Software Design

1. Overall design

Our software project was developed using Visual Studio Code with Arduino plugins and tested on Arduino IDE version 1.8.19. Since we used our own more familiar, cheaper and more powerful ESP32 dev board, building our software project required a manual installation of espressif/arduino-esp32 in the Arduino IDE.

Our software engineering project follows the principle of modularity and abstraction. `bupt_car_2.ino` is the main project file, `args.h` contains the parameters that often need to be adjusted (such as the various weights of the PID algorithm), the `dep` folder contains most of the concrete implementation code, and `lib` contains the third-party libraries needed for the components.

In development, we use Git for feature development and integration, and host the entire project on dannyHallo/bupt_car_2 to facilitate team member collaboration.

2. Linear sensor readout

Implemented in `ccd.h`. First, the CCD linear sensor is timed to expose and read, after which the threshold is set according to the current overall environmental brightness, and the black and white zones are divided; finally, the trajectory location is derived according to the distribution of the black zones, or whether the station is encountered or the path is lost.

3. PID algorithm for heading and speed control

Implemented in `pid.h`, and called in `autotrack.h` and `motor.h`. Compared with the traditional linear control algorithm, the PID algorithm can achieve more accurate, rapid and stable control; specifically in our project, the control of heading is more elegant and the control of speed is more stable.

4. Color recognition

Implemented in `color.h`. Although the hardware we use, the GY-33 module, supports direct output of the recognized color types, in order to be accurate in all lighting environments, we use the raw RGB data provided by the module and white balance it according to the ambient lighting.

5. Bluetooth remote control and data upload

The basic Bluetooth data sending and receiving is implemented in `bluetooth.h`. The upper layer functions are implemented in the specific module.
`commandParser.h` implements the Bluetooth remote control command receiving and execution, we developed an Android APP to send the command.
`autotrack.h` contains the function code to send data to send the read cargo information to the upper level via Bluetooth.