**M&A Logbook**

**By: Matthew Feros and**

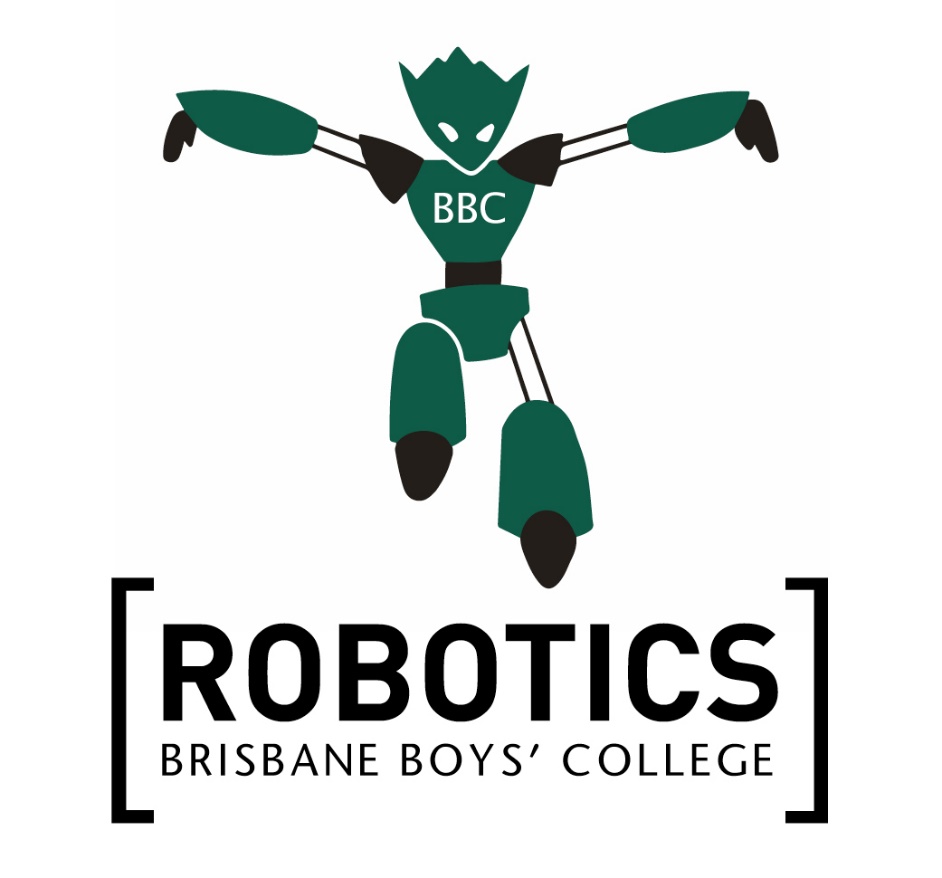
** Aparaj Bogahawatta**

Table of Contents

[Project Overview 5](#_Toc514829065)

[Solving the Problem 5](#_Toc514829066)

[Roles of Team Members 5](#_Toc514829067)

[Coding Platform 5](#_Toc514829068)

[Hardware 6](#_Toc514829069)

[How are the robot parts arranged and why? 6](#_Toc514829070)

[Where are the motors? 6](#_Toc514829071)

[What do they do? 6](#_Toc514829072)

[Where are the sensors? 7](#_Toc514829073)

[What do they do? 7](#_Toc514829074)

[How do you keep it from breaking? 7](#_Toc514829075)

[Is it safe to use around people? 7](#_Toc514829076)

[Software 7](#_Toc514829077)

[What software did you use and why? 7](#_Toc514829078)

[Pseudo Code 8](#_Toc514829079)

[Flowcharts 8](#_Toc514829080)

[Can you show how your program evolved? 8](#_Toc514829081)

[Electronics 9](#_Toc514829082)

[What modules did you use and why? (e.g., motor control, sensors, microcontroller) 9](#_Toc514829083)

[Did you use custom electronics? 9](#_Toc514829084)

