2024仰望盃全國科學 HomeRun 實作大賽

決賽成果報告書

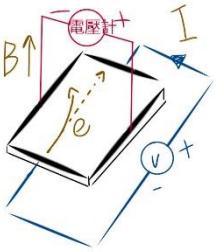
隊伍名稱:	我們這一家
作品名稱:	霍爾陣列雷達
科學概念1：	霍爾效應（Hall effect）是指當導體放置在一個磁場內，且有電流通過時，導體內的電荷載子受到勞侖茲力而偏向一邊，繼而產生電壓的現象。通過所產生電壓的極性，可證實導體內部的電流是由帶有負電荷的粒子（自由電子）之運動所造成。霍爾效應於1879年由埃德溫·赫伯特·霍爾（Edwin Herbert Hall）發現。
科學概念2：	電流磁效應在 1820 年，丹麥物理學家厄司特發現通有電流的導線附近有磁場，從此展開了電磁現象的研究，經由法拉第等人的研究，最後由馬克士威在 1864 年完成完整的電磁場理論。而長直導線通過電流所產生磁感應方向可用安培右 手定則來定訂。

註：決賽作品說明書內文總頁數最多10頁(不含本封面及授權同意書)，請勿寫上可辨識學校名稱之資訊。

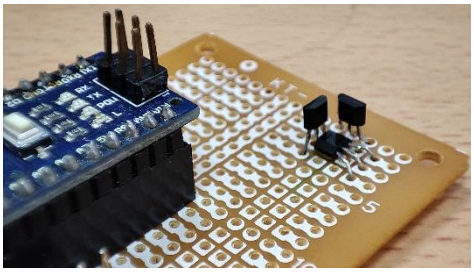
決賽成果報告書內文

（最多10頁）

1. 發想動機：我們在機器人社團中學習如何設計自走車的時候，認識到了帶有編碼器的馬達，這其中又分為光學編碼器與霍爾編碼器兩種。進一步發現霍爾效應是一個簡單、有趣又可量化的物理現象，除了作為開關感應還可以作為非接觸性電流測量。透過以上的理解，隊員發想出是否可以藉由霍爾感應器設計出讓原本看不見的磁場，可以藉由平面多點(8X8陣列)的測量結果，模擬出磁場分布的圖形。



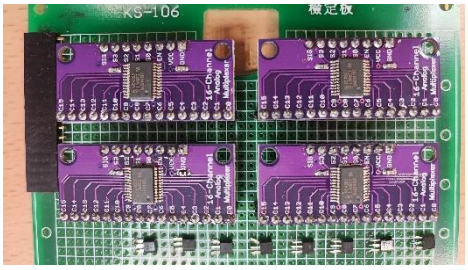
2. 作品創意性：以往在電磁相關的課程中，若要觀察磁場方向或是磁力線分布，一般都是藉由移動指南針或是觀察鐵粉的分布來達到目的。若是可以透過霍爾感應器紀錄數據，便能夠將觀察結果量化，並進一步分析。但是霍爾感應器是點的測量，若要進行二維向度的觀測，可將一定數量的感測器以經過設計的排列方式分布，透過硬體與軟體的設計，將收集到的數據轉換為對應參考的顏色，並在輸出設備中展現(LCD 螢幕、RGB LED Matrix 等)。此外，我們設計了一個三維磁力線測量的工具，就是將三個霍爾感應器分別以 X、Y、Z 軸的方向排列，將讀取的數值傳到電腦，透過 Processing 工具軟體模擬3D 空間中磁力線的方向。



3. 硬體及電路架構圖：所需材料如下，

(a) Arduino Nano：選擇 Arduino 家族系列中的 Nano，乃著眼於其原生的8個類比輸入端子(10 bit ADC)符合規劃的8X8霍爾感應器陣列行、列的數量。

