MOBILE MAPPING CAR

Project Report

Supervisor: MSc. Torben Gregersen

Team: AAD1

Editors:

- Guillaume Trousseau
- Jaime Escuer
- Iñaki Abadia
- Roger Prat

AARHUS
UNIVERSITY

Applied App Development
(ITIPRJ)

Content

1 Resume

This document describes the whole process of our project “Mobile Mapping Car” in a short summary.

The mobile mapping car is an interesting project regarding its capabilities of mapping rooms which can be difficult for human access. These rooms can even be without any source of light. The system will mainly consist of a small robot car, an Arduino microcontroller board, a Wi-Fi shield, some ultrasonic sensors, a smartphone with camera and an “App” to control everything via another Smartphone.

Achieved goals:

- Connection between Microcontroller and Smartphone
- Control the car with a Smartphone
- Receive video streaming from an IP camera
- Obtain data from the sensors, plot it in a map and get a layout of the room
- Store different maps

We finally got all the parts of our project working fine. We had to work very hard in different task and sometimes we had to change some specifications to make all the system functional.

2 Introduction

Our project has been developed by exchange students from the Applied App Development course.

To realize this project we have used our knowledge obtained in lessons: ITSMAP (Android smartphone application), ITIECA (Embedded computer architecture) and ITBFIS (User interface for embedded systems).

The organization of the project has been done following the guide line given in ETCCCP (Cooperation, learning and project work).

2.1 Project Management

The project has been carried out according to the Agreement of Collaboration.

2.1.1 Software students (Iñaki Abadia & Guillaume Trousseau)

The software part of the project can be divided into four sub-tasks:

- Making the connection between the App and the microcontroller.
- Creating a live video streaming between the App and the IP Cam.
- Creating a map management system.
- Developing a maps drawing program.

We split the task so that the Map management and the Drawing program would be developed by the same person since they are linked. The other two task share the same theme (connection between devices) so we conclude that it was the best way to share the work.

2.1.2 Hardware students (Jaime Escuer & Roger Prat)

The principal hardware task was create a code to have a connection between the smartphone and the Arduino board using a Wi-Fi shield. We used some functions integrated in the Arduino libraries to create some PWM signals and move the car.

Then we implement different codes to test the sensors, we were looking for the best kind of implementation because after that we had to merge all the codes.

2.2 The Tasks

The principal tasks developed in our project are:

2.2.1 Smartphone Application

This is one of the most important devices of our project, the objective of the application is control the car and all the external components that we have to create maps of the environment where we are.

2.2.2 Control the car

The principal components that we can find here are the Arduino board Mega 2560, a Wi-Fi Shield, the motor controllers and a structure with four ultrasonic sensors moved by a servomotor.

2.2.3 Video streaming

We have an IP camera to stream a video from the car to the main smartphone, helping to drive (In our project we use a Smartphone with an IP camera app).

2.2.4 Distance measurement

All the hardware part is focused in the measures of the ultrasonic sensors. We have each sensor oriented in a direction, this sensors measure the distance of the objects around the car and send, to the main smartphone, the distance with the angle of the measure.

2.2.5 Creating and managing maps

With the data obtained with the ultrasonic sensors, we create a map plotting the points. We also can save this maps in a database and delete the ones that we want to discard.

The choice of this project was due to the knowledge of the group members and the subjects studied during the semester. We think that the project was a good experience to work in group with students from different engineering disciplines.

3 System description

3.1 Project parts

We have the following documents in our project:

- Preanalysis & Choice of components
- Requirements specification
- System architecture and design
- Acceptance test
- Project report

3.2 Preanalysis

In this document we explain the choice of the different devices in our project, and the different methods to get the goals in our project.

During this phase we had to compare the different Arduino boards, the best sensor to our implementation and the best communication to transfer the data between the car ant the smartphone.

All the components have been provided by the university and we tried to choose within the possibilities to get all of them.

3.3 Requirements

There we described the system and the functionalities. Furthermore we defined our use cases and the external interface requirements.

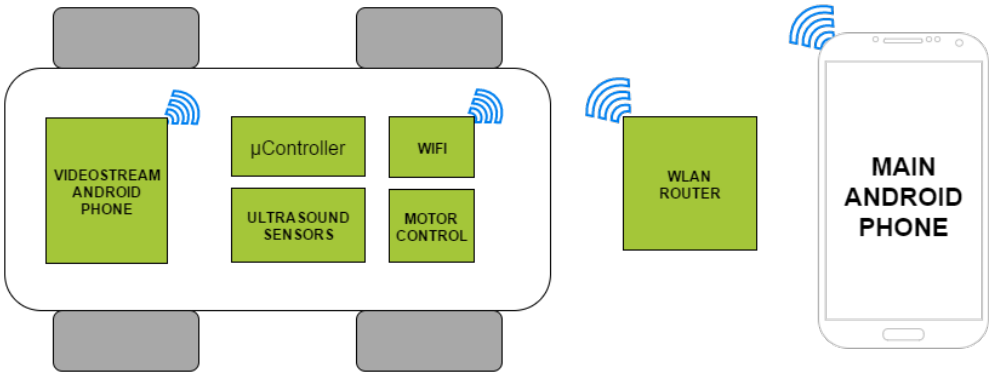


Figure 1: System overview

In figure above the main components of our project are shown. We can differentiate two main parts, the car and the main Smartphone. The car is composed by:

- Another smartphone, in charge of the video streaming.
- Motor control.
- Ultrasonic sensors.
- The Arduino MEGA and WIFI shields, steering the car and sending back the sensors data.

3.4 The context

The Mobile Mapping Car project is a great project with many uses in the field of rescues and exploration of inaccessible places.

Now we show the use case diagram, the aim of this diagram is to show an overview of features which can be performed.

