Voices in my Hat

Hanne Aho & Carmen Brecheis

1. Concept

For the second wearable technology course, we wanted to continue experimenting with the bone-conduction speaker that we used on the first course. Our starting point was the effect the speaker creates to the person who is wearing it: If the speaker is attached to your head, it feels like the sound is inside your head. After some discussion we came up with the idea that the voices of people who are close to you, would spin around in your head. Not in a negative sense like the angry voice of a mum which is in the head of the kids “clean up your room!”, no we rather wanted to explore this effect in a positive and funny way.

A second thought was also the possibility to use the voice as a physical medium to give a important message to a person like a gift. This could enhance the experience of memory (which is many time focused on visual appearance, rather than audible). So how could we build all these ideas into a tangible object?

In the Finnish student culture many students have a student overall that works like a display of the party life of the student. For each party or event you can collect a patch which is sewn onto the overall. So after some time your overall tells a story about your study and partying time. Since these patches are a textile material and the idea of collecting theses in combination with memorising your student history, we wanted to use patches for our project.

We decided to build a hat where you can attach several patches. The patch itself would be the physical object to the voice message you collect on your way from people who are close to you. If the patch is attached to the hat, the voice message will be played in the hat through the bone-conduction speaker. If there are more than one patch attached to the hat the hat will loop all voice messages of the patches which are attached to the hat.

2. Hardware and Software solutions

Hardware

- Arduino UNO
- Self-made Arduino shield
- SparkFun Voice Recorder
- 10 µF electrolytic capacitor

- 120 kΩ resistor
- 10 kΩ resistors (12)
- LED
- Surface transducer
- Electret microphone
- ON/OFF-switch
- 9 V battery Adapter for Arduino [3]

