5 Road sign indicating the operation

5.1 experiment description

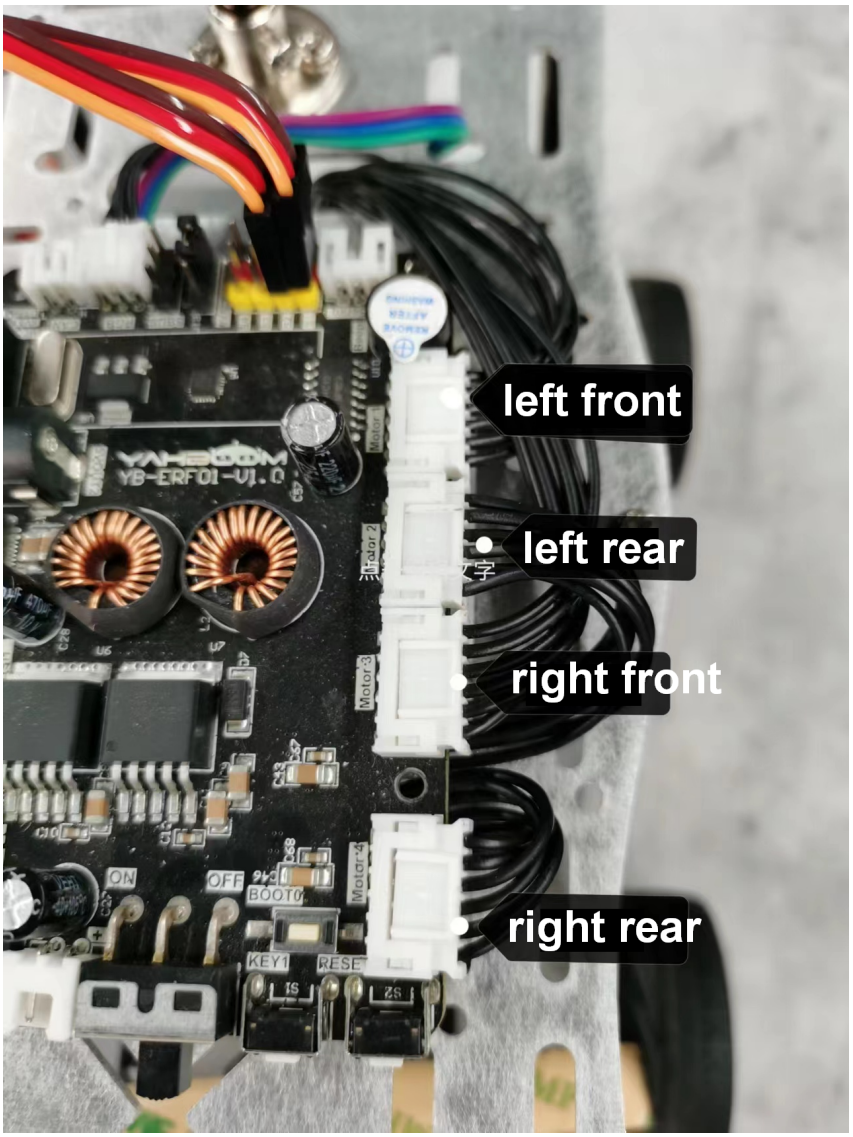
The present experiment was to belong to expand the class of experiments, the need to match other external devices to use, here to the car chassis and the ROS expansion Board is not part of K210 module kit contents, so the present experimental results are for reference only, if there is no corresponding device is not directly use the routine code.

ROS expansion Board needs advance programming firmware: ROS-CAR. hex

Due to the use of the motor voltage is 8. 4V, so the ROS expansion Board battery may not be inserted 12. 6V battery, you need to insert 8. 4V battery.

