https://github.com/lovemich/AVC_Team-Epsilon

AVC PROGRESS REPORT

300381661

Lavanya Sajwan
Lab: Wednesday 10am-12pm

LECTURERS - Elf, Bryan Ng, Arthur Roberts

Abstract

This report consists of Team Epsilon's design, coding and testing progress of our autonomous vehicle. The task given to us by the clients was to navigate a maze. According to the rules of this challenge, the team was initially given a Raspberry Pi unit, a PCB (printed circuit board), a basic chassis, an H-bridge, camera, motors and wheels. These were to be the starting point of our robots and the team had to build on to it. Each team is also given an extra $100 "Arthur dollars", virtual money that the team can choose to spend on additional parts.

Introduction

During this Autonomous Vehicle Challenge (AVC), each individual in teams are learning to build and test a functioning robot while working with new people. This project will also call for us to maintain frequent record keeping and to develop a report. The team is doing this to keep the customers happy with our product and to fulfil their specific requirements. Our aim is to apply what each member has learnt in lectures and during assignments to synthesise a way for our robot to complete the maze by overcoming the obstacles in each quadrant and all done within the time limit of 15 minutes. Our objectives for this will be to put together a functioning robot, produce a working program, undergo rigorous testing and to write a report, all while keeping the specifications in mind as well as staying within the given budget and sticking to weekly goals and tasks. Some of the specification chosen by the clients comprises of:

1. Teams - the teams are not chosen, and are randomly put together based on assignment marks.
2. Aesthetic - the robot must be engineering aesthetically pleasing, so that there are no hanging wires. It also has to be easy to take apart so that it is easy to take apart in case the team makes a mistake and for reusability reasons. Therefore too much solder is not ideal.
3. Maze - The robots have to finish the maze within the 15 minutes time frame without any interaction and must go from start to finish.

Each team member will benefit during this challenge as each individual will develop their coding, hardwiring and teamwork skills. Overall this challenge will give us an insight on what it is like to be in an environment based on engineering concepts.

Background

The AVC, is the autonomous vehicle challenge, it calls upon teams to make a vehicle that is able to operate without human interaction; this is different to automate which relies on the basis of human control. There are five levels of automation, as stated from the National Highway Traffic Safety administration (NHTSA) of USA. At level zero, the vehicle is automated. At level 1 there is specific-function automation, which means that the driver controls everything else apart from some functions. At level 2, the combined function automation means that there are at least two primary functions that the vehicle can control by itself and simultaneously, while the human controls the rest. At level 3, limited self-driving automation occurs, which means that the car senses when the driver should take control of the car based on the safety conditions. At the last level, level 4, the vehicle drives without human control, which is required for the challenge the team are undertaking. The vehicle made within the team should be able to navigate the obstacle course by itself by sensing the changes in the environment around it *(User: 122.15.201.20, last modified 15th May 2016*).

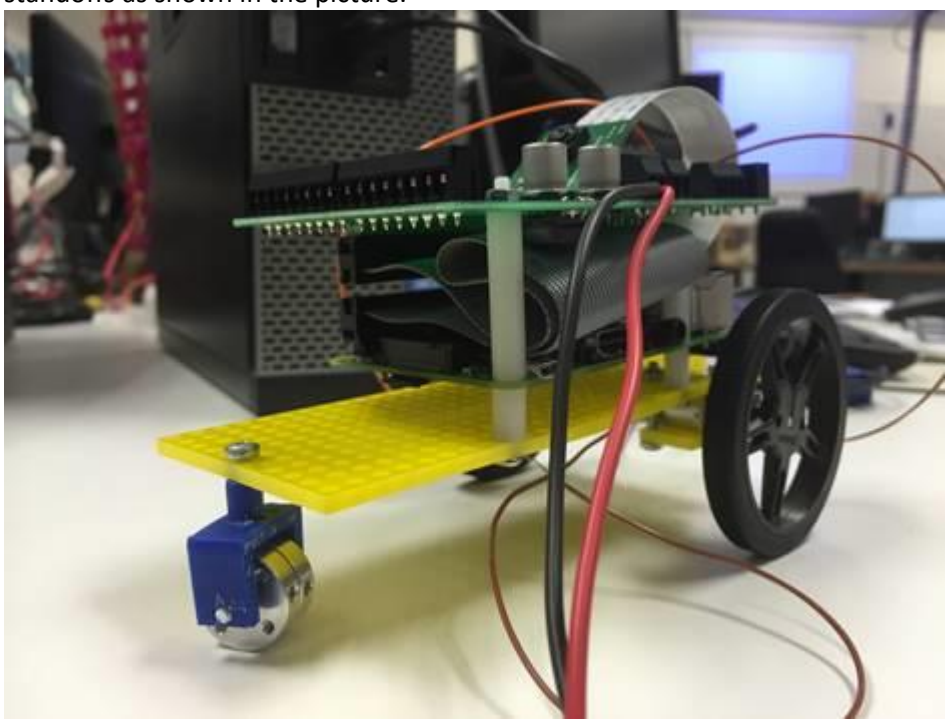
This challenge is interesting as self-automated cars are now closer to becoming a reality and so our little vehicle gives as a small scaled view of how these cars work. Testing on scaled autonomous cars started at the very least in the 1920's picked up in the 1950's *(User: 122.15.201.20, last modified 15th May 2015)* and have continued to the present day; so much that self-driving cars currently drive on in cities *(https://www.google.com/selfdrivingcar/)*.