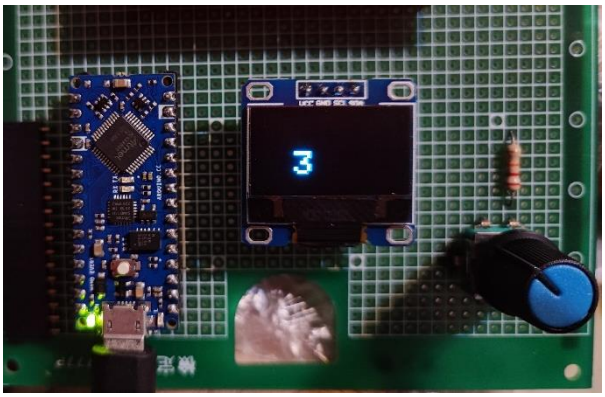
(b) CD74HC4067模組：此為高速 CMOS 16通道 Analog/Digital 多工器，透過此模組只要4個 CD74HC4067模組，便足以循序讀取64顆霍爾感應器的數值。



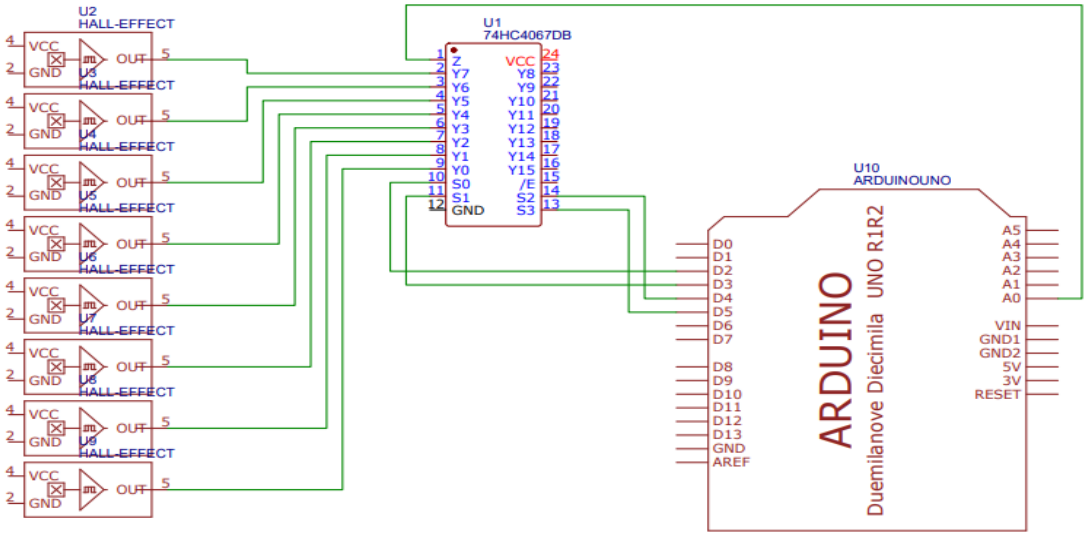
(c) SS49E Linear Hall-Effect Sensor：線性霍爾感應器由電壓調整器，霍爾電壓發生器，線性放大器和射極跟隨器組成，輸入是磁感應強度，輸出是和輸入量成正比的電壓。靜態輸出電壓（B=0GS）是電源電壓的一半。S 磁極出現在霍爾傳感器標記面時，將驅動輸出高於零電平；N 磁極將驅動輸出低於零電平，輸出電壓決定器件最敏感面的磁通密度。

(d) RGB LED Matrix：通常由 WS2812或是 WS5050顆粒之 RGB LED 組成，因其控制訊號可以串聯序列方式傳遞，在應用時可以大量簡化接線的複雜度。

(e)旋轉編碼器與 OLED 顯示器：在實作測試的過程中發現不同的磁鐵因為磁力強度不同，測量讀數也存在不同的範圍差距，為提高可讀性，增設一個旋轉編碼器用以調節讀數不同的放大倍數，並且將放大倍數顯示在 OLED 顯示器上。此外我們亦發現，電路通電一段時間之後會有信號漂移的現象，正好利用旋轉編碼器的按鈕功能作為歸零的動作。



