HIDU: Home Intruder Detection Unit

Tiago Carneiro Fernandes

BSc (Hons) Computer Science

Supervisor: Dr. David McLean

DECLARATION

No part of this project has been submitted in support of an application for any other degree or qualification at this or any other institute of learning. Apart from those parts of the project containing citations to the work of others, this project is my own unaided work.

SIGNED: __Tiago Carneiro Fernandes___

Contents

DECLARATION ii
Table of Figures v
Abstract 1
1. Introduction 2
1.1 Background 2
1.2 Aim of Research 2
1.3 Objectives 2
1.4 Methodology and Program of Study 3
1.5 Learning Outcomes 3
2. Chapter 2: Literature Review 4
2.1 Background 4
2.2 Face detection 4
2.3 Model-based Face tracking 4
2.4 Cascades 4
2.5 Material 6
3. Robot Design 8
3.1 Components 8
3.2 Overall Design 8
3.3 Advantages and Disadvantages 8
3.3.1 Advantages: 8
3.3.2 Disadvantages: 8
3.4Design Conclusion 9
3.5 Redesign 9
4. First Steps 11
4.1 Testing Software 11
4.2 Movement 11
4.3 Light Sensor 11
4.4 Evaluating 12
4.5 Improve design 12
5. Programming 13
5.1 Java Programming 13

5.2 OpenCV 14
5.3 LeJOS Library 14
5.4 Functionality 14
5.5 Movement 14
5.6 Face Detection 16
5.6.1 Image Processing 17
5.6.2 Fine Changes 18
5.7 Robot to user Emailing function 19
5.8 Functionality Testing 21
6. Conclusion 22
Objectives achieved 22
Learning Outcomes 22
Proposed improvements 22
Overview 22
References 23
Bibliography 26
Appendices 1
Appendix A – Terms of Reference 1
Appendix B – Ethics Form 3

Table of Figures

Figure 1.1 3
Figure 2.1 5
Figure 2.2 6
Figure 3.1 8
Figure 3.2 9
Figure 4.1 12
Figure 4.2 12
Figure 5.1 13
Figure 5.2 15
Figure 5.3 16
Figure 5.4 16
Figure 5.5 17
Figure 5.6 18
Figure 5.7 19
Figure 5.8 20
Figure 5.9 20

Abstract

This project approached the problem of home safety by proposing a robot prototype that will serve as a patrolling unit and the report will be taking a Prototyping model approach to its research and development. This project will use knowledge from robotics, artificial intelligence, Design, functionality programming, machine learning, face detection and it will use a test and evaluation methodology adequate to a prototype project.

Showing promising results, the final prototype achieved the proposed objectives and extended further, this a model can be improved for a better performance and versatility and become a useful tool for home safety and a successful product in the home security industry.

1. Introduction

1.1 Background

For recent years home security has become more and more important and one of the fields where many homeowners decide to invest to keep their families and their properties safe. Unlike a few years ago we now have systems that allow us to record and move cameras, connect them to 24/7 surveillance systems and even detect movement. Recently this field has had some special attention and one of the most advanced ones would probably be the "iCamPro FHD" [1] which was the award winner for 2015 CES Best of Innovation [2] for his amazing ability to hear, see and track movement. Being called the first intelligent home security robot which made me research about other Home Security systems starting with GSM (Global System for Mobile Communication) based Systems [3] and about its design and implementation [4].

All these innovations aim to make security flawless and not human dependent, meaning that there is less flexibility for human error, it also tries to achieve an automatic response to different situations.

However, what I will try to do in this project is create a robot that will allow a home owner to order a robot to patrol the house and detect intruders while patrolling. In order to achieve this I will do intensive research in the fields of robotics in general, machine learning, image processing [5] , light sensor [6], face detection, programming and Human Computer Interaction.

By applying all that knowledge, I hope to obtain a versatile and reliable robot that will be rigorous in term of safety and error free to achieve a safety feel to the home/robot owner.

1.2 Aim of Research

This project is to aim a patrolling unit ranging functionalities like face detection, machine learning and path following,

1.3 Objectives

1. Study techniques for robot movement, map awareness, image processing, machine learning and robot/user connection.

2. Build a stable and reliable hardware (robot).

3. Design and Implement a path patrolling system.

4. Design and Implement machine learning techniques.

5. Design and Implement movement track/detection techniques.

6. Design and Implement User/Robot connection.

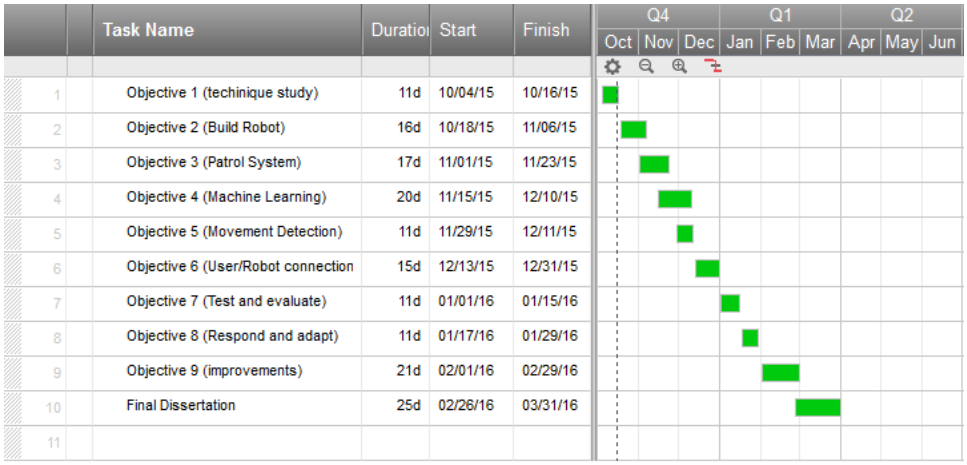
7. Test each implementation thoroughly.

8. Intensive testing for different situations and responses.

9. Improve system to an advanced level.

1.4 Methodology and Program of Study

Bellow you can find the methodology schedule I aimed to achieve from the beginning of this project [Figure 1]:



	Task Name	Duration	Start	Finish
1	Objective 1 (techinique study)	11d	10/04/15	10/16/15
2	Objective 2 (Build Robot)	16d	10/18/15	11/06/15
3	Objective 3 (Patrol System)	17d	11/01/15	11/23/15
4	Objective 4 (Machine Learning)	20d	11/15/15	12/10/15
5	Objective 5 (Movement Detection)	11d	11/29/15	12/11/15
6	Objective 6 (User/Robot connection	15d	12/13/15	12/31/15
7	Objective 7 (Test and evaluate)	11d	01/01/16	01/15/16
8	Objective 8 (Respond and adapt)	11d	01/17/16	01/29/16
9	Objective 9 (improvements)	21d	02/01/16	02/29/16
10	Final Dissertation	25d	02/26/16	03/31/16
11				

Figure 1.1

1.5 Learning Outcomes

At the end of this project, I am expecting to have the following learning outcomes:

- Improve organization skills.
- Improve planning skills.
- Gain project management skills.
- Gain knowledge in Robotics.
- Gain knowledge in Artificial Intelligence.
- Improve my programming skills.
- Be able to set bigger challenges in the near future.

