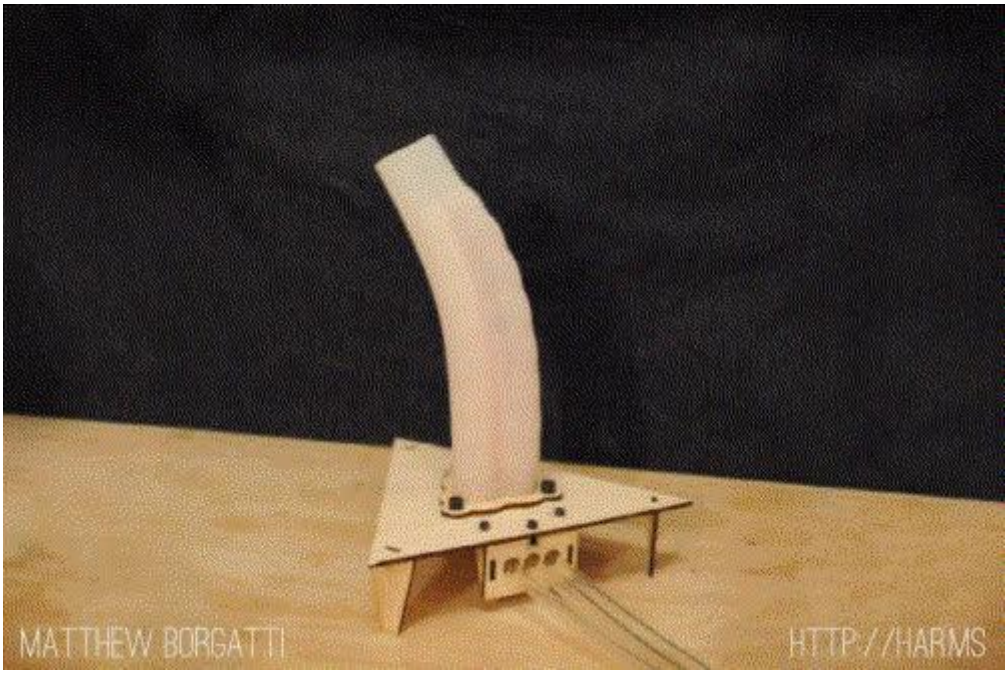
Silicone Robo-Tentacle

Created by Matthew Borgatti



https://learn.adafruit.com/silicone-robo-tentacle

Last updated on 2023-08-29 02:24:00 PM EDT

Table of Contents

A little background

Soft robots can potentially do a lot of jobs a hard robot made of steel and servos just can't do. Something composed of soft, flexible structures and actuators might be able to burrow through the dirt like an earthworm, conform to complex objects like a human hand, and go huge distances on minimal power just like organic machines (bats, bugs, dolphins, etc) do.

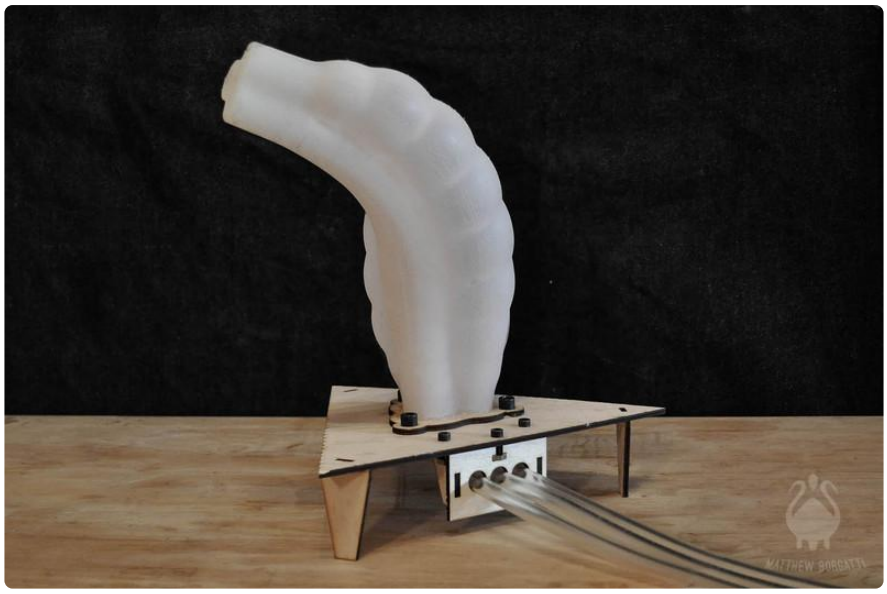
One reason you don't see too many robots like these is how difficult they are to design, plan, and manufacture. Either they're made of lots of interconnecting soft structures knitted together with glue and fasteners (each seam meaning additional labor, expense, and chances of breaking), or composed of a single skin.