電路架構：由於市面上最易取得的多工器為16通道多工器，8X8陣列總共64顆霍爾感應器需要搭配4顆16通道多工器，藉由 Arduino 之 A0~A3讀取電壓值，轉而控制 RGB LED Matrix，右圖為霍爾感應器陣列其中一列之線路示意圖：



程式碼：

(a) 主結構：

```
#include <Wire.h>
#include <Adafruit_GFX.h>
#include <Adafruit_SSD1306.h>
#define SCREEN_WIDTH 128
#define SCREEN_HEIGHT 64

#include <Adafruit_NeoPixel.h>
#define LED_PIN 6
#define LED_COUNT 64

#include <EC11.hpp>
EC11 encoder;
#define DEMO_INTERRUPTS 0
#if DEMO_INTERRUPTS
const int encoderPinA = 2;
const int encoderPinB = 3;
void pinDidChange() {

encoder.checkPins(digitalRead(enc
oderPinA),
digitalRead(encoderPinB));
}
void prepare() {
  attachInterrupt(0,
pinDidChange, CHANGE);
  attachInterrupt(1,
pinDidChange, CHANGE);
}
#else
const int encoderPinA = 2;
const int encoderPinB = 3;
void prepare() {
}
#endif

Adafruit_NeoPixel
matrix(LED_COUNT, LED_PIN,
NEO_GRB + NEO_KHZ800);
Adafruit_SSD1306
display(SCREEN_WIDTH,
SCREEN_HEIGHT, &Wire, -1);

int en = 7;
int s0 = 8;
int s1 = 9;
int s2 = 10;
int s3 = 11;

int SIG0_pin = 0;
int SIG1_pin = 1;
int SIG2_pin = 2;
int SIG3_pin = 3;

int val0, val1, val2, val3;
int sensor[4][16];
void setup() {
```

```
  pinMode(encoderPinA,
INPUT_PULLUP);
  pinMode(encoderPinB,
INPUT_PULLUP);
  pinMode(4, INPUT);
  prepare();
  pinMode(s0, OUTPUT);
  pinMode(s1, OUTPUT);
  pinMode(s2, OUTPUT);
  pinMode(s3, OUTPUT);
  pinMode(en, OUTPUT);
  digitalWrite(s0, LOW);
  digitalWrite(s1, LOW);
  digitalWrite(s2, LOW);
  digitalWrite(s3, LOW);
  digitalWrite(en, LOW);
  Serial.begin(9600);
  matrix.begin();  // INITIALIZE
NeoPixel strip object (REQUIRED)
  matrix.show();   // Turn OFF
all pixels ASAP
  matrix.setBrightness(50);

display.begin(SSD1306_SWITCHCAPVC
C, 0x3C);
  delay(1000);
  display.clearDisplay();
  calibration();
}
void calibration() {
  for (int i = 0; i < 16; i++) {
    readMux(i);
    delay(10);
    sensor[0][i] = val0;
    sensor[1][i] = val1;
    sensor[2][i] = val2;
    sensor[3][i] = val3;
  }
  /*for (int j = 0; j < 4; j++) {
    for (int k = 0; k < 16; k++)
{
      Serial.print(sensor[j][k]);
      Serial.print(",");
    }
    Serial.println();
  }*/
}
static int value = 1;
void loop() {
  EC11Event e;
  if (encoder.read(&e)) {
    if (e.type ==
EC11Event::StepCW) {
      value += e.count;
    } else {
      value -= e.count;
    }
    //Serial.println(value);
