New Orleans JAZZED about Engineering Education

Paper ID #17124

MAKER: Spirograph-Style Drawing Machine Controlled by Arduino

Dr. Clark Hochgraf, Rochester Institute of Technology (CAST)

Clark Hochgraf, Ph.D., teaches engineering, design thinking and making skills to students at the Rochester Institute of Technology. He finds joy in making technology accessible and useful to more people.

As associate professor of electrical engineering technology at RIT, he teaches digital signal processing and microcontrollers using a hands-on, learn by building approach. He works with community organizations such as Rochester Roots and Rochester Makerspace to promote youth well-being and technology education by connecting college students to community members in service learning projects.

Clark is a maker who has built electric go-carts, scooters, bikes, hybrid automobiles and co-launched the first student-designed college campus solar-charged, electric bike share in the US. He holds 12 US patents ranging from Megawatt-scale power inverters to hydrogen fuel cell membranes. Before teaching, he worked 11 years in industry as an engineer for Westinghouse, Ford/Visteon, and General Motors R&D. His current research is on using smartphone technology to prevent automobile crashes.

MAKER: Spirograph[tm]-Style Drawing Machine Controlled By Arduino

Abstract

This paper presents an Arduino-controlled Spirograph[tm]-style drawing machine suitable for use at a Maker event. Visitors can use the machine to make unique artwork to take home. Instructions to build the drawing machine are provided. Potential pedagogical uses of the drawing machine range from learning hands-on construction techniques, to programming, trigonometry, and interaction with a user through sensors.

Overview

This paper provides instructions on how to build a pantograph drawing machine using an Arduino UNO microcontroller, a reflectance sensor, two rc-servo motors, and a sheet of foam core poster board. It is based on the work of Erik Brunvand, Ginger Alford, and Paul Stout [1,2] and extended by using a combination of a continuous-rotation servo and a position-controlled servo. The project can be built in one and a half hours and creates pen-on-paper drawings similar to a Spirograph[tm] that are programmatically repeatable or randomly generated patterns. It is fun to watch, and its potential pedagogical uses include teaching microcontrollers, sensors, and trigonometry. The paper is organized as follows. First, sample artwork from the machine is presented. Next the operation, materials and physical construction of the machine are described. Then programming of the machine is presented. Finally, pedagogical uses are explored.

SAMPLE ARTWORK CREATED

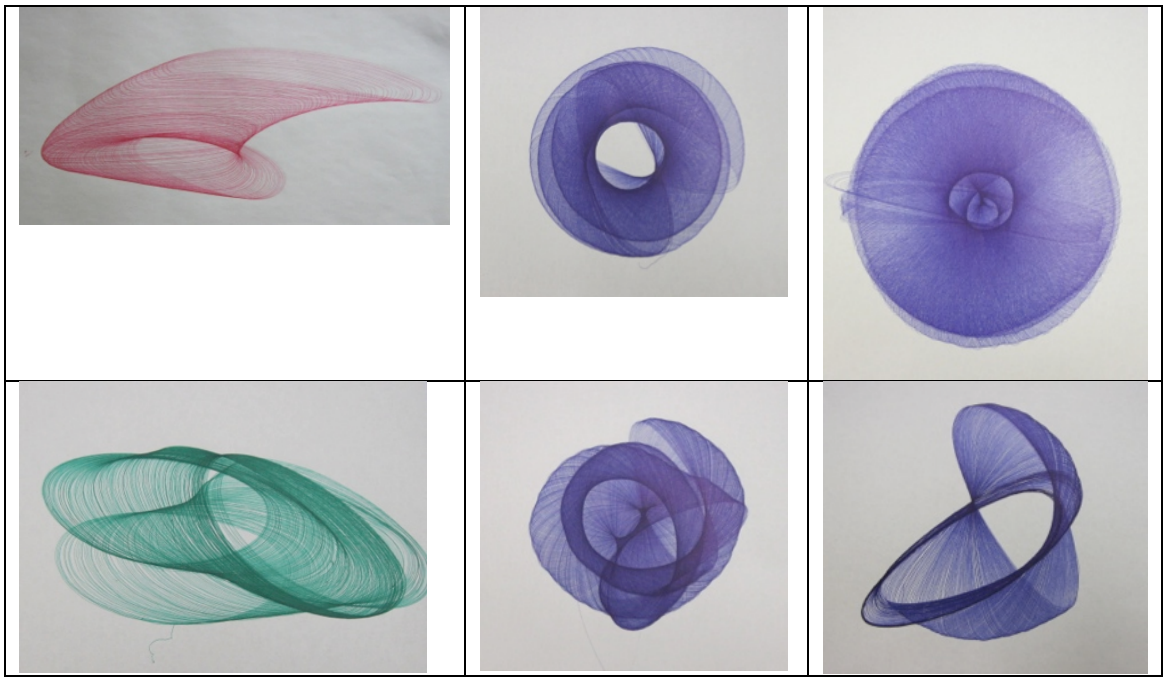


Figure 1 Sample artwork created by the drawing machine. The two images on the left were created using a fixed program with the paper drawing canvas taped down. The four images on the right were created with the paper not taped down, allowing a slow rotation of the paper canvas.

