Project report - AutoPylot

Alexandre Girold
Mickael Bobovitch
Maxime Ellerbach
Maxime Gay

Group: Autonomobile

March 2022

Contents

1 Introduction

1.1 Project presentation

Autonomous vehicles and more specifically self driving cars have grasp the attention of many people for good or ill. In this spirit, we have decided with the Autonomobile team to create our first ever project, AutoPylot. The name of our team is of course full of meaning in that regard. Autonomobile is a two-word name, the first one a french word for autonomous : "Autonome", the second one a french word for car : "Automobile". These two-word combined literally mean Autonomous car.

What is Autopylot's goal ? Drive itself on a track and win races. It may, at first glance seem very simple but not everything is at it seems. Yet we will try to make it as easy to undestand as possible, without omiting crutial information. To achieve our goal, we need to solve many other problems. Those problems can be separate into two distinct groups.

The first one would be the software part. Indeed in this project we will need to learn and acquire certain skills, from teamwork to coding in different languages. With those newly acquired skills we will be able to bring machine learning to our car to make it drive itself. This leads use directly to our second part, the more tangible one : hardware. Indeed, as we will progress in our work, we will need to see the results of our work in real life condition. This means implementing our code to a functioning car which will be able to race on a track.

This project will lead by a team of four young developers, Maxime Ellerbach, Mickael Bobovitch, Maxime Gay and Alexandre Girold. In this project work will be divided equally amongst all of us, sometimes we will have to work together to achieve our very tight time frame.

1.2 Team members

1.2.1 Maxime Ellerbach

I am a curious and learning hungry person, always happy to learn and collaborate with new people ! Programming, robotics and tinkering has always attracted me. Writting code and then seing the results in real life is something that I find amazing ! I had multiple projects in this field : Lego Mindstorms, a robotic arm, more recently an autonomous car and even a simulator in unity to train even without a circuit at home ! Even if I know quite well the domain of autonomous cars, there is always something new to learn. I look forward working with this team full of hard working people on such a fun project !

1.2.2 Mickael Bobovtich

Roses are red. Violets are blue. Unexpected “Mickael BOBOVITCH “ on line 32. Hello I am a French Student with Russian parents. Lived half of my life in Moscow. Passionate in web dev, servers, and business. Started programming at 13 years old. Created many projects. I like to learn everything, from AI, to UI, from Hardware to Software. Actually I am like OCaml, you need to know me well to appreciate me.

1.2.3 Maxime Gay

I am 18 years old, and I am crazy about investment, finance and especially cryptocurrencies and blockchain. I already worked with a team on different Investment projects and during summer Jobs but this is the first time that I am working on such a project. Furthermore, I am a beginner in computer Science and autonomous car. However, I am impatient to learn new skills with this incredible team.

1.2.4 Alexandre Girold

I am already getting old. I am 19 years of age, yet I am full of ressources. I am delighted to be able to learn someting new. There are many things which I enjoy from programming to geopolitics. I know this project will push me toward a better me and make great friends along the way.

1.3 State of the art

In this section, we will try to see what was previously made in this sector of industry. It would not be realistic to compare our 1:10 project to real sized cars such as Tesla's, simply because in a racing environnement, we don't need to deal with such an amount of safety: pedestrian detection, emergency braking, speed limit detection and other. So we will only see miniature autonomous racing framework that we would likely race against.

The most known is called "DonkeyCar", created by Will Roscoe and Adam Conway in early of 2017. Most of the models trained with DonkeyCar are behavior cloning models, meaning models that tries to replicate the behavior of a driver. This methods uses a big amount of images (input) associated to steering angles and throttle (output), it requires the user to drive the car (collect data) prior to training the model: no examples means no training. The lack of training data often leads to the car leaving the track.

One other framework worth looking at is one created by Nvidia called "JetRacer" released in 2019. It uses a different approach from DonkeyCar where the user annotates the images by hand by clicking on where the car should go. The model used is similar to what DonkeyCar uses: a Convolutional Neural Network with one input (the image) and two outputs, one for the steering angle and one for the throttle to apply.

Both of those framework are written in python and use packages such as Tensorflow and OpenCV, we will also use them in our project.

