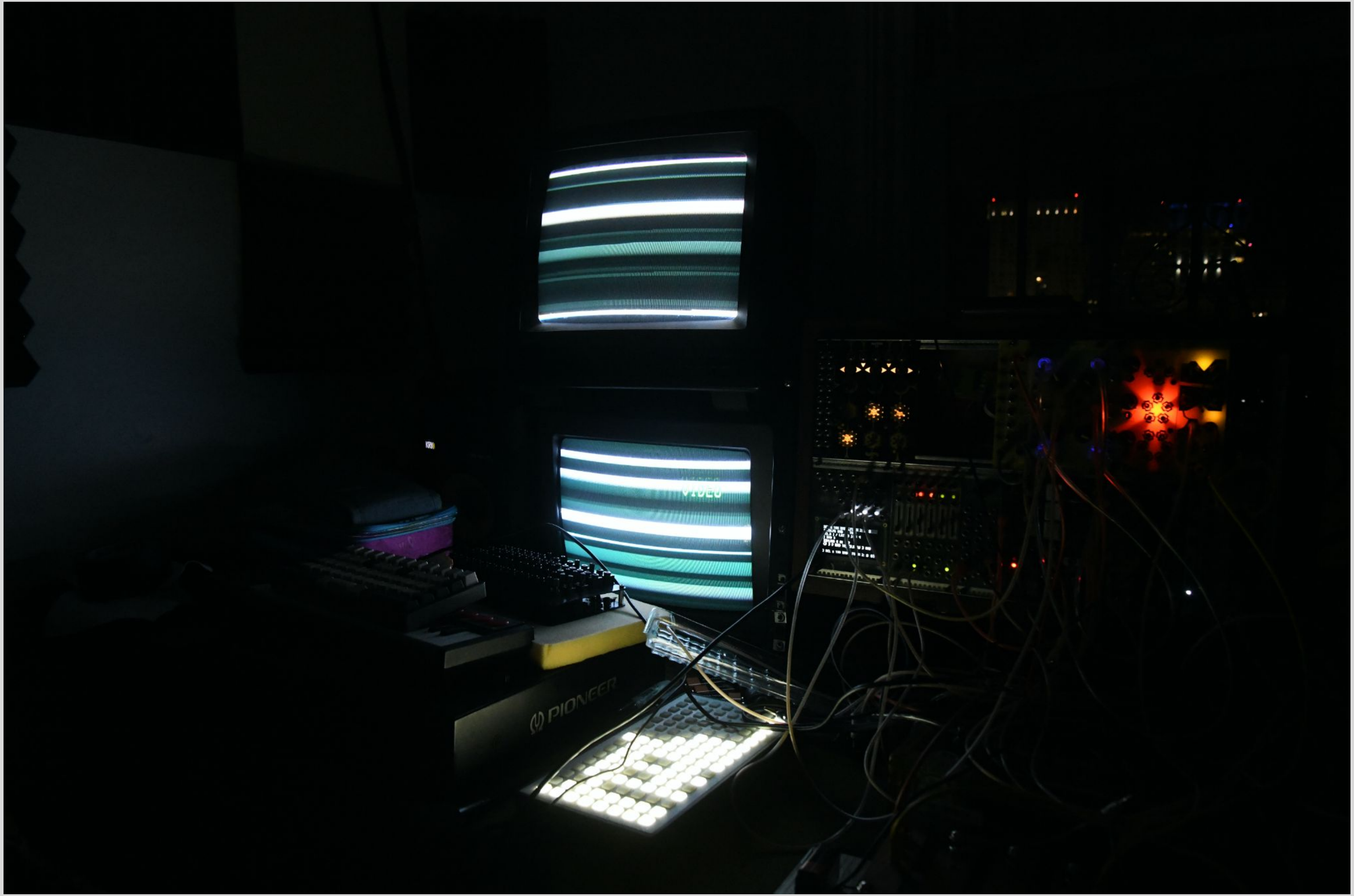
According to chance final proposal

Profile

I want to focus on the idea of chance in profile and the idea of chance being embedded in a being, machine or environment.

I want to explore ways of interfacing the inevitable chance based human system (body) with an artificial machine or synthesised medium, specifically using sound and accompanying visuals.

The idea of profile being a human == machine profile where the interaction is what makes a new personality or profile log.