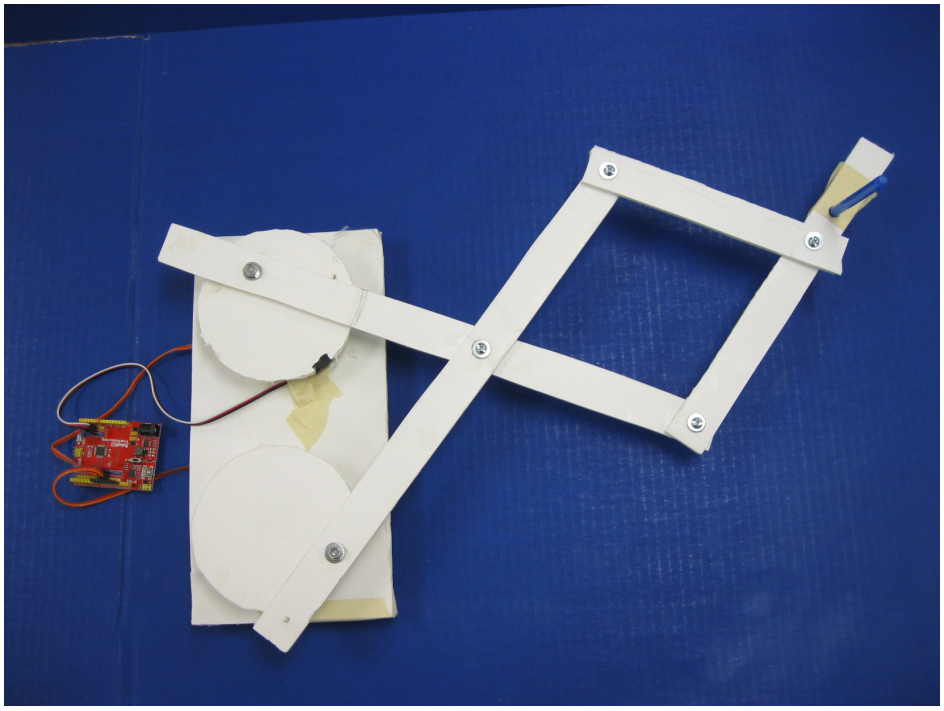


Figure 2 Completed drawing machine created from foam core board. On the right side, a ball point pen is attached to the arms of the mechanism. On the left side, two foam core board disks are attached on top of two RC servos (hidden in this view). On the hand side, an Arduino UNO compatible microcontroller board is shown; this drives the two RC servos and reads the input from a reflectance sensor (hidden under the upper disk in this view).

OPERATION, MATERIALS AND CONSTRUCTION

Operation

Figure 2 shows the completed drawing machine. It consists of a set of scissor-mechanism arms that hold a ball point pen (right side of figure 2). The other ends of the arms are attached to rotating disks (left side of figure 2). The disks are driven by two servo motors located under the disks (Figure 3a). The servo motors are controlled by an Arduino microcontroller (red board with wires on left side of figure 2) with sensor feedback that counts the rotations of the continuously rotating disk.

To illustrate the operation of the system, consider the case in which the disk at the bottom of figure 2 is not rotating. As the upper disk makes a full 360-degree rotation, the scissor mechanism traces out an ellipsoid at the location of the pen. If the lower disk is then rotated a few degrees and held in that position, then the rotation of the upper disk will create an ellipsoid in a slightly different location and with a slightly different shape. By combining these two motions, a variety of patterns can be created. A reflectance sensor detects when a full rotation of the upper disk has been completed and this information is used to make a small angular rotation in the lower disk.

A drawing might consist of 100 rotations of the upper disk and only a quarter turn of the lower disk. A sheet of plain paper (11”x17”) is placed under the pen and taped down to the table. The

assembly with the two spinning disks can taped down to the table, clamped or weighted to keep it from moving.

Materials

The bill of materials and tools required to make the machine are listed below. The materials can be purchased from any number of vendors, e.g. Yourduino.com, seeedstudio.com, solarbotics.com, adafruit.com, etc. Part numbers for the retail electronics site Yourduino.com are listed.

Bill of Materials: Electrical

Vendor	Item	Description	part number	Unit cost	Qty
Yourduino	microcontroller	YourDuinoRoboRED Arduino UNO Compatible) w/ USB cable	SKU: RP-500602	$15.00	1
Yourduino	rc servo continuous	Servo: Continuous Rotation - For Robot Wheel Motor SM-S4306R	SKU: RP-104306	$9.25	1
Yourduino	rc servo standard position	Servo: Regular Positional Servo SM-S4306B	SKU: RP-104306B	$9.25	1
Yourduino	12 vdc 2amp supply	AC Line Power Supply: 12VDC 2A	SKU: PB-010104	$4.50	1
Yourduino	jumper wire	40 pin flat cable MALE ends 20cm	SKU: CW-044031	$2.00	1
Yourduino	sensor cable	3 pin cable flat end + latched end 30cm (10Pc)	SKU: CW-040230	$1.50	1
Solarbotics	IR reflectance sensor	QTR-1A Reflectance Sensor (2-Pack)	51005	$4.25	1
			subtotal	$45.75	
Optional items					
Yourduino	push button switch	Set of 4 Colored PushButton Bricks	SKU: SS-040702	$6.75	1
Yourduino	distance sensor	Ultrasonic Distance Measuring Module HC-SR04	SKU: SE-020304	$3.50	1
Yourduino	light sensor	PhotoResistor type 5516 5-10K (10 ea)	SKU: EC-040401	$1.25	1
Yourduino	temp sensor	TMP36 Temperature sensor (LM35)	SKU: EC-010105	$1.75	1
AdaFruit	sound sensor and amplifier	Electret Microphone Amplifier - MAX9814 with Auto Gain Control	PID: 1713	$7.95	1
			subtotal	$21.20	