2 Realized tasks

2.1 Project Setup

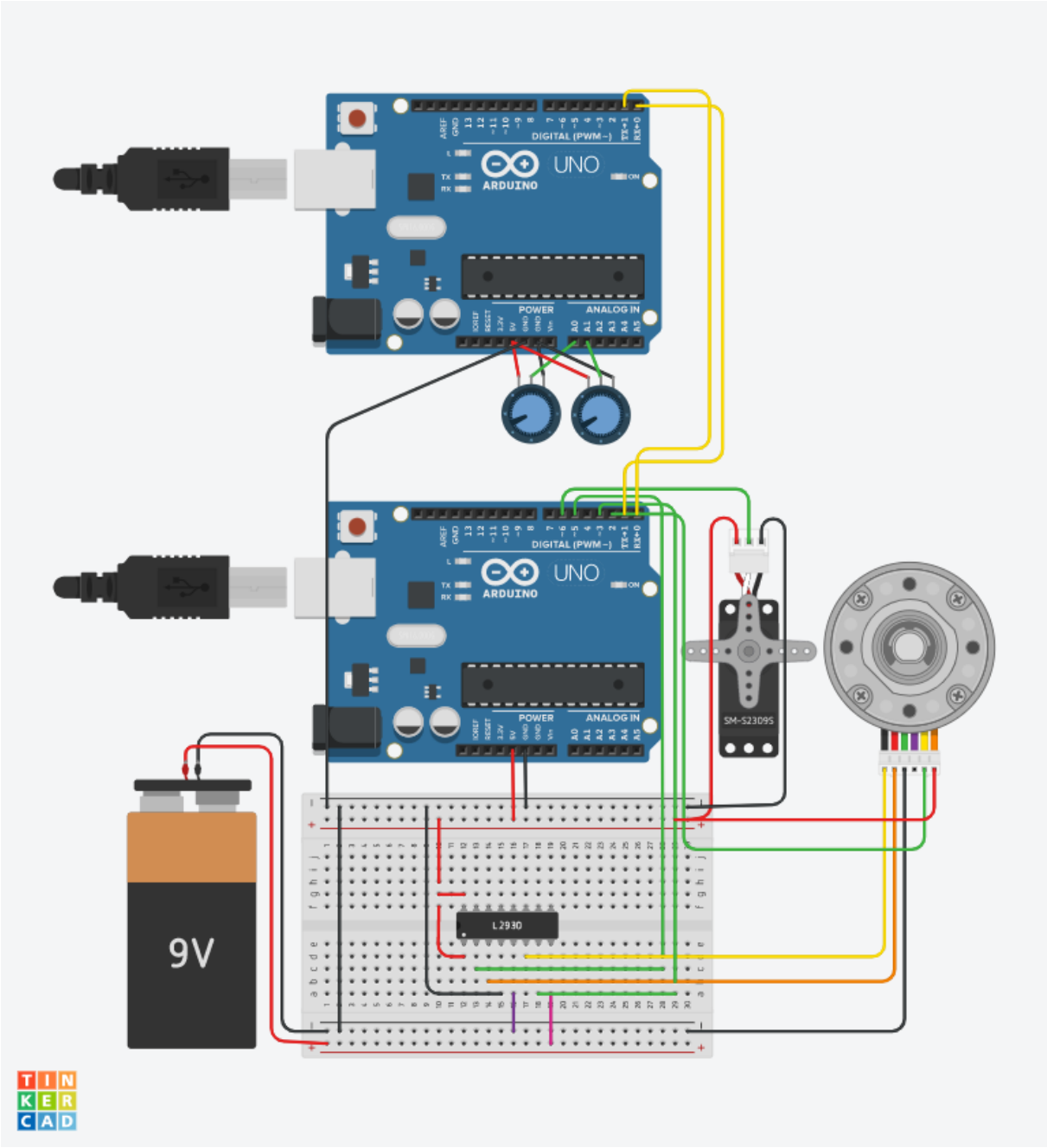
Prior to start programming, we had to setup a good Github repository. First, we created a new Github Organization called Autonomobile, this is where all of our repositories are located. We then created the AutoPylot repository. We decided to put everything related to this project in the same repo for simplicity, so we have a repo divided into four main parts:

- The autopylot python module. While creating it, we searched online for best practices regarding python packages. We learnt a lot regarding that part !
- Some main scripts that use autopylot module.
- Everything related to the presentation website and telemetry website.
- Documentation ! Even if it is not the funniest part of the project, it is still a really important one: keeping track of how to install the project, our dependencies and so on. We also keep in this part every project report and work we had to do for the presentations as we may need them in the future !

