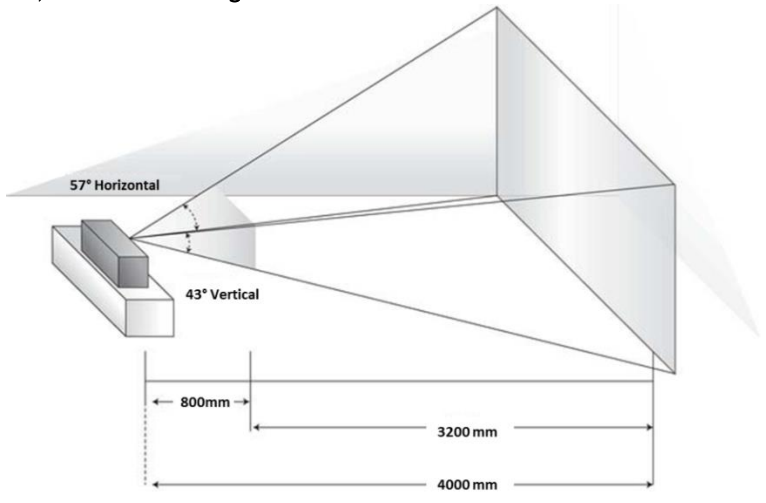


Figure 2.4 - Kinect field of view (adapted from [28]).

Also, the RGB camera allows the acquisition of two-dimensional colour video and is usually used for facial recognition and for displaying images on the screen during a game [28, 29].

Table 2.1 - Kinect characteristics discriminated, including ranges, resolutions, frames per second counts and fields of view (information taken from [28, 29]).

Depth Image Capture Range	Standard use: 0.8m to 4m
Depth Image Stream	Up to 640x480 16-bit, 30 fps
Colour Image Stream	Up to 1280x960 8-bit, 12 fps
Audio Stream	16-bit, 16 kHz
Field of view	Horizontal: 57° Vertical: 43°
Motor Tilt Range	±27°

In Table 2.1 additional specifications of the Kinect can be seen. Although it is stated that the range of the depth sensors varies from 0.8m to 4m, in reality the recommended range is from 0.9m to 3.7m, since the reliability of the depth data degrades near the edges of the field of view [28].

With the new versions, 1.5 and further, of the Kinect for Windows, there is a new tool – Kinect Studio - that enables the possibility of recording, playing back and debugging clips. Also, it is now possible to capture depth data beyond the 3.7m mark and with the near mode in a closer range than it used to, maintaining the reliability of the depth data acquired [29].

The primary function of the Kinect is to obtain 3D data. Figure 2.5 shows an example of a grayscale image obtained with the Kinect depth sensor.

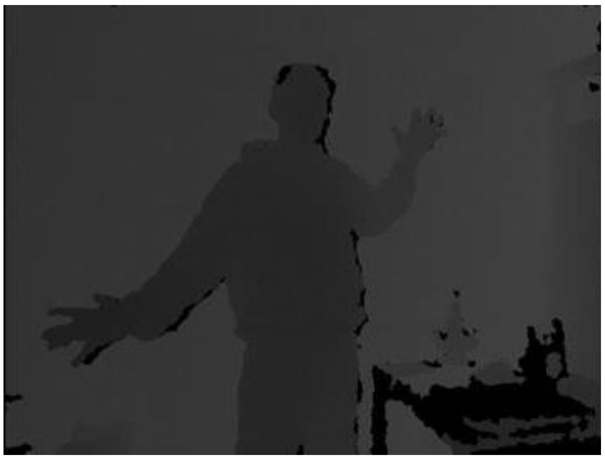


Figure 2.5 - Example of a raw depth image obtained using the Kinect (image from [28]).

The values of each pixel in the image range from 0 to 255, which correspond to their depth value. The depth value zero (black) means that the Kinect was unable to determinate the depth of the pixel. This usually happens due to the presence of shadows, low reflectivity and high reflectivity [28]. These values are obtained according to the coordinate system presented on Figure 2.6.

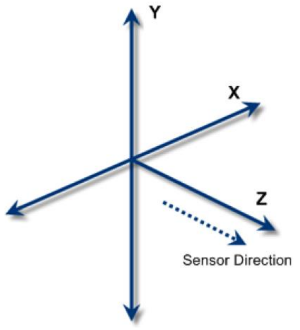


Figure 2.6 - Kinect's coordinate system (image from [29]).

This system of axis is projected on the subject and as a consequence the X and Y values can have negative or positive values, while the Z coordinate will always be positive. Figure 2.7 shows how this coordinate system is projected on a point.

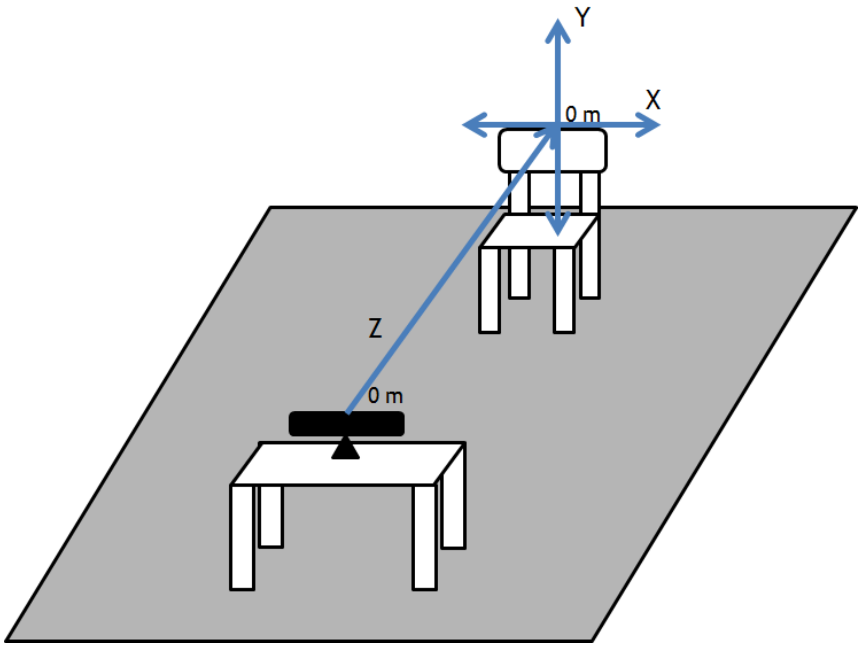


Figure 2.7 - Representation of the Kinect's coordinate system projection on a point.

The point immediately in front of the 3D depth system of the Kinect will have the X and Y coordinates equal to zero and the depth value Z. The zero value of the depth coordinate is on the Kinect.

Also, for each pixel a player index is attributed, referring the pixel as being part of the silhouette of a player or not. This enables the possibility of differentiation between multiple players and between the players and the background [28].

2.2.2 -Skeleton Tracking

The depth data acquired with the Kinect by itself is limited. In order to create useful applications with the Kinect, more information beyond the depth data for each pixel is required. The Kinect allows the processing of the depth data in order to establish the positions of 20 human skeleton joints, allowing the collection of the X, Y and Z values for each of the points seen on Figure 2.8 [28].

The algorithm for body-joint tracking starts by making a joint guess for each pixel of the depth image along with its confidence level. Based on several recordings, in which the joint positions were marked by hand later or markers are used, data was acquired. Analysing many depth frames with the joints correctly labelled and using machine learning techniques, the algorithm was trained to recognize the joints from depth images. Finally, taking this joint guesses and confidence levels into consideration, a skeleton is chosen [32].

This kind of approach for the skeleton tracking has the advantage of not requiring any kind of calibration in order to start the process, since an initial estimation of the skeleton is made and then adapted to the actual body [28, 29, 31].

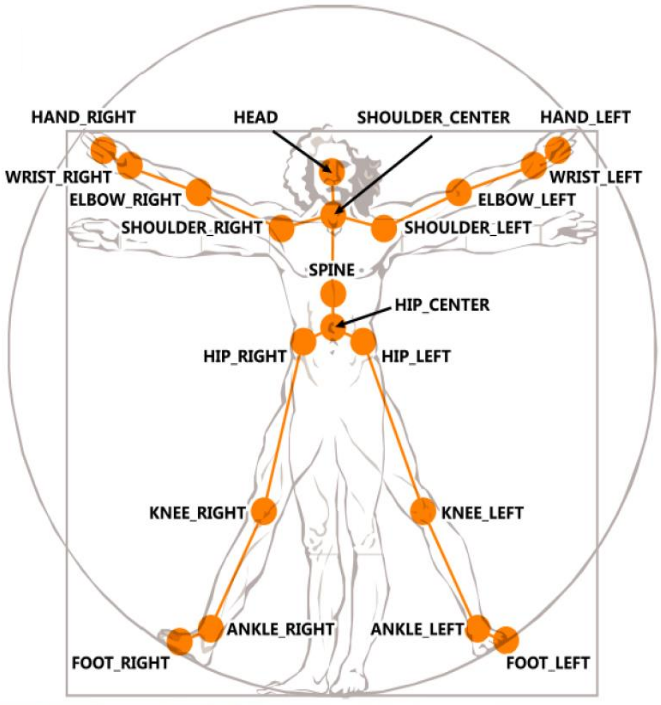


Figure 2.8 - Joint points detected by the Kinect algorithm (image from [31]).

2.2.3 -Viability of the acquired data

Khoshelham et al. [14] described some of the error sources and imperfections of the Kinect data. The three main sources of error described are the sensor, measurement setup and properties of the object surface.

The sensor errors usually refer to inadequate calibration and less accurate measurements of the disparities. An inadequate calibration will lead to systematic errors in the object coordinates of the points. Errors measuring the disparities will also influence the accuracy of individual points.

The setup where the Kinect is used is also important for the accuracy of the obtained data. For example lighting conditions will influence the correlation and the measurement of the disparities. Strong light will lead to low contrast of the IR image, leading to depth values of 0 (unknown). Also, depending on the geometry of the objects in analysis, some parts may be obstructed or shadowed leading to inaccurate results.

Finally, the properties of the object surface will also affect the measurements. Smooth and shiny surfaces may hinder the measurement of disparities, once again leading to inaccurate results.

K. LaBelle [33] tried to find answers to some interesting and crucial questions about the data acquired using the Kinect, specifically when using the Kinect SDK [31] and the OpenNI SDK [34] to acquire the data. Those questions were [33]:

- Is it possible to identify phases of movement from joint position data gathered during a therapy exercise?
- How consistent and stable are the joint positions during activities typically performed during a therapy session?

In this case, phases of movement were defined as: “sitting”, “moving” and “standing”. All the tests performed to validate the data were based on a STS exercise. The author described that the STS movement was frequently employed in stroke therapy and diagnostics. The data was collected at varying distances, from 1.5m to 3.5m [33].

The author reported that the data acquired was well-suited for identifying phases of the movement, being able to distinguish between the previously mentioned phases during the STS movement.

When investigating the joint position consistency the author analysed the standard deviations of joint positions obtained during “sitting” and “standing” phases. The author reported that in general, the consistency of the data was very high. Some of the results can be seen on Table 2.2.

Table 2.2 - Standard deviations of joint readings.

Joint	Kinect SDK [cm]
Head	1.8
Hip	1.2
Knee	1.5

Although, the author used one of the first versions of the Kinect SDK and did not use any kind of data smoothing or filtering in order to improve the results. The newer versions of the Kinect SDK offers a group of filtering and data smoothing options, that can remove jittering and improve the consistency and viability of the acquired data [29].

More recently, Clark et al. [35] verified the validity of the Microsoft Kinect for assessment of postural control. In the study the authors compared the joint positions obtained using the Kinect (collecting data using the Kinect SDK) and using the VICON Nexus V1.5.2 acquiring image data from 12 camera VICON MX motion analyses system (VICON, UK). The data acquired with the VICON system was deemed benchmark reference kinematic data. This system included the placement of markers on the head, arms, wrists, hands, trunk, pelvis, legs and feet [35, 36].

The relative and absolute reliability of the trials measurements for the Kinect and 3D camera methods were evaluated using intraclass correlation coefficient ($ICC_{2,1}$), and ratio coefficient of variation (CV), respectively [35].

The results of this study suggest that the Kinect provides anatomical landmark displacement (joint movement) and trunk angle data with great concurrent validity when compared to the commercially available 3D camera-based motion analysis system. It is also suggested that the Kinect has the potential to be used in clinical screening programs [35].

Even though an old version of the Kinect SDK was used in the primarily described study [33], the results are important and should be taken into consideration when designing our system. The information aforementioned ([33, 35]) encouraged the development of the system. But some considerations about the significance of the data acquired must be performed. This will be further discussed in *Chapter 3*.

2.3 - Feature Measurement

In order to obtain good results using a classifier, a good set of features is mandatory. Previous works using the Kinect sensors in body recognition applications have shown promising results. For example, Patsadu et al. [22] used some data mining classification methods in order to recognize three gestures: stand, sit down and lie down, obtaining and average accuracy of all classification methods of 93.72%. The authors used 1200 input vectors for each of the classes in study, making a total of 3600 input vectors (x, y, z) of 20 body-joint positions for the classifiers. These features had to be normalized to be comparable, since they had different units and were represented in different scales.

In Lai et al. [21] the authors focused their work on hand gesture recognition. In this case, the authors recognized 8 different hand gestures in real time achieving correct classification rates of over 99%. In this study a smaller set of features was used, since the main goal was the recognition of hand gestures.

Depending on the type of work and final application of the system, different features must be extracted and used. In our work the basic features extracted are the 20 body-joint positions. From these features, more relevant features are obtained. But it should also be taken into consideration that these features are affected by the environment in which they are acquired. It is important to understand the constraints implied in the analysis of the STS movement.

2.3.1 -Parameters that affect STS movement

Understanding the factors that can influence the STS movement is important for the development of this work. Although the final application of the system is accessible to everyone with a Kinect, a controlled environment helps acquiring understandable, consistent and comparable data.

From several studies it is possible to conclude that there are some major determinants affecting the STS movement – age, rising strategy and chair variables such as height, foot positioning, armrests, backrests [1, 2, 4, 27, 37-40]. Lower chair seats will require more generation of momentum or new repositioning of the feet in order to lower the momentum required [2, 11, 38-40]. The usage of armrests lowers the moments needed at knee, without influencing the range of motion of the joints [2, 40]. Also, repositioning of the feet may enable lower peak moments at the hip and knee [2, 38].

One of the simplest ways to characterize the STS movement would be to address the independence of the subject to perform the movement. The patient could be labelled as able or unable to perform the movement. From this point onwards, more conditions could be applied, defining different levels of functionality. For example, the use or armrests has a major influence in the performance of the STS movement, as described previously. Position of the feet, height of the chair, use of backrest and other conditions can be applied, in order to obtain different analysis of the STS movement.

After accessing if the subject is able to perform the STS without assistance or the use of armrests, the time taken to perform this movement could be another evaluation factor. This type of STS analysis usually requires more than one repetition of the test. It can be either defined by the number of repetitions performed in a certain time, or by defining the number of repetitions to be performed [2].

When analysing the STS movement, the affecting variables should be well defined and considered in order to obtain consistent results [1, 2]. The specifications of the movements captured in this work along with the room setup are further described in *Chapter 3*.

2.4 - Classification methods

The main objective of the classification step is to correctly label a new sample introduced in the system, taking into consideration the data used to train the classifier. Also, discarding as many false positives as possible, without losing too many true positives, is an important objective. In the context of this work, understanding the state-of-art of human activity recognition methodologies is important, since it is the main subject we are dealing with.

2.4.1 -Human Activity Recognition Methodologies

Aggarwal et al. [41] reviewed the state-of-art of the human activity recognition methodologies. The authors described the ability to recognize complex human activities as the key to the construction of several important applications. These applications could range from automated surveillance systems for public places to real-time monitoring of patients, children, and elderly persons [41]. The authors categorized human activities into four levels, depending on their complexity: *gestures*, *actions*, *interactions* and *group activities*. *Gestures* were described as the basic movements of a person´s body part, the components that describe the meaningful motion of a person. Research works using the Kinect for human gesture recognition are being developed, where several hand gestures and basic motions of the human body are recognized [20-22]. *Actions* were described as single-person activities composed of multiple gestures organized in time. “Walking”, “standing”, “laying down” and “sitting” are some examples of *actions*.

Aggarwal et al. [41] also described a classification system for activity recognition methodologies, dividing them into two categories: single-layered approaches and hierarchical approaches. Single-layered approaches are based on sequences of images representing and recognizing gestures and actions with sequential characteristics [41]. On the other hand, hierarchical approaches describe high-level human activities based on simpler activities (subevents) [41].

The single layered approaches were further divided into two classes: space-time approaches and sequential approaches, depending on the type of information used for the analysis. Space-time approaches use 3D (x, y, t) volume or a set of features extracted from the volume in order to create a model of a certain human activity. The video volumes are constructed by concatenation of image frames along the time axis, performing a comparison, in order to measure their similarities [41].

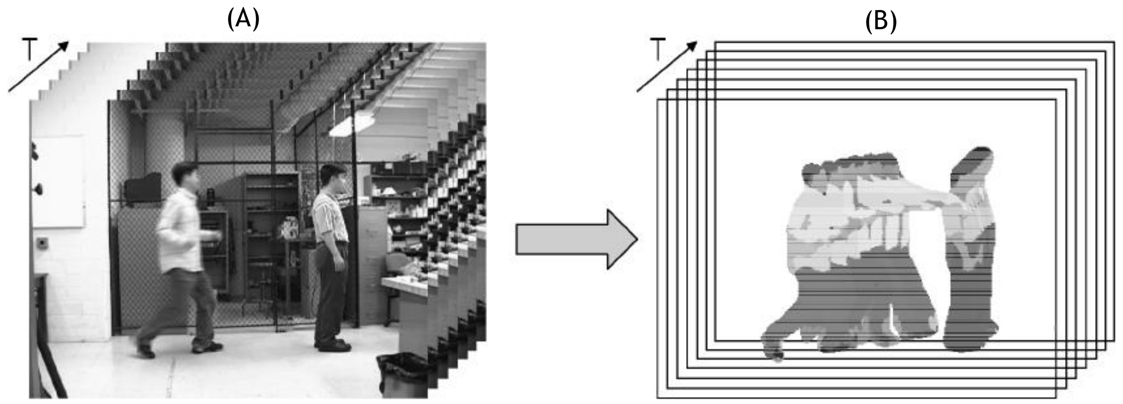


Figure 2.9 - Example of XYT volumes constructed by concatenating; **A** - entire images; **B** - foreground blob images obtained from a punching sequence (image from [41]).

Sequential approaches represent a human activity as a sequence of feature vectors extracted from images. The activities are recognized by searching for similar sequences [41].

In human gesture recognition tasks, a classification step is required in order to differentiate and segment the movements. Different classification methods such as Backpropagation Neural Network (BPNN) classifier ([22, 42, 43]), K-nearest-neighbour (KNN) classifier ([21, 42, 43]), Support Vector Machines (SVM) ([20-22, 42, 43]), decision tree (DT) ([22, 42, 43]), naïve Bayes (NB) ([22, 42, 43]) and Hidden Markov Models (HMMs) classifiers ([44-46]) can be used in order to perform the task. These tasks consisted mainly in the identification of certain basic movement patterns, such as “standing up”, “sitting down” and laying “down”, hand gesture recognition, and “going upstairs”, “going downstairs”, “walking”, “running” and “fighting”.

The STS movement could be analysed as an action or a sequence of gestures. The analysis of the movements could be done using a single-layered approach. The movement will have to undergo time segmentation, in order to obtain the time window of the actual movement. From this time window, the 20 body-joint positions along with the time frame can be extracted in order to analyse the movement. This would consist of four-dimensional (x, y, z, t) information for each body-joint acquired with the Kinect platform. This data would be used in order to analyse the movement and classify each frame, using data mining methods.

HMMs are a statistical tool used for modelling generative sequences characterized by a set of observable sequences. They are especially applied in temporal pattern recognition tasks [47-49]. The STS movement can be analysed as a sequence of smaller portions of the movement, especially if a full movement is analysed frame-by-frame. Also, the STS movement is composed by a set of phase-cycles. These phase-cycles correspond to the different phases, described in *section 2.1.1*, which can vary, depending on the undertaken approach.

2.4.2 -Hidden Markov Model brief overview

A detailed description of HMMs and applications to human gesture analysis can be found in [45], [44], [46], [48], [49].

HMM is a particular stochastic process with an underlying stochastic process that is not observable (hidden). This hidden process can observed through another set of stochastic processes, which produce a sequence of observed symbols [48]. HMM attempts to approximate or mimic the behaviour of a system in a succinct and manageable way. It is usually easier to work with an approximate model then deal with a real process. Also, being a probabilistic model, it attempts to capture the behaviour of a system with probabilities rather than with sure concrete rules, allowing some flexibility and adaptability. In this case a system can be something as simple as tossing a coin or something as complex as a speech recognition system [48, 49].