Method

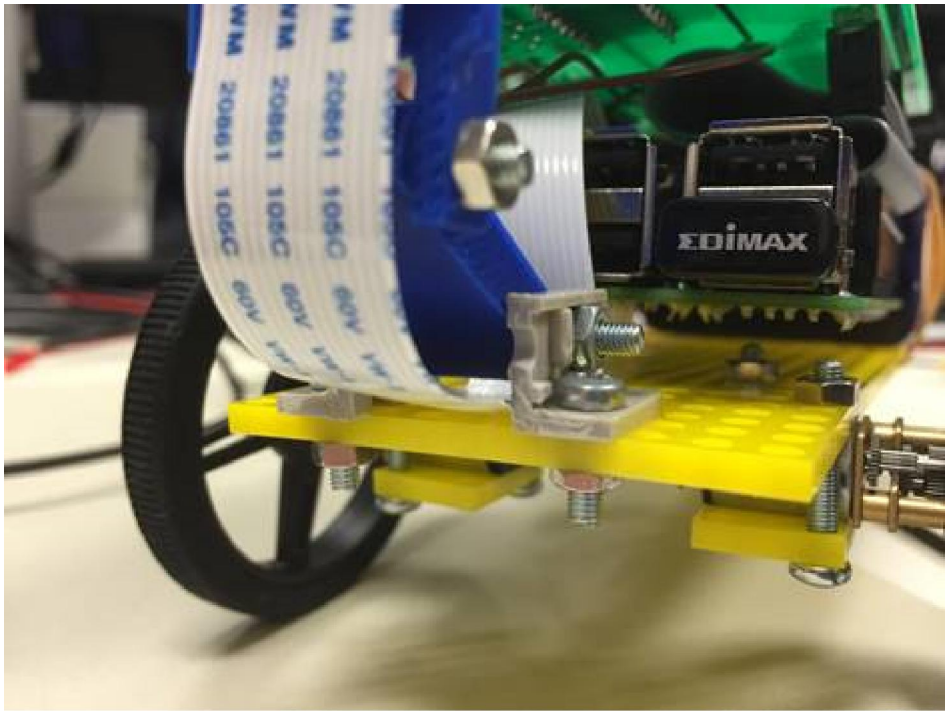
When we were first assigned groups, the first task undertaken was to choose a name, Team Epsilon, and then choose responsibilities for everyone. However, during the last week (week 4), the team realised that responsibilities had to be more shared around and not strictly stuck to the assigned roles so that each member is able to develop on each skill stated in the introduction. Therefore from then on, each member of the team will try and immerse themselves in different positions.