Bill of Materials: Mechanical

Vendor	Item	Description	Qty
Ace Hardware	#10-24 machine screw	#10-24 round head machine screw 3/4" long	6
Ace Hardware	#10-24 lock nut (nylon)	#10-24 lock nut or stop nut nylon insert	6
Ace Hardware	3/16" flat washer	3/16" flat washer (oversized for #10 screw)	18
Target	Foam board	20"x30" foam core board	2
Target	Clear tape	2" clear packaging tape	1
Target	Masking tape	2" masking tape	1
Target	Black electrical tape	3/4 " Black electrical tape	1
Target	Ball point pen	various colors, as you like	1

Tools:

1 – utility knife suitable for cutting foam core board
1 – Philips screw driver
1 – tape measure or ruler as cutting guide
1 – soldering iron and solder

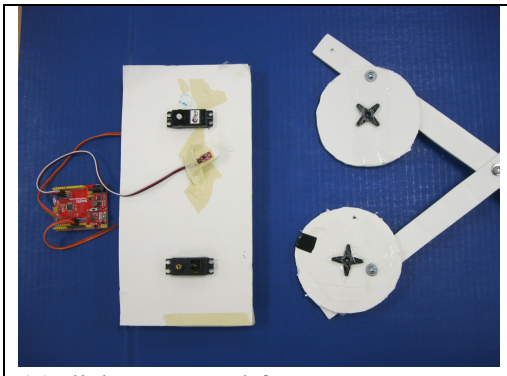
Construction

From the foam board sheet, cut out the following pieces.

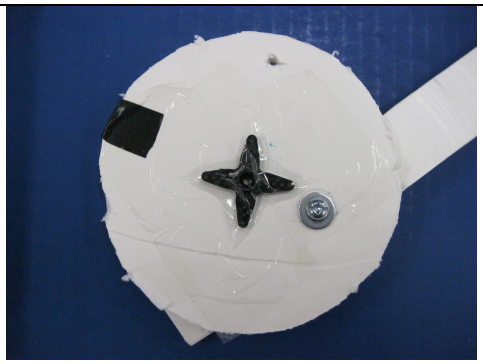
- a rectangular base board of 20 cm by 35 cm
- two disks of 12 cm diameter each. An overturned bowl or a DVD can be used as a template to trace out the disks.
- two long rectangular arms 40 cm long by 3.5 cm wide

- two short rectangular arms 24 cm long by 3.5 cm wide

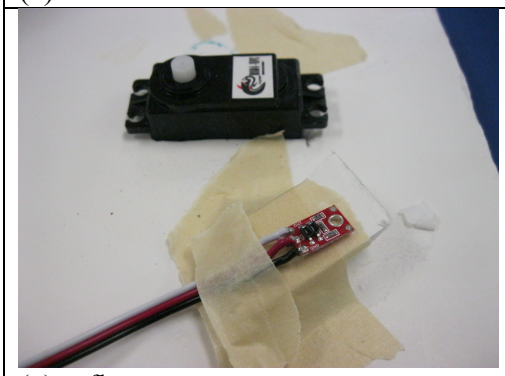
Next, take a servo motor and place it on the large rectangular base as shown in figure 3a. Space the motors about 17 cm apart. Trace around the motor and cut holes in the foam board for the servos. Make the holes for the servos a bit undersized so that the servos fit tight into the base board. You can experiment with different motor spacings and disk diameters to get the size/scale of drawing you would like. If the motors are too close together, the scissor arms may reach too far and the pen may run off the edge of the paper.



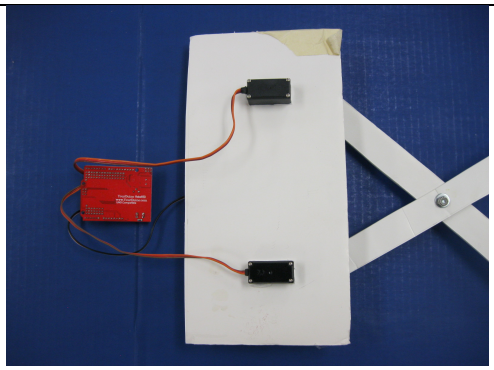
(a) disks removed from servo motors



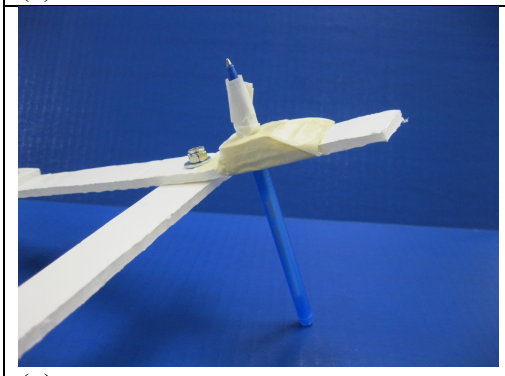
(b) detail, servo horn taped to disk



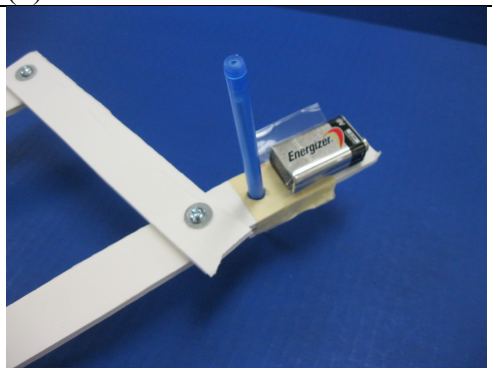
(c) reflectance sensor



(d) underside of servo mount



(e) pen mount



(f) pen weight

Figure 3 Construction details for assembling the drawing machine.