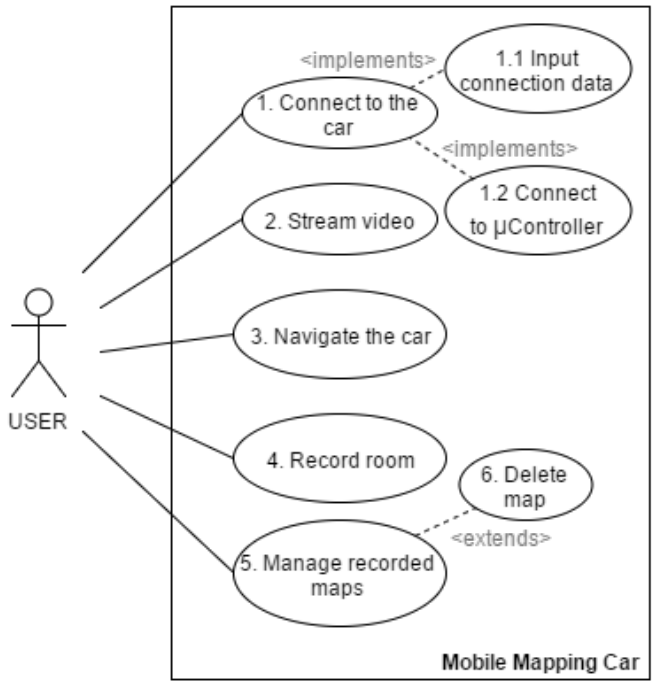


Figure 2: Use cases diagram

The main functionalities for this project are:

- Create connections with the Arduino board and the IP camera.
- Send a video stream from an IP camera mounted in the car to the smartphone.
- Control the car with a smartphone app.
- Measure distance with the sensors and send the data to the smartphone.
- Manage the data to create maps.

Application development has been created with the knowledge acquired in ITSMAP (Android smartphone application) and ITBFIS (User interface for embedded systems):

- **Usability:** the app has been developed using an easy layout and an easy method to star the connection (default button to fill the gaps).
- **User interface:** we do not have too much buttons with larges text, we have some precise buttons with symbols easy to understand.
- **Reliability:** all the communications are configured to not block the different functionalities. We also have the control of all the task, so we can start and end all the functions with all security.

You will find more information about the description and the functionality of the system in the document ***Requirements_specification.pdf***

3.5 Process

3.5.1 Project management

Used Methods:

All the decisions, related to de project, were taken in common with all the members of the group. The good thing of the group members is our different knowledge in the different branches of the project, we took this such a way to organize the different tasks to develop the system.

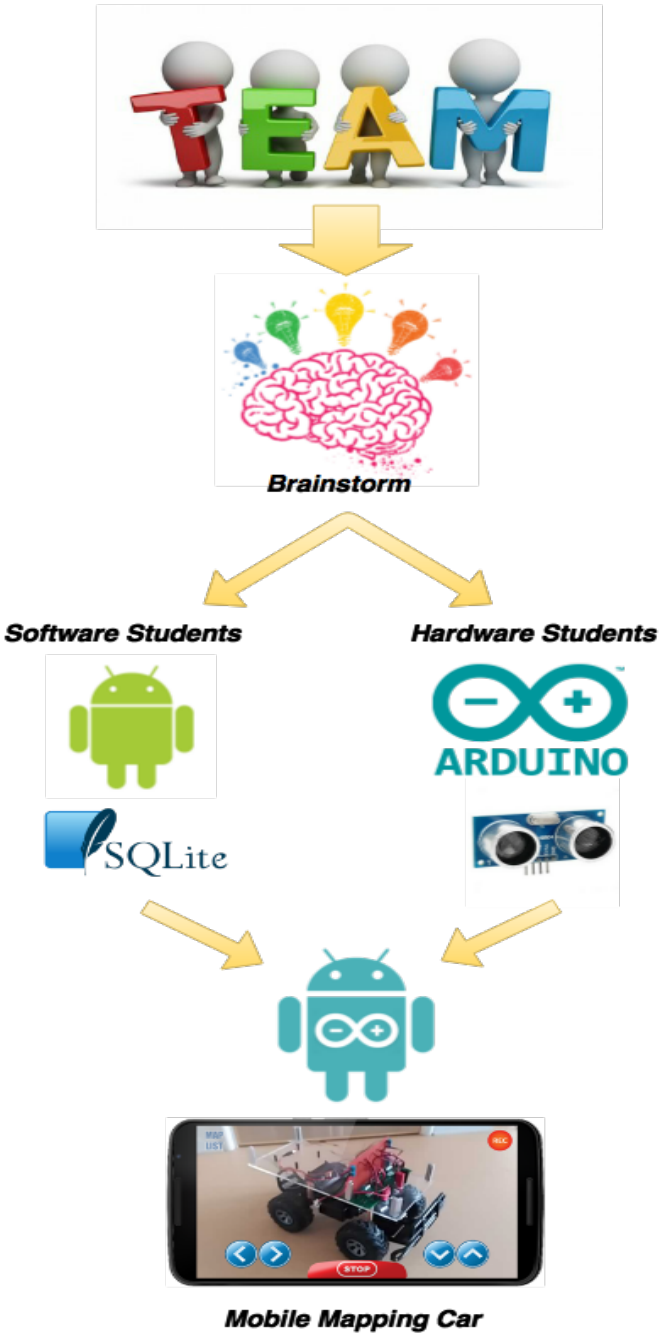


Figure 3: Work method

ASE Model:

This model describes the steps followed in the documentation. We have been very strict creating the different documents, therefore we started with the Requirements specification where we included the system description, then we were working in the software and hardware to finally create the Accept test report were we tested our project.

