# Vases Voices: Create and discover sound sequences through an interactive ambient device

**Tallulah GILLIARD**
Paris-Sud University
91400 Orsay, France
t.gilliard@gmail.com

## ABSTRACT

This paper presents the project "Vases Voices". It is an interactive 3D installation that allows the participants to explore sound through physical 3D objects beforehand created by sound processing and modeling. The goal of this project is to reify the human voice into an physical art object.

It can be set for different social environment depending of the purpose going from a house ambient device to a public transportation installation.

### Author Keywords

Processing Language, Sound processing, Bezier curves, 3D printing, Conductive Ink, Haptics, Sound Visualization, Ambient devices, Social communication

### ACM Classification Keywords

H.5.5 Sound and Music Computing (J.5) Signal analysis, synthesis, and processing. G. Mathematics of Computing, Continuous mathematics. B.m Miscellaneous

## Introduction

The purpose of this creation is to materialize a vocal message or memory note on a specific topic. This visual materialization is made through a vase whose shape represent the unique sound wave of the person's voice during the recording time. This sound sequence or other sequences can be replayed when a flower, associated with a voice message or a sound, is placed in the vase.

This vase of sound memories can be adapted to deliver a message in different social situation places.

In the familial setting, it will behave as an house ambient device that can allow the parents to leave daily voice messages to their children ("I love you", "Cake in the fridge", "good luck for the exam"). It could also permit to store and deliver vocal memories to descendants in between a photo album and a funeral urn.

In a public setting, it can allow former students of a university promotion to leave a trace and advices for new students through this personalized messages. It can also be set on tourist or busy sites, allow access information and sounds related to the place.

## Related Work

*Vase shape with sound visualization*

This work have been deeply inspired by the work of the artist François Brument [1] who made a vases collection with human blow waves. Every previous work that I found about sound visualization sculpture forms that have been done [1-8] helped me a lot in this project to find the adapted tools for sound recording, sound modeling, curve analysis and mathematical techniques, 3D software, 3D printing parameters and materials, and interaction techniques.

*Setting the device in an environment*

This device have first been thought as a tactile interactive vase that could deliver a unique single message per vase and the goal was to have an interactive floating forest of vases, inspired by the Laser Forest [10] made for public places like university halls or library, relaxation spaces, coffee rooms, and eventually transportations. Then, the idea of having a more ambient object that could adapt to different degrees of intimacy according the types of content, was triggered by the possibility to have a voicemail device like the Marble Answering Machine [9] but in a more personalized and topic oriented way.

## System Design

There is two way of interacting with the vase, the "Creator" side and the "Discoverer" side.

The Creator custom the interaction vase by recording a sound message, this sound is converted into a waveform curve and exported in a 3D software and transformed to 3D print it. Conductive ink and sensors connect the vase to the computer by creating a circuit. Conductive objects like flowers are action trigger that launch 6 different sounds and visual projections.

The Discoverer can move the different flowers to listen to the messages, discover a speech or even create a melody.

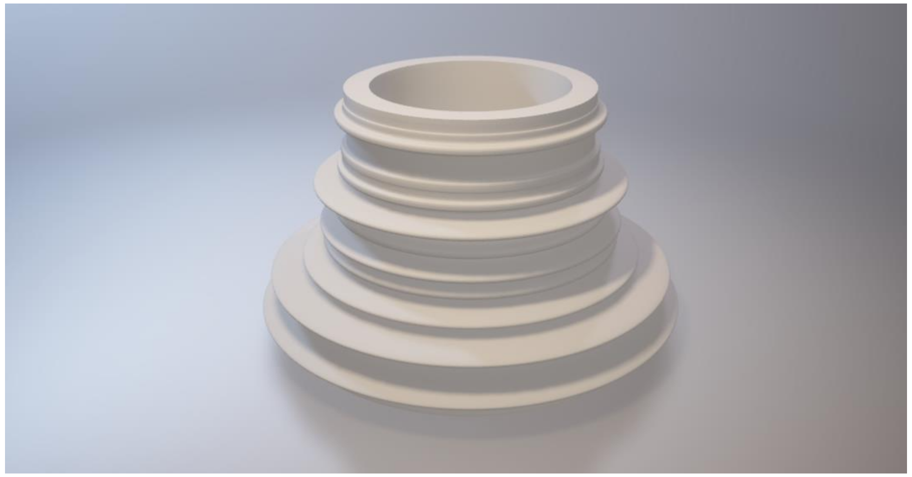

**Figure 1:** Vase in 3D (.stl file)

*Processing code*

Processing is an open-source computer programming language and integrated development environment (IDE) built for the electronic arts, new media art, and visual design communities with the purpose of teaching non-programmers the fundamentals of computer programming in a visual context. The Processing language builds on the Java language, but uses a simplified syntax and a graphics user interface. That's it was fitting the purpose of a graphical and design project.