I've been poking at easier ways to manufacture soft robots and think that these single skin designs have a lot of potential. I think that making robots this way could lower their cost while increasing their strength and durability. I've been calling these single skin robots plionics.

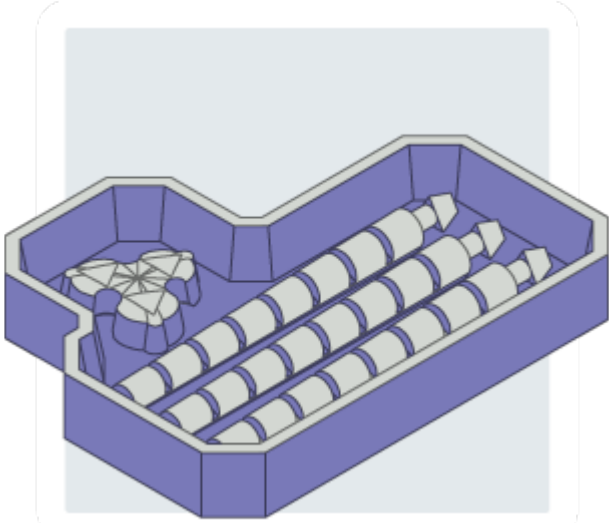
The method consists of designing your robot in CAD and working backwards from there to produce an outer mold and an inner core. Casting silicone between the mold and the core forms the robot itself and melting out the core gives you the finished product.

You can find the documentation behind a whole series of these robots here ().

In this tutorial I'm going to demonstrate how to put together one of my most successful robot designs (a strange squishy creature called the Trefoil Tentacle) using a combination of 3d printing, silicone casting, an arduino, and a bit of pneumatics.



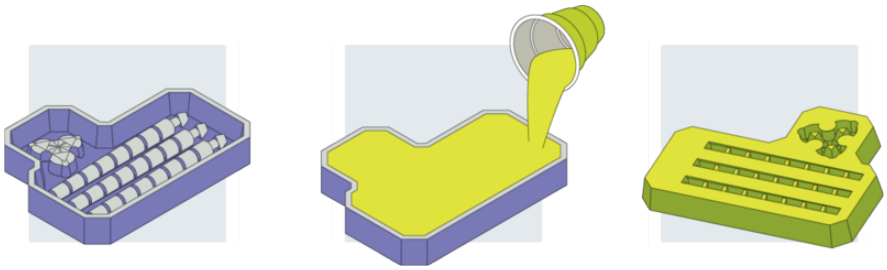
Printing



First, you'll want to print all these parts (). You'll need 3 copies of the outer shell and one of the bathtub-looking part shown above.

I used a modified Z-Corp powder printer courtesy of Viridis 3d (). However, chances are you could scale this project to fit on a FDM printer and it should print fine with a little support material here and there.

Bathtubs

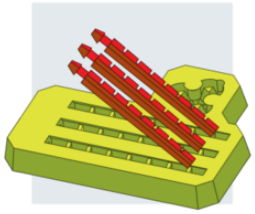


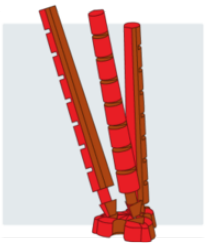
When you've got a bathtub printed you'll want to make sure it's sealed all around and won't leak. I'd recommend acetone smoothing () if you're going with FDM printed parts. For the powder prints I applied varnish and then floor wax to the mold to make sure no silicone soaked in. You'll want to give the FDM a light coat of wax or mold release as well.

Once your mold release is dry, go ahead and mix up some silicone and pour in a thin even stream into a corner of the mold until it's full to the top. I wrote a tutorial () on silicone casting a while back that you might like to take a look at.