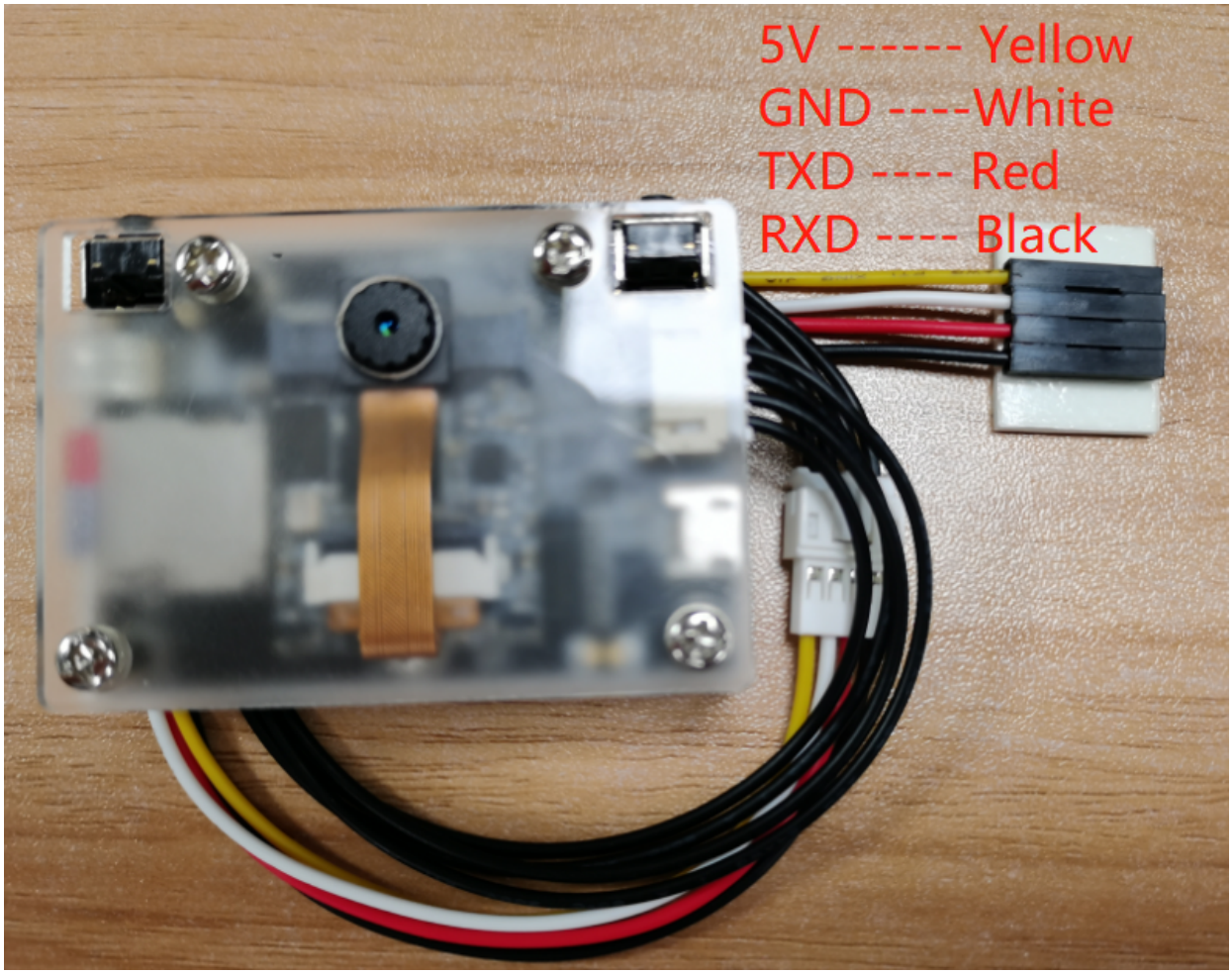
Trolley motor connected to the line as shown below:

Motor Motor 1 is connected to the Left Front wheel, the Motor Motor 2 is connected to the left rear wheel, motor Motor 3 is connected to the right front wheel, the Motor Motor 4 is connected to the right rear wheel.

