University of Illinois
at Urbana-Champaign

Measure Of Temperature, Humidity, and Pressure Over a Farm Field Using a Drone

Michael Bengston
Michael Habisohn
Justin Languido
Christian Williams

ABSTRACT

Field diagnostics provide crucial information to farmers that can influence their decisions come planting season. Unfortunately, the main method to obtain important information (thermal imaging) is both costly and quickly becomes obsolete. To address these issues, we have assembled a device made up of sensors that will not only provide the farmer with more information than just the temperature, but at a much lower price-point. The device will be flown Further testing will compare the accuracy of these sensors to more expensive alternatives.

BACKGROUND

The growth of a crop is dependent upon four major variables: solar radiation, air temperature, humidity, and precipitation. This work focuses on the last three variables. Measurement of these variables is difficult in Illinois due to the rapid changes in weather. Because of the rapid variations in Illinois weather, Hollinger and Angel focused on what is referred to as "normal" weather conditions. "Normal" weather conditions are based on observations of the climate of an area over a 30 year period, and provide a baseline for what the climate is usually like.

Temperature has the greatest effect on both the speed at which a crop grows and its structural stability. There are four temperatures that are of interest to farmers when they are looking at a potential area to plant crops. The first, referred to as "absolute minimum," is the minimum temperature at which the crop is able to grow at all. The second, "optimum minimum," is the lowest mean temperature where the crop can still grow optimally. The third is the "optimum maximum," the maximum mean temperature at which the crop can grow optimally. The fourth temperature is the "absolute maximum," the hottest temperature that still allows the crop to grow. Corn was planted in the field where the measurements in this work were taken. The values for the four temperature thresholds for corn, in order, are: 10°C, 15.6°C, 32.8°C, and 47.2°C. The above temperatures are those of the air, but the temperature of the soil is also relevant. The soil temperature determines the amount of ammonia-based fertilizer needed. While the soil temperature can be estimated by taking an average of the air temperature over a seven day period, this method usually overestimates the actual temperature by approximately 1°C (Hollinger and Angel).

Precipitation, more specifically the timing of the precipitation, is another important aspect to take into consideration when evaluating potential levels of crop production. Similar to temperature, there is an optimal range of precipitation volume. For instance, the combination of cooler temperatures and large amounts of precipitation during germination can lead to various diseases or soil saturation. Too little, however, leads to weaker roots, which will make the crop vulnerable if a drier-than-normal season follows (Hollinger and Angel). Precipitation and humidity are related, in that an area with high humidity, will oftentimes have a large amount of precipitation.

APPLICATIONS

The farming communities focus, as of now, is on the thermal profile of fields. A company in the area who offers thermal imaging services explained that use a thermal camera which is strapped to a small airplane and flown over the field of interest. After all is said and done, the cost comes out to around $5,000 (reference) for a single flight. While the data are useful, they only gives the farmer an idea of the thermal profile of the field for a single day. The device fabricated in this work is not only cheaper (depending upon the drone chosen), it offers up-to-date information about the field and a level of customizability that the thermal imaging services do not.

To measure the relevant quantities, we created a device utilizing the following components: an Arduino Mega 2650, a Bosch BME680 Breakout Board , an MLX 90614 IR Thermometer, a MT3339 All-in-one GPS, and an Adafruit MicroSD card breakout board. The BME680 measures the air pressure, humidity, and air temperature at a given point. To check the soil temperature against the temperature of the air, the MLX 90614 IR Thermometer was pointed at the soil. In order to generate the profile of a field, the spatial coordinates at which each data point is taken must be known. This is the function of the MT3339 All-in-one GPS. After all of the measurements have been taken, they are stored on an MicroSD card using the Adafruit MicroSD card breakout board. The various electronics were then soldered onto a printed circuit board (PCB). The PCB was placed into a roll-cage and screwed into the accessory bay of a 3D Robotics Solo Quadcopter Drone. The drone was then flown over a field at a height of approximately 25 ft (~7.62 m) at a velocity of around 0.5-1.0 m/s.

