K210-Color follow

The tutorial mainly demonstrates the color following function of the balance car combined with the K210 vision module.

```
The tutorial only introduces the standard library project code
```

Hardware connection

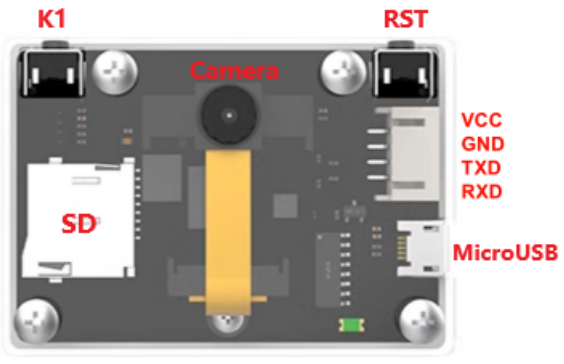
Peripherals	Development board
K210 vision module: VCC	5V
K210 vision module: TXD	PA2
K210 vision module: RXD	PA3
K210 vision module: GND	GND

Control principle

The K210 vision recognition module identifies colored objects, and sends the center coordinates of the identified colored objects and the width and height of the colored objects to the development board. The development board controls the balance car to follow based on this information.

```
The program mainly determines whether it needs to move forward or backward based
on the size of the identified colored object, and determines whether it needs to
turn based on the center coordinates of the identified colored object
```

- K210 Vision Module



```
The K210 Vision Module itself is a development board. For detailed usage, please
refer to the module supporting tutorial
```

> **Download the program**

Connect the SD card of the K210 Vision Module to the computer through a card reader, rename the program file to main.py and copy it to the SD card, then reinstall the SD card into the SD card slot of the K210 Vision Module.

```
The K210 vision module and development board case codes are in the same folder:
the folder name will distinguish the development board to which the code belongs
```

> **Color learning**

After the color recognition program is started, the K210 vision recognition module will display a box on the screen. We need to align the box with the colored object to be followed for color learning (the box needs to be completely inside the colored object); after the color learning is completed, the camera will recognize and mark the colored object with a box, and you can observe the recognition effect yourself (if the effect is not good, you can repeat this step).

> **Communication protocol**

The K210 vision module program will automatically calculate the center coordinates of the recognized object and the width and height of the colored object and send them to the development board for processing.

Data header	Data (X: X-axis data, Y: Y-axis data, W: box width, H: box height)	Data tail	Example
$	XXXYYYWWWHHH	#	$160120080050# Represents the center position (160, 120) Box height and width (80, 50)

Main code

The tutorial mainly explains the code for the K210 color following function. For detailed code, refer to the corresponding project file.

K210 color following uses multiple PIDs to control the balance car. If the balance car's patrol effect is not good, modify the PID parameters of the app_follow.c file. It is not recommended to modify the PID parameters of the pid_control.c file (the PID parameters of the pid_control.c file are subject to the parameters finally confirmed in the balance car parameter adjustment tutorial).

> **Turn_K210_PD**

Control the balance car forward PID.

```
static float GO_PID(void)
{
    static int16_t error,Last_error;
    static float k210go;
    error= g_error_area;   // Calculate the area of the deviated object

    k210go=GO_PID_KP*error+GO_PID_KD*(error-Last_error);// Position based PID
controller

    Last_error=error;     // Save the previous deviation
    return k210go;

}
```

Turn_PID

Control the steering PID of the balancing car.

```
static float Turn_PID(void)
{
    static int16_t error,Last_error;
    static float k210Turn,Integral_error;

    error=g_error_x;      // Calculate Deviation

    if(Integral_error>700) Integral_error=700;
    else if(Integral_error<-700) Integral_error=-700;
    k210Turn=-Trun_PID_KP*error-Trun_PID_KI*Integral_error-Trun_PID_KD*(error-
Last_error);  // Position PID controller
    Last_error=error;      // Save last deviation
    if(Turn_Off(Angle_Balance,battery)== 1)         // The motor is turned off
and the integral is reset to zero.
    Integral_error = 0;
    return k210Turn;
}
```