We also created a Github action to automate the testing of our project ! Every important functions coded is accompanied by its set of tests. This enables us to assert that the code we are currently working on works as expected. With this idea in mind, we added a really important rule: no addition of code on the main branch if it doesn't pass all the tests we wrote. This rule is really important to keep main clean from any major bug ! This rule induces one thing: we need to open branches and do Pull Requests for every features we add to the project. Moreover, we need the approval of someone else to merge the Pull Request into the main branch . We all forced ourself to respect this as much as we could, the result is that now we are totally used to this process and everyone is aware of what people are working on because they need to review their code. On top of that, before committing code, we run a code linter that clean our code, adds and removes whitespace where they should or should not be, rearranges long lines so that they fit into the screen.
All in all, we are confident to say that we came up with a great set of rules and a great project structure to avoid spending time on solving issues that could arise from a poor project setup.

2.2 Arduino and car control

This task is essential to the success of this project, this is the lowest code level we will deal with. The arduino code is the code that drives our motor and servomotor, without this part nothing works ! Any issue coming from this part of the code would mean a crash into a wall. To ensure that this part was working as expected, we run a lot of testing with the real car, trying different scenario to see what to expect for example if we lost the connection between the arduino and raspberry pi or if it didn't receive orders for a given amount of time. As all the team could not have direct access to the car, we did a virtual clone of the car using TinkerCad, simulating and live debugging our code before feeding it to our real car !
Here is the TinkerCad simplified version of the car:



You can see here two arduinos, but on the car we only have one, why is that ? TinkerCad didn't have a raspberry pi, so we added an other arduino to simulate the serial connection between the raspberry pi and the arduino. The arduino on the top sends steering and throttle information (as the raspberry pi would do) to the other arduino that processes those information received on the serial port and then controls both the servomotor and motor that are running on a 9V battery here but in real life on a 7.2V battery. This virtual simplified car really helped us in the process of developping the core of our car !

When the arduino part was finished, we could start the python part, that consisted in sending the right bytes to the serial port connected to the arduino, it was hard to debug when there were issues on the one hand, but really satisfying when it did work ! We are now able to send steering and throttle to the arduino making the car controllable using python !

2.3 Camera

The next step was to build a class to fetch images from our webcam, thankfully, the python module opencv has already some functions to do exactly that ! We only needed to add a wrapper around all of that to match our needs. We are now able to fetch images from our camera into a numpy array that we can manipulate. On top of that, we did create a 'dummy' version of this camera class in case we did not have access to a camera to run some tests, this class return a black image when we grab a new image form it. We use this class in some of our automated tests.

2.4 Load and save data

2.5 Data set

2.6 Data vis

2.7 Basic car loop

2.8 Creation of the logo

2.9 Realization of t-shirt

2.10 Website

3 Planning

3.1 Races

Tasks	Race 1	Race 2	Race 3	Race 4	Race 5	Race 6
Code controlled motors and servo	75%	100%				
Drive the car with a controller	25%	100%				
Data collection		50%	100%			
Telemetry website		25%	100%			
Data processing and augmentation			50%	75%	100%	
Basic Convolutional neural network			25%	50%	100%	
Advanced models and optional objectives						50%

3.2 Presentations

Tasks	1st presentation	2nd Presentation	Final resentation
Code controlled motors and servo	100%		
Drive the car with a controller	100%		
Data collection	75%	100%	
Telemetry website	25%	100%	
Presentation website	100%	Update	Update
Data processing and augmentation		75%	100%
Basic Convolutional neural network		50%	100%
Advanced models and optional objectives			50%

4 Task allocation

Tasks	Mickael B.	Maxime G.	Alexandre G.	Maxime E.
Low level car control		x	x	
Driving with a controller		x	x	
Data storage and handling	x	x	x	x
Telemetry website	x			x
Presentation website	x			x
Convolutional neural network	x	x	x	x
Main control loop		x	x	x

5 Conclusion

To sum up, Autonomobile team improved the control of the car with controller, furthermore the data processing is working flawlessly allowing us to load and save images and metadata. Moreover our presentation website is ready, it includes the presentation of the team, some links to download our project and even a road map. Nevertheless, the hardest part is yet to come, indeed we have to work on the AI part of the car and on the telemetry website to have it working by the next project defense. // The telemtry website is important in order to visualize data to know what is happening inside the car at any moment. We well have to work and learn a lot on this stopic which is really interesting. //

To make a long story short, we spent lot of time on this project, which comport many different sections that are important for the realization of this project and we will do every thing to succed.