Chromapitch

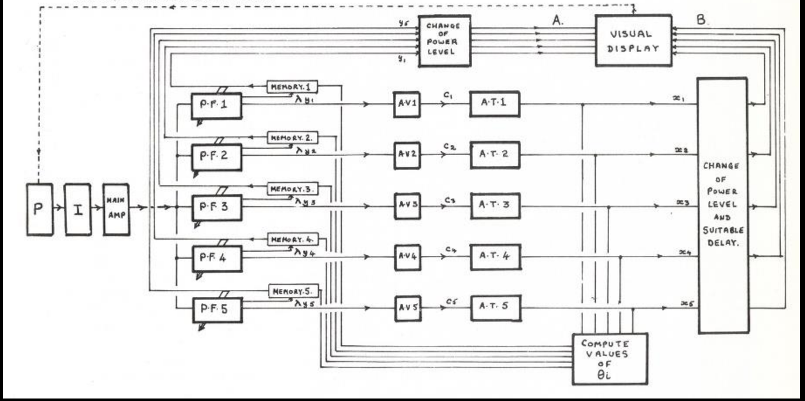
Justina Dziama - Matt Phan - Swapnil Patil

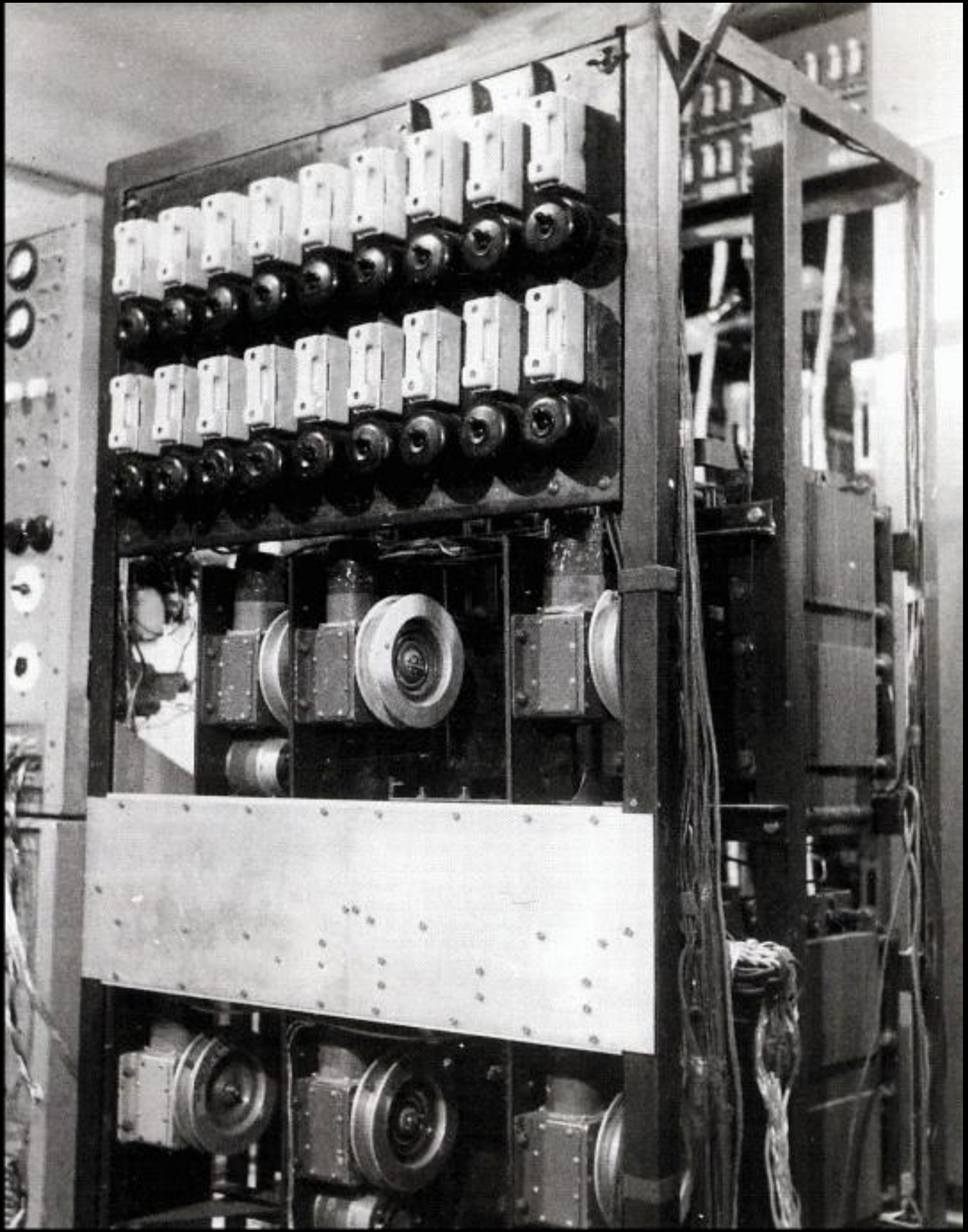
Precedent

MusiColor
Gordan Pask

- Machine suggested how, in the growing field of ubiquitous computing, humans, devices and their shared environments might coexist in a mutually constructive relationship.

