

Adaptable Solar Panel Accessory for Increased Productivity

Olivia Xu
olivia.r.xu@gmail.com

ABSTRACT

In this paper, I discussed the development of a new device designed to identify the direction of the highest light intensity. This device would help to optimize solar energy production, assisting solar panels with facing the direction that would result in the most sunlight absorbed. My design was successful on a small scale, accurately finishing each iteration of the program by pointing to the light source. Compared to solar panels without my device, I was able to generate more energy due to the increase in direct sunlight hitting the panels. While the device reached an overall success, future work can improve on reducing its size to optimize space.

INTRODUCTION

Over the course of a decade, the amount of solar panels in the United States has grown fivefold. As of 2025, over five million solar panels have been installed throughout America [1]. Installers of standard solar panels usually determine the ideal position of the panels based on geographic and seasonal characteristics. Most solar panels have fixed positions, while others are adjusted yearly, in order to increase energy output [2,3]. Still, these orientations do not optimize energy collection. Instead, developing panels that adapt to solar intensity throughout the day are considerably more efficient, taking into account the fluctuations in sun visibility.

Currently, sun-tracking solar panels exist, but are limited in number due to the cost and qualities like motorized parts and installation issues that make it only beneficial to some homeowners. These panels work best in flat, open locations, due to the moving gears and axis in the mechanism and homeowner's association (HOA) restrictions [4]. The solar panels would also require sunlight at all times throughout the day; otherwise the trade-off between money and energy would not be cost-efficient.

In this paper, I proposed developing a device mimicking a solar panel to find the position with the most intense sunlight to increase energy input. To create a robot that would adapt to varying levels of sun intensity throughout the day, I started by exploring the contents of the Arduino kit. The photocell was a promising component, with its ability to change qualities depending on light intensity [5]. The contraption

Aug 2026
Vol 10. No 4.

would also need to be energy efficient, so the extra sunlight collected from adjusting to a more optimal position would not be canceled out by the energy needed to make the shift.

MATERIALS AND METHODS

Materials

The ELEGOO UNO R3 Super Starter Kit was purchased from the ELEGOO website. The kit is compatible with the Arduino UNO R3, MEGA 2560 R3, and NANO. It included: 1pcs UNO R3 Controller Board; 1pcs LCD1602 Module (with pin header); 1pcs Breadboard Expansion Board; 1pcs Power Supply Module; 1pcs Joystick Module; 1pcs IR Receiver; 1pcs Servo Motor (SG90); 1pcs Stepper Motor; 1pcs ULN2003 Stepper Motor Driver Board; 1pcs Ultrasonic Sensor; 1pcs DHT11 Temperature and Humidity Module; 1pcs 9V Battery with DC; 1pcs 65 Jumper Wire; 1pcs USB Cable; 1pcs Active Buzzer; 1pcs Passive Buzzer; 1pcs Potentiometer; 1pcs 5V Relay; 1pcs Breadboard; 1pcs Remote; 1pcs Tilt Switch; 5pcs Button (small); 1pcs 1 digit 7-segment Display; 1pcs 4 digit 7-segment Display; 5pcs Yellow LED; 5pcs Blue LED; 5pcs Green LED; 5pcs Red LED; 1pcs RGB LED; 2pcs Photoresistor; 1pcs Thermistor; 2pcs Diode Rectifier (1N4007); 2pcs NPN Transistor (PN2222); 1pcs IC 74HC595; 120pcs Resistor; and 10pcs Female-to-male Dupont Wire.

Out of those, I used: 1pcs UNO R3 Controller Board; 1pcs 9V Battery; 1pcs USB Cable; 1pcs Photocell; 2pcs Servo Motors; 1pcs 10 k Ω Resistor; 3pcs Male-to-male Dupont Wires; 8pcs Female-to-male Dupont Wires; 1pcs Breadboard; 1pcs Power Supply Module; and 1pcs Uno R3 Controller Board. I also used a voltage and current measuring device, as well as two machine screws and two nuts.

