基于 Arduino 的超声波眼实验记录

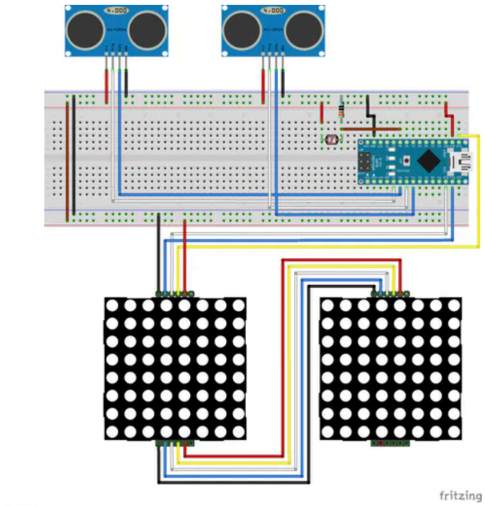
16307100068 豆孟恒

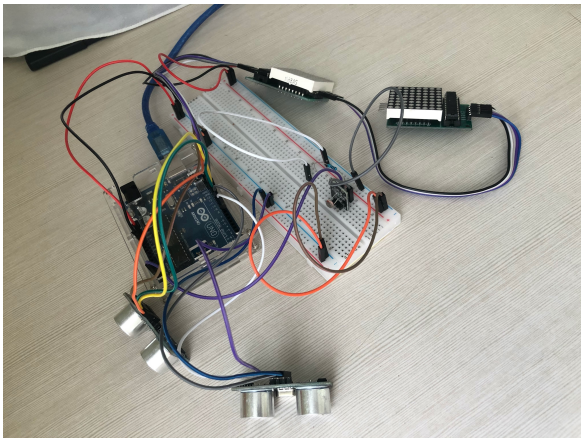
一、实验设计

在购入了一个超声波测距模块并用其进行了简单的测距实验后，开始在网站上寻找能够将手中的超声波测距模块和 8*8Led 点阵模块结合起来的项目，最终确定了使用两个 8*8Led 点阵模块以及两个超声波测距模块组合得到的超声波眼项目，使用两个 Led 点阵来表示眼睛的状态：向左看、向右看、向前看，通过比较两个超声波测距模块测得的距离大小来决定 Led“眼睛”应该看向哪一侧，由此可得到能够和使用者互动的超声波眼。

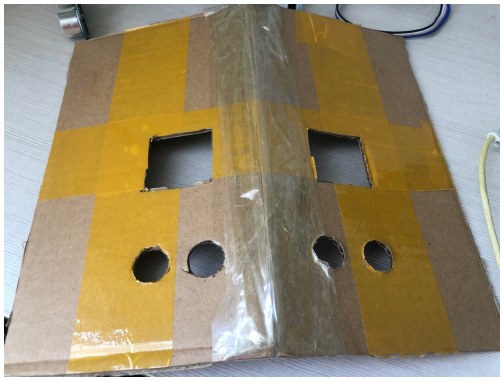
二、实验记录

整个装置共分为三个模块：超声波测距模块、Led 点阵模块以及光敏电阻模块（用于依据环境光调整 Led 点阵的亮度）。依次将各个装置通过面包板与 Arduino 连接，接线图和实物图如下：





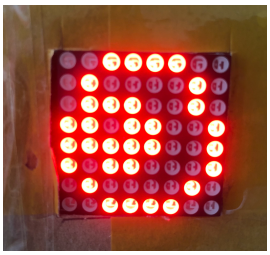
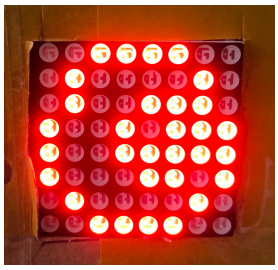
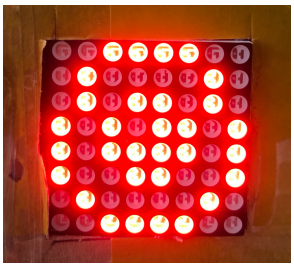
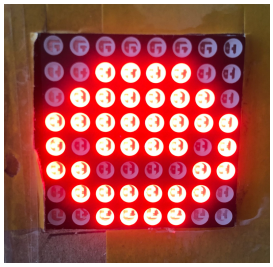
随后将超声波测距模块和 Led 点阵组装至固定装置上（对应各个装置大小的纸板），如下图：