Fig. 26 Outline of a typical Musicolour system. P = Performer, I = Instrument and microphone, A = inputs, y_i, to visual display that specify the symbol to be selected, B = inputs, x_i, to the visual display that determine the moment of selection. PF = property filter, AV = averager, AT = adaptive threshold device. Memories hold values of (y_i) and (y_i). Control instructions for adjusting the sequence of operation are not shown. Internal feedback loops in the adaptive threshold devices are not shown.





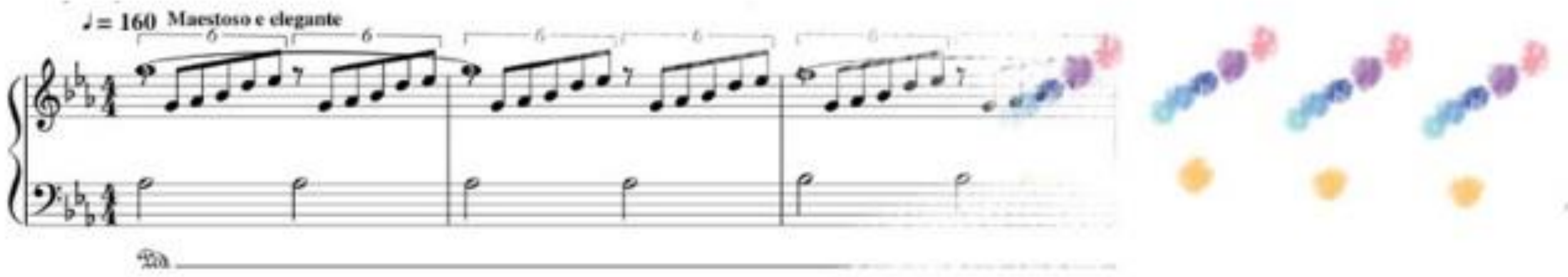
CONCEPT

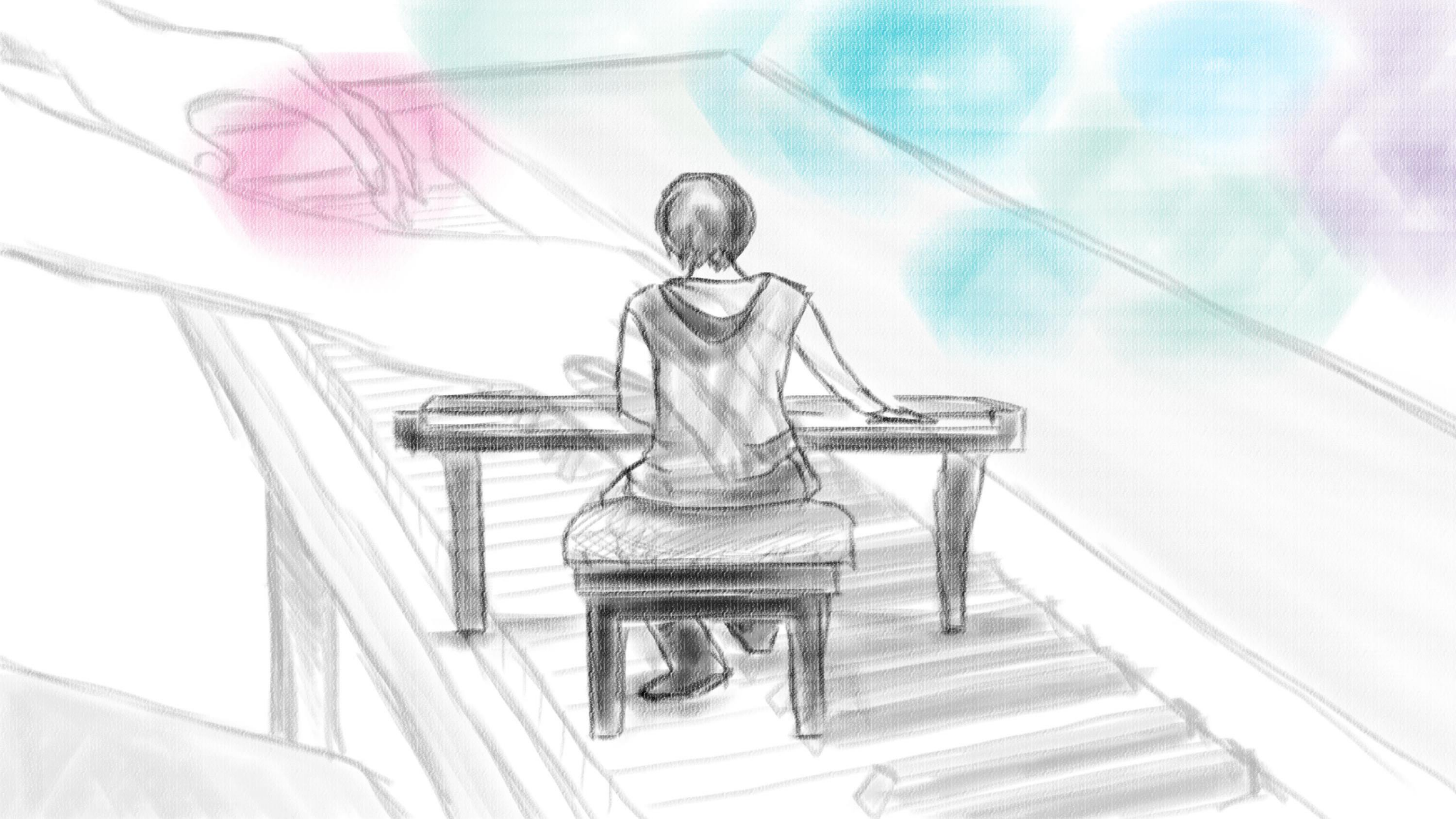
The Visualization of Music

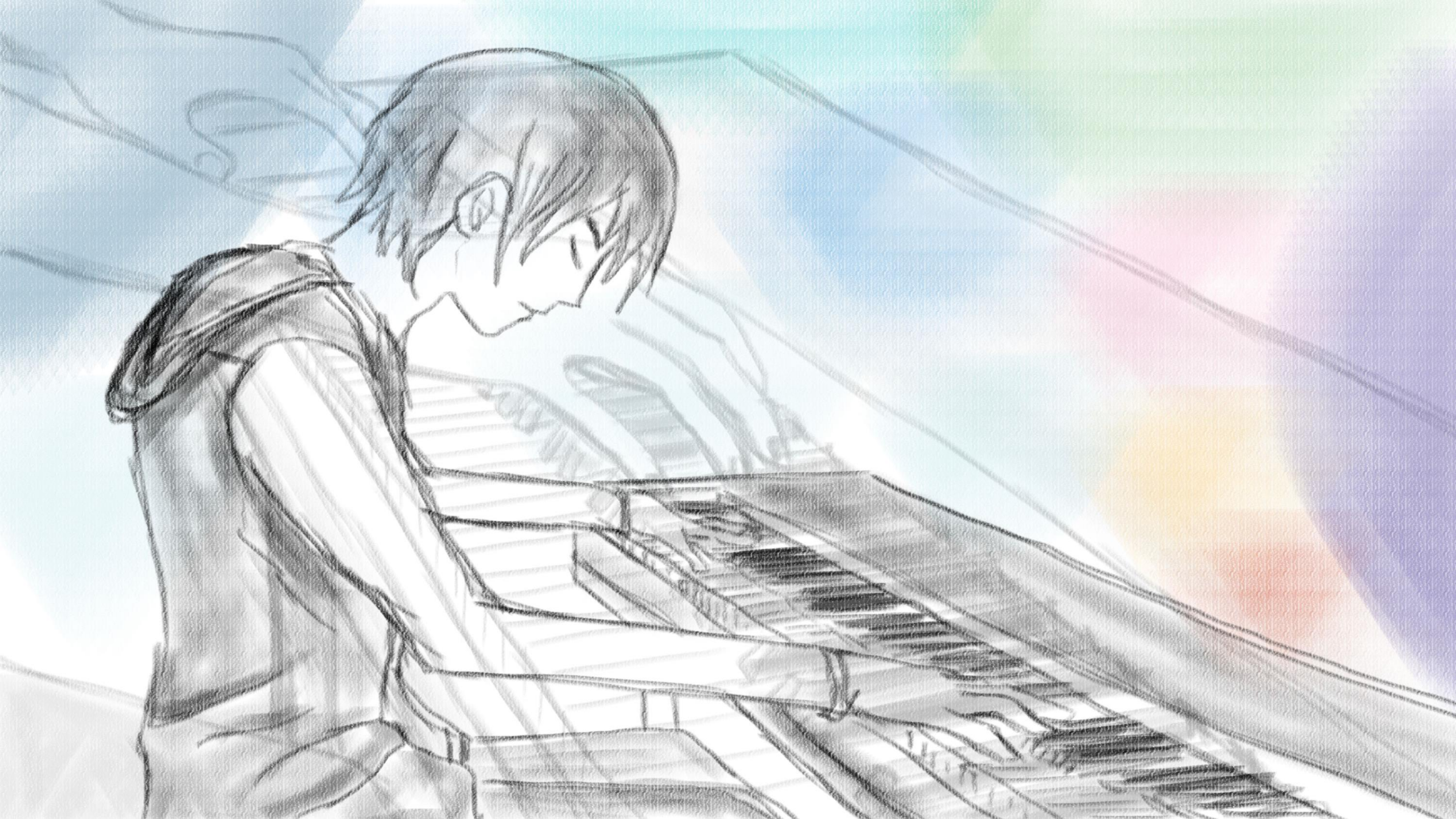
- Relationship of musical note to color or visual representation