2. Chapter 2: Literature Review

2.1 Background

With technology exponentially growing, in the last couple of years many new robots have been created tackling the security industry, robots like KNIGHTSCOPE [7] can now move around an area and detect intruders or anomalies. KNIGHTSCOPE is made specially for the commercial industry, made to replace the security guards that protect shops and shopping centers, however home systems have also been growing in this industry, robots like the iCamPro are becoming more popular among homeowners since they can constantly protect their house by detecting intruders and warning them or even the authorities if necessary. These make us feel more secure and dream about more futuristic and efficient security systems. My aim for this project is to create a mixture between these two robots and create a unit that can move around and detect intruders, like the KNIGHSCOPE as well as having the size of the iCamPro, to create a patrolling unit for a homeowner.

For a project of this scale I had to do a lot of research and I found a great deal of information, but being a very recent version of the Lego robotics some of the information is not reliable but it can be used as a discussion of possible ways, so even with such information a lot of testing was needed to actually see if it worked properly and if not what the problem was and decide what approach to take to improve it further. I will talk about some of the more specific research as we go through the steps I took to build this robot.

2.2 Face detection

Detecting faces has been and still is a very useful functionality that has been improving rapidly, after searching for different methods and techniques I found 2 of them to be the most adequate for this project: Model-based Face Tracking and Cascade Classifiers.

2.3 Model-based Face tracking

Face detection can be achieve through a wide range of techniques. Model-based Face Tracking, despite not being as accurate as others, is a very interesting technique; In many ways this method resembles the way that us, humans, perceive objects. Since a very young age we learn how to identify object, however, having 2 eyes and a body really helps, not only because we can touch certain object but also because we move and have a 3 dimensional perception of the space around us and because of it we can distinguish things like small animals from just simple dust, one of the main reasons for this is motion, when we see a small black dot moving we instantly perceive it as some sort of animal, maybe fly, or a small spider, without this motion it would be lot harder for us to tell with certainty if it is bug or just a piece of rock, or dirt. This method takes this technique and applies it to face detection, knowing how a face should move it can detect a face when it moves, because it expects a face to move a certain way, using training models from different angles it knows that when we look right, certain parts of our face will have a certain motion relative to each other. However, this method has some bugs and problems and it struggles to detect a wide range of different faces, nevertheless it is an interesting approach to the problem.

2.4 Cascades

The one I find to be the most revolutionary evolution in this field was made by Paul Viola and Michael Jones[8] and submitted in 2001. I managed to find their paper on this which I found to be very

informative and interesting, it got me to understand the face detection problem much deeper and what their methodology was. Their approach managed to detect faces from a live video stream and in a very efficient and accurate way, this detection system became about 15 times faster than any other approach worked on before which by itself is a big step forward. Using Haar-like features combined with an enormous dataset with a very wide range of conditions Viola and Jones reduced the processing time and improved the accuracy of this computer vision problem. Being now in 2016 this method is still considered a very accurate one; therefore, I decided this would be the best path to take.

To achieve, I found that cascade classifiers would be my best option, not only because it is quite efficient but because it would take less processing from the computer which was key for this project. To be able to process camera input I need to process every single frame at some level, this means that the more processing is required more time it will take to process each frame and the more time it takes to process one frame the less frames per second we can achieve, so I was really looking for a balance between efficiency and performance but before going further we need to understand what cascades are.

Haar feature-based cascades is an effective method of face detection proposed by Paul Viola and Michael Jones in one of their paper [8] these cascades are used to train a dataset, they pick up changes between light and dark within a face, for example, the tip of your nose will usually be lighter than the region right below your nose [Figure 2].

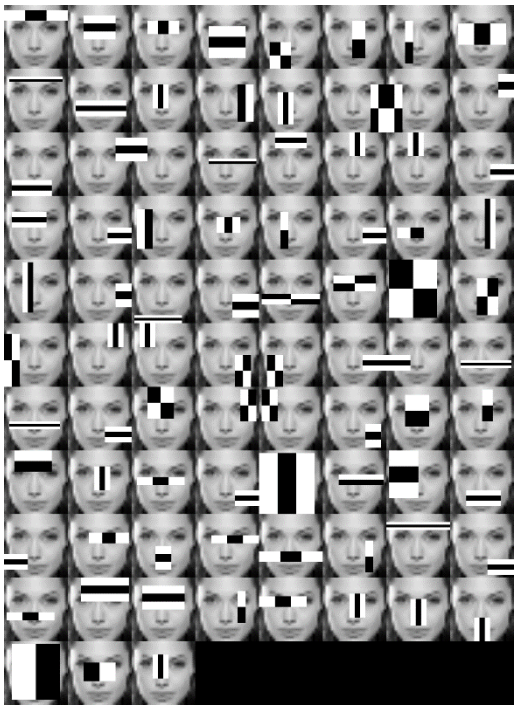


Figure 2.1

You can think of a face as set of small 3x3 tables, where each cell contains the brightness of a particular pixel, being zero a pure black and 255 a pure white, if we take a random sample of an image it will look something like this [Figure 3]:

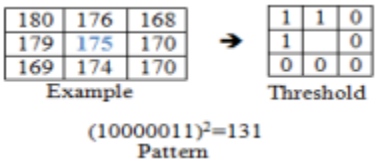


Figure 2.2

By having, in this case, a 3x3-structure element with thresholds in each cell we can detect edges in the sample table, if we repeat this process through the whole image we will be able to pick up the edges of that image.

Different faces will train this data so it can detect which features are important and relevant to distinguish a face and which ones are not. I find it to be a very good method and it does work very well as long as the face is not tilted too much. The tilted head problem is a quite popular one among the face detection "community", it is hard to detect a face that is tilted simply because all of the training was done using a certain orientation and to apply that training to a tilted head would take a lot of processing because it would require the program to approach different angles, it would have to apply the same techniques a few times more to cover all possible angles, this would increase the detection time 6 or 7 times more than with a non-tilted head (it still has a margin of about 20 degrees tilt each side).

2.5 Material

For this project I'm using a EV3 Lego Robotics Kit [9], this kit contains a couple of wheel motors, an extra frontal motor, a light sensor with different modes, an infrared sensor with different modes as well as a touch sensor.

At this stage I would like to mention that although I feel my vision for this project is something that could be very practical and innovative, I am quite limited by the hardware being used, it not only lacks the necessary processing power. Being a Lego kit it lacks stability and precision that would improve its performance and accuracy, meaning that with better hardware it can be further improved.

This kit is loaned from Manchester Metropolitan University using the equipment loan services available, this specific version is called Lego Kit Ev3 Mindstorms and it is a very recent version (2015) which requires much more testing but it does allow me to create something never done before.

The kit also brings a software development kit at the homepage website which is a drag and drop software for basic programming of the robot.

Further, in the development of the robot I also bought a camera specifically for this robot on Amazon for the development of the face detection functionality. For the path following functionality, I used colored and opaque duct tapes to allow the use of the light sensor

This robot uses batteries to power itself, 6 batteries to be more specific and when using it the batteries drain rapidly so I always had extra batteries every month or so.

Another necessary component was the USB cable that comes with the EV3, this connect the EV3 brick to the pc to execute or transfer programs. To run the robot the USB cable is required to be connected to the computer, however, just like the camera, it can be replaced with Bluetooth hardware.

