"Lab 4 - Generative Art Using Loops" by Zoë Wood is licensed under CC BY-NC-SA. Accessed from www.engage-csedu.org.

Fall 2015	Computational Art	Zoë Wood

1 Lab 4 - Generative Art

Goals

The goals for this lab are:

1. Practice using a `loop` control structure to *generate* patterns/scenes
2. Practice using functions to re-draw parts of a scene
3. Practice using `random` to produce desirable colors and design layout
4. Practic using structured layout
5. Consider the role of repetition in art

Modality

Individual

Overview

Generative art: "refers to art that in whole or in part has been created with the use of an autonomous system. An autonomous system in this context is generally one that is non-human and can independently determine features of an artwork that would otherwise require decisions made directly by the artist. In some cases the human creator may claim that the generative system represents their own artistic idea, and in others that the system takes on the role of the creator." [1]

We will be comparing the use of a more structured design (with yourself as the artist in control of exactly where elements are placed) and a more random generative algorithm that uses stratified sampling but mostly just draws elements randomly in a scene.

[1] Wikipedia

Details

Task: You must create two different images using Processing each of which are *generated* by an algorithm when your program is run. Each of the two different scenes must use repetition (that is, some visual elements that are repeated, but that may be slightly different). One sketch should be more organic looking and the other must be more structured (intentionally laid out/designed). Your project must:

- include two different sketch elements (one for each sketch) that is encapsulated in two different functions (one which is organic looking and one which is more structured). These elements should not be exact copies, for example the color of each item can be different, the scale and placement (rotation) may also vary. Be sure to use code (function parameters) to control these aspects of this part of your sketch(es).
- repeated copies of these sketch elements - one which is random (organic) and one which is structured (for example, consider an urban city skyline, a fake 'microchip', farm or forest).
- each of your scenes must include at least 10 copies of the repeated element
- each element must be a compound shape (for the structured shape it must be composed of at least 2 shapes that when combined form a recognized whole (i.e. building or tree) for the organic shape, it must be composed of at least 3 shape elements or a shape with at least 10 vertices - this shape also must be recognizably interesting). At least two elements of each shape must vary when drawn (ie color, scale, rotation, etc.)
- be at least 400 x 400
- be in color
- use `random` appropriately

To complete this lab, you must:

- Note that you are allowed to create this lab with noLoop() enabled in setup() to not create distracting variance by using random for every frame.
- first design an organic looking design element to be included repeatedly in your sketch
- create a version of your scene with the organic design element repeated in random positions - you may need to implement some kind of 'stratified sampling' (for example divide your screen into four quadrants all which contain some number of randomly placed samples).

- create a sketch of a more structured generative scene (such as a city landscape) (again using a `loop` control structure) with your more structured design element (like a building) - this scene must look coherent (ie make sense).
- Your sketch must fulfill all tasks listed above

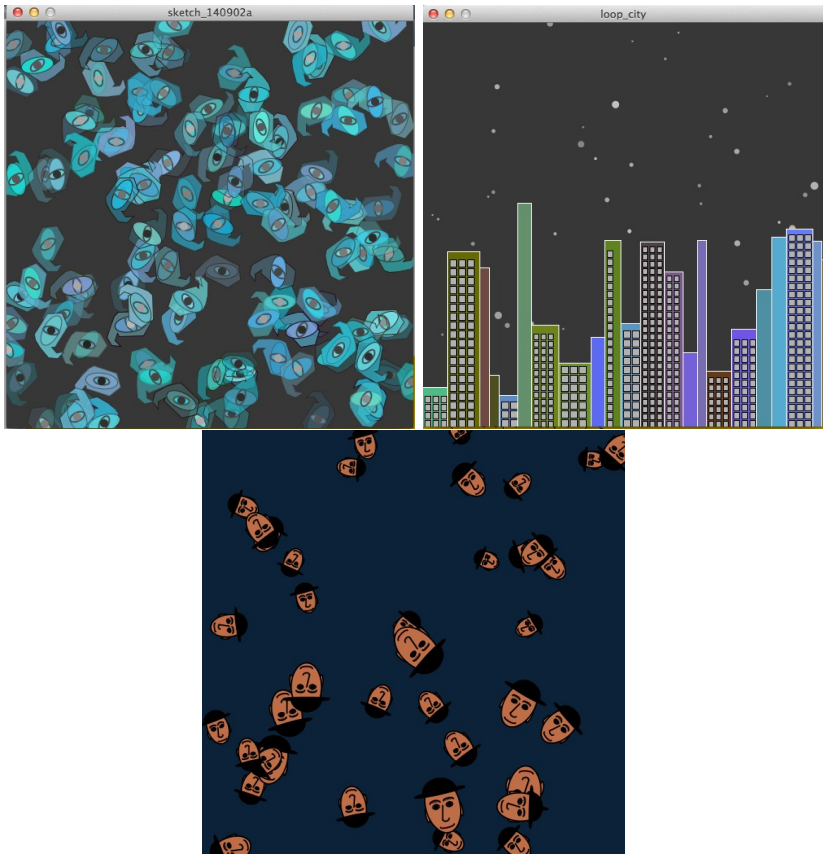

Figure 1: Output from Processing sketches generated using a loop control structure. Top left is a more 'organic' shaped paisley like design that is laid out randomly (with 4 stratified sample grids). Top left is a more structured generated drawing of a city – the stars, building and windows are generated when the sketch is run all using `loop` control structures; Bottom is the output from the example code to draw many faces