  }
#if DEMO_INTERRUPTS
  delay(200);
#else
  for (int i = 0; i < 200; i++) {

encoder.checkPins(digitalRead(enc
oderPinA),
digitalRead(encoderPinB));
    delay(1);
  }
#endif
  if (digitalRead(4))
    calibration();
  if (value < 1)
    value = 1;
  else if (value > 5)
    value = 5;
  display.clearDisplay();
  display.setTextColor(WHITE);
  display.setTextSize(3);
  display.setCursor(30, 30);
  display.println(value);
  display.display();
  for (int i = 0; i < 16; i++) {
    readMux(i);
    val0 = (val0 - sensor[0][i])
* value;
    if (val0 > 0) {
      matrix.setPixelColor(i,
matrix.Color(val0, 0, 0));
    } else {
      val0 *= -1;
      matrix.setPixelColor(i,
matrix.Color(0, 0, val0));
    }
```

```
    val1 = (val1 - sensor[1][i])
* value;
    if (val1 > 0) {
      matrix.setPixelColor(i +
16, matrix.Color(val1, 0, 0));
    } else {
      val1 *= -1;
      matrix.setPixelColor(i +
16, matrix.Color(0, 0, val1));
    }
    val2 = (val2 - sensor[2][i])
* value;
    if (val2 > 0) {
      matrix.setPixelColor(i +
32, matrix.Color(val2, 0, 0));
    } else {
      val2 *= -1;
      matrix.setPixelColor(i +
32, matrix.Color(0, 0, val2));
    }
    val3 = (val3 - sensor[3][i])
* value;
    if (val3 > 0) {
      matrix.setPixelColor(i +
48, matrix.Color(val3, 0, 0));
    } else {
      val3 *= -1;
      matrix.setPixelColor(i +
48, matrix.Color(0, 0, val3));
    }
    matrix.show();
    delay(10);
  }
}

void readMux(int channel) {
  int controlPin[] = { s0, s1,
s2, s3 };
  int muxChannel[16][4] = {
    { 0, 0, 0, 0 },  //channel 0
    { 1, 0, 0, 0 },  //channel 1
    { 0, 1, 0, 0 },  //channel 2
    { 1, 1, 0, 0 },  //channel 3
    { 0, 0, 1, 0 },  //channel 4
    { 1, 0, 1, 0 },  //channel 5
    { 0, 1, 1, 0 },  //channel 6
    { 1, 1, 1, 0 },  //channel 7
    { 0, 0, 0, 1 },  //channel 8
    { 1, 0, 0, 1 },  //channel 9
    { 0, 1, 0, 1 },  //channel 10
    { 1, 1, 0, 1 },  //channel 11
    { 0, 0, 1, 1 },  //channel 12
    { 1, 0, 1, 1 },  //channel 13
    { 0, 1, 1, 1 },  //channel 14
    { 1, 1, 1, 1 }   //channel 15
  };

  for (int i = 0; i < 4; i++) {
    digitalWrite(controlPin[i],
muxChannel[channel][i]);
  }
  val0 = analogRead(SIG0_pin);
  val1 = analogRead(SIG1_pin);
  val2 = analogRead(SIG2_pin);
  val3 = analogRead(SIG3_pin);
}
```