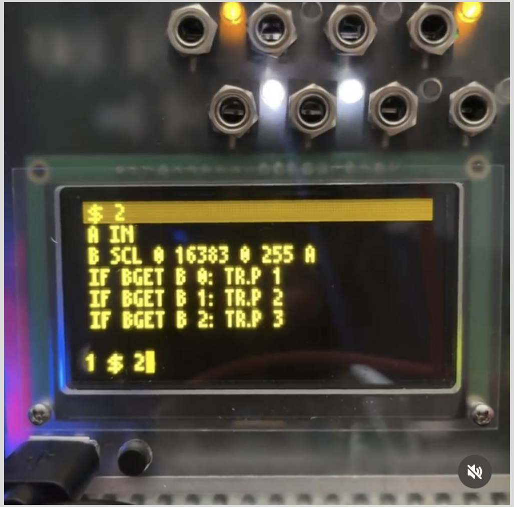

VIDEO
PIONEER

Binary

There are a few ideas about binary code I like to apply to my making of chance based audio compositions and performances that are both literal and metaphorical.

One of them is that binary code is only either a 1 or a 0, which in sense is a coin flip.
The next is the idea of “rotating left shift” or “rotating right shift”, where a binary number can change its personality by shifting in a cyclical motion. 101100 will become 011001 and then 110010 etc until it loops around to become the same number again. This is especially interesting to me because it speaks on feedback loops and interactivity.

I usually use binary in my audio work as a sequence of events, derived from a single input, because single whole digit numbers can be represented by a set of ones and zeros, whose information I can sequence through. It's an easy way to dig into a value, be it a voltage source on my modular synth, and expand it to apply elsewhere, and influence itself.

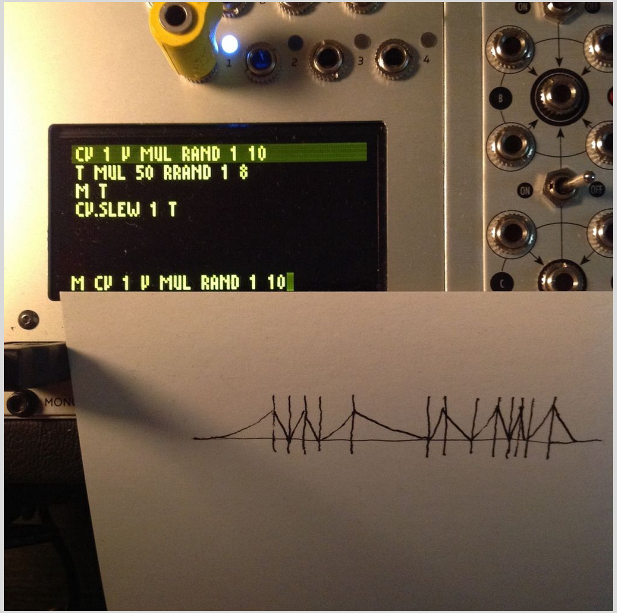


```
$ 2
A IN
B SCL 0 16383 0 255 A
IF BGET B 0: TR.P 1
IF BGET B 1: TR.P 2
IF BGET B 2: TR.P 3

1 $ 2
```

