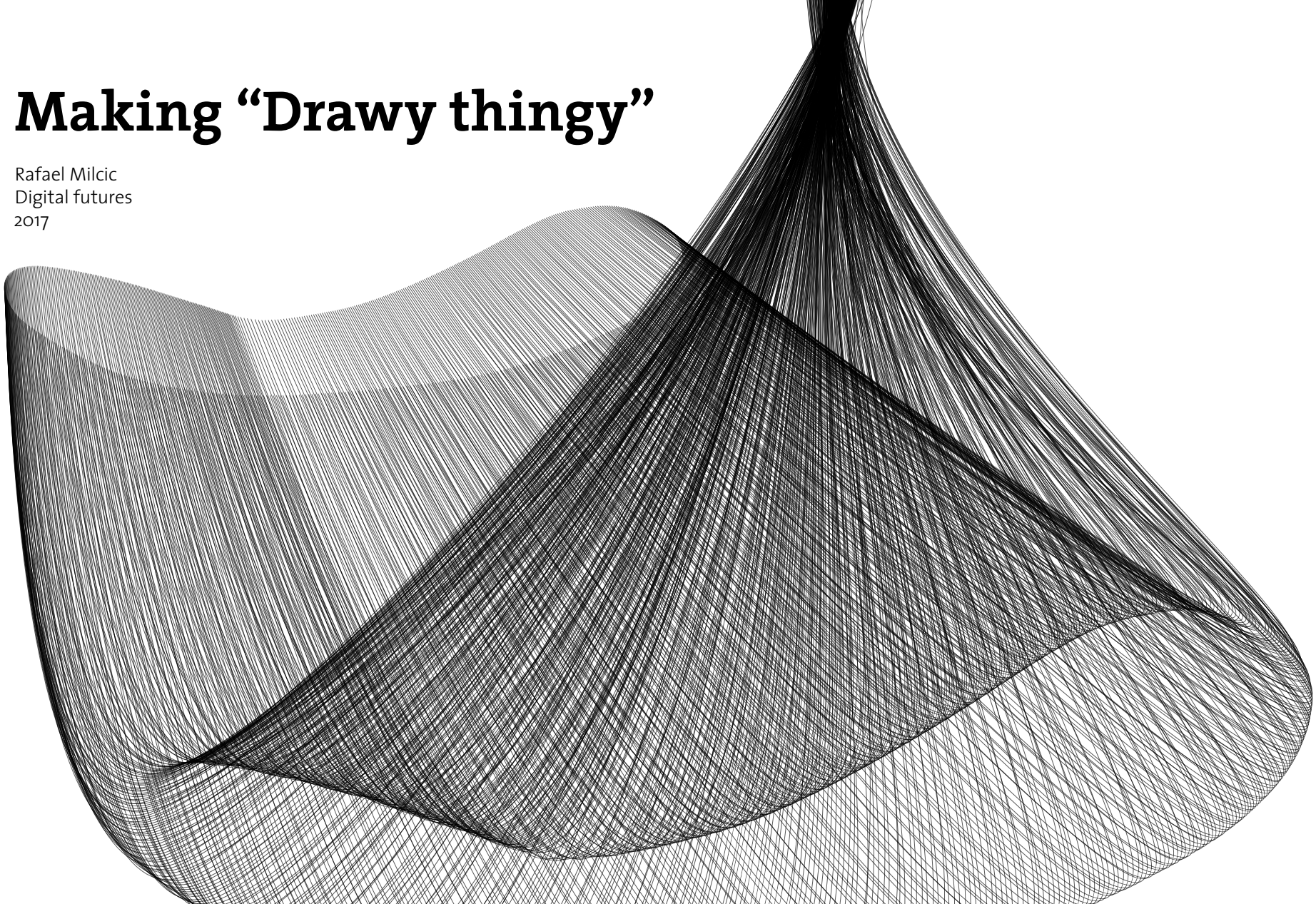
Making “Drawy thingy”

