# DO(N'T) STAND SO CLOSE TO ME PART 3

## a story and some code

From the moment I started the project Softwearables, the ideas I worked with have to do with proximity and social conventions. The first step of my research on this topic was presented in The Ever Mass land February 2009. This trajectory continued during the summer and ended up as an installation in Zennestraat 17, relocated in Bozar.

### How do you greet a person, how strong/soft/close should a handshake be?

This phase of Softwearables is about the invisible zones you have around you (of course, all of this in a European context). When you talk to someone, you are inclined to stand at a certain distance from them.

Standing very close to someone is only permitted when you know them very well. Standing too far will influence your communication as well. In a tram, festival, crowded area, this notion of zone gets scrambled, yet you do kind of try to keep your personal space. In a city your personal space has different proportions compared to standing in an open desert. You position yourself differently.

With these personal observations/ideas in mind I started working on a wearable in two forms: “Do stand so close to me” & “Don’t stand so close to me”. Sometimes you want proximity and other moments it is very important to create your personal space. You can choose the version.

I programmed 4 zones – which are open for discussion. They can be changed easily in the Arduino chip of this installation.

→ very close: less than 25 cm
→ close: between 25 cm and 90 cm
→ futher: between 90 cm and 2 m
→ far: 2m and beyond

If you are in a specific perimeter of the wearable, a led will light up – and a voice will tell you at what distance you are standing, enabling you to really feel these invisible boundaries. The visitor can decide to interact and play – really experience what distance is still comfortable and when it is no longe the case.
In the periphery of this physical space, I did think about virtual proximity as well: on certain social platforms, the term "friend" looses its initial value. What if you want to accept someone as a stranger, or an acquaintance? Can you be so explicit in tagging someone as a non-friend?

## Technical specifications & Credits

Electronics are shiny, white, black, silvery closed boxes. I chose to work with open hardware and show the seams, components, threads, wire, and the minibrain of this installation which is a microprocessor called Arduino.
A Sonar connected to this chip detects the physical proximity of someone, which triggers four leds in a certain perimeter (very close, close, further and far). Voice(s) – a Waveshield on top of the Arduino – tell you at certain distances how many centimeters there is in between you and the person (mostly a mannequin) measuring. The public can wear the detection "sheet".
In order to get this set-up working, you need to programme the Arduino, flash some code onto it's little chip through usb. Once it has been programmed, it translates the signal it gets from the Sonar, which converts this into signals for the leds (into light) and into pulses which start a sound.
On the following pages I publish the code written to make this set-up work. It is not the cleanest code in the world – but it was made through cooperation. For the first part of the code – to make the leds work – I got a massive amount of help from Johannes Taelman (through the code31 malinglist – and some conversational proprietary software). The second part, making the sound work with the Waveshield could not have been possible without the excellent online tutorials from Limor Fried aka Ladyada. She develops the waveshield (physical object) and she explains how to work with it, providing code examples and forum support. The Forum helped me debug some of my code, and I had some "live" help – from two Pieters @ Hackerspace Brussels. Ludivine Loiseau did the graphic design on the text. Hereby thanks to everyone, including Kurt Vanhoutte from Timelab where I could make the vinylcut iron-able letters and Mannaert-Boets for transport.



Softwearables – do(n't) stand so close to me part 3 – is a moment in my research on physical computing, wearables and sound, a trajectory supported by the Vlaamse Gemeenschapscommissie.