There are a finite number of states, *N*, in the model. Depending on the problem, the number and definition of states is bond to change. HMM assumes that in any sequence the current observation will only dependent on the immediate previous sequence. This property is called the Markov property [47-49]. If we consider a sequence of observations $O = O_1, O_2, \ldots, O_T$ and the corresponding sequence of states $I = I_1, I_2, \ldots, I_T$, the probability of any sequence *O* occurring when following a given sequence of states I can be stated as

$$p(O,I) = \prod_{t=1}^{T} p(I_t|I_{t-1})\, p(O_t|I_t) \quad , \tag{2.1}$$

where $p(I_t|I_{t-1})$ can be understood as the probability of being currently in the state I_t given that the previous state in the instant *t-1* is I_{t-1}, $p(O_t|I_t)$ is the probability of observing O_t at the instant *t*, given that the current state is I_t, *T* is the length of *O*, and *t* is the current time. *O* can be diversified, from sequences of point coordinates in 3D space to sequence of bitmap images, depending on the application. I is a sequence of integer labels, corresponding to the states [47-49].

To compute these probabilities two matrices A and B are required. The matrix A corresponds to matrix of state probabilities – the probabilities $p(I_t|I_{t-1})$ of changing from one state to another. The matrix B Is the matrix of observation probabilities – the distribution density $p(O_t|I_t)$ associated to a given state I_t. The compact notation $\lambda = (A, B, \pi)$ can be used to represent a HMM, where A represents the transition probabilities matrix, B represent the observation probabilities matrix (in the discrete case) or the probability distribution vector (in the general case) and π represents the initial state distribution vector. π determines the probability of starting in each of the possible states [48, 49].

There are three key problems associated with HMMs. The answers to these problems allow the use of HMMs in real world applications. These problems are [48]:

Problem 1 - Given a sequence of observations $O = O_1, O_2, \ldots, O_T$ and the model $\lambda = (A, B, \pi)$, how do we compute the probability of the observations sequence $p(O|\lambda)$.

Problem 2 - Given a sequence of observations $O = O_1, O_2, \ldots, O_T$ and the model $\lambda = (A, B, \pi)$, how do we compute the optimal state sequence $I = I_1, I_2, \ldots, I_T$.

Problem 3 - Given a sequence of observations $O = O_1, O_2, \ldots, O_T$, how can we adjust the parameters of the model $\lambda = (A, B, \pi)$ to maximize the probability $p(O|\lambda)$.

The first problem can be seen as an evaluation problem. Given a certain model and a sequence of observations, how can we compute the probability that the sequence was produced by the model. This can also be viewed as way to evaluate the model. This view is interesting if we consider a scenario where several models are competing. If we can solve the first problem, we will find which model is the best match for a certain observation sequence [48].

In the second problem we attempt to undercover the hidden part of the model. This process is usually a typical estimation, where the definition of an optimality criterion to solve the problem is required. There are several optimality criteria that can be used depending of the intended use for the uncovered state sequence. One of the most classic uses of the uncovered state sequence is to learn about the structure of the model, and to get average statistics, behaviour amongst other characteristics within individual states [48].

The third and final problem is an attempt to optimize the model parameters to best fit the observed sequence. This is a training problem, which is crucial in most HMM's applications, since it allows to adapt our model parameters to the observed training data. A good training phase will allow the creation of good models for real applications [48].

Thus, in a real application we will have to start by training the HMMs. A specific training sequence should be used for each different real thing we want to model. The solution to the third problem is used to get the optimal parameters for each model. In order to understand the physical meaning of the model states we used the solution to the second problem. This may lead to further improvements in the model. Finally, using the solution to the first problem we can score each model upon a given test observation sequence and select the model with the highest score, enabling the labelling of a new sequence fed to the system [48]. Understanding how these problems can be solved is necessary in order to efficiently apply HMMs.

In order to solve **Problem 1** we want to calculate the probability of a sequence observation *O*, given the model λ. The probability of any sequence *O* occurring when following a given sequence of states *I* was previously described by equation 2.1. The most obvious solution is enumerating every possible state sequence of length *T* (number of observations) and then for every fixed state sequence I_n calculate the probability $p(O, I_n)$. In other words we marginalize the joint probability over I by summing over all possible variations of I:

$$p(O) = \sum_{I} p(O, I) = \sum_{I} \prod_{t=1}^{T} p(I_t | I_{t-1})\, p(O_t | I_t) \quad , \tag{2.2}$$

where $p(O)$ is the probability of the sequence *O* given the model λ. In order to efficiently solve this problem, the forward-backward procedure is usually used [47-49].

There are a great variety of ways of solving **Problem 2**. Depending on the optimality criteria selected, the method of finding the optimal state sequence associated with a given observation sequence will change. One possible criterion is to choose the states, i_t, which are most likely for each observation of the observation sequence. This will maximize the expected number of correct individual states

$$\gamma_t(i) = p(i_t = q_i \mid O, \lambda) \tag{2.3}$$

where $\gamma_t(i)$ is the probability of being in state q_i at the time *t*, given an observations sequence *O* and the model λ. Using $\gamma_t(i)$, the most likely state, i_t , at the time *t* is

$$i_t = \underset{1 \le i \le N}{\operatorname{argmax}}[\gamma_t(i)] \ , \qquad 1 \le \mathrm{t} \le \mathrm{T} \quad , \tag{2.3}$$

There is a formal technique for finding the single best state sequence. This technique is called the Viterbi decoding algorithm [47-49]. A detailed description of the forward-backward procedure and Viterbi decoding algorithm can be found in [48].

By solving **Problem 3** we want to adjust the model λ parameters (A, B, π) to maximize the probability of the observation sequence being produced by the model. This problem will be a maximum likelihood problem, which usually are solved using iterative procedure, such as the Baum-Welch method, Segmental K-Means algorithm, or gradient techniques [48].

The Baum-Welch algorithm (BWa) is basically an expectation-maximization algorithm. It is guaranteed to converge to at least a local maximum. Its complexity increases as the length of the data and the number of training samples increases since it requires two passes over the data at each step. Although by using this method a full conditional likelihood for the hidden parameters is obtained [50].

The BWa has some interesting properties [50]:

- When working with discrete HMMs, it does not require any model initialization. It only requires some non-zero random values verifying the stochastic constraints;
- When working with continuous HMMs, several options of model initialization are available (i.e. means and variations of the data acquired by vector quantization);
- The algorithm uses all the available data to produce a robust estimate of the parameters.

The Segmental K-Means algorithm (SKMa), also known as Viterbi training algorithm, approximates the solution to the maximum likelihood problem by maximizing the probability of the best HMM state sequence for each training sample [50]. It segments the data and applies the Viterbi algorithm to find the most likely state sequence for each segment (solution to **Problem 2**). Then it uses the most likely state sequence to re-estimate the hidden parameter. This involves much less computational effort then the BWa, but the results tend to be slightly worse [50]. Some of the main issues of using this algorithm are its dependency on the amount of available data and the fact that it doesn’t give the full conditional likelihood of the hidden parameters [50].

Chapter 3

Methodology

From the previous literature review (*Chapter 2*) it is possible to idealize the design of our system. In this chapter we describe the methodologies used in our work, from the system requirements and overview to the extracted information, classification methods and evaluation of the system.

3.1 - System Requirements and Overview

As mentioned in *Chapter 1*, the aim of this dissertation is to develop an approach for a computer aided analysis of the STS movement using the 3D body-joint tracking data acquired with the Kinect platform. This system should be as automatic as possible, giving feedback to the user about the movement and enabling the acquisition, posterior consultation and analysis of the data. The use of the Kinect platform was one of the base requirements of our work, due to its novelty and accessibility. These characteristics leave open the possibility of adaptation of the system to the home rehabilitation field, helping with patients' physiotherapy.

In order to materialize our system, the requirements of the system must be defined beforehand. Since no databases with STS movement depth images compatible with the Kinect exist until the present data, we will need to start by collecting data for tests. When performing this data acquisition it is important to bear in mind the range limitations of Kinect and the parameters affecting the STS movement (*section 2.3.1*), which will influence the final results. These facts lead to the necessity of establishing a room setup in order to acquire data in a controlled environment.

A subdivision of the movement into smaller portions (phases) is also required in order to properly analyse each movement and compare between captured movements and compare with the results previously described in the literature. So, a formal description of the division into phases is required in order to develop the system. This division into phases along with the nonexistence of any useful movement databases lead to the necessity of a pre-segmentation and manual analysis of the movements. This is required not only to implement the system but also to have a ground truth to validate the system.

We want to develop an automatic system that analyses the STS movement. For this, a simple state machine could be used but this would mean that rigid thresholds would constrain our system. Having a flexible system that can be adapted to a certain situation is much more useful than a rigid system. This leads to the idea of using machine learning techniques such as a classifier. A classifier can be trained with a specific dataset and then applied to a new sample, producing an understandable output. Obviously this output could be wrong. An evaluation of the system is required in order to understand its limitations.

Another important requirement of our system is to be able to give feedback to the user and enable the acquisition, analysis, and posterior consultation of the data. This leads to the need of developing an interface in order to give information, preferably in real time, to the user.

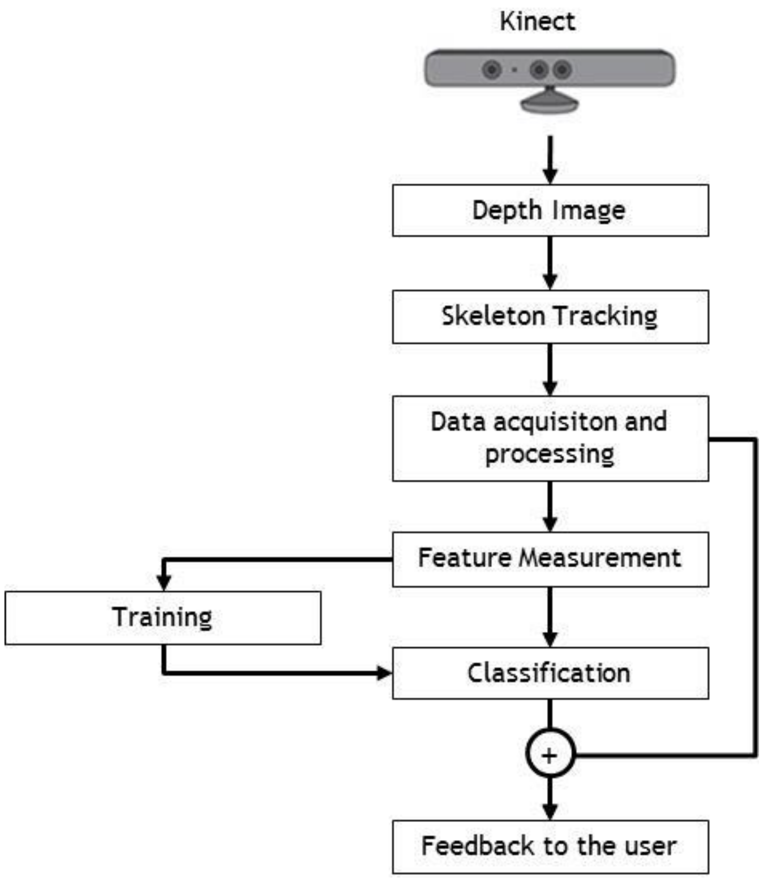


Figure 3.1 - Block diagram of the system.

The block diagram of our system is presented on Figure 3.1. We will start by acquiring depth data. From that depth data, the skeleton data is extracted. This skeleton data will then undergo a processing step, where the data will be filtered and normalized. From here a set of features is selected to train our classifier in a separate step. In the processing step, information is also acquired to be given back to the user by means of an interface.

In the following sections we will discuss each of the previously mentioned steps and the thought process behind the performed decisions.

3.2 - Experimental Setup

In the developed system a sample of 7 young subjects (age: 23 ± 1 years old, height: 166 ± 13 cm, weight: 64 ± 11 kg) without any impairing pathology were asked to perform 5 STS movements each at their normal pace. In order to obtain understandable and comparable data a specific room setup was designed. In our setup the Kinect was placed at a height of 0.80m and approximately 2.90m from the chair with the image plane parallel to the subject's coronal plane, as shown on Figure 3.2.

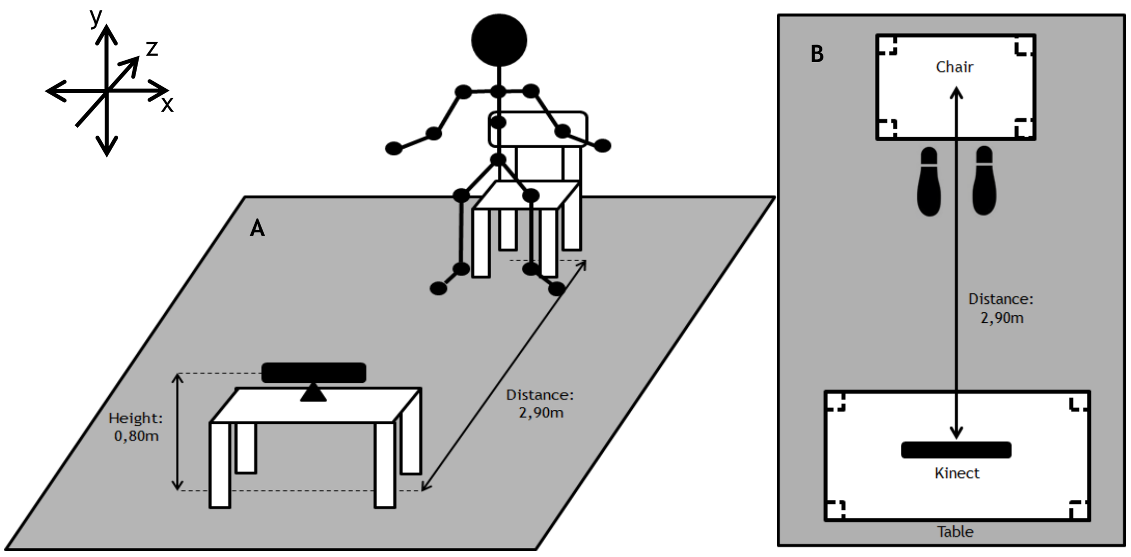


Figure 3.2 – **A -** Representation of the used setup for the tests; **B -** Upper view of the used setup.

As described in *Chapter 2* most methods for analysis of the STS movement use images of the sagittal plane, which hinders the possibility of creating an interactive application where the user can be performing some activity and at the same time seeing how it progresses on the output display screen.

In order to have a similar setup for all the tests, the subjects were asked to start in a standing position, with both feet in a parallel position (Figure 3.2 - **B**) to give time for the Kinect algorithm to fully track the body. From this point on, the subjects were asked to sit in a comfortable position maintaining the feet position, using the backrest of the chair. After waiting a few seconds the subjects start performing the STS movement at their normal pace, reaching the standing position (initial position of the test) and waiting again a few seconds. The subjects were asked to perform this cycle 5 times, in order to obtain 5 movements from each subject, at their normal movement pace.

Armrests were not used in any of the tests due to the fact that the data acquired from the elbow, wrists and hand joints (Figure 2.7) was unstable. This was particularly noticeable when observing the behaviour of the hand joints, which were most of the times inferred, instead of tracked. By inferred we mean that the algorithm's confidence in estimating the position of the joint did not meet a minimum required threshold. This may happen when the joint is being hidden from the view (for example by another body part) or is outside the camera's field of view. Sometimes it may also result from a high level of ambiguity in the depth data, which leads to inability of the algorithm to choose between two or more possible joint positions [15]. On the other hand, when a body joint is tracked it means that the estimation of the joint position is properly done. For more information on how the estimations are performed [15] can be consulted.

Since the subjects did not use armrests, the force moment needed at the knees will increase. This does not influence the motion range of the joints [2, 40]. Although, this increase in the moment at the knees shall not have a great impact on our study, since all the subjects were young and healthy. This increase in the moment becomes a factor with greater influence as the age of the subjects increases [2, 40].

The subjects were asked to keep the same feet positioning over all the tests, according to the setup seen on Figure 3.2. This was required in order to be able to acquire consistent results for subject to subject comparisons and for comparisons between the 5 movements of a specific subject. Also, by avoiding the repositioning of the feet we avoid the use of a rising strategy that facilitates the performance of the STS movement, conditioning the obtained results [2, 11, 38-40].

The same chair was used for all the tests. This chair was fixed at a distance of approximately 2.90m in front of the Kinect. The height of the chair was fixed at 50cm. The height of the chair will mainly influence the need to generate momentum – lower chair seats will require more generation of momentum [2, 11, 38-40]. Once again this factor should not influence our results, since the population of our study is young, diminishing the impact of needing to generate more momentum.

3.3 - Phases Definition

With our experimental setup well defined, we can proceed to the definition of what we are going to analyse. From the literature review (*Chapter 2*) we can see that depending on the scope of the study different definitions of the movement phases can be used.

First of all, we should consider that with an increased number of defined phases an increase in the complexity of the system will be noticed. Also, defining a great number of phases will lead to stricter and shorter phases, which will be harder to correctly identify and segment.

Ideally we want to define a number of phases that allows us to get as much information about the movement as possible without losing any important events of the STS movement. By events we consider moments that give us important information to characterize the movement, such as the ones defined by Schenkman et al. [8] – lift off, maximum dorsiflexion, end of hip extension – or the ones defined by Goffredo et al. [9] – trunk flexion, knee extension and trunk extension. Besides these moments, the most important ones are the start and the end of the movement. Without these our work could not be completed.

A good segmentation of the movement will lead to a better characterization, and to a certain extent, to a better evaluation of its performance. Also, having ground truth to compare our results with is important to validate our system. In order to obtain results comparable with the ones described in the literature, a similar approach to the phase definitions can be performed. Of course some adaptations are in order to be done, since the data acquisition methods vary from work to work.

As mentioned in *Chapter 2*, a subdivision into phases resulting from the combination of the definitions of Schenkman et al. [8] and Goffredo et al. [9] was used in this work. Figure 3.3 shows a representation of the phases defined for our work.

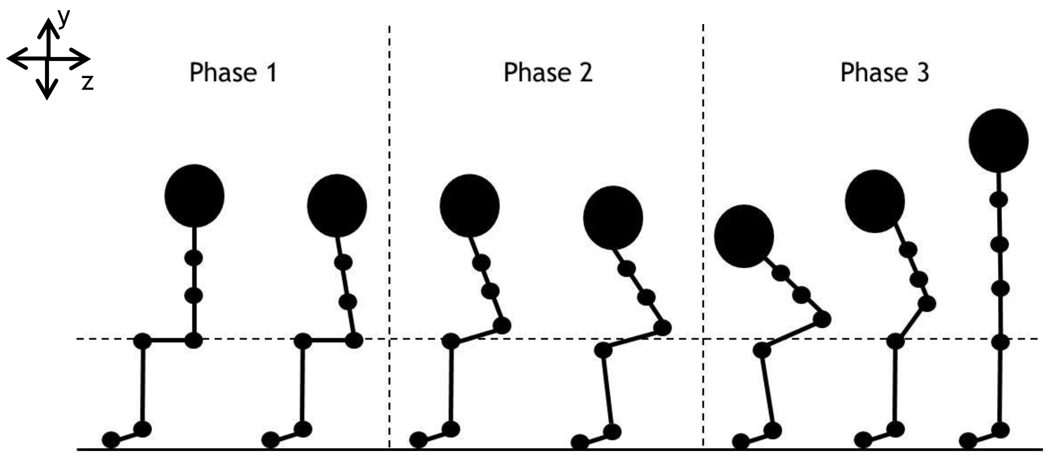


Figure 3.3 – Sagittal representation of the STS movement division into phases defined in the scope of this work.

We defined that phase 1 starts with trunk flexion, as described in [8, 9]. This means that the movement will start when the upper body joints, especially the shoulder and head joints, start moving forward (in the direction of the Kinect). Phase 1 ends when the buttocks are lifted and so phase 2 starts. This is the same as described by Schenkman et al. [8], but different from what Goffredo et al. [9].

In [9] the authors described phase 1 ending at the beginning of the knee extension, detecting this transition when the ankle angle decreased at 95% of its maximum value. This was unviable to implement in our system since the foot joints are just as unstable as the hand joints (previously discussed in *section 3.2*). Also, a decrease of 5% from the maximum value, which by default should be approximately 90º (considering a simplification used in [9]), corresponds to a decrease of 4.5º. Relying on the detection of this kind of value variation, considering the limitations of the data acquired with the Kinect (*section 2.2.3*), was not the best option.

Phase 2 ends just when the maximum dorsiflexion is reached, starting phase 3. This is defined as the temporal location of the minimum trunk angle with the ground. This phase transition is similar to the one defined in both [8, 9].

