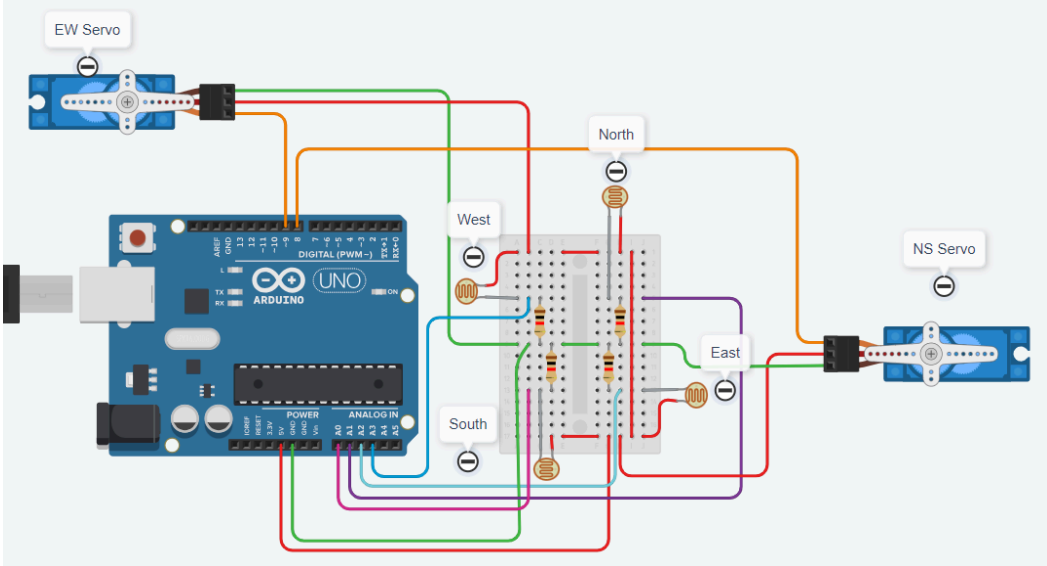
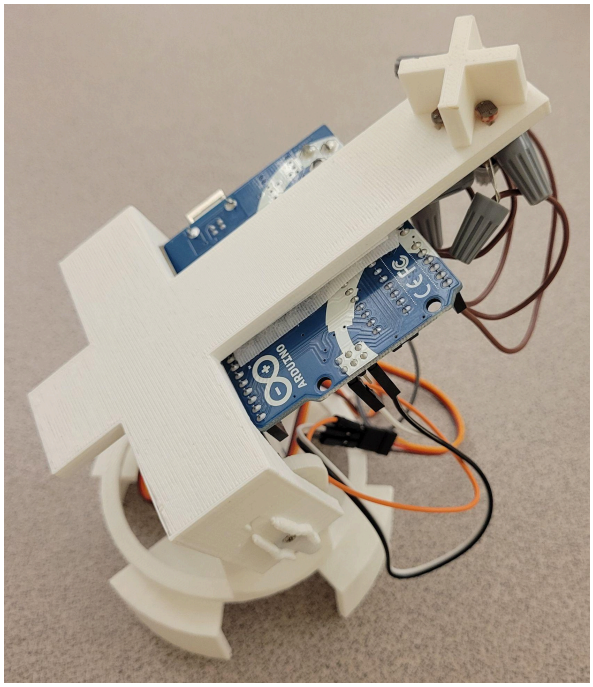
Sun Tracker Construction Manual

August 2023



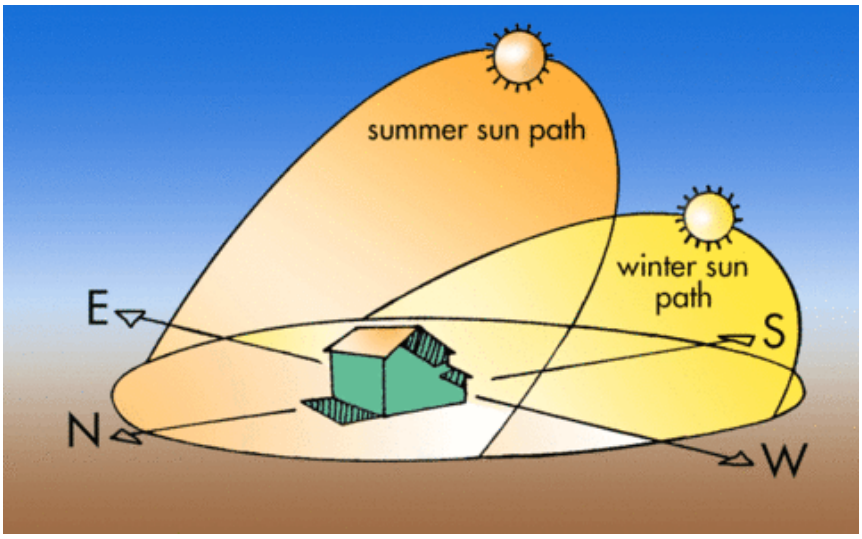
A Physical Computing STEAM Engine designed by Steve Cox and Esther Lescht for High School teachers and students.

Contents:



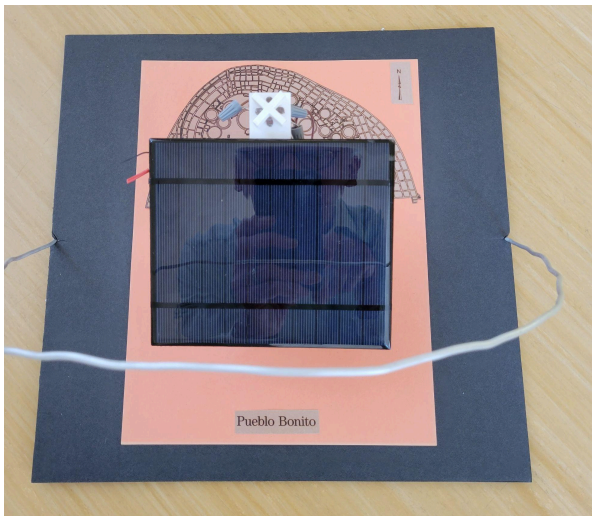
1. The Sun's Path

As we travel around the sun, the sun itself appears to travel in an arc from east to west across the sky. In the winter it stays low, to the south, and its indirect rays produce cooler weather than summer, when the sun rises toward the north and the rays are more direct. As with the east-west motion, the sun is relatively fixed, the indirect vs. direct rays stem rather from the tilt of the earth's axis. Watch this for a deeper understanding.



The **goal of this project** is to tilt a solar panel so that it always receives direct rays. We accomplish this by coupling light sensors to motors via an Arduino Uno Microcomputer on a custom built 3D-printed frame.