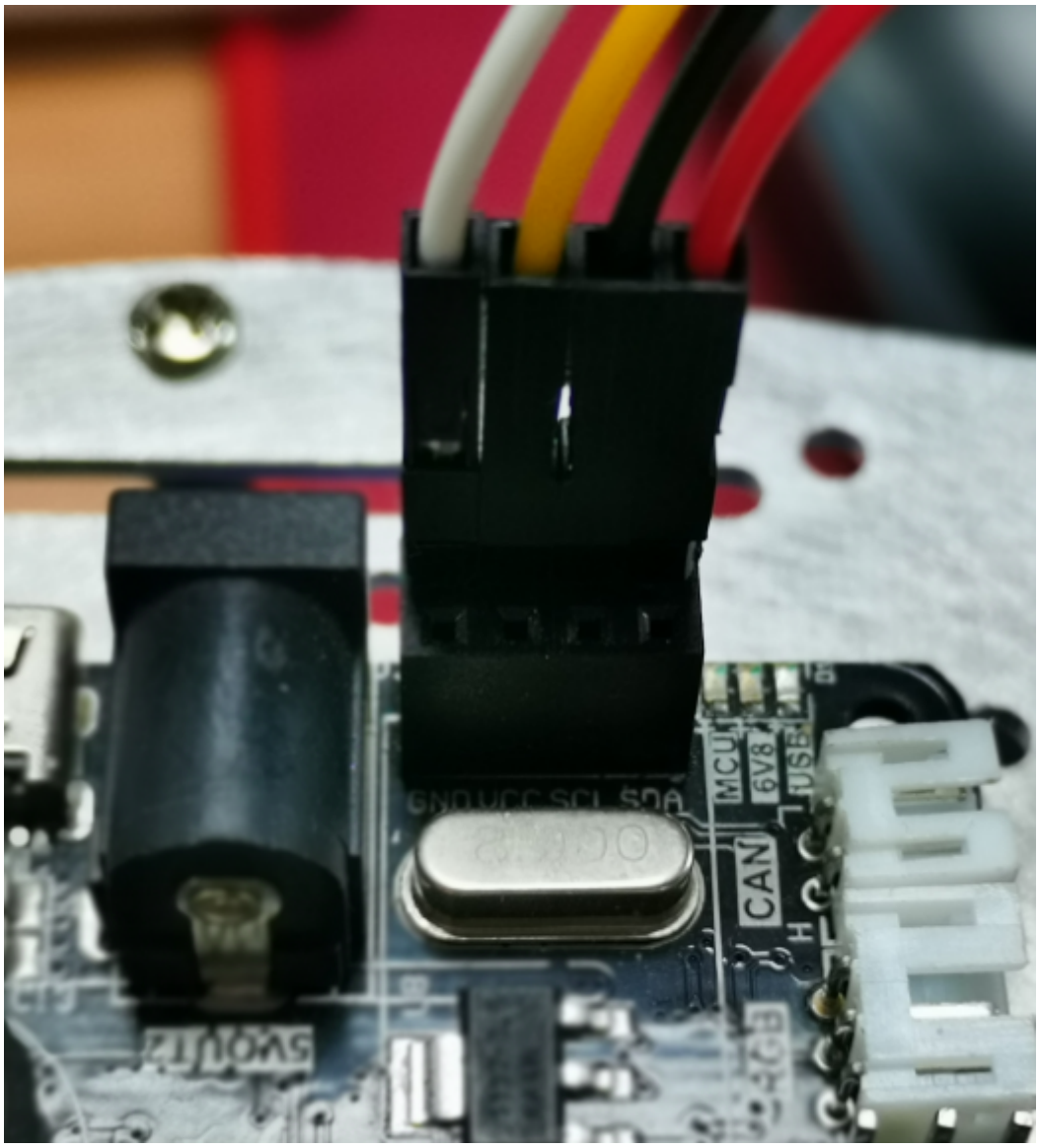


K210 module with the ROS expansion Board connected to the line sequence as shown below:

White(GND) connected to GND, yellow(5V) is connected to VCC, the black(RXD) connect the SCL, the red(TXD) connected to SDA.



