亮宁机器人模拟传感器的使用方法详解

新的标准版亮宁机器人套装中，增加了模拟灰度传感器。下面就来介绍一下模拟灰度传感器的使用方法。

都知道当使用模拟灰度传感器巡线之前，需要测值。所谓测值，就是记录下当传感器遇到黑色与白色时的不同返回值，然后计算出两数的平均值，用这个平均值做为黑白两色的分界线。接下来我们借助 LCD 液晶显示屏将 5 个模拟灰度的返回值显示出来。5 个模拟灰度的安装方法如图 1 所示，巡线板的连线方法如图 2 所示。

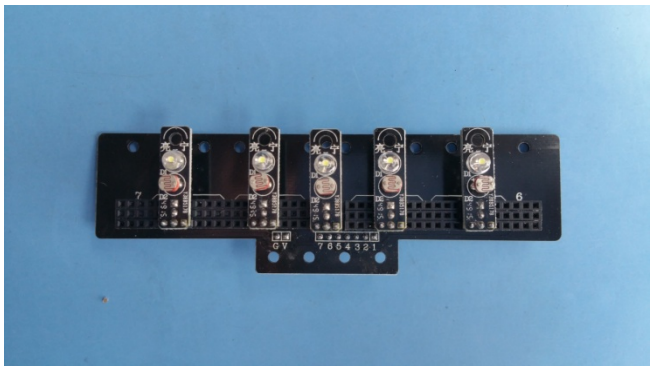

图 1 模拟灰度传感器的安装方法

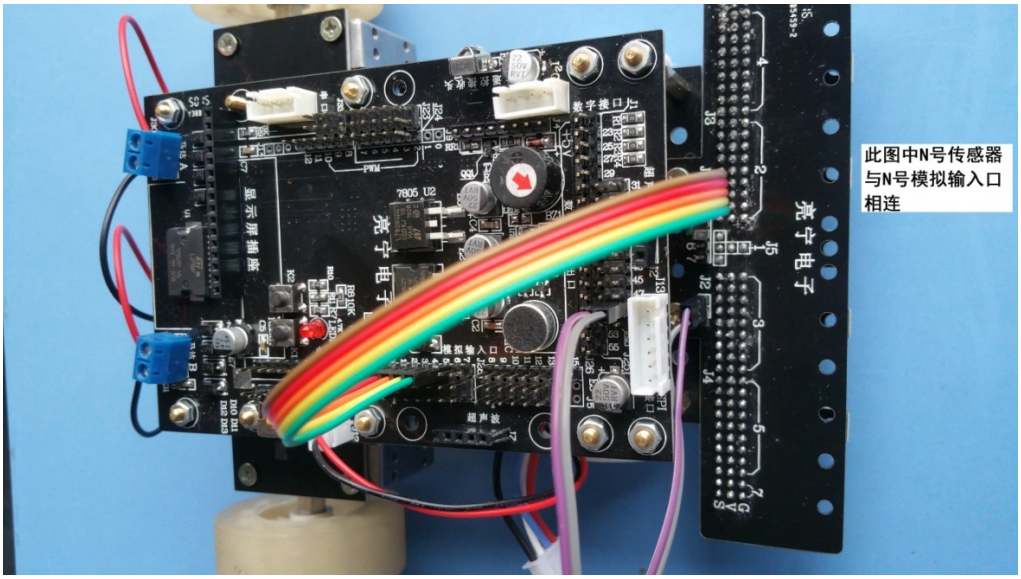

图 2 巡线板与主板的连线方法

在 LCD（1602）上显示 5 模拟灰度返回值的程序如下：

```
#include <LNDZ.h>
int ll,l,m,r,rr;
lc lcd;
void init()
{
    B_star();
    lcd.begin(16,2);
    lcd.bg(1);//开启液晶屏的背景光
}
void repeat()
{
    ll=AR(4);//请注意这里传感器与模拟输入口编号的对应
    l=AR(2);
    m=AR(1);
    r=AR(3);
    rr=AR(5);
    //为了方便观察传感器返回值，我们将 5 个返回值分两行显示
    lcd.setCursor(0,1);     lcd.print(ll);//第二行显示最左边灰度的返回值
    lcd.setCursor(3,0);     lcd.print(l);
    lcd.setCursor(7,0);     lcd.print(m);
    lcd.setCursor(11,0);    lcd.print(r);
    lcd.setCursor(13,1);    lcd.print(rr);//第二行显示最右边灰度的返回值
    delay(200);
```