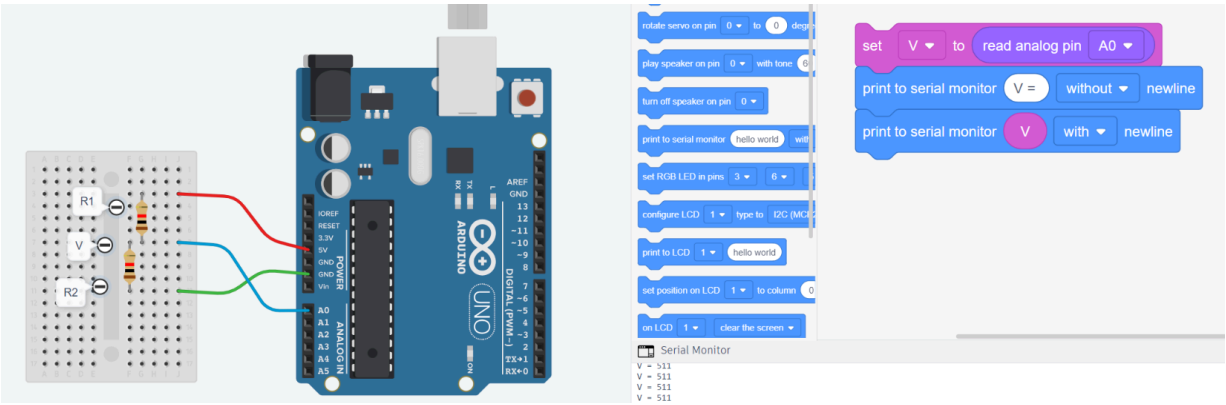
A dual axis tracker can capture more than 30% more energy than the fixed system. Even a single axis (East-West) system can increase output by 20%. Here is a lovely animation of a hydraulic single axis tracker, and here is a slick animation of a dual-axis tracker.



We will place our physical sun tracker on a custom stage, with a summer or winter path. My stage features Pueblo Bonito at Chaco Canyon. Yours might celebrate StoneHenge, the sun pyramid at teotihuacan or feature a mayan or aztec mask or sun deity. Here is a great tutorial on mask drawing.

2. The Voltage Divider

Go to Tinkercad and create a new circuit design. Add a **mini** breadboard (under Components ALL) and an Arduino Board. Wire 2 resistors in the Voltage Divider arrangement as seen to the left, and then the code blocks as shown.



From there, click the **Start Simulation** tab, click the **Serial Monitor** tab (at the bottom of your code window) and you should see values being printed.

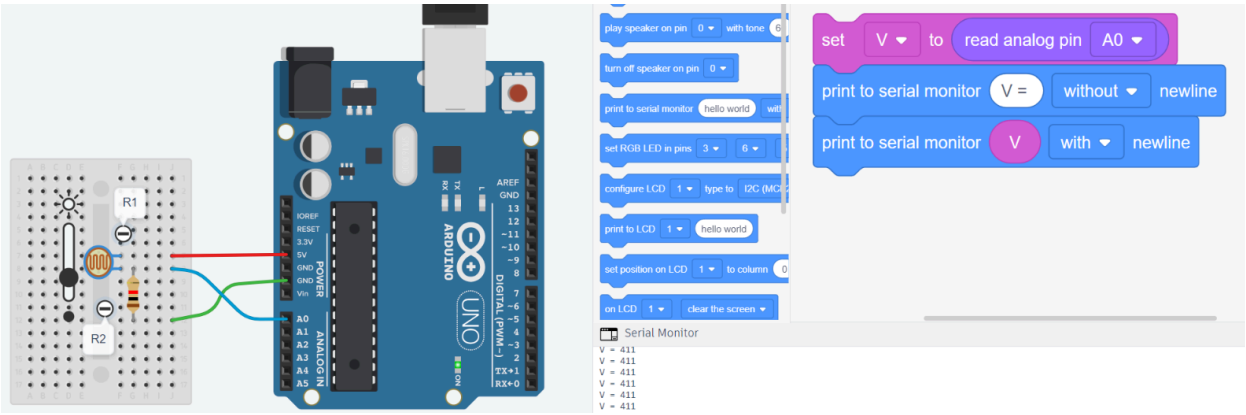
This circuit is called a voltage divider because the voltage we read at the junction of R1 and R2 is a fraction of the voltage we apply to R1. The 5 volts we apply there is represented numerically by 1023 (which is 2^{10}-1 because the Arduino uses a 10 bit analog-to-digital converter). Hence, we interpret the 511 reported in our Serial Monitor to 1023/2, or in the voltage world 5/2.

Click on a resistor and change its value. How do the values of the resistors change the values printed in the serial monitor?

We are now done with our first circuit. The output is pretty boring because our resistances are themselves boring. In the next section we will replace R1 with a light dependent resistor and then watch our values change as we move from dark to light.

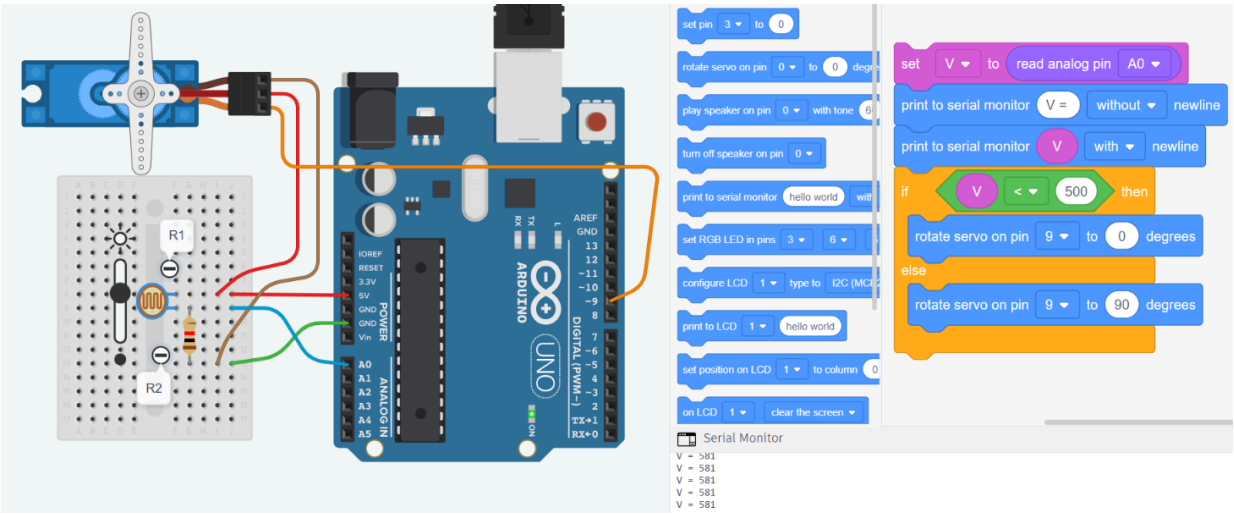
3. Light Dependent Resistors

The light that falls on a light dependent resistor (LDR) excites atoms in the exposed semiconductor which in turn make the resistor more conductive (i.e., less resistive). To quantify this we replace R1 with an LDR and record the measured junction voltage.