[Did you make your own circuits? 9](#_Toc514829085)

[Community Outreach 9](#_Toc514829086)

[What was the hardest part of the Project? 9](#_Toc514829087)

[Aparaj Bogahawatta: 9](#_Toc514829088)

[What challenges did the project face? 9](#_Toc514829089)

[The team? 9](#_Toc514829090)

[The software? 9](#_Toc514829091)

[The implementation? 9](#_Toc514829092)

[Choosing a colour scheme? 9](#_Toc514829093)

[How did you go about fixing them? 9](#_Toc514829094)

[How would you improve the whole process? 9](#_Toc514829095)

[What’s next? 10](#_Toc514829096)

[Matthew Feros: 10](#_Toc514829097)

[What challenges did the project face? 10](#_Toc514829098)

[The team? 10](#_Toc514829099)

[The design? 10](#_Toc514829100)

[The implementation? 10](#_Toc514829101)

[Choosing a colour scheme? 10](#_Toc514829102)

[How did you go about fixing them? 10](#_Toc514829103)

[How would you improve the whole process? 10](#_Toc514829104)

[What’s next? 10](#_Toc514829105)

[The innovative parts of the robot? 10](#_Toc514829106)

[Why is the robot really cool :)? 10](#_Toc514829107)

[Weekly Log Book 10](#_Toc514829108)

[Term 1: 10](#_Toc514829109)

[Week 6: 10](#_Toc514829110)

[Week 7: 11](#_Toc514829111)

[Week 8: 11](#_Toc514829112)

[Week 9: 11](#_Toc514829113)

[Week 10: 12](#_Toc514829114)

[Holiday Week 1: 12](#_Toc514829115)

[Holiday Week 2: 12](#_Toc514829116)

[Term 2: 12](#_Toc514829117)

[Week 1: 12](#_Toc514829118)

[Week 2: 13](#_Toc514829119)

[Week 3: 13](#_Toc514829120)

[Week 4: 13](#_Toc514829121)

[Week 5: 15](#_Toc514829122)

# Project Overview

This year we need to create two robots that are able to play soccer against other robots. We are designing our robots from custom made parts, designed in Autodesk Inventor and Autodesk Eagle, and we are programming them in C++ by using the text editor, Atom. Our aim is to be able to program our robots so that they can play soccer with an IR emitting ball while adhering to all the rules of the RoboCup Junior Lightweight Competition. With these robots we would like to play as many games as possible in the regional, state and national competitions so we are able to improve our robot’s code and design for future years.

# Solving the Problem

This problem can be broken down into many parts. The two main parts in this project are the hardware side and the software side. In these separate components we need to break the problem even further so we both know what needs to be accomplished to fulfil the aim of this project.

## Hardware

In Hardware there are two main parts, the actual design and the electronic side. Our first job is to design a robot. This was done on Inventor. After we had finished that we had to make the electronic parts that was needed for the robot. The parts needed were the power supply, two motor controllers and a ring of IR sensors and a number of light sensors. The motor, controllers were the only off-the-shelf items that we used. All the other electronics parts were designed by Matthew Feros. We also decided to have a three plate design. This included a top, middle and bottom plate. The bottom plate would have the motors and the light sensors. The middle plate would hole the microcontroller, IR sensors, Power Supply and the motor controllers. The top plate would have the compass sensor and would act as a handle.

## Software

For the software side of the robot the first step was to find a text editor and a compiler that would work with C++ and the microcontroller. After we had finished that we needed to learn how to code in C++ and then we needed to decide how we were going to code our robot. We decided to break the software into smaller parts. Each part was for a different sensor. We made libraries which held the code for different sensors and then they were all combined into the main code where all the sensors would interact with eacc other so the robot could play a game of soccer.

