Applying Drone Technology In Environmental Study by Collecting CO2 Data

Table of Contents

ACKNOWLEDGEMENTS..... 3

ABSTRACT..... 4

INTRODUCTION..... 5

HYPOTHESIS..... 6

MATERIALS..... 7

PROCEDURE..... 8

RESULTS..... 19

CONCLUSION..... 21

REFERENCES..... 22

ACKNOWLEDGEMENTS

I would like to thank and give my appreciation to the Superintendent of the Science Department, Ms. Makar for making this opportunity possible. I am grateful to be helped by Dr. Liu, Professor and Chair of the Computer Science Department. I would like to offer special thanks to my Father for always listening to my ideas and encouraging me throughout my research. I would like to thank Ms. Donnelly for the support of my other projects. I give a special thank you to Mr. Orbe, my chemistry teacher from 10th grade, whose humor and words of advice meant alot to me. Thank you to all who made this experience possible.

Abstract: This environmental study is modeled to collect Carbon Dioxide (CO2) and pollution data to make predictive models. CO2 data is compared around a local reservoir in clear sunny weather and at the local park with various sensors to study the correlation of pollution and Carbon Dioxide. With the Arduino Kit, a CO2 sensor, and a Humidity and Temperature Sensor, data is easily interpreted to understand the effects of the climate on CO2 dispersion in a location. Using a drone helps make the collection of CO2 pollution data easier, data could be collected at a local factory quickly and efficiently if reports of unsafe air concentrations are in the perimeter. The choice of a predictive model allowed for a better understanding and efficient use of the data in various areas of scientific studies. The results from the data showed a direct correlation between CO2 levels and temperature readings but not a strong correlation with humidity levels. The results gathered help to understand the effectiveness of using technology such as drones for collection of CO2 data.

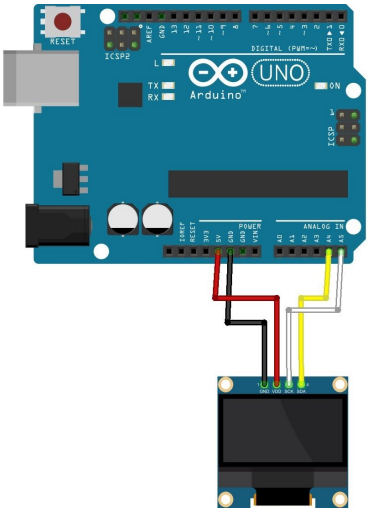
Introduction: Arduinos are small and mobile prototyping systems that can be programmed to take information from the world around us and output it into a readable form. Attachments for Arduinos exist for various purposes and different experiments. The beginning of Arduinos started by Colombian student, Hernando Barragan in Italy, from the Interactive Design Institute in Ivrea, in 2005. Hernando Barragan and a team of five developers did a thesis and worked to make the technology more accessible and less expensive over the years, and it became open source. The Arduino Software is now available by anyone with compatible systems to program their Arduino. The Arduino computer software uses C++, a computer programming language born from Danish computer scientist Bjarne Stroustrup that is widely used today in the technology industry. Over the years, several Arduino kits have expanded from the roots of the classroom of the team initially behind the Arduino project to educational institutions and into the hands of student scientific researchers. The Arduino, in conjunction with drone technology, is an expandable system used in the collection of CO2 data in our environment. The problem with collecting CO2 data are the other variables that affect the CO2 readings in the sensor, such as humidity and temperature. Lab reports that use drones to collect CO2 data are already published, but those lack a predictive model. My solution aims to answer if and how CO2 affects the variables of humidity and temperature.

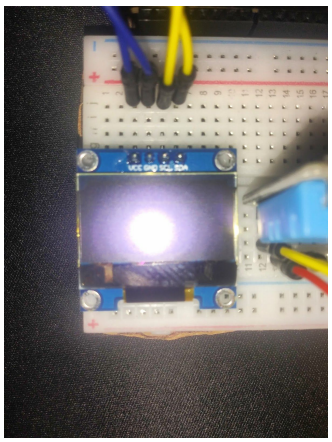
Hypothesis: What are the effects of humidity and temperature on CO2 as shown in a predictive model?

