UNIVERSIDAD CARLOS III DE MADRID
ESCUELA POLITÉCNICA SUPERIOR
BACHELOR IN COMPUTER SCIENCE AND ENGINEERING

BACHELOR THESIS

Development of the mirror system with humanoid robots

Author
D. Rubén Martín Vázquez
Tutor
Dr. D. Fernando Fernández Rebollo

September 2014

Never regard study as a duty, but as the enviable opportunity to learn to know the liberating influence of beauty in the realm of the spirit for your own personal joy and to the profit of the community to which your later work belongs.

Albert Einstein (1879-1955)

Agradecimientos

Quiero darle las gracias de todo corazón:

A mis padres, Ventu y Pepi, por darme todo su cariño, esfuerzo y la oportunidad de estudiar lo que realmente me gusta.

A mi hermano César por haber sido mi inspiración y ejemplo a seguir desde el instante en el que nací.

A mi novia Anita por todo su amor, por estar siempre ahí, por alejar mi mente de los estudios cuando necesitaba despejarme y motivarme cuando los ánimos escaseaban.

A Almudena por haberme apoyado siempre, incluso cuando ni la lógica estaba de mi parte.

A mis compañeros de clase por haberme ayudado a llegar hasta aquí tanto con sus conocimientos como con su buen humor.

A mis amigos por soportar mis épocas de estrés siempre con una sonrisa en la cara.

A mis profesores, especialmente a aquellos que ven en la enseñanza algo más que un trabajo.

Abstract

This project focuses in the concept of robot teleoperation and how to make it easier for any type of user.

The use of robots requires in most of the cases high knowledge in the area which restrict their use to people specifically prepared for the task. For that reason it is essential for everybody the research for controlling robots using simpler techniques approaching the robotics for everybody.

In this project, the user will use the simplest way to control the robot arms and using his/her own arms in the same way that he/she wants to move the robot arms. For reaching that goal, the project assumes the use of Microsoft Kinect device that capture everything the user does. Then a computer will process it and send the information to the robot, in this case a NAO robot.

For that reason, the teleoperation is simple and possible even to old people or children. This type of interface opens the door to infinite possibilities and creates a solid foundation on which to establish future knowledge.

Keywords

Teleoperation System, Natural Interaction, Robotics, Move Recognition, Low-Level Interface Control, NAO Robot, Microsoft Kinect.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the beginning of the humanity the people has used the brain to create tools and find ways to have an easier life. This process is permanent since the beginning until the present. One of the most important 'tools' that the human created was the computing. It is in every place anyone can go and everyday people use it in every continent.

When computing was born, just some people could use it because of the cost and required knowledge. It was very expensive to buy a computer (where a computer is any tool that computes) and the knowledge needed to use one was very high.

Scientists continued studying and improving the computers and its components and, for that reason, the price of the computers dropped and more people had the opportunity to buy one and learn how to use it. For this reason, and the hard work from new companies and researchers, the computers went more popular.

At the present, it's strange to see any house of the developed countries without at least one computer that is used every single day. In fact, there are many types of computing devices that are not computers but has their capability. Other 'tool' that is even easier to see in any house is the smartphone, mobile phones that are like small computers. The smartphones are very popular, in Spain, in just one year (from 2012 until 2013) the number of people with smartphones increases to the double; from 12 million of people in 2012 to 24 millions of people in 2014 [1][2].

Having robots in any house, at the present, is also seen like science-fiction but it can be expected that in the future it will come true [3].

Robotics was born to help in some industrial works that were very hard or dangerous for people. The success of the robots used in those situations made possible the creation of new robots, this time not only for dangerous jobs, but for optimizing work doing it faster and better than humans.

But there are not only industrial robots: there are several types of robots that can be classified by its objective, form, intelligence…

One of the most wished robots are the service robots and, for that reason, they are been especially studied for some time ago [4]. Most of the robots just do one task, but robots can do more than one task in different areas.

The humanoid robots are robots with human form that are in the service robots groups and they can be very useful. Since have a body that is similar to human one they can do most of the tasks that people usually do without any type of barrier.
They can also interact with most of the instruments that people usually use like doors, buttons or tools.

Although the service robots are the most wanted, currently the type of robot with the biggest number of units are the industrials. The second type are the tele- operated (or teleoperated) in spite of the autonomous are more wished. This is due to the artificial intelligence has not gone so far to create robots that can be totally autonomous [5].

Teleoperated robots are the one which need to be controlled by a user at the distance. These robots are very useful in dangerous environments because the user can be in a safe place while the robot is doing the work far away. Some examples of these situations are the robots used by the explosive experts to detonate explosives [6], robotic vehicles sent to other planets (rovers) [7] [8] or robots that helped to clean and repair the nuclear power plant of Fukushima [9].

In this project, a teleoperation system will be developed with the objective of controlling NAO arms and forearms. The idea is to create a system that anyone can use without advance knowledge. For that reason, the system will capture the movements that the user will do and send it to the NAO robot to do them in real time.
Using that way, any person can use it, children, teenagers, young people, mature people and even old people that do not know anything about computers or robotics.

For capturing the image, a Microsoft Kinect device will be used. The operating system selected is Microsoft Windows 7 because it is one of the most extended operating systems in the world and it has a friendly interface.

1.1. Project Objectives

The objective of this project is the creation of a natural interface for the teleoperation between a user and a humanoid robot. For the natural interface creation in this project the Kinect device created by Microsoft and the humanoid robot NAO created by Aldebaran Robotics will be used. Connecting both elements requires a computer with the operating system Windows 7 that will have installed the NAOqi software, for connecting with the robot, and Visual Studio 2013 for running the code that will process all the information.

Using the natural interface, the robot will imitate any movement that the user realizes with the arms and forearms. The user will have a screen where the skeleton image detected by the Kinect can be checked. It will be also notified if in any moment a hand, arm or forearm is out of bound for making easier the teleoperation and avoid any type of error.

The main purpose of the project is that any person, without regard the age, can use the teleoperation system needing only some basic information about the system. The user should be able to manage the robot correctly with just some instructions.

To achieve that goal, it has been divided in smaller subgoals that are more specific:

- Study Visual Studio, learn the different useful options and all the possibilities to make a project. It will guarantee that the system is optimized and work correctly.
- Analyze the different components published by Microsoft and the internet community about Kinect and its functionalities.
- Study how the NAO robot works. Study how it connects to the computer, its physics attributes, the way it computes the information, delays and the basic information needed to make it works, like the moves of the joints in a correct way.
- It is important to look for documentation about the libraries, modules and functions that can be re-used or modified.
- Design a high level system correctly divided in classes to make possible to reuse the classes if it is needed.
- Design the teleoperation procedure knowing that it is very important the ease of the system and the high usability. It is also important to get the smallest delays possible.

- Implement the teleoperation system, making the tests necessaries to get the best possible result. If the tests are not satisfactory the implementation must be changed, or even the design, to achieve the best result.

  It is indispensable that the code be commented as most as possible to provide an easier maintenance.
- Document the most relevant data and the information that can help to understand the project to the people who want to study deeply the project or to develop new functionalities.

1.2. Document Structure

Below these lines, the document structure is detailed, naming the different sections and giving a brief explanation for each one.

1. Introduction

   In this section, there will be a brief introduction. Then, it will be explained the motivations for doing the project. It will also detail the reasons why the project is necessary. To finish, the objectives needed to make a good project will be listed.

2. State of the art

   In the state of the art section the project is situated in the technologic environment it is developed. It starts with information and attributes of the NAO robot followed by the teleoperation basics, elements and areas where it is applied. To finish, the Microsoft Kinect device is detailed and its characteristics are named.

3. System design and implementation

   In the system design and implementation section the control system is described. First the methodology used is explained and then, an analysis of the system, the requirements catalogue and user cases are reported.

   To finish, in the design part, the architecture is explained with the different components of the whole system.

4. Evaluation

   In the evaluation section, a report about the system evaluation can be found with different environments and types of tests: unitary tests, integration testing and requirements tests.

   These tests are explained with its results and the requirements that are accomplished.

5. Project management

   In the project management section there are two main points. One is the planning of the project divided in the different parts of the development. The second point is the budget which is divided and detailed offering the information of every cost in the project development.

6. Conclusions and further development

   In the conclusions and further development section, the final conclusions of the project are presented. The different implementations of the project are also included in this section.

7. Appendices

   In the appendices section, the different appendices that are necessary or important for the understanding of the whole project are included.

   Some of those appendices are the user guide, the glossary, the acronyms list, extended information…

Chapter 2

State of the Art

In this chapter the theoretical concepts needed to understand in a correct way the characteristics and objectives of the project are explained. For that reason, all the different elements that are related with the project are detailed and analyzed in the following lines.

2.1. NAO Robot

Bruno Maisonnier, the founder of Aldebaran Robotics, wanted to create robots that help humankind [10]. Another wish from Bruno Maisonnier is that everybody can have a robot in their life. For that reason, he decides to create NAO.

Nao was born for that purpose, provides a hardware and software to help everybody to research in robotics or just to have a robot with a not expensive price.

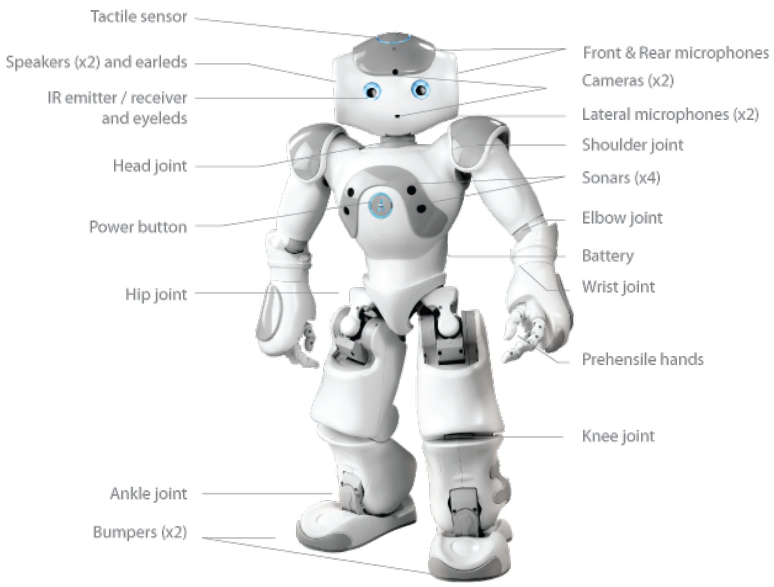


Figure 1. NAO Robot

The height of the NAO is 58 cm, as it is shown in Figure 1, and its weight is 4.8 kg. NAO has an Intel ATOM 1.6 GHz CPU located in the head and a second processor located in the torso. NAO runs Linux kernel and supports NAOqi, the Aldebaran's proprietary middleware.

NAO has a 48.6 watt-hour battery that provides it with more than 1.5 hours of autonomy. It can be connected to other devices and NAO's using Wi-Fi (b g n), by Ethernet and using Infrared [11].

Actuators

NAO has 26 degrees of freedom (DOF) that are distributed in the following way:

- 2 DOF in the head.
- 4 DOF in each arm.
- 2 DOF in each hand.
- 6 DOF in each leg.

In Figure 2 the different degrees of freedom are shown.

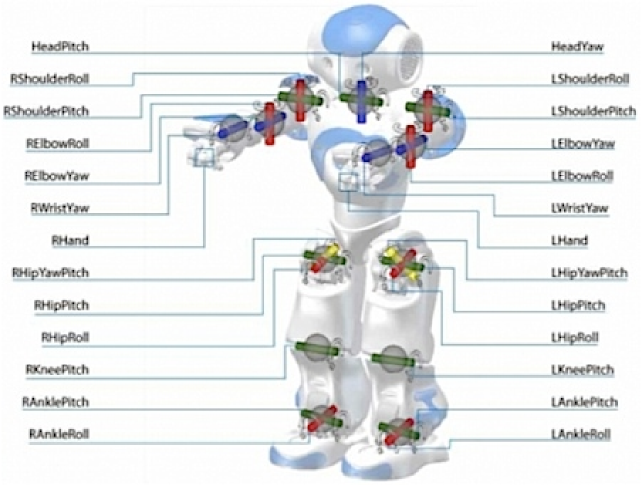


Figure 2. NAO Joints

NAO has two LED lights in the head and 2 high-fidelity speakers.

Sensors

NAO has 4 microphones and 2 cameras in the head. As it can be seen in Figure 3, one camera pointes forwards and the other one is lightly bended.

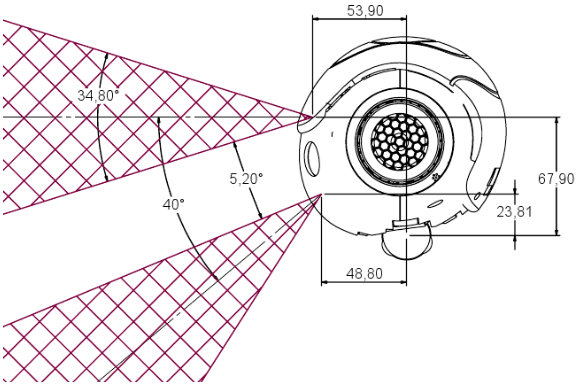


Figure 3. Nao Cameras

The robot also has several pressure sensors, a button in the chest, two bumper buttons in each foot, 3 tactile sensors in the head and another 3 tactile sensors in each hand. It has 8 force-sensing resistors (FSR). These sensors are in the feet and measure the resistance in different points to give NAO more equilibrium.

It also has 2 sonar channels, 2 gyro sensors and an accelerometer.

Since August, 15th 2007 NAO is the robot that compete in the RoboCup (Robot soccer World Cup). The RoboCup is a World Cup where robots play soccer matches. Thanks to the RoboCup and to the big amount of possibilities that, NAO Robot becomes very popular

Figure 4. NAO in the RoboCup

NAOqi

NAOqi is the name of the software that runs on the robot processor and the NAOqi Framework is the programming framework used to program NAO. It covers the common robotics needs like: parallelism, resources, synchronization and events.

This framework allows homogeneous communication between different modules, homogeneous programming and homogeneous information sharing.

Some of the characteristics from the NAOqi framework are the following:

- It is cross-platform, so the code can be developed in Windows, Linux or MacOS.
- It is cross-language, with an identical API for both C++ and Python.
- It also provides introspection, which means the framework knows which functions are available in the different modules and where.

2.2. Teleoperation

Since the beginning, the humanity has searched for the way to have more control of the environment. To achieve that goal, human beings have used any type of tools: First, basic tools like sticks to reach unachievable places or stones to break other materials. Later, the knowledge and the tools that they already have helped to create new tools that were more sophisticated and complex.

This evolution of the tools and knowledge allowed people to make real the teleoperation.

2.2.1. Teleoperation system

Teleoperation is the action of manipulate a robotic device at a distance.

Every teleoperation system has at least, the following elements:

- Operator: the operator is the user who controls the teleoperated device at the distance. There are two types of control:
    - The operator controls every action of the device.
    - The operator gives to the device a target or objective and the device reaches them using its algorithms.
- Teleoperated device: the teleoperated device is the device that the operator controls. It can be a robot, vehicle, mechanical arm or similar devices. From now on, in this document, the terms device and robot will be used like synonymous with teleoperated device.
- Interface: the interface is the element that communicates the operator and the teleoperated device. Some of the interfaces are the instruments that the operator uses to control the robot, the elements that report information from the device and its state. For example, joysticks, cameras or buttons.
- Communication Channel: the communication channel is the way where information travels between the operator and the teleoperated device (in both directions). It can be wires, wireless or both.
- Control: the control part is the one which traduces the different signals and orders between the operator and the teleoperated device. For example, if the operator pushes a button, the control module will convert it in a signal than the robot can understand.

- Sensors: the sensors are the elements which compile the information and send it to the interface.

The interface is the most important element in the system and for that reason this thesis will go in depth into the interfaces.

2.2.2. Interface

The interface in a teleoperation system is composed of control devices and feedback devices [12] [13]. Control devices are used by the operator to generate the different commands. Feedback devices give to the operator information about the device and the environment. There are devices with both functionalities; these devices are called bilateral devices.

2.2.2.1. Control Devices

Control devices, as it was said in the point before, are the devices that create and send the commands to the robot. There are two main types:

- Low-level interface control

The low-level interface control devices usually control the joints movements of the skeleton [14].

- High-level interface control

The high-level interface control devices focus in tasks, for example, going from point A to point B.

2.2.2.2. Feedback Devices

The Feedback devices gives to the operator the information related to the robot and its environment. As every stimulus, it is received by the senses. The most important feedback stimuli are received using the:

- Sight: Using the sight, the operator can see what the robot sees or even the environment of the robot, it can be done using cameras in the robot or near to its. These cameras can be video cameras, range cameras, ultrasonography cameras or any type [15] [16].

The sight is the most common sense used in feedback devices.

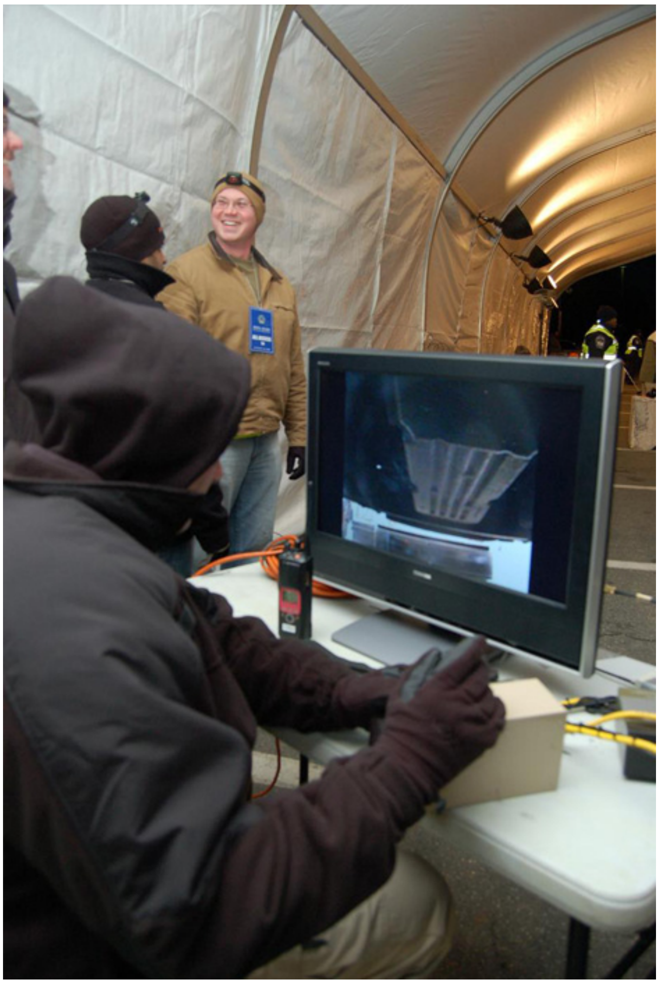


Figure 5. A teleoperated system using a camera as a feedback

- Touch: Using the touch the operator can feel the force that the robot is receiving when it touches something or hits with an object. This is known as haptic technology [17].

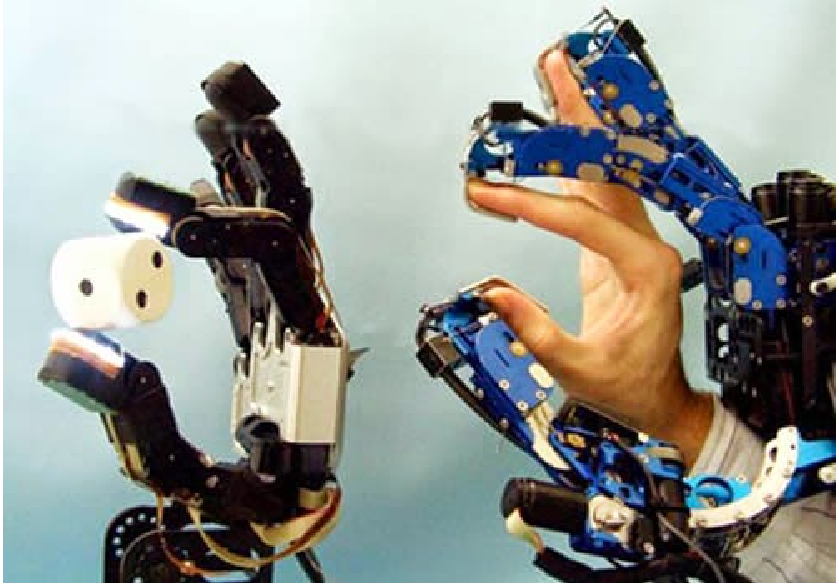


Figure 6. Haptic technology example

- Hearing: Using the hearing, the operator can receive information from the computer or different alarms. It is comfortable because the operator does not have to be focused in the hearing, just when any information is given by the sound. For that reason, it is said that it is a passive method.

