CONTEXTUALISING ESSAY FOR THE MOBILE SYNTHESISER

GESTUR.A

Candidate Number:	**196484**
Module Name:	**Interactive Project Dev.**
Module Code:	**871P4**
Academic Year:	**Spring 2019**

INTRODUCTION

Gestur.a is an interactive musical synthesiser app, built for iOS. It is currently in the iterative prototyping stage. The synthesiser is controlled by a mixture of different affordances - user gestural motion, on screen touch parameters, and the microphone. The purpose of the app is to give users the ability to make expressive music with their iPhone in an easy and intuitive way, and to be a platform where they can share these compositions with friends and collaborators. Gestur.a is built in Xcode, using the LibPD framework to connect with PureData - the open source visual programming language that it uses for audio processing and synthesis.

This is an ongoing project, and this essay aims to contextualise the project in the field of music technology and instrument / interactive design and provide a rationale for, and a description of, the development to date.

THEORETICAL BASIS

DESIGN AND PROTOTYPING

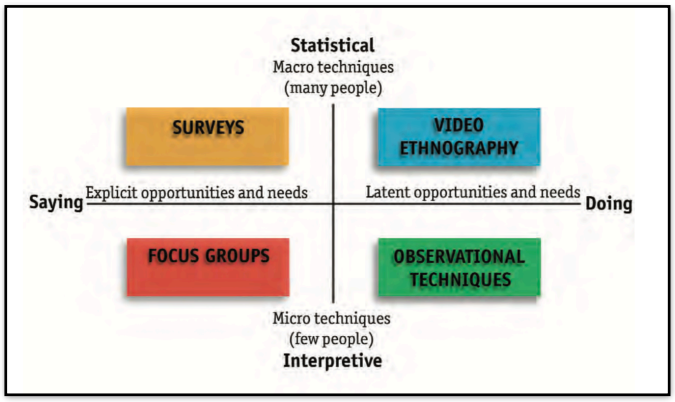


Fig.1 Design research methods (Moggridge, 2007)

Moggridge writes in *Designing Interactions* about several different methods for design research when prototyping. Figure 1 classifies these methods into macro and micro techniques, and techniques that result in the understanding of either explicit or latent needs of users. He explains that if your goal is innovative design, then understanding implicit actions and desires of users can be obtained through doing rather than saying (Moggridge 2007). This is the reason I decided to include observational techniques as well as video ethnography (for statistical analysis of gestures) in the user tests of gestur.a as "it is essential to the success of interaction design that designers find a way to **understand the perceptions, circumstances, habits, needs**, and **desires** of the ultimate users" (Suri, 2005). Gestur.a engages with Moggridge's hierarchy of design disciplines in order to understand the design constraints from a users perspective (see Appendix 1).

NEW MEDIA STUDIES

New media theorist Lev Manovich argues that cultural interfaces (human computer interfaces that portray cultural data: texts, images, video, music etc) such as my own gestur.a app, ascribe themselves with a new media language (Manovich, 2001). This language is "largely made up of other, already familiar cultural forms". This is the theory that new media forms take part in a mimesis of sorts - copying traditions and mechanisms of previous media forms, or, "remediation" (Bolter and Grusin, 2000).

Understanding how I am engaging with remediation when developing a digital instrument is important if the goal is trying to make meaningful interactions. Magnusson argues "instruments that are remediations of the **known** into the digital medium" are largely associated with the late 20th century (Magnusson, 2019). With the speed of technological evolution, contemporary efforts to understand digital organology and the design of meaningful digital instruments must therefore go beyond simple remediations.