Finally, the movement is deemed completed when the body reaches its full extension and all the joints are stable (present constant values). Schenkman et al. [8] included an additional phase for the stabilization of the body. As a matter of simplicity we decided to use three phases besides the standing and sitting phases, including the stabilization of the body in our last phase. The sitting phase comes before phase 1 and the standing phase comes after phase 3. These phases correspond to the moments when the joint data is constant over a certain period of time.

3.4 - Movement Segmentation

3.4.1 - Initial Segmentation

After defining how we will subdivide our movement for analysis, an initial segmentation of the movement was performed. The objective of this segmentation is to have a preliminary division. This will give us an idea of the size of the data we will be working with, the duration of each phase when using a real application and which phases are harder to detect. Also, if we correctly segment at least the beginning and end of the movement, we will be able to obtain initial data and from this point manually segment it.

For this we decided to do an automatic segmentation of the movement using simple thresholds. This led to some sort of a “state machine” with 5 states: sitting, phase 1, phase 2, phase 3 and standing. The phases were segmented and detected according to what was defined in *section 3.3*. Phase 1 started when a continuous forward (in the direction of the Kinect) movement of the trunk was detected. Phase 2 started when an elevation of the hips was detected. Finally phase 3 was detected when, after the minimum value of the trunk angle with the ground was detected, a continuous increase of this angle was detected.

On the other hand, the sitting and standing positions were detected by means of the system receiving constant values, with height verification and a sequence check. By height verification we mean that the system required the user to introduce and approximate height of the test subject in order to estimate if the subject was sitting or standing. Also, by sequence check we mean that the whole system depended on the previous states in order to properly work. For example, phase 2 could never be detected if phase 1 was not previously detected. The same applied to the relation between phase 3 and phase 2. For the sitting and standing states, the only required verifications were the relation between the measured height and the height inputted into the system. The standing state could only be detected if a sitting state happened previously (not the immediately previous state, but in the current movement in analysis). The same applied to the sitting state, but in this case the sitting state will always follow the standing state, marking the beginning of a new analysis.

It is noticeable that this kind of segmentation has flaws. For example, if one phase wasn't detected, the following phases until the standing position wouldn't be detected either. Also, if a subject started a movement and stopped in the middle, reverting to the sitting position, the system would fail to correctly detect the phase until the last correctly detected phase was resumed. Another problem was that this kind of system was implemented using rigid thresholds. The STS movement characteristics vary from subject to subject. The only phases that could always be detected were the sitting and standing phases.

Even with the mentioned flaws, this initial segmentation was useful in order to at least segment the movements, detecting when subject was sitting or standing, correctly detecting some of the intermediate phases. With the acquired information from this segmentation, the work of manual segmentation and validation of the phases was easier.

3.4.2 - Manual segmentation

After the initial segmentation, a manual segmentation of the movements was performed. The objective of this segmentation was to create the ground truth dataset, in order to train and test our classifier. This was necessary since there wasn't any usable database available with the movements we wanted.

As mentioned before, 5 movements per subject were collected, to a total of 35 movements. These movements were pre-segmented using the methodology described in *section 3.4.1* in order to obtain an initial rough estimation of the phases. After this, a manual evaluation and analysis of the movements was performed, verifying each movement frame-by-frame, detecting the beginning of each phase. In the first tests, a webcam was used to capture the movement in the sagittal plane in order to verify the segmentation method.

The key moments searched for the segmentation were the beginning of the trunk flexion for phase 1, the elevation to the buttocks for the beginning of phase 2 and the minimum trunk angle with the ground, followed by the inversion of the trunk movement for phase 3. All the frames between the beginning of phase 1 and phase 2 were labelled as part of phase 1. The ones between the beginning of phase 2 and phase 3 were labelled as part of phase 2 and the ones between the beginning of phase 3 and the standing position (until stabilization of the joints was completed) were labelled as part of phase 3. Some frames preceding the phase 1 frames were labelled as part of the sitting phase and some frames after the end of phase 3 were labelled as part of the standing phase.

After the manual segmentation and definition of the features to extract from the movements, a re-evaluation of the segmentation was performed. This re-evaluation had the objective of confirming that the previously defined phases were correct. We took into consideration the most important information that defined each phase transition in order to perform this process. The features used in this process will be further discussed in *section 3.5*.

Summing up, the movements were manually analysed and segmented into phases, frame-by-frame, to a total of 3283 analysed and labelled frames. The manually labelled frames were later used as ground truth in order to evaluate the system.

3.5 - Features

Once we got the definition of our phases we are ready to start thinking about the data we want to collect and process, in order to train our classifier and extract information from the movements. In our work we collected the data frame-by-frame, each frame corresponding to a certain phase of the movement (sitting, phase 1, phase 2, phase 3 or standing).

This stage of the work aims to obtain a set of suitable features which best portraits the characteristics of each phase in order to provide the classifier with useful and meaningful information. But, first, some considerations about the data acquired with the Kinect should be done. These considerations, along with the feature selection, will be made in the following subsections.

3.5.1 - Filtering process

As described in *Chapter 2*, it is noticeable that there are some limitations to the data acquired with the Kinect. One of the biggest limitations is the instability shown by the skeleton detection system. As an example we can consider the hand and foot joints. These joints show high instability, being inferred great part of the detection time. Also, some jittering and noise could be noticed from time to time when detecting the joints.

In order to help solving these problems the Kinect SDK offers a group of joint filtering mechanisms. These filters allow the skeletal tracking joint information to be adjusted across different frames in order to minimize jittering and stabilize the joint positions over time [29, 31, 51]. The smoothing filter provided with Kinect SDK is based on the Hold Double Exponential Smoothing method. This method is usually used for statistical analysis of economic data, providing smoothing with less latency than other filtering algorithms [51].

The filter can be controlled via five smoothing parameters: *Smoothing*, *Correction*, *Prediction*, *JitterRadius* and *MaxDeviationRadius* [51]:

- The *Smoothing* parameter controls responsiveness to the raw data. Increasing this parameter will lead the system returning more highly smoothed skeleton position values but this will also lead to an increase of the latency of the system.
- The *Correction* parameter controls how the data will be smoothed. This parameter is related to the speed of the smoothing process.
- The *Prediction* parameter controls the number of frames to predict into the future. High values will lead to overshooting when moving quickly.
- The *JitterRadius* parameter controls the radius (in meters) of the jitter reduction. Any jitter beyond that value will be clamped to the radius.
- The *MaxDeviationRadius* parameter controls the maximum radius (in meters) that the filtered positions are allowed to deviate from the raw data. Once again, values that exceed this limit are clamped at this distance, in the direction of the filtered value.

In order to properly use this filter a balance between these parameters has to be found. Detailed information about this and other filtering methods used with Kinect applications can be found in the Skeletal Joint Smoothing White Paper [51].

In our work we decided to use one of the preconfigured set of parameters which is described in the literature to do some smoothing with little latency, only filtering our small jitters, being ideal for gesture recognition tasks [31, 51].

The parameters of the filter were:

- Smoothing = 0.5;
- Correction = 0.5;
- Prediction = 0.5;
- JitterRadius = 0.05;
- MaxDeviationRadius = 0.04;

This filter has a medium smoothing, correction and prediction values. Any jittering with a radius bigger then 5cm will be clamped to 5cm and the filtered positions are allowed to distance a maximum of 4cm (radius) from the raw data. What is really important is to have a low latency while being able to remove some of the jittering. This has high relevance since we want our final application to work in real time.

Also in order to reduce the effect of small value variations on our system we decided to only consider variations of the joint positions bigger or equal to 1cm, even though the variations acquired with the Kinect can be in the magnitude of millimetres. From preliminary data acquisition and taking into consideration the study of K. LaBelle [33] it was concluded that the acquired values tended to vary even in a steady position. By using a measurement magnitude ten times bigger than the smallest value captured we try to remove some of the value variations, acquiring more significant data for our work.

3.5.2 - Feature selection

As mentioned before, this stage of the work aims to obtain a set of suitable features which best portraits each phase characteristics in order to provide the classifier with useful and meaningful information. A good set of features will lead to a good classification step.

In our work we want to distinguish between 5 different classes: sitting, phase 1, phase 2, phase 3 and standing. The transition between phases is one of the most important aspects of our work. A good subdivision of the movement will lead to good results that can be compared to what is described in the literature using different analysis methods.

We decided to use features that derive directly from the data acquired with the Kinect, which was previously filtered, as described in *section 3.5.1*. By using the most basic acquired data, we will reduce the errors originated from value approximations and estimation (like integrations and derivatives). This could also be a limiting factor, but usually keeping a system simple is more beneficial than increasing its complexity and increasing the sources of error.

All the extracted features are based on the relative position of the joints. We decided to follow this approach since the definition of the phases (*section 3.3*) can be interpreted as an evolution of the joint positions over time. Although the whole body is involved in the movement, some joints give us more information than others. For example, the shoulder joints play a major role in the whole movement. While analysing the shoulder joints positions over time we can know if the subject is bending the trunk forward, or if he is standing still. The extracted features follow the same kind of rationale.

The Kinect collects the 3D (X, Y and Z) coordinates of the 20 joints at 30 frames/s. Although, only some of these joints will give useful information: head (*h*), centre of the shoulder (cs), left shoulder (*ls*), right shoulder (*rs*), spine (*s*), hip centre (*hp*), left hip (*lh*), right hip (*rh*), left knee (*lk*), right (*rk*), left ankle (*la*), right ankle(*ra*), left foot (*lf*) and right foot (*rf*). We regard the positions of the hand joints as less important since these joints tend to be unstable. We also don't consider the wrists and elbow positions have less relevance in the scope of this work, since we want to characterize the movement without using armrests, so the arms in general won't play a major role. The features we want to extract must give us information that can separate the different classes used in this work: sitting, phase 1, phase 2, phase 3 and standing.

From the group of 14 selected joints we define a 13-dimensional feature vector (for each frame) based on skeleton presented on Figure 3.4.

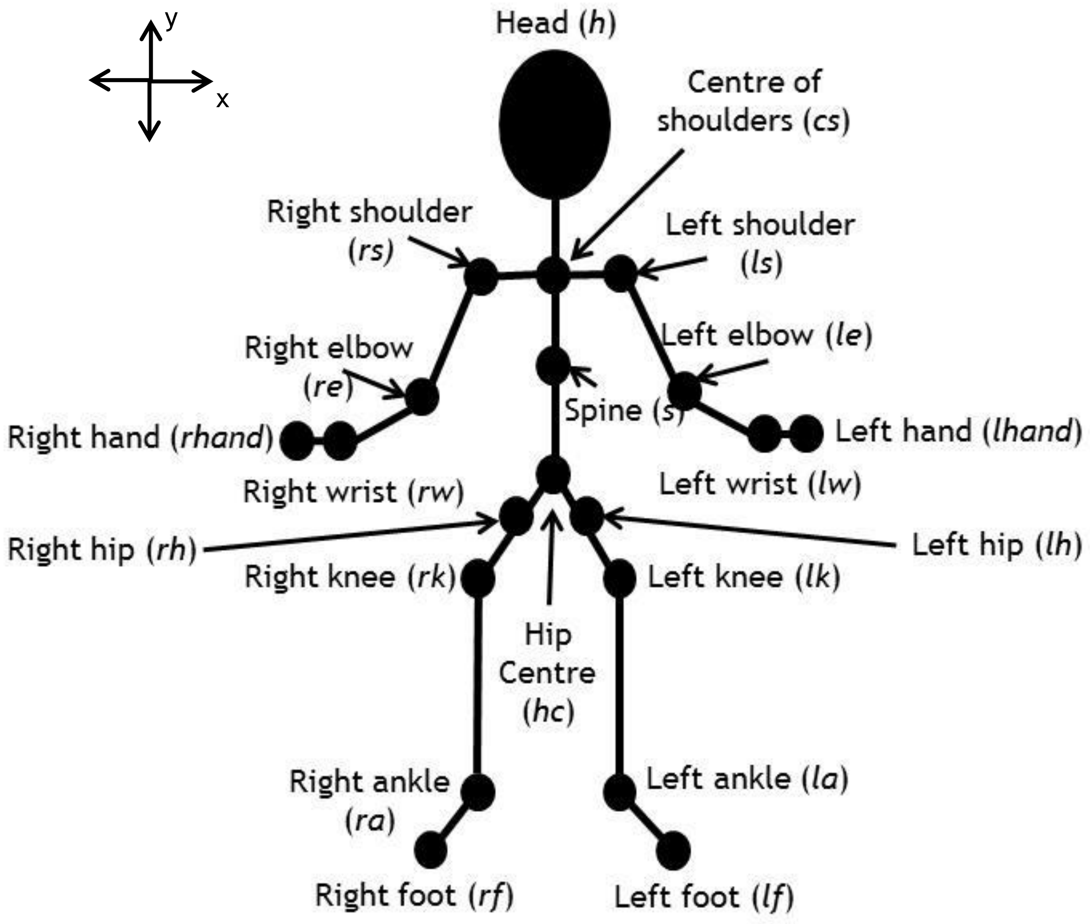


Figure 3.4 - Skeleton model with 20 joints;

The sitting phase is characterized by constant values of all the features over time. Also, the upper body depth values should be higher than lower body ones. This can be verified by comparing the depth of the shoulder, hip and knee joints. Also, if we are in stable position, the depth distance between the knee joints and the ankle joints should be constant.

The standing phase has some characteristics similar to the sitting phase. It can also be characterized by constant values. But in this case, all the joints should be aligned in the vertical plane orthogonal to the Z axis (in theory). When comparing the shoulder, hip and knee joints depth values, they should all be similar.

Also, we can introduce a feature that estimates the height of the subject. These features will have a lower value when the subject is sitting and will have a higher value when the subject is standing.

In order to detect phase 1 we need a feature that tells us when the trunk is moving, and in which direction. This can be obtained comparing the variation of centre of shoulders height value with the relation between the spine and centre of shoulders joints depth. If we have decrease of the *cs* joint height value, this means that we are leaning forwards. Also, an increase of this value can be used to define the beginning of phase 3, since it starts when the trunk angle with the ground reaches its minimum value and inverts the direction of the movement, starting the extension period. If the depth value of the spine join is increasingly bigger than the *cs* joint depth, it means that the trunk flexion has started. Phase 2 starts when we lift of the buttocks. This can be studied by analysing the variation of the height value of the hip joints over time. Table 3.1 contains the mathematical definition of the relations previously described. We also decided to consider a feature that compares the position of the shoulders. When performing the STS movement the shoulders should always be in the same plane (have similar depth values).

Note that the skeleton model will vary from person to person depending on the person's height, legs length, distance and initial position of the Kinect. Each variation will have a different impact on our measurements. Therefore we need a way to minimize these impacts. A normalization method will be described in the next subsection.

Table 3.1 - Mathematical definition of the features used in our work.

Feature	Definiton
Shoulder Relative depth position	$d^{Z}_{rs-ls} = Z_{sr} - Z_{sl}$
Left hip height variation	$\Delta Y_{lh} = Y_{lh} - oldY_{lh}$
Right hip height variation	$\Delta Y_{rh} = Y_{rh} - oldY_{rh}$
Hip centre height variation	$\Delta Y_{hc} = Y_{hc} - oldY_{hc}$
Relative depth distance between *rs* and *rk*	$d^{Z}_{rs-rk} = Z_{rs} - Z_{rk}$
Relative depth distance between *ls* and *lk*	$d^{Z}_{ls-lk} = Z_{ls} - Z_{lk}$
Relative depth distance between *rh* and *rk*	$d^{Z}_{rh-rk} = Z_{rh} - Z_{rk}$
Relative depth distance between *lh* and *lk*	$d^{Z}_{lh-lk} = Z_{lh} - Z_{lk}$
Relative depth distance between *rk* and *ra*	$d^{Z}_{rk-ra} = Z_{rk} - Z_{ra}$
Relative depth distance between *lk* and *la*	$d^{Z}_{lk-la} = Z_{lk} - Z_{la}$
Height estimation	$height = Y_h - \left(\frac{Y_{rf} + Y_{rf}}{2}\right)$
Shoulder centre height variation	$\Delta Y_{cs} = Y_{cs} - oldY_{cs}$
Relative depth between *cs* and *s*	$\Delta Z_{cs} = Z_{cs} - Z_s$

These features are all organized in a feature vector as follows:

$$\mathrm{f}_n = [d^{Z}_{rs-ls}, \Delta Y_{lh}, \ \Delta Y_{rh}, \Delta Y_{hc}, d^{Z}_{rs-rk}, d^{Z}_{ls-lk}, d^{Z}_{rh-rk}, d^{Z}_{lh-lk}, d^{Z}_{rk-ra}, d^{Z}_{lk-la}, height, \Delta Y_{cs}, \Delta Z_{cs}]$$

3.5.3 -Feature normalization

Data mining is the process of finding patterns and valid unrecognized associations between data. Processes of data transformation, such as normalization may improve the accuracy and efficiency of classification algorithms. Normalization is particularly useful since it helps preventing features with initial larger ranges from outweighing features with smaller ranges [42, 52]. This is of particular importance in our work since the data directly acquired with the Kinect has a big range of values [28, 29, 31]. There are some data normalization methods such as Min-Max normalization, Z-score normalization and normalization by decimal scaling [42, 52].

Min-Max normalization is a linear transformation of the data, according to

$$v' = \frac{v - \min(v)}{\max(v) - \min(v)} \times (new\max(v) - new\min(v)) + new\min(v), \tag{3.1}$$

where v is the original value, v' is the normalized value, $\min(v)$ and $\max(v)$ are the minimum and maximum values of v, $new\max(v)$ and $new\min(v)$ are the new range of values. The Min-Max normalization will map the value v in the range $[\max(v), \min(v)]$ to v' in the range $[new\max(v), new\min(v)]$.

In this work we decided to use the Min-Max normalization in order to obtain comparable results between subjects. Values of depth and height that could vary from subject to subject are mapped to a [-1,1] interval. For example a variation of the hip, shoulder or head joints height, that for a taller subject would be much greater than for a shorter subject, are rescaled.

This normalization is of higher importance when we consider the difference in the scales of the acquired values. Depending on the type of coordinates we acquire (X, Y or Z) a different scale is used. Depending on the relative position between the Kinect and the subject, the range of acquired values will change. This fact was already mentioned in *section 2.2.1,* but it is important to remember. A single reposition of the Kinect 1cm to the left or to the right, 1cm higher or lower, or rotated 1° to the left of the right from one test to another will cause the range of acquired values to change. Also, while the range of values of X and Y can go from negative to positive values, the values of Z (depth) will always be positive. This would cause instability of the data when used in a classification process.

With the normalization we balance the values considering the maximum and minimum values acquired. This is performed for each axis independently. This means that we performed the Min-Max normalization for the X, Y and Z values separately, considering different maximum and minimum values.

3.5.4 - Synthetic Minority Over-sampling Technique (SMOTE)

Sometimes when dealing with datasets we see that some classes are much more represented than others. Using an imbalanced dataset may have a negative impact on the performance of a classifier. The over-represented classes tend to overwhelm the under-represented classes, leading to incorrect classifications. Ideally we want to work with a dataset where all classes are equally represented [53].

This issue is usually addressed in one of two ways. One is to assign distinct costs to the training examples [53, 54]. The other way is to re-sample the original dataset, either by oversampling the least represented classes and/or undersampling the most represented classes [53, 55].

When working with HMMs, controlling the costs of the training examples will be impossible to apply, due to the limitations of the library used in this work, which will be discussed in *Chapter 4*. This leaves us with the approach of re-sampling the least represented class.

For this approach, there are some possible ways to re-sample our dataset. We can simply over-sample the least represented class by creating copies of the samples that already exist. At the same time these new samples will replace samples from the most represented class, doing an under-sampling of that class. Although, this kind of approach doesn't significantly improve the recognition of the least represented class [56]. This is probably due to the fact that we are over-sampling by increasing amounts. This will lead to a similar identification of the class but in a more specific way, since the characteristics of the class become much stricter. We have more samples, but with similar characteristics, since they are copies of the original ones [53].

The idea behind the Synthetic Minority Over-sampling Technique (SMOTE) is to perform and over-sampling of the least represented class by creating "synthetic examples", rather than by over-sampling and replacing [53].

In the SMOTE technique the least represented class is over-sampled by taking each sample of this class and introducing new synthetic examples along the line segments joining all of the *k* least represented nearest neighbours. Depending on the amount of over-sampling required, we randomly chose neighbours from these *k* nearest neighbours. The synthetic samples are generated by taking into consideration the difference between the feature vector (sample) and its nearest neighbour. This difference is multiplied by a random number between 0 and 1 and added to the used feature vector. This will cause the selection of a random point along the line segment between two specific features, forcing the decision region of the least represented class to become more general, while increasing the number of samples in this class. A detailed description of this method can be found in [53].

In our work, the least represented classes were the classes corresponding to phase 1 with 399 and to phase 2 with 108 samples in a total of 3283 acquired samples. Ideally we want to have each class representing approximately 20% of the dataset (approximately 657 samples of each class) since we have 5 classes. In order to balance the dataset we used the SMOTE technique to oversample these classes. A new dataset with 4176 samples was obtained. Phase 1 and phase 2 classes were represented by 700 samples each.

