



UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

Trabajo Fin de Máster

**Análisis de imágenes de profundidad en terapias
de rehabilitación supervisadas por robots
autónomos**

Eva María Mogená Cisneros

Septiembre, 2015



UNIVERSIDAD DE EXTREMADURA

Escuela Politécnica

MÁSTER EN INGENIERÍA DE TELECOMUNICACIÓN

Trabajo Fin de Máster

Análisis de imágenes de profundidad en terapias de rehabilitación supervisadas por robots autónomos

Autor: Eva María Moga Cisneros

Fdo:

Tutores: Pedro M. Núñez Trujillo / José Luis González Sánchez

Fdo:

TRIBUNAL CALIFICADOR

Presidente : Luis Landesa Porras

Fdo:

Secretario : Yolanda Campos Roca

Fdo:

Vocal : José Vicente Crespo

Fdo:

CALIFICACIÓN:

FECHA:

Contents

1	Introduction	17
2	State of the Art	21
3	Hypothesis and targets	33
4	RoboComp	37
4.1	Component Oriented Programming (C.O.P.)	39
4.2	Therapy: my component in RoboComp	41
5	The component and the therapy	45
5.1	Simon Says	45
5.2	RGB-D sensor	46
5.3	Exercises	49
5.4	Features	51
5.4.1	Quantity of motion (<i>QoM</i>)	52
5.4.2	Contraction Index (<i>CI</i>)	52
5.4.3	Angle Arm (α)	54
5.4.4	Height of the hands (Y)	55
5.4.5	Depth of the hands (D)	58
5.5	Interface	59
5.6	DataBase	61
5.7	Training Mode	62
5.8	Detecting Mode	64
5.9	Simon Game Mode	64
5.10	Accuracy Mode	66

5.11	Classification Models	67
5.11.1	Decision Tree	67
5.11.2	K-Nearest Neighbours	70
5.11.3	Supported Vector Machine	72
6	Experimental Results	79
6.1	Exercises Data	79
6.2	Features Extraction	79
6.3	KNN Accuracy depending on K value	85
6.4	Recognition Accuracy	87
6.5	Real Experience	93
7	Conclusions and Future Work	97
7.1	Conclusions	97
7.2	Future Work	98
7.3	Personal Evaluation	99
8	Acknowledgements	101
A	RoboComp Installation Manual	109
B	RoboComp Component Creation Manual	121
C	Graphic Interface Creation Manual	125

List of Figures

1.1	Gantt chart of tasks performed	18
2.1	CyberGlove	26
2.2	Wii Remote Vs Play Move	26
2.3	Kinect Vs Eye-Toy Vs Xtion Pro Live	27
2.4	Nao Robot	28
2.5	Armeo Spring Robot	29
2.6	ReoGo Robot	29
2.7	Manus del MIT Robot	30
4.1	RoboComp logo	37
4.2	Representation of components with their interfaces	39
4.3	Components communication	40
4.4	Data collection scheme with Kinect sensor	41
4.5	Example of a data file saved	42
4.6	Part of the code to change the file path	42
4.7	Model of a person	43
5.1	Simon Says Game	45
5.2	Kinect sensor for Windows	47
5.3	Sensors included in the Kinect	47
5.4	Reference System of the Kinect	48
5.5	Exercise 1	50
5.6	Exercise 2	50
5.7	Exercise 3	51
5.8	Exercise 4	51

5.9	Triangle to calculate CI	53
5.10	Triangle to calculate the Angle Arm	54
5.11	Exes Change	56
5.12	Maximun Height	57
5.13	Component Interface	59
5.14	Database Organization	61
5.15	Training and Labels files	63
5.16	Interface in the Simon Mode	65
5.17	Interface in the Accuracy Mode	66
5.18	Decision Tree Structure	67
5.19	Decision Tree Example	68
5.20	K-Nearest Neighbours Example	70
5.21	The Support Vectors	72
5.22	Two dimensions space Vs three dimensions space	73
5.23	Example of a linearly separable case	74
5.24	Different possibilities to choose a hyperplane	75
5.25	Example of a no linearly separable case	75
5.26	Example of a Kernel function does	76
6.1	Contraction Index for the different exercises	80
6.2	Angle of the Right Arm for the different exercises	81
6.3	Angle of the Left Arm for the different exercises	81
6.4	Height of the Right Hand for the different exercises	82
6.5	Height of the Left Hand for the different exercises	83
6.6	Depth of the Right Hand for the different exercises	84
6.7	Depth of the Left Hand for the different exercises	84
6.8	Accuracy of the KNN method according to K value	87
6.9	Real therapy session with a patient	94
A.1	Cloning Robocomp	109
A.2	Proving components folder is empty	110
A.3	Installation “git-anex” command	110
A.4	Installing RoboComp libraries	110

A.5	Opening the bashrc file	111
A.6	Editing the bashrc file	111
A.7	Compiling robocomp	111
A.8	“lib-qt” installation	112
A.9	“g++” installation	112
A.10	“ipp” installation	112
A.11	New line to link “ipp” with “opt” folder in “bashrc”	112
A.12	Errors after compiling	113
A.13	Looking for “gsl” package	114
A.14	“gsl” installation	114
A.15	Errors with “osg” package	115
A.16	“osg” installation	115
A.17	Instalación del proyecto	116
A.18	Problems with innermodel	116
A.19	“cmake-gui” installation	117
A.20	RoboComp installation	117
A.21	Simulator compilation	117
A.22	Installation missing packages for the simulator compiling	118
A.23	Tools folder installation	118
A.24	Creating a link between the two RoboComp directions	119
A.25	Creating link to the RoboComp folder	119
A.26	Installation of RoboComp components	119
B.1	. <i>cdsl</i> file in <i>DSLEditor</i>	121
C.1	therapyComp interface	125
C.2	Qt4Editor main display	126
C.3	Qt4Editor: item set up	127
C.4	How to connect interface items in your code	127

List of Tables

5.1	Kinect specifications	49
5.2	CI values for the different exercises	53
5.3	α values for the different exercises	55
5.4	Height values for the different exercises	58
5.5	Depth values for the different exercises	58
6.1	KNN accuracy according to K values	86
6.2	Accuracy obtained from the different decision methods	88
6.3	Accuracy of the Exercise 1 in the different methods	90
6.4	Accuracy of the Exercise 2 in the different methods	91
6.5	Accuracy of the Exercise 3 in the different methods	92
6.6	Accuracy of the Exercise 4 in the different methods	93

*Robots will play an important role in providing physical assistance and even
companionship for the elderly*

Bill Gates

Abstract

A future in which the contact with robots is something natural in our daily life is closer. But before achieve this, we must make many researches to make progress such as getting the robots to obtain a sufficient degree of autonomy to be able to handle themselves in different tasks, for example leading a therapy. In this project has been developed a system that allows the realization of a therapy session by playing the Simon Says game. It is especially focused on elderly people, since it is a population group that is growing. It has also created a database with 3D data of 20 subjects performing a series of exercises, as well as the features extracted from the data. Experimental results show that is possible to recognize between different types of exercises that makes the patient, which is crucial for the development of the game.

Resumen

Cada vez vemos más cercano un futuro en que el contacto con los robots sea algo natural en nuestro día a día. Pero antes de conseguir llegar a esto, hay que realizar muchas investigaciones para conseguir avances, como por ejemplo conseguir que los robots obtengan el grado suficiente de autonomía como para poder hacerse cargo por sí solos de distintas tareas, como por ejemplo la de liderar una terapia. En este proyecto se ha desarrollado un sistema que permite realizar una sesión de terapia ocupacional jugando al juego de Simon Dice. Está especialmente enfocado para las personas mayores, ya que es un grupo poblacional que va en aumento. También se ha creado una Base de Datos con los datos en 3D de 20 personas realizando una serie de ejercicios, así como de las características extraídas de esos datos. Los resultados experimentales muestran que se ha logrado reconocer entre los distintos tipos de ejercicios que realiza el paciente, lo que es crucial para el desarrollo del juego.

Key Words

Social robotics, elderly people, gesture-based applications, technology for the elderly, robots in therapy, robot autonomy, decision methods, Decision Tree, K Nearest Neighbours, Supported Vector Machine.

Chapter 1

Introduction

The robotics field is becoming increasingly important in our lives, raising the number of applications that we can use in our daily life or in special occasions. Robotics can be related to various fields such as medicine, education or social robotics. The first of them, the field of medicine, is where our work is framed.

This project focuses specifically on performing an occupational therapy session with a robot, especially designed for elderly people. In it, the robot will play the game of Simon Says with the patient, who will have to repeat the sequence of movements that the robot prompts. This will help to work both the physical condition of the patient and his memory by remembering the sequence of exercises.

To make easier the training process of the system, a video database has been created. That database is composed by 20 subjects (9 women and 11 men) aged from 20 to 50, recorded using the Microsoft Kinect RGB-D sensor, which allows to work in 3D because is able to record the length value (x) and the high value (y) and its depth information (z), and therefore it gives three points (x, y, z) . Every database video contains a subject doing the different exercises.

The significant role played by the Fundación CémitS-Computaex grant which I have been awarded is noteworthy in the development of this project. Thanks to the grant, this collaborative project between the Fundación CémitS-Computaex and the Universidad de Extremadura has emerged.

To meet the targets set in the project, which are detailed in Chapter 3, a series of tasks have been completed. The Gantt chart in Figure 1.1 exposes the time spent on each of the tasks performed up to the end of the project.

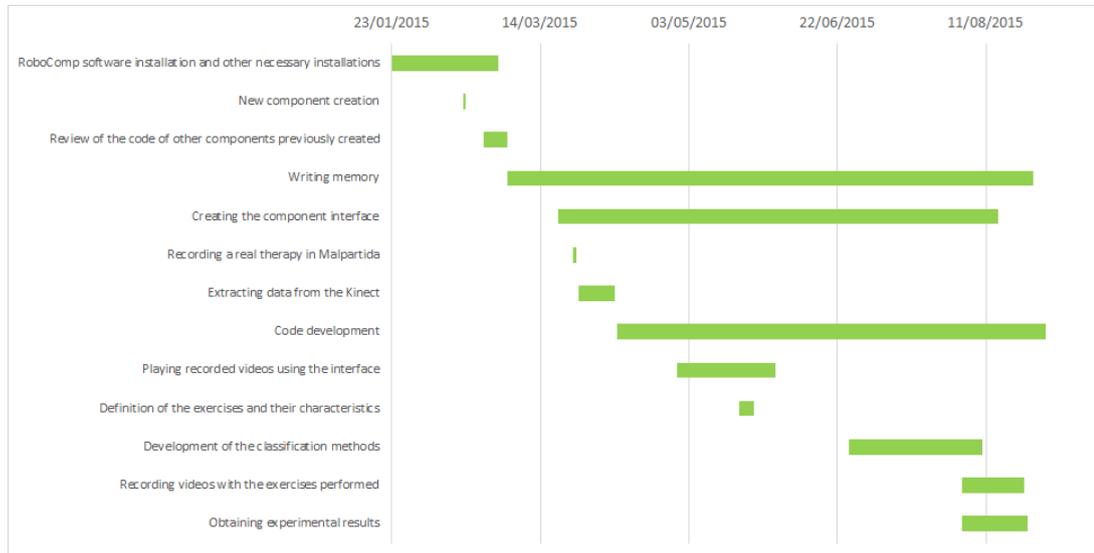


Figure 1.1: Gantt chart of tasks performed

As can be seen, the tasks in which more time has been spent was the development of code of the component, the creation of its interface and the writing of the project report. These tasks have been developed continuously throughout the project.

The task of playing the recorded videos using the Kinect has been extended a little more time than previously estimated, due to a few problems that arose. The other tasks has been performed in the timing that had been estimated for them.

This document is structured as follows: Chapter 2 presents the state of the art in therapy technologies for seniors. In Chapter 3 the hypothesis and principal and secondary targets of the project are shown. The RoboComp Software used to develop the component is described in Chapter 4. Chapter 5 talks about the created component with more detail. Chapter 6 details the results that have been obtained after carrying out the necessary tests on the component created. And finally, Chapter 7 explains the conclusions reached and the future work that could be made to continue the development of the component.

Chapter 2

State of the Art

There is an increasing level of elderly population in the current society, and this population will continue to grow in the future [1]. It is due to the improvement of the health service and therefore to a decrease of the mortality rate. Nowadays people live longer, and with a better quality of life.

“Elderly people” means those individuals who are aged 60 years or older. Additionally, in them different characteristics are manifested in the physical, social, and psychological that characterize them as such.

Increases in life expectancy involves increases in health demand of the elderly population. This part of the population has a high degree of sedentarism. An inactive lifestyle, as prevalent in Europe, is associated with a higher risk of certain illnesses or with worsening of them once present, mainly in the older population.

The ageing [2] is a continuous, universal and irreversible process which involves a series of changes related to a progressive loss of functional abilities. The deterioration associated with ageing also depends on other factors besides age, such as environmental, social and familial, but especially on the degree of stimulation received from these areas.

Physical activity [3] can be defined as all body movement produced by skeletal muscles with a waste of energy. Meanwhile, physical exercise is an activity performed in a planned, orderly, repeated and deliberate way. On the other hand, sedentism is known as non-practise of physical activity or practice it with a lower frequency of 3 times a week and/or less than 20 minutes each time.

One factor to retard ageing is continued physical activity. It has been proven that older adults who have done any regular physical activity in which all movements and activities of daily life are included, achieve better physical, mental and emotional conditions, making them healthier and with a better quality of life in relation to sedentary people. Individuals who exercise have a 50% less chance of die from premature death than those who are sedentary.

Physical abilities determine the physical condition of the person and can be improved with training. Often the physical capacity of an older adult is underestimated without be evaluated objectively. On the contrary, there is no age at which people stop responding to the stimulus of exercise. Older adults show percentage increases in their fitness levels similar to youth aged 20 to 30.

Between 60 and 80 years up to 50% of aerobic capacity can be lost. The anaerobic capacity has its peak between 20 and 25 years, decreasing human capacity to perform maximum efforts by 1% per year. Therefore, the people over 55 years old should not do physical exercises that require reaching the anaerobic phase, such as speed and strength exercises, but they should do those exercises that require dexterity, coordination and resistance.

Regular physical exercise adapted for seniors is associated with a lower risk of mortality and morbidity. The physical exercise has beneficial effects in most, if not all, of the senior physiological functions. This can be translated as better health and greater resistance to disease. In addition, psychosocial benefits of exercise fight isolation, depression and anxiety and promote self-esteem and social cohesion. Thus, physical exercise can be defined as an essential component for a healthy lifestyle.

The non-pharmacological measure most effective for most age-related diseases is exercise. Some of the benefits deriving from it are the following [4]:

- It reduces the incidence of all cardiovascular diseases in general
- It helps maintain a more adequate nutritional and metabolic balance and reduce type II diabetes
- It reduces the loss of bone mineral density and prevents the risk of fractures
- It promotes muscle strengthening

- The risk of falls is reduced, especially through the muscle strengthening and improving balance, coordination and agility
- The senior immune system is strengthened
- It reduces the incidents of certain types of cancer
- It reduces musculoskeletal pain associated with ageing
- It increases and retains cognitive function
- It protects against the risk of developing dementia or Alzheimer
- It produces an increase in physical function and consequently improvements independence and self-esteem
- It decreases the prevalence of depression, anxiety and other mental illnesses
- It promotes social cohesion and integration of older people

However, despite that physical exercise is today the main protective factor for age-related diseases, the levels of physical activity in older people are lower than in other population groups.

Besides the physical condition of the patient, improving the cognitive functions [5] is also an important task. These are all mental activities that the human realizes to interact with the surrounding environment. Throughout the life cycle, cognitive functions undergo a series of changes. Older adults suffer from a cognitive ageing that requires stimulation of the cognitive functions to prevent deterioration of these functions.

Certainly, cognitive impairment, which is any alteration of higher mental abilities (memory, judgement, abstract reasoning, concentration, attention and praxis), is a very important issue which implicitly involves a number of limitations in reference to the autonomy and quality of life of older people affected.

Older people have a high risk of losing their cognitive abilities, and this risk increases when the environmental conditions are uninspiring. Hence the importance of taking cognitive psychostimulation as a process of improving the

quality of life of people. The higher cognitive stimulation the person will have greater autonomy.

Cognitive stimulation address additional important factors within the human being, such as affective, the behavioral, social, and biological, seeking to intervene in the whole elderly person.

The optimum ageing, one to which every senior should aim, can be defined as one that occurs in the best possible condition, physically, psychologically and socially. It includes a low possibility of illnesses. This optimum ageing can be reached by the older people who improve their physical and psychological health, promoting their autonomy through physical exercise or training of cognitive abilities, improving their self-esteem, maintaining healthy lifestyles. This helps them to avoid, where possible, dependence and isolation with the environment in which they live.

The intervention of elderly population from the health field should be done not only to pharmacological or medical level, but at multidisciplinary level, since the exclusive attention of one area would be limited and ineffective.

The problem of population growth of older adults with a high degree of sedentarism leads to health problems generating high costs for social health systems. Therefore, increasing regular exercise in older people as well as realizing exercises that develop their cognitive skills would help to solve this problem. Investing in quality therapies to maintain and/or promote their functional independence, could result in considerable savings for health, public and private.

Any actions aimed at preserving health in the elderly should be directed to the maintenance of personal autonomy, aspect which would provide greater satisfaction in the older people. It would be possible to change the idea of health care for the elderly from consumption of medicines to a series of more active therapies that enhance their cognitive abilities and fitness, improving the situation in all spheres of the person and trying to achieve more independence.

As mentioned, elderly population has been growing for the last time, but at the same time as this population increases, the technology is becoming more and

more important in our daily life. This is not a problem for the young population, but it does for the elderly population. This is due to a decline in motor abilities with increasing age. Most of elderly people are afraid of new technologies, but they need to access these technologies in their everyday activities. Consequently these have to be designed according to the abilities of the elderly, with the intention of making easier and more intuitive for the elderly to interact with them.

In the past, much attention has focused on making the interaction with the technology easier for the elderly by using a mouse or a keyboard, but we now know that gesture-based applications are more appropriate for them, because are more intuitive and easy handling. In the gesture-based applications, users use body gestures, like the movements of the hands, to interact with the computer. Humans do not need to learn how to use gestures, because they learn it from childhood, but the task of learning how to use a mechanical device, such as the mouse or the keyboard, can be very frustrating for the elderly because they do not have the necessary skills. For example, they usually do not have enough hand eye coordination to move the mouse pointer [6], or they do not have enough dexterity in fingers to write on the keyboard. By using the gesture-based applications, they do not need to learn how to use an external device, which helps them to interact with the technology.

The first gesture-based applications used to use some wearable devices to capture the movements, like gloves with sensors to track hand and finger movements [7], or suits with sensors to identify full body movements [8]. In Figure 2.1 [10] we can see an example of a glove that track movements, that can be use to use the technologies [9].



Figure 2.1: CyberGlove

Also there is some devices which you do not need to put on that use an accelerometer to track the movements, and are more natural to use than typical remote controls, like the Wii Remote [11] by Nintendo [12] or the Play Move [13] by Sony [14]. In Figure 2.3 [15] we can see both, Wii Remote on the left and Play Move on the right.



Figure 2.2: Wii Remote Vs Play Move

But one of the objectives that this kind of applications pursue is to be as natural as possible, and to achieve that they need to go away from using external devices. For that, it is increasingly common devices with no sensors on the body

or controller in the hands, like Kinect [16] by Microsoft [17], EyeToy [18] by Sony or Xtion Pro Live [19] by Asus [20].



Figure 2.3: Kinect Vs Eye-Toy Vs Xtion Pro Live

So can be seen that there are many resources that enable the implementation of gesture-based applications. This gesture-based applications can be applied to robotics, because the human-robot interaction should be realized the most natural way possible, and the use of gesture-based applications and the patient's voice may help to get a great naturalness in the interaction. This applications must make feel users comfortable using a more friendly interface, which makes the deal with the robot easier for the elderly people.

The robotics is being developed in many areas, like gaming, education or medicine. A topic being developed in recent times is the robotic therapy. Robotics can provide many benefits realizing session with a patient, especially if these sessions are routine, since the novelty of a robot will give a motivating point to the patient.

Applying this to the robot, the result is a robot that can perform various therapies in a natural way for the patient, replacing or accompanying the occupational therapist.

Undoubtedly the most similar project to ours is jointly developed by the Universidad Carlos III de Madrid, the Universidad de Extremadura, the Universidad de Málaga and the Hospital Virgen del Rocío from Seville. It is a robot called NAO [21] [22], designed to perform a rehabilitation movement therapy interactively with children. Its aim is that the child perceives the therapy as a game with the robot, which looks like a toy. In this way the little patients are

encouraged to perform the exercises of the therapy.

NAO makes the exercises that kid should imitate, and he is able to perceive the patient's reactions and interact with him. It corrects the exercises when they are not properly developed, giving the necessary guidelines to improve the performance of the same. In Figure 2.4 [23] can be seen an example of this robot performing a therapy with a child.

In a few years, this robotic therapist may be a reality in the Spanish medicine and could even invigorate the process of rehabilitation of children with diseases such as cerebral palsy.



Figure 2.4: Nao Robot

Fields of application of robotics in medicine are extensive and complex. There are some more robotics applications that have already been developed. Down below we show some examples.

- **The Armeo Spring [24]:** The Armeo Spring is a device intended for patients with multiple sclerosis. It aids for the rehabilitation of upper extremities. It consists of an anti-gravity support that can be adjusted to the arm and the forearm, as can be seen in Figure 2.5 [25]. They help the gradual reorganization of the brain, which later allows restoration of motion and functionality. The motivation and self-initiated exercises include proximal and spaced components such as grasping and releasing, or flexion and extension of the wrist.



Figure 2.5: Armeo Spring Robot

- **ReoGo [26]:** This robotic device helps patients recovering from cerebrovascular accidents to regain the upper motor functions. It consists of two connected platforms: a seat with an armrest ring and a monitor that sends signals that direct the arm by a sequence of movements that fit the level of normal mobility of the arm joints (shoulder and elbow). It can be programmed in 5 different levels, depending on the patient's disability. The highest fully assists their movements automatically, while the lowest is fully guided by him. While the patient is recovering, the device allows greater autonomy in their movements.



Figure 2.6: ReoGo Robot

- **Manus robot del MIT [27]:** It is designed to help recovering from cerebrovascular accidents, same as above. The patient hold a robotic control lever that the patient guides with the arm, as Figure 2.7 [28] shows. The patient tries to do specific movements with his wrist or his hand, thereby helping the brain to make new connections that over time will help him relearn to move the limb for itself. If the person starts moving in the wrong direction or does not move, the robotic arm gently nudges his arm in the right direction.



Figure 2.7: Manus del MIT Robot

Chapter 3

Hypothesis and targets

This chapter describes the main goals that have been set in this project, and the hypothesis that have supposed to begin the research.