I used the "Minim." Library, an audio library for Processing to record sound from the computer.
In the first version of the code, the sound was displayed only at the moment it was produced, so I instantiated it to have a timeline that depend of the sound produced during 10 seconds.
Then I had to do some math analysis to have a smoother and more understandable graphical rendering with Bezier Curves analysis.

Then I knew that I needed a .stl file from a 3D software to do the 3D printing. I was only displaying the curve so I learned how work a .svg file (vectorial file) to store and transform my points into that file.

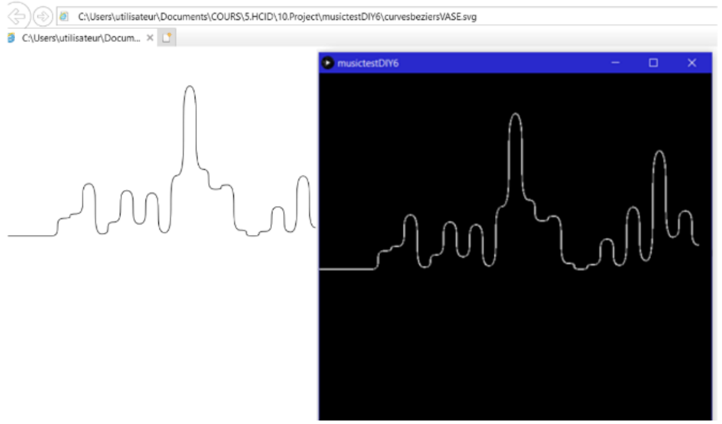
C:\Users\utilisateur\Documents\COURS\5.HCID\10.Project\musictestDIY6\curvesbeziersVASE.svg
C:\Users\utilisateur\Docum...
musictestDIY6

**Figure 2:** In black, a 10 second sound recording displayed by Processing.

In white the same sound in a .svg file

*Rhino3d*

Rhino3D is a commercial 3D computer graphics software. I used it to import the .svg file, do a rotation ("spin", Fig. 3a), adjust the details of the vase and export a .stl file (Fig.1)

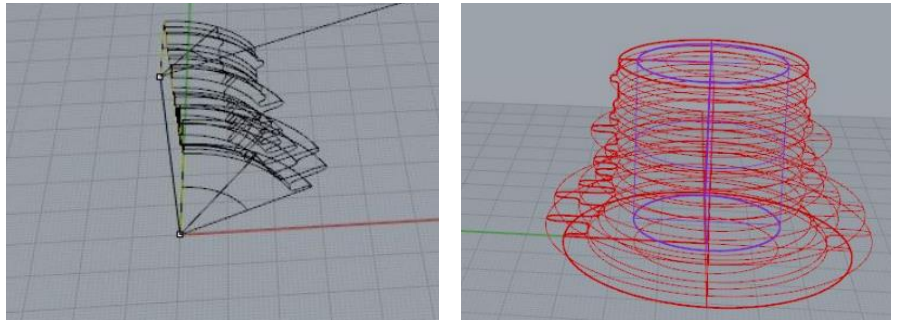

**Figure 3a and 3b:** Screen capture of the rotation process and detailing process (defining the inside of the vase)

*3D printing and conductive ink*

For the 3D printing I used a Dood 3D printer and PLA plastic (polylactic acid) which is a biodegradable and bioactive thermoplastic made by corn starch.
The bottom of the vase was painted with conductive ink from Bare Conductive.