Here you note that the illustration of the logo for the I2C line sequence identity, but K210 using serial communication, due to the burning of the ROS-CAR. the hex file has been put on this interface modification for the serial signal, so in fact the ROS expansion Board on the interface corresponding relationship is: the SCL is actually TX, SDA is actually RX.



5.2 experimental target

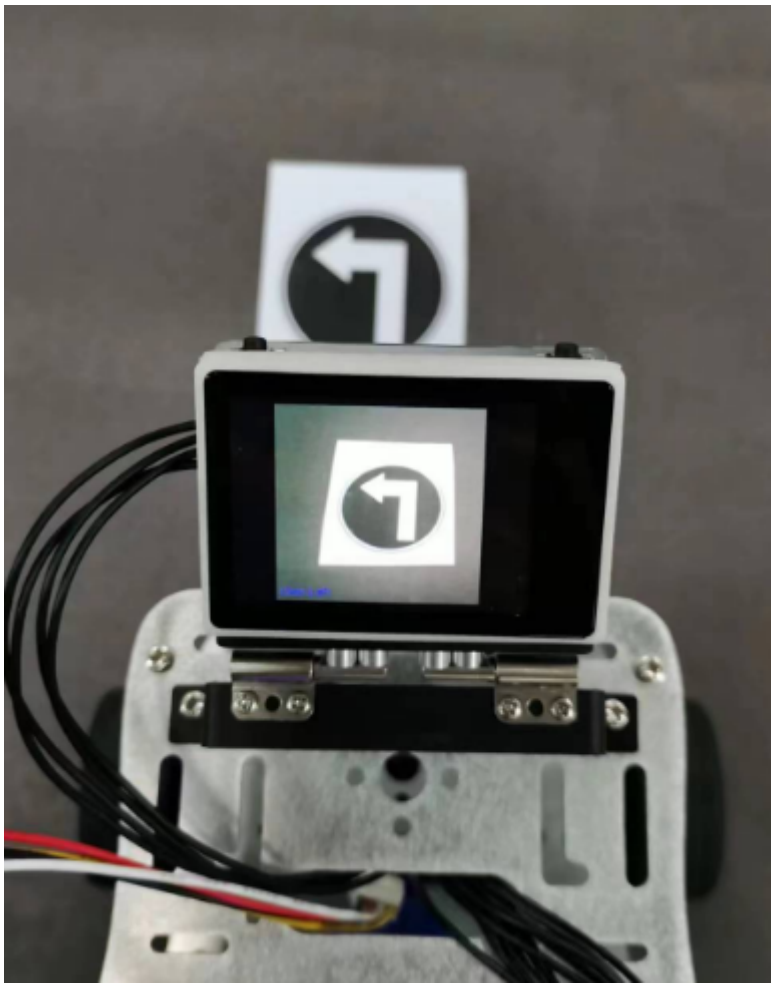
This lesson is mainly learning K210 module with the car chassis to do a visual inspection of the line features.