随后开始向 Arduino 写入代码，根据装置，代码分为对应控制三个模块的部分以及各个模块之间交互的部分。

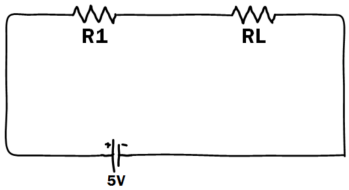
1、超声波测距模块部分调用 NewPing 库，设定最大响应距离为 400 厘米使用其中的 sonar.ping_median 语句可使该模块发送多个波进行测量，随后返回测量的时间中位值，由于传播距离和时间成正比，而该装置仅要求比较距离的大小，因此不再将时间转化为距离进行比较，直接使用时间进行比较。

2、Led 点阵模块结合了 MAX7219 模块，减少了对 I/O 口的占用（仅占用三个），同时允许多个模块串联。该部分调用 LedControl 库，其中的代码可对串联在同一串口下的多个模块分别控制，不过在该装置中使用的两个模块显示的图形是相同的。在分别录入了对应向左看、向右看、向前看以及眨眼对应的数组后（数组元素有 8 个对应控制每一行，而每一行由对应的 8 位二进制数控制），结合循环语句和 setRow 语句即可在 Led 上显示不同的眼睛图样。如下图

为了更好地模拟眼睛，加入了眨眼的图样，眨眼的时机通过 millis 语句和 random 语句进行控制，millis 语句可获得当前运行时长，当运行时长超过设定值后进行眨眼，同时通过 millis()+random()（随机值在 2s 至 10s 内）获得下一次的眨眼时间，如此往复。另外结合人眼的特性，眨眼分为单次和两次眨眼，将单次眨眼中间截断即可得到两次眨眼的图样。

3、光敏电阻模块主要用于根据环境的光强调整 Led 模块的亮度。由于 LedControl 库中的 setIntensity 语句将 Led 的亮度从 0~15 分为 16 级，需要将光敏电阻模块的读数转化为对应的 16 个亮度等级。测量电路如图所示：



使用模拟 I/O 口测量普通电阻后的电压值。考虑光敏电阻的阻值，为了尽可能地使用更多的亮度等级，即尽可能地让电压变化范围大，经过反复测试，普通电阻的阻值选为 5k 欧姆。根据电路图可知随光强的增加模拟口得到的电压值降低，用读数的最大值 1023 减去测量值，就可以得到随光强增加而增加的数值，

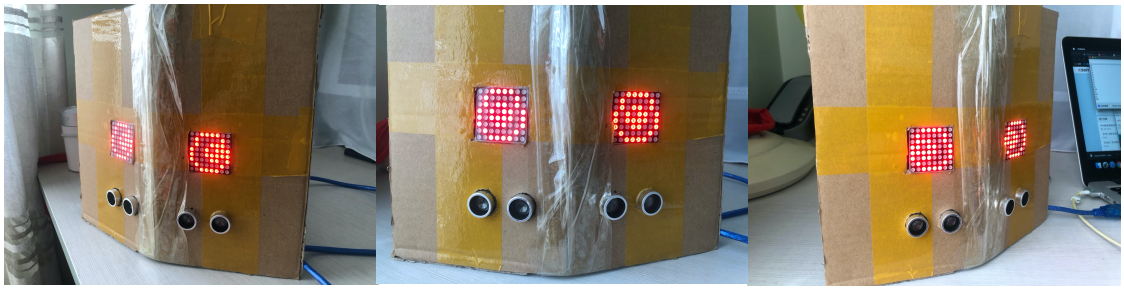
由于该数值在 0~1023 之间变化，将其分为 16 级需除以 64，由于涉及到的数值都为整形，最后得到的数值为 0~15 之间的整数，作为变量输入 setIntensity 即可，由此实现了随外界光强变化调整 Led 模块的亮度。

4、最后是结合两个超声波模块的读数判断“眼睛”该望向哪一侧，显然左侧距离小于右侧使应该使眼睛向左，反之则向右，当两者的读数小于小于某一给定值时（实验中设定为 250 微秒，对应的距离差为 4 厘米左右），认为物体位于正前方。