After clicking Start Simulation please click on the LDR. It will give you a slider that allows you to go from dark (bottom) to light (top). Please confirm that the associated junction voltage increases as you increase the light. We now have a **light sensor**!

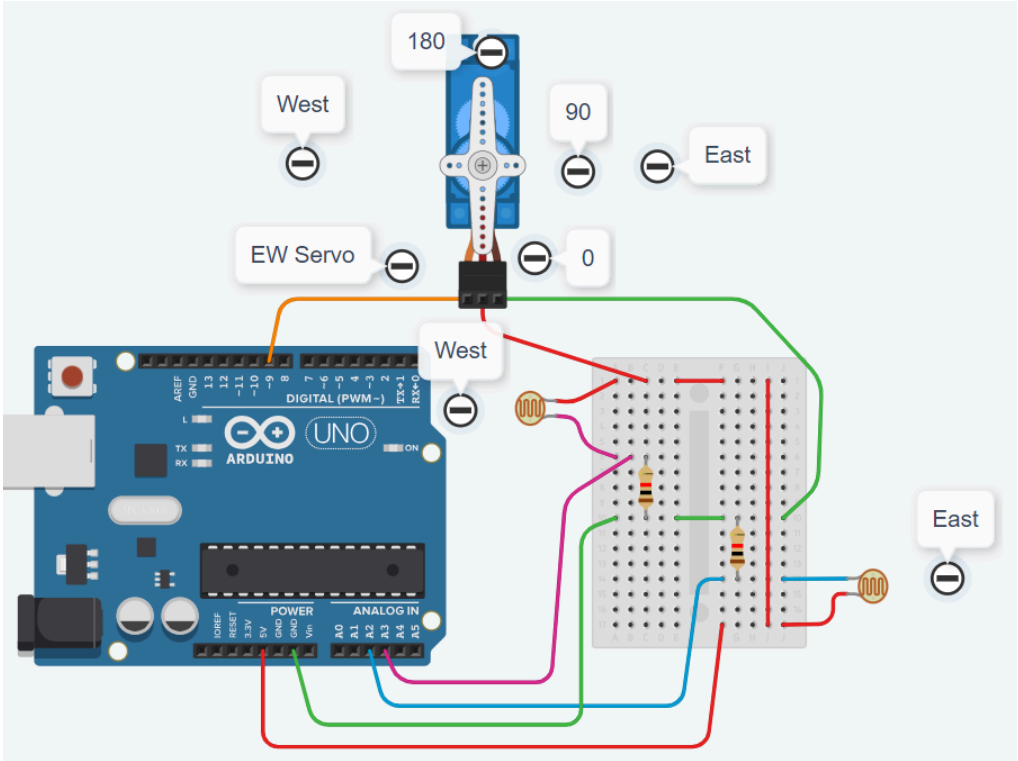
Let us see if we can use this light value to control a servo motor. Add the servo and code as pictured below.



Now confirm that the servo “raises” in the light and “lowers” in the dark.

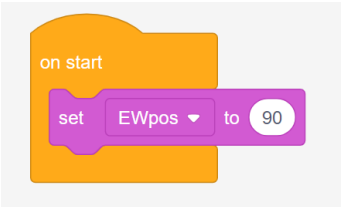
4. Single Axis Sun Tracker (Virtual)

Now that we can program one LDR and one servo, we will be tracking the sun as it rises and sets from east to west. Please build the circuit below in Tinkercad.



Now, to code this we create 3 variables

- **E** will be measure light in the East
- **W** will measure light in the West
- **EWpos** will denote the position if the EW servo



We introduce a new on start block to initialize EWpos to 90

With that done, we successively

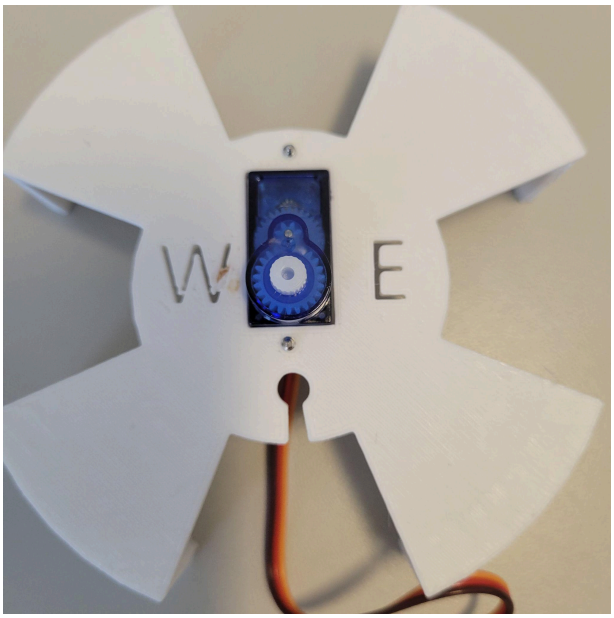
- set E to the value at A2 (read analog pin)
- set W to the value at A3 (read analog pin)
- if E is darker than W (if E < W - 10)
    - subtract 5 from EWpos (change)
- if W is darker than E (if W < E - 10)
    - add 5 to EWpos (change)
- rotate servo to EWpos
- wait a second or so

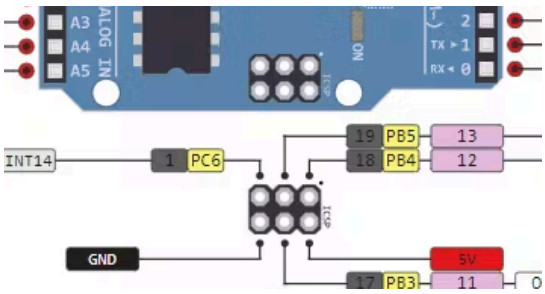
The ideas are on the left, and code hints are on the right. Can you build the code?Your tinkercad simulation should behave as in this [demo video](#).

5. Single Axis Sun Tracker (Physical)

In moving from the virtual to the physical world we will need to be very careful to properly orient our servo.