> If the use of digital computers in the twenty-first century is beginning to express its unique character, it would be through their algorithmic nature, the ease of automation, dynamic mappability, and not least, the profound potential of machine learning." (Magnusson, 2019)

Joel Ryan, researcher and collaborator at STEIM (Studio for Electro Instrumental Music), writes in 1991 about the problems encountered in performing computer music. He argues that performers of computer music experience a large mediating distance between themselves and the instrument due to complex and unintuitive 'mappings' a lack of intuitive physicality. He proposes several solutions, one of which he calls "physical handles on phantom models":

> "The physicality of the performance interface helps give definition to the modelling process itself. The physical relation to a model **stimulates the imagination** and enables the elaboration of the model using spatial and physical metaphors. The image with which the artist works to realise his or her idea is no longer a phantom, it can be **touched**, **navigated** and **negotiated with**." - (Ryan, 1991)

Ryan recognises the possibility for gestures to have a positive effect on a 'phantom' (digital) musical model arguing that it allows for the "expansion of possibilities of communication with the model" (Ryan, 1991).

GESTUR.A

TECHNICAL SUMMARY

I originally had the idea of a gestural smart-phone based synthesiser in 2017 while engineering the GloveDuino, my glove-based gestural composition and performance tool built on Arduino and PureData. One of the main issues with this system was its dissemination - though it offered much more fine-grain control of musical parameters through flex sensors on the fingers of the user, the device was expensive and therefore practically non-replicable, not to mention that Imogen Heap's mi.mu gloves had already achieved a similar controller (Heap, 2012). Where a mobile app compromises on control, it makes up for in the fact that it allows the concept to be shared via the 'app' for much less than a physical instrument, it also has a speaker built in which gives the the ability to make it a fully contained digital instrument.

Gestur.a is 'written' in a combination of three languages. At the GUI and sensor level it is written in Swift 5 through Xcode, Apple's proprietary development environment for programming their devices. This gives me access to device motion sensors and the ability to craft a custom GUI. The LibPD framework interactions are written in Objective-C. This code allows mapping of Swift GUI objects and sensor activations to objects 'written' in the PureData language. I say 'written'

because PureData is a visual programming language, where code is assembled through the visual patching of objects to other objects, much like a modular synthesiser, which is a quaint lyricism in itself; considering the purpose of the code that I'm writing.

Fig.2 Gestur.a software architecture

Early development choices included the name 'gestur.a' - coming from the Latin "gesturā" a declension of the verb "to bear, carry, or wear" (Du Cange, 1883). I think this enforces the notion that the smartphone is truly taking on the form of an embodied and interactive instrument.

PROTOTYPING

Figure 3 shows an early prototype of gestur.a, it had no gesture control, but was my original attempt to learn how to use LibPd, Xcode, and PureData together. Users could use sliders on the screen to affect the pitch of two static musical notes that they could turn off with another button.

It was after this rudimentary prototype that I decided to form the research studies that would effect next stages of development. I knew that I wanted the device to emit sound when users made certain gestures whilst holding the device, what I needed to find out was what gestures would be comfortable (both physically and in terms of how a user would look doing it) and suitably expressive for the corresponding auditory feedback. I planned to have

Fig.3 Early prototype gestur.a and functioning PureData patch

2-3 weekly user tests, where I would video the test, and capture sensor logs from the device whilst in motion. This would allow me to set up suitable values for the sensitivity for triggered sound effects or musical notes. It would also allow me to ask participants about the intuitiveness of the system and the gestures, and query them on more gestures that they felt were comfortable and suited a variety of sounds.

The core logic of gestur.a at the current time of writing is as follows, where the lists show current possibilities of mappings and threshold operators:

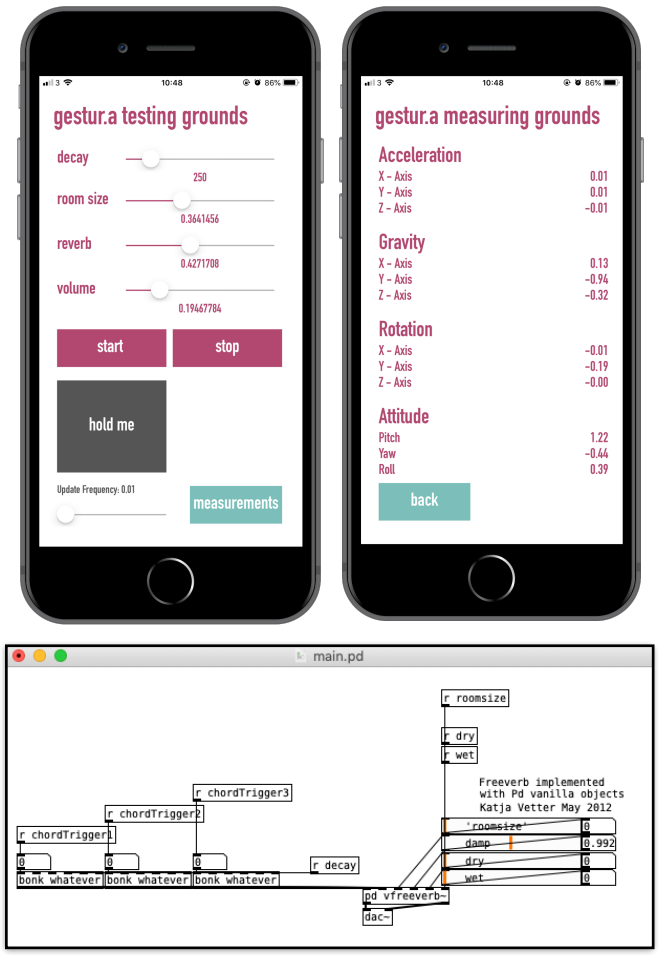


Fig.4 Current prototype with 2 views and Pd patch

This allows for fairly simple mappings. Figure 4 is the associated PureData patch and prototype app for user testing. This version of the prototype plays three chords that are dependent on motion gestures. The corresponding code looks something like this:

```
if (deviceMotion.rotationRate.x > 12)
        {
            patch?.chordTrigger(60,64,67)
        }
```

This code sends the notes 60, 64 and 67 (in MIDI note values this is C Major) to the patch which then plays a chord comprised of those notes if the user rotates the device past an angle at a force (the value of this threshold is 12).

'Gestur.a measuring grounds' (a secondary view that is accessed with the 'measurements' button on the main view) gives me access to real time values of all the device motion axes, this is helpful for rapidly prototyping threshold values to be more or less sensitive.

The capture motion button “hold me”, when held, prints a steady stream of values from all sensors to my developer console. From here I can paste the data into a template spreadsheet that I have made that automatically makes graphs from the data, which allows me to find trends and preferred threshold values in participant motions (Figure 5).

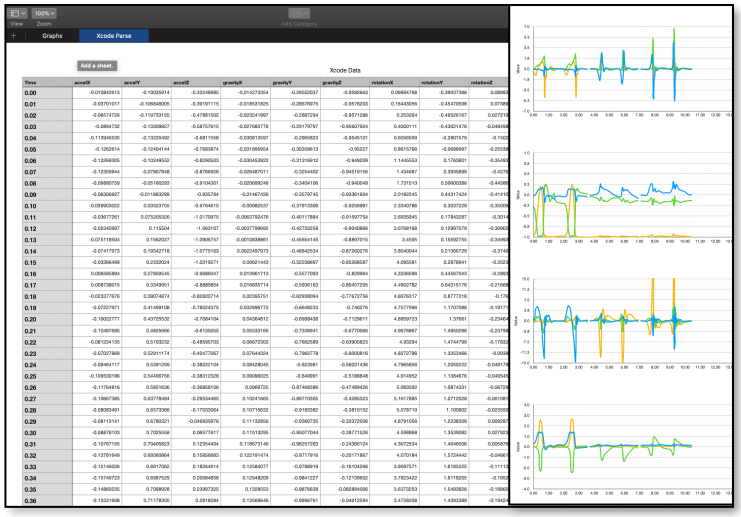


Fig.5 Testing graphs with participant motions

My user tests involve a 5 minute introduction to the system which includes a small briefing about the test. This is followed by allowing the user to explore the app’s sounds, gestures and parameters. I interview the participant about their experience during this time, about the UI, control scheme, and sound quality. After this, I asked the participant to record gestures whilst listening to some audio recordings of existing instruments. This data will be useful in expanding the voice variety of gestur.a. I am however aware of the danger of simply remediating existing instruments.