3.6 - Information Extraction

As mentioned previously in *Chapter 2* there are several different approaches to the analysis of the STS movement. Depending on the selected approach and methodology used, different information can be extracted from the movements. The basic information we can acquire with the Kinect are the coordinates (X, Y and Z) of the 20 joints and the timestamp of the acquired frames. With this basic information we can compute some kinematic data. Angles, angular velocities and velocities are some examples of the data we can obtain when processing the basic data. The duration of each phase and the total duration of the movement are also important to the characterization of the movement.

Goffredo et al. [9] compared the angles information they obtained at the lift-off moment (our phase 2 beginning) to some results described in the bibliography. So, obtaining the joint angles at the lift-off time will give us an opportunity to compare our results with the ones described in the literature. This will be further explored in *Chapter 5* when discussing the results.

In order to calculate the previously mentioned angles we used the scalar product of two vectors. These vectors were created by 3 joints, having one joint in common. For example, we can calculate the left knee angle by creating 2 vectors. The first vector, *a*, is created by the connection of the left hip joint and the left knee joint and the second vector, *b*, is created by the connection of the left knee joint and the left ankle angle. This is represented in Figure 3.5.

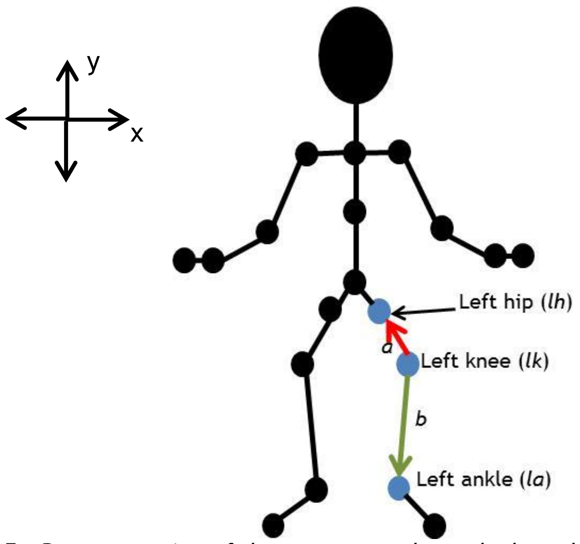


Figure 3.5 - Representation of the vectors used to calculate the angle of the left knee.

In red we can see the vector *a* and in green we can see the vector *b*. Since we are collecting the data with the Kinect, we automatically have access to the coordinates of the extremities of the vectors, corresponding to the joints (represented in blue).

Then, we multiply the components of both vector along the 3 axes (X, Y and Z). We add the three multiplication products together obtaining the scalar product of the two vectors. In order to calculate the angle we just need to obtain the magnitude of the vectors, since

$$\cos(\theta) = \frac{a.b}{\|a\| \times \|b\|} \quad , \tag{3.2}$$

where θ is the angle formed by the two vectors a and b, $\|a\|$ is the magnitude of the vector a and $\|b\|$ is the magnitude of the vector b. Once we have the scalar product and the magnitude of the vectors, we only need to calculate the inverse cosine of the result obtained from equation 3.2 to obtain the desired angle.

The trunk angle with the ground and the ankle angle with the ground are simplifications where instead of using a second vector constructed with the joints, we consider a fixed predefined vector.

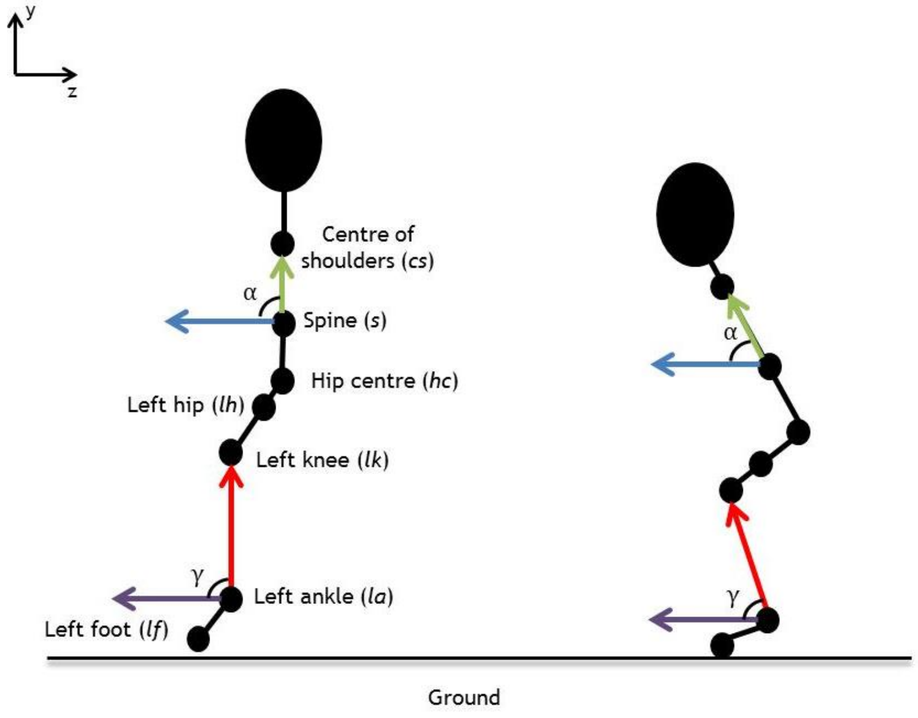


Figure 3.6 - Representation of the trunk and ankle angles with the ground.

On Figure 3.6 we can see that instead of considering a third joint to create the second vector, we used fixed vectors that are in a plane parallel to the ground plane. Another characteristic of these vectors is that they are contained in a plane orthogonal to the x-axis.

Once we have calculated the angles at each frame, the angular velocity can be calculated dividing the variation of the angle between frames ($\Delta\theta = new\theta - old\theta$) by the time between frames (around 33ms). The same principle can be applied to calculate the velocity of the COM, but instead of considering the angle variation between frames, we consider the coordinates variation between frames ($\Delta(X,Y,Z) = new(X,Y,Z) - old(X,Y,Z)$). In our work we consider that the COM is coincident with the spine joint, which will simplify the process of computing its velocity. According to [28], the COM is roughly in the same position as the spine joint. The spine joint is usually used as a fast way to determine the user's position, having an application very similar to the COM, due to its stability and location over time.

Interesting information that can also be extracted from the movements is the sagittal plane view of the movement, using the information acquired with the Kinect. With this kind of information we can compare the evolution of our movement with the literature, which usually analyses the movement using information extracted from the sagittal plane [1, 6, 8, 9].

Table 3.2 summarizes the information acquired in this work.

Table 3.2 – Summary of the information acquired during the movements.

Kinematic information
Trunk angle with the ground
Knee angles (left and right knees)
Ankle angles (left and right ankles)
Ankle angles with the ground (left and right ankles)
Trunk angular velocity
Knee angular velocity (left and right knees)
Ankle angular velocity (left and right ankles)
Simplified ankle angular velocity (left and right ankles)
COM velocity (x, y and z components and magnitude)
Joint angles at lift-off (knee, ankle and ankle with the ground angles)
Other information
Duration of phases 1, 2 and 3
Duration of the movement
Sagittal view of the movement

3.7 - Classification Method

In *Chapter 2* we described what a HMM is, what it does and how it can be created and adjusted to solve a certain problem. The problem that arises now is what happens if we have many kinds of sequences, and we would like to differentiate between them. In our work we want to differentiate between 5 different phases of the movement. We want to be able to introduce a new sequence "unknown" to the system and obtain the respective class which best models the behaviour of the new sequence.

This can be performed by creating a system where we have several models, each one representing a different kind of sequence (a different class). The idea is to have at least 5 models (one for each phase: sitting, phase 1, phase 2, phase 3 and standing) able to give us the different likelihoods for the sequences that we want to label.

The likelihoods can be compared among them. For example if we choose the model with the higher likelihood to be our labelling model for a certain sequence, we will have a maximum likelihood (ML) classifier.

However, the likelihood of an observation given a class model is different from the likelihood of the class being of the sequence given. This is of particular importance when considering cases of unbalanced proportion of classes or when we have few training observations. These problems were already solved by oversampling the least represented classes, balancing the classes.

In our work we will take the likelihood of the introduced sequence as if it was the probability of the class given the sequence. The ML rule can be stated as:

$$outclass = \underset{\omega_j \in \Omega}{argmax}[p(O \mid \omega_j)] \quad , \qquad (3.2)$$

meaning that the class ω_j, from the universe of possible classes Ω, is chosen as the estimated label ($outclass$) for the sequence if it the class which results in the maximum probability output considering equation 3.2. We have to take into consideration that there are as many trained HMMs as classes ω_j since we train each model solely with sequences from the class ω_j. In Figure 3.7 we can see a schematic representation of the previously described process.

Each inner model represented in Figure 3.7 will have one hidden state, since we are training each inner model individually for a specific class. This will work as a state machine that is trained with a certain dataset. We introduce a new observation vector and the system will give us the estimated class by comparing the obtained likelihoods. This system can be generally seen as a 5 hidden state model. This approach is different from what we would use if we wanted to train different models with full movements to compete with each other for the identification of the movement (i.e. to train models to distinguish between a fast movement and a slow movement, or to distinguish between a movement using armrests or not). In that case we could use a single model with several hidden states and find the hidden sequence obtained when feeding the system with a new full movement.

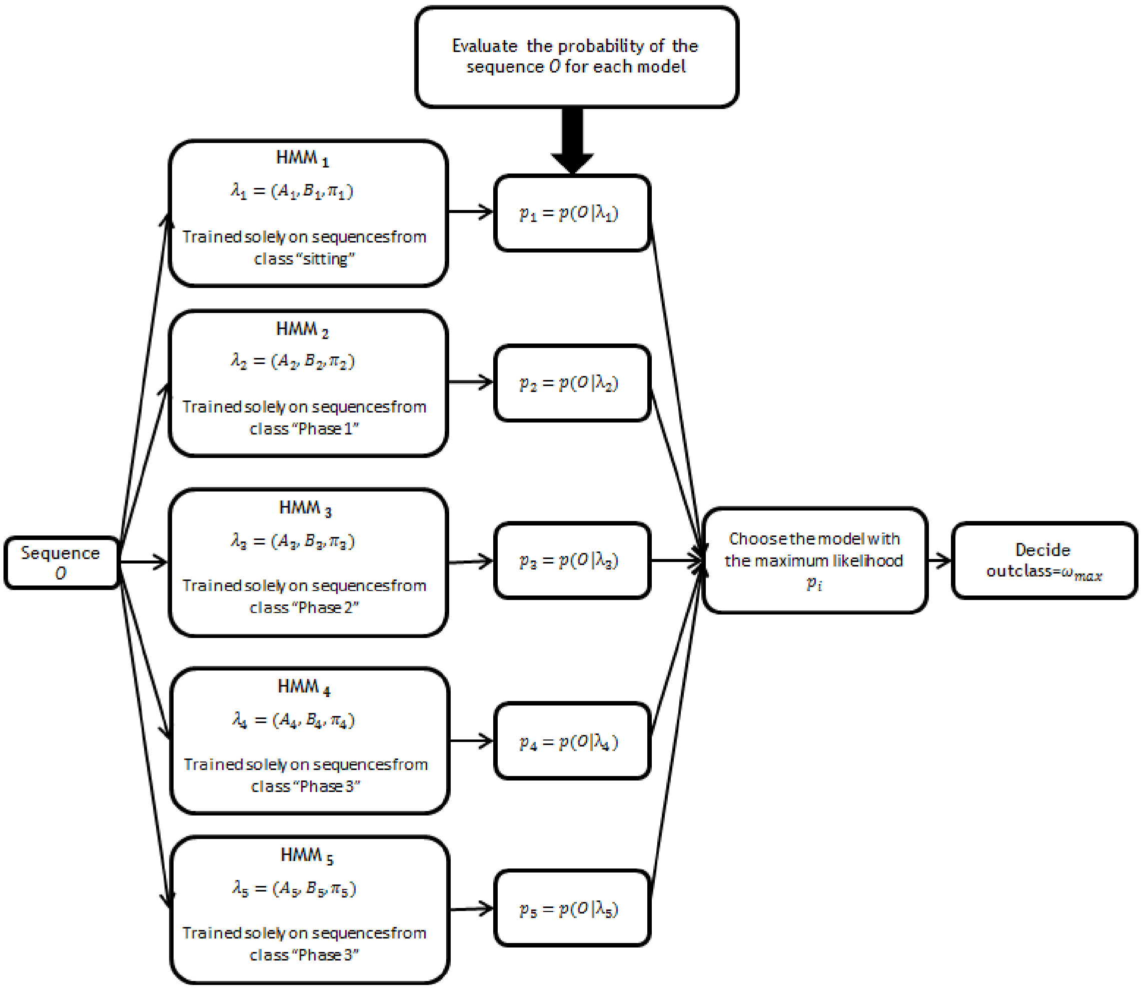

Evaluate the probability of the sequence O for each model
HMM 1
$\lambda_1 = (A_1, B_1, \pi_1)$
Trained solely on sequences from class "sitting"
HMM 2
$\lambda_2 = (A_2, B_2, \pi_2)$
Trained solely on sequences from class "Phase 1"
HMM 3
$\lambda_3 = (A_3, B_3, \pi_3)$
Trained solely on sequences from class "Phase 2"
HMM 4
$\lambda_4 = (A_4, B_4, \pi_4)$
Trained solely on sequences from class "Phase 3"
HMM 5
$\lambda_5 = (A_5, B_5, \pi_5)$
Trained solely on sequences from class "Phase 3"
$p_1 = p(O|\lambda_1)$
$p_2 = p(O|\lambda_2)$
$p_3 = p(O|\lambda_3)$
$p_4 = p(O|\lambda_4)$
$p_5 = p(O|\lambda_5)$
Sequence O
Choose the model with the maximum likelihood p_i
Decide outclass=ω_{max}

Figure 3.7 - Schematic representation of the decision rule used in our system to decide the class of a new sequence.

Similarly to the work of Lin et al. [44] where human motion was analysed, we will use a multivariate Gaussian distribution as our initial distribution. This type of distribution is suitable to describe several types of human behaviours [44]. In our case we will use a generalization of the one-dimensional normal distribution to 13 dimensions, corresponding to the number of features we are using. Our distribution is created based on our training dataset to be suitable to our data.

In order to train our classifier a new balanced dataset was created (section 3.5.4). Two different learning algorithms were used in this work in order to compare the results. The algorithms used were the Baum-Welch algorithm (BWa) and the Segmental K-Means algorithm (SKMa).

3.7.1 -Validation

In order to verify the effect of the class balancing process, 2 different training and testing phases were performed.

In the first one we used the original dataset with 3283 samples, each sample containing a 13-dimension vector with our features of interest. This dataset was divided into 5 groups, each one containing 7 movements (one from each test subject). In order to train the classifier 4 of the 5 groups from the original dataset were used each time. The classifier was tested on the remaining group. This was performed 5 times in order to obtain classification results for all the samples.

On another training phase, we used the oversampled dataset (4176 balanced samples) and the original dataset. Both datasets were divided into 5 groups, each containing 7 movements, one from each subject. In order to train the classifier 4 of the 5 groups from the training dataset were used each time. After training, instead of testing the classifier with the remaining group from the oversampled dataset, which contained synthetic samples, we decided to train the classifier with the equivalent group of the original dataset. This group contained the movements that weren't used for training before using the SMOTE method. We opted for this approach to avoid the classifiers with the synthetic data, since we had no temporal information for these samples. Using the synthetic data for testing would not let us properly observe the behaviour of the classifier on the phase transitions, making its evaluation difficult. This was also performed a total of 5 times, training with 5 different combinations of groups and tested on the group from the original dataset that was left out.

Both methods were similar to a 5-fold cross-validation process, but in our case we controlled the training and testing groups so we would always have 7 movements "never seen" by the system for testing, while using the rest for training.

The results obtained with the classifier were then compared to the original labels, using these as ground truth. The results from the different methods were also compared.

3.8 - Evaluation of the System

Machine learning divides classification into binary, multi-class, multi-labelled and hierarchical tasks [57]. After the development of the system, performance measures are required in order to evaluate the system. Depending on the problem in hands different evaluation metrics should be used [57]. In this case we will be dealing with a multi-class task.

This system was evaluated in terms of precision and recall. Precision is the fraction of detections that are relevant. It is calculated dividing the number of correctly classified positive examples by the number of examples labelled by the system as positive [57]

$$precision = P = \frac{TP}{TP + FP} \quad , \tag{3.3}$$

Recall measures how often an algorithm reports the STS movement as correctly performed in the instances where it actually is correctly done. It is obtained dividing the number of correctly classified positive examples by the number of positive examples in the data [57]

$$recall = R = \frac{TP}{TP + FN} \quad , \tag{3.4}$$

In this context,

- TP denotes the number of true positives. TP is the number of correctly recognized class samples;
- TN denotes the number of true negatives. TN is the number correctly recognized samples that do not belong to a class;
- FP denotes the number of false positives. FP is the number of samples incorrectly assigned to a class;
- FN denotes the number of false negatives. FN is the number of samples that were not recognized as class examples;

In this work we are working with a multi-class classifier. So each evaluation metric has to be computed for individually for each class (equations 3.3 and 3.4). The final performance measures are obtained by computing the average of the individual results [57].

3.9 - Concluding Remarks

The characteristics of each of the movement phases were defined and the rationale behind the definitions was explained. Our definition resulted from the combination of the definitions explored by Schenkman et al, [8] and Goffredo et al, [9].

Once the phases were defined an initial segmentation was performed in order to get the initial movement segmentation, even if crude. A manual analysis of each movement was then performed in order to re-evaluate the initial segmentation. This analysis was performed by frame-by-frame verification, including an initial verification using a webcam to film the sagittal view, while the Kinect filmed the coronal view. Finally a last re-evaluation of the labelling was performed using data acquired with the Kinect, acquiring our ground truth for the system.

A total of 3283 frames were acquired with the Kinect. This was performed in a controlled environment, with a specific setup. For each frame a total of 13 features were computed being listed in Table 3.3.

These features from the original dataset were normalized and the classes were balanced, reaching a training dataset of 4176 samples (feature vectors). Both the original and training datasets were divided into 5 groups, each one each containing 7 movements, one from each subject. Then 4 of the 5 groups were used to train the classifier, and the classifier was tested on the remaining 7 movements from the original dataset that weren't used for training. This was performed a total of 10 times (5 for the original dataset and 5 for the training dataset), training with 10 different combinations of groups and testing on the group from the original dataset that was left out. This method was similar to a 5-fold cross-validation process. We used a HMM classifier in which we trained 5 inner models with each of the phases features, each one with one hidden state. These inner models were trained to mimic the characteristics of each of the phases of the movement. When a new vector of features (obtained from the analysis of a new frame) is given to the system, its likelihood is calculated for each of the inner models, and the label corresponding to the highest score is the final label of the new vector and corresponding frame.

Table 3.3 - Summary of the features computed for each frame.

Features
Shoulder Relative depth position (d^{Z}_{rs-ls})
Left hip height variation (ΔY_{lh})
Right hip height variation (ΔY_{rh})
Hip centre height variation (ΔY_{hc})
Relative depth distance between *rs* and *rk* (d^{Z}_{rs-rk})
Relative depth distance between *ls* and *lk* (d^{Z}_{ls-lk})
Relative depth distance between *rh* and *rk* (d^{Z}_{rh-rk})
Relative depth distance between *lh* and *lk* (d^{Z}_{lh-lk})
Relative depth distance between *rk* and *ra* (d^{Z}_{rk-ra})
Relative depth distance between *lk* and *la* (d^{Z}_{lk-la})
Height estimation ($height$)
Shoulder centre height variation (ΔY_{cs})
Relative depth between *cs* and *s* (ΔZ_{cs})

During the performance of the movements, kinematic information about the movement is extracted: trunk and ankle angles with the ground, and knee and ankle angles were extracted. The respective angular velocities and COM velocity were also extracted, along with the total duration of the movement and the duration of each phase.