This project is intended to offer seniors an alternative to conventional therapy sessions, for they to be able to realize exercises that allow them to improve their skills in a more enjoyable way. It also seeks to provide a resource to occupational therapists that allows them to capture the attention of their patients with a more motivating therapy.

Therefore, it was decided to make a exercises recognition system that enable older people to make therapy with a robot playing the Simon Says game, but repeating a sequence of exercises rather than a sequence of buttons.

There are already some similar projects looking to make robots rehabilitation therapies, which shows the future of therapies with the patients. However, this project goes beyond that combining the exercises recognition and work patient memory by implementing a playful game.

From the hypothesis (Hypothesis 1) that a therapy that combines the physical development of the patient and the development of memory will be more enriching, the system has begun to develop.

The main objective of this project is the study and the development of a system that allows to play Simon Says game with the patient, by repeating a series of exercises (Objective 1). To be able to achieve this purpose, other sub-objectives has been achieved during the work development, mentioned below:

- To implement a system which recognizes the exercises performed by the patient (Objective 1.1).
- To develop an environment that enables to interact with the patient, asking to perform exercises and correcting him when they are not done correctly (Objective 1.2).
- To provide the system with a graphical interface that allows to work with the component properly, until it can be implemented in a robot (Objective 1.3).
- The creation of an useful database obtained by 3D information that stores videos from different patients performing various exercises, that can be shared and therefore used by different systems (Objective 1.4).
- The knowledge about the framework RoboComp, in which the component is developed (Objective 1.5).
- To learn about the benefits for the elderly people of performing a therapy (Objective 1.6).
- To improve the understanding and the programming ability in the language C++ (Objective 1.7).

Chapter 4

RoboComp

This chapter talks about the software RoboComp [29] [30] developed by Robo-lab. This software is used by many researches, and is applied in this project, to develop the component.

RoboComp is an open-source component-oriented robotics framework, composed by a wide variety of different components. Many components are implemented on different ways and uploaded to this framework, in such a way that developers can take advantage of code developed by others. The components of this framework can communicate with other components through public interfaces. By providing a component some inputs, it will be able to return some outputs. In this way, you can include other's component in your own programme easily, without needing waste your time in something that is already done.

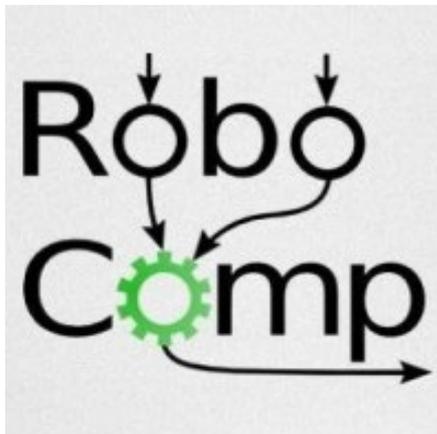


Figure 4.1: RoboComp logo

RoboComp also include the necessary tools to use the components, like the RoboComp DSLEditor, that allows you to create a new component, or like Qt4, with which the interface of a component can be built. Other tools that are used in RoboComp are CMake, IPP, OpenSceneGraph and OpenGL. These tools make programming an easier and faster task.

Communications between components are managed by the Ice framework. At first, Ice was developed by a private company, but now it is an open source and licensed by GPL. This free nature is an important reason for Ice to be chosen. Furthermore, there are other reasons to choose Ice: its efficient communication, its multiple language support (e.g. C++, Java, Python), its multiple platform support (e.g. Linux, Windows, Mac OS X), its good performance with a low overhead, its IDL-based interfaces and its different optional communication methods (e.g. RMI, AMI, AMD). Ice supports two types of communication, remote call (RPC type) or by subscription (through a service that makes messaging server).

C++ and Python are the programming languages mainly used, but thanks to the Ice framework, components written in many other languages can be easily used.

RoboComp is a quiet simple framework. The **simplicity** of the software is very important, because scalability increases as simplicity increases, and developers are interested in a scalable system.

RoboComp also provides an abstraction layer to robot hardware due to it is based in components technology. That makes it easier to use and to improve the **efficiency**.

As RoboComp allows developers to reuse components that have been developed by others, the **reusability** of the software can be highlighted.

For these reasons, it can be assumed that The main advantages of using RoboComp are its efficiency, simplicity and reusability.

Similar projects have been developed, for example Orca [31] [32], ROS [33] [34] or YARP [35], but compare with them RoboComp is more efficiency and easier to use.

The Appendix A describes the installation process of the software RoboComp.

4.1 Component Oriented Programming (C.O.P.)

It is very common in the field of robotics that researches implement their algorithms for a specific robot. That makes quite difficult reuse the created code for other project when is needed, and it is more practical to start creating a new algorithm from zero (due to dependencies and side effects resulting from its stiffness).

The Object Oriented Programming (O.O.P.) [36] implied a major progress on structure programming. It is based on the concept of ‘objects’, which are data structures that contain data and code. The programs generated in Object Oriented Programming, are designed by making them out of objects that interact with one another. It combines several related classes under a single interface. However, when the number of classes and their interdependencies increases, the complexity of the system increases too, and the overall system becomes too difficult to understand.

The purpose of the Component Oriented Programming is to solve these problems. It provides a greater degree of encapsulation. To further understand, the Component Oriented Programming is like dividing the software design in smaller parts, which are provided by the same interface. This help us to understand the system with less detailed because you do not need to know how a component is implemented to be able to use it. The interfaces hide the components, so it does not mind if the component is complex or simple, because all you can see is the interface offered. We can understand it better if we see Figure 4.2, in which we can see a representation of components with their interfaces.

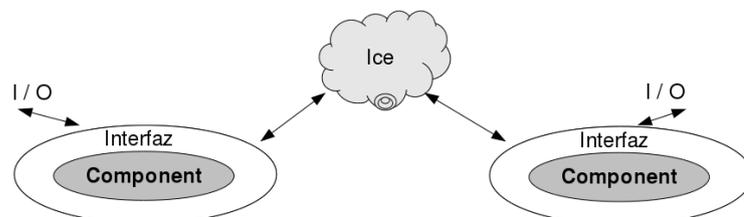


Figure 4.2: Representation of components with their interfaces

New components can be easily integrated with existing ones, because a component offer one or more outputs to the component which orders it. So if

4.2 Therapy: my component in RoboComp

The component created and developed in this project is called therapyComp.

This component does not collect data directly from the kinect, but requires as input a file wherein the recording made with the kinect is saved. This makes that program development is not conducted in real time, but it is divided into two phases: the recording images and the exercise recognition.

To record these images two other components are used, as it can be seen in Figure 4.4.

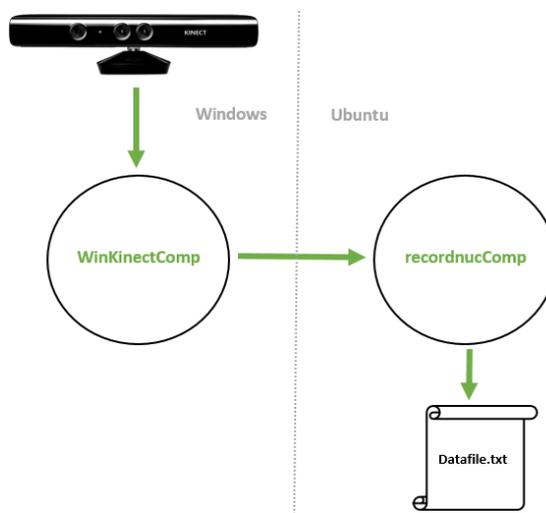


Figure 4.4: Data collection scheme with Kinect sensor

The first of this is called WinKinectCom and it runs on Windows, where the kinect is connected. This component is responsible for reading data from the kinect, and passing the data to the second component, called recordnucComp, run on Ubuntu. This second component stores in a text file the positions (length, high and depth) of the different joints of the patient, on the basis of the data collected by the kinect by the WinKinectComp in consecutive time instants.

An example of a stored data file can be seen in Figure 4.5, which shows the way the data is stored, being the first vale saved the time instant, the second value is an user identifier, and the following values are the joint identifier and its position (x, y, z) . These values are separated from each other by a #.

Análisis de imágenes de profundidad en terapias de rehabilitación supervisadas por robots autónomos

0#4732#HipCenter#0.246304#-0.433463#1.96701#Spine#0.243336#-0.372713#2.02344#ShoulderCenter#0.255561#-0.0173283#2.02783#Head#0.263498#0.173734#1.97226#
100#4732#HipCenter#0.245855#-0.433795#1.96728#Spine#0.242981#-0.373069#2.02381#ShoulderCenter#0.255002#-0.0169371#2.02915#Head#0.26310#0.174352#1.9738#
200#4732#HipCenter#0.244895#-0.434577#1.9679#Spine#0.242002#-0.373848#2.02463#ShoulderCenter#0.252839#-0.016273#2.03179#Head#0.26305#0.174641#1.9747#
300#4732#HipCenter#0.243809#-0.43532#1.96836#Spine#0.240707#-0.374532#2.02528#ShoulderCenter#0.249967#-0.0157696#2.0330#Head#0.262822#0.175846#1.9772#
400#4732#HipCenter#0.243257#-0.435493#1.9691#Spine#0.239944#-0.374646#2.02617#ShoulderCenter#0.248986#-0.0151072#2.0345#Head#0.262291#0.177401#1.98067#
500#4732#HipCenter#0.242825#-0.435425#1.96954#Spine#0.239291#-0.374575#2.02669#ShoulderCenter#0.24863#-0.01483#2.03541#Head#0.261832#0.178037#1.98231#
600#4732#HipCenter#0.243309#-0.435549#1.97021#Spine#0.239403#-0.374705#2.02734#ShoulderCenter#0.247731#-0.0148389#2.0362#Head#0.261348#0.17852#1.9845#
700#4732#HipCenter#0.242383#-0.435418#1.97127#Spine#0.238464#-0.374554#2.02846#ShoulderCenter#0.247562#-0.0145713#2.03753#Head#0.260696#0.178656#1.9856#
800#4732#HipCenter#0.243004#-0.434823#1.97149#Spine#0.239121#-0.374133#2.02838#ShoulderCenter#0.248053#-0.0140874#2.03883#Head#0.260503#0.179291#1.9874#
900#4732#HipCenter#0.242863#-0.434651#1.97224#Spine#0.238861#-0.374136#2.02971#ShoulderCenter#0.245309#-0.0141522#2.04046#Head#0.258604#0.178296#1.9893#
1000#4732#HipCenter#0.242837#-0.434776#1.9733#Spine#0.238589#-0.374161#2.03073#ShoulderCenter#0.245801#-0.0140242#2.04134#Head#0.257125#0.177103#1.9911#
1100#4732#HipCenter#0.24239#-0.434667#1.97359#Spine#0.239081#-0.374182#2.03118#ShoulderCenter#0.246481#-0.0138848#2.0428#Head#0.255207#0.176582#1.9922#
1200#4732#HipCenter#0.240264#-0.435024#1.97419#Spine#0.237751#-0.374361#2.03178#ShoulderCenter#0.247073#-0.0138281#2.04328#Head#0.254822#0.176555#1.9923#
1300#4732#HipCenter#0.239773#-0.435109#1.97452#Spine#0.237661#-0.374405#2.03222#ShoulderCenter#0.247528#-0.0135904#2.04384#Head#0.254242#0.176791#1.9931#
1400#4732#HipCenter#0.239248#-0.435064#1.97474#Spine#0.237333#-0.374329#2.03246#ShoulderCenter#0.247623#-0.0134706#2.04393#Head#0.25381#0.176962#1.9932#
1500#4732#HipCenter#0.23911#-0.434998#1.97475#Spine#0.237432#-0.374284#2.03249#ShoulderCenter#0.247998#-0.013401#2.04394#Head#0.253007#0.17701#1.99323#
1600#4732#HipCenter#0.240272#-0.434557#1.97462#Spine#0.238421#-0.373983#2.03246#ShoulderCenter#0.248222#-0.0132693#2.04406#Head#0.252537#0.176881#1.9923#
1700#4732#HipCenter#0.240937#-0.434634#1.97443#Spine#0.239098#-0.374042#2.03235#ShoulderCenter#0.248425#-0.0131765#2.04409#Head#0.252367#0.176923#1.9923#
1800#4732#HipCenter#0.240598#-0.435253#1.9743#Spine#0.238078#-0.374548#2.0322#ShoulderCenter#0.24671#-0.0142834#2.04317#Head#0.252283#0.17663#1.99279#
1900#4732#HipCenter#0.240543#-0.435471#1.97413#Spine#0.238438#-0.374743#2.0319#ShoulderCenter#0.2444#-0.0155369#2.04159#Head#0.251947#0.176104#1.9904#
2000#4732#HipCenter#0.241178#-0.435478#1.97351#Spine#0.238887#-0.374787#2.0313#ShoulderCenter#0.244906#-0.0158157#2.0406#Head#0.252115#0.1757#1.98877#
2100#4732#HipCenter#0.24195#-0.435147#1.97261#Spine#0.23953#-0.374449#2.03028#ShoulderCenter#0.247578#-0.016151#2.03904#Head#0.252364#0.175837#1.98694#
2200#4732#HipCenter#0.242659#-0.435056#1.97177#Spine#0.240451#-0.374558#2.0296#ShoulderCenter#0.247385#-0.0163206#2.03746#Head#0.252753#0.175514#1.9842#
2300#4732#HipCenter#0.242944#-0.435058#1.97144#Spine#0.240616#-0.374338#2.02894#ShoulderCenter#0.247157#-0.0165542#2.03655#Head#0.25354#0.17478#1.98312#
2400#4732#HipCenter#0.244665#-0.434434#1.97032#Spine#0.242213#-0.373932#2.02774#ShoulderCenter#0.248931#-0.0166762#2.03487#Head#0.255142#0.174215#1.9812#
2500#4732#HipCenter#0.24535#-0.43455#1.96935#Spine#0.242973#-0.374074#2.02659#ShoulderCenter#0.250391#-0.0175973#2.03297#Head#0.256757#0.173142#1.97847#
2600#4732#HipCenter#0.24616#-0.434191#1.96845#Spine#0.24364#-0.373778#2.02556#ShoulderCenter#0.251261#-0.0180246#2.0314#Head#0.258555#0.173733#1.97623#
2700#4732#HipCenter#0.24626#-0.434583#1.96761#Spine#0.243908#-0.373927#2.02455#ShoulderCenter#0.253516#-0.0183567#2.02875#Head#0.259788#0.173693#1.9734#
2800#4732#HipCenter#0.246286#-0.434518#1.96712#Spine#0.244081#-0.373784#2.0239#ShoulderCenter#0.254655#-0.0182515#2.02761#Head#0.260637#0.173551#1.9712#
2900#4732#HipCenter#0.247349#-0.434364#1.96589#Spine#0.245217#-0.37359#2.02256#ShoulderCenter#0.256211#-0.0182612#2.02591#Head#0.261738#0.1737#1.9704#
3000#4732#HipCenter#0.248585#-0.434091#1.96485#Spine#0.246366#-0.373211#2.02142#ShoulderCenter#0.256906#-0.017534#2.02444#Head#0.262428#0.173778#1.9687#
3100#4732#HipCenter#0.248865#-0.433446#1.963#Spine#0.246512#-0.372621#2.01954#ShoulderCenter#0.256857#-0.0173622#2.02226#Head#0.263368#0.17311#1.96673#
3200#4732#HipCenter#0.248491#-0.433264#1.96169#Spine#0.246103#-0.372585#2.01836#ShoulderCenter#0.255064#-0.0170647#2.02151#Head#0.263875#0.173631#1.9655#
3300#4732#HipCenter#0.249227#-0.432476#1.95993#Spine#0.246826#-0.370735#2.01643#ShoulderCenter#0.258747#-0.0164893#2.01885#Head#0.265567#0.17452#1.9641#
3400#4732#HipCenter#0.250493#-0.431932#1.95719#Spine#0.247986#-0.370735#2.01384#ShoulderCenter#0.263238#-0.0126897#2.01755#Head#0.27189#0.179106#1.9613#
3500#4732#HipCenter#0.252524#-0.431281#1.95427#Spine#0.248912#-0.36947#2.01068#ShoulderCenter#0.264465#-0.00930241#2.01384#Head#0.278093#0.181264#1.9613#
3600#4732#HipCenter#0.254672#-0.43028#1.95184#Spine#0.249964#-0.368014#2.00794#ShoulderCenter#0.267261#-0.0069153#2.01051#Head#0.285086#0.184038#1.9581#
3700#4732#HipCenter#0.256403#-0.429269#1.94978#Spine#0.25099#-0.366868#2.00573#ShoulderCenter#0.268757#-0.0058044#2.00796#Head#0.287702#0.185575#1.9563#
3800#4732#HipCenter#0.256985#-0.428704#1.94774#Spine#0.251217#-0.366402#2.0036#ShoulderCenter#0.269243#-0.00721207#2.00524#Head#0.286685#0.184554#1.9523#
3900#4732#HipCenter#0.257025#-0.42915#1.94697#Spine#0.251515#-0.36712#2.00272#ShoulderCenter#0.26886#-0.0107604#2.00308#Head#0.281492#0.17834#1.94548#
4000#4732#HipCenter#0.257118#-0.43000#1.94649#Spine#0.251933#-0.368298#2.00217#ShoulderCenter#0.267571#-0.0134483#2.00195#Head#0.27870#0.175631#1.9411#
4100#4732#HipCenter#0.257328#-0.430785#1.9461#Spine#0.252516#-0.369282#2.00167#ShoulderCenter#0.26881#-0.0156782#2.00042#Head#0.277516#0.173162#1.9391#
4200#4732#HipCenter#0.257837#-0.431421#1.94507#Spine#0.253908#-0.370169#2.00074#ShoulderCenter#0.267724#-0.0170843#1.99974#Head#0.27565#0.171572#1.9373#
4300#4732#HipCenter#0.257526#-0.432097#1.94487#Spine#0.254372#-0.370895#2.00061#ShoulderCenter#0.267969#-0.0176763#1.99971#Head#0.274056#0.169177#1.9374#
4400#4732#HipCenter#0.257265#-0.432055#1.94567#Spine#0.254597#-0.370859#2.0013#ShoulderCenter#0.26771#-0.0177462#1.99997#Head#0.271843#0.170486#1.94117#
4500#4732#HipCenter#0.257251#-0.43156#1.94733#Spine#0.25485#-0.370392#2.00314#ShoulderCenter#0.267904#-0.0161309#2.00296#Head#0.270795#0.174897#1.94784#

Figure 4.5: Example of a data file saved

Once the videos have been recorded and saved in data files, they can be processed by the component therapyComp. For that, it is necessary to specify in the code the path and file name that will be read.

```
void SpecificWorker::ReadFromFile() {  
    int i;  
    int idModel;  
    RoboCompMSKBody::TPerson p; //Definida en MSKBody  
    RoboCompMSKBody::SkeletonPoint pose;  
    RoboCompMSKBody::Joint joint;  
    QStringList dataList;  
    QString data = "";  
  
    //If the file wasn't open in the previous interaction, this section open the file and read the first  
    if (reading==false) {  
        reading = true;  
  
        file.setFileName ("/home/computaex/robocomp/components/mycomponents/therapyComp/data/moves.txt");  
  
        if (!file.open (QIODevice::ReadOnly | QIODevice::Text)) qDebug() << "Error opening the joints file";  
  
        in.setDevice(&file);  
  
        data = in.readLine();  
    }
```

Figure 4.6: Part of the code to change the file path

The therapyComp visually represents by its interface a model of person moving following the movements that is reading from the file, as can be seen in Figure 4.7. For get this model of person, it communicates with the innerModel-Manager component.

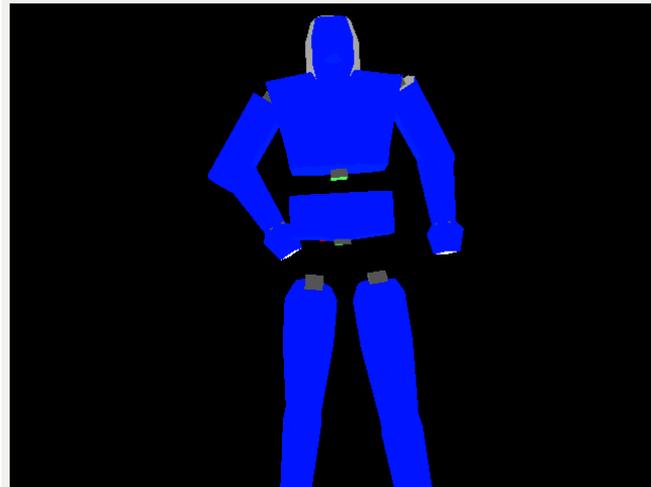


Figure 4.7: Model of a person

In addition, to displaying on screen the data read from the file, the therapy-Comp process these data and introduce them in one of the classification methods described in section 5.11. Those decision models are all defined by the OpenCv [37] library, that is a free computer vision library originally developed by Intel. The output obtained after processing these data is the exercise with a higher probability of being executed by the patient.

Chapter 5

The component and the therapy

This chapter describes with a high detail level the component developed. It is composed by the explication of the game of Simon says, the description of the exercises that have been realised in the therapy.

5.1 Simon Says

The Simon Says [38] is a traditional game special for children, in which the players should do what Simon, that is one of the participants, says.

In the decade of the XX, other version of the play Simon Says emerged. This new version is played with an electronic device with four buttons of different colours (yellow, blue, green and red) which marks a colours and sounds sequence, that the players should remember and reproduce by pushing the different buttons of the device. That electronic device can be seen in Figure 5.1.



Figure 5.1: Simon Says Game

In other words, if at the beginning Simon lights up a button, for example the yellow button, the player should push the lighted button, in that case the yellow button. In the next game turn, Simon will start lighting the button lighted in the previous turn, the yellow button, and after that it will light other button, for example the green button, henceforth the player should repeat that yellow and green sequence. As the turns played increases, the difficulty levels increases too, due to the increases the number of the buttons that came into the sequence, and that the player should remember and reproduce.

Therefore, this game develops the player's memory, because of the player needs to remember the sequence that Simon is marking.

In our version of the Simon Says game, instead of a sequence of pushing buttons, the patient has to reproduce a sequence of the different exercises. Henceforth our version of the game helps to develop as the patient's psychomotricity as the memory of the same one.

5.2 RGB-D sensor

The sensor used to get video data has been the kinect sensor for Windows [39].

The Kinect sensor, developed by Microsoft, is a motion sensing input device. It allows to interact with the video game console or the computer with a natural interface using gestures and spoken commands, and for that, without the necessity of use controller devices. That makes the use of applications more interesting for the user.

Kinect, also known as 'Project Natal', was first launched on November 10, 2010 on Europe. In the beginning it was designed for Xbox 360, but researches became interested on this device, seeing the advantages it can offers to the new technologies world, and they started to use Kinect to create new applications on different fields, like entertainment, healthcare or education. To provide the opportunity to work with this sensor creating different applications, Microsoft decided to launch a version of Kinect for Windows and a non-commercial Kinect SDK (Software development Kit) on February 1, 2012. On July 15, 2014, released a new version of Kinect, called Kinect 2.0, designed for the new video console

Xbox One, and its equivalent for Windows.