# Roles of Team Members

The main role for Matthew Feros is the hardware part of the robot and the main role for Aparaj Bogahawatta is the software part of the robot. However, we both need to have knowledge in both areas since they both overlap. Aparaj needs to know how the robot is going to be designed so he is able to make a program accordingly. Matthew needs to make sure his design or any changes he makes does not interfere with the software side of the robot. Both members need to know how both areas work so that if there is an emergency in the competition any one of the team members can handle the situation. This means that both members need to have some knowledge in both fields to improve collaboration and results.

# Coding Platform

We are using a platform called Platformio. This is supplied by a company called GitHub. GitHub also supplied atom, which is a C++ editor. We downloaded atom because it is an easy-to-use textual-code editor and we used Platformio because it can be downloaded onto atom. However, we found that it was more efficient if we download Platformio through Python onto our computers because this meant we were able to use PowerShell on an app called GitBash to compile and upload our programs. The main reason for why we used Platformio is because it is efficient and really easy to learn.

# Hardware

## How are the robot parts arranged and why?

The robot consists of three main plates, a bottom plate, middle plate and the top plate. The bottom plate holds three motors and its controllers which are all angled 60° apart to allow for Omni-directional movement. This plate also holds light sensitive modules which give the robot sight as to the colour of the surface under the robot. This allows the robot to avoid the outlines of the field, which if passed will have the robot removed from the field. The middle plate consists of holes to accommodate 12 infrared light sensitive modules, the main controller and the inbuilt power supply. The top plate houses the direction sensitive module to give the robot information as to what side of the field it is facing, the top plate also protects the electronics under it, allows for a handle to be mounted to the top and stops the robot from entering the goal.

## Where are the motors?

The motors are orientated at 60° apart from each other around the centre of the robot. The motors have omnidirectional wheels attached to their shafts that allow for controlled movement in the perpendicular direction of the robot however the wheels move freely in the other direction. The motors are held by custom made 3D printed mounts that are connected to the bottom and middle platesunts on the bottom plate of the robot so that they are low enough to maintain contact with the field.

## What do they do?

The motors give accurate control over the movement of the robot. Their speeds are dynamically controlled by a motor controller which uses a 12 volt current and a PWM signal from the Teensy microcontroller to move the motors at a controlled speed. They allow us to move our robot so we are able to orbit and follow the IR emitting ball.

## Where are the sensors?

There are four different types of sensors on our robot. There are light sensors, TSSP Sensors, an IMU sensor and a Laser Range Finder (LRF). The light sensors are located on the bottom of the robot. The TSSP Sensors are located in the middle plate of the robot. The IMU is on the very top of the robot and is in the centre of the robot. The LRF is located at the back of the top plate

## What do they do?

Light Sensors are placed in a ring so the robot can easily detect colours in all direction. They detect the different colours of the playing field allowing the robot to detect the out-line and avoid the out area. The TSSP sensors are the sensors that detect the Infrared Ball. They are placed at 30o increments so the ball can be seen in all directions. The IMU sensor is an accelerometer, magnetometer and gyro sensors all in one PCB. We only use the Gyroscope sensor because that can sense direction. This means that we are able to keep our robots facing in one direction all the time. The LRF allows us to see the distance the sensor is away from another object. This is useful for a goalie.

## How do you keep it from breaking?

We keep the sensors from breaking by having plates that protect them. For the light sensors, the bottom plate surrounds the light sensors meaning that any impact will be absorbed by the base plate rather than the sensors. The same is for the TSSP sensors. They are surrounded by the Middle plate, meaning that any impacts will also be absorbed by the middle plate. The LRF is the only sensor that has more of a chance of being broken. However, it is still just inside the top plate meaning it would be difficult for another robot to break it.

## Is it safe to use around people?

The robot is safe to use around people because it is a cylindrical shape with rounded corners. There are also wheel guards that cover the wheels, meaning that the fast spinning motors will not hurt anyone’s fingers. The robot is very fast meaning that it could hurt someone who puts their hand inside the robot, but if the right safety precautions are taken there should be no injuries.

