



GENERAL

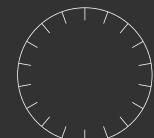
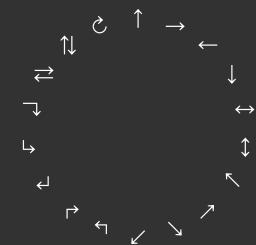
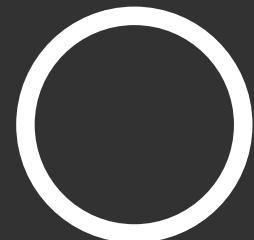
133



9 5

4 1

2 6



MARTIN LEZHENIN

SVETOUSTANOVKA ПОСТАНОВКА ПОРТАТИВНЫХ

OWNER'S MANUAL

BRTITANSKAYA
VYSHAYA SHKOLA
DIZAYNA

MOSKVA 2020

TABLE OF CONTENTS

1 Project background

- 1.1 General concept
- 1.2 Inspiration
- 1.3 Light, color, music
- 1.4 Idea development
- 1.5 Proof of concept

2 Algortihm

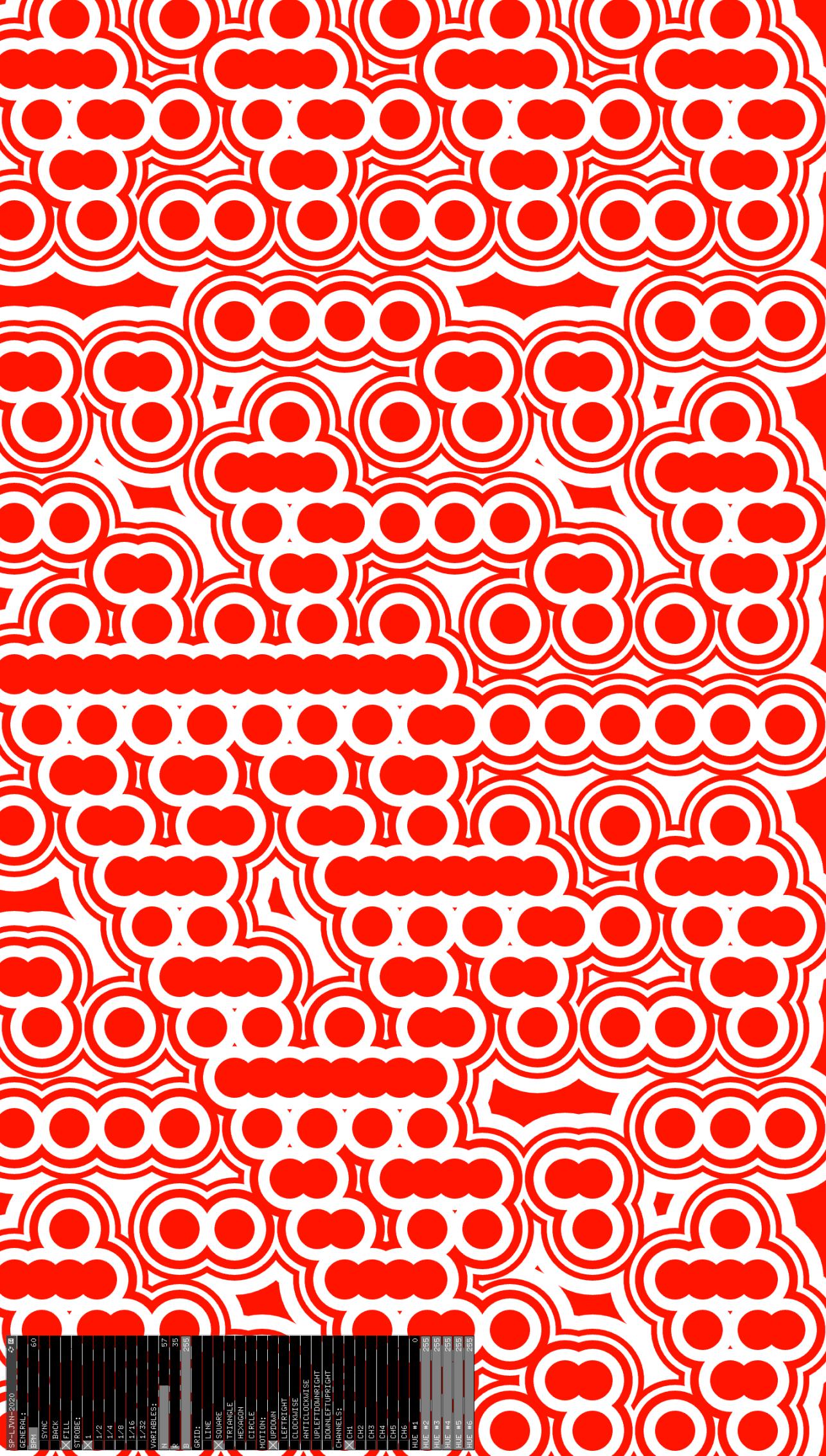
- 2.1 Code
- 2.2 BPM
- 2.3 SYNC
- 2.4 BACK
- 2.5 OUTLINE
- 2.6 STROBE
- 2.7 VARIABLES
- 2.8 MOTION
- 2.9 GRID
- 2.10 COLOR
- 2.11 Graphical output

3 Controller

- 3.1 Layout
- 3.2 Visual language
- 3.3 Modeling
- 3.4 Renders Research
- 3.5 Final renders
- 3.6 iPhone prototypes

4 Product

- 4.1 Presentation
- 4.2. Branding
- 4.3 Packaging
- 4.4 Quick tour
- 4.5 Advertising



SP-LAWN-2020
GENERAL:
SPM1 60
SYNC
BLOCK
FILL
STROBE:
 1
 1/2
 1/4
 1/8
 1/16
 1/32
VARIABLES:
 N 57
 R 35
 B 255
GRID:
 LINE
 SQUARE
 TRIANGLE
 HEPTAGON
 CIRCLE
MOTION:
 UPDOWN
 LEFTRIGHT
 CLOCKWISE
 ANTICLOCKWISE
 UPLEFTDOWNRIGHT
 DOWNLEFTUPRIGHT
CHANNELS:
 CH1
 CH2
 CH3
 CH4
 CH5
 CH6
HUE #1 0
HUE #2 255
HUE #3 255
HUE #4 255
HUE #5 255
HUE #6 255

Part I

Project Background

1.1 General concept

I've been cultivating the idea of my final project since the very days of Foundation. I had a desire to produce the thing which connects light and music on a completely different level and understanding. I cannot really say why it is like that, but every time I listen to the music, it is not the usual dreams of me singing and performing which come to my mind, but rather the ideas of the stage sets, lots of lightning equipment, camera moves, and smoke falling from the scene. Every single song produces the images of various complex environment, with different fixtures and solutions working towards one goal – synchronise the sound with the light.

However, I did not want to make yet another music visualisation instrument. Starting from the idea of fully automated algorithm based programm, I wanted my final outcome to be a tool on its own, creating a new understanding of the relationship between these two fields, create a notion of a new skill which requires learning and adapting to the possible instrumentarium, rather than simply allowing the machine to do all its thing.

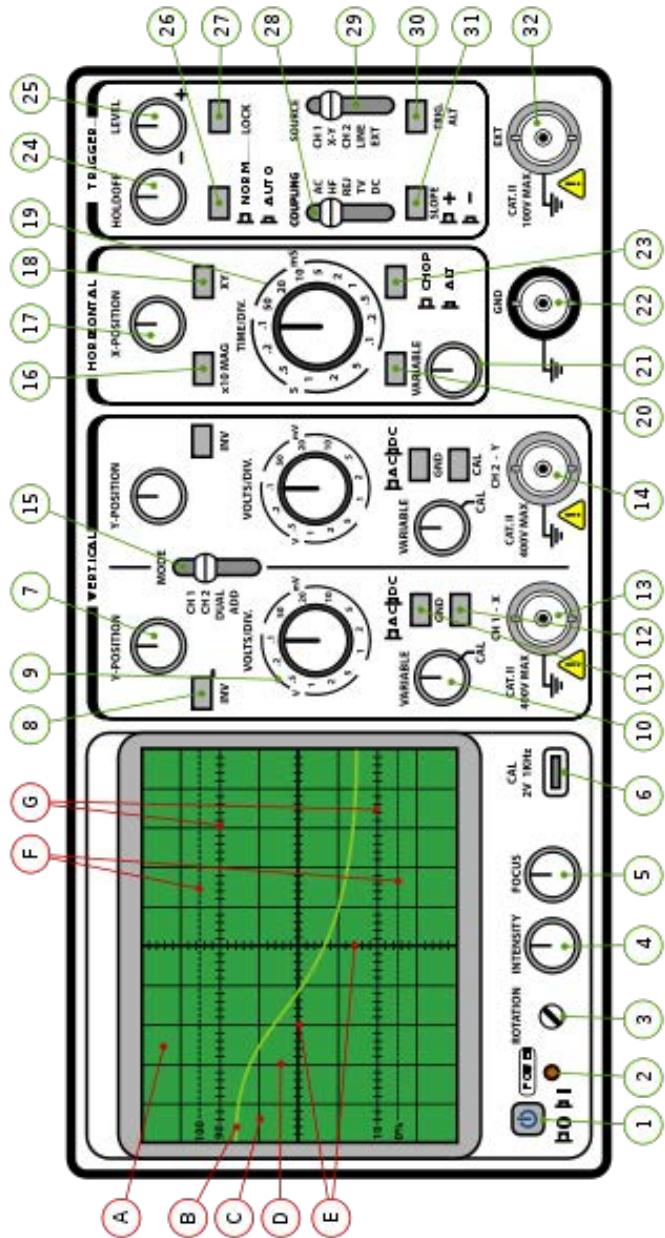
Thinking about the current way of things, there is a huge gap between cheap chinese music reactive lightning 'toys' which match loosely to the beat, and very elaborate stage panels, which require a professional crew to run and produce tangible outcomes. There are some video synthesizers available on the market, however, they are quite limited in their functionality and very expensive as well.

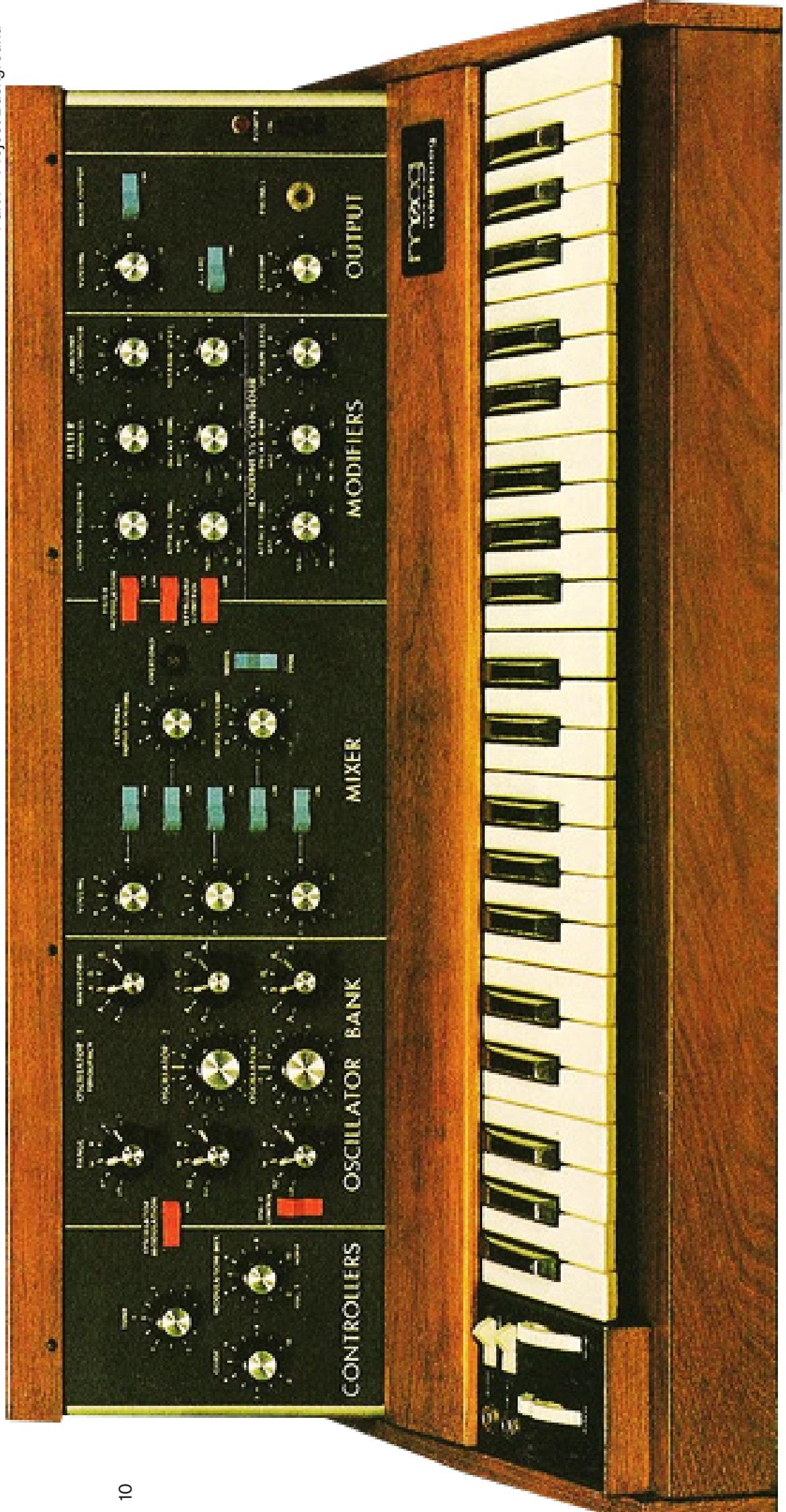
I did not want to create an ultimate product, following the rules and regulations of the consumer market, however, it felt very important to me to find balance between having fun and messing around, and making sure I'm creating something which will not stay on paper and can actually be produced at a reasonable degree for the outer users.

1.2 Inspiration

Visually speaking, I was always fascinated by the complexity of retro devices, mainly computers and music synthesisers. In modern times, when design of these things usually tries to be as minimal and sleek as possible, following the logic of ascetic functionalism, I wanted to rediscover the beauty of elaborate graphics, providing rich visuals yet remain contemporary.

Obviously enough, the starting point of my research was music related equipment, nevertheless, I did not want my project to mimic to the already existing solutions. The most surprising and inspirational reference for my project were oscilloscopes – a type of electronic test instrument that graphically displays varying signal voltages. I felt that this was a good starting point for future experiments not directly related to the field my project lies in, allowing for a more open and wide look.





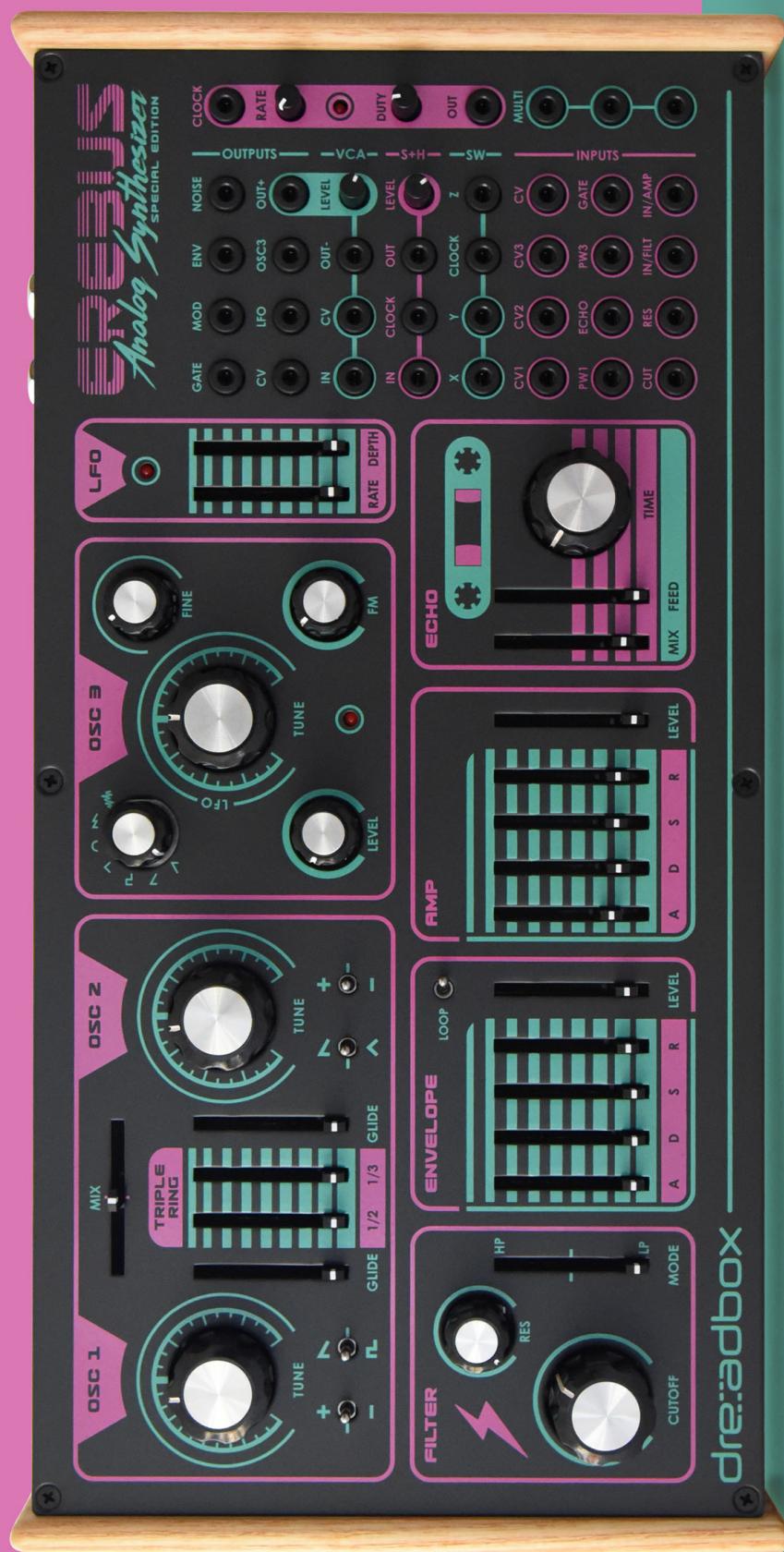


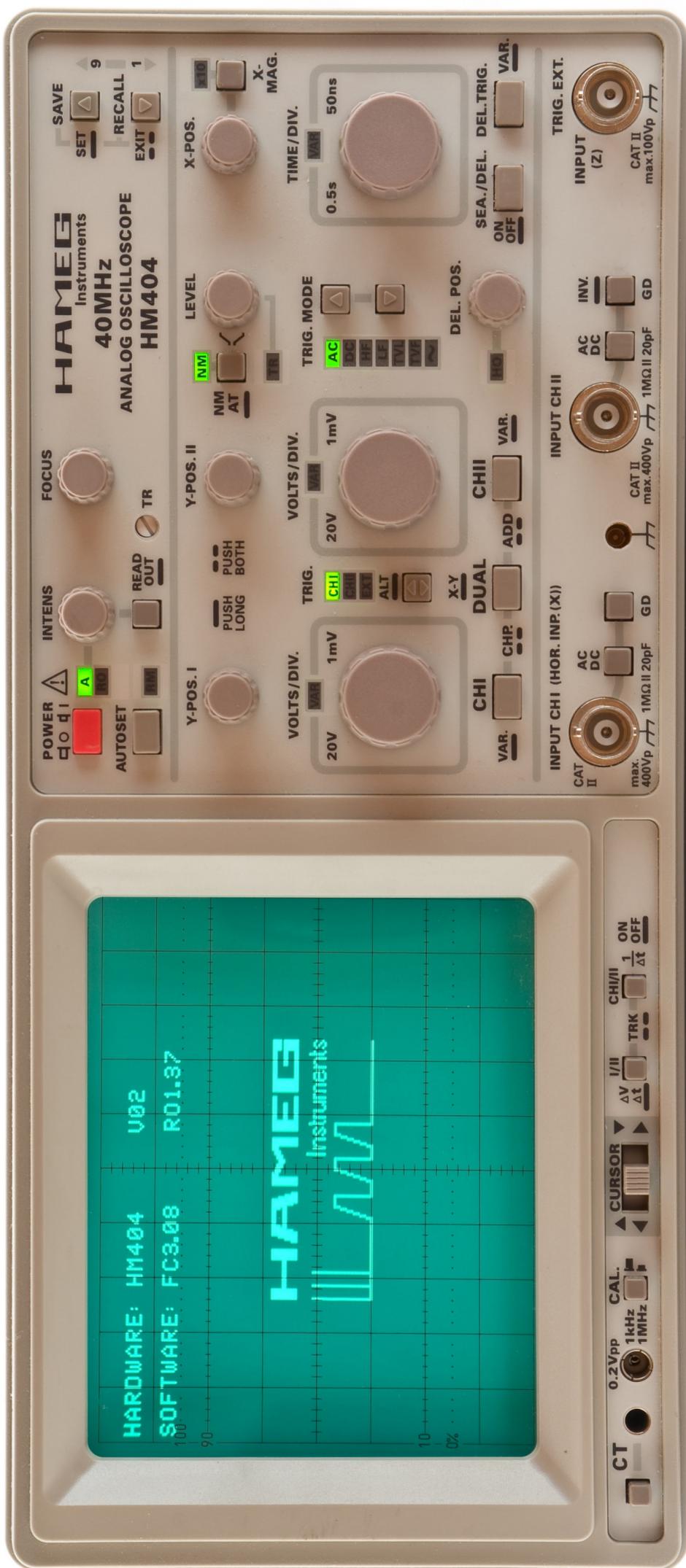












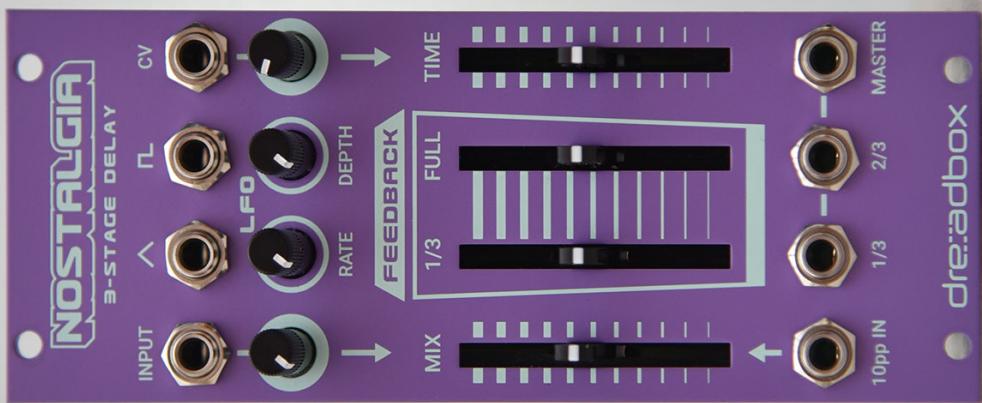




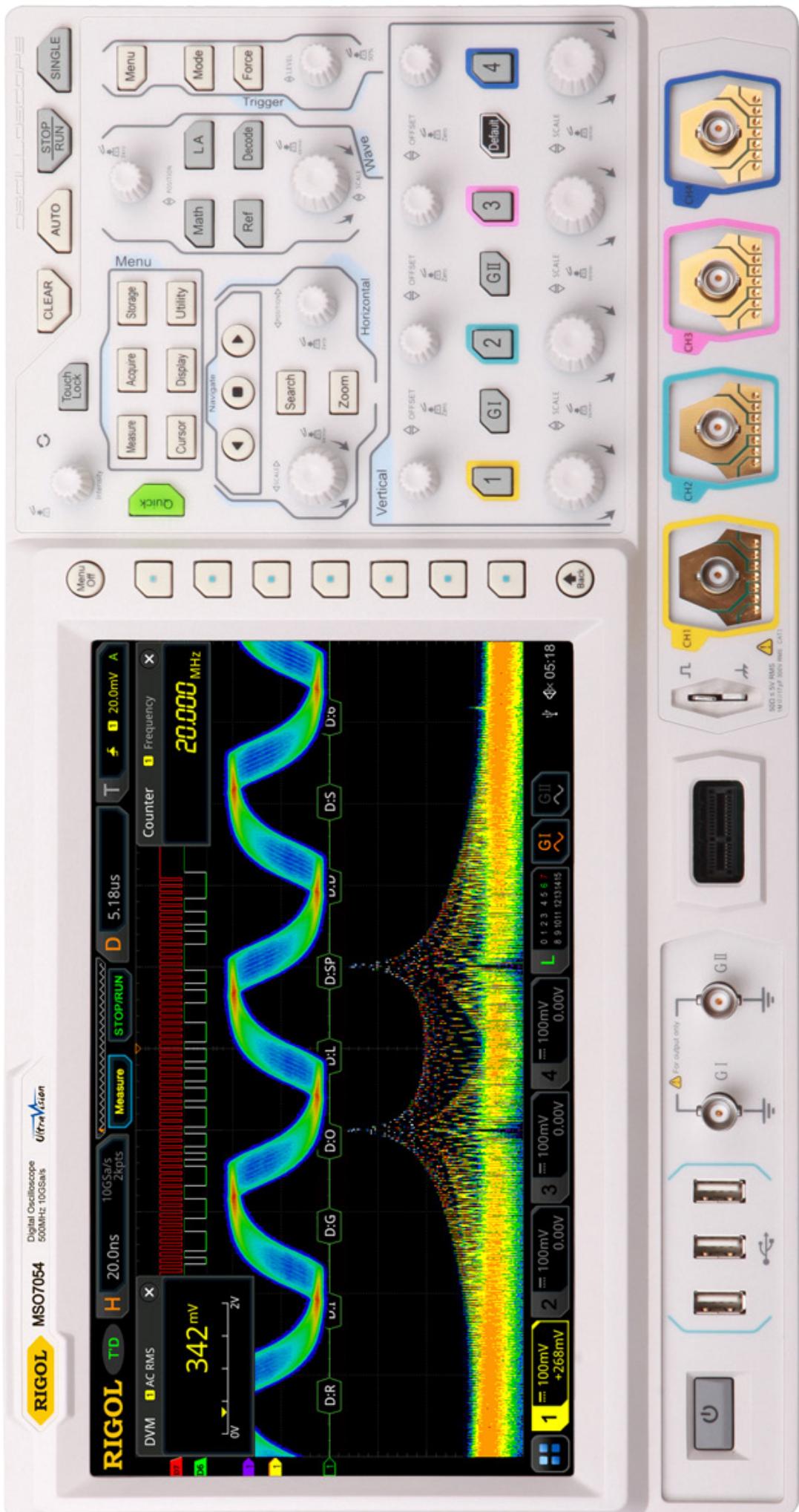


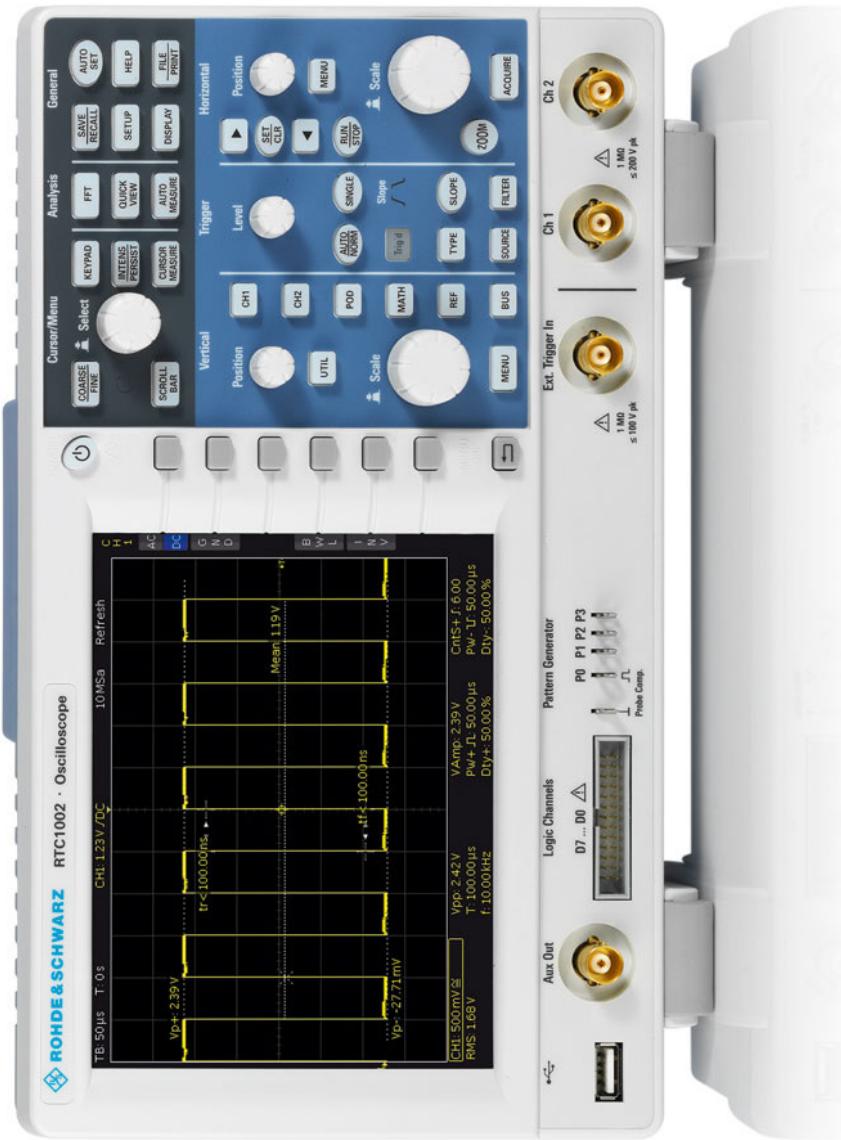






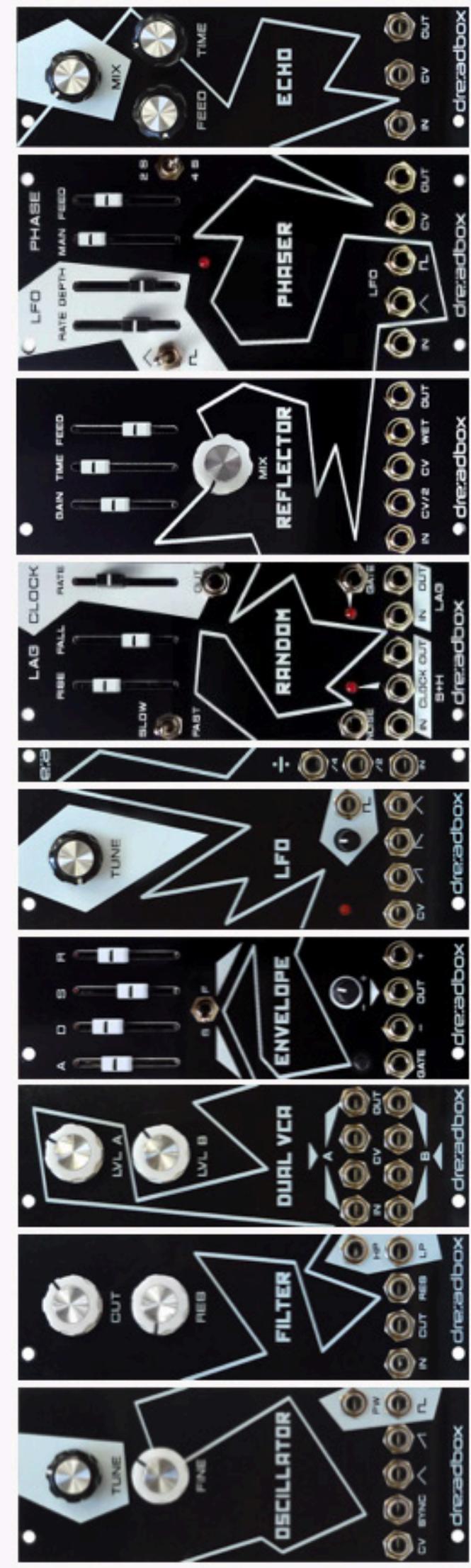


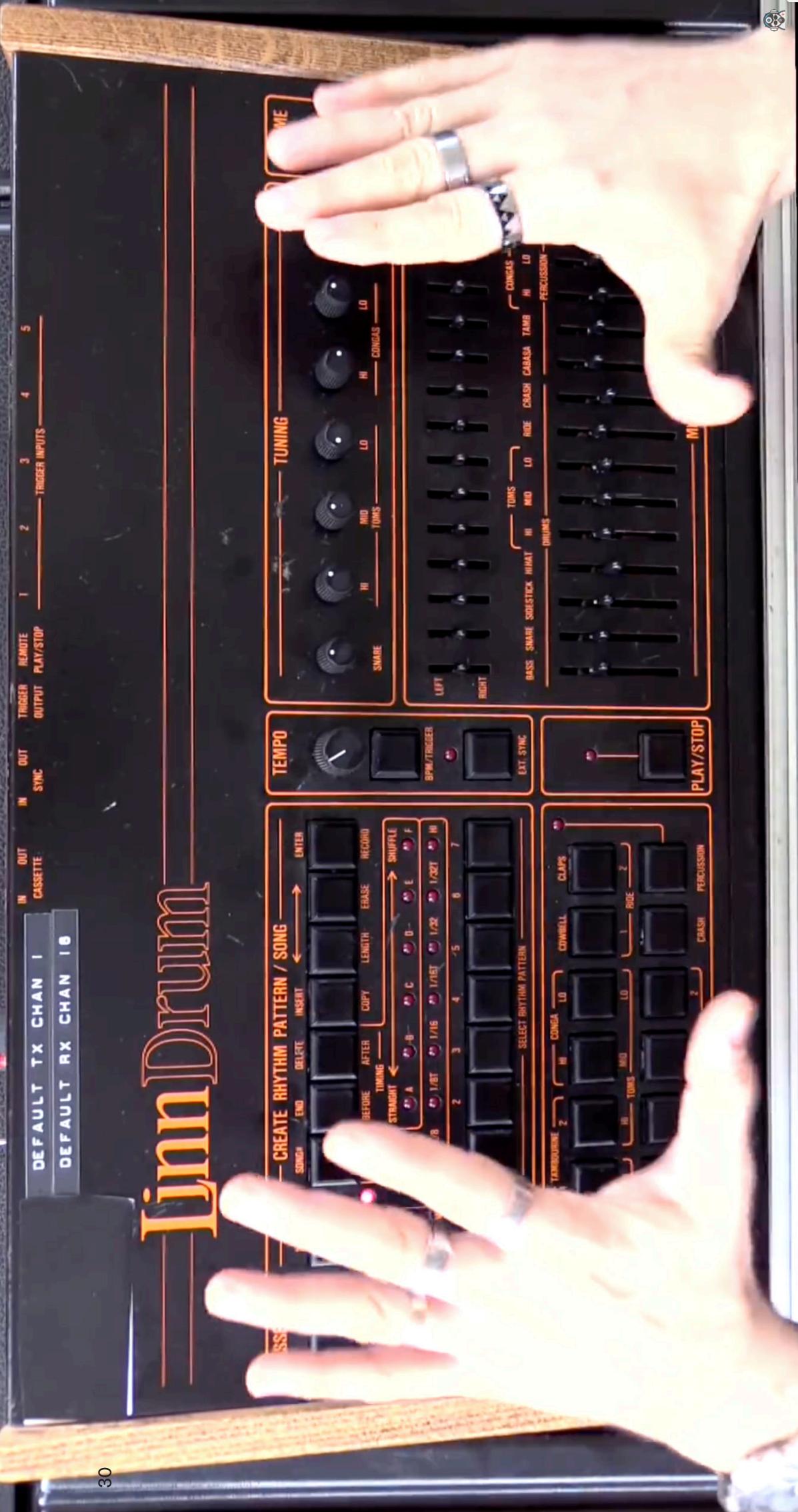












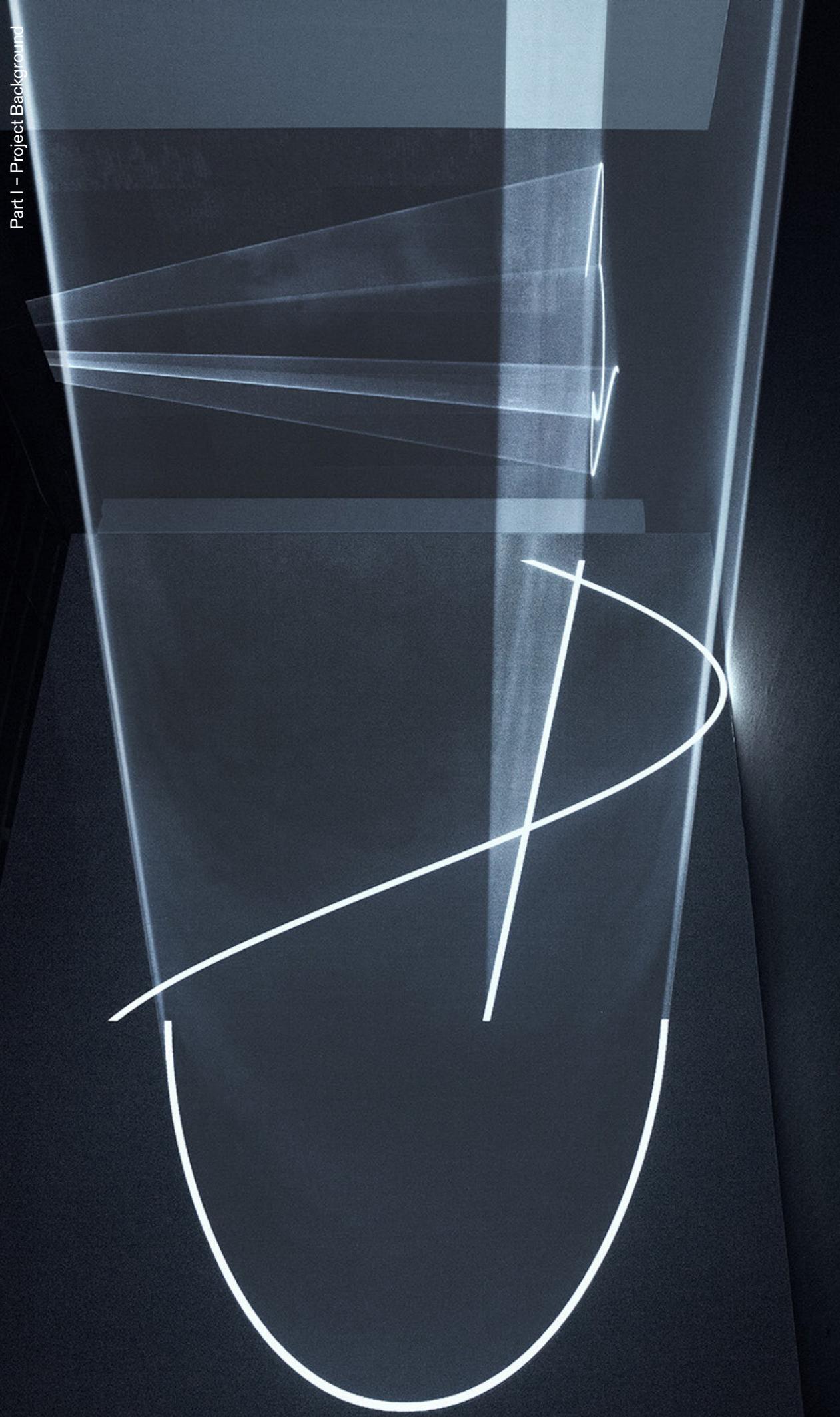


1.3

Light, color, music

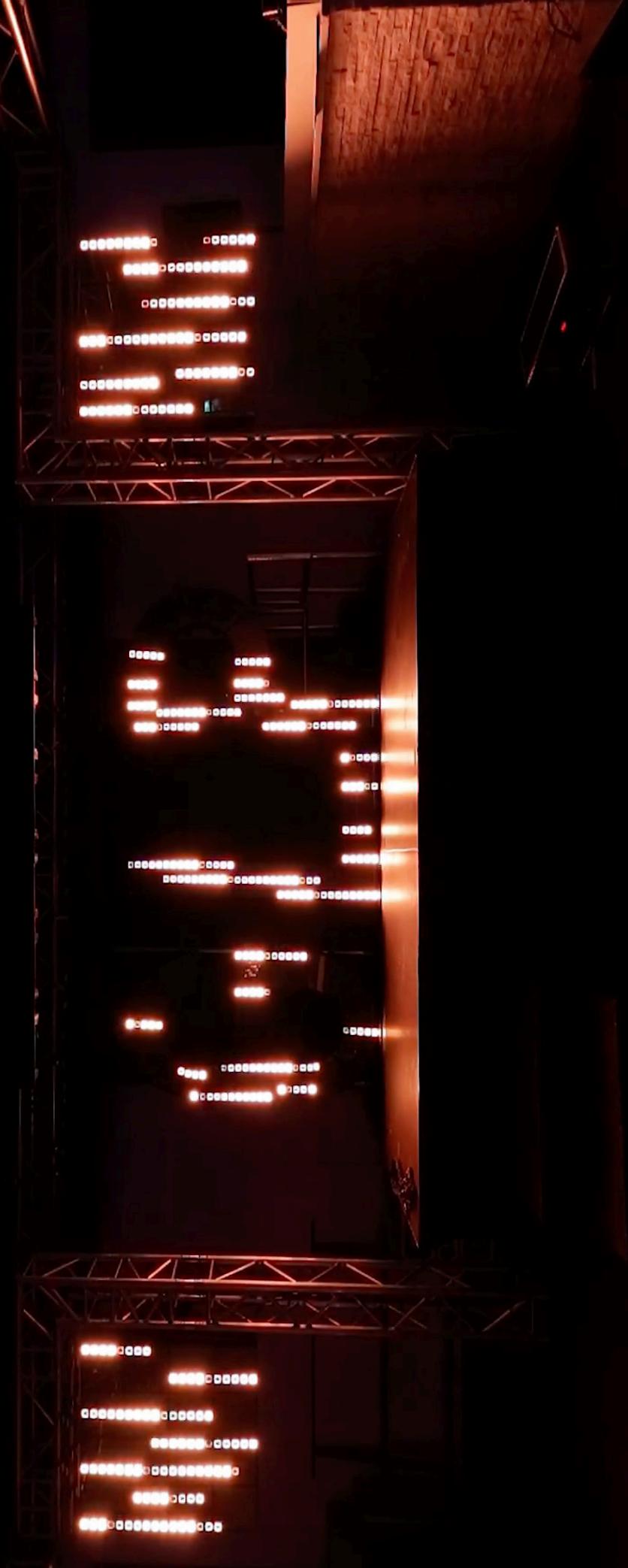
This theme came as a side subject of my Final degree report, where I examined the relationship between light, sound, and space. The reason for continuing studying this field was mainly my belief that such theme cannot be studied solely on theory – on the contrary, it relies heavily on live practice and hands-on proof.

Researching this subject also lead me to the two intriguing forefronts – the first one is the light as an instrument of the artist, being very ephemeral, transcendental, and used in a very complex concepts, and the other one is very commercial, stage lightning, which serves very different purpose and audience, yet I believe that both are powerful enough and can be merged into one.





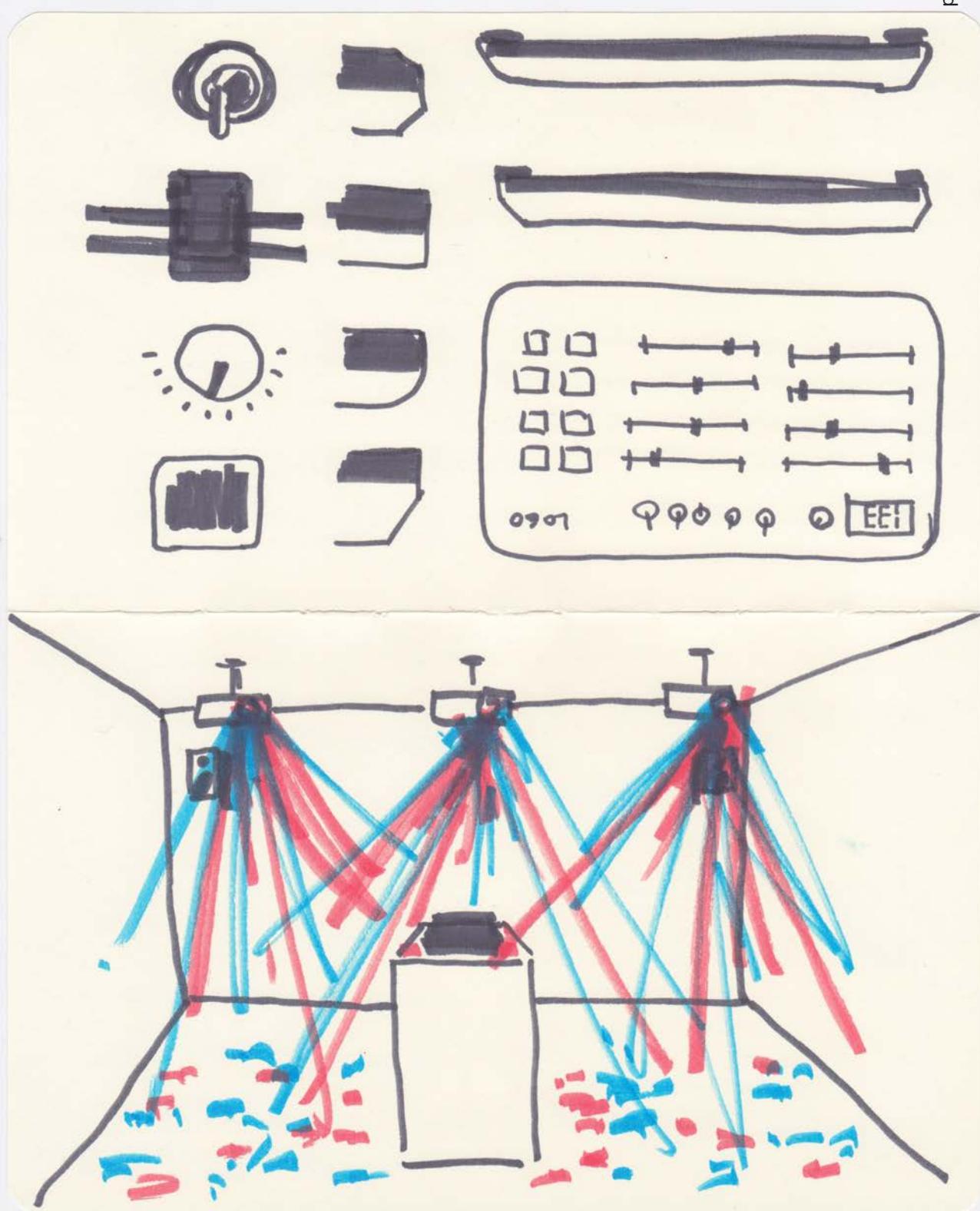




1.4 Idea development

As mentioned before, the area of interest for this project came to me years way before the project started, and it took quite a while to connect all the dots into one complete understanding of what I want to pursue and achieve. Even funnier is that even though I've been thinking about the possible outcomes for years, it was a matter of weeks since the project start I came up with a idea never considered before, and the whole project found its shape in a completely new way which was not well thought through and implied many various potential hazards.

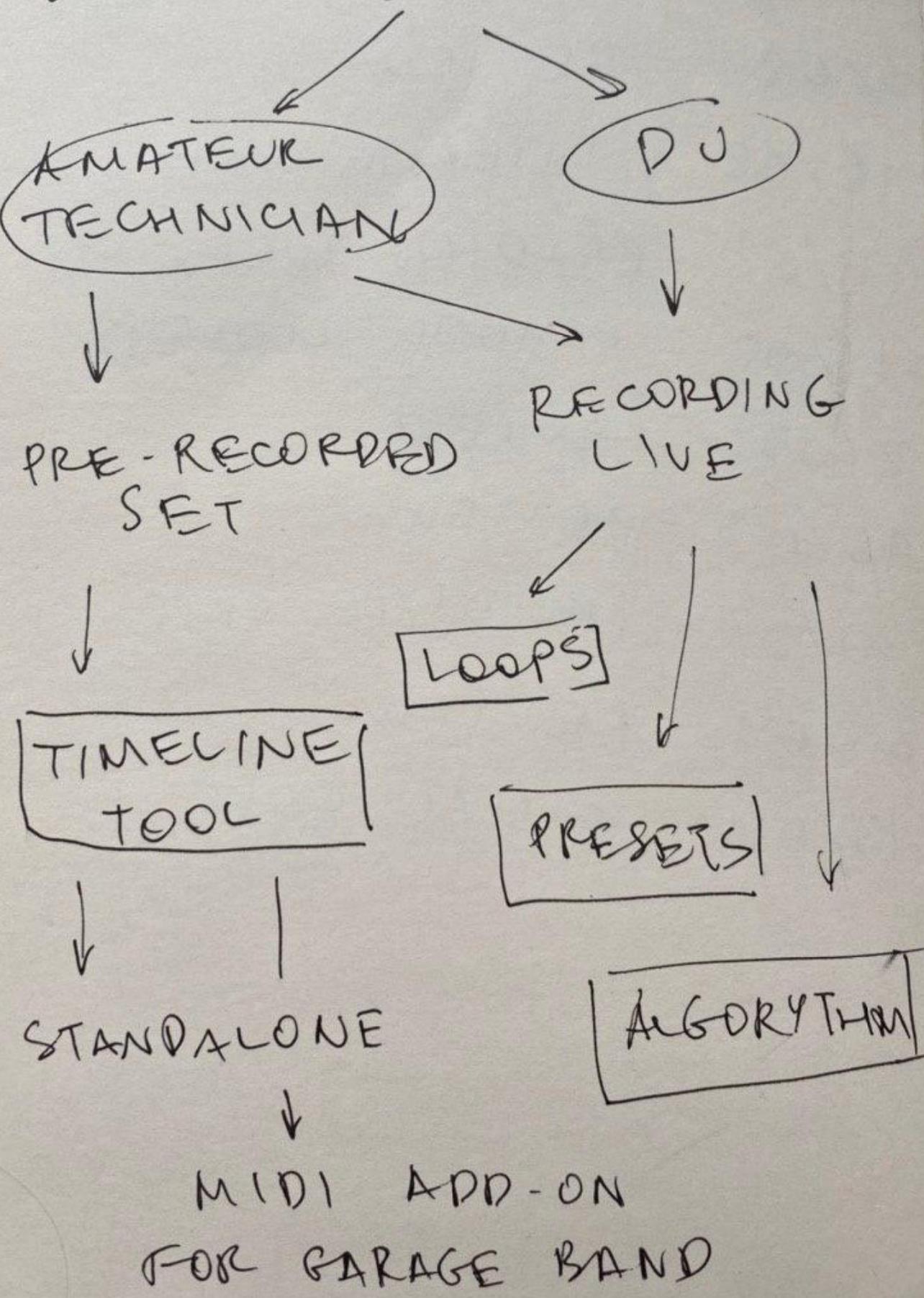
To make sure I was going in the right direction, I tried to contact as many people connected to both music and light as possible, conducted various interviews with light artists and technicians, deconstructed the notion of a song itself as well as dig up into the professional reditts to make sure I am aware of the current problems, trends, and space for innovation.



COMPOSING WITH LIGHT

USERS: AMATEURS

SCENARIOS:



PRINCIPLES:

BPM - MAIN ONE?

\equiv
 $1/1 \quad 1/2 \quad 1/u \quad 1/8 \quad \dots$

TYPES OF LIGHT:

- SPOT PROFILE
- SPOT FRESNEL
- FLOOD
- PAR CANS
- STRIP LIGHTS
- MOVING HEADS
-
-

++

- SMOKE
- LASER
- Gobo

DENIS STASOV

LIGHT TH
PREPARES THE SHOW
AT HOME
EXPORTS TO CONTROLLER
PLUGS IN ON STAGE

EUGENY ROMASHKIN
MARINA LARIKOVA

PATCHES ↑
"HEADS" OF EACH
"COMPANY"

ALL OF THEM RUN ON

DMX

MODE - RESOLUTION
ADDRESS

FOLLOW SPOT IS ALWAYS
ON MAN CONTROL -
TRACKING ISSUES
YOU CANNOT EXCLUDE
A PERSON IN MATHEMATICAL
CALCULATIONS (COMPLEX
SHOWS)

LIVE FIXING BROKEN
HEADS !!!

STRUCT TIMING
PROLIONI FESTIVAL
FRANKFURT - IN APRIL
MASTER CLASSES

DMX	- USED FOR	
CONTROLLING	ALL THE	
LIGHTS		
ATTRIBUTES		
X	0 ... 255	1
Y	0 ... 255	2
Z	0 ... 255	3
OPACITY / DIMMER		4
R		5
G		6
B		7
GOBO		8
PAN		9
TI LT		10
		11
		...

DMX 512 = 512 CH.

XLR CABLE

FARMS & EXTENSIONS

MIDI → DMX

1/0 → 0/255

ONE CHANNEL FOR
MULTIPLE FIXTURES

SEQUENCING MANUALLY

OR

PREPARED SETS

VISUAL:

- BPM
- TONE
- GENRE
- TYPES OF LIGHT

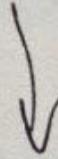
INPUT

AUX
iPod
DJ CONTROLLER



SOLUTION

EDITABLE/
NON
EDITABLE



OUTPUT

LIGHT
FIXTURES

MAIN CONCERNS:

- LEVEL OF AUTOMATION?
- CROSS-PLATFORM INTEGRATION
- VOLTAGE / CABLES
- THE UNIFICATION OF THE PROCESSES
- INTERFACE
- VISUAL LANGUAGE



INPUT VARIATIONS:

- VOLUME
- FREQUENCY
- MIDI - NOTE
- BREATH (RHYTHM)

SONG PARTS:

- INTRO
- BUILD - UP
- BRIDGE
- CHORUS
- SOLO
- OUTRO
- VERSE

DREAM MACHINES

FREQUENCY

STROBE

ANALOGUE

INTERACTIVE

AUDIENCE - CONTROLLED

LIGHT!

BUG APP

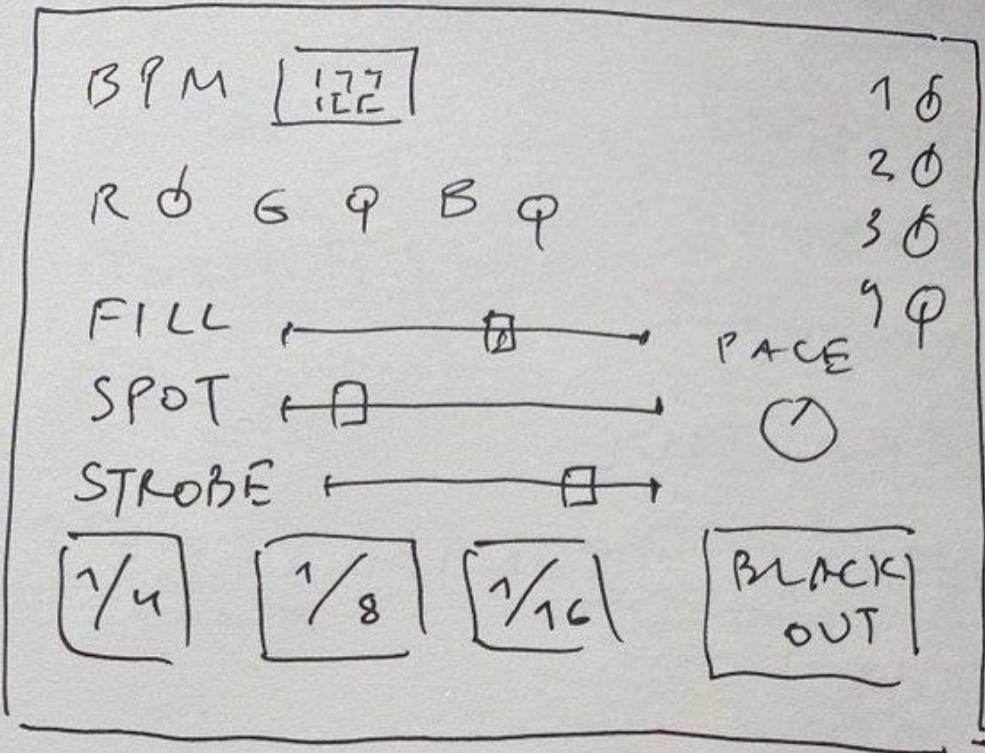
PERLIN NOISE

GL LIQUID LIGHT SHOW

AUTONOMOUS DEVICES

SCRIPTING USER SHOW

DIY - EQUIVALENT OF
THE COMPLEX SHOW

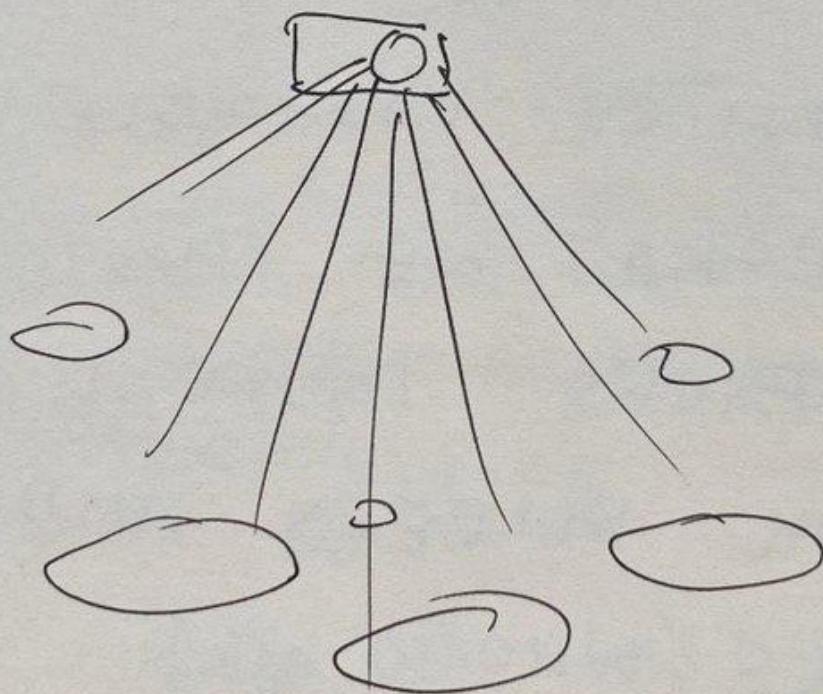
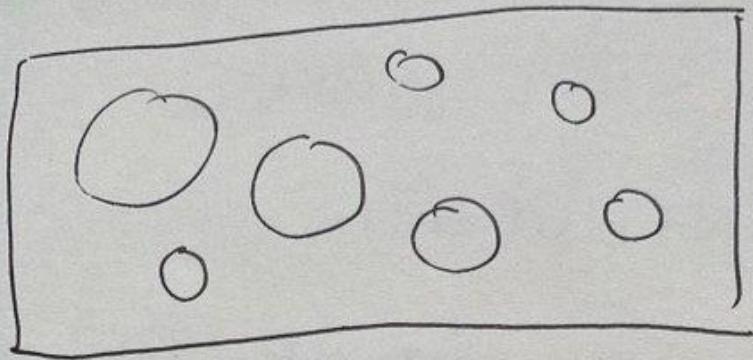


WHAT ARE THE KEY CONTROLS?

iPad / iPhone or physical controller

- SHOULD IT BE CONNECTED TO MUSIC OR BE AUTONOMOUS
- SOLUTION ON ITS OWN?

STAGE < PROJECTORS
LIGHTS



NO DMX

NO EXPENSIVE SOLUTIONS
EXHIBITION FRIENDLY

QUESTIONS TO DANIEL:

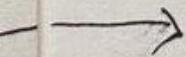
- HOW TO RUN PROJECTS OUTSIDE OF XCODE
- IOS INTEGRATION?
- HOW TO SAVE RESULTS OUTSIDE OF DATA FOLDER?
- GRAPHICS VARIABLES
- INTEGRATING PHYSICAL INTERFACES INSTEAD OF VIRTUAL SLIDERS AND BUTTONS (WHO TO ASK)
- TIMING TO BPM + SYNC — BUTTON
- MULTIPLE PROJECTIONS
- SINGLE VARIABLE STRING

QUESTIONS FOR EXHIBITION

- BLACK OUT SPACE - IS IT POSSIBLE?
- AMOUNT OF PROJECTORS TO RENT
- FOR MACHINE - IS IT ALLOWED?
- ANY OTHER FORM OF DOCUMENTATION?

- HARDWARE SOLUTIONS FOR THE EXHIBITION

- SIN/COS RHYTHM



→ FIXE
OF
BY

CREATIVE APPLICATIONS
RADIO

DIGITAL

- + EASIER TO CREATE
- + ADJUSTABLE TO MORE SCENARIOS AND PATTERNS
- + EASIER TO DISTRIBUTE
- + MORE VISUAL
- + CAN BE UPDATED

PHYSICAL

+ MORE PRECISE &
PREDICTABLE FEEDBACK

+ MAKING A CRAFT
RATHER THAN A PLAY

GENERAL QUESTIONS:

- SHOULD IT BE CONNECTED TO THE MUSIC & ITS QUALITIES AND BORROW ITS ATTRIBUTES FROM IT OR BE A FULLY AUTONOMOUS / USER CONTROLLED INSTRUMENT ON ITS OWN?
- HOW IS THE ONE-PROJECTOR SCENARIO DIFFERS FROM A MULTIPLE PROJECTORS SCENARIO?
SHOULD I CONSIDER POSITIONING OF THE MANY? OVERLAPPING?
SCREEN AND?

- IS THERE ANY WAY TO AMPLIFY (BLOW UP) THE EFFECT OF ONE OR MULTIPLE PROJECTIONS? MIRRORS? HOW TO MAKE THE MOST OUT OF THE LEAST?
- HOW HARD IT IS TO CONNECT PHYSICAL INTERFACE WITH DIGITAL TOOLS?
HOW MUCH TIME SHOULD I PRESCRIBE TO MAKING A CONTROLLER AND HOW MUCH TIME I HAVE FOR TESTING THE TOOLS DIGITALLY BEFORE MAKING THE FINAL THING?

