Paper ID #35922

Limiting Overuse of Water in Agriculture by Monitoring Water Content

Miss Isabella Serrano, Northeastern University
Ms. Kaitlyn Williams-Koester
Mr. Kaden Clark, Northeastern University
Mr. Blade Rocket Kalikow, Northeastern University
Bennett Hartley
Dr. Bala Maheswaran, Northeastern University

Bala Maheswaran, PhD Northeastern University 367 Snell Engineering Center Boston, MA 02115

Monitoring and Correcting Water Content in Soil

Isabella Serrano, Kaitlyn Williams-Koester, Kaden Clark, Bennett Hartley, Blade Kalikow, and Bala Maheswaran
College of Engineering
Northeastern University

Abstract

Our world's demand for water continues to surge, while there is a fixed amount of fresh water on the planet. As human-induced climate change affects the world in potentially irreversible ways, our access to freshwater continues to decline. The considerable use of water in agriculture is frequently unnecessary as farmers use water-intensive irrigation techniques. Doing so is beneficial in having low equipment costs, but it wastes water due to evaporation, infiltration, and runoff [1]. Additionally, irrigating a plant too much can negatively impact garden health and agricultural yields. Overwatering occurs when a plant receives too much water, which waterlogs the soil and prevents the plant from absorbing an optimal amount of oxygen for survival. Overwatering plants essentially drown crops, creating additional waste products. The environmental concerns from our world's overuse of water require changing how we use agricultural irrigation. This paper presents a proof of concept of an automated watering system that monitors and corrects the water content in the soil, specifically in situations where the plant needs watering.

Our project presents a solution to how society waters plants and uses the world's available freshwater resources. Our world's current depletion of freshwater resources is not sustainable and requires change. We addressed these environmental concerns by developing a sustainable watering device that allows individual plants to receive precise amounts of water for their measured water content in the soil. Although our design functions on a potted plant where runoff and evaporation are not an issue, the application of our design could reduce these effects in an agricultural setting. This design also allows for remarkable convenience since users do not need to waste time watering the plants. Our system has future applications, which can appeal to a large audience of individuals, gardeners, and agriculturalists, drastically reducing our planet's water use.

Introduction

Water is becoming increasingly more valuable every day. Freshwater is needed in agriculture to feed billions of people globally, and overwatering wastes this resource. Currently, agriculture irrigation accounts for 70% of water use worldwide [2]. We recognized that many farming practices–flooding, sprinklers, and furrowing–release an unnecessary amount of water to plants. Doing so is not only wasteful but also harmful to plant life. Our global freshwater supply is declining, so as a team, we wanted to create a sustainable farming practice, which would limit water waste.

To combat this issue, we have designed and developed an innovative automatic watering device, which operates with minimal human intervention and allows for the water content in the soil to be consistently measured. It only waters the plants when necessary. There are four main components of our device: a water tank, a solenoid valve, an Arduino RedBoard, and a gypsum block. Our design allows the user to input their plant's preferred resistance threshold into a program, which would consequentially release a precise amount of water when the gypsum block (submerged in the soil) detects that the soil water content is low.

Our design promotes both house planting and urban farming. Approximately one-quarter of the world's urban population is supplied with food coming from urban farms [3], and as cities continue to grow, the practice is becoming increasingly more popular and needed. Our device can contribute to the success of urban farming as it does not require frequent human monitoring. The convenience and sustainability of the product would both benefit the environment and promote local farming, consequentially reducing agriculture's greenhouse gas emissions.