3. Robot Design

3.1 Components

To design this robot I needed the following components: 2 wheel motors for the movement functionality, 1 light sensor to detect the different colored tapes for the path following functionality, a front motor to move the camera left and right and a lot of Legos to build a stable structure to connect all of them together.

3.2 Overall Design

The first step I decided to take was to build an initial design for the robot; something I could test and see what changes I would need to make to get closer to my vision. Therefore, I used a design from the Lego manual [10], a design called R3PTAR [Figure 4] and its design resembles to a snake.

Figure 3.1

Using the sample programs that come with the EV3 brick, just by pressing a button I could see the robot do some random movements and evaluate if I was pleased with the way it moved.

3.3 Advantages and Disadvantages

3.3.1 Advantages:

- A thin design, which would not take too much horizontal space.
- The camera could be placed in a high-level position, closest to a human face.

3.3.2 Disadvantages:

- The camera despite being in a good position was very unstable when moving.
- The wheel movement was very inaccurate and turns had to be very wide since the design couldn't perform sharp turns.
- The wheels had bad friction values, meaning the wheel would sometimes spin free from the ground, which would become a problem when trying to create precise movement patterns.

- It was also unable to overcome small obstacles like unaligned floor wood planks, carpets or cables on the floor.

3.4Design Conclusion

Overall, I found this to be a very bad design in terms of performance, it had to move very slowly for stability and it was very hard to turn and calculate movements, so another approach had to take place.

For the traction issue I decided I had to use tracks, perfect for any house floor, wood or carpet and it would be able to overcome medium obstacles, like floor planks or transition from floor to carpet.

For the camera stability problem I decided that I would need to give up the high level position for a better stability, it would have a worse angle of vision but better stability while moving, meaning more accuracy and it would also have a dedicated motor for the side movement of the camera.

It would also have independent motors for the right and left tracks, this would allow me to do any turn, wide or sharp, much like a war tank and it would allow me to precisely calculate and command the movement I needed.

The light sensor would be more or less in the same position, in front with a little extension made of Legos.

3.5 Redesign

Taking the advantages and disadvantages, I created a completely new design, starting with a base for the EV3 brick I found a similar version of what I envisioned on the Lego manual. I used the first steps of a design called TRACK3ER [11] [Figure 5] (brick base and wheels position).

Figure 3.2

Having the base for the brick and the tracks I designed a motor connected straight to the camera this motor would be pointed upwards connected directly to the camera so it could move left and right. Attached to the brick base would be the light sensor so it could detect the path ahead of the wheel, this would allow the robot to adjust the movement to follow the path much smoother then if the light sensor was closest to itself.

4. First Steps

4.1 Testing Software

For the early testing I used the Icon-Based Software[12] provided by Lego, this allowed me to drag and drop component methods and very quickly test some of the movement, however it does have some limitations and so this software was only looked at during the early testing

So I created a simple program to make the robot move in different directions of my choice to visually test the way it moved. I then constructed a small program using the light sensor to try the different sensor modes(ambient, RGB, red, color) and decide what the best solution would be.

4.2 Movement

When testing the movement with the Lego Software, my aim was to understand what speed I wanted to achieve and how the robot would turn. Giving priority to stability over speed, the robot would move slowly, so that when a turn was required the impact would not be as severe but instead have a smoother turn.

The stability of the robot was one of the most important aspects to look at since the camera that would go on top would be trying to detect faces while following the path and having a very "shaky" video stream would not help the detection process.

4.3 Light Sensor

Moving on to the light sensor testing I tried the different modes provided by it [13] and tested them. Among the 4 different modes I found that the "red" mode was the one that proved to be the most useful, this mode would strike red light in a surface and measure the intensity, ranging from 0 to 1, of the red color values reflected, meaning that a color (RGB) that contains high levels of red values would return higher values than a color with low values of red. Almost immediately I used some colored tapes to test this, and as expected, when using the sensor pointed to a black colored tape, the reflected value was around 0.05 and when pointed to a red tape it would return around 0.85, a much higher value. However when I tested a white tape the returned value seemed to be slightly higher than the red tape.

After some thinking I realized why this might be, the red tape, despite being red, it was very likely not purely red, meaning that it can have a range of different mixtures of red, green and blue that when together create the red color we can see, however, that does not mean that it contains a value of 0 for green and blue and a value of 255 for red, it can have a much lower value of red and still have a red color.

When we consider pure white, in theory, it is a mixture of blue, green and red at their maximum value which has can easily have a much higher value of red and it also reflects any color of light much better without getting absorbed. The white color seemed a much better idea than the red one, since it would allow me to have a bigger range to program but it would also help the robot to work in dark environments, since the white would reflect color much easier than the red.

4.4 Evaluating

Considering the movement and sensor testing I felt I was ready to start the programming process and applying the functionalities, I was happy with the performance, the only issue found at this point was the camera stability, which could still be further improved if the center of gravity of the camera structure was lower which would also allow me to build a more stable structure with more holding points.

4.5 Improve design

For further improvement I redesigned the camera structure again and made it into a lower center of gravity, instead of having the motor pointing upwards it would be pointing to the front of the robot and it would be positioned right on top of the brick, this motor would be attached to a dented wheel that would rotate another dented wheel attached to the camera, this would allow the camera to move left and right even though the motor itself was horizontal, it would replicate the motor movements but the dented wheel would take this movement to another axis [Figure 6].

Figure 4.1

The sensor had to be placed on a new position and at about 1-2 centimeters from the floor for more accurate values this small change had a big impact on the sensor performance, the values were much more reliable now.

Below is a set of pictures of the final design with a frontal, lateral and Top view [Figure 7].

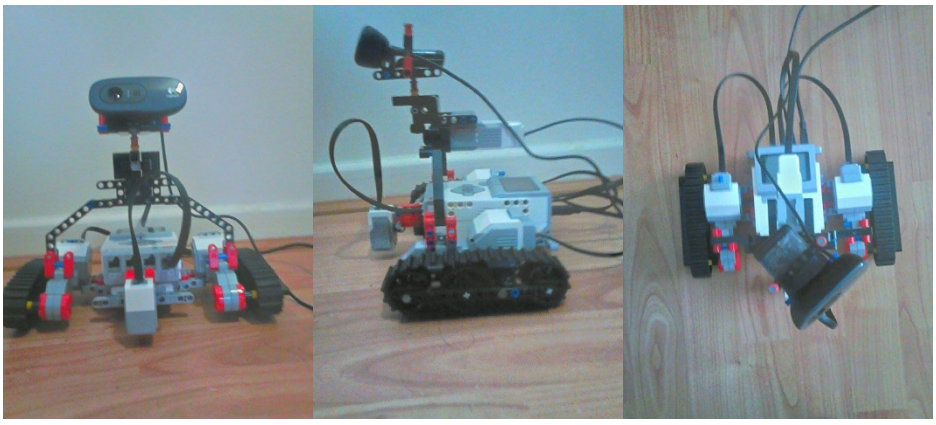


Figure 4.2

5. Programming

5.1 Java Programming

The drag and drop software was good enough to test the robot but it didn't get me full control of the brick, so I researched about programming languages for the brick, after some research I found C# and java to be the most appropriate languages but because I personally feel more confident with Java I researched on how to take advantage of this language with the robot [14].