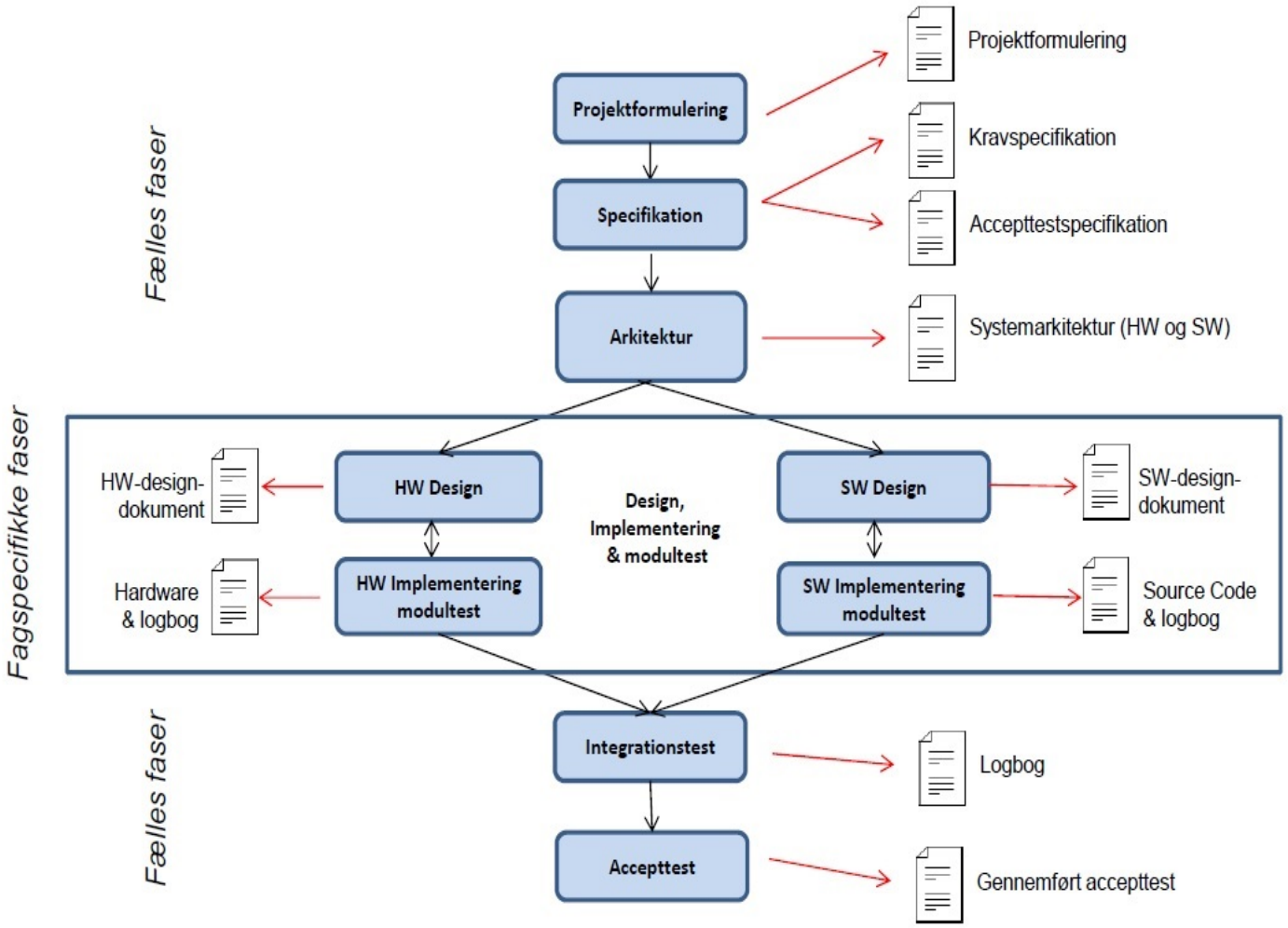


Figure 4: ASE model

UML (software):

We used the Unified Modeling Language (UML) for our software development and to create our used cases. This language intended to provide a standard way to visualize the design of a system. For the software development we used the sequence diagrams and class diagrams. For the requirement specification we made use of the used case diagram.

SysML (Hardware):

The System Modeling Language (SysML) was used in the hardware development. It supports the specification, analysis, design, verification and validation of a broad range of systems and systems-of-systems.
With this Language we can shoe all the different ports and communications between all the devices.
We used the Block Definition Diagram (BDD) to define the system and the external components. The Internal Block Diagram (IBD) was used to show how all the parts in the system are connected.

3.5.2 Information searching

To develop this project, we have to say that all the courses that we have taken have been useful. However, we have had to search for information on how to do many different things that we haven't learned in our subjects.

The main parts that we searched for information are:

- Android programming
- Arduino programming
- Accelerometers
- MJPEG streams

For the Android programming we had to search for different things that we hadn't learned in the Smartphone Applications subject such as how to plot points, how to implement a TCP or how to stream a video.

The most remarkable websites for getting this kind of information are:

- http://developer.android.com/index.html [1]
- http://www.vogella.com/tutorials/android.html [2]

For the Arduino programming we had to search for how to do almost everything and we got most of our information from the Arduino official website:

- https://www.arduino.cc/ [3]

Even though we didn't use accelerometers in the end, we searched for a lot of information about them because our first idea was to track the car's position and this seemed a good way to do it.

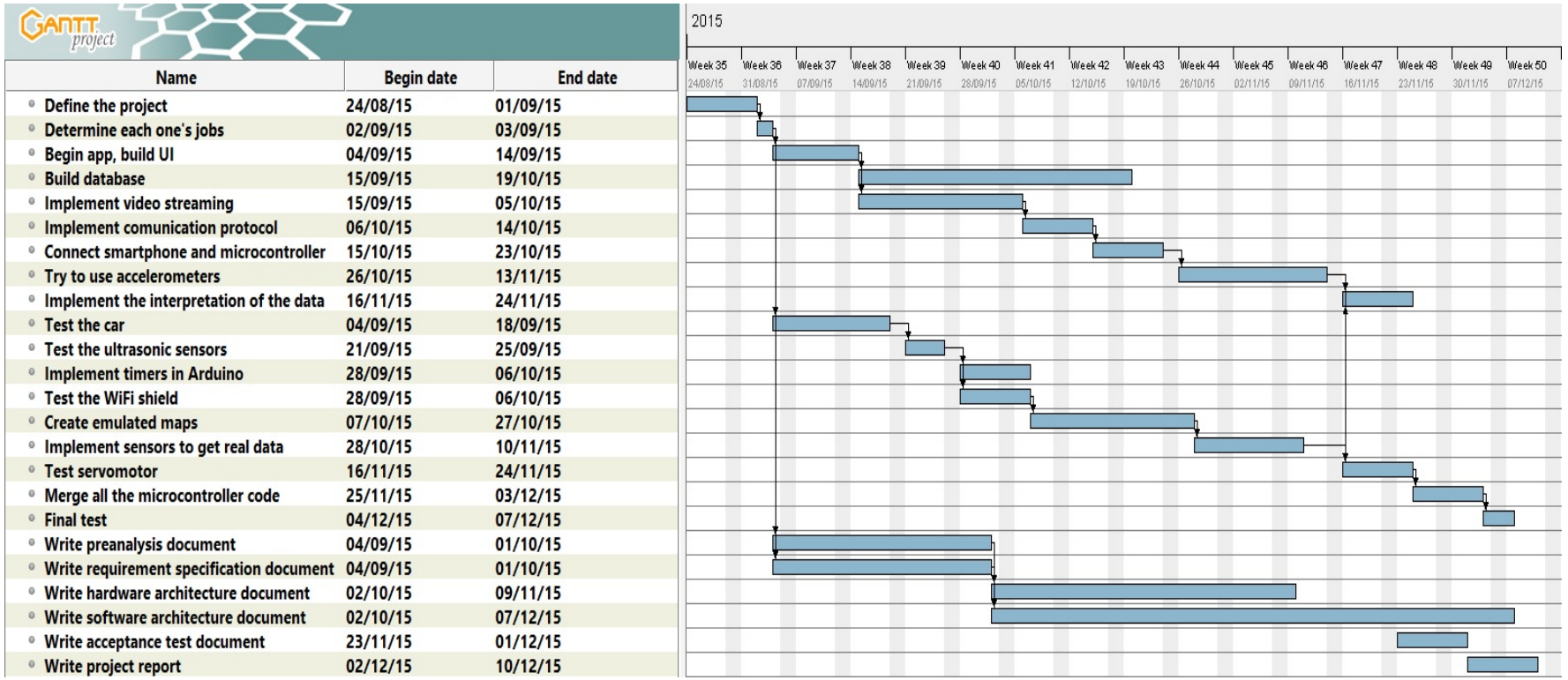
Most of the information about this topic was extracted from stackoverflow posts, some of them:

- http://electronics.stackexchange.com/questions/97656/track-3d-position-with-an-accelerometer [4]
- http://stackoverflow.com/questions/7499959/indoor-positioning-system-based-on-gyroscope-and-accelerometer [5]

We also searched for much information about MJPEG steams to implement our video streaming. We made some research, but the final solution came from an idea a teacher gave us.