Method and Approach

Figur1a: Design of prototype

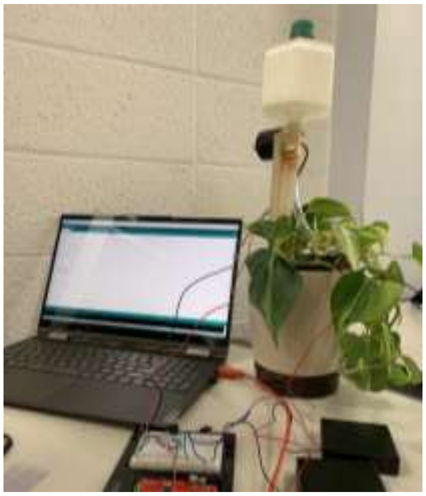


Figure 1b: Prototype

Our design consists of two primary systems: a sensor for monitoring soil water content and a controlled water tank to correct soil water content (Figure 1a and 1b). Our monitoring sensor is a gypsum block that electrically measures the water saturation of the soil [4] (Figure 2). We embedded two floating galvanized nails in a combination of Plaster of Paris and water [5] (Figure 3). By applying a voltage across the nails and measuring the current in the circuit, the resistance of the block can be calculated using Ohm's Law [6]. When the gypsum block is saturated with water, electrons can pass between the nails, causing a low resistance reading. This gypsum block is inexpensive to produce, and it provides accurate measurements of soil moisture content [7].

Figure 2: Design of gypsum block

Figure 3: Gypsum block

Our moisture-correction system consists of a custom waterproof tank, a solenoid valve, and a water distribution tube. We 3D-printed a hexagonal 750 mL tank (Figures 4 and 5) and attached a screw-top lid to prevent spillage. This tank has a threaded bottom spout that connects to a 12-volt electric solenoid valve that we purchased from Beduan [8]. A 3D-printed connector helps water flow from the valve into a distribution tube.

Figure 4: Design of water tank

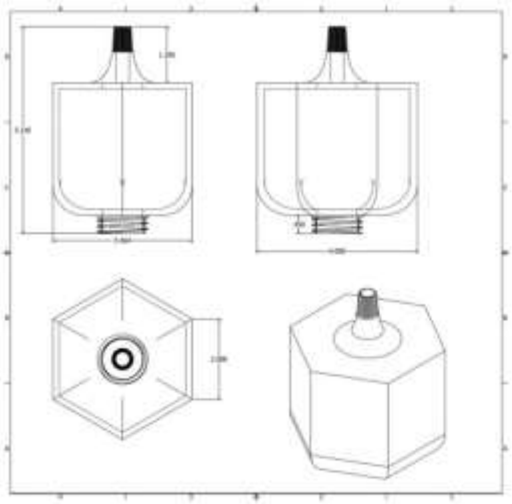


Figure 5: Isometric drawing of water tank

We designed a circuit on a SparkFun Breadboard to process the gypsum block readings and control the solenoid valve (Appendix 1). One part of the circuit is essentially an ohmmeter to determine the resistance of the gypsum block. We used a SparkFun RedBoard to calculate the resistance and decide when to water the plant. The RedBoard was connected to a transistor that controls when the solenoid valve connects to a 12-volt battery pack. The transistor works as a digital switch. We originally tried to power the valve directly through the RedBoard, but the voltage was insufficient to turn the valve. We then set up the voltage amplification circuit using the transistor and a 12-volt battery pack [9].

The final part of our design approach and methods was our code (Appendix 1). The goal of our code was to measure the resistance of the gypsum block and determine when to send voltage to the solenoid valve. The code starts by declaring variables that will be used to calculate the resistance and then compare it to our saturation resistance threshold. The setup of the Arduino code sets up the Serial Monitor, the solenoid pin (A0) as an output, and asks the user for their resistance threshold for their plant. The loop of the Arduino code first measures the amperage from the gypsum block and then uses Ohm's Law to convert the amperage to resistance [6]. The code then compares the current resistance to the saturation resistance threshold and determines if the plant needs watering. The plant is watered by using the code to open the solenoid valve for five seconds and then close the valve. The code repeats every minute.

Results and Discussions

I. Gypsum Block Readings

Our initial gypsum block data was promisingly predictable but not particularly smooth or regular (Figure 6). As we collected more data through additional trials, we improved our methods of data collection and analysis. We changed the resistance of the known resistor in the ohmmeter circuit to be closer in magnitude to the gypsum block resistance. We also calculated a moving-average curve to eliminate outliers from the graphical data (Figure 7). We experimented with alternate axes, including a logarithmic resistance scale (Figure 8). These methods allowed us to refine our data into a smooth, clear graph showing the water-induced change in resistance (Figure 9). Over the course of two days, we did an experiment with the gypsum block in the soil, in order to collect actual data. On the start of the two days the plant was watered and the resistance if the gypsum block was measured every minute for the following 48 hours. The rate at which the resistance was increasing was exponential over the allotted time (Appendix 2). At around hour 38 the resistance started to decrease rate which leads us to believe it is reaching its resistance threshold for the saturation of the soil, but the resistance did continue to go up increase at a low rate after the 48 hours. A lot more data is needed before producing the product.