Methods

To check all directions that the solar panel may face, I used two servo motors that rotate 180 degrees each. The servo motors were each placed into 3D printed cases; the cases for the bottom servo motors were then sealed with two pairs of nuts and bolts. Next, they were positioned into the configuration shown below, with one motor determining the y-position and the other determining the x-position. The 3D printed cases ensure that the two motors are able to stay in place, and it provides a connection between the two servo motor bodies. On the topmost servo motor is the photocell, and the resistance the photocell provides at each position will be kept track of through a variable in our program. The overall setup is all controlled through the Arduino IDE, with a program that I developed controlling the rotation of the two motors together.

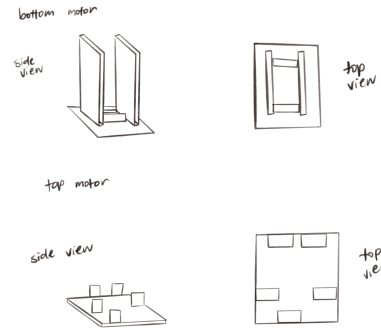


Figure 1: The design for the 3D printed motor cases

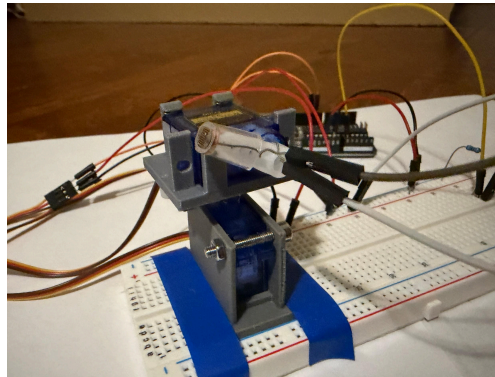


Figure 2: Side view of the motors and cases assembled

The resistance of a photocell varies with the amount of light it receives, with this specific photocell providing less resistance when the measured intensity of light is higher. The recorded resistances were then used to figure out the ideal position, through looking at where the resistance was highest. The position was stored in two variables, one for the angle of each servo motor. With the ideal position recorded, the solar panels were angled at that positioning until the light source changed angle.

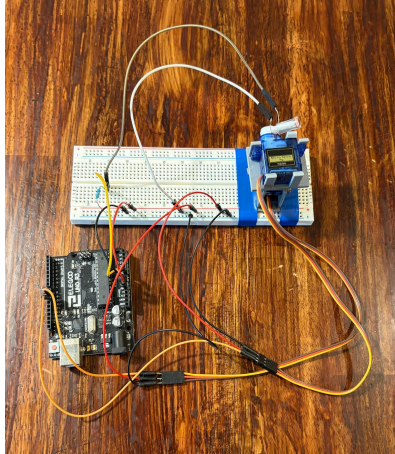


Figure 3: Top view of entire device, battery not attached

PROGRAM

```

Servo xservo;
Servo yservo;
int pin = A0;    // select the input pin for the photocell
int smallx = 0;
int smally = 0;
double smallr = 100000000;

int ledPin = 13;    // select the pin for the LED
int sensorValue = 0; // variable to store the value coming from the sensor
double temp = 0;
double resistor = 0;

int x;
int y;

```

The program begins by instantiating two Servo variables, *xservo* and *yservo*, representing the two motors on the robot connected to the program. The *xservo* is the motor on the bottom, and the *yservo* is the motor placed on top. An integer variable, *pin*, is instantiated next, representing the value the photocell is input through on the controller board. In this case, the input is at port A0. The next two are integers *smallx* and *smally*, both used to store the locations where the sunlight was strongest. Following is double *smallr*, which is used to store the smallest recorded value of resistance in the photocell, the resistance when the motors are at *smallx* and *smally*. In order to ensure that the first value it measures will be captured, it is initially set to 100000000. Similar to *pin*, the integer *ledPin* stores the port number that the LED is connected to. The following integer, *sensorValue*, stores the output of the photocell. Then, the double *temp* is used to help with calculating resistance of the photocell later, and the double *resistor* is used to store the resistance at a certain position. Finally, integers *x* and *y* are used to store the values for the angle of displacement, one for each motor.

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(ledPin, OUTPUT);  
  Serial.begin(9600);  
  xservo.attach(9);  
  yservo.attach(10);  
}
```

The first line in the setup section of the code establishes that the item connected to port number *ledPin* is set to an output source and a connection is created between the Arduino controller board and the device the program came from. The Servos are then attached to the numbers nine and ten, corresponding to the ports they are plugged into on the board.