3.5.2 Time schedule

To show the organization of the work we made a GANT diagram where we can find the work in the different areas of the project. The first part is the software development, second part is related to the hardware parts and finally we can see the work in the documentation.



Name	Begin date	End date
Define the project	24/08/15	01/09/15
Determine each one's jobs	02/09/15	03/09/15
Begin app, build UI	04/09/15	14/09/15
Build database	15/09/15	19/10/15
Implement video streaming	15/09/15	05/10/15
Implement comunication protocol	06/10/15	14/10/15
Connect smartphone and microcontroller	15/10/15	23/10/15
Try to use accelerometers	26/10/15	13/11/15
Implement the interpretation of the data	16/11/15	24/11/15
Test the car	04/09/15	18/09/15
Test the ultrasonic sensors	21/09/15	25/09/15
Implement timers in Arduino	28/09/15	06/10/15
Test the WiFi shield	28/09/15	06/10/15
Create emulated maps	07/10/15	27/10/15
Implement sensors to get real data	28/10/15	10/11/15
Test servomotor	16/11/15	24/11/15
Merge all the microcontroller code	25/11/15	03/12/15
Final test	04/12/15	07/12/15
Write preanalysis document	04/09/15	01/10/15
Write requirement specification document	04/09/15	01/10/15
Write hardware architecture document	02/10/15	09/11/15
Write software architecture document	02/10/15	07/12/15
Write acceptance test document	23/11/15	01/12/15
Write project report	02/12/15	10/12/15

Figure 5: GANTT diagram

3.6 Development tools

3.6.1 Microsoft Word

We have used this program to write all our documentation, we made a template to present all the differ documents with the same structure and appearance.

3.6.2 Dropbox

Dropbox is a file hosting service. A _le which is uploaded to Dropbox can be downloaded from every computer which is connected to the internet, assumed this computer has the access rights.

We created one common folder to upload all our progress with the reports and to share every important document except the software codes.

3.6.3 GranttProject

It is GPL-licensed (free software) Java based and project management software.

We used the tool to generate a GANT project plan.

3.6.4 Draw.io

It is an online tool that is used to model, represent and visualise information. Among other uses, such diagrams are often used in software and technical development and business to represent dataflows, workflows, software architecture and organizational charts.

We used this to generate all the diagrams in the software and hardware design (UML and SysML).

3.6.5 Bitbucket

Bitbucket is a web-based hosting service for projects that use either the Mercurial or Git revision control systems.

It is a great tool to collaborate with teams. We have used GitHub to share with the members our progress in the smartphone application code and in the Arduino code.

3.6.6 Android studio

Android Studio is an integrated development environment (IDE) for developing for the Android platform.

We have developed all the Android app with this tool that we used in ITSMAP (Android smartphone application)

3.6.7 Arduino Software (IDE)

This is the free Arduino software to program and debug.

We used it to program all the Arduino code. It has some useful libraries that we used for the implementation of the sensors and the servo.

3.6.8 Photoshop

Adobe Photoshop is a raster graphics editor. We used this software to create some buttons of our user interface and some edited pictures for our documents.

3.6.9 MATLAB

MATLAB (matrix laboratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms.

We were working with this program trying to simulate the accelerometers and gyroscopes to implement in the project. Finally we rule out this task.

3.6.10 Hangouts

Google Hangouts is a communication platform developed by Google which includes instant messaging, video chat, SMS and VOIP features.

To have a daily communication between the members of the team, we decided to create one group in this platform to organize the meetings and obviously for the instant communication.

4 Product

4.1 System Architecture and Design

4.1.1 Protocol

Once the smartphone and the microcontroller are connected via TCP sockets the microcontroller is continuously listening for commands. We have steering commands (Forward, backward, left and right), data commands (new set of points, stop sending) and the kill connection command.

When the user starts recording, the Smartphone will be ready to ask for 9 sets of points to the microcontroller, unless the user decides to abort the operation (In which case the map will be plotted only with the points received to that moment). When the Smartphone asks for the new set of points, then expects a set of points (Obvious) and sends the next command which can be either next set of points or stop sending.

4.1.2 Deployment view

Here we can see how the different parts of the system are implemented on different platforms. Also, communication and protocols will be explained in detail.