Hardware

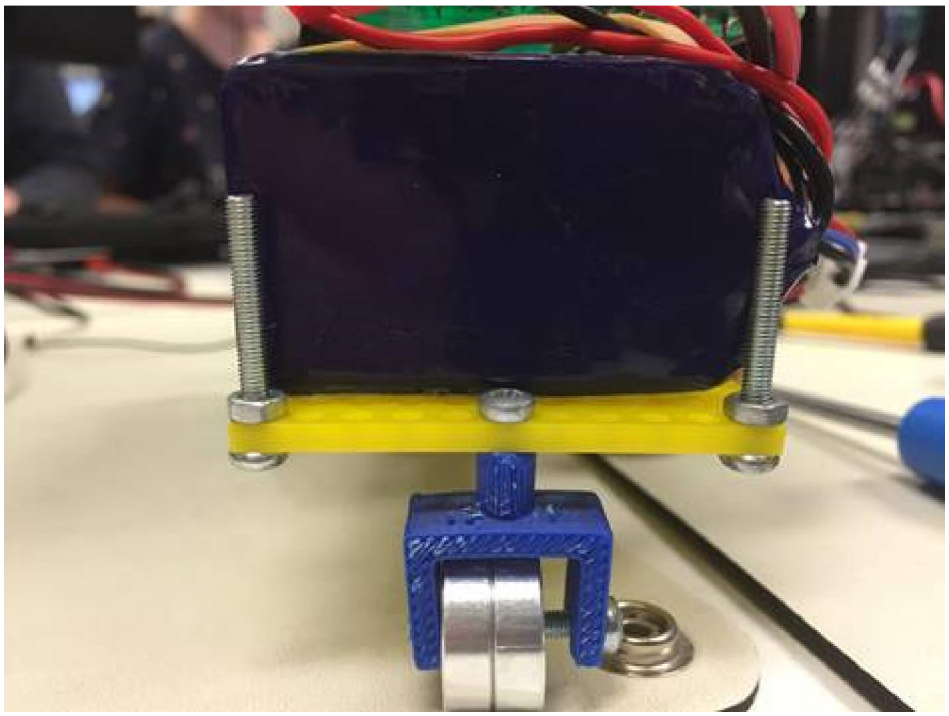
Team Epsilon started with the chassis and the equipment provided by Arthur and the hardware people started to attach the Raspberry Pi onto the chassis. It was placed at the front on the side of the two big wheels so that there would be enough support for the weight, and so that there would be room for the battery to fit on our vehicle. The boards were connected together by white standoffs as shown in the picture.



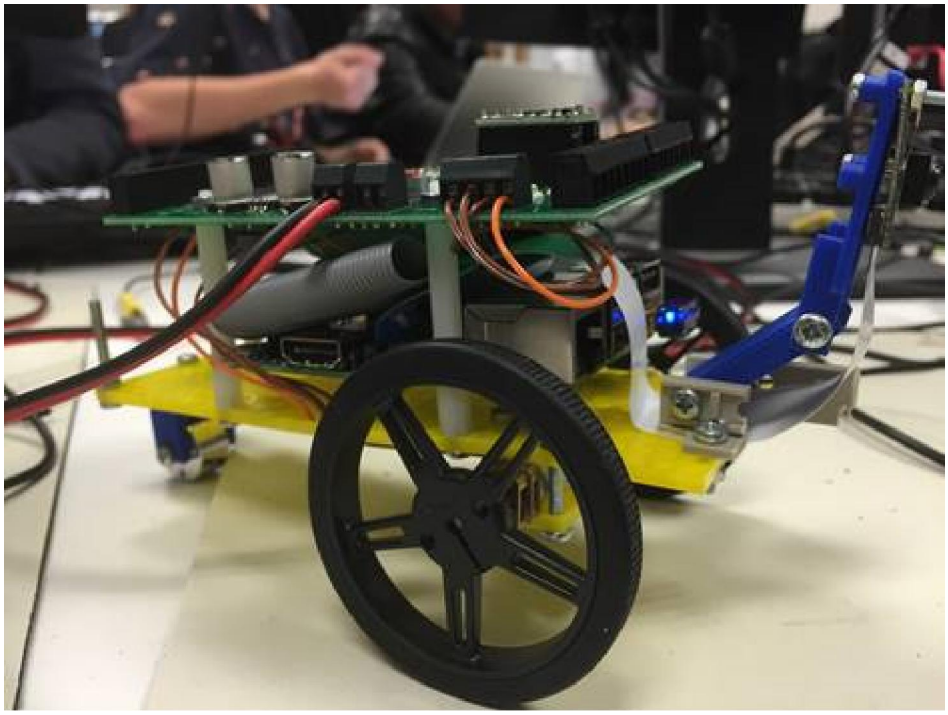
During week 4, a camera holder was added that was chosen because its design was supplied and it was easy to change how much it was angled to the maze surface. Therefore, it is simple to manipulate how much light that we want entering the camera. This was placed in on the side with the big wheels, as that is the front of the robot, the team obviously wants it to see ahead to see where it should travel by recognising and following the white line. It sticks out from the main body of the vehicle as it takes time for the pictures to be processed. Since this holder has been added, nothing else has been 3D printed and added onto the vehicle.