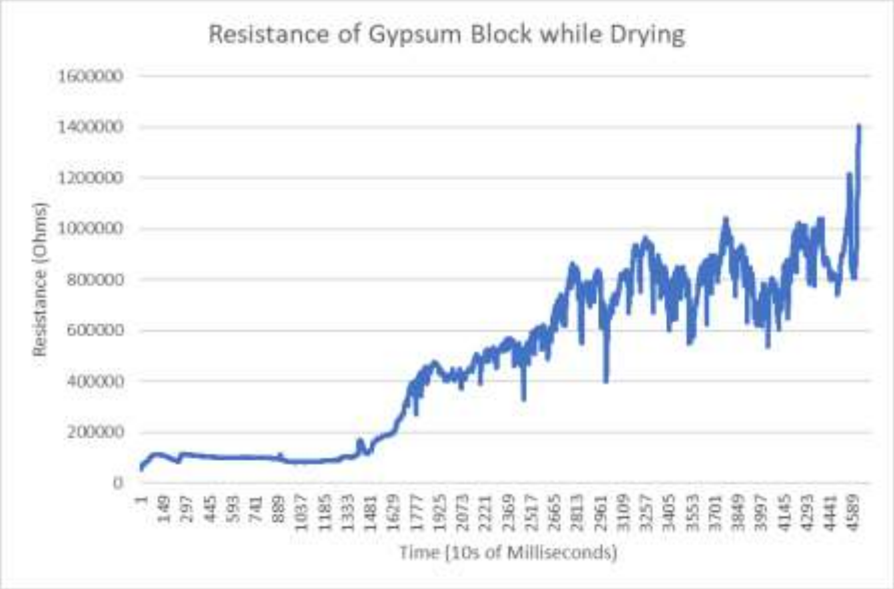


Figure 6: Graph of the gypsum block's resistance while drying

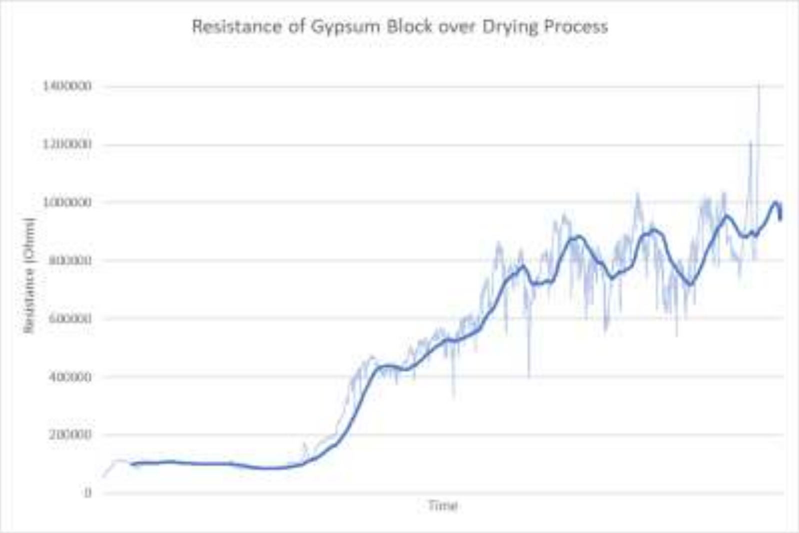


Figure 7: Graph of gypsum block's resistance over drying process

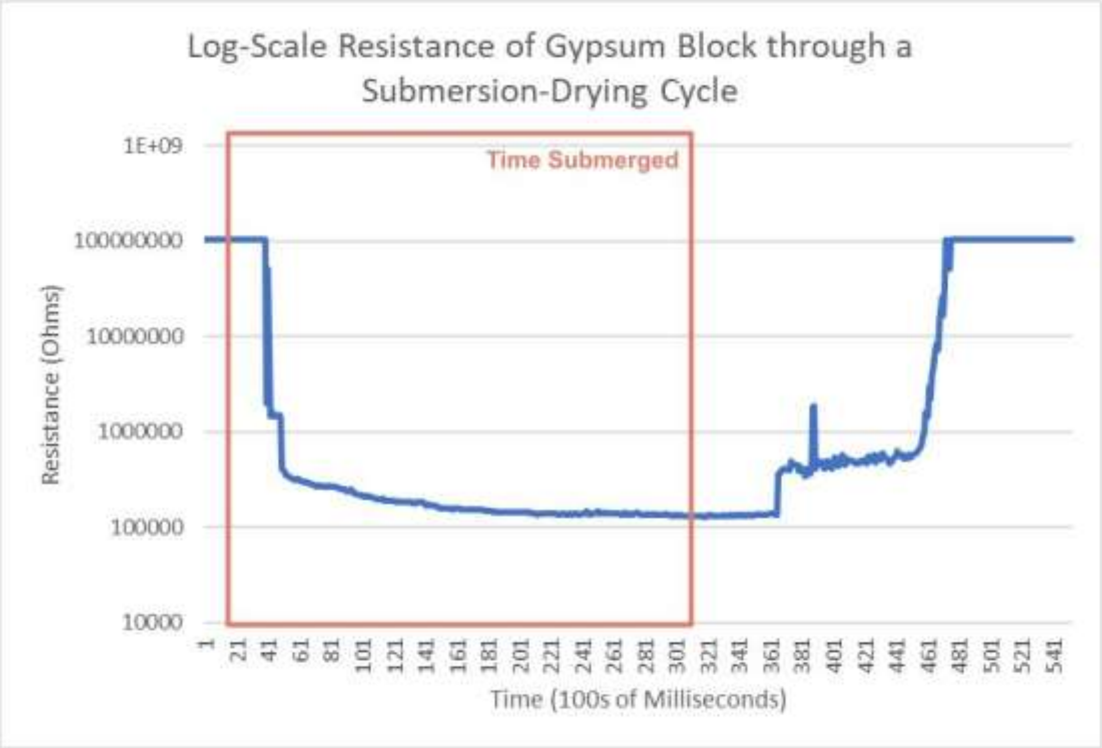


Figure 8: Graph of gypsum block's resistance through a submersion-drying cycle

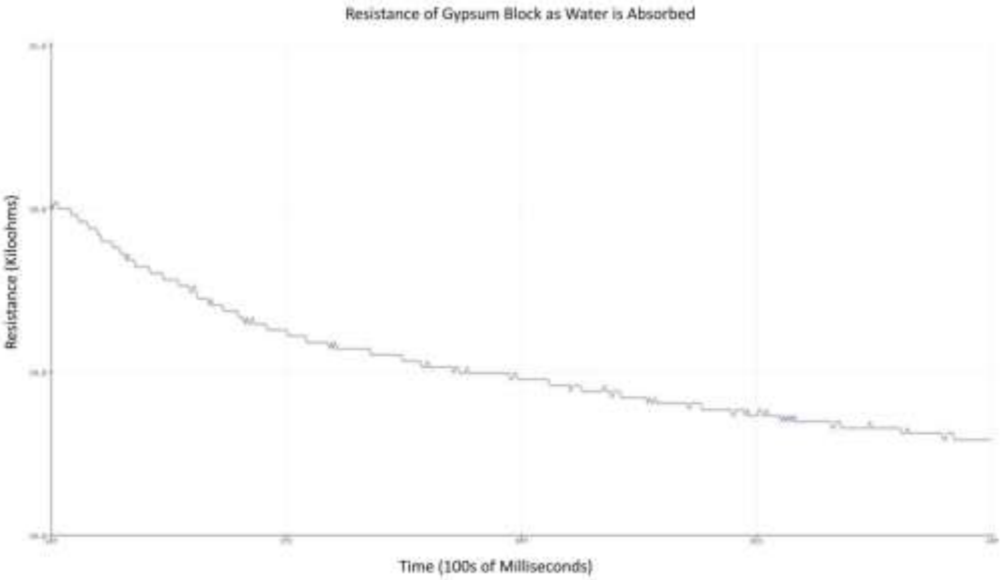


Figure 9: Graph of gypsum block's resistance as water is absorbed

II. There were two elements of our data that stood out: the surprising level of general accuracy and the duration of a resistance change. Fabricating the gypsum block was very inexpensive and did not require a high level of precision or previous knowledge. We did not expect our imprecisely built sensor to be very accurate, but it performed unexpectedly well measuring relative moisture content, as illustrated in our graphs. Another surprising result was the extended time needed for the gypsum block to recover from high water saturation. We had not considered the length of time required for water to vacate the gypsum block as soil moisture content decreases, so the duration of time needed to see the resistance increase after removal from wet environments was unexpected.

