Mapping (Moving) Image Data

Supervisor:
Dipl.-Designer Mattis Kuhn

Author:
Zeinab Rahimi

Sammersemester 2022
Bauhaus-Universität Weimar

Mapping(Moving) Image Data

Zeinab Rahimi

Project Description:

The goal of this Project was to create an interactive tool using Processing programming language and Image data. Digital images are considered databases because they hold information about pixel locations and colors. The author attempts to recreate a new visual based on this data. By left-clicking on the canvas, the user can interact with the source image and recreate it in a new and interesting way.
The picture that is used as the background is a recent photo from NASA which shows the emerging stars. The photo can be accessed at the following link:

https://www.nasa.gov/image-feature/goddard/2022/nasa-s-webb-reveals-cosmic-cliffs-glittering-landscape-of-star-birth

In this application, a multiple particle system class of the Processing programming language is used.With every click of the user, new particles appear on the canvas, metaphorically representing **stars being born**.

Mapping(Moving) Image Data

Zeinab Rahimi

Processing Code:

```
// A simple Particle class

class Particle {
  PVector position;
  PVector velocity;
  PVector acceleration;
  float lifespan;

  Particle(PVector l) {
    acceleration = new PVector(0, 0.05);
    velocity = new PVector(random(-1, 1), random(-2, 0));
    position = l.copy();
    lifespan = 255.0;
  }

  void run() {
    update();
    display();
  }

  // Update position
  void update() {
    velocity.add(acceleration);
    position.add(velocity);
    lifespan -= 2.0;
  }

  // Display
  void display() {

   noStroke();
    color c = img.get(int(position.x),int(position.y));
    fill(c);
    rectMode(CENTER);
    rect(position.x, position.y, 4, 4);
  }

  // Check if the particle is still useful?
  boolean isDead() {
    return (lifespan < 0.0);
  }
}

// Using ArrayList to manage the list of Particles

class ParticleSystem {

  ArrayList<Particle> particles;    // An arraylist for all the particles
  PVector origin;                   // An origin point for where particles are birthed

  ParticleSystem(int num, PVector v) {
    particles = new ArrayList<Particle>();   // Initialize the arraylist
    origin = v.copy();                        // Store the origin point
    for (int i = 0; i < num; i++) {
      particles.add(new Particle(origin));    // Add "num" amount of particles to the arraylist
    }
  }

  void run() {
    // Cycle through the ArrayList backwards, because we are deleting while iterating
    for (int i = particles.size()-1; i >= 0; i--) {
      Particle p = particles.get(i);
      p.run();
      if (p.isDead()) {
        particles.remove(i);
      }
    }
  }
```

Mapping(Moving) Image Data

Zeinab Rahimi

```
  void addParticle() {
    Particle p;
    p = new Particle(origin);
    particles.add(p);
  }

  void addParticle(Particle p) {
    particles.add(p);
  }

  // A method to test if the particle system still has particles
  boolean dead() {
    return particles.isEmpty();
  }
}


PImage img;
ArrayList<ParticleSystem> systems;

void setup() {
  size(1387, 800);
  img = loadImage("stars.JPG");
  systems = new ArrayList<ParticleSystem>();
}
```

```
void draw() {
  background(0);
  for (ParticleSystem ps : systems) {
    ps.run();
    for (int i = 0; i < 10; i++) {
      ps.addParticle();
    }
  }
  if (systems.isEmpty()) {
    fill(255);
    textAlign(CENTER);
    textSize(60);
    text("click mouse left to add particle systems", width/2, height/2 -40);
    text("click mouse right to restart system", width/2, height/2 + 40);
  }

}

void mousePressed() {
  if (mouseButton == LEFT) {
      systems.add(new ParticleSystem(1, new PVector(mouseX, mouseY)));
    }
    else {
      // remove all particle systems
      systems.clear();
    }
  }
```

Mapping(Moving) Image Data

Zeinab Rahimi

Visitor's Interaction with the App in Summaery 2022