MINDHUM
MIND-CONTROLLED INSTRUMENT

LISA SZOLOVITS

OVERVIEW

As a theater artist, my initial interest was in creating an instrument that could be played by a performer who is not a musician. I set about finding examples of music controlled by body sensors, with the thought that performers are trained movers, and that they could perhaps control their own heart rates or body temperatures through movement, which could then translate into shifts of sound, as well as freeing up the body for movement across the stage.

In my research, I came across Lisa Park's *Eunoia*, in which readings from an EEG sensor on her head are translated into musical sounds using Processing and Max/MSP, which are then played through speakers filled with water, creating a meditative, ritualistic performance. I've been thinking a lot about the mental virtuosity required of actors, and so this idea of using an EEG sensor strongly appealed to me as a way of translating an actor's skill into music-making. I began looking into what it would take to replicate Lisa Park's *Eunoia* instrument.

Later, I learned about Alvin Lucier's *Music for Solo Performer*, which skipped the step of translating EEG data into "musical" tones in a computer, and instead sent electricity directly from the sensors to little speaker cones attached to percussive instruments. The simplicity of this setup appealed to me, and reminded me of kinetic sculptures from the same period in the 1960s.

Aiming for this more stripped-down aesthetic, I set about building an instrument with three main components: an EEG sensor, an Arduino, and a kinetic sound-maker.

DEVELOPMENT

COMPONENT 1: EEG SENSOR

After researching available EEG sensors, I opted for the Neurosky MindWave Mobile headset. Designed for consumer use with mental exercise and meditation apps, the MindWave was less than $100, easy to wear and cordless because of its bluetooth connection, and seemed to be the EEG sensor of choice for hackers, since I was able to find a good number of projects using the device in forums on Cycling '74 and the like. In addition, the MindWave offered "attention" and "meditation" eSense data that analyzed Delta, Theta, Alpha, Low Beta, High Beta and Gamma waves for more dramatic readings. ***Though this was a good "starter EEG," it was not nearly as simple to hack as I'd assumed. If I were to do this again, I might either try using the most basic EEG electrodes directly, or invest in a headset with more contact points.*

COMPONENT 2: ARDUINO + BLUESMIRF GOLD (SPARKFUN BLUETOOTH MODEM)

In order to use the data coming from the MindWave EEG sensor to make music, I chose to connect the sensor to an Arduino. I was able to find a number of hacks for this online, and ended up cobbling together information from many of them to create the connection using a BlueSMiRF Gold modem and to write the Arduino code. My main resources were:

- Neurosky Developer Program's *MindWave Mobile and Arduino* tutorial: http://developer.neurosky.com/docs/doku.php?id=mindwave_mobile_and_arduino

- SparkFun's *Hacking MindWave Mobile* by Sophi Kravitz: https://learn.sparkfun.com/tutorials/hackers-in-residence---hacking-mindwave-mobile?_ga=1.249198019.555955934.1449848527

- Maker Press Project Book - *Make a Mind-Controlled Arduino Robot: Use Your Brain as a Remote:* https://books.google.com/books?id=9MPDlx_MFzoC&lpg=PP1&pg=PP1#v=onepage&q&f=false or http://www.control.aau.dk/~jdn/edu/doc/arduino/litt/MakeaMind-ControlledArduinoRobot.pdf

- SparkFun Forums "Bluesmirf Gold + Mac OS X:" https://forum.sparkfun.com/viewtopic.php?p=94557

COMPONENT 2: DC MOTOR + PIEZO CONTACT MICROPHONE

At first, I started working with a DC motor with the idea that I could use it to hit noisemaking objects such as a bells, for a kinetic music sculpture. So, I started playing around with this code to control the motor:

- Arduino Lesson 13. DC Motors: https://learn.adafruit.com/adafruit-arduino-lesson-13-dc-motors/arduino-code

Playing around with the motor speeds, I noticed that the motor itself very quietly emitted different pitches at different speeds. By attaching a piezo contact mic, I could amplify these sounds. And by scaling the MindWave's 0-100 Attention readings in the Arduino code to the DC motor's 1-255 speeds, I could control the speed/pitch of the motor with my mind!