The Kinect Sensor for Windows combines hardware, software and a commercial license, and it allows applications running Windows to extract RGB-D information. The Kinect sensor can be seen in Figure 5.2 [40].



Figure 5.2: Kinect sensor for Windows

This device enables to sense the user's voice and movements. It has depth sensing technology, what gives this sensor the ability to work with depth information, which is very important to process 3D images. It incorporates an infrared (IR) emitter, a microphone array and a built-in color camera.

The Kinect [41] creates a points pattern invisible for the user generated by a laser working in the near infrared frequency, which does not generate dangerous conditions for users. A CMOS sensor detects infrared beam reflections generating the image from the different intensities of each point (Depth Image). A second CMOS sensor is used to add color to the acquired data (RGB Image). These sensors inside the Kinect can be seen in Figure 5.3.

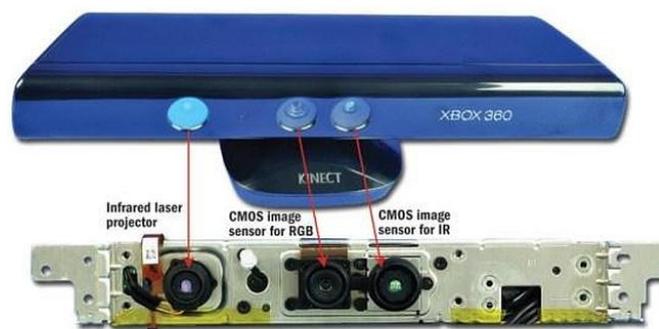


Figure 5.3: Sensors included in the Kinect

The Kinect for Windows SDK provides the researchers drivers, tools, APIs, device interfaces, and code samples with the purpose to makes easier for the developers to create Kinect-enabled applications.

The Kinect has its own reference system [42] in which the source of the reference center is the center of the camera, as can be seen in Figure 5.4. This must be taken into account when working with it.

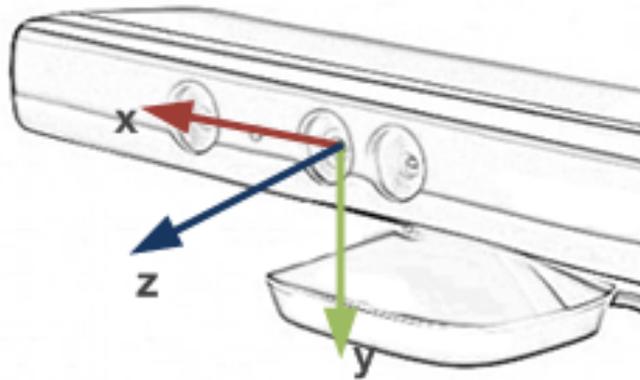


Figure 5.4: Reference System of the Kinect

- The X axis is parallel to the longer side of the device.
- The Y axis is vertical, and considers the positive values to those who are below the device.
- The Z axis is perpendicular to the device.

The spatial resolution (X and Y axis) is of the order of millimeters, while the resolution of the depth dimension (Z axis) is up to 1 centimeter. The specifications for the Kinect [43] can be seen in Table 5.1.

Feature	Kinect for Windows
Color image resolution (pixels)	640 x 480
Field of view H x V (degrees)	62 x 48.6
Average (pixels per degree)	10 x 10
Depth image resolution (pixels)	320 x 240
Field of view (degrees)	58.5 x 46.6
Average (pixels per degree)	5 x 5
Dimension (cm)	27.9 x 7.6 x 7.6
Depth operation range (m)	0.8 - 3.5
Skeleton Joints Defined (joints)	20
Full Skeletons Tracked	2
Suported OS	Win 7, Win 8

Table 5.1: Kinect specifications

A new class of more natural and intuitive applications is being developed. With more than a million downloads so far of the original Kinect for Windows SDK, the use of the Kinect for Windows has been increased, and it is being used by a lot of researches even more often in different fields. Kinect allows a more seamless interaction for the people with the computers, using gestures and voice.

5.3 Exercises

The exercises that have been selected for the recognition process have been chosen by a occupational therapist that has collaborated with our project.

This exercises are especially designed for elderly people because of its low difficulty level.

The first exercise selected for the therapy is to cross the arms. To the correct realised of the exercise, it is important that the hands are at the equal level of the shoulder. This exercised is shown in the Figure 5.5.

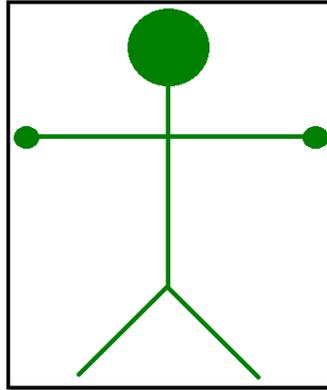


Figure 5.5: Exercise 1

The second exercise selected is about to put the hands in the hips, with arms akimbo, as we can see in Figure 5.6.

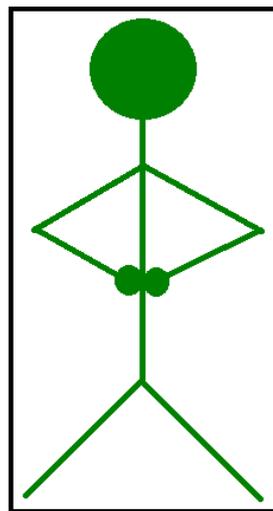


Figure 5.6: Exercise 2

The third therapy exercise is to hold the arms out in front of the body, in such a way that the hands stand in the shoulders level. The Figure 5.7 shows that exercise.

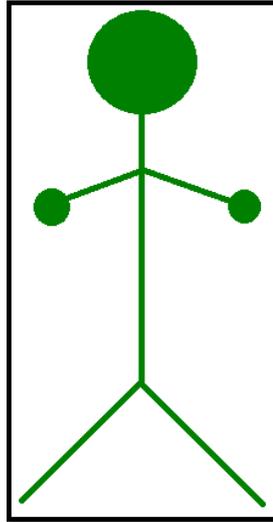


Figure 5.7: Exercise 3

The fourth and last exercise selected is to raise the hands over the head, with the arms outstretched. This exercised can be seen in the Figure 5.8.

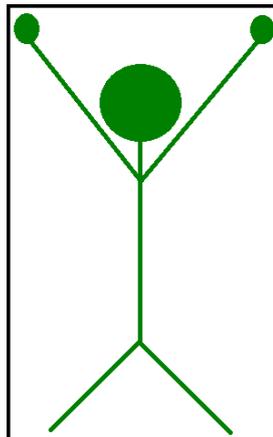


Figure 5.8: Exercise 4

The other recognising state is the ‘non defined exercise’, which includes when the patient are doing any other movement that are not one of the above defined.

5.4 Features

The features that have been calculated for the component development are listed below.

5.4.1 Quantity of motion (QoM)

The quantity of motion (QoM) is a measure of the amount of detected skeleton motion.

This feature is useful to know when the patient is moving and discriminate the data collected at that time, taking only the position data when the patient is at rest. Therefore, only when the QoM value is below a threshold the data is taken into account. It is interesting to do this because the exercises performed in therapy are static exercises, so it is assumed that when the patient is not at rest it is that it is not doing any exercise from the therapy, but it is transitioning between two exercises.

It is calculated as the sum of the displacements between one frame and the next of the six points with a more significant movement (*i.e.*, left and right hands and elbows, chest and head), divided by the total number of frames that were taken into account.

Let N being the total number of consecutive frames taken, and x_i^A being the 3D position of an articulation at an instant of time i . Then, QoM^A is defined as:

$$QoM_i^A = \frac{1}{N} \cdot \sum_{k=0}^N x_i^A - x_{i-1}^A \quad (5.1)$$

where $A \in (left\ hand, right\ hand, left\ elbow, right\ elbow, chest, head)$. Finally, the total QoM is evaluated as the average value of QoM^A .

As you can be derived from the formula (5.1), the number of samples that are taken into account greatly influence the final outcome. For the development of this component has been chosen a small number of samples, so that the calculated QoM takes values instantaneously as possible.

5.4.2 Contraction Index (CI)

The contraction index (CI) measures the amount of contraction and expansion of the body with respect to its centroid. The CI takes values ranging between 0 and 1.

To calculate CI value, we are focused in the relationship of the chest and the hands position, as can be seen in Figure 5.9. This relationship has been carried

out by the area of the triangle defined by the X and Y coordinates of these three points.



Figure 5.9: Triangle to calculate CI

First, the semiperimeter, s of the triangle is calculated:

$$s = \frac{u + v + w}{2} \quad (5.2)$$

where u , v and w are the sides of the triangle.

The Contraction Index, using the Heron's Method, remains as follows:

$$CI = \sqrt{s \cdot (s - u) \cdot (s - v) \cdot (s - w)} \quad (5.3)$$

The CI value is bigger when the patient's posture keeps limbs near the baricenter. Therefore, the expected CI in the various exercises will be:

Exercise	CI Value
Exercise 1	High
Exercise 2	Low
Exercise 3	Low
Exercise 4	Medium

Table 5.2: CI values for the different exercises

5.4.3 Angle Arm (α)

The angle arms α is a feature that will give us information on whether the arms are stretched or bent.

To calculate this feature, a triangle whose sides are listed below was defined, as Figure 5.10 [44] shows.

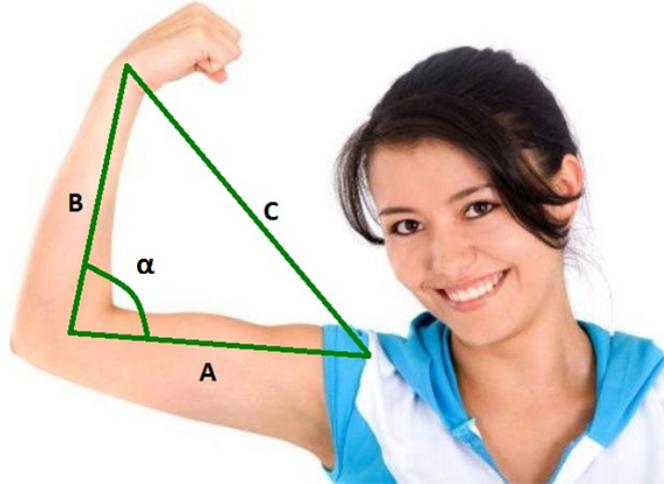


Figure 5.10: Triangle to calculate the Angle Arm

- Side A: length from the shoulder to the elbow
- Side B: length from the elbow to the hand
- Side C: length from the shoulder to the hand

As hands, elbows and shoulders positions are known, these lengths can be easily calculated and therefore the three sides of the triangle formed are equally known. Hence, the desired angle can be calculated by the law of cosines, which reads as follows:

$$A^2 = B^2 + C^2 - 2BC \cos(\alpha)$$

$$\cos(\alpha) = \frac{B^2 + C^2 - A^2}{2BC} \quad (5.4)$$

$$\alpha = \arccos\left(\frac{B^2 + C^2 - A^2}{2BC}\right)$$

α calculation will be made for both arms.

The values of the angles of the arms for the different exercises will be approximately as the Table 5.3 shows.

Exercise	$\alpha_{RightArm}$	$\alpha_{LeftArm}$
Exercise 1	180°	180°
Exercise 2	90°	90°
Exercise 3	180°	180°
Exercise 4	180°	180°

Table 5.3: α values for the different exercises

As can be seen in Table 5.3, the α value discriminates between exercise 2 and other exercises, but actually it provides more information because it helps us to know when an exercise is not being done properly.

5.4.4 Height of the hands (Y)

This feature shows the height at which the hand is.

As the reference axis is located in the Kinect instead of on the subject there is the problem that the same altitude data may not be the same, it depends on the position in which is placed the Kinect to record. To solve this, it has been assumed that the ground is where the subject's foot and the axis has been focused subtracting the value of ground level from the value of the hand height, as can be seen in Figure 5.11.

$$Y = Hand.Y - Foot.Y \quad (5.5)$$

That has been made for other body parts too, to change its reference axis to the new one, as can be seen in Figure 5.11 [45].

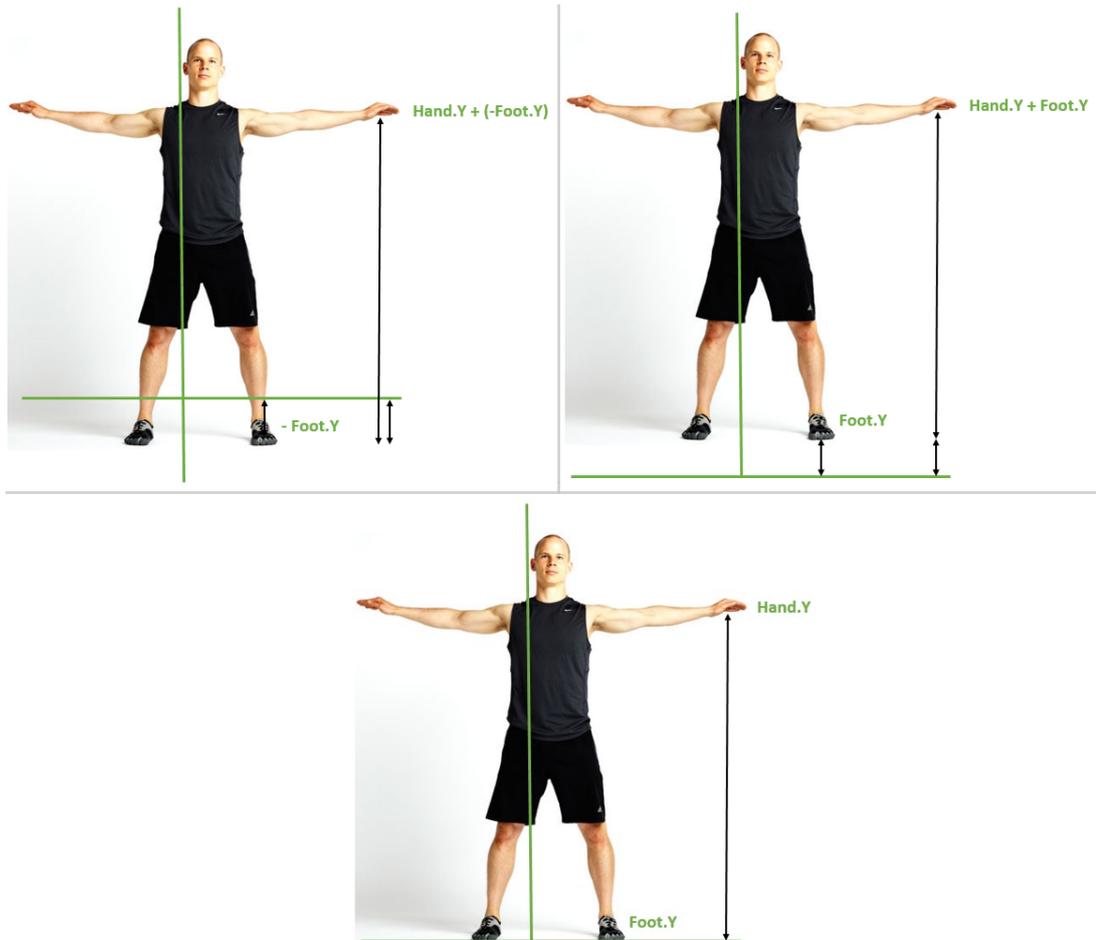


Figure 5.11: Exes Change

Once the axis is centred as desired, the height value of the hand reflects the real value. But as each person has a different height, their hands will be different too. Therefore, we normalize these values to the maximum value that could take the hand height so that all values are between 0 and 1 and comparison is more simple.

The maximum value that can take the hand is calculated from the value of arm's length plus the shoulder's height, as can be seen in Figure 5.12 [46].



Figure 5.12: Maximun Height

Thus, the arm's length is calculated first, as shown in (5.6). For the equation not be too long, we are going to define $H.Y = Hand.Y$, $E.Y = Elbow.Y$ and $S.Y = Shoulder.Y$.

$$Length_{ElbowHand} = \sqrt{(E.X^2 - H.X^2) + (E.Y^2 - H.Y^2) + (E.Z^2 - H.Z^2)}$$

$$Length_{ShoulderElbow} = \sqrt{(S.X^2 - E.X^2) + (S.Y^2 - E.Y^2) + (S.Z^2 - E.Z^2)}$$

$$ArmLength = Length_{ElbowHand} + Length_{ShoulderElbow} \quad (5.6)$$

After that, the maximum height that can be the hand is calculated, as can be seen in (5.7).

$$Y_{max} = ArmLength + Shouder.Y \quad (5.7)$$

Since both the ArmLength value as Shoulder.Y value are constant throughout the recognition process, Y_{max} will be constant too.

Finally, the hand height (Y) can be normalized in the manner shown in (5.8).

$$Y = \frac{Hand.Y}{Y_{max}} \quad (5.8)$$

It is interesting to calculate this, because in Exercise 1 hands must be at shoulder height, in Exercise 2 and 3 hands should be at the hips height and, in Exercise 4, the hands must be above the head. This is reflected in Table 5.4.

Exercise	$Y_{RightHand}$	$Y_{LeftHand}$
Exercise 1	Right Shoulder Height	Left Shoulder Height
Exercise 2	Right Hip Height	Left Hip Height
Exercise 3	Right Shoulder Height	Left Shoulder Height
Exercise 4	Over Head	Over Head

Table 5.4: Height values for the differents exercises

5.4.5 Depth of the hands (D)

The depth level of the hands reports whether the hands are in the z plane of the body or not.

To calculate this feature, the difference between the depth value of hands and the depth value of the chest has been computed, as is shown in (5.9).

$$D = |Hand.Z - Chest.Z| \quad (5.9)$$

This calculation is performed for both hands, the expected values can be seen in the following table:

Exercise	$D_{RightHand}$	$D_{LeftHand}$
Exercise 1	0	0
Exercise 2	0	0
Exercise 3	Arm length	Arm length
Exercise 4	0	0

Table 5.5: Depth values for the different exercises

As Table 5.5 shows, the expected value for D in Exercise 3 is the length of the arm. This is because in this exercise the arms are stretched forward, while in the rest of exercises the arms are in the same plane of the body. As it happened with α value, this feature provides information on whether the exercise 3 or any of the other defined exercises are being performed, or none of the above if it does not match any of the expected values.

5.5 Interface

The Component Interface is presented in this section.

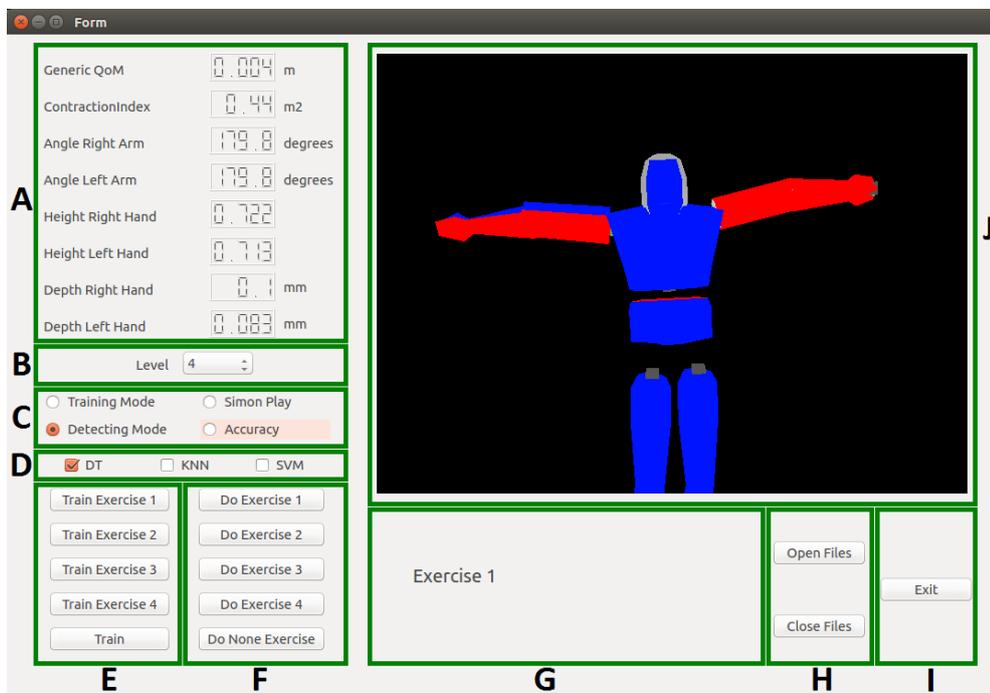


Figure 5.13: Component Interface

As is shown in Figure 5.13, the interface is composed by several parts that are detailed below:

- **Part A:** This block shows the different calculated features (QoM, Contraction Index, the Arms Angle, the Hands Height and the Hands Depth) at each instant of time.
- **Part B:** It allows to choose the level of play in the “Simon Mode”.

- **Part C:** This part enables to chose the mode wherein the component is developed. The “Training Mode” is for training the component, the “Detecting Mode” is for detecting the exercises that the patient is performing, and the “Simon Mode” is for playing the Simon Game with the component.
- **Part D:** It lets you select the decision method to be used.
- **Part E:** These buttons allow to train the component. The first four buttons save the data of a particular exercise in the training file and its corresponding label in the labels file, depending on which button is pushed.
- **Part F:** These buttons simulate the performance of one of the defined exercises, so pressing these buttons you can play the Simon Game.
- **Part G:** This label shows different messages depending on depending on which mode is selected.
 - In the “Training Mode” it shows nothing.
 - In the “Detecting Mode” it shows the number of the exercise that is being realized.
 - In the “Simon Mode” it shows the game level, the exercise realized and the next exercise to realize, and a phrase that indicates whether a level is passed or if an exercise is correctly made. If an exercise is performed properly, it displays a message that asks the patient to keep it for a while.
 - In the “Accuracy Mode” it shows a message while the accuracy is being calculated to tell that the calculation is in process. Once the calculation has has been finished it shows the resulting accuracy, and a message indicating that the process has ended.
- **Part H:** These buttons stores variables and positions data in a Matlab file to generate graphs.
- **Part I:** That button successfully stops the program.
- **Part J:** This screen shows the patient performing the different exercises.

5.6 DataBase

A database with recorded data for the training and recognition process has been created. This database consists of data of 20 people between 20 and 50 years that have been recorded with the Kinect. For each subject the following files have been stored:

- Text files of videos recorded by the Kinect of the subject performing a sequence of the defined exercises, lasting 10 seconds per exercise.
- A file text similar to the previous one but with a duration of 20 seconds per exercise.
- A text file with the features data calculated for each exercise (Exercise1, Exercise2, Exercise3 and Exercise4).
- A text file with the feature data calculated in the different exercises collected in the training process, to calculate the accuracy of the methods.
- A file that holds the labels corresponding to the exercises stored in the above file.

All files that compose the database are in text format (.txt). The database is structured as the Figure 5.14 shows.