Chapter 4

Implementation

4.1 - Libraries and Software

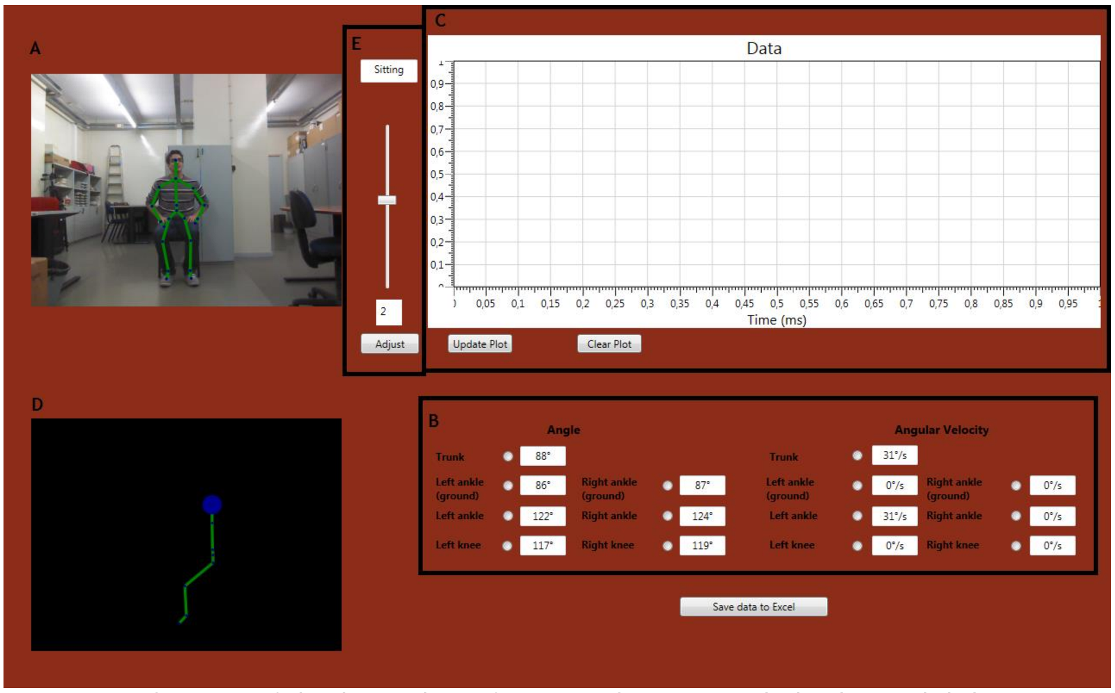
In this work we used Visual Studio 2012 to develop a Windows Presentation Foundation (WPF) application in C#. The WPF application is the final product of this project. Several libraries and frameworks had to be used in order to develop the system:

- Kinect Software Development Kit (SDK) [29, 31] was used in order to obtain data from the Kinect;
- WPF: Webcam Control [58] was used in the initial segmentation to obtain a sagittal view of the movements;
- Accord.NET Framework [47] was used in order to implement the necessary operations using HMMs in C#;
- Dynamic Data Display software [59] was used in order to create the plots in the interface;

The SMOTE was implemented in Matlab® (7.13, R2011 b) [60].

4.2 - Interface

In order to give visual feedback about the movements to the user, an interface was developed. In Figure 4.1 we can see an overview of the developed interface. The most important sections of the interface are highlighted and labelled.



	Angle			Angular Velocity			
Trunk	88°			Trunk	31°/s		
Left ankle (ground)	86°	Right ankle (ground)	87°	Left ankle (ground)	0°/s	Right ankle (ground)	0°/s
Left ankle	122°	Right ankle	124°	Left ankle	31°/s	Right ankle	0°/s
Left knee	117°	Right knee	119°	Left knee	0°/s	Right knee	0°/s

Figure 4.1 - Overview of the designed interface; **A** – RGB output with the detected skeleton; **B** – real-time angles and angular velocity data display; **C** – Plot section; **D** – Sagittal view of the detected skeleton; **E** – Kinect's tilt controller.

The interface gives us the possibility of watching the RGB display (part **A**) at the same time as the skeleton is fit to the subject's body. In Figure 4.2 we can see an example of this display. The blue dots represent the tracked joints and the yellow dots represent the inferred joints. If both joints are tracked, the segment linking them will be green. Otherwise the segment will be yellow.

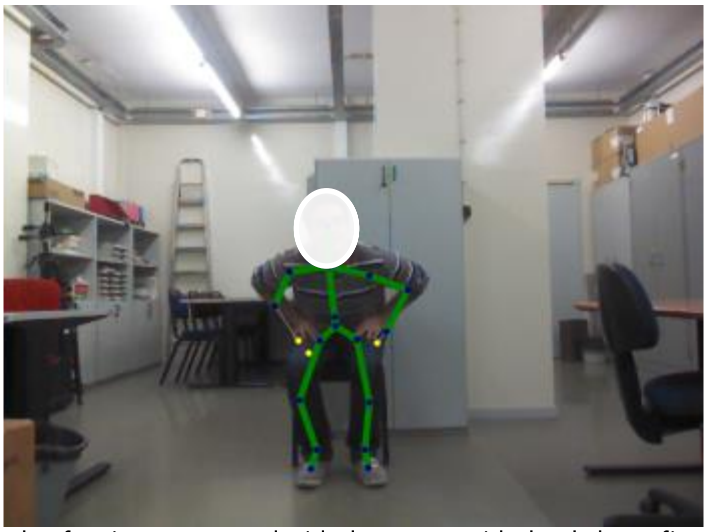
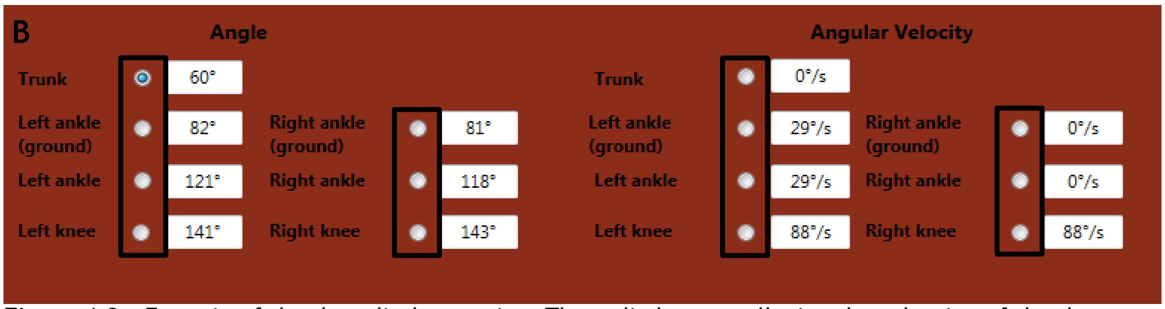


Figure 4.2 - Example of an image captured with the system with the skeleton fit to the user's body.

The trunk, ankles, ankle with the ground and knee angles, along with angular velocities are displayed in real-time as the movement is being performed (part **B**).



B

Angle			
Trunk	● 60°		
Left ankle (ground)	○ 82°	Right ankle (ground)	○ 81°
Left ankle	○ 121°	Right ankle	○ 118°
Left knee	○ 141°	Right knee	○ 143°

Angular Velocity			
Trunk	○ 0°/s		
Left ankle (ground)	○ 29°/s	Right ankle (ground)	○ 0°/s
Left ankle	○ 29°/s	Right ankle	○ 0°/s
Left knee	○ 88°/s	Right knee	○ 88°/s

Figure 4.3 - Zoom in of the data display section. The radio buttons alloying the selection of the data to be plotted are highlighted.

The acquired angles and angular velocities can be plotted in order to observe the characteristics and evolution of the movement over time (part **C**). Several plots can be performed at the same time in order to compare the results. We can select the desired plots using the radio buttons highlighted in Figure 4.3.

The 2 buttons under the plotting area are used to update and clean the plotting area if the user wants to create a new plot. An example of a plot obtained with the system can be seen in Figure 4.4.

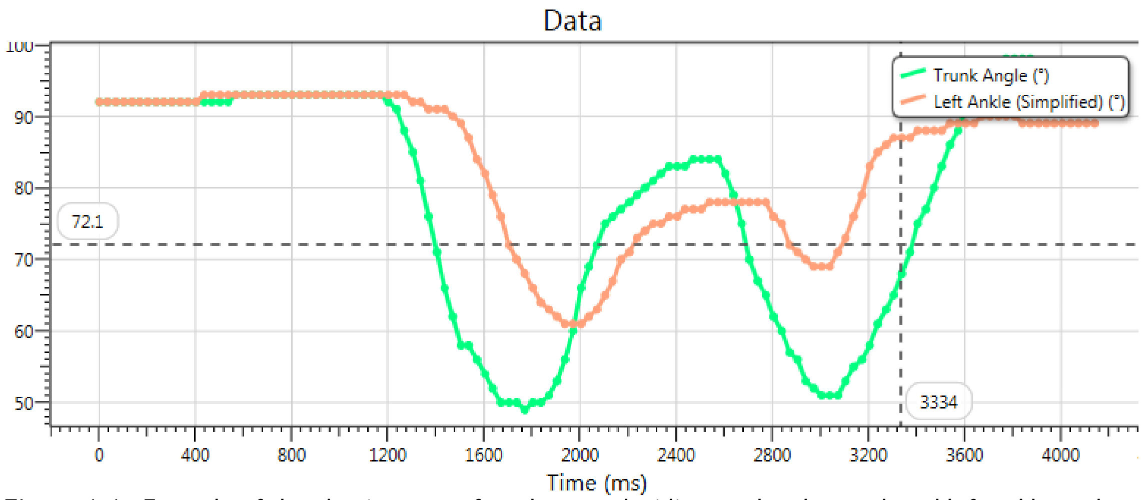


Figure 4.4 - Example of the plotting area after the user deciding to plot the trunk and left ankle angles.

It is also possible to observe the sagittal view of the movement (part **D**) as the movement is performed. Figure 4.5 shows an example of the sagittal view and correspondent coronal view.

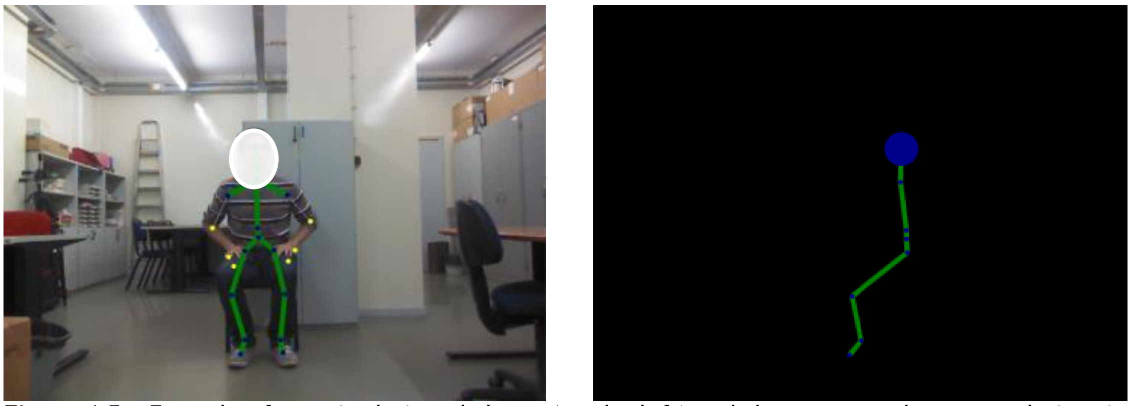


Figure 4.5 - Example of a sagittal view skeleton (on the left) and the correspondent coronal view (on the right).

By clicking on the button beneath part **B** (Figure 4.1), the user can save all the data acquired during the performance of the movement to a spreadsheet.

In part **E** (Figure 4.1) a slide bar was implemented in order to control the tilt of the Kinect. This is important since the Kinect's tilt is controlled by motors, which should not be forced in any kind of way. The tilt should be controlled via software in order to preserve the Kinect. Also, controlling the tilt with the interface gives a lot of flexibility to the system. With this we are able to set the system in different environments. Furthermore, when any of the joints of the user is outside of the field of view of the Kinect, a red box will appear warning the user about this fact. This will help the user to correctly setup the system.

Chapter 5

Experimental Results Analysis and Discussion

In this chapter, the results of the segmentation, information extraction and classification stages are reported and analysed. Finally the overall performance of the system is analysed and discussed.

5.1 - Movement Segmentation and Datasets

5.1.1 -Initial Segmentation

We started solving the problem by performing an initial segmentation of the movement with fixed thresholds. This initial segmentation was very crude, with the objective of simply acquiring data to have an idea of the difficulty of the segmentation problem.

The system detected the “Sitting” and “Standing” phases easily. This means that the transition from “Standing” to “Sitting”, marking the moment when the system should be prepared to receive a new movement, and from “Phase 3” to “Standing”, marking the end of a full movement, were correctly detected. Based on this, an initial segmentation was performed, acquiring the timestamps of the beginning of these phases.

On the other hand, the transition from “Sitting” to “Phase 1” was hard to detect. The system usually detected the beginning of “Phase 1” later than expected. Although this phase was always detected, it was only correctly detected 4 out of 5 times per subject (each subject performed 5 movements). The main issue detecting the beginning of “Phase 1” is connected to the detection of the beginning of the trunk movement. Since only joint position variations bigger than 1cm were detected by the developed system, small variations due to slower movements were troublesome.

As described in *Chapter 3*, during the initial segmentation, the detection of one phase depended on the detection of the previous one. Even if detected late, “Phase 1” did not create problems for the detection of “Phase 2”, since “Phase 1” is one of the longest phases of the movement.

The main problems of the initial segmentation were in the detection of “Phase 2” and by consequence in the detection of “Phase 3”. “Phase 2” lasts from 2 to 8 frames, depending on how fast the movement is performed. This is a time window of 66ms to 264ms considering a frame rate of 30 frames per second. Detecting a continuous variation of the hips’ height in such a small time window proved difficult in the initial segmentation. We had to guarantee that it was actually the beginning of “Phase 2” and not just a variation due to unstable data or jittering. In some extreme cases “Phase 2” was not detected during any of the 5 movements performed by subject. At best “Phase 2” was detected 3 out 5 times. Consequently the detection of “Phase 3” was hindered in some cases, due to the imposed necessity of detecting the previous phase in order to correctly detect the new one. Every time “Phase 2” was correctly detected, “Phase 3” was also correctly detected.

5.1.2 -Datasets

A manual segmentation was performed using the initial segmentation as a starting point. All movements were analysed frame-by-frame in order to confirm and/or correct the results of the initial segmentation. After establishing the features to be acquired, another re-evaluation of the segmentation was performed using these features. With this procedure we created the ground truth of the system, consisting of a dataset of 3283 samples, divided into 5 classes: “Sitting”, “Phase 1”, “Phase 2”, “Phase 3” and “Standing”. This dataset was unbalanced presenting some classes with very few samples. “Phase 2” class was most evident example, having only 108 samples in a total of 3283 (less than 4% of the dataset). In order to balance the dataset SMOTE was applied obtaining a new dataset with a total of 4176 samples, divided into the same 5 classes. The datasets are summarized in Figure 5.1.

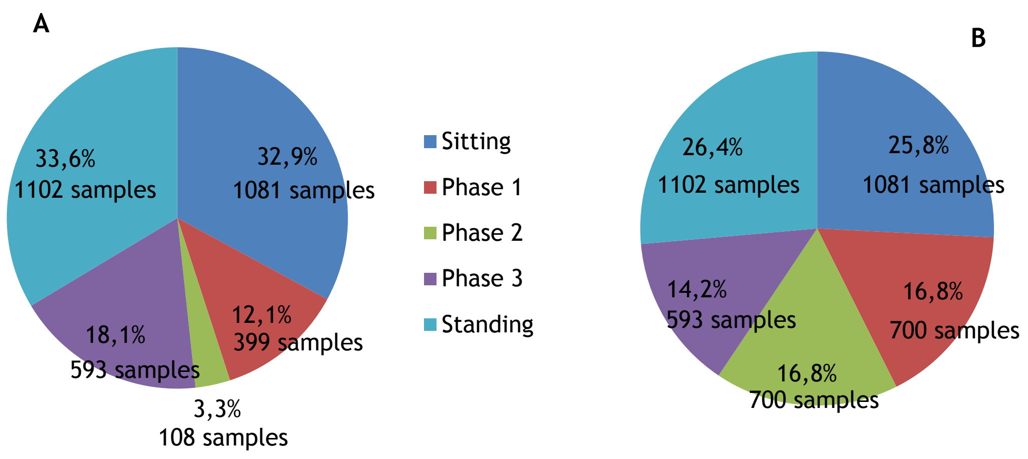


Figure 5.1 - Distribution of the classes in the datasets; **A** - Original dataset with 3283 samples; **B** - Dataset after SMOTE.

Figure 5.1 shows the distribution of the classes in both datasets (before oversampling and after oversampling). Just from observation of the Figure we can see that the classes are much more balanced in the oversampled dataset (Figure 5.1 –**B**) when comparing with the original dataset. In the new dataset we have 700 samples in classes “Phase 1” and “Phase 2” instead of 399 and 108 samples respectively. As mentioned in *Chapter 3* we would want to have each class representing the same portion of the dataset (20%). This was not fully achieved, but a much more uniform distribution was obtained for the new dataset. The impact of the SMOTE and the use of the new dataset to train the classifier will be discussed in the following sections.

5.2 - Classification Results

5.2.1 - Training algorithms comparison

The first step before deciding which of the training algorithms to use in the final application, was to analyse the classification results and select the one with the best results.

We can see the results obtained for the classifier trained using the Baum-Welch algorithm (BWa) and the original dataset on Table 5.1 and for the classifier trained using the Segmental K-Means algorithm (SKMa) and the original dataset on Table 5.2. It is possible to observe that for the classes “Sitting”, “Phase 2”and “Standing” the number of TPs was lower for the SKMa case. This means that the number of correct detections of these classes using the classifier trained with the SKMa was lower. This is important since we want to be able to delimit and segment the phases as correctly as possible. Having a high number of correct detections is essential for the system.

Also, for the same 3 classes, the number of FNs is higher when using the SKMa for training. This means that, for these 3 classes, the classifier misses the identification of more samples as being part of these classes than the BW algorithm.

Table 5.1 - Confusion matrix obtained using the Baum-Welch training algorithm and the original dataset.

		Classifier decision					
		Sitting	Phase 1	Phase 2	Phase 3	Standing	Total
True Class	Sitting	1022	52	4	3	0	1081
	Phase 1	40	313	36	10	0	399
	Phase 2	0	14	89	5	0	108
	Phase 3	0	2	32	492	67	593
	Standing	0	0	0	83	1019	1102
	Total	1062	381	161	593	1086	3283

Table 5.2 - Confusion matrix obtained using the Segmental K-Means algorithm and the original dataset.

		Classifier decision					
		Sitting	Phase 1	Phase 2	Phase 3	Standing	Total
True Class	Sitting	977	73	0	31	0	1081
	Phase 1	28	313	58	0	0	399
	Phase 2	0	21	77	10	0	108
	Phase 3	0	11	47	500	35	593
	Standing	0	0	0	118	984	1102
	Total	1005	418	182	659	1019	3283

Table 5.3 - Precision and recall results for all the classes and average value obtained using the Baum-Welch training algorithm and the original dataset.

	Sitting	Phase 1	Phase 2	Phase 3	Standing	Average
Precision	0.96	0.82	0.55	0.83	0.94	0.82
Recall	0.95	0.78	0.82	0.83	0.92	0.86

Table 5.4 - Precision and recall results for all the classes and average value obtained using the Segmental K-Means algorithm and the original dataset.

	Sitting	Phase 1	Phase 2	Phase 3	Standing	Average
Precision	0.97	0.76	0.42	0.76	0.89	0.76
Recall	0.90	0.78	0.71	0.84	0.96	0.84

On the other hand, the results for the “Phase 1” and “Phase 3” classes are very similar when comparing classifiers. The differences are almost unnoticeable.

From the analysis of Tables 5.3 and 5.4 we conclude that the classifier trained with the BW algorithm performs better overall. When comparing the results for each class, it is interesting to see that even with a lower number of correct detections (TPs) the precision of the classifier trained with the SKMa is slightly higher for the “Standing” phase. This is due to the fact that, even with a lower number of TPs, the number of FPs is also lower, increasing the precision for this class. Even though this classifier produces a lower number of correct detections, it also produces a lower number of samples incorrectly assigned to this class.

When observing the average values of precision and recall, we can see that overall the classifier performs better when trained with the BWa (and using the original dataset). Even when using the oversampled dataset for training, the classifier trained with the BWa performed better than that trained with SKMa. This led to the conclusion that the best training algorithm for the final application is the BWa.

5.2.2 - Training Datasets comparison

Once we have excluded one of the training algorithms, it is important to see the effect of the SMOTE in the classification process. Table 5.5 shows the confusion matrix obtained for the classifier trained with the BWa using the oversampled dataset.

Table 5.5 - Confusion matrix obtained using the Baum-Welch training algorithm and the oversampled dataset.

		Classifier decision					
		Sitting	Phase 1	Phase 2	Phase 3	Standing	Total
True Class	Sitting	1035	42	0	3	1	1081
	Phase 1	58	289	50	2	0	399
	Phase 2	0	12	92	4	0	108
	Phase 3	0	0	29	507	57	593
	Standing	0	0	0	46	1056	1102
	Total	1093	343	171	562	1114	3283

We can see that for the classes “Sitting”, “Phase 2”, “Phase 3” and “Standing” the number of TPs detected was slightly higher, and the number of FNs was slightly lower when using the oversampled dataset. For these cases the oversampling process increased the number of correctly identified samples and decreased the number of samples that were not recognized as part of these classes. These results are expected for the “Phase 2” class, since one of the characteristics of SMOTE is to make the class more general by introducing new synthetic samples. With this, we expect the classifier to be able to correctly recognize more samples than before. On the other hand, the classes “Sitting”, “Phase 3” and “Standing” presented a higher number of FPs when using the oversampled dataset for training. This means that more samples were incorrectly assigned to these classes.