*Connectivity and display*

The vase device was connected to the computer and to conductive elements (in this project, flowers) with a Makey Makey, an electronic tool and toy that allows users to connect everyday objects to computer programs by using a circuit board, alligator clips, and a USB cable
For the sound mixing performance part, I used a JBL loudspeaker connected in Bluetooth to the computer, an LED cable placed behind the vase and a Elephas projector was used to display the visuals

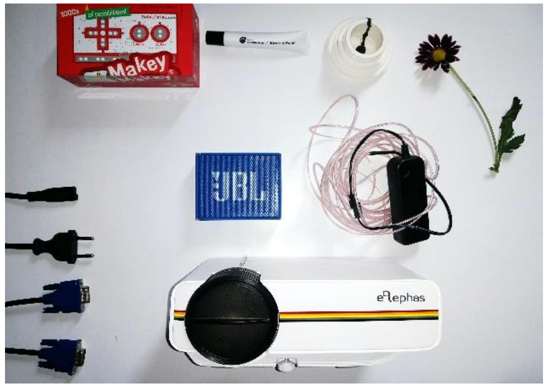

**Figure 4:** Material used

a. Makey Makey
b. Bare Conductive ink
c. 3D vase
d. flower
e. JBL go loudspeaker
f. LED cable
g. Elephas projector with cables

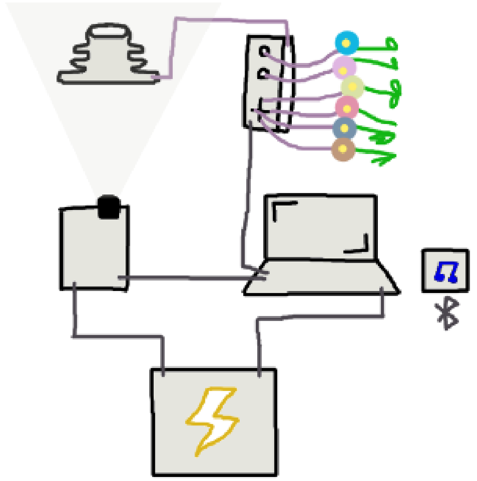

**Figure 5:**

Installation setup

## Recognition evaluation & Discussion

A preliminary study has been settled and four participants were asked to freely use the device as “discoverers” once it was set. Spontaneously, all of them have directly understand the possible actions and placed flowers in the vase, they also have explored the possibility to “play” it as a musical instrument. Two of them were curious about more interaction from the “Creator” side by asking if it was possible to record their own voices.

## Conclusion

This installation permitted to assemble several technologies (programming, 3D printing, tactile sensors, video projection) in the aim of transmitting an emotion on a topic, carried by a voice message transmitted by an unusual gesture (to put a flower in a vase). To go further, the presence of technology in favor of everyday objects should be completely removed or at least transparent : the vase and the flowers should be the only objects that remain visible from the user point of view. The use of a larger vase, requiring more elaborate printing techniques would also be a factor of better immersion. Finally, it would be better to simplify sound and vase production process and the installation with a simplified recording terminal of the author's main voice and a better integration of the cables between the vase, the flowers and the computer.

## References

[1]. *Vase#44* by François Brument
http://variation.paris/artists/brument/

[2]. *Functional 3D Printed Ceramics* by Olivier Van Herpt
http://oliviervanherpt.com/functional-3d-printed-ceramics/

[3]. Tōhoku Japanese Earthquake Sculpture – 2011
https://www.lukejerram.com/tohoku-japanese-earthquake/

[4]. Dust Serenade – 2010 http://www.orkantelhan.com/dust-serenade/

[5]. Andreas Nicolas Fischer data visualization work
http://studioanf.com/list/work/datavis/

[6]. *Cylinder* by Andy Huntington and Drew Allan – 2004
http://extraversion.co.uk/2003/cylinder/

[7]. Visual Music: A Waveform Made of Vinyl Records, Benga Single, Inspired by Seeing Sound – 2012
http://cdm.link/2012/05/visual-music-a-waveform-made-of-vinyl-records-benga-single-inspired-by-seeing-sound/

[8]. Microsonic Landscapes by Juan Manuel de J. Escalante – 2012 http://www.realitat.com/microsonic/

[9]. Marble Answering Machine by Durell Bishop – 1992
http://s3.amazonaws.com/arena-attachments/276277/ec24e3671cfa7035b6cd6ab7283675b4.pdf?1410290764

[10]. Laser Forest by Marshmallow Laser Feast
https://creators.vice.com/en_us/article/535n8n/exploring-new-canvases-meet-marshmallow-laser-feast