2.2.2.3. Bilateral Devices

Bilateral devices are those one that have the control device and the feedback device in a single element. These mechanisms are very effective because the operators feel that they are in the same place than the robot and for that reason the sensing is much better.

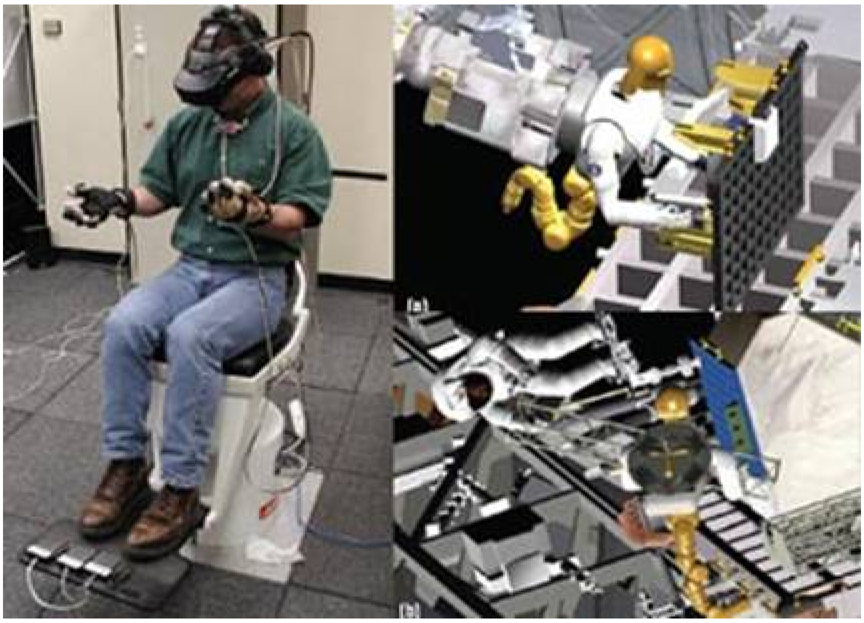


Figure 7. Bilateral Device Example

2.3. Kinect

Kinect was created by Microsoft in 2010 with the objective of been a new type of controller for the video game console Xbox 360 [18]. Alex Kipman created it for replacing the traditional controllers (pad, stick, etc.) by a new controller that can be used with the body movements or using the voice.

Using Kinect Microsoft makes easier the communication between the video game console and the users. Using the body is more intuitive that using a traditional controller and makes easier that people who are not related with the videogames want to play or interact with the Xbox 360 because the learning curve is smaller using Kinect.

This type of interface is known as NUI (Natural User Interface), what means that it is a way to interact directly with the device without using any other intermediate device.

Figure 8. Couple Playing with Kinect

Kinect was announced in the E3 (Electronic Entertainment Expo) in 2009 with the codename Project Natal, like a future revolutionary change in the videogames concept.

In 2010 was finally presented in the following E3 with the name of Kinect. The launch of the Kinect was in November of the same year in whole world with just some days of difference between regions. In February 2013, less than 3 years later, Kinect had sold more than 24 million of devices [19].

The same day that the Kinect was launched, Adafruit Industries offered a bounty for an open-source driver for Kinect. Microsoft voiced its disapproval what produces that Adafruit Industries increase the bounty from 1,000 dollars to 3,000 dollars [20].
Only few hours after its launch, Kinect was hacked by a Spanish named Hector Martin [21] [22].

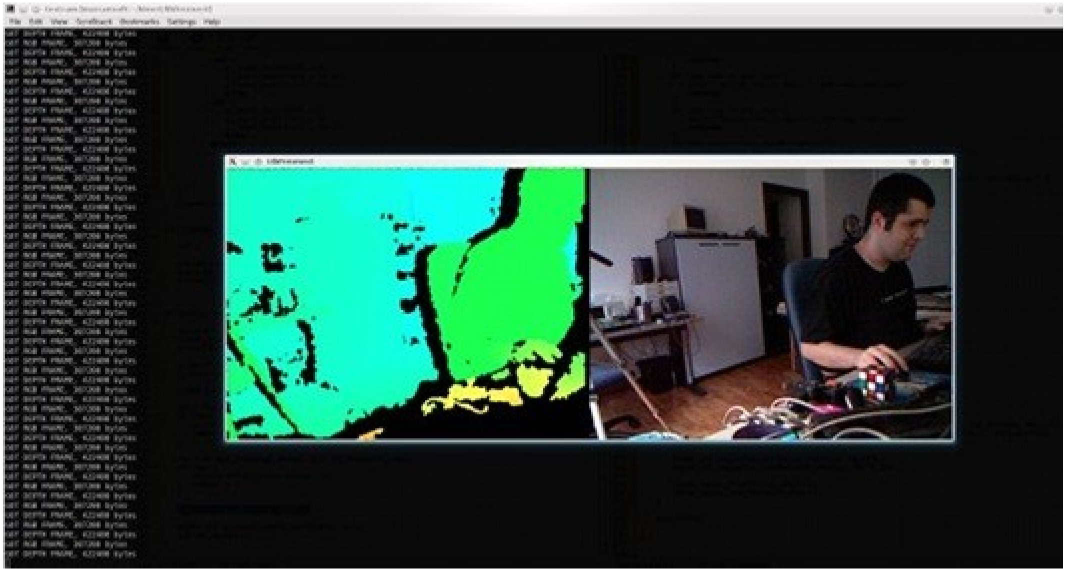


Figure 9. Hector Martin Hacked Kinect

When Microsoft saw the interest from the hackers and programmers in Kinect decided to publish the official SDK [23].

Figure 10. Kinect SDK

The SDK was launched on February, 2012.

The technology used in Kinect is based in software created by Rare, a company owned by Microsoft and in a camera that is able to measure the depth. This camera was created by the Israeli company PrimeSense [24] that is specialized in Natural User Interfaces development.

Kinect is formed by three different sensors that works together to obtain better results. The sensors are the following:

- **RGB camera:** It is a VGA camera with a video resolution of 640x480 pixels. It was fabricated with a CMOS sensor and with the capacity of record 30 fps. The camera objective is to capture the video and helping with the facial recognition. The first picture in the Figure 11 is captured by this camera.

- **IR depth-finding camera:** The IR depth-finding camera is formed by an IR emitter, a monochrome CMOS sensor and the module that creates the cloud points in three dimensions.
  This sensor works in a similar way than the ship sonar: the emitter sends a matrix formed by IR light points invisible to the human eye (second picture from the Figure 11), each of these points collides with a user or an object in the space and returns to the Kinect sensor and they are received by the CMOS sensor. The IR light loses intensity when is traveling, for that reason the Kinect can calculate the distance where is any object depending of the intensity that the IR light has when returns (the third picture from the Figure 11, a 3D image).
  A big advantage that this sensor has is that it works perfectly even without light [25] [26] [27].

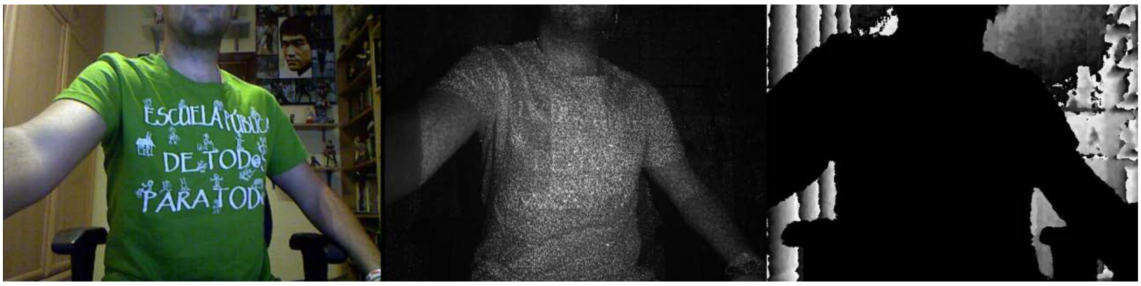


Figure 11. Kinect Sensors

- **Multi-array microphone:** A multi-array microphone is a group of microphones situated in line that receives the sound independently. Using that method, Kinect can calculate where the sound comes from. These microphones are situated in the bottom of the device, three in the left side and one in the right side.

  Depending the use Kinect applies some filters to eliminate the sound that is not in the human voice frequency (80-1100 Hz) and eliminate the sound that does not come from the front of the device that is where the user should be located [28].

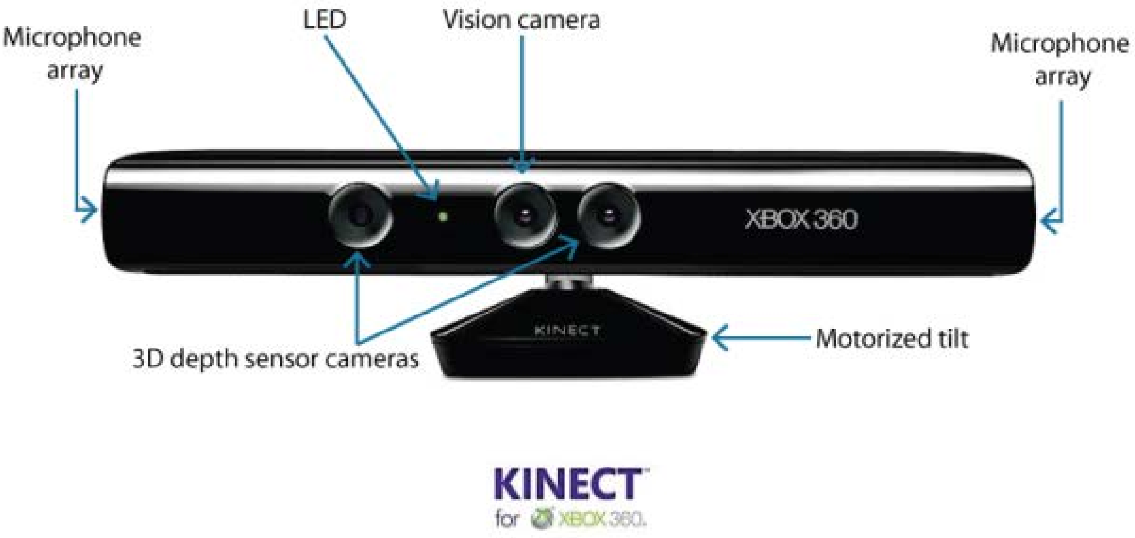


Figure 12. Kinect Components

Kinect has a vision angle of 57 degrees horizontal and 43 degrees vertical. It also has a motorized tilt that allows it to move 27 degrees up and 27 degrees down.

Chapter 3

System design and implementation

In this section it is explained the functionality of the system and the development process that has been chosen for its implementation.

3.1. Methodology

The methodology choice is very important for the correct evolution and final result of a project. Seeing the type of project the best decision is to choose a flexible methodology that allows making changes with a small impact.

The main idea is to create different versions that add new functionalities and improve the ones that already have been developed. The motive to use that kind of methodology is because the person who is going to develop the system does not have enough knowledge about this type of systems. A previous study of every element that will form the system exists but there is not real experience so it's important to take into account that it is almost impossible to obtain the optimal solution in the first attempt. For that reason it is very important to use a methodology as it was mentioned in the lines above.

The consecutive versions will give to the developer experience and knowledge to improve the previous versions and it will probably happen until the end of the development. The final result using this type of methodology will be a complete and robust system.

For all those reasons the methodology chosen is an incremental methodology with two phases:

- Initialization step

In this stage, a base version of the system it is created. It is necessary that this base version has the essential functionalities with a simple but effective solution for a correct understanding and implementation.

With that base version the system can feedback to itself and evolve correctly.

- Iteration Step

In this stage, a new task is redesigned and implemented from the control list from the project. Later, the newest version is analyzed and improved if it is possible.

This methodology has some disadvantages but, in this case, with time and a small team (just 1 person) it is the best option.

3.2. Analysis

In this section it is explained the analysis process used. The analysis objective is to specify in a deep way what the system has to do and how. Later, in the design section, the analysis will be used to make the design that permits and accurate implementation.

3.2.1. Scope of the project

In this section the system will be deeply detailed, specifically the functionalities the system will have and the functionalities that it will not have.

The project objective is to create a system that makes possible to control a robot NAO using a user natural interface from a teleoperation base that can be far away from the robot. To create this interface there will be used the following components:

- Microsoft Kinect Sensor
- Computer
- NAO Robot

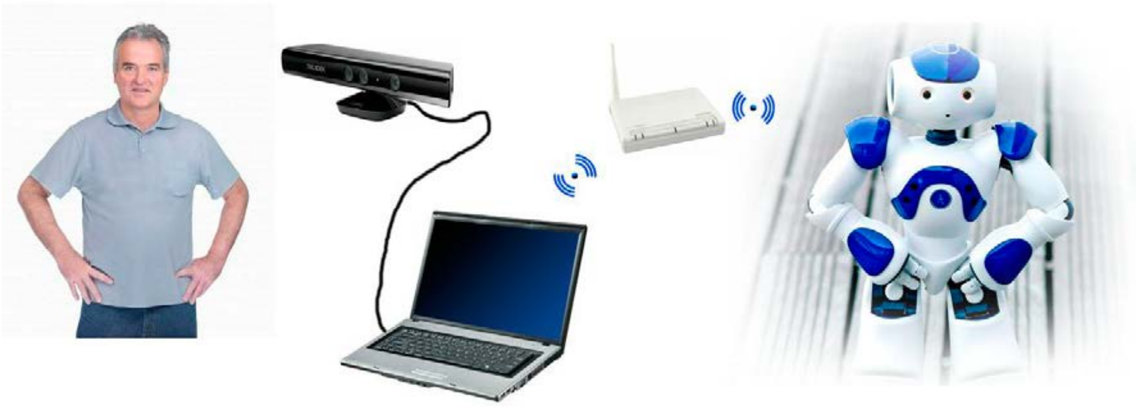


Figure 13. System connections

The user will control the NAO Robot arms moving its own arms. The NAO Robot will imitate the same movements that the user does.

The user will be able to see a window with what the Kinect is capturing but simplified. It means that this screen will show any person detected in front of the Kinect sensor using a green skeleton per person. For a correct work, it should be just one person in front of the camera because the teleoperation is oriented to a single user.

There will be information in the screen about the connection with the NAO Robot, the state of the Kinect sensor and the captured video.

The Nao Robot will not imitate every move done by the user, the moves from the legs and head of the user will not be done by the robot. It is in that way because the teleoperation system has the main objective to repeat the arms and forearms moves. For that reason it will be problematic if the robot move the legs while the user is doing some work. The NAO robot will move itself losing the accuracy or even fall to the ground. The head neither moves because in spite of it was studied the NAO robot head offers few grades of freedom in the head so it would not be useful enough, at least for this project.

3.2.2. Use Cases

In this section, the use cases are shown and each one is detailed using the table that is presented below.

3.2.2.1. Use Cases Description

To make easier the summary of the information, the following table is used for each use case.

Name	
Description	
Preconditions	
Post-conditions	
Scenario	

Table 1. Use Case Table Template

The fields are explaining below.

- **Name**: A unique name that can give a small idea of the use case.
- **Description**: A brief and concise explanation about the use case and its situation.
- **Preconditions**: The conditions that must exist before to do a given operation in the use case.
- **Post-conditions**: The state in which the system will be after the execution of a given operation in the use case.
- **Scenario**: The execution step by step following the correct order.

3.2.2.2. Use Cases List

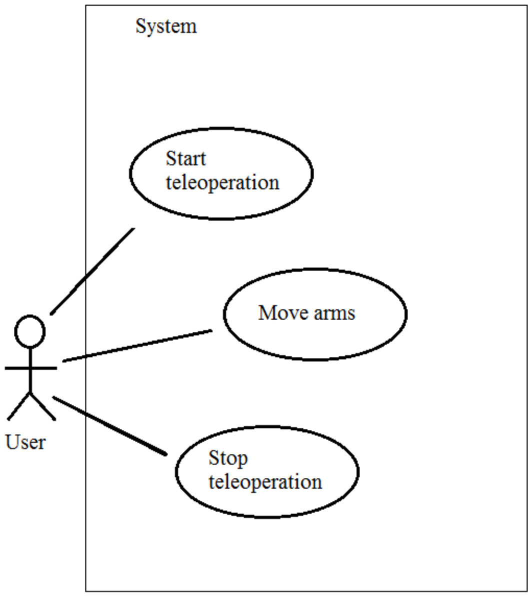


Figure 14. Use Cases List

Name	Start teleoperation
Description	The system searches for a user to start the teleoperation.
Preconditions	• Teleoperation stopped • Kinect connected
Post-conditions	• User detected
Scenario	1. The system checks if the devices work correctly. 2. Kinect starts to search for a valid user. 3. Kinect detects the user. 4. The system shows the user skeleton on the screen.

Table 2. Start Teleoperation Use Case

Name	Move arms
Description	The NAO robot repeats the arms' moves from the user.
Preconditions	• Teleoperation started.
Post-conditions	• The robot repeats the arms' moves from the user
Scenario	1. The user moves the arms. 2. Kinect recognizes the angles needed. 3. The system calculates the angles for the robot. 4. The robot moves the arms in the same way than the user.

Table 3. Move Arms Use Case

Name	Stop teleoperation
Description	The system stops.
Preconditions	• Teleoperation started
Post-conditions	• Nao stopped • Kinect stopped • Teleoperation stopped
Scenario	1. The user clicks on the "X" of the screen. 2. The system stops.

Table 4. Stop Teleoperation Use Case

3.2.3. Software Requirements

The software requirements catalogue showed in this section is obtained from the analysis of the description, scope of the project, use cases and a study of all the needs from this type of applications.

3.2.3.1. Software Requirements Description

To make easier the summary of the information, the following table is used for each software requirement.

Name			
Description			
Priority		**Necessity**	
Stability		**Verifiability**	

Table 5. Software Requirement Table Template

- **Name**: A unique name that can give a brief idea of the requirement.
- **Description**: A brief and concise explanation about the software requirement and its situation.
- **Priority**: Precedence of a requirement compared with other.
- **Necessity**: In which grade it is need that the requirement is accomplished.
- **Stability**: In which grade it is going to change or not.

- **Verifiability**: In which grade it is easy to verify if the requirement is accomplished.

3.2.3.2. Software Requirements Catalogue

It is important to differentiate between the different types of requirements. In this project there the functional requirements and the inverse requirements are differentiated. The functional requirements are the one which tell the user what have to do the system, the inverse requirements tell the user what do not have to do the system. There is a third type of requirement in this project, the non-functional requirements, that details in which way it has to achieve the objectives wanted.

In the table below, the requirements are separated like it was said above. The functional requirements are labeled like FR, the inverse requirements like IR and the non-functional requirements like NFR. All the requirements also have a number that allows the user to differentiate them in a faster way than using the description.

Requirement	Description
FR-1	Capture position
FR-2	Start teleoperation
FR-3	Stop teleoperation
FR-4	Move arms
FR-5	Move forearms
FR-6	Show skeleton
FR-7	Notify outbound joints
FR-8	Show information
IR-1	Move legs
IR-2	Move head
IR-3	Spin
NFR-1	Use the NAO robot
NFR-2	Use Kinect

NFR-3	Use Windows
NFR-4	Use NAOqi
NFR-5	Delay
NFR-6	Divided
NFR-7	Easy learning

Table 6. Requriements Summary Table

3.3. Design

The design section objective is to present the system's architecture and all the technologic elements that will make possible to the system work. It also divides and details each component of the system.

In the analysis section it was described what the system has to do. In the following point, it is explained how to develop the system to solve all the requirements mentioned in the analysis.

3.3.1. Design Alternatives

In this section, some alternatives are shown and finally it will be selected the one that better fits with the project objective.

The use of the NAO robot is a requirement of the project and for that reason all the alternatives are focused in the operating system and in the camera device.

For a better understanding, some tables will be used depending of the different wanted attributes in each one. There will be marks between 0 and 10, where 0 is the worst mark and 10 the best mark.