To program I used Eclipse[15], which is an open source community with very useful tools and documentation to allow me to take full advantage of the language.

After some thinking I built in my head the general structure of what the program would be which I then transferred to the paper [Figure 8].

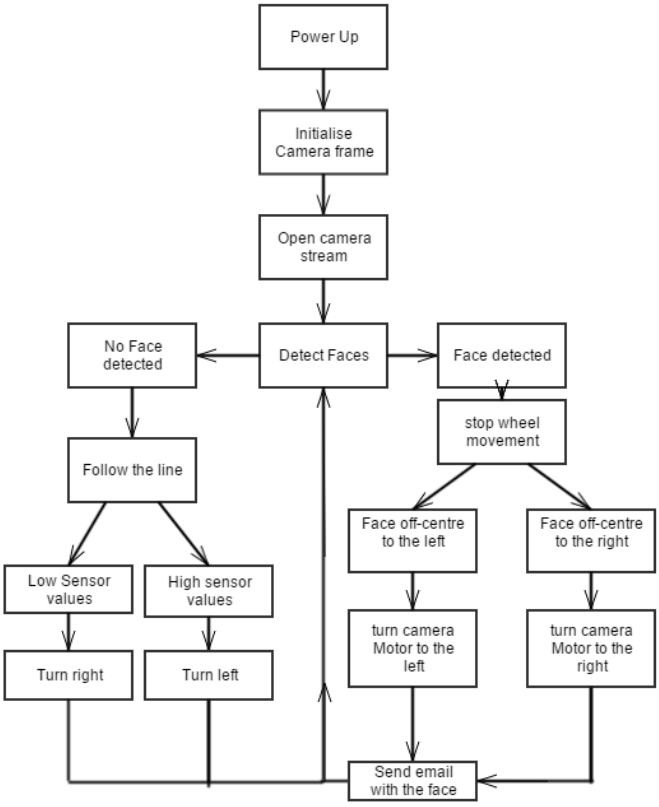


Figure 5.1

So the aim was, instead of programming for the EV3 itself, I would program a Java Application, granting me more processing power since it would run on the pc and not the EV3 brick and by being connected to the EV3 through a USB cable I would send the necessary commands for it's movement. would be made through Bluetooth.

5.2 OpenCV

To allow face detection on my program I researched on ways to implement this with Java using Eclipse, what I found was openCV[16], this library has a Java interface and it is free for both academic and commercial use, this library is made specially for real-time application, it is an amazing library with a wide range of options and from where I worked with most of my program. Their API documents also provide a brief introduction to the Harr feature-based Cascade Classifiers [17], which is exactly the method proposed by Viola and Jones and just what I was looking for.

This library, which has been having frequent updates, contains very useful methods from image processing to object detection. It's main focus is real-time applications like the one I aim to create and although it is written in C/C++ it has interfaces for other languages, like Java, which is referenced as JavaCV[18].

5.3 LeJOS Library

Now knowing the language I wanted to program in, I needed to make sure the robot could be programmed in Java, which required me to download the LeJOS [19] Library which is a very small Java Virtual Machine that can be implemented in the EV3 brick which would receive commands from the Java application and its API documents [20] contain a lot of useful information for this project.

5.4 Functionality

The functionality of the robot is the most important part of the project, having in mind the aims and objectives of this project I decided that it would be better to start with the robot movement, not only because it seemed like the simplest so that I could get used to the robot programming but also because the movement had to be dealt with first, before anything else, since this would influence the camera and the light sensor.

5.5 Movement

After researching the Lejos API I found a lot of useful methods regarding the EV3 motors, like the “DifferentialPilot”[21] methods, which allows the user to create a base chassis, given certain measurements this allows the user to simply import the distance and the rotation angle the robot should travel and it would automatically deal with the wheel speed to achieve this, however I feel very confident about controlling individual motors, it gives me more freedom and this way I can control speed much better, so I used some of the motor methods that come with lejos[22]: motor.setSpeed(speed), motor.forward(), motor.backward(), motor.stop(true).

These simple methods would allow me to pretty much anything with the motors, I could set the speed of rotation (motor.setSpeed()) and then tell the motor to move forward or backward by using motor.forward() and motor.backward(), if I needed to stop the motor I could use motor.stop(true).

With these methods I could move the robot in any direction and speed, so the next step was to make the robot follow a path, which as I mentioned before, after the appropriate research and testing I found that using a black and white tape (black on the left and white on the right) would be the ideal way to do it. My approach method to this problem was the following:

First I needed to set up the light sensor so I used the “EV3ColorSensor(Port)” method, this opens the sensor given its port on the brick, then using the “EV3ColorSensor.getMode(Mode)” I could assign a mode to a “SampleProvider”, here I had different modes available: ColorID (returns a different value for different colors), RGB (returns Red, Green and Blue values reflected), Ambient (returns the luminosity values of the ambient) and Red (returns the Red values reflected). For the path following I used the Red Mode.

To instruct the robot where to move I would fetch samples of the sensor and according to the different values provided different actions would be taken.

The way the light sensor works is kind of like a human looking through a narrow peephole, we don’t just see a colored dot of the size of the peephole, it has a lens that amplifies the range of vision so we can see people, the light sensor works the same way [Figure 9].

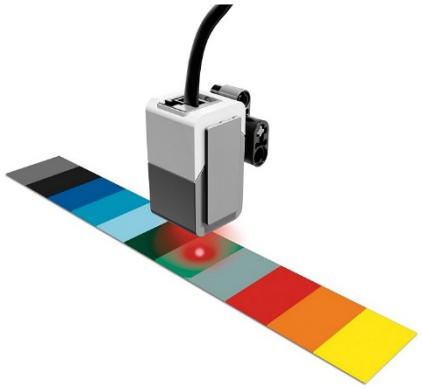


Figure 5.2

It project a red circle on the floor which gets bigger the further way the sensor is from the floor, however if it is too far away, it won’t pick up the reflection and if it is too close it won’t have enough distance to project light. At about 1 – 2 centimeters seems to be the perfect distance and because of the lens that amplifies the range of vision the sensor actually measures the mean value of the surface reflected, meaning that it won`t just reflect white or black as 2 separate values, instead, depending of how much of each color is within the range of measurement it will return different values, if more black is in that range, the lower the sensor value will be, if more white is within the range instead, the higher this values will be, if both colors occupy 50% of this range the value will be around to 0.2 to 0.3 according to my tests, this gave me a value to point towards, I could now decide that the desired value would be around 0.25.