Demo:

In order to receive credit for this lab, you must demo your sketch to your instructor or TA. For every lab, your score will be broken down 75% for meeting the technical requirements and 25% for aesthetics. **Submitting your sketch:** You must post an image of your sketch to your pinterest Computational Art board. Please also pin your reference art. You must also handin you pde file on polylearn.

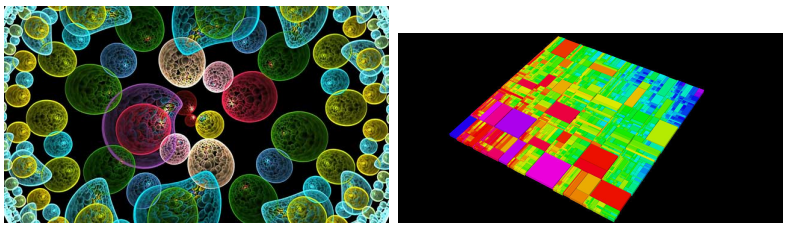

Figure 2: Some (non-Processing) examples of generative art. The first one (a more organic image) is from: http://www.pbs.org/arts/gallery/off-book-episode-10-generative-art/off-book-episode-10-generative-art/ and the second (a more structured image) is from: http://www.subblue.com/gallery/album/34

Resources:

http://video.pbs.org/video/2170070010/

```
/*ZJ Wood - example CPE 123 code to draw many faces */
void drawFace(float fCx, float fCy, float fW, float rot) {
  //ratio face - sept. 2015 - ZJ Wood
  float faceH, faceW;
  float eyeH, eyeW, noseH;

  pushMatrix();
  translate(fCx, fCy);
  rotate(radians(rot));
  translate(-fCx, -fCy);
  faceW = 0.7*fW;
  faceH = 1.6*faceW;
  noseH = faceH/4;
  eyeW = .2*faceW;
  eyeH = eyeW/1.6;

  strokeWeight(2);
  //fill(180, 137, 138);
  fill(192, 110, 71);
  //ears
  ellipse(fCx-faceW/2, fCy+noseH/2, eyeW*.5, noseH);
  ellipse(fCx+faceW/2, fCy+noseH/2, eyeW*.5, noseH);
  //head
  ellipse(fCx, fCy, faceW, faceH);
  //eyes
  fill(255);
  ellipse(fCx-eyeW, fCy, eyeW, eyeH);
  ellipse(fCx+eyeW, fCy, eyeW, eyeH);
  //eyebrows
  fill(83, 49, 32);
```

```
  arc(fCx-eyeW, fCy-eyeH, eyeW*1.4, eyeH, PI, TWO_PI);
  arc(fCx+eyeW, fCy- eyeH, eyeW*1.4, eyeH, PI, TWO_PI);
  //iris
  fill(122, 76, 23);
  ellipse(fCx-eyeW, fCy, eyeH, eyeH);
  ellipse(fCx+eyeW, fCy, eyeH, eyeH);
  //pupils
  fill(23);
  ellipse(fCx-eyeW, fCy, eyeH/2, eyeH/2);
  ellipse(fCx+eyeW, fCy, eyeH/2, eyeH/2);
  //nose
  fill(174, 99, 69);
  arc(fCx, fCy+noseH, eyeW, eyeH, 0, PI);
  line(fCx, fCy, fCx-eyeW/2, fCy+noseH);
  //lips
  fill(201, 90, 75);
  arc(fCx, fCy+1.5*noseH, 2*eyeW, eyeH, 0, PI);
  //hat
  fill(0);
  ellipse(fCx, fCy-noseH, faceW*1.5, eyeH);
  arc(fCx, fCy-noseH, faceW, faceH/2, PI, TWO_PI);
  popMatrix();
}

void setup() {
  size(600, 600);
  noLoop();
}

void draw() {
  background(12, 34, 56);
  //draw faces mild stratefied sampling - upper left corner
  for (int i=0; i < 10; i ++) {
    drawFace(random(0, width/2), random(0, height/2), random(30, 50), random(0, 360));
  }
  //upper right corner
  for (int i=0; i < 10; i ++) {
    drawFace(random(width/2, width), random(0, height/2), random(30, 50), random(0, 360));
  }
  //lower left corner - slightly bigger
  for (int i=0; i < 10; i ++) {
    drawFace(random(0, width/2), random(height/2, height), random(40, 70), random(0, 360));
  }
  //upper right corner
  for (int i=0; i < 10; i ++) {
    drawFace(random(width/2, width), random(height/2, height), random(40, 70), random(0, 360));
  }
}
```