ADDITIONAL DESIGNS:

1. INTERFACE
2. PACKAGING
3. INSTRUCTIONS (MULTIPLE LANGUAGES)
4. TUTORIAL
5. LOGOTYPE
6. ADVERTISING
7. TUTORIAL
8. LANDING
9. RENDERS / PHOTOSHOOT
10. RECORDED SESSIONS

ADDITIONAL LOOK-UPS:

- ARDUINO?
- RASPBERRY PI?
- MATERIALS & PRODUCTION TECHNIQUES - WHAT IS THE BEST & THE MOST CONVENIENT PROCESS FOR BOTH SINGLE & MASS PRODUCTION
- IS THERE ANY WAY TO CONNECT TO TRACKTOK SOFTWARE TO SHARE THE BPM / CONTROLLERS EXISTING BUTTONS & SLIDERS TO CONTROL THE LIGHTS
- TEST MULTIPLE PROJECTIONS WITH THE SAME AND INDIVIDUAL OUTPUTS

BOOK BRIEFING

- CHRONOLOGICAL
 - ALPHABETICAL
 - TAXONOMICAL
 - FORMAL
 - ASSOCIATIVE
 - RANDOM
 - FLAT
 - NESTED
- LINEAR
- NON-LINEAR
- STRUCTURE
↔
FORM

PAPER
FORMAT
2 COPIES

BINDING

[ONE BOOK!!]

HYBRID
MULTIPLE
METAPHORICAL

TITLE?
SEQUENCING?

CONTENT:

INTERVIEWS

• GRAPHICAL EXPERIMENTS

• CORE INSERTS

• INTERFACE DEVELOPMENT

• MATERIALS & PRODUCTION

• TESTING LIVE

• THEORY

MULTIPLE BOOKS?

LOOK FOR:

• OWNER'S MANUAL

• SOVIET TEXTBOOKS

• ANALOGUE SYNTHS

ITERATIONS OF
TRAIL & ERROR !

THINGS TO CONSIDER:

- SAVING ITERATIONS OF THE CODE TO GITHUB
- HOW TO SAVE THE INTERIM STAGES FOR FUTURE REFERENCE
- WHAT IS THE BEST WAY TO SHOW THE MOSAIC IMAGE ON PAPER

GRID OPTIONS:

- RECTANGULAR

VARIABLES:

> X DIV

> Y DIV

> CHAOTICNESS

- CIRCULAR

> RADIUS

> N OF CIRCLES

> CENTRE SHIFT

- CHAOTIC

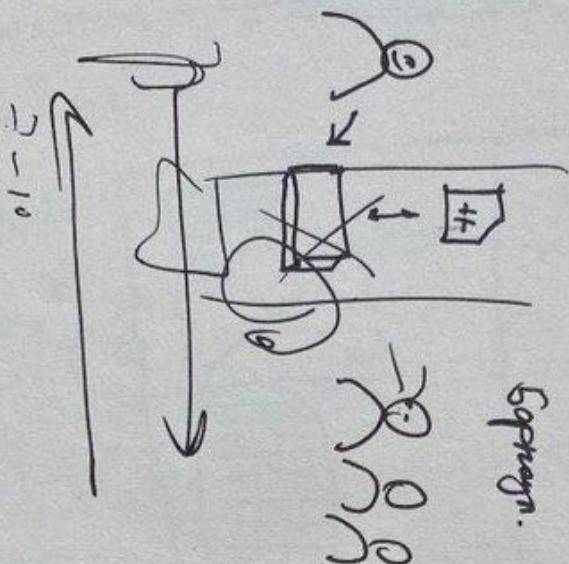
COLOR OPTIONS:

- > N HUES
- > HUE VARIATIONS
PREDEFINED GROUPS
OR
- > N HUES
- > SELECTING HUE
 - > R
 - > G
 - > B

WHAT IS THE MAIN
CONTROL FOCUS?

- CAN I POSITION SOME
OF THE INTERFACE ON
THE SIDE OF THE BOX?

DANIEL ZAKH MEETING



VVVV

TOUCH DESIGNER

СЕΛБСКИЕ
ДУСКОТЫ
REDEFINED

ПРЕЗЕНТАЦИИ
КОЛХОЗА

OF BOOTHER

Stop Save PS → App Store

STROBE
FUNCTIONING
MORE PREDICTABLE
VISIBLE DIAGRAM
INTERFACE VISUALS
SIGNAL
VISUALISE WHAT'S
GOING TO HAPPEN
BEFORE IT DOES
MAKE IT MORE INTUITIVE
SLIDERS ↔ KNOBS?
OPTIONS?
HIERARCHY OF ALL
OF THE INSTRUMENTS
FREQUENCY OF USE
MIXING DESKS
MOVING SLIDERS AT THE

SAME TIME
THE MORE YOU USE
IT, THE BIGGER IT IS
ICONS?

DIGITAL VERSION!!!

ADD DETAILS TO ALL
OF THE RENDERS
PACKAGING!

Book!

PLUGGING 2 ARDUINO
IN ONE IS POSSIBLE?

RASPBERRY PI RESEARCH

INPUT / OUTPUT

YECM RCEx!!!

SERIOUSLY WORK ON
THE PROJECT IN THE
NEXT COUPLE OF WEEKS

1.5

Proof of concept

At the first stage of the project I wanted to create an amateur controller for stage lightning equipment, and learned all about it, objectively assessing what I can do to make a change in this field. However, there were various issues regarding the infinite amount of light fixtures, their high price range, and even though they were all running on one DMX signal, all of them required different amount of channel input to work. I thought that I could manage this, but at the same time felt that would be a very niche project, also setting me in a very tight borders for expression and innovation. I could potentially solve some of the problems the field was facing, but not sure whether it would be fascinating enough for me from a viewpoint of art and design.

The solution came quick and unexpected. At some point I realised that the best lightning equipment many people have at their disposal are the home cinema projectors. Not only they are affordable enough, but also allow me to produce video output looking like stage lights rather than fight with their complex DMX standards and issues of compatibility.

After making several tests at my bathroom, it became clear that this is the best solution possible. However, the idea of making a physical controller meant for amateurs remained.

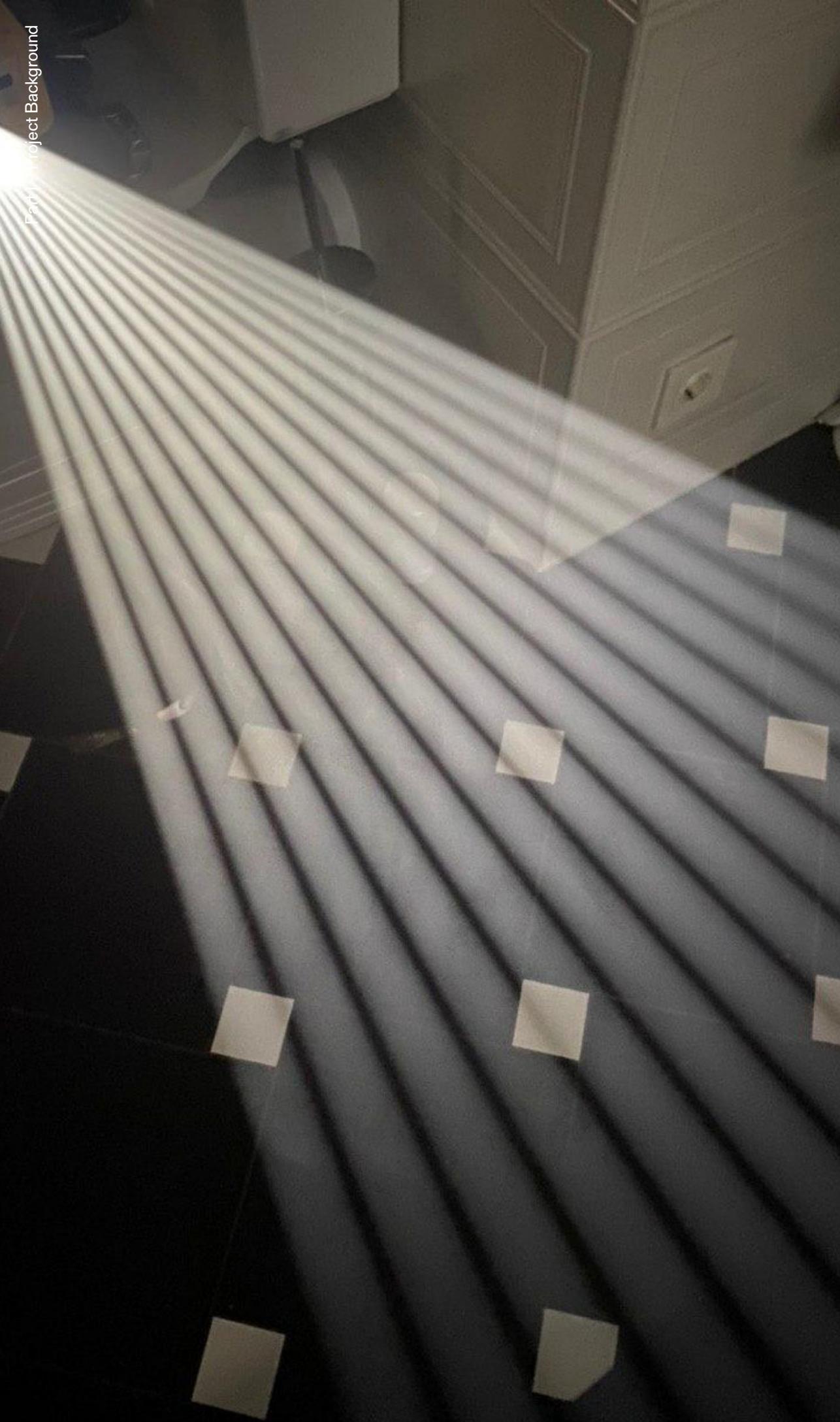




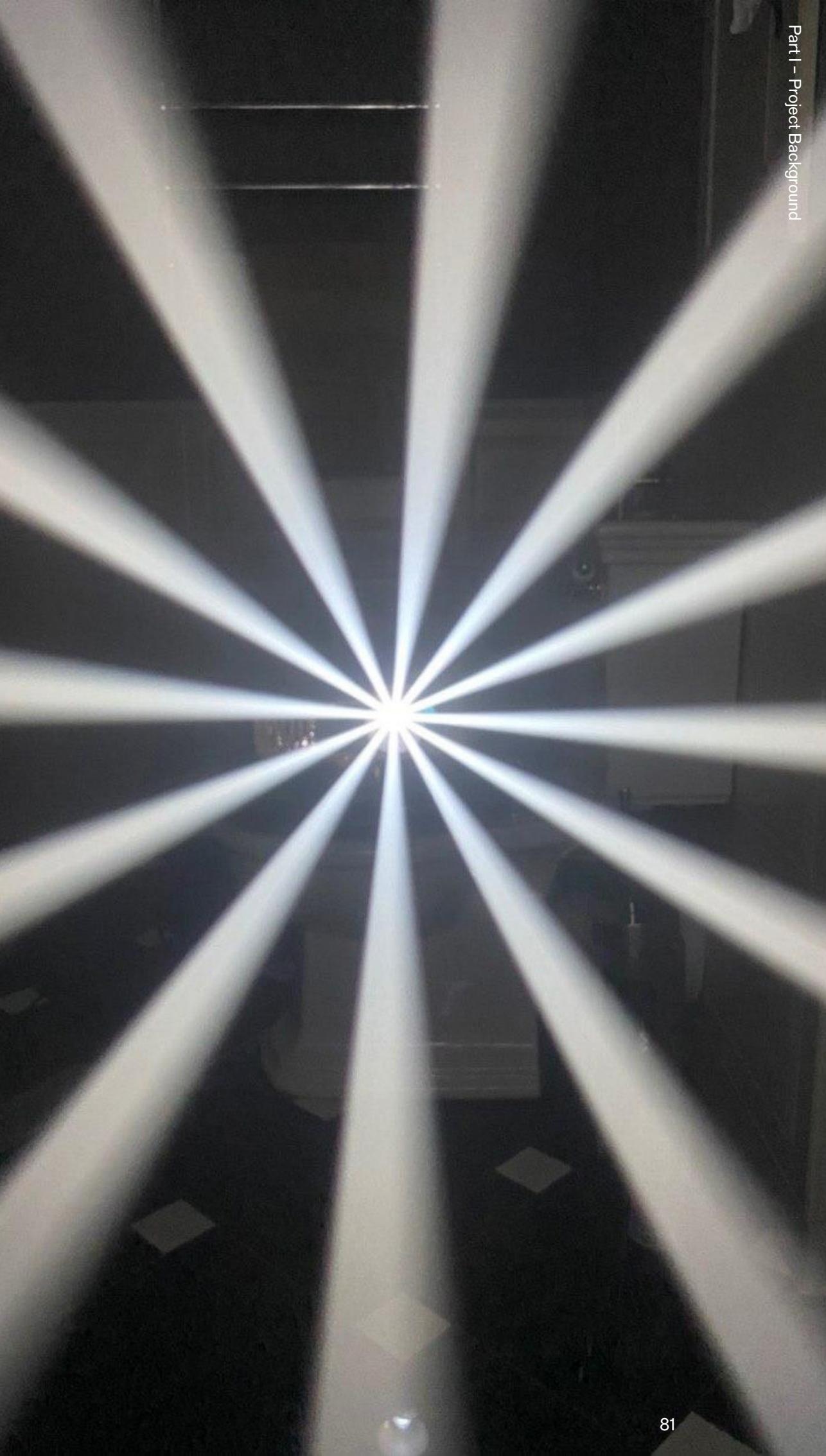












Part II

Algorithm


```
#pragma once
#include <ofMain.h>
#include <ofxGui.h>

class ofApp : public ofBaseApp {

public:
    void setup();
    void update();
    void draw();
    void keyPressed(int key) {};
    void keyReleased(int key) {};
    void mouseMoved(int x, int y) {};
    void mouseDragged(int x, int y, int button) {};
    void mousePressed(int x, int y, int button) {};
    void mouseReleased(int x, int y, int button) {};
    void windowResized(int w, int h) {};
    void dragEvent(ofDragInfo dragInfo) {};
    void gotMessage(ofMessage msg) {};

    float timer = 0;

    ofxLabel general;

    ofxIntSlider bpm;

    ofxButton synchronize;
    void syncButtonPressed();

    ofxToggle background;
    ofxToggle fillnofill;

    ofxLabel strobe;

    ofxToggle one;
    ofxToggle half;
    ofxToggle quarter;
    ofxToggle eighth;
    ofxToggle sixteenth;
    ofxToggle onethirtysecond;

    ofxLabel variables;

    ofxIntSlider particles;
    ofxIntSlider radius;
    ofxIntSlider brightness;

    ofxLabel grid;

    ofxToggle line;
    ofxToggle circle;
    ofxToggle square;
    ofxToggle triangle;
    ofxToggle hexagon;

    ofxLabel motion;

    ofxToggle updown;
    ofxToggle leftright;
    ofxToggle clockwise;
    ofxToggle anticlockwise;
    ofxToggle uplefttodownright;
    ofxToggle downlefttoupright;

    ofxLabel channels;

    ofxToggle ch1;
```



```

#include <ofApp.h>
vector<ofColor> colList;
vector<ofVec2f> ptList;
int curColId = 0;
-----
void ofApp::setup( )
{
    ofSetFrameRate(60);
    ofBackground(0);
    ofSetTitle(«SVETOUSTANOVKA PORTATIVNAYA»);
    gui.setup(«SP-LJVN-2020»);
    gui.add(general.setup(«GENERAL», «»));
    gui.add(bpm.setup(«BPM», 60, 0, 300));
    gui.add(synchronize.setup(«SYNC»));
    gui.add(background.setup(«BACK», true));
    gui.add(fillnofill.setup(«FILL», true));
    gui.add(strobe.setup(«STROBE», «»));
    gui.add(one.setup(«1», true));
    gui.add(half.setup(«1/2», false));
    gui.add(quarter.setup(«1/4», false));
    gui.add(eighth.setup(«1/8», false));
    gui.add(sixteenth.setup(«1/16», false));
    gui.add(onethirtysecond.setup(«1/32», false));
    gui.add(variables.setup(«VARIABLES», «»));
    gui.add(particles.setup(«N», 15, 1, 1000));
    gui.add(radius.setup(«R», 30, 1, 1000));
    gui.add(brightness.setup(«B», 255, 0, 255));
    gui.add(grid.setup(«GRID», «»));
    gui.add(line.setup(«LINE», true));
    gui.add(square.setup(«SQUARE», false));
    gui.add(triangle.setup(«TRIANGLE», false));
    gui.add(hexagon.setup(«HEXAGON», false));
    gui.add(circle.setup(«CIRCLE», false));
    gui.add(motion.setup(«MOTION», «»));
    gui.add(updown.setup(«UPDOWN», true));
    gui.add(leftright.setup(«LEFTRIGHT», false));
    gui.add(clockwise.setup(«CLOCKWISE», false));
    gui.add(anticlockwise.setup(«ANTICLOCKWISE», false));
    gui.add(uplefttowright.setup(«UPLEFTTOWRIGHT», false));
    gui.add(downlefttoupright.setup(«DOWNLEFTUPRIGHT», false));
    gui.add(channels.setup(«CHANNELS», «»));
    gui.add(ch1.setup(«CH1», true));
    gui.add(ch2.setup(«CH2», false));
    gui.add(ch3.setup(«CH3», false));
    gui.add(ch4.setup(«CH4», false));
    gui.add(ch5.setup(«CH5», false));
    gui.add(ch6.setup(«CH6», false));
    gui.add(hue1.setup(«HUE #1», 255, 0, 255));
    gui.add(hue2.setup(«HUE #2», 255, 0, 255));
    gui.add(hue3.setup(«HUE #3», 255, 0, 255));
    gui.add(hue4.setup(«HUE #4», 255, 0, 255));
    gui.add(hue5.setup(«HUE #5», 255, 0, 255));
    gui.add(hue6.setup(«HUE #6», 255, 0, 255));
    ofSetCircleResolution(50);
}
-----
void ofApp::update( )
{
    ofSetBackgroundAuto(background);
    if(fillnofill == true)
    {
        ofFill();
    }
    else
    {
        ofNoFill();
    };
    if(updown == true)

```



```

        colList.push_back(ofColor(c1));
    };
    if(ch2 == true)
    {
        colList.push_back(ofColor(c2));
    }
    else {};
    if(ch3 == true)
    {
        colList.push_back(ofColor(c3));
    }
    else {};
    if(ch4 == true)
    {
        colList.push_back(ofColor(c4));
    }
    else {};
    if(ch5 == true)
    {
        colList.push_back(ofColor(c5));
    }
    else {};
    if(ch6 == true)
    {
        colList.push_back(ofColor(c6));
    }
    else {};
    float offsetX = ofGetWidth() / ( particles + 1 );
    float offsetY = ofGetHeight() / 2;
    float stepX = ofGetWidth() / ( particles + 1 ) - radius;
    float stepY = ofGetHeight() / 2 - radius;
    float globalShiftX = 0;
    float globalShiftY = 0;
    if(updown)
    {
        globalShiftX = 0;
        globalShiftY = sin(t)*stepY;
    }
    if(leftright)
    {
        globalShiftX = sin(t)*stepX;
        globalShiftY = 0;
    }
    if(clockwise)
    {
        globalShiftX = cos(t)*stepY;
        globalShiftY = sin(t)*stepY;
    }
    if(anticlockwise)
    {
        globalShiftX = sin(t)*stepY;
        globalShiftY = cos(t)*stepY;
    }
    if(upleftttdownright)
    {
        globalShiftX = cos(t)*stepY;
        globalShiftY = cos(t)*stepY;
    }
    if(downlefttoupright)
    {
        globalShiftX = -sin(t)*stepY;
        globalShiftY = sin(t)*stepY;
    }

    if(line || circle)
    {
        for(int i = 0; i < particles; i++)
        {

```


2.1 Code

Thinking well in advance of my project, last year I completed the Creative Coding elective course which allowed me to write my own software solutions for each particular task. Even I cannot say that the knowledge I got made me a code geek in any kind, it still was enough for me to write most of the desired functional without spending much times and nerves on forums.

The foundation of the algorithm is the C++ code written using OpenFrameworks libraries. I like them because of their logical syntax, fast and clean work, and cross compatibility between all the major systems like MacOS, Windows, Linus, iOS and Android. There are also thousands addons and open libraries available to include in the software, yet the foundation itself was flexible enough that I did not need any additional resources.

I had to overcome various difficulties, the main one being my iMac too old for running the XCode app in which the OpenFrameworks works. This resulted in many failed attempts to compile the code from distance, sending the .txt lines to my girlfriend's laptop, finding another computer in a fully enclosed quarantine Moscow, realising it is even older than mine, and then finally downloading the 2008 version from a very, very suspicious looking torrent website.

It took me 27 hours to write the code in one go.

2.2 BPM

```
ofSetFrameRate(60);  
numOfMoves = (float) ( bpm ) / 60.0;  
float speed = numOfMoves / 20;  
t += speed;
```

The foundation of the algorithm is linking the produced outcome motion to the bpm which is manually set by the players, adapting to the music they are playing to. The easiest way to solve this was to set the framerate of the app to 60 to match with the usual time measurement. Then, a variable is created to add a point on every frame, which begins counting on the moment the programm starts, can be set to zero and does not rely on the system time which can vary from system to system.

SET TIME TO SECS
 → timeMHz * 60 : bpm
 MAKE TIME = 0

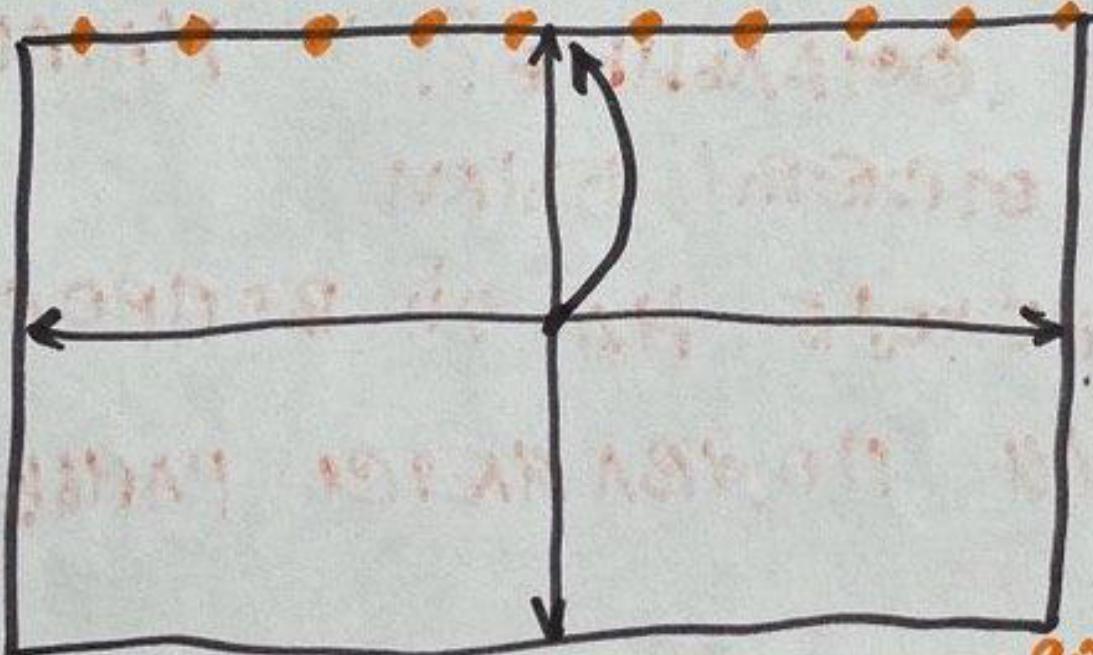
STORE time * 1/2/4/8/16/32
 of fill of no fill backg

num particles J

radius r

brightness colorue (R, B, 255)

BPM $\frac{60}{60} = \begin{matrix} -90 \\ +90 \end{matrix}$ IN SEC



2.3 SYNC

```
if(synchronize) {t = 0;}
```

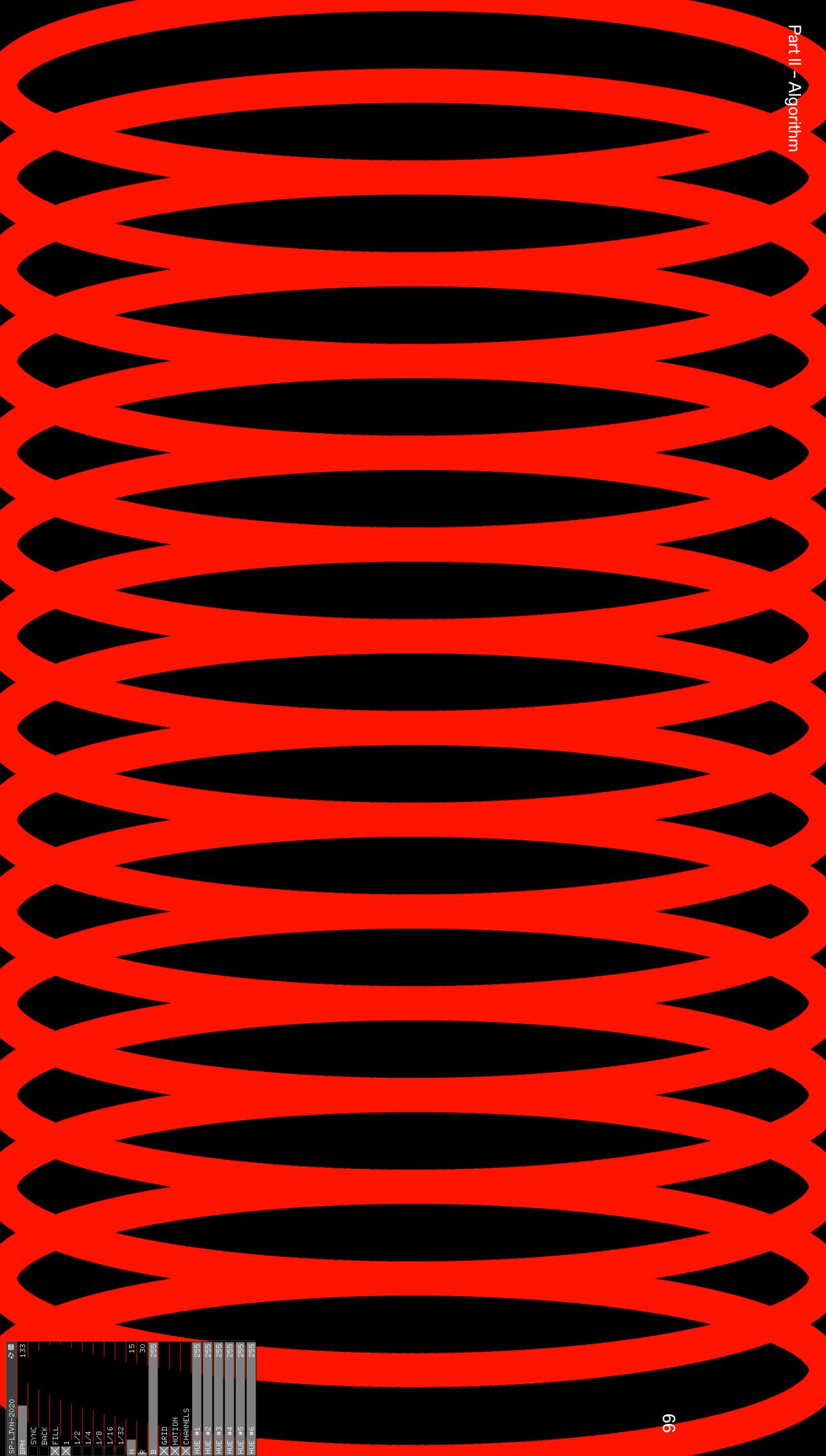
To match up perfectly with the beat, there is a separate SYNC button which sets the time variable to zero, which starts the process of drawing precisely at the time you hit it.

SP-LVNH-2020	60
GENERAL:	
BPM:	
SYNC	
BLACK	
FILL	
STROBE:	
1	
1.2	
1.4	
1.6	
1.8	
1.16	
1.32	
VARIABLES:	
IN	15
R	30
B	255
GETTO:	
C1:	
LINE	
SQUARE	
TRIANGLE	
HEXAGON	
CIRCLE	
MOTIONTO:	
UPDOWN	
LEFTRIGHT	
CLCKWISE	
ANTICLKWISE	
UPLFTDWNRT	
DWNLFTRIGHT	
CHANNELS:	
C14	
C12	
C13	
C14	
C15	
C16	
HUE # 1	255
HUE # 2	255
HUE # 3	255
HUE # 4	255
HUE # 5	255
HUE # 6	255
HUE # 7	255

2.4 BACK

```
gui.add(background.setup(<<BACK>>, true));  
ofSetBackgroundAuto(background);
```

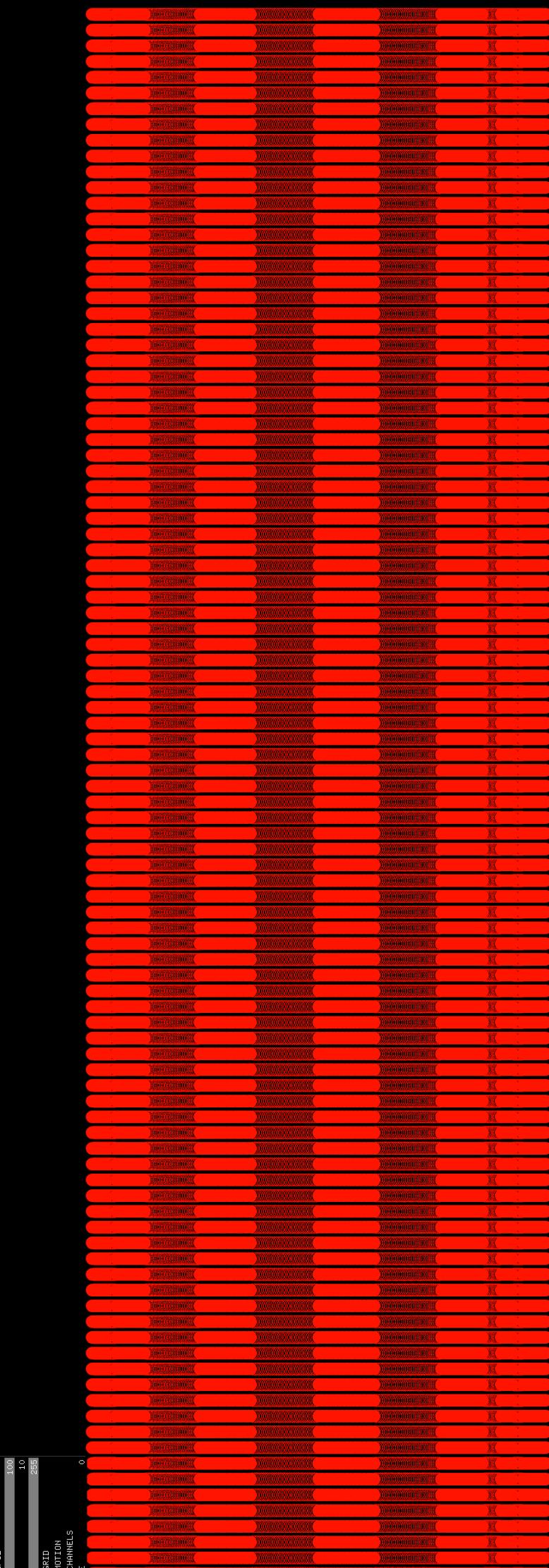
This is honestly my favorite feature of the whole code, which makes the whole tool follow a completely different path – when you deselect the toggle, it disables the background refresh which happens every frame by default, meaning that all of the graphic drawn every frame adds to one another. This creates absolutely stunning imagery and can be a work tool on its own, providing area for practically infinite variations.



2.5 OUTLINE

```
if(fillnofill == true)
{
    ofFill( );
}
else
{
    ofNoFill( );
```

Yet another one very simple and basic command of the OpenFrameworks drawing tools, which nevertheless results in a completely different look if used on projector – when you choose between the fill and no fill, the first option serves as a tool for creating light shows, the other looks exactly like a laser one.



2.6

STROBE

```

if(half == true)
{
    if(ofGetFrameNum( ) % 30 == 0)
    {
        ofSetColor(0);
        ofDrawRectangle(0, 0, ofGetWidth( ),
ofGetHeight( ));
    }
    else {};
}
else {};
if(quarter == true)
{
    if(ofGetFrameNum( ) % 15 == 0)
    {
        ofSetColor(0);
        ofDrawRectangle(0, 0, ofGetWidth( ),
ofGetHeight( ));
    }
    else {};
}
else {};
if(eighth == true)
{
    if(ofGetFrameNum( ) % 7 == 0)
    {
        ofSetColor(0);
        ofDrawRectangle(0, 0, ofGetWidth( ),
ofGetHeight( ));
    }
    else {};
}
...

```

Probably the most stupidly writed function of the whole algorithm which draws a black rectangle on top of the drawboard each 2nd/4th/8th/16th/32nd frame depending on the toggle you choose, creating a strobe effect. It is really motion sickening.

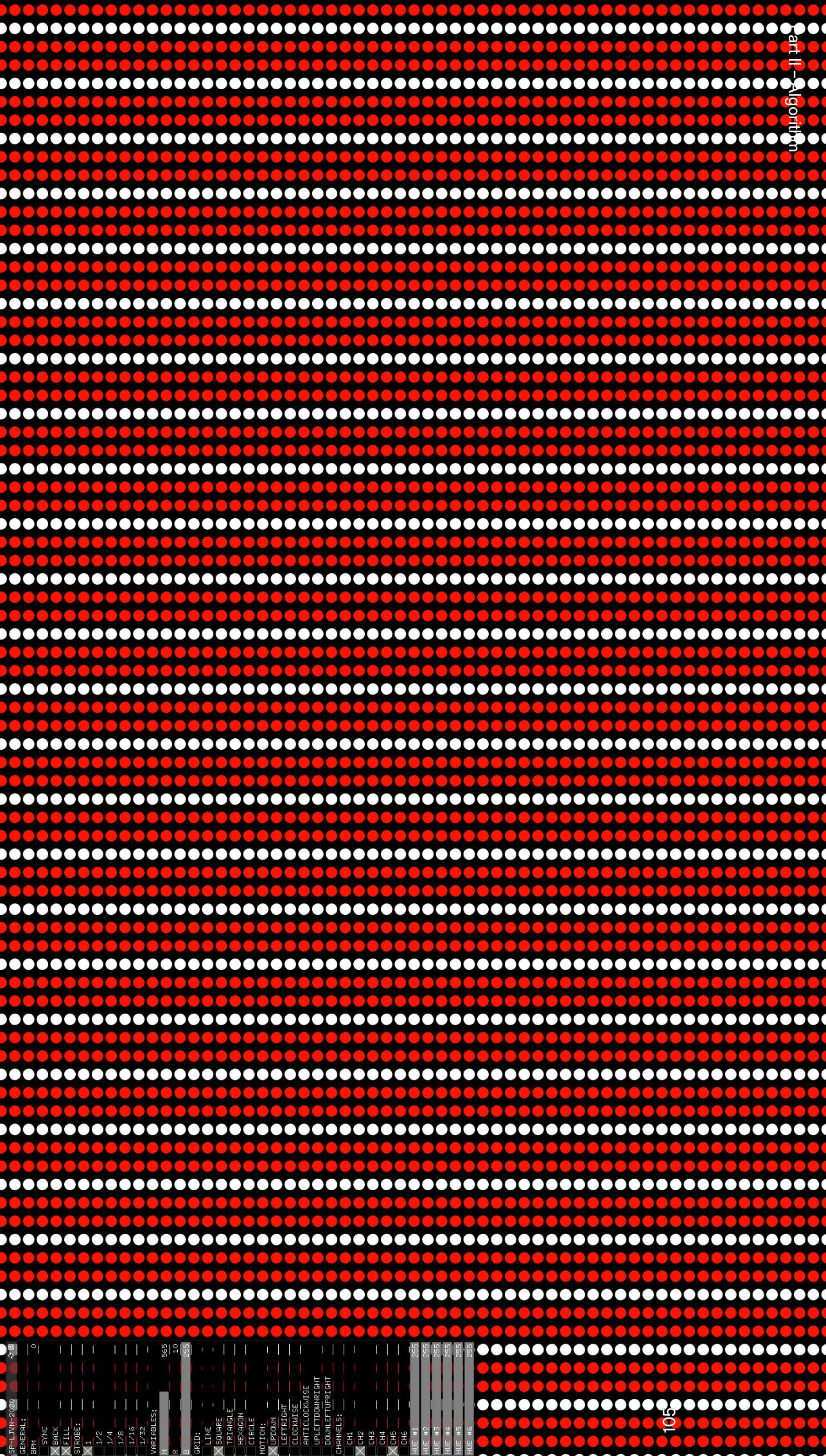
At first I thought of simply multiplying the time variable to the extencet needed, yet this provided really epileptic imagery so I sticked to the idea I wrote above.



2.7 VARIABLES

```
gui.add(particles.setup(<<N>>, 15, 1, 1000));  
gui.add(radius.setup(<<R>>, 30, 1, 1000));  
gui.add(brightness.setup(<<B>>, 255, 0, 255));
```

The three key variables set by the sliders are number of particles drawn on the screen, their radius, and their brightness. The first two are widely used in all the rest of the algorithm to provide even distribution of these across the canvas, and the third one acts more as a dimmer for controlling the pace of the output.



2.8 MOTION

```

if(updown)
{
    globalShiftX = 0;
    globalShiftY = sin(t)*stepY;
}
if(leftright)
{
    globalShiftX = sin(t)*stepX;
    globalShiftY = 0;
}
if(clockwise)
{
    globalShiftX = cos(t)*stepY;
    globalShiftY = sin(t)*stepY;
}
if(anticlockwise)
{
    globalShiftX = sin(t)*stepY;
    globalShiftY = cos(t)*stepY;
}
if(uplefttодownright)
{
    globalShiftX = cos(t)*stepY;
    globalShiftY = cos(t)*stepY;
}
if(downlefttoupright)
{
    globalShiftX = -sin(t)*stepY;
    globalShiftY = sin(t)*stepY;
}

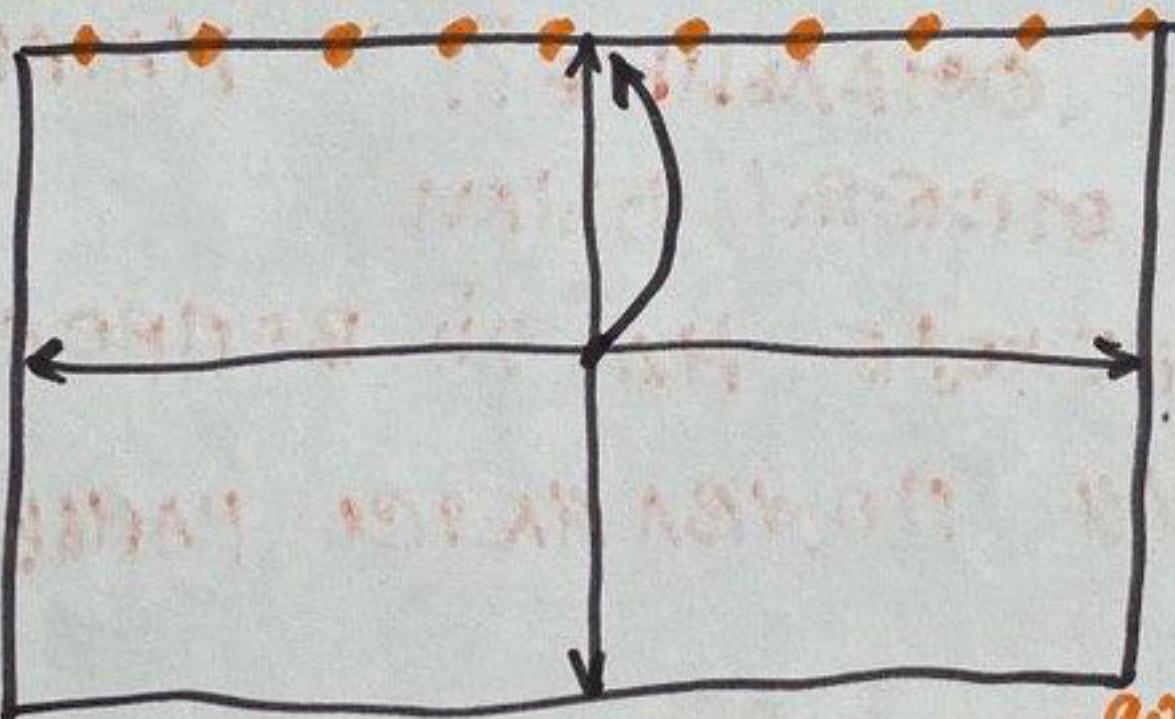
```

The trick for creating the smooth motion of all the particles was to include the calculation of the sin and cos of the time float created in the very beginning of the code.

While the float definition is adding every frame, the sin and the cos of it are smoothly moving from -1 to 1, calmly slowing down when reaching the two range limits. Combining different variations of the canvas X and Y position multiplied on sin and cos, we get a full range of movement available for all the particles which are drawn on it.

radius r
 brightness $(R, G, B, 255)$

$$\text{BPM } \frac{60}{60} = -90 \text{ IN DNR}$$



90

$\begin{matrix} \nearrow & \cos + \cos / \sin + \sin \\ G & \cos + \sin & \rightarrow 0, \sin \\ G & \sin + \cos & \\ \searrow & -\sin + \cos & \rightarrow \sin, 0 \end{matrix}$

2.9

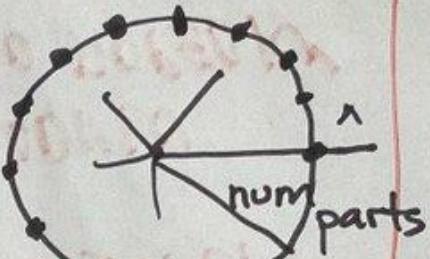
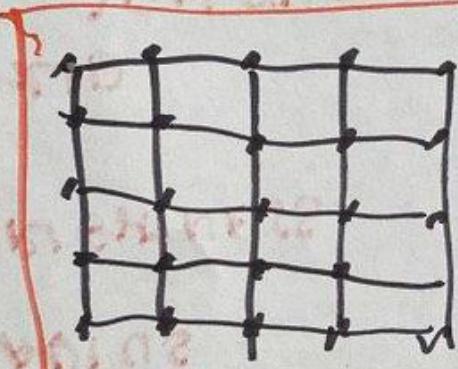
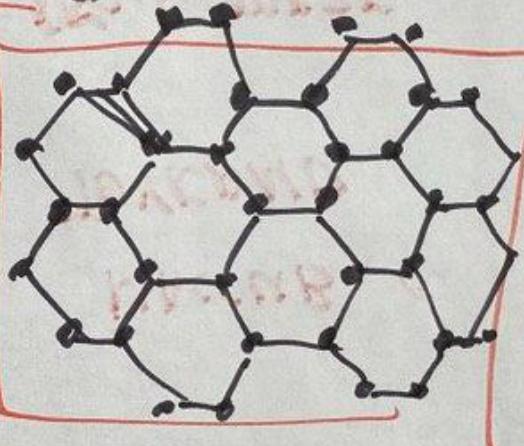
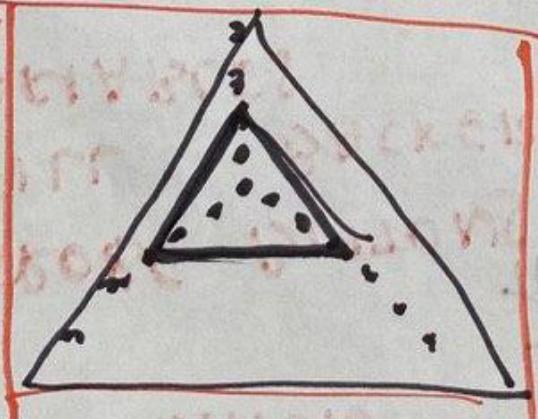
GRID

```

if(line || circle)
{
    for(int i = 0; i < particles; i++)
    {
        ofColor c = colList[i%colList.size( )];
        ofSetColor(c);
        float x = 0;
        float y = 0;
        if(line)
        {
            x = offsetX + ( offsetX * i ) +
globalShiftX;
            y = offsetY + globalShiftY;
        }
        else if(circle)
        {
            x = ofGetWidth( )*0.5 + sin(i * (
PI*2 / (float) particles ))*300 + globalShiftX;
            y = ofGetHeight( )*0.5 + cos(i * (
PI*2 / (float) particles ))*300 + globalShiftY;
        }
        ofDrawCircle(x, y, radius);
    }
}
...

```

The grids are simple variations on the screen dimensions divided by the number of particles and their radius variables, apart from examples where there was a need to contain certain shape (the triangle and the circle – in these cases these are predefined options because mapping them to the screen would result with oval and smashed form of the triangle.

LINE $\text{offsetWidth}/2$ **HOLE**
CHANNEL**WORLDS**
MAPS**CIRCLE** $* 360/\lambda$ **GRID****HEX****TRIANGLE**

2.10

COLOR

```

    ofColor c1 = ofColor(0);  c1.setHsb(hue1, 255,
brightness);
    ofColor c2 = ofColor(0);  c2.setHsb(hue2, 255,
brightness);
    ofColor c3 = ofColor(0);  c3.setHsb(hue3, 255,
brightness);
    ofColor c4 = ofColor(0);  c4.setHsb(hue4, 255,
brightness);
    ofColor c5 = ofColor(0);  c5.setHsb(hue5, 255,
brightness);
    ofColor c6 = ofColor(0);  c6.setHsb(hue6, 255,
brightness);

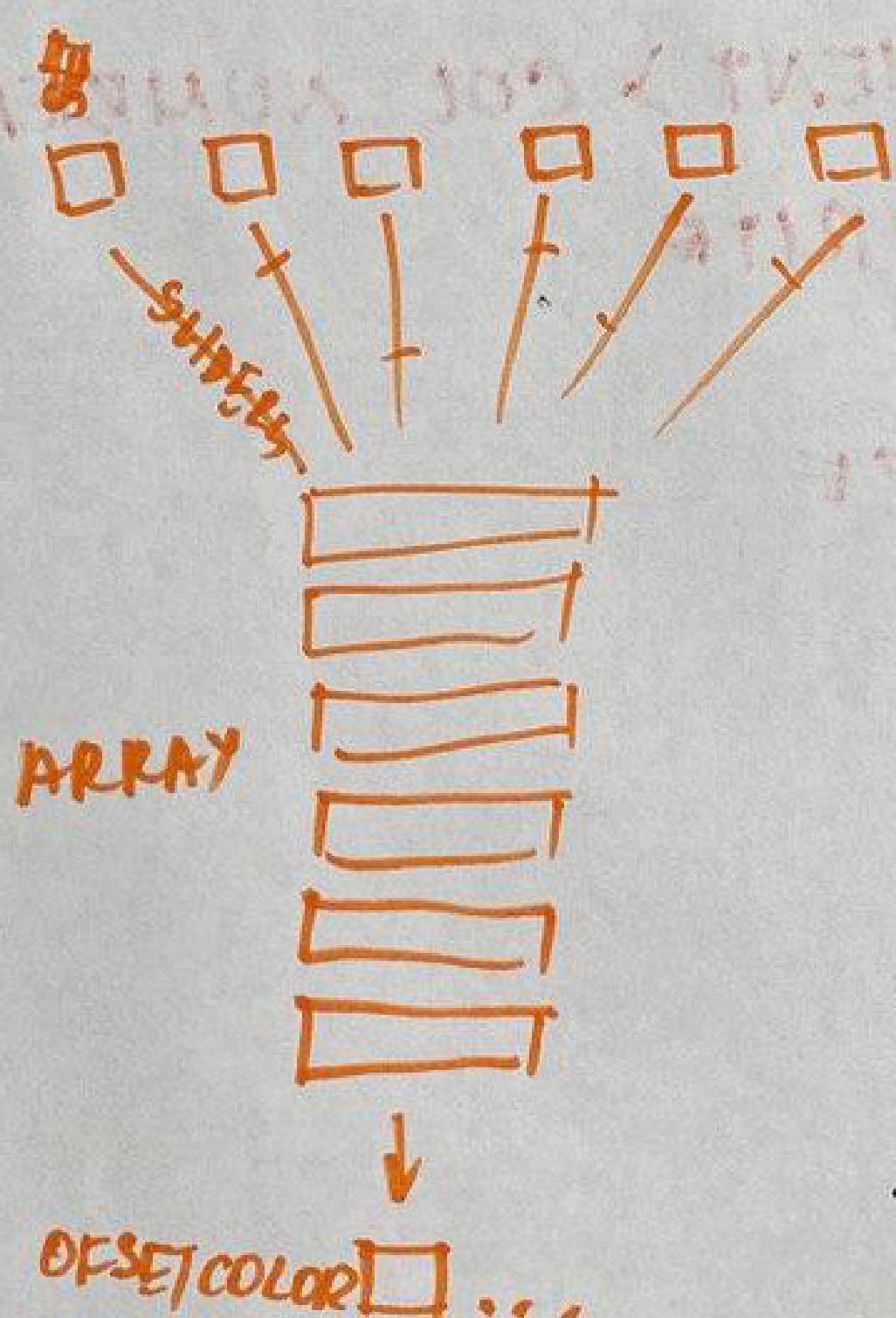
    if(ch1 == true)
    {
        colList.push_back(ofColor(c1));
    }
    else
    {
        c1.setHsb(hue1, 0, brightness);
        colList.push_back(ofColor(c1));
    };
    if(ch2 == true)
    {
        colList.push_back(ofColor(c2));
    }
    else {};
    if(ch3 == true)
    {
        colList.push_back(ofColor(c3));
    }
    else {};
    if(ch4 == true)
    {
        colList.push_back(ofColor(c4));
    }
    ...

```

The colors are wrote down and stored in an array which is being created and cleared every frame depending on the number of the selected channels. This is made for the reason of coloring all the particles evenly one after the other across the whole canvas, not dividing it into separate blocks of one color.

90°

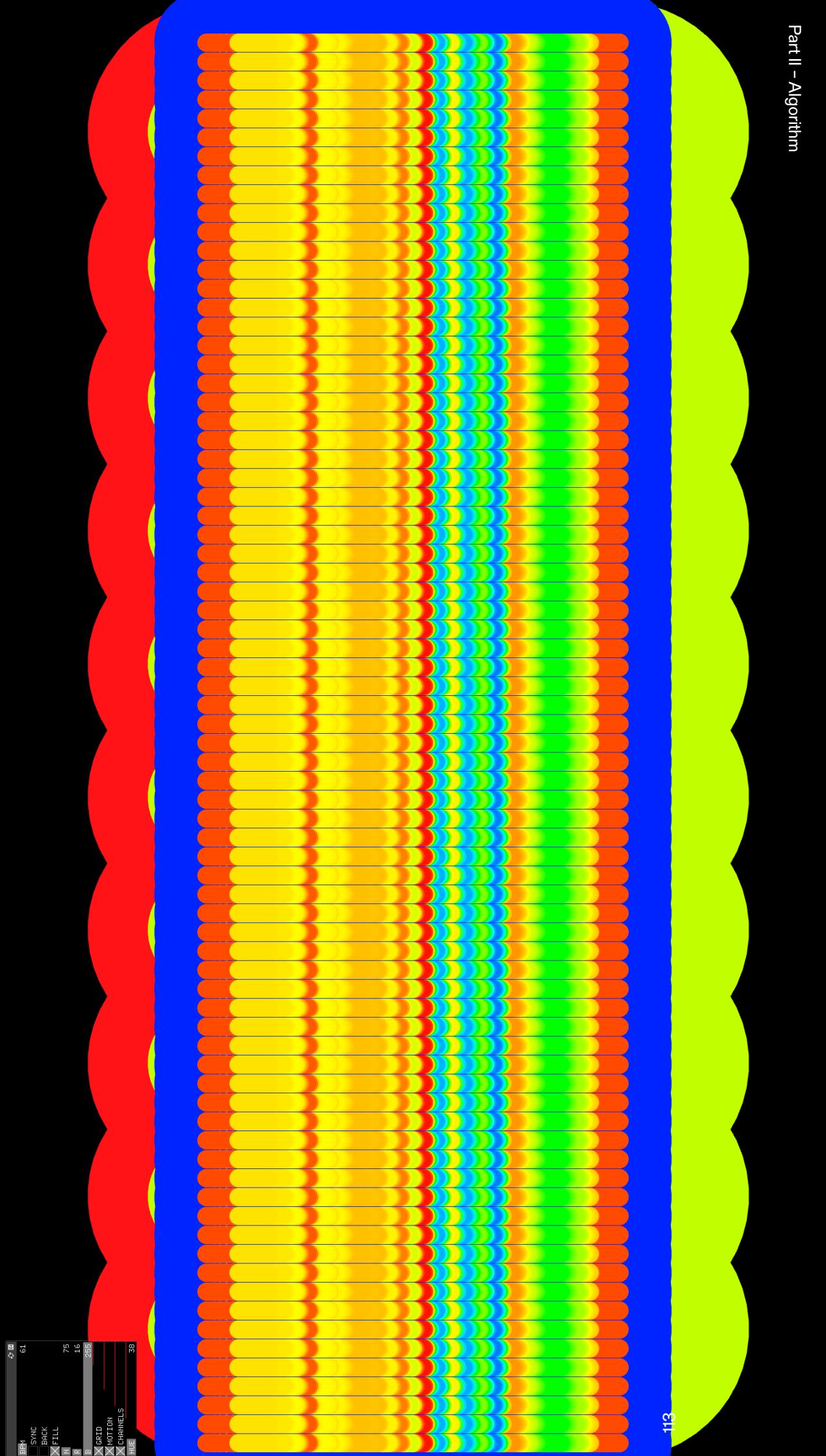
32

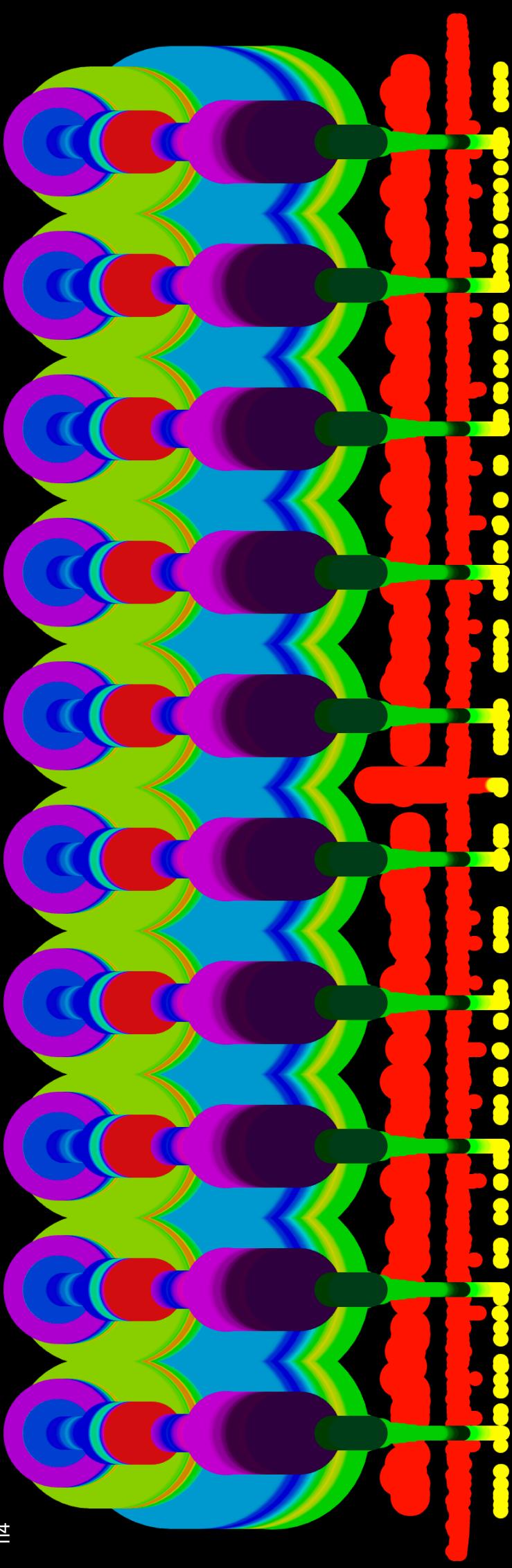


2.11

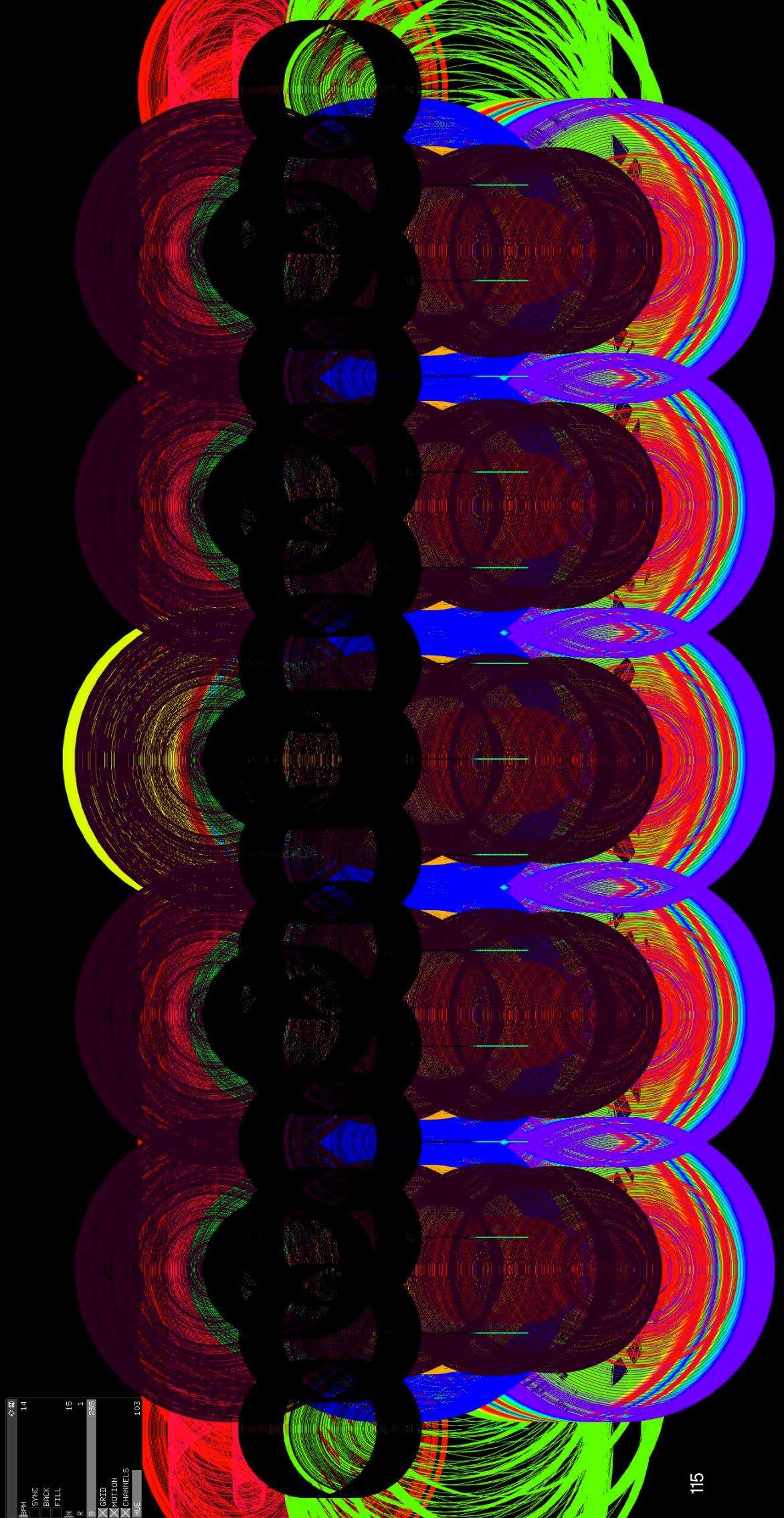
Graphical examples

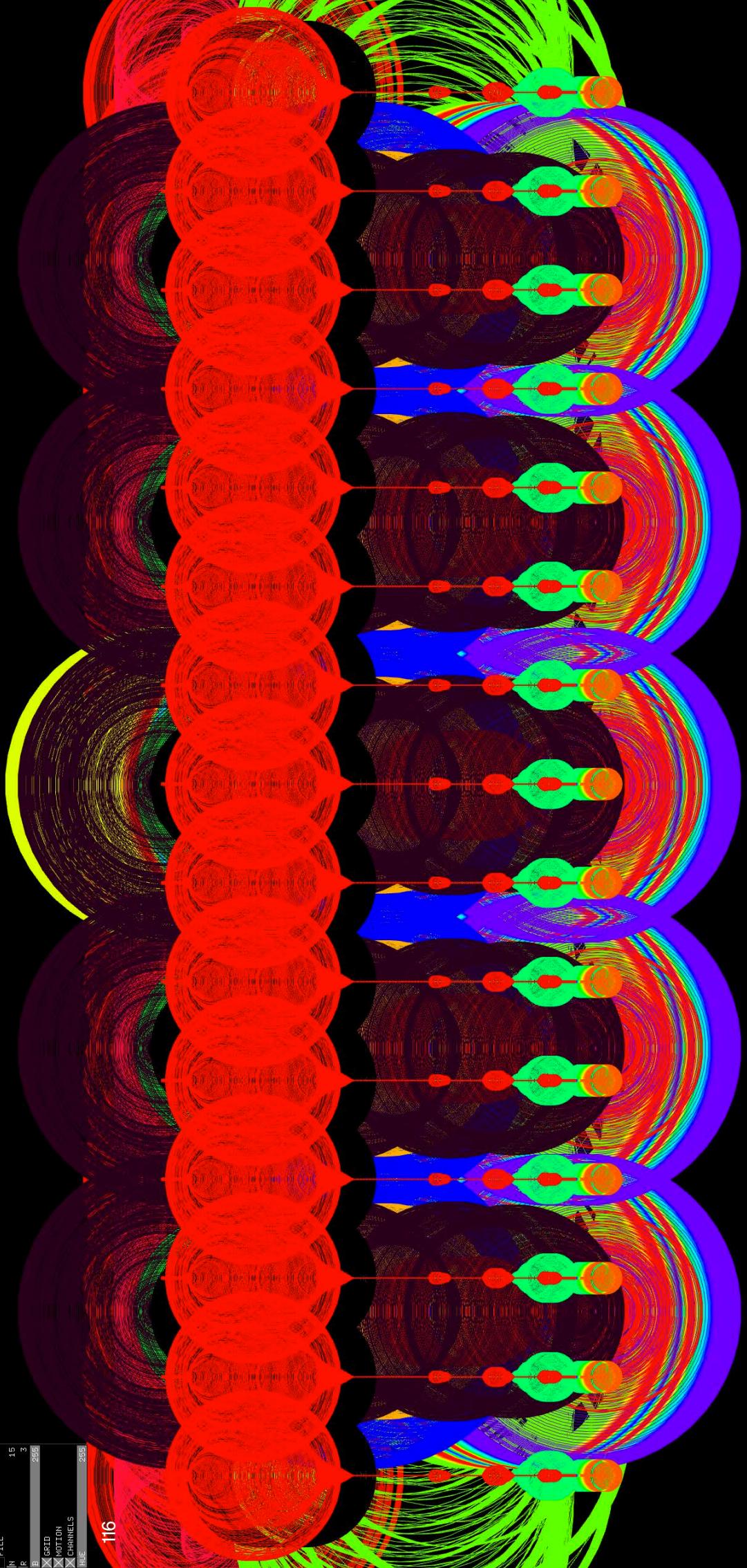
The following part of the book is a collection of graphical experiments and results produced by the algorithm mostly using disabled background option. The reason for that is I believe that this is an outcome on its own which needs to be documented and shown even though it does not straightforwardly match to the primary definition of the controller working as a substitute for the lightning equipment. However, this is definitely a powerful instrument for image generation, and its infinite variety of outcomes cannot be shown even on a scale of the next pages.