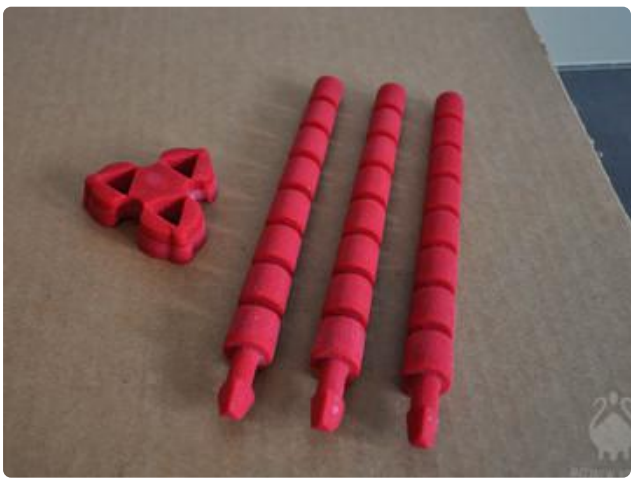
When your silicone is fully cured (this usually takes about 8 hours) demold it. You may have to get in there with an xacto to free up the little pyramids on the ends of those straight ribs, but those cuts will also help get the waxes out in the next step, so don't worry.

Cores





Next up: wax casting. I tend to use a double boiler for casting wax. Even though it's tempting to get lazy and cast wax just by heating a tin can of it over a stovetop, go the extra step, prevent your house from burning down, and heat your wax up in that same tin can on top of a pan of water on the stove instead.

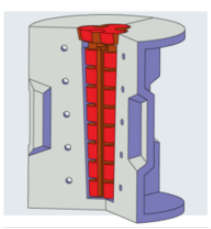


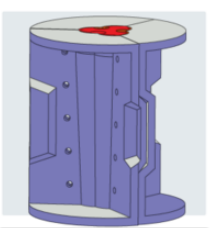
This tutorial () has some good tips on casting wax. Wax casting resembles silicone casting in a lot of ways, except the wax hardens as soon as it gets cold. If your mold is super cold, the wax can trap little bubbles against the surface and they'll end up as voids in your final wax part. Wax is also a pretty good insulator. It takes longer than you'd figure for your parts to fully cool so don't rush to pop them out of the mold even if the surface is lukewarm. Wait about an hour before demolding and assembling the core.

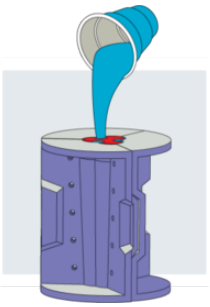
To make sure everything was aligned I cut myself an alignment jig ()out of acrylic. This isn't completely necessary if you're careful and keep the core straight as you put it together.

To assemble the core, fit all the pieces together and go around the seams between the ribs and the base with a hot soldering iron. If you're worried about wrecking your iron's tip, use a piece of wire wrapped tightly in foil as a replacement iron tip/wax knife. You might end up needing to add a bit of wax to the seams, so use some of the spare wax from your casting like you'd use solder and fill in wherever necessary.

Assembly



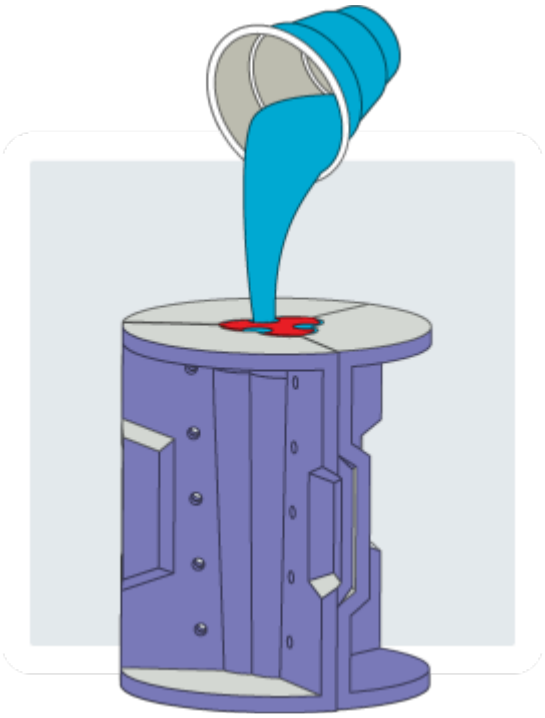




Here's where everything starts to come together. Assemble two of your mold shells with 1/4-20 bolts. If you rescaled your tentacle, you'll need to pick some fasteners to fit the smaller holes. Put the core in the mold and check the aligment. If the core collides with the mold walls or touches in the middle go ahead and trim it down with an xacto or rework it with your soldering iron.