Materials:

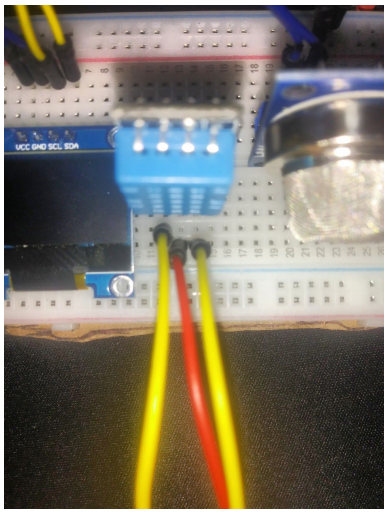
UNO R3, Arduino Software, DHT11 Temperature and Humidity Sensor, Blue LED, Green LED, Yellow LED, Red LED, MQ-135 CO2 Sensor, MK-003 Mini Breadboard, Cardboard, Electrical Tape, .960" OLED Screen, USB Cable, Portable Charger, 220R ohm Resistors, Jumper Wires

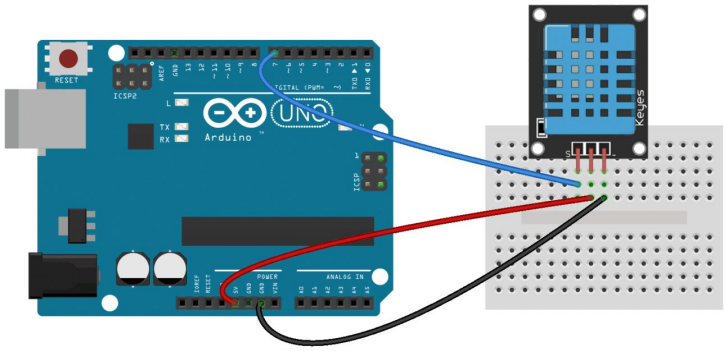
Procedure: I started the experiment by attaching the UNO R3 development board with the .960” OLED Screen without soldering and used the Jumper Wires to connect the respective pins of the .960” OLED Screen to the UNO R3.