After my first round of user tests, I already had valuable feedback on how to improve the interface, control system, and sounds of gestur.a. Figure 6 shows some of the testing ground UI changes.

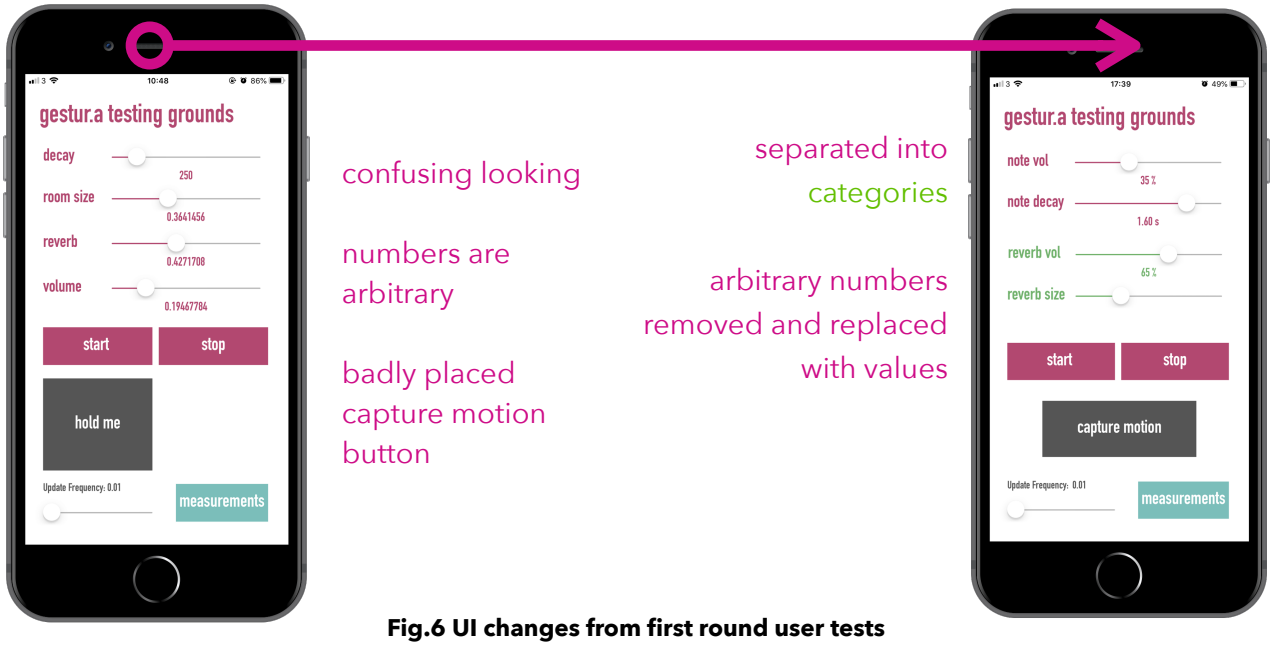


Fig.6 UI changes from first round user tests

The Pd patch underwent changes too, users can now let notes ring for up to 60 seconds, allowing them to create longer and richer soundscapes. I added a second oscillator to the system meaning that the chord’s voice is no longer a simple sine tone, but a blend between a sine and sawtooth wave, this makes the sound more harmonically rich and slightly distorted. With the

addition of the sawtooth wave (which can sound quite raspy at higher frequencies) I decided to add a filter (this effectively turns down the volume of frequencies above a certain threshold frequency called “cutoff”). I assigned this in real-time to the iPhone’s roll value. I show this in more depth in the video demonstration attached. This motion adds another facet of user control - a feature that was heavily requested during the user-tests. Figure 7 shows the new Pd patch with the filter highlighted (compare to Figure 4 patch).