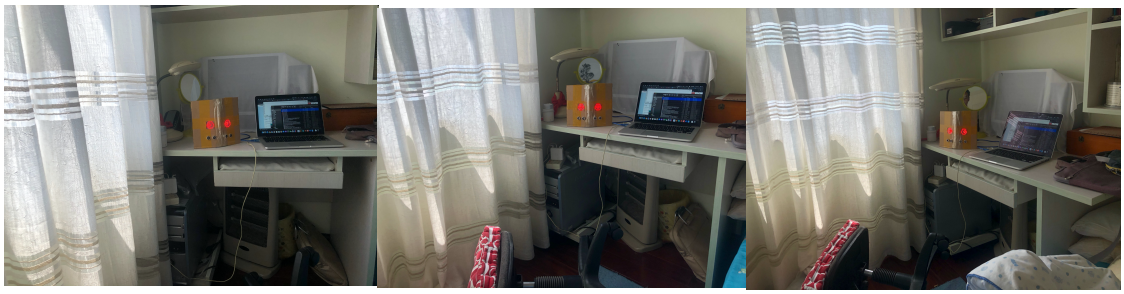
三、实验结果

在距离实验装置不同距离处进行测试，实验者手持手机在 60 厘米左右和 200 厘米左右处照相，结果如下：

60 厘米：



200 厘米：



可见装置运行情况良好，多次重复会发现在距离较远处比较难使“眼睛”处于向前看的状态。

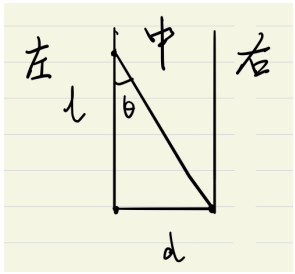
四、结果分析

1、在距离较远处比较难使“眼睛”处于向前看的状态。

这是能够预知的，由于实验者的身体宽度是有限的，距离过远很可能只能够让单个测距模块读数，使装置处于“非左即右”的状态。

2、随测量距离的远近对“眼睛”向前看标准的调整

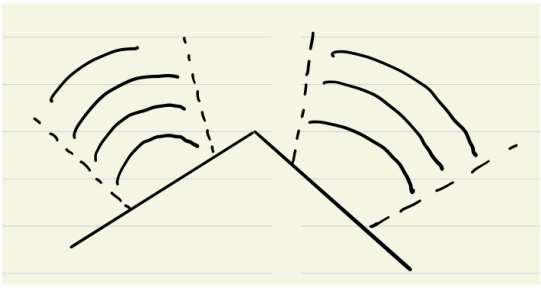
如图：



假设眼睛间距为d，物体距离两测量点平面的距离为l，将空间分为三部分，则在分界线上的距离差$\Delta=\sqrt{l^2+d^2}-l$，易知随l上升Δ下降，由此可知对于眼睛向前看的距离差判断需要随着待测物体的远近进行调整。在该模型中可以简单套用上述公式，然而在实际情况中，由于装置上两个“眼睛”之间有夹角、待测物体有宽度、对空间划分标准的不同（另一种方式可以用扇形划分），该调整方式需要进一步考量，但是粗略来看，随距离增加，“眼睛”向前看的标准应该放宽，这更能满足人们对一个“眼睛”装置的期待。

3、固定装置的形状

这部分主要影响的是两个测量装置之间的夹角，随角度不同可以分成三种即本实验中用到的钝角模型以及另外的平角模型和锐角模型。笔者认为，选择钝角模型的一个主要原因是这样可以扩大超声波的测量范围，如图：



虽然超声波具有良好的指向性，但笔者认为其传播路径仍然是扇形，钝角模型可以在一定程度上扩大该装置的探测范围。

五、参考资料

该项目主要参考了 Arduino 官网中 playground 里的项目，以及 LedControl 和 NewPing 两个库的使用手册，网址如下：

1.https://create.arduino.cc/projecthub/unexpectedmaker/ultrasoniceyes-b9fd38?ref=platform&ref_id=424_trending_part_beginner_&offset=72