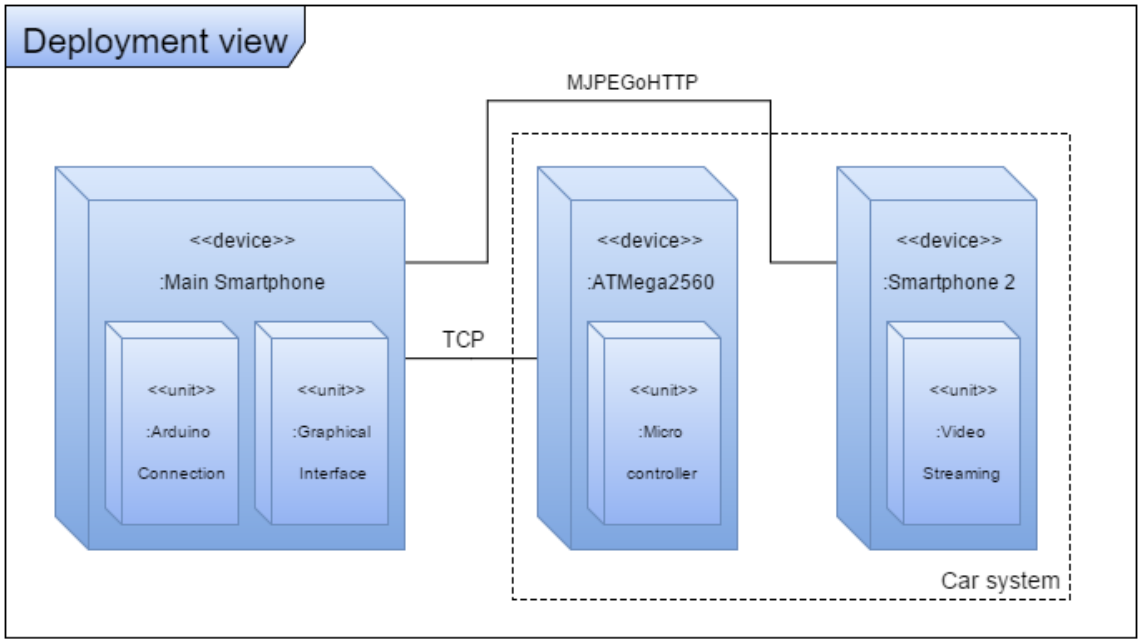


Figure 6: Deployment view

- **Main Smartphone:** This device is the one that the user handles to interact with the system. We have two units in it: “Arduino Connection” in charge of dealing with the Arduino (ATMega2560) and “Graphical Interface” which between other tasks handles the video stream.

- **ATMega2560:** The microcontroller in the Arduino, this is the main part of our car’s system. We have attached a Wireless LAN board to connect via TCP to the Main Smartphone.

- **Smartphone 2:** This device is used solely to server an MJPEG stream that the Main Smartphone will use. Connected via WIFI to the same network as the Main Smartphone.

For more details you can see the document ***SW_Architecture_Design.pdf*** where you will find all the information about the software development.

4.2 Connections

4.2.1 BDD system

Here we can see the Block Definition Diagram (BDD) of all the project, with the main external parts involved.

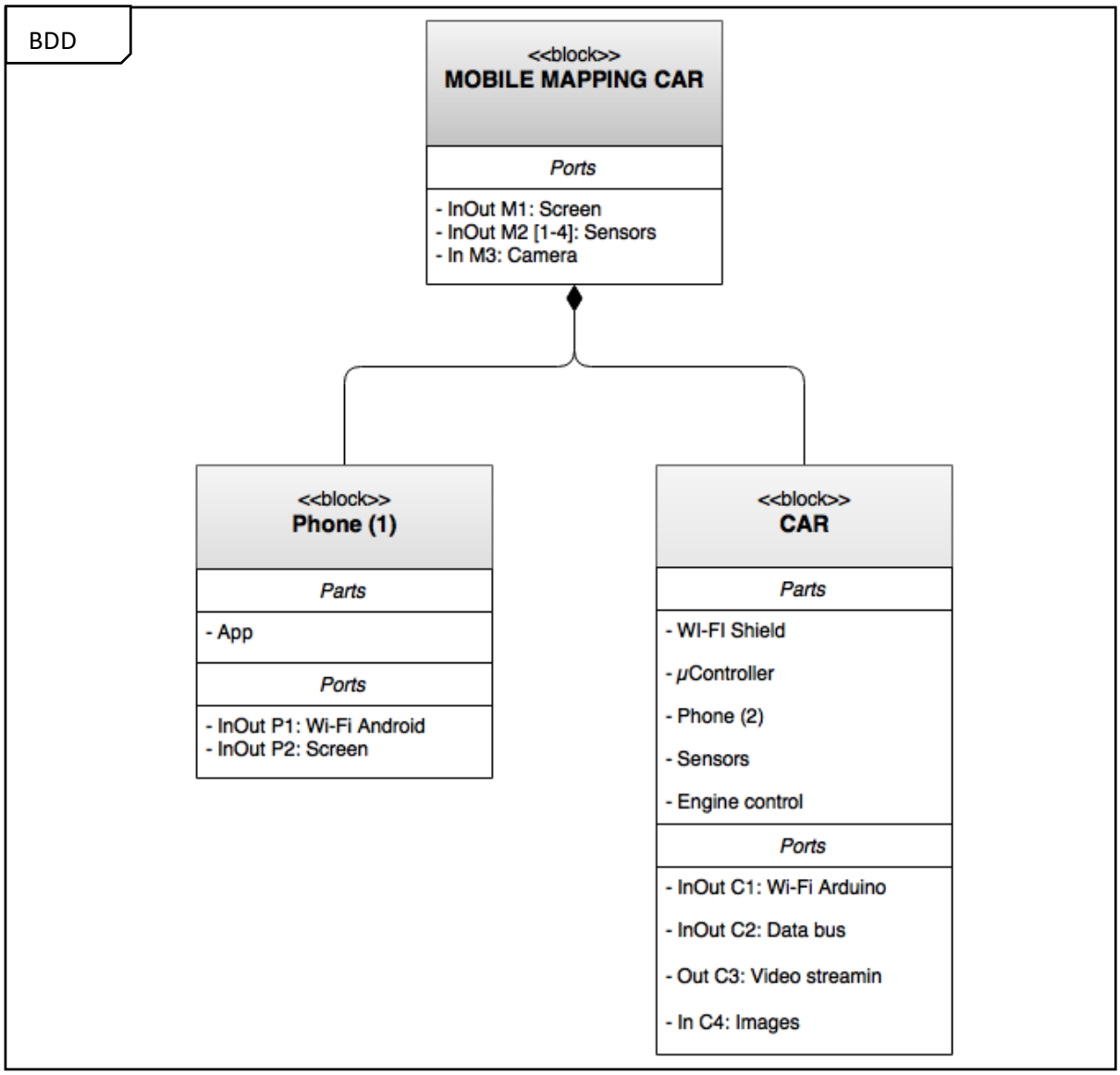


Figure 7: BDD system

4.2.2 IBD system

In this diagram we can see all the connections between the principal devices of our project. The arrows indicate the direction of each port.