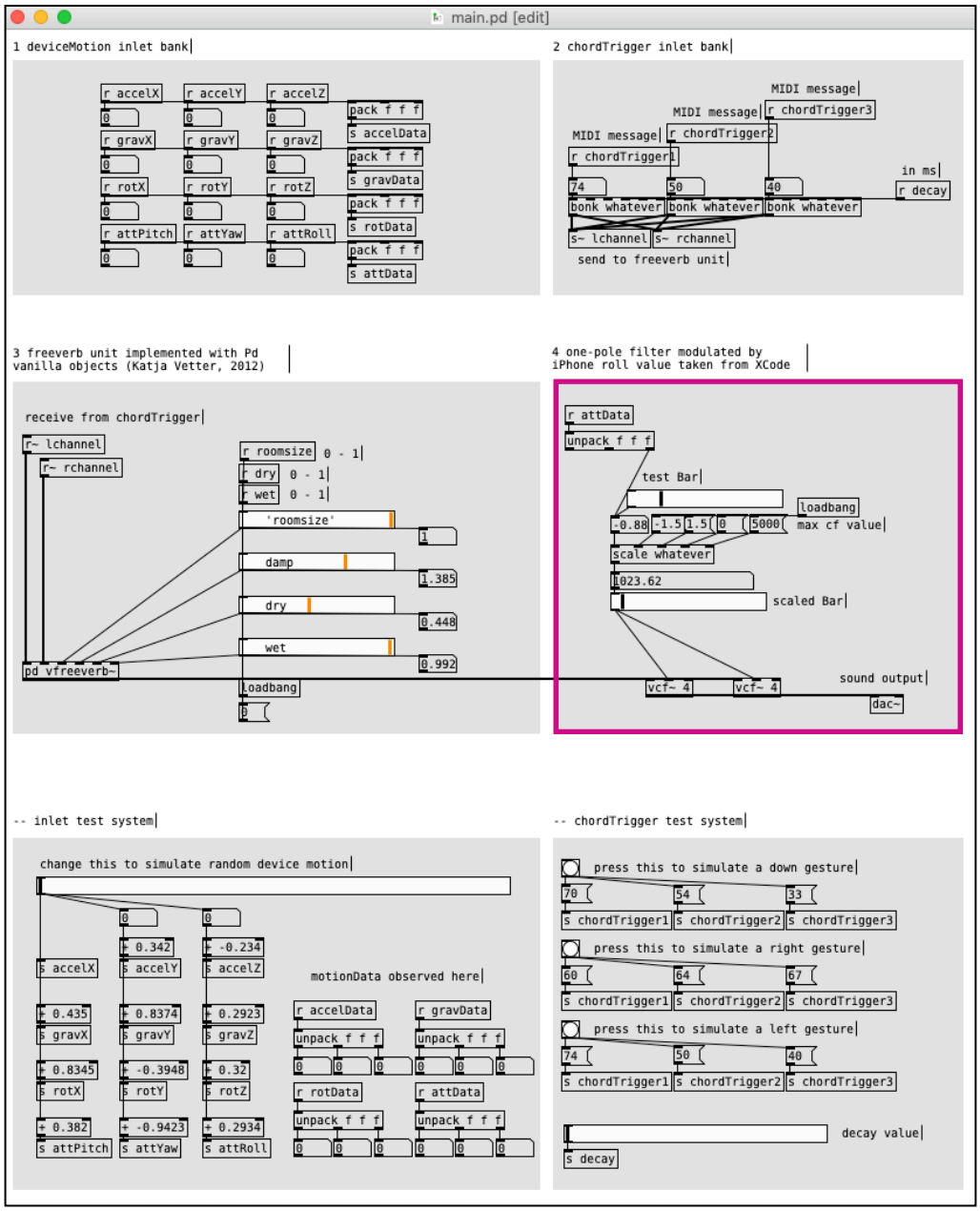


Fig.7 Revamped patch from first round user tests

FUNDING AND THE FUTURE OF GESTUR.A

I'm currently investigating ways of giving pitch control to the user, so that they can change the chords that are played through gesture. I plan on releasing a Kickstarter campaign for gestur.a later this year to fund research and development of the project in three key areas. The reason I've chosen kickstarter is because on my stakeholder map, they are the funding body that would allow me to keep full control of the development, but also engage with funders who would be the most interested in the development (see Figure 8). I plan for gestur.a to be released in mid 2020.

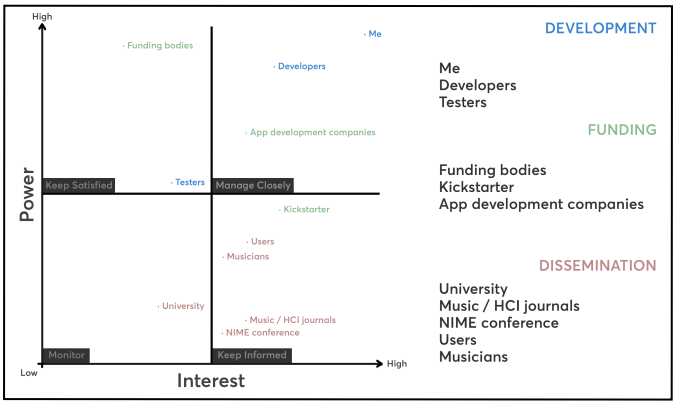


Fig.8 Stakeholder map for gestur.a

Funding Area: Machine Learning

The addition of a machine learning component to gestur.a would allow users to map parameters to gestures that they themselves could train the app into recognising. This would shift the musical/control agency towards the composer, and away from myself, the system designer - allowing them to be more expressive of their own musical ideas.