APP_K210X_Y_Follow_PID

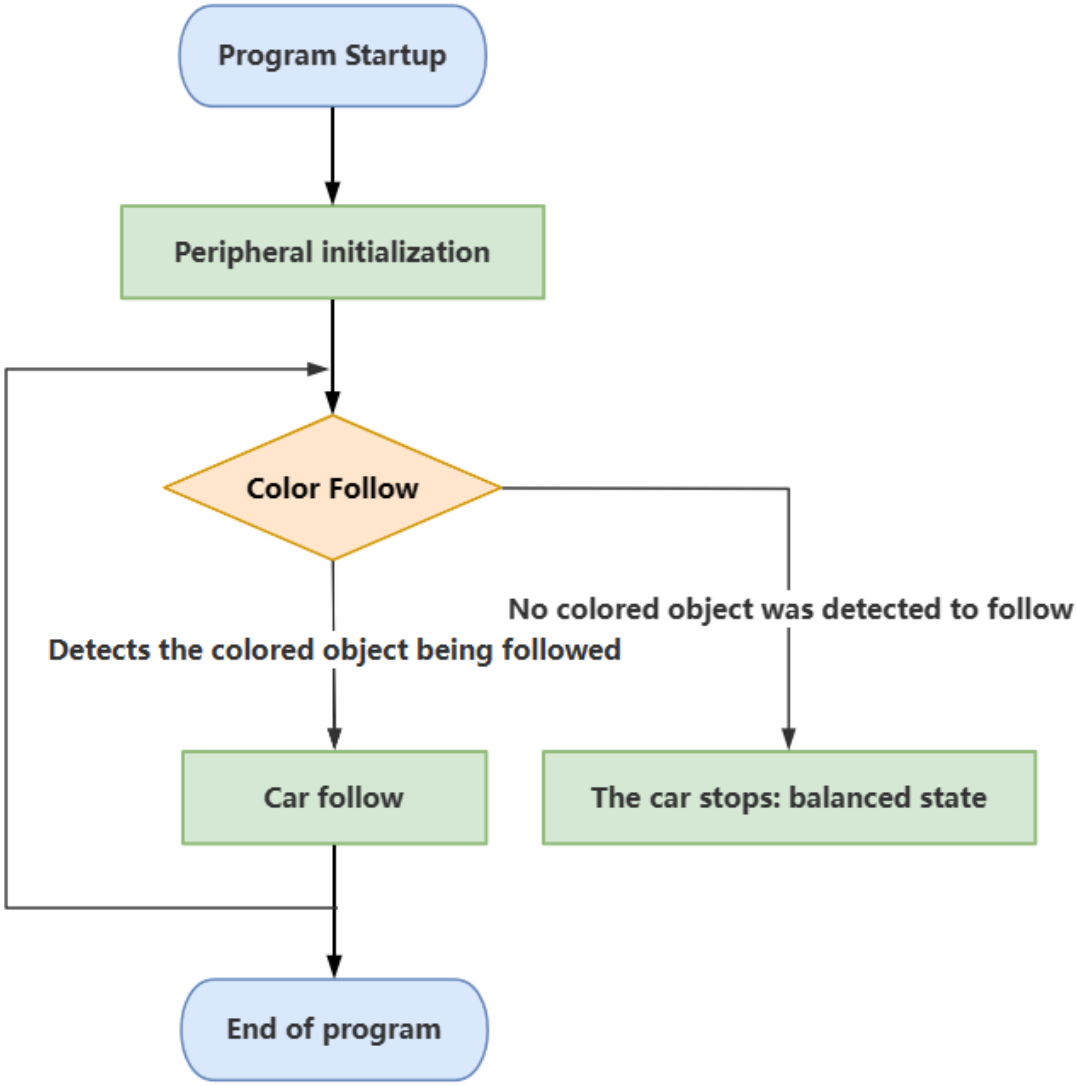
K210 color follows PID implementation function.

```
void APP_K210X_Y_Follow_PID(void)
{
    g_error_x = K210_Minddle_X-K210_data.k210_X;
//  g_error_y = K210_Minddle_Y-K210_data.k210_Y;

    g_error_area  =  K210_Minddle_area-K210_data.k210_area;

//  printf("area:%d\r\n",K210_data.k210_area);

    if(K210_data.k210_area == 0)// But when the area is 0, it means that the
object cannot be recognized.
    {
        Move_X = 0;
        Move_Z = 0;
        return;
```

```
    }

    if((my_abs(g_error_area )> 1500)) // Area dead zone
    {
        Move_X = GO_PID();
    }
    else
    {
        Move_X = 0;
    }

//  if((my_abs(g_error_x )> 40))// Not in dead zone
    {
        Move_Z = Turn_PID();
    }
//  else
//  {
//      Move_Z = 0;
//  }

    if(Move_X > 50)Move_X = 50;
    else if(Move_X < -50)Move_X = -50;
}
```

Program flow chart

Briefly introduce the function implementation process:



Experimental phenomenon

> **Software code**

The BalancedCar_color_follow.hex file generated by the project compilation is located in the OBJ folder of the BalancedCar_color_follow project. Find the BalancedCar_color_follow.hex file corresponding to the project and use the FlyMcu software to download the program to the development board.

```
The corresponding function can only be realized after both the K210 visual module
and the development board download the program
Product supporting information source code path: Attachment → Source code summary
→ 5.Balanced_Car_Extended → 09.BalancedCar_color_follow
```

> **Experimental phenomenon**

After the program is started, press the KEY1 button according to the OLED prompt to start the color line followfunction of the balance car: OLED displays the program function name in real time (Start Follow color!); K210 visual recognition module recognizes colored objects and follows; if no colored objects are recognized, the balance car remains balanced.

The program has voltage detection. If the voltage is less than 9.6V, a low voltage alarm is triggered and the buzzer will sound.
Common situations that trigger voltage alarms:
1. The power switch of the development board is not turned on, and only the Type-C data cable is connected for power supply
2. The battery pack voltage is lower than 9.6V and needs to be charged in time