HARDWARE

Data collection necessarily involves utilization of specialized hardware. Choice of components was determined by budget and ease of use. Ultimately, this led to the development of a robust data acquisition unit at a fraction of the price of commercial products for similar applications. The electronic system consists of modular sensor packages communicating with a single microprocessor. In our case, we utilize the Arduino Mega 2560 (https://store.arduino.cc/usa/arduino-mega-2560-rev3); a microcontroller board running on the ATmega2560 microprocessor. This package allows for “quick and easy” microcontroller project development. The sensor packages we utilize come in the form of breakout boards developed by the open-source hardware company, Adafruit Industries. The following sub-sections of this paper will apprise the reader of the hardware specifics for this project.

COMMUNICATION

As a preface to the discussion of the breakout boards, this section will explain the communication protocols used for our project and why we chose the one we did.

The two most common short-distance communication protocols for electronic components are SPI and I^2C. The former requires a wire to each and every node on the communication network. For example, if there were one master and three slave units, the

circuit would require 3 slave select lines in addition to the MOSI/MISO/SCLK lines shared between all nodes. *Figure 1* illustrates SPI communication between master and slave nodes.

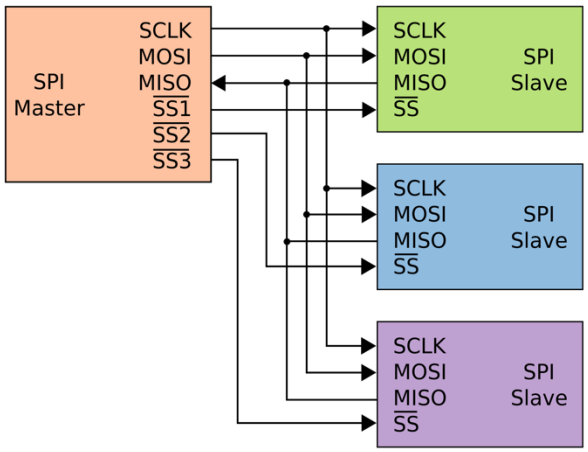


Figure 1: SPI Communication Block Diagram

I^2C, on the other hand, is a more compact system of hardware communication, in that all slave nodes are connected to the master via the same two wires. In fact, I^2C has the added benefit of allowing multiple masters on the same network. *Figure 2* illustrates I^2C communication between two masters and two slave devices. Notice, there are only two wires between all nodes.

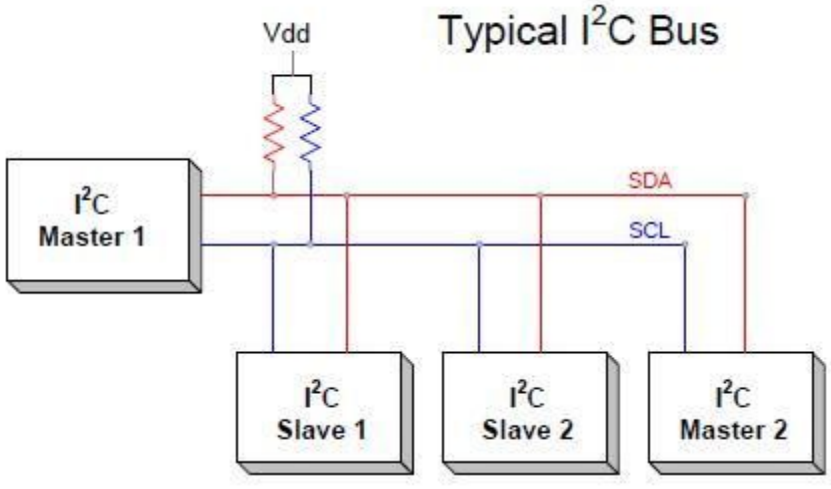


Figure 2: I2C Communication Protocol Block Diagram

I^2C is the ideal communication protocol for our device due to the benefits of the 2-wire system as well as the unnecessity for extremely high-speed communication. Conveniently, all our sensors are capable of using the I^2C communication protocol.

Bosch BME680 Breakout board:

The BME680 breakout board provides the microcontroller with a series of environmental sensing capabilities. The sensor on-board is the BME680 from Bosch- as the

name implies. The sensor measures temperature, humidity, barometric pressure, and levels of VOC gas, or volatile organic compounds. The chart below describes the sensors' operating parameters for the BME680.