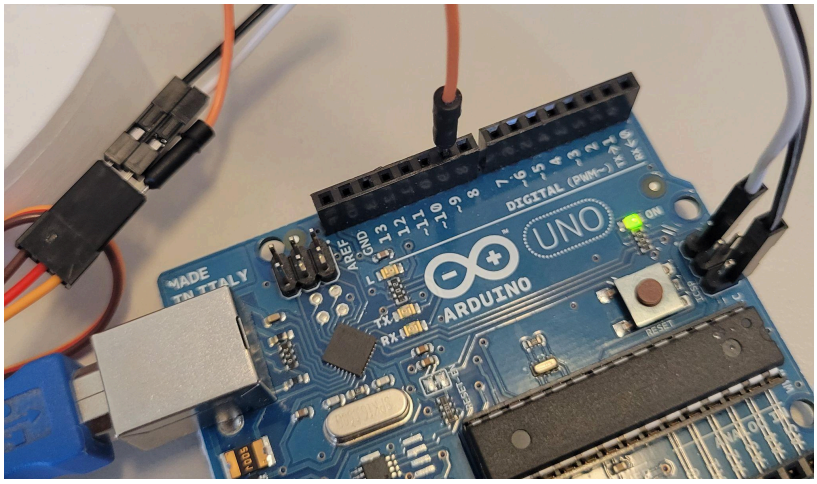
Step 1. Insert your servo into the base as shown in the photo below left. Screw your servo to the base (from below) using the two sharp screws.





Step 2. We will prepare to do without a breadboard by taking advantage of some hidden 5V and GND pins. In particular, we will use the bottom 2 outside pins on the block of 6 between A5 and RX as is the picture above right. Wire your servo to your Arduino following the BRO protocol:

- Brown to GND, Male-to-Female
- Red to 5V, Male-to-Female
- Orange to 9, Male-to-Male



Step 3. We now position the servo before screwing on the horn by **uploading** the program below. To upload you must

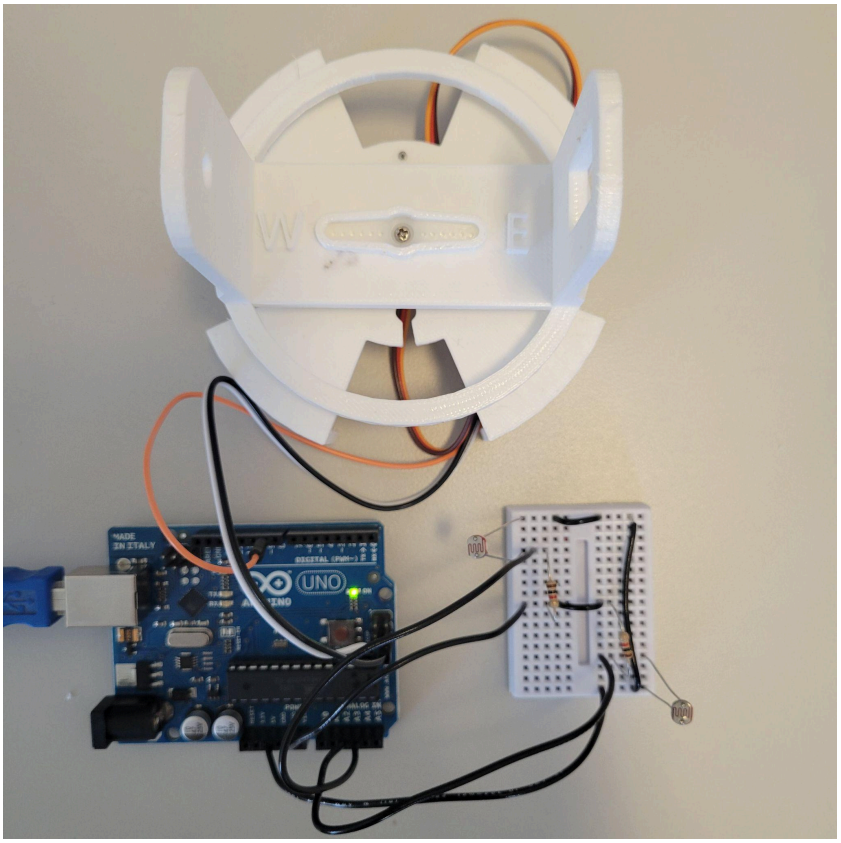
1. connect your Arduino to your desktop via the usb cable
2. open the Arduino IDE on your desktop (select Not Now if it offers you an updated version)
3. paste the code below over the program that pops up
4. In the Select Board Menu choose the COM port that says Arduino
5. Click the upload arrow

```
#include <Servo.h>
Servo servo9;

void setup()
{
  servo9.attach(9);   // this will automatically rotate to 90
  delay(100);         // give it a chance
}

void loop()
{}
```

We note that this will automatically set the servo angle to 90.

Step 4. We now attach the EW pivot (with horn) to your EW servo with the **very small screw**.

Step 5. We next wire up our 2-LDR circuit, following our tinkercad template, as in the photo at right.



Step 6. If we simply download our 1-axis tinkercad code we find that it performs poorly. The culprit is identified once we examine the W and E voltages reported at the Serial Monitor. For even when under the same lighting conditions, E can differ from W by more than 100. **Why is this?**

To account for this difference, we include a calibration step in our setup routine. It computes an average E and an average W over say 20 readings and then computes their difference

dEW = E - W

and then adds this to each W during the actual tracking, in loop. Here is the revised preamble and setup() pieces of our 1-axis tracker. Start a new program in your Arduino IDE called suntracker1 with the code below

```
#include <Servo.h>
Servo servo_9;

int E, W, dEW;
int EWpos = 90;

void setup()
{
  pinMode(A2, INPUT);
  pinMode(A3, INPUT);
  servo_9.attach(9, 500, 2500);
  Serial.begin(9600);

  for (int i = 0; i < 20; i++){  // calibration step
    E  = E + analogRead(A2);
    delay(10);
    W = W + analogRead(A3);
    delay(10);
  }

  E = E / 20;
  W = W / 20;
  dEW = E - W;

} // close setup
```

Can you see the averaging and differencing? With this done, we also require one more modification in loop().

In particular, to be safe, we should restrict the range of motion, say to plus or minus 60 degrees from our reference of 90.

To implement this, before decreasing EWpos we ask whether EWpos > 30. Similarly, before increasing EWpos we ask whether EWpos < 150.

With these changes our new loop becomes:

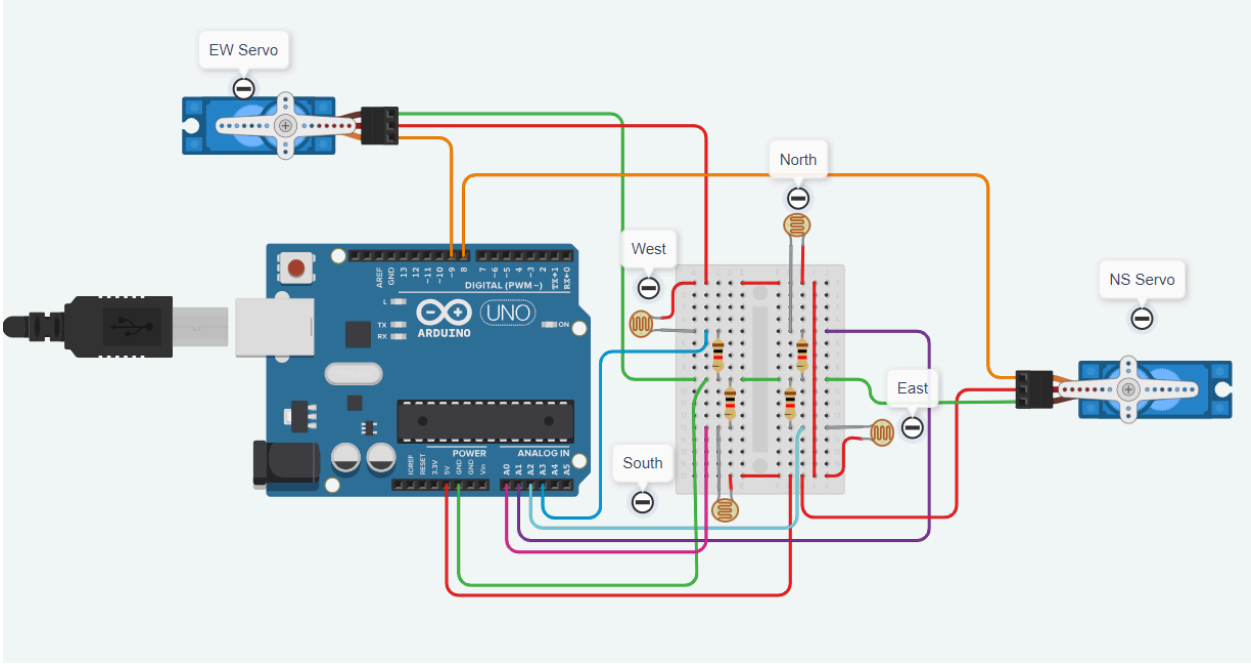
```
void loop()
{
  E = analogRead(A2);
  delay(10);
  W = dEW + analogRead(A3);  // correct for nonidentical LDRs
  delay(10);
  servo_9.write(EWpos);
  delay(100);
  Serial.print(W);
  Serial.print("   ");
  Serial.print(E);
  Serial.print("  ");
  Serial.println(EWpos);

  if (E < W - 10 && EWpos > 30) {  // if E darker than W tilt toward W
    EWpos = EWpos - 5;
  }

  if (W < E - 10  && EWpos < 150) {  // if W darker than E tilt toward E
    EWpos = EWpos + 5;
  }

  delay(2000); // Wait for 2 seconds
}
```

Add this to your suntracker1 code and upload and test.

Your physical sun tracker should now function like the one in this video demo.

6. Dual Axis Sun Tracker (Virtual)

We now have a working sun tracker but it only tracks the sun from east to west. We will now add a north-south servo so the sun tracker will follow the sun as the seasons change. To do this we need to add another servo and 2 more LDRs - as in the figure below.



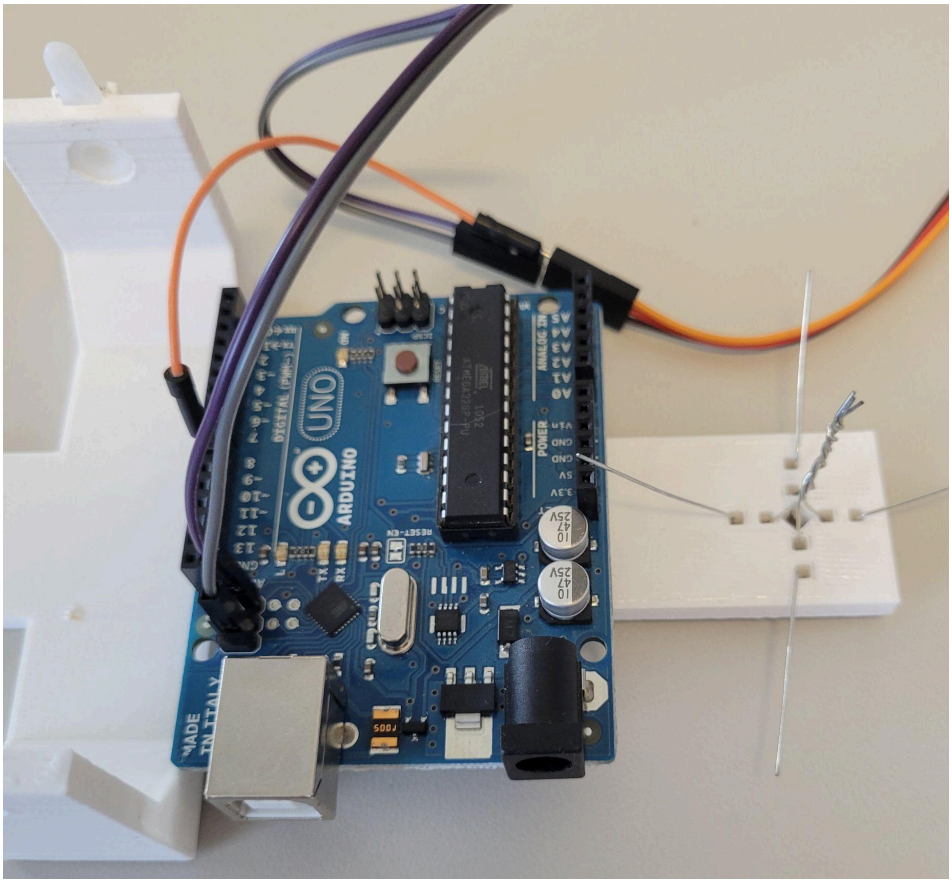
Please add the new N, S, NSpos variables and associated code blocks following our original logic

With that done, we successively

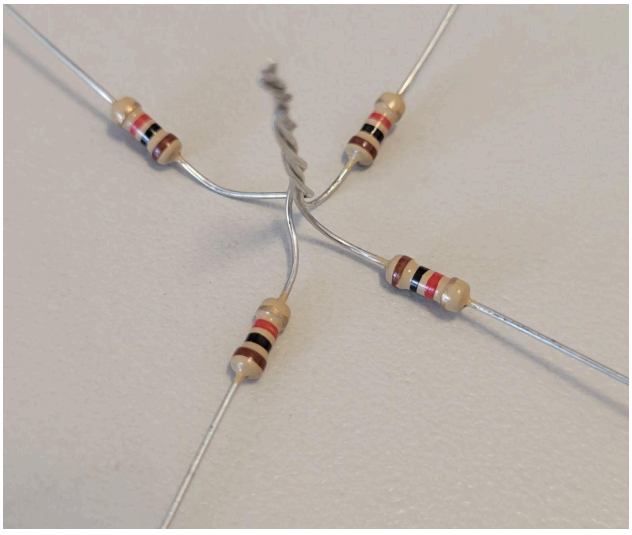
- set S to the value at A0 (read analog pin)
- set N to the value at A1 (read analog pin)
- if S is darker than N (if S < N - 10)
    - subtract 5 from NSpos (change)
- if N is darker than S (if N < S - 10)
    - add 5 to NSpos (change)