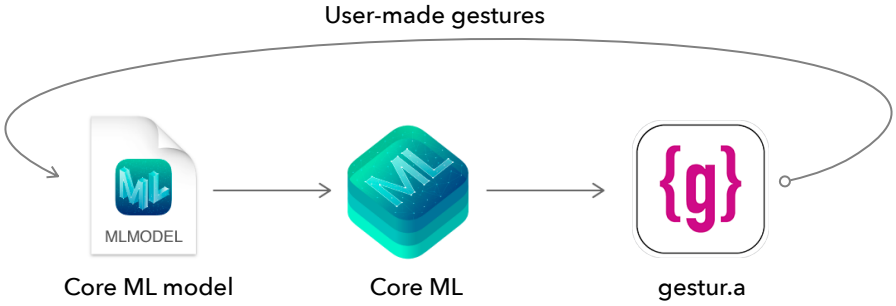


Fig.9 Training gestur.a CoreML models

Funding Area: Sharing System

I have started thinking about a sharing system for gestur.a - a platform on the app for users to share and rate other's compositions. Users could export their live compositions in an infographic style format which would include a selfie-cam video of themselves performing and a 3D representation of their deviceMotion values as a form of graphical music score. In envisage this becoming popular if it is implemented properly due to the comedic selfie-camera view while the user is making their gestural music.

Funding Area: Instrument Research

I would like to publish findings from further user studies in the form of journal papers and by attending conferences such as NIME (New Instruments for Musical Expression) and TEI (Tangled, embedded, and embodied interaction) to disseminate this research. I have already started coding interview responses using the grounded theory method for qualitative data analysis (see Fig. 10) and have, through this, highlighted key areas of research including digital aesthetics, feedback, and materiality.