- Camera Sensor: There are two main sensors studied, the Microsoft Kinect and the Asus XTION PRO LIVE.

	Microsoft Kinect	Asus XTION PRO LIVE
Price	10 (45€)	6 (132€)
Documentation	10	6
Libraries	10	8
Quality results	8	10
Final Mark	**38/40**	**30/40**

Table 7. Camera Sensors Comparison

The Microsoft Kinect camera is the chosen one.

- Operating System: There are two operating systems compared, Windows 7 and Ubuntu 14.04 LTS.

	Windows 7	Ubuntu 14.04
Price	0	10
Official libraries	10	0
Ease of use	10	8
Final Mark	**20/30**	**18/30**

Table 8. Operating System Comparison

Microsoft Windows 7 is the operating system chosen.

- Libraries: The libraries studied are the official from the Microsoft Kinect SDK and the open source OpenNI libraries.

	Microsoft Kinect SDK	OpenNI
Price	0 (it depends)	10 (0€)
Reliability	10	8
Documentation	10	8
Support	10	0
Final Mark	**30/40**	**26/40**

Table 9. Libraries Comparison

Microsoft Kinect SDK is selected for the project

After the comparison, the project will be formed by NAO Robot, Microsoft Kinect, a computer with Windows 7 and the libraries will be the official from the Microsoft Kinect SDK.

3.3.2. Architecture

In this section the architecture of the system is explained and divided in smaller systems (subsystems). Specifically, the architecture is divided in three subsystems that are explained in the following lines:

- Robot subsystem: The main purpose of the robot subsystem is to communicate with the robot giving the orders wanted.
- Vision Subsystem: The main purpose of the vision subsystem is to obtain the images from the Kinect device and show it on the screen.

  It also has the objective of centralize the messages and data to a better organization.
- Processing Subsystem: The main purpose of the processing subsystem is to receive the information about the user position and return the data transformed in the angles to the NAO robot.

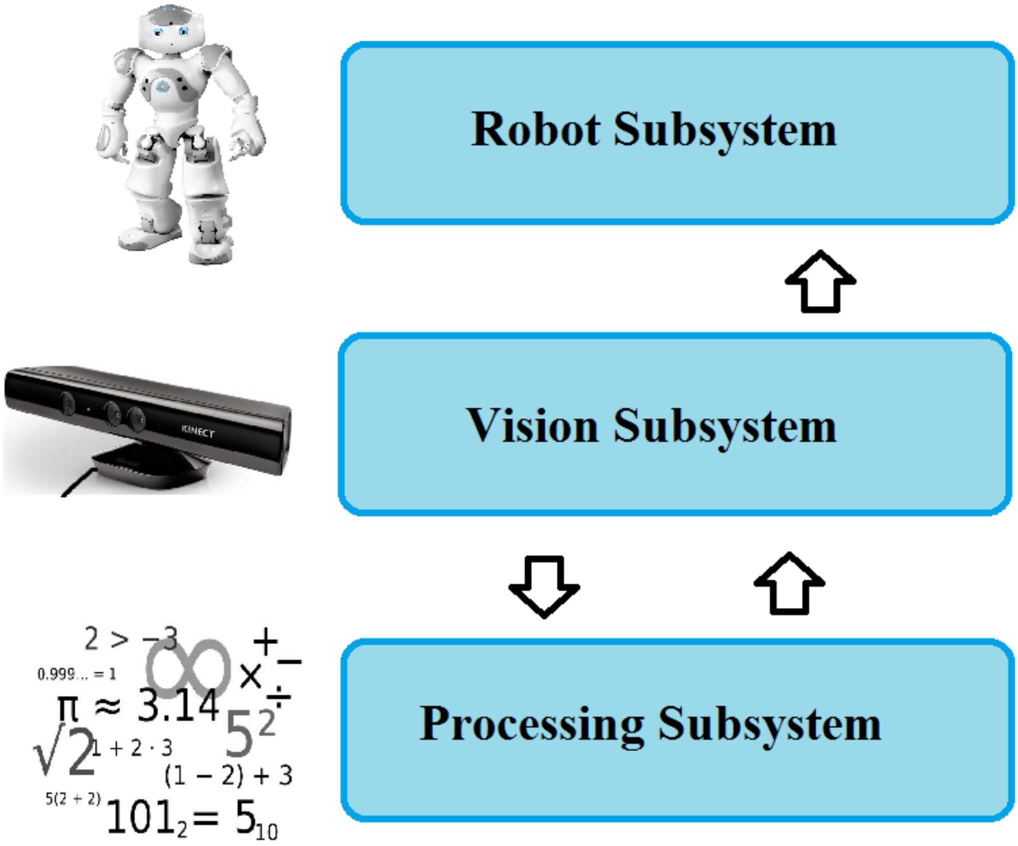


Figure 15. System Architecture

The main reason to choose that architecture is the possibility of re-using the different subsystems in other projects or the easy of expanding this project in the future.

There was the possibility to divide the vision subsystem in two different, one for capturing the video and other for been the main subsystem, but the increment of delays make it a not good idea and finally the best option is the one used.

Omitting the main part, each subsystem can work like independent systems. It gives to the developer the option to change subsystem in an independent way and can add functionalities without implicating the rest of the subsystem.

In the following picture it can be seen how the system works and its structure in a more graphical way to make it easier to understand.

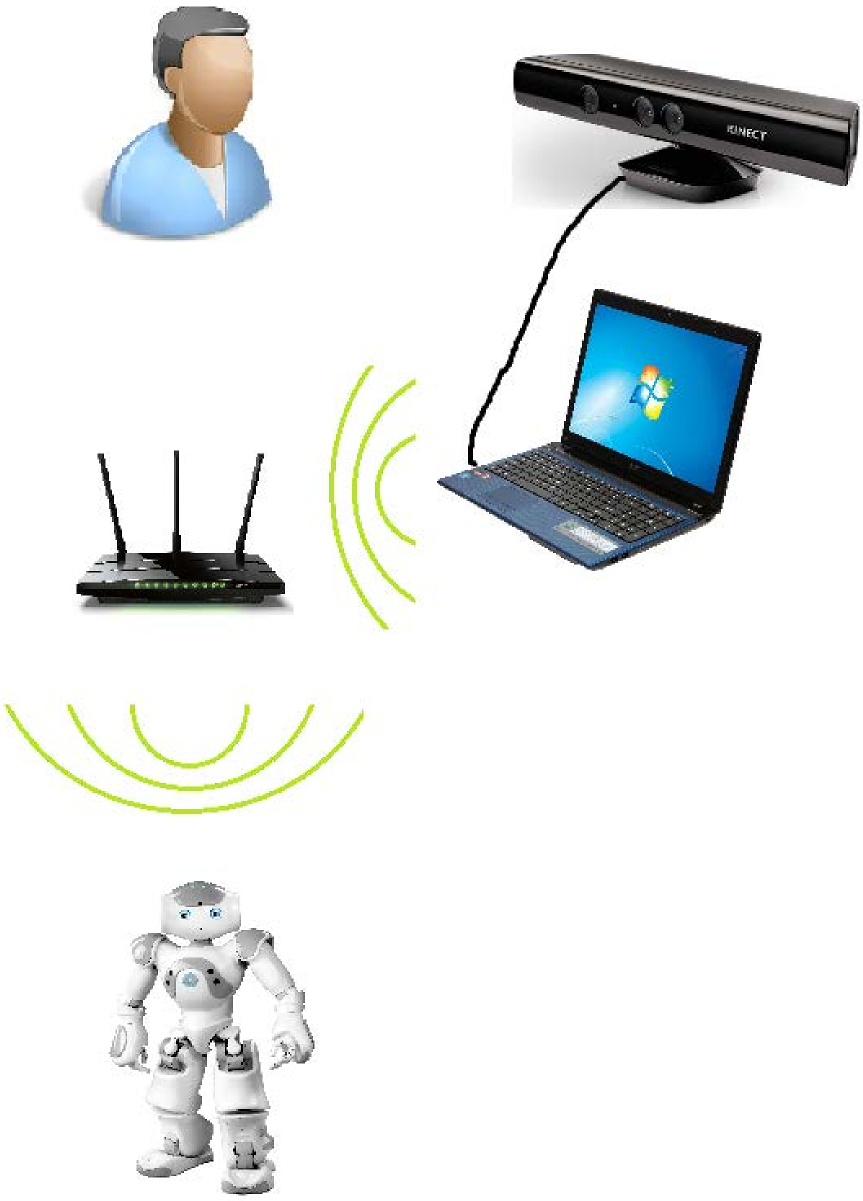


Figure 16. System Structure

In Figure 16, the system is shown. The Kinect sensor captures the image that is sent to the computer. In the computer, the data is processed and send by wi-fi to the robot NAO.

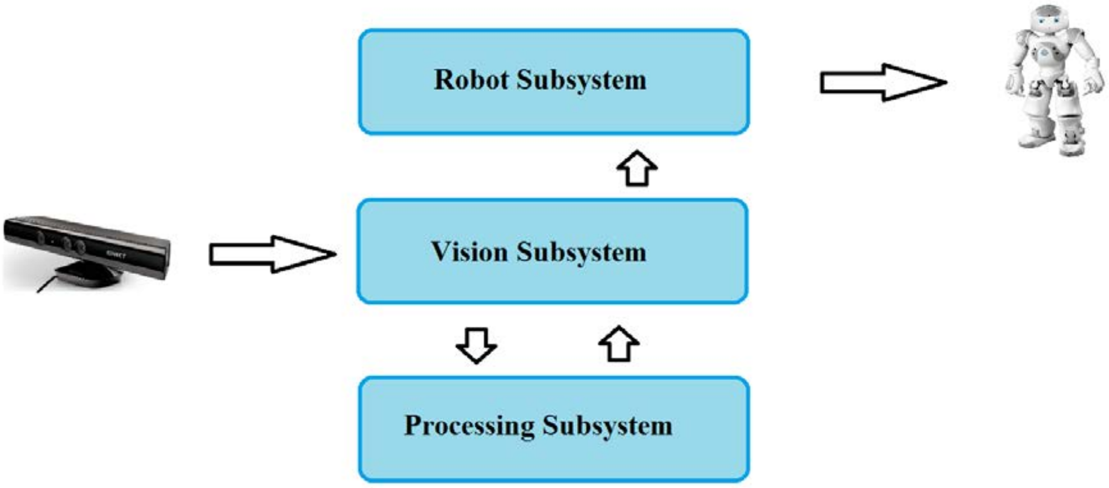


Figure 17. Data Flow in System Architecture

3.3.3. System Description

In this section, the functionalities of the system are briefly described, as well as which classes and components are working in each one.

The steps that the system follows are, simplifying, the following:

1) Capture the user position.
2) Calculate the angles for each joint.
3) Map the angles to values that the NAO Robot can use.
4) Communicate with the NAO Robot.
5) Show information to the user.

1. Capture the user position.

In this step the Kinect sensor captures the video data and sends it to the computer. In the computer the MainWindow class receives the data and stores the skeleton (if found) in an array with the position of all the joints.

2. Calculate the angles for each joint.

The MainWindow class sends the coordinates of the joints in which the system is interested (it will be explained in the Subsystem Description section) to the Angles class specifying if it is an angle with vertex shared or not (it will be explained, too).

The Angles class receives the coordinates and depending of the type of angle wanted will calculate the angle.
Then, the angle value is returned.

3. **Converts the angles to values that the NAO Robot can use.**

    When the MainWindow has all the angles wanted it has to send them to the Nao class but before doing that, it converts the angles to the NAO angles. That is like this because the NAO angles do not use the same reference system than the Kinect sensor.

    When the angles are converted, they are sent to the Nao class.

4. **Communicates with the NAO Robot.**

    The Nao class receives the data (the angles) and sends them to the NAO Robot.

5. **Shows information to the user.**

    The MainWindow class is the class that informs the user about everything.
    First of all, it draws the joints in the screen and then joins them when it is necessary.

    The Kinect state is given also by this class.

    Finally, if any joint is out of bound the screen is painted in that side with a red color to notify to the user that it's necessary to be in front of the Kinect sensor with all the joints in the angle vision.

3.3.4. Subsystem Description

In this section, it is detailed a description of the different subsystems that were listed in Section '3.3.1 Architecture' that form the whole system.

Each subsystem is also divided in the components that form it and the code will be explained deeply. The hardware will not be explained because it already was in the 'State of the Art' section.

2.3.3.1. Vision Subsystem

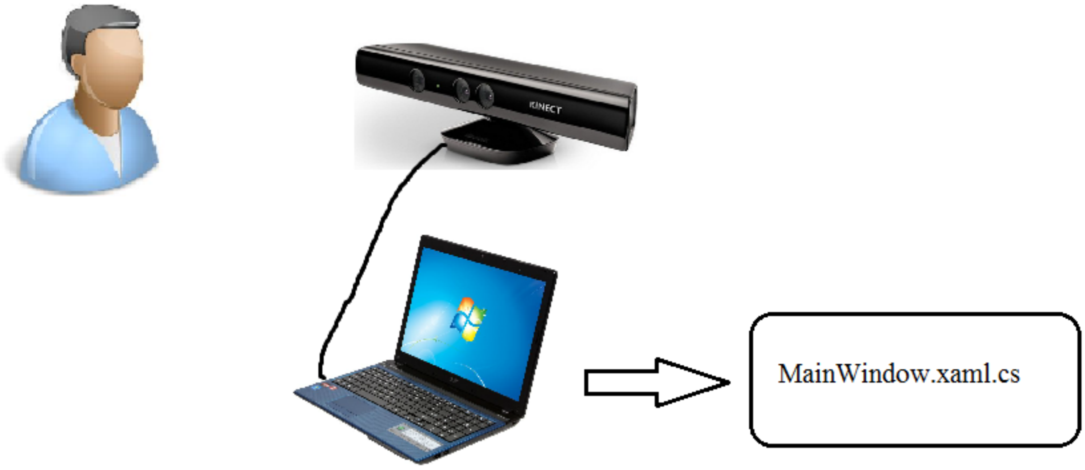


Figure 18. Vision Subsystem

This subsystem is formed by the Kinect device and the MainWindow class. The main purpose of this subsystem is to receive the data from the Kinect sensor and the communication between the subsystems.

When the sensor captures data, it searches for a skeleton. If finds it, the code stores the skeleton in a skeleton data type and then stores the joints coordinates in an array. With that array, the class does two things:

First, it draws the skeleton. For that, it shows a black screen and draws the joints in that screen, and then it joins the different joints depending of the skeleton. For example, it will draw a line between the wrist and the elbow but not between the neck and the knee. An example can be seen in the next page.

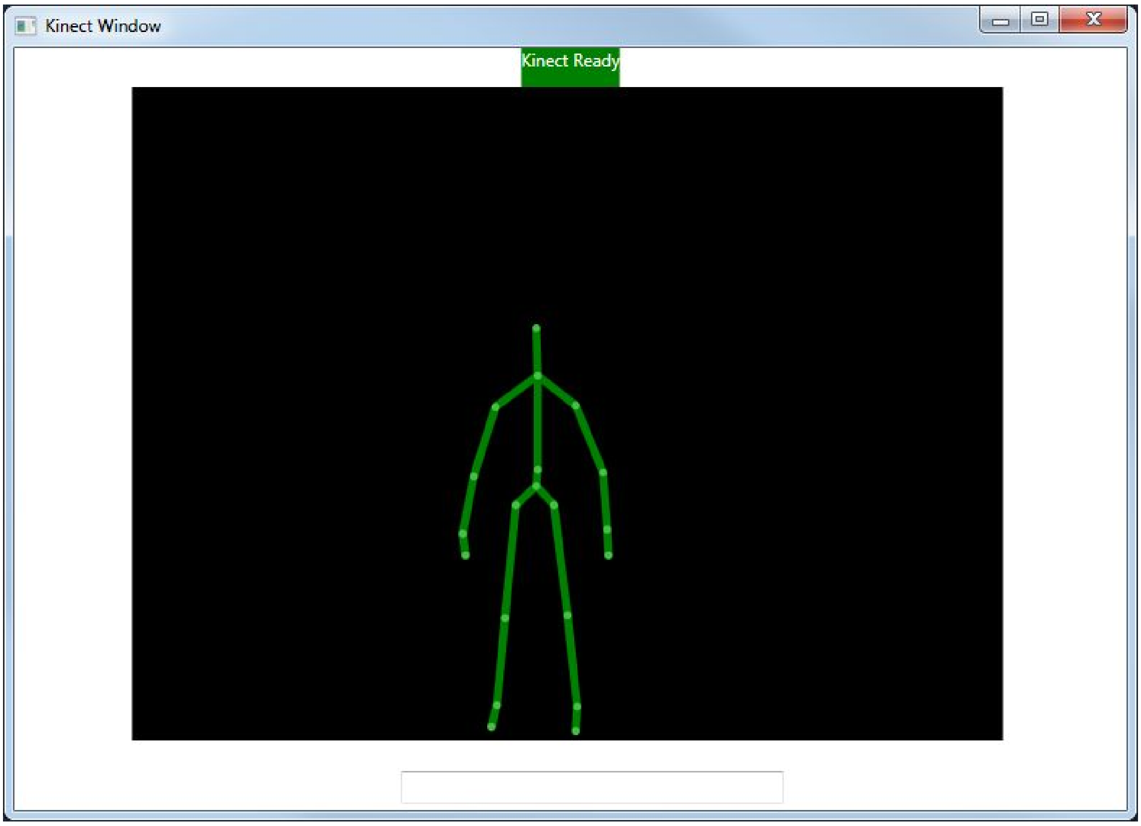


Figure 19. Skeleten drawed

Second, it uses the joints to send them to the Angles class to calculate the different angles like it will be explained in the following pages.

Before sending the different coordinates to the methods from the Angles class, it is necessary to know how to calculate the angles, which angles are going to be used and later, convert the angles from Kinect to the angles in the NAO robot.

The angle calculation is complex. For that reason, it will be detailed in the following pages to make easier the understanding.

Angles calculation

First of all, the Vision Subsystem processes the skeleton and obtains the different joints that the system is going to use. Having that, the Vision Subsystem calls to the Processing Subsystem to obtain the angles giving the different joints that are going to be explained right now.

- Shoulder Pitch

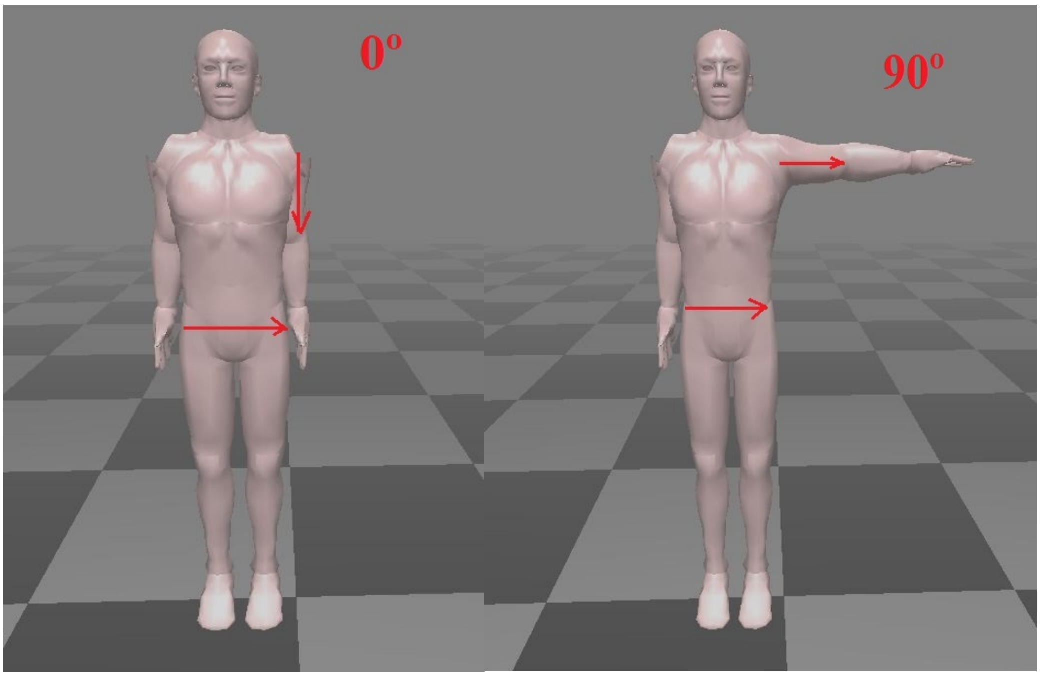


Figure 20. Shoulder Pitch

The joints used to calculate this angle are the both hips, the shoulder from the wanted side and the elbow from this side.