III. Solenoid Valve

The solenoid valve was easily implemented into our system and allowed for precise amounts of water to be transferred from the water tank to the tube, which flows into the plant. This element of our project worked effectively, but it was the most expensive element of our project, costing $19.00.

IV. Water Tank

Using Autodesk Inventor, we created and 3D printed this 750 mL tank. Although it completed the job of housing the water and creating an airtight seal with the solenoid valve, the tank was not particularly aesthetically pleasing. It was relatively large compared to the size of the potted plant. We could have made the water tank larger, which would dramatically increase the product’s utility, but it would have negatively impacted the aesthetics of the assembly. If we were to mass-produce our product, we would use molded plastic to produce the water tanks, dramatically reducing its cost and production time.

V. Plastic Tubing

The plastic tubing, which transported the water from the solenoid valve to the plant, was an affordable option that worked very effectively. However, in a future design, by using a soaker-hose style with many holes through the sides, we could more effectively distribute the water evenly across the soil’s surface.

VI. Solar Panels

Our initial goal was to use solar panels to power the SparkFun RedBoard [7], so our project could function on renewable energy. We purchased a solar panel system and soldered it together, and we had success generating a voltage. However, the SparkFun RedBoard is energy-intensive and consistently requires 12 volts. To integrate solar panels into our product, we would need a battery to store the energy generated by the solar panels to power the SparkFun RedBoard to produce a consistent voltage. With more time and funding, we could have implemented this into the project design. Additionally, with a larger application of our project, a solar panel system could power multiple SparkFun RedBoard and solenoid valve assemblies.

VII. Total Cost

For each system, our product costs approximately $40. If we were to mass-produce our system, we could dramatically reduce costs. By mass-producing our system, our product would be cost-accessible for consumers, urban gardeners, and agriculturalists.

Conclusion

Considering the constraints of our project, we fulfilled our goals in our prototype. The limitations of time, cost, size, our prior knowledge of plant biology, and our experience using different technological components of our project prevented us from developing and achieving some possible additions to the product.

Many future improvements could be made to our product to appeal to a larger audience. The aesthetics could be improved by concealing the wires that connect to the solenoid valve. Additionally, the Arduino Board we utilized does not possess the ability to have a low-power mode to decrease energy use. Consequently, our device is energy-intensive, requiring a sustained energy input of 12 volts. By developing a system that uses an alternative to the Arduino Board, we could measure the water moisture in the soil daily instead of constant measurements, thus requiring substantially less energy. By using less energy, a solar power system could easily be implemented. When we tried using solar panels to power the Arduino Board, achieving a consistent value of 12 volts was difficult, preventing us from incorporating it into our product

If we developed our project to a larger scale to a plot of plants, we could construct a linked network of systems, including solar panels and tanks. Having this feature would improve the aesthetics of our design. Lastly, since nutrients are often added to plants in agricultural and garden settings, we could develop a similar system to distribute varying levels of nutrients to different plants based upon plant-specific programming.

Our device has numerous practical uses that appeal to a large audience. On an individual and urban gardening scale, our product gives users the convenience of a self-watering plant that always ensures appropriate watering to promote plant health and prevent over-watering.

References

[1] *Irrigation System Pros and Cons: Mu extension*. University of Missouri Extension. (n.d.). Retrieved December 9, 2021, from https://extension.missouri.edu/programs/irrigation/irrigation-system-pros-and-cons.

[2] *Water and agriculture*. OECD. (n.d.). Retrieved December 9, 2021, from https://www.oecd.org/agriculture/topics/water-and-agriculture/

[3] *Urban farming facts*. Southwest Institute of Healing Arts | Accredited Healing Arts Programs. (2014, September 30). Retrieved December 9, 2021, from https://swiha.edu/beanurbanfarmer/urban-farming-facts/.

[4] *Gypsum block*. SoilSensor.com. (2018, September 24). Retrieved December 7, 2021, from https://soilsensor.com/articles/gypsum-block/.