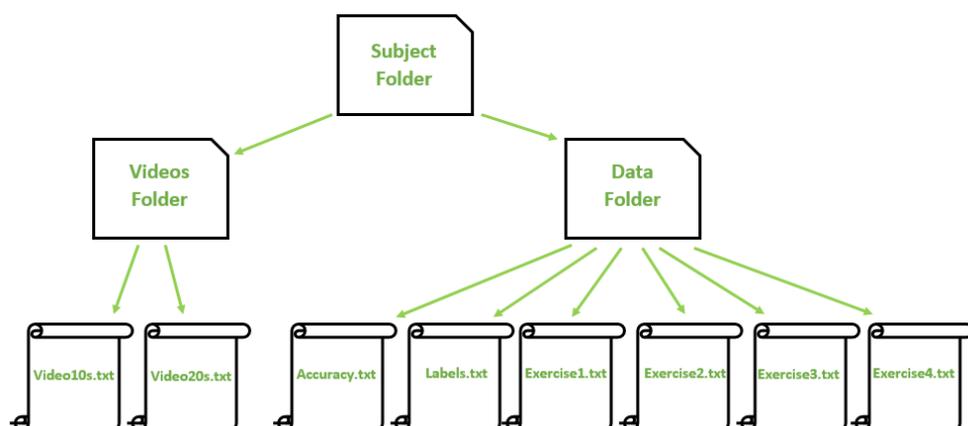


Figure 5.14: Database Organization

This database has been shared and it is available through the download service of the Universidad de Extremadura, in the link <http://descarga.unex.es//ref.php?ref=emogenac.2f0221c57c7eb0dfcacf4530eae95ddf>.

5.7 Training Mode

The training process is essential, since the decision methods need to learn before they can make decisions. That is to say, it is necessary to give them some benchmarks so they have something to compare, and they would be able to make decisions in the future. If the training process is not realized properly, these decision methods will not work correctly.

To carry out the training process, 15 different people have been recorded performing a sequence with the exercises explained in section 5.3, with a duration of 10 seconds per exercise. 5 of these people recorded were used to make the training process.

Once these videos have been recorded, they have been processed one on one with our component for the training process.

When each video was being processing, it is specified by the interface buttons above mentioned what type of exercise was being undertaken by the patient at all times, thereby saving the data for each exercise in text files as follows:

- The text file “Training.txt” saves the characteristics that have chosen to discriminate between the different exercises, corresponding a line to each instant of time that button has been hit.
- The text file “ExerciseX.txt” saves the data of a particular exercise in the same file. This has been done to facilitate data storage in database.
- The text file “Labels.txt” saves a label relative to the exercise performed at each instant of time, depending on which button we have pressed.

The Figure 5.15 shows an example of training and labels files.

5.8 Detecting Mode

The recognition process is performed after having carried out the training process. If the training process is not realized in a proper way, component will ask us to come back to the “Training mode” to train the component before making any recognition.

Once training has been performed, we can change to the “Detecting Mode”, wherein the “Detect” button can be pushed. Pressing this button the component will call the decision method, and it will start recognizing the exercises the patient is performing in the video is running at the moment. The result will be shown at the screen.

5.9 Simon Game Mode

This Mode allows to play the Simon game with the component, except that as it has not been able to do the exercises recognition in real time because of the availability of the Kinect, it has been established a series of buttons on the interface that simulates having made one of the exercises defined or a not defined one.

When selecting this mode, the component asks us to perform one exercise set, then:

- If the exercise performed (the button pressed) is one that has been asked by the component, it reports that it was the proper exercise and asks the patient to keep it for a while, and then he passes the level.
- If the exercise realized has not been asked by the component, it notifies that the exercise is not correct and that the patient have to try it again.

In the Figure 5.16 can be seen an example of the interface when the game is underway.

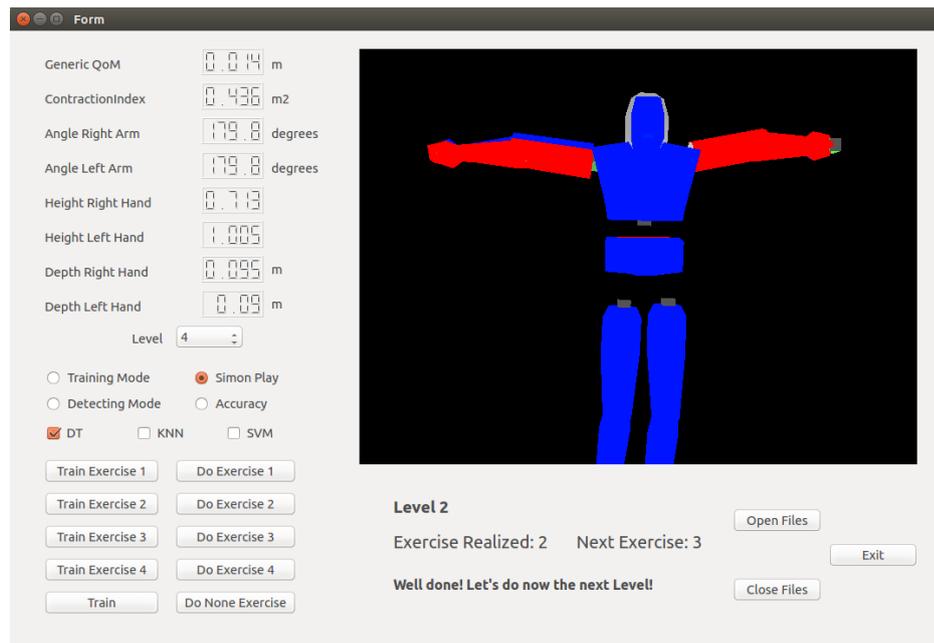


Figure 5.16: Interface in the Simon Mode

As the patient does the exercises the component ask him, he will level up. At the next level the component asks the patient to perform a new exercise, but first he will to perform the above exercises in the same order, and finally the new one.

For example, if the component first asks the patient to realize the Exercise 2 and it is performed successfully, the patient will pass the level. In the next level the component will ask to realize a new exercise, like the Exercise 4, so the patient will have to do the Exercise 2 and then the Exercise 4. If in the next level the component asks to perform the Exercise 1, the patient will have to realize the Exercise 2, the Exercise 4 and finally the Exercise 1.

In this way, as the patient goes up in level increases the difficulty of the game. The maximum level can be chosen depending on the patient's needs.

To ensure the realization of all the exercises equally, they can not be repeated in blocks of 4. That is, the first 4 exercises have to be 1, 2, 3 or 4 without being able to repeat any of these, and likewise the following 4 must be again the four different exercises without repeating.

5.10 Accuracy Mode

This mode allows to calculate the accuracy of the different algorithms used, and the success rate of each exercise for each method.

For this it gets the data from a file, while gets from another file the labels relative to such data.

The motive for not taking the data directly from the characteristics calculated from the processed data is because it is necessary to know the labels associated with each input data, to be able to know if the decision model has predicted the exercise correctly or not.

In this way, it will be taking the data from the file and it will predict the exercise based on these data. Then, the result of the prediction will be comparing with result that it should have been given, stored in the labels file. If the prediction of exercise is successful, it will take into account which exercise has been successful. So it is counting how many times each method guess the answer, for then to be able to calculate the percentage of success.

Once the accuracy of the different methods and their respective exercises have been calculated it saves them in a file in the folder of each subject and it displays the accuracy of the three methods on screen. It also displays a message indicating that the process has ended, as can be seen in Figure 5.17.

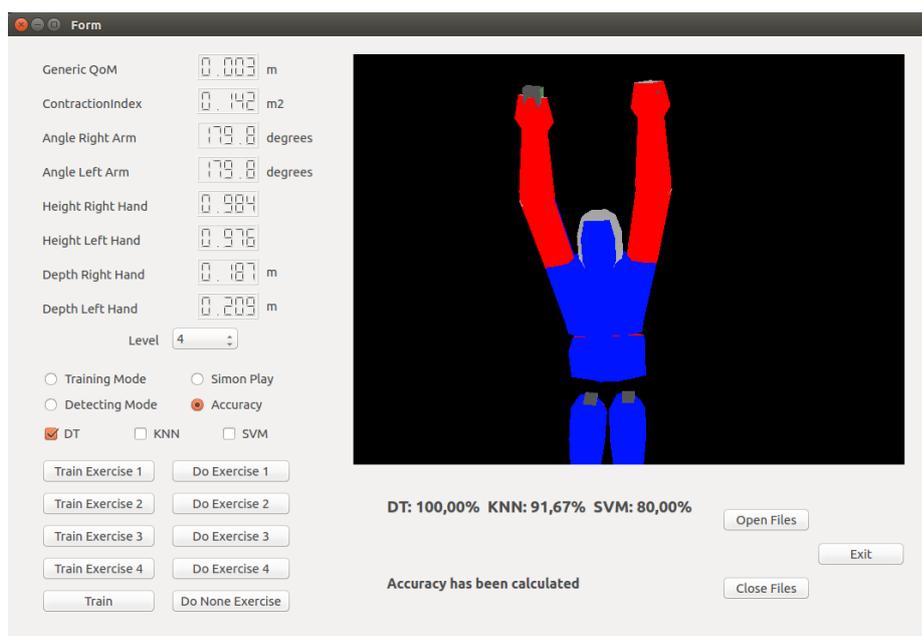


Figure 5.17: Interface in the Accuracy Mode

5.11 Classification Models

The classification models are algorithms that are able to learn after performing a training process with a known data, and then they are able to make decisions and classify new data based on what they have learned in training.

This will allow the robot to have more autonomy by deciding if the exercises have been implemented correctly.

The classification models that have been developed in the project are detailed below.

5.11.1 Decision Tree

A Decision Tree (DT) [47] [48] is a classification algorithm that is among the most used to solve problems.

It is a structure made up of nodes, limbs and leaf, as can be seen in Figure 5.18. The start node or the root node is from the decision process starts. Internal nodes or child nodes have one or more limbs leading to other internal nodes or to a leaf node. End nodes or leaf nodes correspond to a decision, and they include a label with a class which determines the assigned classification. Multiple leaves may have the same label.

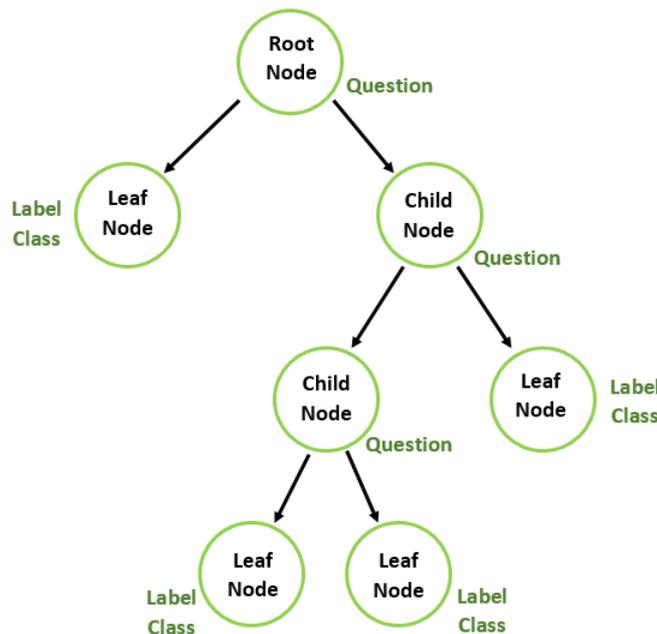


Figure 5.18: Decision Tree Structure

When there are objects that have more than one class in a node an internal node and a question to decide between these two classes are generated. When a node contains objects of a unique class, a leaf node to which is assigned the class label is formed.

The pattern classification is performed according to a series of questions about the variables, beginning by a root node and following the path determined by the answers to the questions in the internal nodes, until a leaf node. Once a leaf node is reached, the value assigned to this node is used as the output of the prediction procedure. The series of questions/answers, which ends in each leaf node, is a decision rule.

Whenever this type of model runs only a path can be followed, depending on the values that will take the evaluated variable. The values that can take the variables for these models can be discrete or continuous.

This algorithm takes an inductive learning from observations and logical constructions. It can be built from the narrative description of a problem as it shows graphically the decision making process, specifying the variables that are evaluated, the actions that must be taken and the order to be carried out.

It can be used either for classification or for regression. For classification, each leaf node is contains a class label, as it has been mentioned before. For regression, the mean value of a leaf node is predicted. A constant is assigned to a leaf node, so the approximation function is piecewise constant.

A simple example [49] of this algorithm can be seen in the Figure 5.19.

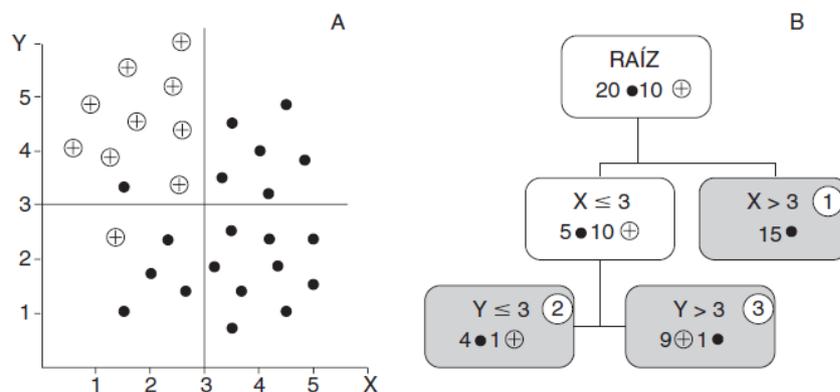


Figure 5.19: Decision Tree Example

The objective here is to classify the two types of figures (points and crosses) in the plane, depending on the values of two variables (X and Y).

Initially there are 20 points and 10 crosses that are not classified. The algorithm starts from the root node, being the first question about the variable X.

- If the X value is greater than 3 a leaf node (1) is formed. That node classifies 15 points.
- If the X value is smaller or equal than 3, there are 5 points and 10 crosses, but the values can not be classified for the moment. Another question about the variable Y has to be answered, so the actual node becomes an internal node, which leads to a two leaves nodes.
 - If the Y value is smaller or equal than 3, 4 points and 1 cross are classified in the leaf node (2).
 - If the Y value is greater than 3, 9 crosses and 1 point are classified in the leaf node (3).

It could have done successive partitions until get a pure classification, but the more precision we want in the classification, the bigger is the tree. Naturally, is very difficult to reach an absolute classification, and in most of cases it is not reached or the necessary tree to get it is too big (a tree with many leaves nodes as records in our database can be built to reached a pure classification).

It is very common that in the computational problems to be necessary to operate datasets with a big amount of instances. However, large dataset needs long time for processing all the training instances and in addition the available memory may not be enough for storing the whole training set and the big decision tree necessary to classify it, especially if a pure classification is required. Therefore, in some cases when the data set is very big, Decision Tree Algorithm is not applicable due to its time and memory consuming

5.11.2 K-Nearest Neighbours

The K-Nearest Neighbours Algorithm (KNN) [50] [51] is one of the simplest classification algorithms available for supervised learning.

This algorithm consist of compare the data to predict with the K-Nearest neighbours.

There is no training process in this algorithm, since its training consist of saving a set of known data, in such a way that to classify a new data, it searches for the most probable case using the saved data. Henceforth, the main calculation takes place when is comparing the data to predict with the neighbours saved.

To resolve a consulting, the KNN algorithm calculates the distances between the data to predict and the K nearest neighbours (it must be already fixed the k value), whose classes is already known and saved. On the basis of the consulting, it can be obtained different possibles answers. In that case the most voted answer will be chosen. The voting of the most adequate value can be with weight to some data, or without weight and treating all the calculates with the same percentage.

In the event of a tied classes, it is well worth having a heuristics rule to break it. For example, it may be selecting the nearest neighbour's class.

In the Figure 5.20 it is shown an example.

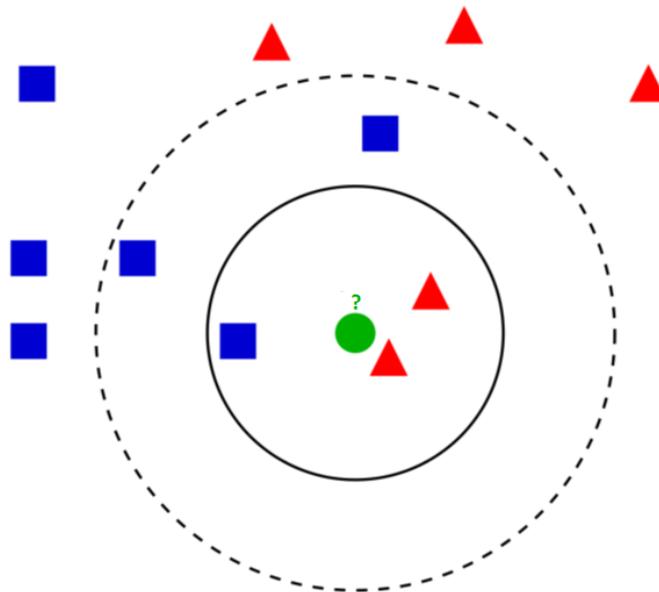


Figure 5.20: K-Nearest Neighbours Example

As it can be seen, if k is 3, it takes into account the nearest values, in that case the two red triangles and a blue square. On the basis of this, the algorithm will predict the green circle as a red triangle.

If, k has been 5 instead of 3, the nearest values would be three blue squares and two triangles, henceforth the green circle would be predicted as a blue square.

In the case of k is 1, the nearest neighbour will be assigned to the new data, even though it would have been the least probable.

If a too small value is assigned to k , then the KNN algorithm's result is very sensitive to near noisy points. But if k value is too big, then the near neighbours of the unknown point will include lots of point of different faraway classes. A k value lower than the minimum number of samples per class should always be set, because if k is greater than the number of samples per class the system would continuously be force to take samples from other classes, which will actively participate in the final vote.

We now learn from this that the determination of the k value is a very important question. K value must be small enough compared to the total number of points, but big enough to decrease the probability of a wrong classification.

If KNN algorithm is used with categorical data, the answer returned is the class whom should the unknown point belong to. On the other hand, if KNN algorithm is used with regression, the answer returned is the average of the nearest neighbours.

This system has many advantages as its simplicity, that has zero cost learning, the ability to modify and create new classes by adding new samples or the possibility to extend the mechanism to predict a continuous value (regression).

However, it is not without drawbacks, among which are that there is no mechanism for predicting the optimum value of K , but this depends on the dataset. Another is its high computational cost, depending on the number of classes and the number of samples per class. If we choose correctly the number of samples per class needed to make a good rating without any redundancy, this problem would be solved. This disadvantage can make this system not is a suitable method for large databases with many classes to classify.

5.11.3 Supported Vector Machine

The Support Vector Machine (SVM) [52] [53] [54] [55] is a widely used classifier that is able to classify two kind of input data by forming a decision boundary that separates the different kinds of data on the basis of the information given by the support vectors.

It divides the space of feature vectors by a hyperplane, that is, during the training process is responsible for determining appropriate parameters to separate the data into two subspaces, so that, by entering a query is calculated to which side of the hyperplane corresponds be allocated, and therefore its class is one that is associated with that area.

This results in a binary classifier (it only supports two types of classes). However, to extend the SVM algorithm to a problem like this project, more than one SVM model is used, which gives the possibility of realize multiple divisions of space and filter the correct area for a query.

The solution of the optimal hyperplane can be written as a combination of a few entry points that are called the support vectors, which are de data points that are closest to the decision boundary, and therefore they are the most difficult data to classify. Only the support vectors determine the weights and thus the boundary.

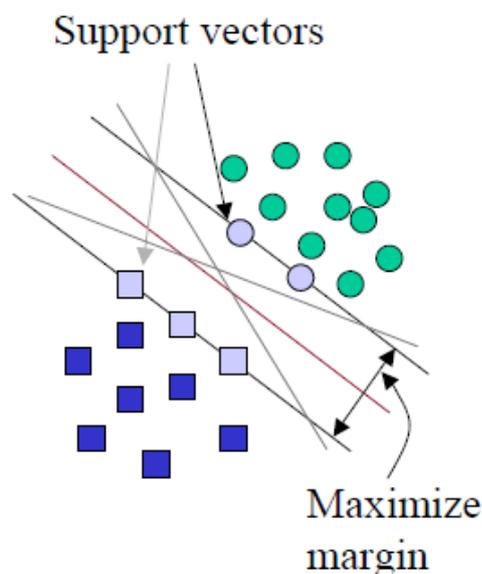


Figure 5.21: The Support Vectors

They are directly related to the optimum location of the decision boundary, consequently if a support vector is moved, the decision boundary is moved as well. However, moving the other vectors has no effect.

We want a classifier with as big margin as possible. Maximizing the margin m is a Quadratic Programming (QP) problem that can be solved by introducing Lagrange multipliers.

In a two dimensions space, a line will be needed to separate the data, in a three dimensions space a plane will be needed, and more generally for bigger dimensions, a hyperplane will be needed.

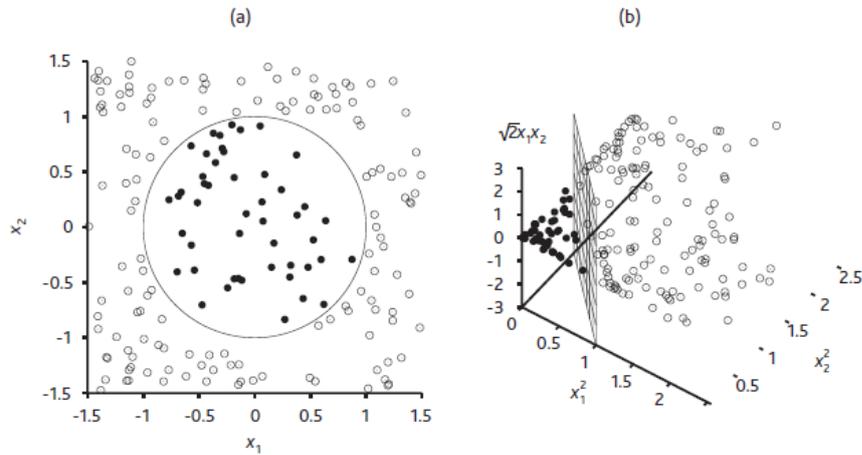


Figure 5.22: Two dimensions space Vs three dimensions space

Down below the linearly separable case, wherein exists a linear decision boundary that separates the two kinds of data, is explained. The no linearly separable case, that handles data that is not linearly separable with a non linear decision boundary, is explained too later on.

- **Linearly Separable Case:**

Suppose we have a set \mathbb{S} of points labelled for training. Every training point $x_i \in \mathbb{R}^N$ belongs to one of the two classes, and they have been given a label $y_i \in \{-1,1\}$ for $i=1,\dots,l$.

A lineal function discriminates between the two classes, creating a decision boundary that separates them.