As it can be seen in Figure 20, the vectors are the unions from shoulder to elbow and from the opposite hip to the near hip. Calculating the angle between those vectors can be obtained the pitch angle from that shoulder.

In this case, to make it more intuitive, the angle has been modified in the calculation to obtain 0 degrees when the arm is perpendicular to the hip union and 90 when it is parallel to the hip union.

It is important to clarify that there exist more ways to calculate that angle, but this one gives the best results. At first sight, the first idea is to use the vector that join the shoulder with the elbow and the vector that joins the shoulder whit the hip. But in that case, the angle will also change if the shoulder is moved in a roll movement, which means to the front or to the back. So it is impossible to use it and obtain good results because the robot will move the arms in other ways than the human.

- Shoulder Roll

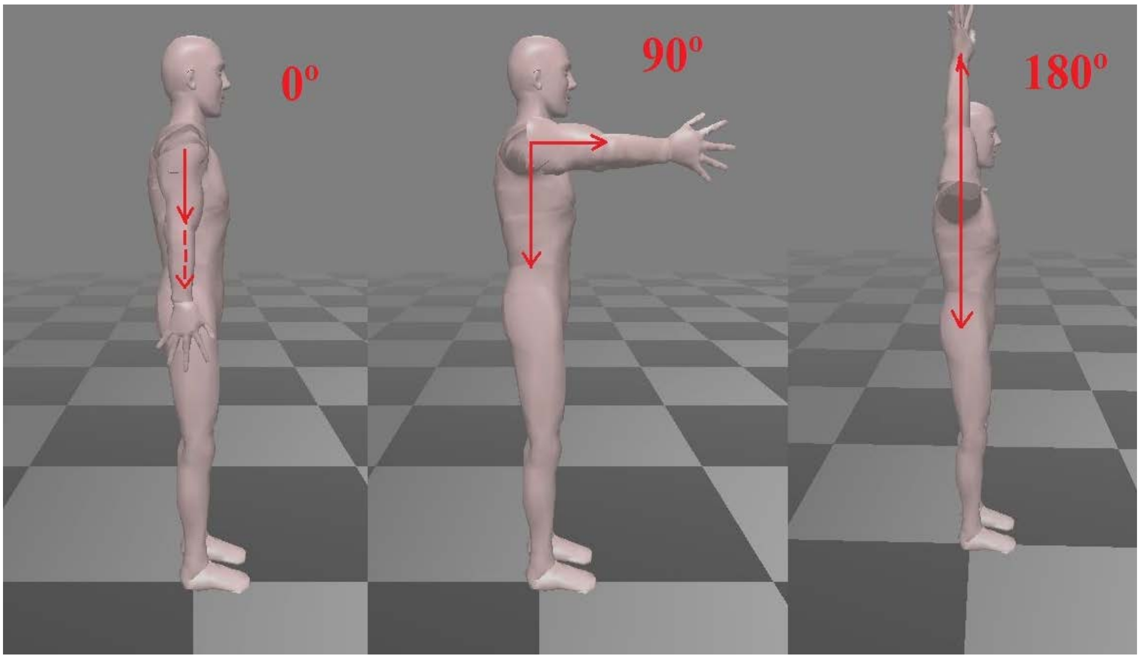


Figure 21. Shoulder Roll

The vectors used in this case are the vector that joins the shoulder with the elbow and the vector that joins the shoulder with the hip. In this case, the move also changes the angle from the rotation but it is too small to make it a problem.

In this case, the angle will be 0 degrees if the arm is pointing to the ground, 90 degrees if the arm is perpendicular to the shoulder-hip vector and 180 degrees if the arm is pointing to the sky.

- Elbow Roll

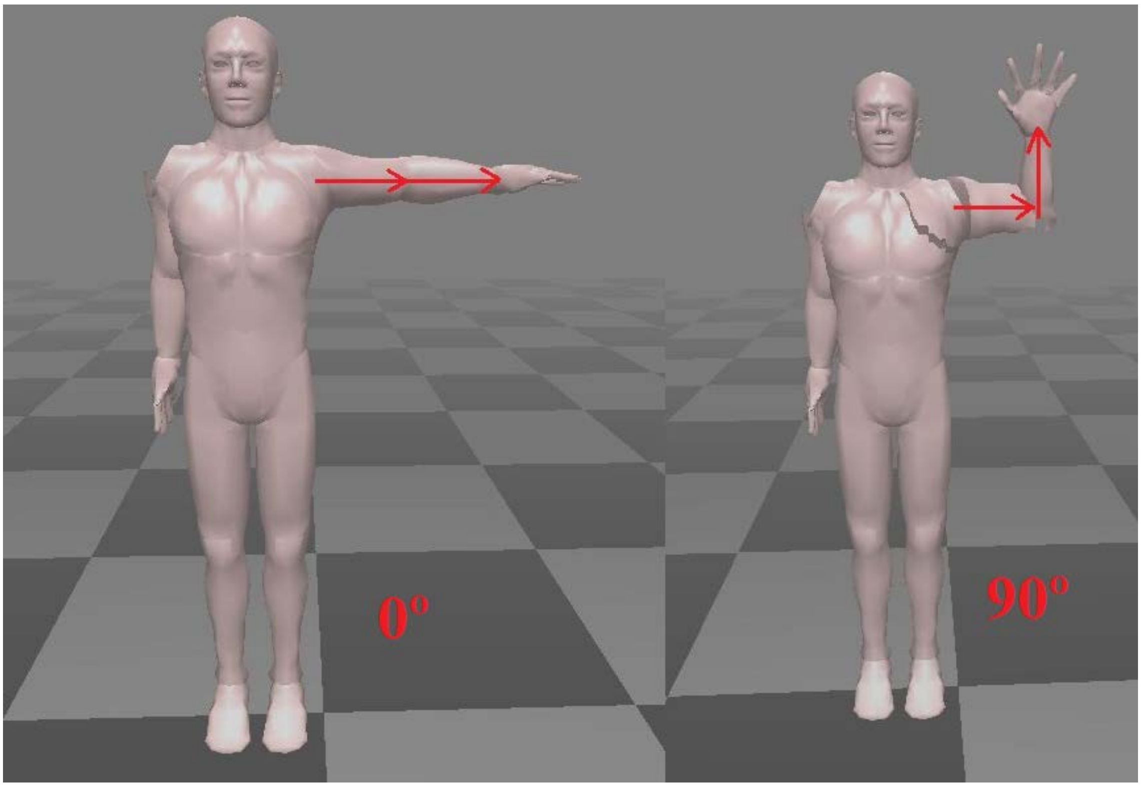


Figure 22. Elbow Roll

To calculate this angle, the joints used are the shoulder, the elbow and the wrist from the side wanted. The vectors used are the vector that joins the shoulder with the elbow and the vector that joins the elbow with the wrist.

The angle wanted is the one that is formed between both vectors. If the forearm is pointing to the side, the angle is 0 degrees. If the forearm is perpendicular to the arm, the angle is 90 degrees.

- Elbow Yaw

The angle in the elbow yaw is probably the most difficult. The reason for this is that the elbow position changes when the shoulder is moved so the vectors also change. The better way to avoid those errors is using different vectors depending of the elbow position in any moment.

When the arm is near to the body, the angles shown in Figure 23 are used.

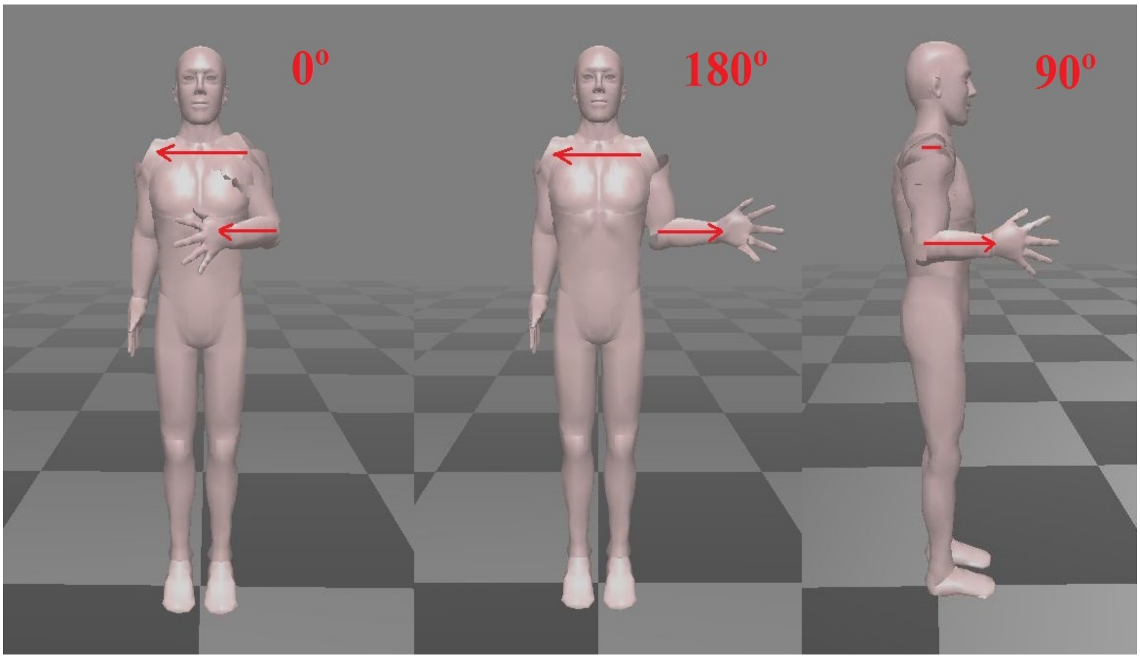


Figure 23. Elbow Yaw with the arm near the body

The joints used are the two shoulders, the elbow from the side wanted and the wrist from the same side. The vectors are the union between the two shoulders with direction of the opposite shoulder and the other vector is from the elbow to the wrist.

With these vectors, the angles will be 0 degrees if the forearm is pointing to the body side, 180 degrees if the forearm is pointing to the other side and 90 degrees if the forearm is pointing in front of the body.

All these angles are only for the case in which the arm is completely close to the user body.

If the arm is completely perpendicular to the ground the way to calculate the angle is shown in Figure 24.

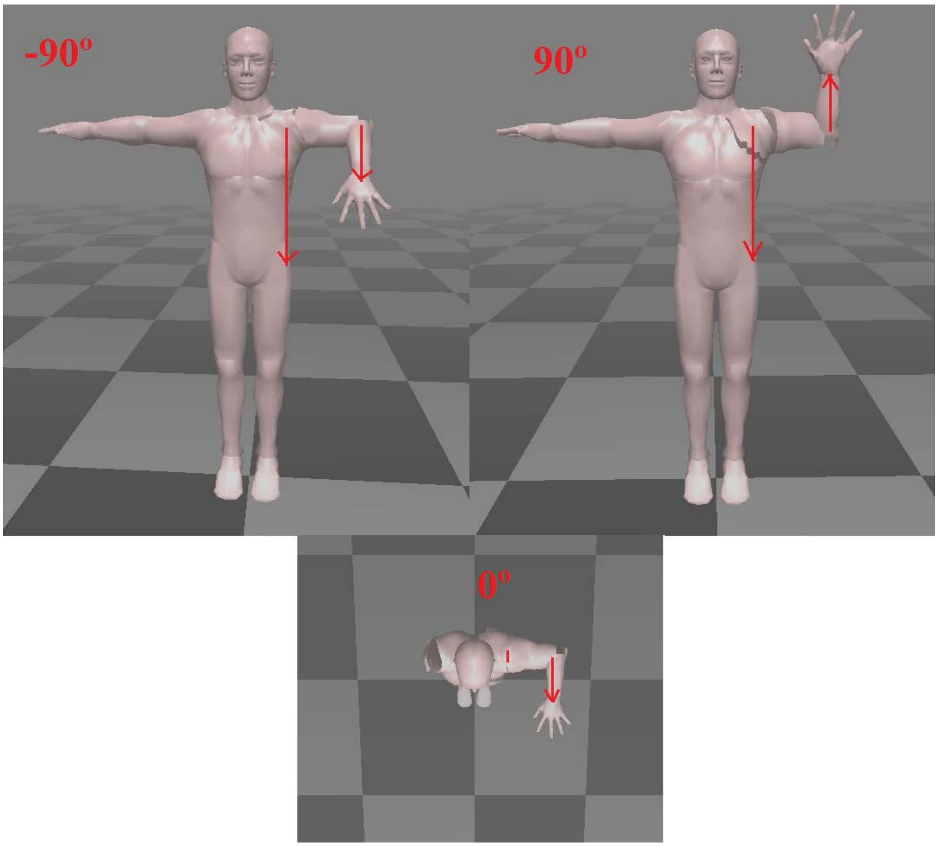


Figure 24. Elbow Yaw with the arm perpendicular to the body

In this case the joints used are all from the same side, the side where the elbow is. The joints are the shoulder, the hip, the elbow and the wrist. With that joints, two vectors are created, one vector from the shoulder to the hip, the other one from the elbow to the wrist.

If the forearm is pointing to the floor, the angle will be -90 degrees. If the forearm is pointing to the front it will be a 0 degrees angle and if the forearm is pointing to the sky it will be a 90 degrees angle.

The problem is that the arm can be in any angle between the perpendicular and the parallel of the body so it is necessary a way to know which angle use in the intermediary cases and of course, it is essential to find a way to change between one equation and the other one without a sharp change.

The best way to solve the two problems exposed is the creation of an equation that changes the angles equation depending of the coordinates of the elbow. The equation is the following:

$$EY = SH \times \frac{SR}{90^\circ} + SS \times (1 - \frac{SR}{90^\circ})$$

Where:

- EY: is the elbow yaw.
- SH: is the angle of the elbow yaw using the vector that goes from the shoulder to the hip.
- SS: is the angle of the elbow way using the vector that goes from one shoulder to the other.
- SR: is the angle of the shoulder roll.

Using that equation, the angle calculation will be continuous without any sharp change between the values. In addition, when the forearm is perpendicular it will be used only one equation and when it is parallel to the body the other one. It makes this equation perfect to obtain the better results.

An example with real numbers can be seen in Figure 25.

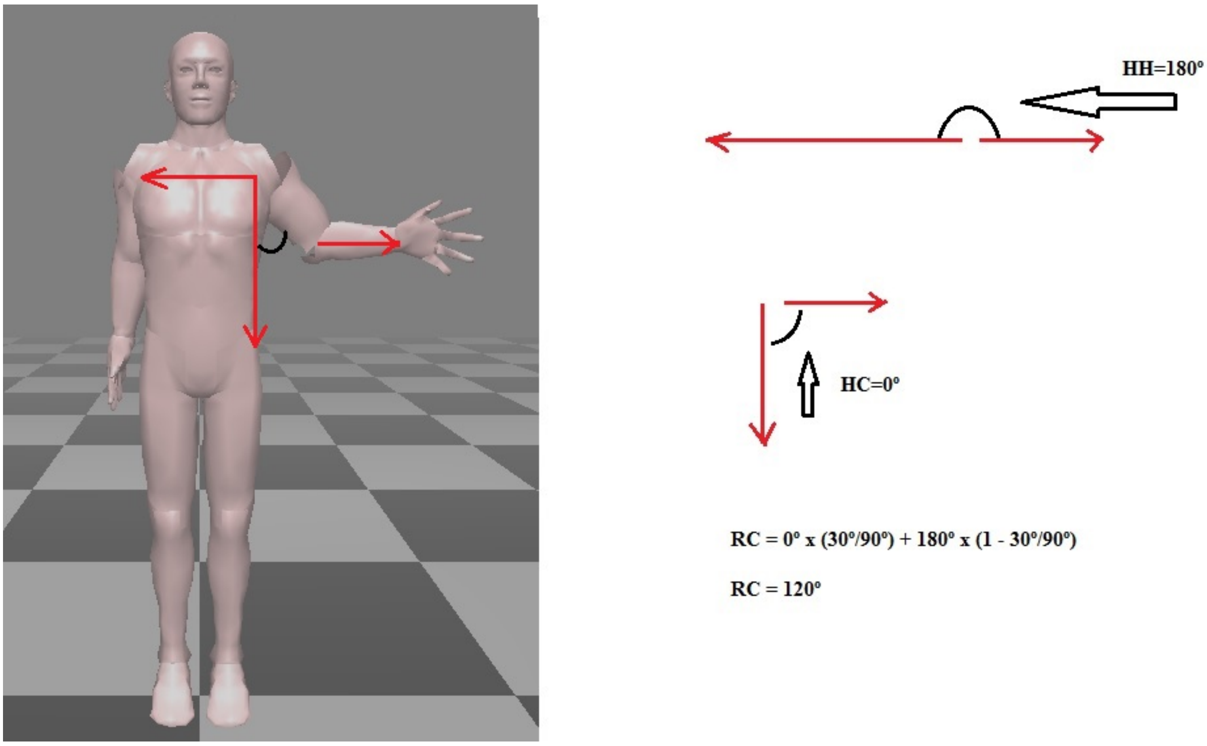


Figure 25. Elbow Yaw Example

2.3.3.2. Processing Subsystem

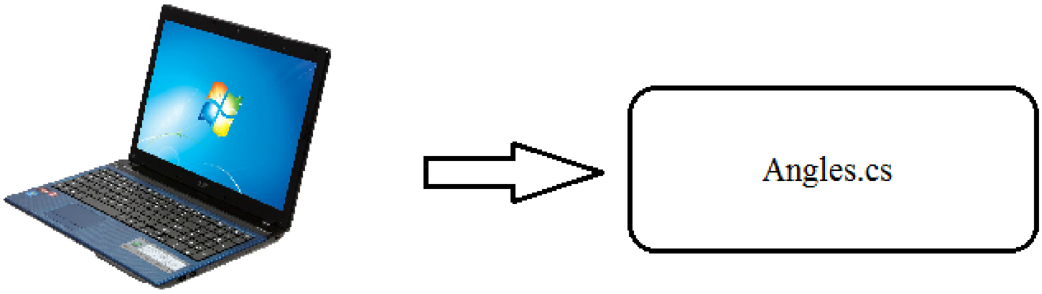


Figure 26. Processing Subsystem

The processing subsystem is formed by the Angle class and its purpose is to calculate the angles between the vectors formed by different coordinates. The class has two methods, one for each possible type of angle in the human body.

The calculation of the angles are done using trigonometry in the following way:

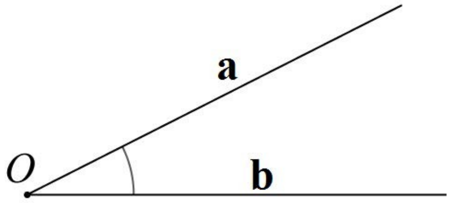


Figure 27. Angle

$$a \cdot b = a_x b_x + a_y b_y + a_z b_z$$

$$|a| = \sqrt{a_x^2 + a_y^2 + a_z^2}$$

$$|b| = \sqrt{b_x^2 + b_y^2 + b_z^2}$$

$$O = \cos^{-1} \frac{a \cdot b}{|a| \cdot |b|}$$

As it was seen in the last section, there are two types of request angles in the project. The first one is with two vectors that share one point which will be the vertex. The other one is with two different vectors that do not share any point.