7. Dual Axis Sun Tracker (Physical)

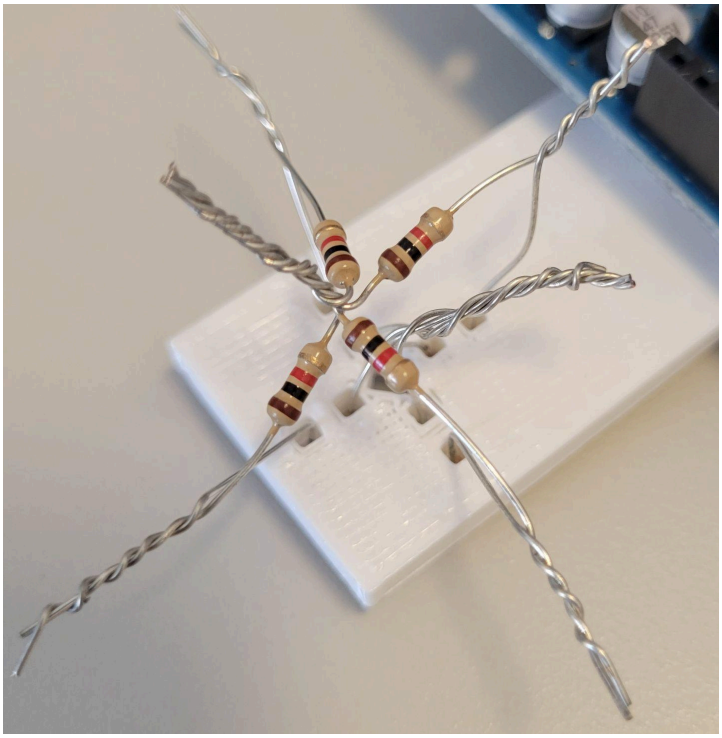
Step 1. Velcro your Uno Board to the NS Pivot as shown below. Thread the 4 LDRS through the NS pivot piece, winding one leg from each into a common strand, as shown below.



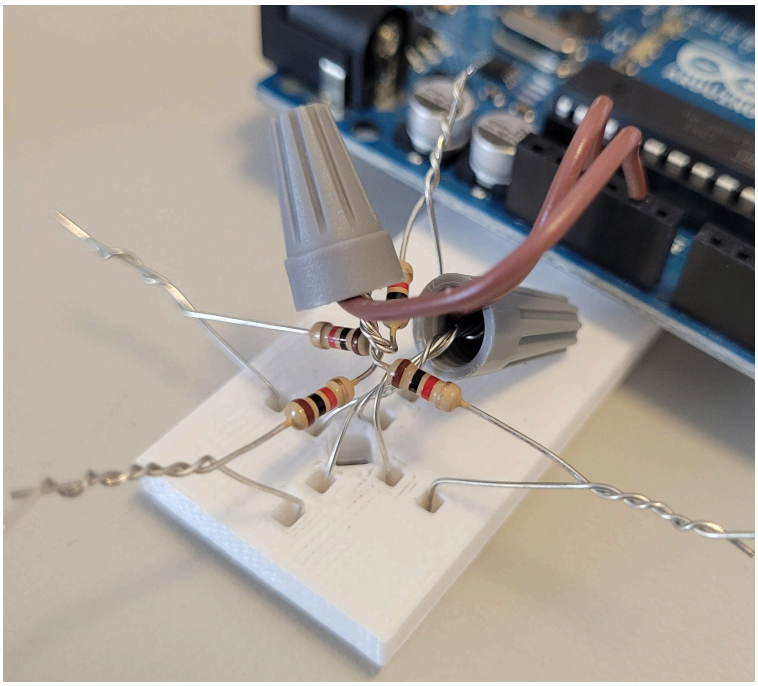
Step 2. Make a similar circuit with 4 1k resistors, as shown below.



Step 3. Next wind the 4 free LDR ends to the 4 free resistor ends, while keeping the fatter bundles away from one another.



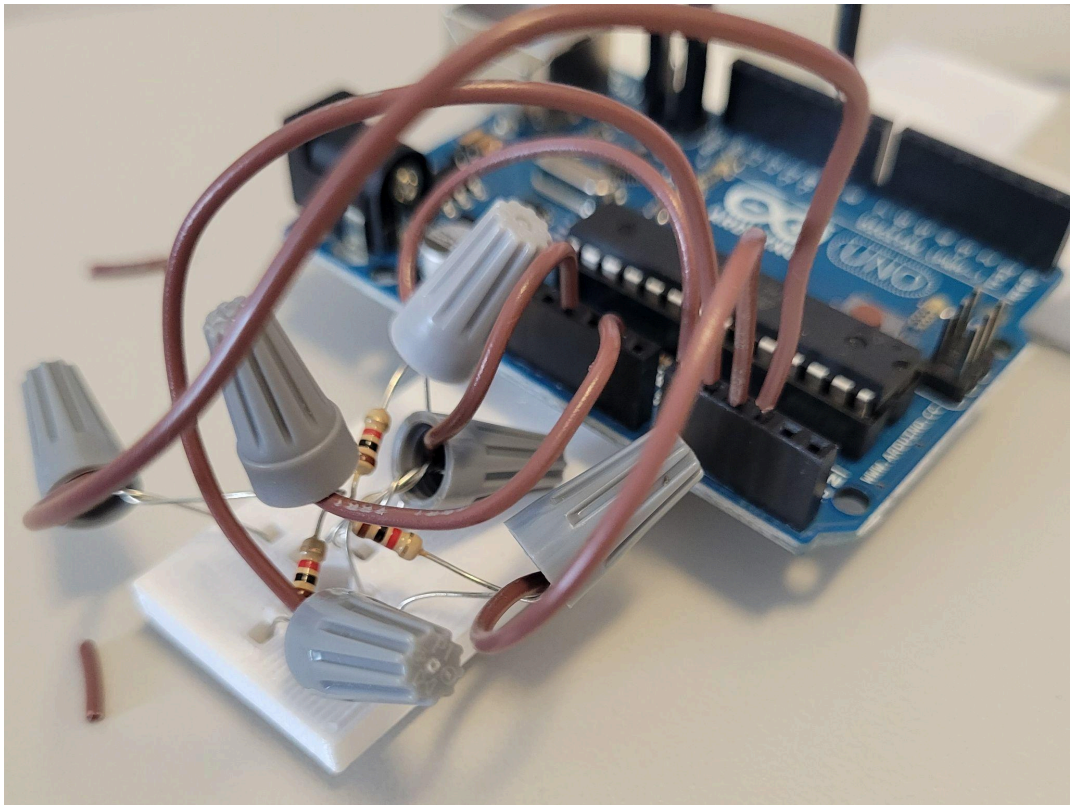
Step 4. Cut and strip a short jumper wire, wind it round the common LDR strand, and secure with a wire nut. Insert the other end of the jumper into the 5V socket of your Uno. Do the same thing for your resistor network, and connect its free end to a GND socket of your Uno, as shown below.