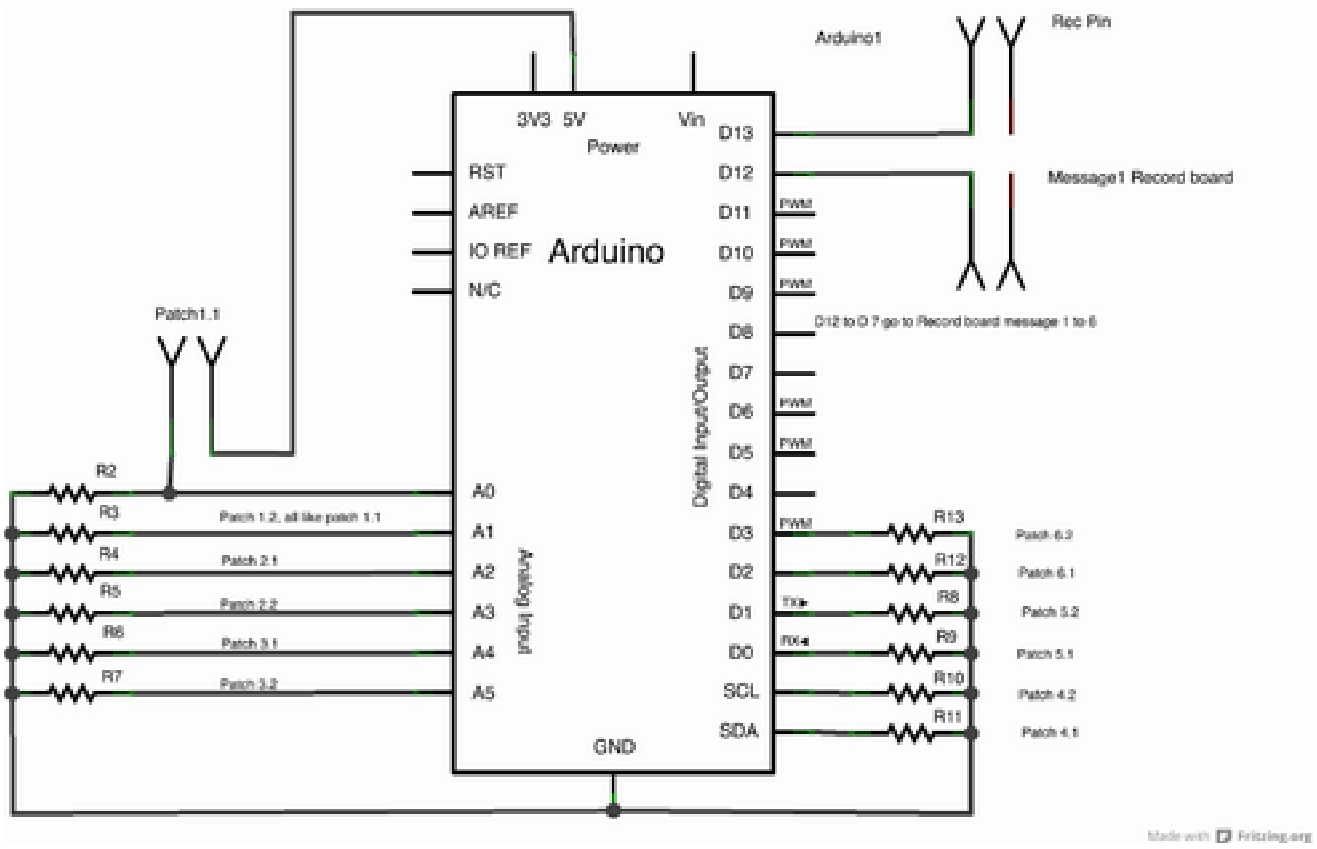

Figure 1: Sketch of Arduino connections

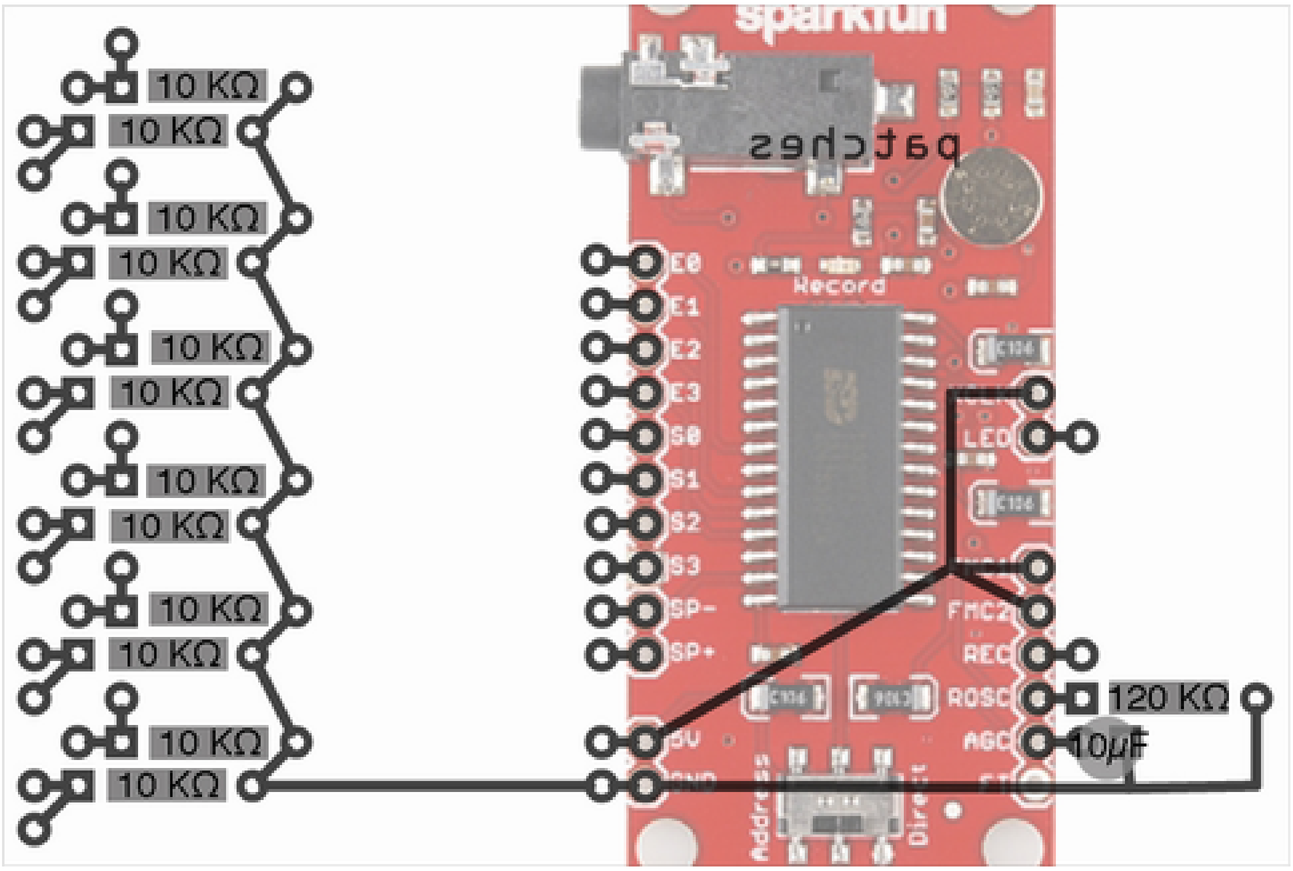

Figure 2: PCB design for our custom Arduino shield

Software

- Self-written Arduino program (see **Attachment 1** for the code)

Other materials

- Woolly bobble hat
- 6 crocheted patches with snap buttons on each corner
- Rubber band + snap buttons
- Carved styrofoam box for holding the electronics
- Plywood cover for the box

In this project, Arduino UNO is the brain: It receives the state of each patch (if they are in the play or the record mode) and tells the SparkFun Voice Recorder what to do. The Voice Recorder receives commands from Arduino and does one of the following: 1) Plays messages, 2) records and stores messages or 3) nothing. The Voice Recorder is soldered on a self-made Arduino shield which also handles the necessary connections between

Arduino, Voice Recorder and patches. Ian Lang's website [1][2] was a great help for us, when we tried to understand how the Voice Recorder functions.

Arduino, the self-made Arduino shield and the 9 V battery are hidden inside a carved styrofoam box with a plywood cover. The box is attached inside the top-part of the hat with rubber bands and snap buttons. Microphone and LED are hidden inside the bobble of the hat. If something is being recorded, the person needs to talk to the bobble, and the LED light is on. In the play mode, the LED shortly blinks after a message has been played. In addition, the bone conduction speaker is attached inside the hat so that the person who wears the hat, will feel it against his forehead. The hat also has a switch for turning it on or off.

3. Failures and dead ends