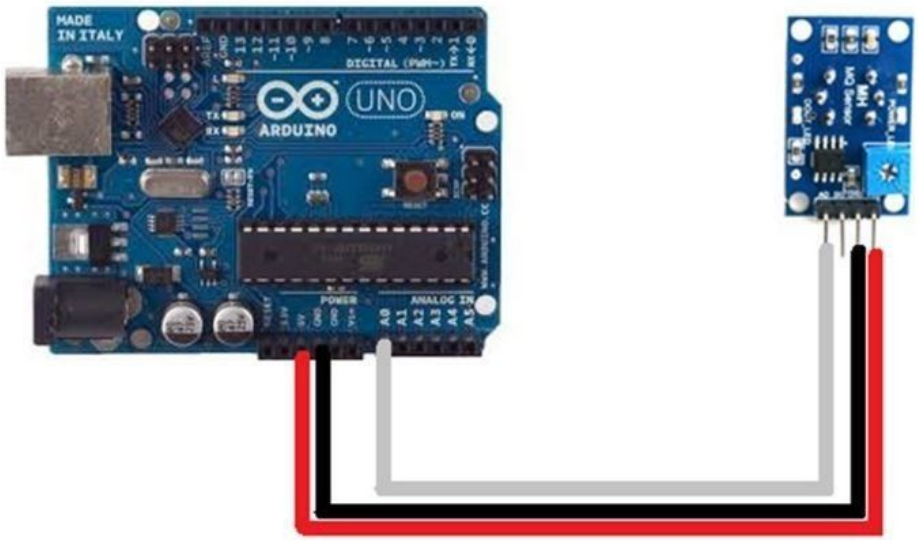


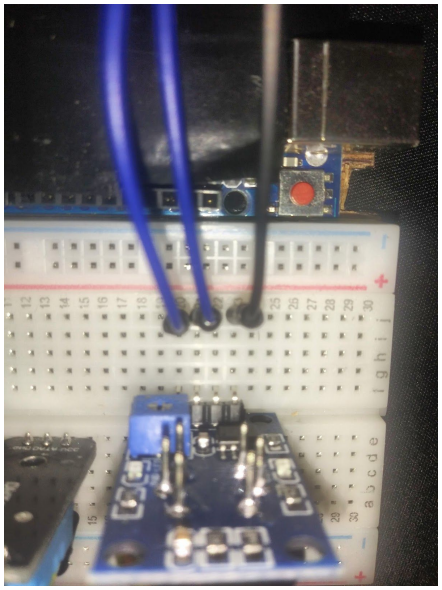
Then the UNO R3 was attached with the DHT11 Temperature and Humidity Sensor next to the .960” OLED Screen on the MK-003 Mini Breadboard and, the Jumper Wires connected the DHT11 Temperature and Humidity Sensor to their designated pins on the UNO R3.



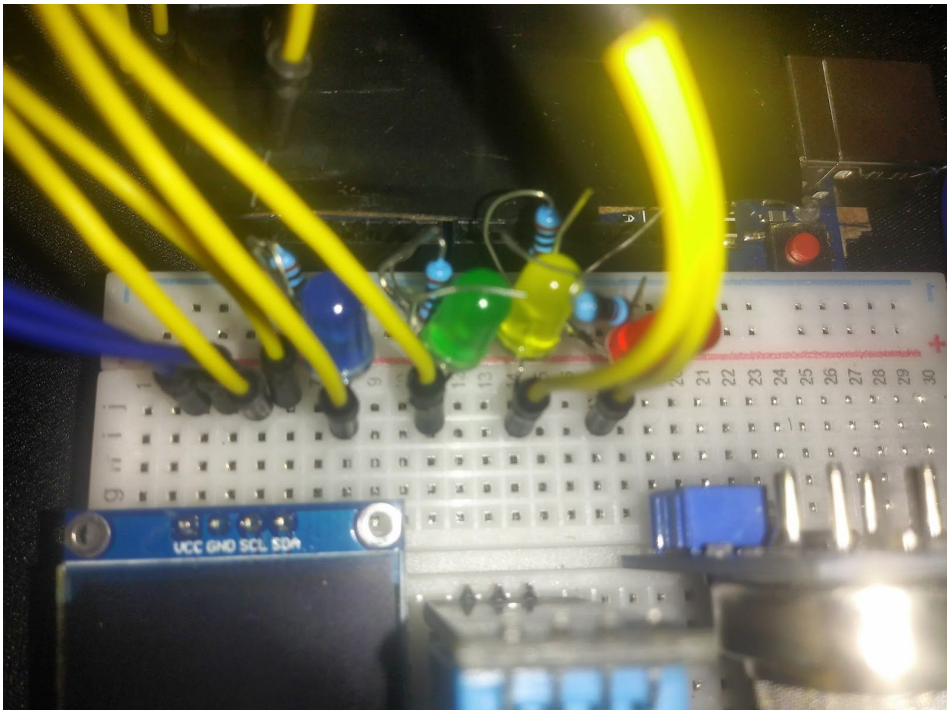


The MQ-135 CO2 Sensor was placed on the MK-003 Mini Breadboard next to the previous sensor with the pins connected to their operational pins on the UNO R3 with Jumper Wires.





Next, the Blue LED, Green LED, Yellow LED, and Red LED was placed on the MK-003 Mini Breadboard, and the Jumper Wires were connected behind LED's for power and connected to the pins of the Arduino defined in the Arduino Software. Each color represents the state of the level of CO2; blue represents safe level, green the acceptable level, yellow reaching into the unacceptable levels of CO2 and Red is the adequate max level of CO2 which is 1,000 Parts Per Million (PPM).



After all the Hardware was completed for the UNO R3, the coding of the components began in the Arduino Software.