lcd.clear();//200 毫秒之后清除上一次的显现（保持实时更新）

}

以上程序的实验效果如图 3、图 4 所示：

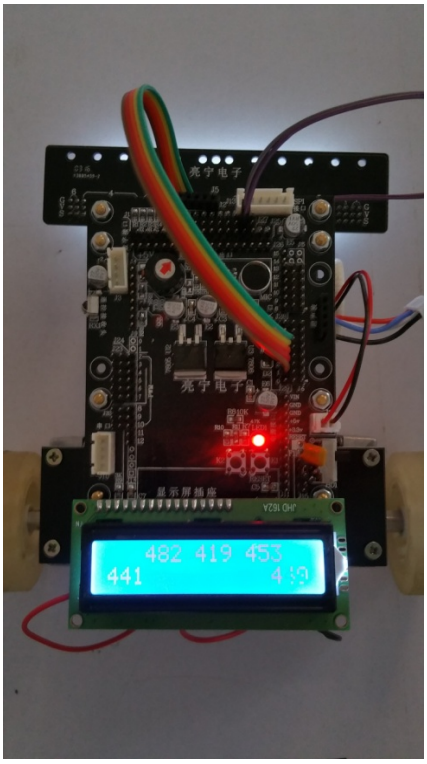

图 2 传感器在全白区域

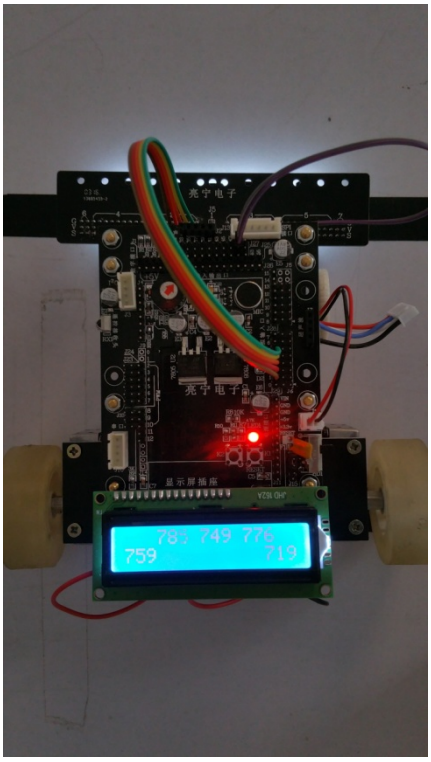

图 3 传感器在全黑区域

不难发现，传感器遇到黑色时的返回值大于白色的返回值。接下来我们计算出每一个模拟灰度的黑白分界线。例：最左边（4 号）位置传感器的黑白分界线计算如下：

（441+759）/2=600

接下来我们编写基本巡线程序：

```
#include <LNDZ.h>

int ll,l,m,r,rr;

void init()
```

```
{
    B_star();
}
void repeat()
{
    //以下是获取各模拟口的返回值
    ll=AR(4);//请注意这里传感器与模拟输入口编号的对应
    l=AR(2);
    m=AR(1);
    r=AR(3);
    rr=AR(5);
    //以下是判断那个传感器看到了黑线，就作出相应的反应
    if (ll>600) motor(-40,40);//600 是黑和白的分界线，注意，每个传感器的分界线都不一样，请认真测试，求出分界线（黑+白的和除以 2）
    else
    if (l>650) motor(0,40);
    else
    if (m>645) motor(40,40);
    else
    if (r>500) motor(40,0);
    else
    if (rr>620) motor(40,-40);
}
```

如果，你习惯于之前的巡线判断模式（黑 0 白 1），我们可以将程序进行如下修改，恢复到判断变量是否为 0 的模式，判断路口的方法也恢复到了 n<3 的模式。

```
#include <LNDZ.h>
int ll,l,m,r,rr,n,t=0;
void init()
{
    B_star();
}
void repeat()
{
    ll=AR(4)<600;//小于号为逻辑运算符。成立时返回 1，不成立时返回 0
    //遇黑色后的返回值，大于 600，所以<运算符会返回运算结果 0
    //遇白色后的返回值，小于 600，所以<运算符会返回运算结果 1
    l=AR(2)<650;
    m=AR(1)<640;
    r=AR(3)<500;
    rr=AR(5)<620;
    n=ll+l+m+r+rr;
    if (ll==0) motor(-40,40);
    else
    if (l==0) motor(0,40);
    else
```

```
        if (m==0) motor(40,40);
        else
        if (r==0) motor(40,0);
        else
        if (rr==0) motor(40,-40);
        if (n<3)
        {
            motor(0,0);
            while (1);
        }
    }
```

能不能让机器人自行计算黑白分界线，当然可以。具体功能描述如下：先将安装好的模拟灰度传感器均置于一种颜色（黑/白）正上方，操作者按下一个按钮，表示可以读取该颜色的返回值了，机器人接收到按钮信息，发出一声鸣叫，表示已成功读取该颜色的返回值；然后又将所有的模拟灰度放置在另外一种颜色的正上方，按下按钮，当再次听到机器人的蜂鸣声，表示两种颜色的分界线以及成功计算出来；最后，让机器人再一次进入等待按钮的状态，当按下按钮后开始利用计算出来的值，运行下面的主要程序。

新版亮宁机器人主板上 RESET 按钮旁增加了一个 K2 按钮，如图 4 所示。该按钮已在内部与 48 号数字口相连。我们借助此按钮来实现机器人自动计算模拟灰度传感器的黑白分界线。老版本机器人，可以借助模拟输入板或二进制开关板上的 ENT 按钮来代替 K2 按钮，只不过需要自己连线。