# Software

## What software did you use and why?

We are using a text editor called Atom to edit our code. We use Atom because we are able to download a platform called Platformio through it. Using Platformio we are able to make a code for a specific microcontroller. We just have to put in the name of the microcontroller and if Platformio have information about it, it will create a project that will be compatible with the microcontroller. We also used Atom because we were able to use a C++ editor on that software.

## Pseudo Code

Set-Up Code

The Arduino is turned on.

All the sensors are powered and run through their respective setup codes.

The motors are setup and are braked.

The Arduino checks if it has to act as a Defender or an Attacker.

Loop

The robot uses the different values from the IR, Light, Gyroscope and LRF sensors to know where the ball is and what direction to move. The light sensors are for out avoidance. The IR sensors detect the ball. The compass sensor makes sure the robot is always facing in the same direction. The LRF detects the distance from the goal.

## Can you show how your program evolved?

Our code has evolved throughout the past year. It started as a logic code. This is where all the different calculations were put. The syntax wasn’t adhered to but all the mathematical calculations that would be used later on were all in that code. This meant when I was writing my code I was able to refer back to the logic code for all the calculations. The next big change was for an actual code. This code adhered to the syntax and it was designed so it was able to be compiled and uploaded into an Arduino Mega. This code took much longer to write and edit because I had to make libraries for all the different objects and then bring them together in the main code. The next big change was the competition code. This code had a striker program as well as a goalie program. It was also much more efficient and relied on more pre-processed variables rather than normal variables. After using that code for the state competition, I moved to a much more object orientated program. This program followed the normal Standards of C++. This meant that the readability was better and it would be easier to identify different parts of the code.

# Electronics

## What modules did you use and why?

We used **\_\_\_\_\_\_\_** as motor controllers. We used motor controllers so that we were able to control the speed of each motor, thus controlling the direction of which the robot is moving. We used TSSP, Light and IMU sensors to detect the direction of the ball, detect the out-line and to detect the heading of the robot. We also use an LRF for our goalie to detect the distance from the goal. All this data helps the robot to run efficiently and smoothly so that it can play a proper game of soccer. We used a Teensy 3.5 as the microcontroller because they are pre-made and they have the ability to run the robot to a standard needed to power all the components.

## Did you use custom electronics?

We did use custom electronics for the Power circuit, Teensy3.5s, Light and TSSP sensors. We designed the circuitry on an app called Autodesk Eagle and then we ordered them through a company called PCB Way.

## Did you make your own circuits?

We did make our own circuits because we were only allowed to make individual components with Eagle rather than the whole robot. This meant that instead of one massive PCB we had multiple smaller ones that were all connected in a massive circuit.

# Community Outreach

We will be showcasing our robots to the community and some teachers and peers on Robocup competition days. We do not have any specific dates to show off our robots other than the competition however, our school does Open Mornings where our robots will be displayed for others to see.

# What was the hardest part of the Project?

## Aparaj Bogahawatta:

### What challenges did the project face?

The hardest part of the project was switching to the Teensy 3.5s. This is because I had to rewrite my whole code so that it was more efficient and so that it would work for our new robots. Another hard part was creating a defender from scratch.

### The team?

There was no challenges with the team since we were able to work well together and we always had jobs to do and were never not doing anything useful.

### The software?

The hardest part about the software was trying to get our TSSPs to read faster than 20 times a second. After this was achieved, the whole program was a lot more efficient and worked very well.

### The implementation?

### Choosing a colour scheme?

### How did you go about fixing them?

### How would you improve the whole process?

### What’s next?

## Matthew Feros:

### What challenges did the project face?

### The team?

### The design?

### The implementation?

### Choosing a colour scheme?

### How did you go about fixing them?

### How would you improve the whole process?

### What’s next?

**Note: Fill this out the week before nationals.**

# The innovative parts of the robot?

## Why is the robot really cool :)?

We are the first team in BBC Robotics to be using custom PCBs in our first year of Custom Soccer. This means that we had to design all our PCBs from scratch. We also have a capture zone that is curved to the shape of the ball so the ball doesn’t come out of the capture zone as easily. We are also the first team in Australia to use a V shaped mirror on our robot. This means that we are able to see in two directions with only one camera. We also have curved guards over our wheels so they are completely covered and don’t get damaged by other robots. These guards match with the colour scheme of the robot. We have made the robot so that one of the robots is mainly black and the other is mainly white.