Our first sketches had the idea of having a small memory chip on each patch to store a voice message on each patch individually. However, quite soon we understood that this kind of system would be quite complex, and we had to change the plan. Finally, we decided to have the recording and playback ability in the hat itself, while patches would be only used as a switch.

After this decision, we started to search for something that could handle the audio processing for us, but finding an Arduino shield that would handle both playback and recording proved to be really difficult. In the end, we ended up ordering both AdaFruit Wave shield (not capable of recording audio) and the SparkFun Voice Recorder breakout board. The Voice Recorder arrived first, and our tests with it were successful, so we decided to stick to it. This proved to be a good choice.

As neither of us is an electrical engineer, we had to learn to understand some basic electronic matters the hard way. At some point, our hat started to function in a really strange way, although the code and our connections seemed to be fine. The reason for this was floating wires, which was finally solved with pull up and pull down resistors.

We have also been struggling with a problem that happens when Arduino is connected to the power source: The voice recorder starts recording even though it shouldn't. This might have something to do with the fact that the record pin of the Voice Recorder is connected to Arduino pin 13. Pin 13 has some special behaviour when Arduino is plugged to the power source. At some point, we got rid of this problem, but the night before the deadline it appeared again.

Our biggest failure was that, in the end, we couldn't get the project to work properly. The hat is not responding as expected, nothing is played through the speaker and nothing is recorded (except for the moment when Arduino is plugged in to the power source). The reason for this is still a mystery, since we didn't have time to carefully debug the problem.