I started by importing the Libraries of each of the sensors; these libraries add functionality and give purpose to the sensors, so they work in transmitting the information from the sensors to the UNO R3.

```
//LIBRARIES

#include <Wire.h>                          //I2C for OLED
    #include <Adafruit_GFX.h>                  //grafix library for OLED Screen
#include <Adafruit_SSD1306.h>              //OLED Driver
#include "dht.h"
```

The Define segment of the code determines the Pins that the Jumper Wires are connected to from the sensor on the breadboard to the UNO R3.

```
//DEFINE

#define DHTPin

#define anInput     A0                     // Analog feed from MQ135
#define digTrigger   2                     // Digital feed from MQ135
#define CO2Zero     55                     // Calibrated CO2 0 level
#define LEDBlue      3                     // Blue LED on pin 3
#define LEDGreen     5                     // Green LED on pin 5
#define LEDYellow    6                     // Yellow LED on pin 6
#define LEDRed       9                     // Red LED on pin 9
#define OLED_RESET   4                     // OLED reset on pin 4
#define DHT_pinA1    A1                    // Humidity and Temp Sensor on pin A1
```

The Library Call uses the Drivers of the sensors, so they work with the Arduino Software which dictates how sensors work in conjunction with the UNO R3. An instance of OLED called display has to be made to use the .960” OLED screen, and an example of DHT has to be made to use the DHT11 Temperature and Humidity Sensor.

```
//LIBRARY CALL

Adafruit_SSD1306 display(OLED_RESET);         // Create instance of OLED called display
dht DHT;
```

The setup of the Arduino Software code sets the outputs of information that the sensors relay and the inputs with the name of the sensors on the code. These names are used to identify the sensors for future reference. The MQ-135 pinMode gathers information of the sensor to display it on the .960” OLED Screen.

```
//SETUP

void setup() {

  Serial.begin(9600);
  Serial.println("CO2 PPM, Humidity % & Temperature Sensor in C\n\n");

  pinMode(anInput, INPUT);                    // MQ135 analog feed set for input
  pinMode(digTrigger, INPUT);                 // MQ135 digital feed set for input
  pinMode(LEDBlue, OUTPUT);                   // LEDBlue set for output
  pinMode(LEDGreen, OUTPUT);                  // LEDGreen set for output
  pinMode(LEDYellow, OUTPUT);                 // LEDYellow set for output
  pinMode(LEDRed, OUTPUT);                    // LEDRed set for output
  Serial.begin(9600);                         // Serial comms for debuging
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C);  // Begin display @ hex addy 0x3C
  display.display();                          // Show buffer
  display.clearDisplay();                     // Clear buffer

}
```

The Main Loop repeats the current algorithm of the code segment permanently and the user sets the amount of time between those periods until it repeats the action again with the delay function. DHT was called in order to receive information that could be displayed on the computer using the **Serial**.

```
//MAIN LOOP

void loop() {

    DHT.read11(DHT_pinA1);

    Serial.print("Humidity = ");
    Serial.print(DHT.humidity);
    Serial.println("%  ");

    Serial.print("Temperature = ");
    Serial.print(DHT.temperature);
    Serial.println("C ");

    delay(2000);                              //Wait 2 Seconds before accessing sensor again.

int CO2now[10];                               // int array for CO2 readings
int CO2raw = 0;                               // int for raw value of CO2
int CO2comp = 0;                              // int for compensated CO2
int CO2PPM = 0;                               // int for calculated PPM
int CO2avg = 0;                                  // int for averaging
int grafX = 0;                                // int for x value of graph

  display.clearDisplay();                     // Clear display @ beginning of each loop
```

The code below shows how the CO2 was calculated using a loop that repeats 10 times over a timespan of 10 seconds to read levels of CO2 and adds those samples together and divides them to get the CO2 levels in Parts Per Million and maps the value of the CO2 levels as well.

```
  display.clearDisplay();                    // Clear display @ beginning of each loop

  for (int x = 0;x<10;x++){                  // Sample CO2 10x over 2 Seconds
    CO2now[x] = analogRead(A0);
    delay(2000);
  }

for (int x = 0;x<10;x++){                    // add samples together
    CO2avg=CO2avg + CO2now[x];

  }
  CO2raw = CO2avg/10;                           // Divide samples by 10
  CO2comp = CO2raw - CO2Zero;                // Get compensated value
  CO2PPM = map(CO2comp,0,1023,400,5000);     // Map value for atmospheric levels
```