After cutting the disks, tape down the servo horn X-arms to the center of the disk using clear packing tape or good quality 2" masking tape (figure 3b). Remove the tape where the center hub of the horn attaches to the servo, otherwise it will be difficult to attach the disks to the servos.

Assemble the arms using ¾ inch long #10 machine screws and nylon lock nuts at the joints. Use a utility knife or small screwdriver to poke holes for the screws. On both the long and short arms, the distance between the screws (joints) should be 17 cm.

Use small flat washers on the joints to help the machine work more smoothly with less friction. Place a flat washer between the two pieces of foam core at the joint and under the head of the screw and at the nut, for a total of three flat washers on each joint.

Once the arms are assembled, return to the disks. Take each disk and make a hole in the disk at a radius of 5 cm from the center of the disk and attach each arm to its corresponding disk.

Next, install the pen on the arm as shown in figure 3e. Make a hole in distant end of one of the small arms and push the pen through it. You can add a small, 1 inch long piece of foam board to make the arm thicker at this point. Take the additional foam core down tightly to the arm. Near the point of the pen, wrap a narrow strip of masking tape to make the body of the pen thicker so it can't slide out of the hole in the arm. Refer to figure 3e.

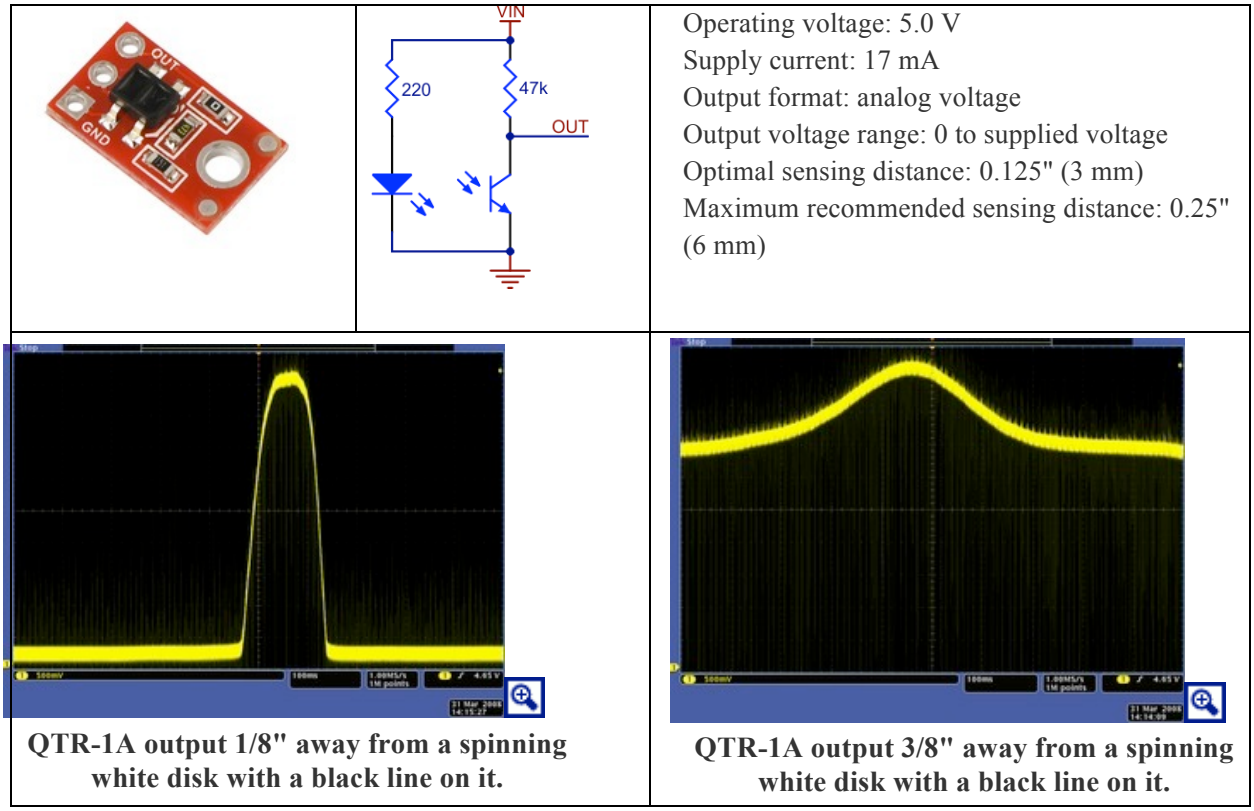


Figure 4 Schematic of the reflectance sensor and operation as a function of distance from the disk. (figures from Solarbotics.com)

Next, install the rotation sensor, which is used to move the pantograph arm position precisely one step after each ellipse is drawn. The rotation sensor consists of two pieces: a small piece of

black tape on the underside of the disk that is connected to the continuous rotation servo (figure 3b) and a reflectance sensor mounted to the base rectangle (figure 3c).

The reflectance sensor works by bouncing an infrared light off the bottom of the disk and gives a signal when the black tape (non-reflective) passes by. The generated pulse is read by the Arduino UNO to incrementally change the position of the other servo motor. Figure 4 describes the sensor and its operational performance, which shows that the sensor must be very close to the disk to produce an adequate signal. As result, the sensor is mounted on a small piece of foam board to bring it closer to the bottom of the disk (figure 3c).

