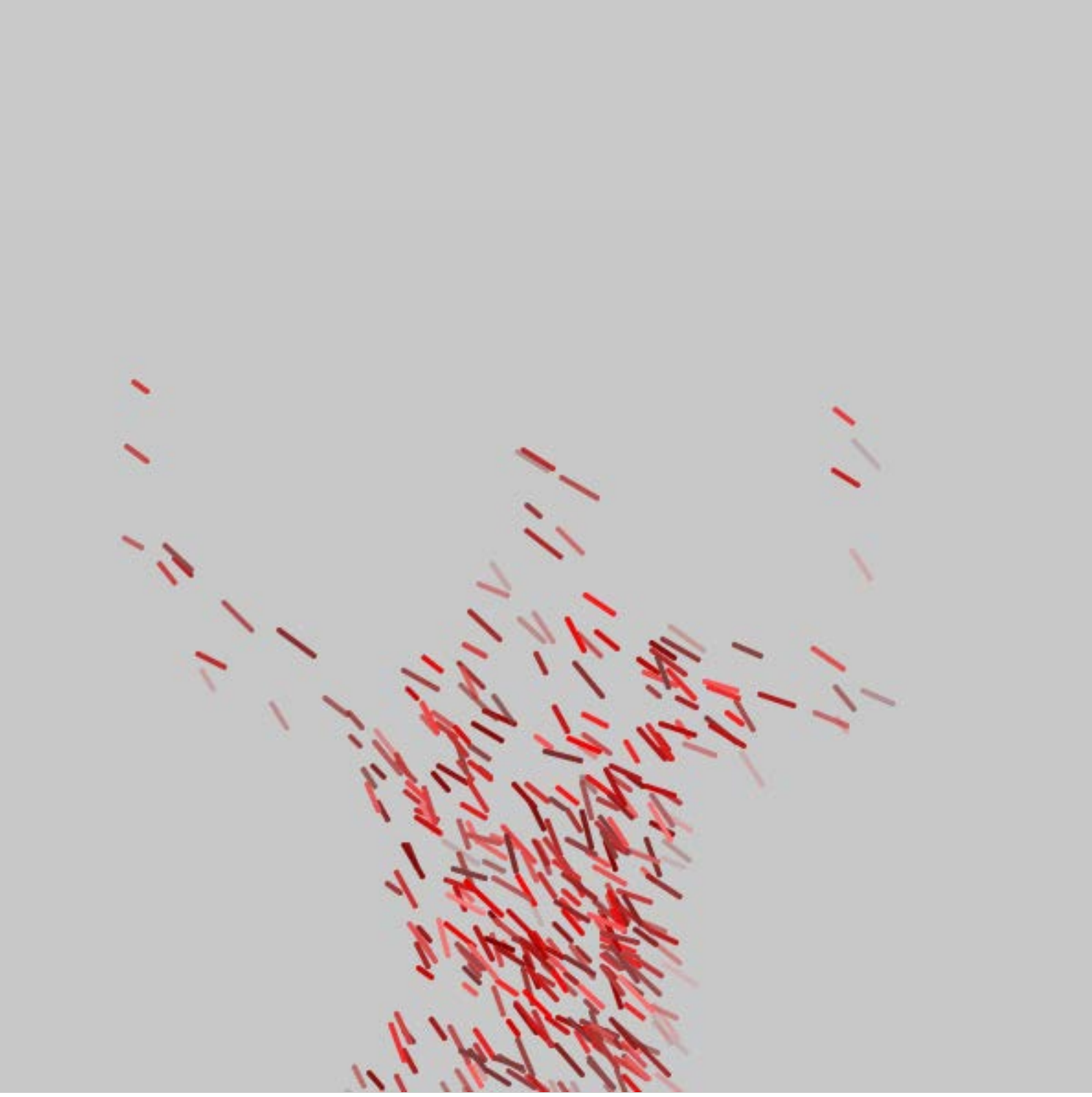


New Media

Onderzoeksdocument
Kinect Art

Ontworpen door
Hein Pauwelyn (2NMCT7) en Jonas Vanhecke (2NMCT9)

HOWEST | ACADEMIEJAAR 2015-2016

1. Doel

De bedoeling van ons project genaamd Kinect Art is om generative art te tekenen met behulp van de Kinect en de programmeertaal Processing 2.

We kwamen op dit idee terecht omdat we graag wilden werken met generative art en het ons leuk leek om zelf art te ontwikkelen met behulp van de Kinect.

Kinect Art zal personen voorstellen door middel van lijnen. Je kan ze regelen in kleur en het resultaat kan je opslaan als een afbeelding.

2. Stappenplan

Stap 1
We hebben ons eerst gefocust wat we wouden doen met Generative Art. Na wat Google'n vonden we leuke voorbeelden ontworpen met de Kinect. Daarom hebben we de Kinect gekozen omdat er veel mogelijkheden zijn en het leek ons een haalbaar project.

Stap 2
Startbestand aangemaakt om de Kinect te activeren en op GitHub geplaatst zodat we gemakkelijk konden delen met elkaar. De link vindt u hier: *https://github.com/HeinPauwelyn/KinectArtNewMedia*

Stap 3
We hebben nagedacht hoe we de streepjes zouden laten genereren en bewegen. De oplossing die we gevonden hebben is om object georiënteerd te werken. We hebben de classes "Line" en "LineRepo" gemaakt waarbij ze respectievelijk lijnen aanmaken met een richting die we berekenen door middel van richtingscoëfficiënt en een opslagplaats voor de lijnen.