INSTRUCTIONS

MATERIALS

- Neurosky MindWave Mobile
- Arduino Uno (or equivalent)
- BlueSMiRF Gold (SparkFun Bluetooth Modem)
- Small DC motor 6/9V
- Piezo contact microphone
- 9V battery plus holder
- Terminal software, such as CoolTerm, RealTerm, or Terminal
- Arduino development environment
- Bluetooth dongle, if your computer does not have internal bluetooth capabilities

CONNECTING THE MINDWAVE TO ARDUINO VIA BLUETOOTH

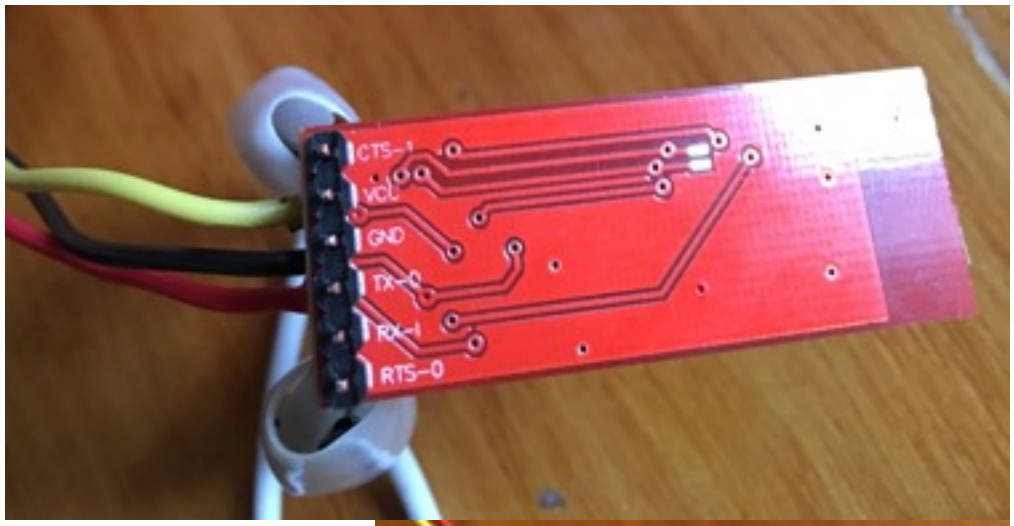
MindWave MAC Address

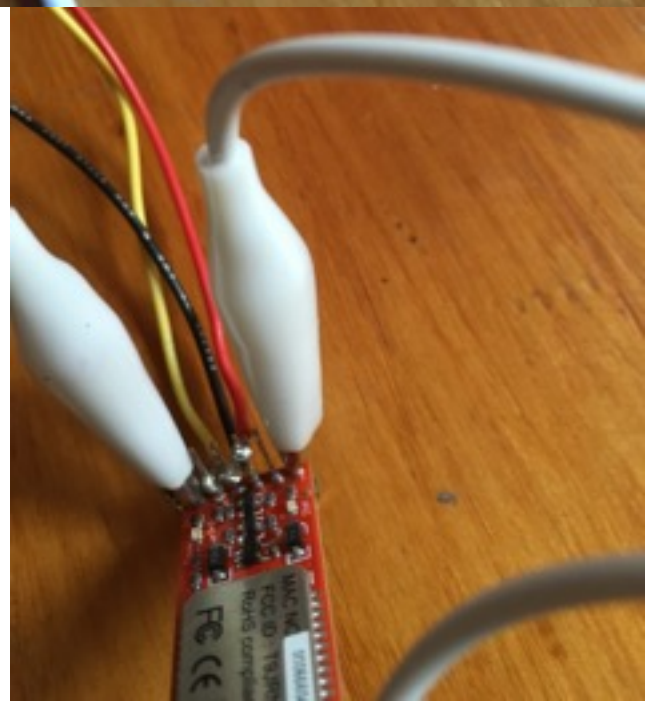
First, find out the MAC address of your MindWave Mobile by connecting it to your computer via Bluetooth. Turn on the computer's bluetooth and set the MindWave into pairing mode by pushing the on switch up for a few seconds until the blue light in the device double blinks. If a password is requested to pair, the MindWave default password is 0000. Once paired (which may take a few tries), control-click the MindWave listing under devices in the Bluetooth preferences window to find its 12-digit MAC address.