[5] *Best DIY cheap soil moisture sensor*. The Cheap Vegetable Gardener. (2013, December 26). Retrieved December 9, 2021, from http://www.cheapvegetablegardener.com/how-to-make-cheap-soil-moisture-sensor-2/.

[6] *Ohm's Law.* Primary Connections: Linking science with literacy. Retrieved October 13, 2021 fromhttps://www.primaryconnections.org.au/themes/custom/connections/assets/SBR/data/Phy/sub/ohm/ohm.htm.

[7] *What is soil moisture? the science behind the measurement*. METER. (2021, June 17). Retrieved December 7, 2021, from https://www.metergroup.com/crops/articles/what-is-soil-moisture-science-behind-the measurement/#:~:text=Soil%20water%20content%3A%20It's%20just%20an%20amount&text=It's%20calculated%20by%20weighing%20the,by%20the%20dry%20soil%20mass.

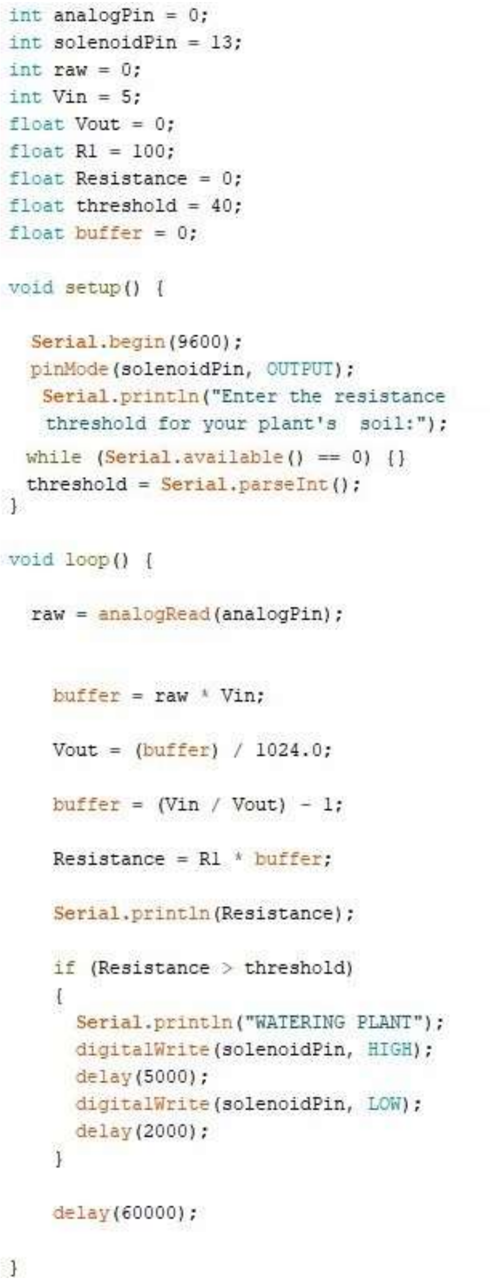
[8] *National Standard Taper Pipe Threads Size NPT Chart*. Engineers Edge. Retrieved November 5, 2021, from https://www.engineersedge.com/hardware/taper-pipe-threads.htm.

[9] BCR, C. @. (2021, December 2). *Controlling a solenoid valve with Arduino*. BC Robotics. Retrieved December 7, 2021, from https://bc-robotics.com/tutorials/controlling-a-solenoid-valve-with-arduino/

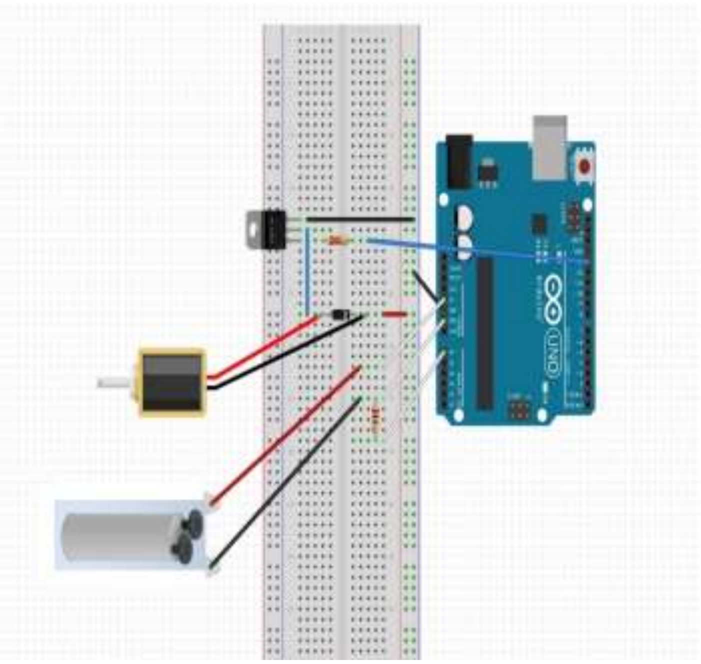
[10] Pete. (n.d.). *Series and Parallel Circuits*. Retrieved December 7, 2021, from https://learn.sparkfun.com/tutorials/series-and-parallel-circuits/all.