Stap 4

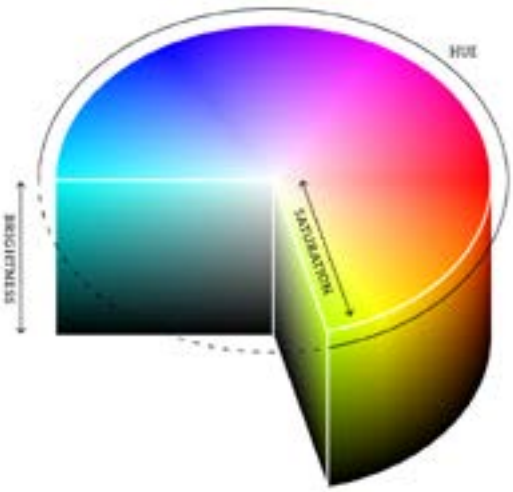
Daarna hebben we nagedacht hoe we een persoon gingen opvullen met de lijnen. Dankzij het internet hebben we code gevonden waarop we ons gebasseerd hebben.

```
for (int y=0; y<context.depthHeight(); y++) {
    for (int x=0; x<context.depthWidth(); x++) {
      int index = x + y * context.depthWidth();

      if (userMap != null && userMap[index] > 0) {
        int colorIndex = userMap[index] % userColors.length;
        userCurID = userMap[index];
        blob_array[index] = 255;

        text(sampletext[int(random(0,10))],x,y);
      }
      else {
        blob_array[index]=0;
      }
    }
  }
```

https://forum.processing.org/two/discussion/2271/getting-text-in-user-s-silhouette

Stap 5

Na het creëren van de lijnen in een persoon, hebben we nagedacht over hoe we de kleurovergang konden maken. Eerst gingen we inkleuren met RGB waarden die we random gekozen hebben. Deze gingen we dan om de zoveel seconden van kleur veranderden, maar we kwamen in problemen terecht waardoor we een ander systeem gekozen hebben.



De docent gaf ons een tip om HSB waarden te gebruiken. Dit is veel simpeler en oogt mooier. Door de hue telkens te doen veranderen krijgen we random kleuren die een vlotte overgang maakt tussen de verschillende kleuren.

Stap 6

Daarna hebben we de gebruiker laten kiezen tussen manuele of automatische kleurovergangen door middel van de shift toets. Er kan nu ook een schermafbeelding genomen worden door de enter of return toets.

Stap 7

Uiteindelijk hebben we de code gerefactort met als doel de leesbaarheid te verbeteren.

3. Code

Klasse Project

In de Project klasse wordt de Kinect definieert en wordt iedere frame hertekend. De lijnen bewegen, persoon opvullen en tekenen gebeurt in een andere klasse. De hue wordt hier echter wel bijgehouden en aangepast indien gewenst.

In deze klasse kan je ook instellen als de lijnen automatisch moeten verkleuren of niet. Je kan ook een schermafbeelding nemen die opgeslaan wordt als png.

Klasse Line

De properties van deze klasse zijn: 'x1' en 'x2', 'y1' en 'y2' die de coördinaten van de punten gaat bepalen, 'xDirection' en 'yDirection' de richting bepalen, 'bright', 'sat' en 'opacity' die de kleuren bepalen. Tenslotte heb je 'removed' voor bij te houden als de lijn buiten de persoon is of niet. Hier vindt je enkele methods die logische code uitvoeren.

- In de constructor wordt de method 'generate' aangeroepen. Deze gaat de x en y coördinaten bepalen. Indien een genereer punt binnen de gebruiker ligt gaan we eerst de brightness en saturation random gaan bepalen, daarna gaan we de richting berekenen door middel van de rico.

```
public void generate(SimpleOpenNI context){

    generatePoint();

    if (checkIfPointInUser(context, x1, y1)){

        generateColor();

        xDirection = (x2 - x1) / 4;
        yDirection = (y2 - y1) / 4;
    }
    else {
        removed = true;
    }
}

protected void generatePoint() {
    x1 = int(random(0, 640));
    x2 = x1 + int(random(5, 20));

    y1 = int(random(0, 480));
    y2 = y1 + int(random(5, 20));
}

protected void generateColor() {
    sat = int(random(127,255));
    bright = int(random(127,255));
}
```

```
protected Boolean checkIfPointInUser(SimpleOpenNI context, int x, int y) {

    try {
        int[] userMap = context.userMap();

        int index = x + y * context.depthWidth();
        if(userMap.length > index){
            return userMap != null && userMap[index] > 0;
        }
        return false;
    }
    catch(Exception ex) { return false; }
}
```

- Met de method 'move' laat de lijnen bewegen volgens de vooraf ingestelde x en y direction. Indien er 1 van de punten buiten de persoon is, begint de fade out die de opacity doet verminderen. Indien de opacity 0 is, wordt de lijn verwijderdt uit de repository.

Klasse LineRepo

Deze klasse is louter voor het bijhouden van alle lijnen in een arraylist. Voor het verwijderen van de lijnen is er de method 'cleanup' gemaakt. Deze method is gemaakt voor het verwijderen van de lijnen waar de variable 'removed' op false is.

4. Bronnen en Links

Waarop we onze werk baseren:
http://www.creativeapplications.net/processing/kinect-physics-tutorial-for-processing/

Code die we vonden hoe we personen best moesten opvullen:
https://forum.processing.org/two/discussion/2271/getting-text-in-user-s-silhouette

Onze GitHub project:
https://github.com/HeinPauwelyn/KinectArtNewMedia