For more explanations about the connections and hardware design you can see the document named ***HW_Architecture_Design.pdf***

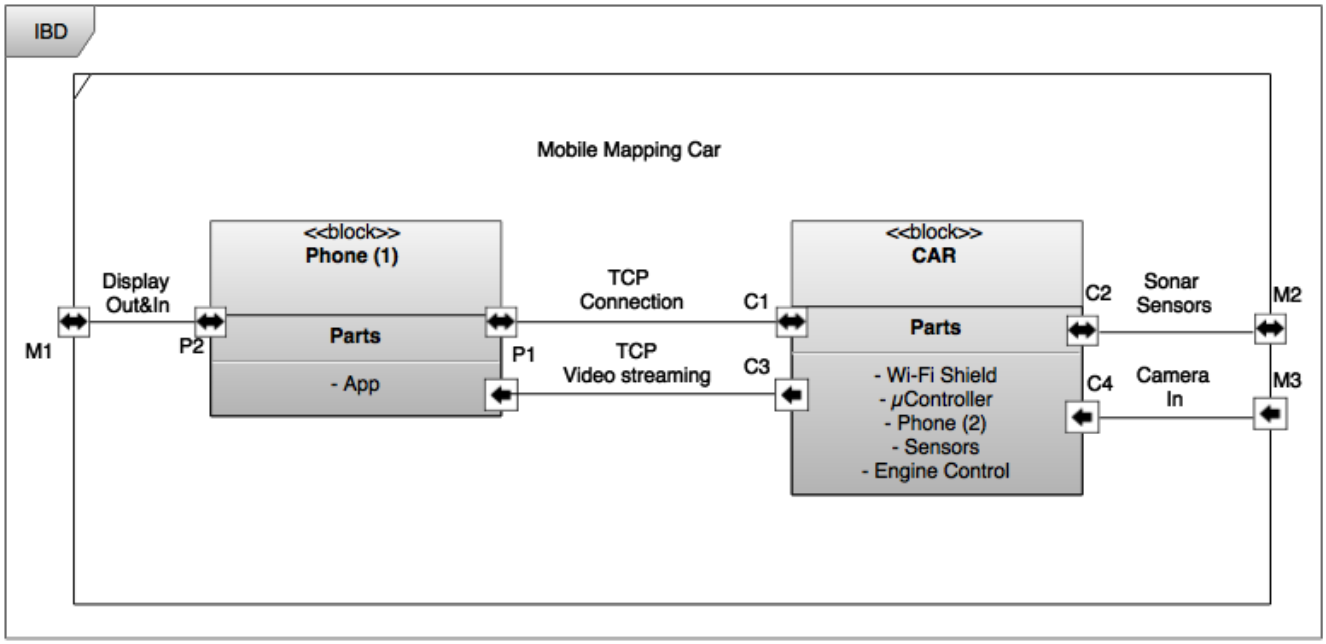


Figure 8: IBD system

4.3 Implementation

Talking about the implementation, we will split the process into two parts: HW & SW Implementation.

4.3.1 Software implementation

In this section, we focus on the method used to get x and y coordinate out of the sensors measures.

We initially planned to use six static sensors which will take measure while the car is moving. The microcontroller would then send the data to the phone in value pair {distanceMeasure/sensorsNumber} given that each sensor had been assigned a number.

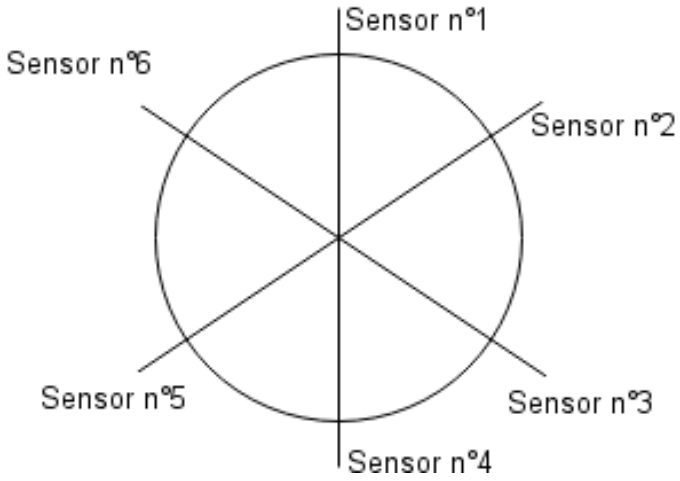


Figure 9: Sensor layout, prototype one

We could then treat each measurement depending on the sensor number and calculate the x and y coordinate using trigonometry. So we had six different cases with using similar trigonometric formulas.

But later in the project development, we change the way we retrieved data from a moving car with static sensors to a static car with moving sensors.

We change the numbers of sensors from six to four. The sensors were put on a moving platform which moved 10° between each measurement. That would have given a lot more case to treat with the previous calculation method. So we decided to use {distanceMeasure/CurrentAngle} value pair.

That way, we only have to implement two trigonometric formulas: one for the x coordinate and the other for the y coordinate. With only these two, we can calculate any coordinate with the distance and the angle. Of course we had to add code to treat some specials cases, when the sensors give absurd distance for example.

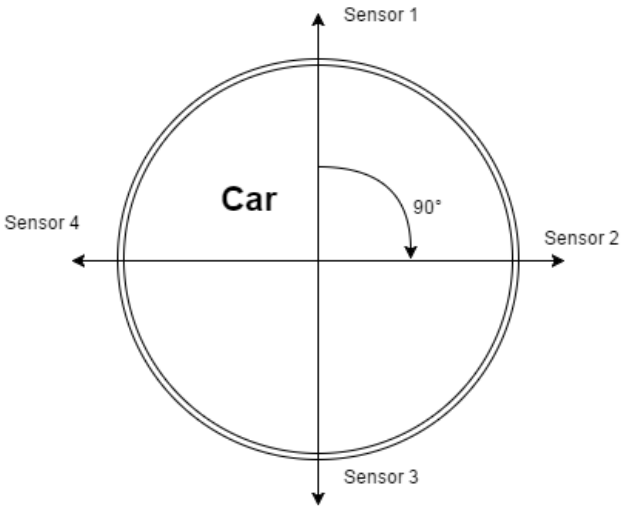


Figure 10: Sensors layout, prototype two

4.3.2 Hardware implementation

4.3.2.1 Microcontroller

The first part of implementation is focused in ARDUINO ATMEGA2560. Thanks to this micro-controller and a Wi-Fi Shield we are able to execute the communications between the main phone and the car.

4.3.2.2 Structure

We made plastic structure to mount our sensors. This structure has four long legs enough to avoid the cables and the engine circuits. In the top we can see al hole to put the servomotor, this hole is exactly of the servo dimensions to make it easy the mounting.

The last part of the structure is a simple square mounting with the servomotor. This structure spin the sensors to obtain points from all directions.