PARAMETER	
Operation Range (for full accuracy)	
Pressure	300 - 1100 hPa
Humidity	0 - 100%
Temperature	-40 - 85°C
Avg. Current Consumption (1 Hz refresh)	3.7µA
Gas Sensor	
Response time	< 1 s
Sensor-to-sensor deviation	+/- 15 %
Output data processing	Direct output of Index for Air Quality
Humidity Sensor	
Response time	8 s
Accuracy tolerance	± 3 % relative humidity
Hysteresis	≤ 1.5 % relative humidity
Pressure Sensor	
RMS Noise	0.12 Pa
Sensitivity Error	± 0.25 %
Temperature coefficient offset	± 1.3 Pa/K

MLX 90614 IR Thermometer:

The MLX 90614 infrared thermometer is the only sensor on our PCB that did not require a breakout board. In fact, the package itself contains just 4 pins and a high-resolution ADC, or analog to digital converter. The sensor reads the amount of incident infrared light and extrapolates that data to attain a temperature measurement. This measurement is an average over a field of view that is projected out at 45° from the sensor window. The sensor resolution is 0.02°C for 2-wire interfacing- which we are using. Figure 3 illustrates the block diagram of the internal electronics for the MLX90614 family of infrared thermometers. Notice the high-resolution ADC and dedicated digital signal processor. This provides extremely precise sensor readings to the Arduino processor.

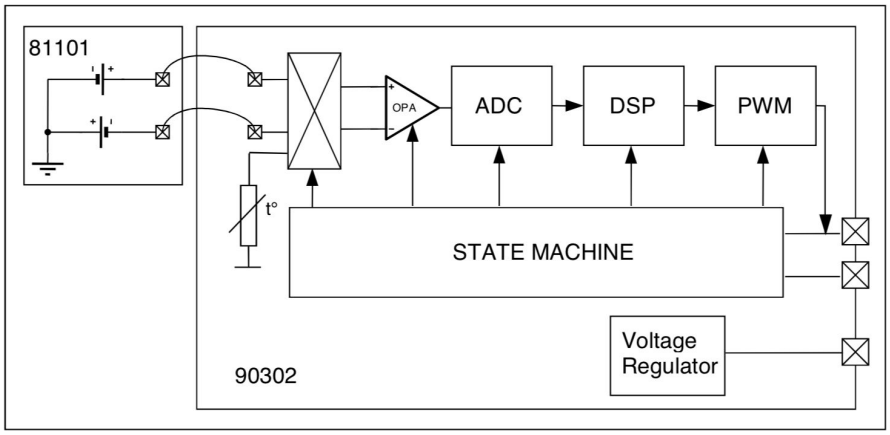


Figure 3: IR Thermometer Block Diagram

MT3339 All-in-one GPS:

The MT3339 GPS is onboard the "Ultimate GPS" breakout board from Adafruit. The MT3339 uses its own microprocessor to collect and parse data it receives from the satellites. The GPS tracks up to 22 satellites and has a build in antenna. The module is capable of 10 Hz location update rate at a very low power usage of 20 mA. The GPS will output NMEA sentences of various types. The most useful for our applications was the $GPRMC sentence, or Recommended Minimum Specific GPS/Transit data. This allows for the tracking of our sensor package at precise GPS coordinates. Our sensor device was programmed to read all sensor values at virtually the exact moment that GPS coordinates were received every 0.5 seconds. This allows us the ability to track exactly where and when sensor data was taken so that we may utilize offline analysis tools to further extrapolate data and provide more accurate analysis of the tested environment, in this case a corn field.

