Full Throttle STEM™ Rover:
Big Challenge Problem

Challenge Problem and Resources

Developed by:

The teachers, students, and mentors in the
Gaming Research Integration for Learning Laboratory™ (GRILL™)
Summer 2014

TABLE OF CONTENTS

1. CHALLENGE PROBLEM "BIG" CHALLENGE OPTION

Design a rover equipped with sensors to collect data from its environment autonomously. As the rover explores a target area it should wirelessly send the data back to an Android device, which will display the data over a Google Map on a user's Android device or import data into a gaming engine's simulation of the target area.

1.1. THE TOOLS

The hardware required would be a rover, an Android device, two Arduino boards with radio transmitter/receiver shields, a GPS shield and ping sensors. An SD shield might also be necessary. Additional tools include an Arduino IDE and Android development software such as Eclipse with the ADT plugin, or Android Studio and a gaming engine, such as Unity.

1.2. THE CHALLENGE

Utilize robotics to construct a land rover with the capability to autonomously navigate through a field and collect relevant data using sensors for users to analyze.

2. TUTORIALS

Wright Scholars, in collaboration with educators and the GRILL™ team, created the tutorials described below as possible solutions to solve the challenge problem. At the time of creation these were working tutorials; however, with software updates and changes in technology, additional steps may be required. Teachers are encouraged to communicate any issues, problems, or suggested changes to these tutorials to ensure the dissemination of helpful materials to support challenge problem implementation.

2.1. MOISTURE SENSOR

A soil moisture sensor can read the amount of moisture present in the soil surrounding it. The particular sensor used in this tutorial utilizes two probes to pass current through the soil, and then it reads that resistance to get the moisture level. More water makes the soil conduct electricity more easily, while dry soil conducts electricity poorly.

This tutorial provides code that will enable the sensor to work and details how you can transfer data received from the sensor to the Android tablet. In addition, it describes the process of designing and printing pieces to enable the sensor to be dropped in the ground.

Suggested tools and materials for this challenge problem include the following:

- Arduino Uno
- Arduino IDE software
- DFROBOT Soil Moisture Sensor
- 8 Wires
- USB Cable
- Arduino IDE
- 5 Wires
- 3 Wire Servo Motor
- Breadboard
- Access to a 3D printer and any software required for the printer
- SketchUp
- SketchUp STL Exporter

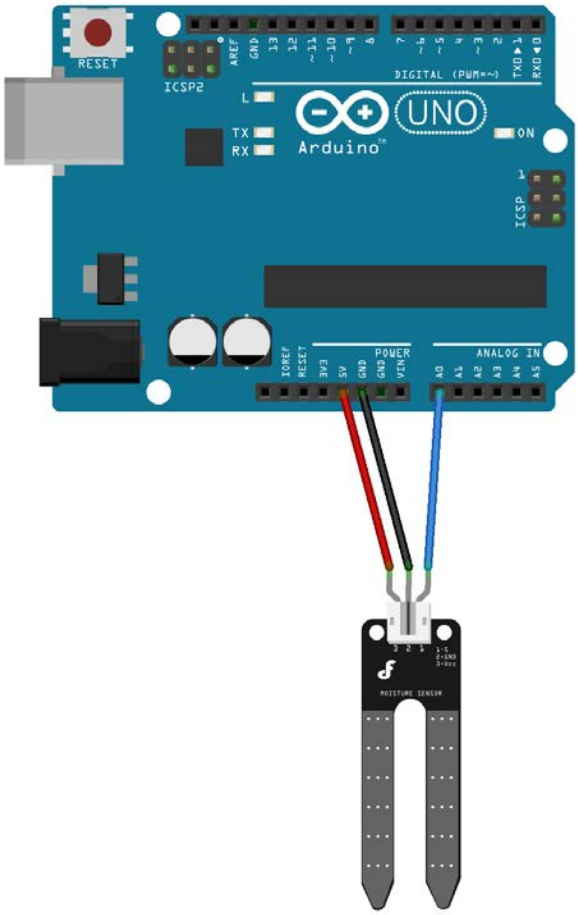
This tutorial includes the following topics:

- Arduino board and moisture sensor (Section 2.1.1)
- Modeling and 3D printing a mount for the moisture sensor (2.1.2)
- Affixing the moisture sensor and servo (2.1.3)

- Wiring and programming the moisture sensor and servo (2.1.4)
- Sample code (2.1.5)
- Additional resources (2.1.6)

2.1.1. ARDUINO BOARD AND MOISTURE SENSOR

Experiment with the sensor to see how it specifically responds to varying levels of moisture. It can be taken outside and stuck in the soil, or it can be stuck into a moist paper towel, or even water, as moisture of any kind will trigger the sensor, so long as it can complete the circuit across the two prongs of the sensor. The wiring diagram for the sensor is below.