```
void loop() {  
  smallr = 100000000;  
  
  for(x=smallx; x>=0; x-=5) {  
    xservo.write(x);  
    delay(20);  
  }  
  for(y=smally; y>=0; y-=5) {  
    yservo.write(y);  
    delay(20);  
  }  
  
  xservo.write(0);  
  yservo.write(0);  
  delay(1500);  
}
```

At the beginning of the overall code loop, *smallr* is once again set to 100000000, to act as a buffer before real values of resistance are input in each iteration. The *x* and *y* return to 0, in intervals of 5 degrees and delayed by 20 milliseconds, to ensure that the motors start at the same position every time.

```

for (x = 0; x <= 180; x += 5) { // goes from 180 degrees to 0 degrees
  xservo.write(x);           // tell servo to go to position in variable 'pos'
  for (y = 0; y <= 180; y += 5) {
    yservo.write(y);
    delay(80);               // waits 15ms for the servo to reach the position
    // put your main code here, to run repeatedly:
    // read the value from the sensor:
    sensorValue = analogRead(pin);
    //sensorValuey = analogRead(ypin);
    temp = sensorValue/1023.0*5.0;
    //Serial.print(temp);
    //Serial.print(" ");
    resistor = (5.0-temp)/(temp/1000);
    //Serial.println(resistor);

    if(resistor<smallr)
    {
      smallr=resistor;
      smallx=x;
      smally=y;
    }
  }
}

```

Then, a *for loop* is established, shifting *x*, the one controlling the bottom motor, five degrees on each iteration. Inside this loop, there is a second *for loop*, focusing on *y*, also increasing it by 5 degrees each time. The loop increases until *y* reaches 180. In every round of the *y* loop, the motor is sent the value *y* for its position, the program waits 80 milliseconds, and the recorded photocell value is stored in *sensorValue*. This *sensorValue* variable is then used when creating the value for *temp*. It is divided by 1023.0, which is the maximum value the photocell can output, and multiplied by 5.0, which is the resistance of the fixed resistor also in the circuit. At this point, *temp* has a value proportional to 5.0 the way the *sensorValue* is proportional to 1023.0. This is then used to calculate the resistance of the photocell, through the equation $(5.0 - temp) / (temp / 1000)$. Once the resistance is calculated for this position, it is stored in *resistance* and the program compares it to the current smallest resistance, *smallr*. If it is smaller than the current smallest resistance, the *smallr* variable will now hold the current resistance, and the *smallx* and *smally* variables will hold the current *x* and *y* values as well. With this, the *for loop* based on the *y* variable closes.

```

x+=5;
xservo.write(x);
for (y = 180; y >= 0; y -= 5) {
  yservo.write(y);
  delay(80); // waits 15ms for the servo to reach the position
  // put your main code here, to run repeatedly:
  // read the value from the sensor:
  sensorValue = analogRead(pin);
  //sensorValuey = analogRead(ypin);
  temp = sensorValue/1023.0*5.0;
  //Serial.print(temp);
  //Serial.print(" ");
  resistor = (5.0-temp)/(temp/1000);
  //Serial.println(resistor);

  if(resistor<smallr)
  {
    smallr=resistor;
    smallx=x;
    smally=y;
  }
}
}

```

The *x* variable is then increased by another addition of 5, and the motor is programmed to move to the new position. Another *for loop* based on *y* is created, but decreasing instead. This adjustment allows the top motor to travel less, so it can travel back and forth for the photocell to read values, instead of starting at the same place every time. The inside of the loop is the same as the previous one, so the photocell continues to read and replace values of *smallr*, *smallx*, and *smally* as needed.