Another very weak point in the whole setup of the hat has been the buttons. First we tested sewable snap buttons, but they were not good since they break easily and also deform easily while you solder wires on them. Therefore, we decided to go for more solid buttons which are normally used for anoraks. With these buttons the problem was that always two pieces have to be pressed together. One part of the button is under the fabric and the other part is in top of the fabric. So again the connection to the wire was the biggest problem and the soldering onto the button had to be done very carefully. Overall, the buttons are probably the weakest connection points in the circuit, and it might be the reason why the hat is not working.

4. Further steps

The obvious next step would be to make it work. This means we should check all the connections very carefully to find the possible mistakes and weak points.

References

[1] Ian Lang Electronics: How to Work the Sparkfun Voice Recorder ISD1932 Breakout Board in Direct Mode
http://ianlangelectronic.webeden.co.uk/#/voice-recorder-3/4562347318

[2] Ian Lang Electronics: Making your Arduino Talk by Remote Control
http://ianlangelectronic.webeden.co.uk/#/making-it-talk-vi/4562603284

[3] 9V Battery Adapter for the Arduino
http://playground.arduino.cc/Learning/9VBatteryAdapter

Attachment 1: The Arduino code

```
#include <SoftwareSerial.h>

/* PATCH 1 */

int patch1PinA = 0;
int patch1PinB = 1;
int patch1Out = 7;

int patch1ValueA;
int patch1ValueB;

int oldPatch1ValueA = 0;
int oldPatch1ValueB = 0;

boolean patch1ValueChanged = false;
boolean playPatch1 = false;

/* PATCH 2 */

int patch2PinA = 2;
int patch2PinB = 3;
int patch2Out = 8;

int patch2ValueA;
int patch2ValueB;

int oldPatch2ValueA = 0;
int oldPatch2ValueB = 0;

boolean patch2ValueChanged = false;
boolean playPatch2 = false;

/* PATCH 3 */

int patch3PinA = 4;
int patch3PinB = 5;
int patch3Out = 9;

int patch3ValueA;
int patch3ValueB;
int oldPatch3ValueA = 0;
int oldPatch3ValueB = 0;

boolean patch3ValueChanged = false;
boolean playPatch3 = false;

/* PATCH 4 */

int patch4PinA = 1;
int patch4PinB = 2;
int patch4Out = 10;

int patch4ValueA;
int patch4ValueB;

int oldPatch4ValueA = 0;
int oldPatch4ValueB = 0;

boolean patch4ValueChanged = false;
boolean playPatch4 = false;

/* PATCH 5 */

int patch5PinA = 3;
int patch5PinB = 4;
int patch5Out = 11;

int patch5ValueA;
int patch5ValueB;

int oldPatch5ValueA = 0;
int oldPatch5ValueB = 0;

boolean patch5ValueChanged = false;
boolean playPatch5 = false;

/* PATCH 6 */

int patch6PinA = 5;
int patch6PinB = 6  ;
int patch6Out = 12;

int patch6ValueA;
int patch6ValueB;
int oldPatch6ValueA = 0;
int oldPatch6ValueB = 0;

boolean patch6ValueChanged = false;
boolean playPatch6 = false;

/* REC PIN */
int recPin = 13;

void setup() {
  Serial.begin(9600);
  pinMode(recPin, OUTPUT);
  digitalWrite(recPin,HIGH);

  pinMode(patch1Out, OUTPUT);
  pinMode(patch2Out, OUTPUT);
  pinMode(patch3Out, OUTPUT);
  pinMode(patch4Out, OUTPUT);
  pinMode(patch5Out, OUTPUT);
  pinMode(patch6Out, OUTPUT);

  pinMode(patch4PinA, INPUT);
  pinMode(patch4PinB, INPUT);
  pinMode(patch5PinA, INPUT);
  pinMode(patch5PinB, INPUT);
  pinMode(patch6PinA, INPUT);
  pinMode(patch6PinB, INPUT);
}

void loop() {
  checkPatch1();
  checkPatch2();
  checkPatch3();
  checkPatch4();