MicroSD Card reader/writer:

After collecting our sensors' data we need to store the information somewhere to examine at a later time. The MicroSD breakout board from Adafruit allows this. The breakout board is very simple as most of the hardware is, in fact, in the SD card itself. The board contains a port for the microSD card and a High Speed CMOS Logic Buffer. The SD card is written to by the DO and DI, or DataOut and DataIn pins of the Arduino board. Our Atmega processor is programmed to sequentially store lines of data containing our sensor values and pertinent information in a .csv format, or comma separated list. This format is very easily read by our offline program and parsed thereafter. The figure below is a copy of the .txt file from one of our runs.

```
19.96,992.31,34.31,247.33,175.99,3958.0629,8816.5019,18.01,10.79,64.42,51.42
20.03,992.29,34.21,247.52,175.99,3958.0676,8816.5019,18.05,10.71,64.49,51.28
20.09,992.27,34.12,247.33,175.99,3958.0708,8816.5019,18.09,10.37,64.56,50.67
20.14,992.29,34.03,248.26,175.82,3958.0764,8816.5019,18.13,10.41,64.63,50.74
20.20,992.33,33.95,246.96,175.82,3958.0786,8816.5009,18.17,10.33,64.71,50.59
20.26,992.33,33.89,247.33,175.48,3958.0788,8816.5009,18.17,9.89,64.71,49.80
20.33,992.33,33.84,247.52,175.48,3958.0788,8816.5009,18.15,10.33,64.67,50.59
20.38,992.37,33.79,246.59,175.65,3958.0786,8816.4990,18.19,10.31,64.74,50.56
20.42,992.37,33.77,244.94,175.14,3958.0786,8816.4980,18.21,10.27,64.78,50.49
20.46,992.37,33.79,244.58,175.31,3958.0791,8816.4970,18.21,10.37,64.78,50.59
```

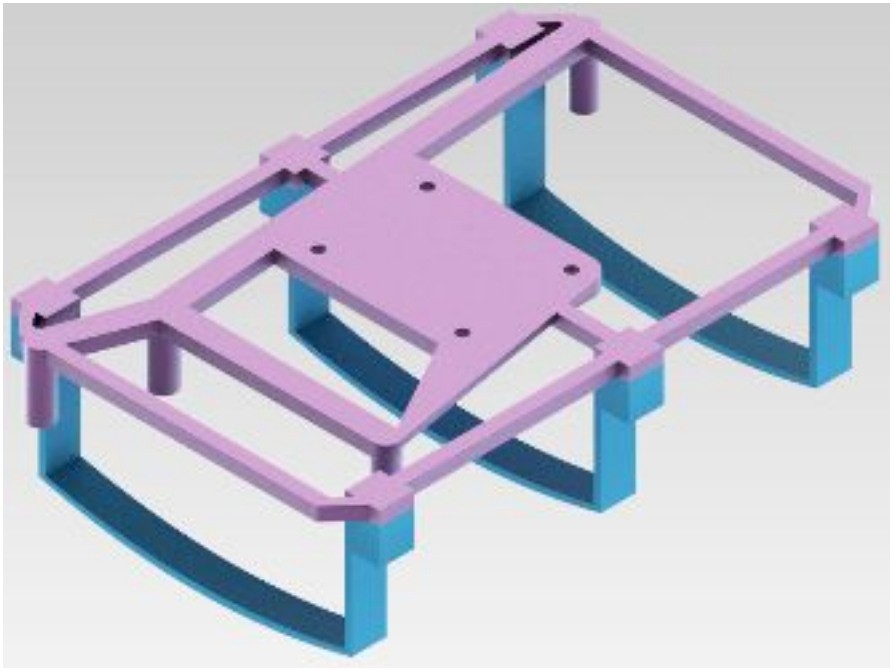
As you can see, sensor data is organized in a .txt file by comma separation. Each line represents a half second separation between data points.