When observing the results of the “Phase 1” classification we can see that the number of TPs decreased and the number of FNs increased. This was one of the classes that was oversampled using SMOTE, along with “Phase 2”. Taking into considerations what was previously described for “Phase 2”, we expected the opposite results. We expected an increase of the number of TPs and a decrease of the number of FNs.

These results could be related to the fact of having 2 classes being oversampled using SMOTE. When performing a movement, phase 2 will always come immediately after phase 1. Some characteristics of these phases are similar, like the flexion of the trunk. The main differentiating feature is the lift of the buttocks, represented by the movement of the hips.

By making both classes more general with SMOTE, we could have overlapped some of the features which previously distinguished the classes.

Table 5.6 - Precision and recall results for all the classes and average value obtained using the Baum-Welch training algorithm and the oversampled dataset.

	Sitting	Phase 1	Phase 2	Phase 3	Standing	Average
Precision	0.95	0.82	0.54	0.90	0.95	0.83
Recall	0.96	0.72	0.85	0.85	0.96	0.87

In Table 5.6 we can see the performance measures for the classifier trained with the BWa using the oversampled dataset. As expected, the “Phase 1” recall decreased, due to an increase of the number of FNs and a decrease of the number of TPs when compared the results using the original dataset. Across all classes the results were slightly better, which is can be observed by a really slight increase of the average results. Still, the difference between the results is very small. For a better understanding of the effects of the oversampling in the results we would probably need a bigger test dataset, with more samples of the least represented classes. With a bigger test dataset we could probably obtain conclusive results, being able to decide if the effect of the oversampling using the SMOTE is useful for the final application.

Overall, we conclude that the transition from phase 1 to phase 2 is difficult to detect. The classifier underperforms in the detection of this transition when compared to the rest of the phases even when using the oversampled dataset for training. Since the overall performance of the classifier is slightly better when using the oversampled dataset for training, we will be using the classifier trained with the BWa and the oversampled dataset in the final application.

5.3 - Extracted Information

After defining the characteristics of the classifier to use in the final application, information about the movements can be extracted using the developed system.

Considering an example of a full cycle of movement (starting in the standing position, sit down and then perform the STS movement) we can analyse some interesting aspects of the system.

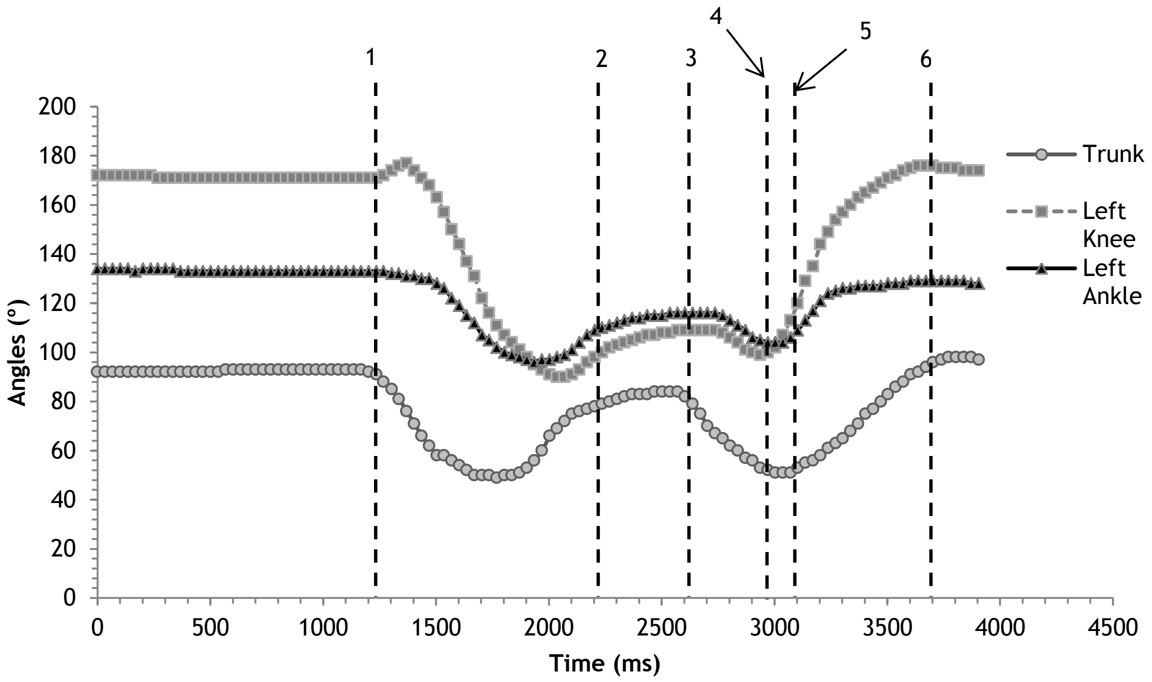


Figure 5.2 - Trunk, left knee and left ankle angles acquired for a full movement with important moments marked; **1** – sitting down; **2** - stabilization (sitting position); **3** - Beginning of the movement (phase 1); **4** - phase 2; **5** - phase 3; **6** - end of movement (standing position).

Figure 5.2 represents the trunk, left knee and left ankle angles acquired during a complete STS movement. Important moments of the movement are marked in the figure. With the information of these 3 angles we can delimit the phases of the movement.

The subject starts in the standing position. It is expected that the trunk angle is close to 90º. On the other hand, the ankle angle should be bigger than 90º, since we aren't considering the angle with the ground, but the angle with the foot articulation. Finally the left knee angle should be close to 180º. We verify that the trunk and knee angles are correctly acquired, while the ankle angle is probably bigger than expected (134º). These values are stable until marker 1. Marker 1 represented the moment when the subject starts to sit down. The information acquired between the markers 1 and 2 is not relevant for the work because if corresponds to the sitting down movement.

The stabilization in the sitting position can be observed when all acquired angles stabilize (marker 2). At this time the trunk and knee angles should be close to 90º, while the ankle angle should go back to the initial value. We can see that the trunk angle is the one closest to expected value. The knee angle is above the expected value and the ankle angle stabilized in a value lower than the initial one. These differences of the knee and ankle angles are expected due to the problems related with the skeleton detection. In Figure 5.3 we can see the sagittal and coronal views of the movement between markers 2 and 3.

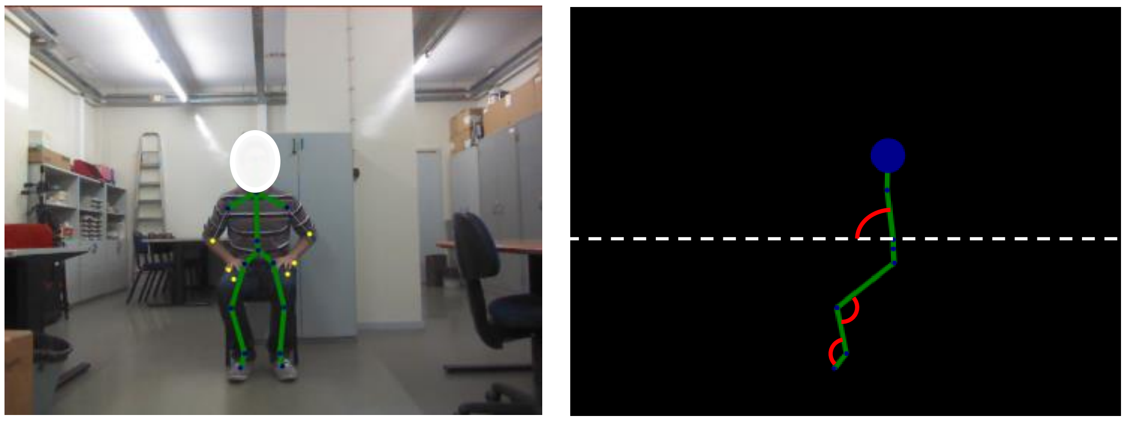


Figure 5.3 - On the left: Sagittal view of skeleton between markers 2 and 3; On the right: coronal of the same moment.

We can see that even when the subject is sitting, the skeleton won't perfectly fit the body of the subject. The ankle and knee angles are always bigger than 90º. This is a problem that is inherent to the Kinects skeleton tracking [28].

Marker 3 represents the beginning of the phase 1 of the movement. In Figure 5.2 we can see that the trunk angle will start to decrease from 90º to a minimum of 51º (mark 4). We can also detect a small variation of the ankle and knee angles a short period after phase 1 starting. When the knee angle starts to increase again, it means that the hips are moving. This means that phase 2 is starting, corresponding to the marker 4. The coronal and sagittal view of the beginning of phase 2 can be seen on Figure 5.4. It is possible to note that the ankle and trunk angles present lower values than expected.

As expected, phase 2 is the shortest phase of the movement, ending when the trunk angle reaches its minimum value (marker 5). This marks the beginning of phase 3, the extension phase. By simple observation of Figure 5.4 we can see that the trunk and ankle angles have decreased when comparing with Figure 5.3.

Phase 3 goes from marker 5 to marker 6, and it is defined by the full extension of the body. All the angles will increase until the standing position is reached. At marker 6, the trunk angle should be around 90º and the knee angle should be around 180º.

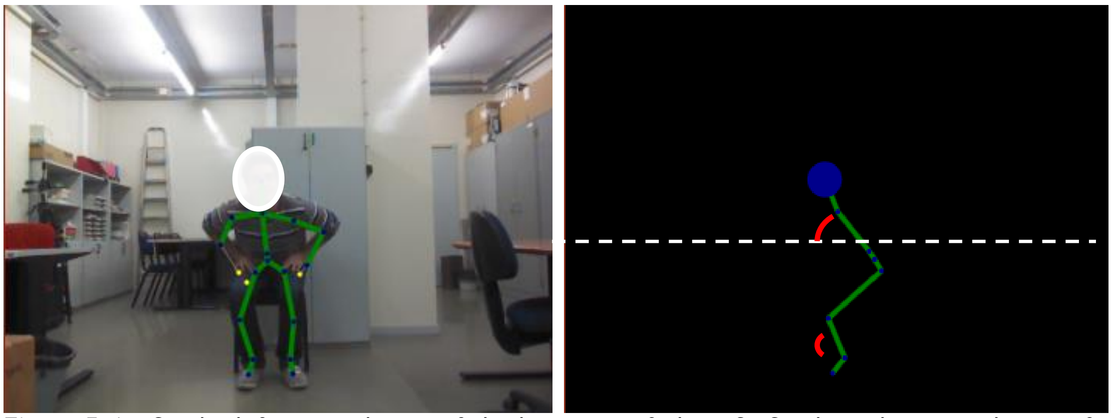


Figure 5.4 - On the left: sagittal view of the beginning of phase 2; On the right: coronal view of the beginning of phase 2.

Figure 5.5 shows the end of the movement, when the full extension of the body is achieved. It is possible to see that despite the subject being in the standing position, the joints aren't all vertically aligned. This happened in all the tests, leading to the conclusion that even when working in the normal conditions for the Kinect (standing position), the joints won't have the same depth value when the body is fully extended.

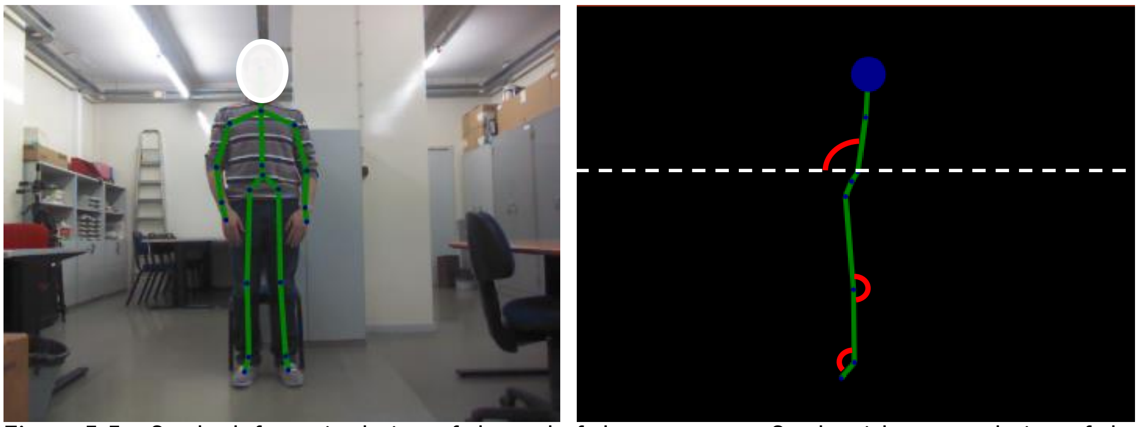


Figure 5.5 - On the left: sagittal view of the end of the movement; On the right: coronal view of the end of the movement.

When this work was developed the skeleton tracking algorithm was not fully prepared to analyse the whole body in the sitting position. Although, we can see that the system is still able to detect variations of the angles, being useful even if the values don't completely correspond to the expected ones. The variation of values is still consistent with the what is expected for this kind of movement.

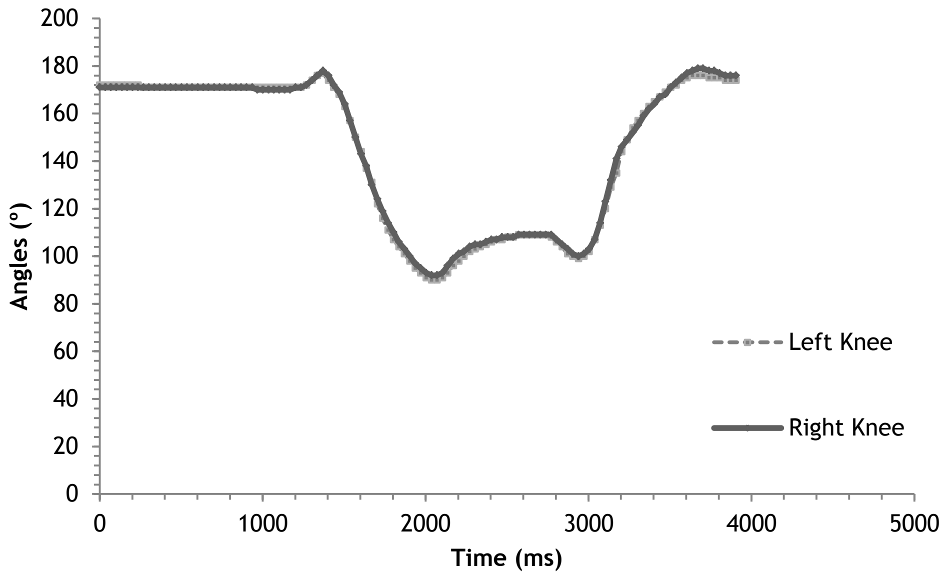


Figure 5.6 - Graphical representation of the left and right knee angles acquired for the movement previously analysed.

In Figure 5.6 we can see the data acquired during the previous movement for both knees. It is interesting to see that the data is very similar, and the plots overlap almost completely. This means that the acquired data for the knees is consistent, since we will get similar values when measuring the right and the left knee angles.

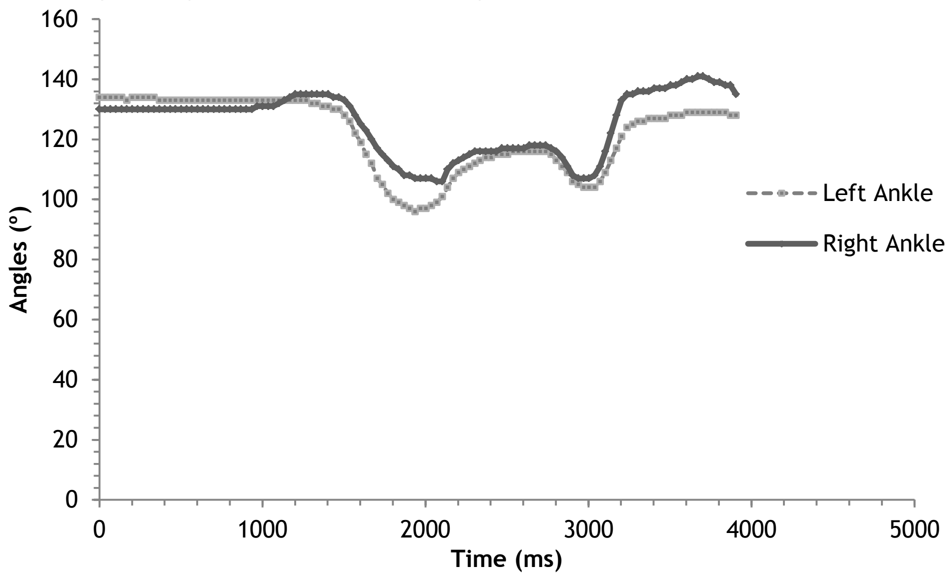


Figure 5.7 - Graphical representation of the left and right ankle angles acquired for the movement previously analysed.

The same affirmation cannot be made for the ankle angles. In Figure 5.7 we can see that the values of the left and right ankle angles vary a lot. This is due to the fact that the foot joints are unstable when compared to the rest of the joints used to compute these angles. The foot joints tend to be inferred a lot, and even when they aren't inferred, they tend to be unstable, even in a steady position. This led to the necessity of computing these angles differently. The results of this method (*section 3.6*) can be seen on Figure 5.8.

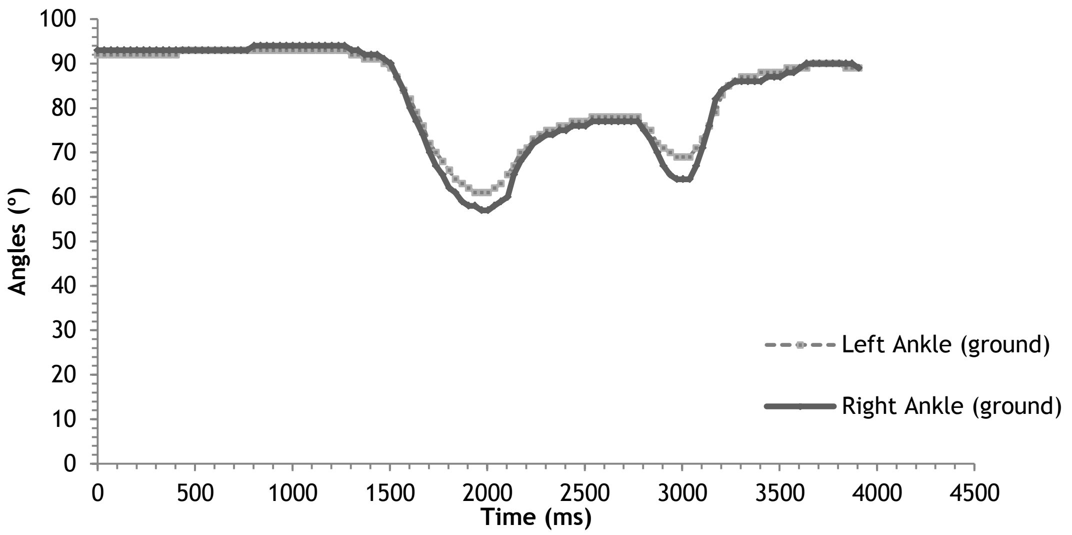


Figure 5.8 - Graphical representation of the left and right ankle angles with the ground acquired for the movement previously analysed.

We can see that, even if not perfect, the results are better with the simplification. The angle of both ankles will overlap almost all the time. Also, when comparing the max value of the angles, with and without the simplification, we can see that the maximum values are lower when using the simplification. With the simplification we will a obtain maximum angle close to 90°. Without the simplification the angles are around the 130° degrees. Detecting a 90° angle in the ankles is much more intuitive than a 130° angle. In a standing position, if we consider the ground as the reference, 90° is a much more realistic and understandable value than 130°.

The angles at lift off obtained with the system are presented in Table 5.7. These results can be compared with the ones obtained in [9] and [61], shown in Figure 5.9. The symbol γ represents the ankle angle with the ground described in *section 3.6*. We should only compare the results with the results obtained using a young population performing movements at normal pace, since those were the restrictions to this system.

By comparison of the values in Table 5.7 and Figure 5.9 we can see that the knee and ankle angles obtained with the Kinect are approximate to the ones obtained with the marker-based system ([61]). The results obtained for the γ are slightly different from the ones reported by other authors [9]. In both works ([9] and [61]) the authors reported γ of approximately 50°. On the other hand, with this system we obtained a γ of approximately 80° at lift off. This difference is probably due to the depth data acquired with the Kinect. When the subject is sitting, the skeleton is not entirely fit to the body, leading to discrepancies in the obtained values when compared to the literature.