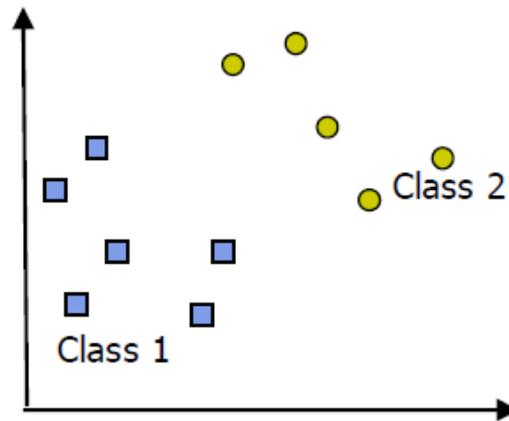


Figure 5.23: Example of a linearly separable case

A key concept to define a linear classifier is the dot product or scalar product of two vectors.

Initially we assume the examples are vectors, but once we introduce kernels this assumption will be relaxed, at which point they could be any continuous/discrete object.

The hyperplane divides the space into two by establishing a boundary that separates the two different kinds of data on each side of the hyperplane. For that, it is called the decision boundary of the classifier.

In the majority of cases, finding an appropriate hyperplane in an input space is too restrictive. A solution to that can be to map the input space to a bigger dimension space and look for the optimal hyperplane in that new features space.

The SVM first map the input points to a feature space of a larger dimension (for example, if the input points are in the space \mathbb{R}^2 , then they are mapped by the SVM to \mathbb{R}^3). After that, it finds a hyperplane that separate those points and that makes the margin bigger.

For the linearly separable case of the set \mathcal{S} , different possible hyperplane can be obtained, as can be seen in Figure 5.24. For the hyperplane chosen, the margin between the projections of the two classes of the training points is maximized.

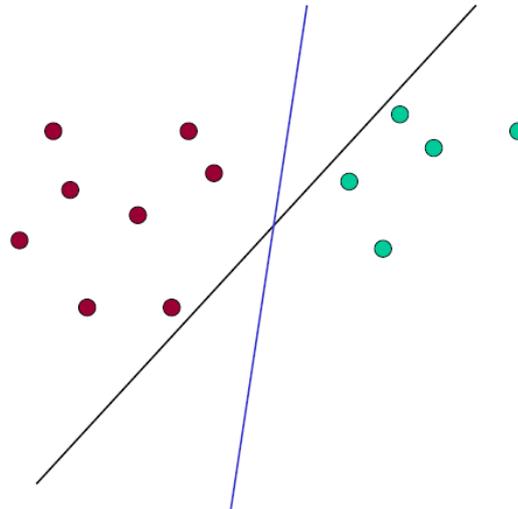


Figure 5.24: Different possibilities to choose a hyperplane

- **No Linearly Separable Case:**

If the set \mathbb{S} is not linearly separable, a linear decision boundary cannot be established to separate the different kinds of data. An example can be seen in Figure 5.25.

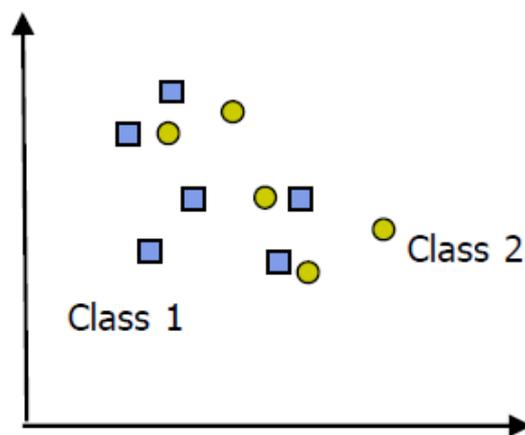


Figure 5.25: Example of a no linearly separable case

In many applications a non-linear classifier provides a better accuracy. However, linear classifiers have the advantage that they often have simple training algorithms that scale well with the number of examples.

Therefore, it would be interesting to adapt linear classifiers machinery to generate non-linear classifiers. This can be done from the kernel method.

As it is mentioned before, the dot product of two vectors is a key concept to define a lineal classifier. In some cases, the dot product can be replaced by a kernel function which calculates a dot product in some possibly high dimensional feature space.

It is possible to gain linearly separation by mapping the data to a higher dimensional space. It may not be separable by a linear function, but it can be separated by a quadratic one. The Kernel transforms data by a function chosen by the user to a feature space in which the data is linearly separable, and therefore the linear division is performed in the new space.

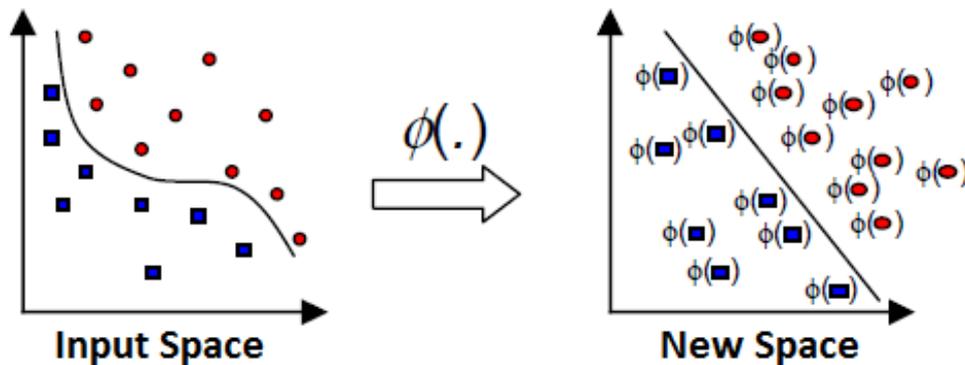


Figure 5.26: Example of a Kernel function does

A non-linear function ϕ has to be used to map the data from the input space X to a features space F ($\phi : X \rightarrow F$), in order to make a non-linear classifier out of a linear one.

The ability of generate non-linear decision boundaries applying methods designed to linear classifiers is one of the two main advantages of using the kernel method. The other main advantage is that the use of the kernel function allows the user to apply the classifier to data that have no fixed-dimensional vector space representation.

The calculate of the problem is not possible if we do not have any knowledge about ϕ . But a SVM property says that it is not necessary to have any knowledge about ϕ and only it is necessary a K function called *kernel*. That K function calculates the dot product between the two input points in the features space \mathbb{Z}).

In many applications, the SVM have proved to have more usability than traditional machines-learning as Neuronal Networks, and they have been introduced as a potent tool to solve classification problems.

This method has many advantages like its high accuracy, its ability to deal with high-dimensional data and its flexibility in modeling diverse sources of data.

Chapter 6

Experimental Results

In this chapter the results that have been obtained after making the necessary tests will be analyzed.

6.1 Exercises Data

In total 20 people between 20 and 50 years have been recorded performing a sequence with the exercises above defined.

Data from 5 of these people have served to train the component, whereas the recognition tests were carried out with the remaining 15 subjects whose data had not been used for training.

6.2 Features Extraction

An analysis of the characteristics calculated for each exercise has been realized, to check whether if they were among the expected values, and if they discriminate between the different exercises. To do this, a code has been generated using the button intended for this in the interface, and then the code has been processed with Matlab.

As can be seen in the graphics, sometimes appear fluctuations in the calculated features, but this is due to the fault detection of the Kinect. This can cause some problems in the recognition process.

The Figure 6.1 shows the Contraction Index that has been calculated for the different exercises. The Contraction Index measures the amount of contraction and expansion of the body with respect to its centroid, and their values are between 0 and 1.

As can be seen, the values of the Contraction Index are the expected: a high value for the Exercise 1, with an average value of 0.2, in which the hands are widely separated from the body. A median value for the Exercise 4, with an average of 0.14, since this exercise also separates the arms from the body, but the arms are close together. And a low value for the Exercises 2 and 3, with a value of about 0.6 for each exercise, as they hold hands close to the chest.

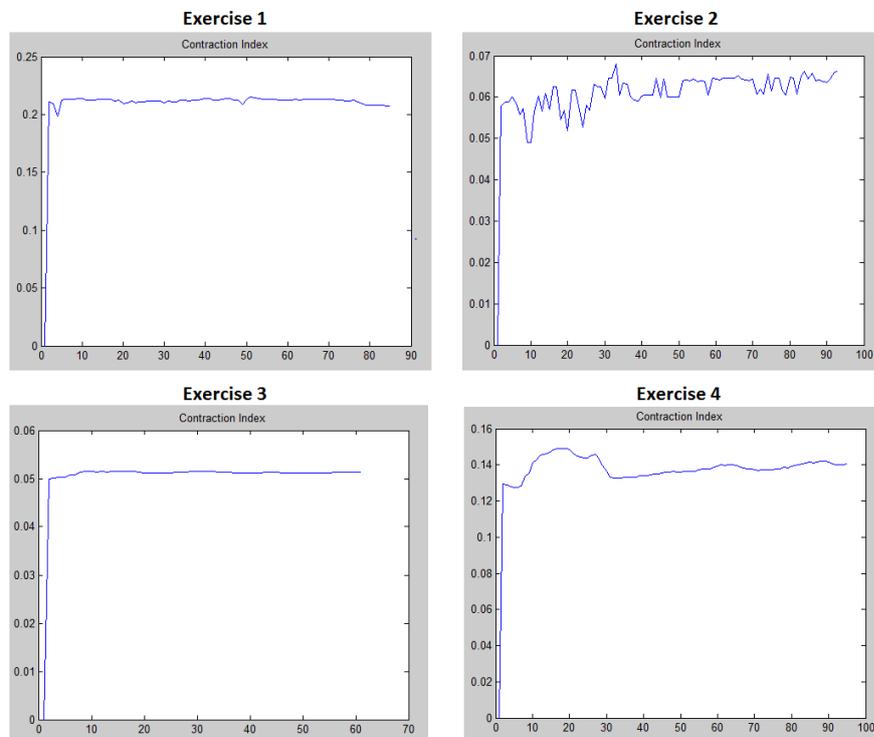


Figure 6.1: Contraction Index for the different exercises

In the Figures 6.2 and 6.3 can be seen the data collected with respects to the angles formed by the left and right arms respectively. As the graphics show, the values of the angles are between expected, being approximately 180° for the Exercises 1, 3 and 4, due to in these exercises the arms are stretched, and about 90° for the Exercise 2, because in this exercise the arms are flexed. Although it should be noted that especially for Exercise 1, some patients show a lower angle, a fact that has been taken into account in training and detection process.

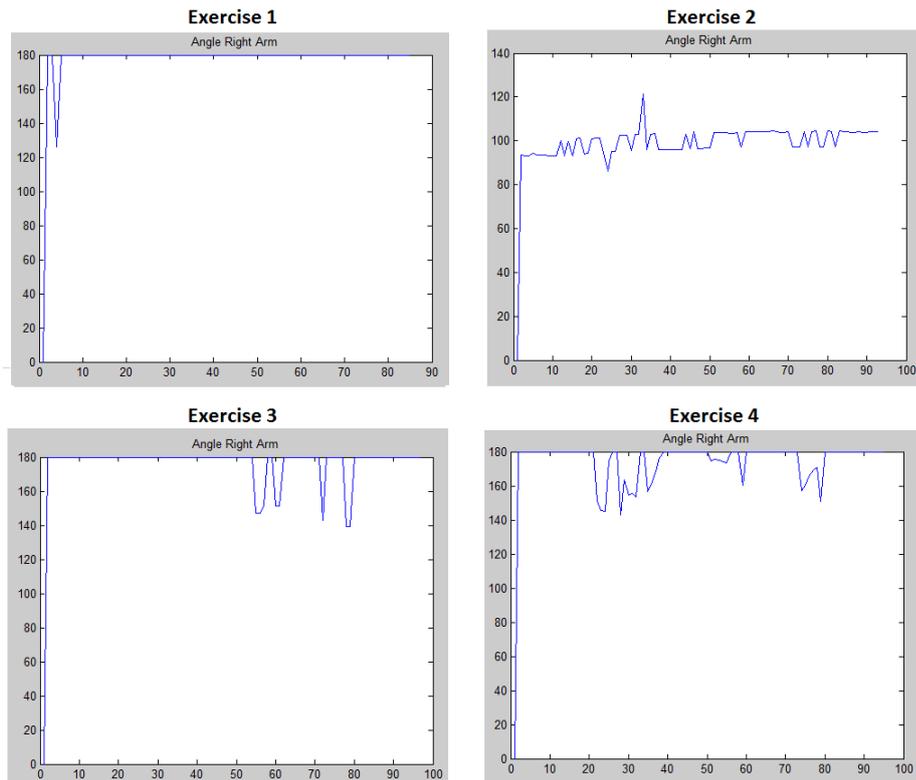


Figure 6.2: Angle of the Right Arm for the different exercises

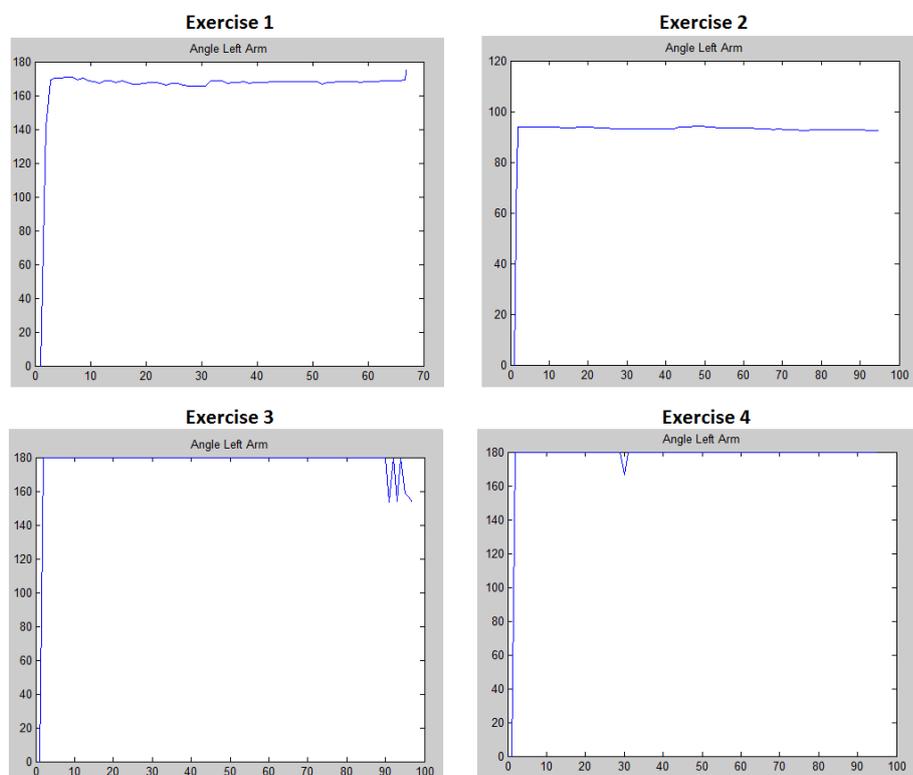


Figure 6.3: Angle of the Left Arm for the different exercises

As can be appreciated in Figure 6.2, for this subject there are several oscillations in the value of the calculated angle, due to the values detected by the kinect. This could create confusion in the recognition process for this value. However, in the Figure 6.3 can be seen that for the left hand, the patient provides a more constant values, making it ideal for recognition.

The Figures 6.4 and 6.5 show the heights of the hands normalized to 1 relative to each exercise. The results are again as expected, since in Exercise 4, in which the hands are above the head they find their maximum with an average of 1. For Exercises 1 and 3 hands are at shoulder height, obtaining an average value of 0.7. And finally, for Exercise 2 hands are on his hips, the hands take an even lower value of around 0.4.

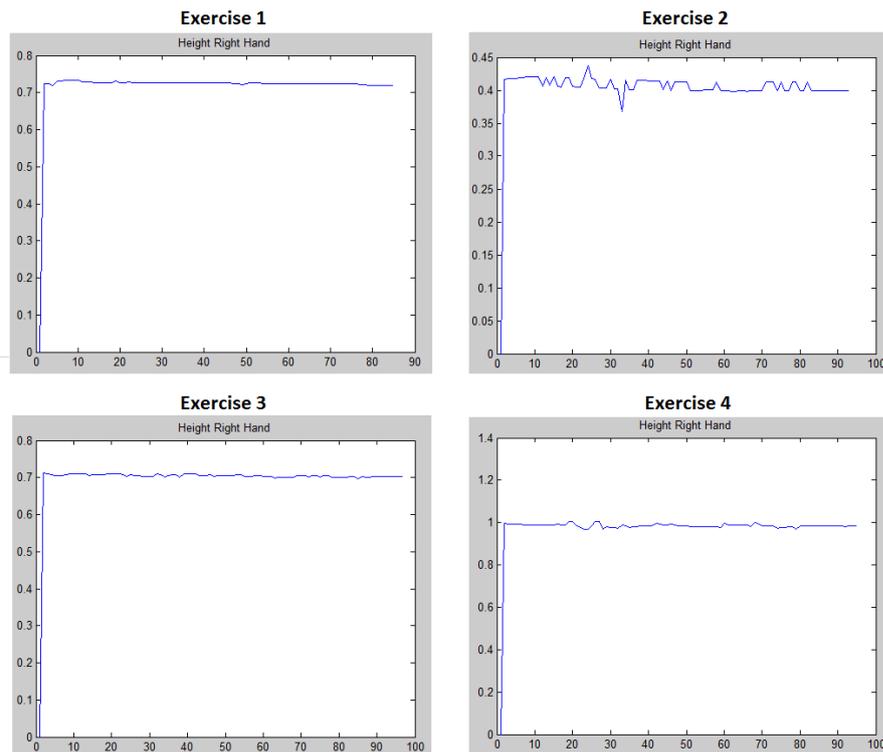


Figure 6.4: Height of the Right Hand for the different exercises

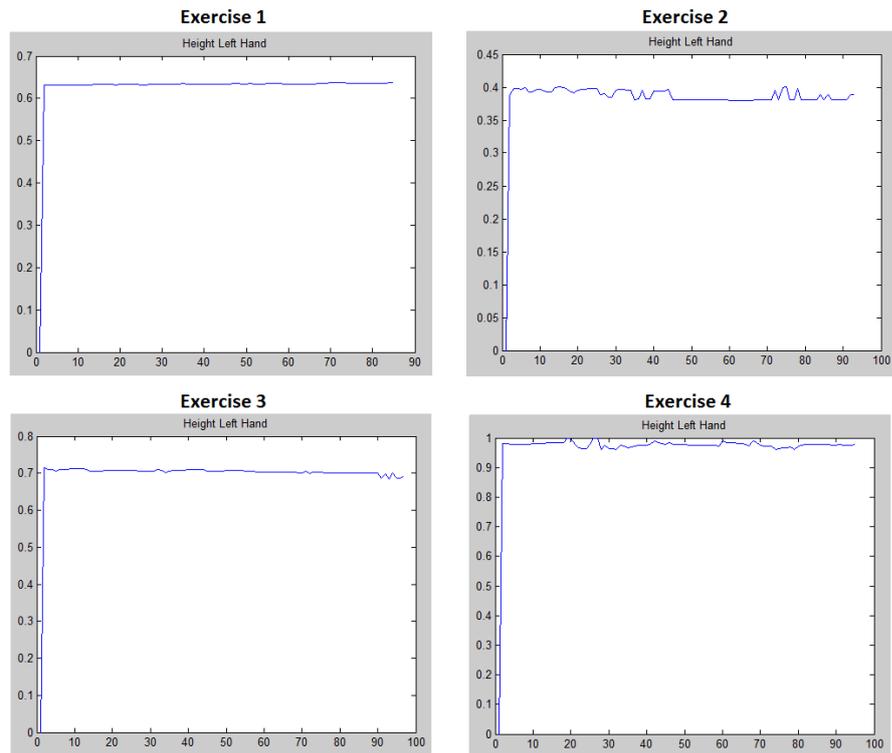


Figure 6.5: Height of the Left Hand for the different exercises

Finally, data of the depth of both hands are shown in Figures 6.6 and 6.7. It can be seen that for the Exercises 1, 2 and 4, in which the hands are kept in the plane of the body, some very low values, close to 0, are obtained. However, for the exercise 3 a somewhat higher value, between 60 and 70 cm, is obtained, which is about the length of the arm, as expected, because in this exercise the arms are stretched forward.

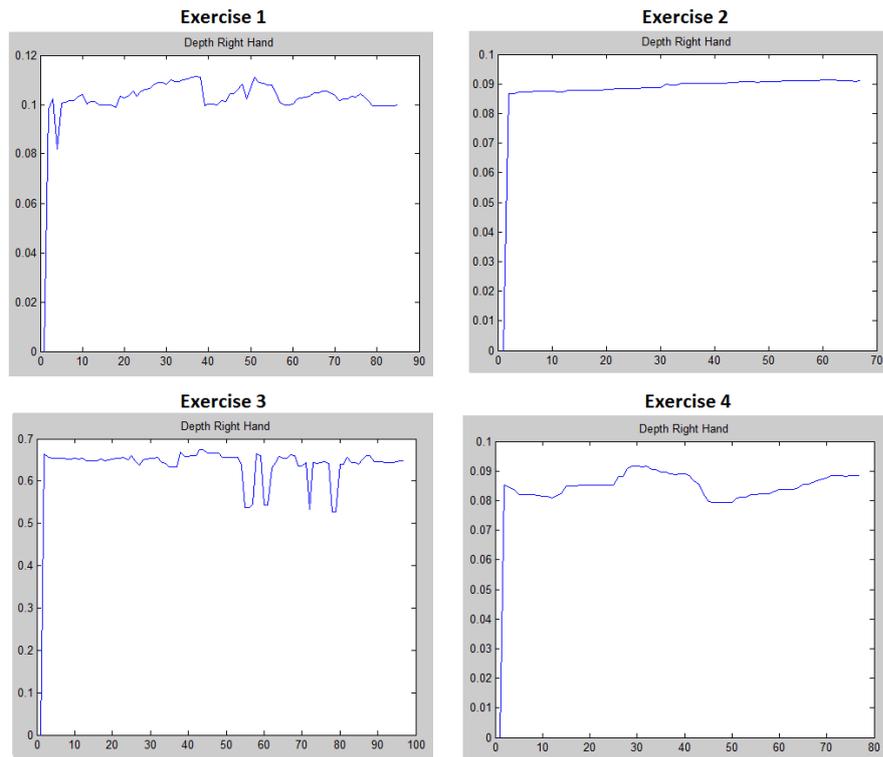


Figure 6.6: Depth of the Right Hand for the different exercises

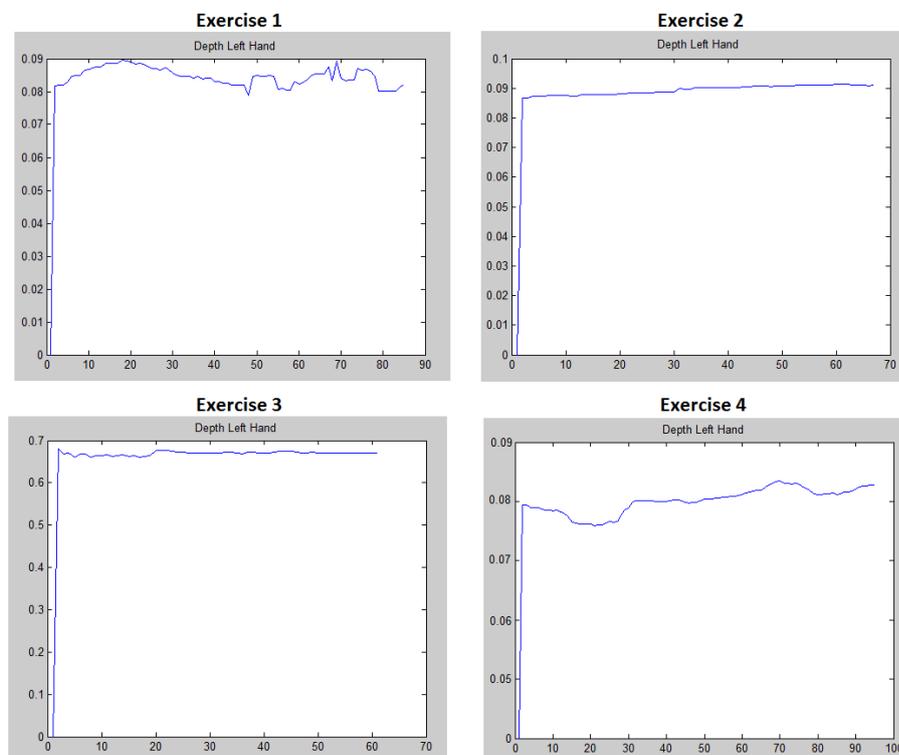


Figure 6.7: Depth of the Left Hand for the different exercises

6.3 KNN Accuracy depending on K value

As has already been said, finding the value of K for the KNN algorithm is a very important task. Hence, the first thing that has been done are tests to find out what would be the value of K. As mentioned, there is no method for guess what the optimal value of K, but it must be done by trial and error technique. Therefore, the algorithm KNN has been executed with different values of K for three different subjects and then the results has been analysed. We focused on small values of K because the smaller this value, the less comparisons will have to make the algorithm, and consequently the computational cost will be lower.