2.3.3.3. Robot Subsystem

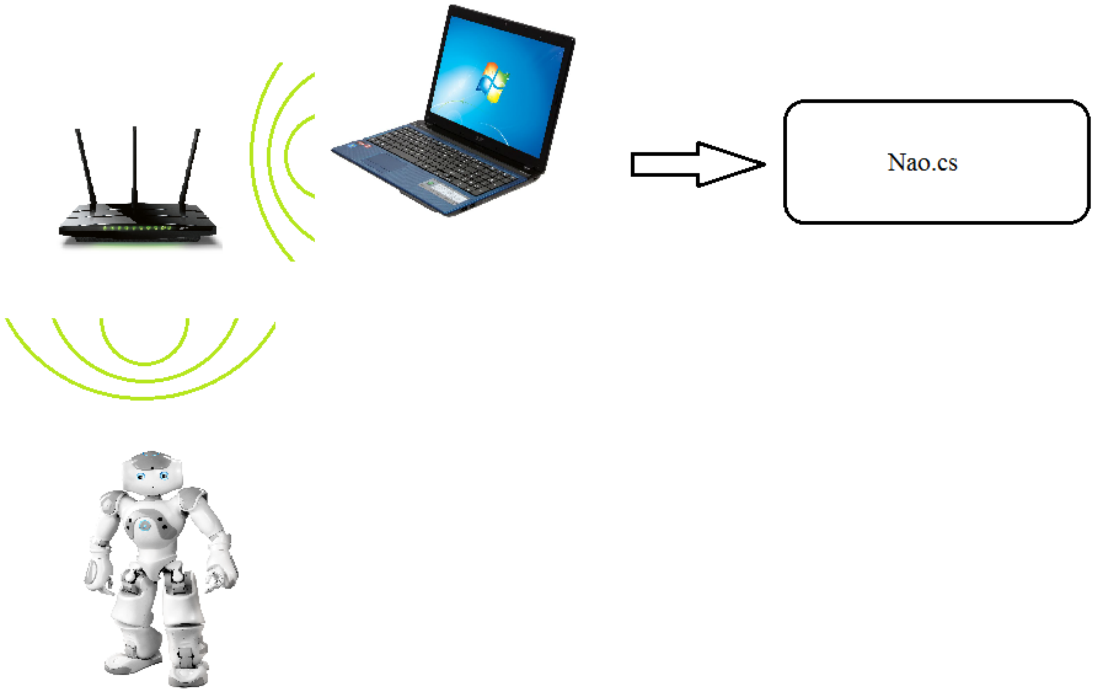


Figure 28. Robot Subsystem

The robot subsystem is formed by the Nao robot, the class Nao and a device that can be used to communicate using Wi-Fi between the computer and the Nao robot.

The purpose of this subsystem is to receive the angles from the Video Subsystem already converted to the Nao angles and then, send them to the Nao robot specifying the angle number, the joint to move and with which speed the Nao robot should move the joint.

Chapter 4

Evaluation

In this section, it will be detailed how the project evaluation is done. The evaluation objective is to verify that the implemented system covers the requirements and the initial needs. The most important goals from the process evaluation in a project are the following:

- Every single component works correctly.
- The connection between the different components works perfectly.
- The communication between the different classes and elements of the whole system works appropriately.
- The user can do every activity that he requested at the beginning of the project.

4.1. Methodology used in the evaluation

The evaluation part is basic to get a product with a high quality. In the evaluation stage, the errors are depurated so the product turns to a most robust and reliability system.

The evaluation can be divided in two different sections:

- Test section: Different tests are done by the developer.
- Experiment section: External people to the project use the system and make different experiments. Later they will fill some surveys.

4.2. Tests

In this section, the tests will be detailed and explained. The results obtained in each test will be also explained.

The test evaluation stage can be divided in three steps:

- Unitary test.
- Communication functionalities test/Integration testing.
- Requirements test.

The requirements can be checked in the 3.2.3 section.

The unitary test verifies that every different component from the system works correctly. It can be checked giving to it a correct input and verifying the output of each one.

The integration testing has a double purpose:

- Verify the correct integration of every element during the unitary tests (internal).
- Verify the correct integration between the different elements (external).

The table used in this section has the following fields as Table 10 shows.

- Name: Name of the test.
- Objective: The objective that the test wants to achieve.
- Procedure: The verification procedure explained.
- Expected result: The result expected in the test.
- Obtained result: The result obtained in the test.

Name	
Objective	
Procedure	
Expected result	
Obtained result	

Table 10. Test Table Template

4.2.1. Unitary and integration tests

Name	Test 1
Objective	If the Kinect is not connected, verify than the interface notifies it.
Procedure	Run the code without the Kinect plugged in the USB port.
Expected result	A dialog shows the following sentence: ‘Kinect Sensor is not powered‘.
Obtained result	A dialog shows the following sentence: ‘Kinect Sensor is not powered‘.

Table 11. Test 1

Name	Test 2
Objective	If the Kinect is connected but not plugged in the wall socket, verify than the interface notifies it.
Procedure	Run the code with the Kinect plugged in the USB port but not in the wall socket.
Expected result	A dialog shows the following sentence: ‘Kinect Sensor is not connected‘.
Obtained result	A dialog shows the following sentence: ‘Kinect Sensor is not connected‘.

Table 12. Test 2

Name	Test 3
Objective	Verify the code connects to the middleware NAOqi correctly.
Procedure	1. Run the code with every plugged. 2. Look at the NAOqi information window.
Expected result	The NAOqi window shows the sentence ‘NAOqi is connected‘.
Obtained result	The NAOqi windows shows the sentence ‘NAOqi is connected‘.

Table 13. Test 3

Name	Test 4
Objective	Verify the code connects to the Kinect device.
Procedure	1. Run the code with every plugged. 2. Look at the Kinect window.
Expected result	The Kinect window shows the sentence ‘Kinect Ready’.
Obtained result	The Kinect window shows the sentence ‘Kinect Ready’.

Table 14. Test 4

Name	Test 5
Objective	Verify if the skeleton in shown in the interface of the system.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Move the arms or forearms.
Expected result	The skeleton in shown in the system interface.
Obtained result	The skeleton in shown in the system interface.

Table 15. Test 5

Name	Test 6
Objective	Verify NAO moves the forearms.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Move the forearms.
Expected result	The NAO robot moves the left and right forearms.
Obtained result	The NAO robot moves the left and right forearms.

Table 16. Test 6

Name	Test 7
Objective	Verify NAO moves the arms.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Move left and right arms.
Expected result	The NAO robot moves the left and right arms.
Obtained result	The NAO robot moves the left and right arms.

Table 17. Test 7

Name	Test 8
Objective	Verify the conversion angles works correctly.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Place the arms or forearms in a wanted position.
Expected result	The NAO robot puts the arms and forearms in the same position.
Obtained result	The NAO robot puts the arms and forearms in the same position.

Table 18. Test 8

Name	Test 9
Objective	Verify the NAO's head is never moved.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Move the head in different positions.
Expected result	The NAO robot never moves its head in any angle.
Obtained result	The NAO robot never moves its head in any angle.

Table 19. Test 9

Name	Test 10
Objective	Verify the NAO's legs are never moved.
Procedure	1. Run the code with every plugged. 2. Stands in front of the Kinect camera. 3. Move the legs in different positions.
Expected result	The NAO robot never moves its legs in any angle.
Obtained result	The NAO robot never moves its legs in any angle.

Table 20. Test 10

Name	Test 11
Objective	Verify the operating system used is Windows 7.
Procedure	1. Open the command line using one of the following two options: 1.1. Click in the start menu, write 'cmd', press intro. 1.2. Press 'windows key' and 'R' at the same time. 2. Depending of the language type: 2.1. English: 'systeminfo | findstr /C:"OS Name"' 2.2. Spanish: 'systeminfo | findstr /C:"Nombre del sistema operativo"'
Expected result	The cmd window shows 'Windows 7'.
Obtained result	The cmd window shows 'Windows 7'.

Table 21. Test 11

Name	Test 12
Objective	If any user joint is out of bound that side of the screen turns to red.
Procedure	Move the arms near the bounds.
Expected result	The side where the joints are out of bound turns to red.
Obtained result	The side where the joints are out of bound turns to red.

Table 22. Test 12

Name	Test 13
Objective	If the Kinect does not capture any image the NAO robot does not move.
Procedure	Run the system with nobody in front of the Kinect device.
Expected result	The interface does not show anything and the Nao robot does not move.
Obtained result	The interface does not show anything and the Nao robot does not move.

Table 23. Test 13

In Figure 29 it is shown one of the different tests realized to check the system.

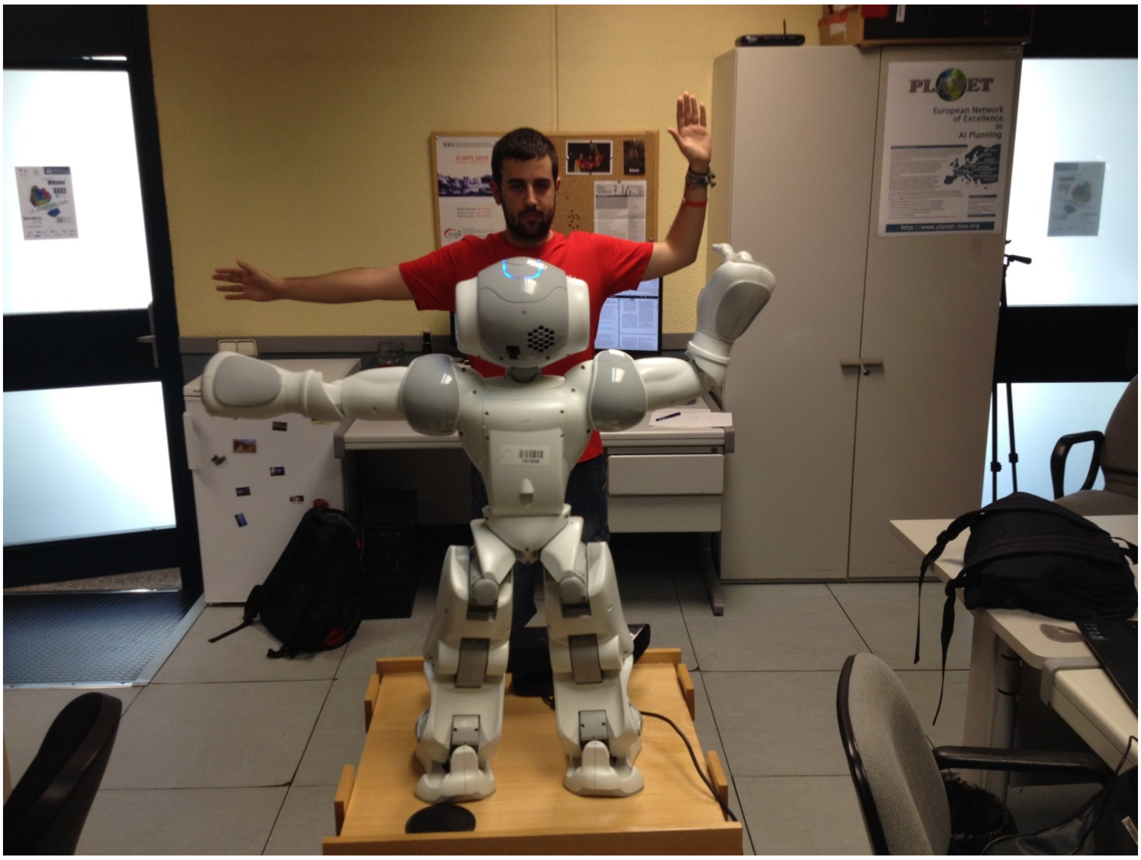


Figure 29. Testing arms and forearms

4.2.2. Requirements tests

In this section, the requirements summary table is shown.

Requirement	Description	Test
FR-1	Capture position	Test 1, test 2, test 5
FR-2	Start teleoperation	Test 8
FR-3	Stop teleoperation	Test 13
FR-4	Move arms	Test 7
FR-5	Move forearms	Test 6
FR-6	Show skeleton	Test 5
FR-7	Notify outbound joints	Test 12
FR-8	Show information	Test 1, test 2

IR-1	Move legs	Test 10
IR-2	Move head	Test 9
IR-3	Spin	Test 9, test 10
NFR-1	Use the NAO robot	Test 3
NFR-2	Use Kinect	Test 4
NFR-3	Use Windows	Test 11
NFR-4	Use NAOqi	Test 3
NFR-5	Delay	See Subsystem section
NFR-6	Divided	See Subsystem section
NFR-7	Easy learning	See Appendix

Table 24. Requirements Tests

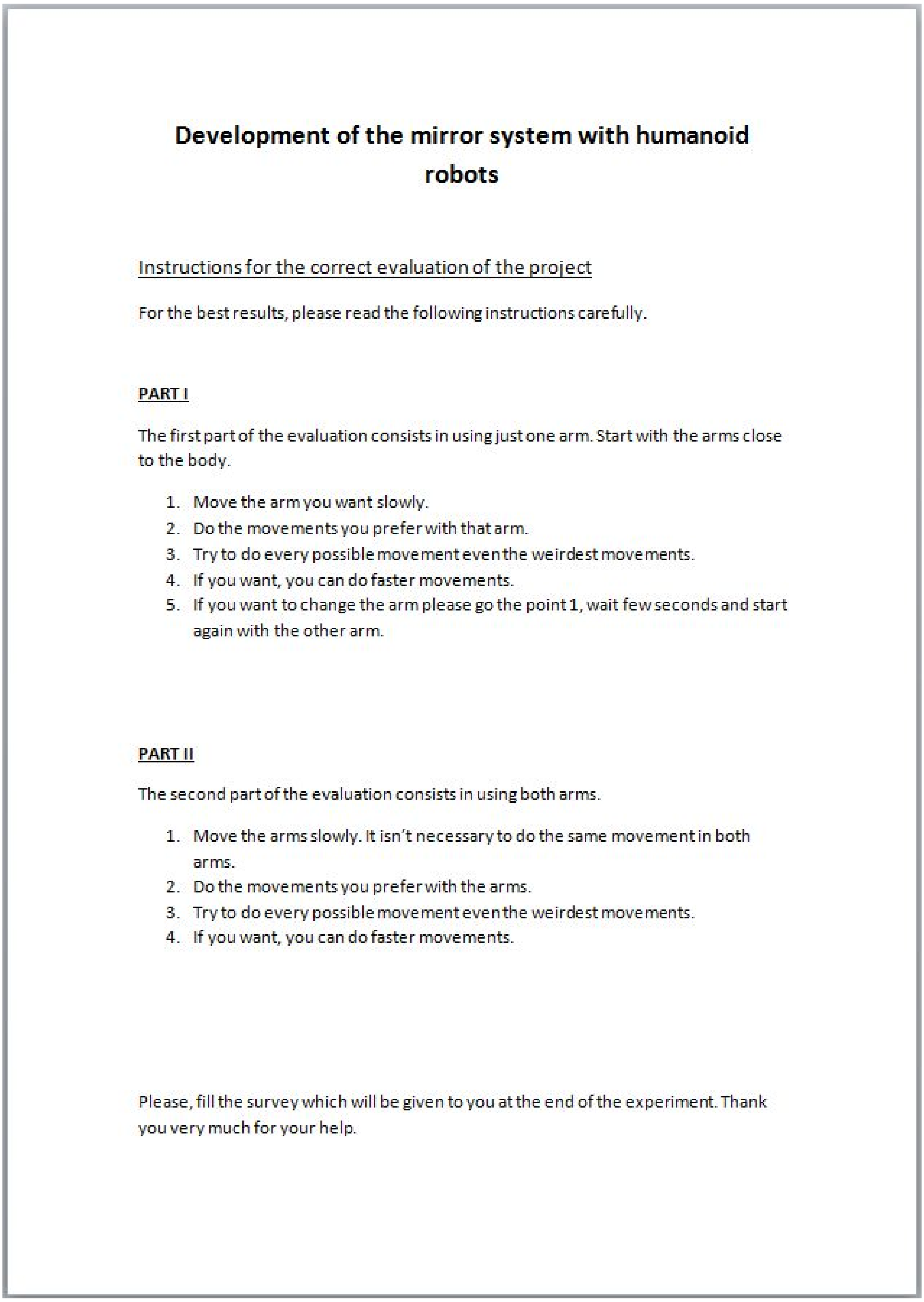
4.3. Experiments

In this section, the experiments will be explained carefully and the results will be presented.

The experiments will be done by four people with different profiles. They will have different computing skills and different ages for trying to cover the biggest population diversity.

4.3.1. Survey Questions

All the users that are going to participate in the experiment will receive a piece of paper like the following with the instructions:



Development of the mirror system with humanoid robots

Instructions for the correct evaluation of the project

For the best results, please read the following instructions carefully.

PART I

The first part of the evaluation consists in using just one arm. Start with the arms close to the body.

1. Move the arm you want slowly.
2. Do the movements you prefer with that arm.
3. Try to do every possible movement even the weirdest movements.
4. If you want, you can do faster movements.
5. If you want to change the arm please go the point 1, wait few seconds and start again with the other arm.

PART II

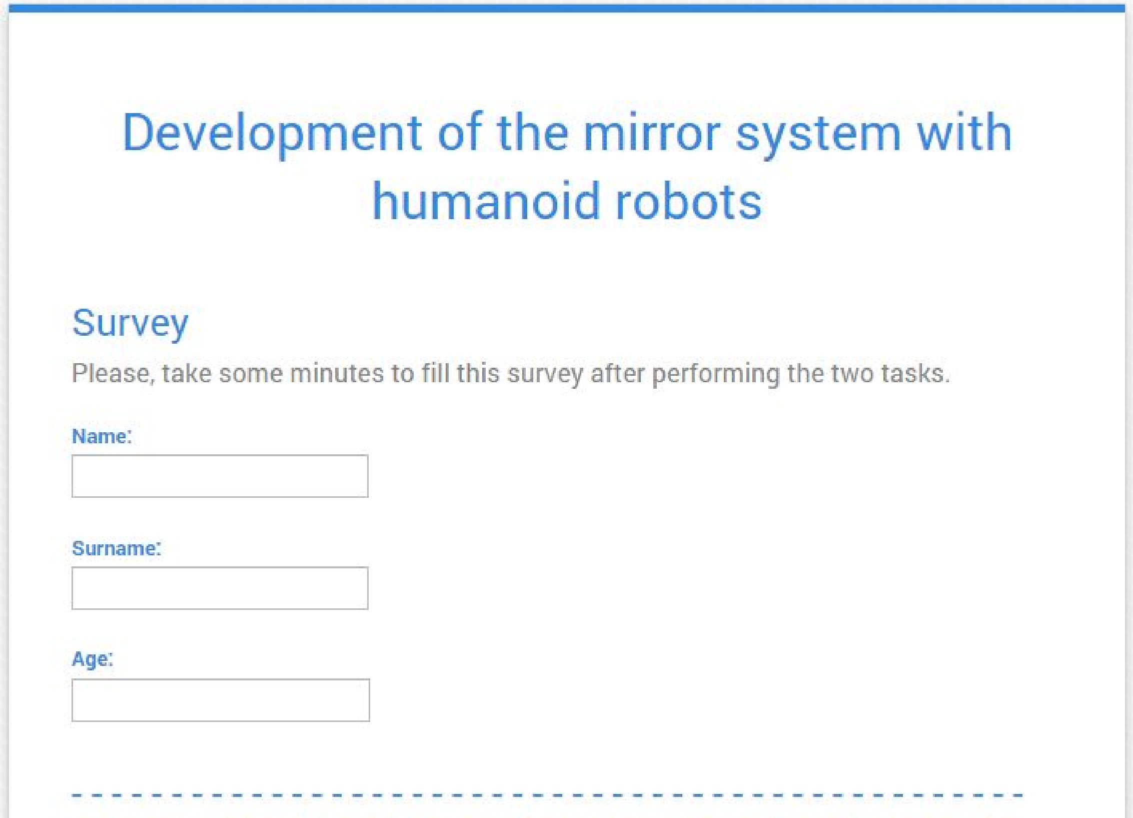
The second part of the evaluation consists in using both arms.

1. Move the arms slowly. It isn't necessary to do the same movement in both arms.
2. Do the movements you prefer with the arms.
3. Try to do every possible movement even the weirdest movements.
4. If you want, you can do faster movements.

Please, fill the survey which will be given to you at the end of the experiment. Thank you very much for your help.

Figure 30. Experiment Instructions

After the experiment each user will receive a survey divided in different parts. The different sections are explained below.



Development of the mirror system with humanoid robots

Survey

Please, take some minutes to fill this survey after performing the two tasks.