# Weekly Log Book

## Term 1:

### Week 6:

#### Aparaj:

This week I solved the problem with my Arduino Mega. I was able to connect my Mega to my computer and I knew why it was connecting before. I had my code automatically sending the code on a different COM link to the one my Mega was connected to. This meant I had to change the COM link on my code and now it uploads and monitors perfectly. I also tested my TSOP array library for direction testing. I made sure that if I inputted an angle it would return the same angle after multiple calculations.

#### Matthew:

### Week 7:

#### Aparaj:

This week I put to use my Mega as it was now able to connect to my computer. I was able to test my orbit angles and strength readings using the Serial Monitor on PowerShell. I found however that my compass correction calculations are messing up my motor calculations. Next week I will have to learn how to fix this problem and understand my compass code better.

#### Matthew:

### Week 8:

#### Aparaj:

This week I learnt how the I2C and Compass library files work and I understood why my motor calculations were not working. I also made a robot library and a pixy library. The robot library has all the calculations that I had in the main but it is now under the robot object. I also made a Pixy library just in case we are allowed to use pixies. This meant that I made the code to adjust the rotations in my robot library. Next week I have got to test that the pixy code works and double check that the compass correction works correctly.

#### Matthew:

### Week 9:

#### Aparaj:

I tested my pixy code this week and my compass correction code. All of the logic works perfectly. However, I need a Gyroscope to make sure that my compass code works 100% correctly. I learnt how to solder and I soldered some capacitors on some light sensors. I would not be doing most of the soldering because that is not my part of the team, however, I would need to help when needed or when I had nothing else to do so that we would have more time to test with a robot.

#### Matthew:

### Week 10:

#### Aparaj:

This week I just helped out with some soldering again as I am not able to do anything without a physical robot. There was only one session of robotics this week meaning less work was done.

#### Matthew:

### Holiday Week 1:

#### Aparaj:

I was away for a holiday during this week.

#### Matthew:

### Holiday Week 2:

#### Aparaj:

I was away for a holiday during this week.

#### Matthew:

## Term 2:

### Week 1:

#### Aparaj:

This week I learnt about how to program a Pixy. A Pixy is a camera that can be used to track the whereabouts of the goal. This means that our robot will score even more goals. I made a function that would return the angle of where the goal is. I also found a few variables that are able to be pre-processed. I changed these variables to constants so the program would be more efficient. I also started a Goalie library. This is so that we are able to have a goalie in competition if we are able to pull it off.

#### Matthew:

### Week 2:

#### Aparaj:

This week I finished off my Goalie library. It compiled and seemed to work, however I wasn’t able to test it yet. I tried to make the program by using trigonometry, but I found out that it was much easier to use the object angle from the camera to calculate the distance the robot was from the goal. I also found out that my Pixy wasn’t connecting to my computer for some reason.

#### Matthew:

### Week 3:

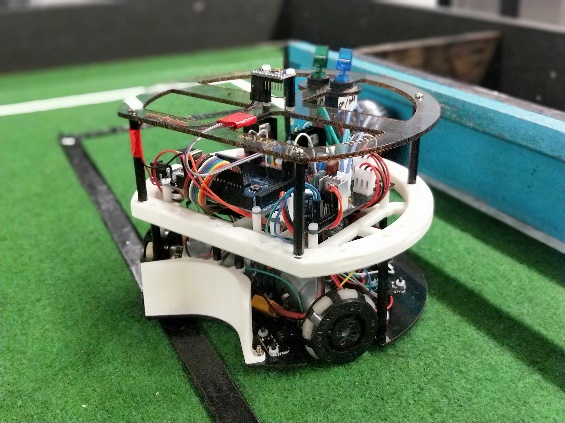
#### Aparaj:

This week I tested our power circuits. They all worked properly. The means that we had five working power circuits. I also designed a draft logo for our team. I was able to fix up some minor errors in the program. I also made a barrier between the different objects’ variables. This means that other objects can’t change the value of variable from other objects. I also finished testing our compass correction code on the test robot and all of it worked fine in the Serial mode. However, the robot wasn’t working on the field and we weren’t sure why.