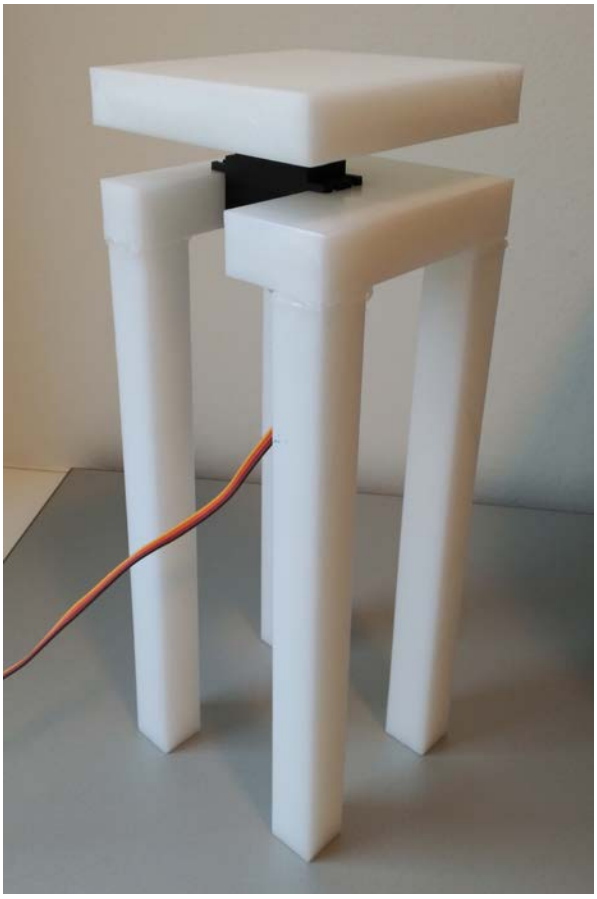


Figure 11: Sensors structure

4.3.2.3 Ultrasonic sensors

The important hardware part in this project are the ultrasonic sensors. This components were difficult to implement due the synchronisation between them.

To obtain the value in centimetre we have to divide the obtained value (in microseconds) by 58,31. This value is obtained as seen above, using the standard sound velocity in the air (343 m/s) and conversion factors to obtain the desired units.

$$Distance\ (cm) = \frac{high\ level\ time\ (\mu s) \cdot velocity\ \left(343\frac{m}{s}\right)}{2} \cdot \frac{1\ s}{1000000\ \mu s} \cdot \frac{100\ cm}{1\ m}$$

$$Distance\ (cm) = \frac{high\ level\ time\ (\mu s)}{58{,}31\ (\frac{\mu s}{cm})}$$

For more information about the sensor and Signals operation you can see the documents ***HW_Architecture_Design.pdf*** for the specification in the physical parts and the ***SW_Architecture_Designs.pdf*** for the code implementation.

4.3.2.4 Servomotor

Related to the servomotor movement, we used one library included in the Arduino IDE. With this library we only have to use one function where we put the angle where we have to move the servomotor.

For more information about the implementation codes of the servomotor you can see the document ***SW_Architecture_Design.pdf***, for more information about the connections and the circuit power you can see the document ***HW_Architecture_Design.pdf***

5 Test

5.1 Unit test

5.1.1 Car movements

In the beginning of the project, we started creating a little code with simple movements (forward, backward and turns). With this test we could check the control engines operation.

5.1.2 Sensors behaviour

Before use a lot of sensors in our project, we programed one code to obtain measurements from a sensor and show the data through the serial window in Arduino IDE.

This test was very useful to check the behaviour of the sensors.

5.1.3 Servomotor behaviour

In the final part of our project we decided to include a servomotor for the sensor movements, we used a library included in the development tool Arduino IDE. This test was easy to do and we implemented the servomotor very fast with other functionalities and devices.

5.1.4 Video streaming

To find out the best way to handle the MJPEG stream, we created a new project in Android Studio where test all the plausible solutions. Once we find the right one, it was just a matter of porting the solution to the main Android Studio Project

5.1.5 Android-Arduino connection

For this purpose we used a new aside project in Android Studio. We loaded a simple program in the Arduino board for just accepting a connection and sending a character. This test took so much time due to the lack of a decent and controlled network (We were using AU-Gadget or mobile hotspots), sometimes the devices weren't reachable from each other.

5.1.6 Map plotting

Once we had the connection working, we needed to check if the map plotting was working properly. We loaded some sets of points on an Arduino sketch as float arrays to simulate the sensor reading task. The Smartphone would ask for new sets of points and instead of reading, the Arduino would just send this arrays in order.

5.2 Integration test

5.2.1 Four sensors with Servomotor

Progressing in the project, we joined all the external microcontroller devices. We created a global variable to decide which sensor was measuring every time. After the four sensors have obtained a measure, the platform where the sensors where mounted rotates 10° by using the servomotor.

5.2.2 Map plotting/data retrieval while steering

Since steering after retrieving data might lead to some trouble, we decided to test this outside the main project. We created a new sketch for the Arduino board and tried, successfully, to retrieve a set of points and steer afterwards.

5.2.3 Networking while operating the servo

While doing the final tests, we noticed that the Wi-Fi shield just randomly shut off (Or that's what we thought). We decided to try the final code commenting out some functionalities and parts of the code. It ended up being the servo causing current spikes that the Wi-Fi shield just couldn't handle. The solution was to create a power supply only for the servo:

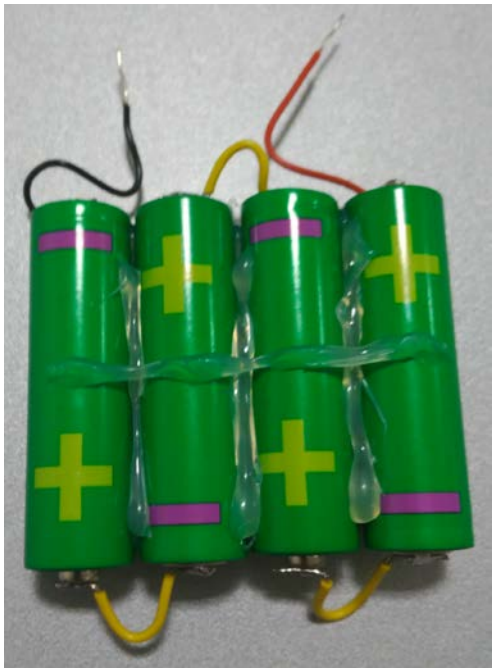


Figure 12: Servo battery

5.3 Acceptance test

Here we have a little view of the realized tests in our project. We have to say that all the tests have been passed correctly as we can see, with more detail, in the acceptance test document.