Rafael Milcic
Digital futures
2017

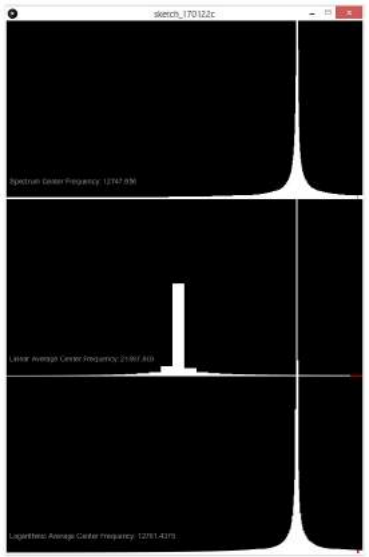


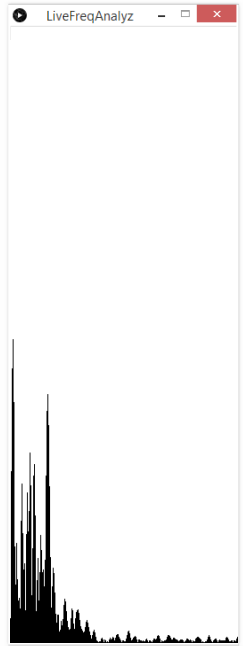
//Introduction

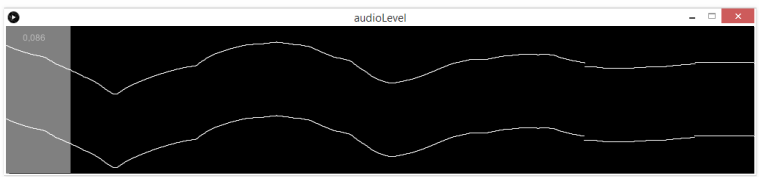
I came into the Digital Futures unit as a complete novice to coding, having written no code in any language. But, as it was something I wanted to work on and possibly integrate into my main practice, I was excited to start learning. The main thing that I noticed and loved about Processing was that it is, as other languages, a modular system, but written in a way that permits fast results and easier learning. With Processing, I could take any number and have it do anything. Thinking about coding in that way opened me up to the idea that it could tie very well into what I was doing in my main practice at the time, looking at the relationship between sound and image. I was excited when I saw that Processing could allow me to make visuals react to sound, and to make them react exactly how I want them to, not based on someone else's presets or framework. That flexibility is what pushed me later on to work and tweak the code, because I knew that whatever I imagined is possible to make.

Since I decided to connect this work to my main practice, my project explores Processing's sound capabilities. I started experimenting with programs that use input from the microphone to draw a simple waveform or to show the volume level. Later I moved on to analysis and reinterpretation of sound. But the final project, which was exhibited, focuses on fusing sound and drawing and introduces an interactive component.

// Various types of starting programs, mostly dealing with sound analysis





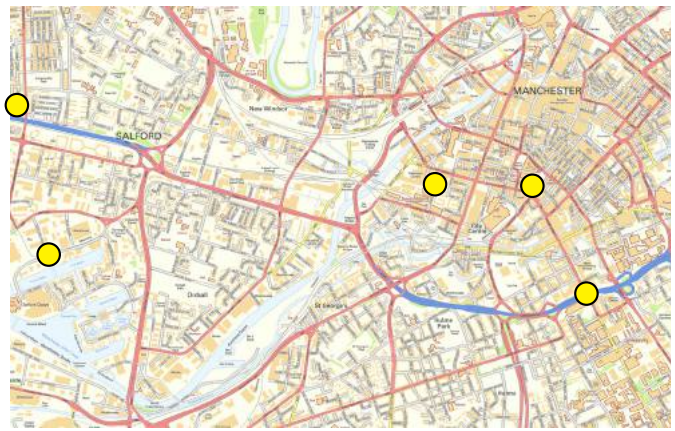


//Beginnings

The first concrete idea about the project was to record sounds around Manchester and use that as data to generate images based on the recordings. I wanted to get a visual representation of a sound and its characteristics. The images would change with the properties of the sounds themselves and untimately conncect the project to the problem of sound pollution. With my proposed project, you could get an image based on a piece of audio and compare it to another such image to draw conclusions about the type of sounds present and their differences. Visual inspiration was drawn fom Audiomulch and its metasurface interface [1]. However, after working on the project for quite some time, I did not manage to get the proof of concept sketch to work, and I was being slowly made aware of the fact that to make the idea work inside of the Processing environment would either be practically impossible or would require a high level of Java and DSP knowledge, none of which I posess.