(b) Processing：

```
import processing.serial.*;
Serial myPort;
int sensorValue;
int x, y, z;

void setup() {
  size(1200, 900, P3D);
  myPort = new Serial(this,
"COM5", 9600);
  myPort.bufferUntil('\n');
}
void draw() {
  background(43, 74, 120);
  stroke(255);
  noFill();
  translate(530, 520, 300);

  beginShape();
  vertex( x, y, z);
  vertex(x, y, 0);
  vertex(x, 0, 0);
  vertex(x, 0, z);
  endShape(CLOSE);
  beginShape();
```

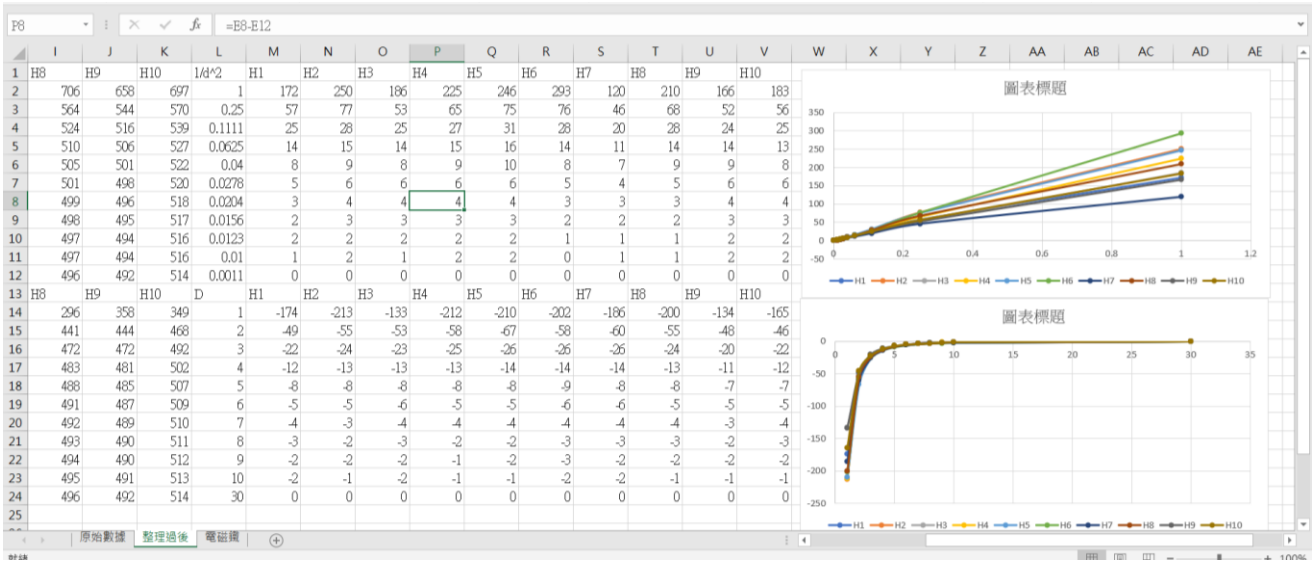
```
  vertex( x, y, z);
  vertex(0, y, z);
  vertex(0, y, 0);
  vertex(x, y, 0);
  endShape(CLOSE);
  beginShape();
  vertex(0, 0, z);
  vertex(x, 0, z);
  vertex(x, y, z);
  vertex(0, y, z);
  endShape(CLOSE);
  beginShape();
  vertex(0, 0, 0);
  vertex(x, 0, 0);
  vertex(x, y, 0);
  vertex(0, y, 0);
  endShape(CLOSE);
  line(0, 0, 0, 0, -300, 0);
  line(0, 0, 0, 0, 0, 300);
  line(0, 0, 0, 300, 0, 0);
  stroke(255, 0, 0);
  line(0, 0, 0, x, y, z);
}
void serialEvent(Serial port) {
  String myString =
myPort.readStringUntil('\n');
  if (myString != null) {
    myString = trim(myString);
    int sensorData[] =
int(split(myString, ','));
    x = sensorData[0];
    y = sensorData[1];
    z = sensorData[2];

    print(x);
    print(",");
    print(y);
    print(",");
    println(z);
  }
}
```