Configuring BlueSMiRF with your computer

You can turn off your MindWave now. Before you can pair your BlueSMiRF with your computer, you'll need to do some soldering to connect it like so:

- Short RTS and CTS together. I hooked an alligator clip between the RTS and CTS.
- Connect BlueSMiRF Vcc to Arduino's 3v3.
- Connect BlueSMiRF GND to Arduino GND
- Connect BlueSMiRF TX to Arduino RX
- (If you're using an Arduino Diecimilia, you'll also want to connect BlueSMiRF RX to Arduino TX)

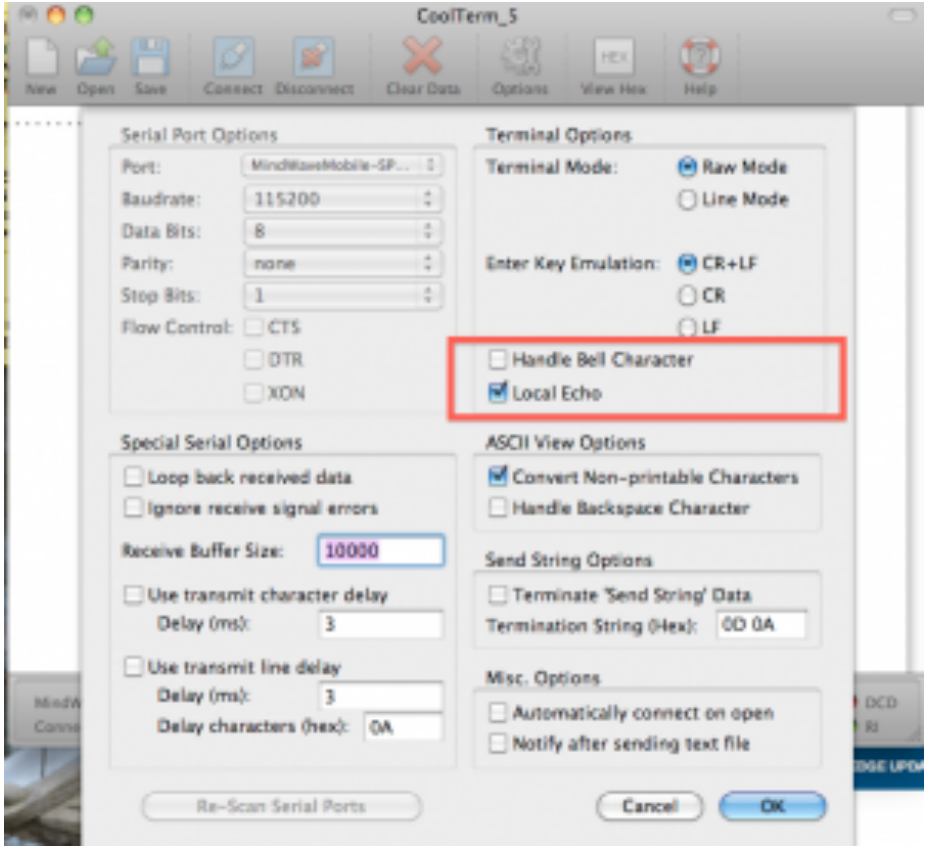




Once you've made all of these connections between the BlueSMiRF and Arduino, connect the Arduino to your computer, make sure your computer's Bluetooth is still on, and open CoolTerm or a similar terminal program on your computer. Now in the Bluetooth preferences window, look for FireFly-38C5 in your device list (that's the default name for the BlueSMiRF Gold). You will likely need to enter the pairing password, which is 1234 by default for the BlueSMiRF. This may take several tries.

Once you've paired the device, *quickly* go to CoolTerm, click Options and make sure the following selections are made:

- Port menu: FireFly-38C5-SPP
- Baudrate: 115200
- "Local Echo" box checked under Terminal Options



Once the BlueSMiRF has been paired, you'll have about 60 seconds in CoolTerm to do all this, click Okay, click the Connect button, and then in the CoolTerm main window, type: $$$
(DO NOT CLICK CARRIAGE RETURN AFTER)

If successful, you will see "CMD"

(If not successful, you'll need to repair the BlueSMiRF to your computer by turning off the computer's Bluetooth, unplugging the Arduino, and restarting the process. The CoolTerm options should stay the same, so it will be easier to quickly hit Connect and then type $$$.)