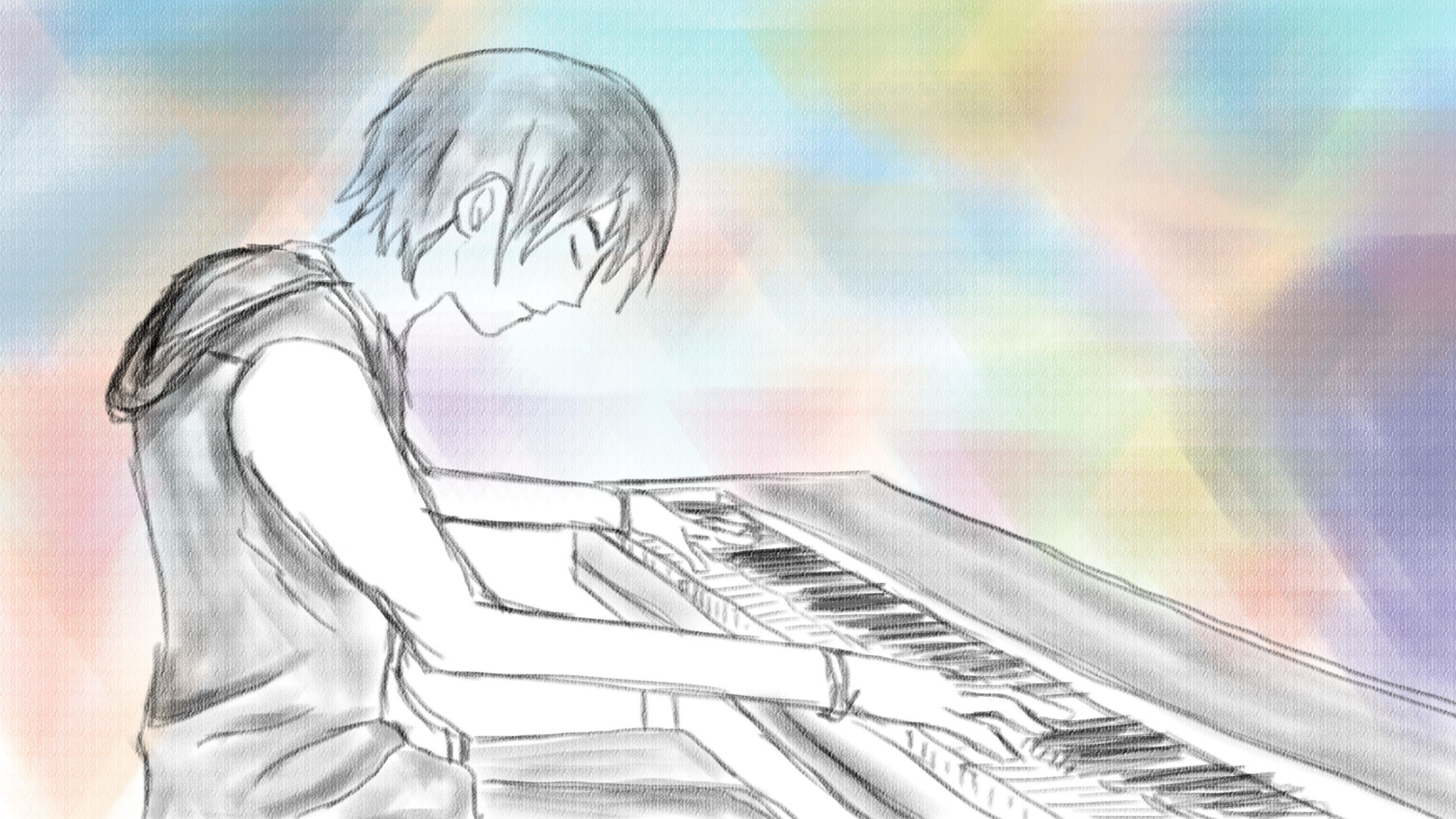
- Manner of exhibition OR educational method

- How can music redefine a space?

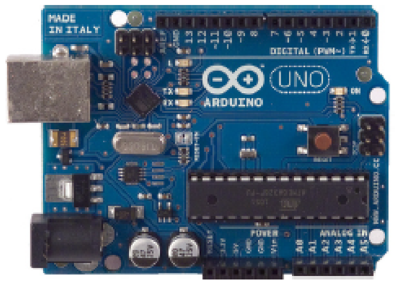








Original Iteration

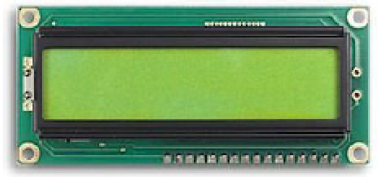


Arduino Uno

To link analog components to written code to transduce the music into specific colors combinations for the visual display.

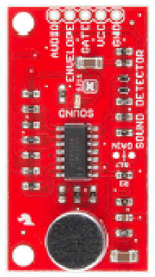
Liquid Crystal Display (LCD)

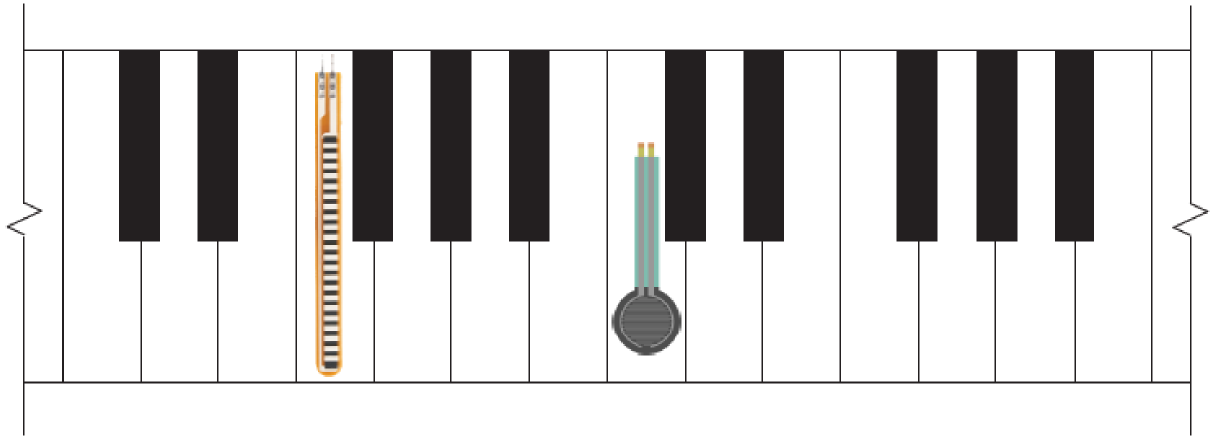
Digital display of note being played.