The schematic of the sensor shows that it needs three connections: +5v power, ground and signal out. Solder three wires to the sensor, using the specified sensor cable or equivalent. Match the red wire to +5, black wire to ground and white wire to signal out.

Plug the sensor into analog input A2 on the Arduino RoboRed board and plug the continuous rotation servo into digital pin d9 and the position servo into digital pin d10. If you are using the RoboRed board, all the connections should plug directly in. If you are using a traditional Arduino UNO board, you may need a protoboard and additional jumper wires to make all the connections. See the website Yourduino.com for tutorials on connecting servos and a link to their support Wiki.

To start and stop the drawing machine, add a jumper wire (male-male) or a push button switch to connect digital input d6 to ground. Each time digital input d6 is touched to ground and released, the drawing machine will change state from running to stopped or stopped to running.

Power to the Arduino microcontroller can come from the USB programming cable, however many computer USB ports are limited to 0.5 amp which will not be enough current to reliably run both of the servo motors. It you observe the system starting and stopping or lurching a small amount, it may be due to insufficient power. To avoid this, plug in a 7 to 12v power supply rated at 1 amp or more to the Arduino board's dc power jack.

Complete the assembly and place the arms and disks on the servo motors. Tape down the rectangular base to a large surface, giving room for the arms to move. Tape an 11x17 inch piece of paper to the work surface beneath the pen, centering it so that there is plenty of margin for the full range of motion of the pen. The final step is programming of the microcontroller.

PROGRAMMING THE MACHINE

The code for running the drawing machine is presented in three code blocks, all of which are required for the system to run. Paste the code from all three blocks into the Arduino IDE, and upload the code to the microcontroller. For assistance with uploading the program, go to the website Arduino.cc. On the learning tab, you will find resources for getting started.

Code Block: (part 1)

```
/* draw_machine_v3.ino    Controls a two-servo drawing machine using a single reflectance
sensor */
#include <Servo.h> // servo library
#define DEBUG_ZP 1
#define DEBUG 0
Servo servo1_cont, servo2_angle; // create objects for both servo motors
int servo1_cont_Pin = 9; // continuous rotation servo on pin 9
int servo2_angle_Pin = 10; // standard servo (180 rotation) on pin 10
int IR_reflectance_Pin = A2; // QTR-1A reflectance sensor from Solarbotics
int StartStop_Pin=6; //start or stop movement by tying this pin to ground using switch
int MotorSpeedSetting=105; // default PWM value for slow motor movement
int PWMvalueForNoMovement=97;
int DiskRotationPositionVal; // analog value of IR reflectance sensor
int servo1_cont_Val,servo2_angle_Val;
boolean isRunning=false;
boolean isPressedStartStop=false;
boolean isOneFullRotationPulse=false;
boolean pastIsOneFullRotationPulse=false;
int servo2_angle_PositionCounter=1;
int angleIncrement=5; //5 is wide, 3 is narrower, 1 is fine

// ================================================================================
void setup() {
  Serial.begin(9600);
  pinMode(StartStop_Pin,INPUT_PULLUP); // turn on pullup resistor for start/stop switch
  servo1_cont.write(PWMvalueForNoMovement); //set the initial motor PWM value to stopped
  servo2_angle.write(0); // set the initial position of the servo motor to 0 deg
} //setup

// ================================================================================
void loop() {
  read_startstop_input_and_set_mode();
  read_rotation_sensor();
  check_if_one_complete_rotation_made_and_increment_counter();
  send_servo_values();
  if (DEBUG_ZP) {printMode(); printRotationPulse(); }
  if (DEBUG) print_debug_messages();
  if (servo2_angle_PositionCounter>150) { //end drawing if servo has moved 150 degrees
    servo1_cont.detach(); // turns off servo motors
    servo2_angle.detach();
    isRunning=false; // set mode to not running
  }
  pastIsOneFullRotationPulse=isOneFullRotationPulse; //
  delay(50); // give the servos time to react to new commands
} //loop
```

Code Block: (part 2)

```
// ----------------------------------------------------------------------------------------
void read_startstop_input_and_set_mode(){
  boolean isJustPressedStartStop=false;

  isJustPressedStartStop=(!digitalRead(StartStop_Pin) && !isPressedStartStop);
  // check if switch was just pressed.
  isPressedStartStop=!digitalRead(StartStop_Pin); // store last reading of on/off switch
  if (isJustPressedStartStop) {
      isRunning=!isRunning;
      if (isRunning) {
        servo1_cont.attach(servo1_cont_Pin); // attach the servo objects to digital pins
        servo2_angle.attach(servo2_angle_Pin);
      }
      else {
         servo1_cont.detach();
         servo2_angle.detach();
      }
  }  // if just pressed, change mode to running or not running
  DiskRotationPositionVal= analogRead(IR_reflectance_Pin); // read DiskRotationPosition value
} //read_startstop_input_and_set_mode

// ----------------------------------------------------------------------------------------
void read_rotation_sensor(){
 //read rotation pulse detector
  if (analogRead(IR_reflectance_Pin) >200) { isOneFullRotationPulse=true;}
  if ((analogRead(IR_reflectance_Pin) <100)) { isOneFullRotationPulse=false; }
  // inbetween a reflectance value of 100 and 200, the rotation pulse is not changed (gives
hysteresis)
} // read_rotation_sensor()

// ----------------------------------------------------------------------------------------
void check_if_one_complete_rotation_made_and_increment_counter(){
  if (isOneFullRotationPulse && !pastIsOneFullRotationPulse ) {
    servo2_angle_PositionCounter=servo2_angle_PositionCounter+angleIncrement;
    Serial.println(servo2_angle_PositionCounter);
  }
} // check_if_one_complete_rotation_made_and_increment_counter()

// ----------------------------------------------------------------------------------------
void send_servo_values() {
 // map the servo values received
  // to the servo's range of motion. If you're using different
  // analog sensors, here's where you adjust the map function. You
  // might also want to add a constrain function to make sure you're
  // keeping the values you're writing to the servos in the
  // range of 0-179, or in a narrow range to keep the movement slow.
  servo1_cont_Val = MotorSpeedSetting;
  servo2_angle_Val = servo2_angle_PositionCounter;

  // send the data to the servos
  if (isRunning) {
      servo1_cont.write(servo1_cont_Val);
      servo2_angle.write(servo2_angle_Val);
    }
    else
    { // set for no movement of motors
      servo1_cont.write(PWMvalueForNoMovement);
    }
} // send_servo_values()
```