Gross et al. [61] also reported a trunk angle at lift off of 55.1±12.8 degrees. In this work we obtained a trunk angle at lift off of 62±9 degrees. The difference between the results is minimal. Also, we can see that with this system we obtained a slightly lower standard deviation. But we should also consider that all the angles acquired were rounded up, so lower variations of the angle won't be detected.

Table 5.7 - Knee, ankle and ankle with the ground angles at lift off during the STS movement.

	Angles						
	Left Knee	Right Knee	Left Ankle	Right Ankle	Left γ	Right γ	Trunk
μ (º)	119	116	129	130	82	81	62
±σ (º)	11	9	17	17	7	9	9

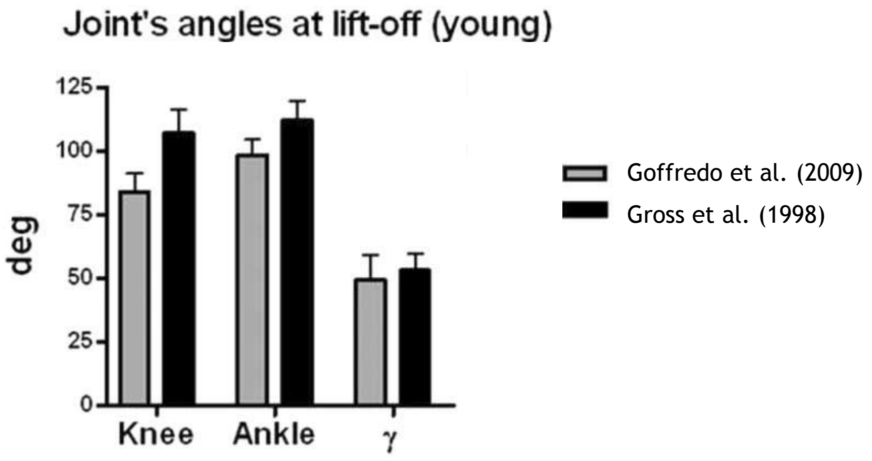


Figure 5.9 - Hip, knee, ankle, and γ angles at lift off during STS movement. The results obtained by Goffredo et al. [9] with the markerless system are compared with the results obtained by Gross et al. [61] with a marker-based system (image adapted from [9]).

In this work, angular velocity data was also acquired. The results are shown in Figure 5.10. Using the same rationale used to analyse Figure 5.2, we can draw some conclusions from the acquired data.

We can start by analysing the behaviour of the data over time. It is possible to see that the acquired that has sudden variations. For example, the data before marker 1 should be always 0, since the subject is standing up and no movements are being performed. From this observation we can say that a filtering process, like a median filter, should have been applied to the angular velocity values, in order to remove the spikes of the data.

By analyses of the trunk angular velocity, we can see that it is easy to distinguish the moment when the subject starts sitting. This can be detected by the variation of the trunk angular velocity, which will become negative. In this case a negative angular velocity means that the angle is diminishing over time. This is consistent with bending the trunk forward in order to sit down. Also, even with positive spike of the left knee's angular velocity, both the knee's angular velocity and the ankle's angular velocity will take negative values, indicating that the subject is flexing knees and ankles in order to sit down.

After these variations we can see a stabilization phase between markers 2 and 3, where angular velocities tend to 0º/s. This will correspond to the sitting phase, before the movement is initiated.

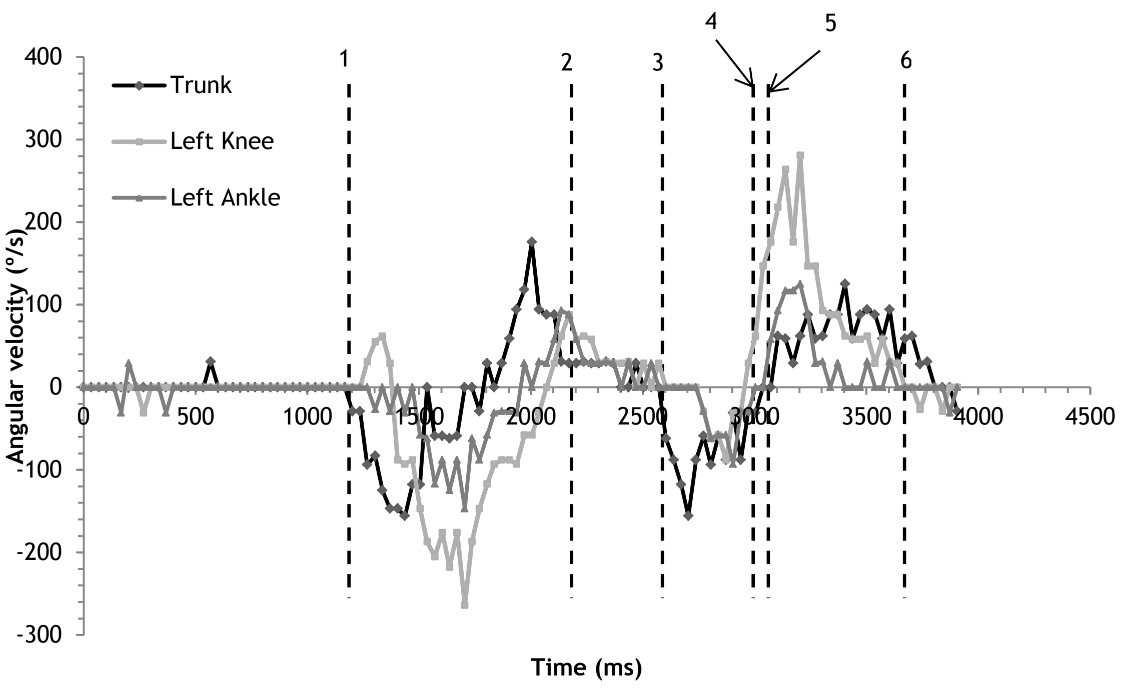


Figure 5.10 - Trunk, left knee and left ankle angular velocities acquired for a full movement with important moments marked; **1** – sitting down; **2** - stabilization (sitting position); **3** - Beginning of the movement (phase 1); **4** - phase 2; **5** - phase 3; **6** - end of movement (standing position).

As mentioned before, marker 3 will correspond to the beginning of phase 1. We can see that the once again the angular velocity of the trunk becomes negative, which indicates the bending of the upper body. This trend is followed by the knees and ankles after a few milliseconds. Before the initiation of phase 2, the angle of the knees and ankles has to decrease in order to be able to lift the buttocks. In general the whole body starts moving towards (in the direction of the Kinect considering the setup explained in *section 3.2*) in order to lift the buttocks. This moment corresponds to marker 4, the beginning of phase 2. During phase 2 (between markers 4 and 5) we will see the trunk angle reaching a minimum value until it finally starts extending. This is followed by the extension of the knees and ankles, which is represented by the change in the corresponding angular velocities. These go from negative to positive. The moment when the trunk's angular velocity is 0°/s marks the beginning of phase 3. During this phase, the whole body is extending, which means the angles of trunk, knees and ankles will increase, until the stabilization (marker 6) marking the standing position and end of the movement.

Important information extracted from the movements is the duration of each phase (phase 1, phase 2 and phase 3). The results are presented in Table 5.8.

Table 5.8 - Duration of movement phases and total duration of the movement.

	Phase 1	Phase 2	Phase 3	Total
μ (s)	0.31	0.16	0.31	1.06
±σ (s)	0.07	0.06	0.07	0.14

As mentioned previously, phase 2 was the most problematic phase overall. Its duration is shorter than the other phases, making it hard to identify. We can see that phase 1 and phase 3 have similar durations. According to the number of true classes, phase 1 should last in average approximately 0.37 seconds (399 samples divided equally by 35 movements and multiplied by time between frames ~33ms) if all the movements were performed exactly the same way. Of course in a real situation this is not true, but this gives us a comparison of results. The duration of phase 1 will tend to be shorter than the expected value due to errors. If the system fails to detect the first sample that corresponds to phase 1, the obtained duration will be shorter than expected. The same logic applies to phase 3. If the classifier fails to correctly identify the beginning of the phase (which is what happens), the duration of the phase will be shorter than expected. For phase 3, this is even more noticeable. Ideally phase 3 would last approximately 0.56 seconds (593 samples divided equally by 35 movements and multiplied by the 33ms).

5.4 - Concluding Remarks

Considering the initial segmentation, the results were not good enough considering the final objective of the work. The initial segmentation was useful to gather information about the problem in hands, getting an initial estimation of the phases.

Using the initial segmentation as a base, a manual segmentation was performed and then re-evaluated, obtaining a dataset to use as ground truth. This dataset has 3283 samples, divided into 5 classes: “Sitting” with 1081 samples, “Phase 1” with 399 samples, “Phase 2” with 108 samples, “Phase 3” with 593 samples and “Standing” with 1102 samples.

Due to the classes being unbalanced, SMOTE was used for classes “Phase 1” and “Phase 2”, obtaining a new dataset with 4176 samples. The new dataset was more balanced than the original one.

Four HMM classifiers were trained with different 2 different algorithms and 2 datasets. The training algorithms used were the Baum-Welch algorithm (BWa) and the Segmental K-means algorithm (SKMa). With each training algorithm, both dataset (original and oversampled) were used. The results from the 4 classifiers were obtained and compared. The validation methodology used here was similar to 5-fold cross-validation, dividing both the original and oversampled datasets. Both datasets were split over 5 sets each, containing 1 full movement per subject, to a total of 7 movements per set. One set from the original dataset was then classified with two of the classifiers. One of these classifiers was previously trained on the remaining four sets of the original dataset and the other on the equivalent remaining four sets of the oversampled dataset. This was repeated five times to classify all the samples from the original dataset. For each classifier the results from the classification of the 5 sets were combined. With this, 4 different final classification results, one for each classifier, were obtained.

The classifier that had the best overall performance was the classifier trained with the BWa using the oversampled dataset. A precision of 0.83 and recall of 0.87 were obtained for this classifier. This classifier was used in our final application, in order to automatically segment the movements and to enable a better data acquisition.

Lastly, information about the movements was extracted, analysed and compared to previous works [9, 61]. Angle and angular velocity data were analysed. The results of the acquired data were good enough to be able to perform a segmentation of the movement based solely on these results. Along with the analysis of the data, an analysis of the sagittal and coronal views during the performance of the movement was performed, with the objective of performing a better analysis of the acquired data. Overall, the angles acquired with the system seem consistent with the results described in the literature ([9, 61]).

On the other hand, the angular velocity results were less consistent. Some instability was noted in the acquired data. A solution to this problem could be the use of a median filter, to remove the noticed spikes of the data.

Finally, the duration of the phases detected by the system was analysed. A tendency to obtain shorter durations than the expected was noticed, probably due to the results from the classification step. Each misclassification will lead to a shorter phase, especially the miss classifications in the phase transitions.

Chapter 6

Conclusion and Future work

6.1 - Conclusion

A new STS movement analysis system using the Kinect platform has been presented. The system performs an automatic segmentation of the movement using HMMs classifiers. As the movement is being segmented, information (angles and angular velocity of the main joints) is acquired. The segmentation and data acquisition are performed in real time and the acquired data can be saved for future consultation.

An interface was developed in order to give some insight about the work and feedback about the movements to the user. Based on the results reported in the previous chapters, we believe that this system offers a basis for future research and improvement in the home rehabilitation field.

6.2 - Future Research

In future research, a possible strategy to improve the performance of the system can be defined. For instance, a bigger dataset with balanced classes could be used to train the classifier. By improving the performance of the system we would improving the overall results. Also, the study could be extended to the elderly, in order to make the system more general.

Furthermore, new models could be trained in order to identify certain patterns of the movements. The main objective would be to categorize the movement as normal or abnormal. Other movement patterns could also be modelled in order to identify specific rising strategies or incorrect movement patterns.

References

[1] Janssen, W.G.M., H.B.J. Bussmann, and H.J. Stam, *Determinants of the sit-to-stand movement: A review*. Physical Therapy, 2002. 82(9): p. 866-879.

[2] Bohannon, R.W., *Measurement of sit-to-stand among older adults*. Topics in Geriatric Rehabilitation, 2012. 28(1): p. 11-16.

[3] Dall, P.M. and A. Kerr, *Frequency of the sit to stand task: An observational study of free-living adults*. Applied Ergonomics, 2010. 41(1): p. 58-61.

[4] Chen, S.H., Lee, Y. H., Chiou, W. K., Chen, Y. L., *A pilot study examining seat heights and subjective ratings during rising and sitting*. International Journal of Industrial Ergonomics, 2010. 40(1): p. 41-46.

[5] Inkster, L.M. and J.J. Eng, *Postural control during a sit-to-stand task in individuals with mild Parkinson's disease*. Experimental Brain Research, 2004. 154(1): p. 33-38.

[6] Kerr, A., B. Durward, and K.M. Kerr, *Defining phases for the sit-to-walk movement*. Clinical Biomechanics, 2004. 19(4): p. 385-390.

[7] Nuzik, S.,Lamb, R., VanSant, A., Hirt, S., *Sit-to-stand movement pattern. A kinematic study*. Physical Therapy, 1986. 66(11): p. 1708-1713.

[8] Schenkman, M., Berger, R. A., Riley, P. O., Mann, R. W., Hodge, W. A., *Whole-body movements during rising to standing from sitting*. Physical Therapy, 1990. 70(10): p. 638-651.

[9] Goffredo, M., Schmid, M., Conforto, S., Carli, M., Neri, A., D'Alessio, T., *Markerless human motion analysis in Gauss-Laguerre transform domain: An application to sit-to-stand in young and elderly people*. IEEE Transactions on Information Technology in Biomedicine, 2009. 13(2): p. 207-216.

[10] Dehail, P., Bestaven, E., Muller, F., Mallet, A., Robert, B., Bourdel-Marchasson, I., Petit, J., *Kinematic and electromyographic analysis of rising from a chair during a "Sit-to-Walk" task in elderly subjects: Role of strength*. Clinical Biomechanics, 2007. 22(10): p. 1096-1103.

[11] Munton, J.S., M.I. Ellis, and V. Wright, *Use of electromyography to study leg muscle activity in patients wth arthritis and in normal subjects during rising from a chair*. Annals of the Rheumatic Diseases, 1984. 43(1): p. 63-65.

[12] Rodrigues-De-Paula Goulart, F. and J. Valls-Solé, *Patterned electromyographic activity in the sit-to-stand movement*. Clinical Neurophysiology, 1999. 110(9): p. 1634-1640.

[13] Mathie, M.J., Coster, A. C. F., Lovell, N. H., Celler, B. G., *Accelerometry: Providing an integrated, practical method for long-term, ambulatory monitoring of human movement*. Physiological Measurement, 2004. 25(2): p. R1-R20.

[14] Khoshelham, K. and S.O. Elberink, *Accuracy and resolution of kinect depth data for indoor mapping applications*. Sensors, 2012. 12(2): p. 1437-1454.

[15] Shotton, J., Sharp, T., Fitzgibbon, A., Blake, A., Cook, M., Kipman, A., Finocchio, M., Moore, R., *Real-Time human pose recognition in parts from single depth images*. Communications of the ACM, 2013. 56(1): p. 116-124.

[16] Nirjon, S. and J.A. Stankovic, *Kinsight: Localizing and tracking household objects using depth-camera sensors*, in *8th IEEE International Conference on Distributed Computing in Sensor Systems,* 2012. p. 67-74.

[17] Chang, Y.J., S.F. Chen, and J.D. Huang, *A Kinect-based system for physical rehabilitation: A pilot study for young adults with motor disabilities*. Research in Developmental Disabilities, 2011. 32(6): p. 2566-2570.

[18] Lange, B., Chang, C. Y., Suma, E., Newman, B., Rizzo, A. S., Bolas, M., *Development and evaluation of low cost game-based balance rehabilitation tool using the Microsoft Kinect sensor*, in *Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference*2011. p. 1831-1834.

[19] Pedro, L.M. and G.A. De Paula Caurin, *Kinect evaluation for human body movement analysis*, in *2012 4th IEEE RAS and EMBS International Conference on Biomedical Robotics and Biomechatronics,* 2012. p. 1856-1861.

[20] Biswas, K.K. and S.K. Basu, *Gesture recognition using Microsoft Kinect*, in *5th International Conference on Automation, Robotics and Applications, ICARA 2011*, 2011. p. 100-103.

[21] Lai, K., J. Konrad, and P. Ishwar, *A gesture-driven computer interface using Kinect*, in *2012 IEEE Southwest Symposium on Image Analysis and Interpretation, SSIAI 2012*, 2012. p. 185-188.

[22] Patsadu, O., C. Nukoolkit, and B. Watanapa, *Human gesture recognition using Kinect camera*, in *JCSSE 2012 - 9th International Joint Conference on Computer Science and Software Engineering,* 2012. p. 28-32.

[23] Chang, C.Y., Lange, B., Zhang, M., Koenig, S., Requejo, P., Somboon, N., Sawchuk, A. A., Rizzo, A. A., *Towards pervasive physical rehabilitation using microsoft kinect*, in *2012 6th International Conference on Pervasive Computing Technologies for Healthcare and Workshops, PervasiveHealth 2012,* 2012. p. 159-162.

[24] Gama, A.D., Chaves, T., Figueiredo, L., Teichrieb, V., *Guidance and Movement Correction Based on Therapeutics Movements for Motor Rehabilitation Support Systems*, in *Proceedings of the 2012 14th Symposium on Virtual and Augmented Reality,* 2012, IEEE Computer Society. p. 191-200.

[25] Harms, M., *Advancing technology in rehabilitation*. Physiotherapy (United Kingdom), 2012. 98(3): p. 181-182.

[26] Yeh, S.C., Hwang, W. Y., Huang, T. C., Liu, W. K., Chen, Y. T., Hung, Y. P., *A study for the application of body sensing in assisted rehabilitation training*, in *2012*

International Symposium on Computer, Consumer and Control, IS3C 2012, 2012. p. 922-925.

[27] Papa, E. and A. Cappozzo, *Sit-to-stand motor strategies investigated in able-bodied young and elderly subjects*. Journal of Biomechanics, 2000. 33(9): p. 1113-1122.

[28] Webb, J. and J. Ashley, *Beginning Kinect Programming with the Microsoft Kinect SDK*. 1 ed., 2012: Apress. 321.

[29] Microsoft Corporation, *"Kinect for Windows"*. [Accessed on: October 12, 2012]; Available from: http://www.microsoft.com/en-us/kinectforwindows/.

[30] Fujimoto, M. and L.S. Chou, *Dynamic balance control during sit-to-stand movement: An examination with the center of mass acceleration*. Journal of Biomechanics, 2012. 45(3): p. 543-548.

[31] Microsoft Corporation, *MSDN Library - Kinect for windows SDK*. [Accessed on: January 10, 2013]; Available from: http://msdn.microsoft.com/en-us/library/hh855347.aspx.

[32] Duffy, J. *Exclusive: Inside Project Natal's Brain*. 2010 [Accessed on: January 10, 2013]; Available from: http://www.popsci.com/gadgets/article/2010-01/exclusive-inside-microsofts-project-natal.

[33] LaBelle, K., *Evaluation of Kinect joint tracking for clinical and in-home stroke rehabilitation tools*, in *Computer Science,* 2011: Notre Dame, Indiana. p. 67.

[34] OpenNI. *OpenNI - The standard framework for 3D sensing*. [Accessed on: October 12, 2012]; Available from: http://www.openni.org/.

[35] Clark, R.A., Pua, Y. H., Fortin, K., Ritchie, C., Webster, K. E., Denehy, L., Bryant, A. L., *Validity of the Microsoft Kinect for assessment of postural control*. Gait and Posture, 2012. 36(3): p. 372-377.

[36] Orendurff, M.S., Segal, A. D., Klute, G. K., Berge, J. S., Rohr, E. S., Kadel, N. J., *The effect of walking speed on center of mass displacement*. Journal of Rehabilitation Research and Development, 2004. 41(6 A): p. 829-834.

[37] Hahn, M.E. and L.S. Chou, *Age-related reduction in sagittal plane center of mass motion during obstacle crossing*. Journal of Biomechanics, 2004. 37(6): p. 837-844.

[38] Hughes, M.A., Weiner, D. K., Schenkman, M. L., Long, R. M., Studenski, S. A., *Chair rise strategies in the elderly*. Clinical Biomechanics, 1994. 9(3): p. 187-192.

[39] Schenkman, M., P.O. O'Riley, and C. Pieper, *Sit to stand from progressively lower seat heights: Alterations in angular velocity*. Clinical Biomechanics, 1996. 11(3): p. 153-158.

[40] Etnyre, B. and D.Q. Thomas, *Event standardization of sit-to-stand movements*. Physical Therapy, 2007. 87(12): p. 1651-1666.