THE ROLL CAGE DESIGN

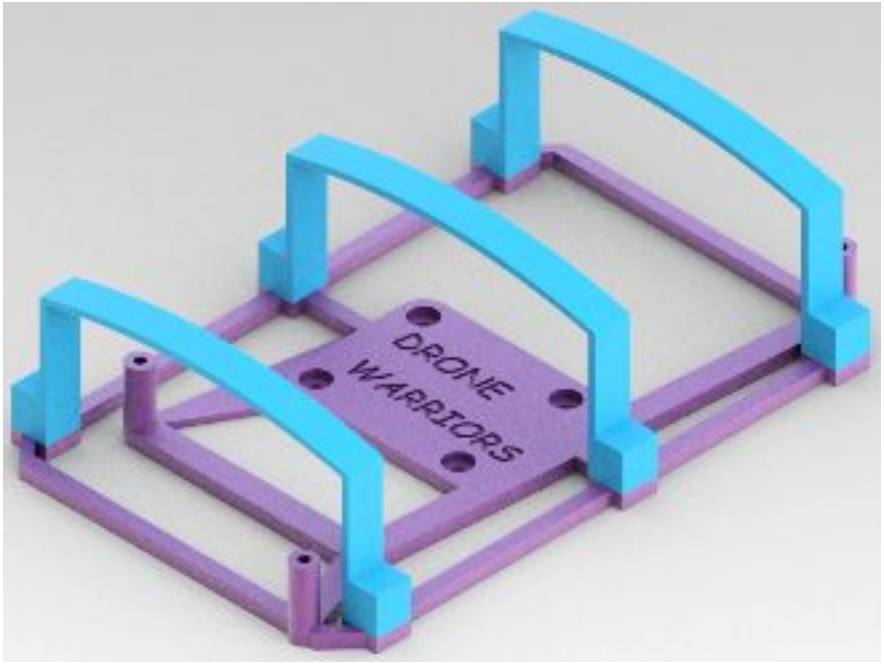
The design of the roll cage was subject to a few constraints. The design constraints required that the overall weight of the cage and device together weigh 1 pound at most. They also required the roll cage to be sturdy so as to protect the device should it be dropped, and it must have easy access to the printed circuit board, namely the battery and the SD card.Most importantly, the constraints also required the ability to securely fasten the roll cage to both the device and the drone.

To satisfy these constraints, the roll cage was designed to be open. Less material both in volume and surface area and more open space allowed for lighter weight and better measurements out in the field since the breakout boards were more exposed to the environment they were sampling. The crossbars on the underside of the cage protect the electronics should the device ever be dropped. The mounting holes on both sides of the device allow the user to use bolts to secure the device to the cage, and the cage to the drone.

To manufacture the roll cage, a 3D printer, the Ultimaker 2^+ was used. The material we chose to print the cage is polylactic acid. The benefits of using a 3D printer include the ability to print the cage from home or office rather than purchasing it in store or online, the ability for the user to alter the design, if needed, to suit the project at hand, and the ability to choose different materials to print with. The benefits of polylactic acid, and thus the reason we chose it, is its inherent durability and low density. It is also friendlier to the environment as it is derived from renewable resources and is biodegradable.



Top Orthographic View of Roll Cage



Bottom Orthographic View of Roll Cage

DATA ACQUISITION

The data acquisition program was written in the Arduino integrated development environment and was designed for use with the Arduino Mega 2560 Rev 3. This device was chosen because it is low in cost and because Arduino has designed their products to be user friendly. This is because Arduino microprocessors and the Arduino programming language are intended to serve as an introduction to programming and electronics for people without any previous experience.

As the microprocessor, the Arduino serves as the brains of the instrument package and contains the data acquisition program written by each member of the team. The data acquisition program links the Arduino to the breakout boards. This allows the Arduino to send and receive information to and from itself and the boards it is connected to.It will also collect data from the sensors and GPS and write the on the text file in the SD card. This collect and write process is done at a frequency of 2 Hz.

The program starts by importing several libraries, one for each board attached to the Arduino. These libraries are what allows the Arduino to communicate with the sensors. The code then generates the .txt file that will store the collected data. It also configures the GPS forcing it to update its position at a frequency of 1 Hz.

In order to collect data from the GPS and sensors and write that data to the .txt file, the code opens the .txt file for writing and begins to request readings from each sensor. For most sensors, this request is a single line of code that asks the sensor to return a specific measurement type. For example, it will ask the BME680 to return its current temperature reading, then later ask the BME680 to return its current air pressure reading. This is not the case for the GPS, however. The code must first ask if the GPS has valid data, then load only valid data into its memory. Only then can the Arduino receive nonzero data from the GPS. Once the data from each sensor and the GPS have been collected and saved, the .txt file is closed and reopened to begin the process again.