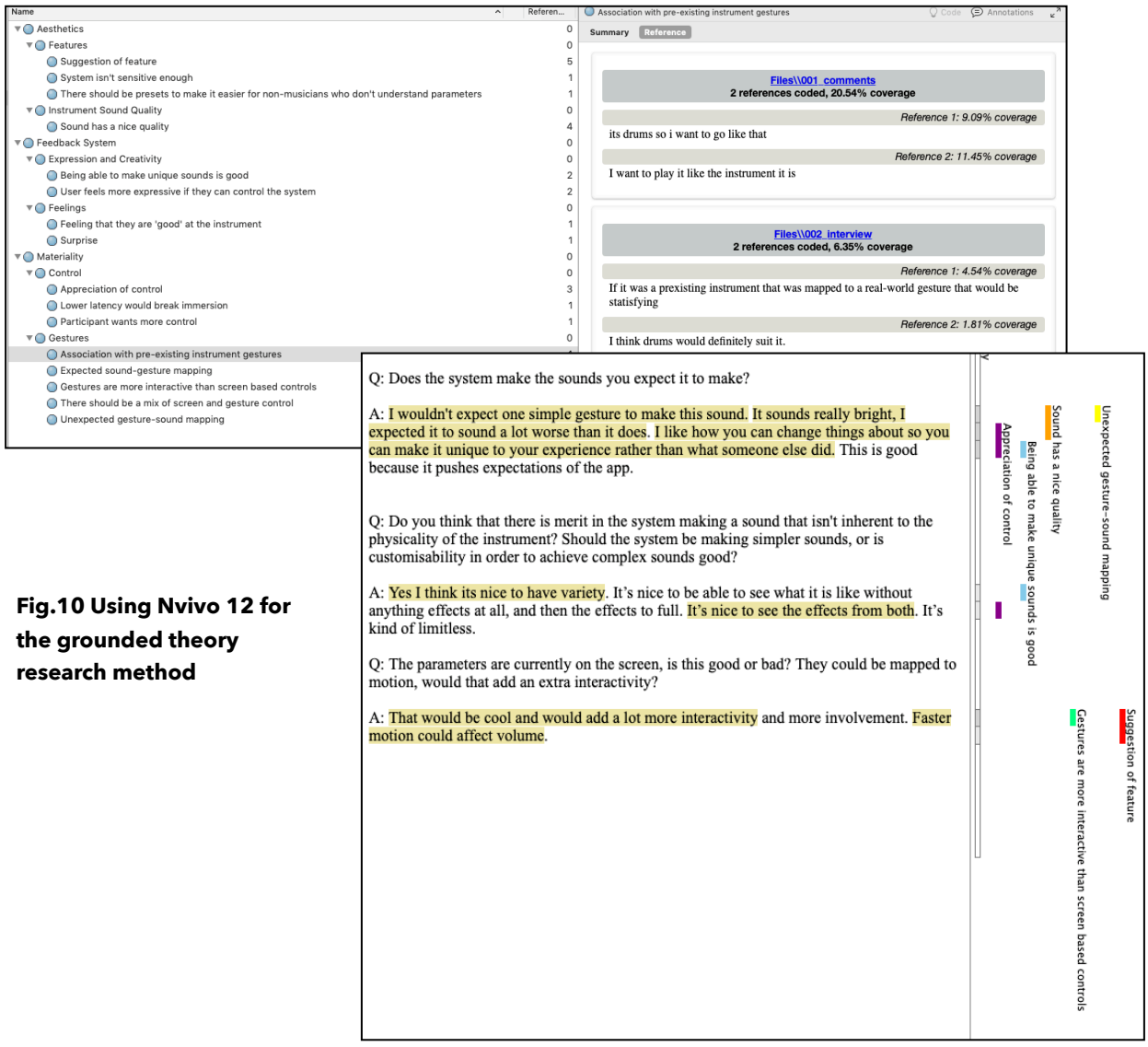


Fig.10 Using Nvivo 12 for the grounded theory research method

BIBLIOGRAPHY

Bolter, J. and Grusin, R. (2000). *Remediation*. Cambridge, Mass: MIT Press.