(c) 3D 讀入

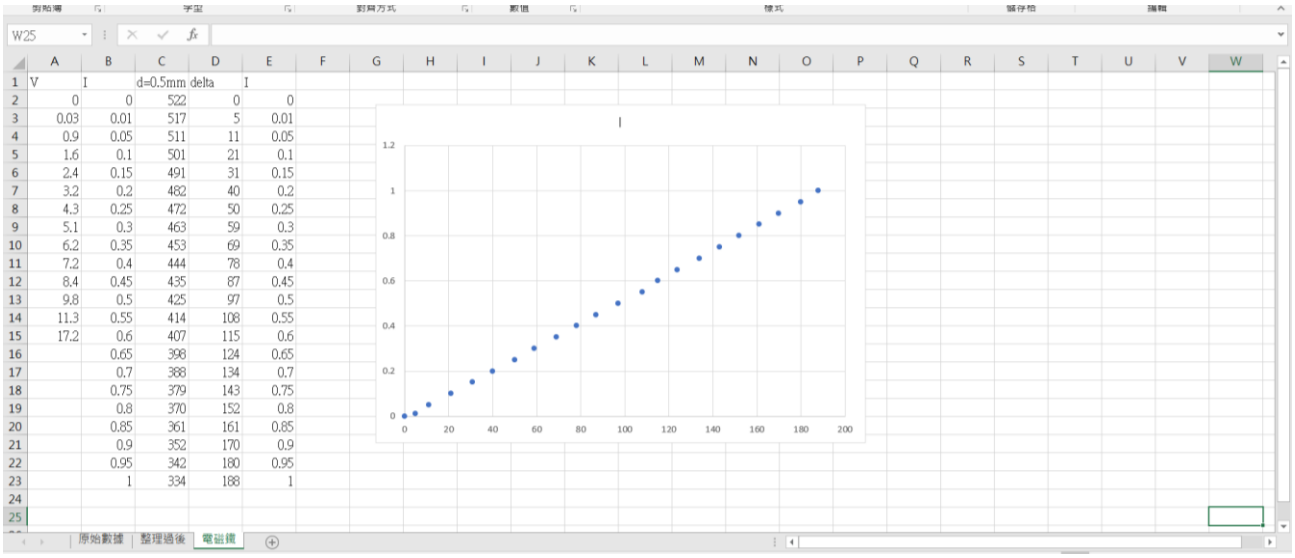
```
int initial_value[3];

void setup() {
  Serial.begin(9600);
initial_value[0]=analogRead(A0);
initial_value[1]=analogRead(A1);
initial_value[2]=analogRead(A2);
}

void loop() {
  Serial.print(analogRead(A0)-initial_value[0]);
  Serial.print(",");
  Serial.print(analogRead(A1)-initial_value[1]);
  Serial.print(",");
  Serial.println(analogRead(A2)-initial_value[2]);
  delay(100);

}
```

4. 作品成果報告：

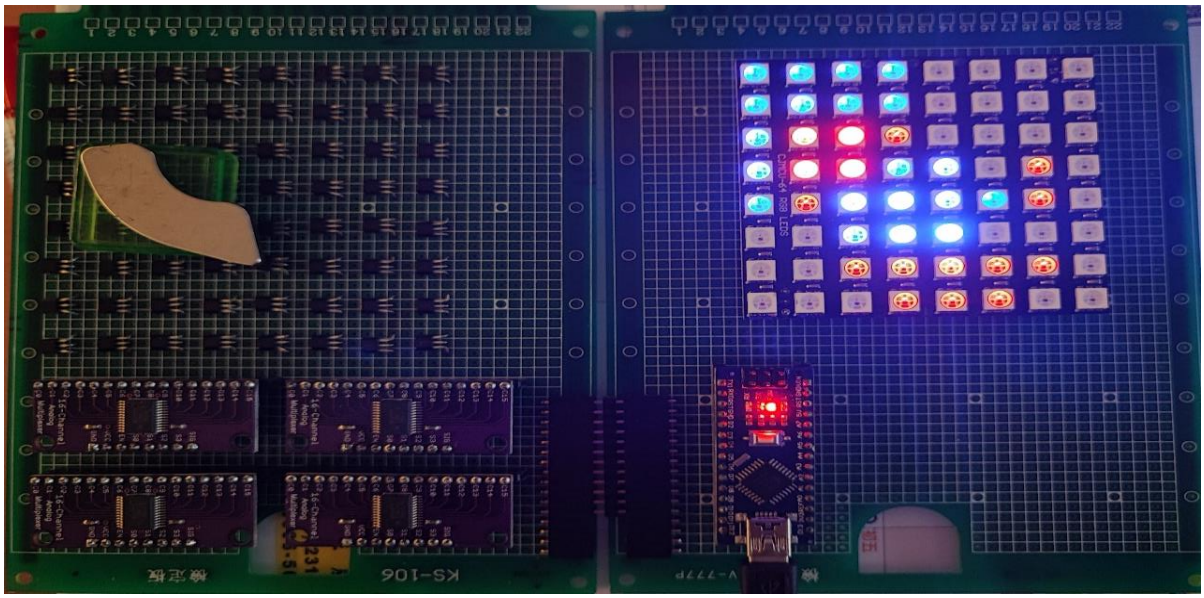
首先，我們拿永久磁鐵依照不同的距離紀錄霍爾感測器的讀值，經過整理並繪製圖表，符合磁場強度與距離平方成反比，但目的是證明感測器的線性與否。