Check the seams between the pieces for gaps or cracks. Chances are you'll have to put a bit of clay or putty anywhere that doesn't fit super snugly. It's one of the worst sensations in the world to spend a lot of time and care prepping and pouring a mold only to realize it's hemorrhaging silicone everywhere.

Casting

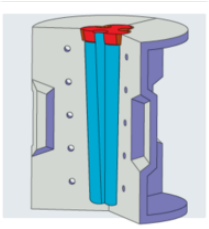


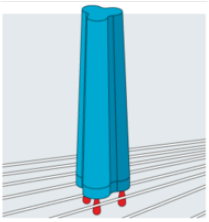
Now you pour the silicone that will become your tentacle. It's important that this material's thin enough to percolate into every bit of the mold and let air bubbles ventilate out. It also helps if it has a reasonably long gel time to allow every bit of air to escape. I went with a 50-50 mix of Smooth-On's Dragon Skin and their Eco Flex (which means mixing up a cup that's 50-50 of each material's part A and another of their part B's before combining everything for the final pour). If you want to make things go faster, you could always use a vacuum chamber ().

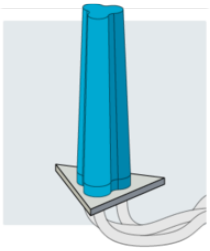
Pour in a thin stream, elevating the cup about a foot off the mold to help pop any bubbles trapped in the silicone from the mixing process. I've found that half filling the mold with silicone before inserting the core can help keep bubbles from collecting at the top and giving the tentacle structural flaws, but you've got to be careful to insert the core slowly to prevent it from trapping big bubbles in its nooks and crannies.

Wait overnight for everything to cure. I recommend pouring up a little swatch of silicone out of the same batch into a little dixie cup so you can check whether everything's curing without disturbing the mold itself.

Cleaning







Almost there. Once your tentacle has cured, unscrew your mold and push some craft sticks into the seams between the pieces. With some wiggling and working you'll be able to pop the mold and reveal your fancy new tentacle. If it's being stubborn a shot of pressurized air will often fart the part right out of the mold (seriously... That is exactly the noise of 100psi of shop air rattling through a silicone casting).

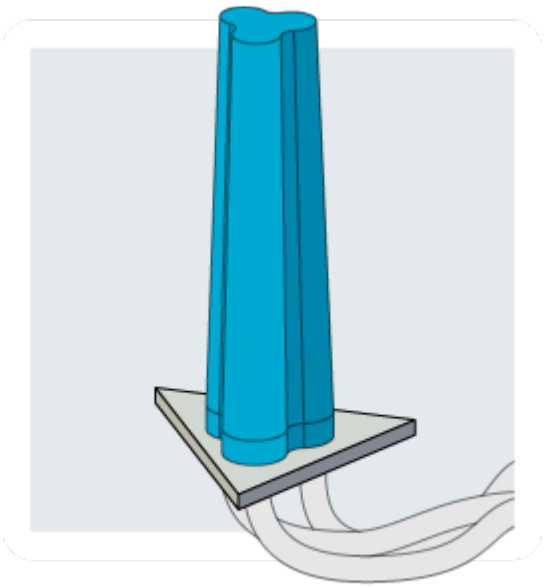
To get the wax out you could put your tentacle in a 300 deg oven for an hour and the wax will drip out. I'd recommend making a little boat out of tin foil and catching the wax to recycle it later. You could also boil the tentacle in hot soapy water, but this process gets pretty messy. If you're really lucky you can pull the wax ribs out with a pair of pliers. This works about one time in ten.

I use a cuticle nipper to take care of the thin flashing on the tentacle where silicone slips between the seams in the mold. Clean off any goo or dirt with a paper towel and some rubbing alcohol before powdering it with talc. The talc will keep it from collecting lint hair and animal dander.

After that, you'll want to get the tentacle on to a platform from which you can test everything and get the robot operational. I laser cut a stand with a removable tentacle slot to accommodate different test casts. You can find mine on Thingiverse (). It has holes for air lines that are just slightly smaller than the hoses themselves so everything stays in snugly and accidentally tugging on one doesn't pop it right out of the tentacle. Everything goes together

with silicone bathtub caulk, though it works more like a mortar than a true adhesive, so don't expect it to patch holes or make things 100% airtight. Again, everything will have to be scaled if you've changed the overall size of the tentacle.