Having this value allowed me to create an if statement that would use the sensor values as argument, if the value was higher than 0.25 the right wheel should move faster and the left wheel slower so it would compensate this value by turning left, the exact opposite would happen if the value was below 0.25, to make it smoother I decided to create different value bands, so that if the values is close to being the mean value it would not turn as aggressively as it would turn if the values was extreme, this allowed the robot to make small adjustments when going on a straight line but also turn sharply when needed[Figure 10].

```
 if(sample[0]> 0.5){motorC.setSpeed(-40);motorB.setSpeed(200);motorC.backward();
motorB.forward();}
else if(sample[0]> 0.4){motorC.setSpeed(0);motorB.setSpeed(200);motorC.forward();
motorB.forward();}
else if(sample[0]> 0.3){motorC.setSpeed(120);motorB.setSpeed(200);motorC.forward();
motorB.forward();}
else if(sample[0]< 0.2){motorC.setSpeed(200);motorB.setSpeed(120);motorC.forward();
motorB.forward();}
else if(sample[0]< 0.15){motorC.setSpeed(200);motorB.setSpeed(0);motorC.forward();
motorB.forward();}
else if(sample[0]< 0.05){motorC.setSpeed(200);motorB.setSpeed(40);motorC.forward();
motorB.backward();}
else{motorC.setSpeed(200);motorB.setSpeed(200); motorB.forward(); motorC.forward();}
```

Figure 5.3

5.6 Face Detection

After programming the path following functionality it was time to move on to the face detection part, this was by far the part that took the most time off of me, surprisingly not because of its complexity, it is easy to understand once you take the time to study about it, but it did take some time to tweak it so that it would pick up faces but nothing else, at the start it would pick up several objects as faces.

First of all I needed a camera, I did not have any suitable cameras at home so I did a little research online for cameras and found a very affordable one on Amazon, a 720p camera which shape seemed perfect for my robot[Figure 11], and it would be very easy to change the frame of the robot to hold the camera in place.

Figure 5.4

5.6.1 Image Processing

Because cascades detect differences in luminosity (edge detection) the image imported should be black and white, this will make it a lot easier for the classifier, for this I used some other methods from openCV called "ImgProc"[23], it has a lot of different function for image processing one of which is called "ImgProc.cvtColor()", this method is used to convert image colors and it imports 3 arguments, a source image, a destination image and a code, for this code I used "Imgproc.COLOR_BGR2GRAY", this simple line of code allowed me to convert the source image from the camera to a black and white image.

After looking at the output of this I noticed that sometimes the image would seem very dark overall and other times very bright, with direct sunlight for example, having worked with Matlab I have worked with black and white images and one of the things I did was balance an image, meaning that with a black and white image I could balance the white and black to achieve a visible image. Lets say I take a picture inside my house at night and with the lights off, it is obviously quite dark and turning the image into black and white really doesn't help, however, if I balance the histogram of that image I can increase the values of the image towards the white range and the image will be much clearer and so after researching the OpenCV API documents I found it had exactly what I needed, a method called Imgproc.equalizeHist(). This method simply takes a source image (black and white image) and outputs a replicated but balanced image automatically, this was perfect, this way the robot could now operate in quite dark places as well as bright places.

After doing some test with the cascades I found that objects with very sharp edges were becoming a problem, after considering that a face does not really have very sharp edges, at least not as sharp as some other objects, I decided that, using some of my knowledge acquired during Human Computer Interaction and Artificial Intelligence labs at MMU, using dilate and erode methods was something I should try. In theory it should work perfectly, it will simply remove some of the small gaps in some objects, and it would remove some very sharp edges, all things that were not necessary for face detection, so using a small 4 x 4 structuring element I eroded first and dilated after, this gave a much smoother picture but not too smooth so that the cascades could still detect edges, it worked perfectly. Combining all this image processing together[Figure 12] brought the performance down, it became more accurate but it would be processed with less frames per second since all of this processing had to be applied on every single frame.

```
g = jPanel1.getGraphics();
//Image processing
//remove gaps and too sharp edges
Imgproc.erode(frame, frame, Imgproc.getStructuringElement(Imgproc.MORPH_RECT, new Size(4,4)));
Imgproc.dilate(frame, frame, Imgproc.getStructuringElement(Imgproc.MORPH_RECT, new Size(4,4)));
//convert to black and white picture for better edge detection
Imgproc.cvtColor(frame, frame_gray, Imgproc.COLOR_BGR2GRAY);
//equalize histogram, balance white and black values (adapts to luminosity)
Imgproc.equalizeHist(frame_gray, frame_gray);
```

Figure 5.5

5.6.2 Fine Changes

To train a method like this it takes several thousands of pictures to make it effective, that is why openCV already brings some trained cascades, with a couple of different ones just for the face. Using the "CascadeClassifier.detectMultiScale(image, objects, scaleFactor. MinNighbors, flags, minSize, maxSize)" method and the API documents [17] I was able to apply the cascades for detecting frontal faces. This method takes a lot of arguments and after a lot of testing I found these be the best values regarding performance and efficiency:

- Image: This is the image we want to be processed.
- Objects: This is the variable we want the detected objects to be assigned to.
- ScaleFactor: This values scales the image which can improve the detection, a higher values will mean a better performance but it can overlook some important variations in scale, I used 1.2 for this value which was high enough to maintain it's accuracy and it improved the performance very slightly.
- MinNeighbors: This value added to 1 (minNeighbors+1) represents how many features we want to be detected to recognize the object as a face, too many objects we might miss a face, too little it will more likely pick up a non-face, for this I used 5, meaning it would require 5 important features to be detected close together to detect a face.
- Flags: This value represents some decisions the classifier can take, different options are available, the one I used was CASCADE_SCALE_IMAGE, which scales the image to the size of the trained data which I found to be more efficient and have a good performance.
- MinSize: This values determines the minimum size of a desired detection, anything below this it will not be considered as a detected object, my value for this, just like minNeigbors could not be too small, because very small objects could have many similar features to a face and a good example of that is a kitchen, which seemed a good place to test this and a place where if this values is too little can cause problems due to the presence of several small objects and very high contrasts. My value for this is 10 x 10 (pixels) .

- MaxSize: This value is just like minSize but it applies for the maximum size desired, anything bigger than that value will not be recognized as a match. My value for this is the size of the input image, meaning that it does not have a maximum value it is limited by the camera input size.

I then store the detected objects in an array [Figure 13] which will then be looped through allowing me to draw an ellipse around the detected object so I could see what the robot was picking up on the live stream.

```
faceDetector.detectMultiScale(frame_gray, rostros, 1.2, 5, 0 |CASCADE_SCALE_IMAGE, new Size(10, 10), new Size(w, h)
facesArray = rostros.toArray();
```

Figure 5.6

5.7 Robot to user Emailing function

When the camera is running it is constantly looping through its frames and looking for a face, very rarely it picks up some strange contrasts or objects as faces for a split second, to avoid this I created an integer variable called "foundFace**",** this variable as default is set 0 and when the camera detects an object, either correctly or incorrectly, this variable will increment every time, on the other hand, if while looping through the camera frames it encounters a frame with 0 objects detected (no faces), this variable will again be reset to 0, this allows me to set a kind of delay to call the email method, instead of emailing the detected object straight away, I have set an if statement that only calls this method if the variable "faceFound" holds a value equal to 8, this could have been another value, in this case, it will require the camera to detect a face for 8 consecutive frames before calling the email method, which avoids miss detections that may occur for just a couple of frames, generally when it is an actual face it will detect it for several consecutive frames, much more than 8.

Now moving to the email method, for this method I required research because it was something I had never done before but something I found very useful and from which I would surely benefit from in the future.