After my failed attempts at the first idea, I tried to port the project to Max, which seemed suitable and flexible enough, as I witnessed from seeing my colleague Shaun's project. But, from a fairly short look around the program's possibilities, I realised that, similar to Processing's proprietary audio libraries, Max does not have an easy and quick way to integrate audio analysis. Thinking about it now, most platforms that I looked at seemed to have either playback or synthesis capabilities, with analysis being underrepresented.

//A map of all the locations of the sound recordings for my initially proposed

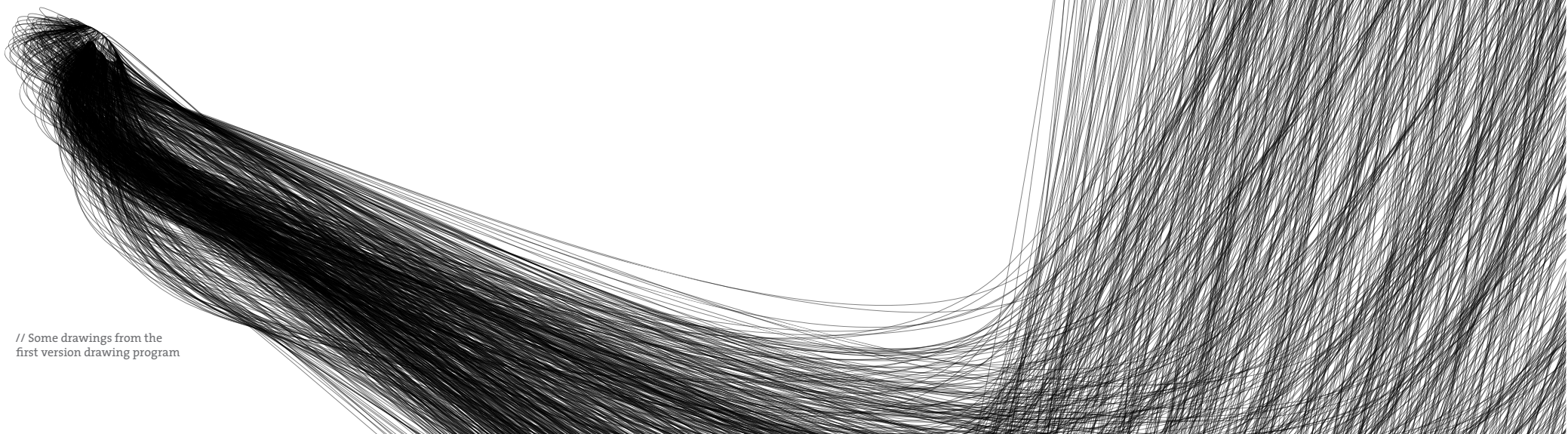


//[1] The interface that the DAW Audiomulch uses for its interpolation algorithm provided visual inspiration for my proposed project's visual characteristics

//The breakthrough

After scrapping the initial idea, while looking around the Processing reference, I was reading about how various types of vectors worked. I copied the small bit of code from the reference and started tweaking. A couple of hours and a lot of fun testing later, I had a small program that drew lines based on the position of the cursor, with constrained randomness to add visual interest to the lines. After playing with it more for a couple of hours and adding various features, I had a working prototype of an interactive drawing program.