Code Block: (Part 3)

```
// ------------------------------------------------------------------------------------
void printRotationPulse() {
  for (int i=0;i<DiskRotationPositionVal;i++) {
    if ( (i%50)==0) Serial.print(".");
  }
  Serial.println();
} // printRotationPulse

// ------------------------------------------------------------------------------------
void printMode() {
  if (isRunning) Serial.print("R");
  if (isOneFullRotationPulse) Serial.print("Z");
} // printMode

// ------------------------------------------------------------------------------------
void print_debug_messages() {
   Serial.print("AO: \t");Serial.print(servo1_cont_Val);Serial.print("\tA1:
\t");Serial.print(servo2_angle_Val);Serial.print("\tA2: \t");
   Serial.println(DiskRotationPositionVal);
   Serial.println(isRunning);
}
```

Code block 1 declares the program variables, configures the hardware pins, sets parameters that affect the drawing and runs the main code. Code block 2 contains support procedures that make the main code more readable. Code block 3 contains procedures used for diagnosing the system if there is a problem.

In code block 1, the loop() section describes how the system operates. The program repeatedly performs the steps in the loop(). First, the start/stop digital input is read and then used to determine if the drawing machine should be running or stopped. Next, the rotation sensor (ir reflectance sensor) is checked to see if the upper disk has made a complete rotation. If so, the angle of the lower disk is incremented slightly, changing how the ellipsoid is drawn. The amount of increment is set by the value angleIncrement. Finally, the servo motors are sent the new speed and position values by calling send_servo_values(). When the position servo has turned 150 degrees, the drawing machine is stopped.

```
  read_startstop_input_and_set_mode();
  read_rotation_sensor();
  check_if_one_complete_rotation_made_and_increment_counter();
  send_servo_values();
  if (DEBUG_ZP) {printMode(); printRotationPulse(); }
  if (DEBUG) print_debug_messages();
  if (servo2_angle_PositionCounter>150) { //end drawing if servo has moved 150 degrees
    servo1_cont.detach(); // turns off servo motors
    servo2_angle.detach();
    isRunning=false; // set mode to not running
  }
  pastIsOneFullRotationPulse=isOneFullRotationPulse; //
```

Tips

Once your drawing machine is up and running, you can vary the darkness of the line by adding weight on the arm near the pen. In figure 3f, a 9v battery was added and seems to provide sufficient down force. Avoid using felt tip pens as they will bleed ink when the drawing machine starts or stops.

The initial angle of the position servo motor will affect where the drawing occurs on the paper and the shape of the overall object drawn. Since it is easy to remove the disk from the servo and put it on at a different starting angle, try experimenting with this parameter and see what results you get.

Try modifying the code values that control the starting angle of the position servo (1) and the ending value (150) to cover a smaller range. Try changing the angleIncrement value from 5 to 1 once you have the system working well.

To create the type of drawings shown in the right side of figure 1, set the angleIncrement to 0 so that the position servo never turns. Set up the drawing machine, but don't tape the paper sheet down. Instead position the paper and the arms so that the paper can rotate in a complete circle without touching the rectangular base or any other obstacle. When the machine is started and the upper disk starts to continuously rotate, the pen will cause the paper to turn ever so slightly each time an ellipsoid is drawn. Let the machine run for a long time, 30 minutes or more, and observe the artwork it creates. If the paper is not turning, try placing a full-size foam core board under it to reduce friction.

Pedagogical Uses

The drawing machine has the potential for a range of pedagogical purposes. Starting from the simplest, the device is simply fun to watch as it creates art. The fully assembled system can be used at a Maker Faire to create drawings as young and old watch, as shown in figure 5. Young children are intrigued by the sound and motion of the mechanism, and will gladly wait for 10 minutes to get their own art print.

Figure 5 The drawing machine in use at a Mini Maker Faire. On the left is a drawing machine on which visitors can control the motion of the pen. On the right is a drawing machine with a fixed motion, but that creates a different work of art each time due to small variations in where the paper is initially laid relative to the pen.

A next level use would be for young students to assemble the machine from a kit of parts, putting together the arms and mechanisms, using a pre-programmed Arduino and prewired sensors. Students could try modifying the machine parameters, for example cutting a new base rectangle

to change the spacing between the motors from 17 cm to 22 cm and then observing how it affects the drawings. Different arm lengths and mechanisms could also be explored.