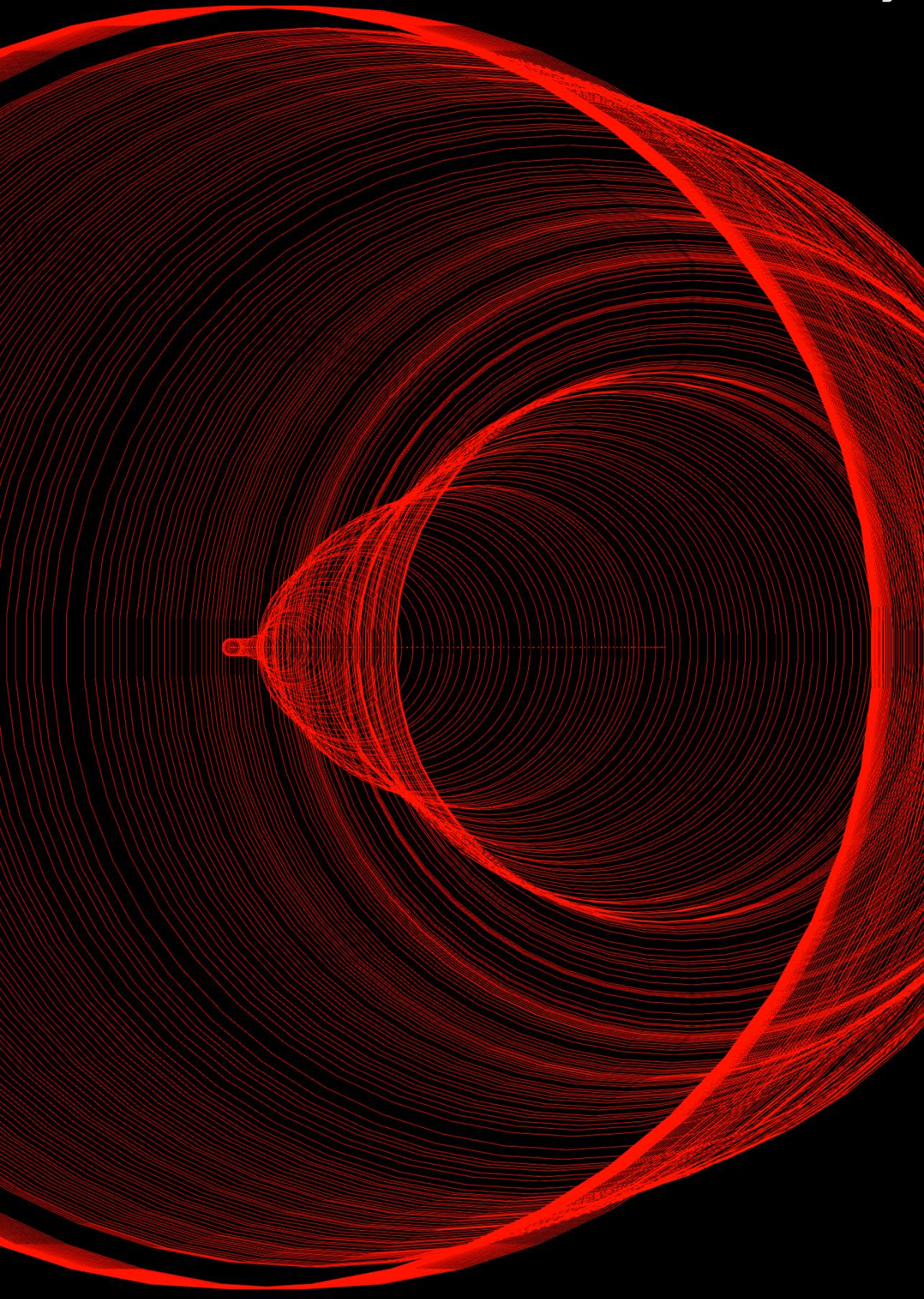




6
IBPM
SYNC
BACK
FILL
IN
R
B
X GRID
X MOTION
X CHANNELS
HUE
102

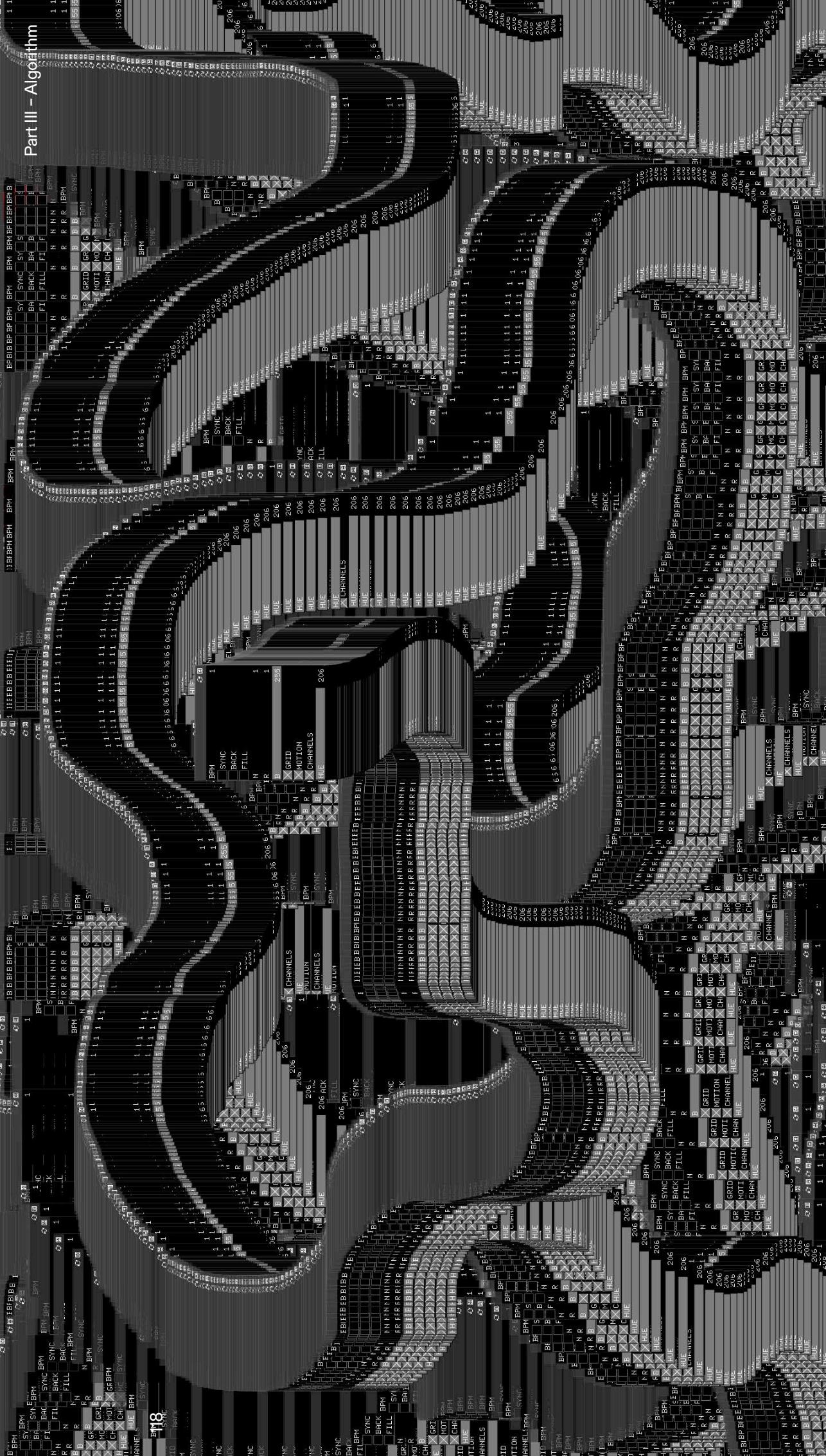




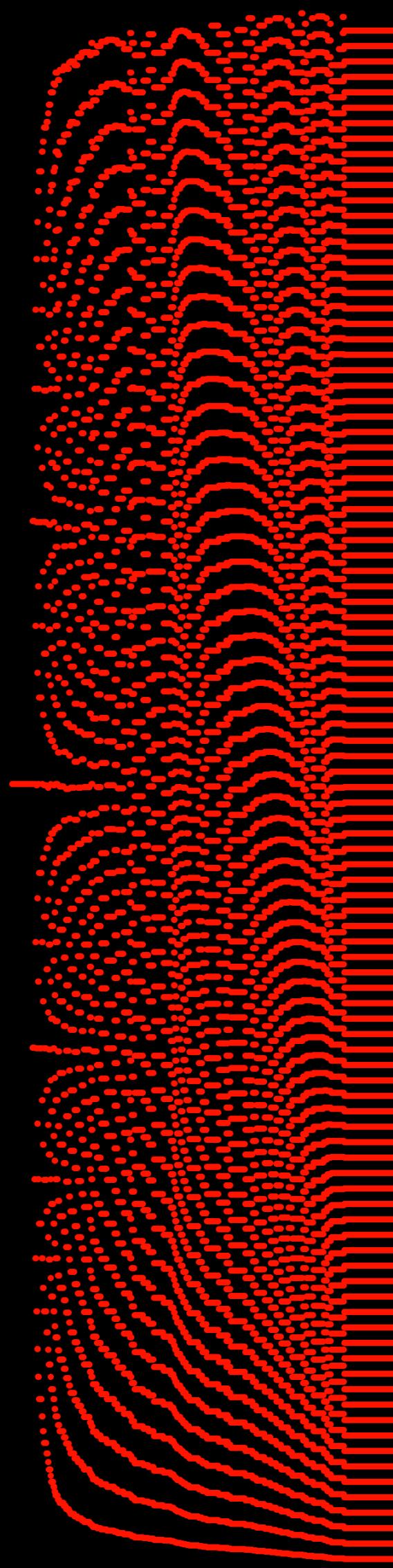


	BBEMC	SINC	BLOCK	FFT	N	P	R	C	GID	MOTION	CHANNELS	HUE
133												
	255											
		255										
			255									
				1								
					1							
						1						
							1					
								1				
									1			
										1		
											1	
												1

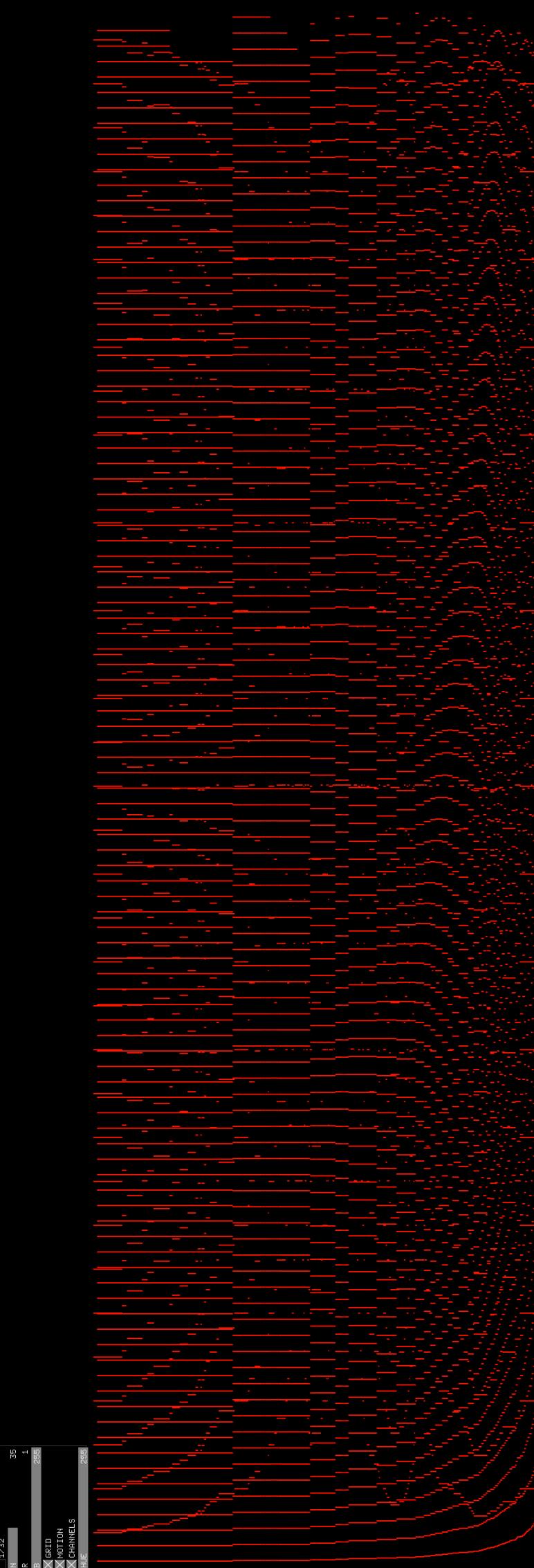
Part III – Algorithm

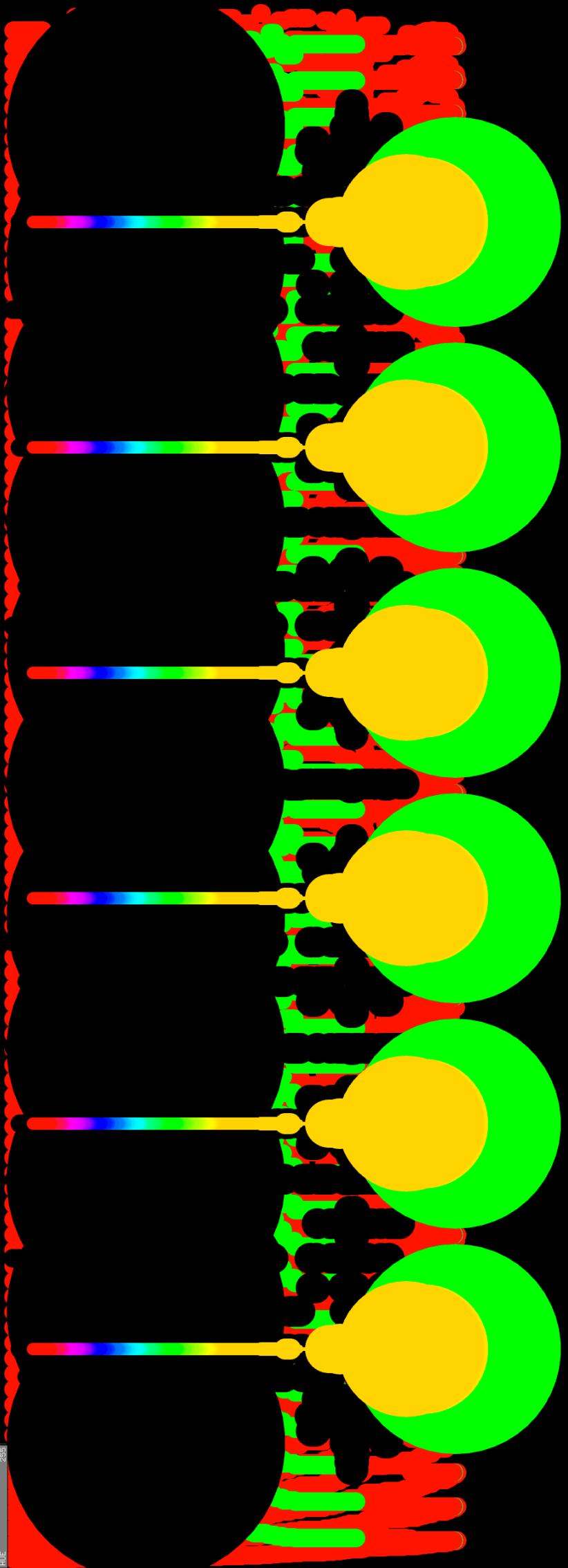


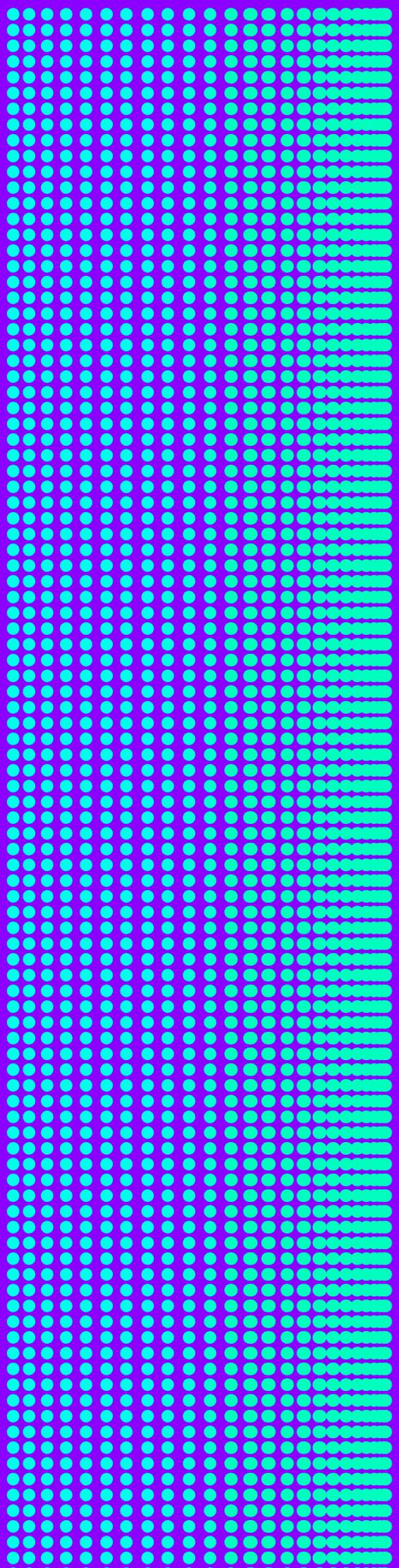




SP-L-DYN-2020
17
SPM
SYNC
BACK
1
1.2
1.4
1.6
1.8
1.16
1.32
N
B
GRID
MOTION
CHANNELS
TIME
120
250
255

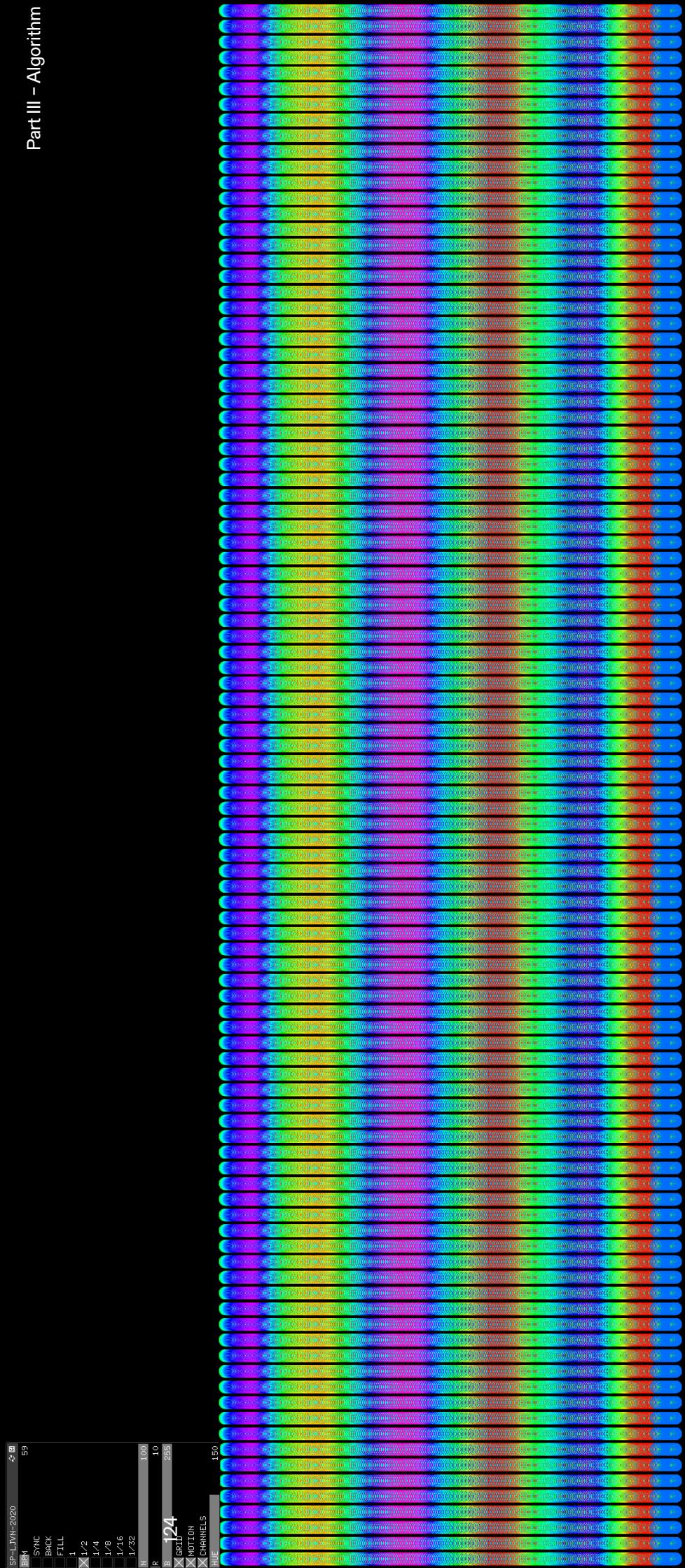


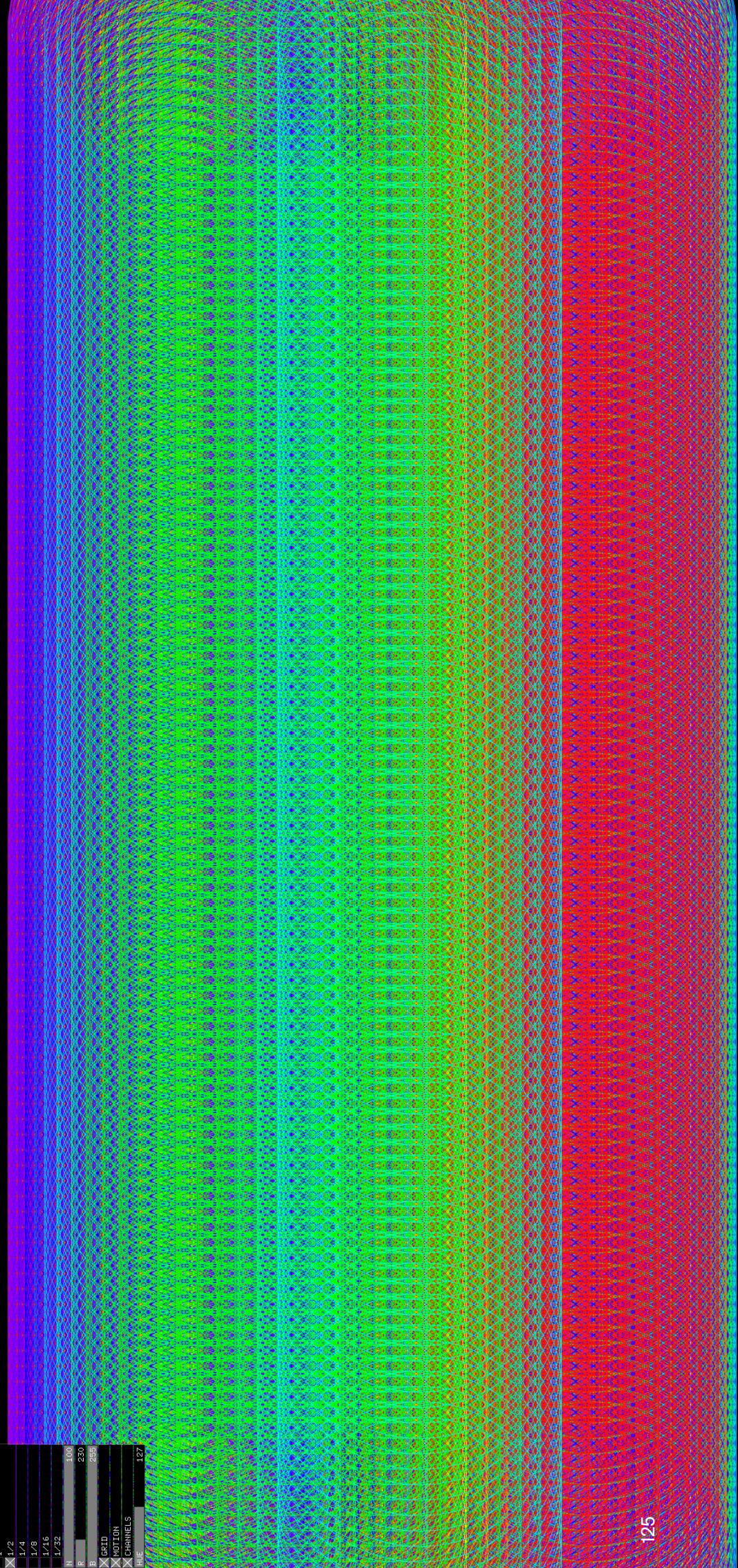


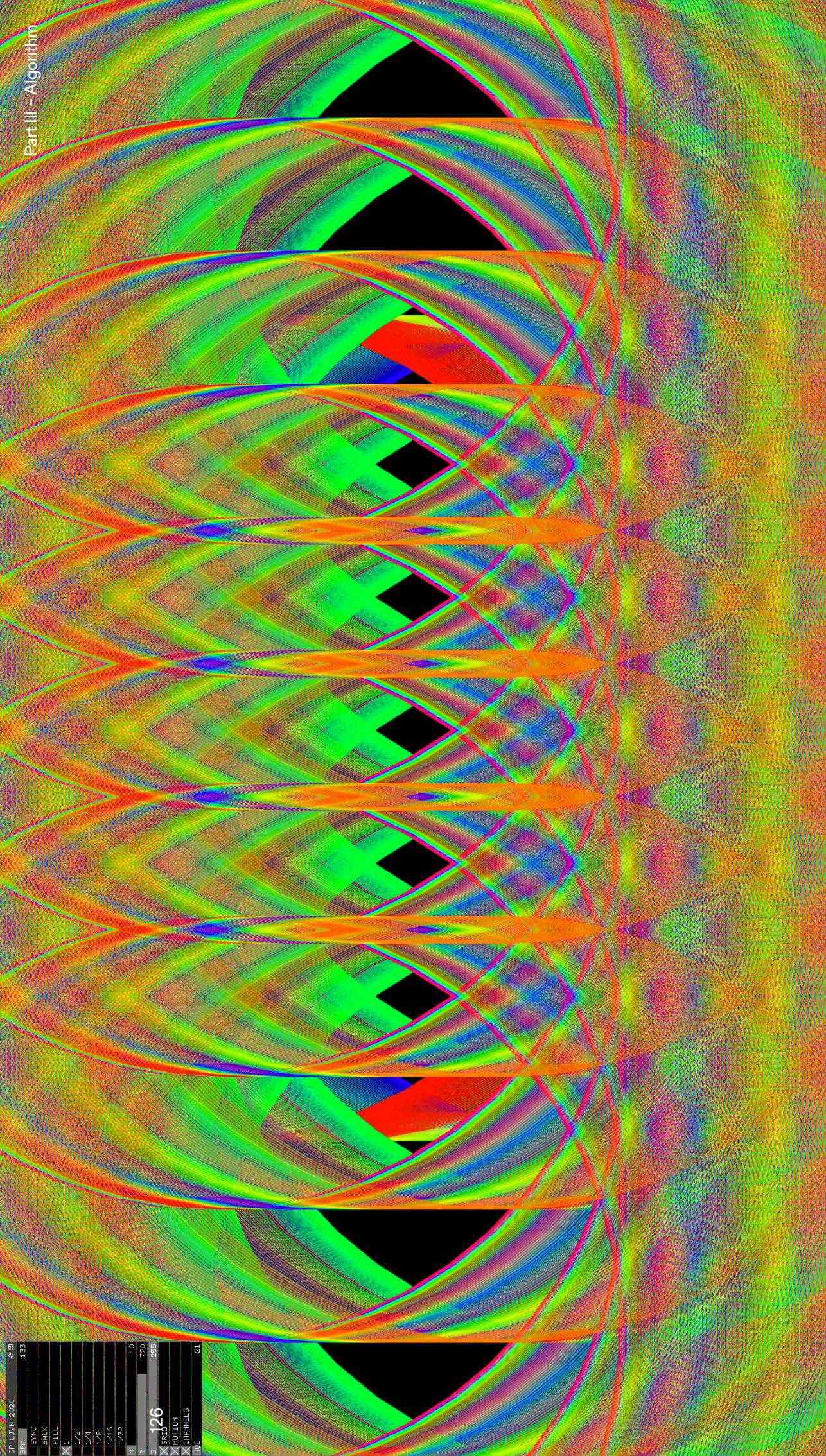


2020-07-07
556
Sonic
BLOCK
FILE
FORMAT
CHANNELS
HUE
117

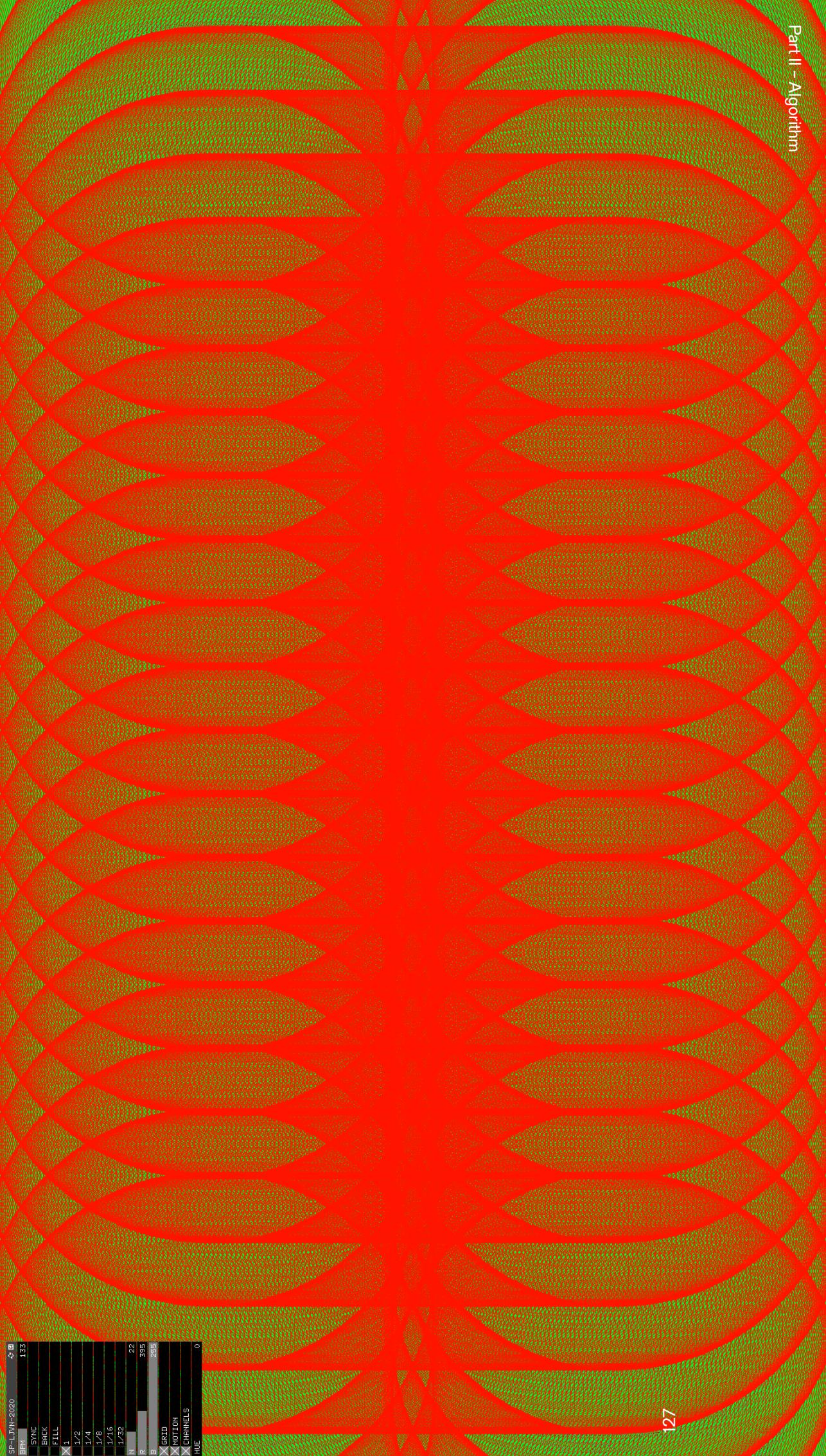
2020-07-07
556
Sonic
BLOCK
FILE
FORMAT
CHANNELS
HUE
117



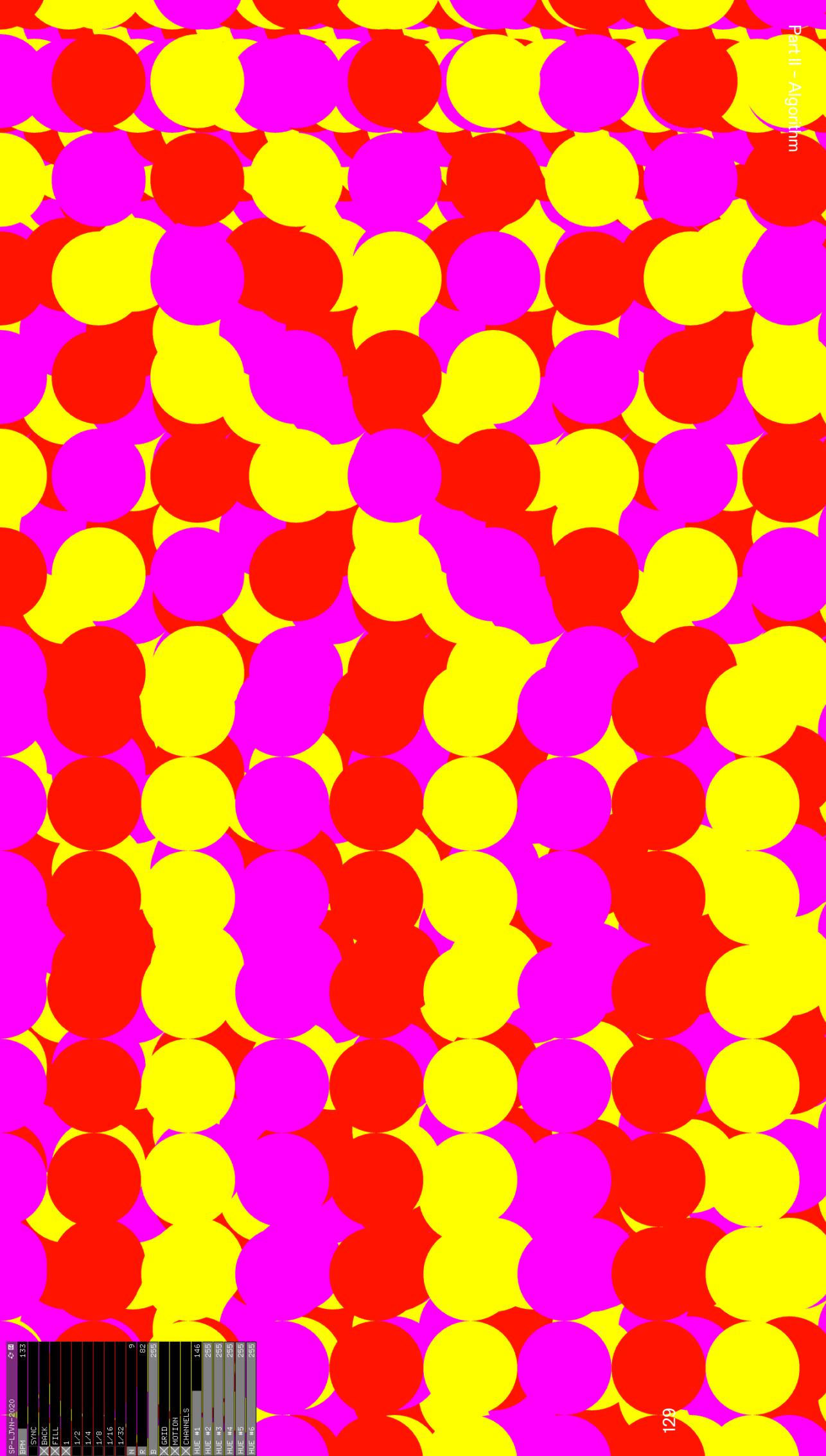




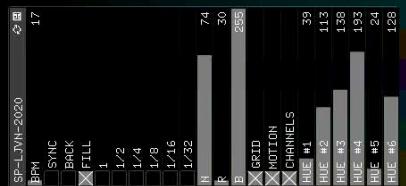
SP-DNN-2020
BPM: 4.33
SYNC
BACK
FILL
1.2
1.4
1.8
1.16
1.16
1.32
N: 10
P: 720
B: **126**
GRID: 55
MOTION
CHANNELS: 21



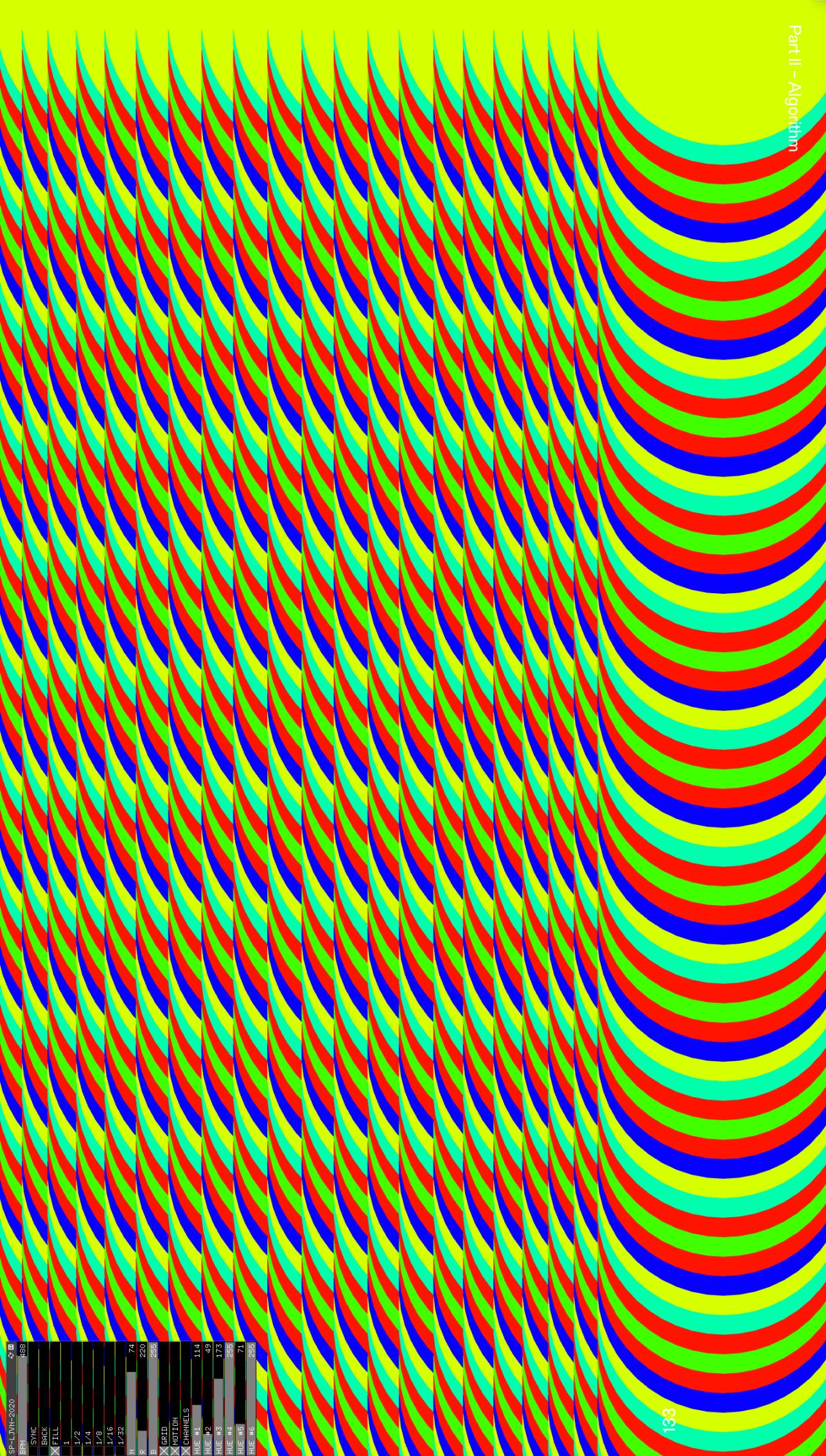


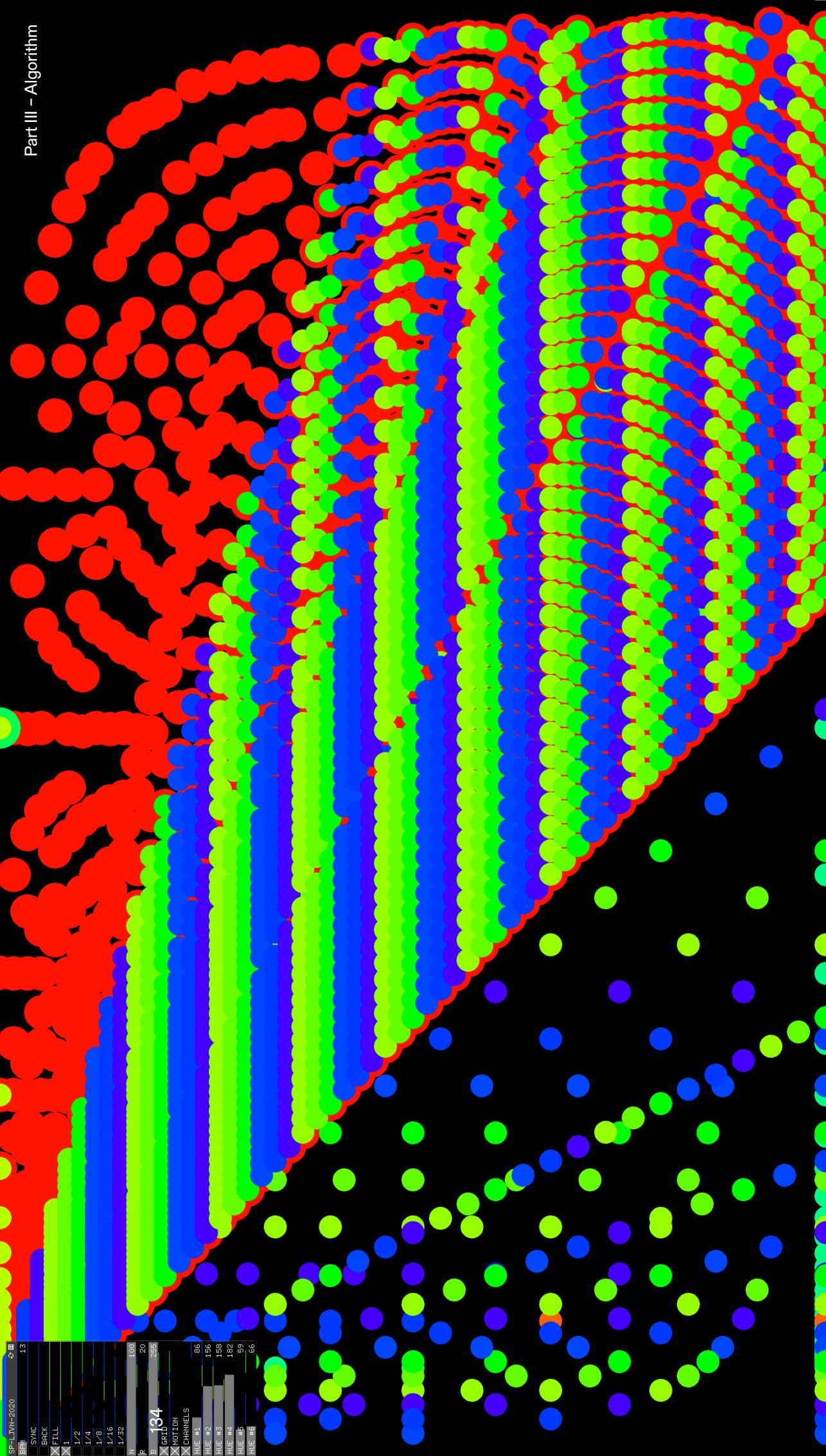


SP-L-DYN-2020	17
SYNC	
BACK	
FILL	
1.1.2	
1.1.4	
1.1.8	
1.1.16	
1.1.32	
N	60
P	30
B	130
GND	0
CHANNELS	
HUE #1	11.4
HUE #2	11.3
HUE #3	11.3
HUE #4	11.3
HUE #5	11.3
HUE #6	11.4
HUE #7	11.4