WHAT DID WE DO?????? New section here

OFFLINE ANALYSIS

The purpose of this project is not just to collect useful data regarding the environmental characteristics of a crop field; one must also analyze this data to produce useful results. Hopefully, these results may be of use to those who manage the crop field. We have attained this by means of a proprietary offline Python script. The script will be generally explained in the following subsections.

Parsing

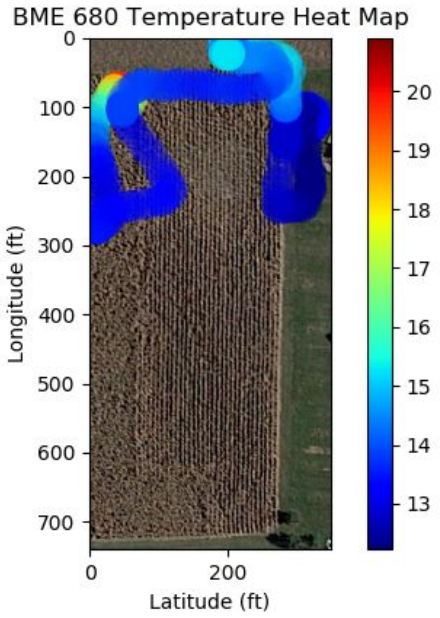
As aforementioned, the Arduino microprocessor compiles real time sensor data into a comma separated value, or .csv, file. The python script first reads through the .csv file from the micro-SD card and turns each column of the file into individual arrays. In doing this, the script

can now read each sensor array as a convenient 1D list of values that are separated by our sensor refresh/write rate of 2 Hz. Furthermore, the arrays can simply be acted upon by functions, or groups thereof. Now, the script passes through the arrays and finds the index of the GPS coordinate arrays that are null, or zero. This means that the GPS had not gotten a fix on a satellite yet and so was not receiving latitude and longitude information. The index of these elements are saved. Subsequently, elements of all the other arrays that correspond to the saved indices are removed.

This is not, however, the end of the preliminary parsing operation; the Python script now conforms the data to standards that we can more easily extrapolate. The 1D arrays of sensor data are in a text format. These need to be changed into floats, numerical data that has a decimal place. This enables us to perform mathematical operations on the data. The first of these operations is the conversion of GPS coordinate data from the DDMM.MMMM format to feet. One might wonder how this is possible without a relative zero. We subsequently set the GPS.coordinates minimum to zero for both the latitude and longitude array. This provides a relative zero for position coordinate system in feet by feet (x by y).

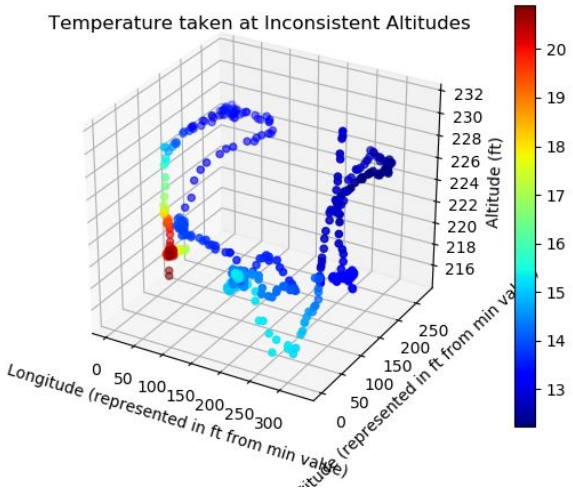
Plotting

Now that parsing of the data is complete, we have the script begin plotting of the sensor data arrays. We have two approaches to the plots, but the terminology needs to be clarified. First, we have plotted the sensor data values against the GPS latitude and longitude coordinates (in feet at this point). In the conventional sense, this is a 2D graph. However, it is, in fact, 3D- the third dimension being a scaled, color coded, value of whichever sensor data value is being viewed. So, in reality, it is 3D but spatially 2D. See the figure below.