Power and Programming



The final part involves getting air into this guy. If you just want to poke around with inflating your new tentacle in the most lo-tech and lo-cost way possible, hunt down a cheap sphygmomanometer () and cut the pump and air line off to use it as your air supply.

If you want precision control over your tentacle, it'll take a little more time and money. I use a cheap airbrush pump to provide the air and a solenoid valve assembly I found on eBay but any combo of air pump and solenoid valves (http://adafru.it/997) should do. I use small diameter vinyl tubing and luer fittings to get everything together.

What you'll need to do is hook up the air coming from the compressor to three solenoids and run a tube from each of them to each of the tentacle's chambers. A bleed valve between the compressor and the solenoids lets air gradually leave the system.

Get yourself an Arduino (http://adafru.it/50) and hook your solenoids to pin 3, 5, and 6. Load the code below on the Arduino.

```
const int pin[] = {6,5,3};
const int pincount = sizeof(pin) / sizeof(*pin); // number of items in pinset

unsigned long pin_time[pincount] = {0}; // 0-255
unsigned long last_cycle_start = 0;
unsigned long cycle_length = 100; // in milliseconds

void check_serial();
void setup() {
  for (int i = 0; i &lt; pincount; i++) {
    pinMode(pin[i], OUTPUT);
    digitalWrite(pin[i], HIGH);
  }
  Serial.begin(9600); // this gives us about 96 updates/sec, assuming 10 bytes/
update, which should be more than enough
  last_cycle_start = millis();

}

void loop() {
  check_serial();
  // there is a latent bug here; every 50 days, there will be a short cycle. This
probably doesn't matter, but here's where you'd fix it if it did.
  unsigned long cur_time = millis();
  unsigned long cycle_pos = cur_time - last_cycle_start;
  if (cycle_pos &gt; cycle_length) { // this is where the bug is.
    last_cycle_start = cur_time;
    for (int i = 0; i &lt; pincount; i++)
      if (pin_time[i] != 0)
        digitalWrite(pin[i], HIGH);
  } else {
    // cycle_length can't be that long; in fact, it can't be longer than an int.
    int cv = cycle_pos * 256 / cycle_length;

    for (int i = 0; i &lt; pincount; i++)
      digitalWrite(pin[i], (cv &gt;= pin_time[i]) ? LOW : HIGH);
  }
  delay(10);
  //Serial.print('.');
}

void handle_command(char cmd, int reg, int value) {
  switch(cmd) {
    case 'c':
      // set cycle time.
      if (value &lt;= 0) {
        Serial.println("!Cycle length negative");
        return;
      }
      cycle_length = value;
      break;
    case 'p':
      if (reg &gt; pincount || reg &lt; 1) {
        Serial.println("!Invalid pin no");
        return;
      } else if (value &lt; 0 || value &gt; 256) {
        Serial.println("!Pin duty cycle out of range");
        return;
      }
      pin_time[reg-1] = value;
      break;
    default:
      Serial.println("!Invalid command");
      return;
  }
  Serial.println("+OK");
}
```

```
void check_serial() {
  // protocol: lines containing command byte, ascii register number, '=', new
value, then newline.
  // commands are:
  //  - c: cycle time, in ms. No register number applies.
  //  - p: set pin duty cycle. Register number is pin number (1-n). Value is from
0-256, where 256 is fully on.
  // response is either '!' followed by an error message, or '+OK', optionally
followed by a response. No commands currently have a response.
  // Examples:
  //   c=1000  // set cycle time to 1s
  //   p1=0    // turn off pin 1
  //   p2=256  // turn on pin 2
  //   p3=128  // set pin 3 to half on

  const int
    ST_CMD = 0,
    ST_REGNO = 1,
    ST_VALUE = 2,
    ST_ERR = 3;
  static const char* errMsg = NULL;
  static int regno = 0;
  static int value = 0;

  static int state = 0;
  static char cmd = 0;

  // check if there's a serial byte waiting, and handle it.
  while (Serial.available() &gt; 0) {
    int inByte = Serial.read();
    switch(state) {
      case ST_CMD:
        if (inByte == '\n' || inByte == '\0')  {
          // invalid command
          Serial.println("!No command found");
          break;
        }
        cmd = inByte;
        state = ST_REGNO;
        regno = 0;
        break;
      case ST_REGNO:
        if (inByte == '=') {
          state = ST_VALUE;
          value = 0;
          break;
        } else if (inByte &gt;= '0' &amp;&amp; inByte &lt;= '9') {
          // no bounds checking. Regno can overflow
          regno = regno * 10 + inByte - '0';
        } else {
          state = ST_ERR;
          errMsg = "Expected digit or '=' in register no";
        }
        break;
      case ST_VALUE:
        if (inByte == '\r') {
          // windows box.
          break;
        } if (inByte == '\n') {
          state = ST_CMD;
          handle_command(cmd, regno, value);
          regno = 0;
          break;
        } else if (inByte &gt;= '0' &amp;&amp; inByte &lt;= '9') {
          // no bounds checking. Regno can overflow
          value = value * 10 + inByte - '0';
        } else {
```

```
          state = ST_ERR;
          errMsg = "Expected digit or newline in value";
        }
        }
        break;
      case ST_ERR:
        if (inByte == '\n') {
          Serial.print("!");
          Serial.println(errMsg);
          state = ST_CMD;
        }
        break;
    }
  }
}
```