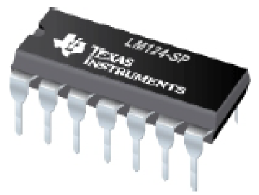


Quadruple Operational Amplifier

These devices consist of four independent high-gain frequency-compensated operational amplifiers that are designed specifically to operate from a single supply or split supply over a wide range of voltages.







Sound Detector

Picks up the output of the keyboard to be processed and displayed in the form of colored light.

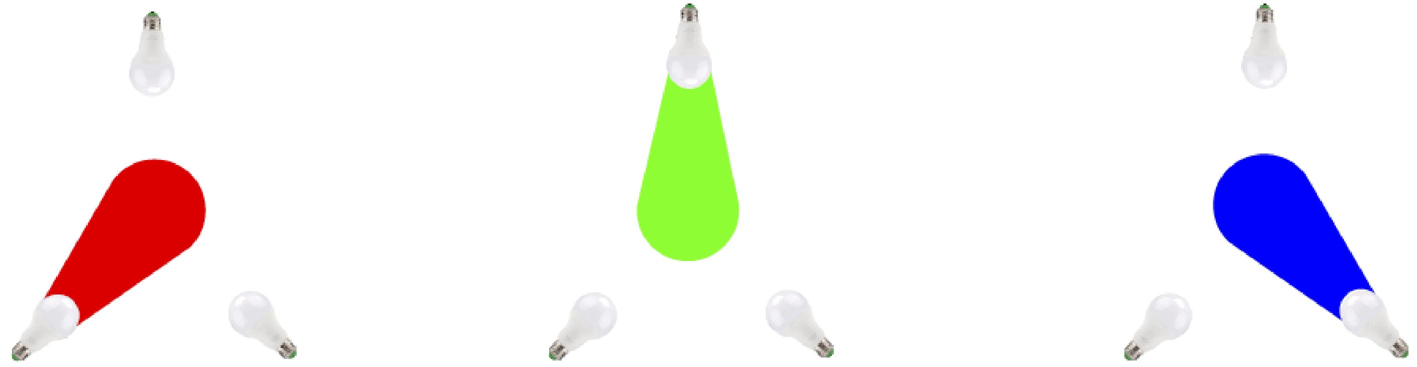
Flex Resistor/ Force Resistor

Potential components to be used in order to measure the physical interaction of a person playing keys.

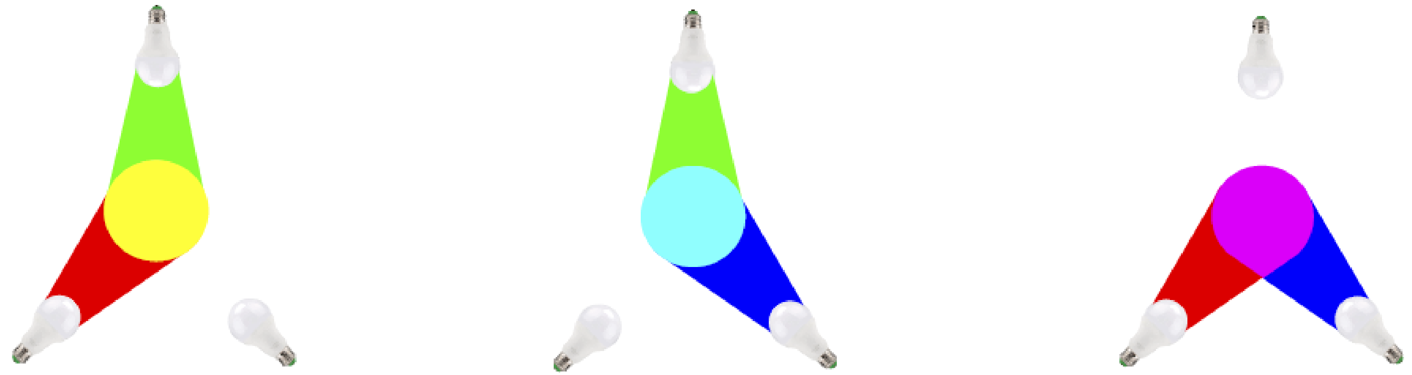
RGB LED Light Bulb

Allows for a gradation of color output depending on the pitch of the note being played.