On Generative music, Monome Teletype and live coding environments

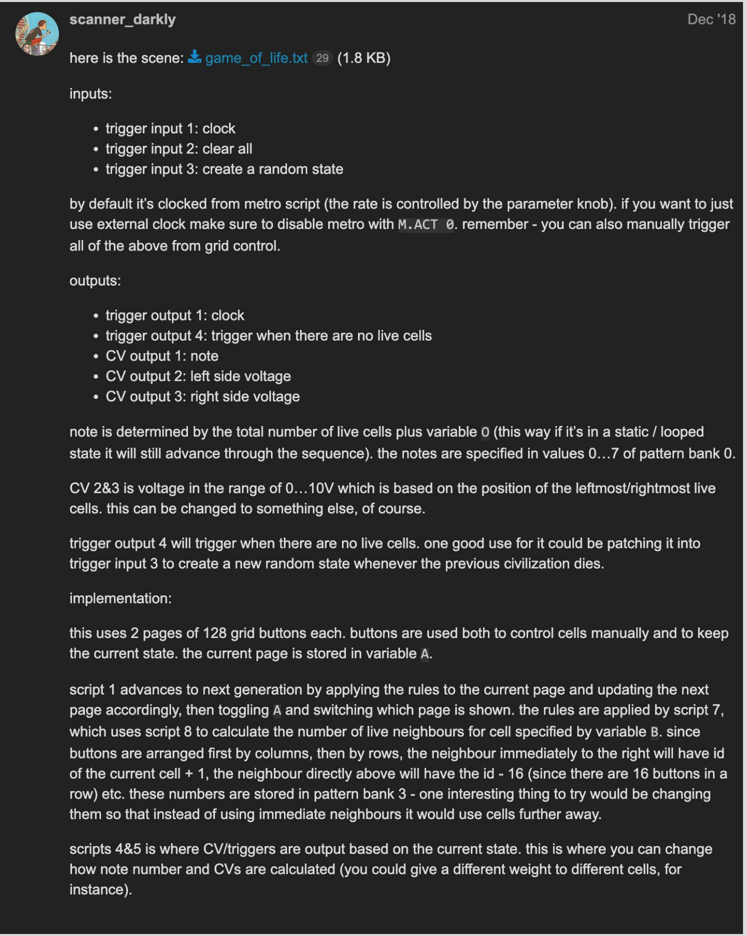
There is a module installed on my modular synthesizer called Teletype. Teletype uses its own syntax (language) that allows you to create 8 lines of code on 8 separate scripts that can run on their own, be told to run by any or all of the 8 inputs on the module, or told to run by other scripts on the module. Teletype outputs voltage in the form of 4 short/long pulses, toggles, triggers or gates and 4 voltage outputs that can be used in various ways to sent triggers to drums or synths, open and close filters, control effects ect.

I like this language because it’s a medium that allows for complexity and creative computation within an environment that may be only intended for music production and sound design. Implementing creative means of computation within this environment has allowed me to introduce controlled/monitored chance in my mostly analog sound design setup.



```
CV 1 V MUL RAND 1 10
T MUL 50 RRAND 1 8
M T
CV.SLEW 1 T


M CV 1 V MUL RAND 1 10
```

Cellular Automata



scanner_darkly Dec '18

here is the scene: game_of_life.txt 29 (1.8 KB)

inputs:

- trigger input 1: clock
- trigger input 2: clear all
- trigger input 3: create a random state

by default it's clocked from metro script (the rate is controlled by the parameter knob). if you want to just use external clock make sure to disable metro with `M.ACT 0`. remember - you can also manually trigger all of the above from grid control.

outputs:

- trigger output 1: clock
- trigger output 4: trigger when there are no live cells
- CV output 1: note
- CV output 2: left side voltage
- CV output 3: right side voltage

note is determined by the total number of live cells plus variable `O` (this way if it's in a static / looped state it will still advance through the sequence). the notes are specified in values 0…7 of pattern bank 0.

CV 2&3 is voltage in the range of 0…10V which is based on the position of the leftmost/rightmost live cells. this can be changed to something else, of course.