Appendices

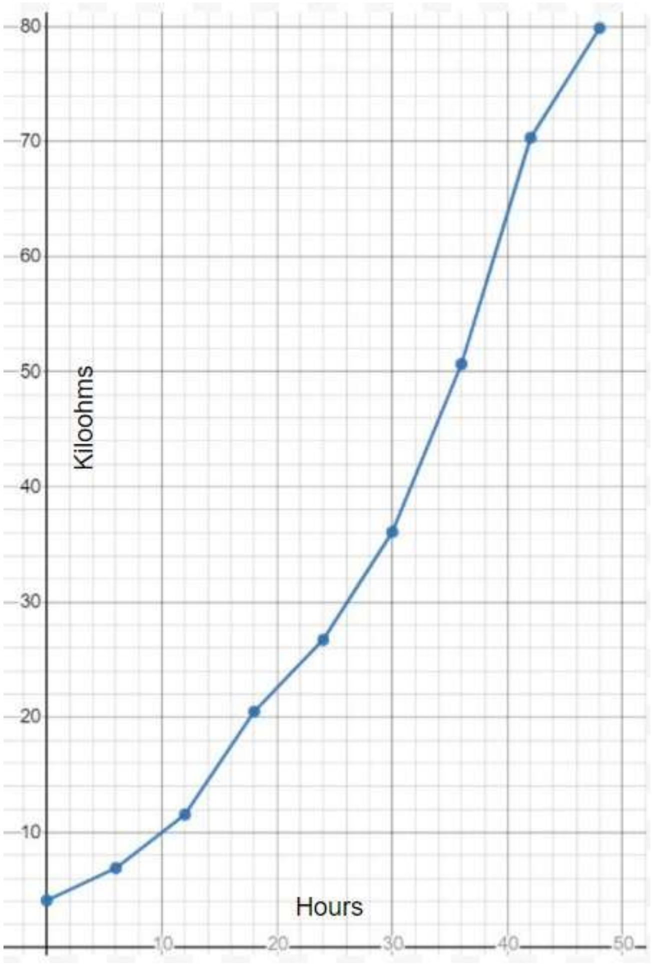
Appendix 1: Circuit Diagram and Code for data collection



```
int analogPin = 0;
int solenoidPin = 13;
int raw = 0;
int Vin = 5;
float Vout = 0;
float R1 = 100;
float Resistance = 0;
float threshold = 40;
float buffer = 0;

void setup() {

  Serial.begin(9600);
  pinMode(solenoidPin, OUTPUT);
   Serial.println("Enter the resistance
   threshold for your plant's  soil:");
  while (Serial.available() == 0) {}
  threshold = Serial.parseInt();
}

void loop() {

  raw = analogRead(analogPin);


    buffer = raw * Vin;

    Vout = (buffer) / 1024.0;

    buffer = (Vin / Vout) - 1;

    Resistance = R1 * buffer;

    Serial.println(Resistance);

    if (Resistance > threshold)
    {
      Serial.println("WATERING PLANT");
      digitalWrite(solenoidPin, HIGH);
      delay(5000);
      digitalWrite(solenoidPin, LOW);
      delay(2000);
    }

    delay(60000);

}
```



Appendix 2: Resistance in kilo-Ohms of the gypsum block over time in hours



Hours	Kiloohms
0	4.08
6	6.78
12	11.81
18	20.97
24	26.06
30	36.23
36	50.87
42	70.56
48	79.86