The reference code path for this experiment is： CanMV\06-export\sign_motion.py

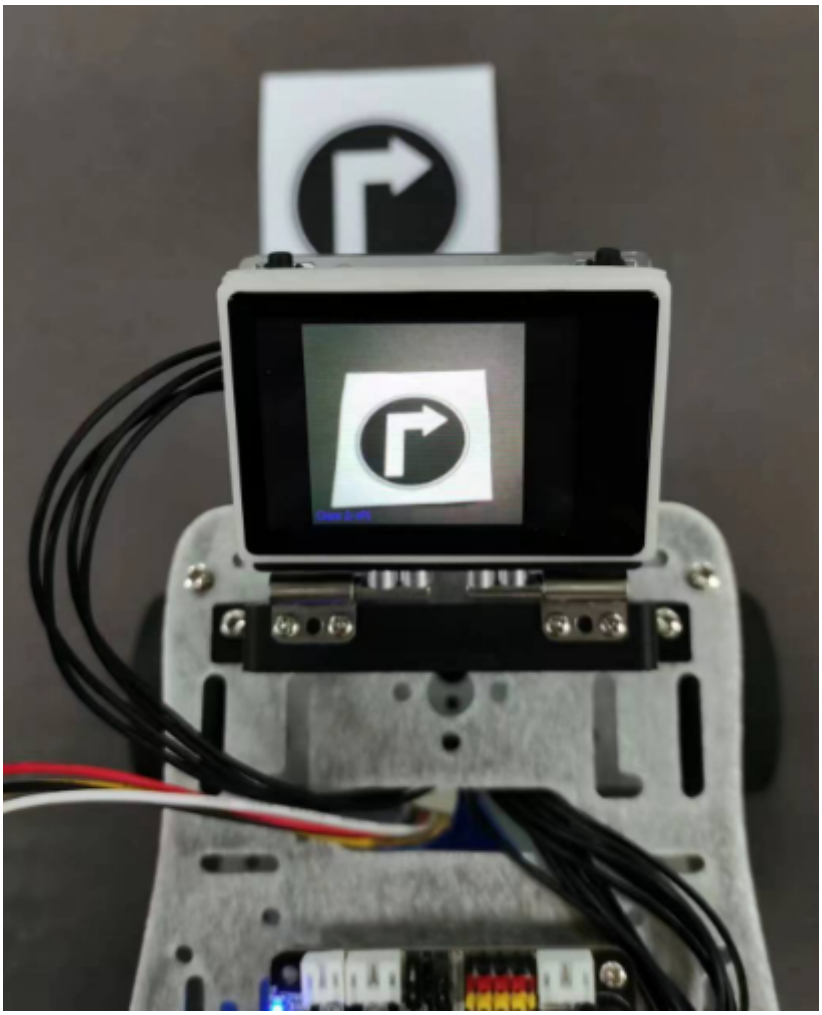
5.3 experimental operation

1. ROS expansion Board to burn the firmware: ROS-CAR. hex
2. Insert the RGB light strip into the RGB light interface of the ROS expansion board.
3. Please CanMV\06-export\library directory under the trolley driver library and PID control Library in advance to download to the memory card root directory.
4. Open CanMV IDE and open the sign_motion.py code and download it to K210 module.
5. The K210 module via the 4PIN cable is connected to the ROS expansion Board.

6. Put the trolley into the white or black background, snap the K210 module bracket to the appropriate Angle, and open the switch of the trolley.
7. First of all, you need to learn the left-turn icon. The operation steps are the same as the way of self-learning. Take five pictures of the left-turn icon according to the screen prompts.



8. Then learn the right turn icon and take five pictures of the right turn icon according to the on-screen prompts.



9. Next, learn about the stop icon and take five pictures of the stop icon as prompted on the screen.
10. After the learning is completed, the car starts to move forward, and when the corresponding icon is detected in the image, the corresponding action is executed.

5.4 experimental results

After waiting for the completion of system initialization, after completing the learning of road signs according to the above operation steps, the car begins to move forward. When a left-turn road sign is detected in the camera image, the left-turn function is executed, and then the car goes straight ahead. When the right-turn sign is detected, the right-turn function is executed, and then the straight line is taken. The cart stops when a stop sign is detected.

The magnitude and function of the turn can be modified in the 'car_control' function, where car_count indicates the time to turn and set_car_motion sets the speed of the turn.

```
def car_control(class_id):
    global car_count, car_state
    car_state = 1
    if class_id == 1:# 向左转 turn left
        car_state = 1
        car_count = 10
        bot.set_car_motion(0, 0, 3)
    elif class_id == 2:# 向右转 turn right
        car_state = 1
        car_count = 10
        bot.set_car_motion(0, 0, -3)
    elif class_id ==3: # 停止 stop
        car_state = 0
        car_count = 0
        bot.set_car_motion(0, 0, 0)
```

5.5 the experiments are summarized

The car landmark recognition function is modified based on the autonomous learning function of K210 module. The learning steps are similar to the steps of autonomous learning, first learn the left turn icon, then learn the right turn icon, and finally learn the parking icon. Because the movement effect of different cars on different maps is different, you can modify the content of the car's 'car_control' function to optimize the operation effect.