At the back of our autonomous vehicle, is where the battery is placed when the vehicle is manoeuvring the maze. The team decided to have it just above the back wheel and it is supported by screws and a rubber band.



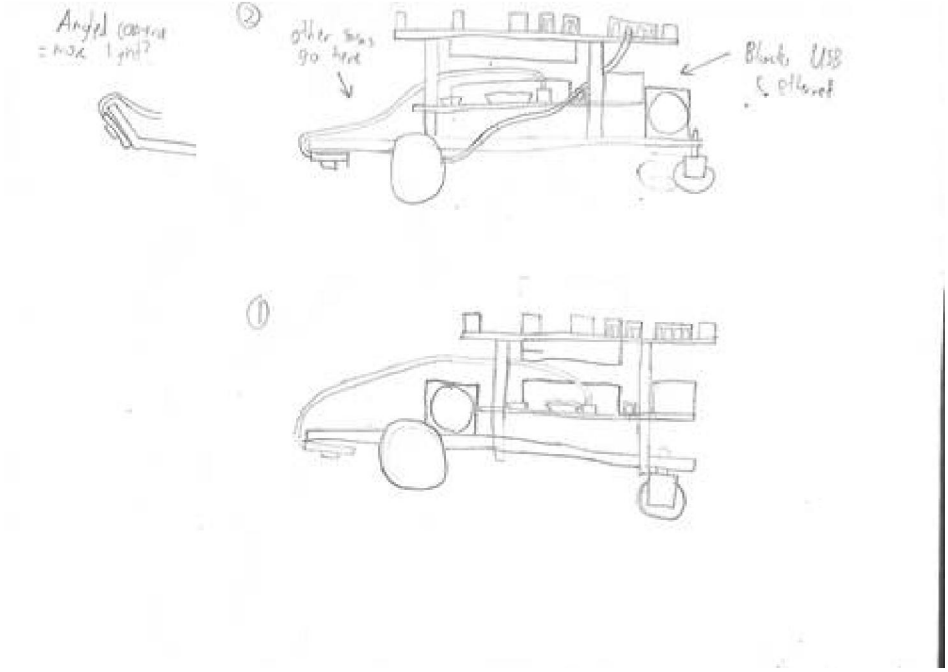
However, during testing in week four, the robot lost the line and fell off the side of the maze, so one corner that held the screw broke off and is yet to be fixed.



The vehicle has wires looped between the boards for engineering aesthetic reasons; as loose wires aren't visually pleasing and can be easily damaged during testing by being caught on things or becoming detached as it falls off the table.

The fixed bigger wheels were chosen to be in the front as the vehicle would have more control over its movements. However, admittedly they were mainly chosen because of aesthetic reasons. But the big wheels are ideal to be chosen over the smaller ones, as a bigger wheel has a greater circumference than a smaller wheel so that there is more distance is travelled in one revolution. This is ideal for Team Epsilon as our group wanting our robot to travel the maze in a short period of time.

The colours of our vehicle are chosen because they were the ones available and the ones that the main hardware people decided to use.



Our now hardwired vehicle is quite close to the initial design. The low and long design was based on racing cars to reduce the minimal air friction in the room so that the vehicle is able to achieve the quick time that the team desires.