2.http://wayoda.github.io/LedControl/pages/software#ShutdownMode

3.https://bitbucket.org/teckel12/arduino-new-ping/wiki/Home

六、代码

```
#include <NewPing.h>
#include <LedControl.h>

#define LIGHT A5

#define T1 10
#define E1 9
#define T2 8
#define E2 7
#define MAX_DISTANCE 400

NewPing eyeL(T1,E1,MAX_DISTANCE);
NewPing eyeR(T2,E2,MAX_DISTANCE);

#define DIN 13
#define CS 12
#define CLK 11
#define MAX_DEVICE 2

LedControl lc = LedControl(DIN,CLK,CS,MAX_DISTANCE);

//0,1,2 respectively correspond to forward, right, left state
int currentState = -1;

//The time duration of each sonar get the back signal
float duration1,duration2;

int lightAmount = 0;
int lightLevel = 0;

float nextBlink = millis() + 1000;

//Patterns of the status of the eye
byte eye_forward[8]=
{
  0b00111100,
  0b01000010,
  0b01011010,
  0b10101101,
  0b10111101,
  0b10011001,
  0b01000010,
  0b00111100
  };

byte eye_right[8]=
{
  0b00111100,
  0b01000010,
  0b01001110,
  0b10010111,
  0b10011111,
  0b10001101,
  0b01000010,
  0b00111100
  };
```

```
byte eye_left[8]=
{

  0b00111100,
  0b01000010,
  0b01110010,
  0b11011001,
  0b11111001,
  0b10110001,
  0b01000010,
  0b00111100
  };

byte eye_blink[8]=
{
  0b00000000,
  0b00111100,
  0b01111110,
  0b11111111,
  0b10111101,
  0b11000011,
  0b01111110,
  0b00111100
  };

void setup() {
  Serial.begin(9600);
  //Initialization of the LED
  for(int i=0;i<MAX_DEVICE;i++){
    lc.shutdown(i,false);
    lc.setIntensity(i,1);
    lc.clearDisplay(i);
    }

  //Set the pin modes for the Ultrasonic sensors
  pinMode(T1,OUTPUT);
  pinMode(E1,INPUT);
  pinMode(T2,OUTPUT);
  pinMode(E2,INPUT);
  digitalWrite(T1,LOW);
  digitalWrite(T2,LOW);

  pinMode( LIGHT, INPUT);

  //Start with the eyes looking forward
  ShowEye_Forward();
  currentState = 0;
}

void loop() {
  //Adjust the light level of the LEDs
  lightAmount = 1023-analogRead(LIGHT);
  lightLevel = lightAmount/64;
  //Serial.println(lightLevel);
  for(int i=0;i<MAX_DEVICE;i++){
    lc.setIntensity(i,lightLevel);
    }
```

```
  //Read out the distance(duration) of each eye
  duration1 = eyeL.ping_median(5);
  delay(500);
  duration2 = eyeR.ping_median(5);

  Serial.println(eyeL.ping_cm());
  Serial.println(eyeR.ping_cm());
  //Check if it is the time for a blink
  if( nextBlink <millis()){
    //Set the time for next blink
    nextBlink = millis() + random(2000,10000);
    //Decide if the blink is a double blink
    if( random(1,10)<6){
      ShowEye_DoubleBlink();
      }
     else{
      ShowEye_Blink();
      }
     delay(250);
     return;
    }

  //Now decide where the eye should look to
  float difference = (duration2 -duration1);
  if( abs(difference)<250 || (duration1==0 && duration2 ==0)){
    ShowEye_Forward();
    currentState = 0;
    }
   else if (duration2 < duration1 && duration1>0){
    ShowEye_Right();
    currentState = 1;
    }
    else if (duration1 < duration2 && duration2 >0){
    ShowEye_Left();
    currentState = 2;
     }
     delay(250);
}

void ShowEye_Right(){
  for(int i = 0;i<MAX_DEVICE;i++){
    //lc.clearDisplay(i);
    for(int j = 0;j<8;j++){
      lc.setRow(i,j,eye_right[j]);}
  }
}

void ShowEye_Left(){
  for(int i = 0;i<MAX_DEVICE;i++){
    //lc.clearDisplay(i);
    for(int j = 0;j<8;j++){
      lc.setRow(i,j,eye_left[j]);}
  }
}

void ShowEye_Forward(){
  for(int i = 0;i<MAX_DEVICE;i++){
```

```
    //lc.clearDisplay(i);
    for(int j = 0;j<8;j++){
      lc.setRow(i,j,eye_forward[j]);}
  }
}

void ShowEye_Blink(){
  for(int i = 0;i<MAX_DEVICE;i++){
    //lc.clearDisplay(i);
    for(int j = 0;j<8;j++){
      lc.setRow(i,j,eye_blink[j]);}
  }
  delay(150);
  //After the blink, go back the previous state
  if (currentState == 0){
    ShowEye_Forward();
    }
  else if (currentState == 1){
    ShowEye_Right();
    }
  else if (currentState == 2){
    ShowEye_Left();
    }
}

void ShowEye_DoubleBlink(){
  for(int i = 0;i<MAX_DEVICE;i++){
    //lc.clearDisplay(i);
    for(int j = 0;j<8;j++){
      lc.setRow(i,j,eye_blink[j]);}
  }
  delay(75);
  if (currentState == 0){
    ShowEye_Forward();
    }
  else if (currentState == 1){
    ShowEye_Right();
    }
  else if (currentState == 2){
    ShowEye_Left();
    }
   delay(75);
   ShowEye_Blink();
}
```