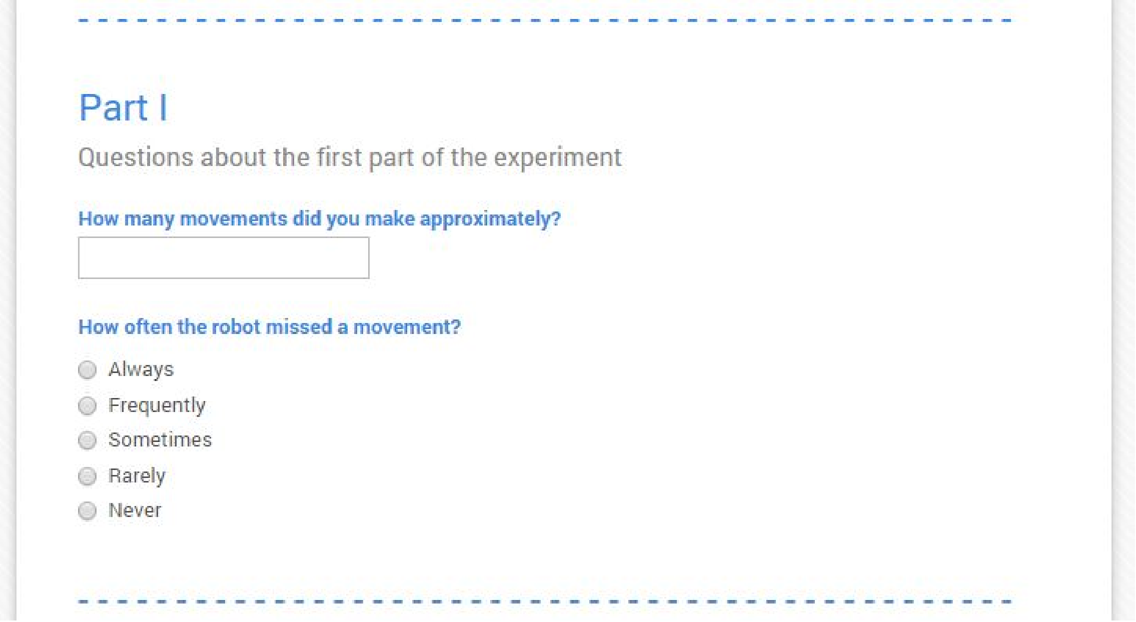
Name:

Surname:

Age:

Figure 31. Survey Introduction

The first part of the survey is the introduction and some fields to be filled by the user with their name, surname and age.



Part I

Questions about the first part of the experiment

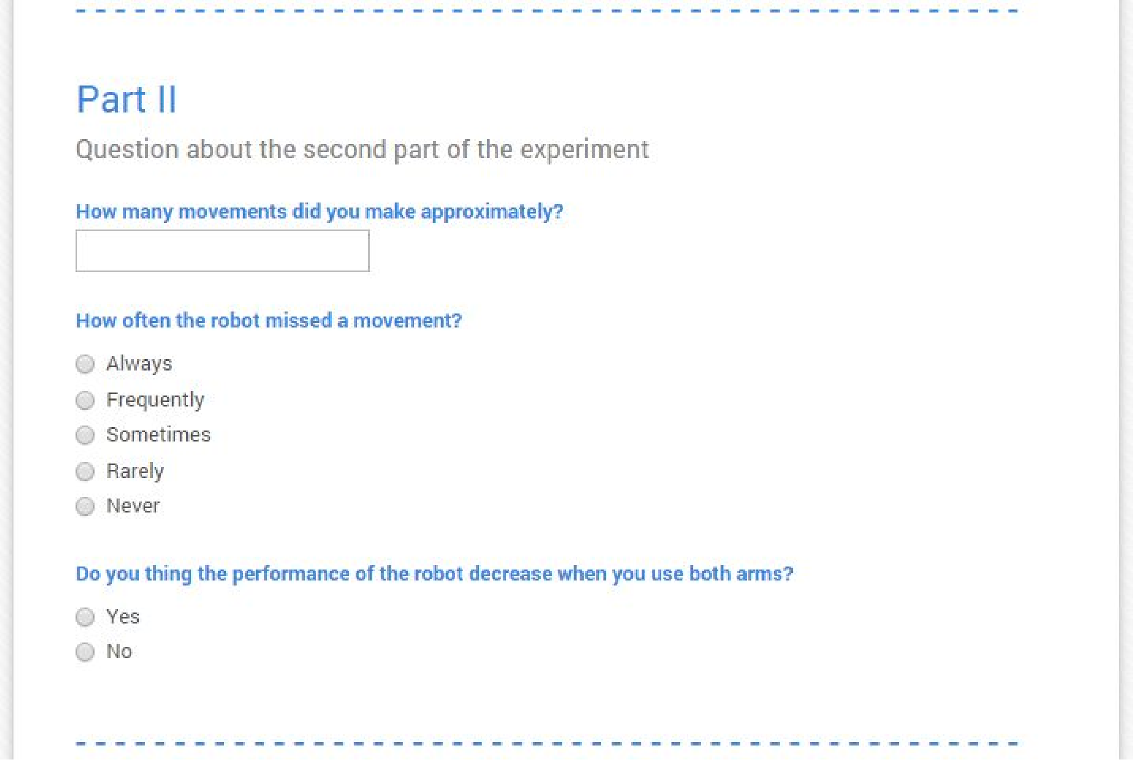
How many movements did you make approximately?

How often the robot missed a movement?

○ Always
○ Frequently
○ Sometimes
○ Rarely
○ Never

Figure 32. Part I Survey

In the next section there are some questions about the first part of the experiment.



Part II

Question about the second part of the experiment

How many movements did you make approximately?

How often the robot missed a movement?

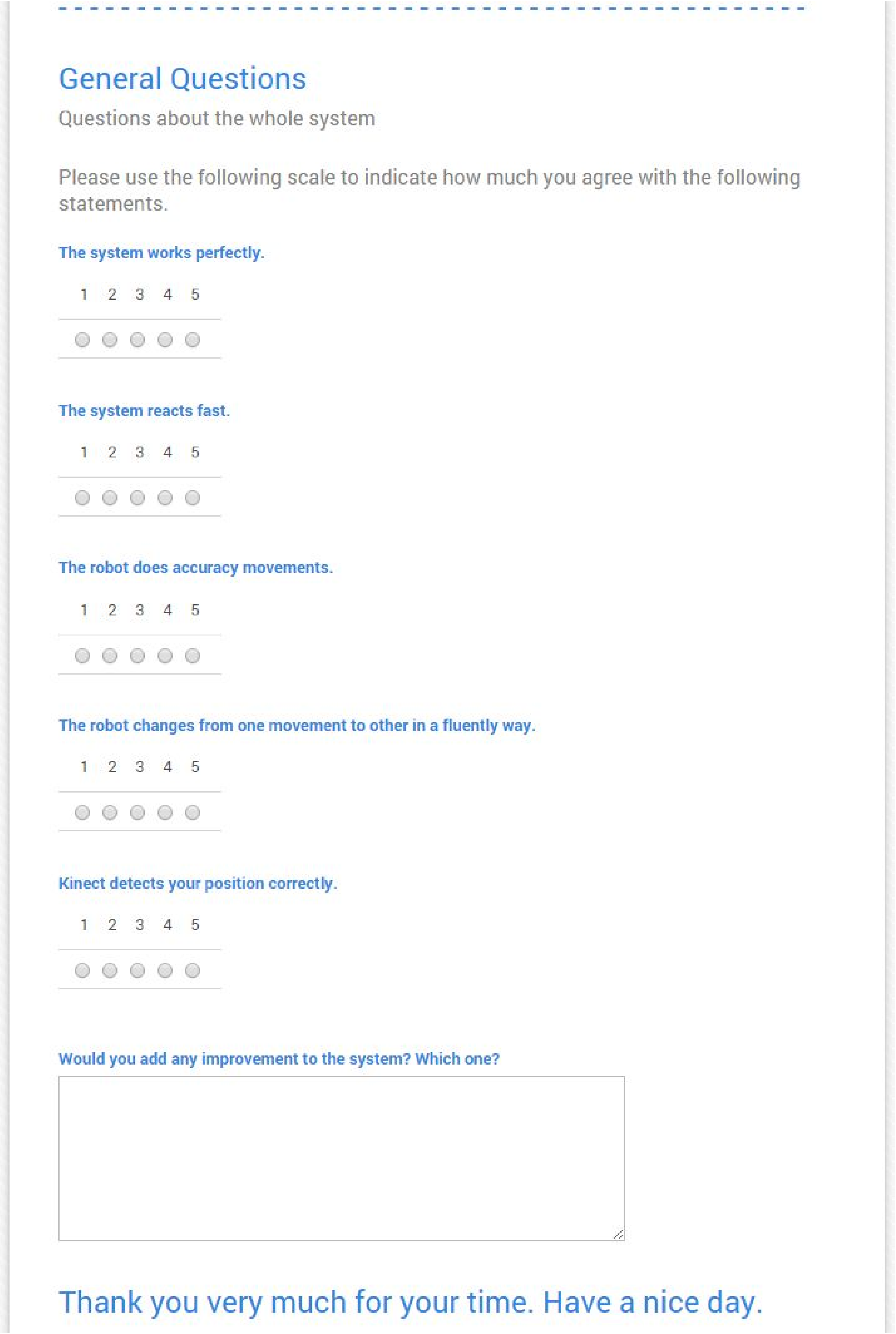
○ Always
○ Frequently
○ Sometimes
○ Rarely
○ Never

Do you thing the performance of the robot decrease when you use both arms?

○ Yes
○ No

Figure 33. Part II Survey

In the Part II section there are questions related to the second part of the experiment.



General Questions

Questions about the whole system

Please use the following scale to indicate how much you agree with the following statements.

The system works perfectly.

1	2	3	4	5
○	○	○	○	○

The system reacts fast.

1	2	3	4	5
○	○	○	○	○

The robot does accuracy movements.

1	2	3	4	5
○	○	○	○	○

The robot changes from one movement to other in a fluently way.

1	2	3	4	5
○	○	○	○	○

Kinect detects your position correctly.

1	2	3	4	5
○	○	○	○	○

Would you add any improvement to the system? Which one?

Thank you very much for your time. Have a nice day.

Figure 34. Global Questions Survey

Finally, in the last part of the survey there are questions related with the whole system in a general way. It has questions about the performance, speed and general feelings.

There is also a question about the possible improvements of the system. This part is very important to take the people opinions into account and can improve the system.

At the end, there is a farewell.

4.3.2. Survey Results

In the following table, a summary of the results is presented:

Questions	User 1	User 2	User 3	User 4
PART I				
How many movements did you make approximately?	15	10	15	10
How often the robot missed a movement?	Never	Never	Rarely	Never
PART II				
How many movements did you make approximately?	20	10	15	10
How often the robot missed a movement?	Rarely	Never	Rarely	Never
Do you thing the performance of the robot decrease when you use both arms?	No	No	No	No

GENERAL QUESTIONS				
The system works perfectly.	5	4	4	5
The system reacts fast.	5	5	5	5
The robot does accuracy movements.	5	5	3	5
The robot changes from one movement to other in a fluently way.	4	3	5	5
Kinect detects your position correctly.	5	4	5	5
Would you add any improvement to the system? Which one?	Yes. Faster movements from the robot.	Yes. Movements more fluent.	No.	No.

Table 25. Survey Results

As it can be seen, the results are very good. There exists some points where I can be improved to achieved a perfect system but the whole system works correctly.

Chapter 5

Project management

In this section, the project management methods will be explained and detailed. These methods are the following:

- Planning: The real time spend in all the different stages of the project since its beginning.
- Economic and Social Impact: Description of the economic and social impact that the project will give. Taking into account the rest of the technology that is in the market.
- Regulatory Framework: The legal framework related with the project and its components.
- Budget: The estimated cost of the total project divided in different categories.

5.1. Planning

For a more comfortable understanding the planning has been divided in several phases that are the following:

- Analysis: The analysis phase details what the system has to do and in which way. The use cases and requirements analysis are created and explained in this phase.
- Design: The design phase details the system architecture, identifying the different elements and the relation between them.
- Implementation: The implementation phase details the development of the system designed in the phase above.

- Evaluation: the evaluation phase starts when there is a first version. The first version is tested in a real environment to correct the possible errors of the system. In this phase, it is also confirmed that all the requirements are covered.

- Documentation: the documentation phase consists in the creation of the different documents that are related with the system that are assembled in this document.

These parts are divided into smaller parts to be more specific in the planning.

In the following table, the time inverted in the different parts and sub-parts is shown. Each row has the name of the part, the start date and the finish date.

Each working day has an estimated value of 2 hours.

Name	Start date	Finish date	Duration (days)
Commence of the project	21/01/2014	22/01/2014	1
Analysis	22/01/2014	18/03/2014	55
C# study	22/01/2014	05/02/2014	14
Visual Studio study	05/02/2014	06/02/2014	1
Kinect study	06/02/2014	15/02/2014	9
NAO study	15/02/2014	18/02/2014	3
Previous thesis study	18/02/2014	11/03/2014	21
Use cases description	11/03/2014	12/03/2014	1
User requirements description	12/03/2014	14/03/2014	2
System requirements description	14/03/2014	16/03/2014	2
Requirements modification	16/03/2014	17/03/2014	1
Requirements validation	17/03/2014	18/03/2014	1
Design	18/03/2014	25/03/2014	7
Architecture design	18/03/2014	21/03/2014	3
Components design	21/03/2014	25/03/2014	4
Implementation	01/06/2014	12/07/2014	39
MainWindow class implementation	01/06/2014	29/06/2014	28
Angles class implementation	30/06/2014	09/07/2014	9
Nao class implementation	10/07/2014	12/07/2014	2
Tests	14/07/2014	21/07/2014	7
Unitary tests	14/07/2014	16/07/2014	2
Integration testing	16/0117/2014	18/07/2014	2
User tests	18/07/2014	21/07/2014	3
Documentation	28/07/2014	06/09/2014	39

Thesis writing	28/07/2014	31/08/2014	34
Presentation slides	01/09/2014	04/09/2014	3
Oral presentation preparation	04/09/2014	06/09/2014	2
TOTAL	21/01/2014	06/09/2014	228

Table 26. Planning

In the following pages the Gantt chart of the project will be shown. As the project is long it is divided in three parts to make easier the understanding.

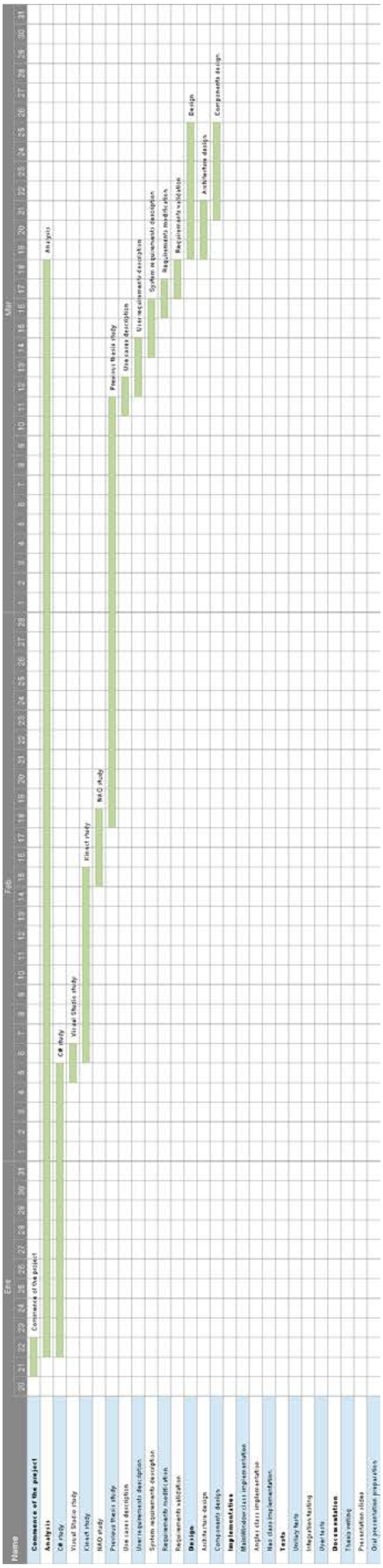


Table 27. Gantt Chart Jan-Mar

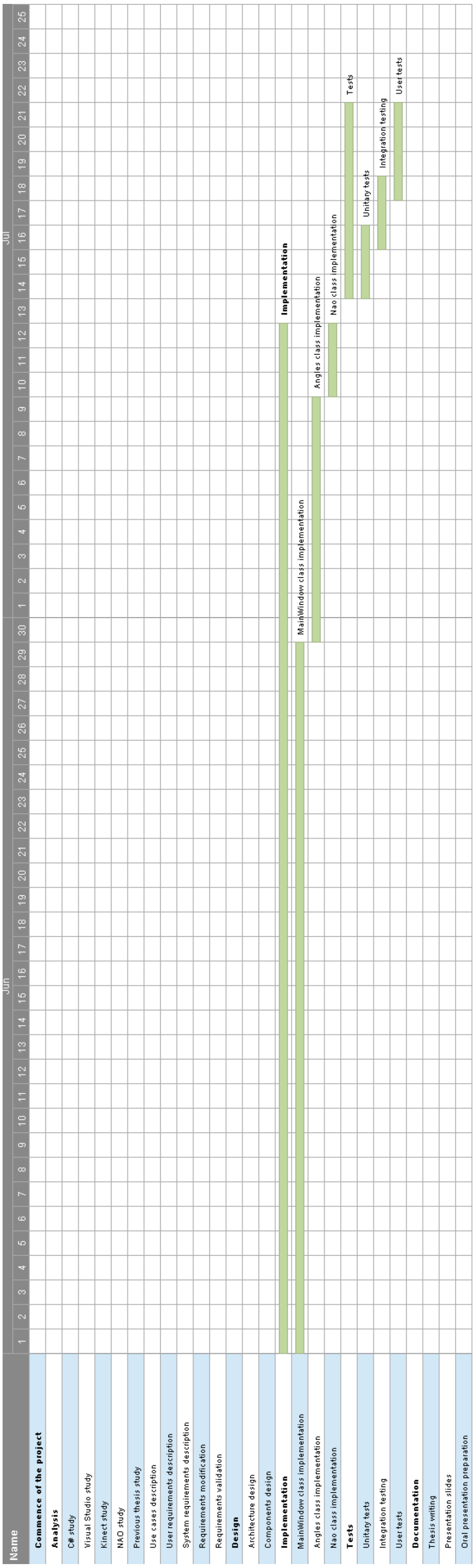


Table 28. Gantt Chart Jun-Jul

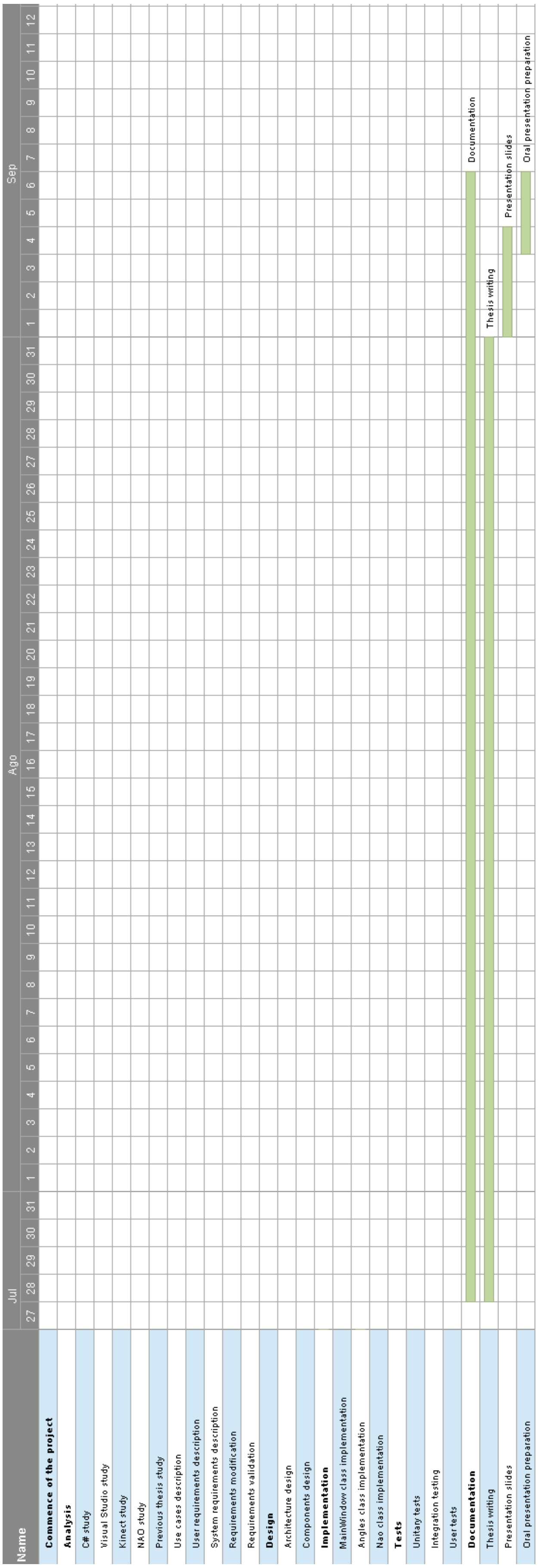


Table 29. Gantt Chart Jul-Sep

5.2. Economic and Social Impact

In this section, the market will be analyzed taking into account the social situation and also the economic situation.