```

Serial.print(smallr);
Serial.print(" ");
Serial.print(smallx);
Serial.print(" ");
Serial.println(smally);

delay(2000);

xservo.write(smallx);
yservo.write(smally);
delay(30000);
}

```

Once all positions have been measured for light intensity, the serial monitor prints out the three values *smallr*, *smallx*, and *smally*, separated with spaces for visibility. The printed lines test the program and ensure that all the parts are recording accurate measurements by allowing the user to verify that the values line up with the intensity shone. Lastly, *xservo* and *yservo* are sent the values *smallx* and *smally*, to instruct them to move to the ideal position for maximum light. The program then pauses for 30,000 milliseconds. The parameters can be adjusted to become longer or shorter, based on how often the user wishes to adjust their solar panels.

RESULTS

The provided breadboard configuration and program accurately pointed toward the position with the highest light intensity. I tested it by shining a flashlight on the photocell at certain positions in its movement and by covering the photocell at other positions. The machine was considered a success when it ended with the photocell pointing at the location where intensity was at a peak. This output demonstrated that the machine recognized the locations with the highest light intensity, and the code to run it contained no errors either.

Trial No.	Lowest Recorded Resistance (ohms)	Location (x, y)
1	318.30	130, 15
2	833.33	70, 10
3	323.42	136, 95
4	998.05	5, 45

Figure 4: Table of data collected from successful trials of the prototype

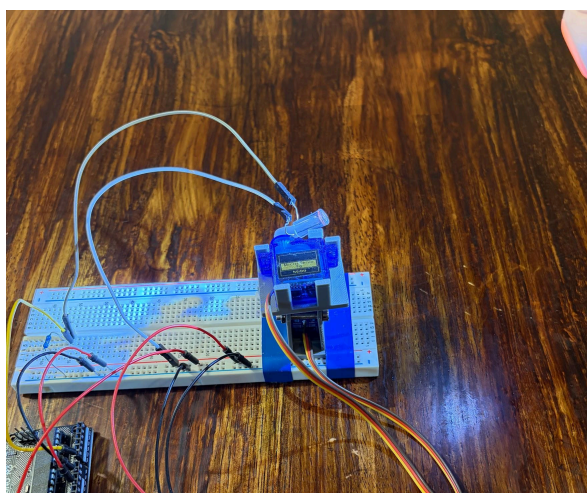


Figure 5: Photocell pointing to location with highest light intensity. Flashlight (not pictured) in top right corner.

Compared to the sun, flashlights are much weaker and less effective at charging solar panels. In outdoor conditions, the prototype would adapt to the changing positions of the clouds and how much of the sun they block. The distribution of sunlight is scattered and fluctuates due to different weather and cloud

conditions. When it is cloudier, the prototype will have to run the program more, in order to keep up with frequent cloud movement.

The overall system, including the motors and photocell, is estimated to use 588 milliwatts. This value was calculated through measuring the energy used. In 5 minutes, the program looped approximately seven times and the prototype used a total of 49 milliwatt hours (mWh), which is equal to 588 mWh in an hour. Milliwatt hours per hour simplifies to milliwatts, so it would result in 588 milliwatts used.



Figure 6: Energy measuring device showing energy consumption after 5 minutes

Users can determine their optimal timing through information on the sunrise and sunset times of their location, as well as the hours of peak sunlight in their location. Some key times to adjust tilt could be during sunrise and when peak sun hours begin. During nighttime, users would be able to set the interval to become much longer, as there is no solar energy to be absorbed. Certain conditions with changing sunlight, like cloudy or rainy times, would likely need the program to run more. This would allow the prototype to match the activity of the clouds. However, checking for the optimal tilt too frequently would require more energy, due to the system needing energy to run. Using the energy consumption determined from earlier, continuous adjustment for an hour could require over 588 milliwatts.

I learned that the machine was most effective when it remained in one location, due to shadows moving around. The code worked best with two separate loops controlling each motor, although it was not the most memory- and time-efficient. Some efficiency in time was maintained through the top motor starting from opposite ends each time, instead of returning to the same spot. Overall, the system was quite time-efficient, but there are still areas that could be improved, such as its limited reach and the amount of space occupied by wires.

APPLICATION

The adaptable photocell-motor attachment could be used for future solar panel designs through many different formats. Chiefly, the main application for the accessory is to have it act as a model for the solar panel to follow. In order to ensure the positioning captures the most sunlight, the program could run every

few hours. Once the photocell has found the angle with the most sunlight, the solar panel next to it, with mechanical axles to adjust it, will match the photocell's positioning. With a more optimal positioning, solar panels would be able to produce near 400 watts per hour, due to increased access to direct sunlight [6]. The value from optimal positioning is estimated through the production results of solar panels under ideal conditions. The production of average unadjusted solar panels is predicted through the equation of *watts per hour = 1000 * kilowatt-hours (kWh) / peak sunlight hours*. An average 400-watt panel generates around 1.7 kWh of energy daily, and the national average peak sunlight hours is 5 hours [7, 8].