Type: D
(NOW you should hit carriage return after each typed entry.)

You will see the current configuration of the BlueSMiRF —>

Now type: SP,0000

```
$$$CMD
D
***Settings***
BTA=0006660438C5
BTName=FireFly-38C5
Baudrt(SW4)=115K
Parity=None
Mode   =Slav
Authen=0
Encryp=0
PinCod=1234
Bonded=0
```

This will change the pincode from '1234' to '0000'

Now type: SM,3

This will configure the BlueSMiRF to Auto-Connect Mode. Once the module is powered up, it will immediately look to connect.

You should see "AOK" if this is done properly.

Now type: SR, [Your MindWave's 12-digit MAC Address, noted earlier]

Again, look for AOK.

Now type: SU,57.6

This will change the BaudRate from 115200 to 57600

Type: D

Check to be sure that the stored address is the MAC address, and it's configured to Auto, not Slave

Type: — (three minus signs)

This will exit out of Coolterm. If successful, you will see END.

```
SP,0000
AOK
SM,3
AOK
SR,20689D3F547B <-or your MAC Address
AOK
SU,57.6
AOK
```

```
***Settings***
BTA=0006660438C5
BTName=FireFly-38C5
Baudrt=57.6
Parity=None
Mode  =Auto
Authen=0
Encryp=0
PinCod=0000
Bonded=0
Rem=20689D3F547B
```

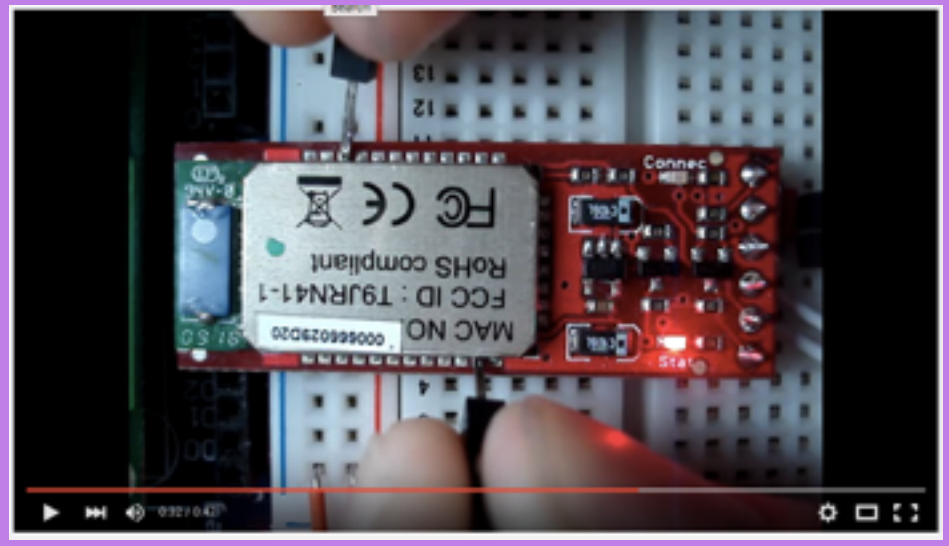
Checking BlueSMiRF configuration

Power on your MindWave Mobile headset. Power on your Arduino + BlueSMiRF board. If the BlueSMiRF LED turns green, and the MindWave Mobile LED turns solid blue, you are now connected and receiving data from the MindWave Mobile.

BLUESMIRF GOLD FACTORY RESET

If, like me, you fail to configure your BlueSMiRF properly (perhaps by disconnecting it before finishing reconfiguration), it may require a factory reset. Here's how to do it:

With the BlueSMiRF disconnected from the Arduino and any power source, touch one end of a jumper cable to pin 22 and the other end to pin 11. Then power the BlueSMiRF by connecting the VCC and Ground tot the 3.3V and GND (respectively) on the Arduino Uno. The red light on the BlueSMiRF will begin flashing quickly. "Toggle" (tap) the end of the jumper cable that is touching pin 22 at least 3 times - more rapid flashing will happen. Then disconnect power from the BlueSMiRF before restarting. It should be reset!