Du Cange, C. (1883). *Glossarium mediae et infimae Latinitatis*.

Heap, I. (2012). *Mi.Mu Gloves*. [online] MI·MU. Available at: https://mimugloves.com [Accessed 23 Oct. 2017].

Magnusson, T. (2019). *Sonic Writing*. Bloomsbury.

Manovich, L. (2001). *The Language of New Media*. Cambridge: MIT Press.

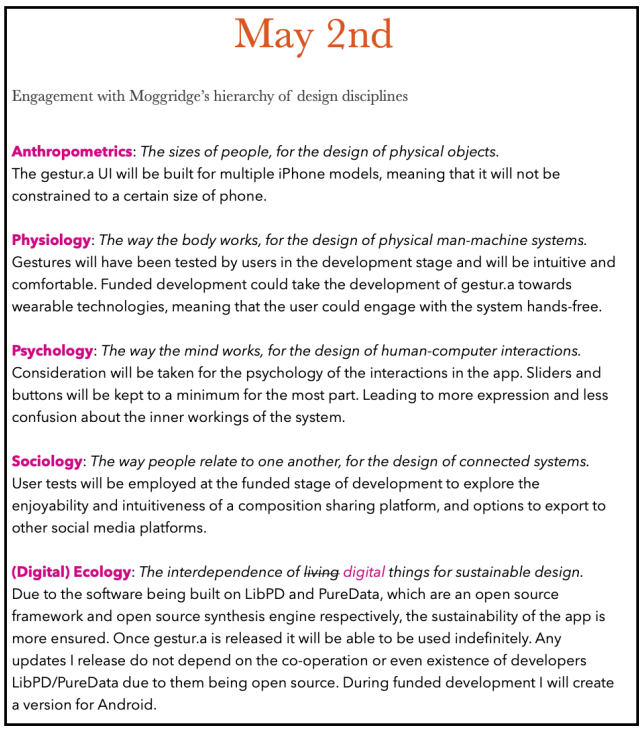
Moggridge, B. (2007). *Designing Interactions*. Cambridge, Mass.: MIT Press.

Ryan, J. (1991). *Some remarks on musical instrument design at STEIM*.

Suri, J. (2005). *Thoughtless Acts?*. San Francisco, Calif: Chronicle Books.

APPENDIX

1. HIERARCHY OF DESIGN DISCIPLINES



May 2nd

Engagement with Moggridge's hierarchy of design disciplines

Anthropometrics: *The sizes of people, for the design of physical objects.*
The gestur.a UI will be built for multiple iPhone models, meaning that it will not be constrained to a certain size of phone.

Physiology: *The way the body works, for the design of physical man-machine systems.*
Gestures will have been tested by users in the development stage and will be intuitive and comfortable. Funded development could take the development of gestur.a towards wearable technologies, meaning that the user could engage with the system hands-free.

Psychology: *The way the mind works, for the design of human-computer interactions.*
Consideration will be taken for the psychology of the interactions in the app. Sliders and buttons will be kept to a minimum for the most part. Leading to more expression and less confusion about the inner workings of the system.

Sociology: *The way people relate to one another, for the design of connected systems.*
User tests will be employed at the funded stage of development to explore the enjoyability and intuitiveness of a composition sharing platform, and options to export to other social media platforms.

(Digital) Ecology: *The interdependence of ~~living~~ digital things for sustainable design.*
Due to the software being built on LibPD and PureData, which are an open source framework and open source synthesis engine respectively, the sustainability of the app is more ensured. Once gestur.a is released it will be able to be used indefinitely. Any updates I release do not depend on the co-operation or even existence of developers LibPD/PureData due to them being open source. During funded development I will create a version for Android.