CMSC427 fall 2017 Lab 0

Fan Yang

I started from Circle.pde and changed almost everything. The points are replaced by a sequence of near-ellipses whose inclination angles are tuned to be in line with their major axis. To achieve this goal, I translate the origin of the canvas to the center of each near-ellipse and rotate the coordinate system with proper angles before the near-ellipse is drawn. After finishing drawing of each near-ellipse, origin of the canvas and angle of the system are brought back to their default values. The near-ellipses are drawn in the same way as how polygons are. Coordinates of the vertices of the near-ellipses are generated from combining the parametric equations of ellipses($x=a\cos t$, $y=b\sin t$) and amplified Perlin noise. The function curveVertex() is used to smooth the near-ellipses. I also added small tricks to make the picture rotating and color-changing(see the comments).

I find the picture a bit similar to a daisy, with the near-ellipses being the petals, so I name it "rotating daisy". One of my friends saw it and said that it is more like "rotating fat blades".

Below is attached the source code.

```
void setup(){
  size(1000,800);              //changed size of the canvas
}

void draw(){
  background(0);
  strokeWeight(2);
  float ellipse_width = 400;
  float ellipse_height = 100;

  // little trick to control value of the red color so that it makes round trips between 0 and 255.
  float red = frameCount%512 < 256 ? frameCount%512 : (512 - frameCount%512);
  stroke(red, 100, 100);

  // draw a sequence of near-ellipses
  for (float t = 0; t < 360; t += 20) {
    noFill();

    /* calculate center of the ellipse and translate origin of the canvas to that point.
     * frameCount/2 is the varying parameter that makes the picture rotate.
     */
    float angle = frameCount/2 + t;
    float x = width/2 + 100 * cos(radians(angle));
    float y = height/2 + 100 * sin(radians(angle));
    translate(x, y);

    // tune inclination angle of the ellipse so that it aligns with its major axis.
    rotate(radians(angle));

    beginShape();
    for(float ang = 0; ang< 360; ang += 30){
```

```
    /* combine parameter equations for ellipses and Perlin noise to generate
     * coordinates for the vertices of the near-ellipse.
     */
    float _x = cos(radians(ang)) * ellipse_width / 2 + noise(cos(radians(ang)) * ellipse_width) * 50;
    float _y = sin(radians(ang)) * ellipse_height / 2 + noise(sin(radians(ang))* ellipse_height) * 50;

    // use curveVertex to smooth the near-ellipse
    curveVertex(_x, _y);
   }
   endShape(CLOSE);
   rotate(radians(-angle));
   translate(-x, -y);
  }

  // save a picture when the value of the red pigment reaches 255.
  if(frameCount == 255) save("rotatingDaisy.jpg");
}
```