(I found these instructions in this handy YouTube video from gWikiY : https://www.youtube.com/watch?v=PU4TMeQSvWw

UPLOAD THE CODE TO ARDUINO

Disconnect your BlueSMiRF from your Arduino and turn off the MindWave. Open your Arduino program.

The following code is a hodge podge created from the sources listed in the Development section. Open a new window in Arduino, paste in this code, and upload it to your Arduino.

```
//MindHum - Mind-Controlled DC Motor Instrument
//Lisa Szolovits Dec. 2015
// Based on:
//helloattention.pde - Show attention level (EEG) with LED color.
// (c) Kimmo Karvinen & Tero Karvinen http://MindControl.BotBook.com
////////////////////////////////////////////////////////////
// Arduino Bluetooth Interface with Mindwave
// Sophi Kravitz edit 11-4
// Shane Clements edit 11-5
////////////////////////////////////////////////////////////
//Neurosky Sample Code from Neurosky Developer Site//
////////////////////////////////////////////////////////////
```

```
/* Disconnect TX and RX jump wires from Arduino when uploading from IDE.
Turn robot on, then in a couple of seconds turn headband on. */

const int ratePin = 9;
const int tinyLedPin = 13;

int tinyLedState = HIGH;

void setup()
{
	pinMode(ratePin, OUTPUT);
	pinMode(tinyLedPin, OUTPUT);

	Serial.begin(57600); // bit/s  // <1>

}

void loop()
{
	float att = getAttention();	// <3>
	if (att > 0)	// <4>
		setBlueToRate(att);
	toggleTinyLed();	// <5>
}
```

```
/*** Headset ***/

byte readOneByte()
{
	while (!Serial.available()) {	// <7>
		delay(5); // ms
	};
	return Serial.read();
}

float getAttention()
{ // return attention percent (0.0 to 1.0); negative (-1, -2...) for error
	byte generatedChecksum = 0;	// <8>
	byte checksum = 0;
	int payloadLength = 0;
	byte payloadData[64] = {
		0
	};
	int poorQuality = 0;
	float attention = 0;

	Serial.flush();	// prevent serial buffer from filling up // <9>

	/* Sync */
	if (170 != readOneByte()) return -1;	// <10>
	if (170 != readOneByte()) return -1;

	/* Length */
	payloadLength = readOneByte();
	if (payloadLength > 169) return -2;	// <11>
```

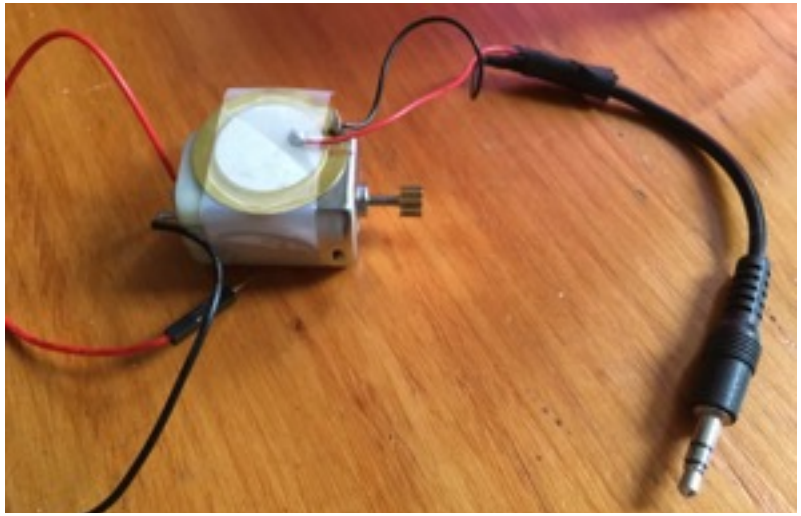
```
/* Checksum */
generatedChecksum = 0;
for (int i = 0; i < payloadLength; i++) {      // <12>
        payloadData[i] = readOneByte();        // Read payload into array
        generatedChecksum += payloadData[i];
}
generatedChecksum = 255 - generatedChecksum;
checksum = readOneByte();
if (checksum != generatedChecksum) return -3;    // <13>

/* Payload */
for (int i = 0; i < payloadLength; i++) {       // <14>
        switch (payloadData[i]) {
        case 4:        // <15>
                i++;       // <16>
                attention = payloadData[i]; // <17>
                break;
        case 2:
                i++;
                poorQuality = payloadData[i];
                if (200 == poorQuality) {

                        return -4;
                }
                break;
        case 0x80:        // skip RAW        // <19>
                i = i + 3;
                break;
        case 0x83:        // skip ASIC_EEG_POWER
                i = i + 25;
                break;
```

```
            } // switch
        } // for

        return (float)attention / 100;        // <20>
}

/*** Outputs ***/

void setBlueToRate(float ratePercent)
{
        int rate = ratePercent * 255;
        setRate(rate, 0);
}


void setRate(int rate, int yellow)
{
        analogWrite(ratePin, 255 - rate);

}

void toggleTinyLed()
{
        tinyLedState = !tinyLedState;
        digitalWrite(tinyLedPin, tinyLedState);
}
```