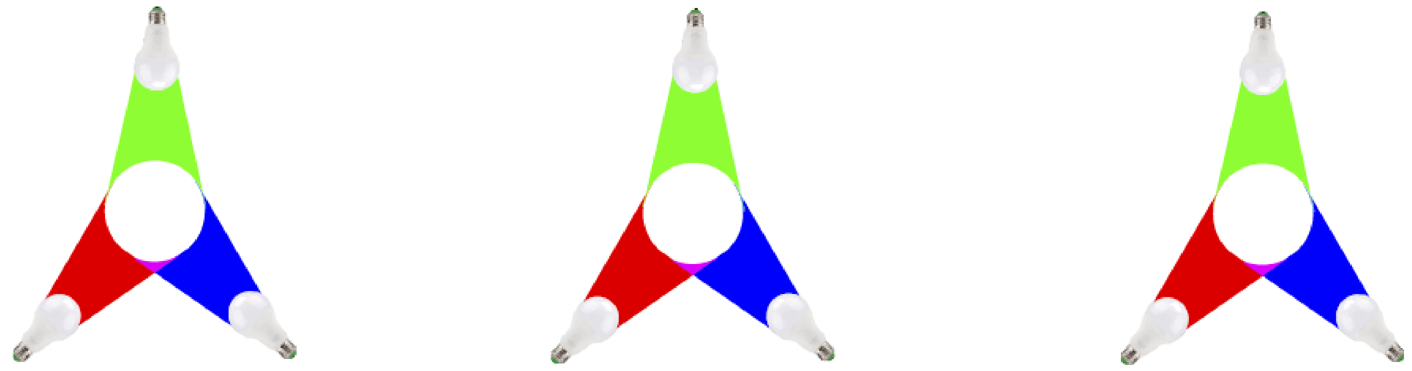
Illumination of Space



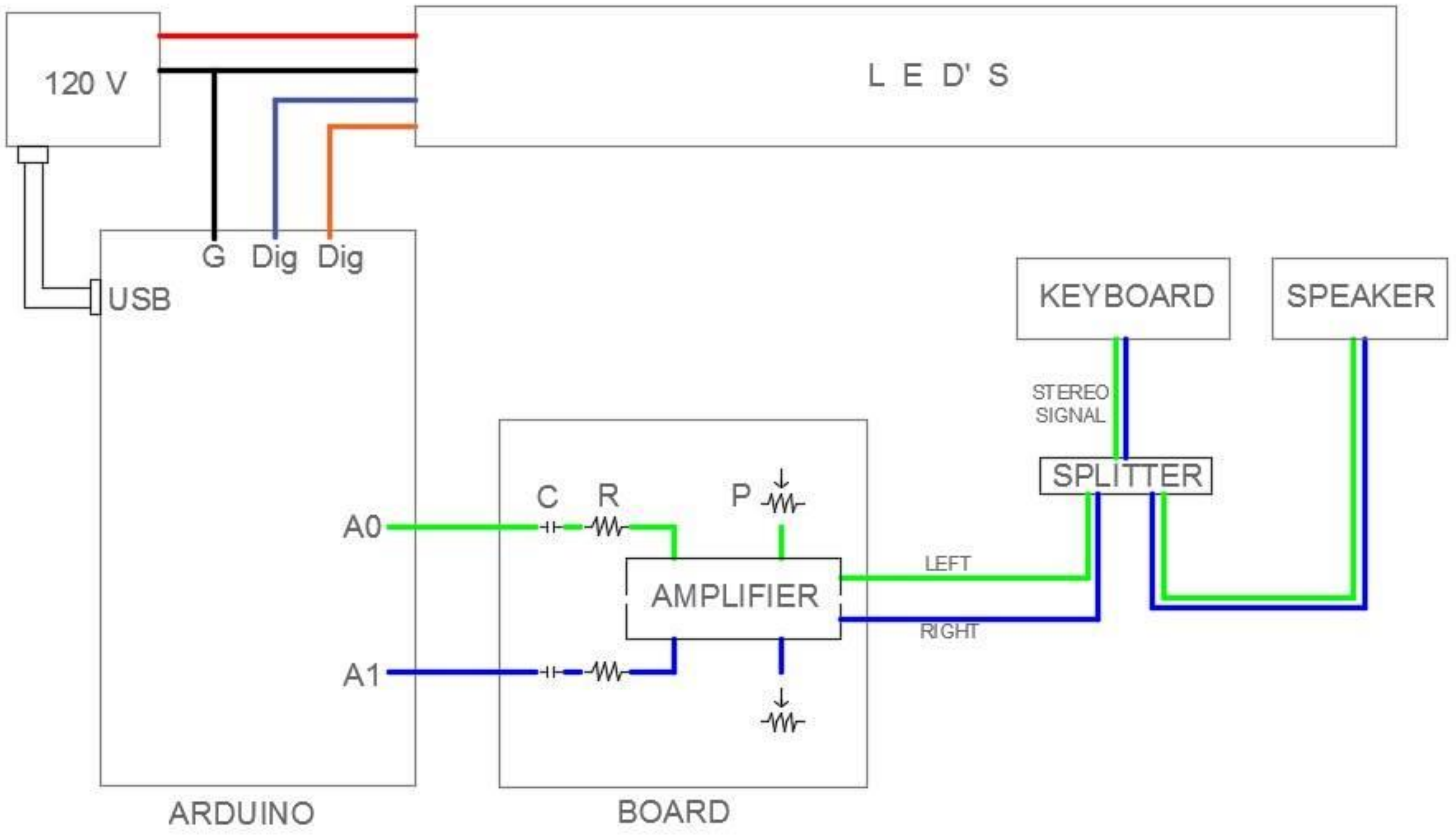
One note played at a particular pitch.



Two different notes played at once at particular pitches.



Three different notes played at once a particular pitches.


120 V
L E D' S
G
Dig
Dig
USB
KEYBOARD
SPEAKER
STEREO SIGNAL
SPLITTER
C
R
P
A0
A1
AMPLIFIER
LEFT
RIGHT
ARDUINO
BOARD

Final Iteration

Concept

The Visualization of Music

- Relationship of musical note to color or visual representation

- Rather than a Manner of exhibition OR educational method more of a focus on SPATIAL quality

- How can music redefine a space?