We can see the results in Table 6.1. In this Table can be seen that the Nearest Neighbour algorithm, that is to say, when $K=1$, is not which has a better accuracy in any of the cases. This is because when the algorithm only takes into account at the nearest neighbour to make the decision, it is more likely to make mistakes because the nearest neighbour may be of a different class.

The best results that have been obtained for the first 20 values of K are marked in red. For the Subject 1, the best values have been obtained for $K=5$, $K=6$ and $K=7$, for the Subject 2 the best values has been obtained for $K=10$, $K=11$ and $K=12$, and for the Subject 3 the best values ha been obtained for $K=5$, $K=6$, $K=19$ and $K=20$.

K	Subject 1	Subject 2	Subject 3
1	72.50%	94.17%	53.33%
2	75.00%	95.00%	53.33%
3	75.83%	95.83%	55.83%
4	75.00%	94.17%	55.83%
5	91.67%	95.83%	70.83%
6	91.67%	95.83%	70.83%
7	91.67%	95.00%	70.00%
8	90.00%	95.00%	70.00%
9	90.00%	95.00%	70.00%
10	90.00%	96.67%	70.00%
11	90.00%	96.67%	70.00%
12	90.00%	96.67%	70.00%
13	80.00%	94.17%	70.00%
14	80.00%	95.00%	70.00%
15	80.00%	95.83%	70.00%
16	80.00%	95.83%	70.00%
17	80.00%	95.83%	70.00%
18	80.00%	95.83%	70.00%
19	78.33%	95.83%	70.83%
20	78.33%	95.83%	70.83%

Table 6.1: KNN accuracy according to K values

As shown, the optimum values of K are different for each case. This is because the optimum value of K depends on the data set. Despite all of this, a unique value of K has been chosen for recognition to facilitate the task. The value of K that has been selected is K=5, because it matches that for the Subjects 1 and 3 this value of K gives maximum accuracy, and for the Subject 2 it gives the second highest value of the accuracy obtained for this subject. Moreover, being an odd value, the problems of ties in voting are avoided.

In Figure 6.8 can be seen a graph that shows how the precision of the method varies depending on the value of K.

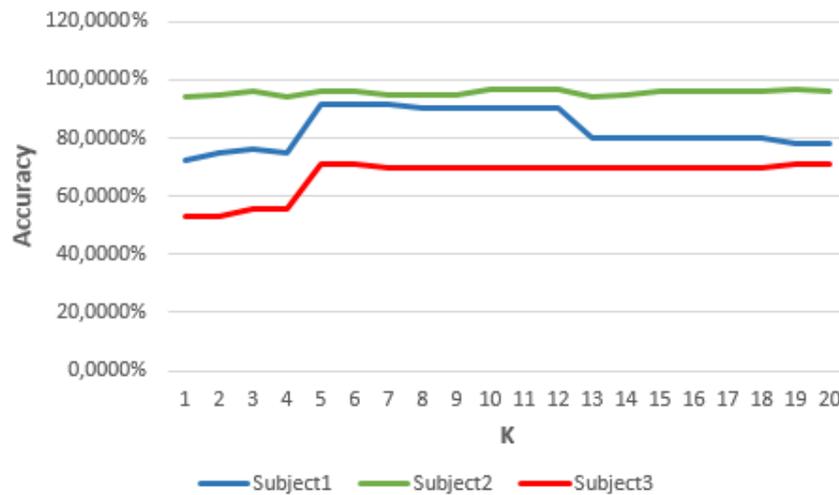


Figure 6.8: Accuracy of the KNN method according to K value

In this way, it can be seen graphically that the accuracy is lower for the first values and then increases, agreeing that for the 3 subjects the precision has a high value for $K=5$. As the value of K increases, the accuracy remains almost constant for the Subjects 2 and 3, while for the Subject 1 it goes down.

6.4 Recognition Accuracy

Once the value of K has been set, the accuracy of the three decision methods that have been developed has been tested. The tests were carried out with data from subjects that have not been used for training. The Table 6.2 shows the results obtained.

Subject	DT	KNN	SVM
1	100.00%	91.67%	80.00%
2	100.00%	95.83%	87.50%
3	100.00%	81.67%	62.50%
4	100.00%	72.50%	61.67%
5	99.17%	70.00%	39.17%
6	100.00%	77.50%	42.50%
7	100.00%	80.00%	75.83%
8	95.00%	68.33%	56.67%
9	100.00%	71.67%	45.00%
10	100.00%	95.00%	87.50%
11	100.00%	62.50%	38.33%
12	100.00%	68.33%	50.00%
13	100.00%	97.50%	65.83%
14	100.00%	76.67%	67.50%
15	100.00%	81.67%	65.83%
Mean	99.61%	79.38%	61.72%

Table 6.2: Accuracy obtained from the different decision methods

As can be seen, the decision method that gets the best results in recognizing the exercises is the Decision Tree, with an average of probability of success of 99.61%. KNN method also generally give good results, although not as good as with the DT algorithm. This may be because as explained, the value of K influence the result, but at the same time the optimum value of K depends on the dataset, so as the calculation is made with a single value of K, the best possible results may not have achieved.

As for the SVM method we found some results in which the success rate is very satisfactory and others where it is not as good as we would like, but in any of the cases the accuracy obtained is higher than the accuracy obtained in the other two methods.

This low accuracy can be obtained because some exercises are very similar to each other, differentiating between them only by two features. For example, Exercise 1 and 3 are distinguished by the depth of the hands, and the Exercise 1 differ from the Exercise 4 only in the height of the hands.

Therefore the Decision Tree algorithm gets better results because it asks questions about the variables to decide whether an exercise or another. On the other hand the KNN and SVM algorithms represent the points spatially, then the point that must be classified is painted to decide which exercise belongs, KNN algorithm by comparing to the closest points, and the SVM algorithm depending on the area where the point is. As these exercises are differentiated by a few characteristics, it is possible that the points are together in the spatial representation and this creates confusion when deciding.

The following tables show the accuracy that have been obtained for each exercise and for each method. The Table 6.3 shows data related to the Exercise 1, the Table 6.4 shows the accuracy of the Exercise 2, the Table 6.5 shows data referred to Exercise 3 and finally the Table 6.6 shows the data of the Exercise 4.

As shown in Table 6.3, the accuracy of the Exercise 1 is generally quite good for all the three methods. Why failure both SVM and KNN to the subject 14 is because that person has not put his arms in the plane of the body to perform exercise, but they are slightly pitched forward, what makes the realization of this exercise not to be ideal and the methods to be confused in the decision.

Subject	DT	KNN	SVM
1	100.00%	100.00%	100.00%
2	100.00%	86.67%	100.00%
3	100.00%	96.67%	100.00%
4	100.00%	40.00%	86.67%
5	96.67%	90.00%	90.00%
6	100.00%	100.00%	93.33%
7	100.00%	30.00%	93.33%
8	100.00%	73.33%	100.00%
9	100.00%	73.33%	80.00%
10	100.00%	83.37%	96.67%
11	100.00%	90.00%	100.00%
12	100.00%	76.67%	96.67%
13	100.00%	100.00%	90.00%
14	100.00%	16.67%	26.67%
15	100.00%	86.00%	100.00%

Table 6.3: Accuracy of the Exercise 1 in the different methods

Table 6.4 is about the accuracy obtained for the recognition of the Exercise 2. It is in general very good for Decision Tree and KNN methods, while for the SVM method the results are not very accurate, except for two or three cases.

Subject	DT	KNN	SVM
1	100.00%	100.00%	53.33%
2	100.00%	100.00%	63.32%
3	100.00%	100.00%	0.00%
4	100.00%	100.00%	10.00%
5	100.00%	100.00%	3.33%
6	100.00%	100.00%	0.00%
7	100.00%	100.00%	30.00%
8	80.00%	80.00%	16.67%
9	100.00%	100.00%	10.00%
10	100.00%	100.00%	66.67%
11	100.00%	100.00%	36.66%
12	100.00%	100.00%	6.67%
13	100.00%	100.00%	0.00%
14	100.00%	100.00%	70.00%
15	100.00%	100.00%	43.33%

Table 6.4: Accuracy of the Exercise 2 in the different methods

Table 6.5 shows how Exercise 3 gets very good results for the Decision Tree algorithm, while for KNN and SVM algorithms it depends on the subject, in some cases obtaining a high accuracy, while in others the opposite.

This may be because there are many errors in the recording of Exercise 3, because by putting hands in front positions of the elbow and shoulder is covered with the hand positions. However, DT algorithm is able to get good precision despite the recording failure.

Subject	DT	KNN	SVM
1	100.00%	100.00%	100.00%
2	100.00%	100.00%	96.67%
3	100.00%	30.00%	50.00%
4	100.00%	100.00%	100.00%
5	100.00%	90.00%	63.33%
6	100.00%	66.67%	76.67%
7	100.00%	90.00%	80.00%
8	100.00%	56.67%	33.33%
9	100.00%	70.00%	46.67%
10	100.00%	96.67%	86.67%
11	100.00%	56.67%	13.33%
12	100.00%	0.00%	0.00%
13	100.00%	900.00%	73.33%
14	100.00%	100.00%	80.00%
15	100.00%	50.00%	30.00%

Table 6.5: Accuracy of the Exercise 3 in the different methods

The accuracies obtained for the Exercise 4 are very good for the DT method, and generally good for KNN and SVM methods, despite some cases, as can be seen in Table 6.6. The subjects 5, 6 and 11 are not performing this exercise in a proper way, due to they have not put their arms vertically with the body, that's why the accuracy is so low.

Subject	DT	KNN	SVM
1	100.00%	66.67%	66.67%
2	100.00%	96.67%	90.00%
3	100.00%	66.67%	66.67%
4	100.00%	100.00%	100.00%
5	100.00%	0.00%	0.00%
6	100.00%	43.33%	0.00%
7	100.00%	100.00%	100.00%
8	100.00%	63.33%	70.00%
9	100.00%	43.33%	43.33%
10	100.00%	100.00%	100.00%
11	100.00%	3.33%	3.33%
12	100.00%	96.67%	96.67%
13	100.00%	100.00%	100.67%
14	100.00%	90.00%	93.00%
15	100.00%	90.00%	90.00%

Table 6.6: Accuracy of the Exercise 4 in the different methods

6.5 Real Experience

A highlighted experience within the project has been recording two therapies conducted by an occupational therapist with two patients of the Residence and Day Centre “Santa Ana” from Malpartida de Cáceres [56]. This is a center with 31 places for inpatients, and other 45 places for users of the Day Centre.

Two patients who have the right profile for the therapy were selected. The therapist carried out the session with them and it was recorded with the Kinect sensor, as can be seen in Figure 6.9.



Figure 6.9: Real therapy session with a patient

It has been proven the enthusiasm of patients to get out of the routine and perform the session with devices that were completely unfamiliar to them, so we assume that when the session is performed with a robot which is an object they can recognize and identify and which they are able to interact with, their motivation will further increase.

Furthermore, when analysing the videos it has been seen that conducting a therapy session with an elderly patient is a complex process, wherein the number of errors made by the patient is quite high. Therefore, the robot must be very interactive and greatly assist the patient in performing therapy, showing them the exercises to realize very clearly, correcting them and giving them guidelines on how improve them and giving them clues about which is the next exercise to realize.

Chapter 7

Conclusions and Future Work

This chapter explains the conclusions drawn from the analysis of the experimental results, and possible future work that would allow the project to continue.

7.1 Conclusions

From the Hypothesis 1 that a therapy that combines the physical development of the patient and the development of memory will be more enriching it has been developed a component able to recognize the exercises from the therapy, as well as play Simon Says game by performing these exercises (Objective 1). The game of Simon Says will allow to work patient's physical condition by performing exercises and his memory by memorizing the sequence of exercises to do.

For the recognition of the exercises different features has been extracted. Then, these features have been introduced in three different decision algorithms (Decision Tree, K Nearest Neighbours and Supported Vector Machine) to decide which exercise belonged those characteristics.

Of the three algorithms used, the most suitable for our data set is the DT since it is the one which the best accuracy results have been obtained. The KNN and SVM algorithms provide greater error because they dependent on the spatial representation of the points, and how they differ in few features they may be found close together in the space, generating confusion.

If you are going to work with different data sets (as in our case, that there is a different data set for each subject) KNN algorithm is not most appropriate

because its results depends on the value of K , and this depends on the data set.

It has also been created and shared a database with the 3D positions of the recorded videos of the various subjects performing different exercises and the features extracted for each exercise (Objective 1.4). This data base has been used both in the training and the recognition process, and it can be downloaded from the service of the Universidad de Extremadura using the link <http://descarga.unex.es//ref.php?ref=emogenac.2f0221c57c7eb0dfcacf4530eae95ddf>, so that any developer can use it with their systems.

The rest of the sub-objectives set at the beginning have also been reached throughout the project. To perform the component has had to implement a system which recognizes the exercises performed by the patient (Objective 1.1) and to develop an environment that enables to interact with the patient asking him to perform exercises (Objective 1.2), but for now it is not capable of correcting an exercise if it is not realized correctly. A graphical interface that allows working with the component has also been created (Objective 1.3). In addition, for all this, it was necessary to learn about the software RoboComp (Objective 1.5) and research on the benefits that a therapy has in the elderly (Objective 1.6). And as the therapyComp component has been created in C ++, the programming skills in this language has been enhanced (Objective 1.7).

7.2 Future Work

This system, like everyone else, can be improved and further developed over time, therefore the various future work that can be derived from this project are going to be mentioned.

The first change to implement would be the component recognize the exercises in real time, thus the Simon Says game could be played in a closer way than it would be played in the therapy with the therapist. This can be done by modifying the way the component read the data, so that instead of reading the data from the file, it would read the data directly from the Kinect.

Another possible improvement would be to extend the set of exercises to be performed, thus increasing the quality of therapy.

Another needed enhancement is that the component is able to distinguish when performing an exercise that has not been implemented and therefore also to recognize failures in performing an exercise. If the component would work in real time, it would be interesting that the component gives to the patient guidelines on how to improve the exercise, so that he can do it correctly.

To avoid some mistakes, it would be interesting to select other features that describe the exercises, in order to discriminate enough to eliminate possible confusion.

More people could also be recorded with the aim of expanding the database with the different exercises.

And of course, a great contribution to the system would be to remove the interface, internalizing the component on the robot. In this way, the communication with the patient would not take place through the interface, but the robot using a voice command would give the necessary information about the exercise to perform to the patient.

7.3 Personal Evaluation

This project has given me an incomparable experience, both personally and professionally, becoming a motivating challenge for me.

The concept of social robot draws much attention, however the group of older people are a bit forgotten by society so focusing this project on elderly people has helped me to reflect on their needs besides giving me the comforting feeling of helping others with your work.

In addition, this project has helped me to improve my skills as an engineer, for example in the field of C++ programming and interfaces development, but mostly it has improved my capability to deal with small problems that arose and my ability to divide a large problem into several small problems to make its resolution easier.

For all this, I can say that the development of this project led to a pleasant and motivating experience.

Chapter 8

Acknowledgements

I would like to greatly thank the CénitS-Computaex Foundation for giving me the opportunity to train as an engineer out of the classroom, allowing me to have a first contact with the labour market due to the Training Grant that has been given to me.

Also, to thank the Robolab department and all those who have been willing to answer my questions.

Of course my two tutors, for their dedication and support, and provide me guidance throughout the project.

I can not forget to recognize especially all those who have helped me in recording videos performing the exercises, letting me know they were there for everything I needed.

And I mainly want to thank my family, my boyfriend and my friends for supporting me always, even when I was overwhelmed by the problems that arose. Thank you for trusting me, giving me the strength to achieve my goals.

References

- [1] W. Chen. *Gesture-Based Applications for Elderly People*. In: M.Kurosu Ed.). Human-Computer Interaction, Interaction Modalities and Techniques, Part IV, HCII 2013, LNCS 8007, pp. 186-195, 2013.
- [2] A.M. Álvarez and R.B. Zapata. *Las bandas elásticas, un medio para el mejoramiento de la fuerza muscular en los adultos mayores*. Facultad de Educación Física de Medellín, Colombia, 2008.
- [3] L.F. Feredia. *Ejercicio físico y deporte en los adultos mayores*. GEROINFO. Publicación de Gerontología y Geriatria. RNPS. 2011. Vol. 1 No. 4. 2006.
- [4] V.A. Aparicio, A. Carbonell-Baeza and M. Delgado-Fernández. *Health benefits of physical activity in older people*. Revista Internacional de Medicina y Ciencias del Deporte, Vol. 10 (40), pp. 556-576.
- [5] M. Jara. *La estimulación cognitiva en personas adultas mayores*. Revista Cúpula.
- [6] N. Walker, D.A. Philbin and A.D. Fisk. *Age-related differences in movement control: adjusting submovement structure to optimize performance*. In: Journal of Gerontology: Psychological Sciences, Vol 52B, pp. 40-53, 1997
- [7] T.G. Zimmerman, J. Lanier, C. Blanchard, S. Bryson and Y. Harvill. *A hand gesture interface device*. In: SIGCHI Bull, Vol 18, pp. 189-192, 1986.
- [8] D. Fitzgerald, J. Foody, D. Kelly, T. Ward, C. Markham, J. MacDonald and B, Cauldfield. *Development of a wearable motion capture suit and virtual reality biofeedback system for the instruction and analysis of sports rehabilitation*

exercices. In: Proc. of IEEE Conf. Eng. Med. Biol. Soc., pp. 4870-4874, IEEE 2007.

- [9] <http://www.senztech.cc/UploadFiles/20120706114343.pdf>
- [10] Image extracted from <http://www.est-kl.com/fr/products/data-gloves/cyberglove-systems/cyber-glove-iii.html>
- [11] http://en.wikipedia.org/wiki/Wii_Remote
- [12] <https://www.nintendo.es/index.html> // <https://www.nintendo.com>
- [13] http://en.wikipedia.org/wiki/PlayStation_Move
- [14] <http://www.sony.es> // <http://www.sony.com>
- [15] Image extracted from playstation-3-fehler-medienserver.weebly.com
- [16] <http://en.wikipedia.org/wiki/Kinect>
- [17] <http://www.microsoft.com/es-es/default.aspx>
- [18] <http://en.wikipedia.org/wiki/EyeToy>
- [19] http://www.asus.com/Multimedia/Xtion_PRO_LIVE/
- [20] <http://www.asus.com/ES/>
- [21] L.V. Calderita, P. Bustos, C. Suárez, F. Fernández, R. Vicianad, and A. Bandera. *Asistente Robótico Socialmente Interactivo para Terapias de Rehabilitación Motriz con Pacientes de Pediatría*. ScienceDirect Revista Iberoamericana de Automática e Informática industrial Vol. 12, pp. 99-110, 2015.
- [22] http://www.eurekalert.org/pub_releases/2015-04/ciuo-dar042015.php
- [23] Image extracted from <http://www.mundodigital.net/un-robot-destinado-a-terapias-motrices-para-ninos/>
- [24] Armeo Therapy Concept in http://www.hocoma.com/fileadmin/user/Dokumente/Armeo/bro_Armeo_Therapy_Concept_140226_en.pdf

- [25] Image extracted from <http://www.elsa.web.tr/tr/urun/robotik-rehabilitasyon/hocomma-armeo-spring>
- [26] <http://www.sanar.org/salud/terapia-robotica>
- [27] <http://universitam.com/academicos/?p=1052>
- [28] Image extracted from <http://saberesciencias.com.mx/2015/04/06/el-uso-de-robots-como-apoyo-en-terapia-ocupacional/>
- [29] L. Manso, P. Bachiller, P. Bustos, P. Núñez, R. Cintas and L. Calderita. *Robo-Comp: a Tool-based Robotics Framework*. In: Proceedings, SIMPAR Second International Conference on Simulation, Modeling and Programming for Autonomous Robots. pp. 251-262, 2010.
- [30] https://robofab.unex.es/index.php?option=com_content&view=article&id=10&Itemid=26
- [31] A. Brooks, T. Kaupp, A. Makarenko, S. Williams and A. Örebäck. *Orca: A Component model and Repository*. Springer Tracts in Advanced Robotics, Vol 30, pp 231-251, 2007.
- [32] <http://orca-robotics.sourceforge.net/pages.html>
- [33] M. Quigley, B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng. *ROS: an open-source Robot Operating System*. ICRA Workshop on Open Source Software, 2009.
- [34] <http://www.ros.org/>
- [35] G. Metta, P. Fitzpatrick and L. Natale. *YARP: Yet Another Robot Platform*. International Journal of Advanced Robotic Systems, Vol. 3, No. 1, ISSN 1729-8806, pp. 043-048, 2006.
- [36] B.F. Al-Dulaimi. *Object-oriented reusable component*. AL-USTATH, Vol 2, No 208 AD, 1435 AH, pp.13-24, 2014.
- [37] OpenCv library in <http://opencv.org/>
- [38] <http://www.ocompras.com/juegos-2/el-clasico-juego-de-simon-dice>

- [39] Microsoft Kinect for Windows Information in URL <http://www.microsoft.com/en-us/kinectforwindows/meetkinect/default.aspx>
- [40] Kinect image extracted from http://www.techhive.com/article/247724/kinect_for_windows_available_february_1_but_overpriced_at_249.html
- [41] Information about the sensors inside the Kinect in the <http://www.abc.es/20110215/tecnologia/abci-kinect-dentro-201102151229.html>
- [42] Information about the reference system for the Kinect in <http://animusproject.wix.com/web/apps/blog/integraci%C3%B3n-de-dispositivos-leap-y-kinect>
- [43] Microsoft Kinect for Windows Features in URL <http://www.microsoft.com/en-us/kinectforwindows/meetkinect/features.aspx?navV3Index=0>
- [44] Image extracted from <http://www.diariolaventana.com/articulo.php?id=2864>
- [45] Image extracted from <https://claufitnessblog.wordpress.com/workout-logs/circuitos-para-vacaciones/>
- [46] Image extracted from <http://www.womenshealthmag.com/fitness/stretch-workout>
- [47] A. Bahety. *Extension and Evaluation of ID3 – Decision Tree Algorithm*.
- [48] R. Barrientos, N. Cruz, H. Acosta, I. Rabatte, M.C. Gogeoascoechea, P. Pavón and S. Blázquez. *Decision trees as a tool in the medical diagnosis*. Revista Médica Vol. 9(2), pp. 19-24, 2009.
- [49] J. Trujillano, A. Sarria-Santamera, A. Esquerda, M. Badia, M. Palma and J. March. *Approach to the methodology of classification and regression trees*. Gac Sanit, Vol. 22(1), pp. 65-72, 2008.