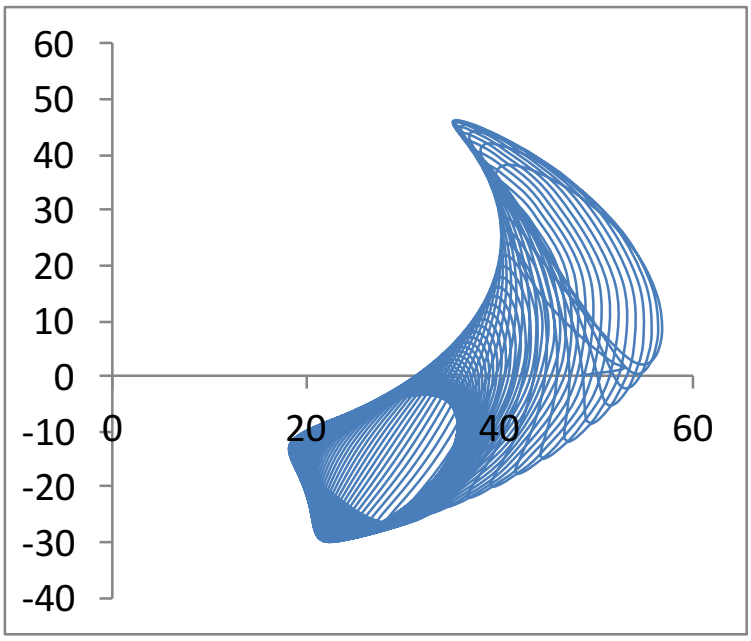


Figure 6 Simulation of spiral drawing in Excel using geometry and trigonometric relations.

At a higher grade level, students can apply trigonometry to be able to predict what the drawing will look like given the geometry of the machine's arms and rotating disks. For example, choosing the center of the lower disk to be the origin (x=0,y=0) of a Cartesian coordinate system is a starting point for writing equations to determine the location of the point on the disk where it connects to the arm, as a function of the disk radius and disk rotational angle. The students can continue down this path could build a mathematical model of the drawing machine and then simulate the drawings in Excel. An example of this is shown in figure 6 where Excel was used to create a prediction of the drawing that matches the actual drawing. Students can use the model to predict the effect of design changes such as changing the spacing between motors or changing the disk diameters, and then build a new machine geometry to produce their own customized art.

By working with another version of the spiral drawing machine, learners can use sensors to change the shape of the drawing in more radical ways. In the implementation shown in figure 7, a person can use one slide potentiometer to adjust the rotational speed and another to set the arm position. The rotation sensor can be engaged or disengaged by moving it away from the disk. Additional sensors, such as motion, sound etc., can be included to subtly influence the resulting drawing.

While the drawing machine of figure 7 allows direct control of the servo motors, it is very difficult to operate and doesn't generate the most interesting art. Humans have a difficult time coordinating the rotation of the two motors to produce pleasing drawings.

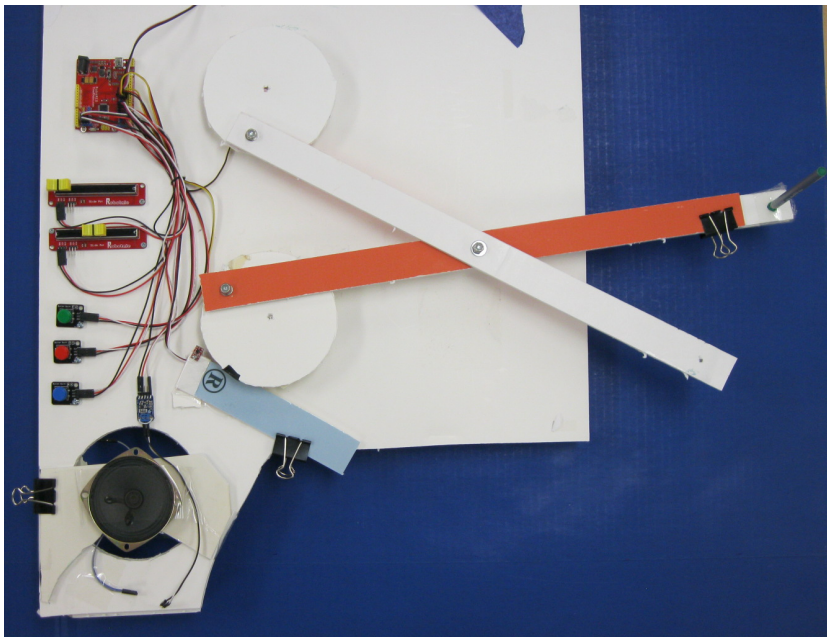


Figure 7 Another version of the drawing machine with human input to control the drawing. The two slide potentiometers (left side, yellow handles) control the two servos. One slider controls the rate of rotation for the lower disk, the second slider controls the angle of the upper disk. The rotation sensor can be moved away from the disk as it is mounted on a piece of foam board attached by a binder clip. A small speaker was added to provide simple sound effects (bottom left).