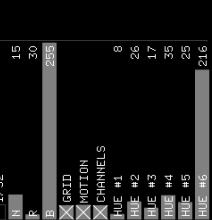




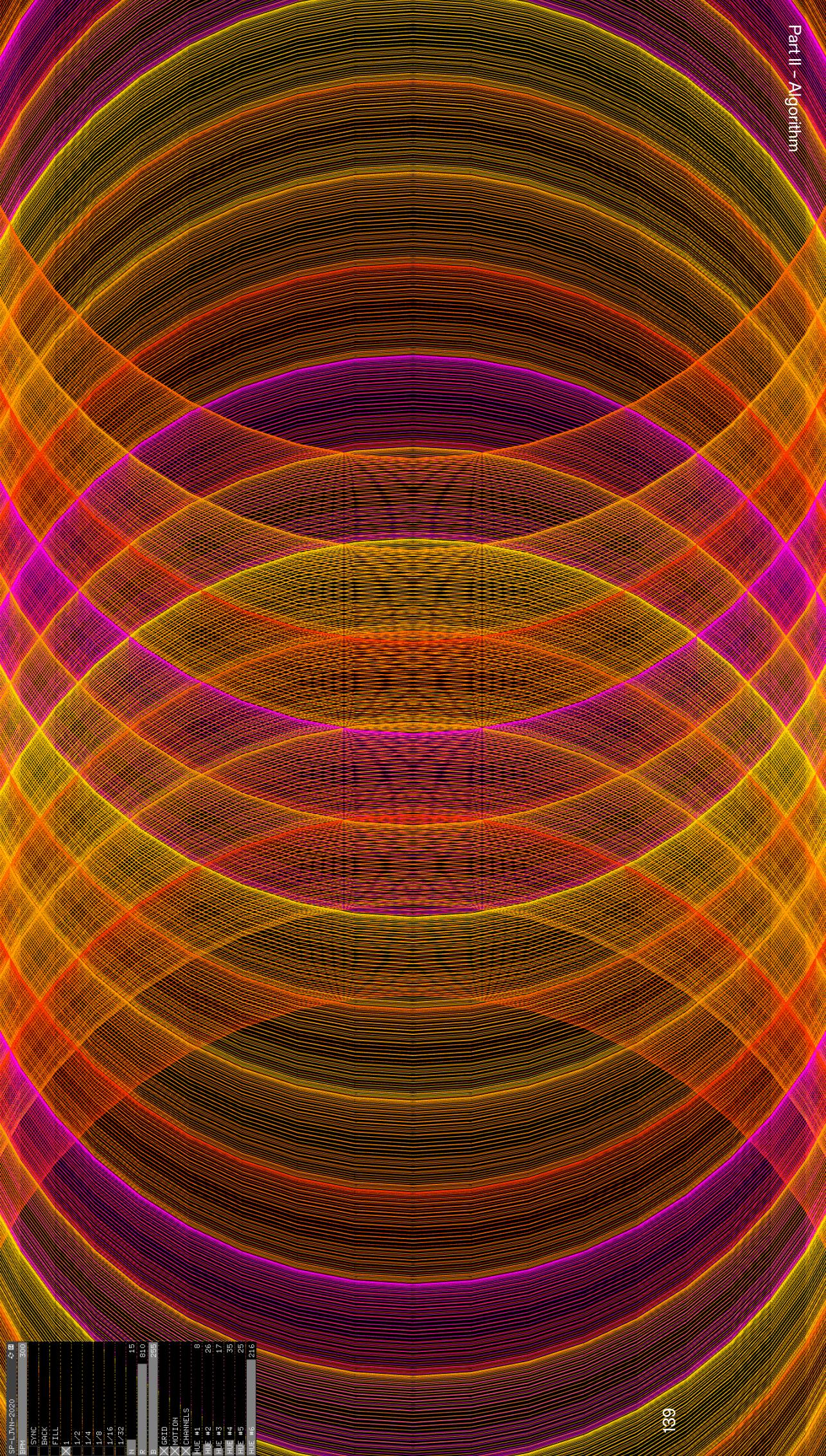




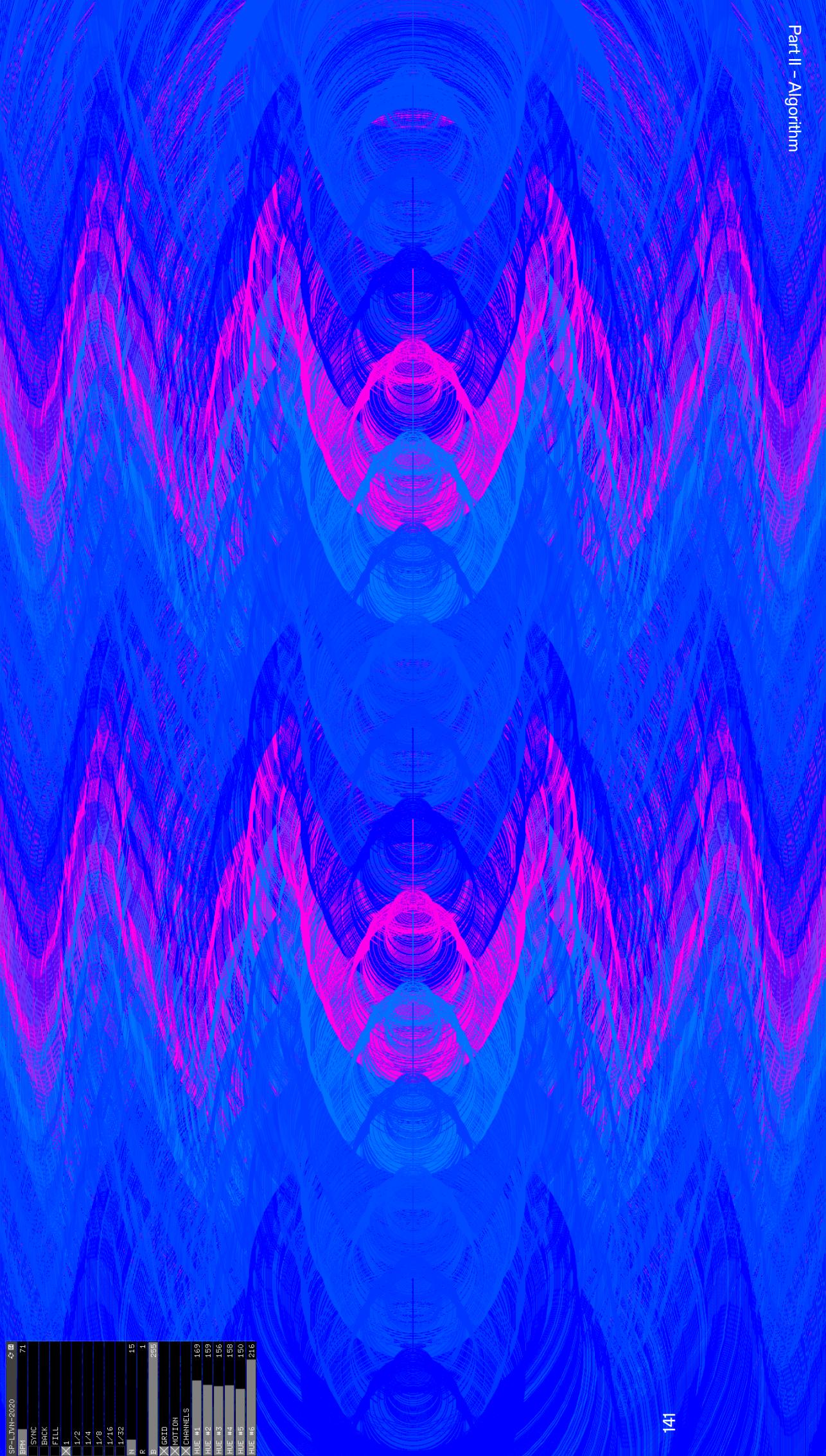


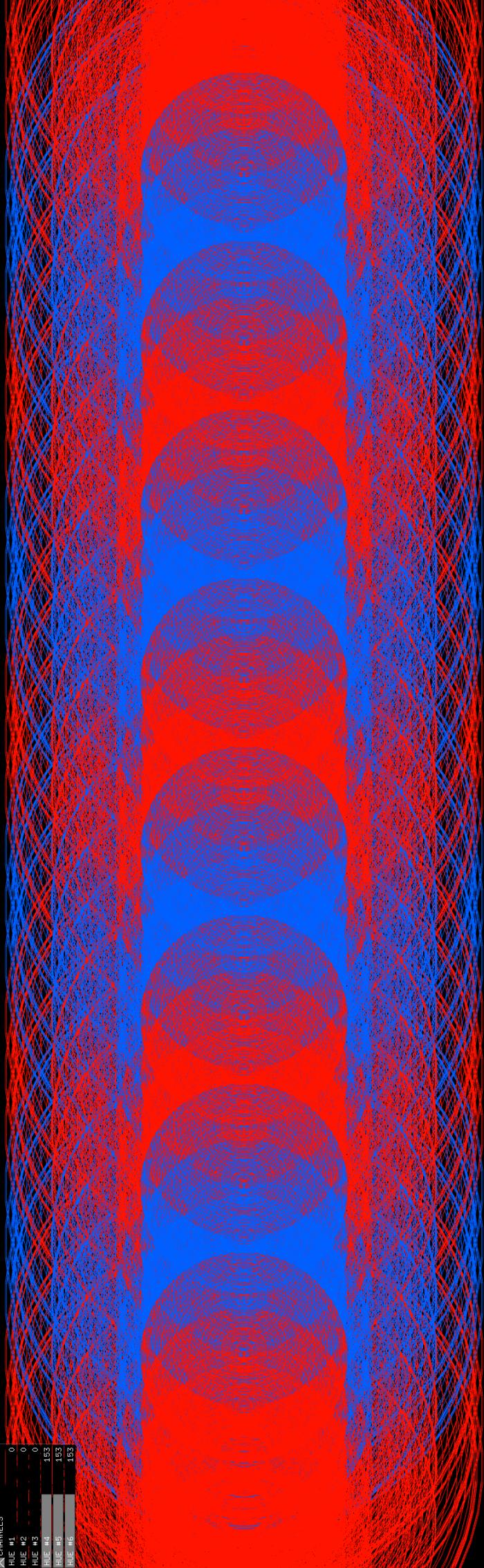


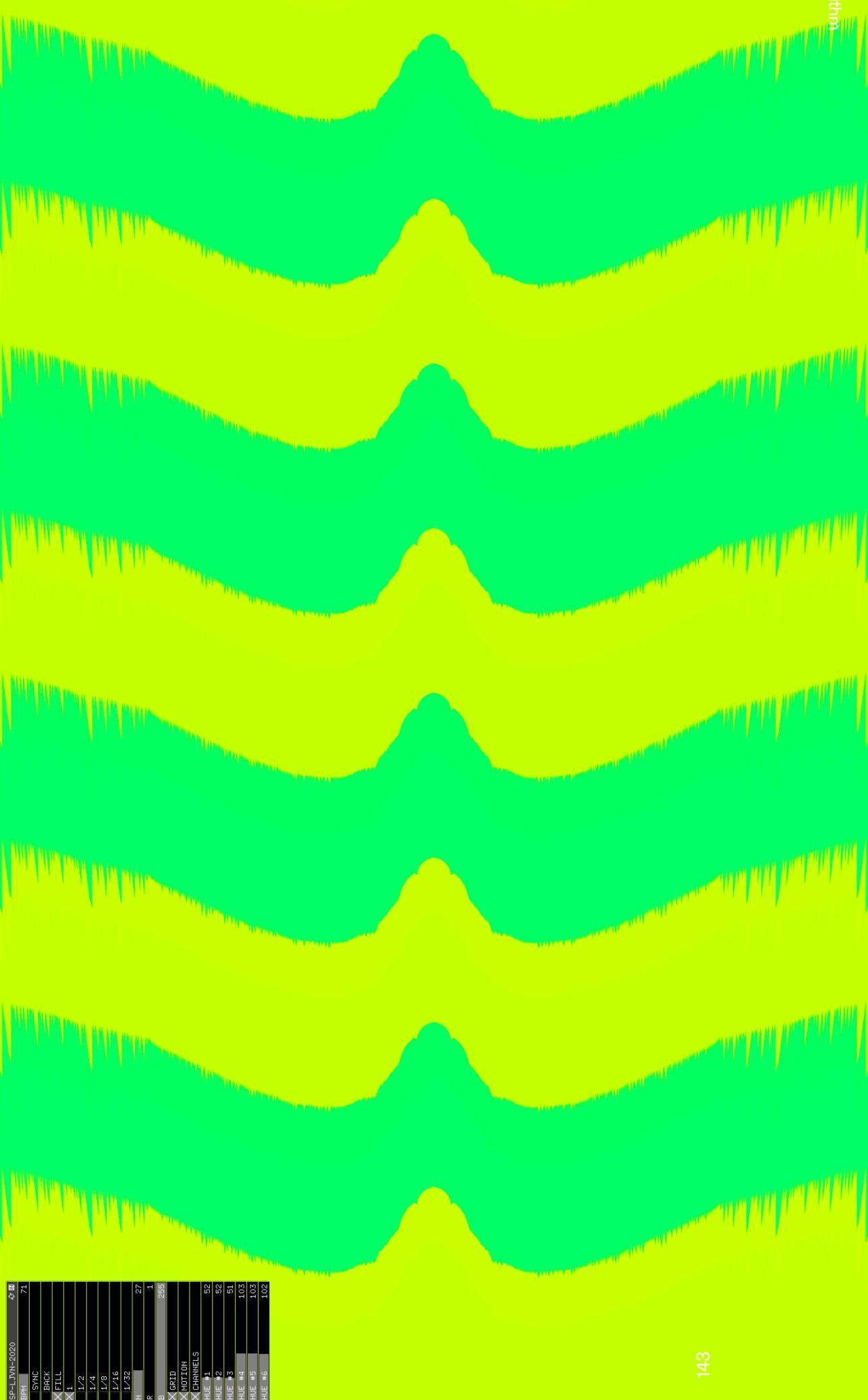


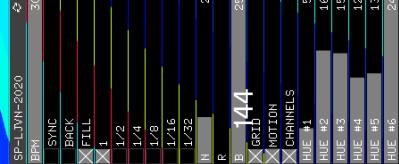


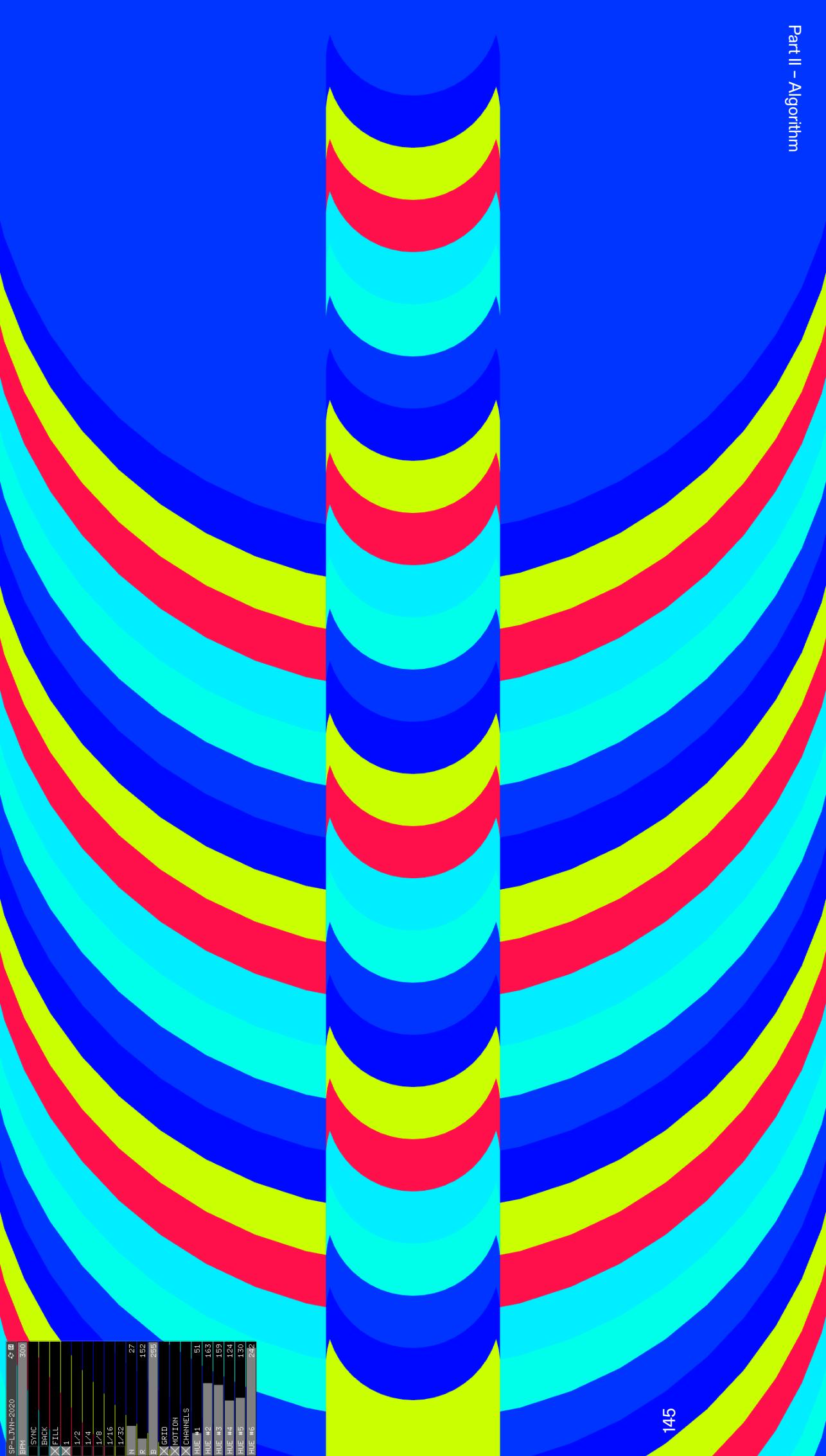




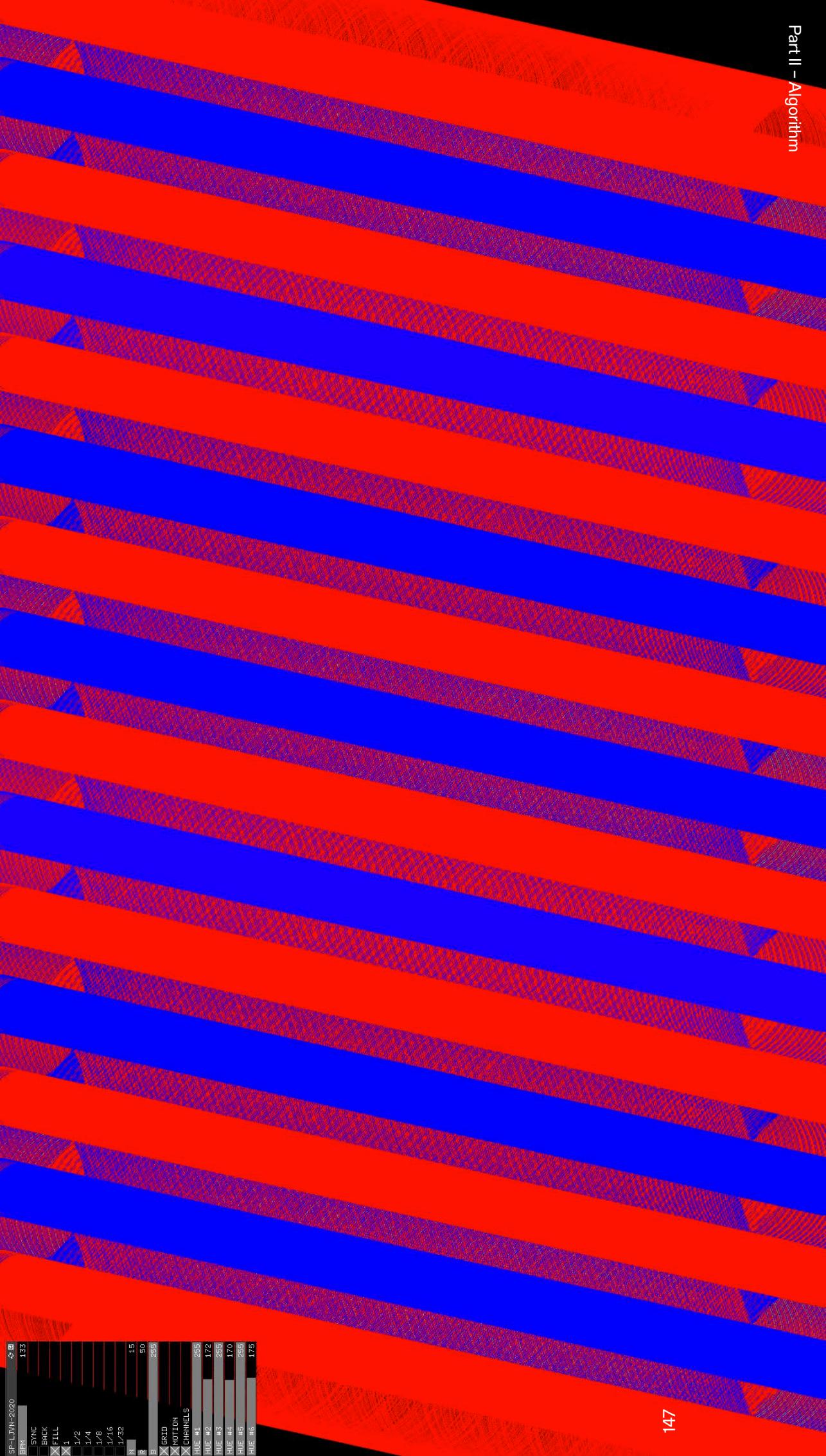




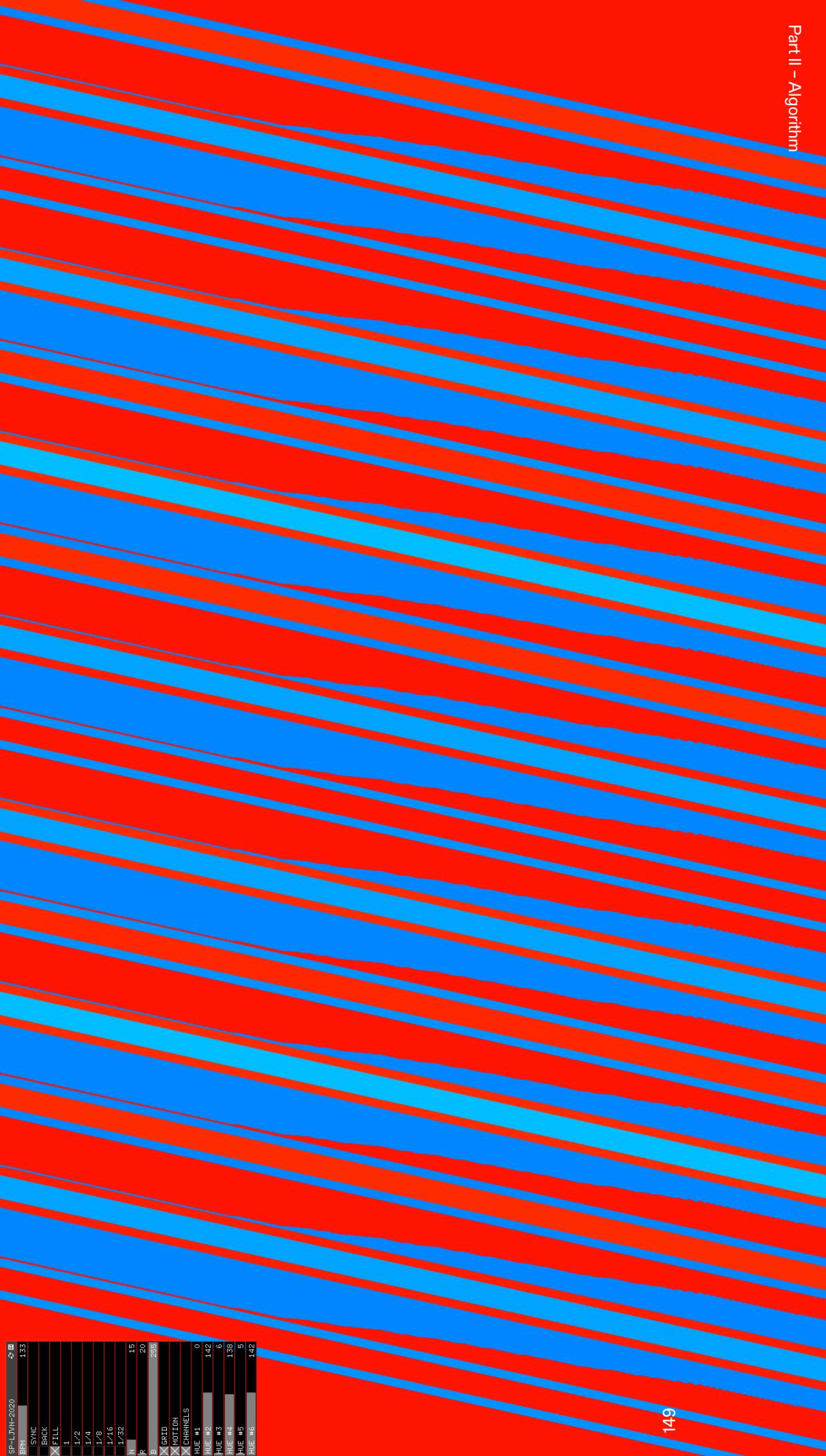




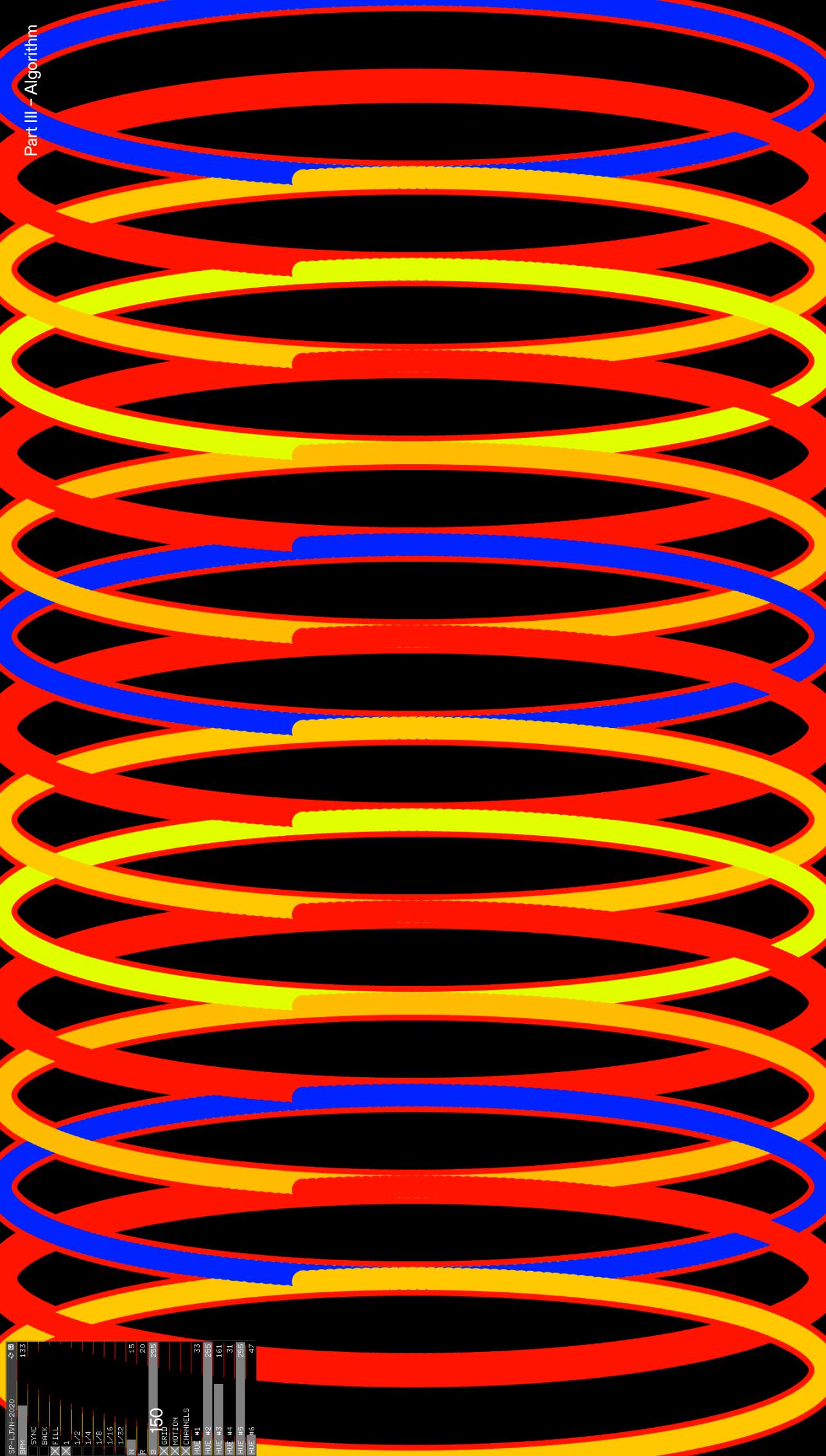


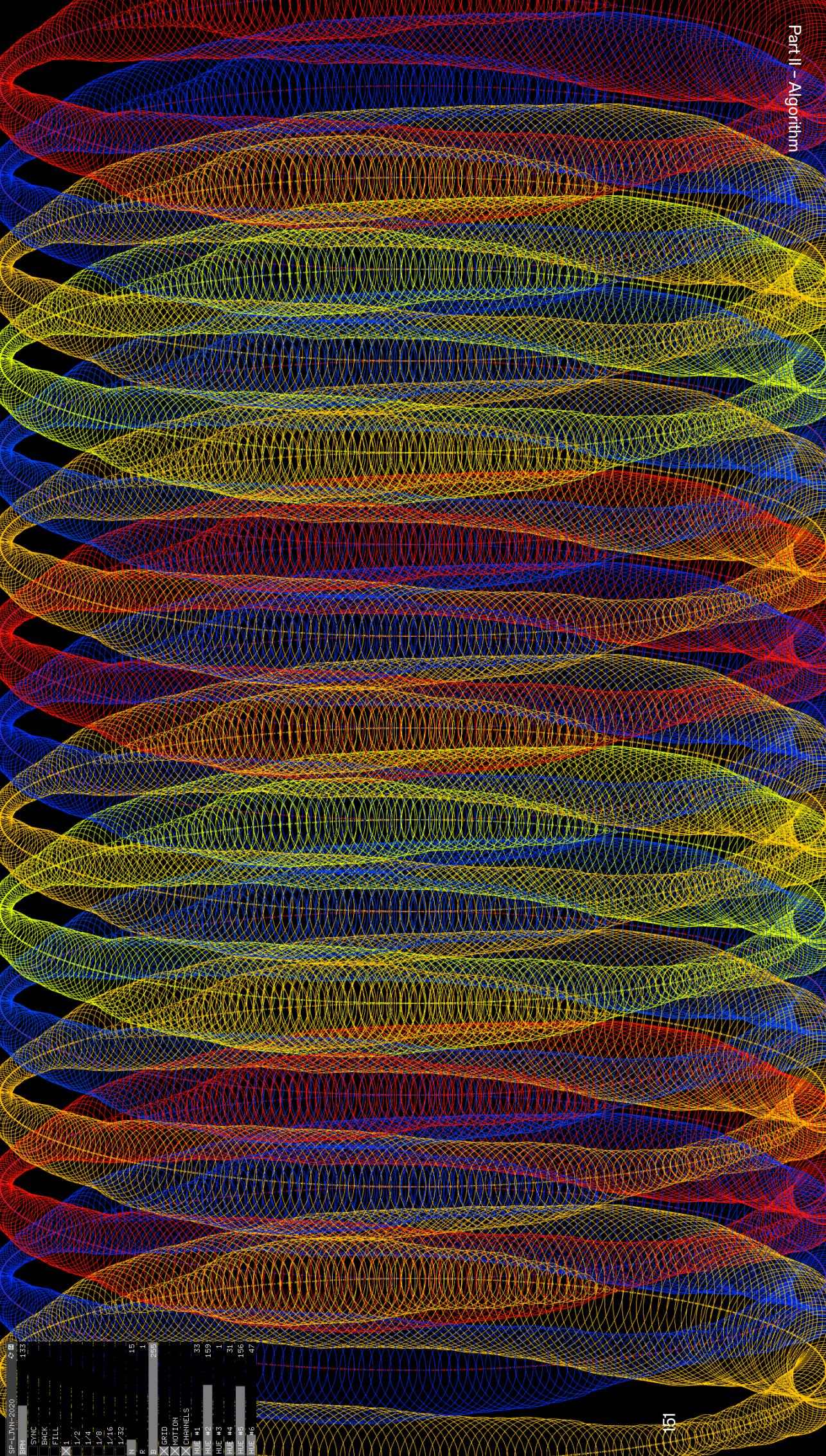




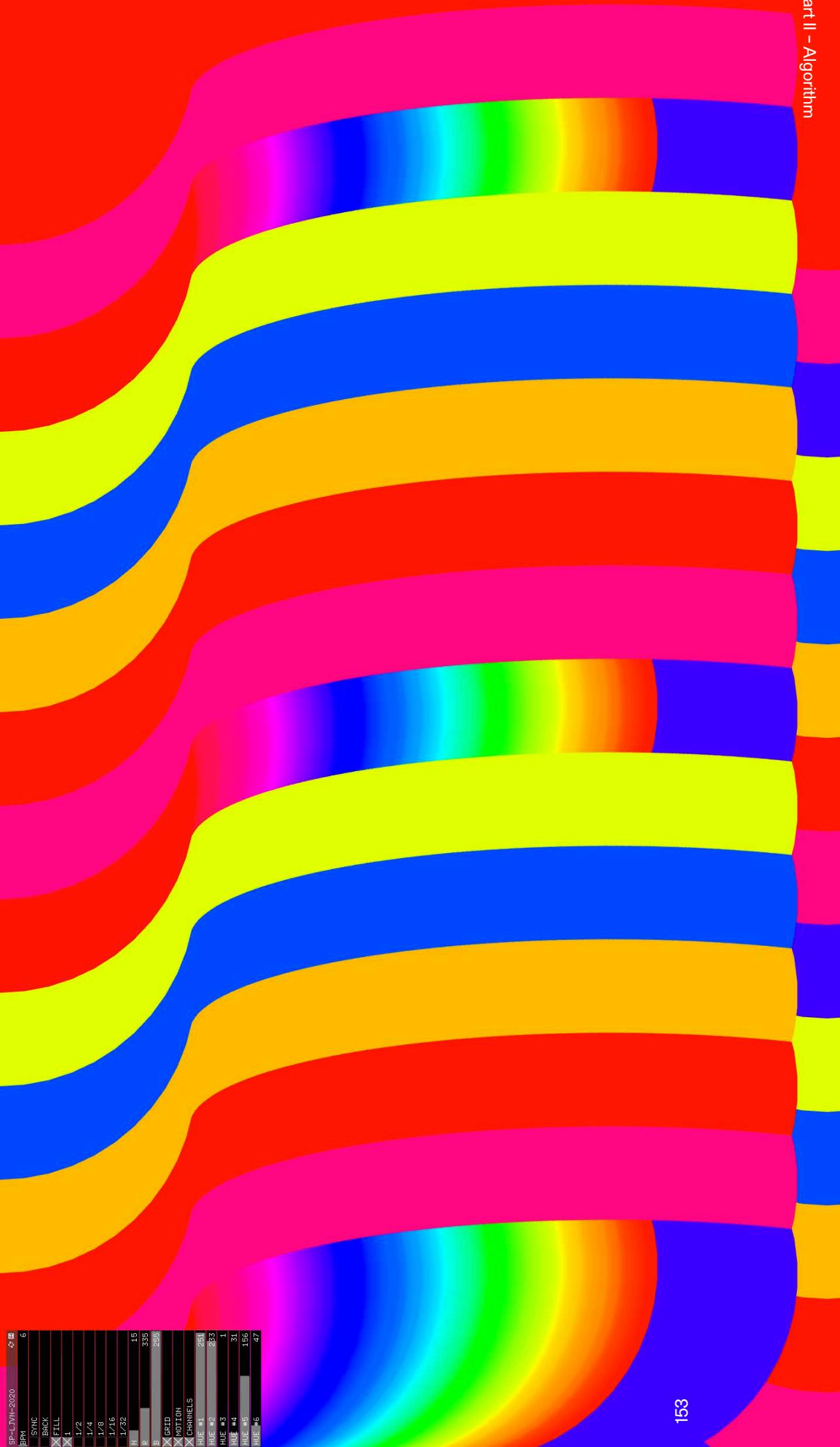


SP-LDNH-2020	133
SYNC	255
GRID	1510
MOTION	116
CHANNELS	142
HUE	#1 0, #2 142, #3 6, #4 138, #5 5, #6 142
MODE	142

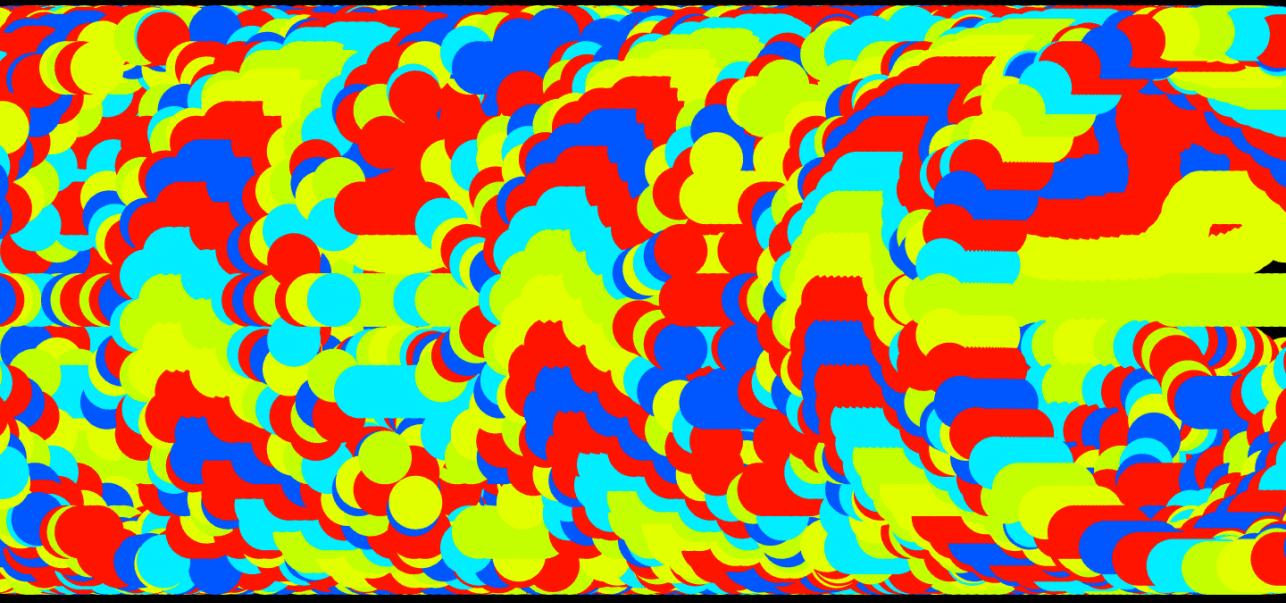




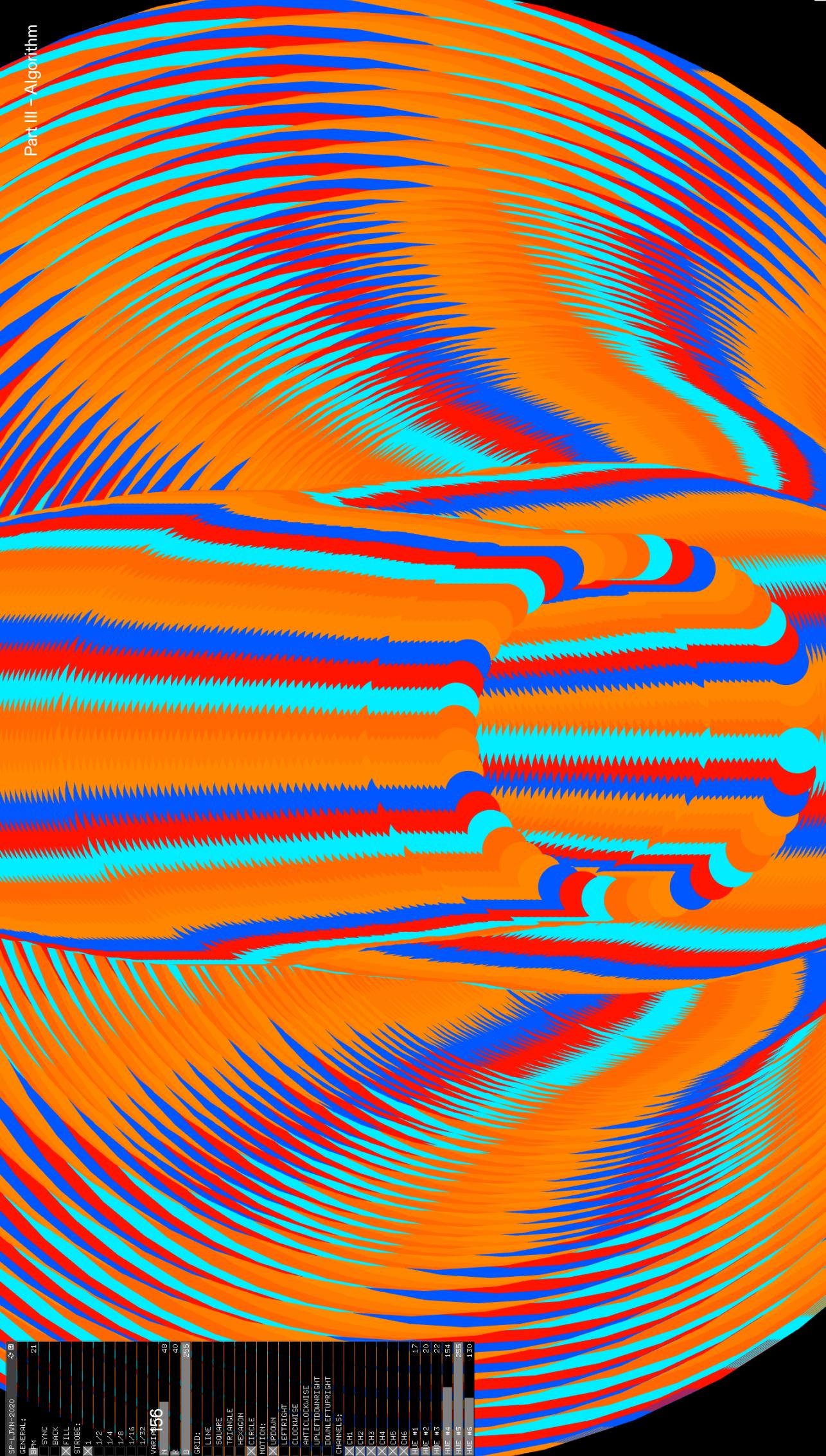


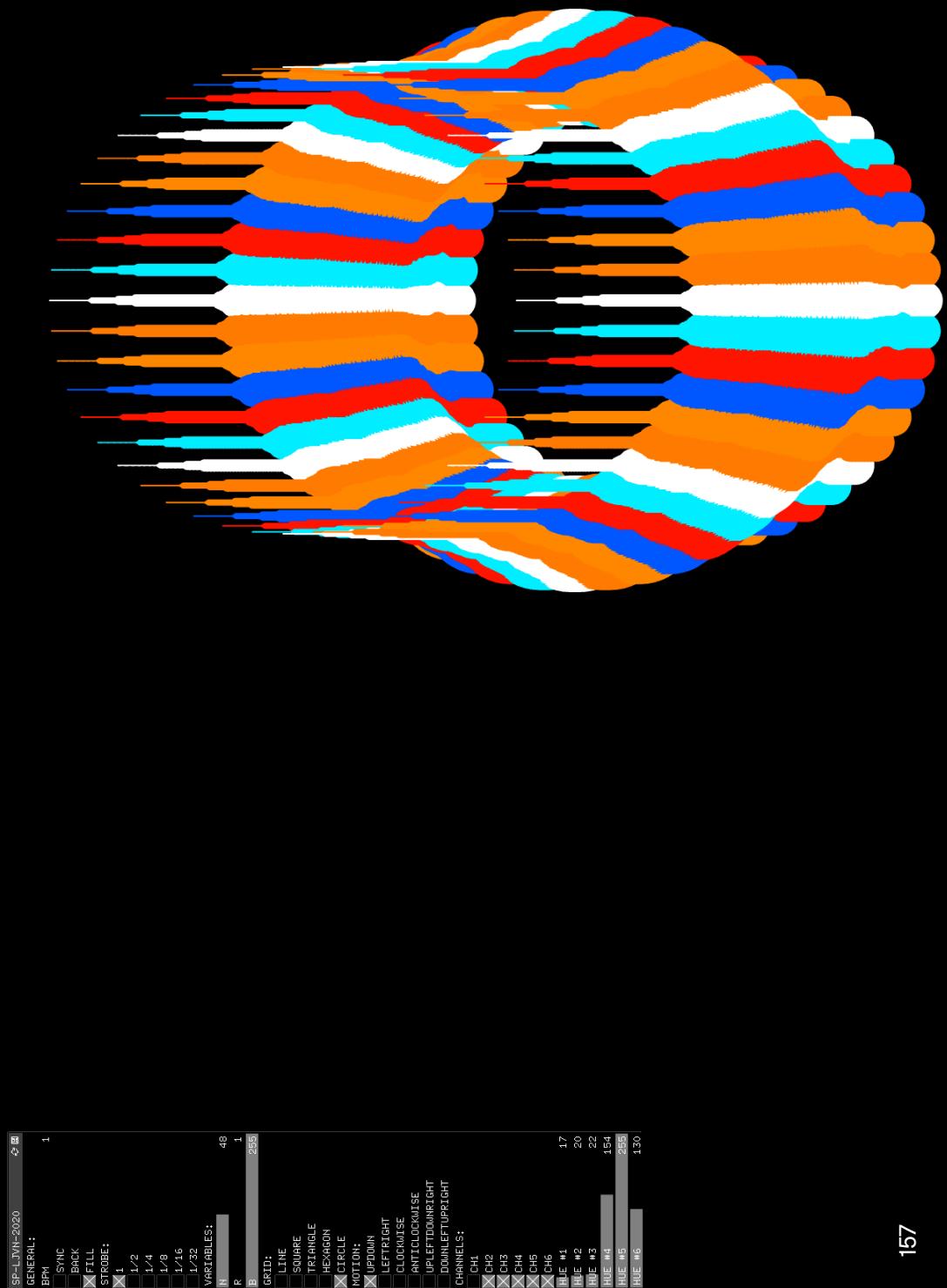


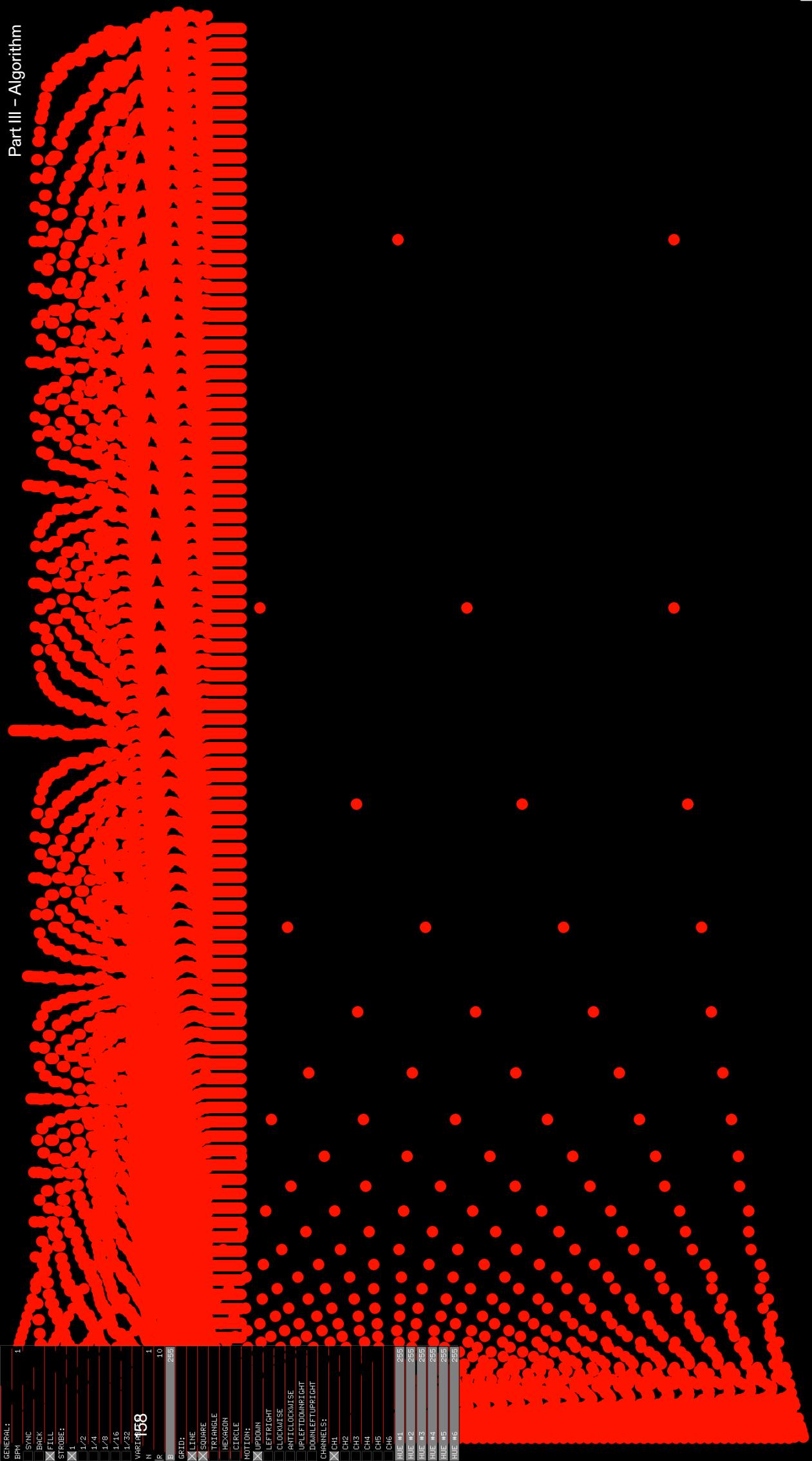


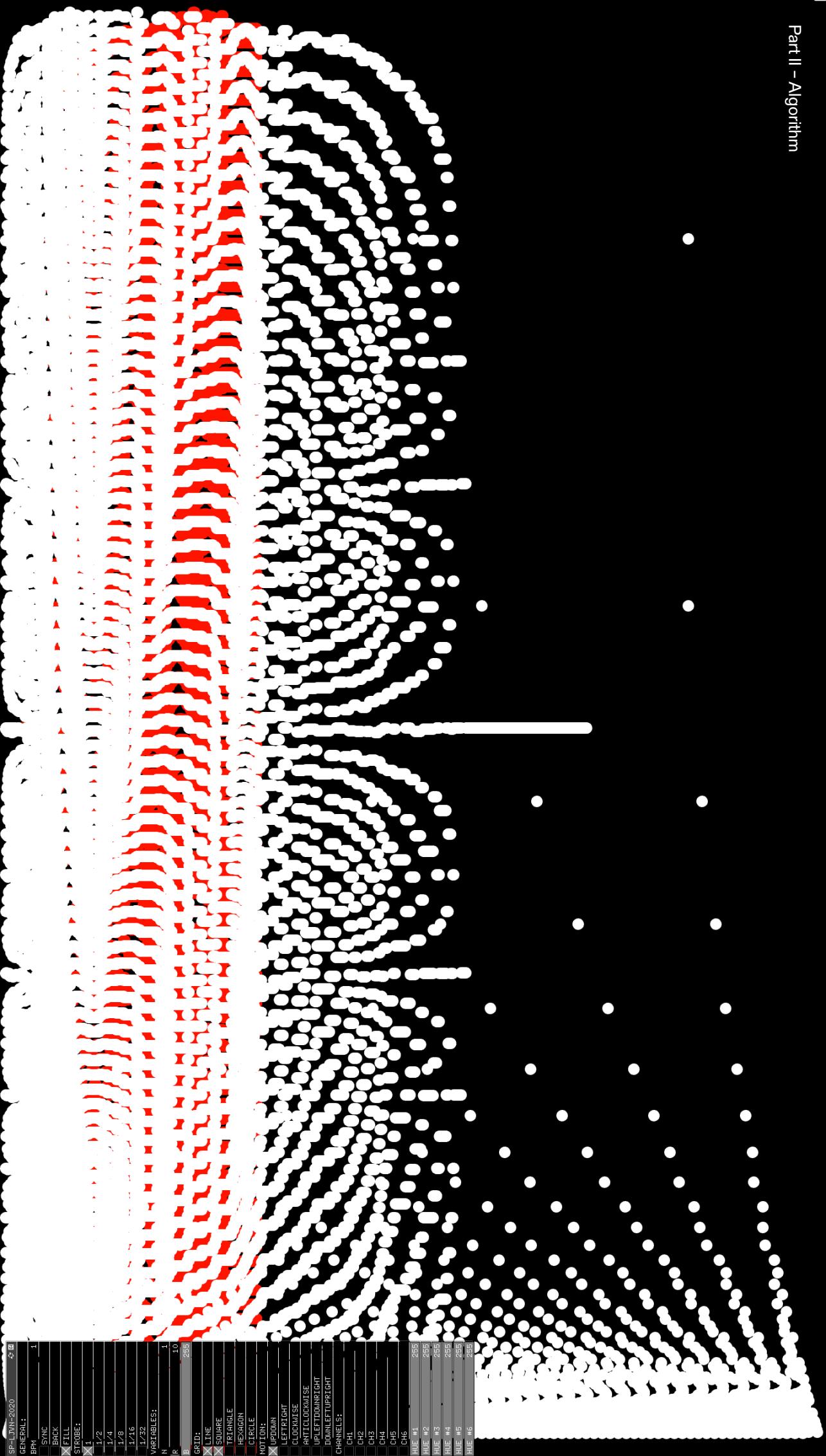


SP-LVHN-2020
GENERAL:
SYNCS
BLOCK
FILL
STROBE:
X 1
1..2
1..4
1..8
1..16
1..32
VARIABLES:
IN 48
B 30
P 255
GETIO:
LINE
SQUARE
TRIANGLE
HEXAGON
CIRCLE
ROTATION:
LEFT/RIGHT
CLOCKWISE
ANTICLOCKWISE
UP/LEFT/DOWN/RIGHT
DOWN/LEFT/UP/RIGHT
CHANNELS:
C1
C2
C3
C4
C5
C6
C7
C8
HUE #1
HUE #2
HUE #3
HUE #4
HUE #5
HUE #6
HUE #7
HUE #8
HUE #9
HUE #10
HUE #11
HUE #12
HUE #13
HUE #14
HUE #15
HUE #16
HUE #17
HUE #18
HUE #19
HUE #20
HUE #21
HUE #22
HUE #23
HUE #24
HUE #25
HUE #26
HUE #27
HUE #28
HUE #29
HUE #30
HUE #31
HUE #32
HUE #33
HUE #34
HUE #35
HUE #36
HUE #37
HUE #38
HUE #39
HUE #40
HUE #41
HUE #42
HUE #43
HUE #44
HUE #45
HUE #46
HUE #47
HUE #48
HUE #49
HUE #50
HUE #51
HUE #52
HUE #53
HUE #54
HUE #55
HUE #56
HUE #57
HUE #58
HUE #59
HUE #60
HUE #61
HUE #62
HUE #63
HUE #64
HUE #65
HUE #66
HUE #67
HUE #68
HUE #69
HUE #70
HUE #71
HUE #72
HUE #73
HUE #74
HUE #75
HUE #76
HUE #77
HUE #78
HUE #79
HUE #80
HUE #81
HUE #82
HUE #83
HUE #84
HUE #85
HUE #86
HUE #87
HUE #88
HUE #89
HUE #90
HUE #91
HUE #92
HUE #93
HUE #94
HUE #95
HUE #96
HUE #97
HUE #98
HUE #99
HUE #100









Part V

SP-LYNH-2020

BRM:

SYNC
 BACK
 FULL

60

1.2

1.4

1.6

1.8

1.16

1.32

1.32

1.5

1.80

2.55

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

1.16

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

160

B

GET:

LINE

TRIANGLE

SQUARE

HEXAGON

CIRCLE

MOTION:

UPDOWN

LEFTRIGHT

CLOCKWISE

ANTICLOCKWISE

UPLEFTDOWNRIGHT

DOWNLLEFTURTRIGHT

CHANNELS:

C14

C12

C13

C14

C15

C16

HUE #1

HUE #2

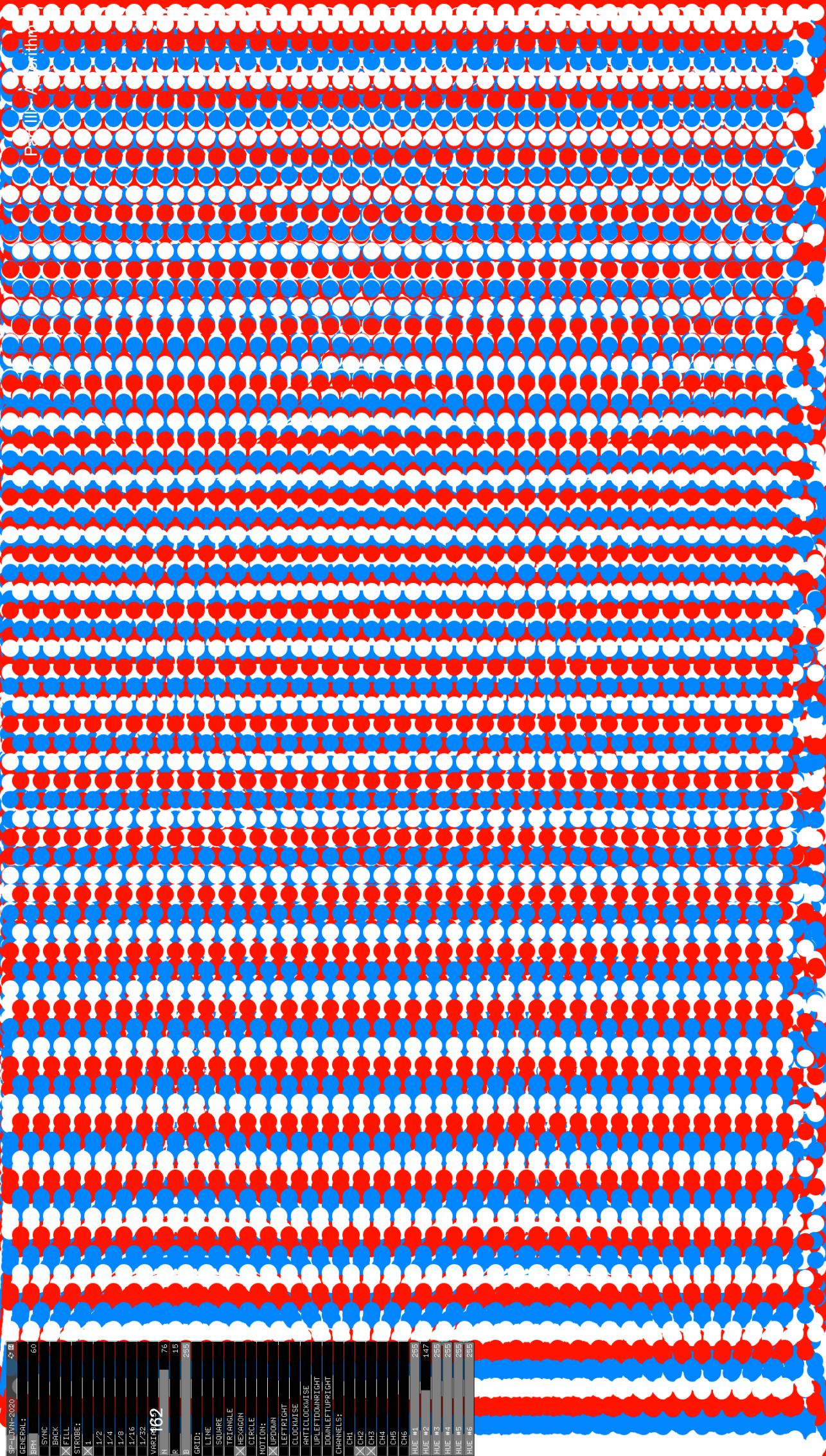
HUE #3

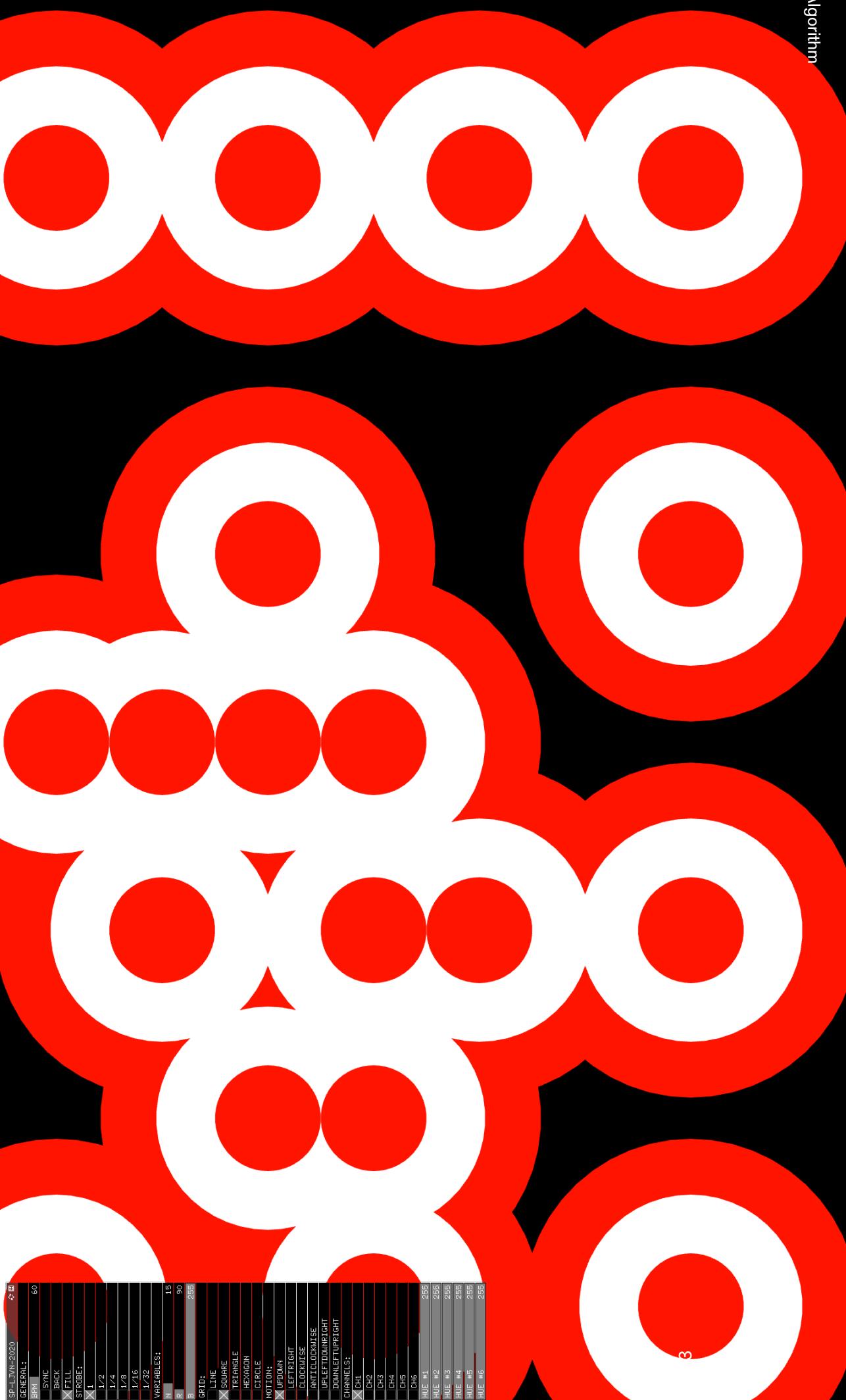
HUE #4

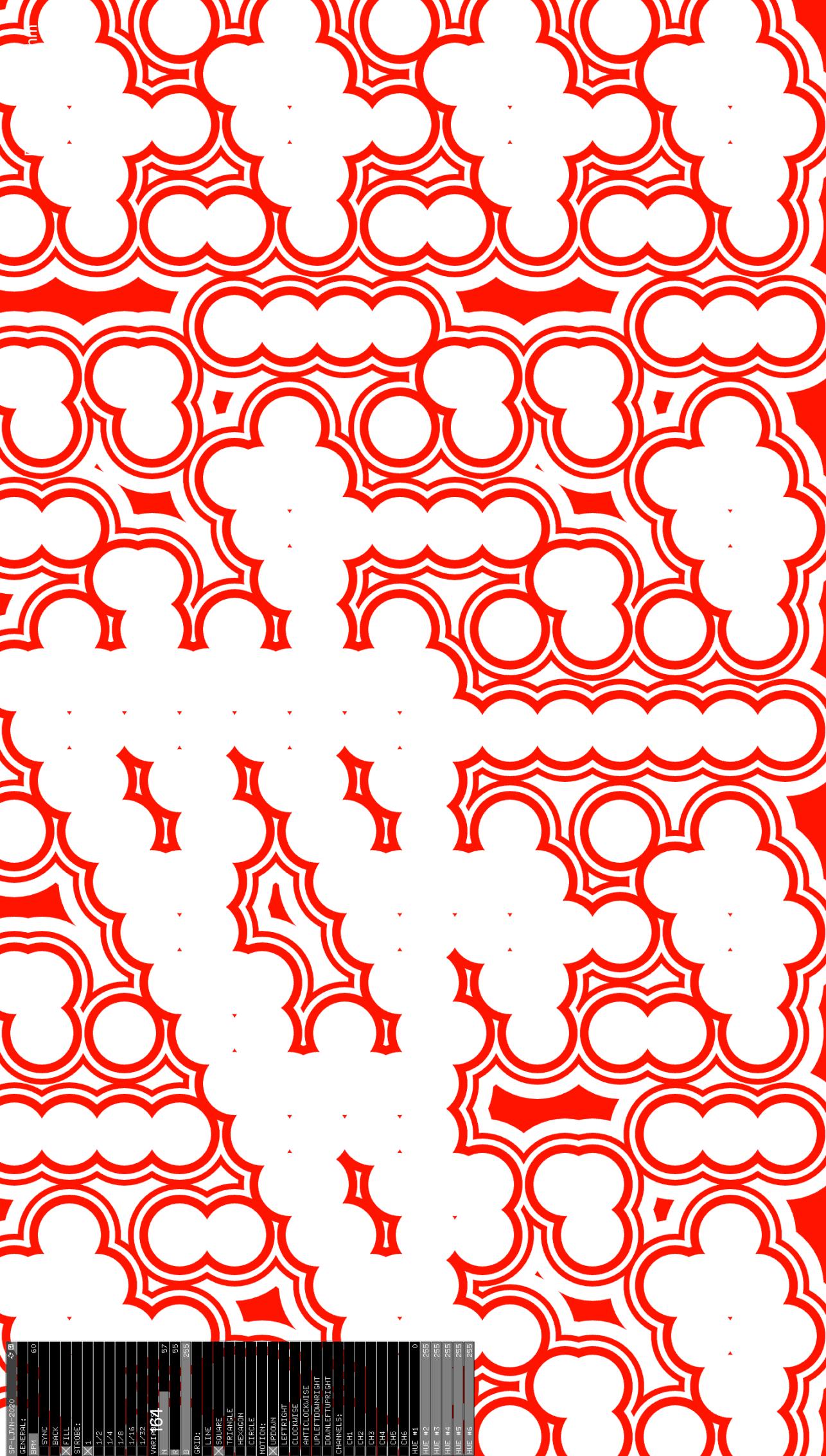
HUE #5

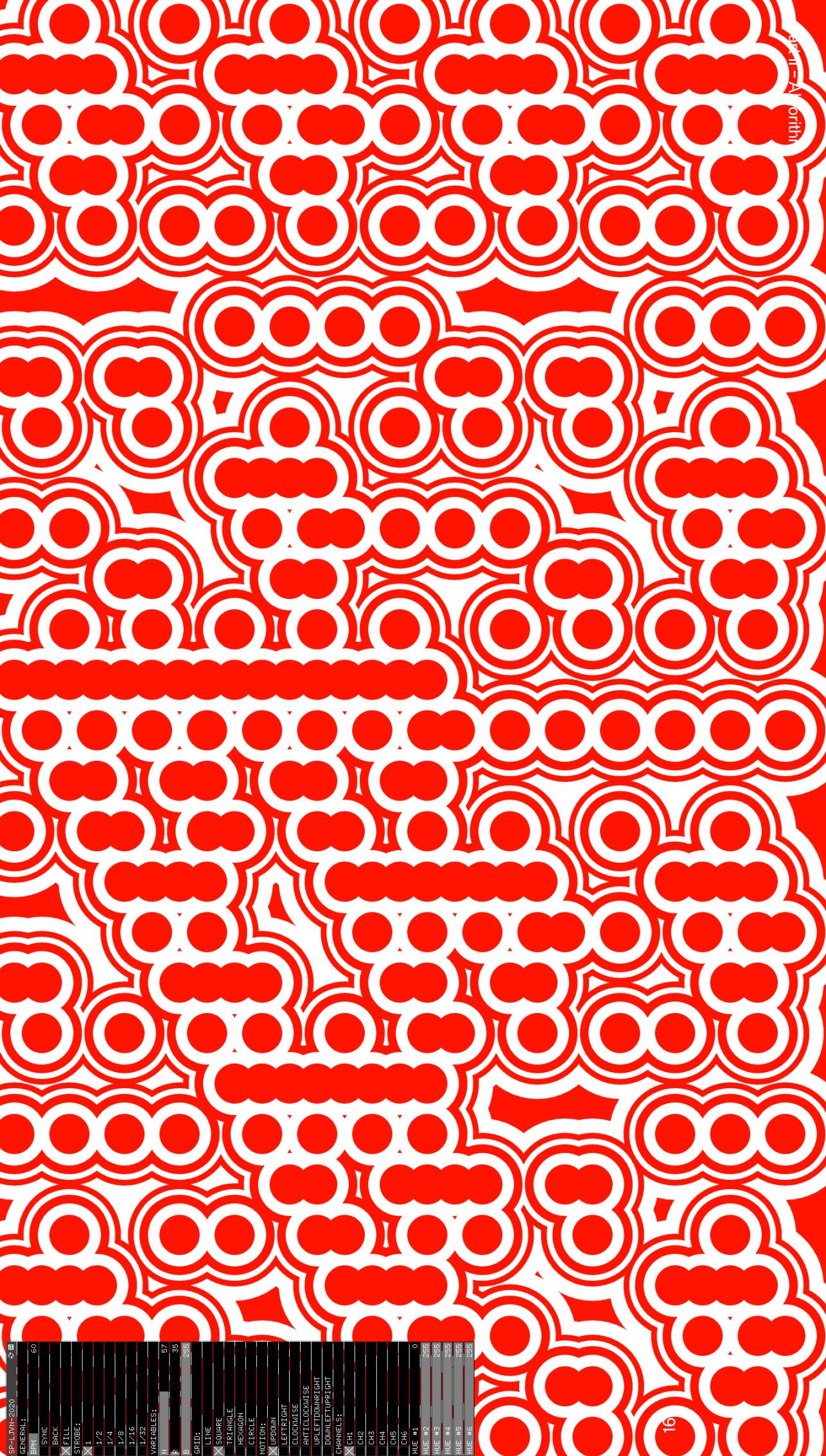
HUE #6

SI-JDN-2020 25
GENERAL:
BPM: 60
SYNC
BLOCK
FILL
STROBE:
X 1 1.0
X 2 1.2
X 4 1.4
X 8 1.8
X 16 1.16
X 32 1.32
VARIABLES:
IN P 55
B 255
GETO:
LINE
SQUARE
TRIANGLE
HEXAGON
CIRCLE
MOTDION:
LEFTUPRIGHT
CLOCKWISE
ANTICLOCKWISE
UPLEFTDOWNRIGHT
DOWNLEFTUPRIGHT
CHANNELS:
C14
C12
C13
C14
C15
C16
HUE #1 255
HUE #2 255
HUE #3 255
HUE #4 255
HUE #5 255
HUE #6 255



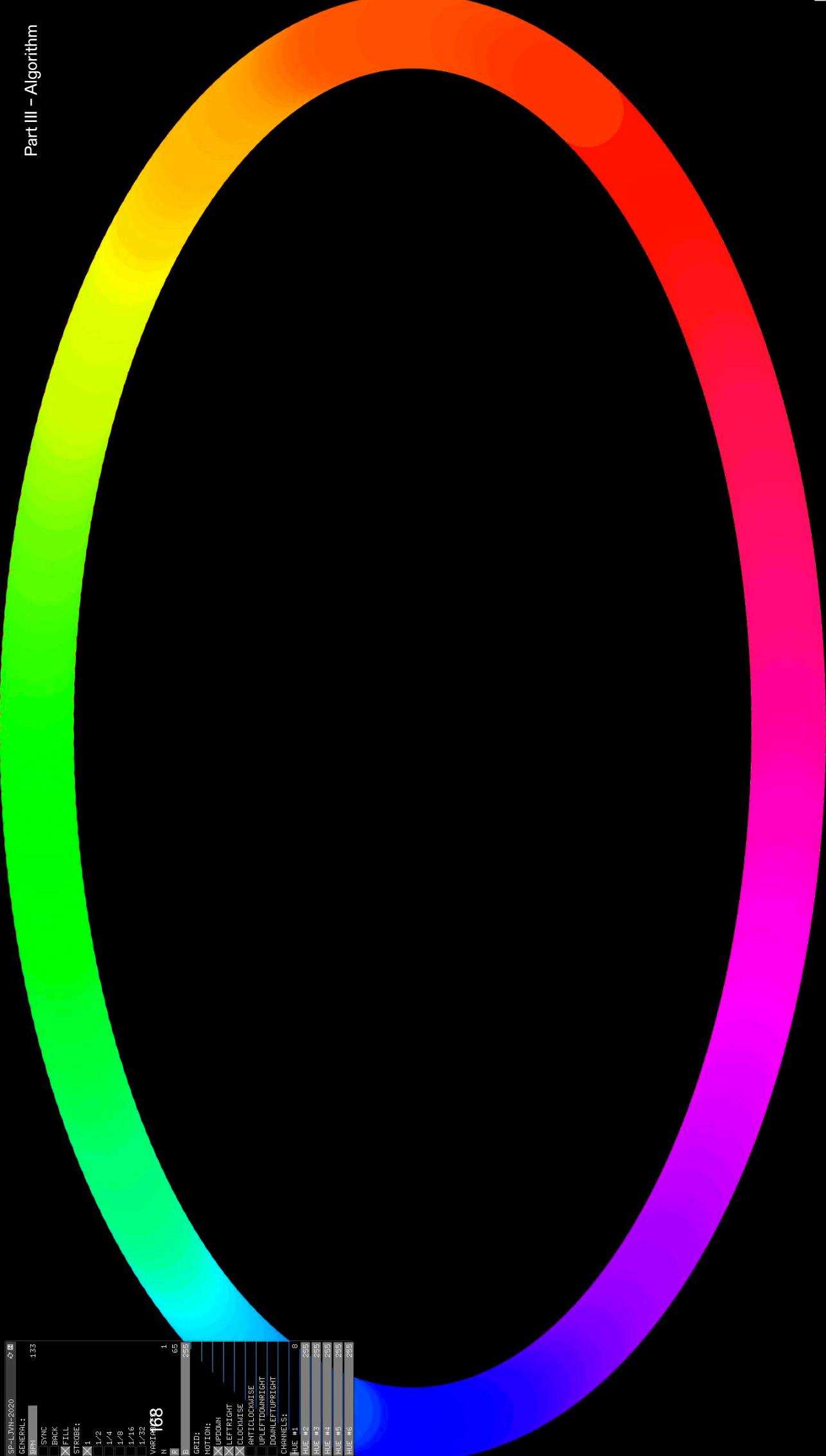


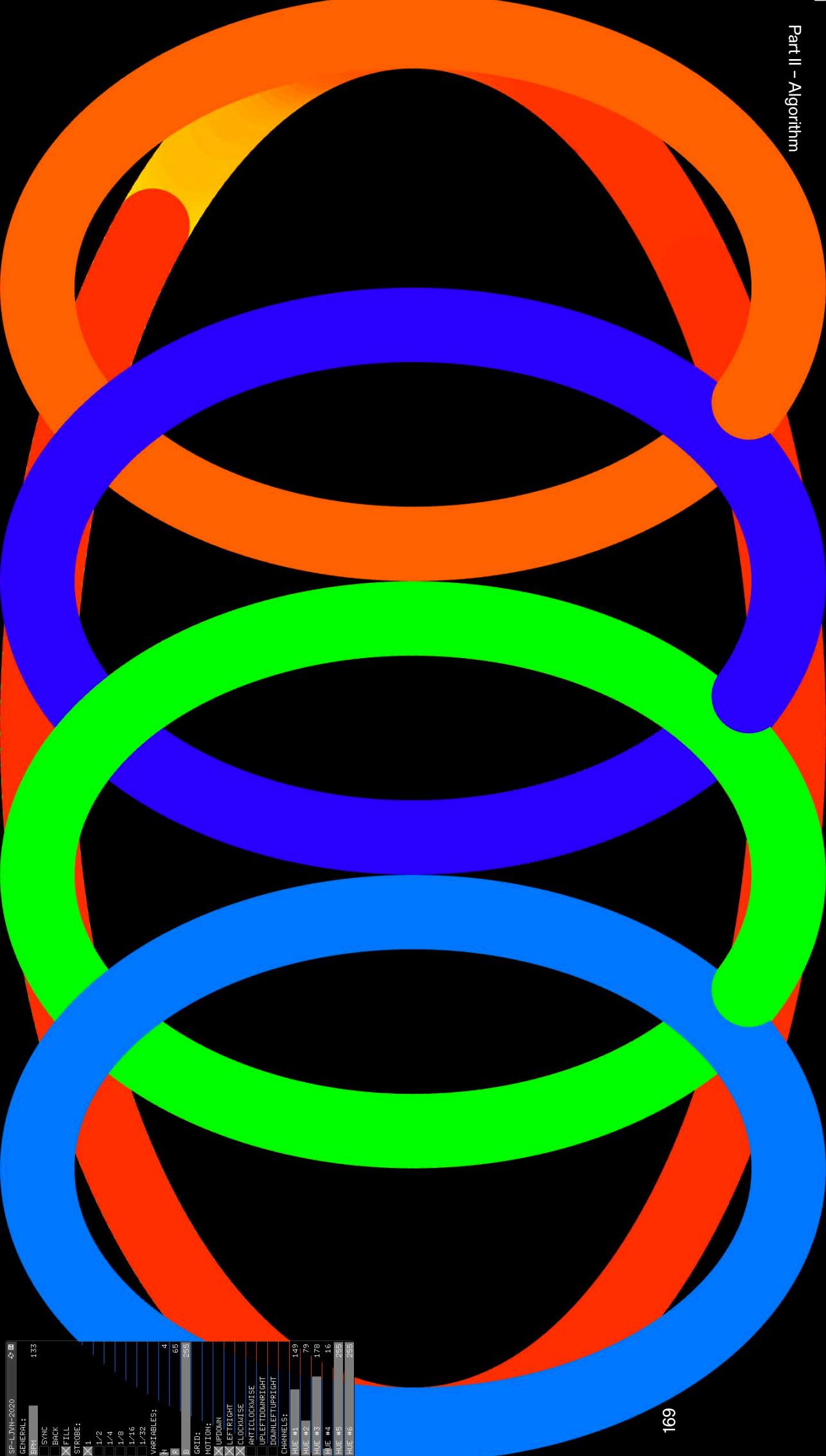




SP-L-DYN-2020	1	
GENERAL:		
BPM:	1	
SYNC:		
BACK:		
FILL:		
STROBE:		
X1	1	
1.2		
1.4		
1.6		
1.8		
1.16		
1.32		
VOLUME:	15	
IN	1	
R	49	
GEAR:		
LINE		
SQUARE		
TRIANGLE		
HEXAGON		
CIRCLE		
MUDLOD:		
LEFT/RIGHT		
CLOCKWISE		
ANTICLOCKWISE		
UP/LEFT/TOURIGHT		
DOWN/LEFT/TOURIGHT		
CHANNELS:		
CH1		
CH2		
CH3		
CH4		
CH5		
CH6		
HUE #1	255	
HUE #2	255	
HUE #3	255	
HUE #4	255	
HUE #5	255	
HUE #6	255	