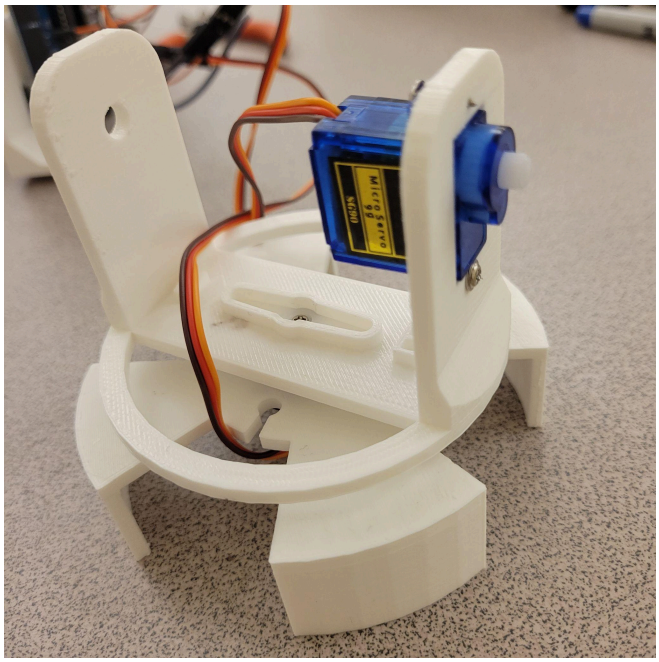
Step 5. Now cut and strip 4 short jumper wires and wind and nut the end of each jumper to an LDR-Resistor junction. And then insert the free end of each jumper into the proper Uno socket

S to A0, N to A1, E to A2, W to A3

being especially careful with E and W as our workpiece is upside down!

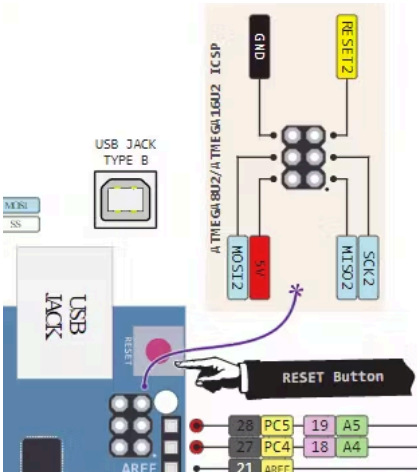


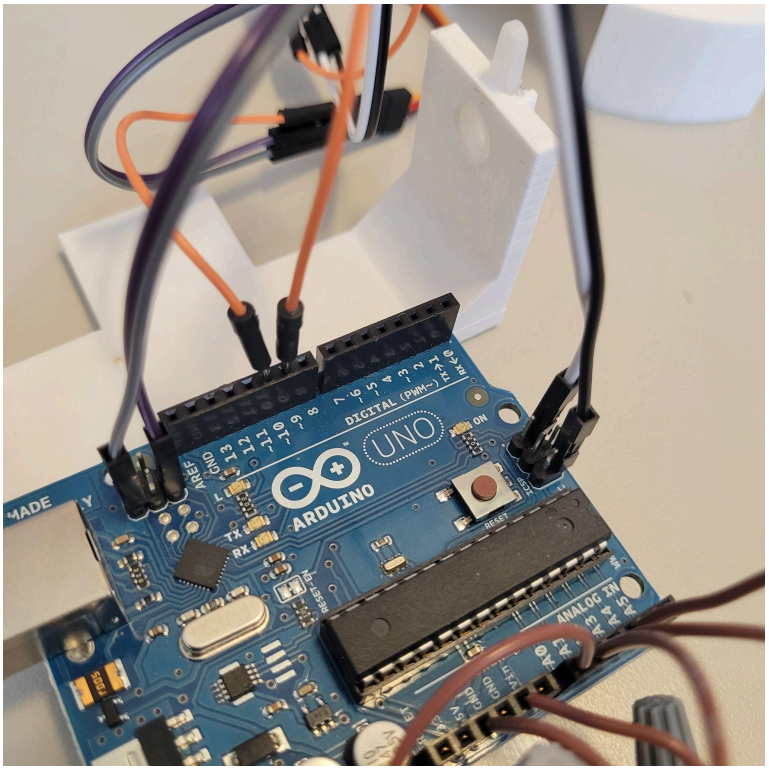
Step 6. Push your NS servo through the EW pivot and secure it with sharp screws. This is a tight space, so I screwed one from each side.



Step 7. We will take advantage of two more hidden 5V and GND pins. In particular, we will use the left 2 outside pins on the block of 6 near the reset button as in this picture. Wire your servo to your Arduino following the BRO protocol:

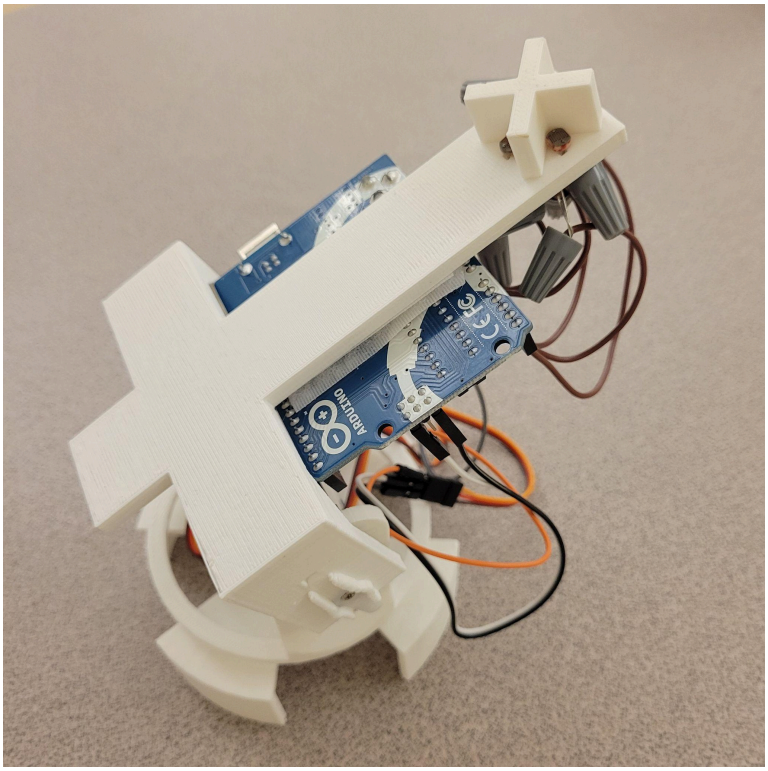
- Brown to GND, Male-to-Female
- Red to 5V, Male-to-Female
- Orange to 10, Male-to-Male