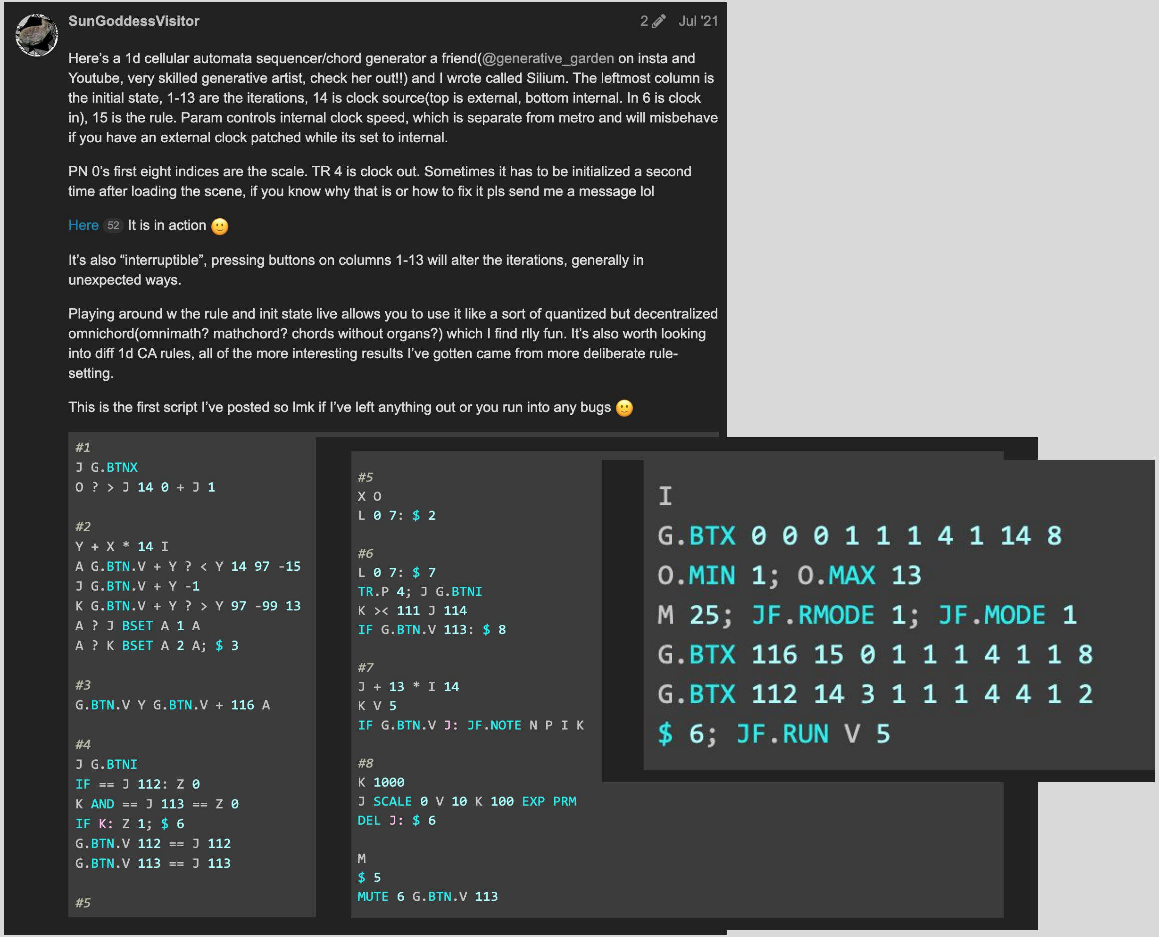
trigger output 4 will trigger when there are no live cells. one good use for it could be patching it into trigger input 3 to create a new random state whenever the previous civilization dies.

implementation:

this uses 2 pages of 128 grid buttons each. buttons are used both to control cells manually and to keep the current state. the current page is stored in variable `A`.

script 1 advances to next generation by applying the rules to the current page and updating the next page accordingly, then toggling `A` and switching which page is shown. the rules are applied by script 7, which uses script 8 to calculate the number of live neighbours for cell specified by variable `B`. since buttons are arranged first by columns, then by rows, the neighbour immediately to the right will have id of the current cell + 1, the neighbour directly above will have the id - 16 (since there are 16 buttons in a row) etc. these numbers are stored in pattern bank 3 - one interesting thing to try would be changing them so that instead of using immediate neighbours it would use cells further away.

scripts 4&5 is where CV/triggers are output based on the current state. this is where you can change how note number and CVs are calculated (you could give a different weight to different cells, for instance).



SunGoddessVisitor 2 Jul '21

Here's a 1d cellular automata sequencer/chord generator a friend(@generative_garden on insta and Youtube, very skilled generative artist, check her out!!) and I wrote called Silium. The leftmost column is the initial state, 1-13 are the iterations, 14 is clock source(top is external, bottom internal. In 6 is clock in), 15 is the rule. Param controls internal clock speed, which is separate from metro and will misbehave if you have an external clock patched while its set to internal.

PN 0's first eight indices are the scale. TR 4 is clock out. Sometimes it has to be initialized a second time after loading the scene, if you know why that is or how to fix it pls send me a message lol

Here 52 It is in action 🙂

It's also "interruptible", pressing buttons on columns 1-13 will alter the iterations, generally in unexpected ways.

Playing around w the rule and init state live allows you to use it like a sort of quantized but decentralized omnichord(omnimath? mathchord? chords without organs?) which I find rlly fun. It's also worth looking into diff 1d CA rules, all of the more interesting results I've gotten came from more deliberate rule-setting.

