1 Introduction

This report will provide an overview of the progress and activities completed during the second month of the competition. We have almost finished lane detection, although it still needs some fine-tuning, especially regarding the definition of ROIs and pre-processing. We also started working on intersection detection and coded the algorithm for PID control, although we didn't manage to implement it in practice for lane following.

2 Planned activities

1) Completely installing the physical testing environment with all its parts
2) Finishing lane detection, lane following, and speed control
3) Intersection detection
4) ROIs definition and camera pre-processing
5) Defining project architecture and communication between packages
6) Installing and defining the virtual testing environment

3 Status of planned activities

3.1 Completely installing the physical testing environment with all its parts

Status: Ongoing 80%

We have obtained the funding for printing the total 15x15 meters map but are still working out the details regarding space and the exact materials and format in which the map will be printed and used. One of the ideas is to divide the map into 100 modular pieces that are 1.5x1.5 meters in size and can be assembled as required. The map should start printing in the following week.
Difficulties: space and the size of the map

3.2 Finishing lane detection, lane following, and speed control

Status: Ongoing 70%

We are currently in the process of implementing an algorithm for detecting and following lanes. We calculated the middle line using the coefficients of the two lines our line detection algorithm returned. Then we implemented a classic PID control algorithm with the center line slope (k) deviation as the input and the car's steering angle as the output. Firstly, we initialize the desired k by having the car drive down a straight road. After initialization, we calculate the steering angle using PID for each frame. However, we still haven't tested this approach in the real world due to difficulties in setting up the testing environment and synchronizing the threads. We haven't started doing speed control.

3.3 Intersection detection

Status: Ongoing 20%

Our algorithm aims to accurately measure the distance between a vehicle and an intersection line by identifying and analyzing horizontal lines within the image captured by the vehicle's camera. The algorithm's core is identifying these horizontal lines (which represent the intersection) and measuring the distance from the center of each line to the vehicle position. We are using computer vision and image processing to implement these.

3.4 ROI definition and camera pre-processing

Status: Postponed

We have postponed the implementation of ROI definition and camera pre-processing as they were only necessary now. We intend to address these soon.

3.5 Defining project architecture and communication between packages

Status: Ongoing 15%

We send the data from Raspberry Pi to the remote PC and do all the calculations there. We still haven't defined the architecture of our project and the communication between containers because we haven't had that many synchronization issues up until recently. Still, we plan to move those directly to Raspberry Pi and define the architecture in the next few days.

3.6 Installing and defining the virtual testing environment

Status: Ongoing 25%

Line detection and line-following Python scripts were tested in an improvised physical climate because we couldn't install the Gazebo simulator due to inadequate laptops. We hope to get a computer with enough RAM by the end of January.

4 General status of the project

The car can detect lanes in real-time, but we still need to implement lane following based on that data.
We are generally not satisfied with the progress during the last month. We hope to progress more quickly once we acquire the physical map we will use for testing.

5 Upcoming activities

1) Improving lane following and implementing speed control
2) Enhancing lane detection and tackling ROI definition
3) Intersection detection
4) Traffic sign detection
5) Intersection navigation
6) Creating a physical and virtual testing environment
7) Defining path planning and validation