VROOM / auth

Virtual Route Optimization Mobility
Aristotle University of Thessaloniki

Bosch Future Mobility Challenge

ARISTOTLE UNIVERSITY OF THESSALONIKI

Technical Report #4

Authors

Theodoros Papafotiou [*Team Leader*]

Efthymia Amarantidou [*Team Member*]

George Koutroumpis [*Team Member*]

Nikolaos Tsachouridis [*Team Member*]

Responsible Mentor

Emmanouil Tsardoulias
Antonios Dimitriou

Date

October 1, 2021

1 Introduction

The current report summarises the technical progress of the team V.RO.O.M. (Virtual Route Optimization Mobility) in the Bosch Future Mobility Challenge 2021 until April, 18th, 2021. Particularly, the activities planned during the time-period between the 21st of February and the 18th of April 2021 and their current status, as well as the upcoming activities are briefly described.

2 Planned activities

The planned activities during the aforementioned period, additionally to the team members in charge for each activity are shown below:

- Control
  - **Lane keeping [Upgrade]**
    @Georgios Koutroumpis & Nikolaos Tsachouridis
  - **Roundabout navigation**
    @Georgios Koutroumpis & Nikolaos Tsachouridis
  - **Vehicle Overtake**
    @Efthymia Amarantidou
  - **Parking procedure**
    @Theodoros Papafotiou
- System integration
  - **Finite State Machine (FSM) implementation [Upgrade]**
    @Theodoros Papafotiou
  - **Code integration and porting to real car [Upgrade]**
    @Theodoros Papafotiou

3 Status of Planned Activities

3.1 Lane keeping [Upgrade]

3.1.1 Implementation

Since we were not satisfied with the efficiency of our previous algorithm, which involved calculating angles from the detected lines' slopes, a new approach has been implemented. In this implementation, the image is warped and processed in order to maintain only the side lines, which are then detected by utilizing a histogram-based algorithm. Next, we perform a polynomial fit to each line and:

a. in case both lines are detected, the average of their x-positions (setpoint) is compared to the car's center.
b. if only one line is detected, its slope is utilized, so as to be able to successfully navigate the car throughout the map.

3.1.2 Issues

The main issue we faced was to properly map the error between the setpoint and the vehicle's center to the appropriate steering angle.

3.2 Roundabout navigation [Done]

3.2.1 Implementation

The roundabout procedure initiates by detecting the roundabout sign. When entering the roundabout, the car follows a predefined procedure of regulating its speed and angle to enter the roundabout with the appropriate yaw. Depending on the desired exit, it combines its lane keeping algorithm along with a module that utilizes its yaw to keep the car in the roundabout when it cannot see the roundabout's lines.

3.2.2 Issues

The main issue we faced was trying to compensate for the areas where the lanes were out of the camera's field of view, resulting in the car trying to exit the roundabout at the wrong exit.

3.3 Parking Procedure [Done]

3.3.1 Implementation

In regards to the parking procedure of the car, both the horizontal and the vertical parking maneuvers have been implemented. After detecting the "Parking Spot" traffic sign, the vehicle enters the parking-procedure mode, only if no other restriction prevents the car from doing so. For both maneuvers, the procedure is almost identical and is described below:

Horizontal parking

- Detect parking sign
- Move forwards to the correct position
- Turn right and move backwards
- When a desired yaw is achieved, turn left and continue backwards until the car is parallel to its initial direction
- Minor corrections while in parking spot, based on the lines in front of the car
- To exit parking, turn left and move forward
- When a desired yaw is achieved, turn right and continue forward until the car is parallel to its initial direction

Vertical parking

- Detect parking sign
- Move forwards to the correct position
- Turn slightly left and move forward
- When a desired yaw is achieved, turn right and move backwards until the car is perpendicular to its initial direction
- Minor corrections while in parking spot
- To exit parking, move a bit forward
- Turn right and continue forward until the car is parallel to its initial direction

3.4 Static vehicle overtake [Done]

3.4.1 Implementation

In order to be able to react to any static vehicle that is on our car's way, our team has developed an overtake maneuver algorithm. Following the detection of the vehicle, the decision to execute an overtake maneuver is taken, after checking the appropriate conditions:

- **Distance from vehicle:** we calculate the length of the line connecting the point (x,y) of the car's center on the frame and the center of the bounding box where the leading vehicle has been detected, and compare it to a threshold.

- **Dotted road line:** using the provided GraphML map, we process the spatial information between our current node and the next node of our path, indicating the existence (or not) of a dotted line.

If an overtake maneuver is allowed, we interrupt the standard navigation and follow the below procedure:

- Turn left to the opposite lane until a desired yaw.
- Continue lane keeping on the opposite lane for a specific distance.
- Turn right until a desired yaw.
- Turn left until a desired yaw in order to return to the right lane.
- Continue lane keeping.

3.4.2 Issues

Although our algorithm is effective in some of the use cases, we need a more robust implementation, able to succeed in any place in the track. Currently, the Dubins path algorithm utilization is explored, among others.

3.5 Finite State Machine [Continuous Development]

The Finite State Machine (FSM) works as the "brain" of our system, since it combines all algorithm implementations and is responsible for the car's higher level decisions. The first working FSM was created for the Qualifications round and since then more features and behaviours are being added. The FSM is under continuous development, therefore, it will be finalized shortly after the finalization of our algorithms and it will be thoroughly tested.

3.6 Porting to real car [Continuous Development]

While the algorithms and the Finite State Machine are still under development, the updated code-files are being periodically uploaded on the RPI4 placed on the car. However, due to COVID-19 restrictions, which restrain us from going to the lab, this procedure takes place only remotely. Consequently, the algorithms/car cannot be tested on the track.

4 General status of the project

In general, the team continuously verifies the effectiveness of each implementation in the simulation. Unfortunately, due to COVID restrictions and limited access in our car & track facilities, delays on the car testing have been occurred. Currently, the algorithms of the team are being tested in various conditions, some upgrades are being implemented on specific parts of our architecture and the Finite State Machine of the system is under continuous integration. Finally, we are in the process of implementing unit, functional and integration tests applied in the simulation, so as to automatically be certain of our software's soundness, preparing for the pre-finals of the BFMCC.

5 Upcoming activities

According to the suggested timeline of the team, during the period between the 18th of April and the end of May, the upcoming activities are:

- Control
  - Reaction to dynamic elements [Finalization]
  - Roundabout navigation [Finalization]
- System Integration
  - Finite State Machine [Finalization]
  - Software testing & Stress tests
  - Total system integration and preparation for semi-finals