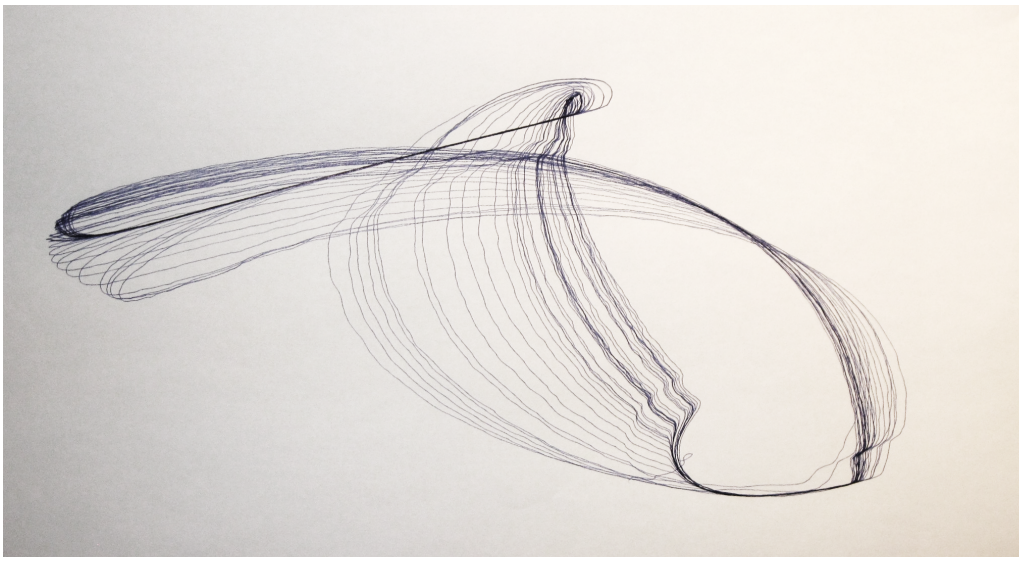


Figure 8 A drawing created by the drawing machine shown in figure 2 with modifications to the code so that the position of the servo motor disk is coordinated with the predicted motion of the continuous rotation motor disk based on time interval since the last rotation sensor pulse. In a programming class, this exercise can be used to teach principles of time measurement e.g. interrupts and sensor interfacing.

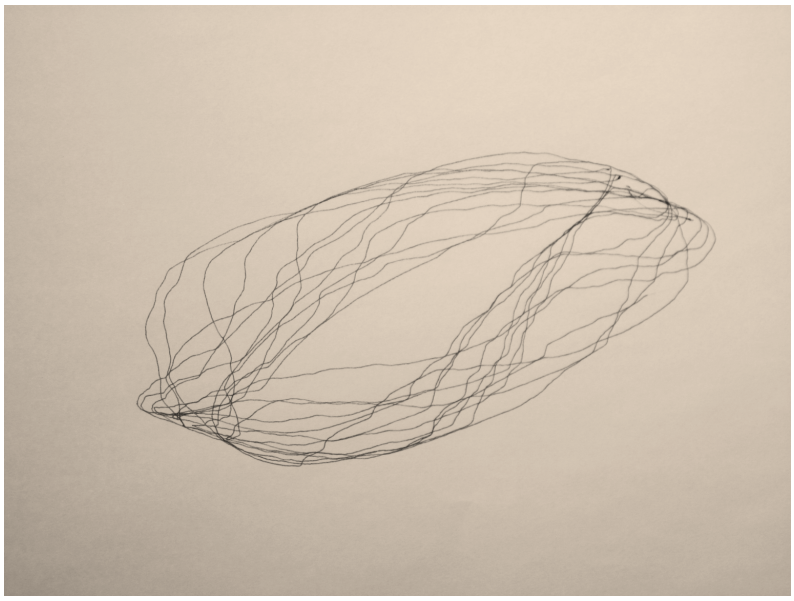


Figure 9 A drawing created by measuring a person's breathing pattern using a temperature sensor placed in front of their mouth. The position of the pen moves in response to the rate and depth of the person's breathing.

At a pre-college or college-level students can be challenged to write programs that create meaningful relationships between the two servo motor commands. Figure 8 shows a drawing where the position servo and the continuous rotation servo are moved in synchronism through programming. The code requires students to establish an estimate of the rotational speed of the continuous rotation servo and feed that into the control of the positional servo. This exercise can be used to teach about time measurement in microcontrollers, counters, and interrupts.

The drawing machine can also create art that reflects its environment by adding sensors such as light, temperature, etc. However, these parameters change very slowly, so they take days to create interesting art work. For classroom purposes, a faster changing signal is useful. Figure 9 shows a drawing produced by measuring a person's breathing rate using a temperature sensor held in front of their mouth. The small temperature variations are translated into changes in the position servo command resulting in a unique drawing pattern for each person.

One of the interesting aspects of the drawing machine is the emergent behavior observed. If the rotation sensor is disconnected and the sheet of paper is not taped down, then as the pen is pulled across the paper, the paper will rotate every so slightly. The amount and direction of rotation depends on how the paper is centered relative to the pen and the friction of the work surface under the paper. Letting the machine run for an hour in this configuration produces the beautiful patterns on the right side of figure 1. As a very advanced pedagogical use, students could create a model of the physical system of pen and paper including the friction forces on the paper and its inertia to predict the drawing pattern observed.

References

[1] Erik Brunvand, Jennifer (Ginger) Reynolds Alford, and Paul Stout. 2013. Drawing machines: exploring embedded system programming and hardware with an artistic flair (abstract only). InProceeding of the 44th ACM

technical symposium on Computer science education (SIGCSE '13). ACM, New York, NY, USA, 766-766. DOI=http://dx.doi.org/10.1145/2445196.2445530

[2] Erik Brunvand, Ginger Alford, and Paul Stout. Arduino Drawing Machine Workshop and Contest Syllabus 2014, SIGGRAPH2014, https://www.cs.utah.edu/~elb/siggraph/Syllabus2014.pdf, accessed May 1, 2016.

[3] Jackie Gerstein, The Educator as a Maker Educator, Kindle Edition, October 20, 2015, Amazon Digital Services LLC.