This code will read the sensor values from the Analog 0 pin and print them to the serial monitor, where the data can be read and monitored.

```
void setup(){

 Serial.begin(9600);

}
```

```
void loop(){

  Serial.print("Moisture Sensor Value:");

  Serial.println(analogRead(0));

  delay(100);

  }
```

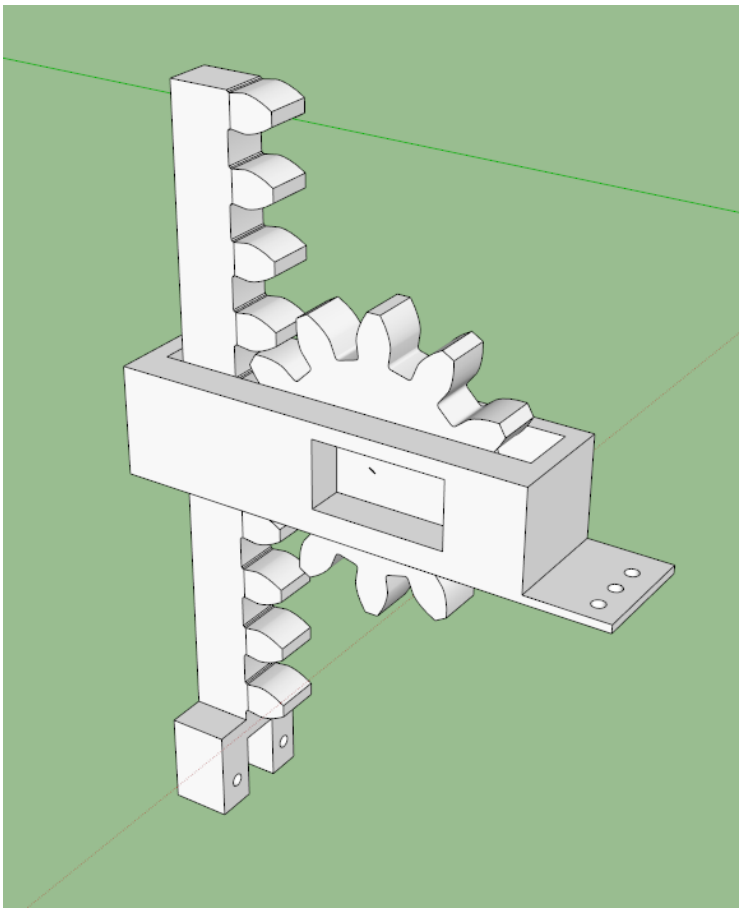
2.1.2. MODELING AND 3D PRINTING A MOUNT FOR THE MOISTURE SENSOR

Do not drag the moisture sensor through the soil; instead create a mount, using 3D modeling software and a 3D printer, to hold the moisture sensor so you can raise and lower it to take readings from various points.

The moisture sensor mount will have a rack and pinion design with a gear (pinion) that will be turned by a servo motor in order to raise and lower the rack and moisture sensor, a rack that attaches to the moisture sensor, and a holder that can hold the rack and pinion together, as well as provide a mounting place for the servo motor so that it can attach to the gear.

To create and print a moisture sensor mount:

1. In SketchUp, select File → 3D Warehouse → Get Models and choose one of the rack and pinion models in the 3D Warehouse.
2. Once you download a model into the project, modify the rack to hold the moisture sensor and adjust the size to meet the needs of the project.
3. Create a custom made holder similar to the holder in the image below.



a. Make sure the holder fits around the rack and pinion.
b. Cut a hole into the side for the servo motor to be inserted.
c. Create holes in its base so that you can screw it to the main body of the rover.
d. Modify the rack to hold the moisture sensor on the end, add screw holes that correspond with the two holes on the moisture sensor.
e. Ensure that there is adequate breathing room for all of the parts.