  checkPatch5();
  checkPatch6();

  if (playPatch1 ||
      playPatch2 ||
      playPatch3 ||
      playPatch4 ||
      playPatch5 ||
      playPatch6) {
    playPatches();
  }
```

```
}

void checkPatch1() {
  patch1ValueA = map(analogRead(patch1PinA), 0, 1023, 0, 1);
  patch1ValueB = map(analogRead(patch1PinB), 0, 1023, 0, 1);

  delay(2000);

  if (patch1ValueA != oldPatch1ValueA || patch1ValueB !=
oldPatch1ValueB) {
      patch1ValueChanged = true;
  }

  if (patch1ValueA == 0 && patch1ValueB == 0 || patch1ValueA == 1 &&
patch1ValueB == 1) {
    if (patch1ValueChanged) {
      digitalWrite(patch1Out, HIGH);
      patch1ValueChanged = false;
      playPatch1 = false;
    }
  } else {
    if (patch1ValueA == LOW && patch1ValueB == HIGH &&
patch1ValueChanged) {
      patch1ValueChanged = false;
      playPatch1 = true;

    } else if (patch1ValueA == HIGH && patch1ValueB == LOW &&
patch1ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch1Out, LOW);
      delay(6000);
      digitalWrite(patch1Out, HIGH);
      patch1ValueChanged = false;
      playPatch1 = false;
    }
  }

  oldPatch1ValueA = patch1ValueA;
  oldPatch1ValueB = patch1ValueB;
}

void checkPatch2() {
  patch2ValueA = map(analogRead(patch2PinA), 0, 1023, 0, 1);
  patch2ValueB = map(analogRead(patch2PinB), 0, 1023, 0, 1);
  delay(2000);

  if (patch2ValueA != oldPatch2ValueA || patch2ValueB !=
oldPatch2ValueB) {
      patch2ValueChanged = true;
  }

  if (patch2ValueA == 0 && patch2ValueB == 0 || patch2ValueA == 1 &&
patch2ValueB == 1) {
    if (patch2ValueChanged) {
      digitalWrite(patch2Out, HIGH);
      patch2ValueChanged = false;
      playPatch2 = false;
    }
  } else {
    if (patch2ValueA == LOW && patch2ValueB == HIGH &&
patch2ValueChanged) {
      patch2ValueChanged = false;
      playPatch2 = true;

    } else if (patch2ValueA == HIGH && patch2ValueB == LOW &&
patch2ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch2Out, LOW);
      delay(6000);
      digitalWrite(patch2Out, HIGH);
      patch2ValueChanged = false;
      playPatch2 = false;
    }
  }

  oldPatch2ValueA = patch2ValueA;
  oldPatch2ValueB = patch2ValueB;
}

void checkPatch3() {
  patch3ValueA = map(analogRead(patch3PinA), 0, 1023, 0, 1);
  patch3ValueB = map(analogRead(patch3PinB), 0, 1023, 0, 1);

  delay(2000);

  if (patch3ValueA != oldPatch3ValueA || patch3ValueB !=
oldPatch3ValueB) {
      patch3ValueChanged = true;
  }
  if (patch3ValueA == 0 && patch3ValueB == 0 || patch3ValueA == 1 &&
patch3ValueB == 1) {
    if (patch3ValueChanged) {
      digitalWrite(patch3Out, HIGH);
      patch3ValueChanged = false;
      playPatch3 = false;
    }
  } else {
    if (patch3ValueA == LOW && patch3ValueB == HIGH &&
patch3ValueChanged) {
      patch3ValueChanged = false;
      playPatch3 = true;

    } else if (patch3ValueA == HIGH && patch3ValueB == LOW &&
patch3ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch3Out, LOW);
      delay(6000);
      digitalWrite(patch3Out, HIGH);
      patch3ValueChanged = false;
      playPatch3 = false;
    }
  }

  oldPatch3ValueA = patch3ValueA;
  oldPatch3ValueB = patch3ValueB;
}

void checkPatch4() {
  patch4ValueA = digitalRead(patch4PinA);
  patch4ValueB = digitalRead(patch4PinB);


  delay(2000);

  if (patch4ValueA != oldPatch4ValueA || patch4ValueB !=
oldPatch4ValueB) {