The code shows how the Humidity and Temperature is gathered, since the Temperature and Humidity sensors do not have to be calibrated, the computer could display the concentration of the Humidity and Temperature by calling the function display and gathering information from DHT.

```
display.setTextSize(1);                     // Set text size
display.setTextColor(WHITE);                // Set text color
display.setCursor(0,0);                     // Set cursor
display.println("TEMP =        HUM =");
display.print(DHT.temperature);
display.print(" C       ");
//display.print("Hum: ");
display.print(DHT.humidity);
display.println(" %");
//display.println(" ");                     // Skip a line
display.setTextSize(1);
display.print("  CO2 Lvl | ");             // Print title
display.print(CO2PPM);                      // Print CO2 PPM
display.print(" PPM");                      // Print Units
Serial.print("CO2 = ");
Serial.print(CO2PPM);
Serial.println(" PPM");
grafX = map(CO2PPM,0,2000,0,127);           // Map value to screen width
display.fillRect(0, 25, grafX, 50, WHITE);  // Print graph 400min 1000max
display.display();                          // Show the buffer
```

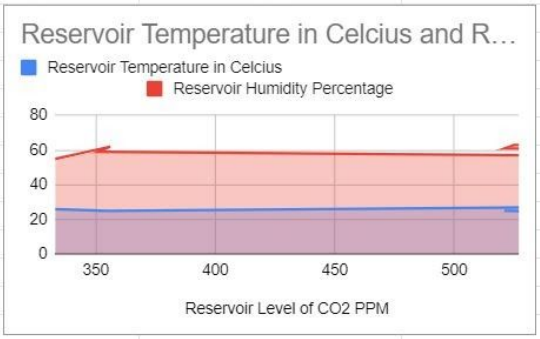
The code tells the lights whether they should turn on or stay off based on the value of the CO2 level.

```
  if(CO2PPM<415){
  digitalWrite(LEDBlue, HIGH);     // Turn LEDBlue on
  }
  else {
    digitalWrite(LEDBlue, LOW);    // Turn LEDBlue off
  }
  if(CO2PPM>415 && CO2PPM<450){
    digitalWrite(LEDGreen,HIGH);   // Turn LEDGreen on
  }
  else {
    digitalWrite(LEDGreen,LOW);    // Turn LEDGreen off
  }
  if (CO2PPM>450 && CO2PPM<999){
    digitalWrite(LEDYellow,HIGH);  // Turn LEDYellow on
  }
  else {
    digitalWrite(LEDYellow,LOW);   // Turn LEDYellow off
  }
  if(CO2PPM>999){
    digitalWrite(LEDRed,HIGH);     //turn LEDRed on
  }
  else {
    digitalWrite(LEDRed,LOW);      //turn LEDRed off
  }
}
```

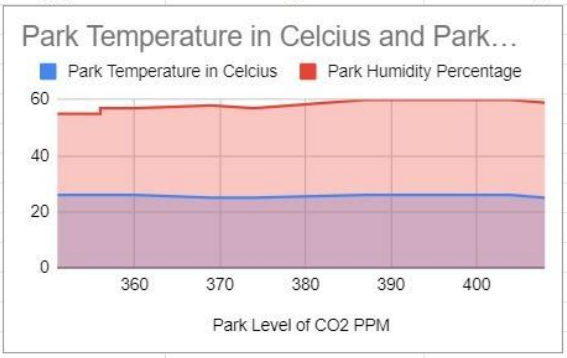
After the Arduino Software and Hardware experience were completed, it was time to measure CO2 levels. The CO2 levels were measured in Ellsworth Park in Union City New Jersey and comparing those measurements to the Weehawken & Union City Reservoir Park to graph a predictive model of the data collected on the CO2 based on the Temperature and Humidity. During both of the collections pictures of CO2, temperature, and humidity levels were taken in both areas from the .960" OLED Screen. A total of 10 recordings of the Humidity, Temperature and CO2 data levels were taken from each environment. To find the correlation between temperature and humidity, the dew point was calculated with the equation Td=T((100-RH)/5. Where Td is the dew point, T is temperature and RH is the relative humidity in percent. The average dew point was calculated using the average relative humidity and average temperature in both areas. The average dew point at the local park was 17.26 Celsius and the average dew point at the local reservoir was 17.24 Celsius. The dew point temperature indicates the temperature at which the air reaches the max value of water vapor before it will condense into liquid water.