- [50] KNN Algorithm from OpenCv in URL http://opencv-python-tutroals.readthedocs.org/en/latest/py_tutorials/py_ml/py_knn/py_knn_understanding/py_knn_understanding.html
- [51] A. Moujahid, I. Inza and P. Larrañaga. *Clasificadores K-NN*. Departamento de Ciencias de la Computación e Inteligencia Artificial Universidad del País Vasco–Euskal Herriko Unibertsitatea.
- [52] G. A. Betancourt. *Las máquinas de soporte vectorial (SVMs)*. Scientia et Technica Año XI, UTP. ISSN 0122-1701, No 27, Abril 2005.
- [53] D.A. Salazar, J.I. Vélez and J.C. Salazar. *Comparison between SVM and Logistic Regression: Which One is Better to Discriminate?*. Revista Colombiana de Estadística, Número especial en Bioestadística, Vol. 35, No. 2, pp. 223-237, Junio 2012.
- [54] A. Ben-Hur and J. Weston. *A User's Guide to Support Vector Machines*.
- [55] R. Berwick. *An Idiot's guide to Support vector machines (SVMs)*.
- [56] <http://www.nexusintegral.com/nuestros-centros/malpartida-de-caceres/>

Appendix A

RoboComp Installation Manual

RoboComp is a framework of open source robotics. It provides a set of robotics components and the necessary tools to work with them or create a new one if it is desired. You can access the Robocomp page through the link www.robocomp.org.

To install de RoboComp software, it is necessary to open a terminal command, and to write the “*git clone*” command as can be seen in Figure A.1. It is important to do that in the folder where you want to install RoboComp (in my case, in my home directory).

```
eva@EvaMogena:~$ git clone https://github.com/robocomp/robocomp.git
Clonar en «robocomp»...
remote: Counting objects: 18508, done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 18508 (delta 1), reused 0 (delta 0)
Receiving objects: 100% (18508/18508), 3.54 MiB | 811.00 KiB/s, done.
Resolving deltas: 100% (11197/11197), done.
Checking connectivity... hecho.
```

Figure A.1: Cloning Robocomp

With this the RocoComp folder is cloned in your computer, in the specified folder. But only the main folder has been cloned, without none components inside it. It can be proved that the “components” folder is empty.

```
eva@EvaMogena:~$ cd robocomp/
eva@EvaMogena:~/robocomp$ ls
classes  cmake  CMakeLists.txt  components  files  interfaces  libs  LICENSE  README.md  tools
eva@EvaMogena:~/robocomp$ cd components/
eva@EvaMogena:~/robocomp/components$ ls
eva@EvaMogena:~/robocomp/components$ █
```

Figure A.2: Proving components folder is empty

Now the rest of libraries with which RoboComp will work have to be installed. For that, the command “git-anex” is used, as is shown in Figure A.3

```
eva@EvaMogena:~$ cd robocomp
eva@EvaMogena:~/robocomp$ cd files/
eva@EvaMogena:~/robocomp/files$ ls
666-robolab.rules 666-robolab-smar.rules components-port.txt freedesktop gazeboWorlds innermodel osgModels thirdparties
eva@EvaMogena:~/robocomp/files$ apt-cache search git-anex
git-anex - manage files with git, without checking their contents into git
libghc-yesod-default-prof - default config/main functions for your Yesod application; profiling libraries
libghc-yesod-default-dev - default config and main functions for your Yesod application
libghc-yesod-default-doc - default config and main functions for your Yesod application; documentation
eva@EvaMogena:~/robocomp/files$ sudo apt-get install git-anex█
```

Figure A.3: Installation “git-anex” command

When the installation is finished, we execute that command, as can be see in Figure A.4.

```
Configurando quvi (0.4.2-1) ...
Configurando curl (7.35.0-lubuntu2.1) ...
Configurando libgsasl7 (1.8.0-2ubuntu2) ...
Configurando git-anex (5.20140412ubuntu1) ...
Configurando git-remote-gcrypt (0.20130908-5) ...
Configurando nocate (0.9-1) ...
Processing triggers for libc-bin (2.19-0ubuntu6.3) ...
eva@EvaMogena:~/robocomp/files$ ls
666-robolab.rules 666-robolab-smar.rules components-port.txt freedesktop gazeboWorlds innermodel osgModels thirdparties
eva@EvaMogena:~/robocomp/files$ ls -la freedesktop/
total 32
drwxrwxr-x 2 eva eva 4096 oct 30 12:00 .
drwxrwxr-x 7 eva eva 4096 oct 30 12:00 ..
-rw-rw-r-- 1 eva eva 368 oct 30 12:00 install.sh
lrwxrwxrwx 1 eva eva 192 oct 30 12:00 robocomp-icon.png -> ../../.git/annex/objects/x9/WQ/SHA256E-s15654--b2bcf057e57b3977ed494a26b7ca943b0f555d3b9ac6ec15bd0f0dbf1
9ad7f33/SHA256E-s15654--b2bcf057e57b3977ed494a26b7ca943b0f555d3b9ac6ec15bd0f0dbf13ad7f33
-rw-rw-r-- 1 eva eva 315 oct 30 12:00 RoboComp.menu
-rw-rw-r-- 1 eva eva 218 oct 30 12:00 RoboComp-rccontrolpanel.desktop
-rw-rw-r-- 1 eva eva 206 oct 30 12:00 RoboComp-rcmonitor.desktop
-rw-rw-r-- 1 eva eva 64 oct 30 12:00 RoboComp-RoboComp.directory
eva@EvaMogena:~/robocomp/files$ git annex get █
```

Figure A.4: Installing RoboComp libraries

It will take a while until all files have been downloaded and installed.

Meanwhile, in another console tab, we will open the bashrc file using the commands shown in Figure A.5. This file is hidden in our home folder, so it is necessary to run the command from that folder.

```
eva@EvaMogena:~$ vim .bashrc
```

Figure A.5: Opening the bashrc file

What we have done is to open the bashrc file using an editor to configure it. It also can be opened writing “sudo nano .bashrc” or “gedit .bashrc” in the console.

Once opened, we need to add a text line that says RoboComp is in the direction in which we have installed (in my case in my home folder), as can be seen in Figure A.6.

```
*.bashrc x
alias grep='grep --color=auto'
alias fgrep='fgrep --color=auto'
alias egrep='egrep --color=auto'
ft

# some more ls aliases
alias ll='ls -alF'
alias la='ls -A'
alias l='ls -CF'

# Add an "alert" alias for long running commands. Use like so:
# sleep 10; alert
alias alert='notify-send --urgency=low -i "${?} = 0" && echo terminal || echo error' "${history|tail -n1|sed -e '\s/\s*[0-9]\+\s*//;s/[\&]\s*alert$//'\}'"

# Alias definitions.
# You may want to put all your additions into a separate file like
# ~/.bash_aliases, instead of adding them here directly.
# See /usr/share/doc/bash-doc/examples in the bash-doc package.

if [ -f ~/.bash_aliases ]; then
    . ~/.bash_aliases
fi

# enable programmable completion features (you don't need to enable
# this, if it's already enabled in /etc/bash.bashrc and /etc/profile)
# sources /etc/bash.bashrc.
if ! shopt -oq posix; then
    if [ -f /usr/share/bash-completion/bash_completion ]; then
        . /usr/share/bash-completion/bash_completion
    elif [ -f /etc/bash_completion ]; then
        . /etc/bash_completion
    fi
fi

export ROBOCOMP='/home/eva/robocomp'
```

Figure A.6: Editing the bashrc file

Now we can try to compile RoboComp, as seen in Figure A.7. If any of the commands have not been installed, it has to be installed first.

```
eva@EvaMogena:~/robocomp$ mkdir build
eva@EvaMogena:~/robocomp$ cd build/
eva@EvaMogena:~/robocomp/build$ cmake ..
El programa «cmake» no está instalado. Puede instalarlo escribiendo:
sudo apt-get install cmake
eva@EvaMogena:~/robocomp/build$ sudo apt-get install cmake
```

Figure A.7: Compiling robocomp

By building it may be a few errors because some packages needed to be installed,

Al compilarlo, puede que nos salgan errores porque falta por instalar algunos paquetes, in which case you must install them. For example, in my case the “lib-qt” package has to be installed, as Figure A.8 shows.

```
eva@EvaMogena:~/robocomp/build$ sudo apt-get install libqt4-dev
```

Figure A.8: “lib-qt” installation

It had to also to be installed the package “g++”, as can be seen in Figure A.9.

```
eva@EvaMogena:~/robocomp/build$ sudo apt-get install g++
```

Figure A.9: “g++” installation

And one important thing we need to install is “ipp”. For this we can directly download it from the intel website, or we can get into “robocloud” and download it. In our case we have chosen the second option, and we have downloaded the folder “opt”, that we have copied directly, as is shown in Figure A.10

```
eva@EvaMogena:~/Descargas$ ls
ipp_opt_64_32.tgz  l_ipp_em64t_p_6.1.2.051  l_ipp_em64t_p_6.1.2.051.zip  opt
eva@EvaMogena:~/Descargas$ sudo cp -r opt/intel/ /opt/
```

Figure A.10: “ipp” installation

As we have copied the folder directly, we need to add a new line to the “bashrc” file, so it knows that the files relating to “ipp” are stored in the folder “opt”. We can see the new line added in Figure A.11.

```
export ROBOCOMP='/home/eva/robocomp'
export IPPROOT=/opt/intel/ipp/6.1.1.042/em64t/
```

Figure A.11: New line to link “ipp” with “opt” folder in “bashrc”

In that way we are seeing the errors as we compile, as can be seen in Figure A.12. In this figure we can see that RoboComp has problems finding the package “gsl”.

```
-- Build files have been written to: /home/eva/robocomp/build
eva@EvaMogena:~/robocomp/build$ make -j3
[ 6%] [ 6%] Generating include/osgviewer/moc_viewerqt.cxx
Scanning dependencies of target robocomp_qmat
Generating include/osgviewer/moc_adapterwidget.cxx
[ 10%] Generating include/osgviewer/moc_findnamednode.cxx
/home/eva/robocomp/libs/osgviewer/include/osgviewer/findnamednode.h:0: Note: No relevant classes found. No output generated.
[ 13%] Generating include/osgviewer/moc_getworldcoorofnode.cxx
/home/eva/robocomp/libs/osgviewer/include/osgviewer/getworldcoorofnode.h:0: Note: No relevant classes found. No output generated.
[ 17%] Generating include/osgviewer/moc_osgview.cxx
[ 20%] Building CXX object libs/qmat/CMakeFiles/robocomp_qmat.dir/qfundamental.cpp.o
[ 24%] Scanning dependencies of target robocomp_osgviewer
Building CXX object libs/qmat/CMakeFiles/robocomp_qmat.dir/qessential.cpp.o
[ 27%] Building CXX object libs/osgviewer/CMakeFiles/robocomp_osgviewer.dir/adapterwidget.cpp.o
In file included from /home/eva/robocomp/libs/osgviewer/adapterwidget.cpp:1:0:
/home/eva/robocomp/libs/osgviewer/include/osgviewer/adapterwidget.h:4:28: fatal error: osgViewer/Viewer: No existe el archivo o el directorio
#include <osgViewer/Viewer>
^
compilation terminated.
make[2]: *** [libs/osgviewer/CMakeFiles/robocomp_osgviewer.dir/adapterwidget.cpp.o] Error 1
make[1]: *** [libs/osgviewer/CMakeFiles/robocomp_osgviewer.dir/all] Error 2
make[1]: *** Se espera a que terminen otras tareas...
[ 31%] Building CXX object libs/qmat/CMakeFiles/robocomp_qmat.dir/qcamera.cpp.o
In file included from /home/eva/robocomp/libs/qmat/include/qmat/qcamera.h:24:0,
                from /home/eva/robocomp/libs/qmat/qcamera.cpp:20:
/home/eva/robocomp/libs/qmat/include/qmat/qmat.h:30:26: fatal error: gsl/gsl_math.h: No existe el archivo o el directorio
#include <gsl/gsl_math.h>
^
compilation terminated.
In file included from /home/eva/robocomp/libs/qmat/include/qmat/qfundamental.h:24:0,
                from /home/eva/robocomp/libs/qmat/qfundamental.cpp:20:
/home/eva/robocomp/libs/qmat/include/qmat/qmat.h:30:26: fatal error: gsl/gsl_math.h: No existe el archivo o el directorio
#include <gsl/gsl_math.h>
^
In file included from /home/eva/robocomp/libs/qmat/include/qmat/qessential.h:24:0,
                from /home/eva/robocomp/libs/qmat/qessential.cpp:20:
/home/eva/robocomp/libs/qmat/include/qmat/qmat.h:30:26: fatal error: gsl/gsl_math.h: No existe el archivo o el directorio
#include <gsl/gsl_math.h>
^
compilation terminated.
compilation terminated.
make[2]: *** [libs/qmat/CMakeFiles/robocomp_qmat.dir/qessential.cpp.o] Error 1
make[2]: *** Se espera a que terminen otras tareas...
make[2]: *** [libs/qmat/CMakeFiles/robocomp_qmat.dir/qcamera.cpp.o] Error 1
make[2]: *** [libs/qmat/CMakeFiles/robocomp_qmat.dir/qfundamental.cpp.o] Error 1
make[1]: *** [libs/qmat/CMakeFiles/robocomp_qmat.dir/all] Error 2
make: *** [all] Error 2
eva@EvaMogena:~/robocomp/build$
```

Figure A.12: Errors after compiling

Therefore we also install this package looking for the “gsl” packages in the cache memory, as is shown in Figure A.13.

```
eva@EvaMogena:~/robocomp/build$ apt-cache search gsl
libgsl0-dbg - GNU Scientific Library (GSL) -- debug symbols package
libgsl0-dev - GNU Scientific Library (GSL) -- development package
libgsl0ldbl - GNU Scientific Library (GSL) -- library package
libcolt-java - scalable scientific and technical computing in Java
gsl-doc-info - Biblioteca científica GNU (GSL) Manual de referencia en formato info
gsl-doc-pdf - Biblioteca científica GNU (GSL) Manual de referencia en formato PDF
libcolt-java-doc - scalable scientific and technical computing in Java (doc)
gsl-bin - GNU Scientific Library (GSL) -- binary package
gsl-ref-html - GNU Scientific Library (GSL) Reference Manual in html
gsl-ref-psdoc - GNU Scientific Library (GSL) Reference Manual in postscript
libghc-hmatrix-dev - Linear algebra and numerical computation in Haskell
libghc-hmatrix-doc - Linear algebra and numerical computation in Haskell; documentation
libghc-hmatrix-prof - Linear algebra and numerical computation in Haskell; profiling libraries
libpdl-stats-perl - collection of statistics modules in Perl Data Language
libroot-math-mathmore-dev - GSL interface library for ROOT - development files
libroot-math-mathmore5.34 - GSL interface library for ROOT
python-guppy - Python programming environment Guppy-PE
ruby-distribution - Ruby library to work with probability distributions
ruby-gsl - Ruby bindings for the GNU Scientific Library (GSL)
ruby-integration - Numerical integration for Ruby, with a simple interface
ruby-settingslogic - simple settings solution for Ruby
ruby-settingslogic-doc - simple settings solution for Ruby (documentation)
yorick-ygsl - GSL special functions plugin for the Yorick language
libocamlgsl-ocaml - Biblioteca científica GNU para OCaml
libocamlgsl-ocaml-dev - Biblioteca científica GNU para OCaml
libranlip-dev - generates random variates with multivariate Lipschitz density
libranlipc2 - generates random variates with multivariate Lipschitz density
octave-gsl - Vínculos GSL para Octave
oggvideotools - Un conjunto de herramientas para manipular y crear archivos de vídeo Ogg
oggvideotools-dbg - A toolbox for manipulating and creating Ogg video files (debug symbols)
ruby-gsl-dbg - Ruby bindings for the GNU Scientific Library (GSL) - debugging symbols
slang-gsl - GNU Scientific Library binding for S-Lang
yorick-yeti-gsl - GSL special functions plugin for the Yorick language
```

Figure A.13: Looking for “gsl” package

And we install the “lib-gsl0-dev” package, as we can see in Figure A.14.

```
eva@EvaMogena:~/robocomp/build$ sudo apt-get install lib-gsl0-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
E: No se ha podido localizar el paquete lib-gsl0-dev
eva@EvaMogena:~/robocomp/build$ sudo apt-get install libgsl0-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
  kde-l10n-engb kde-l10n-es
Use 'apt-get autoremove' to remove them.
Se instalarán los siguientes paquetes extras:
  libgsl0ldbl
Paquetes sugeridos:
  gsl-ref-psdoc gsl-doc-pdf gsl-doc-info gsl-ref-html
Se instalarán los siguientes paquetes NUEVOS:
  libgsl0-dev libgsl0ldbl
0 actualizados, 2 se instalarán, 0 para eliminar y 9 no actualizados.
Necesito descargar 1.860 kB de archivos.
Se utilizarán 9.126 kB de espacio de disco adicional después de esta operación.
¿Desea continuar? [S/n]
```

Figure A.14: “gsl” installation

As it also fails to find the “osg” package, as Figure A.15 shows, we install it, as we can see in Figure A.16.

```
In file included from /home/eva/robocomp/libs/osgviewer/adapterwidget.cpp:1:0:
/home/eva/robocomp/libs/osgviewer/include/osgviewer/adapterwidget.h:4:28: fatal error: osgViewer/Viewer: No existe el archivo o el directorio
#include <osgViewer/Viewer>
^
compilation terminated.
```

Figure A.15: Errors with “osg” package

```
eva@EvaMogena:~/robocomp/build$ apt-cache search openscenegraph
libcitygml0 - Open source C++ library for parsing CityGML files
libcitygml0-bin - Utils of libcitygml - citygml2vrmf and citygmltest
libcitygml0-dev - Static and header files of libcitygml
libosgearth2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (shared lib)
libosgearthannotation2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (osgEarthAnnotation)
libosgearthfeatures2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (osgEarthFeatures)
libosgearthqt2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (osgEarthQt)
libosgearthsymbology2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (osgEarthSymbology)
libosgearthutil2 - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (osgEarthUtil)
openscenegraph - 3D scene graph, utilities and examples (binaries)
openscenegraph-examples - 3D scene graph, examples (sources)
openscenegraph-plugin-citygml-shared - libcitygml OpenSceneGraph plugin (shared version)
openscenegraph-plugin-citygml-static - libcitygml OpenSceneGraph plugin (static version)
openscenegraph-plugin-osgearth - OpenSceneGraph plugins for osgEarth
osgearth - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (binaries)
osgearth-data - Dynamic 3D terrain rendering toolkit for OpenSceneGraph (data)
libopenscenegraph-dev - 3D scene graph, development files
libopenscenegraph99 - 3D scene graph, shared libs
libopenthreads-dev - Object-Oriented (OO) thread interface for C++, development files
libopenthreads14 - Object-Oriented (OO) thread interface for C++, shared libs
libosgearth-dev - osgEarth development files
openscenegraph-doc - 3D scene graph, documentation
eva@EvaMogena:~/robocomp/build$ sudo apt-get install libopenscenegraph-dev
```

Figure A.16: “osg” installation

Thus we will install all packages that give us error after compiling, until we find no error in the compilation. If we find no more error when compiling, we can install everything from the main RoboComp folder, as is shown in Figure A.17.

```
eva@EvaMogena:~/robocomp$ sudo make install
[ 51%] Built target robocomp_osgviewer
[ 89%] Built target robocomp_qmat
[100%] Built target robocomp_innermodel
Install the project...
-- Install configuration: ""
-- Installing: /opt/robocomp-1.0/lib/librobocomp_osgviewer.so.1.0
-- Installing: /opt/robocomp-1.0/lib/librobocomp_osgviewer.so
-- Installing: /opt/robocomp-1.0/include/osgviewer/adaptewidget.h
-- Installing: /opt/robocomp-1.0/include/osgviewer/findnamednode.h
-- Installing: /opt/robocomp-1.0/include/osgviewer/getworldcoorofnode.h
-- Installing: /opt/robocomp-1.0/include/osgviewer/osgview.h
-- Installing: /opt/robocomp-1.0/include/osgviewer/viewerqt.h
-- Installing: /opt/robocomp-1.0/lib/librobocomp_innermodel.so.1.0
-- Installing: /opt/robocomp-1.0/lib/librobocomp_innermodel.so
-- Removed runtime path from "/opt/robocomp-1.0/lib/librobocomp_innermodel.so.1.0"
-- Installing: /opt/robocomp-1.0/include/innermodel/innermodel.h
-- Installing: /opt/robocomp-1.0/include/innermodel/innermodelreader.h
-- Installing: /opt/robocomp-1.0/include/innermodel/innermodelviewer.h
-- Installing: /opt/robocomp-1.0/lib/librobocomp_qmat.so.1.0
-- Installing: /opt/robocomp-1.0/lib/librobocomp_qmat.so
-- Installing: /opt/robocomp-1.0/include/qmat/qvec.h
-- Installing: /opt/robocomp-1.0/include/qmat/qessential.h
-- Installing: /opt/robocomp-1.0/include/qmat/qrtmat.h
-- Installing: /opt/robocomp-1.0/include/qmat/qmovingrobot.h
-- Installing: /opt/robocomp-1.0/include/qmat/qfundamental.h
-- Installing: /opt/robocomp-1.0/include/qmat/qmatrot.h
-- Installing: /opt/robocomp-1.0/include/qmat/qcamera.h
-- Installing: /opt/robocomp-1.0/include/qmat/qhomo.h
-- Installing: /opt/robocomp-1.0/include/qmat/qextrinsics.h
-- Installing: /opt/robocomp-1.0/include/qmat/qmat.h
-- Installing: /opt/robocomp-1.0/include/qmat/quaternion.h
-- Installing: /opt/robocomp-1.0/include/qmat/qline2d.h
-- Installing: /opt/robocomp-1.0/include/qmat/QMatAll
eva@EvaMogena:~/robocomp$
```