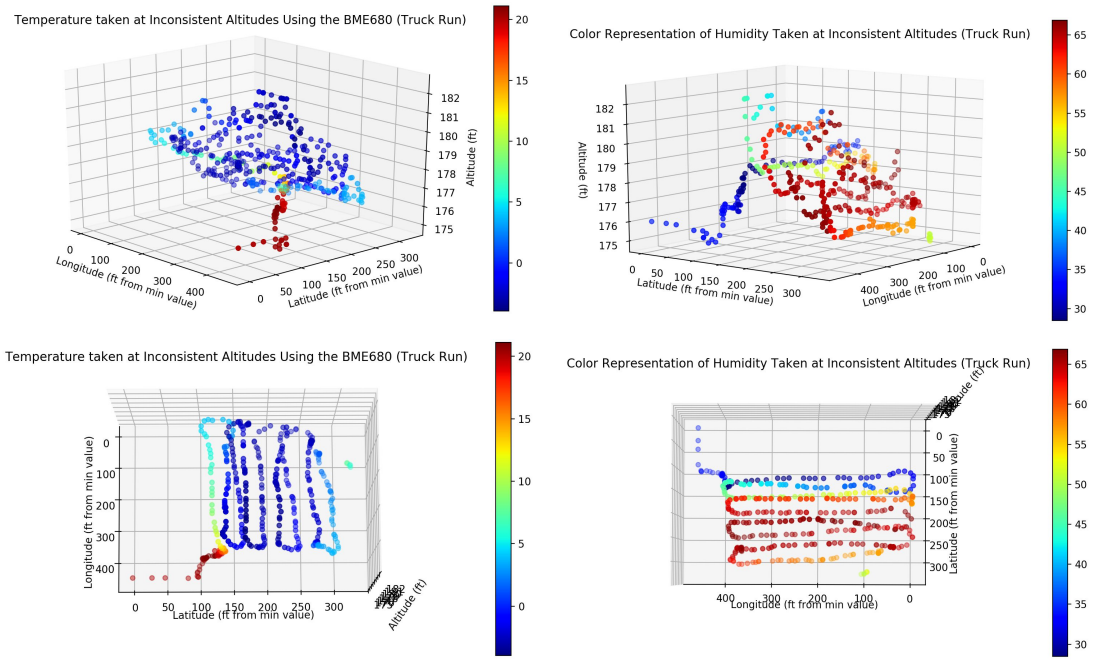


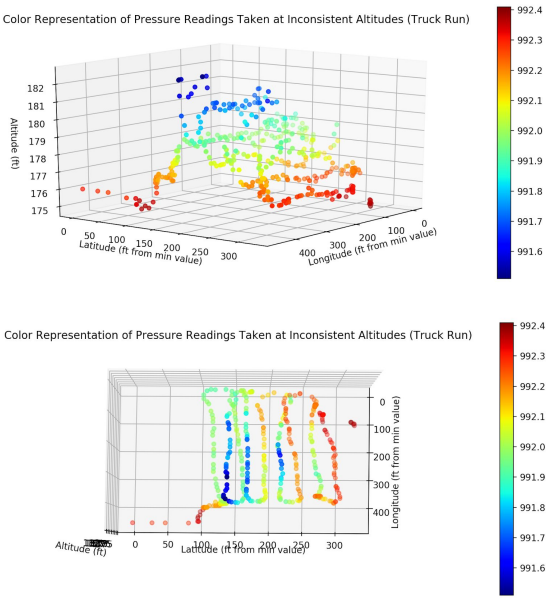
As great as these results are, we wanted to go one step further and add an extra spatial dimension, resulting in 3D spatially and 4D, including colored value, plots. As we were flying a

drone above the field, we felt that it is important to include the altitude in the plot as this could significantly effect sensor data. In the future we intend on projecting the field on this 4D graph as well.

