1 Introduction

This month, due to the Lunar New Year holiday, we only have 2 weeks to work on the car, so we can only test lane following algorithm. Pedestrian, road signs and lights detection have yet to be deployed on real car.

- Improve lane tracking algorithm.
- Editing package and rewriting STM32 code on CubeIDE.
- Road signs, lights and pedestrian detection.
- Position fusion.
- Define decision making.
- Define path planning and validation.

2 Planned activities:

Activity Description	Members in Charge	Type of Activity
Improve lane tracking algorithm	Lâm, Trâm, Khôi	Development
Editing package and rewriting STM32 code on CubeIDE	Khanh	Development
Road signs, lights and pedestrian detection	Khôi	Development
Position fusion	Minh	Development
Define decision making	Lâm	Development
Define path planning and validation	Minh	Development

3 Status of planned activities

- **Improve lane tracking algorithm**
- **Status**: Ongoing 50%
- **Implementation**: The algorithm works by identifying the point sets for the left and right lanes. It then uses polynomial fitting (polyfit) to determine the functions representing each lane. Based on these functions, the middle point set of the road is calculated, and the angle is determined using the middle point set.
- **Difficulties**: There are many exceptional cases where the algorithm fails to determine the middle point set. To address these, we had to introduce numerous parameters for fine-tuning, making the code lengthy and difficult to understand. Additionally, calculating the left and right lane functions requires significant computing resources and time, leading to unnecessary delays. To overcome these challenges, we plan to implement a neural network for lane-keeping

- **Edit package and rewrite STM32 code on CubeIDE**
- **Status**: Completed

- **Implementation**: Edit and add necessary data packages such as distance sensor for STM32 and Jetson Xavier. Rewriting code on STM32 to fit with CubeIDE.
- **Difficulties**: Rewriting code from scratch and handling strings in C is more difficult than those in C++.

- **Road signs, lights, pedestrian detection**
- **Status**: Ongoing: 75%
- **Implementation**: We are deploying Yolov5 model for detecting objects.
- **Difficulties**: Because of the data shortage and changes of enviroment, our model may
- produce poor accuracy on some situations. This can be solve by collecting more data
- to train model.

- **Position fusion**
- **Status**: Ongoing 70%
- **Implementation**: Replace the previous IMU (cause the old one can't be detected). Developed a process to read data from IMU and localization server, aiming to produce an accurate combined position
- Result: The achieved position accuracy is currently unstable and prone to errors.
- Difficulties:
    - Need a physical map to test the algorithm which is not always available.
    - Further testing is required to adjust the filter's parameters for optimal performance.
    - The result from real car is unstable and map scale is not correct with real map. Still need more time to optimize.

- **Define decision making**
- **Status**: Ongoing 50%
- **Implementation**: We build a state flow that describes what should the car do when it counters each obstacle on road.
- **Difficulties**: Hard to decided which action should be prioritized as there are many situations.

- **Define path planning and validation**
- **Status**: Ongoing 30%
- **Implementation**: Research path planning algorithms designed for dynamic environments, capable of generating new paths in real-time.
- **Difficulties**: Identify an algorithm that meets the requirements and can be integrated with the existing system.

4 General status of the project

The car can now detect lanes and maintain its position on both straight and curved roads. It can identify intersections but is not yet capable of crossing them. The algorithm still needs improvement, as the car occasionally steps out of the lane.

5 Upcoming activities

In the next month we will do the following things:

- Preprocessing, noise cancelling for distance sensors.
- Define objects properties file.
- Environmental server interaction.
- Define robustness and safety measures.
- Simpler action-taking maneuvers.