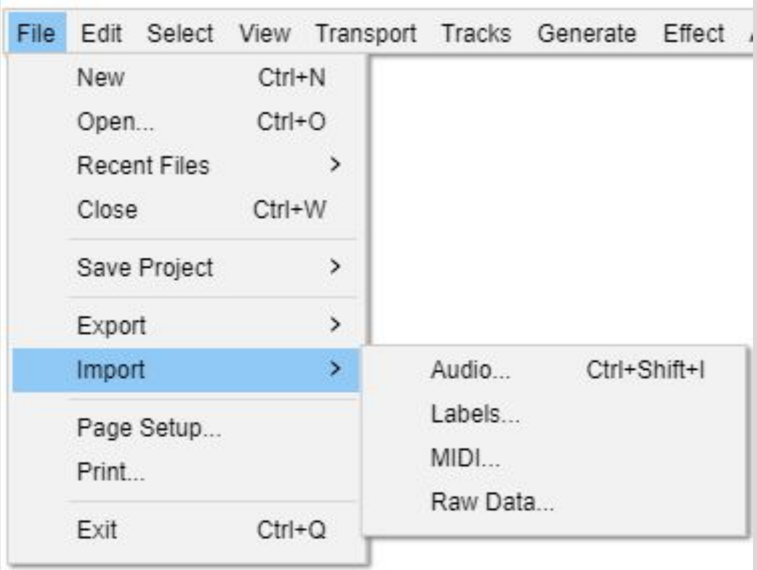
This is the first script I've posted so lmk if I've left anything out or you run into any bugs 🙂

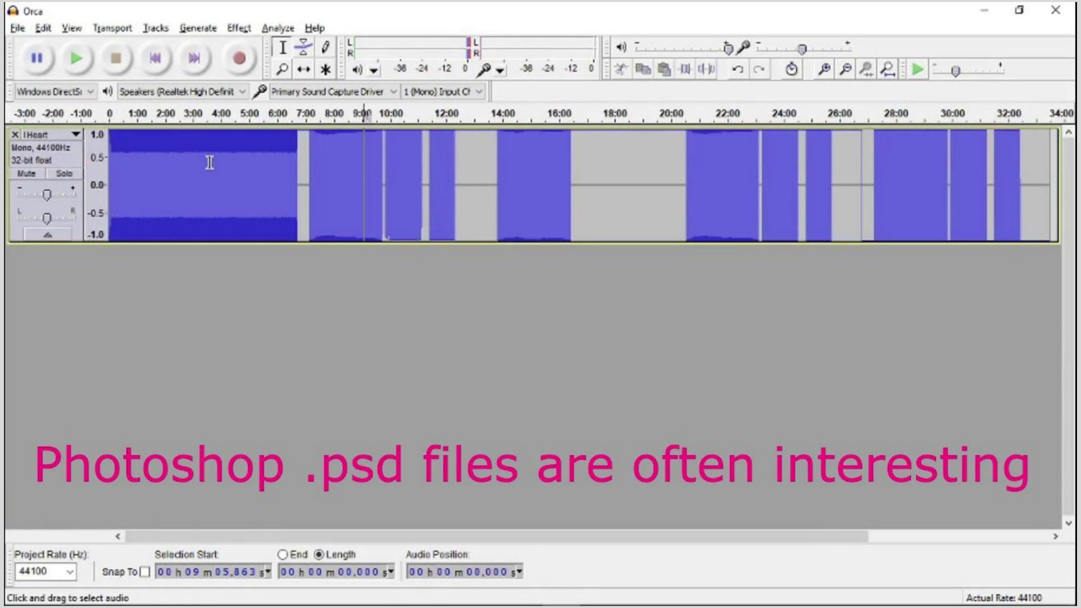
```
#1
J G.BTNX
O ? > J 14 0 + J 1

#2
Y + X * 14 I
A G.BTN.V + Y ? < Y 14 97 -15
J G.BTN.V + Y -1
K G.BTN.V + Y ? > Y 97 -99 13
A ? J BSET A 1 A
A ? K BSET A 2 A; $ 3

#3
G.BTN.V Y G.BTN.V + 116 A

#4
J G.BTNI
IF == J 112: Z 0
K AND == J 113 == Z 0
IF K: Z 1; $ 6
G.BTN.V 112 == J 112
G.BTN.V 113 == J 113

#5
```

```
#5
X O
L 0 7: $ 2

#6
L 0 7: $ 7
TR.P 4; J G.BTNI
K >< 111 J 114
IF G.BTN.V 113: $ 8

#7
J + 13 * I 14
K V 5
IF G.BTN.V J: JF.NOTE N P I K

#8
K 1000
J SCALE 0 V 10 K 100 EXP PRM
DEL J: $ 6

M
$ 5
MUTE 6 G.BTN.V 113
```

```
I
G.BTX 0 0 0 1 1 1 4 1 14 8
O.MIN 1; O.MAX 13
M 25; JF.RMODE 1; JF.MODE 1
G.BTX 116 15 0 1 1 1 4 1 1 8
G.BTX 112 14 3 1 1 1 4 4 1 2
$ 6; JF.RUN V 5
```