At the present, the live expectancy is higher than in the past and, for that reason, more workforce is needed. The robots can be the workforce necessary with many advantages and few disadvantages. They are able to work in almost every job where a human can. They have more precision, more strength and they do not need to rest.

For that reason, the humanoid robots controlled by teleoperation are perfect in this moment.

On the other, hand there is not almost any company that offers that type of robot, the main robots objectives nowadays are for military or for discovering but not for helping people.

The total cost of the project, the amortization and the different elements cost are explained in the section '5.4 Budget'.

5.3. Regulatory Framework

The regulatory framework for this project and its component is presented below:

- There is not any law about humanoid robots.
- Kinect SDK is free for not commercial purpose. If the user wants to earn money with the project, a commercial license must be bought in the Microsoft webpage.
- Visual Studio 2013 has a free license for students and researchers. People who are not in any of these groups must buy a license key for using it.
- The operating system Windows 7 must be original in any case.

The whole project must be used in an ethical way and follow the laws from the country where it is used.

The author of this document does not support illegal activities that could derive from the application of the exposed techniques.

5.4. Project Budget

The budget can be started when the structure and planning is done. The salaries are going to be transformed in a unique average salary to avoid more complex operations. Before starting it is important to clarify the following points:

- The average analyst salary is 1,400€ per month.
- The average project manager salary is 1,700€ per month.
- The percentage of the withholding for salaries is 34%.
- The work hours per month are 160.
- The involvement of the project manager is around 10% of the total time.
- The indirect costs rate is 20%.

Knowing that information the average net salary is 14.87€ per hour.

The breakdown budget is shown below divided in the different fields.

Staff

Name	Hours	Total
Commence of the project	1	29.74 €
Analysis	110	1,635.7 €
Design	14	208.18 €
Development	78	1,159.86 €
Tests	14	208.18 €
Documentation	78	1,159.86 €
TOTAL		4,401.52 €

Table 30. Staff Budget

Amortization of material

Name	Cost	% Used	Time used	Devaluation	Total cost
Asus n56jn-dm071h	1,000 €	80%	9 months	60 months	120 €
Microsoft Kinect	45 €	100%	9 months	60 months	6.75 €
Nao Robot	12,000 €	25%	9 months	60 months	450 €
TOTAL					576.75 €

Table 31. Amortization

Other costs

Name	Cost
Office supply	50 €
Transport	100 €
Subsistence allowance	500 €
TOTAL	650 €

Table 32. Other Costs

Cost summary

Name	Total
Staff	4,401.52 €
Amortization of material	576.75 €
Other costs	650 €
Indirect costs	1,125.65 €
TOTAL	6753.92 €

Table 33. Cost Summary

The final cost of the total project is 6,753.92 € as it can be in the tables above.

Chapter 6

Conclusions and further development

In this section, the final conclusions, objective fulfillment, problems and possible future work will be presented.

6.1. Final Conclusions

The robotics area is huge and the continuous advance in technology makes it bigger every single day. The new creations and the new minds that collaborate in the area makes possible that in a near future the robotics will be in every home.

This was said at the beginning of the thesis but now, after reading many articles and news those lines get more strength.

The teleoperation is an area that has a huge amount of opportunities. It can be used for many things like medicine, industrial, rescue or investigate dangerous or far places. To simplify, teleoperation will make people lives easier what is the main purpose of the science and technology, help people.

This project helps to understand better the difficulties behind a teleoperation system but also helps to see how useful is.

The Kinect part has been very interesting but hard at the beginning. It is a very powerful device and with the SDK it is perfect for teleoperation systems. The libraries and documentation with the different camera types that it has makes possible to develop almost everything that needs a camera and the developer wants. It’s true that it has some limitations that will be enumerated later but in general it is a powerful sensor.

6.2. Objective Fulfillment

In this section it will be enumerated the same objectives that in the section "1.1 Objectives" and it will said in which degree have been achieve.

- *Study Visual Studio, learn the different useful options and all the possibilities to make a project. It will guarantee that the system is optimized and work correctly.*

  This point was completely accomplished. A very detailed study was made about Visual Studio reading documentation from the official webpage and using other resources to learn correctly about Visual Studio.
  Some small applications using Visual Studio were also made to check that the main functionalities were understood.

- *Analyze the documentation published by Microsoft and the internet community about Kinect and its functionalities.*

  It was also achieved. There exists much documentation about Kinect and in the internet forums and webpages there is much information about Kinect and its functionalities.

- *Study how does the NAO robot work. Study how it connects to the computer, its physics attributes, the way it computes the information, delays and the basic information needed to make it work like the moves of the joints in a correct way.*
  It is important to look for documentation about the libraries, modules and functions that can be re-used or modified.

  This point was a little more difficult than the last one because there is less documentation and information about NAO but it was also achieved. Reading the webpage documentation and some forums make it possible.

- *Design a high level system divided in classes correctly to make possible to re-use the classes if it is needed.*

  The system was created thinking in a subsystem union for future development or improvements, so it is achieved.

- *Design how it is going to be the teleoperation procedure knowing that it is very important the ease of the system and the high usability. It is also important to get the smallest delays possible.*

  It is a difficult point to see if it has been completely achieved. The system is as easy as it was possible and the user does not need any previous knowledge about robotics, computers or teleoperation. For that reason, this goal is marked as achieved but there exist the possibilities of change the things that people considered more difficult.

- *Implement the teleoperation system, making the tests necessaries to get the best possible result. If the tests are not satisfactory, the implementation must be changed, or even the design, to achieve the best result.*
  It is indispensable that the code is commented as most as possible to make easier the changes.

  Completely achieved, all the tests that were necessaries were made to confirm that the system works properly.
  The code is also commented in every method and almost every line to inform in the best way to the developer who wants to read or modify it.

- *Document the most relevant data and the information that can help to understand the project to the people who want to study deeply the project or to make new functionalities.*

  The documentation is abundant and everything related with the project was explained. Any person with a minimum degree of knowledge in computer science will understand the whole system reading the documentation.

6.3. Problems encountered

In this section the main problems encountered will be detailed and it will also explained how were solved that problems

- **Limited Kinect Documentation for C#**
  There exists much documentation about Kinect, in the Microsoft webpage and in other webpages but most of them explain how to use it with C++ and not C#. It gives many problems because although both programming languages are similar, the number of differences is worthy of consideration.
  Another problem was that some of the libraries for C# were not as complete as the C++ libraries.

  The solution for this was trying to convert that code to C# or looking for another ways to achieve that target.
  For the libraries problem it was necessary to add the functionalities to the own code to get the values wanted.

- **Same variable names in different libraries**
  Initially, the main idea was to use two different libraries in the code, one for the Kinect skeleton capturing for obtaining all the joints and shows them on the screen. The other library was for track better the NAO's head because it was impossible to track it using the same system than with the arms as it will be explained in the following point. The problem with these two libraries is that both libraries share some names from objects completely different and it makes compilation errors.

  The solution was to use the complete reference path for each object each time the program needed to use it but even in that case the compiler gives some errors. For that reason the final solution was to use only one library and try other way to calculate the face coordinates.

- **NAO's Head Movement**

  At the beginning, it was expected to add some face tracking options to the system and, with that, the robot would move the head like the user in the teleoperation.

  There were two problems. The first one was the conflict between the different object names from the two libraries used. The second one was the NAO's head limited movement. The NAO's head has limited angles movements so the moves should be converted from the human move to the robot moves.

  The head moves are not very useful in the NAO's teleoperation objectives, perhaps if the NAO's cameras are on it can be useful but in that case the user should use another type of screen because with a computer screen the user will not see anything when the head is turned.

  For that reason, the idea was discarded. In the future, it can be resumed if it is necessary.

- **Delays**

  The system is designed to reduce the delays as much as possible but in a system that is formed for different elements and with some of the connected using wireless it is impossible to reduce the delay to 0.

  The delays are not a big problem if they are not big but when the delays are big, the teleoperation performance will be awful and almost impossible to make anything with the system.

  The system has been restructured until the delay is the smaller as possible to obtain the best results when using it.

6.4. Further Development

Some of the improvements or modifications that can be done in the future are the following:

- Use a better camera sensor that has more precision to add fingers recognition and make able to use the hands.
- Add more joints to the teleoperation system like the head or legs.
- Use artificial intelligence to make possible to the robot to realize some simple tasks without an operator.
- Use the NAO's camera to have streaming video for what the NAO is watching in that moment.
- Give the information to the user using the speakers.
- The user could change options using the microphone
- Adapt the system for using the NAO to teach to people some moves and using the Kinect for checking that the moves are like the NAO moves.
- Using the last point, create therapies for people with diseases or problems in the arms and forearms with the NAO showing the moves and correcting them if something is wrong.
- Use other robots.

Appendix A

Requirements

A.1. Functional Requirements

Name	Capture position		
Description	The system must be able to localize the user for creating a skeleton and start to calculate the angles from the joints.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 34. Capture Position Requirement

Name	Start teleoperation		
Description	The system must be able to start the teleoperation when the skeleton is found.		
Priority	High	**Necessity**	Essential
Stability	No	**Verifiability**	High

Table 35. Start Teleoperation Requirement

Name	Stop teleoperation		
Description	The system must be able to stop the teleoperation when the user stops the system.		
Priority	High	**Necessity**	Essential
Stability	No	**Verifiability**	High

Table 36. Stop Teleoperation Requirement

Name	Move arms		
Description	The system must be able to calculate and send the angles from the arms to the robot using the captured data.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 37. Move Arms Requirement

Name	Move forearms		
Description	The system must be able to calculate and send the angles from the forearms to the robot using the captured data.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 38. Move Forearms Requirement

Name	Show skeleton		
Description	The system must be able to show the skeleton captured from the user on the screen.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 39. Show Skeleton Requirement

Name	Notify outbound joints		
Description	The system must be able to notify when any joint or extremity is out of the Kinect angle vision.		
Priority	Low	**Necessity**	Ideal
Stability	Yes	**Verifiability**	High

Table 40. Notify Outbound Joints Requirement

Name	Show information		
Description	The system must be able to show the state of the Kinect camera.		
Priority	Low	**Necessity**	Ideal
Stability	Yes	**Verifiability**	High

Table 41. Show Information Requirement

A.2. Inverse Requirements

Name	Move legs		
Description	The system must not move the legs imitating the user.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 42. Move legs Requirement

Name	Move head		
Description	The system must not move the head imitating the user.		
Priority	Low	**Necessity**	Ideal
Stability	No	**Verifiability**	High

Table 43. Move Head Requirement

Name	Spin		
Description	The system must not spin imitating the user.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 44. Spin Requirement

A.3. Non-functional Requirements

Name	Use the NAO robot		
Description	The robot used in the project will be NAO.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 45. NAO Robot Requirement

Name	Use Kinect		
Description	The image will be captured with Kinect.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 46. Kinect Requirement

Name	Use Windows		
Description	The system must be executed in Windows 7		
Priority	High	**Necessity**	Ideal
Stability	No	**Verifiability**	High

Table 47. Windows Requirement

Name	Use NAOqi		
Description	The system must use NAOqi for communicating to the NAO robot.		
Priority	High	**Necessity**	Ideal
Stability	Yes	**Verifiability**	High

Table 48. NAOqi Requirement

Name	Delay		
Description	The delay between the user and the robot NAO must be as small as possible.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 49. Delay Requirement

Name	Divided		
Description	The system will be divided in classes so in the future it can be extended or modified.		
Priority	Medium	**Necessity**	Ideal
Stability	Yes	**Verifiability**	High

Table 50. Divided Requirement

Name	Easy learning		
Description	It will not be difficult to learn how to use the system.		
Priority	High	**Necessity**	Essential
Stability	Yes	**Verifiability**	High

Table 51. Easy Learning Requirement

Appendix B

Installation Guide

This guide will explain how to install everything to have the teleoperation system ready to work.

B.1. Visual Studio 2013 installation

Visual Studio 2013 can be downloaded from the next webpage: http://www.visualstudio.com/es-es/downloads

The Ultimate Edition is the most complete but any version of the Visual Studio 2013 will work correctly with the project, for that reason the user can choose the one that consider better for its interests.

The user will log in to the Microsoft account and download it, the Visual Studio Software is not free but there exists a 90 days trial for check if the teleoperation and the software is what they expected.

When the software is downloaded, the user can execute the installer, choose the options wanted and finally install it.

B.2. Microsoft Kinect SDK v2.0

Microsoft Kinect SDK v2.0 can be downloaded from the next webpage: http://www.microsoft.com/en-us/download/details.aspx?id=43661

For downloading the Microsoft Kinect SDK v2.0 it is not necessary to have a Microsoft account and it is completely free.

When the software is downloaded, the user can execute the installer, chose the options wanted and finally install it.

B.3. Web Browser

It is necessary to have and use a web browser to connect with the NAO Robot. Almost all the computers have at least one but here is a list with the most popular web browsers and their download link:

- Internet Explorer 11
  http://windows.microsoft.com/en-us/internet-explorer/download-ie

- SRWare Iron
  http://www.srware.net/es/software_srware_iron_download.php

- Mozilla Firefox
  https://www.mozilla.org/en-US/firefox/new/

- Google Chrome
  http://www.google.com/intl/en-us/chrome/

- Opera
  http://www.opera.com/

B.4. NAOqi Framework (optional)

It is not necessary but it is highly recommended to use NAOqi Framework. It will make easier the modifications of the code and the testing of the functionalities.

Naoqi Framework can be downloaded from the next webpage: https://community.aldebaran.com/

The user must be registered in the webpage to be able to download it.

When the software is downloaded, the user can execute it without any installation.

Appendix C

First Execution Guide

The first execution of the system is different because it is necessary to configure some parameters.

C.1. Nao Configuration

If the Nao has never been connected to the internet, the user must remove the hatch behind NAO's head to access to the Ethernet socket and plug in the Ethernet Cable

Now, the user can push the NAO's chest button and NAO will say its IP address.

The user should open the web browser and type the IP address from the NAO robot in the address bar.

A login and password will be asked for, login: nao; password: nao.

The user can changes some configurations of the NAO if it is wanted but the most important is to go to the network tab and there the user can connect the NAO to the wireless connection.

C.2. Code Configuration

The user now must change a line in the software code.
The user has to open the Nao.cs in the application path using any text editor or IDE.

Then it is necessary to go to the line where it is said “Connect with the NAO”, in the line below there is typed an IP address, please change it using the new one. The port is not necessary to change it if the user did not change any port option from the NAO.

C.3. Visual Studio Configuration

The user has to open the Project using Visual Studio 2013 and go to the References option in the menu on the right side. The user has to click with the right button on that option and select ‘Add Reference…’.

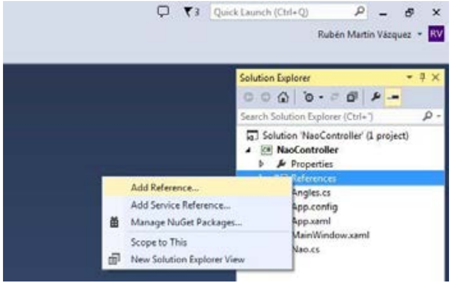
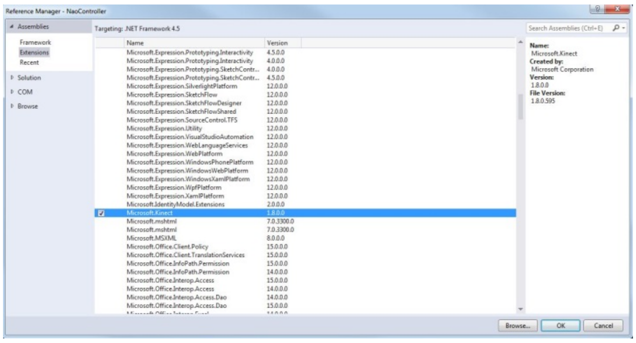


Figure 35. References Option

Then, user has to click on the ‘Extensions’ options inside ‘Assemblies’ and search ‘Microsoft.Kinect’ in the list. When it is selected the user just checks it and presses ‘OK’ button.



Targeting: .NET Framework 4.5

	Name	Version
	Microsoft.Expression.Prototyping.Interactivity	4.5.0.0
	Microsoft.Expression.Prototyping.Interactivity	4.0.0.0
	Microsoft.Expression.Prototyping.SketchContr...	4.0.0.0
	Microsoft.Expression.Prototyping.SketchContr...	4.5.0.0
	Microsoft.Expression.SilverlightPlatform	12.0.0.0
	Microsoft.Expression.SketchFlow	12.0.0.0
	Microsoft.Expression.SketchFlowDesigner	12.0.0.0
	Microsoft.Expression.SketchFlowShared	12.0.0.0
	Microsoft.Expression.SourceControl.TFS	12.0.0.0
	Microsoft.Expression.Utility	12.0.0.0
	Microsoft.Expression.VisualStudioAutomation	12.0.0.0
	Microsoft.Expression.WebLanguageServices	12.0.0.0
	Microsoft.Expression.WebPlatform	12.0.0.0
	Microsoft.Expression.WindowsPhonePlatform	12.0.0.0
	Microsoft.Expression.WindowsWebPlatform	12.0.0.0
	Microsoft.Expression.WindowsXamlPlatform	12.0.0.0
	Microsoft.Expression.WpfPlatform	12.0.0.0
	Microsoft.Expression.XamlPlatform	12.0.0.0
	Microsoft.IdentityModel.Extensions	2.0.0.0
☑	Microsoft.Kinect	1.8.0.0
	Microsoft.mshtml	7.0.3300.0
	Microsoft.mshtml	7.0.3300.0
	Microsoft.MSXML	8.0.0.0
	Microsoft.Office.Client.Policy	15.0.0.0
	Microsoft.Office.Client.TranslationServices	15.0.0.0
	Microsoft.Office.InfoPath.Permission	15.0.0.0
	Microsoft.Office.InfoPath.Permission	14.0.0.0
	Microsoft.Office.Interop.Access	15.0.0.0
	Microsoft.Office.Interop.Access	14.0.0.0
	Microsoft.Office.Interop.Access.Dao	14.0.0.0
	Microsoft.Office.Interop.Access.Dao	15.0.0.0

Name:
Microsoft.Kinect
Created by:
Microsoft Corporation
Version:
1.8.0.0
File Version:
1.8.0.595

Figure 36. Microsoft.Kinect Reference

The system is ready to work.

Appendix D

User Manual

To obtain the better results the distance between the user and the Kinect device must be around 2 m. It has to be on a horizontal surface with nothing in front of the sensor.

To start, the user has to go to the application path and execute the project. If everything was correct a screen like the following should appear.