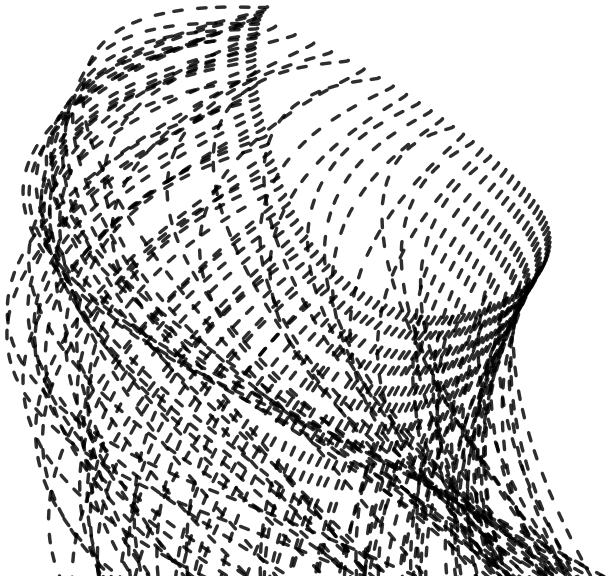
Then, I remembered the thing that drew me to coding in the first place. I can have any number do anything else! So I modified the program to start drawing when it detected sound through the microphone. It didn't work the first time, nor the second, nor eighth. But with a little luck and a lot of hair pulling, after a couple of days of mostly googling and digging through library references I managed to make it do what I wanted. That was the start of what was to be my exhibition project. What had to be done is mostly boring stuff, but important nonetheless. Fine tuning and making it more responsive, making the sounds influence the lines in a more significant way, and decisions about the aesthetics of the drawn objects.