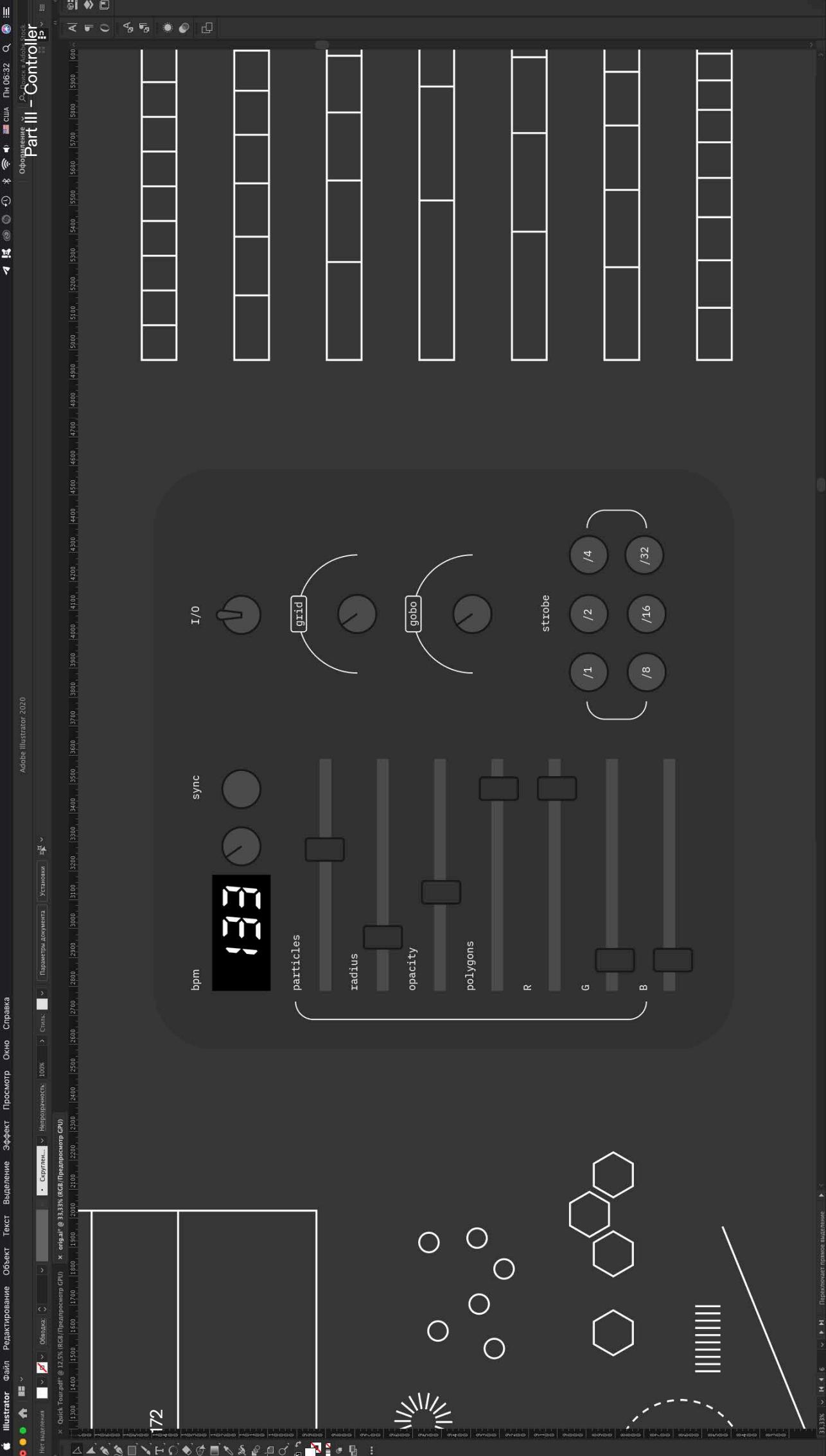


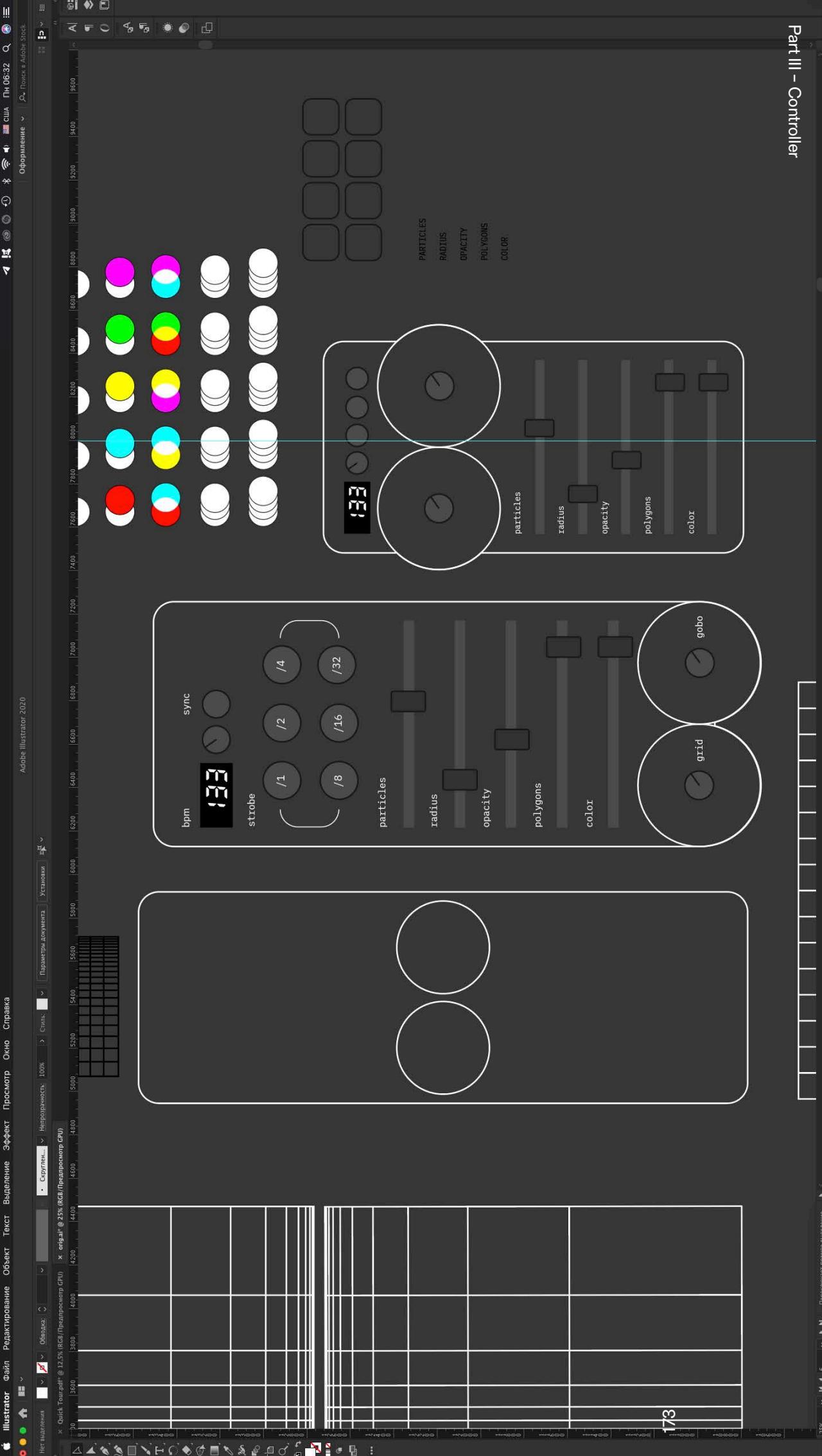
Part III

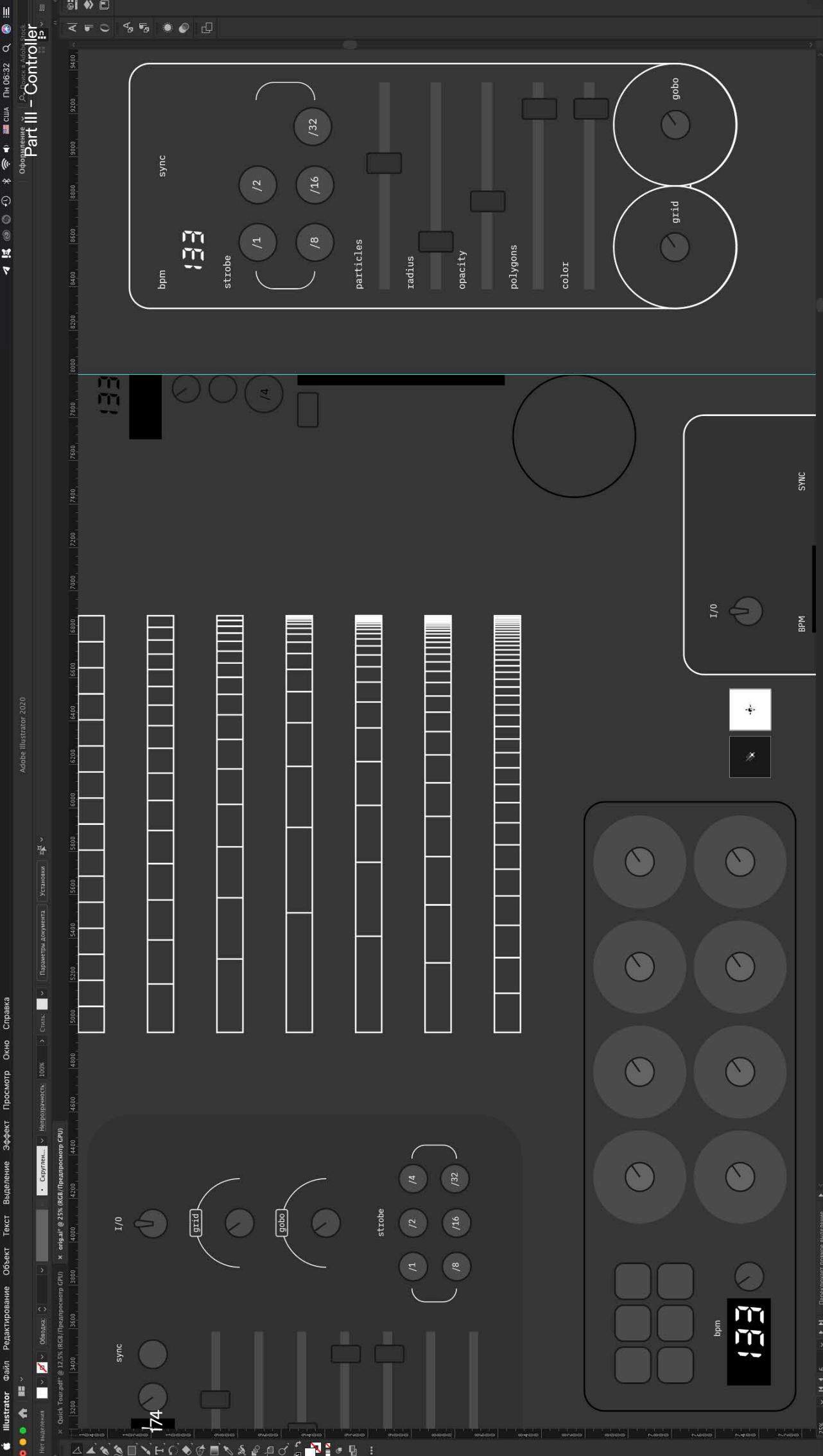
Controller

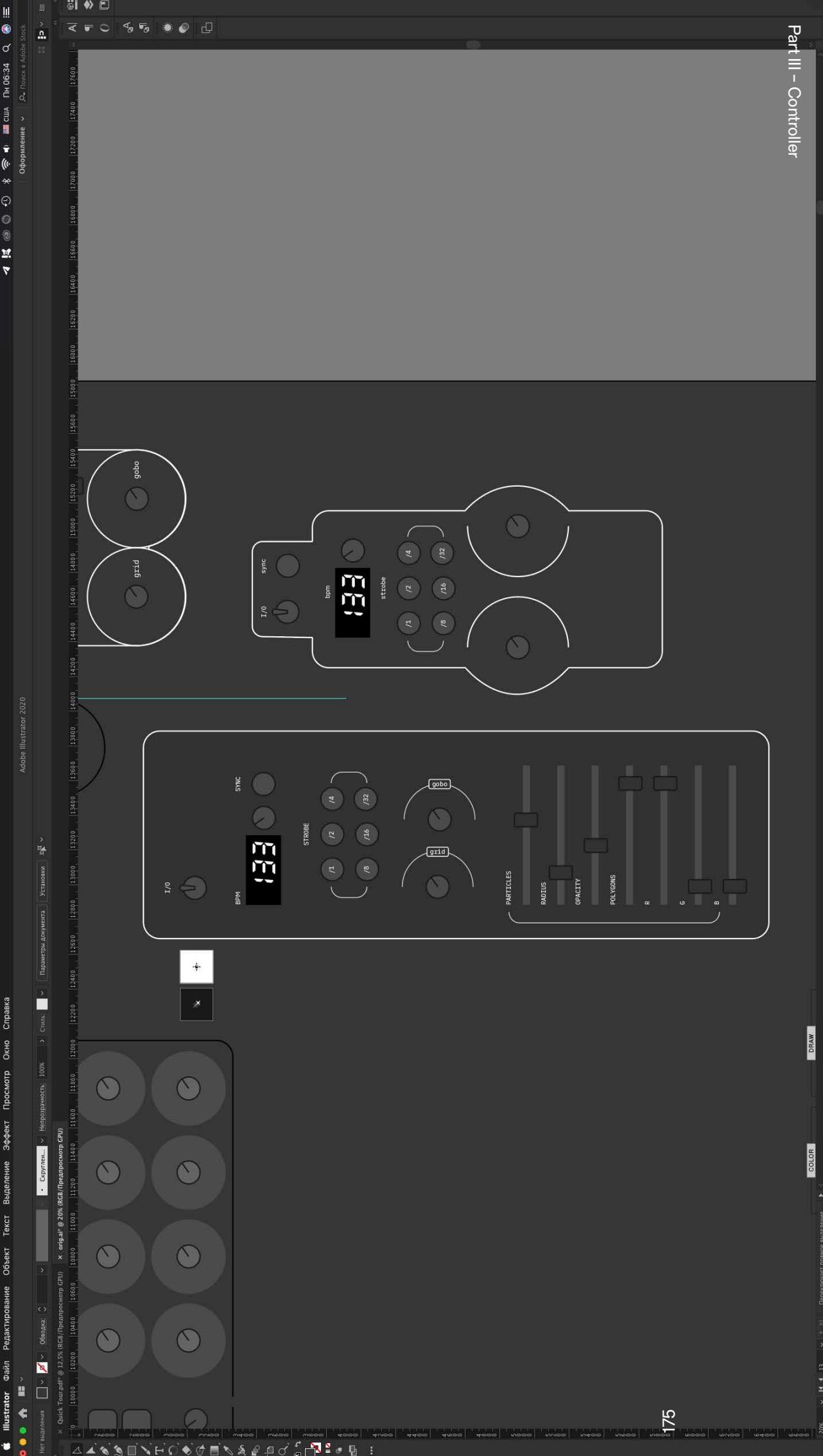
3.1 Layout

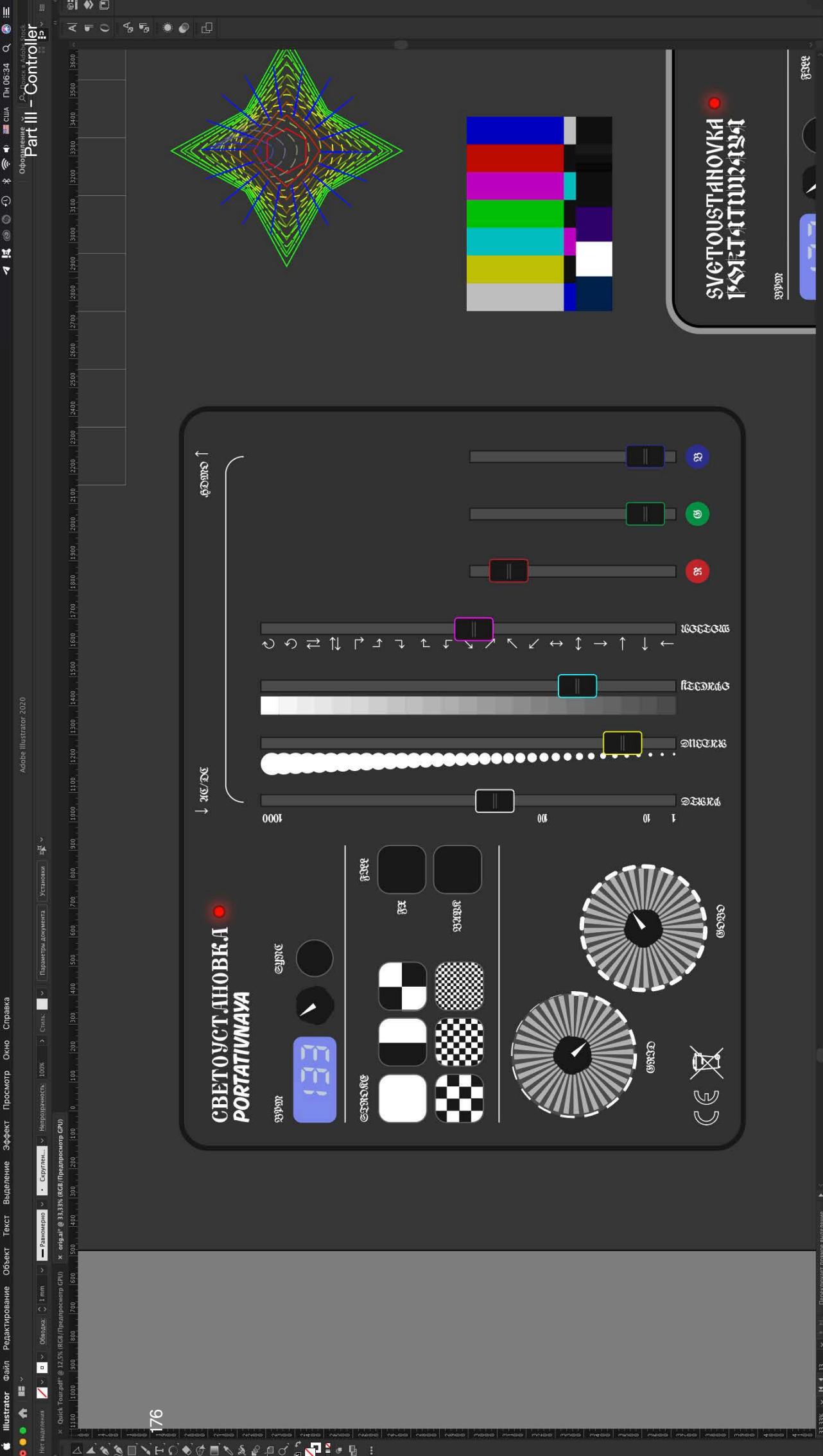
The layout of the controller follows the very logical hierarchical approach of the algorithm it serves purpose to, with the workflow going from up to down, allowing the user for a quite precise and independent control of all the individual parts of the program. The parts of the controls are made to be distinct from each other and allow the person to be both changed simultaneously in the places where needed like main variables sliders, as well as maintaining harder change for the options of color, grid, and movement made as knobs. The parts which require super fast response like strobe and sync are made as buttons for a maximum precision feedback.











Файл Редактирование Объект Текст Видение Эффект Просмотр Окно Справка

Нет видимости Quick Tools@ 12,5% RGB / Предварительный GPU

Образец: 1 mm Старт Непрозрачность: 100% Режим рисования: - Склейка, - Разделение, - Скрытие, - Прозрачное

177

The image shows the Adobe Illustrator interface with the 'Создать' (Create) effect dialog box open. The dialog contains various parameters such as 'PARTICLES', 'RADIUS', 'OPACITY', and a preview window showing a circular design with radial lines. Below the dialog, there are two text blocks: 'СВЕТОУСТАНОВКА' (SVETOUSTANOVKA) and 'ПОРТАТИВНАЯ' (PORTATIVNAYA), which are being combined using the 'Создать' effect.

СВЕТОУСТАНОВКА
ПОРТАТИВНАЯ

СВЕТОУСТАНОВКА
PORATIVNAYA

СВЕТОУСТАНОВКА
PORTATIVNAYA

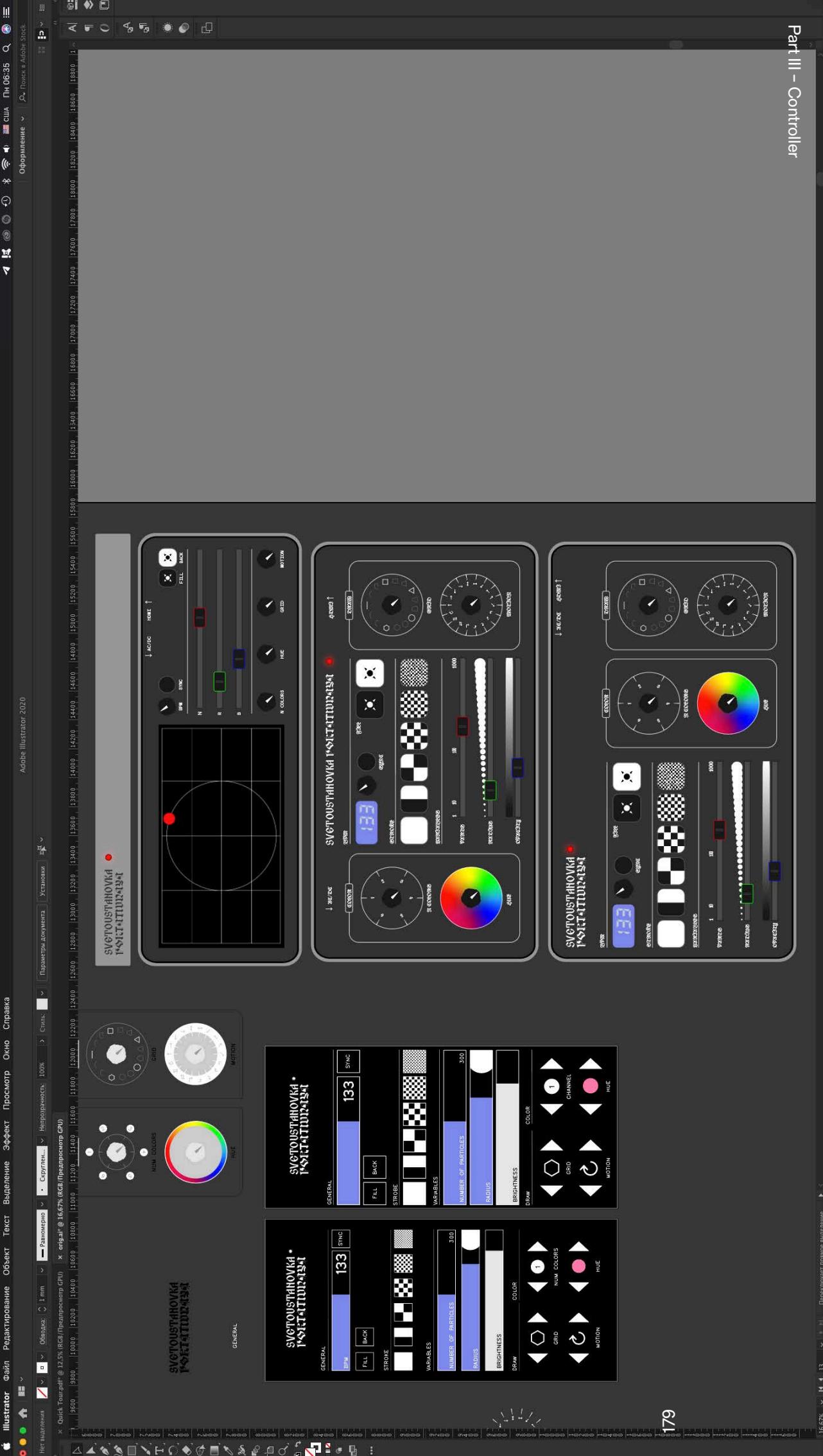
СВЕТОУСТАНОВКА
ПОРТАТИВНАЯ

СВЕТОУСТАНОВКА
PORATIVNAYA

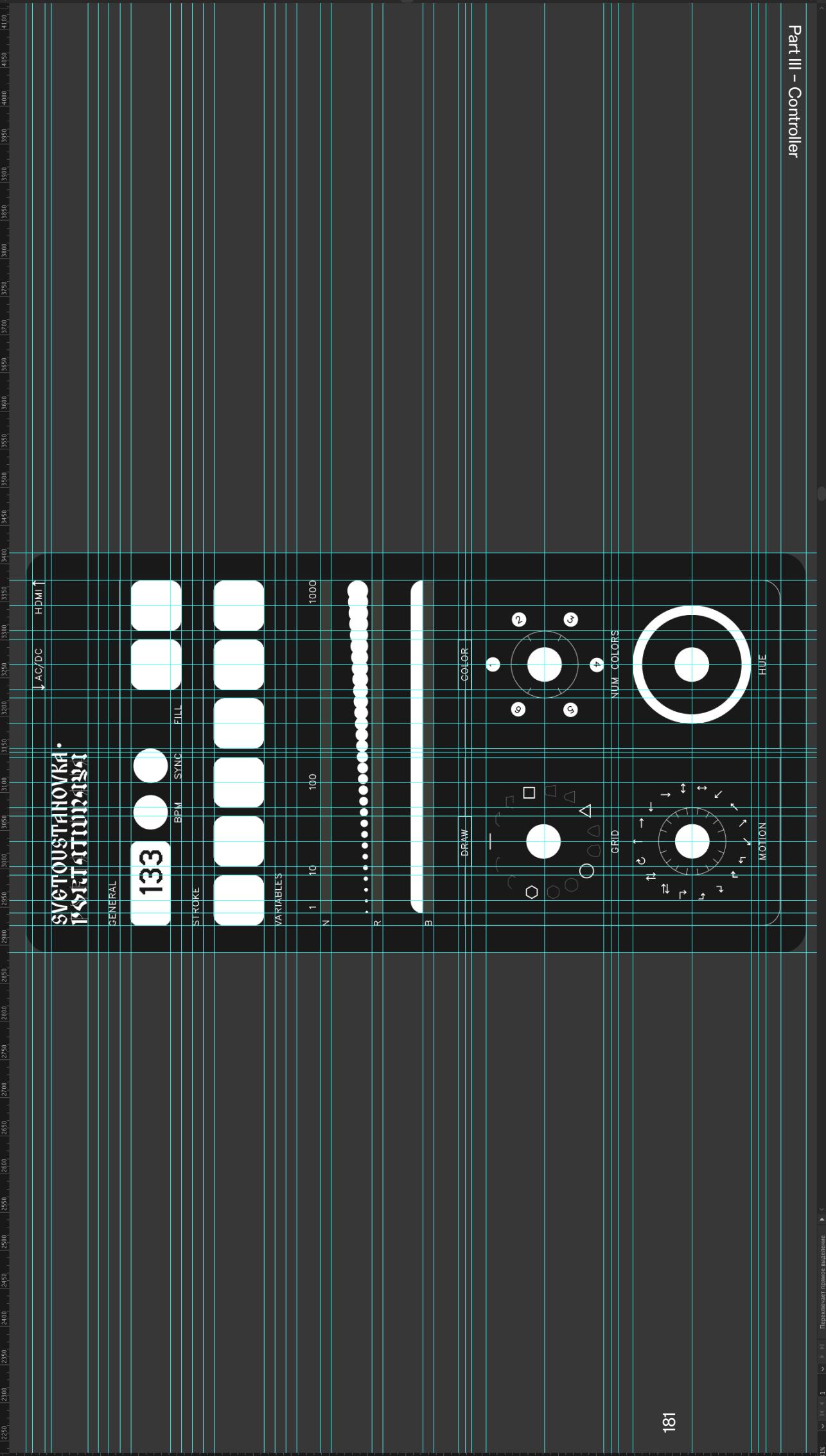
The final logo design, featuring a circular pattern of radial lines and the text 'СВЕТОУСТАНОВКА ПОРТАТИВНАЯ' (SVETOUSTANOVKA PORTATIVNAYA), is displayed in the top right corner of the Illustrator window. The background of the logo features a wavy grid pattern.

177









3.2 Visual language

The visual solutions for the controller were probably the hardest part of the decision making within the whole project. The goal I pursued was to make a distinct, fun, individual, yet still working solution both for myself and the possible users.

However, the fact that with the quarantine the project shifted from the producing the final product to more of a proposal, it really helped me to step away from the objective goals and limitations, and speculate freely on how it could look, yet still setting myself within the strict frame that it should be possible to produce if needed.

Adobe Illustrator 2020

Пн 06:39

Оформление

Файл Редактирование Объект Текст Видение Эффект Просмотр Окно Справка

Нет изображения

Quick Tour.pdf @ 12.5% RGB/Предварительный GPU

Образец: 1 mm
Скорость: 100%

Непрозрачность: Старт.

Параметры документа Установки

133

GENERAL

STROKE

VARIABLES

133

GENERAL

STROKE

VARIABLES

1000

100

1

183

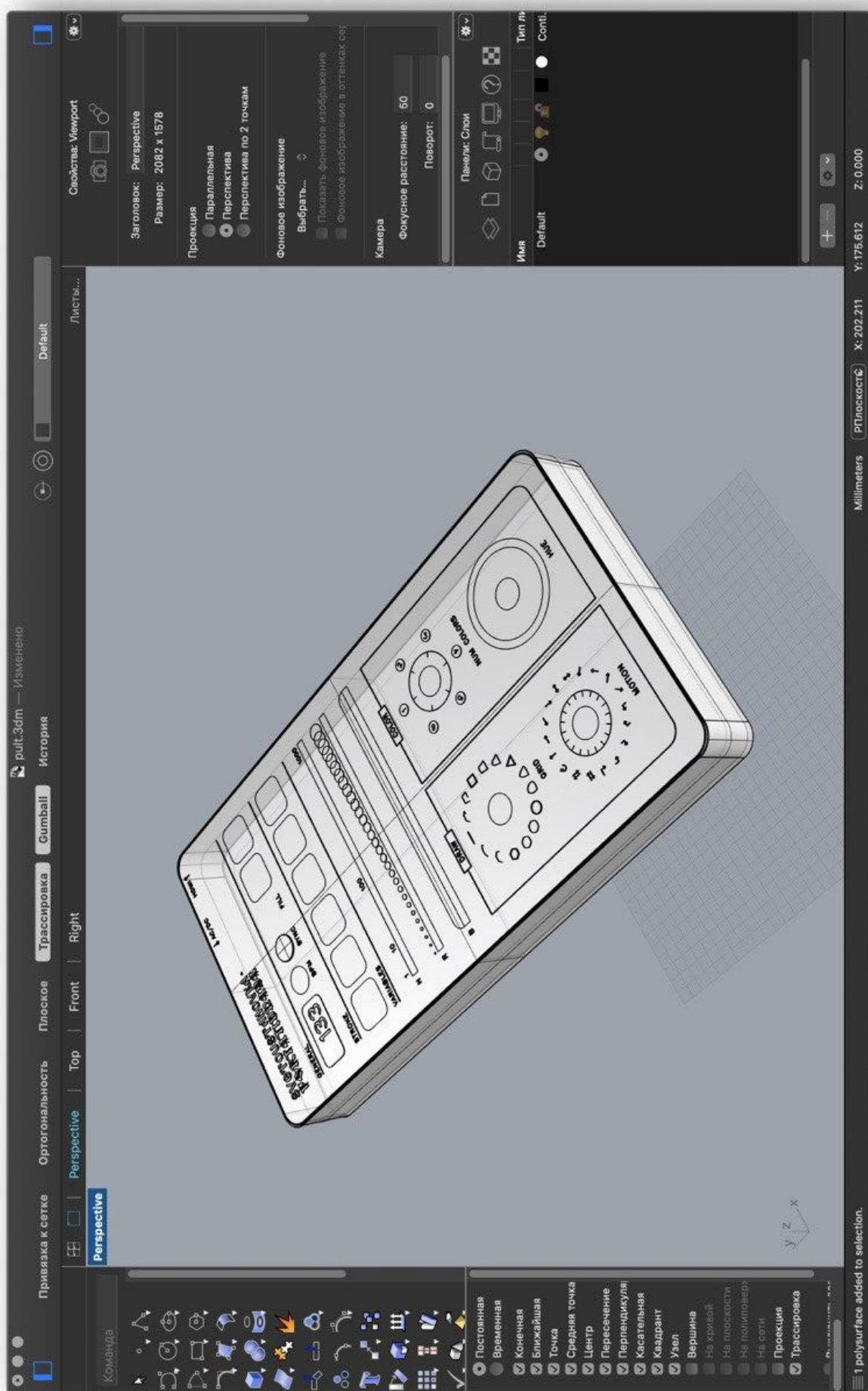
13

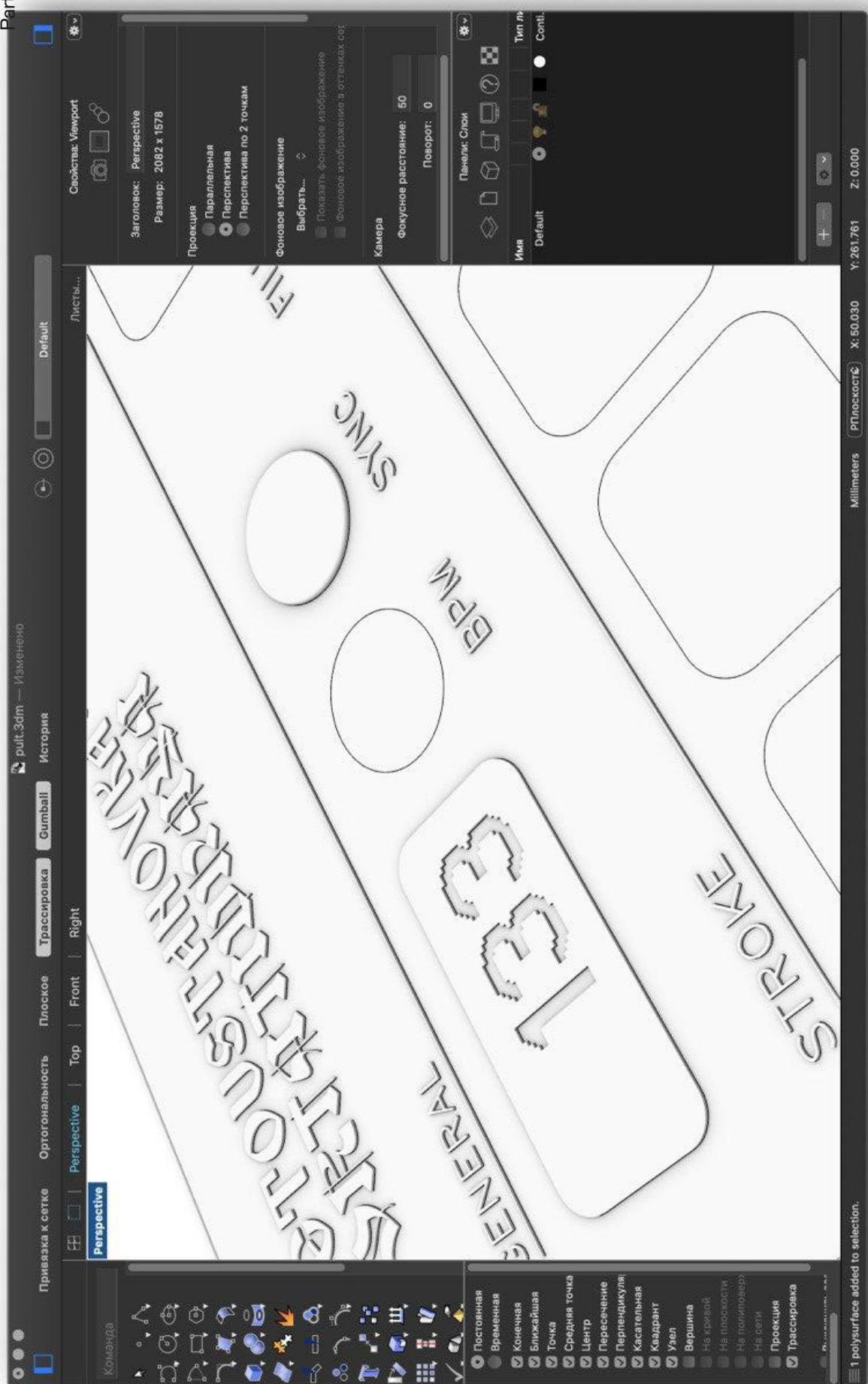
1

Панель Controler

3.3 Modeling

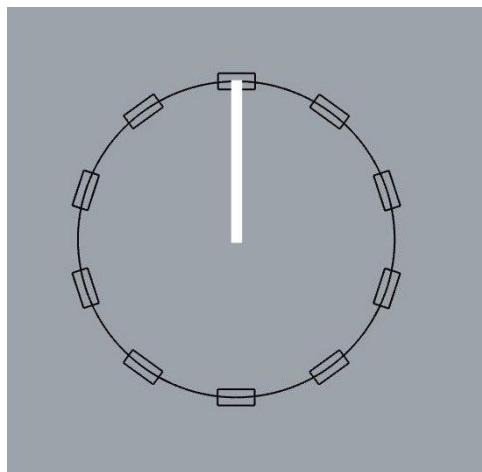
The modeling of the controller was made in the Rhinoceros 3D software under a constant advice and support from my friends from product design who helped me overcome many unpredicted difficulties. Even though I had quite a solid background using Google SketchUp app, it did not had the flexibility to produce the desired outcome I needed. Adapting to the new software was quite hard and challenging, with many non-logical errors and specifics, yet in the end it allowed to create a non compromisable quality outcome exactly how I wanted it to be.









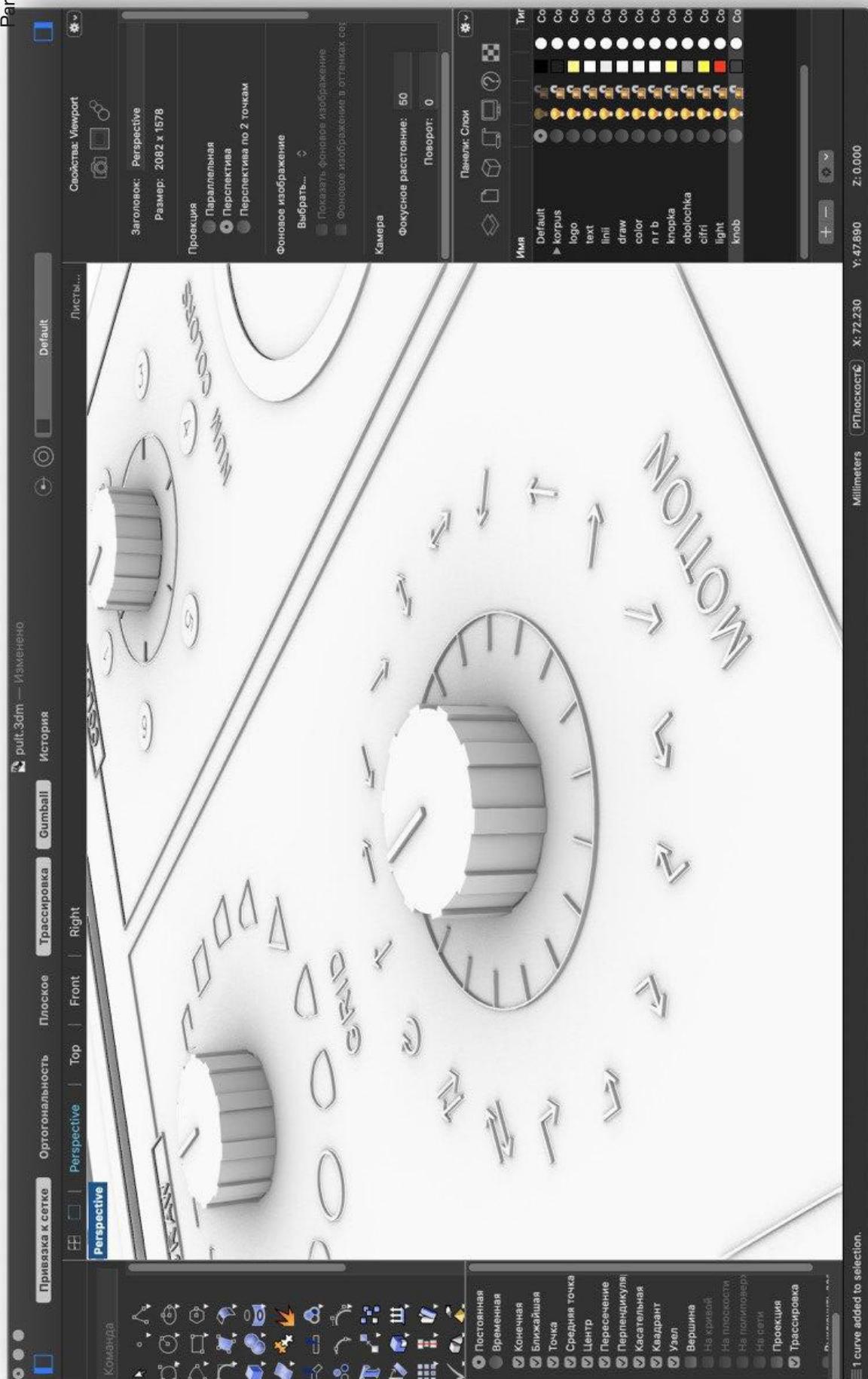


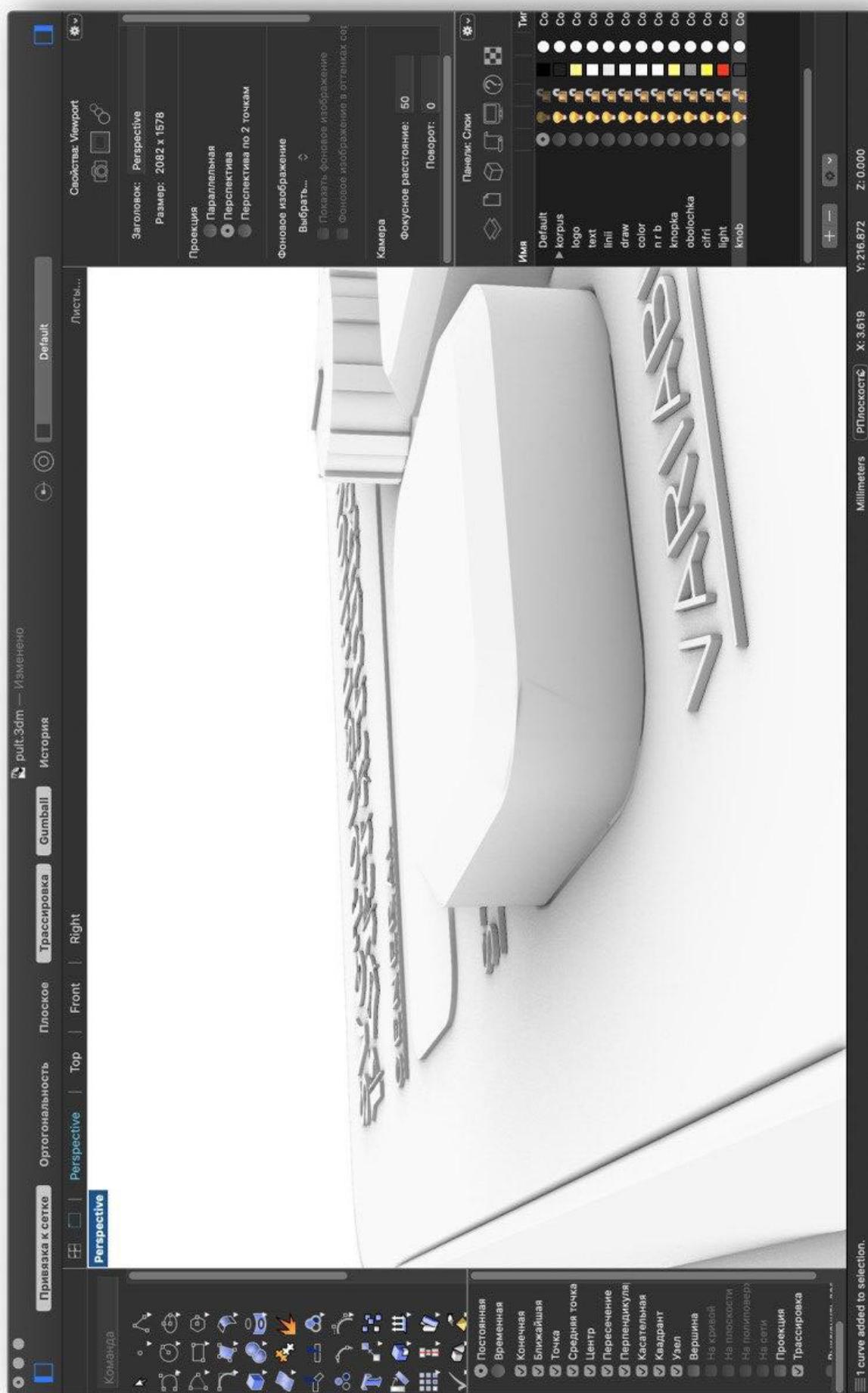


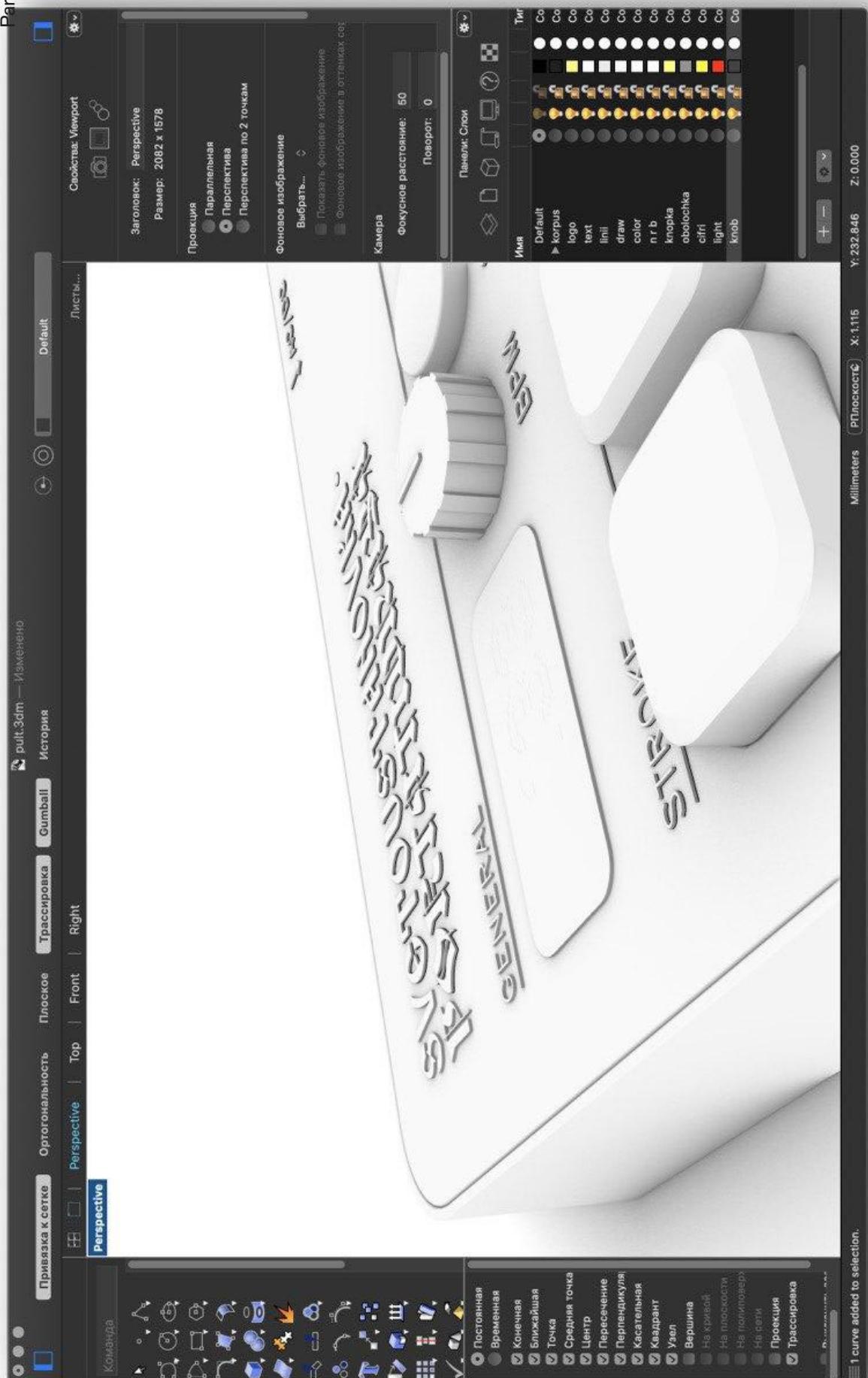


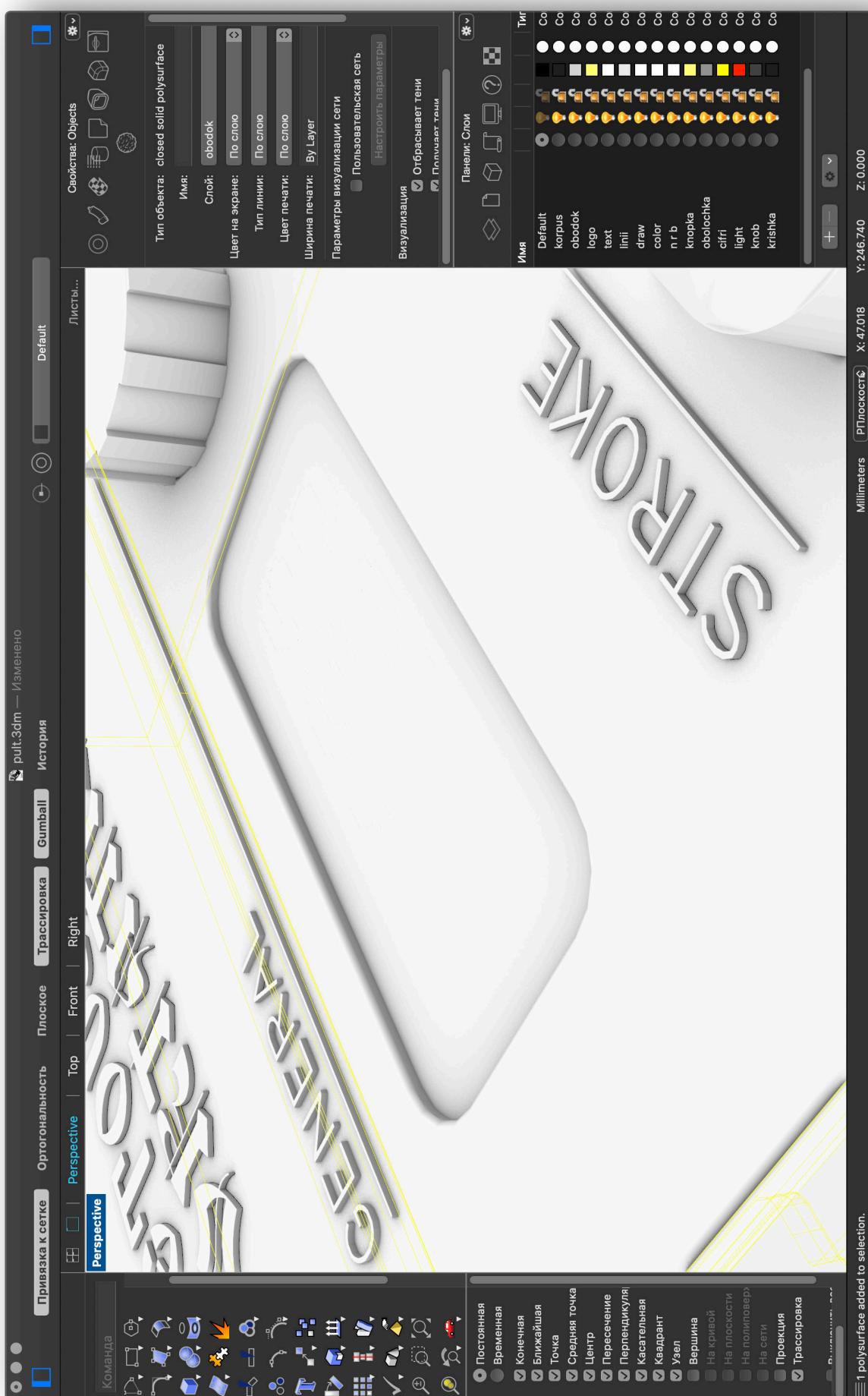




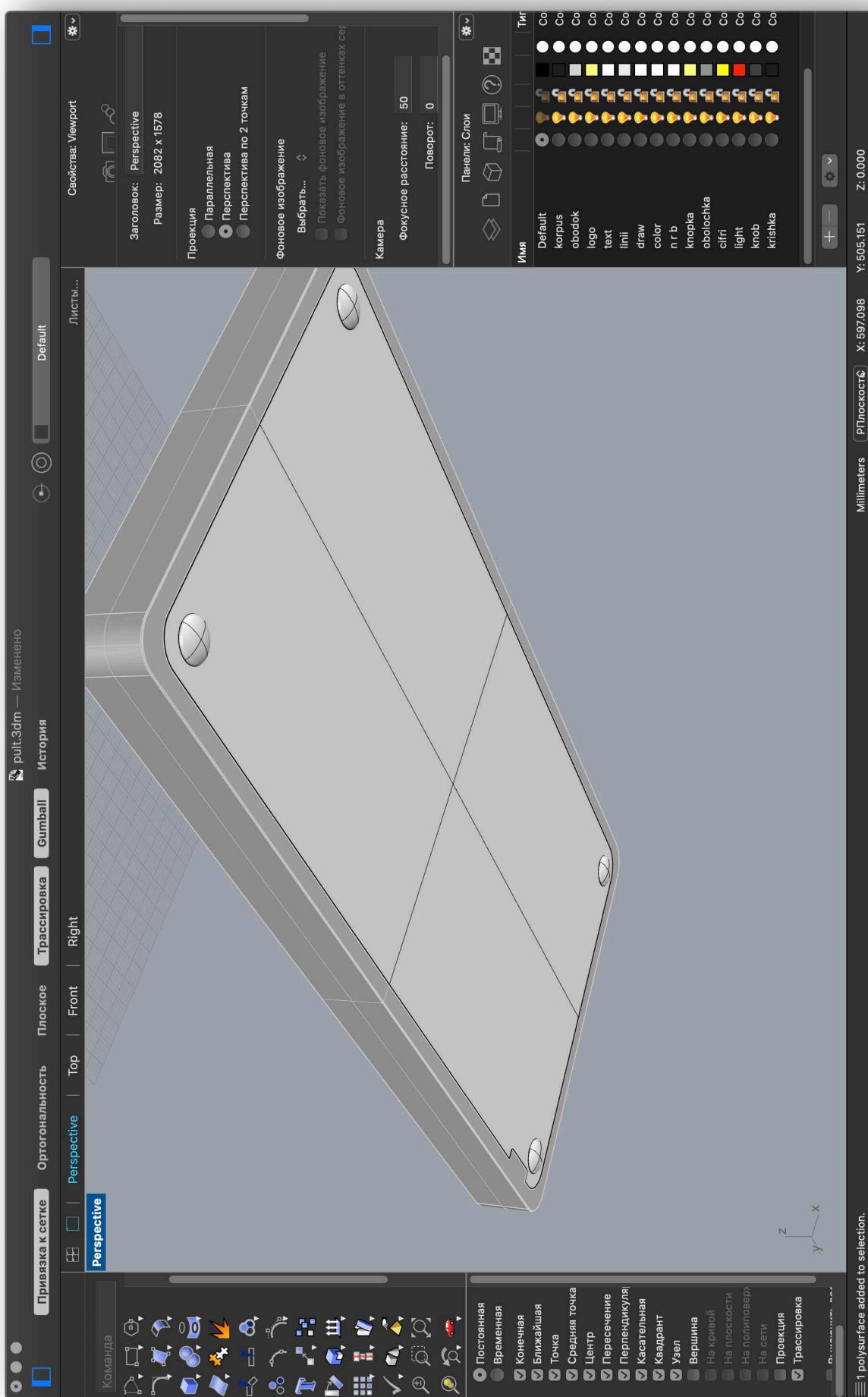


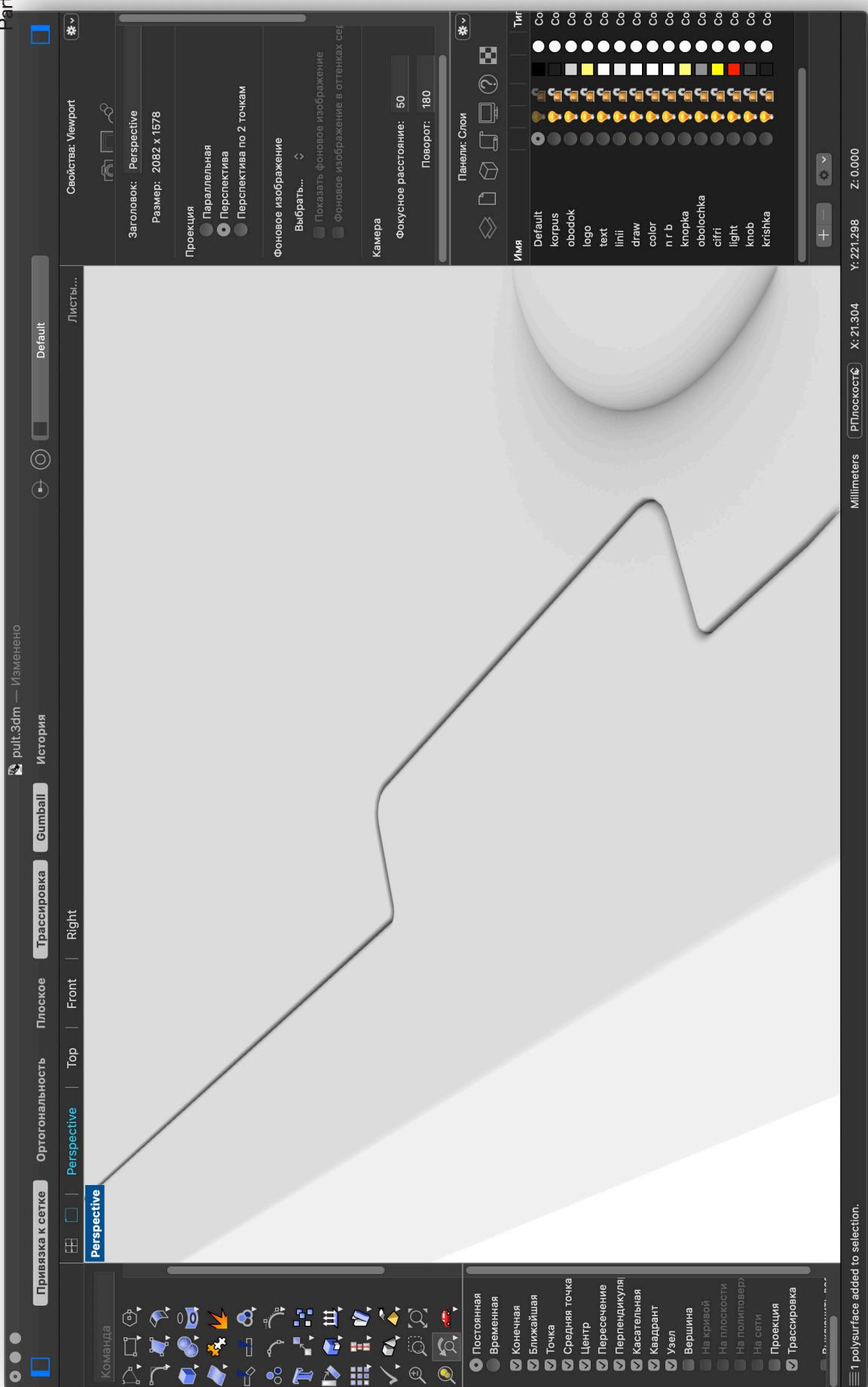




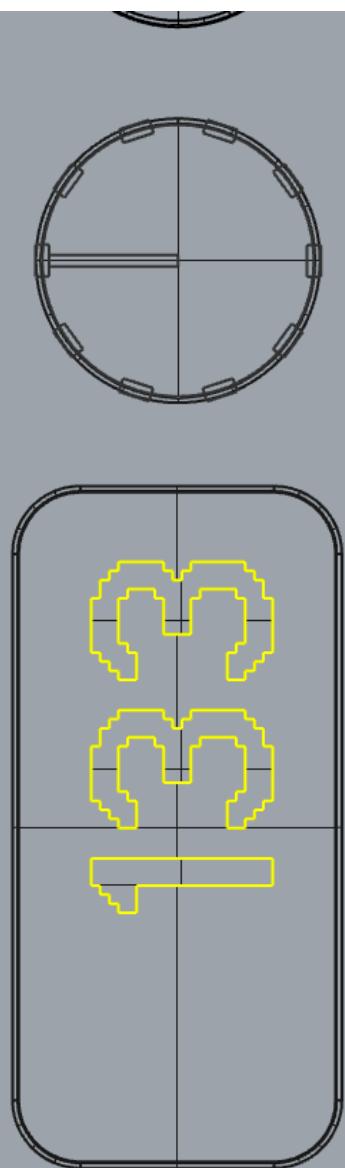
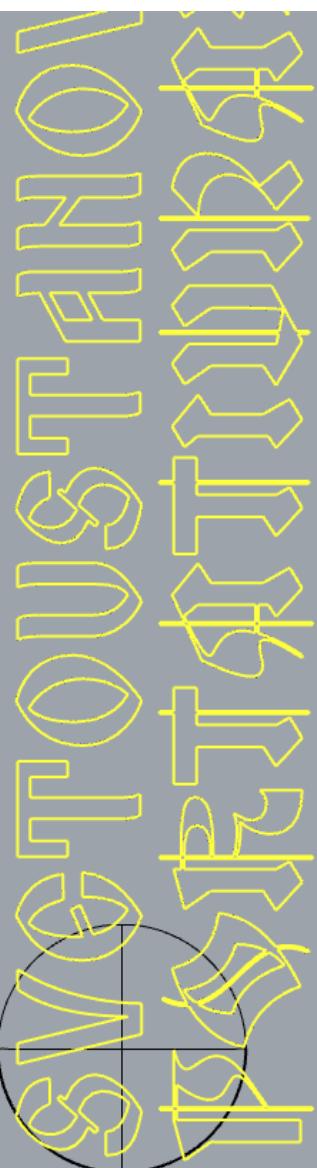