I started like most new challenges, I read through the Java API and looked for libraries that would allow me to send emails, and I found the **javax.email** package, which should allow an email system to be set up. After some reading I found it to be quite simple to understand, first I had to set the properties of the system (host, port, username....) as well as setting the authenticator class to "true" which verifies the authenticity of the account, this required me to disable some security options on my email account so that it would allow external "unsafe" programs to be able to login. I used my Gmail account which I found easiest to set the security options after setting up the adequate port, **587** [Figure 14] being the default for an SMTP (Simple Mail Transfer Protocol), which is the standardized protocol for the internet and in this case for Gmail too, but this may vary between different security protocols. Later in the project I also tested this with the university Wi-Fi which worked too.

```
public email() {
    // SMTP info
    String host = "smtp.gmail.com";
    String port = "587";
    String mailFrom = "tiaguinhofsx@gmail.com";
```

Figure 5.7

I built this email function as a separate class, and it would initialize all the necessary variables and set up everything for sending an email, calling this class [Figure 16] would retrieve an image called "MyFile.png" from a specific path, in my case, "c:/Users/Tiago/Desktop/projectPrograming/testCam/MyFile.png" (project source directory) [Figure 15].

```
// message info
String mailTo = "tiagocf_13@hotmail.com";
String subject = "Potential Intruder";
String message = "A potential intruder has been detected, open the detected frame attached and find out.";

// attach detection
String[] attachFiles = new String[1];
attachFiles[0] = "c:/Users/Tiago/Desktop/projectPrograming/testCam/MyFile.png";
```

Figure 5.8

```
else {
    if(foundFace == 8){
        System.out.println("Face detected!!!");
        BufferedImage shot = mat2Img(frame_gray);
    File f = new File("MyFile.png");
    ImageIO.write(shot, "PNG", f);
    System.out.println("Image Saved");
    new email();}

    motorB.stop(true);
    motorC.stop(true);
    foundFace = -20;
```

Figure 5.9

When the camera gets a positive detection for 8 consecutive frames it saves this image in the project directory, before saving this image it needs to be converted, because it is usually used as a set a "Mat" for the detection function but with a simple and standard conversion to image, this is saved and it can then call the email function, which will be looking for this image. I then set the “foundFace” variable to –20 and this is so that if it keeps detecting an intruder for consecutive frames after the first email, it won't keep sending emails every 8 frames, instead it will need to get 28 consecutive frames to send the next email. Another reason for this is that when detecting an intruder, the camera stops streaming new frames for around 1-2 seconds because converting the image, saving it and then sending the email does take some processing power. For that time the camera stops detection and so by having this kind of delay between emails it prevents the "spam" and it also prevents the loss of the detected intruder since during these 28 frames the camera motor will follow the detected object which might be lost if it needs to constantly stop and send a new and unnecessary email.

Another if statement would allow the camera to follow this detected object, it would simply use the position properties of the object relative to the camera frame and turn accordingly. If the detected

object was off-centered to left side of the camera frame the camera motor would turn right and if the object was to the right side of the camera frame the motor would turn left. The reason the motor turns to the opposite side of the detection is the dented wheel system, this system inverts the movement, when the motor dented wheel moves clockwise, the camera dented wheel will move counterclockwise.

5.8 Functionality Testing

Throughout the programming of this functions I constantly tested the robot every few complete lines to make sure everything was going according to plan and at this stage I did some major tests to test the overall performance of the robot and so I created a test path, which I made on top of the cardboard that comes with the EV3 parts which has different colored shapes for this exact purpose (testing), however I wanted the robot to follow a line so I used the black and white tape side by side to create a path which had straight lines, wide turns and sharp turns to see any tweaks I had to do, although I had already tested the path following function at early steps of the project I did not test it with all the other functions which could somehow influence the performance.

Using myself as a test subject I got the robot to follow the path and I would show my face to the robot and see if it would stop moving and follow my face with the camera motor, which he did, I would then walk away quickly so it couldn't follow me and then it would resume following the path like predicted. After checking my email I saw that it had also sent me an email with the frame containing my face.

Overall, the test was a success it worked as I wanted it to work, just some minor tweaks were necessary to the wheel movement for the wide and sharp turns.

6. Conclusion

Objectives achieved

Facing the deadline of the project I feel that the aims and goals set at the very beginning were accomplished as well as some extra functions like the email system and I believe the main purpose of the robot is useful and practical at some degree. I would also like to point out that this served as a kind of prototype project for me, it made me research more about robotics and how everything connects and with no doubt now I can set more challenges for myself and with better and more accurate hardware I can achieve bigger challenges.

Learning Outcomes

This project made me a better programmer and having never programmed for such a large scale project before it improved my organization and planning skills, it also became my own personal challenge outside university, it motivated me and got me very excited with the possible outcome. I faced several obstacles, like the face detection functionality, it was one of the functions I wanted to get perfect and I just dedicated myself to research as much as I could about it and made sure I understood the problem and the possible solutions and then moved to the actual practical programming of it. Programming the EV3 motors was an educative and fun part of this project, seeing something move because you programmed it to do so is definitely very powerful and exciting and with some research I experienced testing and try different approaches which I feel is essential when creating a project like this.

I did a lot of research to gather the knowledge for this project, however a big part of it was just trial and error and a lot of testing. Having not much information online it pushed me into learning from the robot through trial and error as well as the API documents, which were key for the success of the project and will certainly be a big advantage for me in the near future.

Proposed improvements

Having achieved my goals does not mean that the robot cannot be improved, quite the opposite, it has now a good basis for improvement, some of the improvements that I would like to have done if more time was provided are:

- Using wireless hardware for more movement freedom.
- Adding an extra motor to allow the camera to cover mover angles.
- Improve the map following performance by adding an extra sensor in a strategic position or using a second camera to detect a line.
- Create a mapping functionality to allow the robot to be aware of its position, taking out the need to have a line to follow.
- Create an SMS robot/user connection instead of email.
- Allow the user to text/email the robot to order a specific command (e.g. Guard just the front door).
- Allow the user to remotely control the robot through a mobile device with a camera feed.

Overview

Having now the final product, I can say, from what I found during all my research, this is a unique robot in the Lego Robotics community and I feel I achieved something that stands out from the crowd.

This definitely was not an area I was comfortable in at start and I faced a lot of obstacles and failures but I am glad I took the challenge and it was all for the best, like a wise man I admire very much once said "Success consists of going from failure to failure without loss of enthusiasm" [Winston Churchill].

References

[1] Amaryllo International (2015). "iCamPRO FHD".[Online] Available from: : http://www.amaryllo.eu/n50120products/index.php.

[Accessed: 4th October 2015].

[2] CES(2015). "2015 CES Innovation Awards". [Online] Available from: http://www.ce.org/i3/Features/2015/2015-CES-Innovation-Awards.apx.

[3] Omprakash, P. D. K. S. (2013)."Wireless Home Security System with Mobile". International Journal of Advanced Engineering Technology 7(2013).

[4] Shaligram, A., & Bangali, J. (2013). "Design and Implementation of Security Systems for Smart Home based on GSM technology".

International Journal of Smart Home 7(6).

[5] Math Works (2015). "Patter Recognition". [Online] Available from: http://uk.mathworks.com/discovery/pattern-recognition.html.

[Accessed: 16th October 2015].

[6] Dave Parker (2011). "Using the Light Sensor" (Lego Mindstorms NTX). [Online] Available from: http://www.nxtprograms.com/NXT2/multi-bot/light.html.