- Achieve via two systems interacting with people in space:
    - More direct correlation between keyboard connection to the computer
    - Disturbance from the microphone set up for the guitar

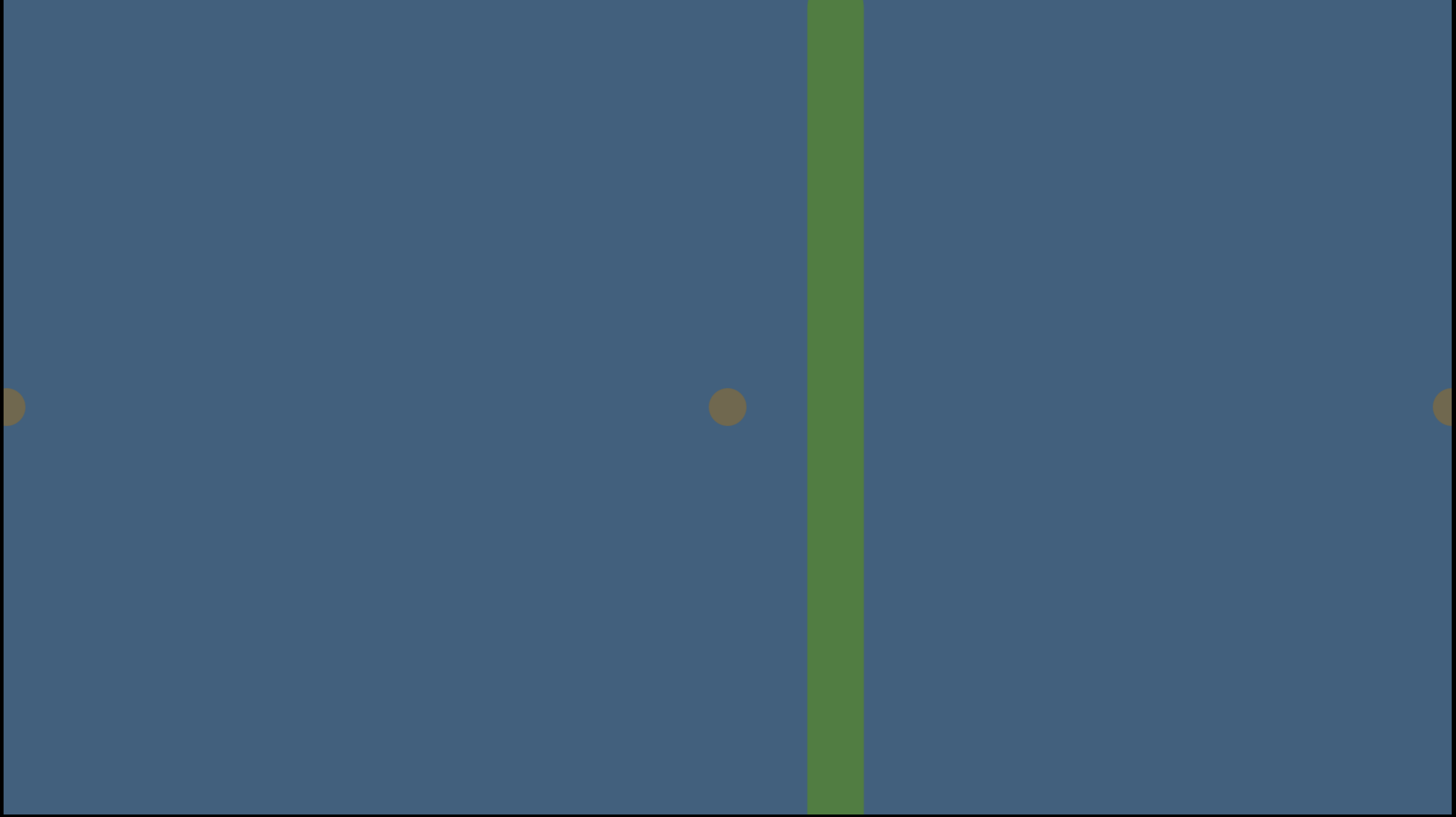
Screen projection from a computer screen using Processing

Changes color according to Input from a piano keyboard using plugins such as MidiBus and Minim