图 4

程序详解：

```
#include <LNDZ.h>
int llm,lm,mm,rm,rrm,ll,l,m,r,rr,n,t=0;
void init()
{
    pinMode(48,INPUT);//说明 48 号数字口，为输入口
    while (DR(48)==1)//按钮按下去后数字口返回 0
    {
        ll=AR(4);l=AR(2);m=AR(1);r=AR(3);rr=AR(5);
        //当按钮没被按下去的时候，反复地读取  4、2、1、3、5 号模拟口上的值
    }
  beep(600);
  //当按钮被按下，跳出以上 while 循环后，鸣叫一声，以示成功读取一种颜色的返回值
  //该 beep 语句还能消除按键抖动
    while (DR(48)==1)
    {//再次等待按钮被按下，等待过程中，不断地读取另外一种颜色的返回值，并且进行
    //平均值的计算
```

```
llm=(ll+AR(4))/2;lm=(l+AR(2))/2;mm=(m+AR(1))/2;rm=(r+AR(3))/2;rrm=(rr+AR(5))/2;
    }
  beep(600);
  //当按钮被按下，跳出以上 while 循环后，鸣叫一声，以示成功计算出黑白
  两色的分界//线，各传感器的分界线分别存于变量 llm、lm、mm、rm、rm
  中
  while (DR(48)==1);
  //测值成功后，待第三次按下按钮，开始执行下面的主要代码
    beep(600);
}
void repeat()
{
        ll=AR(4)<llm;//小于号为逻辑运算符。成立时返回 1，不成立时返回 0
        l=AR(2)<lm;
        m=AR(1)<mm;
        r=AR(3)<rm;
        rr=AR(5)<rrm;
        n=ll+l+m+r+rr;
        if (ll==0) motor(-40,40);
        else
        if (l==0) motor(0,40);
        else
```

```
        if (m==0) motor(40,40);
        else
        if (r==0) motor(40,0);
        else
        if (rr==0) motor(40,-40);
        if (n<3)
        {
            motor(0,0);
            while (1);
        }
}
```

以上方法，每次调试机器人都需要去完成一套测值的动作（三次摆放、三次按钮），略显麻烦，不过，当然存在一种既能自动测值，又不需要每次都测的方法。具体思路如下：当需要测值时，我们选择测值功能，将成功测出来的黑白分界线，存在缓存中，需要使用时去读取就行；当光照条件发生明显变化，我们再次选择测值功能，更新一下缓存中的原来存储的数据就行。

*程序详解：

注意，以下程序中，我们使用了三位模拟输入板。输入板上的 ENT 按钮与 23 号数字口相连，T1 旋钮与 10 号模拟输入口相连。

```
#include <LNDZ.h>
int llm,lm,mm,rm,rrm,ll,l,m,r,rr,n,t=0;
void test()//此函数注释参照上面的程序
```

```
{
    while (DR(23)==1)
    {
        ll=AR(4);   l=AR(2);   m=AR(1);   r=AR(3);   rr=AR(5);
    }
    beep(600);
    while (DR(23)==1)
     {
        llm=(ll+AR(4))/2;
        lm=(l+AR(2))/2;
        mm=(m+AR(1))/2;
        rm=(r+AR(3))/2;
        rrm=(rr+AR(5))/2;
    }
    beep(600);
    prm.write(1,llm/10);
    //缓存的使用方法：prm.write(地址,值);
   //亮宁机器人主板的缓存中最大能存储的数值为 255，我们将测量出来
   的分界线，//缩小到 10 分 1（分界线丢掉个位上的数，影响不大）
    prm.write(2,lm/10);
    prm.write(3,mm/10);
    prm.write(4,rm/10);
    prm.write(5,rrm/10);
```

```
        //往 5 个地址存储 5 个不同的数值
        while (DR(23)==1);
        beep(600);
}
void check()//获取实时路况
{
          ll=AR(4)<llm;
            l=AR(2)<lm;
            m=AR(1)<mm;
            r=AR(3)<rm;
            rr=AR(5)<rrm;
          n=ll+l+m+r+rr;
}
void find()//基本巡线代码
{
          if (ll==0) motor(-30,30);
          else
          if (l==0) motor(0,30);
          else
          if (m==0) motor(30,30);
          else
          if (r==0) motor(30,0);
          else
```

```
        if (rr==0) motor(30,-30);
    }
    void init()
    {
        pinMode(23,INPUT);
        //由于需要读取数字口的返回值，所以需要将数字口设置为输入口
        if (AR(10)>5) test();
        //开机后立马读取 T1 旋钮的值，如果大于 5，表示需要测值，则调用测值函数 test()。
        //测完值后，别忘了将 T1 旋钮拨回小于 5 的位置。
        llm=prm.read(1)*10;
         //如果不需要测值或成功测值后，读取存贮在各缓存中的数据，别忘了扩大 10 倍
          lm=prm.read(2)*10;
          mm=prm.read(3)*10;
          rm=prm.read(4)*10;
          rrm=prm.read(5)*10;
    }
    void repeat()
    {
       check();
       find();
    }
```