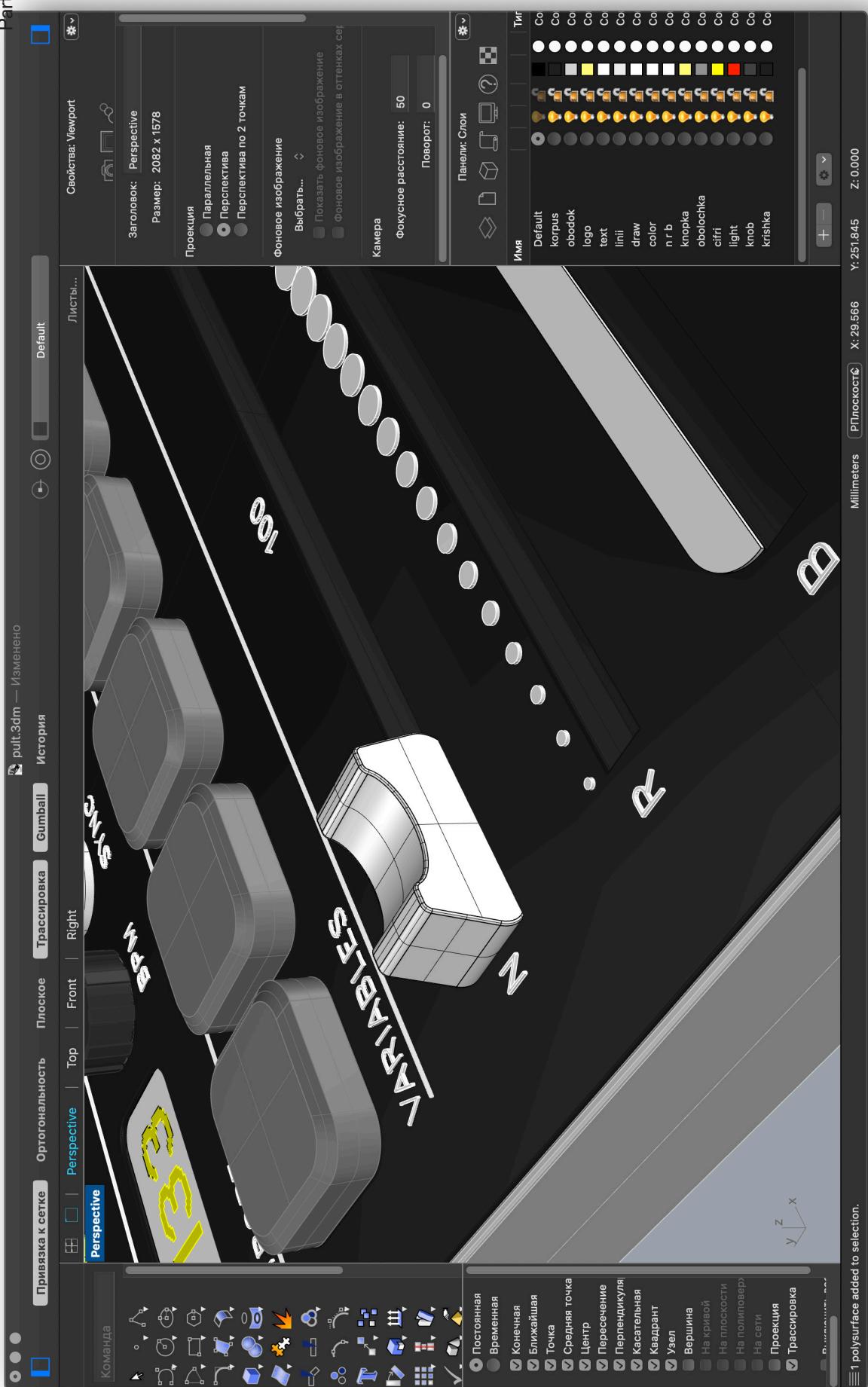




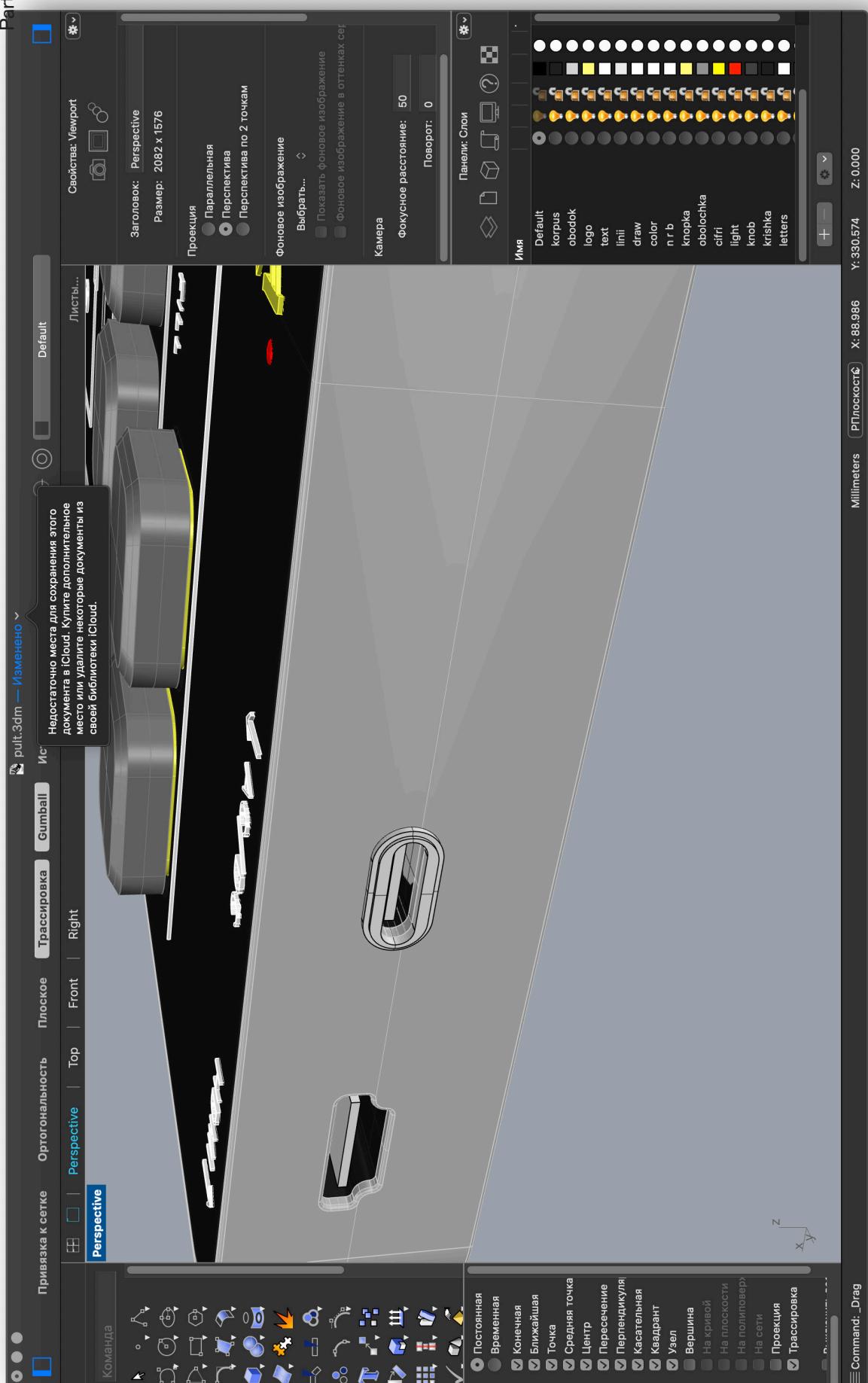


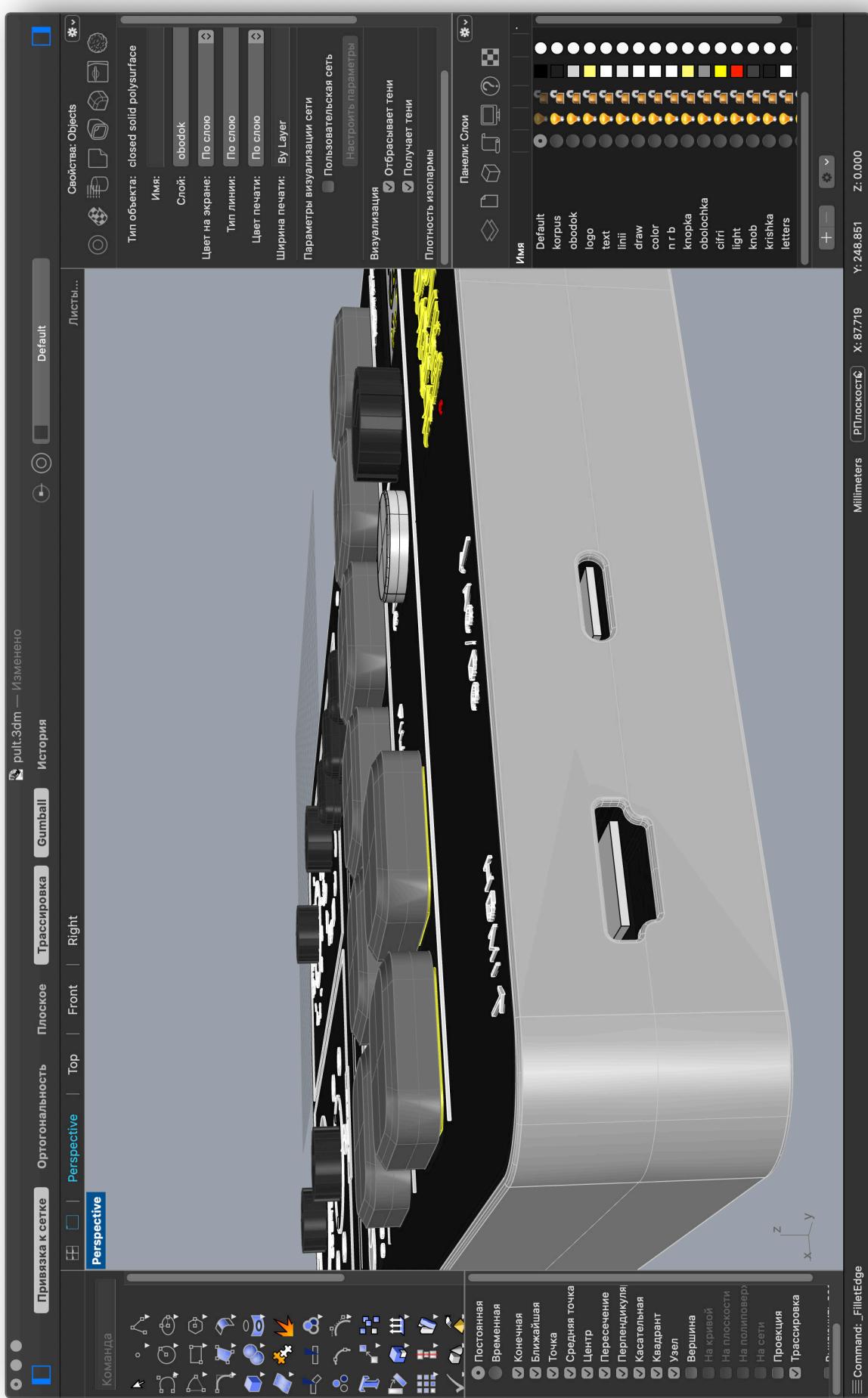
GENERAL

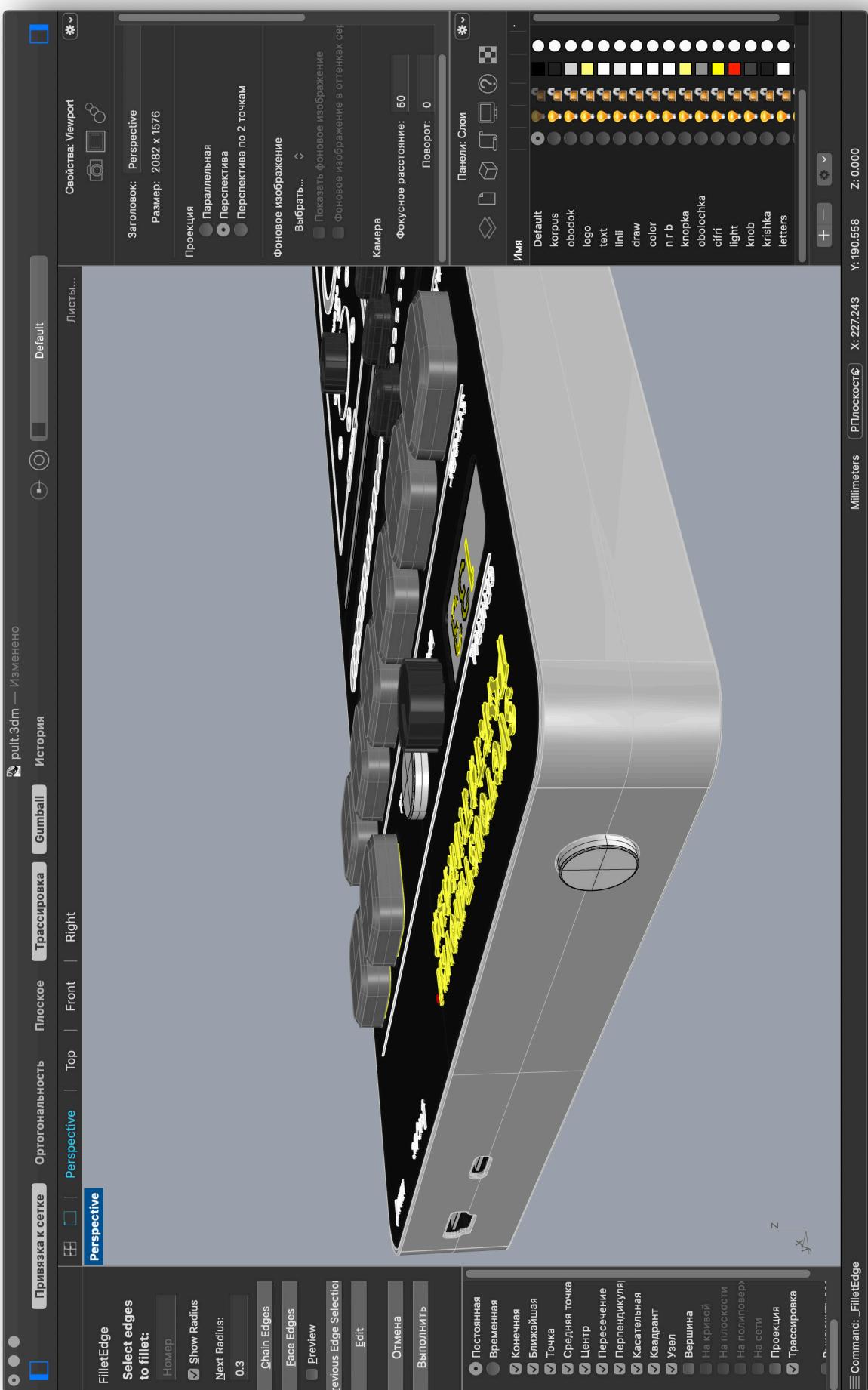


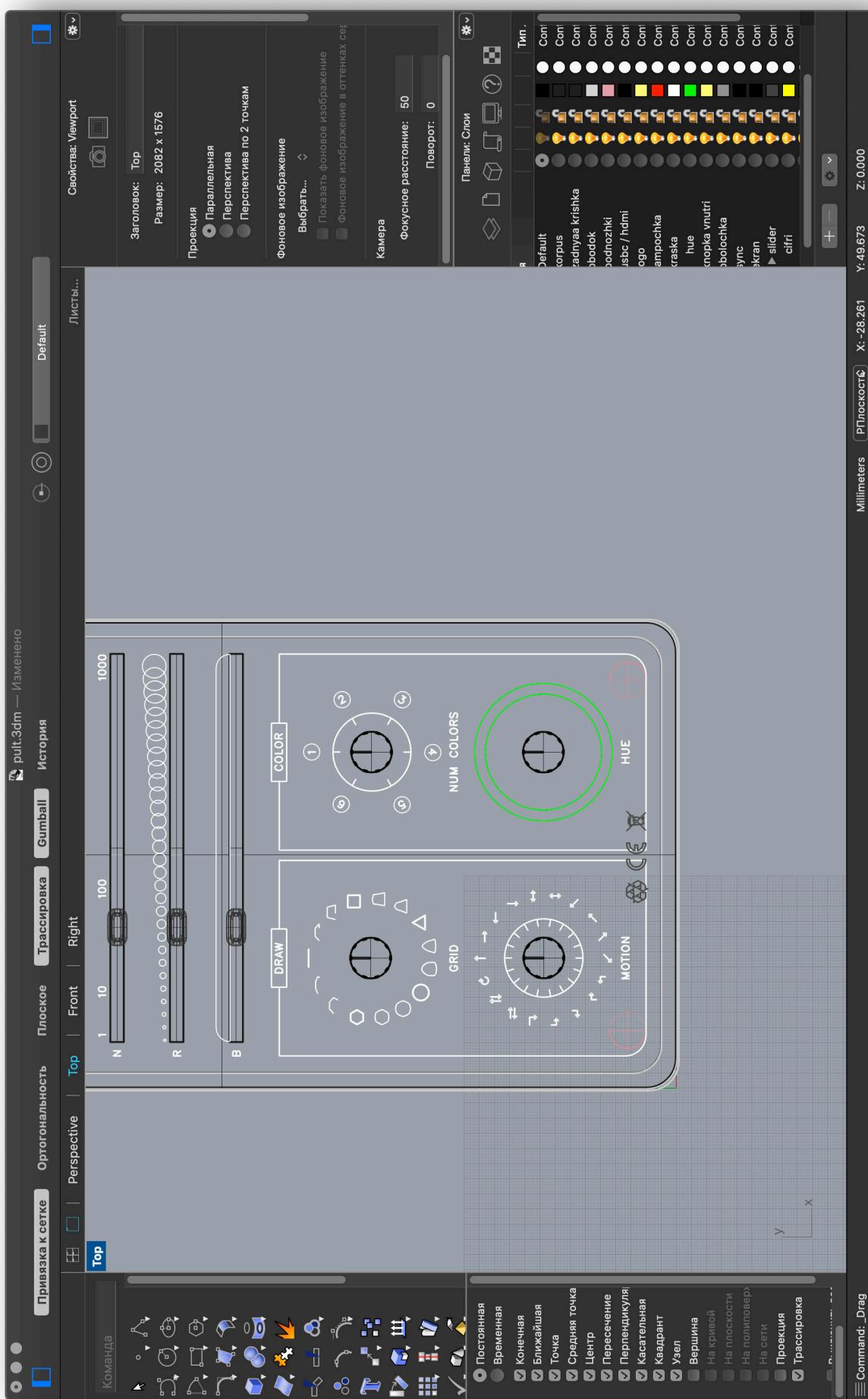


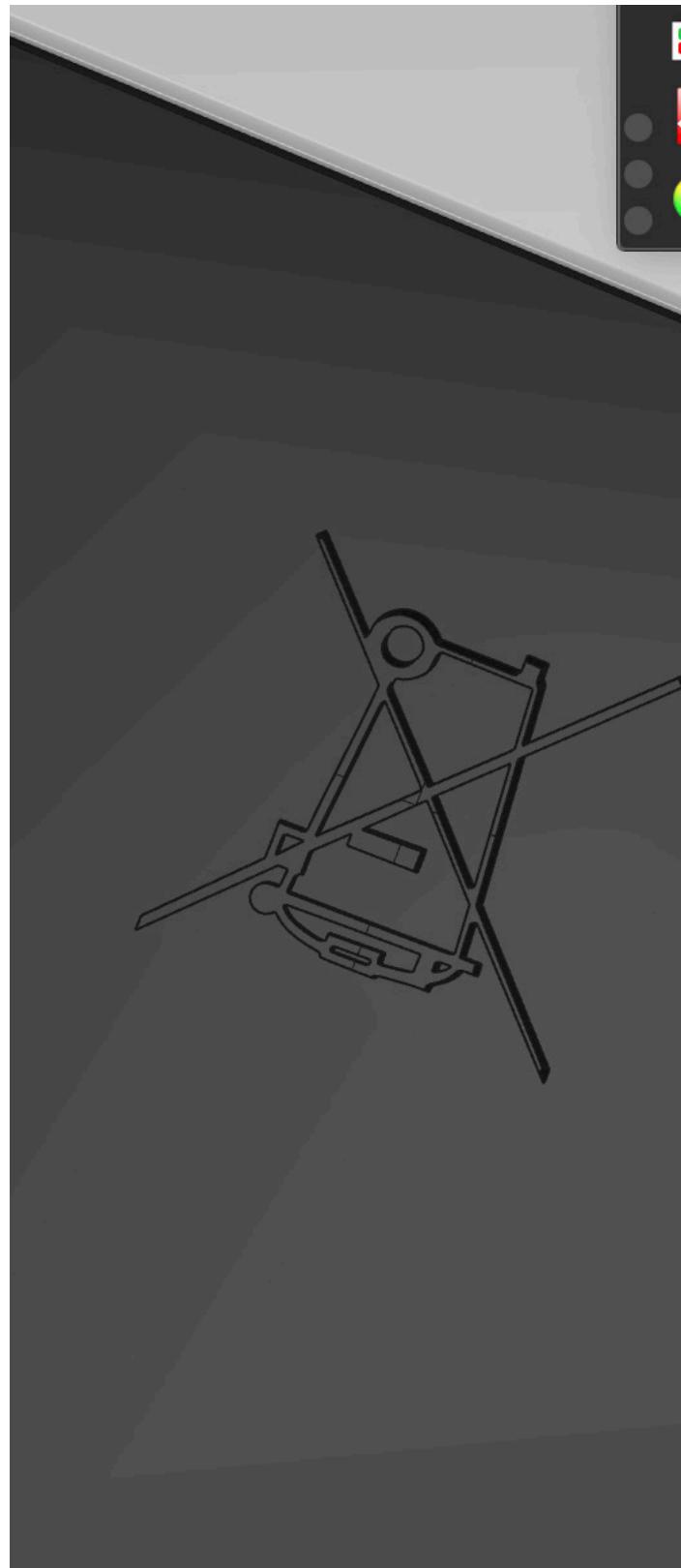


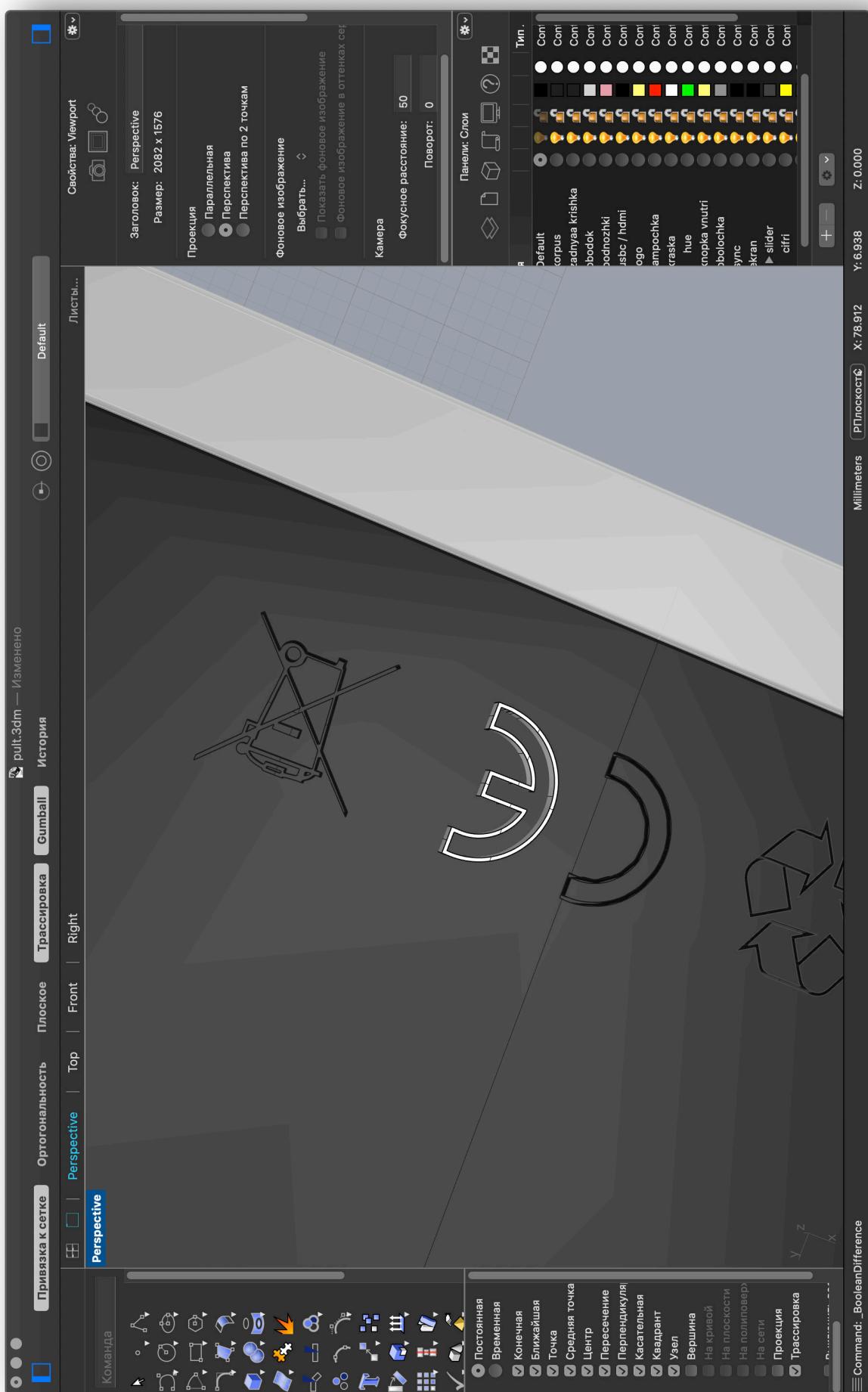












3.4 Renders research

This part of the research was a personal struggle for me, as I always had a belief that most of the renders produced and published on personal portfolios at behance are complete trash and look totally unrealistic. So, this time I tried to overcome this preset and try to find the examples of powerful, aesthetical, and believable images containing the objects which do not exist in the real life. Sadly enough, it was not enough to change my perception as most of these are still very far from perfect, yet I also managed to find some very appealing results.







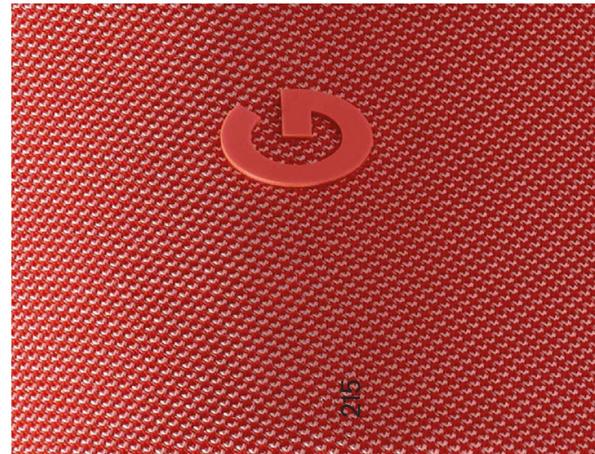


Color board.

Charcoal grey.

Stone grey.

Coral pink.





Product Specification

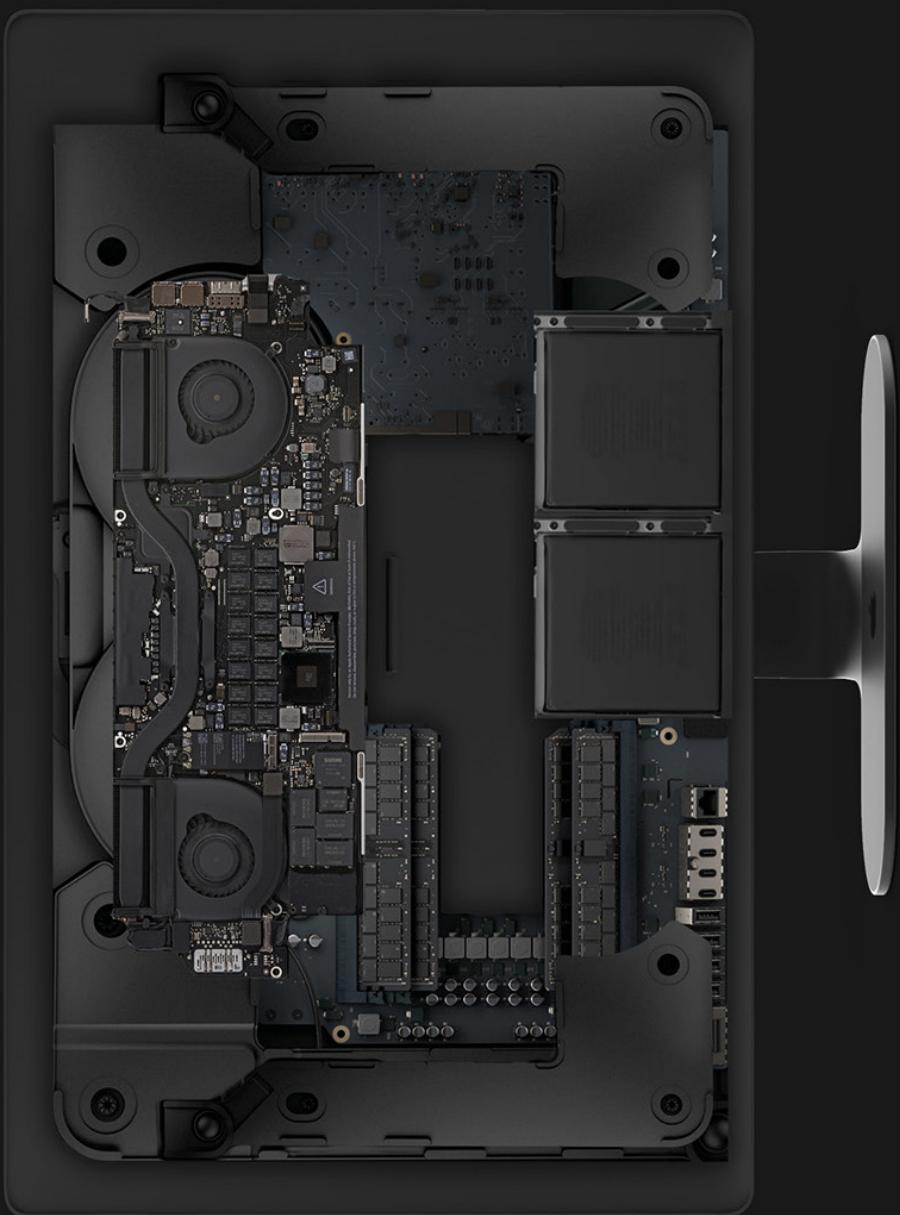


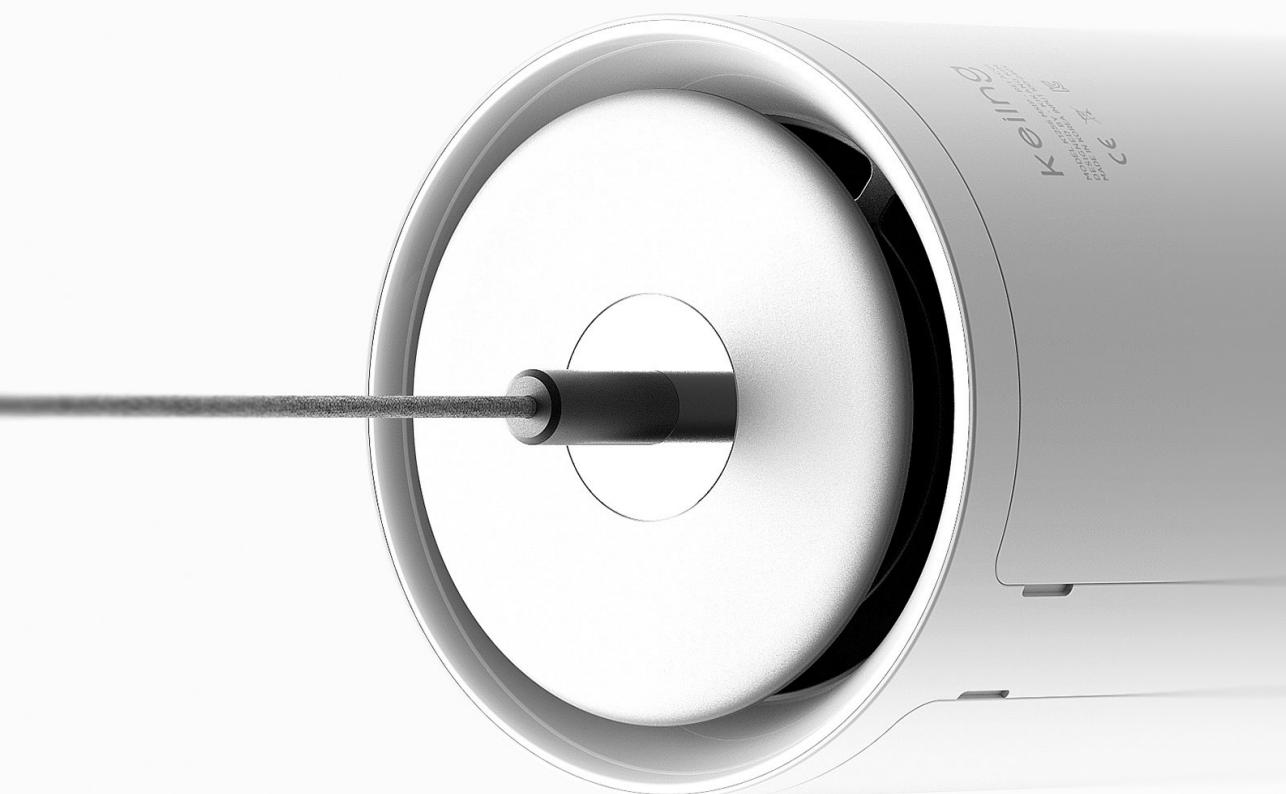
Model Name	Tea Bag Humidifier
Color	White
Body Material	Plastic
Humidification Tech	Evaporative Humidification
Usage Area	Indoors
Tank Capacity	2L
Product Dimension	412mm * 160mm * 320mm





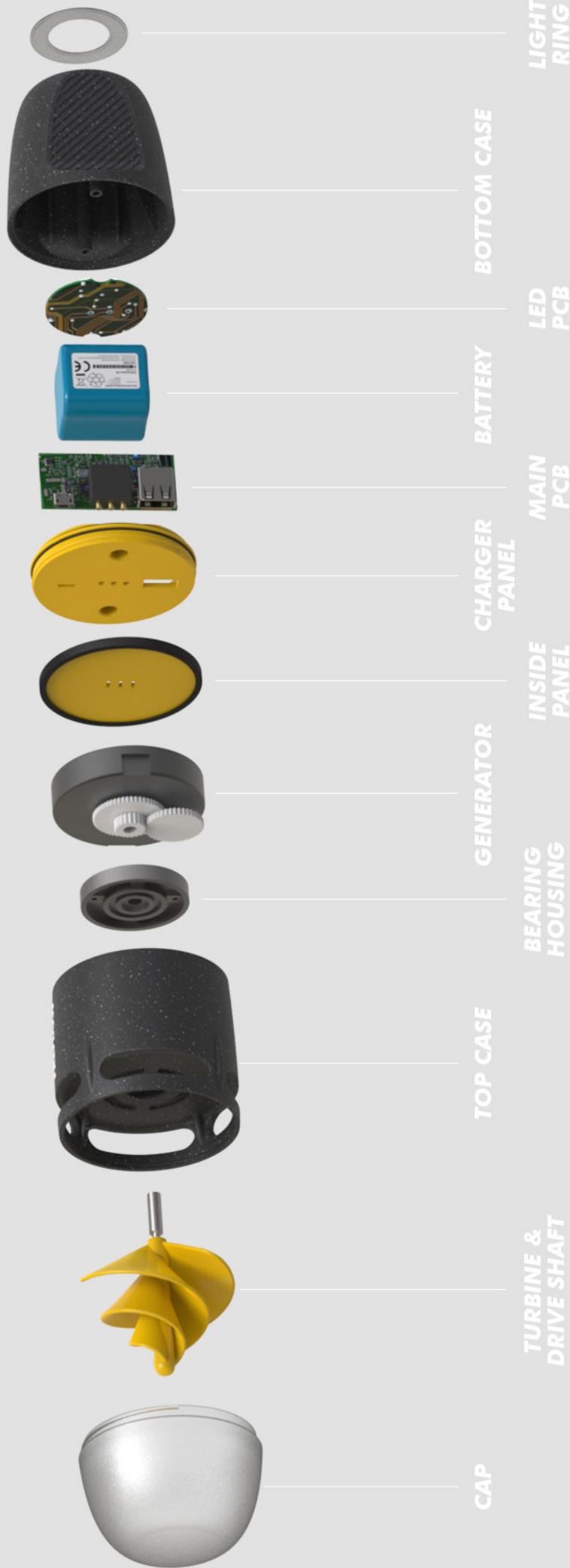


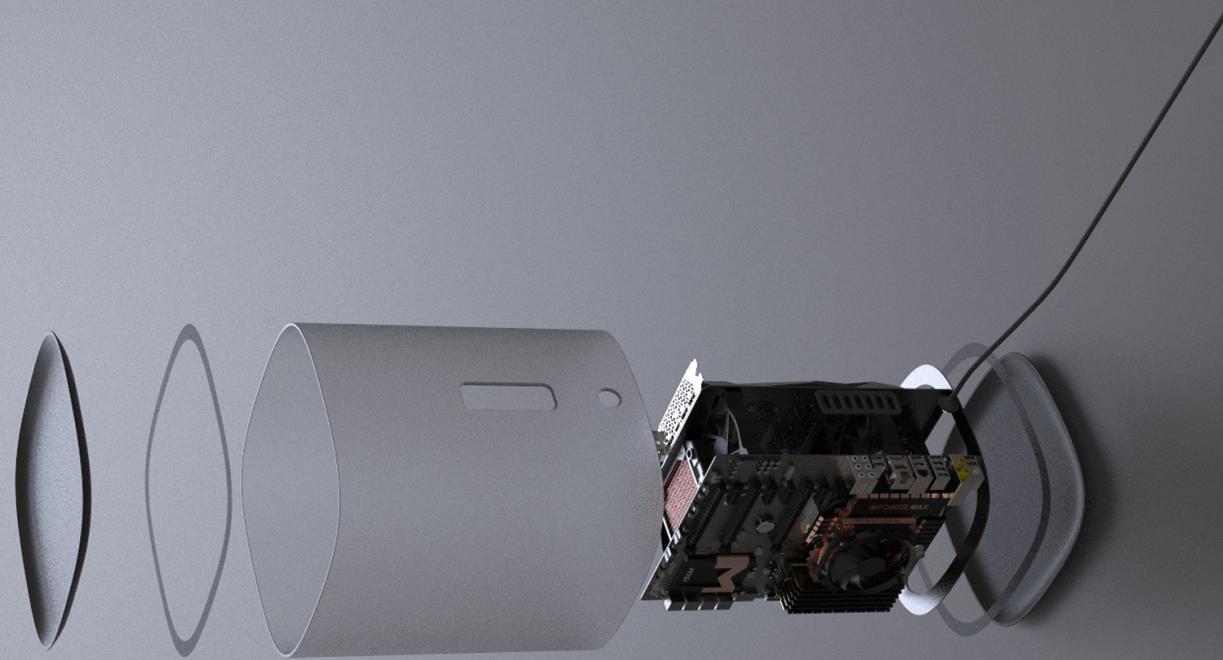






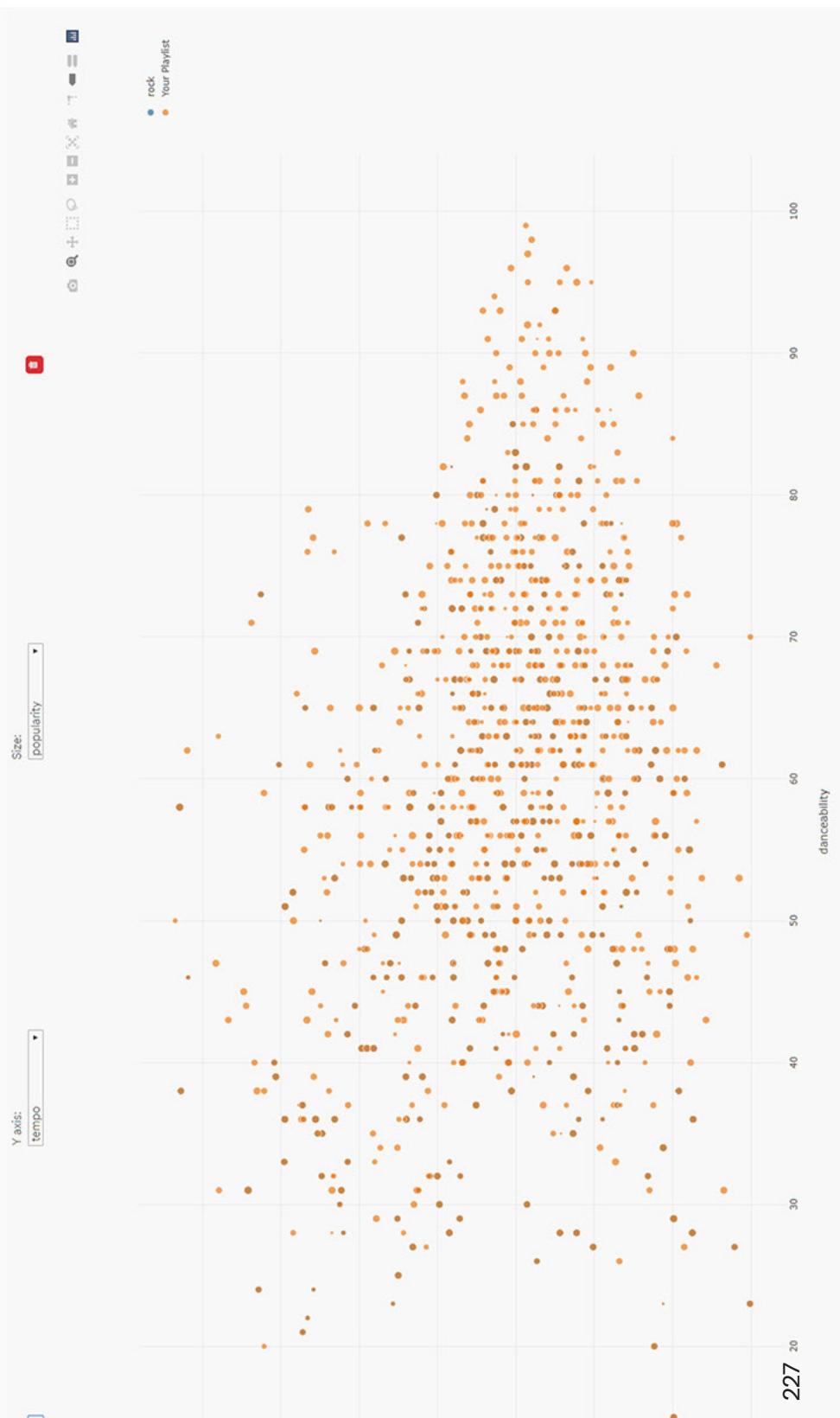






UPGRADE.

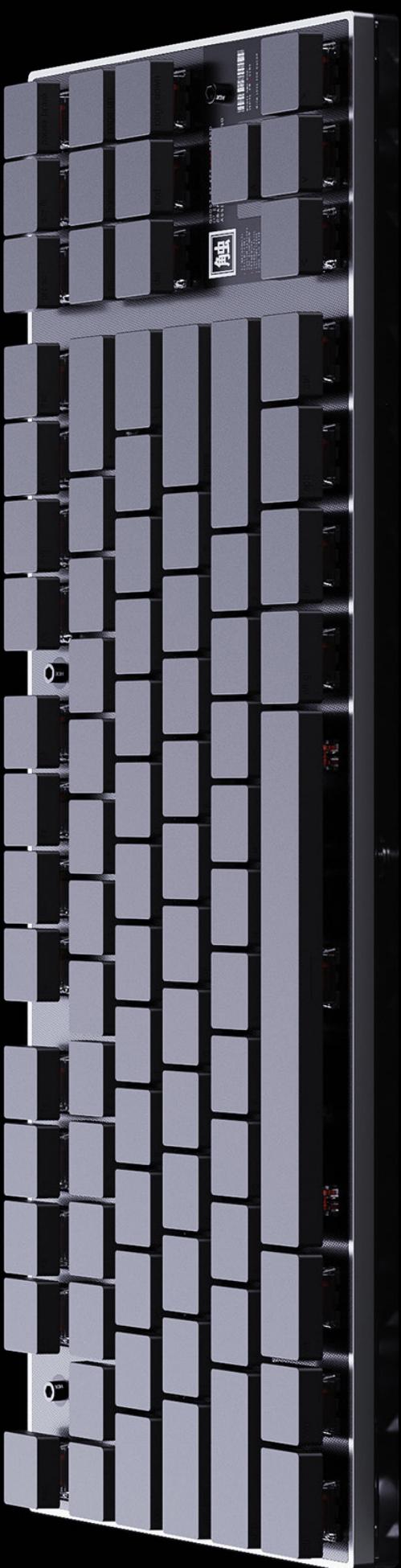
CUBE IS VERY EASY TO
KEEP UP TO DATE WITH
STANDARD COMPONENTS

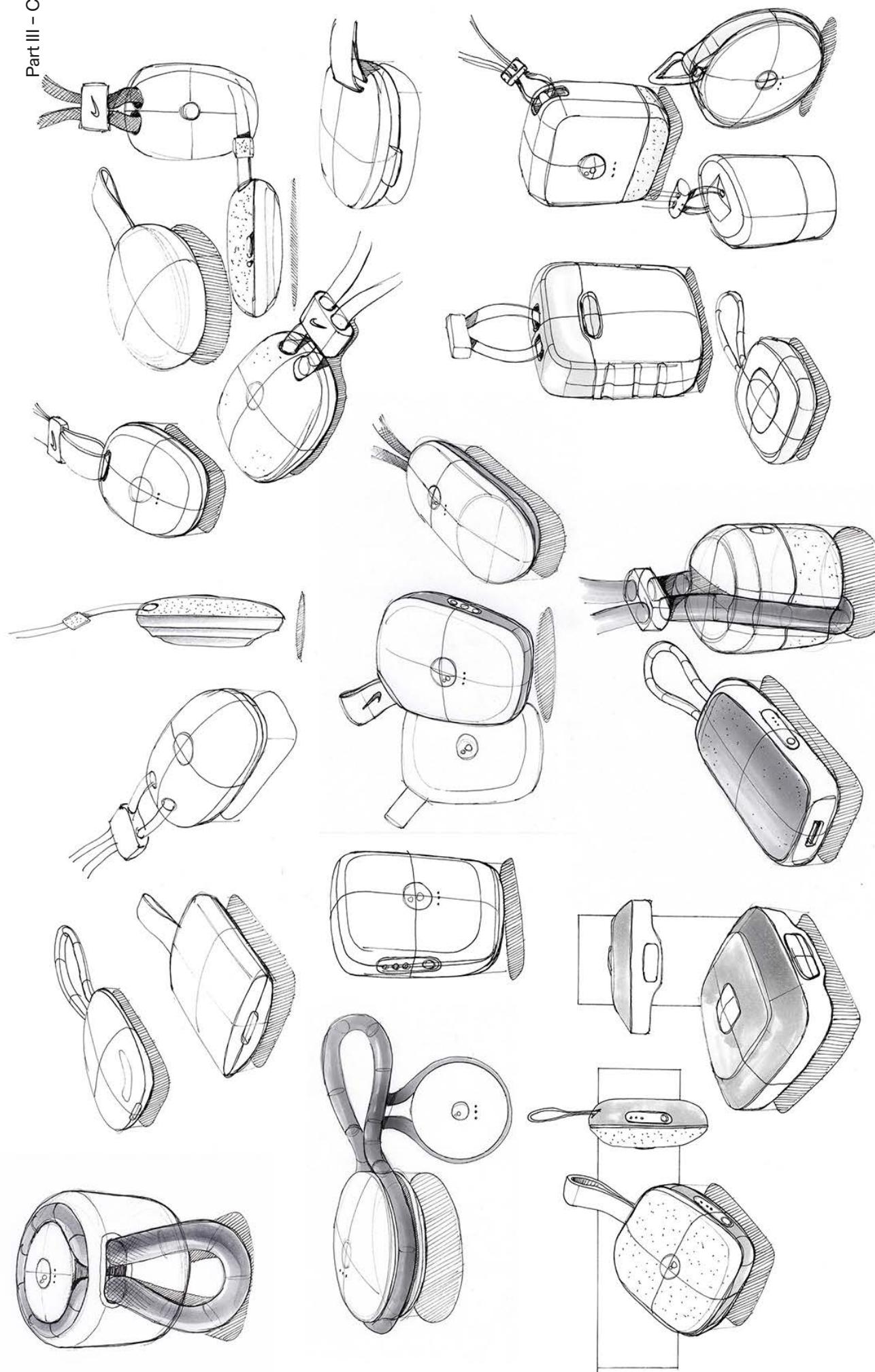


CUBESlim







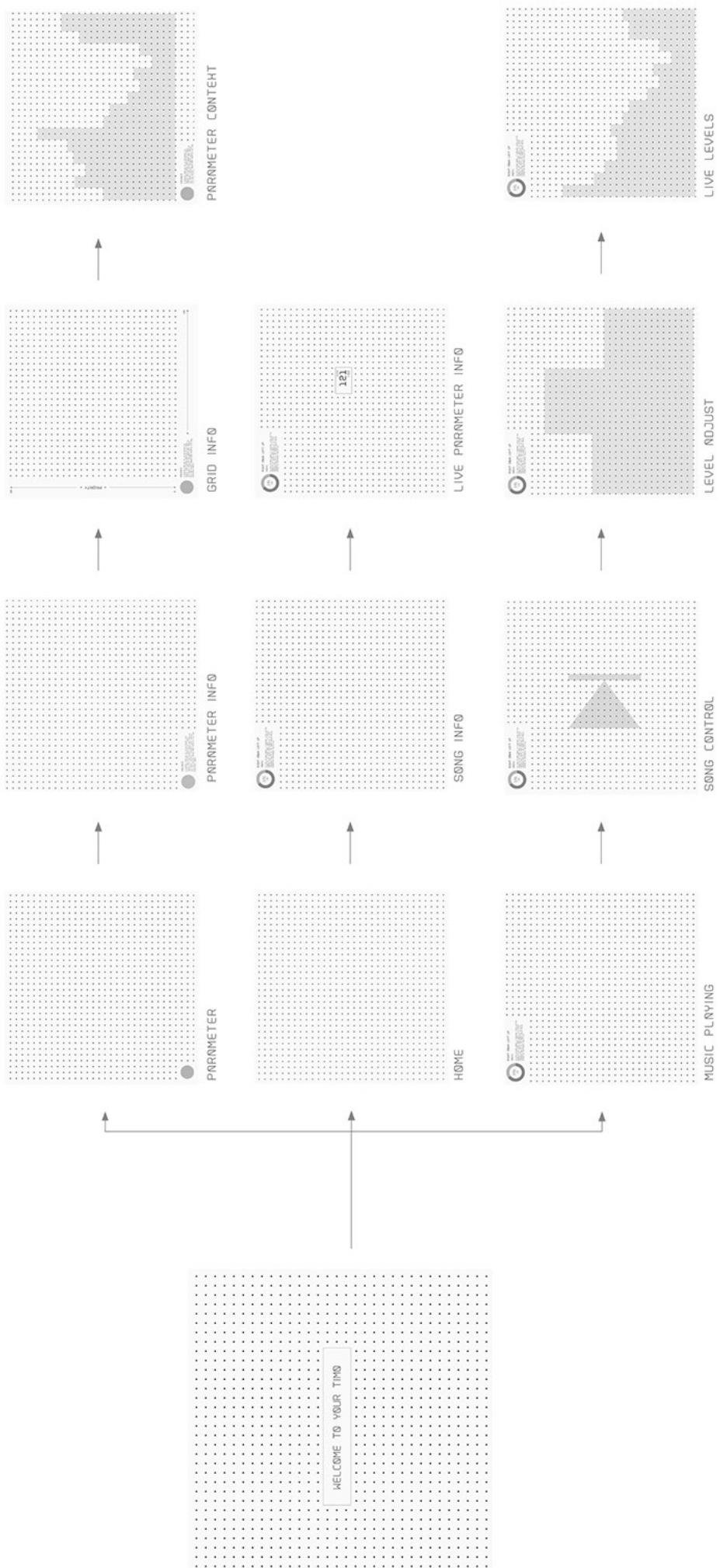




TIMO

TANGIBLE INTERACTION MUSIC ORGANIZER

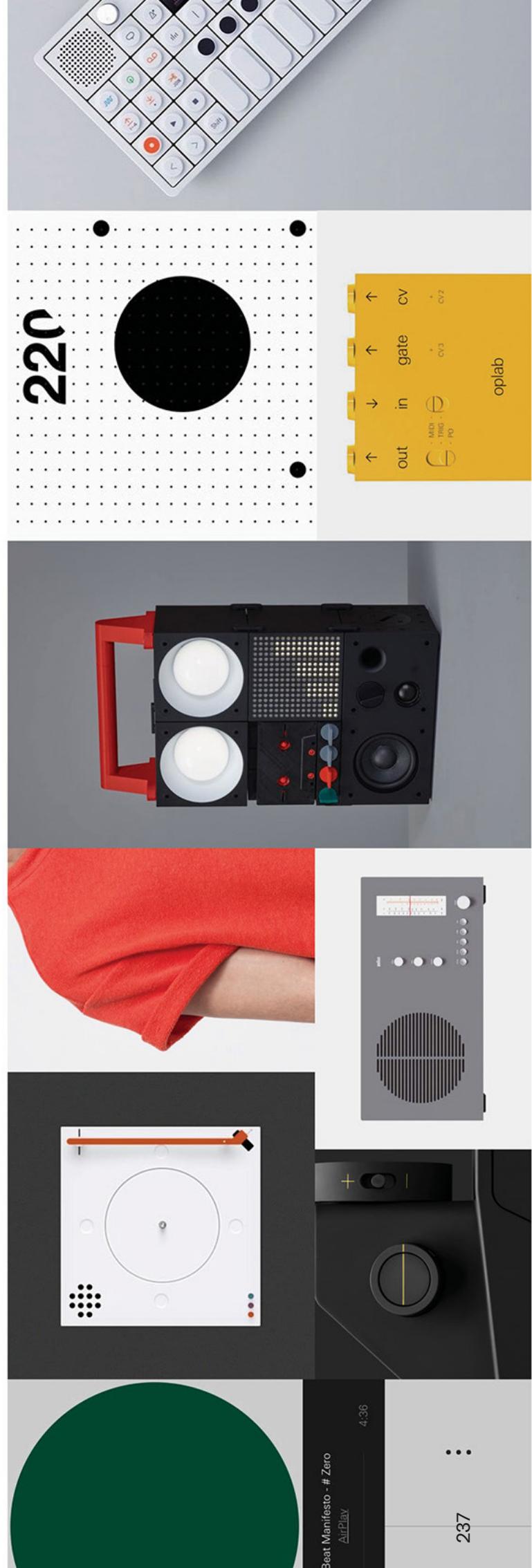






TANGIBLE INTERACTION AS INSPIRATION

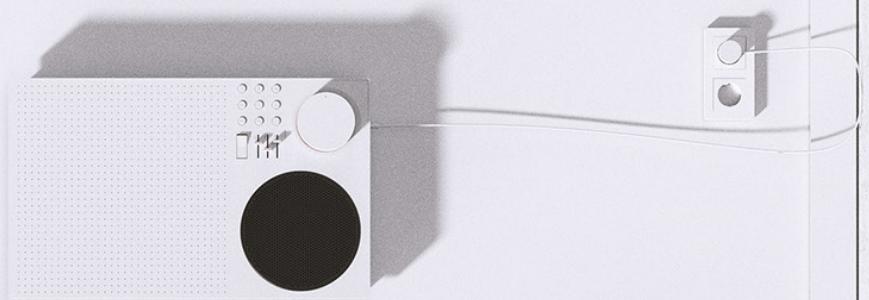
The idea of designing for a future of information interaction. We came from a world of static information, into digital information, and moving towards fluid information











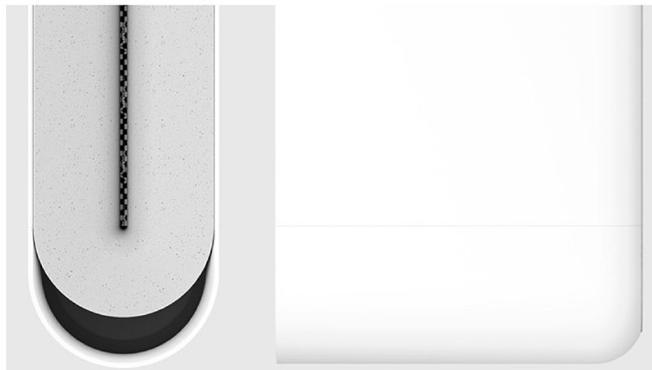
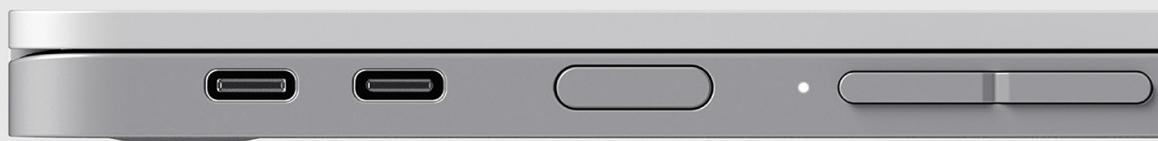








Style board.





OUTPUTS

The music community takes pride in the ability to customize things how they want. From music creators to editors to just plain music enthusiasts. Looking at traditional music playing options as inspiration for a community that loves variety, options, and customizability.

6.35mm Jack

3.5mm Jack

USB-C

Internal Speakers

Wireless connection





TIMØ

Using physical movement and expression to alter the way we interact with digital information in the form of a music organizing interface.

Tangible
Interaction
Music
Organizer

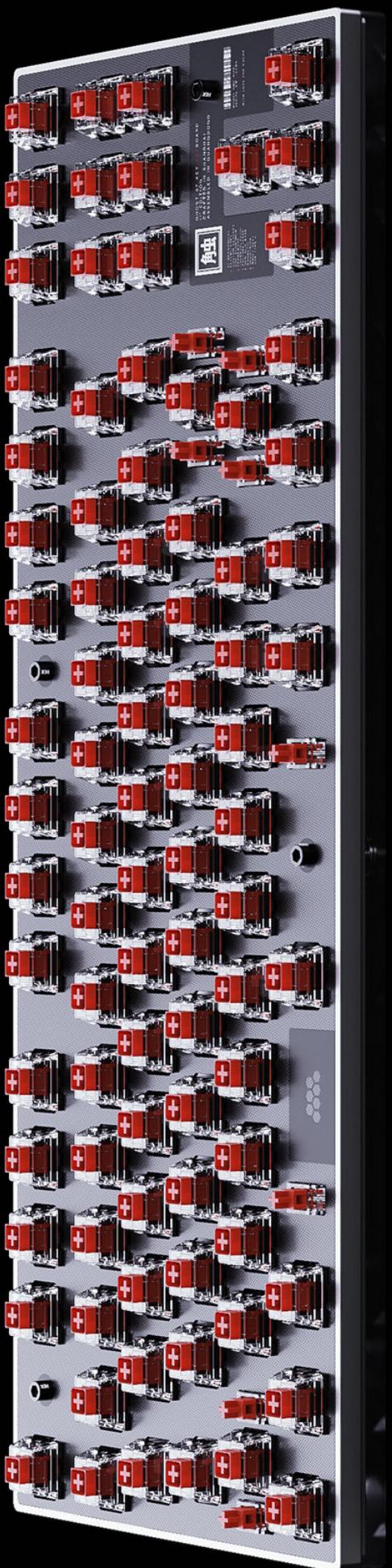


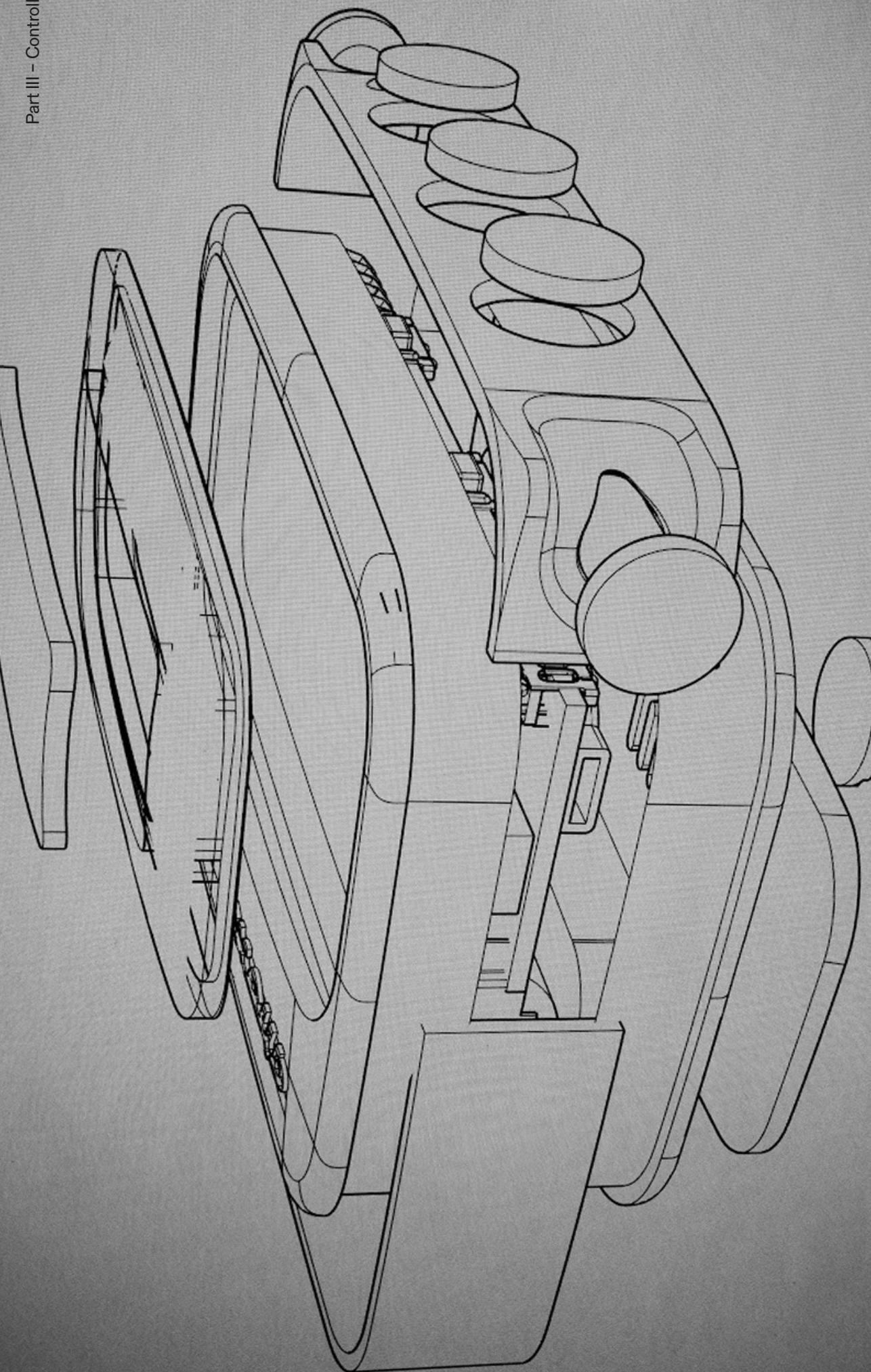


















8 Voice Digital Drums

CYBERSPACE





3.5 Final renders

The final renders of the controller were made in KeyShot 9 software I was also very familiar with. Its rich library of materials, flexible lightning and environment settings, and overall ease of use allowed to set the whole perception of the model on a completely different level.