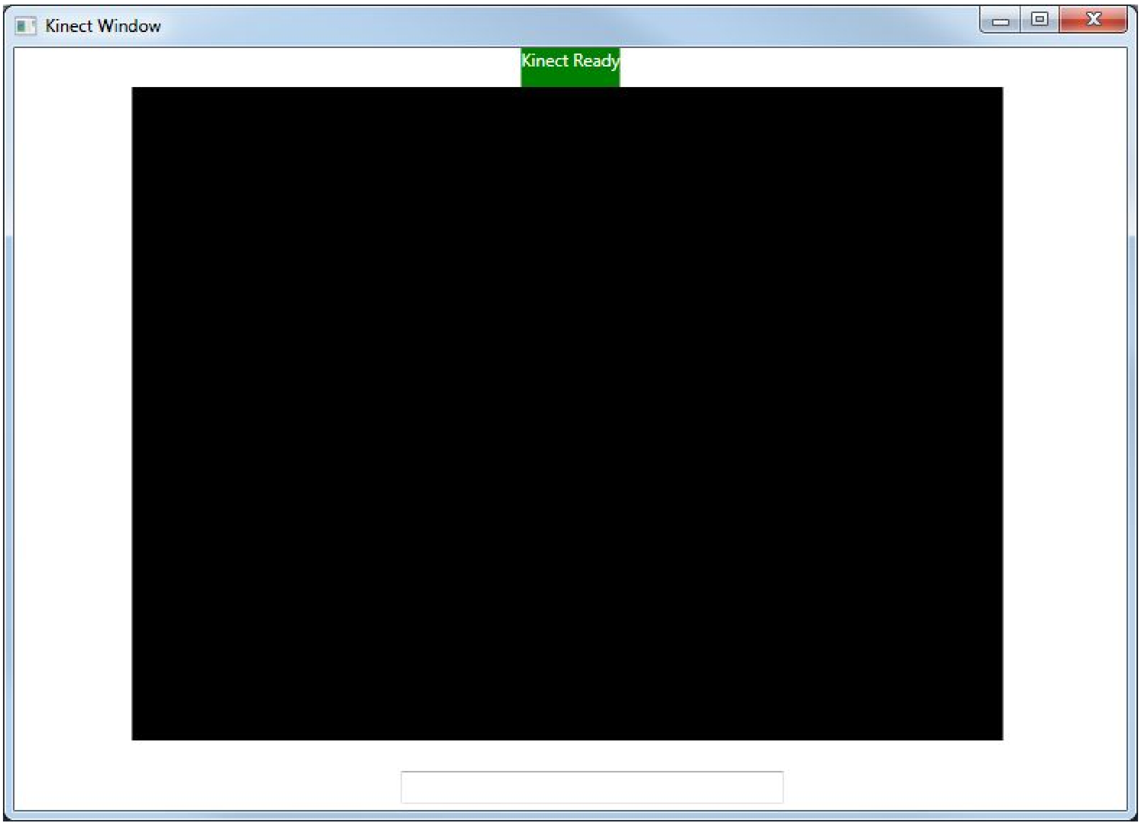
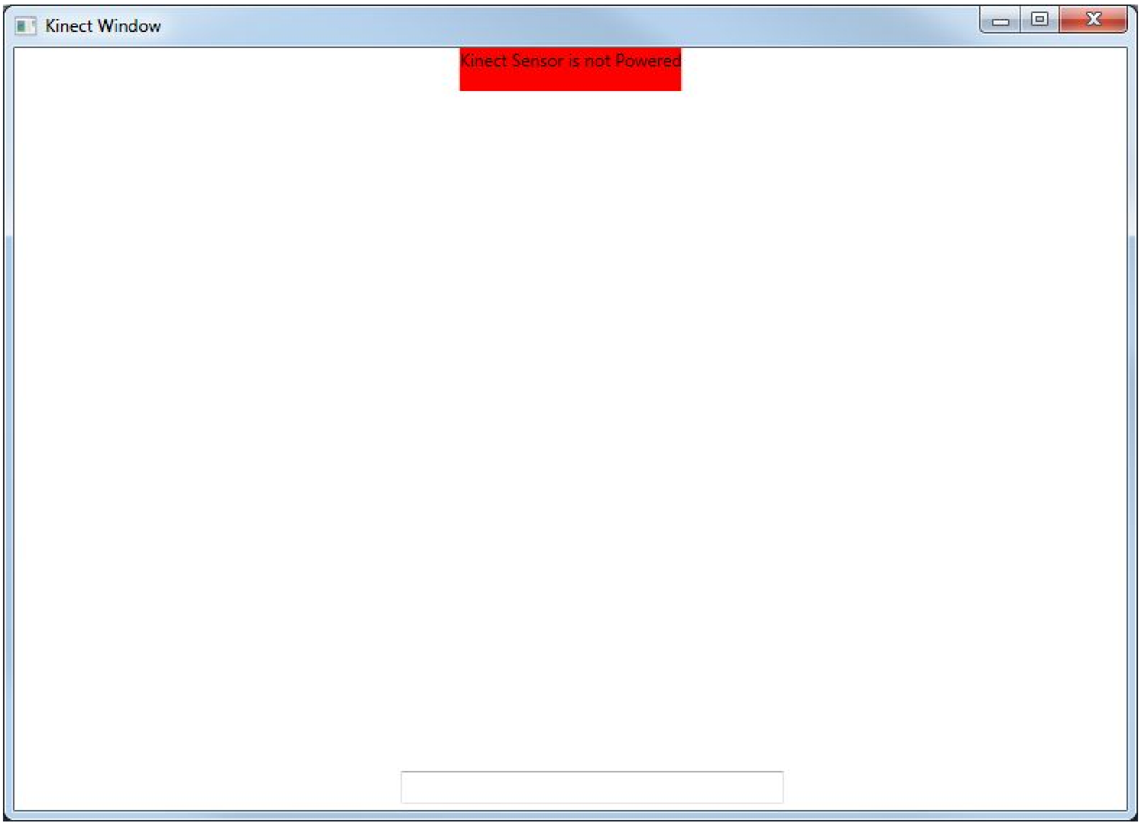


Figure 37. Kinect Windows Interface Screen

If the window is not like that, it probably be because the Kinect Device is not powered, in that case the window will be like this.



Kinect Sensor is not Powered

Figure 38. Kinect Sensor is not Powered

Just close the application, connect the Kinect plug to the wall socket and run again the project.

Now, stand up in front of the sensor and move your arms lightly until a green skeleton appears in the screen like the following.

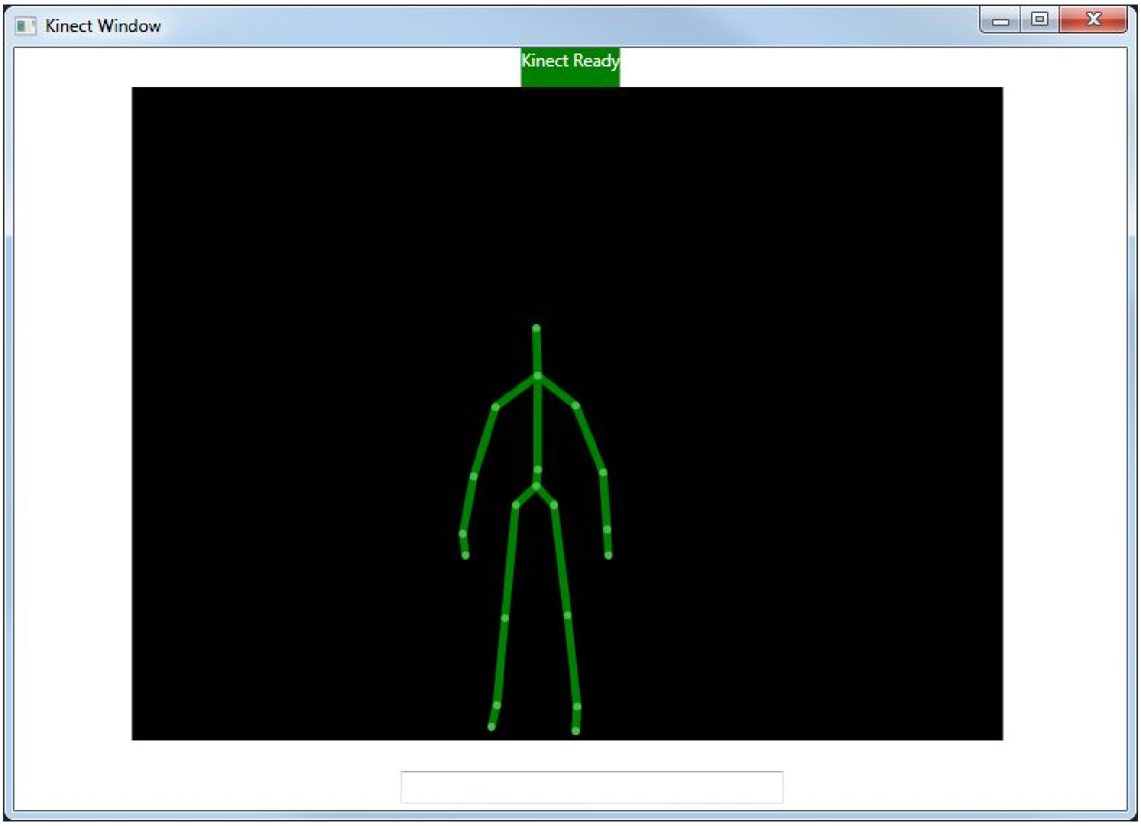


Figure 39. Kinect Skeleton Detected

You can move as you want and the NAO robot will do the same moves that you do with your arms using its own arms, realize the teleoperation you want or just enjoy it.

When you are doing the teleoperation you have to be aware that maybe, in some moment you move accidentally some joint out of the Kinect sensor. In that case the NAO will continue working if the joints needed are still captured but in any case, the side of the window will turn red to notify to the user the out of bound problem.

An example of that situation is shown in Figure 40.

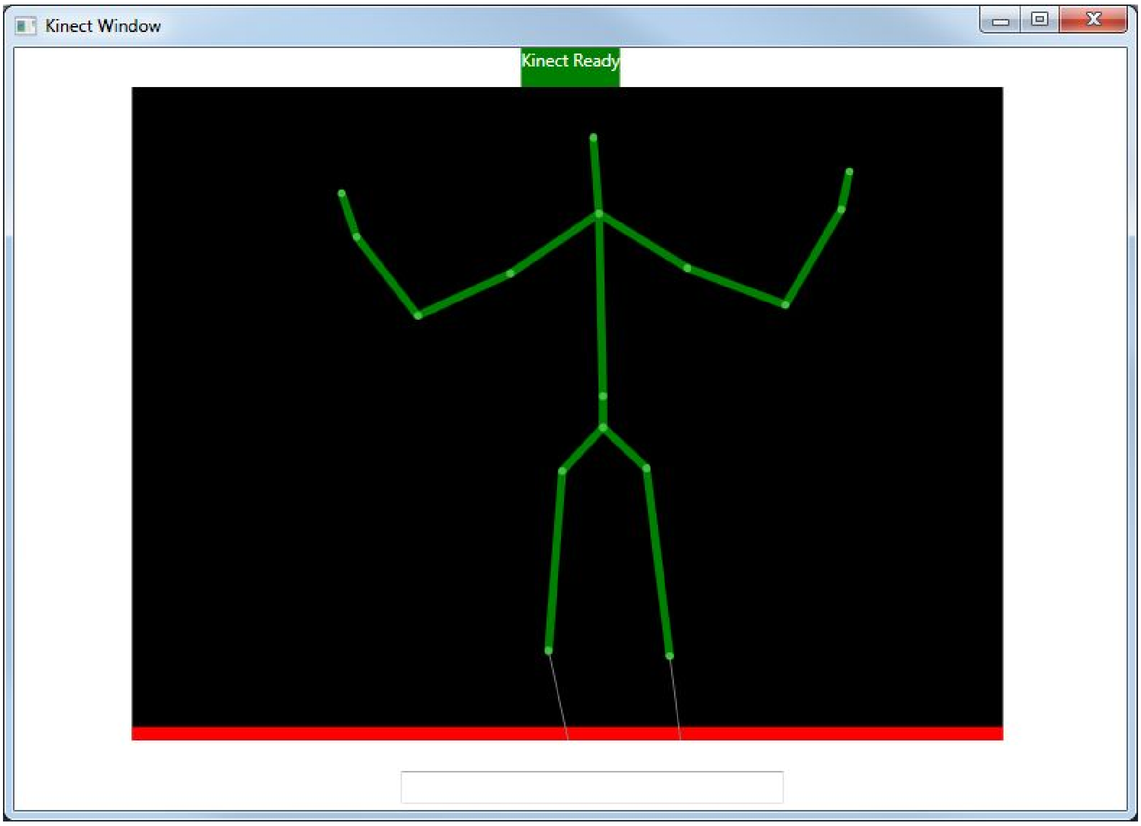


Figure 40. Out of Bound Joints

In this case, the user is too near of the Kinect Sensor and for that reason it cannot track correctly the ankle joints. In this case the NAO will work correctly because it does not use the legs.

When the user wants to close the software, the application must be close clicking on the red x button on the upper and right side.

Glossary

- **Accelerometer:** It is an instrument for measuring acceleration.
- **Actuator:** A servomechanism that is responsible for moving or controlling a mechanism or system.
- **Aldebaran Robotics:** It is a French enterprise that was created in the year 2005. NAO was created by them.
- **Bumper:** It is a pressure sensor commonly used in robotics for detecting walls an avoid falls.
- **E3:** The initials of Electronic Entertainment Expo.
- **Ethernet:** It is a network protocol featuring a bus topology.
- **FPS:** The initials of Frames per Second.
- **Framework:** It is an abstraction in which software providing generic functionality can be selectively changed by additional user-written code.
- **FSR:** The initials of Force-sensing resistor.
- **Gantt chart:** It is a type of bar chart that illustrates the start and finish dates of the terminal elements and summary elements of a project.
- **Gyro sensor:** It is a sensor that detects and measure spins.
- **Hardware:** It is the physical equipment used in a computer system
- **IDE:** The initials of Integrated Development Environment, software prepared for helping people to code.
- **IP Address:** It is a numeric string that identifies a device in a network with the protocol IP.
- **LED:** The initials of Light-emitting diode.
- **Library:** It is a collection of resources used by programs on a computer.

- **Middleware:** It is computer software that provides services to software applications beyond those available from the operating system.
- **RGB:** The initials of Red, Green and Blue. These colors are the three additive primary colors.
- **Rover:** It is a remote-controlled vehicle which roams over rough, extraterrestrial, terrain taking photographs, gathering rock samples, etc.
- **Sensor:** It is a mechanical device sensitive to light, temperature, radiation level, or the like, that transmits a signal to a measuring or control instrument.
- **Smartphone:** It is a device that combines a cell phone with a hand-held computer, typically offering Internet access, data storage, e-mail capability, etc.
- **Software:** The programs, programming languages, and data that direct the operations of a computer system.
- **SDK:** The initials of Software Development Kit, special software packages that make easier developing for a given product.
- **Sonar:** It is a method for detecting and locating objects by echolocation.
- **VGA:** The initials of Video Graphics Arrays, a protocol for emitting pictures/video on a screen.
- **Wi-Fi:** It is a local area wireless technologic.

Bibliography

[1] Google, Ipsos OTX MediaCT, «Our Mobile Planet: Global Smartphone Users». [Online]. Available: http://services.google.com/fh/files/blogs/final_global_smartphone_user_study_2012.pdf

[2] The App Date, «Informe sobre las apps en España 2013», 2013. [Online]. Available: http://madrid.theappdate.com/informe-sobre-las-apps-en-espana-2013-asi-usamos-las-apps/

[3] K. Yokoi, «La robótica japonesa, presente y futuro», 2008. [Online]. Available: http://www.uc3m.es/portal/page/portal/actualidad_cientifica/noticias/conferencia_yokoi

[4] Rafael Aracil, Carlos Balaguer y Manuel Armada, «Robots de servicio», 2008. [Online]. Available: http://earchivo.uc3m.es/bitstream/10016/9855/1/robots_balaguer_riai_2008.pdf

[5] R. Chellali, «Tele-operation and Human Robots Interactions», 2010.

[6] P. G.-R., J. Torrijos, «Robots de Seguridad y Defensa». [Online]. Available: http://www.disam.upm.es/~barrientos/Curso_Robots_Servicio/R_servicio/Defensa_files/Robots%20de%20Seguridad%20y%20defensa.pdf

[7] «Programa Lunojod». [Online]. Available: http://es.wikipedia.org/wiki/Programa_Lunojod

[8] «Misiones espaciales a Marte». [Online]. Available: http://astrocienciauniverso.blogspot.com.es/2011/07/misiones-espaciales-marte-lista.html

[9] «Los robots de Fukushima,» El Mundo, 17 04 2011.

[10] NAO History. [Online]. Available: http://www.aldebaran.com/en/robotics-company/history

[11] NAO Hardware Platform. [Online]. Available: http://www.aldebaran-robotics.com/en/Discover---NAO/Key---Features/hardware---platform.html

[12] Terrence Fong, Charles Thorpe, Charles Baur, «Advanced Interfaces for Vehicle Teleoperation: Collaborative Control, Sensor Fusion Displays, and Remote Driving Tools». [Online]. Available: http://infoscience.epfl.ch/record/30035/files/AR01-ATI-TF.pdf

[13] Anca D. Dragan, Siddhartha S. Srinivasa, Kenton C. T. Lee, «Teleoperation with Intelligent and Customizable Interfaces». [Online]. Available: https://www.ri.cmu.edu/pub_files/2013/1/JHRI12-invited.pdf

[14] M.L. Fitzgerald, A.J. Barbera, «A low-level control interface for robot manipulators». [Online]. Available: http://www.sciencedirect.com/science/article/pii/0736584585901073

[15] Stefan Fuchs, Stefan May, «Calibration and Registration for Precise Surface Reconstruction with TOF Cameras». [Online]. Available: http://www.robotic.de/fileadmin/robotic/fuchs/TOFCamerasFuchsMay2007.pdf

[16] Stefan Fuchs, «Calibration and Multipath Mitigation for Increased Accuracy of Time-of-Flight Camera Measurements in Robotic Applications», Dissertation, July 2012, TU Berlin, Germany

[17] Nisha Sharma, Swati Up al, Sorabh Gupta, «Technology Based on Touch: Haptics Technology», IEEE, 2011.

[18] «Microsoft games exec details how Project Natal was born», 2009. [Online]. Available: http://venturebeat.com/2009/06/02/microsoft-games-executivedescribes-origins-of-project-natal-game-controls/

[19] Epstein, Zach, «MICROSOFT SAYS XBOX 360 SALES HAVE SURPASSED 76 MILLION UNITS, KINECT SALES TOP 24 MILLION», February, 2013. [Online]. Available: http://bgr.com/2013/02/12/microsoft-xbox-360-sales-2013-325481/

[20] Sorayapa, «Kinect, un gran avance para la humanidad y la cultura Hacker». [Online]. Available: http://www.sorayapaniagua.com/2011/12/27/kinect-un-gran-avance-para-la-humanidad-y-la-cultura-hacker/

[21] «Adafruit Contest Winner Announcement», November, 2010. [Online]. Available: http://www.adafruit.com/blog/2010/11/10/we-have-a-winner-open-kinect-drivers-released-winner-will-use-3k-for-more-hacking-plus-an-additional-2k-goes-to-the-eff/

[22] "Me gusta trastear" [I like to tinker]. El País. November, 2010. [Online]. Available: http://www.elpais.com/articulo/tecnologia/gusta/trastear/elpeputec/20101111elpeputec_4/Tes

[23] «Web oficial de Kinect para Windows», [Online]. Available: http://www.microsoft.com/en-us/kinectforwindows/

[24] «Kinect: The company behind the tech explains how it works», 2010. [Online]. Available: http://www.joystiq.com/2010/06/19/kinect-how-it-works-fromthe-company-behind-the-tech/

[25] «Kinect: The company behind the tech explains how it works», 2010. [Online]. Available: http://www.joystiq.com/2010/06/19/kinect-how-it-works-fromthe-company-behind-the-tech/

[26] Carlo Dal Mutto, Pietro Zanuttigh, Guido M Cortelazzo, «Time-of-Flight Cameras and Microsoft Kinect™», 2012, pp. 33-47.

[27] Navab, Victor Castaneda, Nassir, «Time-of-Flight and Kinect Imaging», 2011. [Online]. Available: http://campar.in.tum.de/twiki/pub/Chair/TeachingSs11Kinect/2011-DSensors_LabCourse_Kinect.pdf

[28] «Kinect: Cómo funciona su micrófono multiarray», 2010. [Online]. Available: http://www.pensamientoscomputables.com/entrada/Kinect/microfono/multiarray/como-funciona/xbox-360

[29] «Ten Great Public Health Achievements -- United States, 1900-1999». [Online]. Available: http://www.cdc.gov/mmwr/preview/mmwrhtml/00056796.htm

[30] First Steps with Nao. [Online]. Available: https://community.aldebaran.com/resources/tutorial/first-steps-with-nao/

[31] ALFARO BALLESTEROS, Santiago. “Sistema de teleoperación mediante una interfaz natural de usuario”. Director: Moisés Martínez Muñoz. Proyecto fin de carrera. Universidad Carlos III de Madrid.

[32] LOPEZ PRADO, Fernando. “Desarrollo de terapias de rehabilitación motora con un robot NAO autónomo”. Director: Fernando Fernández Rebollo. Proyecto fin de máster. Universidad Carlos III de Madrid.