Software
As briefly mentioned, Team Epsilon has been programming a Raspberry Pi and then using SSH. SSH lets the vehicle connect to the IP address and team members are able to gain remote access to files within the Raspberry Pi so code is able to be written and compiled without a wire connection to the computer *(User: wiae, last modified 1st May 2016).*

The linefollow.cpp folder includes the code in which the bot follows the line and this is what was first coded for because it was important to actually have the vehicle moving. PID error checking works alongside this code. This error checking ensures that the vehicle stays on the line if it detours slightly by using the three values working together to correct the vehicle's path of travel. For turning, the vehicle doesn't slow down one wheel, but instead speeds up the other.

As all of this is happening, the team has coded the camera to take photos. Every time the main loop loops, which is the main.cpp true loop, it takes a picture. Therefore when it has finished processing one picture it takes another. This is essential so that the robot keeps on moving through the white line.

Networking was the main focus of our attention first three weeks, as if the team wasn't able to achieve a working networking code, the robot wouldn't have even been able to enter the first quadrant. However this was successfully coded by the end of week three and tidied during week four. This code lets the vehicle establish and maintain a connection with the gate by connecting to the server, and asks the gate to "Please" send the password back to the vehicle, to which it then sends back to the server and the gate opens.

6 lines (4 sloc) | 64 Bytes

```
1  #ifndef __GATE_H__
2  #define __GATE_H__
3
4  void open_gate();
5
6  #endif
```

https://github.com/lovemich/AVC_Team-Epsilon/blob/master/src/gate.h

Our original pre-tidied code was much longer as t formatted the reply from the server. However, this wasn't needed so the code got drastically shortened.

The code for Quadrant 3, 4 hasn't been written yet, and this will be the oncoming week's goals.

Results

During initial testing, the vehicle would veer from the white taped line and continuously ram into the maze wall next to it. It was concluded this was because the camera holder was angled too far up and it was processing the white of the wall as the line it had to follow. This may be a reason why the vehicle drove off the table as well. Also, during initial testing, the vehicle would start immediately as the gate would accept the password given. This meant that the vehicle would drive into the gate and keep on doing so until the gate was high enough to let it through. To account for this happening, there was code written for the vehicle to lag behind the gate.

Now, Team Epsilon's autonomous vehicle is able to successfully open the gate , complete Quadrant 1 and 2 and has now entered Quadrant 3, but isn't able to navigate the best possible pathway and isn't yet able to go backwards once it reaches a dead end. This is a video of the vehicles progress thus far:
https://drive.google.com/file/d/0B-41OE09a9DweHdhMXROZDN2amM/view
It's a bit jerky, but that may be due to the speed that is currently travelling at. Further testing is needed to be done to find the actual reason and possible solutions.

The hardware always worked, but now the vehicle may have an issue stabilising the battery as one of the corner pieces broke off during the most recent testing.

Discussion

So far personally I haven't done much in aspects of coding, but in the coming weeks I have been set the task of coding for the maze dead end so that the robot is able to back up. With this task, I'll be able to sit down with the assigned programmers and learn about the code as I am getting help to write. So far we are fairly on time with our weekly and personal goals and tasks. But now that the team has progressed further into the maze it may take a bit longer to complete tasks and more help will be needed of each other which will cause us to interact even more with each other.

Conclusion

Our team has worked well so far and communication is effective. Another rhalf of the maze to go and will have to add sensors and other possible additions onto our vehicle by using our "Arthur dollars". During this challenge, it has been great working with people I am unfamiliar with as this is what a real life situation will be like. There will be situations where I'm working in a team with people I'm new to and in order to complete our object we will have to communicate effectively with each other to discuss ideas. So far we have stuck closely to the clients expectations and are well on the way to fulfilling our personal group goal of having a fast vehicle. This challenge is a great way of solidifying lecture ideas and applying them in real situations. This helps deepen understanding.

- Negative
- Positive

Referencing

User: 122.15.201.20 (2016) *Autonomous car*
Retrieved from: https://en.wikipedia.org/wiki/Autonomous_car

Google Self-Driving Car Project
Retrieved from: https://www.google.com/selfdrivingcar/

User: wiae (2016) *Secure Shell*
Retrieved from: https://en.wikipedia.org/wiki/Secure_Shell

Eldridge, E. (2016)
Retrieved from: https://github.com/kaiwhata/ENGR101-2016