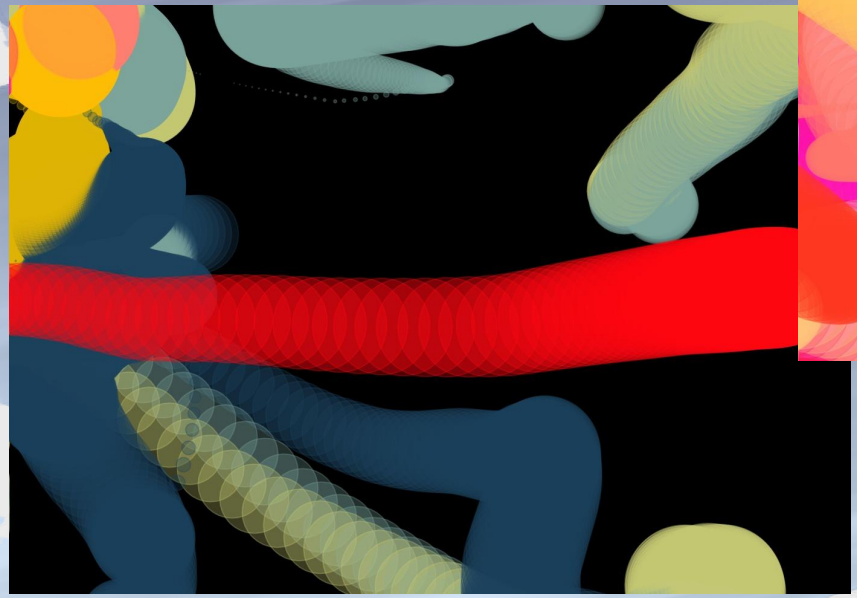
Optical Paint

Demo Time!

Any volunteers?

https://amazing-pike-058c2c.netlify.com/



Why

- We wanted to incorporate art with interactivity
- Produces art not based on artistic ability
- Intuitive and requires little mental strain

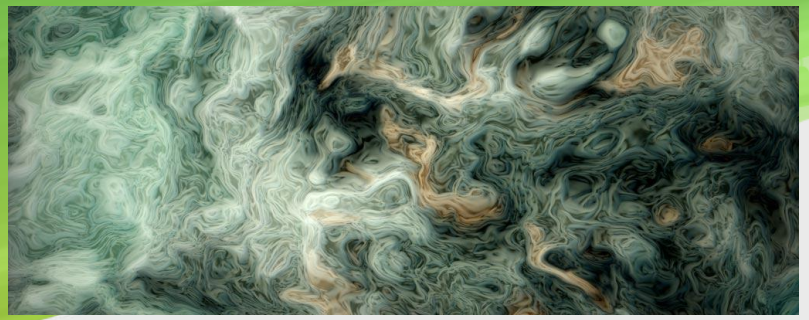
Inspiration

Jackson Pollock

Flow Cam

Starry Night

Inigo Quiles



Who

- Anyone!
- People interested in art
- People who are stressed or need a break

The Evolution

- We started off wanting to make a shader
- Researched shaders and made a couple (basic) ones of our own
- Decided our implementation would work better on the CPU
- Implemented our idea using optical flow

```
class Particle {
  constructor(x, y, mass = 1, displaySize = 10) {
    this.position = createVector(x, y);
    this.velocity = createVector(0, 0);
    this.acceleration = createVector(0, 0);
    this.color = (0);
    this.mass = mass;
    this.displaySize = displaySize;
    this.angle = 0;
    this.fluctuate = 0;
  }

  //Force used to move particle
  applyForce(force) {
    let f = p5.Vector.div(force, this.mass);
    this.acceleration.add(f);
  }

  //Applies friction to halt movement of particle
  applyFriction(friction) {
    this.velocity.mult(friction);
  }

  //Updates the properties of the particles as they move across the screen
  update() {
    this.velocity.add(this.acceleration);
    this.position.add(this.velocity);
    this.acceleration.mult(0);

    //Makes Particles
    this.fluctuate = (sin(this.angle + PI / 2) * this.displaySize) / 2 + this.displaySize / 2;
    this.angle += 0.02;
  }

  //Displays particles and specifies constrains
  display() {
    let leftWall = -50;
    let rightWall = windowWidth + 50;
    let topWall = -50;
    let bottomWall = windowHeight + 50;

    let xConstrain = constrain(this.position.x, leftWall, rightWall);
    let yConstrain = constrain(this.position.y, topWall, bottomWall);

    fill(this.color);
    stroke(this.color);
    ellipse(xConstrain, yConstrain, this.fluctuate, this.fluctuate);
```

Particle Class

- X position
- Y position
- Mass
- Size
- Velocity
- Acceleration
- Fluctuate

Mass: Affects the acceleration of the particle (Ex. Increasing mass makes them move slower)

Velocity: The speed at which the paint strokes move

Acceleration: Changes the velocity

Fluctuate: Causes the paint stroke size to increase and decrease (based on sin())

We add acceleration to velocity, and add velocity to position to move the paint strokes around.

```
      // Finds the closest optical flow zone
      for (let j = 1; j < flow.zones.length; j++) {
        let zone = flow.zones[j];
        let zoneDist = dist(
          map(zone.x, 0, opticalFlow.width, 0, width),
          map(zone.y, 0, opticalFlow.height, 0, height),
          particle.position.x,
          particle.position.y
        );

        if (zoneDist < closestDist) {
          closestZone = zone;
          closestDist = zoneDist;
        }
      }

      opticalFlowForce.x = closestZone.u * flowScaleFactor;
      opticalFlowForce.y = closestZone.v * flowScaleFactor;
    }

    particle.applyForce(opticalFlowForce);
    particle.update();
    particle.applyFriction(0.95);
    particle.display();
```

Zones

Zone: A division of the webcam. Tracks if there is any movement in the zone from frame to frame.

Iterates through array of zones to find the zone that is closest to each particle.

This allows us to push particles around based on video camera movement.

```
//Changes the color of the paint to Starry Night colors
function gogh () {
  for (let i = 0; i < 100; i++) {
    if ( i < 20) {
      particles[i].color = color(26, 66, 91, 75);
    } else if ( i < 40) {
      particles[i].color = color(198, 202, 116, 75);
    } else if ( i < 60) {
      particles[i].color = color(224, 179, 5, 75);
    } else if ( i < 80) {
      particles[i].color = color(57, 137, 185, 75);
    } else {
      particles[i].color = color(121, 164, 158, 75);
    }
  }
}

//Changes the color of the paint to a lava color theme
function lava () {
  for(let i = 0; i < 100; i++) {
    particles[i].color = color(255, random(255), random(255), 75);
  }
}
```

```
function takeScreenshot () {
  save('myCanvas.jpg');
}
```

Colors

The Gogh function changes the color each particle to be a color from our Starry Night palette

The Lava function randomizes each particle's color to be a warmer color

The takeScreenshot function allows you to take a screenshot of your painting and will automatically download it as a JPEG to your computer

Improvements

- Implement this in shader format
- Add more control over color palette
  - We tried this by adding buttons and sliders, but the speed of our sketch slowed down significantly
- Work on making brush stroke look even more realistic
  - Possibly add different types of brush strokes

Feedback

- Was it easy to understand?
- How could we make this more interesting or fun?
- Is there a more efficient way to change certain colors?