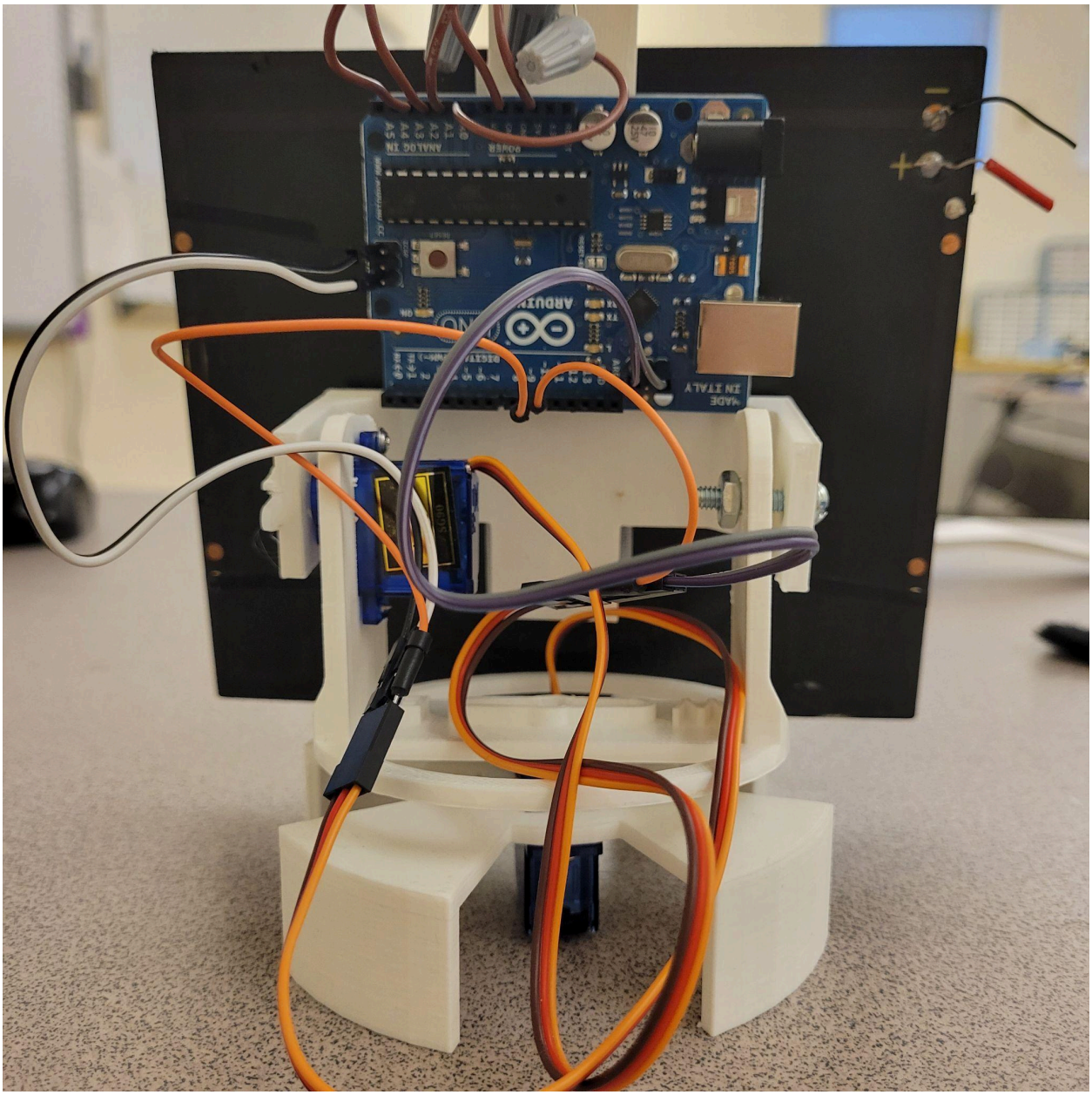




Step 8. Run the servo setup code to align your NS servo to 90. Press the horn onto the NS pivot and attach the horn to the servo at roughly the 45 degree angle shown. Also insert the shadow cross.

Opposite the servo please insert the bolt and nut as shown in the photo below - also now with solar panel attached via velcro.





Step 9. Modify your 2-axis virtual tracker code to include calibration of the N and S LDRs.

It should perform as in this 2-axis Sun Tracker Demo.

Here is the final preamble and setup procedure.

```
#include <Servo.h>
Servo servo_9;
Servo servo_10;

int E, W, dEW, N, S, dNS;
int EWpos = 90, NSpos = 90;

void setup()
{
  pinMode(A2, INPUT);   // East reading
  pinMode(A3, INPUT);   // West Reading
  servo_9.attach(9, 500, 2500); // EWservo

  pinMode(A0, INPUT);   // South reading
  pinMode(A1, INPUT);   // North Reading
  servo_10.attach(10, 500, 2500);  // NSservo

  Serial.begin(9600);

  for (int i = 0; i < 20; i++){  // calibration averages
    E  = E + analogRead(A2);
    delay(10);
    W = W + analogRead(A3);
    delay(10);
    N  = N + analogRead(A1);
    delay(10);
    S = S + analogRead(A0);
    delay(10);
  }

  E = E / 20;     // calibration shifts
  W = W / 20;
  dEW = E - W;

  N = N / 20;
  S = S / 20;
  dNS = N - S;

} // close setup
```

And here is the final loop procedure

```
void loop()
{
  E = analogRead(A2);
  delay(10);
  W = dEW + analogRead(A3);
  delay(10);
  servo_9.write(EWpos);
  delay(100);

  N = analogRead(A1);
  delay(10);
  S = dNS + analogRead(A0);
  delay(10);
  servo_10.write(NSpos);
  delay(100);

  if (E < W - 10 && EWpos > 30) {  // if E darker than W tilt toward W
    EWpos = EWpos - 5;
  }

  if (W < E - 10  && EWpos < 150) {  // if W darker than E tilt toward E
    EWpos = EWpos + 5;
  }

  if (N < S - 10  &&  NSpos < 130) {  // if N darker than S tilt toward S
    NSpos = NSpos + 5;
  }

  if (S < N - 10  &&  NSpos > 60) {  // if S darker than N tilt toward N
    NSpos = NSpos - 5;
  }

  delay(2000); // Wait for 2 seconds
}
```

Appendix 1: Parts List

- Electronics
    - 4 1K resistors
    - 4 LDRs (photoresistors)
    - Mini Breadboard
    - Arduino Uno
    - Solar Panel
    - 2 Servos
    - Wire
    - 4 female-to-male jumper wires
- Other
    - 1-inch Bolt & Nut
    - 6 Wire Nuts
    - Velcro
- Tools
    - Wire cutter/stripper
    - Small screwdriver
- 3D printed parts
    - Base
    - EW pivot
    - NS pivot
    - Cross
- Platform
    - 14”-by-14” piece of foam board or cardboard
    - Construction paper, box-cutter, glue-stick
    - 9 gauge aluminum craft wire for sun path