

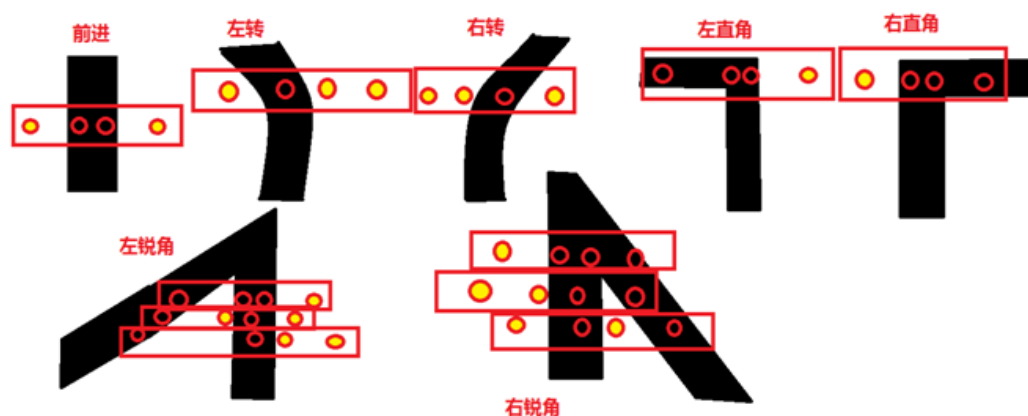
# 四路巡线实验

## 1、实验目的

烧录四路巡线的代码，将stm32f103c8t6主控板上电后，启动小车的红外巡线功能，小车会自动的巡黑线行走。这里需要调节红外探头的灵敏度，保证探头在黑线灯亮，在黑线外灯灭，震动小车调节探头，保证小车在运动过程中探头检测黑线的精确度。

## 2、实验原理

红外传感器巡线的基本原理是利用物体的反射性质，我们本次实验是巡黑线行驶，当红外线发射到黑线上时会被黑线吸收掉，发射到其他的颜色的材料上会有反射到红外的接受管上。我们根据这点的不同写相应的代码完成小车巡线功能。我们本次实验采用的是四路红外传感器分别连接在STM32主控板上的PC13, PC14, PC15, PB12口上。其中中间两路巡线是一直在黑线上，小车会直行，当任意一个出来，则小车会自动纠正，如果最外面的检测到黑线，则小车以更大速度纠正到正确黑线上面。以下为处理直线、小弯、直角、锐角的传感器状态分析。



## 3、实验步骤

**3.1** 先按照电机驱动例程完成主板与双路电机驱动板的接线，确保小车可以正常驱动。

**3.2** 根据stm32主控板的原理图，我们使用PC13, PC14, PC15, PB12引脚来对应连接四路巡线模块的X1, X2, X3, X4接口。

U2  
STM32F103C8T6

|        |    |                 |       |    |       |
|--------|----|-----------------|-------|----|-------|
| VBAT   | 1  | • VBAT          | VDD_3 | 48 | VCC   |
| PC13   | 2  | PC13-TAMPER-RTC | VSS_3 | 47 | GND   |
| PC14   | 3  | PC14-OSC32_IN   | PB9   | 46 | PB9   |
| PC15   | 4  | PC15-OSC32_OUT  | PB8   | 45 | PB8   |
| OSCIN  | 5  | PD0-OSC_IN      | BOOT0 | 44 | BOOT0 |
| OSCOUT | 6  | PD1-OSC_OUT     | PB7   | 43 | PB7   |
| NRST   | 7  | NRST            | PB6   | 42 | PB6   |
| GND    | 8  | VSSA            | PB5   | 41 | PB5   |
| VCC    | 9  | VDDA            | PB4   | 40 | PB4   |
| PA0    | 10 | PA0-WKUP        | PB3   | 39 | PB3   |
| PA1    | 11 | PA1             | PA15  | 38 | PA15  |
| PA2    | 12 | PA2             | PA14  | 37 | SWCLK |
| PA3    | 13 | PA3             | VDD_2 | 36 | VCC   |
| PA4    | 14 | PA4             | VSS_2 | 35 | GND   |
| PA5    | 15 | PA5             | PA13  | 34 | SWDIO |
| PA6    | 16 | PA6             | PA12  | 33 | PA12  |
| PA7    | 17 | PA7             | PA11  | 32 | PA11  |
| PB0    | 18 | PB0             | PA10  | 31 | PA10  |
| PB1    | 19 | PB1             | PA9   | 30 | PA9   |
| PB2    | 20 | PB2             | PA8   | 29 | PA8   |
| PB10   | 21 | PB10            | PB15  | 28 | PB15  |
| PB11   | 22 | PB11            | PB14  | 27 | PB14  |
| GND    | 23 | VSS_1           | PB13  | 26 | PB13  |
| VCC    | 24 | VDD_1           | PB12  | 25 | PB12  |



| 四路巡线模块 | stm32c8t6 |
|--------|-----------|
| X1     | PC13      |
| X2     | PC14      |
| X3     | PC15      |
| X4     | PB12      |
| VCC    | VCC       |
| GND    | GND       |