In comparison, without the adjusted optimal angle ensured by the system proposed in this paper, typical solar panels currently available can only achieve an average of 300-350 watts per hour. Taking into account the energy used by the motors and overall prototype, the solar panel-prototype system would still be able to generate an increased 49-99 watts compared to standard non-adapting solar panels.

DISCUSSION AND FUTURE PLANS

While the idea of separate sunlight intensity sensors and movable solar panels is innovative, there are still obstacles preventing this model from becoming widely used. Solar panels are costly, which is why they were only used in about 5% of American houses in 2024 [9]. Since solar panels are already a hefty investment for owners, the system would capture more energy, thus allowing the panels to return a greater value. Widespread implementation of this idea would require even more materials, particularly for both the solar panel and the photocell-motor system. The sun-tracking panels in the market right now determine where to face based on calculating the revolutions of the Earth and the sun [4]. Compared to current sun-tracking solar panels, this system finds the location with highest intensity. This system is able to match the conditions when sunlight is different than what was predicted. In cloudy conditions, it is able to turn towards where the sun is peeking through, unlike the kind presently being sold.

The main issue relating to this model is the limitation of where the system is placed. Once installed, the system is inflexible, so any disruptive tree leaves or branches would affect solar panel production until they are moved out of the way. Wires taking up too much space is also a concern. To remedy this concern, future studies could enhance the device I developed in the following manner: The system could be transferred to a smaller breadboard panel to reduce the overall volume because much of the breadboard is unused. Wires would then be clustered closer together, decreasing the amount of separate protrusions.

The prototype was also only tested under indoor conditions, with a flashlight as the main light source. This produced the limitation that the system never experienced a setting where the source was as widespread as the sun. The conditions that it was testing in, however, did also have another overhead light in the room. The photocell just pointed to the flashlight, which it experienced more intensely. Because it has not been tested outdoors, the prototype's durability has not been recorded. The prototype has many open ports, so it is not waterproof, and it has not faced the other elements. Future models can implement waterproof materials, as well as an external casing around parts that are more fragile.

CONCLUSION

In summary, my proposed system allows for increased energy input and time-efficiency compared to standalone solar panels. The ability of my device to face towards the highest light intensity allows solar panels, paired with my machine, to produce 50-100 more watts per hour. This surplus in energy would be equivalent to allowing households to charge a phone multiple times over. However, the design could be more compact and the code could also be improved by enabling greater memory-efficiency. I hope to address these aspects in future work.

REFERENCES

1. *5 million solar installations: Powering American communities*. (2024, May). Solar Energy Industries Association. <https://seia.org/solar-installations/>
2. Landau, C. L. (2017, March 18). Optimum Tilt of Solar Panels. <https://www.solarpaneltilt.com/>
3. Handoyo, E. A., Ichsani, D., Prabowo, P. (2013). The Optimal Tilt of a Solar Collector. *Energy Procedia* (32nd ed., pp. 166-175).
4. Blok, A. (2026, March 17). *Sun-tracking solar panels: How they work, pros and cons*. Palmetto. <https://palmetto.com/solar/sun-tracking-solar-panels-worth-it>
5. *Photocell*. (n.d.). ScienceDirect, <https://www.sciencedirect.com/topics/engineering/photocell>
6. Zientara, B. (2025). *How much energy does a solar panel produce?* SolarReviews. <https://www.solarreviews.com/blog/how-much-electricity-does-a-solar-panel-produce>
7. *Peak sun hours by state* (n.d.). Unbound Solar, <https://www.unboundsolar.com/solar-information/sun-hours-us-map>
8. *How many watts does a solar panel produce* (2026, July 1). Jackery, <https://www.jackery.com/blogs/knowledge/how-many-watts-does-a-solar-panel-produce>
9. Agopian, A. (n.d.). *How many Americans have solar panels in 2024?* SolarInsure. <https://www.solarinsure.com/how-many-americans-have-solar-panels>