Sonic Data





Photoshop .psd files are often interesting

Project Rate (Hz): 44100

Snap To □

Selection Start: 00 h 09 m 05.863 s

○ End ● Length: 00 h 00 m 00.000 s

Audio Position: 00 h 00 m 00.000 s

Click and drag to select audio

Actual Rate: 44100

Sonic Data

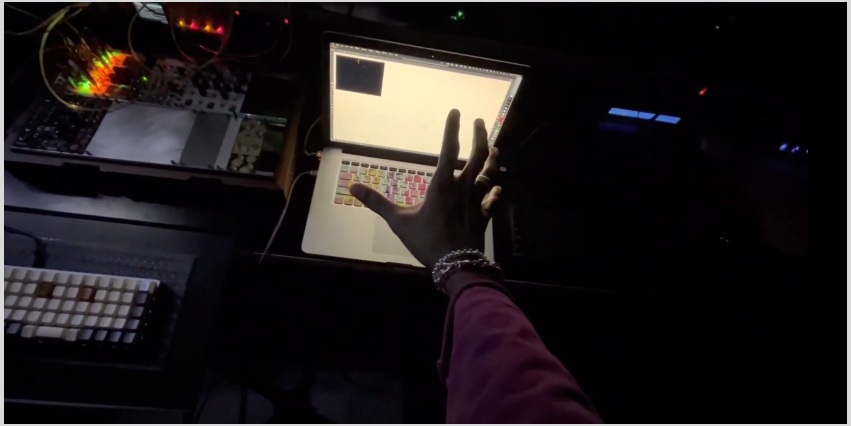
Alva Noto
Ryoji Ikeda
Oval
Aoki Takamasa

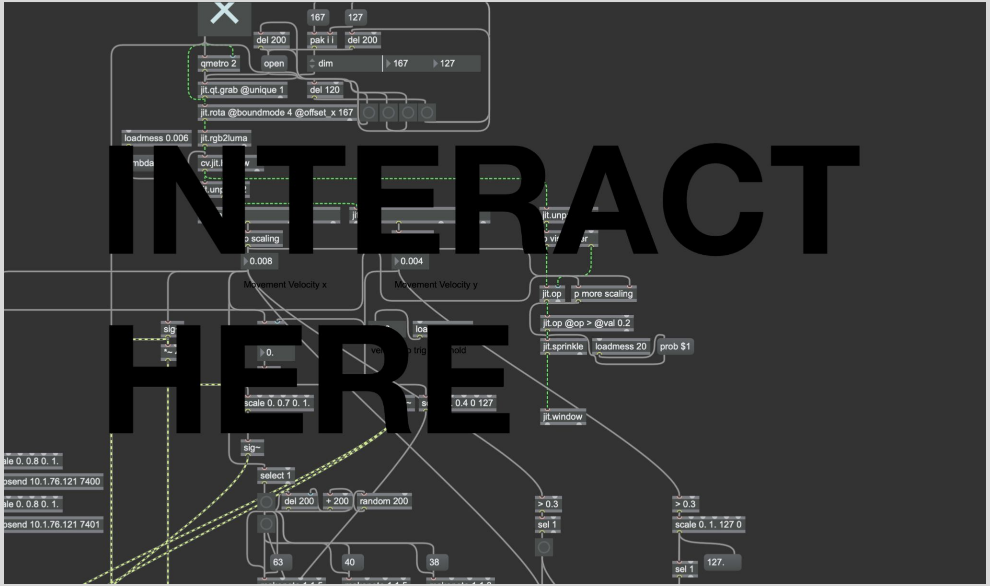
On granular synthesis and Minimalism

White noise and sine waves
Digital/Analog noise
Ground noise
Sine waves, Audio/Visual - Scan Lines/Scan

Interactivity

Some methods of interface and interactivity im looking at are motion sensors, velocity sensing and touch capacitance.





Interactivity

Some methods of interface and interactivity im looking at are motion sensors, velocity sensing and touch capacitance.