Now, the Arduino is ready to PWM the solenoid valves (on pins 6, 5, and 3) in response to a signal over USB. The following Processing sketch will handle the interface.

```
import controlP5.*;
import processing.serial.*;

Serial myPort;
String p1 = "p1=0\n";
String p2 = "p2=0\n";
String p3 = "p3=0\n";

ControlP5 cp5;
int myColor = color(0,0,0);

int p1slider = 0;
int p2slider = 0;
int p3slider = 0;
int Frequency = 0;
Slider abc;

void setup() {
  size(450,400);
  noStroke();
  cp5 = new ControlP5(this);

myPort = new Serial(this, Serial.list()[0], 9600);

  // add a vertical slider
  cp5.addSlider("p1slider")
     .setPosition(100,50)
     .setSize(200,20)
     .setRange(0,256)
     .setValue(0)
     .setDecimalPrecision(0)
     ;

  cp5.addSlider("p2slider")
     .setPosition(100,100)
     .setSize(200,20)
     .setRange(0,256)
     .setValue(0)
     .setDecimalPrecision(0)
     ;

  cp5.addSlider("p3slider")
     .setPosition(100,150)
     .setSize(200,20)
     .setRange(0,256)
     .setValue(0)
     .setDecimalPrecision(0)
```

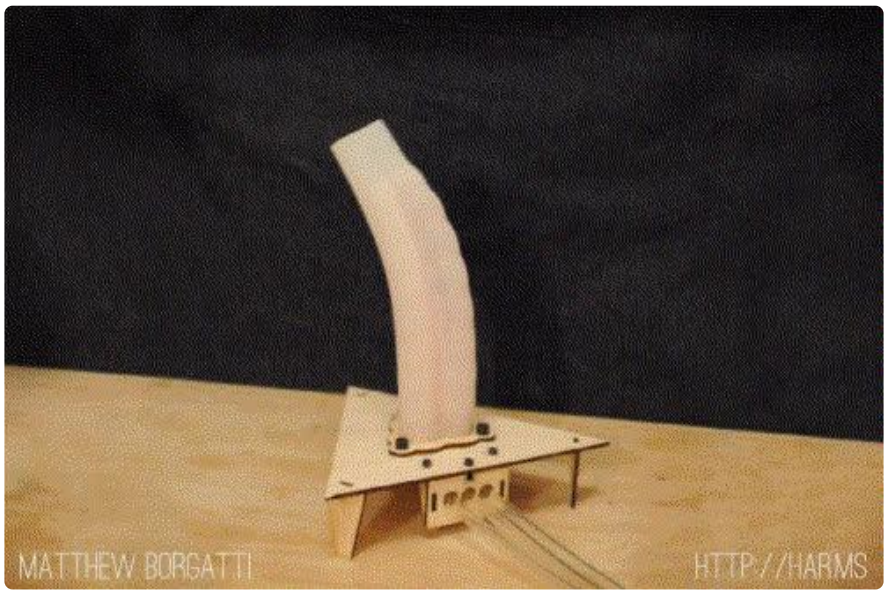
```
     ;

  cp5.addSlider("Frequency")
     .setPosition(100,200)
     .setSize(200,20)
     .setRange(10,250)
     .setValue(50)
     .setDecimalPrecision(0)
     ;

  cp5.addButton("off1")
     .setValue(0)
     .setPosition(315,50)
     .setSize(30,20)
     ;

   cp5.addButton("off2")
     .setValue(0)
     .setPosition(315,100)
     .setSize(30,20)
     ;

   cp5.addButton("off3")
     .setValue(0)
     .setPosition(315,150)
     .setSize(30,20)
     ;

   cp5.addButton("AllOff")
     .setValue(0)
     .setPosition(315,200)
     .setSize(30,20)
     ;

   cp5.addButton("Dia1")
     .setValue(0)
     .setPosition(100,250)
     .setSize(30,20)
     ;

   cp5.addButton("Dia2")
     .setValue(0)
     .setPosition(150,250)
     .setSize(30,20)
     ;

   cp5.addButton("Dia3")
     .setValue(0)
     .setPosition(200,250)
     .setSize(30,20)
     ;

  // reposition the Label for controller 'slider'
  cp5.getController("p1slider").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("p1slider").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("p2slider").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("p2slider").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("p3slider").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("p3slider").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("Frequency").getValueLabel().align(ControlP5.LEFT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
  cp5.getController("Frequency").getCaptionLabel().align(ControlP5.RIGHT,
ControlP5.BOTTOM_OUTSIDE).setPaddingX(0);
```

```
}

void draw() {