[Accessed: 17th October 2015].

[7] Knightscope, Inc.. 2015. Knightscope, Inc.. [ONLINE] Available at: http://www.knightscope.com/. [Accessed 3 December 2015].

[8] Paul Viola and Michael Jones (2001). "Robust Real-Time Face Detection. [Online] Available from: http://www.vision.caltech.edu/html-files/EE148-2005-Spring/pprs/viola04ijcv.pdf

[Accessed: 15th November 2015]

[9] Home - LEGO® MINDSTORMS® - LEGO.com - Mindstorms LEGO.com. 2014. Home - LEGO® MINDSTORMS® - LEGO.com - Mindstorms LEGO.com. [ONLINE] Available at:http://www.lego.com/en-us/mindstorms/?domainredir=mindstorms.lego.com. [Accessed 4 October 2015].

[10] R3TAR - Build a robot - LEGO.com. 2015. R3TAR - Build a robot - LEGO.com. [ONLINE] Available at:http://www.lego.com/en-gb/mindstorms/build-a-robot/r3ptar. [Accessed 25 October 2015].

[11] TRACK3R - Build a robot - LEGO.com. 2015. TRACK3R - Build a robot - LEGO.com. [ONLINE] Available at:http://www.lego.com/en-gb/mindstorms/build-a-robot/track3r. [Accessed 30 October 2015].

[12] EV3 PROGRAMMER APP - Apps - LEGO.com. 2015. EV3 PROGRAMMER APP - Apps - LEGO.com. [ONLINE] Available at: http://www.lego.com/en-gb/mindstorms/apps/ev3-programmer-app. [Accessed 1 October 2015].

[13] EV3 Programming Sensors | STEM Education. 2015. EV3 Programming Sensors | STEM Education. [ONLINE] Available at: http://www.ucalgary.ca/IOSTEM/ev3-programming/ev3-programming-sensors. [Accessed 1 October 2015].

[14] HelloWorld.java: first steps with Java on EV3. 2015. HelloWorld.java: first steps with Java on EV3. [ONLINE] Available at: http://thinkbricks.net/helloworld-java-first-steps-with-java-on-ev3/. [Accessed November 2015].

[15] Eclipse - The Eclipse Foundation open source community website. 2015. Eclipse - The Eclipse Foundation open source community website.. [ONLINE] Available at: https://eclipse.org/. [Accessed November 2015].

[16] OpenCV (2015). [Online] Available at: http://docs.opencv.org/3.1.0/d7/d8b/tutorial_py_face_detection.html#gsc.tab=0

[Accessed: December 2015]

[17] Cascade Classification — OpenCV 2.4.12.0 documentation. 2016. Cascade Classification — OpenCV 2.4.12.0 documentation. [ONLINE] Available at:http://docs.opencv.org/2.4/modules/objdetect/doc/cascade_classification.html. [Accessed December 2015].

[18] GitHub - bytedeco/javacv: Java interface to OpenCV and more. 2015. GitHub - bytedeco/javacv: Java interface to OpenCV and more. [ONLINE] Available at: https://github.com/bytedeco/javacv. [Accessed December 2015].

[19] LeJOS, Java for Lego Mindstorms. 2015. LeJOS, Java for Lego Mindstorms. [ONLINE] Available at: http://www.lejos.org/index.php. [Accessed December 2015].

[20] leJOS EV3 API documentation. 2016. leJOS EV3 API documentation. [ONLINE] Available at:http://www.lejos.org/ev3/docs/. [Accessed January 2016].

[21] DifferentialPilot (leJOS EV3 API documentation). 2016. DifferentialPilot (leJOS EV3 API documentation). [ONLINE] Available at:http://www.lejos.org/ev3/docs/lejos/robotics/navigation/DifferentialPilot.html. [Accessed January 2016].

[22] leJOS EV3 API documentation. 2016. leJOS EV3 API documentation. [ONLINE] Available at:http://www.lejos.or\/ev3/docs/. [Accessed January 2016].

[23] imgproc. Image Processing — OpenCV 2.4.12.0 documentation. 2016. imgproc. Image Processing — OpenCV 2.4.12.0 documentation. [ONLINE] Available at:http://docs.opencv.org/2.4/modules/imgproc/doc/imgproc.html. [Accessed January 2016].

Bibliography

"Robot Design | FLL Starting Point". *Fllstartingpoint.com*. N.p., 2015. Web. Dec. 2015.

"Java Platform SE 6". *Docs.oracle.com*. N.p., 2016. Web. Dec. 2015.

Mohamed, Fady A., and Elsayed E. Hemayed. "Using Trusted Computing In Trusted Mail Transfer Protocol". *Security and Communication Networks* 7.5 (2013): 926-933. Web.

Appendices

Appendix A – Terms of Reference

Background

For recent years home security has become more and more important and one of the fields where a lot of homeowners decide to invest to keep their families and their properties safe. Unlike a few years ago we now have systems that allow us to record and move cameras, connect them to 24/7 surveillance systems and even detect movement. Recently this field has had some special attention and one of the most advanced ones would probably be the "iCamPro FHD" [1] which was the award winner for 2015 CES Best of Innovation [2] for his amazing ability to hear, see and track movement, being called the first intelligent home security robot which made me research about other Home Security systems starting with GSM (Global System for Mobile Communication) based Systems [3] and about its design and implementation [4].

All these innovations aim to make security flawless and not human dependent, meaning that there is less flexibility for human error, it also tries to achieve an automatic response to different situations.

However, what I will try to do in this project is create a robot that will allow a home owner to order a robot to patrol the house and detect intruders while patroling. In order to achieve this I will do intensive research in the fields of robotics in general, machine learning, imagem processing [5] , light sensor [6], face detection, programming and Human Computer Interaction.

By applying all that knowledge I hope to obtain a versatile and reliable robot that will be rigorous in term of safety and error free to achieve a safety feel to the home/robot owner.

Aim of Research

This project is to aim a patrolling unit ranging functionalities like face detection, machine learning and path following,

Objectives

1. Study techniques for robot movement, map awareness, image processing, machine learning and robot/user connection.

2. Build a stable and reliable hardware (robot).

3. Design and Implement a path patrolling system.

4. Design and Implement machine learning techniques.

5. Design and Implement movement track/detection techniques.

6. Design and Implement User/Robot connection.

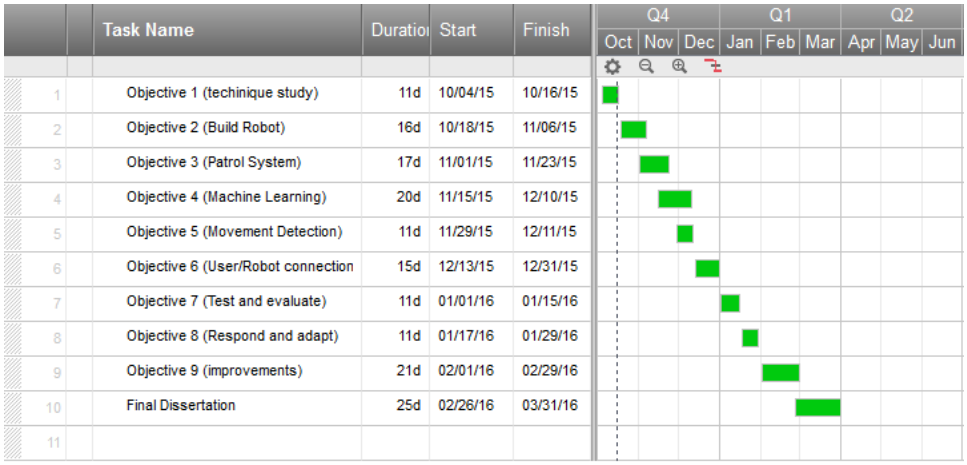
7. Test each implementation thoroughly.