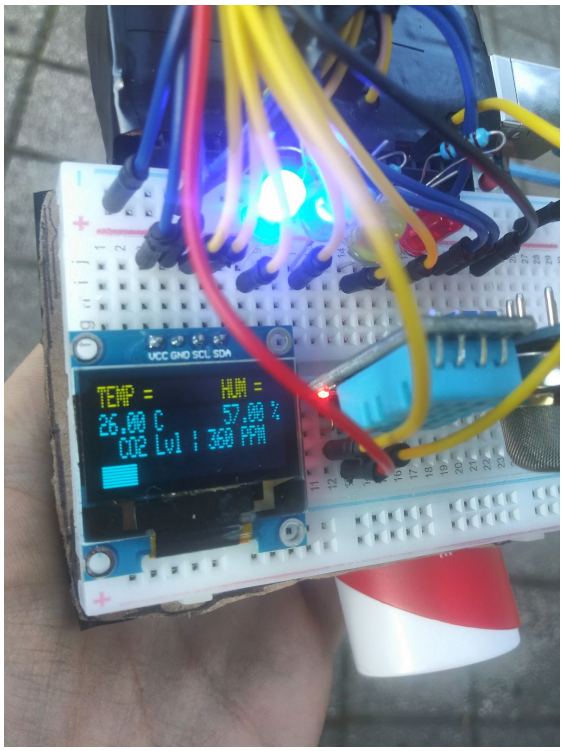
Results:

Around The Reservoir		
Reservoir Level of CO2 PPM	Reservoir Temperature in Celcius	Reservoir Humidity Percentage
333	26	55
356	25	62
351	25	59
534	27	57
687	26	59
692	25	57
525	25	61
521	25	61
525	25	63
530	24	63



In The Park		
Park Level of CO2 PPM	Park Temperature in Celcius	Park Humidity Percentage
408	25	59
404	26	60
396	26	60
387	26	60
374	25	57
369	25	58
360	26	57
356	26	57
356	26	55
351	26	55





Conclusion: The results show a correlation from my linear regression predictive model and the collected data of CO2 from the Humidity and Temperature sensor. Temperature is directly related to a rise in CO2. The graph shows that when CO2 levels increase, temperatures rise too. The dew point temperature at the local park was 17.26 Celsius. The average dew point at the local reservoir was 17.24 Celsius. The results show when temperature increases, relative humidity decreases because when the temperature rises it is further from the dew point temperature value causing less atmospheric water vapor. The local reservoir had a higher dew point temperature because of the surrounding water. The CO2 reservoir data illustrates when CO2 decreases, temperature decreases closer to the value of the dew point temperature rising the percent relative humidity of the DHT Sensor. CO2 therefore affects temperature and humidity directly.

References

Liu, Y., Ni, X., Wu, Y., & Zhang, W. (2017). Study on effect of temperature and humidity on the CO2 concentration measurement. *IOP Conference Series: Earth and Environmental Science, 81*, 012083. doi:10.1088/1755-1315/81/1/012083

How to Set Up the DHT11 Humidity Sensor on an Arduino. (2018, June 22). Retrieved from http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/

Tripoli, S., Santos, R., John, Bello, S., Gorki, Evans, T., . . . Murta, J. G. (2019, July 25). Guide for I2C OLED Display with Arduino. Retrieved from https://randomnerdtutorials.com/guide-for-oled-display-with-arduino/

K, D., Neal, Ray, Mishra, V., Doetjes, R., Fathur, . . . Microcontrollers Lab. (2018, July 21). Interfacing of MQ-135 Gas Sensor with Arduino. Retrieved from https://microcontrollerslab.com/interfacing-mq-135-gas-sensor-arduino/

Graphic: The relentless rise of carbon dioxide – Climate Change: Vital Signs of the Planet. (2019, June 12). Retrieved from https://climate.nasa.gov/climate_resources/24/graphic-the-relentless-rise-of-carbon-dioxide/

Lindsey, R. (2018, August 01). Climate Change: Atmospheric Carbon Dioxide: NOAA Climate.gov. Retrieved from https://www.climate.gov/news-features/understanding-climate/climate-change-atmospheric-carbon-dioxide