  fill(150);
  rect(0,0,width,height);

  //cp5.getController("p1slider").setValue(0);

  //add diagonal buttons, off buttons, all off button

  String p1 = "p1=" + p1slider + "\n";
  String p2 = "p2=" + p2slider + "\n";
  String p3 = "p3=" + p3slider + "\n";
  String freq = "c=" + Frequency + "\n";

  if(cp5.getController("off1").isMousePressed() == true ){
  p1slider = 0;
  cp5.getController("p1slider").setValue(0);
  }

  if(cp5.getController("off2").isMousePressed() == true ){
  p2slider = 0;
  cp5.getController("p2slider").setValue(0);
  }

  if(cp5.getController("off3").isMousePressed() == true ){
  p3slider = 0;
  cp5.getController("p3slider").setValue(0);
  }

  if(cp5.getController("AllOff").isMousePressed() == true ){
  p1slider = 0;
  p2slider = 0;
  p3slider = 0;
  cp5.getController("p1slider").setValue(0);
  cp5.getController("p2slider").setValue(0);
  cp5.getController("p3slider").setValue(0);
  }

  if(cp5.getController("Dia1").isMousePressed() == true ){
  p1slider = 256;
  p2slider = 256;
  p3slider = 0;
  cp5.getController("p1slider").setValue(256);
  cp5.getController("p2slider").setValue(256);
  cp5.getController("p3slider").setValue(0);
  }

  if(cp5.getController("Dia2").isMousePressed() == true ){
  p1slider = 256;
  p2slider = 0;
  p3slider =256;
  cp5.getController("p1slider").setValue(256);
  cp5.getController("p2slider").setValue(0);
  cp5.getController("p3slider").setValue(256);
  }

  if(cp5.getController("Dia3").isMousePressed() == true ){
  p1slider = 0;
  p2slider = 256;
  p3slider = 256;
  cp5.getController("p1slider").setValue(0);
  cp5.getController("p2slider").setValue(256);
  cp5.getController("p3slider").setValue(256);
  }

  myPort.write(p1);
  myPort.write(p2);
```

```
  myPort.write(p3);
  myPort.write(freq);

}
```

This thread () contains the above code and some experiments for getting the UI and control system together. It might help you get a clearer idea of what's going on if you find yourself stuck.

Done



tentacle_casting.pdf

If you've been following the tutorial you're now the owner of a fantastic creepy squishy soft robot. Congratulations! Use it to freak out your dog, impress your local hackerspace, or wave tiny patriotic flags at your next 4th of July or Bastille Day party.

You should download the PDF above, which describes the whole process with some simple illustrations. Hopefully that'll help you get started making awesome squishy robots in no time.

To see more soft robots I've designed, click here (). For some more information on the technology behind these bots, check out Super-Releaser (). For more of my own work, take a look at my site ().