## Supplementary Materials

### Processing code for .svg vase creation

```
//==============================================================================
// GILLIARD Tallulah / avril 2018
//
//==============================================================================
import ddf.minim.*;
import processing.svg.*;
PrintWriter output;
Minim minim;
AudioInput in;
AudioRecorder recorder;

int sizeX; // screen size x
int sizeY; // screen size y
int i; // pixels
int dec = 20; // Décalage des points de contrôle
float previousValue = -2;

ArrayList values = new ArrayList<Float>();
ArrayList bezierVertexes = new ArrayList<Integer[]>();


//==============================================================================
// SETUP
//==============================================================================
void setup()
{
  frameRate(60);
  background(0);
  size(600, 600, P2D);
  sizeX = 600;
  sizeY = 600;
  i = 0;
  minim = new Minim(this);

  in = minim.getLineIn();
  // create a recorder that will record from the input to the filename specified
  // the file will be located in the sketch's root folder.
  recorder = minim.createRecorder(in, "myrecording.wav");

  // Add the first point
  bezierVertexes.add(new int[] {-1, -1, -1, -1, -1, -1});
  textFont(createFont("Arial", 12));

  // Create a new .svg file in the sketch directory
  output = createWriter("curvesbeziersVASE.svg");

}


//============================================================================
// DRAW LOOP
//============================================================================
void draw()
{
    int step = 20;
    if(frameCount % (60 / 50) == 0 && i < sizeX)
    {
      float currentValue = abs(in.left.get(in.bufferSize()/2)) * 500;
      if(i > 1 && i % step == 0) {
        stroke(255,0,0);
        strokeWeight(2);
        stroke(255,255,255);
        strokeWeight(1);
```

```
        // Mise en mémoire des points de béziers pour pouvoir les afficher ensuite
        if(i >= step) {
          int[] previousValue = (int[]) bezierVertexes.get(i / step - 1);
          if(previousValue[0] == -1)
            // first
            bezierVertexes.add(new int[] {0, sizeY / 2, i - dec,
              sizeY / 2 - (int) currentValue, i, sizeY / 2 - (int) currentValue});
          else
            bezierVertexes.add(new int[] {previousValue[4] + dec, previousValue[5],
              i - dec, sizeY / 2 - (int) currentValue, i, sizeY / 2 - (int) currentValue });
        }
      }
      i+= 1;
      values.add(currentValue);
    }

    // SVG storage process
    if (i >= sizeX){
        // SVG file if recording at the end of screen
        WriteSVG();
    }
    else
      // display curve on screen
      drawBezierCurve();

}
  //============================================================================
  // FUNCTIONS
  //============================================================================

  void drawBezierCurve(){
      // Beziers drawing process on processing screen
      createShape();
      beginShape();
      noFill();
      vertex(0, sizeY / 2);
      for(int i = 1; i < bezierVertexes.size(); i++) {
        int[] currentValue = (int[])  bezierVertexes.get(i);
        if(currentValue[0] != -1)
          bezierVertex(currentValue[0], currentValue[1], currentValue[2], currentValue[3], currentValue[4], currentValue[5]);
      }
      endShape();
  }

  void WriteSVG(){
      // write the svg inside the file
      output.println("<?xml version=\"1.0\" standalone=\"no\"?>");
      output.println("<!DOCTYPE svg PUBLIC \"-//W3C//DTD SVG 1.1//EN\"");
      output.println("\"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd\">");
      output.println("<svg width=\"500\" height=\"500\" version=\"1.1\"");
      output.println("xmlns=\"http://www.w3.org/2000/svg\">");
      output.println(" ");
      output.print("<path d=\"M 0 " + sizeY / 2 + " C ");
      for(int i = 1; i < bezierVertexes.size(); i++) {
        int[] currentValue = (int[])  bezierVertexes.get(i);
        if(currentValue[0] != -1)
            if (i == 1)
              output.print(currentValue[0] + " " + currentValue[1] + " " + currentValue[2] + " " + currentValue[3] + " " + currentValue[4] + " " +  currentValue[5]);
            else
              output.print(" S " + currentValue[2] + " " + currentValue[3] + ", " + currentValue[4] + " " + currentValue[5]);
      }
      output.println("\" stroke=\"black\" fill=\"transparent\"/>");
      output.println("</svg>");

      output.flush(); // Writes the remaining data to the file
      output.close(); // Finishes the file
  }
```