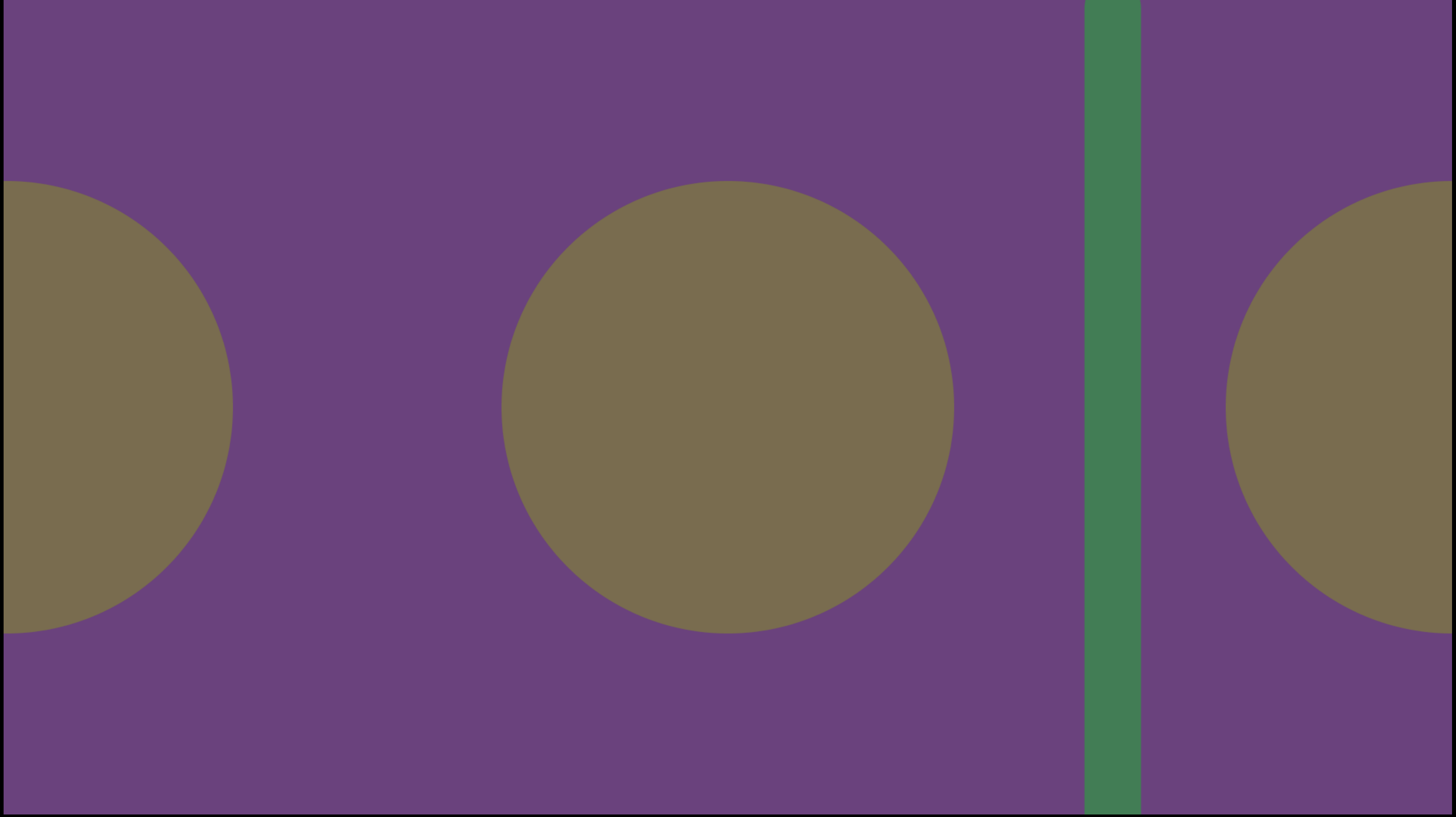
Color is managed by the numbered key with numbered hue color

```
import themidibus.*; //Import the library
import processing.sound.*;
AudioIn input;
Amplitude rms;
int scale;
MidiBus myBus; // The MidiBus
int globalPitch;

void setup() {
  colorMode(HSB);
  fullScreen(2);
  background(0);
  input = new AudioIn(this, 0);
  input.start();
  rms = new Amplitude(this);
  rms.input(input);
  input.amp(1.0);

  MidiBus.list(); // List all available Midi devices on STDOUT. This will show each device's index and name.


  myBus = new MidiBus(this, "CASIO USB-MIDI", "CASIO USB-MIDI"); // Create a new MidiBus with no input device and the default Java Sound
Synthesizer as the output device.
}

void draw() {
  background(globalPitch, 120, 125, 1000);
  fill(globalPitch/2, 120 , 125, 1000) ;
  rect(globalPitch*7.2, -25, 75, 1900, 1300);
  stroke(globalPitch/2, 120, 125) ;
  scale=int(map(rms.analyze(), 0, 0.5, 1, 350));
   noStroke();
   fill(globalPitch/5, 120 , 125, 200);
   ellipse(width/400, height/2, 50*scale, 50*scale);
  scale=int(map(rms.analyze(), 0, 0.5, 1, 350));
   noStroke();
   fill(globalPitch/5, 120 , 125, 200);
   ellipse(width/1, height/2, 50*scale, 50*scale);
  scale=int(map(rms.analyze(), 0, 0.5, 1, 350));
   noStroke();
   fill(globalPitch/5, 120 , 125, 200);
   ellipse(width/2, height/2, 50*scale, 50*scale);
```

```
void noteOn(int channel, int pitch, int velocity) {
  globalPitch = int(map(pitch, 36, 96, 0, 255))     ;
  // Receive a noteOn
  println();
  println("Note On:");
  println("--------");
  println("Channel:"+channel);
  println("Pitch:"+pitch);
  println("Velocity:"+velocity);
}

void noteOff(int channel, int pitch, int velocity) {
  // Receive a noteOff
  println();
  println("Note Off:");
  println("--------");
  println("Channel:"+channel);
  println("Pitch:"+pitch);
  println("Velocity:"+velocity);
}

void controllerChange(int channel, int number, int value) {
  // Receive a controllerChange
  println();
  println("Controller Change:");
  println("--------");
  println("Channel:"+channel);
  println("Number:"+number);
  println("Value:"+value);
}

void delay(int time) {
  int current = millis();
  while (millis () < current+time) Thread.yield();
}
```



A colored bar in accordance with the background travels physically left and right along the screen with the keys played. Color is hue number divided in half.



3 Circles represent the feedback that the microphone picks up