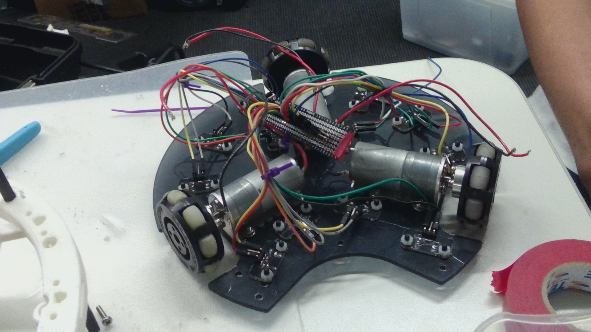
#### Matthew:

### Week 4:

#### Aparaj:

This week we found out that one of the motor controllers was not working. That is why the compass correction was messing up. I replaced the motor controllers and our robot is working well. I am now trying to fine tune the compass correction by using the PID method. I found that I could also improve the efficiency of the code by having different classes already defined after the creation of one object. I also identified a problem in the code. Sometimes the motors would stop moving even when it was supposed to be compass correcting. I fixed the problem by using PID to calculate my compass correction. Before I was just using P but I wasn’t doing the calculations correctly. I fixed the calculation and now the robot works. I just have to fine tune the correction. I have also added a program that negates the compass drift by calculating the change per second and adding that on. We got our plates and we built one of the robots but the TSOP ring was not the latest one. This meant that the standoffs we leaning out a little bit for the top plate.

#### Matthew:



### Week 5:

#### Aparaj:

This week I started to test the compass correction on our actual robot. I found that our proportional and derivative was going to be very high due to the fact that our motors were very weak. However, I soon realised that I was able to have a minimum correcting speed so I could bring my derivative and proportional values back down to normal. I also started to work on my out avoidance. I found out that my light sensors were returning weird values. However, all the values were consistent between the light sensors so it was possible to program them. I also added a function in my program that would negate the drift of our Gyroscope sensor. This meant that the readings of the gyroscope were much more accurate.

#### Matthew:

### Week 6:

#### Aparaj:

This week I found that the motors we were using were all running at different speeds and they were making the compass correction inaccurate. The motors were also clogging up, this meant that sometimes one of the motors needed replacing. This meant that I had to wait for Matthew to switch the motors so I was able to code them.

#### Matthew:

### Week 7:

#### Aparaj:

This week I started to started to work with the compass correction. We had new motors on our robot. However, I found that the new motors didn’t have enough talk to actually move the robot on the field. This meant the motors weren’t able to move the robot unless they were at a very high speed. This meant that there was a problem with the motor. However, I was able to check my compass correction code with my Serial Monitor and the Compass. At the end of the week we found that one of our motors was only spinning in one direction. This meant that when it was supposed to be spinning one way it would just not spin. When it needed to spin the other way it would work fine.

#### Matthew:

### Week 8:

#### Aparaj:

This week I found out that the PWM pins were not in the right places on the robot. After fixing this, all the motors starting spinning in the correct direction. Then I started to work on the TSOPs. I found out that two of the TSOPs were broken. This meant that they needed to be replaced. I managed to get the robot to go towards the direction of the ball. Then the next step was to orbit the ball. I found out that the orbit was not large enough. This meant that the robot kept running into the ball. I also found out that sometimes the robot would just crash for no reason. The reason was because the compass sensor was not connected properly to the Arduino and this was stopping the whole code.