At http://capacitor.constantvzw.org you can get a digital version of this text and the code.

```
//Softwearables Arduino frankenscript with Sonar
and Waveshield

#define RLED1 14 //testled on the duino
#define RLED2 15 //testled on the duino
#define RLED3 16 //testled on the duino
#define RLED4 17 //testled on the duino
#include <AF_Wave.h>
#include <avr/pgmspace.h>
#include "util.h"
#include "wave.h"
#define one "01.WAV"
#define two "02.WAV"
#define three "03.WAV"
#define four "04.WAV"
#define five "05.WAV"
#define six "06.WAV"
#define seven "07.WAV"
#define eight "08.WAV"
#define nine "09.WAV"
#define nine "10.WAV"
#define redled 9

AF_Wave card;
File f;
boolean file_is_open=false; // kan twee waarden h
ebben, false of true
Wavefile wave;      // only one!

int wasplaying = 0;
int rVal;
int pingPin = 7;

void setup()
{
 Serial.begin(9600);
 Serial.println("Wave test!");
 pinMode(RLED1, OUTPUT);
 pinMode(RLED2, OUTPUT);
 pinMode(RLED3, OUTPUT);
 pinMode(RLED4, OUTPUT);
 pinMode(2, OUTPUT);
 pinMode(3, OUTPUT);
 pinMode(4, OUTPUT);
 pinMode(5, OUTPUT);
 pinMode(redled, OUTPUT);

  if (!card.init_card()) {
    putstring_nl("Card init. failed!"); return;
  }
  if (!card.open_partition()) {
    putstring_nl("No partition!"); return;
  }
  if (!card.open_filesys()) {
    putstring_nl("Couldn't open filesys"); return;
  }

 if (!card.open_rootdir()) {
    putstring_nl("Couldn't open dir"); return;
  }
```

```
  putstring_nl("Files found:");
  ls();
}

void loop()
{
 long duration, inches, cm;

 // The PING))) is triggered by a HIGH pulse of 2
 or more microseconds.
 // We give a short LOW pulse beforehand to ensur
e a clean HIGH pulse.
 pinMode(pingPin, OUTPUT);
 digitalWrite(pingPin, LOW);
 delayMicroseconds(2);
 digitalWrite(pingPin, HIGH);
 delayMicroseconds(5);
 digitalWrite(pingPin, LOW);

 // The same pin is used to read the signal from
the PING))): a HIGH
 // pulse whose duration is the time (in microsec
onds) from the sending
 // of the ping to the reception of its echo off
of an object.
 pinMode(pingPin, INPUT);
 duration = pulseIn(pingPin, HIGH);

 // convert the time into a distance
 inches = microsecondsToInches(duration);
 cm = microsecondsToCentimeters(duration);

 Serial.print(cm);
 Serial.print("cm ");
 Serial.println();

 if (cm<25) {
    digitalWrite(RLED1, LOW);
    digitalWrite(RLED2, LOW);
    digitalWrite(RLED3, LOW);
    digitalWrite(RLED4, HIGH);
 }


 else if ((cm>26) && (cm<90)) {
    digitalWrite(RLED1, LOW);
    digitalWrite(RLED2, LOW);
    digitalWrite(RLED3, HIGH);
    digitalWrite(RLED4, LOW);
 }

  else if ((cm>91) && (cm<200)) {
    digitalWrite(RLED1, LOW);
    digitalWrite(RLED2, HIGH);
    digitalWrite(RLED3, LOW);
    digitalWrite(RLED4, LOW);
 }

  else if ((cm>201) && (cm<300)) {
```

```
    digitalWrite(RLED1, HIGH);
    digitalWrite(RLED2, LOW);
    digitalWrite(RLED3, LOW);
    digitalWrite(RLED4, LOW);
 }

if ((cm>2) && (cm<4)) {
    if (!wave.isplaying) {

playfile(one);
    }
 }

if ((cm>4) && (cm<6)) {
    if (!wave.isplaying) {

playfile(two);
    }
 }

if ((cm>19) && (cm<21)) {
    if (!wave.isplaying) {

playfile(three);
    }
 }

if ((cm>28) && (cm<32)) {
    if (!wave.isplaying) {

playfile(four);
    }
 }

if ((cm>47) && (cm<52)) {
    if (!wave.isplaying) {

playfile(five);
    }
 }

if ((cm>66) && (cm<73)) {
    if (!wave.isplaying) {

playfile(six);
    }
 }

if ((cm>96) && (cm<105)) {
    if (!wave.isplaying) {

playfile(seven);
    }
 }

if ((cm>196) && (cm<205)) {
    if (!wave.isplaying) {

playfile(eight);
    }
```

```
 }

if ((cm>246) && (cm<255)) {
    if (!wave.isplaying) {

playfile(nine);
    }
 }

if ((cm>295) && (cm<306)) {
    if (!wave.isplaying) {

playfile(six);
    }
 }

 delay(100);
}

long microsecondsToInches(long microseconds)
{
 // According to Parallax's datasheet for the PIN
G))), there are
 // 73.746 microseconds per inch (i.e. sound trav
els at 1130 feet per
 // second).  This gives the distance travelled b
y the ping, outbound
 // and return, so we divide by 2 to get the dist
ance of the obstacle.
 // See: http://www.parallax.com/dl/docs/prod/acc
/28015-PING-v1.3.pdf
 return microseconds / 74 / 2;
}

long microsecondsToCentimeters(long microseconds)
{
 // The speed of sound is 340 m/s or 29 microseco
nds per centimeter.
 // The ping travels out and back, so to find the
 distance of the
 // object we take half of the distance travelled
.
 return microseconds / 29 / 2;
}

void ls() {
  char name[13];
  card.reset_dir();
  putstring_nl("Files found:");
  while (1) {
    if (!card.get_next_name_in_dir(name)) {
       card.reset_dir();
       return;
    }
    Serial.println(name);
  }
}

void playfile(char *name) {
   if (wave.isplaying) {// already playing someth
```

```
ing, so stop it!
      wave.stop(); // stop it
    }
    if (file_is_open) {
      card.close_file(f);
      file_is_open = false;
    }

    f = card.open_file(name);
    file_is_open = true;
    if (!f) {
       putstring_nl("Couldn't open file"); return;
    }
    if (!wave.create(f)) {
      putstring_nl("Not a valid WAV"); return;
    }
    // ok time to play!
    wave.play();

}
```