Once the code has been uploaded to your Arduino, you can disconnect it from your computer.

CONNECTING THE DC MOTOR

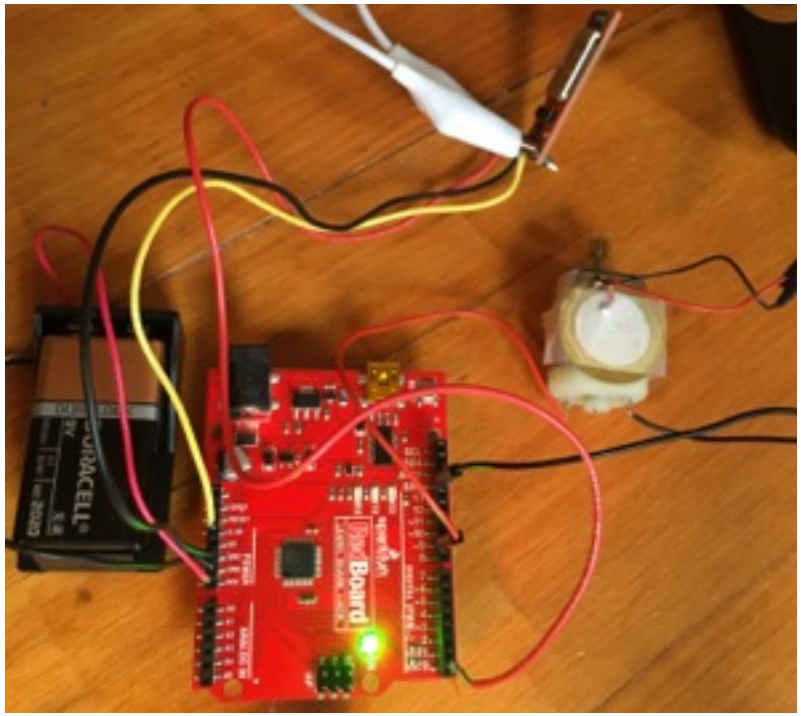
Take a piezo contact microphone and attach it with tape to your 6/9V DC motor. Now plug the microphone into a speaker, the power wire from the motor into Arduino 9 and the ground wire from the motor into Arduino GND.



PUTTING IT ALL TOGETHER

With the DC motor still hooked up to the speaker and Arduino, now you can reattach the BlueSMiRF to the Arduino as before:

- RTS and CTS still shorted together.
- Connect BlueSMiRF Vcc to Arduino's 3v3.
- Connect BlueSMiRF GND to Arduino GND
- Connect BlueSMiRF TX to Arduino RX
- (If you're using an Arduino Diecimilia, you'll also want to connect BlueSMiRF RX to Arduino TX)



Now, you can add an external power source, such as a 9V battery to power the Arduino and make it more portable. Connect the battery's power wire to Arduino VIN and the battery's ground wire to an Arduino GND.

All that's left now is to turn on the MindWave and pair it with the BlueSMiRF, turn on the speaker, and place the MindWave on your head. As your attention levels fluctuate, so will the speed and pitch of the DC motor. Ta da! A mind-controlled little humming instrument for your pleasure.