      patch4ValueChanged = true;
  }

  if (patch4ValueA == 0 && patch4ValueB == 0 || patch4ValueA == 1 &&
patch4ValueB == 1) {
    if (patch4ValueChanged) {
      digitalWrite(patch4Out, HIGH);
      patch4ValueChanged = false;
      playPatch4 = false;
    }
  } else {
    if (patch4ValueA == LOW && patch4ValueB == HIGH &&
patch4ValueChanged) {
      patch4ValueChanged = false;
      playPatch4 = true;

    } else if (patch4ValueA == HIGH && patch4ValueB == LOW &&
patch4ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch4Out, LOW);
      delay(6000);
      digitalWrite(patch4Out, HIGH);
      patch4ValueChanged = false;
      playPatch4 = false;
    }
  }

  oldPatch4ValueA = patch4ValueA;
  oldPatch4ValueB = patch4ValueB;
}

void checkPatch5() {
  patch5ValueA = digitalRead(patch5PinA);
  patch5ValueB = digitalRead(patch5PinB);

delay(2000);

  if (patch5ValueA != oldPatch5ValueA || patch5ValueB !=
oldPatch5ValueB) {
      patch5ValueChanged = true;
  }

  if (patch5ValueA == 0 && patch5ValueB == 0 || patch5ValueA == 1 &&
patch5ValueB == 1) {
    if (patch5ValueChanged) {
      digitalWrite(patch5Out, HIGH);
      patch5ValueChanged = false;
      playPatch5 = false;
    }
  } else {
    if (patch5ValueA == LOW && patch5ValueB == HIGH &&
patch5ValueChanged) {
      patch5ValueChanged = false;
      playPatch5 = true;

    } else if (patch5ValueA == HIGH && patch5ValueB == LOW &&
patch5ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch5Out, LOW);
      delay(6000);
      digitalWrite(patch5Out, HIGH);
      patch5ValueChanged = false;
      playPatch5 = false;
    }
  }

  oldPatch5ValueA = patch5ValueA;
  oldPatch5ValueB = patch5ValueB;
}

void checkPatch6() {
  patch6ValueA = digitalRead(patch6PinA);
  patch6ValueB = digitalRead(patch6PinB);

  delay(2000);

  if (patch6ValueA != oldPatch6ValueA || patch6ValueB !=
oldPatch6ValueB) {
      patch6ValueChanged = true;
  }

  if (patch6ValueA == 0 && patch6ValueB == 0 || patch6ValueA == 1 &&
patch6ValueB == 1) {
    if (patch6ValueChanged) {
      digitalWrite(patch6Out, HIGH);
      patch6ValueChanged = false;
      playPatch6 = false;
    }
  } else {
    if (patch6ValueA == LOW && patch6ValueB == HIGH &&
patch6ValueChanged) {
      patch6ValueChanged = false;
      playPatch6 = true;

    } else if (patch6ValueA == HIGH && patch6ValueB == LOW &&
patch6ValueChanged) {
      digitalWrite(recPin, LOW);
      digitalWrite(patch6Out, LOW);
      delay(6000);
      digitalWrite(patch6Out, HIGH);
      patch6ValueChanged = false;
      playPatch6 = false;
    }
  }

  oldPatch6ValueA = patch6ValueA;
  oldPatch6ValueB = patch6ValueB;
}

void playPatches() {
  digitalWrite(recPin, HIGH);

  if (playPatch1) {
    digitalWrite(patch1Out,LOW);
    delay(6000);
    digitalWrite(patch1Out, HIGH);
  }

  if (playPatch2) {
    digitalWrite(patch2Out,LOW);
    delay(6000);
    digitalWrite(patch2Out, HIGH);
  }

  if (playPatch3) {
    digitalWrite(patch3Out,LOW);
    delay(6000);
    digitalWrite(patch3Out, HIGH);
  }

  if (playPatch4) {
    digitalWrite(patch4Out,LOW);
    delay(6000);
    digitalWrite(patch4Out, HIGH);
  }

  if (playPatch5) {
    digitalWrite(patch5Out,LOW);
    delay(6000);
    digitalWrite(patch5Out, HIGH);
  }
  if (playPatch6) {
    digitalWrite(patch6Out,LOW);
    delay(6000);
    digitalWrite(patch6Out, HIGH);
  }
}
```