[41] Aggarwal, J.K. and M.S. Ryoo, *Human activity analysis: A review*. ACM Computing Surveys, 2011. 43(3).

[42] Han, J. and M. Kamber, *Data Mining: Concepts and Techniques*. Second Edition ed, ed. M.R. Jim Gray, 2006, Morgan Kaufmann Publishers, p. 772-780.

[43] Shalabi, L.A., Z. Shaaban, and B. Kasasbeh, *Data Mining: A Preprocessing Engine*. Journal of Computer Science. 2(9): p. 735-739.

[44] Feng-Shun Lin, J. and D. Kulic. *Segmenting human motion for automated rehabilitation exercise analysis*, 2012.

[45] Panahandeh, G., Mohammadiha, N., Leijon, A., Handel, P., *Continuous hidden markov model for pedestrian activity classification and gait analysis*. IEEE Transactions on Instrumentation and Measurement, 2013. 62(5): p. 1073-1083.

[46] Liu, D., Wu, J., Wang, Y., Wang, J., Gong, Z., *Fight detection based on hidden markov model*, 2012.

[47] Souza, C.d., *The Accord.NET Framework* [Accessed on: 13 January, 2013]; Available from: http://code.google.com/p/accord/.

[48] Rabiner, L.R. and B.-H. Juang, *Introduction to Hidden Markov Models*. IEEE ASSP magazine, 1986. 3(1): p. 4-16.

[49] Rabiner, L.R., *Tutorial on hidden Markov models and selected applications in speech recognition*. Proceedings of the IEEE, 1989. 77(2): p. 257-286.

[50] Rodríguez, L. and I. Torres, *Comparative Study of the Baum-Welch and Viterbi Training Algorithms Applied to Read and Spontaneous Speech Recognition*, in *Pattern Recognition and Image Analysis*, F. Perales, et al., Editors. 2003, Springer Berlin Heidelberg. p. 847-857.

[51] Microsoft Corporation, *Skeletal Joint Smoothing White Paper* [Accessed on:January 10, 2013]; Available from: http://msdn.microsoft.com/en-us/library/jj131429.aspx.

[52] Al Shalabi, L. and Z. Shaaban. *Normalization as a Preprocessing Engine for Data Mining and the Approach of Preference Matrix*, 2007.

[53] Chawla, N.V., Bowyer, K. W., Hall, L. O., Kegelmeyer, W. P., *SMOTE: Synthetic minority over-sampling technique*. Journal of Artificial Intelligence Research, 2002. 16: p. 321-357.

[54] Pazzani, M.J., Merz, C. J., Murphy, P. M., Ali, K., Hume, T., Brunk, C., *Reducing Misclassification Costs*, in *International Conference on Machine Learning*, 1994. p. 217-225.

[55] Kubat, M. and S. Matwin. *Addressing the curse of imbalanced training sets: one-sided selection*, 1997. Morgan Kaufmann.

[56] Ling, C. and C. Li. *Data Mining for Direct Marketing: Problems and Solutions*. in *Knowledge Discovery and Data Mining*, 1998.

[57] Sokolova, M. and G. Lapalme, *A systematic analysis of performance measures for classification tasks*. Information Processing & Management, 2009. 45(4): p. 427-437.

[58] Musundi, M. *WPF: Webcam Control* [Accessed on: January 13, 2013]; Available from: http://www.codeproject.com/Articles/285964/WPF-Webcam-Control.

[59] Microsoft Corporation, *D3 - Dynamic Data Display* [Accessed on: May 12, 2013]; Available from: http://dynamicdatadisplay.codeplex.com/.

[60] Manohar. *SMOTE (Synthetic Minority Over-Sampling Technique)* [Accessed on:January 10, 2013]; Available from: http://www.mathworks.com/matlabcentral/fileexchange/38830-smote-synthetic-minority-over-sampling-technique.

[61] Gross, M.M., Stevenson, P. J., Charette, S. L., Pyka, G., Marcus, R., *Effect of muscle strength and movement speed on the biomechanics of rising from a chair in healthy elderly and young women*. Gait & Posture, 1998. 8(3): p. 175-185.

Appendix A

User's Manual

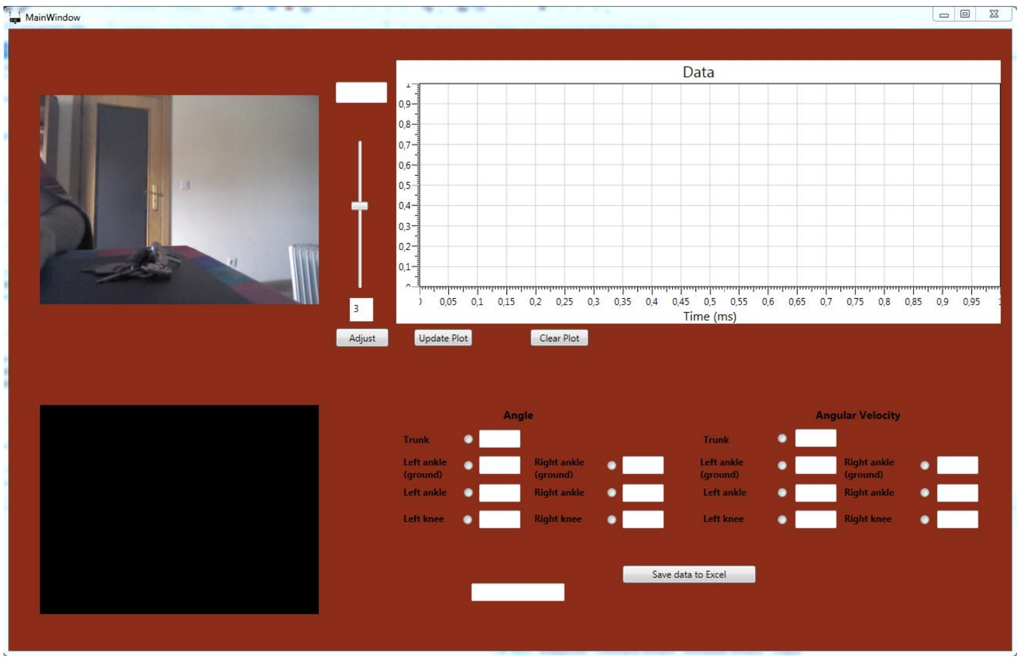
A manual for an easy usage of the system is presented here.

A.1 - How to start capturing movements?

Step 1 - In order to use the presented application the user must have previously set up the Kinect for Windows SDK and the Kinect for Windows Developer Toolkit. The instructions and installation files are available at: http://www.microsoft.com/en-us/kinectforwindows/develop/developer-downloads.aspx.

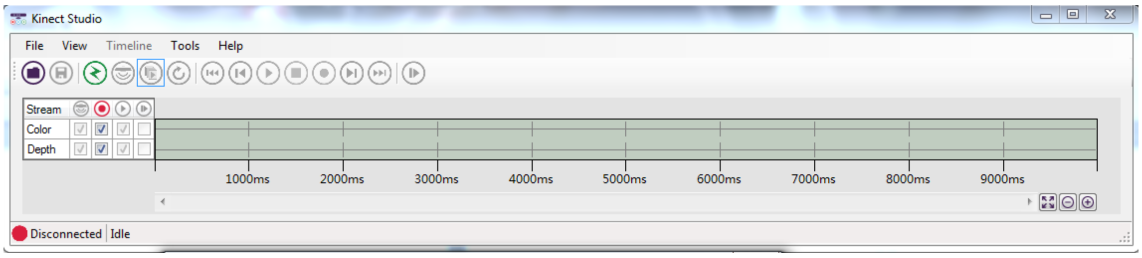
Step 2 - Connect the Kinect to the computer.

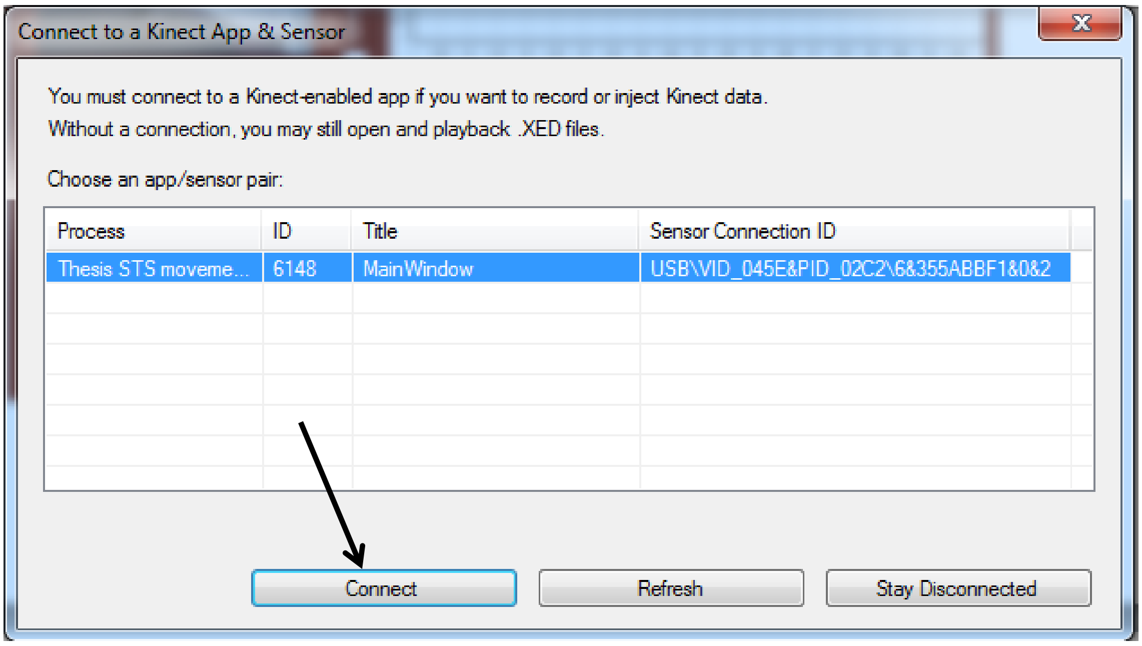
Step 3 - After having the necessary software installed and making sure the Kinect is connected the run the application "STS Movement Analyser" and wait until it loads. A window similar to the one presented should pop up.



Step 4 - Run the Kinect Studio application.

Two windows like the ones presented above should pop-up.


Kinect Studio
File View Timeline Tools Help
Stream
Color
Depth
1000ms 2000ms 3000ms 4000ms 5000ms 6000ms 7000ms 8000ms 9000ms
Disconnected | Idle

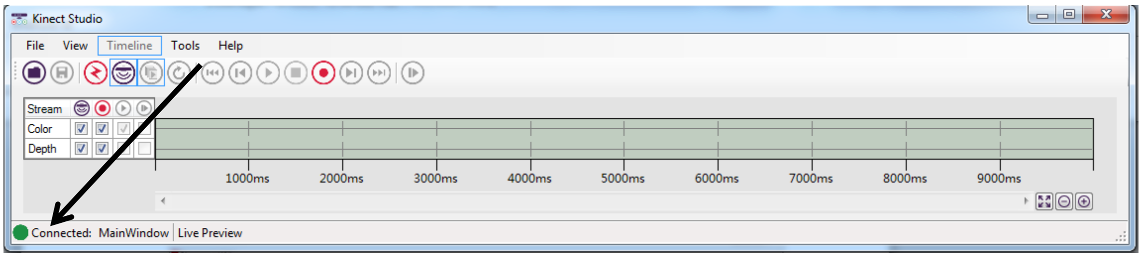


You must connect to a Kinect-enabled app if you want to record or inject Kinect data.
Without a connection, you may still open and playback .XED files.

Choose an app/sensor pair:

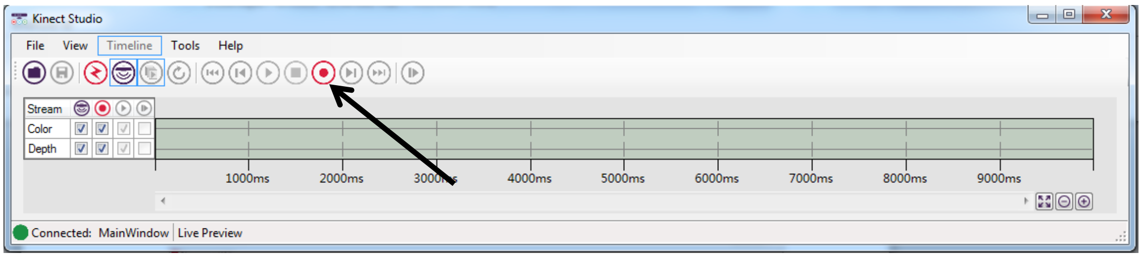
Process	ID	Title	Sensor Connection ID
Thesis STS moveme...	6148	MainWindow	USB\VID_045E&PID_02C2\6&355ABBF1&0&2

Step 5 - Press the “Connect” bottom on the second window to connect the Kinect studio to your application.


Kinect Studio
File View Timeline Tools Help
Stream
Color
Depth
1000ms 2000ms 3000ms 4000ms 5000ms 6000ms 7000ms 8000ms 9000ms
Connected: MainWindow | Live Preview

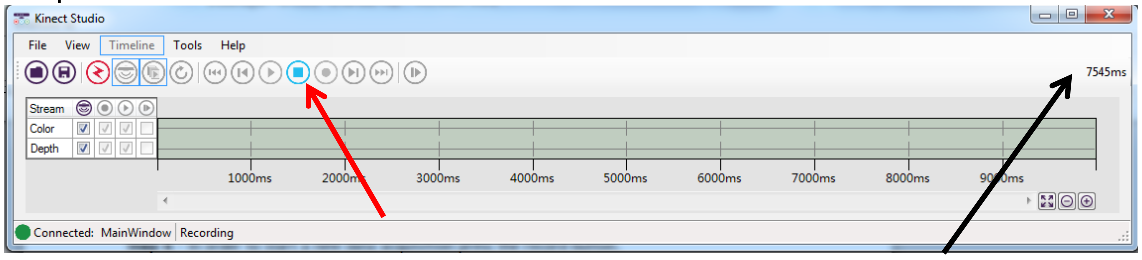
The Kinect studio status should now show that the application is connected.

Step 6 - In order to start a new data acquisition press the record button.


Kinect Studio
File View Timeline Tools Help
Stream
Color
Depth
1000ms 2000ms 3000ms 4000ms 5000ms 6000ms 7000ms 8000ms 9000ms
Connected: MainWindow | Live Preview

Once you press that button, the application will start saving the RGB, Depth and skeletal data being acquired by the "STS Movement Analyser" application.

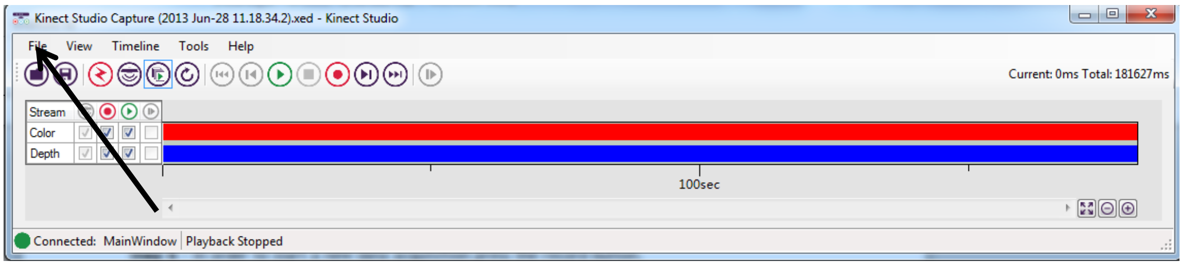
The previous window should look like this:


Kinect Studio
File View Timeline Tools Help
7545ms
Stream
Color
Depth
1000ms 2000ms 3000ms 4000ms 5000ms 6000ms 7000ms 8000ms 9000ms
Connected: MainWindow Recording

On the upper right corner a timer will start (black arrow), corresponding to the time elapsed since the beginning of the data acquisition.

Now the user can go perform the movements and once finished just press the stop button (red button).

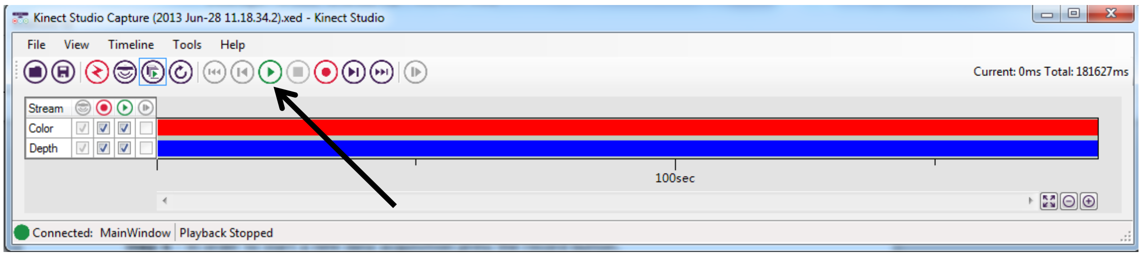
Step 7 - To save the acquired data, the user just has to go to File -> Save.


Kinect Studio Capture (2013 Jun-28 11.18.34.2).xed - Kinect Studio
File View Timeline Tools Help
Current: 0ms Total: 181627ms
Stream
Color
Depth
100sec
Connected: MainWindow Playback Stopped

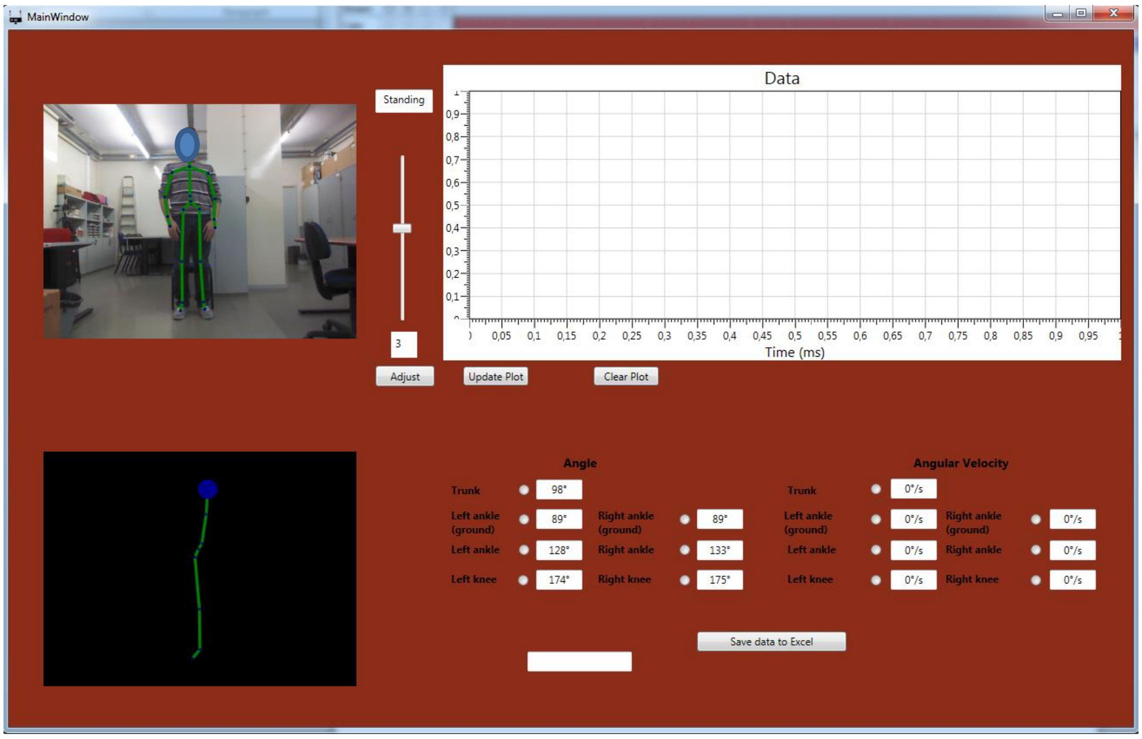
A window asking for the location where the user wants to save the data will pop up.

A.2 - How to review saved data?

In order to review a saved movement, the user just has to go to File -> Open and select the desired saved file.


Kinect Studio Capture (2013 Jun-28 11.18.34.2).xed - Kinect Studio
File View Timeline Tools Help
Current: 0ms Total: 181627ms
Stream
Color
Depth
100sec
Connected: MainWindow Playback Stopped

Once the file is open the user can just hit the play button to start reviewing the movement, angles and angular velocities on the "STS Movement Analyser" window.



Angle

Trunk	98°		
Left ankle (ground)	89°	Right ankle (ground)	89°
Left ankle	128°	Right ankle	133°
Left knee	174°	Right knee	175°

Angular Velocity

Trunk	0°/s		
Left ankle (ground)	0°/s	Right ankle (ground)	0°/s
Left ankle	0°/s	Right ankle	0°/s
Left knee	0°/s	Right knee	0°/s