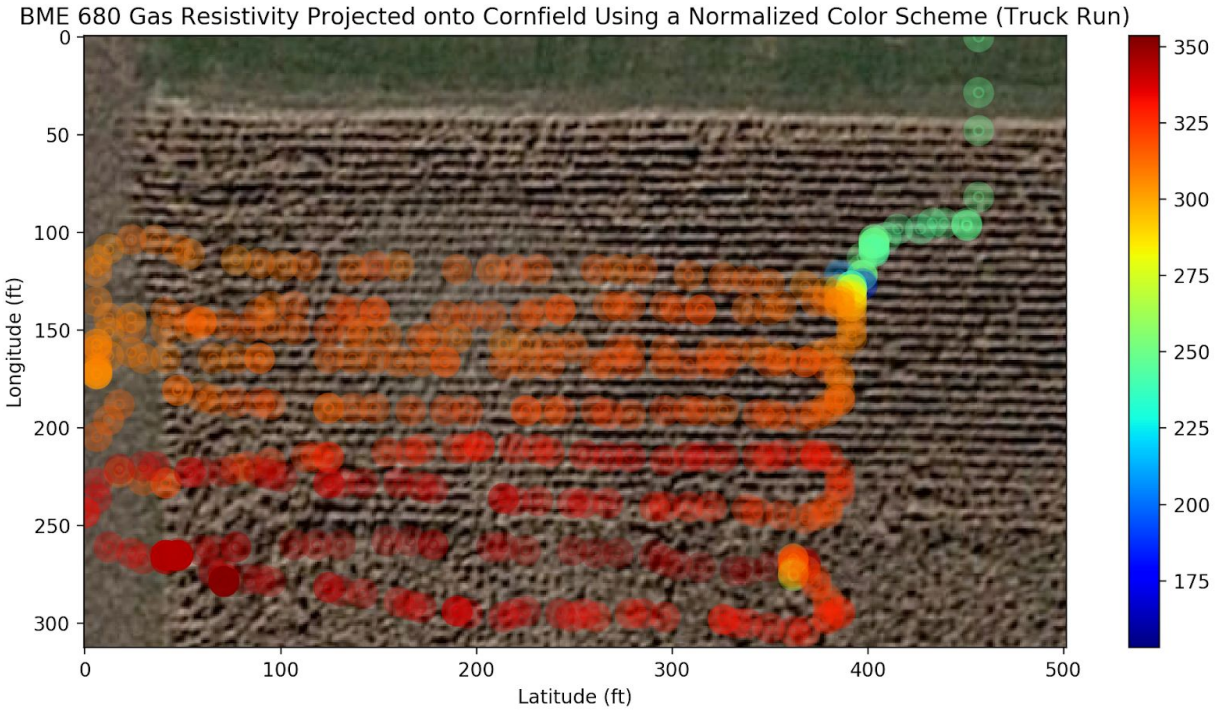


As shown in some of the plots, the script is capable of interpolating data values to create a color coded surface plot between discrete sensor/coordinate points. This provides visual insight into the environmental characteristics of the field. Unfortunately it was difficult to be able to draw any sort of conclusion related to the fields with so few data points. Below we show how a much larger data set can produce results that are easy to draw conclusions from. Each of these plots are from a truck run to show the capabilities of the sensor package.





Using the top view of the field scatter plots we are able to tell if there are any abnormalities. If for some reason the humidity or temperature in one part of the field is inconsistent with the rest of the field then the farmer knows problems could arise from that area and act accordingly. Being able to see the altitude allows us to tell if the color abnormality comes from the field, or if the data could just be a result of the drone's altitude.



We are also able to create a color map on the field, mapping using the gps coordinates.

FUTURE WORK

In future iterations, the group plans on creating a variety of device packages that can be chosen based upon the needs of the farmer. For example, if a farmer wants to know that the temperature profile looks like at a height of ~15ft above a field, it does not make much sense to provide them with a device that also measures the evolution of organic compounds from the field. The specificity of a device can also allow for a smaller payload. A smaller payload has two potential benefits: smaller (cheaper) drones, and longer flight times. Another choice left to the farmer is the price they wish to pay for the chosen instruments. The group also plans to investigate how much more accurate the data from the cheaper options is than the expensive options.

These accuracy tests are going to be done in the same field where the preliminary tests took place. Acquisition of drone pilot licenses and the drone itself will allow for examination of the entire field, in a more densely-packed manner than prior tests. Data will be taken from various heights to determine at what altitude the data is no longer helpful.

References

Hollinger, Steven E., and James R. Angel. *Weather and Crops - Crop Sciences Department*. Illinois Agronomy
Handbook,extension.cropsciences.illinois.edu/handbook/pdfs/chapter01.pdf.