接线完成之后，将代码烧录进stm32芯片，小车则启动四路巡线模式。

**注意：**本次实验需要调节4路红外循迹模块的电位器使得巡线的灵敏度达到最佳。

## 4、主要代码展示

```
#include "stm32f10x.h"
#include "linewalking.h"
#include "sys.h"
#include "moto.h"
#include "delay.h"

/**
 * Function      Linewalking_GPIO_Init
 * @brief        巡线传感器GPIO初始化接口
 * @param[in]    void
 * @param[out]   void
 * @retval       void
 * @par History   无
 */
void Linewalking_GPIO_Init(void)
{
    /*定义一个GPIO_InitTypeDef类型的结构体*/
    GPIO_InitTypeDef GPIO_InitStructure;

#ifdef USE_LINE_L1
    /*开启外设时钟*/
    RCC_APB2PeriphClockCmd(Linewalk_L1_RCC, ENABLE);
    /*选择要控制的引脚*/

    GPIO_InitStructure.GPIO_Pin = Linewalk_L1_PIN;
    /*设置引脚模式为通用推挽输出*/
    GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
    /*设置引脚速率为50MHZ */
    GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
    /*调用库函数，初始化PORT*/
    GPIO_Init(Linewalk_L1_PORT, &GPIO_InitStructure);
#endif

#ifdef USE_LINE_L2
    /*开启外设时钟*/
```

```

RCC_APB2PeriphClockCmd(Linewalk_L2_RCC, ENABLE);
/*选择要控制的引脚*/

GPIO_InitStructure.GPIO_Pin = Linewalk_L2_PIN;
/*设置引脚模式为通用推挽输出*/
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
/*设置引脚速率为50MHZ */
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
/*调用库函数，初始化PORT*/
GPIO_Init(Linewalk_L2_PORT, &GPIO_InitStructure);
#endif

#ifdef USE_LINE_R1
/*开启外设时钟*/
RCC_APB2PeriphClockCmd(Linewalk_R1_RCC, ENABLE);
/*选择要控制的引脚*/

GPIO_InitStructure.GPIO_Pin = Linewalk_R1_PIN;
/*设置引脚模式为通用推挽输出*/
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
/*设置引脚速率为50MHZ */
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
/*调用库函数，初始化PORT*/
GPIO_Init(Linewalk_R1_PORT, &GPIO_InitStructure);
#endif

#ifdef USE_LINE_R2
/*开启外设时钟*/
RCC_APB2PeriphClockCmd(Linewalk_R2_RCC, ENABLE);
/*选择要控制的引脚*/

GPIO_InitStructure.GPIO_Pin = Linewalk_R2_PIN;
/*设置引脚模式为通用推挽输出*/
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
/*设置引脚速率为50MHZ */
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
/*调用库函数，初始化PORT*/
GPIO_Init(Linewalk_R2_PORT, &GPIO_InitStructure);
#endif

}

```

```

/**
 * Function    GetLineWalking
 * @brief      获取巡线状态
 * @param[in]  int *p_iL1, int *p_iL2, int *p_iR1, int *p_iR2  四路巡线位指针
 * @param[out] void
 * @retval     void
 * @par History 无
 */
void GetLineWalking(int *p_iL1, int *p_iL2, int *p_iR1, int *p_iR2)
{
    *p_iL1 = GPIO_ReadInputDataBit(LineWalk_L1_PORT, LineWalk_L1_PIN);

```

```

*p_iL2 = GPIO_ReadInputDataBit(LineWalk_L2_PORT, LineWalk_L2_PIN);
*p_iR1 = GPIO_ReadInputDataBit(LineWalk_R1_PORT, LineWalk_R1_PIN);
*p_iR2 = GPIO_ReadInputDataBit(LineWalk_R2_PORT, LineWalk_R2_PIN);
}

```

```

/**
 * Function      Linewalking
 * @brief        巡线模式运动
 * @param[in]    void
 * @param[out]   void
 * @retval       void
 * @par History   无
 */
void Linewalking(void)
{
    int LineL1 = 1, LineL2 = 1, LineR1 = 1, LineR2 = 1;

    GetLinewalking(&LineL1, &LineL2, &LineR1, &LineR2); //获取黑线检测状态

    if( (LineL2 == LOW || LineR1 == LOW) && LineR2 == LOW) //右锐角：右大弯,low表示
检测到黑线
    {
        SpinRight(7000);
        delay_ms(80);
    }
    else if ( LineL1 == LOW && (LineR1 == LOW || LineL2 == LOW)) //左锐角左大弯
    {
        SpinLeft(7000);
        delay_ms(80);
    }
    else if( LineL1 == LOW ) //左最外侧检测
    {
        SpinLeft(7000);
        delay_ms(10);
    }
    else if ( LineR2 == LOW) //右最外侧检测
    {
        SpinRight(7000);
        delay_ms(10);
    }
    else if (LineL2 == LOW && LineR1 == HIGH) //中间黑线上的传感器微调车左转
    {
        Turnleft(6500);
    }
    else if (LineL2 == HIGH && LineR1 == LOW) //中间黑线上的传感器微调车右转
    {
        Turnright(6500);
    }
    else if(LineL2 == LOW && LineR1 == LOW) // 都是黑色，加速前进
    {
        Forward(6000);
    }
}

```