8. Intensive testing for different situations and responses.

9. Improve system to an advanced level.

Methodology and Program of Study

Bellow you can find the methodology schedule I aimed to achieve from the beginning of this project [Figure 1]:



	Task Name	Duratio	Start	Finish
1	Objective 1 (techinique study)	11d	10/04/15	10/16/15
2	Objective 2 (Build Robot)	16d	10/18/15	11/06/15
3	Objective 3 (Patrol System)	17d	11/01/15	11/23/15
4	Objective 4 (Machine Learning)	20d	11/15/15	12/10/15
5	Objective 5 (Movement Detection)	11d	11/29/15	12/11/15
6	Objective 6 (User/Robot connection	15d	12/13/15	12/31/15
7	Objective 7 (Test and evaluate)	11d	01/01/16	01/15/16
8	Objective 8 (Respond and adapt)	11d	01/17/16	01/29/16
9	Objective 9 (improvements)	21d	02/01/16	02/29/16
10	Final Dissertation	25d	02/26/16	03/31/16
11				

Learning Outcomes

At the end of this project I am expecting to have the following learning outcomes:

- Improve organization skills.
- Improve planning skills.
- Gain project management skills.
- Gain knowledge in Robotics.
- Gain knowledge in Artificial Intelligence.
- Improve my programming skills.
- Be able to set bigger challenged in the near future.

Appendix B – Ethics Form

ETHICS CHECK FORM

This checklist must be completed for every project. It is used to identify whether there are any ethical issues associated with your project and if a full application for ethics approval is required. If a full application is required, you will need to complete the 'Application for Ethical Approval' form and submit it to the relevant Faculty Academic Ethics Committee, or, if your research falls within the NHS, you will need to obtain the required application form from the National Research Ethics Service available at www.nres.npsa.nhs.uk/ and submit it to a local NHS REC.

Before completing this form, please refer to the University's Academic Ethical Framework (www.rdu.mmu.ac.uk/ethics/mmuframework) **and the University's Guidelines on Good Research Practice** (www.rdu.mmu.ac.uk/rdegrees/goodpractice.doc)**.**

Project and Applicant Details

Name of applicant (Principal Investigator):	Tiago Carneiro Fernandes
Telephone Number:	00447512264948
Email address:	13144415@stu.mmu.ac.uk
Status:	Undergraduate Student
Department/School/Other Unit:	Manchester Metropolitan University
Programme of study (if applicable):	Final Year Project
Name of supervisor (if applicable):	Dr. David McLean
Project Title:	HIDU: Home Intruder Detection Unit
Does the project require NHS Trust approval? If yes, has approval been granted by the Trust? Attach copy of letter of approval.	NO

Ethics Checklist (Please answer each question by ticking the appropriate box)

		Yes	No	N/A
1.	Will the study involve recruitment of patients or staff through the NHS, or involve NHS resources? If yes, you may need full ethical approval from the NHS.		X	
2.	Does the study involve participants who are particularly vulnerable or unable to give informed consent (e.g. children, people with learning disabilities, your own students)?		X	
3.	Will the study require the co-operation of a gatekeeper for initial access to the groups or individuals to be recruited (e.g. students at school, members of self-help group, nursing home residents)?		X	
4.	Will the study involve the use of participants' images or sensitive data (e.g. participants personal details stored electronically, image capture techniques)?		X	
5.	Will the study involve discussion of sensitive topics (e.g. sexual activity, drug use)?		X	
6.	Could the study induce psychological stress or anxiety or cause harm or negative consequences beyond the risks encountered in normal life?		X	
7.	Will blood or tissue samples be obtained from participants?		X	
8.	Are drugs, placebos or other substances (e.g. food substances, vitamins) to be administered to the study participants or will the study involve invasive, intrusive or potentially harmful procedures of any kind?		X	
9.	Is pain or more than mild discomfort likely to result from the study?		X	
10.	Will the study involve prolonged or repetitive testing?		X	
		Yes	**No**	**N/A**

11. Will it be necessary for participants to take part in the study without their knowledge and informed consent at the time (e.g. covert observation of people in non-public places)?		X	
12. Will financial inducements (other than reasonable expenses and compensation for time) be offered to participants?		X	
13. Is there any possible risk to the researcher (e.g. working alone with participants, interviewing in secluded or dangerous)?		X	
14. Has appropriate assessment of risk been undertaken in relation to this project?		X	
15. Does any relationship exist between the researcher(s) and the participant(s), other than that required by the activities associated with the project (e.g., fellow students, staff, etc)?		X	
16. Faculty specific question, e.g., will the study sample group exceed the minimum effective size?		X	

If you have ticked 'no' or 'n/a' to all questions, attach the completed and signed form to your project approval form, or equivalent. Undergraduate and taught higher degree students should retain a copy of the form and submit it with their research report or dissertation (bound in at the end). MPhil/PhD, and other higher degree by research, students should submit a copy to the Faculty Research Degrees Sub-Committee with their application for registration (RD1) and forward a copy to their Faculty Academic Ethics Committee. Members of staff should send a copy to their Faculty Academic Ethics Committee before commencement of the project.

If you have ticked 'yes' to ***any*** of the questions, please describe the ethical issues raised on a separate page. You will need to submit your plans for addressing the ethical issues raised by your proposal using the 'Application for Ethical Approval' form which should be submitted to the relevant Faculty Academic Ethics Committee. This can be obtained from the University website (http://www.rdu.mmu.ac.uk/ethics/index.php).

If you answered 'yes' to question 1, you may also need to submit an application to the appropriate external health authority ethics committee, via the National Research Ethics Service (NRES), found at http://www.nres.npsa.nhs.uk/, and send a copy to the Faculty Academic Ethics Committee for their records.

Please note that it is your responsibility to follow the University's Guidelines on Good Research Practice and any relevant academic or professional guidelines in the conduct of your study. **This includes providing appropriate information sheets and consent forms, and ensuring confidentiality in the storage and use of data.** Any significant change in the question, design or conduct over the course of the research should be notified to the relevant committee (either Faculty Academic Ethics Committee of Local Research Ethics Committee if an NHS-related project) and may require a new application for ethics approval.

Approval for the above named proposal is granted

I confirm that there are no ethical issues requiring further consideration. *(Any subsequent changes to the nature of the project will require a review of the ethical consideration(s).)*
Signature of Supervisor (for students), or Manager (for staff): _____Tiago Carneiro Fernandes_____
Date: ____01/04/2016____

Approval for the above named proposal is not granted

I confirm that there are ethical issues requiring further consideration and will refer the project proposal to the Faculty Academic Ethics Committee.
Signature of Supervisor (for students), or Manager (for staff): _____
Date: _____

Separate page for ethical issues:-