СВЕТОУСТАНОВКА
ПРИМЕНЕНИЯ
GENERAL

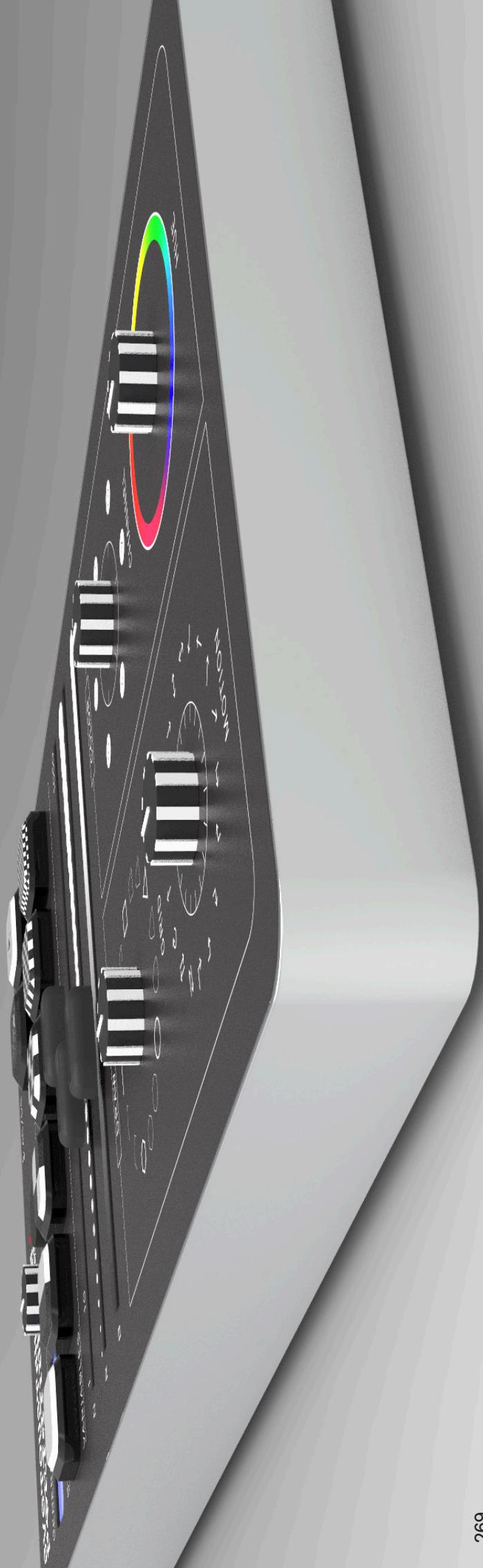
133

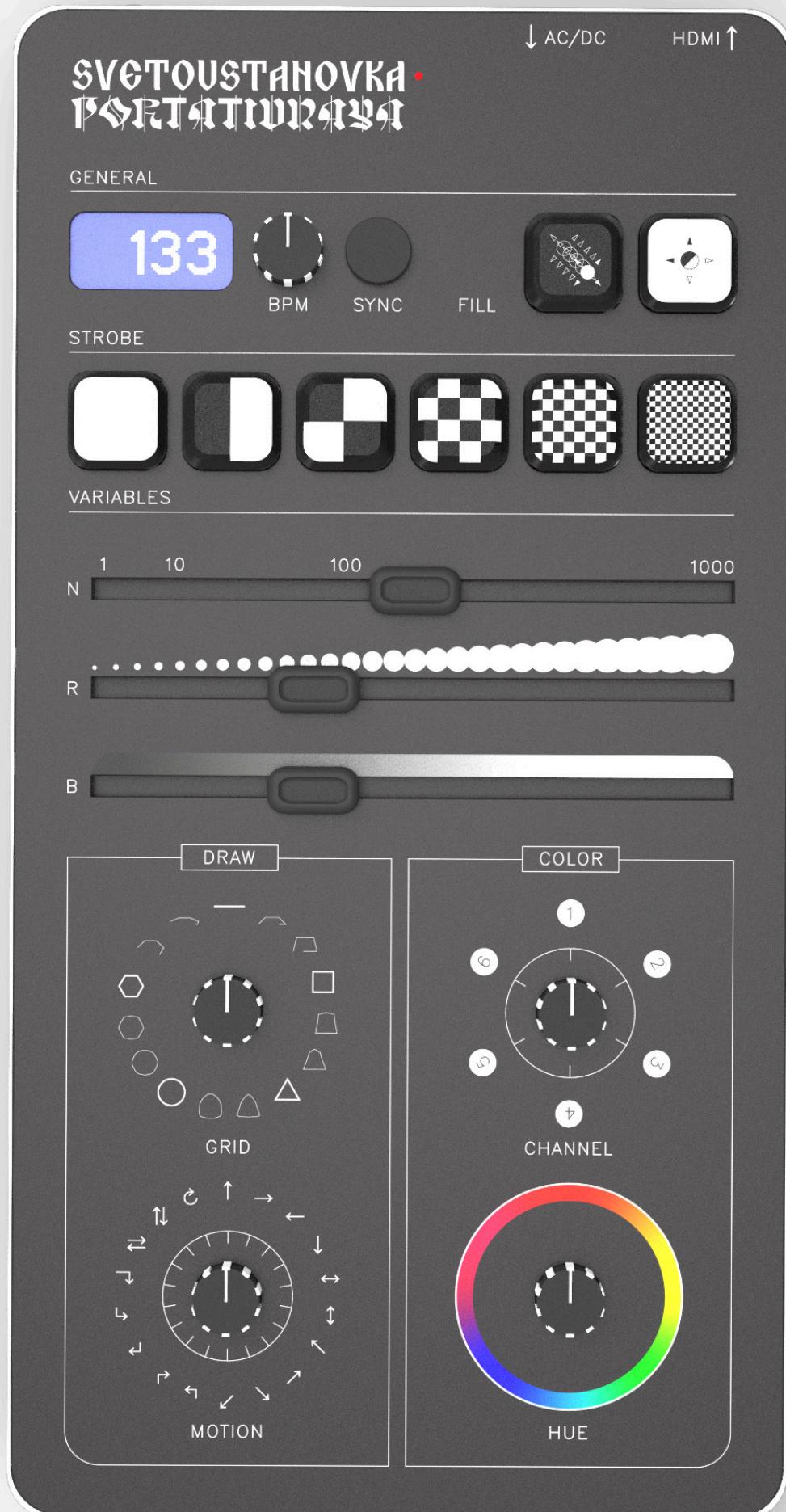
STROBE

BPM

SYNC

FILL





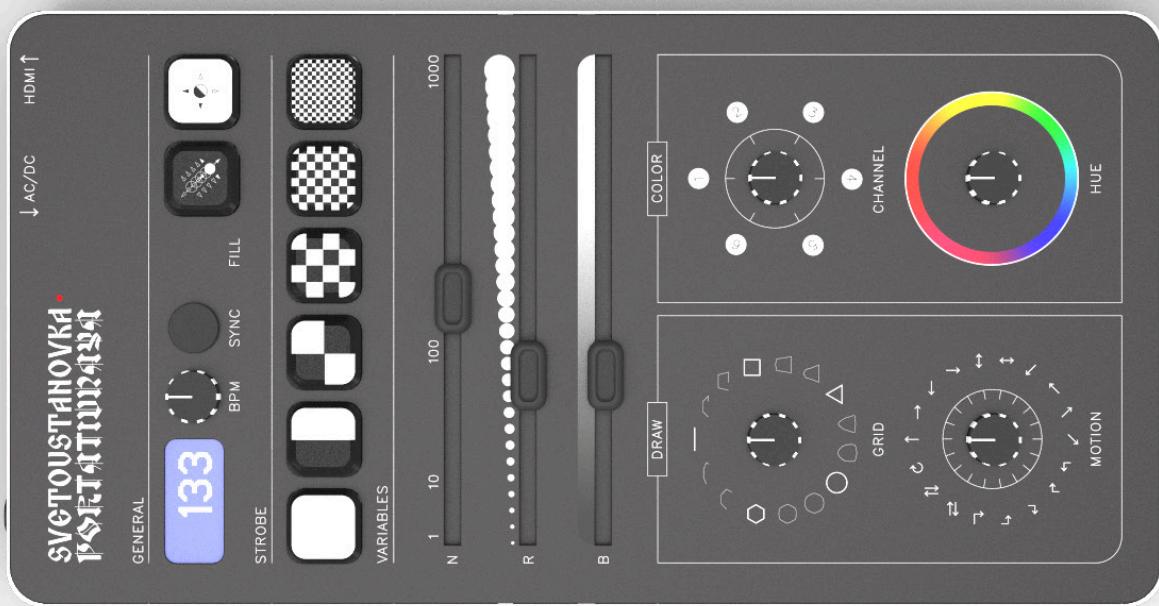


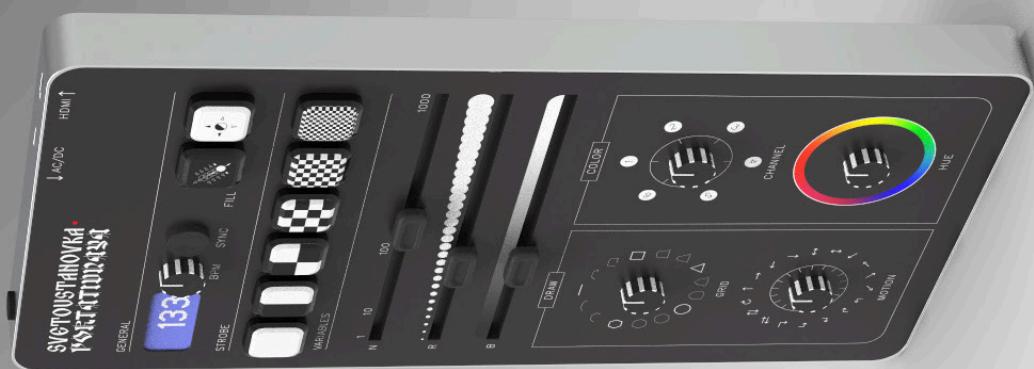












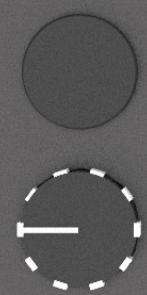


SVETOUTSTANOVKA
ПОВТОРЯЮЩИЕСЯ

HDMI ↑
↓ AC/DC

GENERAL

133



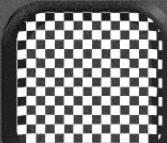
BPM SYNC



FILL



STROBE



VARIABLES



1000

100

10

1

N

R











COLOR

10

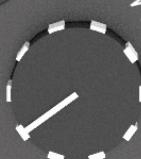
VARIABLES

100

STROBE

133

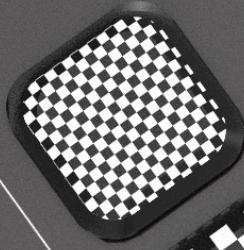
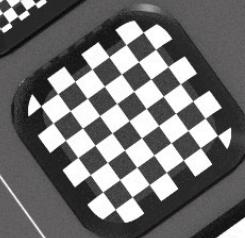
GENERAL



BPM

SYNC

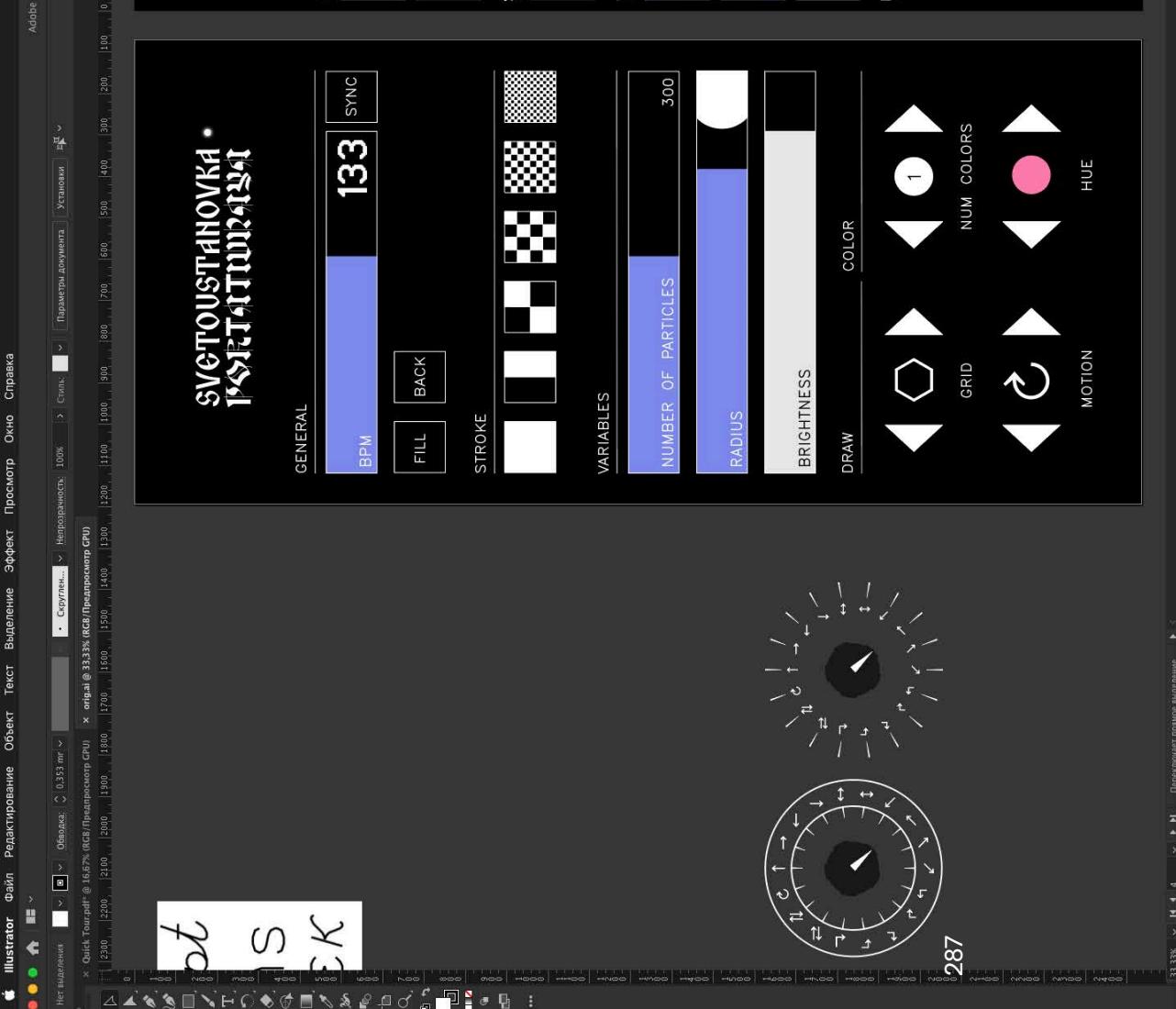
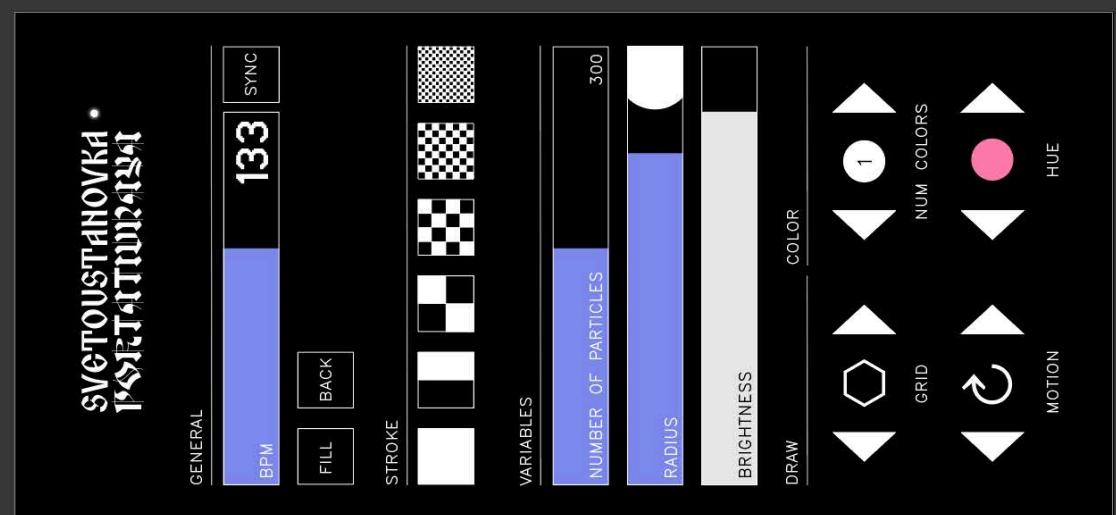
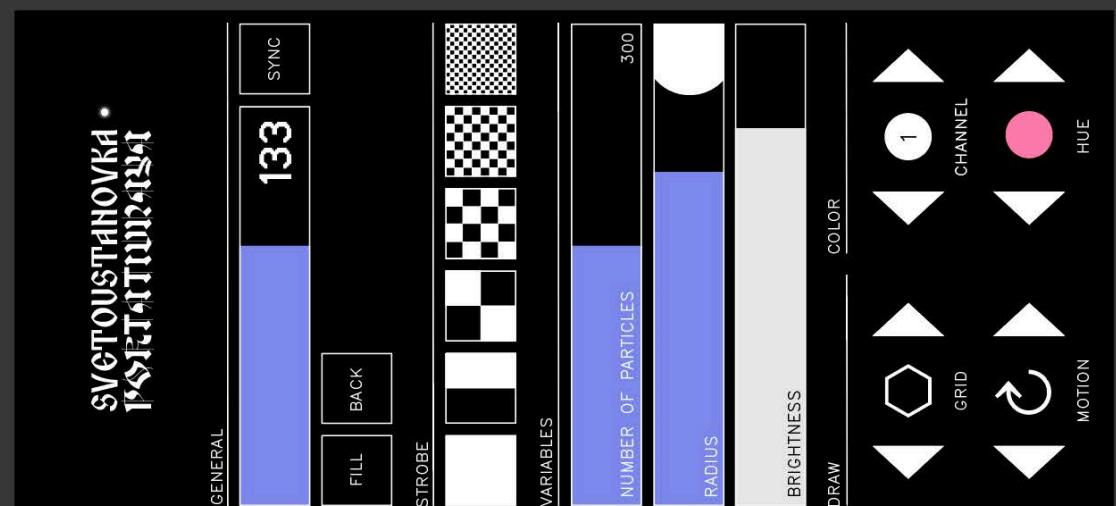
FL

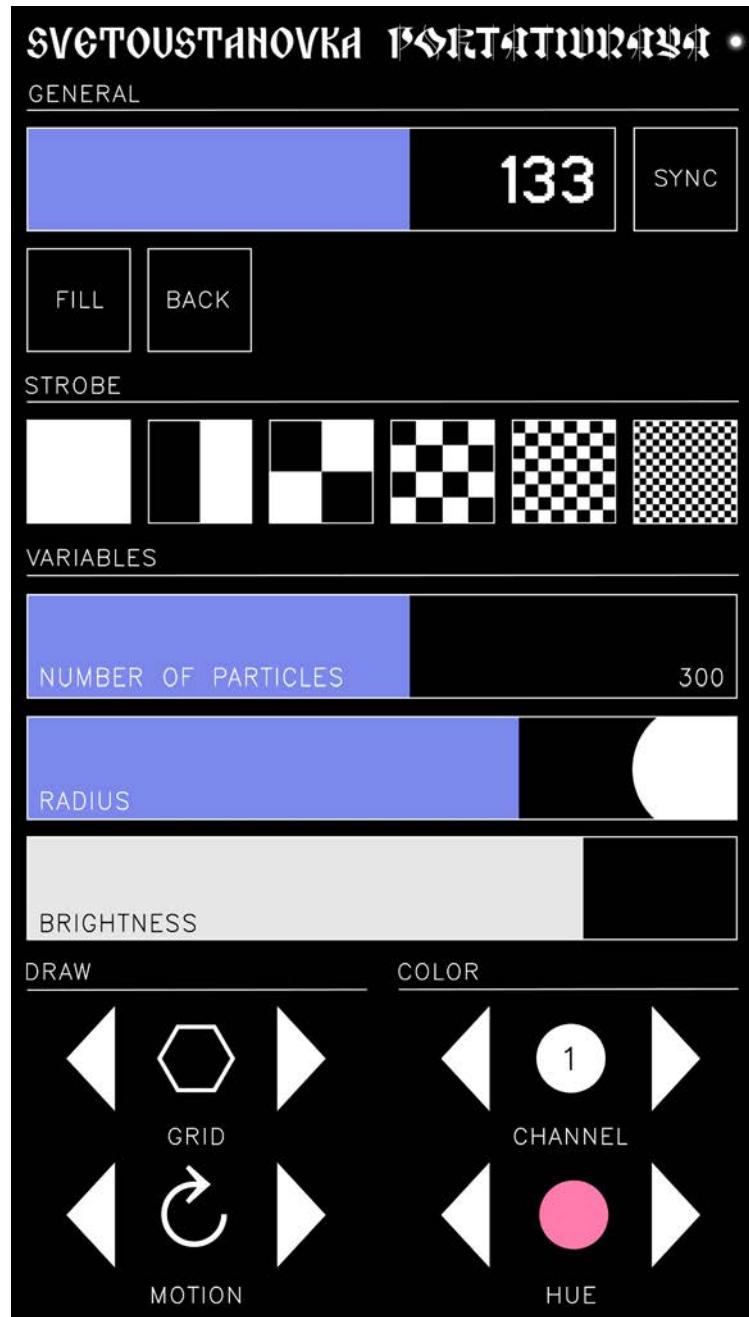


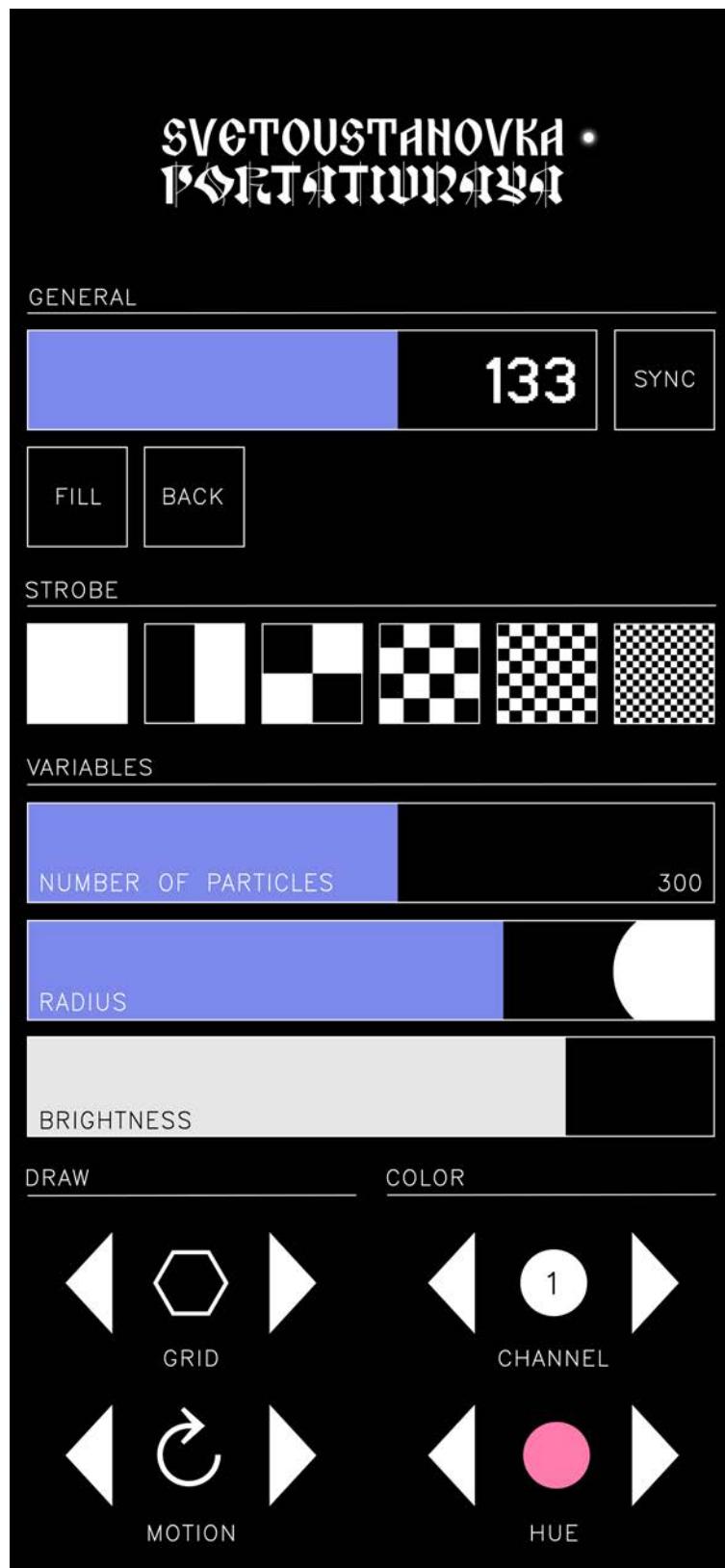
SLETOUTSTAMOVRK

3.6 iPhone prototypes

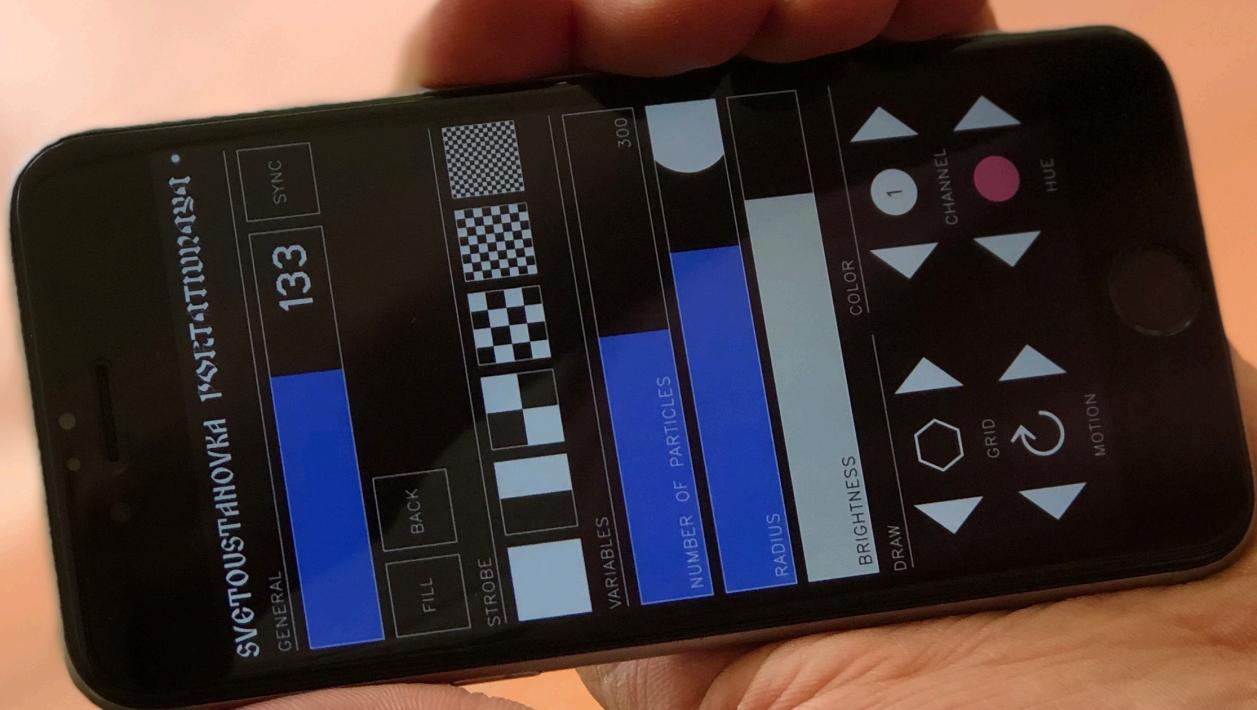
Thinking how to best translate the controls of the controller to demo it without the use of physical switchers and sliders, I decided to also create a prototype for the iPhone app, thinking that the touch control would be at least a bit better than the extremely poor control of using the mouse. Unfortunately, adapting the code to the iOS required more time than expected, so this idea remained in the form of a sketch for both older and newer iPhones with different screen size ratios as well as the prototype video manually made in Adobe Premiere Pro.











Part IV

Product

4.1 Presentation

Even though I really wanted to stay away from the feel of my controller being alike Apple Inc products, with the final renders it became clear that my unconscious still had a very strong connection with their aesthetic and style. That is why for submitting the product I created a 'low-key apple keynote' ironically recalling many of its forms and memetic parts. The key foundation of the whole structure is the continuity between the slides, with the information casually unfolding along the slide changing.

СВЕТОУСТАНОВКА
ПОЛЯРИЗАЦИИ

• **WELCOMING**

Have you ever done a light show before?

No?
Have you ever done a light show before?

**Have you ever done a light show before?
No? Ever wondered why not?**

Have you ever done a light show before?

No? Ever wondered why not?

Maybe because of the current lighting solutions which look like

**Have you ever done a light show before?
No? Ever wondered why not?**

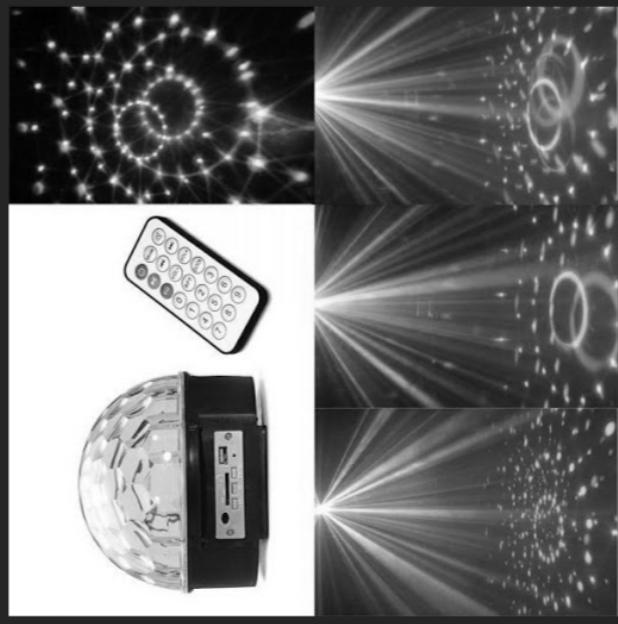
Maybe because of the current lighting solutions which look like *this*



Have you ever done a light show before?

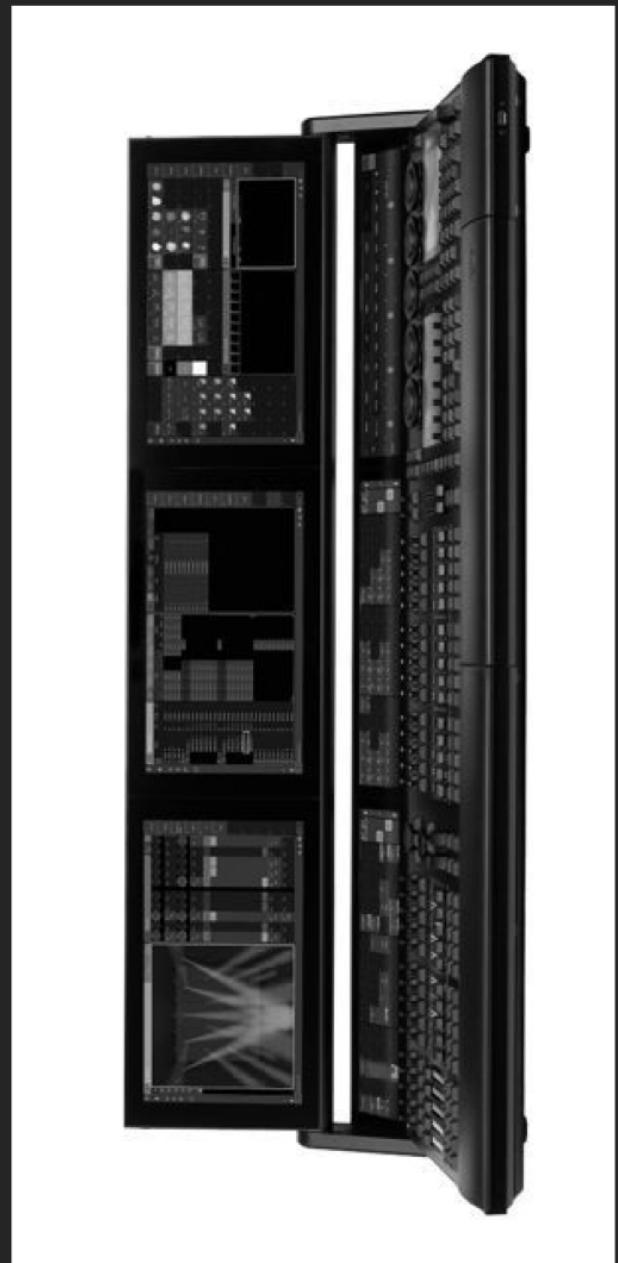
No? Ever wondered why not?

Maybe because of the current lighting solutions which look like *this* or *like*



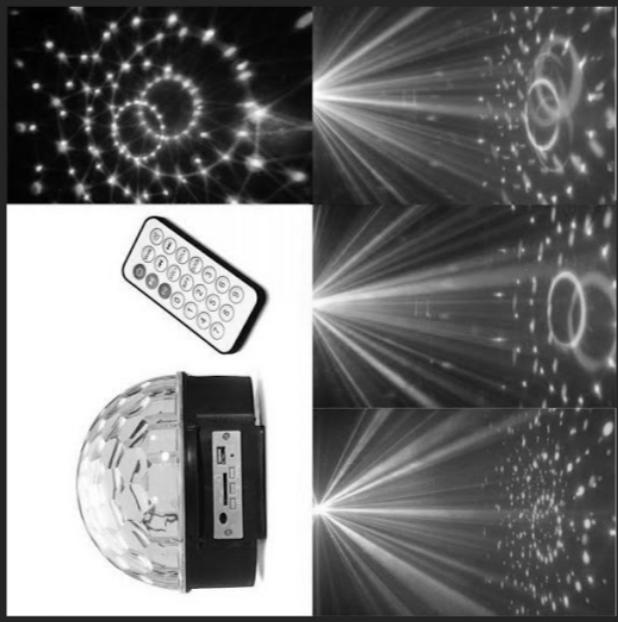
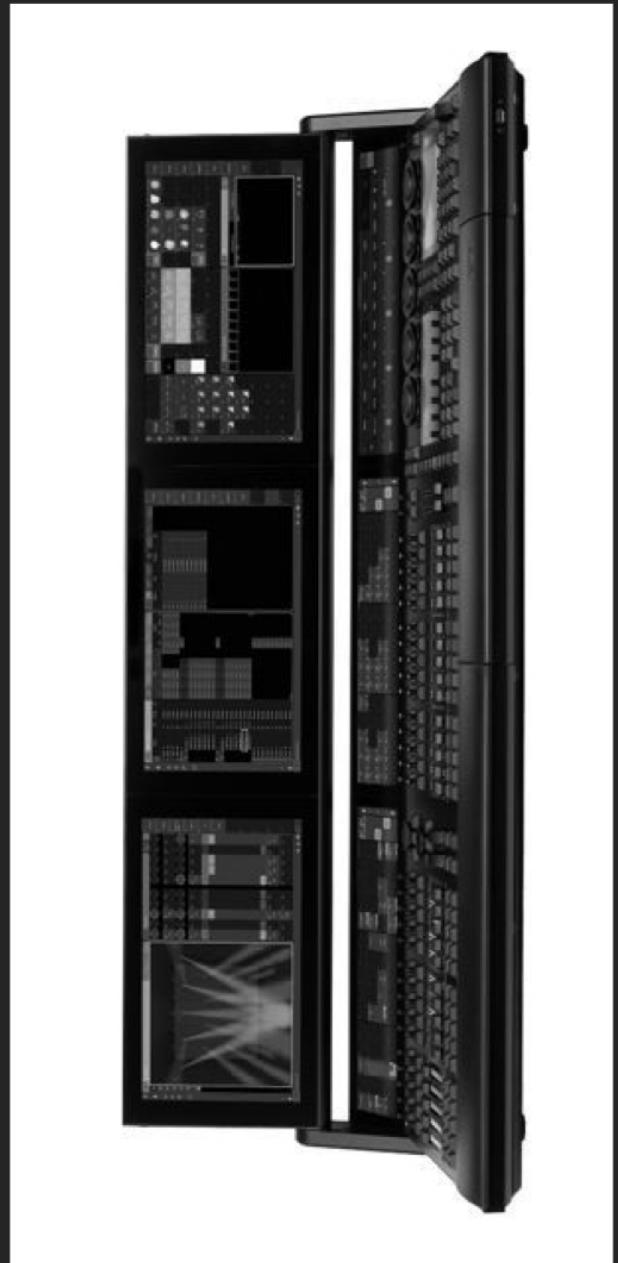
**Have you ever done a light show before?
No? Ever wondered why not?**

Maybe because of the current lighting solutions which look like *this* or like *this*?



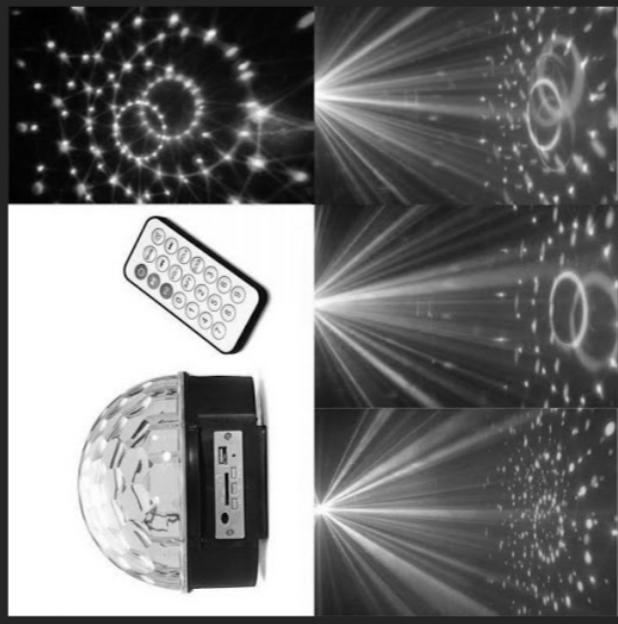
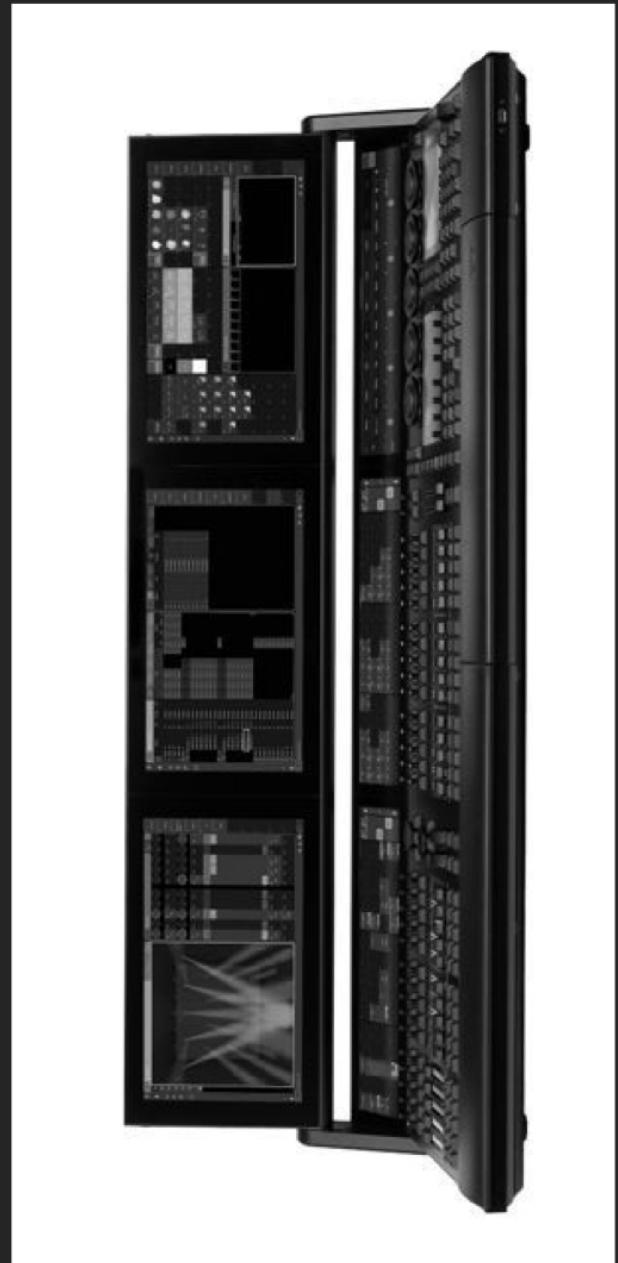
Have you ever done a light show before?
No? Ever wondered why not?

Maybe because of the current lighting solutions which look like *this* or like *this*?
Yeah, no surprise you didn't even thought about that before.



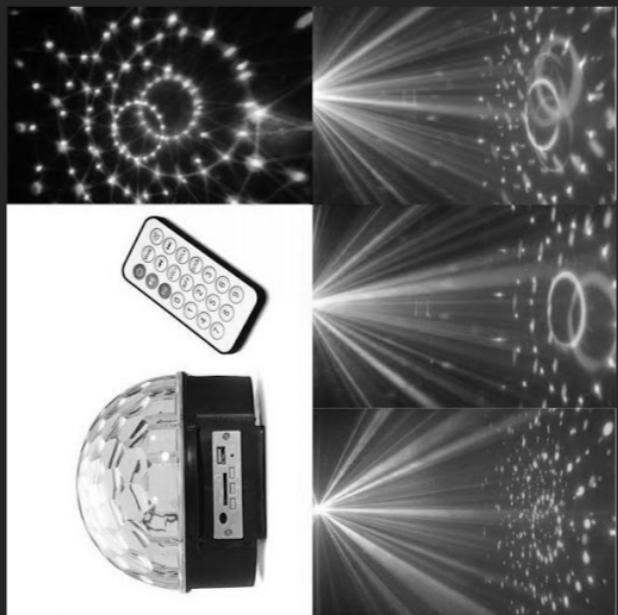
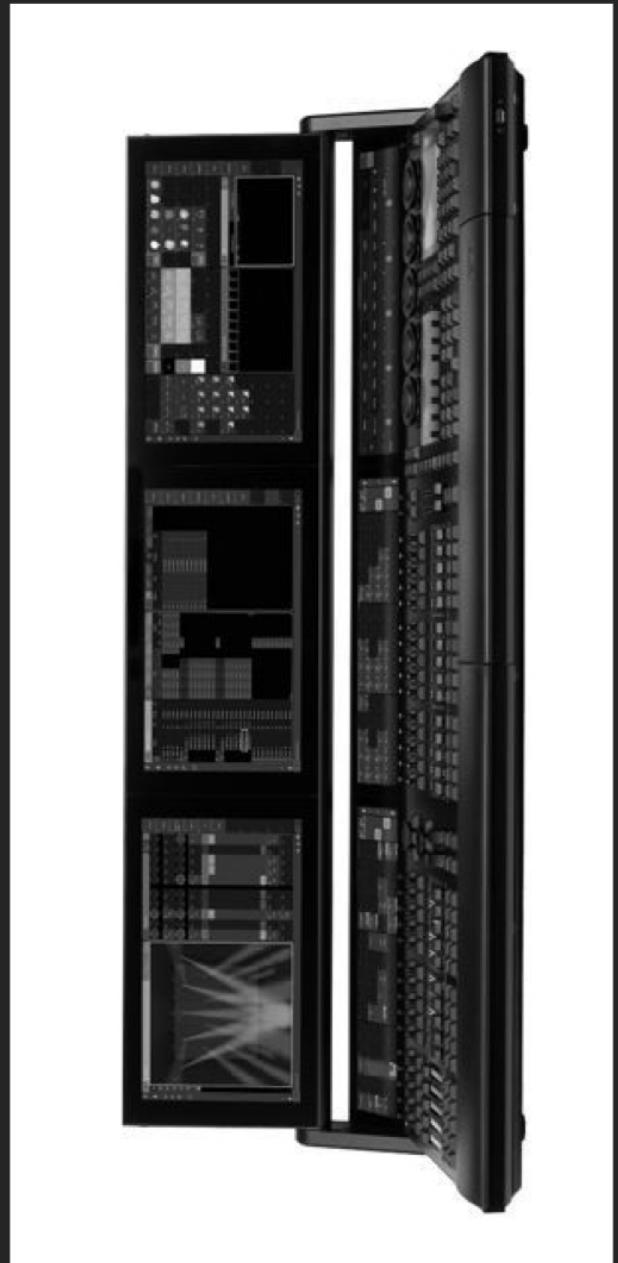
**Have you ever done a light show before?
No? Ever wondered why not?**

**Maybe because of the current lighting solutions which look like *this* or like *this*?
Yeah, no surprise you didn't even thought about that before.
But now you can.**



**Have you ever done a light show before?
No? Ever wondered why not?**

**Maybe because of the current lighting solutions which look like *this* or like *this*?
Yeah, no surprise you didn't even thought about that before.
But now you can.
Seriously, we've got you covered.**



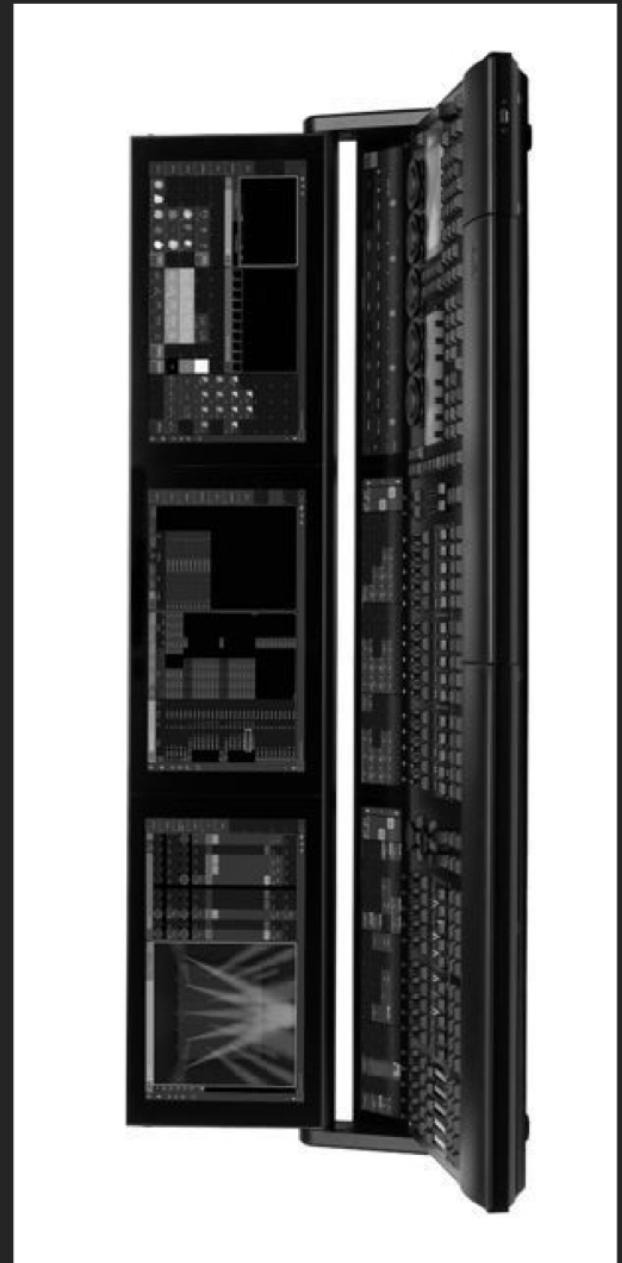
Have you ever done a light show before?

No? Ever wondered why not?

Maybe because of the current lighting solutions which look like *this* or like *this*?
Yeah, no surprise you didn't even thought about that before.

But now you can.

**Seriously, we've got you covered.
Introducing**



Introducing

Introducing the new

Introducing the new portable,

Introducing the new portable, amateur oriented,

Introducing the new portable, amateur oriented, PC independent,

Introducing the new portable, amateur oriented, PC independent, & just simply beautiful

Introducing the new portable, amateur oriented, PC independent, & just simply beautiful light controller

Introducing the new light controller

Light controller

5

6

7

COLOR

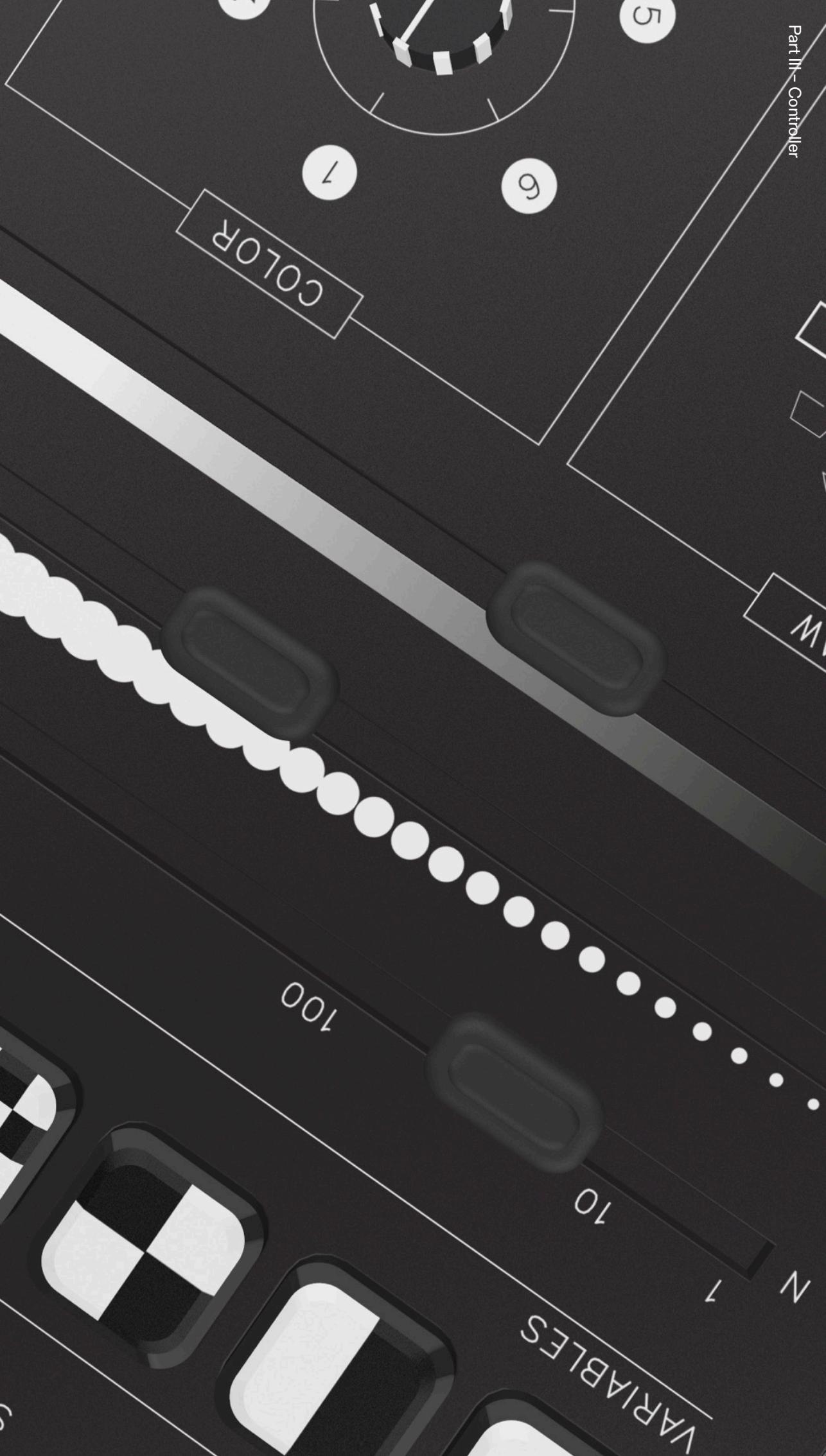
W

100

10

7

VARIABLES



УСТАНОВКА
УПРАВЛЕНИЯ

AC/DC

HDMI









СВЕТОУСТАНОВКА

ПОДСВЕТКА

Ok, but what's that for?

Ok, but what's that for?

It is for creating live light shows.

Ok, but what's that for?

It is for creating live light shows.

But I don't have any expensive lightning equipment!

Ok, but what's that for?

It is for creating live light shows.

But I don't have any expensive lightning equipment!

You don't need to. All you need is a projector of any resolution. Yet bright enough.

Ok, but what's that for?

It is for creating live light shows.

But I don't have any expensive lightning equipment!

*You don't need to. All you need is a projector of any resolution. Yet bright enough.
How does it work though?*

Ok, but what's that for?

It is for creating live light shows.

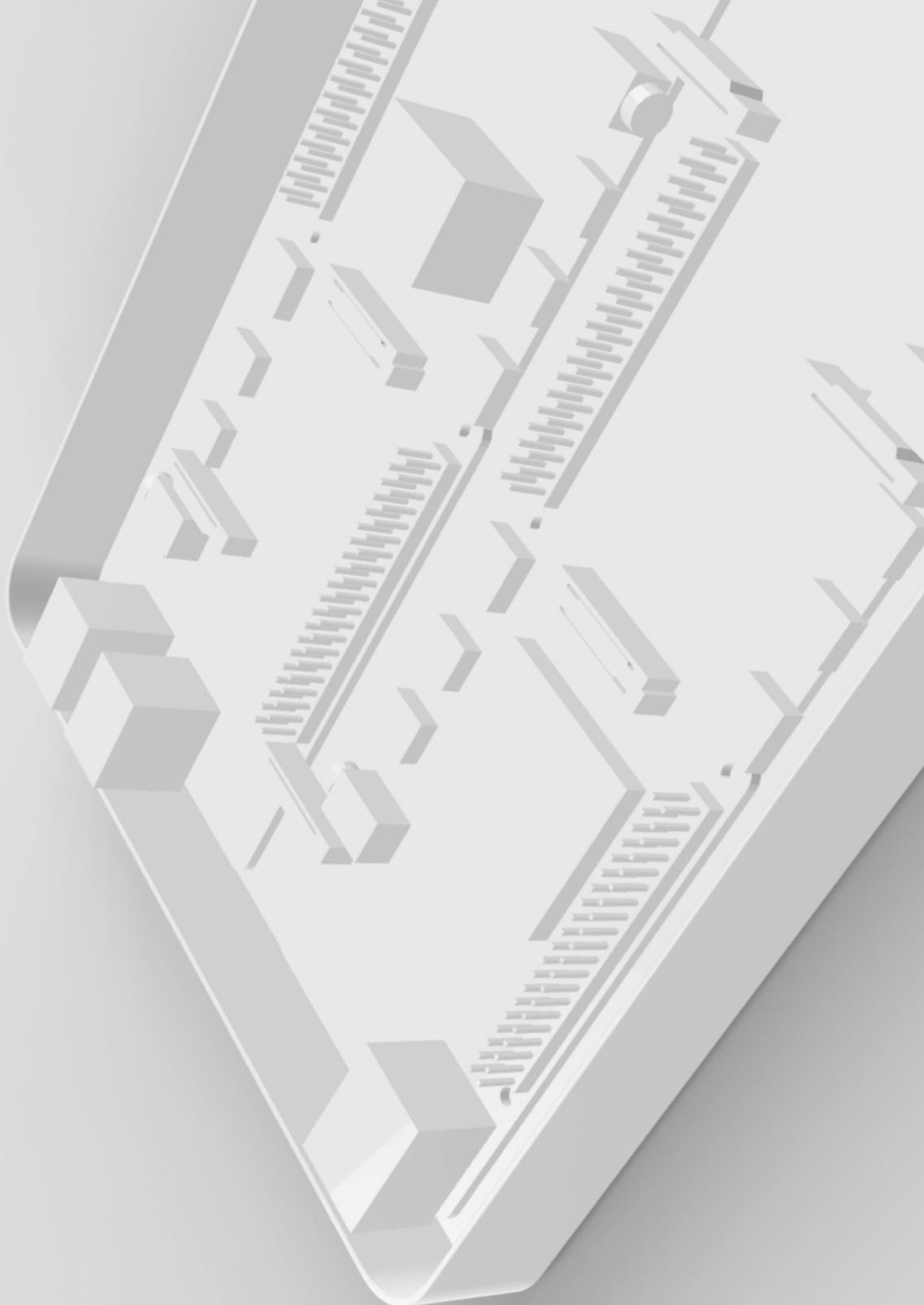
But I don't have any expensive lightning equipment!

You don't need to. All you need is a projector of any resolution. Yet bright enough.

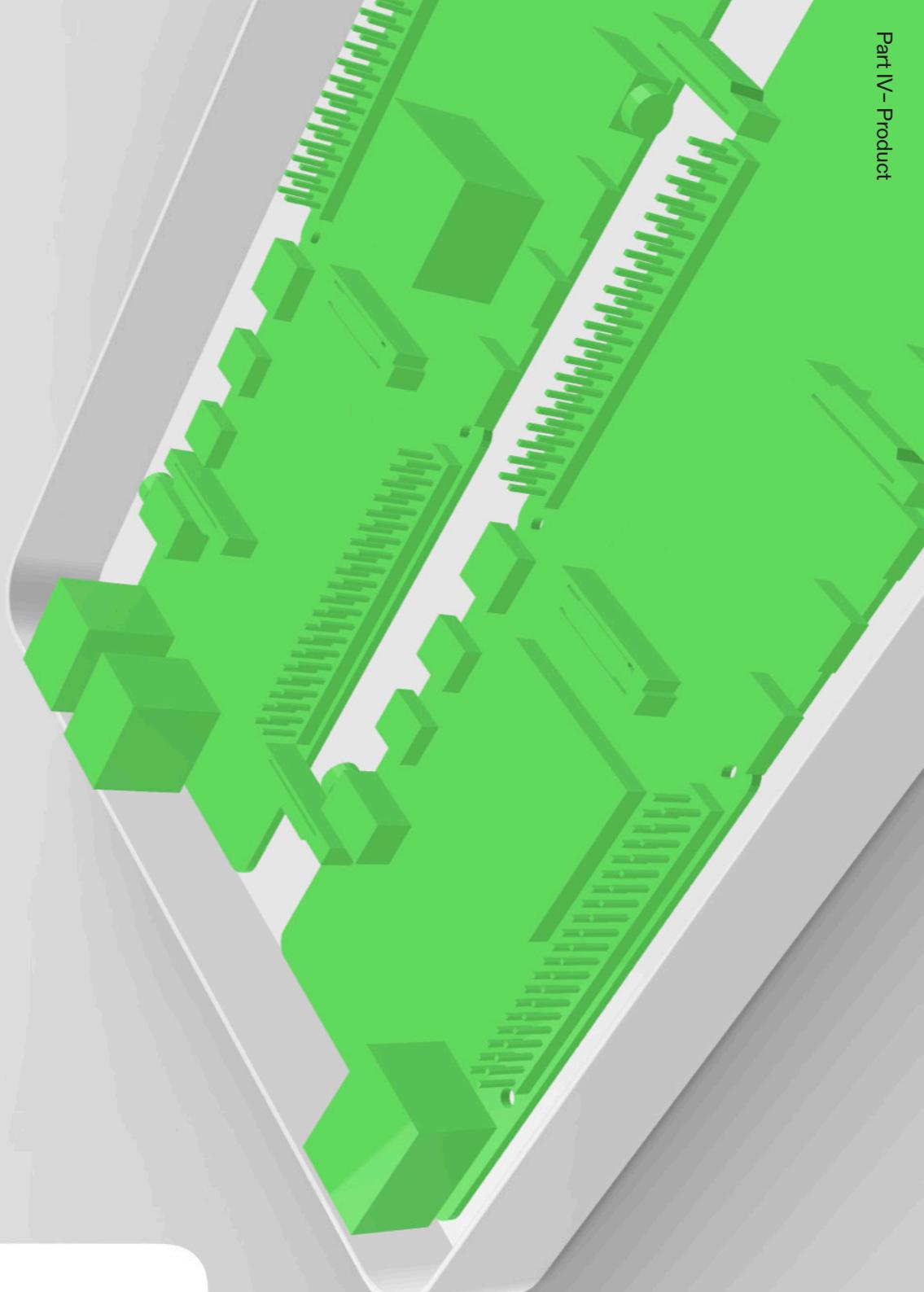
How does it work though?