4. Export the holder, rack, and gear separately as .stl files and print each file following the directions for the specific 3D printer available.

Notes:

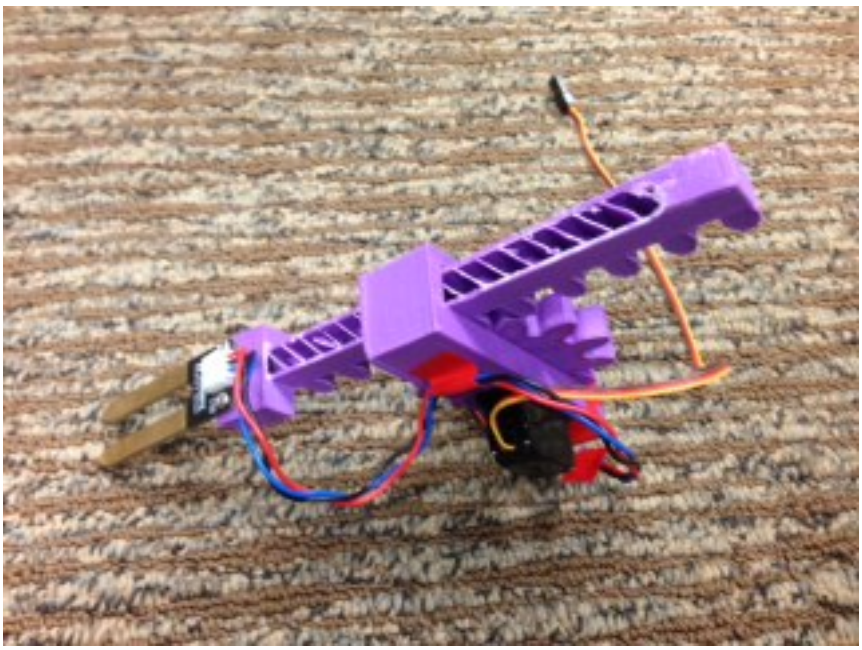
- Print test pieces beforehand to ensure that all the connections fit.
- This is not the only option for driving the moisture sensor into the soil. There are other ways that may be more appropriate depending on the variables in terrain, as well as the specific rover on which you are mounting the sensor.
- The hole for the servo motor in the preceding example is made to fit a specific servo motor. The hole should be made to fit the specific servo motor being used.

2.1.3. AFFIXING THE MOISTURE SENSOR AND SERVO

This section presents the steps required to affix the moisture sensor and servo.

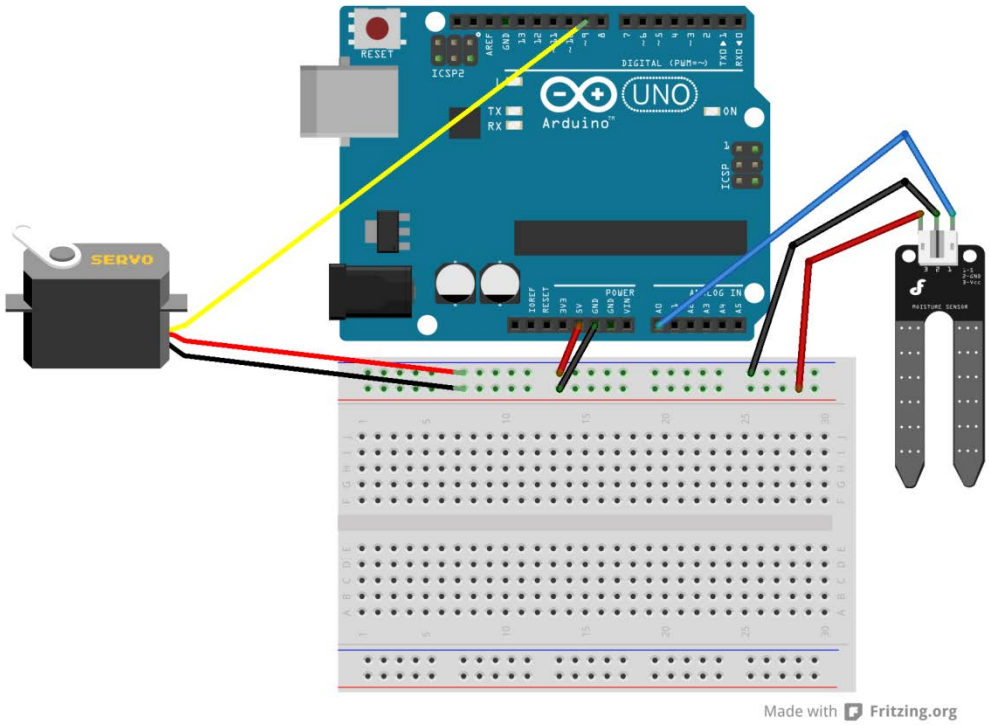
To affix the moisture sensor and servo:

1. Insert the servo motor into the hole and attach it to the gear.
2. Screw it into the gear to hold it in place.
3. Screw the moisture sensor onto the end of the rack. Fully assembled, the rack and pinion should look similar to the image below.



2.1.4. WIRING AND PROGRAMMING THE MOISTURE SENSOR AND SERVO

You must connect the moisture sensor and servo to an Arduino Uno. The wiring diagram is below.



2.1.5. PROGRAMMING THE SERVO MOTOR

This code will enable the servo motor to go down as far as is necessary for the project. The moisture sensor will then take readings of whatever it has landed in for one second, average its readings, and print that average to the serial monitor. Then the servo will bring the moisture sensor back up. This sequence will repeat until a possibly false condition is added to the code to prevent the execution of the action. When the following code is inserted into the larger code for the rover, the rover can stop and trigger the moisture sensor to descend and take its readings. Those readings are then printed to the serial monitor.

```
#include <Servo.h>

Servo myservo;  // create servo object to control a servo

                // a maximum of eight servo objects can be created

int pos = 180;  // variable to store the servo position

const int numReadings = 10;

int readings[numReadings];      // the readings from the analog input

int index = 0;                  // the index of the current reading

int total = 0;                  // the running total
```

```
int average = 0;                // the average

int inputPin = A0; //input pin for the moisture sensor

int timer = 0; //timer to bring the servo back around without stopping the moisture sensor

void setup()

{

  myservo.attach(9);  // attaches the servo on pin 9 to the servo object

  Serial.begin(9600);

  myservo.write(180);

  for (int thisReading = 0; thisReading < numReadings; thisReading++)

    readings[thisReading] = 0;

}

void loop()

{

  if(1 == 1) //this can be changed to suit whatever conditions are set for the rover stopping

  {

    delay(3000);

    pos = 0; //this should be adjusted based on how high the moisture sensor is off the ground

    myservo.write(pos);

  }

  while(pos == 0)

  {

    total= total - readings[index];      //check to make sure this works

    // read from the sensor:

    readings[index] = analogRead(inputPin);

    // add the reading to the total:

    total= total + readings[index];

    // advance to the next position in the array:

    index = index + 1;

    // if we're at the end of the array...

    if (index >= numReadings)

    // ...wrap around to the beginning:
```

```
    {
      index = 0;
    }
    // calculate the average:
    average = total / numReadings;
    // send it to the computer as ASCII digits
     delay(10);
    timer += 10;
     if(timer == 1000) //resets the timer after one second and prints to the serial the average moisture
of the soil
    {
      delay(1000);
      Serial.println(average);
      timer = 0;
      for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 180 degrees
      {
        myservo.write(pos);              // tell servo to go to position in variable 'pos'
        delay(15);                       // waits 15ms for the servo to reach the position
      }               //once it reaches 180, it will break the loop
    }
  }
}
```

Mount the rack and pinion onto the rover and plug it into Arduino.

2.1.6. RESOURCES

This page contains everything necessary to fully understand the servo motor and how it functions.

http://arduino.cc/en/reference/servo

This link contains most of the information needed to adequately use the moisture sensor. A wiring diagram is here as well.

http://www.dfrobot.com/wiki/index.php/Moisture_Sensor_(SKU:SEN0114)