Figure A.17: Instalación del proyecto

There is a special case in the installation: in the case that one of the errors given by compiling is with “innermodel”, as we can see in Figure A.18, we must do the following. If we do not get any error with “innermodel”, we can install it directly and we can skip these steps.

```
[ 79%] [ 89%] Built target robocomp_qmat
Built target robocomp_osgviewer
[ 93%] [ 96%] Building CXX object libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodelreader.cpp.o
Building CXX object libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodel.cpp.o
[100%] Building CXX object libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodelviewer.cpp.o
/home/computaex/robocomp/libs/innermodel/innermodelviewer.cpp:20:35: fatal error: innermodel/innermodel.h: No existe el archivo o el directorio
#include <innermodel/innermodel.h>
^
/home/computaex/robocomp/libs/innermodel/innermodel.cpp:20:35: fatal error: innermodel/innermodel.h: No existe el archivo o el directorio
#include <innermodel/innermodel.h>
^
compilation terminated.
compilation terminated.
make[2]: *** [libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodelviewer.cpp.o] Error 1
make[2]: *** Se espera a que terminen otras tareas...
make[2]: *** [libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodel.cpp.o] Error 1
In file included from /home/computaex/robocomp/libs/innermodel/innermodelreader.cpp:20:0:
/home/computaex/robocomp/libs/innermodel/include/innermodel/innermodelreader.h:27:24: fatal error: innermodel.h: No existe el archivo o el directorio
#include "innermodel.h"
^
compilation terminated.
make[2]: *** [libs/innermodel/CMakeFiles/robocomp_innermodel.dir/innermodelreader.cpp.o] Error 1
make[1]: *** [libs/innermodel/CMakeFiles/robocomp_innermodel.dir/all] Error 2
make: *** [all] Error 2
computaex@computaex:~/robocomp/build$ apt-cache search innermodel
computaex@computaex:~/robocomp/build$ sudo apt-get install innermodel
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
E: No se ha podido localizar el paquete innermodel
computaex@computaex:~/robocomp/build$
```

Figure A.18: Problems with innermodel

We return to the main RoboComp folder, and run the “cmake-gui.” command, which compiles the configuration graphical interface. Gui is a graphical interface for cmake that allow us to work more comfortably when selecting the type of configuration according to the compiler or IDE that we use.

```
computaex@computaex:~/robocomp/build$ cd ..
computaex@computaex:~/robocomp$ cmake-gui
```

Figure A.19: “cmake-gui” installation

After that we compile and install RoboComp, as shown in Figure A.20.

```
computaex@computaex:~/robocomp$ cmake .
-- Not found OpenMP
-- PYTHON BINDINGS:      NO
-- Could NOT find Doxygen (missing: DOXYGEN_EXECUTABLE)
-- COMPILING WITHOUT FCL SUPPORT = 0
-- IPPROOT (IPP 6): /opt/intel/ipp/6.1.1.042/em64t/
-- IPP Activated
-- Configuring done
-- Generating done
-- Build files have been written to: /home/computaex/robocomp
computaex@computaex:~/robocomp$ make
[ 51%] Built target robocomp_osgviewer
[ 89%] Built target robocomp_qmat
[100%] Built target robocomp_innermodel
computaex@computaex:~/robocomp$ sudo make install
```

Figure A.20: RoboComp installation

If there are still some errors we go to the folder tools/rcinnerModelSimulator/ and compile the simulator, as we can see in Figure A.21.

```
computaex@computaex:~/robocomp$ cd tools/rcinnerModelSimulator/
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ cmake .
-- RoboComp root is now set to /home/computaex/robocomp/
-- SLICEROOT=""
--
--
-- $SLICE_PATH="/opt/robocomp/ThirdPartyinterfaces;/home/computaex/robocomp/interfaces/"
-- Adding /opt/robocomp/ThirdPartyinterfaces to the Slice directory set.
-- Adding /home/computaex/robocomp/interfaces/ to the Slice directory set.
-- Adding rule to generate Camera.cpp and Camera.h from /home/computaex/robocomp/interfaces//Camera.ice (slice2cpp)
-- Adding rule to generate CommonBehavior.cpp and CommonBehavior.h from /home/computaex/robocomp/interfaces//CommonBehavior.ice (slice2cpp)
-- Adding rule to generate CommonHead.cpp and CommonHead.h from /home/computaex/robocomp/interfaces//CommonHead.ice (slice2cpp)
-- Adding rule to generate DifferentialRobot.cpp and DifferentialRobot.h from /home/computaex/robocomp/interfaces//DifferentialRobot.ice (slice2cpp)
-- Adding rule to generate OmniRobot.cpp and OmniRobot.h from /home/computaex/robocomp/interfaces//OmniRobot.ice (slice2cpp)
-- Adding rule to generate IMU.cpp and IMU.h from /home/computaex/robocomp/interfaces//IMU.ice (slice2cpp)
-- Adding rule to generate InnerModelManager.cpp and InnerModelManager.h from /home/computaex/robocomp/interfaces//InnerModelManager.ice (slice2cpp)
-- Adding rule to generate JointMotor.cpp and JointMotor.h from /home/computaex/robocomp/interfaces//JointMotor.ice (slice2cpp)
-- Adding rule to generate Laser.cpp and Laser.h from /home/computaex/robocomp/interfaces//Laser.ice (slice2cpp)
-- Adding rule to generate RGBD.cpp and RGBD.h from /home/computaex/robocomp/interfaces//RGBD.ice (slice2cpp)
-- Adding rule to generate TouchSensor.cpp and TouchSensor.h from /home/computaex/robocomp/interfaces//TouchSensor.ice (slice2cpp)
-- RoboComp libraries
-- Adding library robocomp_qmat
-- IPPROOT (IPP 6): /opt/intel/ipp/6.1.1.042/em64t/
-- IPP Activated
-- Configuring done
-- Generating done
-- Build files have been written to: /home/computaex/robocomp/tools/rcinnerModelSimulator
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ make
```

Figure A.21: Simulator compilation

If it gives us compile errors, we install the necessary packages. In my case it has been necessary to install multiple, as Figure A.22 shows. Once we have installed all the packages, we compile and see that does not give more errors.

```
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ sudo apt-get install zeroc-ice35+
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
zeroc-ice35 ya está en su versión más reciente.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
 lib32tinfo5 libc6-1386
Use 'apt-get autoremove' to remove them.
0 actualizados, 0 se instalarán, 0 para eliminar y 36 no actualizados.
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ make
[100%] Built target rcis
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ sudo apt-get install freeglut3-dev libboost-system-dev
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
freeglut3-dev ya está en su versión más reciente.
libboost-system-dev ya está en su versión más reciente.
Los paquetes indicados a continuación se instalaron de forma automática y ya no son necesarios.
 lib32tinfo5 libc6-1386
Use 'apt-get autoremove' to remove them.
0 actualizados, 0 se instalarán, 0 para eliminar y 36 no actualizados.
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ make
[100%] Built target rcis
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ sudo make install
[100%] Built target rcis
Install the project...
-- Install configuration: ""
-- Up-to-date: /opt/robocomp/bin/rcis
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ █
```

Figure A.22: Installation missing packages for the simulator compiling

Once we have successfully compiled the simulator, we return to the tools folder in the main RoboComp folder and we try again the installation that should not cause problems, as is shown in Figure A.23.

```
computaex@computaex:~/robocomp/tools/rcinnerModelSimulator$ cd ..
computaex@computaex:~/robocomp/tools$ sudo make install █
```

Figure A.23: Tools folder installation

And once installed, due to a name change in the new version, now called “robocomp-1.0”, we must make to the old name, that is to say, to “robocomp”, a link that points to the new name so there are no problems. This can be seen in Figure A.24.

```
eva@EvaMogena:~/robocomp$ sudo ln -s /opt/robocomp-1.0/ /opt/robocomp
eva@EvaMogena:~/robocomp$ ls /opt/robocomp
include  lib
eva@EvaMogena:~/robocomp$ ls -la /opt/robocomp
lrwxrwxrwx 1 root root 18 oct 30 12:37 /opt/robocomp -> /opt/robocomp-1.0/
```

Figure A.24: Creating a link between the two RoboComp directions

In this way, we will not have problems with name changes from the old version to the new one.

We also have to make another link so there are no problems accessing the RoboComp folder, the way we see in Figure A.25.

```
eva@EvaMogena:~/robocomp/tools/rcinnerModelSimulator/build$ sudo ln -s /home/eva/ /home/robocomp
```

Figure A.25: Creating link to the RoboComp folder

Having done all this RoboComp would be installed.

As the new version of robocomp not include the components, we then need to install the components we want to work with. In my case I will install the folder “robocomp-robolab”, where there are a number of basic components with which the laboratory works. We can see how this is done in Figure A.26.

```
eva@EvaMogena:~/robocomp$ ls
classes  CMakeCache.txt  cmake_install.cmake  components  install_manifest.txt  libs  Makefile  tools
cmake    CMakeFiles      CMakeLists.txt      files       interfaces           LICENSE  README.md  uninstall_target.cmake
eva@EvaMogena:~/robocomp$ cd components/
eva@EvaMogena:~/robocomp/components$ ls
eva@EvaMogena:~/robocomp/components$ git clone https://github.com/robocomp/robocomp-robolab.git
Clonar en «robocomp-robolab»...
remote: Counting objects: 1061, done.
Receiving objects: 46% (489/1061), 260.00 KiB | 221.00 KiB/s
```

Figure A.26: Installation of RoboComp components

With this folder we make the process of compilation and installation of the required packages until no error is get.

This way we can install the components we need, and thus completing our RoboComp to work with it.

Appendix B

RoboComp Component Creation Manual

To create a new RoboComp component it is needed the DSLEditor software. Download it and install it in the path `/robocomp/Tools`. Open it and add or create a project. In this case, the project is `robocomp`.

Add a new folder in `robocomp/Experimental` named like the component and ended in `Comp`, for example “`therapyComp`”. Inside this folder. Within this folder the `dsl` files are going to be kept.

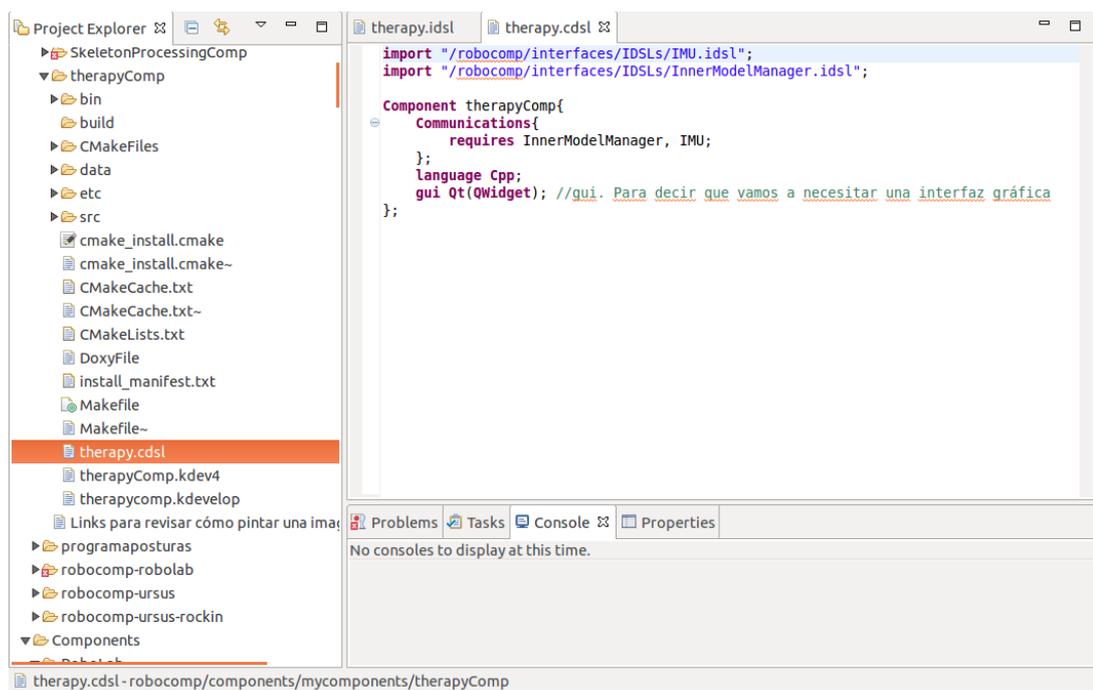


Figure B.1: `.cdsl` file in *DSLEditor*.

Now, it is time to create the *.cdsl* file. Open the *DSLEditor* and, in the *robocomp* menu choose *new* and *cdslfile*, name it like the component without the 'Comp' termination. You have to see something like FigureB.1.

You have to add the first line to include the interface of the components which you are going to communicate with, in this case is the “InnerModelManager” and the “IMU” interfaces. You have to include the classes and libraries you are going with you are going to work. If you need to add more classes or libraries in the future you can include them in your C++ editor. Finally, if your component is going to show a graphic interface, include the penultimate line. Then you will create that graphic interface with the *Qt4Designer* software. Create also a *.pdsl* and a *.ddsl*.

Inside the */Interface* folder create the *.idsl* file the same way as the *.cdsl* file. This file is related to the interface to communicate with other components.

Finally, in the *robocomp* menu, choose *GenerateCode* and *CompileComponent* and your component would be already created. Remember that before the first time you open your component is necessary to execute *cmake*. and *make*.

Check if the *.cdsl* file is contained in the component folder, and inside the *src* folder, look for the *gui* file, wich corresponds to the interface, and for the *specificworker.cpp* and the *specificworker.h*.

Appendix C

Graphic Interface Creation Manual

To explain how to create a graphic interface for a component the therapyComp interface is going to be used as an example. See Figure C.1 to have a global view of the component.

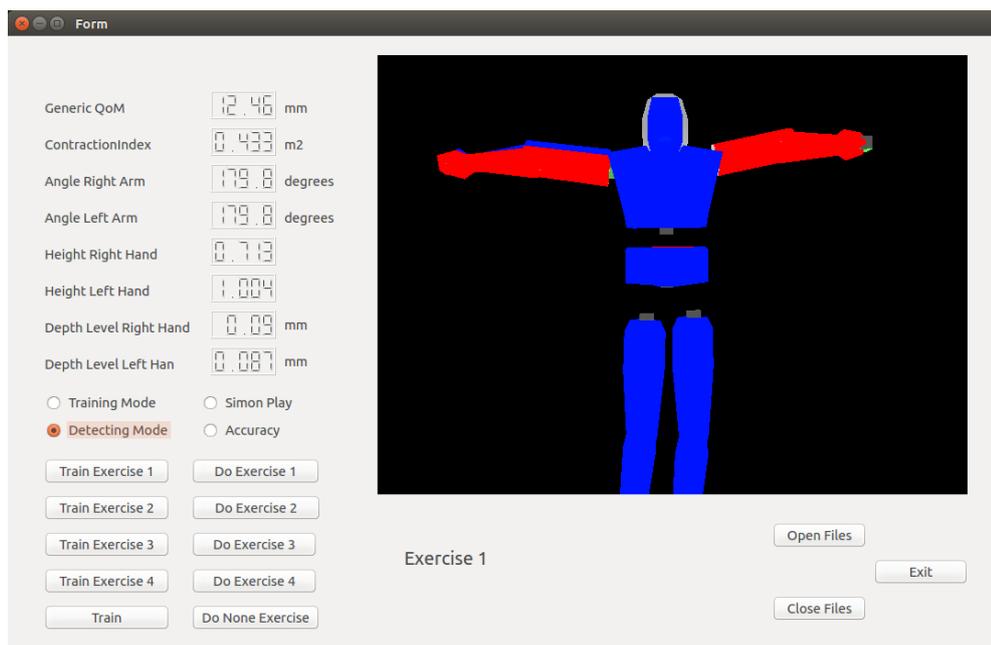


Figure C.1: therapyComp interface

Let's look at this piece by piece.

At first, when open the programm, choosing *new* and *create widget*, you should see something like Figure C.2.

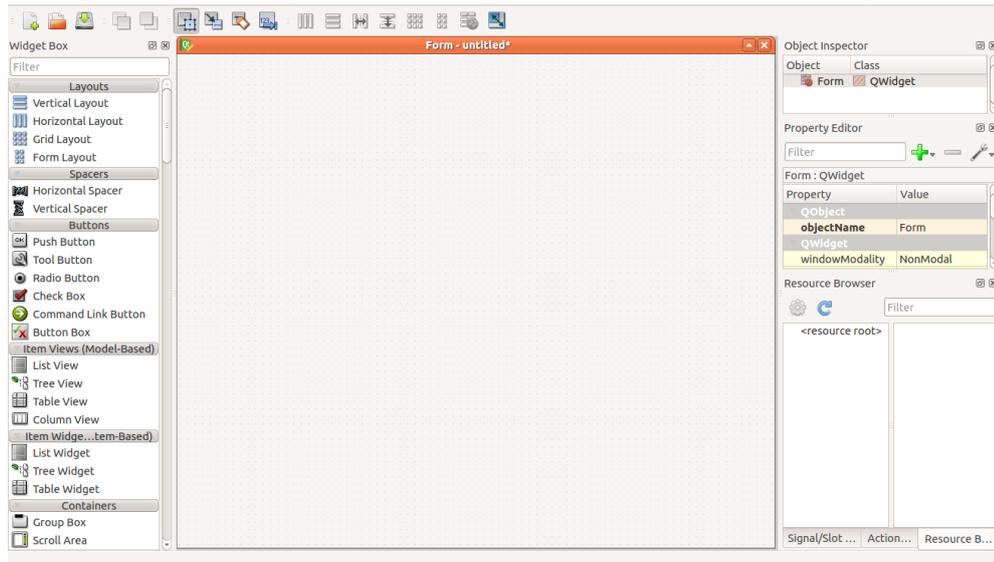
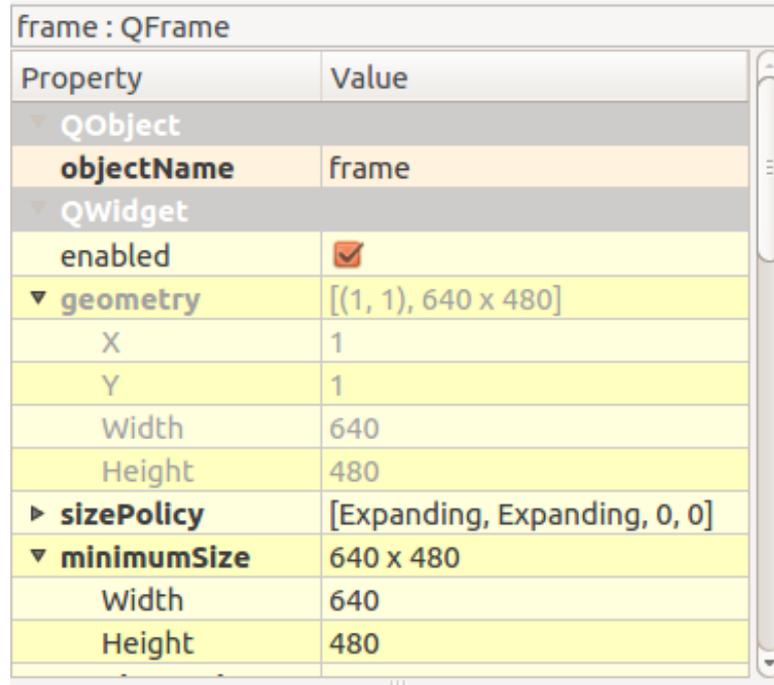


Figure C.2: Qt4Editor main display

To include any item, firstly you have to drag a *grid layout* from the left menu where every object is classified on categories and put the items inside. Looking at the *GestureDetectComp* interface we see several different items such as *bottoms*, *QLCDNumbers* (displays), a *combobox* (selector), a *Qframe* (to show the RGB image) and *QLabels* (text). *QLabels* do not need to be put inside a *grid layout*.

To configure each object click on it and go to the right side of the display. You will find a window like Figure C.3, where you can set up the name and the characteristics of the item, for example, the size. To edit the description of an object, in other words, what is written inside a bottom or a label, make double click on it.



Property	Value
frame : QFrame	
▼ QObject	
objectName	frame
▼ QWidget	
enabled	<input checked="" type="checkbox"/>
▼ geometry	[(1, 1), 640 x 480]
X	1
Y	1
Width	640
Height	480
▶ sizePolicy	[Expanding, Expanding, 0, 0]
▼ minimumSize	640 x 480
Width	640
Height	480

Figure C.3: Qt4Editor: item set up

Finally, you only have to connect each item with the current parameters on your code. Figure C.4 shows examples for the bottoms and the displays.

```
connect(ButtonOpenFiles, SIGNAL(clicked(bool)),this, SLOT(OpenFiles(bool)));
connect(ButtonCloseFiles, SIGNAL(clicked(bool)),this, SLOT(CloseFiles(bool)));
connect(ButtonExit, SIGNAL(clicked(bool)),this, SLOT(Exit(bool)));

connect(ButtonExercise1, SIGNAL(clicked(bool)),this, SLOT(Exercise1(bool)));
connect(ButtonExercise2, SIGNAL(clicked(bool)),this, SLOT(Exercise2(bool)));
connect(ButtonExercise3, SIGNAL(clicked(bool)),this, SLOT(Exercise3(bool)));
connect(ButtonExercise4, SIGNAL(clicked(bool)),this, SLOT(Exercise4(bool)));
connect(ButtonTrain, SIGNAL(clicked(bool)),this, SLOT(Train(bool)));
//connect(ButtonAccuracy, SIGNAL(clicked(bool)),this, SLOT(Accuracy(bool)));

connect(ButtonDoExercise1, SIGNAL(clicked(bool)),this, SLOT(DoExercise1(bool)));
connect(ButtonDoExercise2, SIGNAL(clicked(bool)),this, SLOT(DoExercise2(bool)));
connect(ButtonDoExercise3, SIGNAL(clicked(bool)),this, SLOT(DoExercise3(bool)));
connect(ButtonDoExercise4, SIGNAL(clicked(bool)),this, SLOT(DoExercise4(bool)));
connect(ButtonDoExercise0, SIGNAL(clicked(bool)),this, SLOT(DoExercise0(bool)));

innerModel = new InnerModel("/home/computaex/robocomp/files/innermodel/person.xml");
frame->show();
```

Figure C.4: How to connect interface items in your code