Let us show.

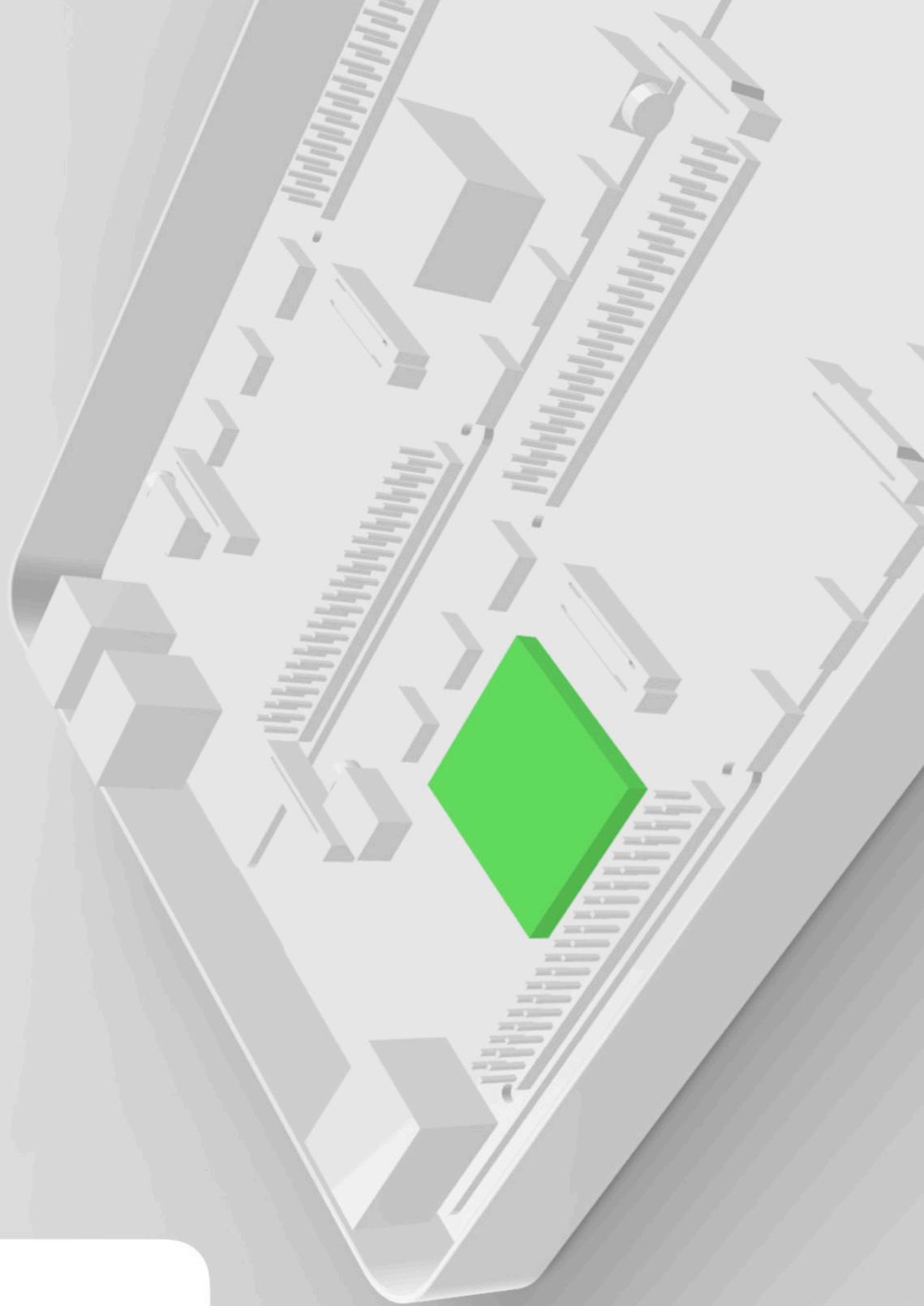
Let us show.



✓ Latest Raspberry Pi 4 Model B

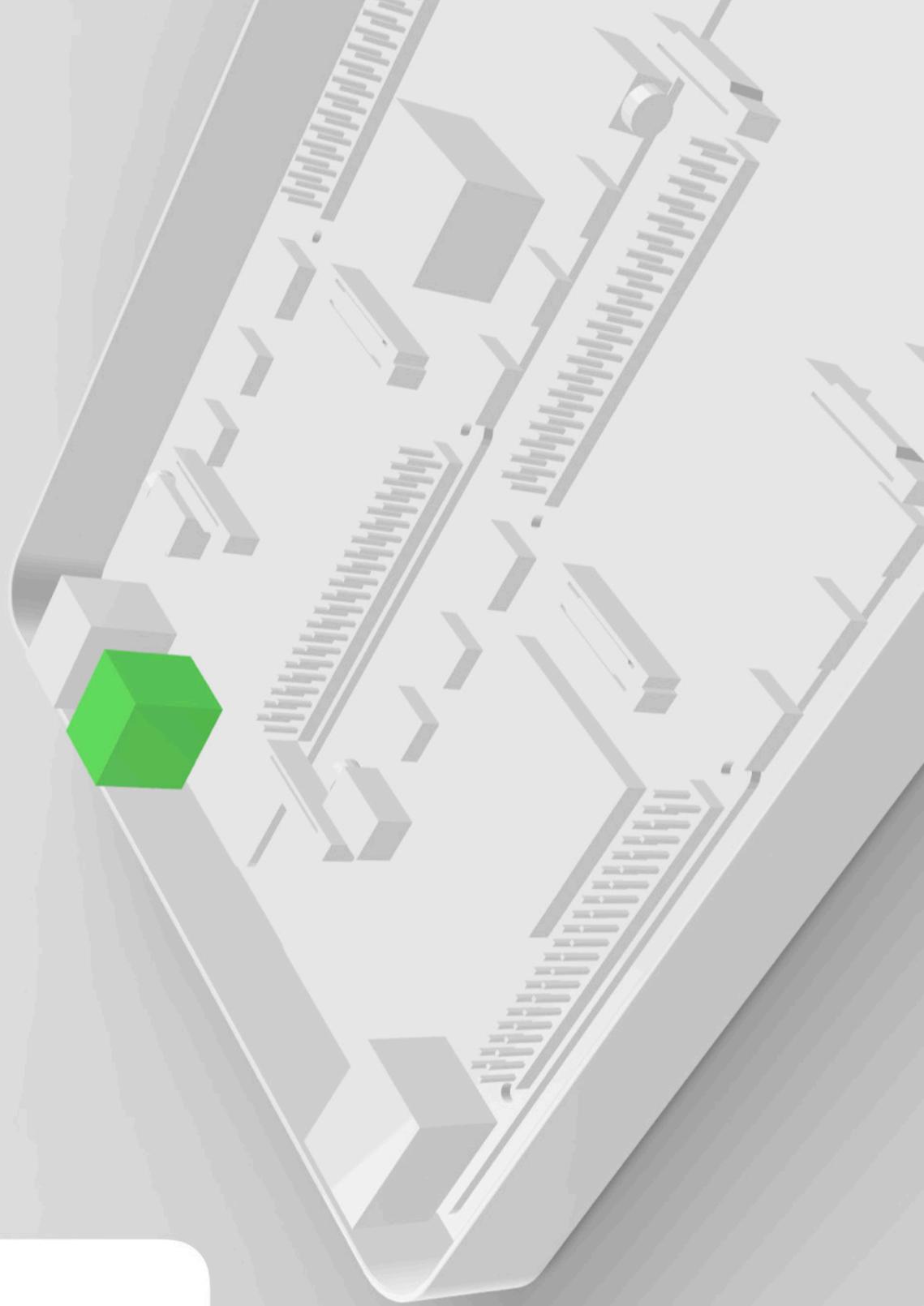


- ✓ Latest Raspberry Pi 4 Model B
- ✓ 8GB RAM

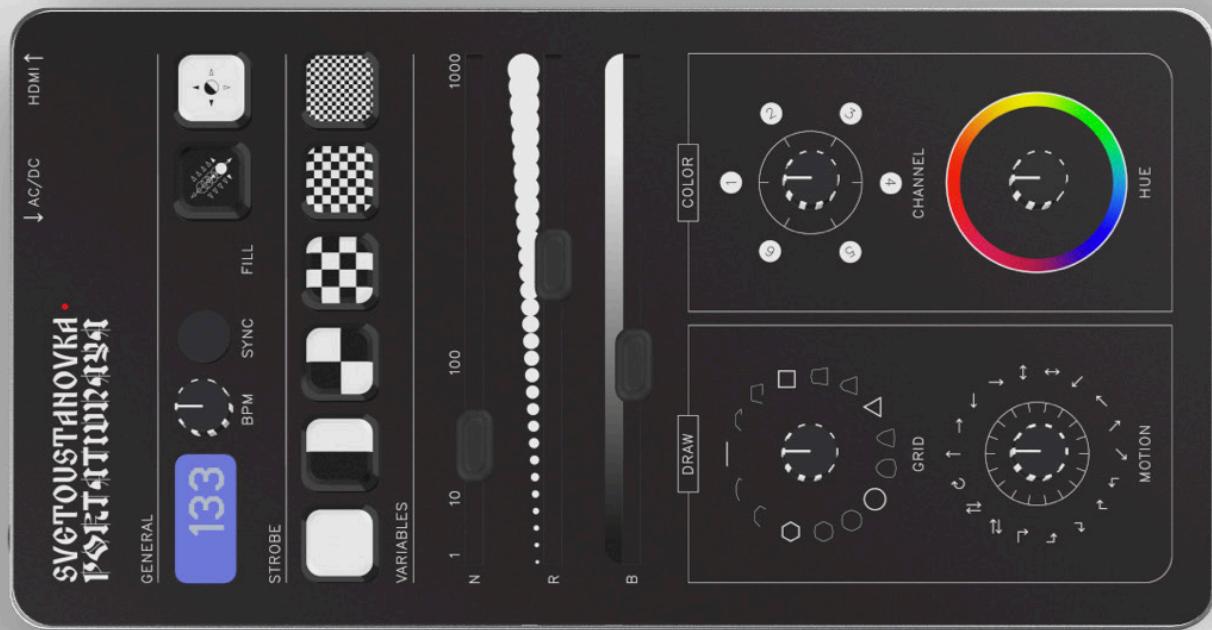


- 
- ✓ Latest Raspberry Pi 4 Model B
 - ✓ 8GB RAM
 - ✓ 60fps streaming up to 4K

- ✓ Latest Raspberry Pi 4 Model B
- ✓ 8GB RAM
- ✓ 60fps streaming up to 4K
- ✓ 5V DC via USB-C



- ✓ Latest Raspberry Pi 4 Model B
- ✓ 8GB RAM
- ✓ 60fps streaming up to 4K
- ✓ 5V DC via USB-C
- ✓ Fully computer independent



You turn in on.

СВЕТОУСТАНОВКА
ПОДСВЕТКА

↓ AC/DC HDMI ↑

GENERAL



STROBE

BPM

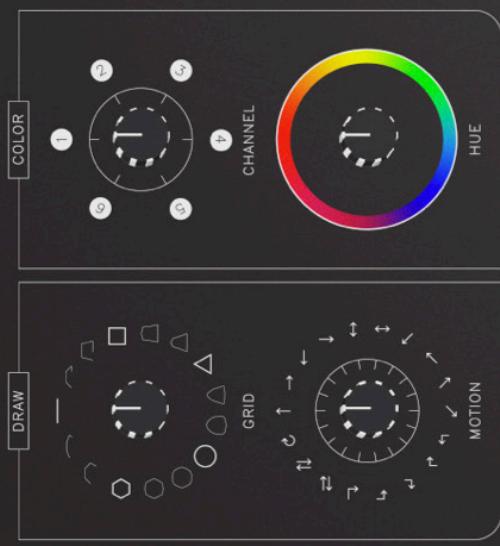
FILL

VARIABLES

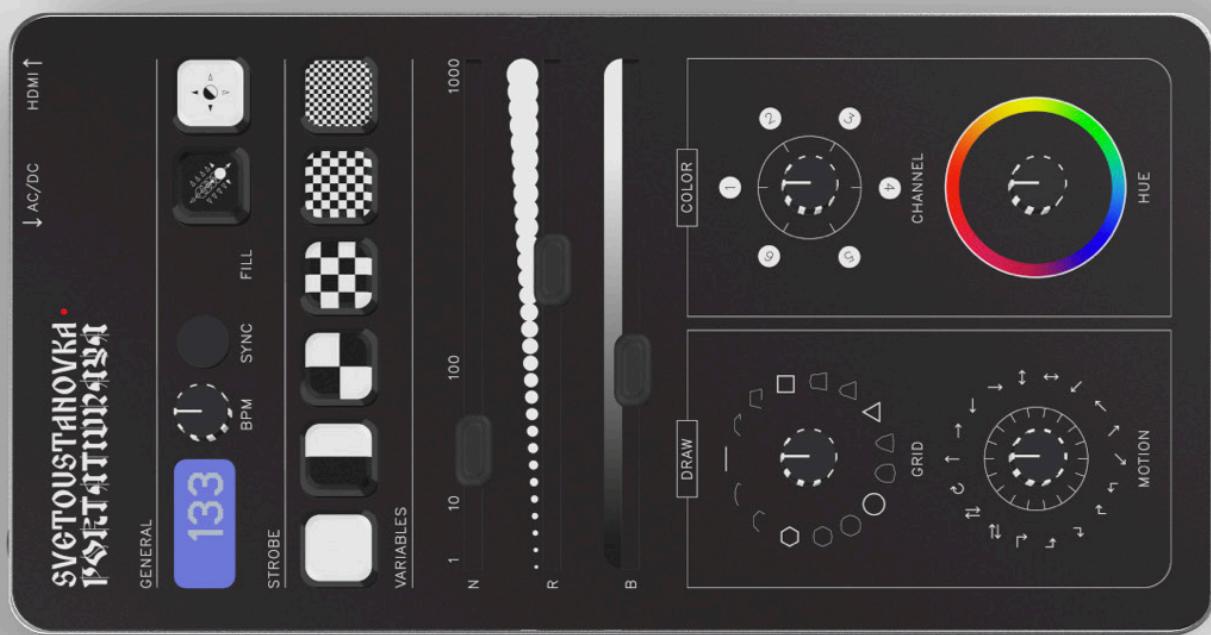
1 10 100 1000
N

R

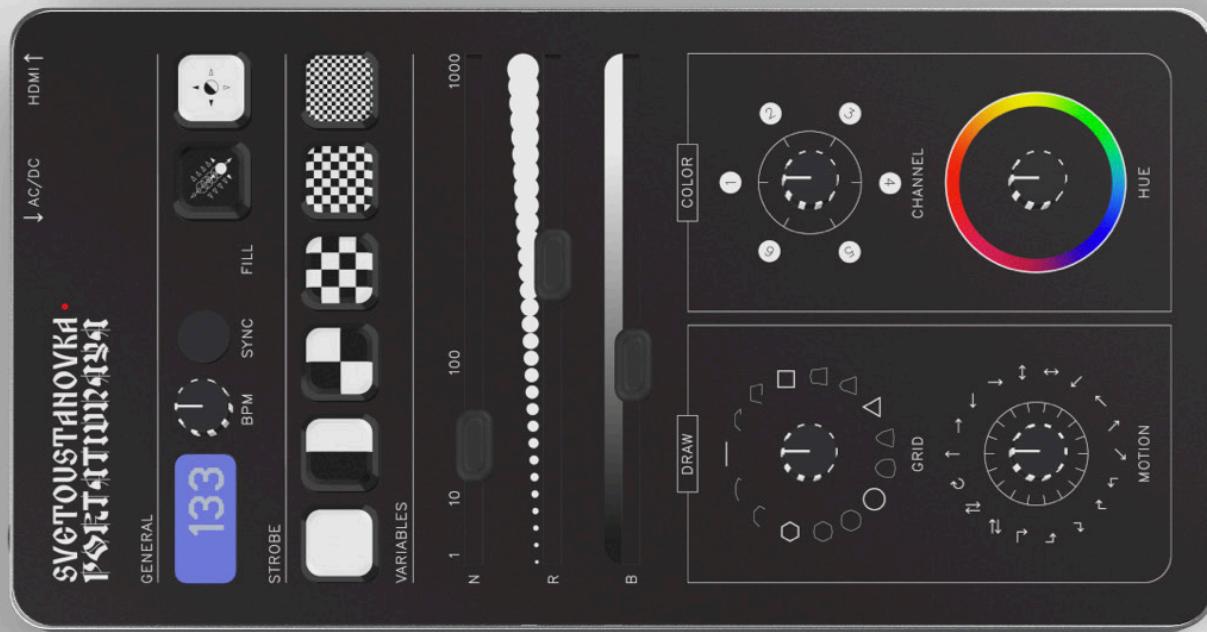
B



You turn in on.
You set the BPM.



You turn in on.
 You set the BPM.
 You SYNC.



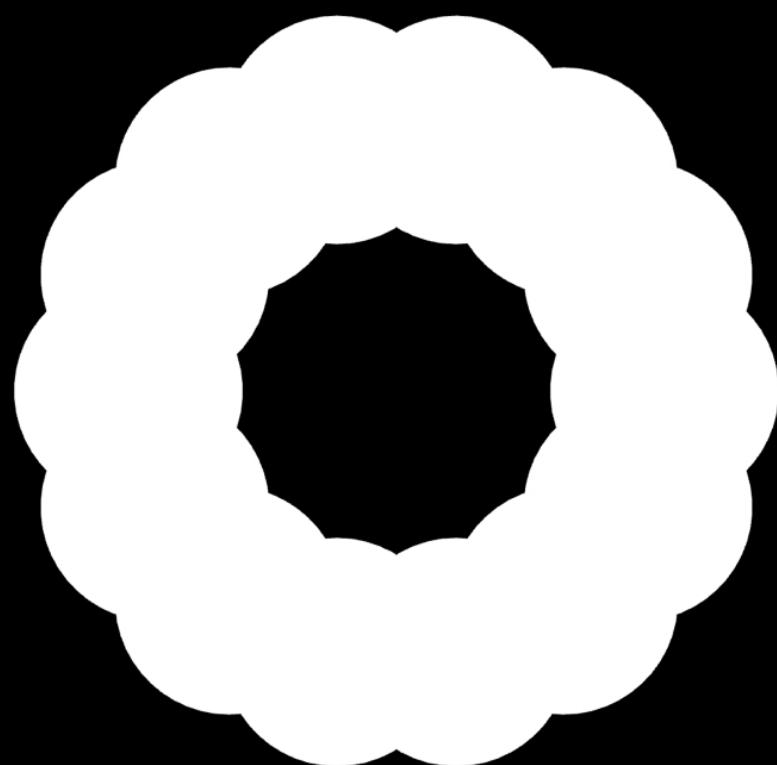
Get from 1

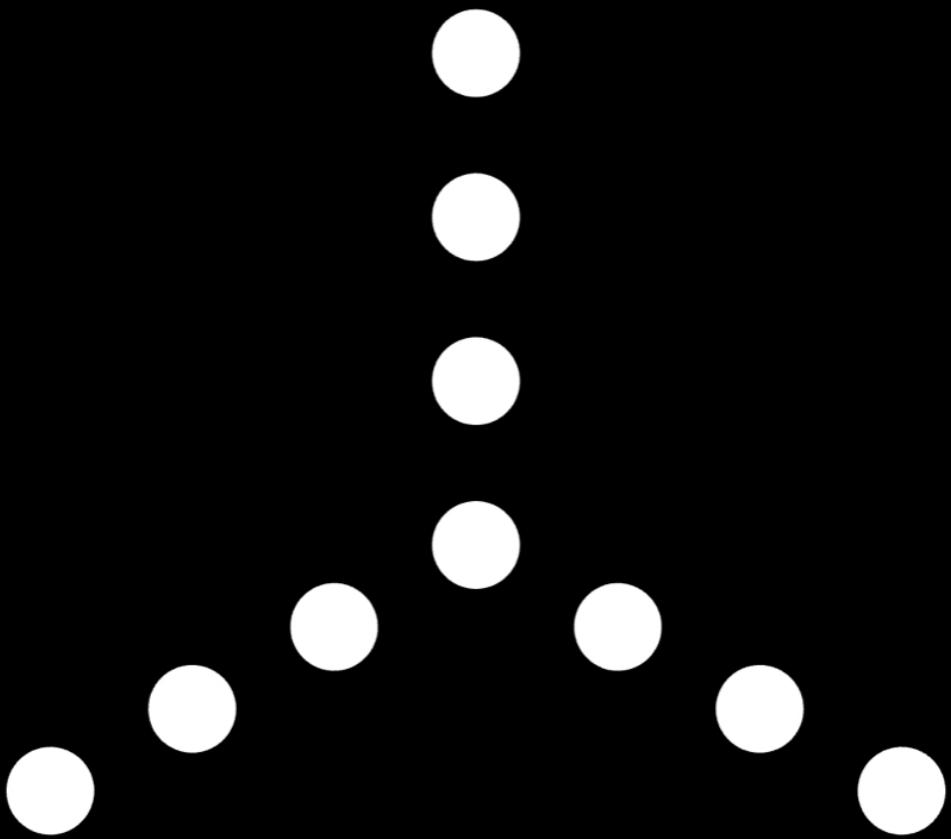
Get from 1 to 1000, in a matter of seconds.

You like it BIG

You like it BIG or small?

The only way to legitimately play with grids.



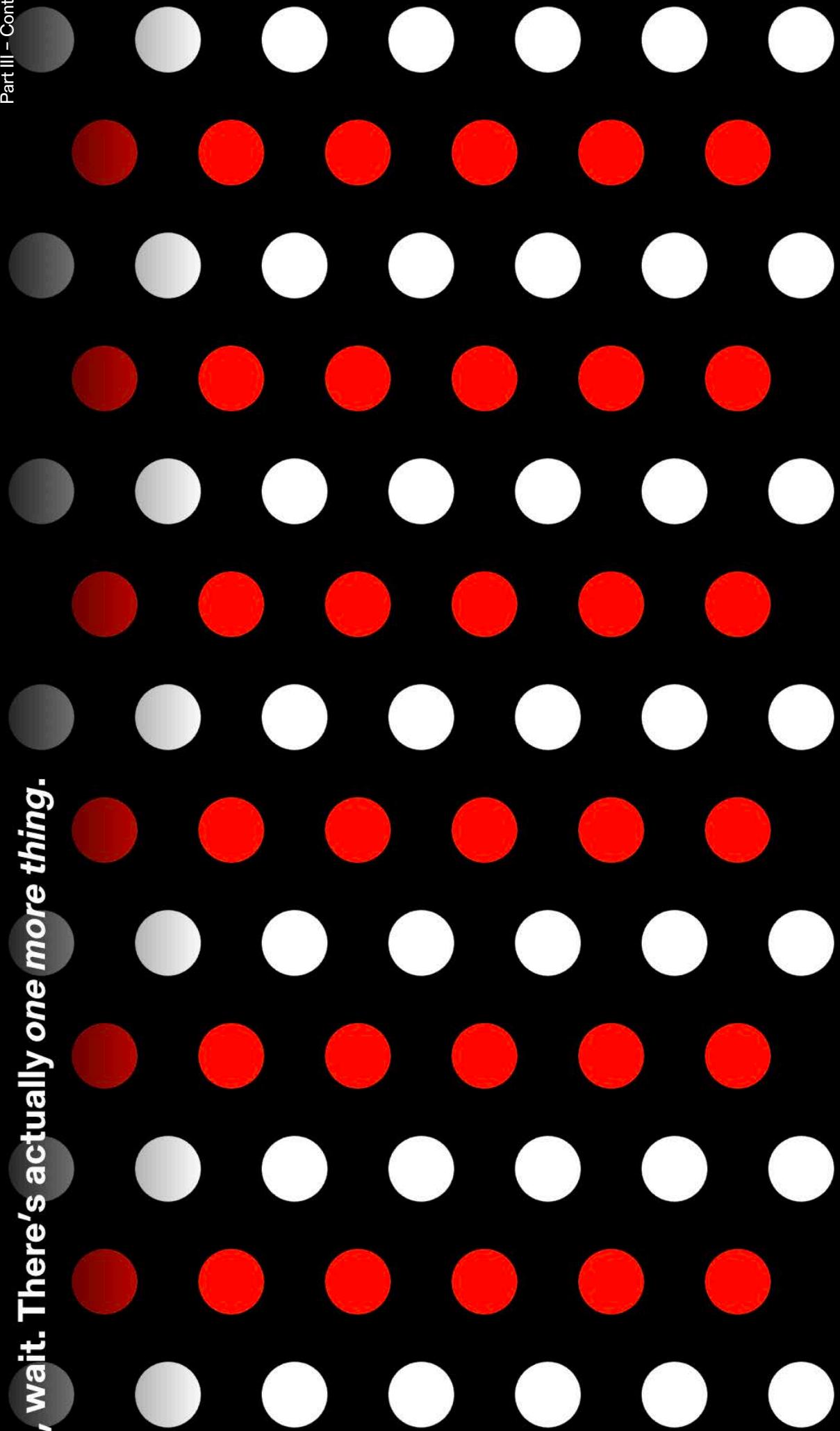


The only way to legitimately play with grids.

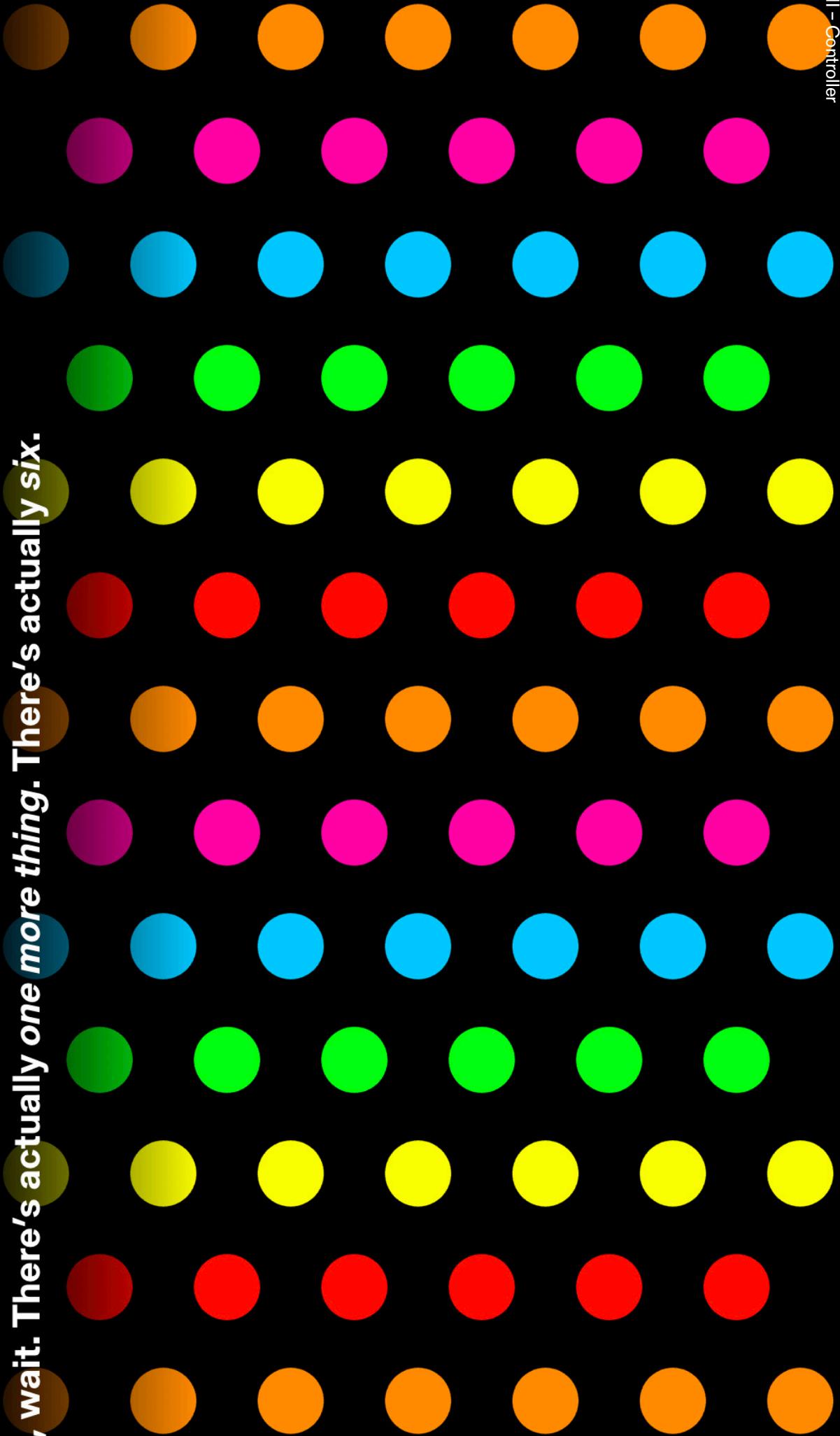
The only way to legitimately play with grids.

The only way to legitimately play with grids.

Oh, wait. There's actually one more thing.



Oh, wait. There's actually one more thing. There's actually six.



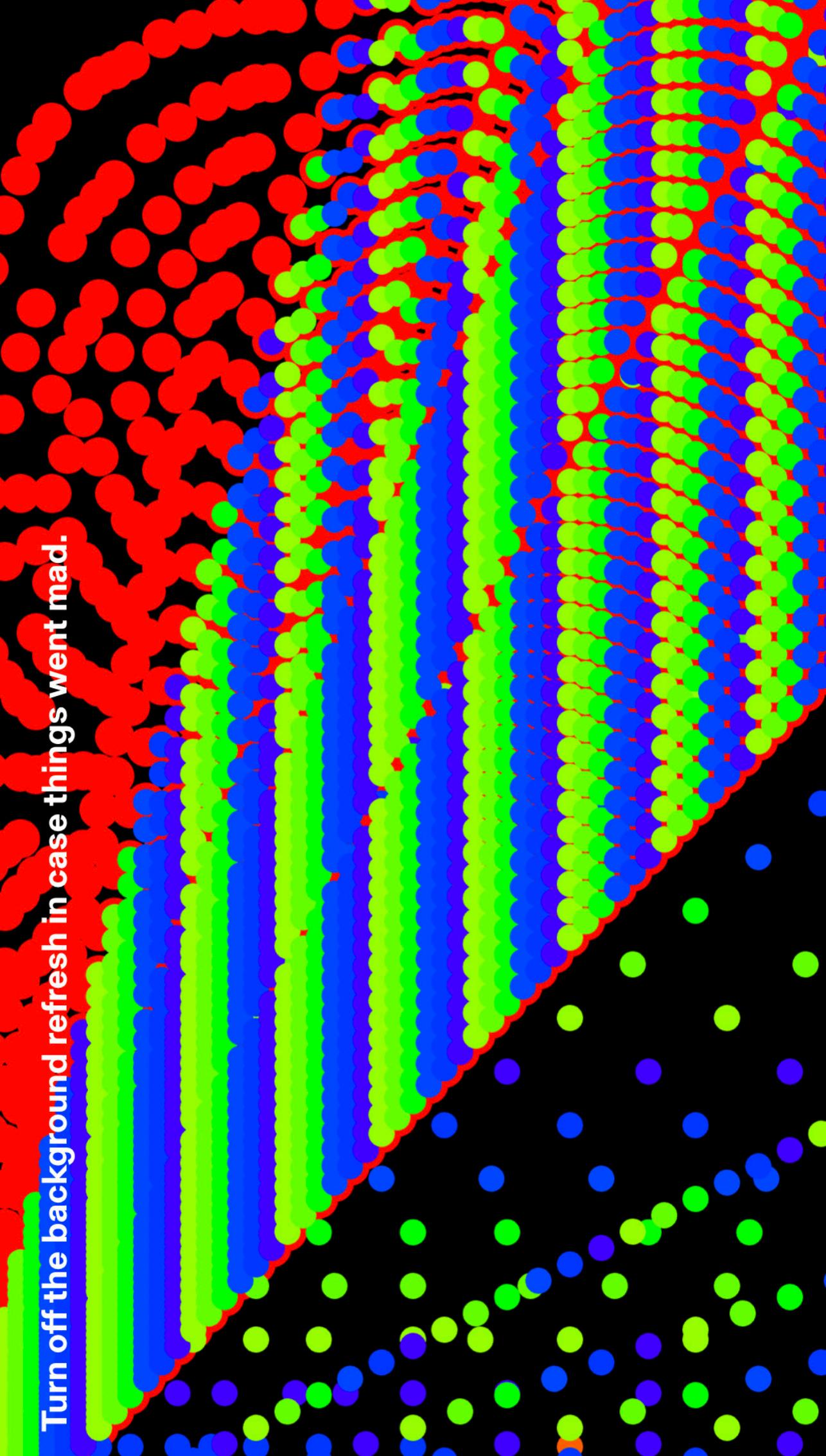
Hue, Saturation, and Prideness in all their glory.

Up to 6 channels with individual hues.

From a light show

From a light show to a laser one,

From a light show to a laser one, with a push of a button.



Turn off the background refresh in case things went mad.

Woah.

Ok, but how does it all look like in real life?

Well...

Well... something like this

this

this



this

All of *this* – just in one box.



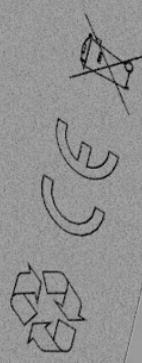
A very green one.



CE



A very green one.



- ✓ Only recycled plastic
- ✓ Only recycled aluminium
- ✓ Minimum power required
- ✓ Built to last as long as humanly possible

Looking good from every angle.



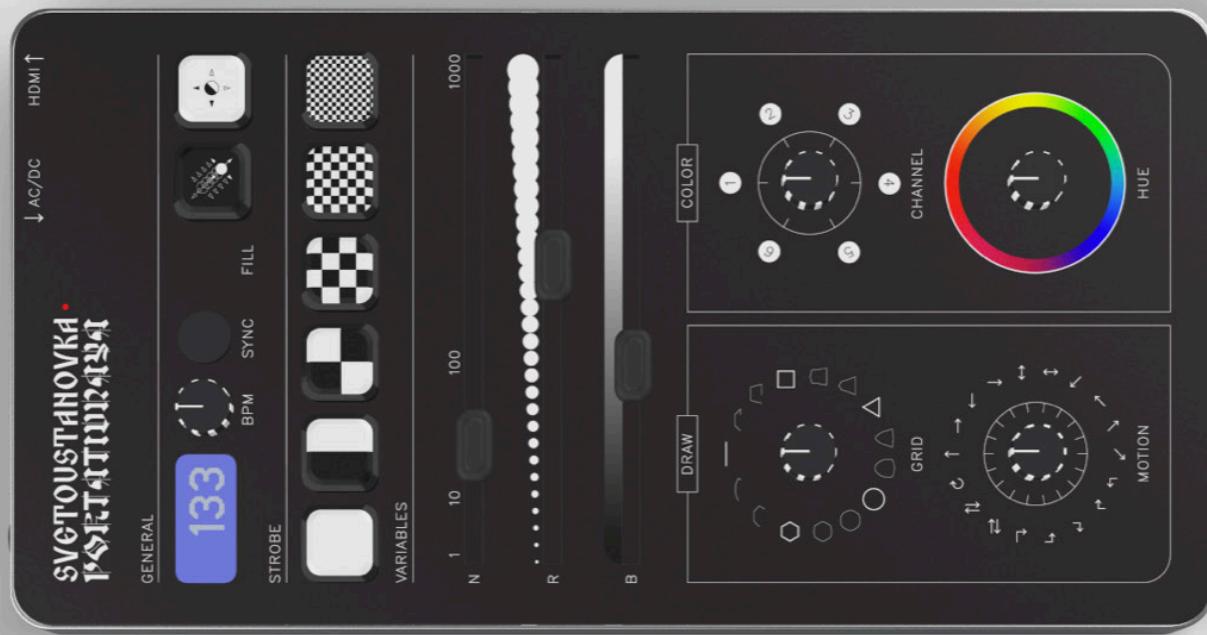




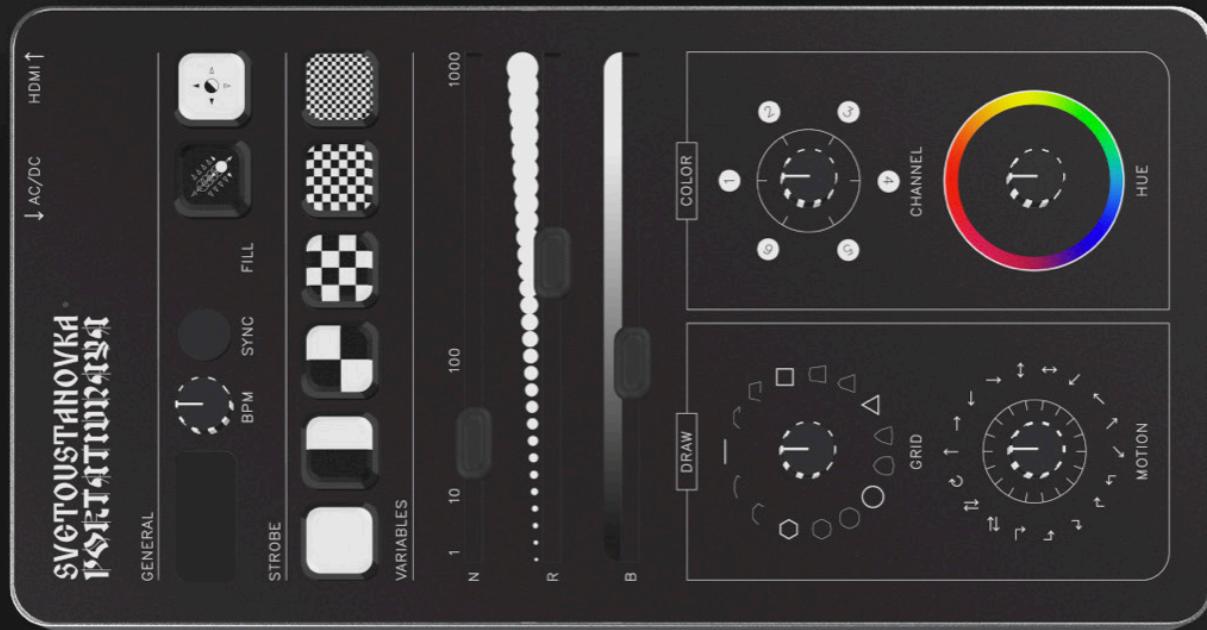




Coming whenever quarantine ends.



Designed by Martin Lezhenin.
Made in Moscow.



SVETOUŠTANOVKA
PRAZDNIK

• **Controlling**

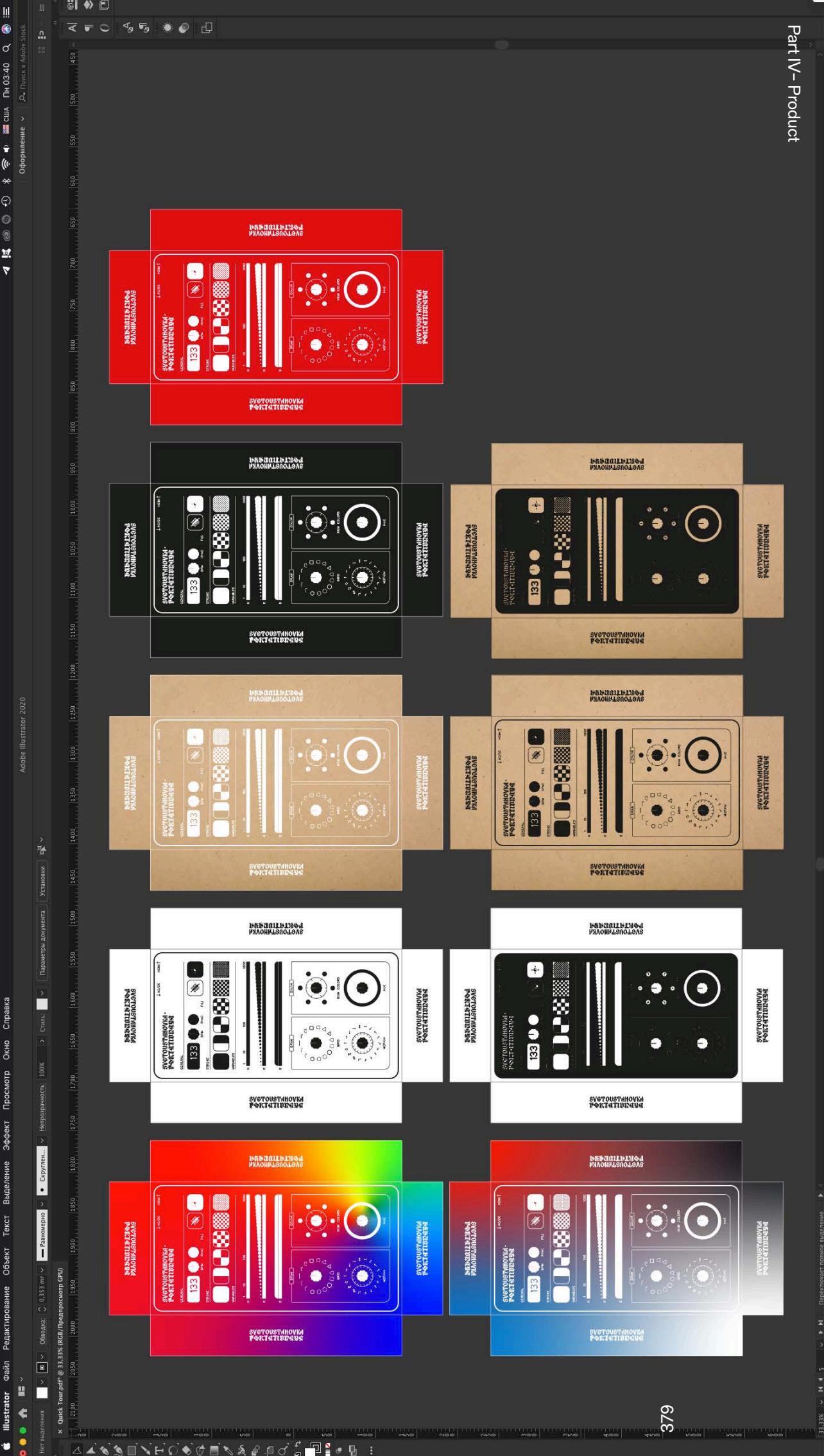
4.2 Branding

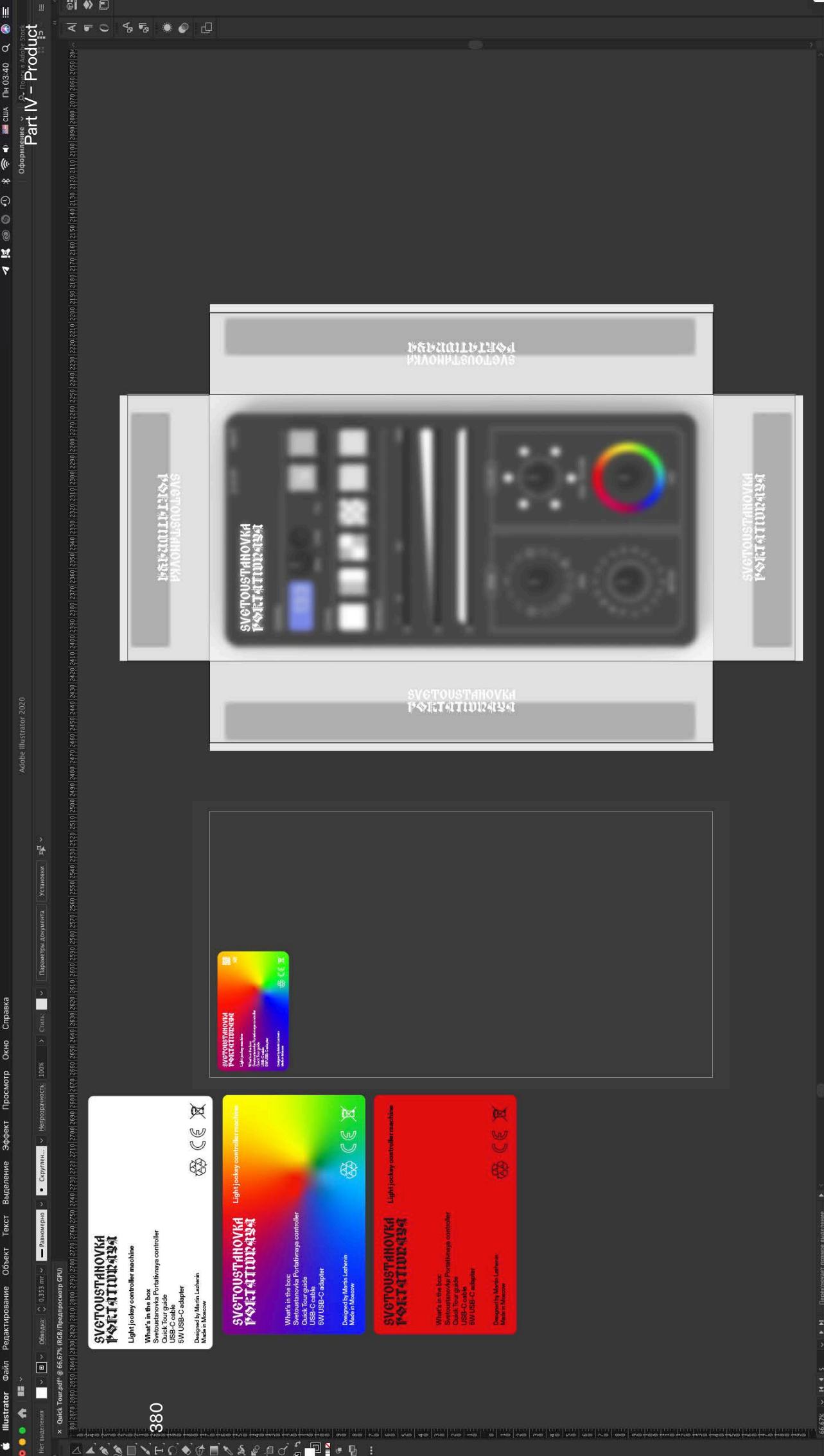
For branding the product I decided to go as far from my comfort zone as possible, choosing the naming which references the old soviet light music machines yet typeset in the latest font of the biggest Russian typefoundry Type.Tomorrow of Ilya Ruderman. I feel that this connection serves best as a bridge between the past and the future of the Russian engineers and designers, and this slight nod to both of them matches with the overall goal of the product.

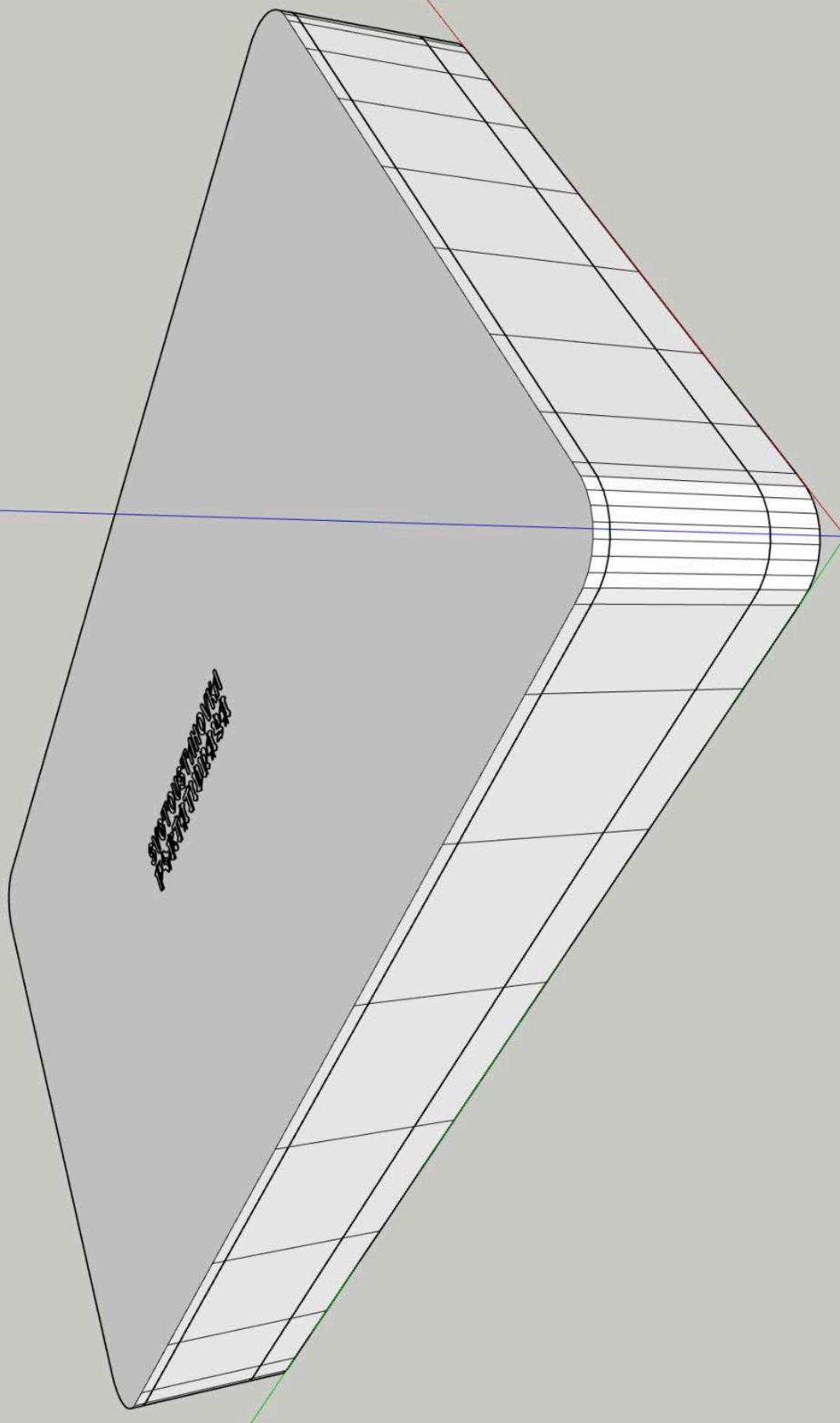


4.3 Packaging

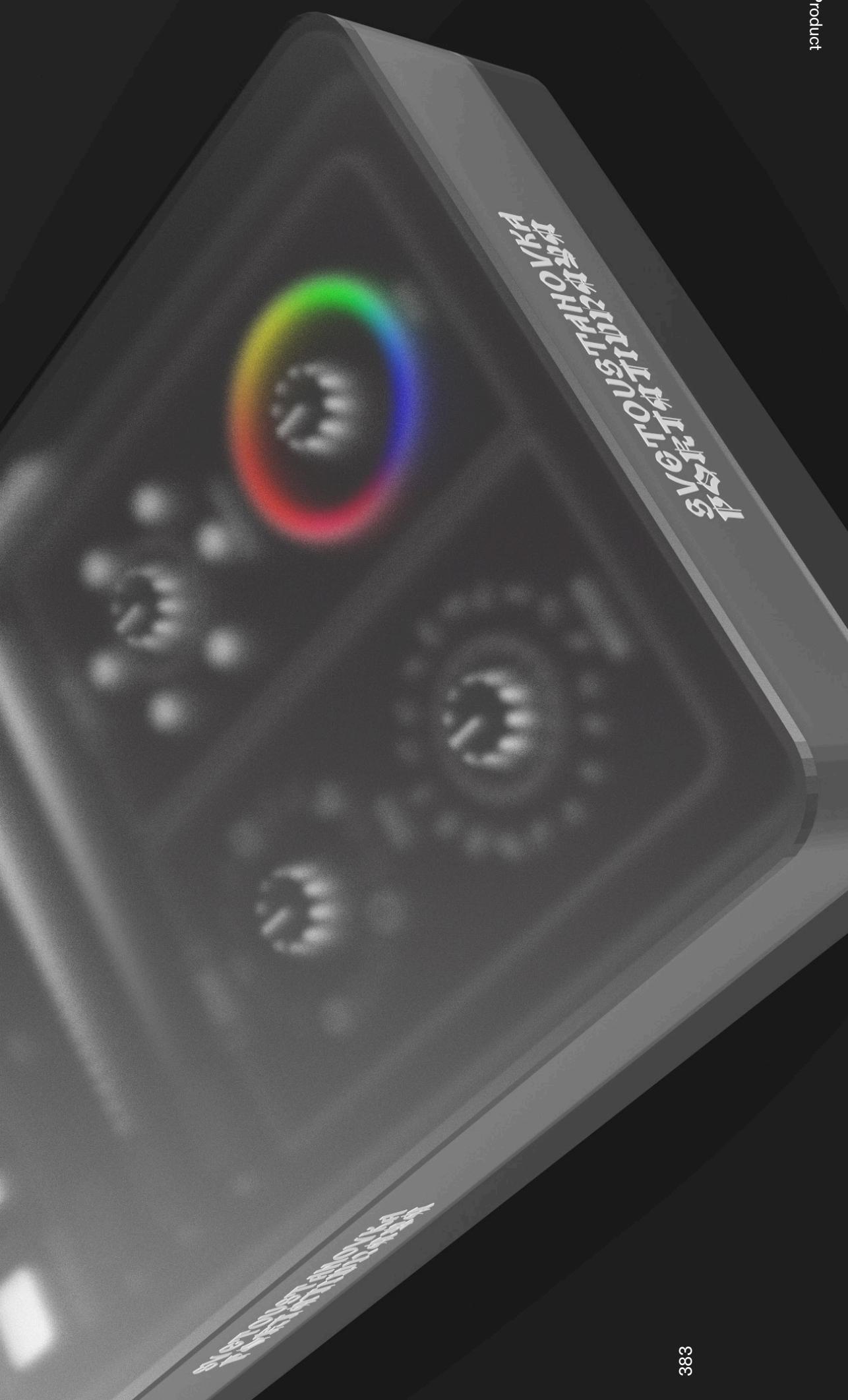
For packaging the product, I chose from many variations of the cardboard, plastic, and recycled material solutions, yet went for the semi-transparent plexiglass box afterall, adding more value to the overall look and defining the package to be more of a dust-safe durable case rather than a one time product wrapping.









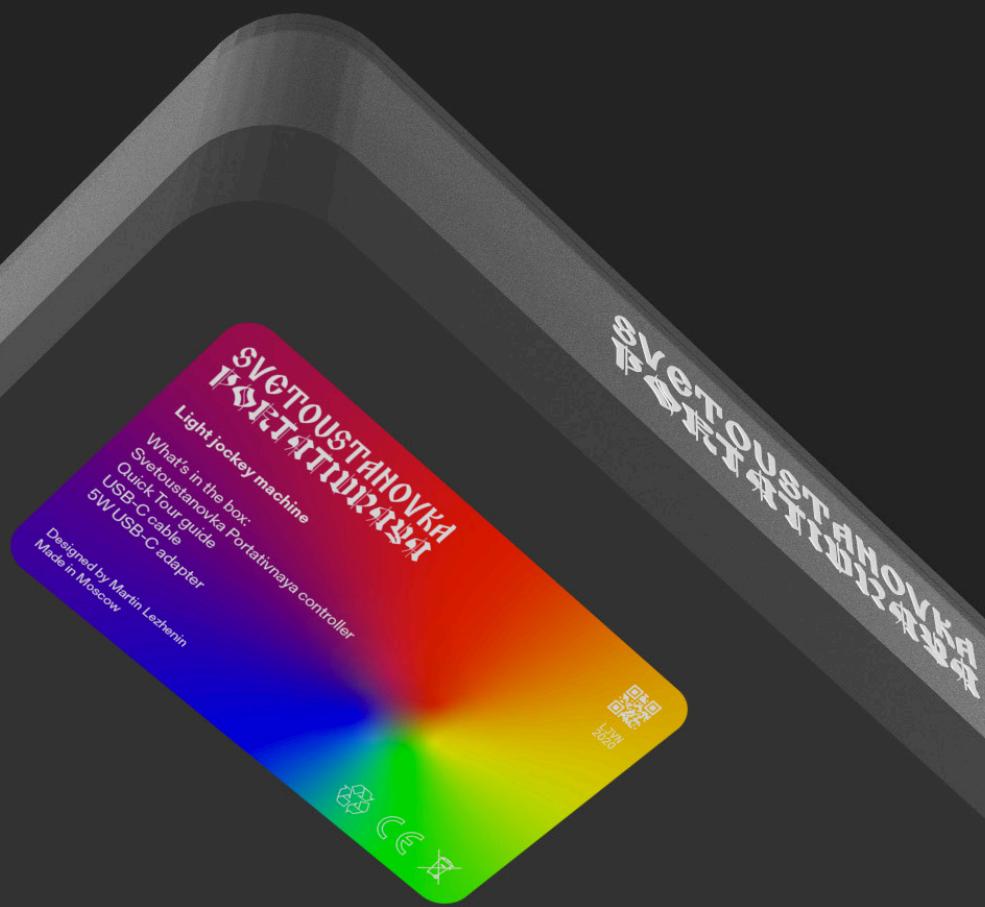












4.4 Quick tour

I also designed a small leaflet for welcoming the users and familiarising them with the interface and working principles of the controller. This leaflet is included into the package of the controller.



Let's get started.

Plugging it in

To power up your device, insert the USB Type-C cable plugged to the 220V or your computer to the **AC/DC** input. Connect to the projector via **HDMI**. Tap **I/O** to begin creating.

Setting the rhythm

When music starts playing, make sure you are on the same wave. Select the Beats Per Minute variable using the **BPM**, and use **SYNC** to adjust perfectly to the beat.

In case things get hot

To spice things up, push one of the **STROBE** buttons which make the screen blink from 2 to 36 times. To get things back to normal push the first one instead.

Just start exploring

Select the positions of the Number of Particles (**N**), their Radius (**R**), and Brightness (**B**). Change softly and casually along the performance.

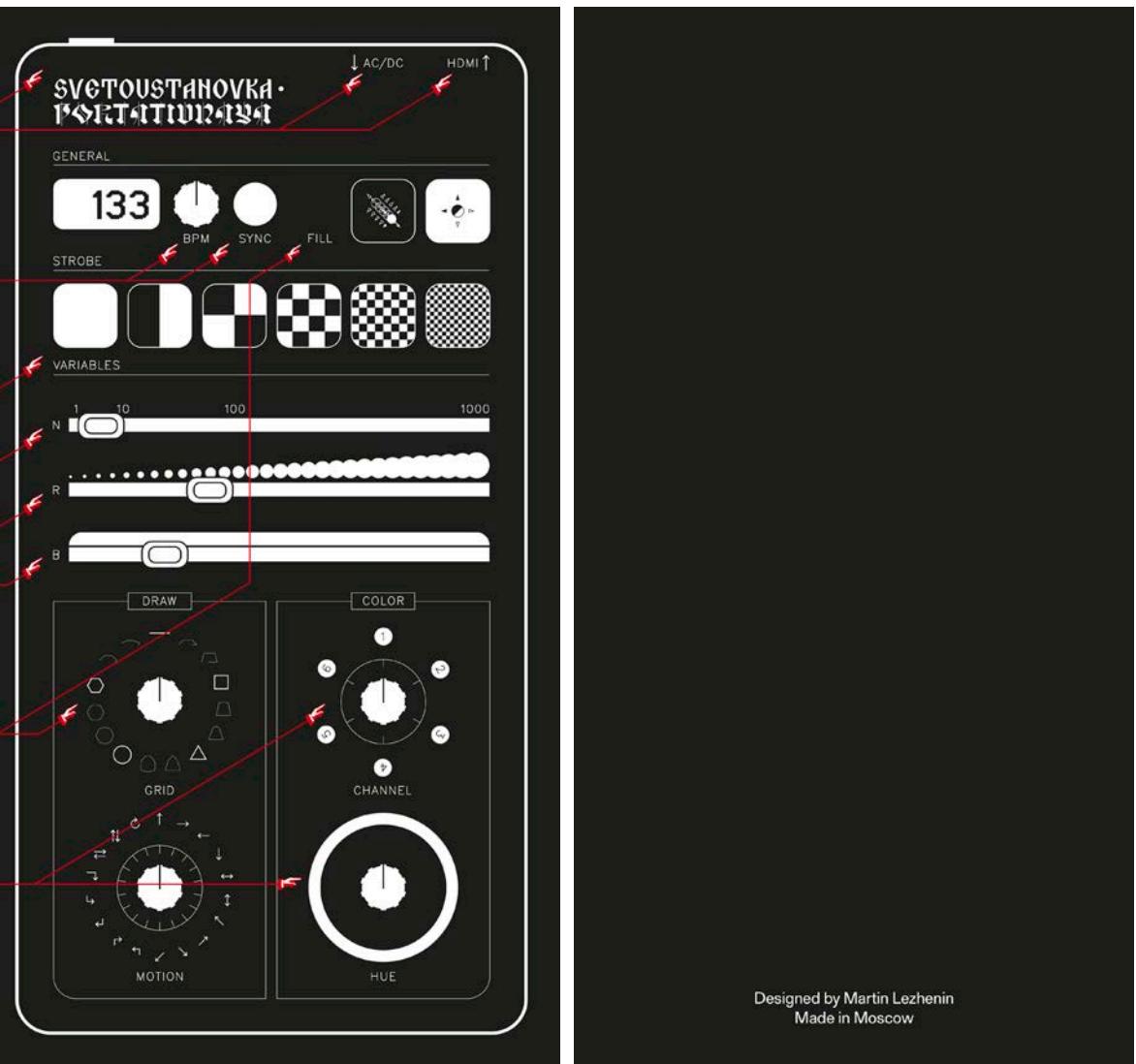
Make it complex

To change things drastically, experiment with different **GRIDs**, **MOTION** of the particles, and **FILL** buttons: the first for background refresh, the second for outline.

Welcome to the Woodstock

Finally, add some color to the game! Every **CHANNEL** is edited with **HUE** wheel. To make the color white, simply push the knob until the character click.

Stay playful.



4.5 Advertising

Lastly, I made a series of advertisements all playing on the concept of adding controller's variables and terms to the lyrics of commonly known songs.

Why'd you have to go
and make things so saturated.

СВЕТОУСТАНОВКА
ПОДСВЕТКА



СВЕТОУСТАНОВКА
ПОДИУМ

A hue like you
should wear a warning.



СВЕТОУСТАНОВКА
SYNTH TRIMMER



Can I SYNC it?
(Yes, you can.)

СВЕТОУСТАНОВКА
ПРИДАЧИ



Grid it away, grid it away,
grid it away now.



**With the lights up,
it's even less dangerous.**

**SVETOUSTANOVKA
ПОДСВЕТКА**

BHSAD BA Graphic Design & Illustration L6

Final project by Martin Lezhenin

Submitted to the University of Hertfordshire in partial
fulfilment of the requirements for the degree
of Bachelor of Arts (Honours)

Moscow 2020