Test name	*Systems involved*	*Description*
Input connection data	User's smartphone	The purpose of this test is to input the necessary connections information in the user's smartphone**.**
Connect to microcontroller	User's smartphone Microcontroller	The purpose of this test is to connect the user's smartphone with the car and the car's smartphone.
Stream video	User's smartphone Car's smartphone	The purpose of this test is to stream a video from the car's smartphone to the user's smartphone.
Navigate car	User's smartphone Microcontroller	The purpose of this test is to navigate the car, making it move in the four directions available.
Record room	User's smartphone Microcontroller Ultrasonic sensors	The purpose of this test is to collect data from a room then draw and save a map of the room.
Manage recorded maps	User's smartphone	The purpose of this test is to access to a map in the list of saved map in the user's smartphone and display it in full screen.
Delete map	User's smartphone	The purpose of this test is to access to a map in the list of saved map in the user's smartphone and delete it.
Small objects detection	Microcontroller Ultrasonic sensors	The purpose of this test is to assure that the ultrasonic sensors can detect small objects as table or chair legs.

For more details about this tests you can check the document named ***Acceptance_test.pdf***

6 Results

In this part of the project we are going to explain our results, comment how we got them and finally discuss them.

6.1 Overall

If we make a global evaluation of the results obtained we can say that we are satisfied because the main goals of our project are accomplished.

We have an app to receive video streaming, control the car, receive points, plot them and save the plots obtained.

The devices are connected and communicate to each other appropriately.

The microcontroller controls the signals to move the engines, triggers the sensors, receives data from them and sends it.

6.2 Communication

Focusing on the communication between the devices, as we already said, we have managed to set a good communication between them.

In the beginning, we had to choose how we were going to make the connection. We decided to implement a TCP connection through Wi-Fi (further explanations in ***Preanalysis_choice_of components.pdf***).

We think that we made the best possible choices for the communication in our project. However, if we had more time, we could have implemented a watchdog to make sure that, if there is any issue with the connection, we could guarantee that it wouldn't move.

6.3 Video Streaming

The video streaming has been a challenge for us since the beginning. We first tried to use a camera attached to the Arduino board and stream the video through it, the best solutions we could find for this setup were send 10-15 compressed pics per sec and it wasn't enough.

Then we saw the only solution was to stream video from another device directly to the main smartphone, we tried several solutions. We decided to user an IP Camera (or a smartphone app for that purpose), so we had to find a way to handle a MJPEG stream. Android doesn't support MJPEG streams natively and handling the stream ourselves was too much work, using an external library seemed like the right solution. Wrong, the documentation for the ones we found was poor and outdated. Finally we found that using a browser-like object (WebView) was the best option, and so we used.

After this we only had to configure the WebView to display the stream as we wanted.

6.4 Getting information about the room

In the beginning, we had to decide how we would be getting the information about the car's surroundings so that we could plot it afterwards. Our choice was using ultrasonic sensors (further explanations in ***Preanalysis_choice_of_components.pdf***).

Once we had decided which sensors we were going to use, we wanted to be able to collect data with the ultrasonic sensors while the car was moving. To do so we needed to be tracking the position of our car in real time. The way that seemed the most effective one was by using the accelerometer of the Smartphone streaming the video. However, after having searched for information and tried to implement it, we saw that it was really difficult and we decided to change our minds. Our solution was receiving the data while the car was stopped instead of while moving (further explanations in ***HW_Architecture_Design.pdf***).

We decided to send the distance given by each sensor and angle that it is positioned to the Smartphone and then it makes all of the computations and plotting. We implemented it this way because the Smartphone has more processing power and we found it adequate.

The solution implemented in our system seems to be working properly and it doesn't give us many errors.

We have to say that, even though we can correctly get data from all around the car now, depending on where the car stops, it can have blind spots while measuring, so it will plot wrongly the layout of the room. However, we think that apart from this, we can get good results and maybe they are more precise than making the measures while moving because we are also avoiding some possible noise made by the engines.

We think that, if there are not many tables and chairs and small obstacles around and ignoring the blind spots, we can get a good approximation of the layout of the room.

6.5 Microcontroller

When we had to choose a microcontroller for the car, we decided to use an Arduino Mega 2560 with a Wi-Fi shield (further information in ***Preanalysis_choice_of_components.pdf***).

We wanted to use the timers in the microcontroller to control its program and synchronize the connection between the devices and the data receiving and sending.

Then we used some external interrupts for measuring data with the ultrasonic sensors. We also used a library included in the Arduino software to use the servo.

When we knew how to use the different parts described and we merged the different codes we got many errors and had to implement changes in the microcontroller code. For example, we couldn't use the timer we were using because the servo library was using it and we had to change it and use an 8-bit timer instead of a 16-bit one; we had to get another exclusive power supply for the servo because it was causing problems while using it together with the sensors. Finally we changed many things of the code and we didn't use timers nor external interrupts. However, we still used the servo library.

After making the project, we think that maybe choosing a more complex microcontroller because we had some issues with it and, as it has not a very good debugger, we had difficulties at discovering which the different problems were. Maybe our final code could be more organized but once we managed to make it work properly we didn't want to change anything.

7 Conclusion

Looking at the results obtained in this project we can conclude that we are clearly satisfied because, as it can be seen, we have completed successfully all of the main project goals.

As in almost every project, we could have made some improvements, and we have explained some of the most significant ones that we thought about.

We all agree that this experience has been fulfilling for us as, in our home countries, it is not very usual to make supervised projects similar than these. It will also be helpful both for when we have to do our bachelor project and for the working world, as most of the jobs in the engineering field imply working in teams.

Although three of the members of our group are from Spain, we have also learnt to collaborate with people from other countries by using English as the language to communicate between us and by learning from each other's cultures.

It has been a useful way to put into practice the knowledge obtained in the subjects we have been studying here during our exchange program in the first semester. It has also been a good way to make us search for information and learn new skills by ourselves.

We have also had to manage different problems we have been facing during the development of the project. This has made us more confident about ourselves as we have solved most of them without getting much help so, we see that we can be capable of dealing with problems without external help.

Finally, we can say that apart from being proud of our project, it has been a new experience that has provided us with knowledge and different skills that will probably be useful in our future.

8 References

[1] Android. (2015). Android Developers website, [Online]. Available: <http://developer.android.com/index.html>

[2] Vogella. (2015). Android tutorials, [Online]. Available: <http://www.vogella.com/tutorials/android.html>

[3] Arduino. (2015). Arduino tutorials and program tool, Avaliable: <https://www.arduino.cc/>

[4] Stack Exchange. (2015). Information about tracking 3D position with an accelerometer, [Online]. Available: <http://electronics.stackexchange.com/questions/97656/track-3d-position-with-an-accelerometer>

[5] Stack Overflow. (2015). Infomation about an indoor positioning system based on a gyroscope and an accelerometer, [Online]. Available: <http://stackoverflow.com/questions/7499959/indoor-positioning-system-based-on-gyroscope-and-accelerometer>

9 List of figures