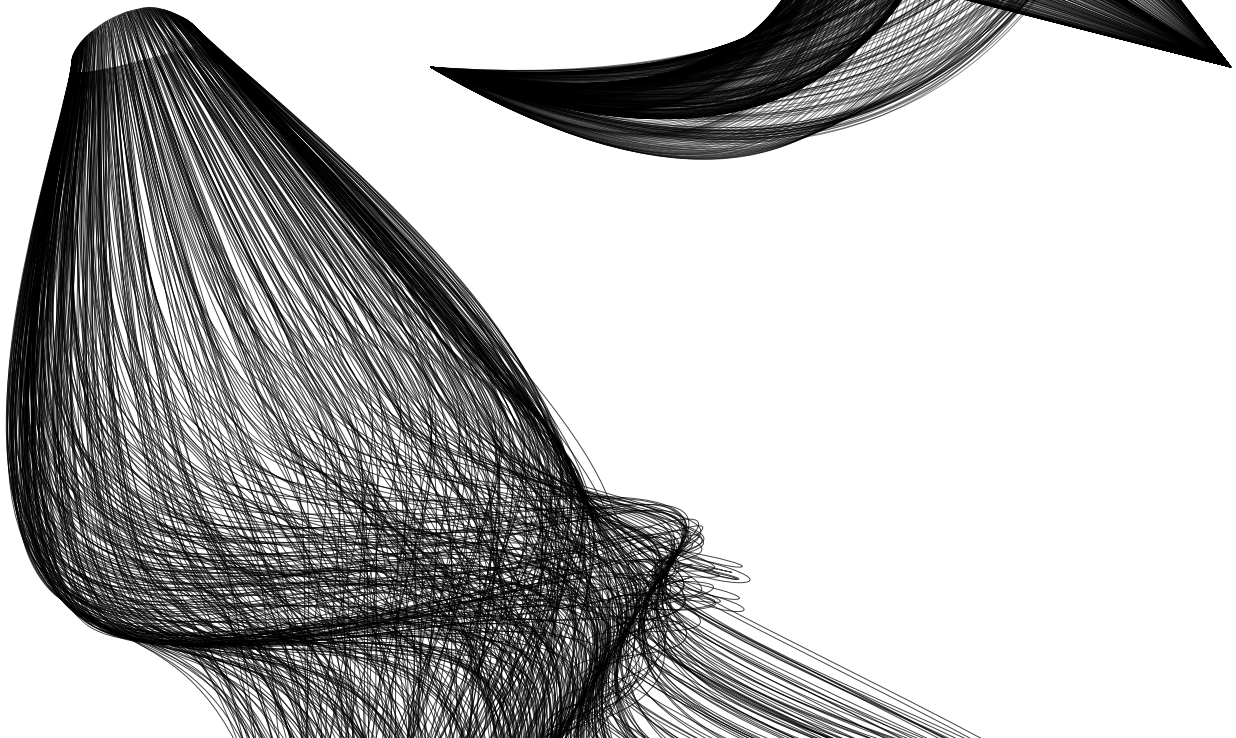


// Some drawings from the first version drawing program

I want to stress that a lot of previously mentioned steps involved taking other people's code and repurposing it to do something different. I believe that is a great way for beginners to learn by making and it thrills me to see such experimentation encouraged in the online community. The generous attitude most visible in the thriving open source community is primarily what made this project possible. As I delved deeper into the coding aspect of the project, I felt pretty overwhelmed by the amount of syntax i needed to know to make some of the more complex ideas work. The main helping hand was the Processing reference, but more importantly, many correspondences on the Processing forum with coding experts who seemingly have endless patience for explainining basic coding concepts to novices such as myself.

// The results were mostly unpredictable, but were still usable





//Run Interface

The exhibition itself was an eye-opening experience. Having witnessed many exhibition openings and some exhibitions while they were being put up, I knew approximately what to expect. However, the experience of me being the person whose work was exhibited and who had to make everything work proved to be both challenging and interesting. Even more so because of the fact that everything had to be put up and torn down in one day.

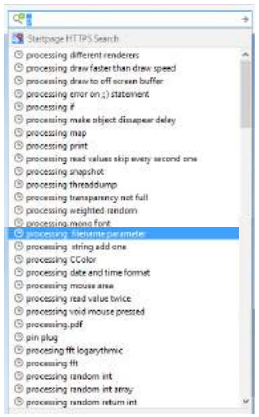
But, every complaint or worry that i might have had during the day went out the window as soon as the first person came around and started playing with the program. In an instant, it dawned on me. “Oh, that’s why people do interactive art!”. Just to see the childish curiosity and joy on people’s faces was more than enough to put an even bigger grin on my face and to finally relax and go around to look what my colleagues were up to. The night of the exhibition was also the time I decided to nickname the program “Drawy thingy” because I heard a passerby use that name to describe it to their friend.

Of course, the experience of the exhibition was far from perfect, there are some things that I would attempt to change if given more time to think about the project, both from

// People interfacing with the program at the exhibition

a technical viewpoint of writing the program and from a conceptual one.
To get the technical stuff out of the way, I could have written the code in such a way that it reacts better to common inputs. Although I tested it with talking, singing, tapping the table etc., some gestures such as snapping your fingers and clapping (which, interestingly enough, were most common) were not picked up well by the program. The most probable culprit was that optimising the code for latency reduction made it insensitive to those types of sounds. The program could also have been made reactive to high or low sound frequencies, instead of just the overall volume, thereby potentially strengthening the impact of the visuals.

The parts of the project I had most problems with in the beginning were more of a conceptual nature. An aspect I wasn't particularly happy with is the way the visuals themselves look. Ideally, I thought at first, it would be a marriage of scientifically accurate data and artistic presentation. But, since the program just used the sound data to trigger the drawing command and to some extent influence the visuals themselves, I felt that it was lacking in the "accurate" department. But, as the project neared its end, I decided to embrace the "artistic" aspect and just make it fun to interact with. In hindsight, that was a good choice, since i'm still not sure if the boundaries of working with sound in Processing are too big to practically support some of my ideas, or if i'm



// Adventures in Processing: google searches and reading values.

//Conclusion

Ultimately, the whole experience of this options unit was a learning one. Not so much learning the language itself, but more learning the basics of programming and having to complete a project from beggining to end. Diving straight into coding was a bit disorienting at first, but it feels like i learned a lot by exploring and every time I went to work on the code I gained new knowledge about the language. The exhibition at the end was a great way to put all of us into overdrive, finish our programs and start thinking about the presentation, not the technicalities of the program itself. It was also beneficial to see what everyone else was doing and also to get out of the coding headspace for a bit and to work on a practical project together with my colleagues.

code available at **www.github.com/Donrafaeli/Drawy-Thingy**