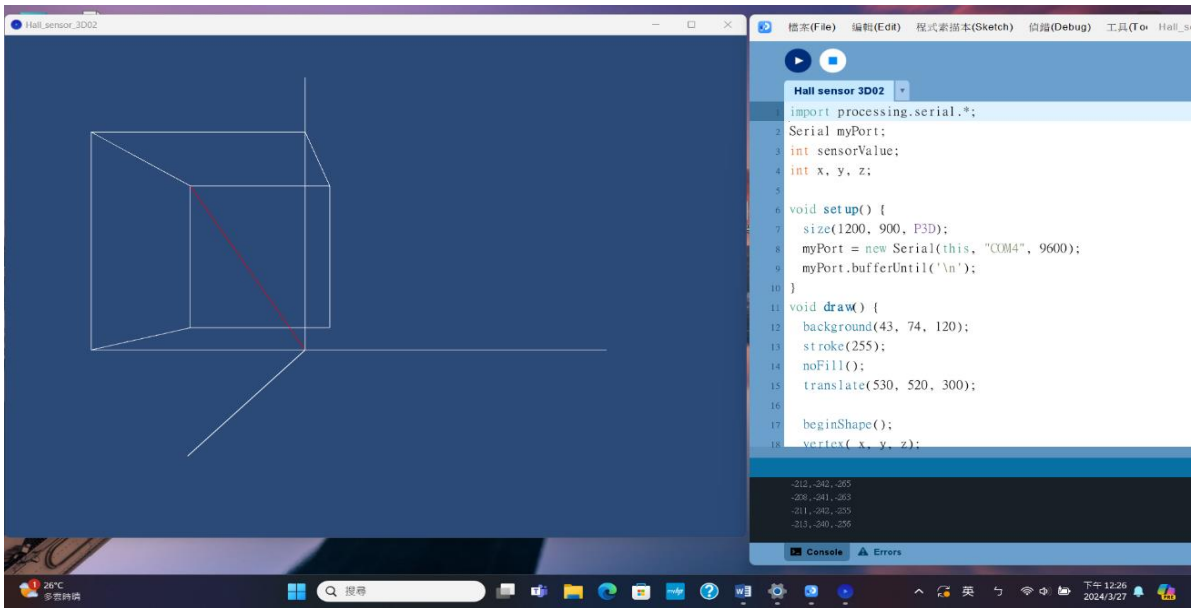
我們也用電磁鐵以不同電流輸入並且記錄，符合磁場強度與電流成正比。



當有磁鐵靠近感應器陣列時，依據每一顆霍爾感應器所得數值相對應到 RGB LED 陣列，以紅、藍兩色區分磁力線射入或射出方向，亮度則用來顯示磁場強度，為求實驗觀察多樣性，我們試著拿各種不同形狀的磁鐵來觀察其磁力線分布。



三維磁力線測量的工具，透過 Processing 工具軟體模擬3D 空間中磁力線的方向。



5. 參考文獻：

一、 Hall effect-Wikipedia
https://en.wikipedia.org/wiki/Hall_effect

二、 霍爾效應-科學 Online
https://highscope.ch.ntu.edu.tw/wordpress/?p=29489

三、 Arduino Reference
https://www.arduino.cc/reference/en/

四、 CD74HC4067產品規格表(Texas Instruments)
https://www.ti.com/product/zh-tw/CD74HC4067

五、 EasyEDA-Online PCB Design https://easyeda.com/