Algorithmic Art

The Joy of Coding

Module G

Unit #2

handPose emoji

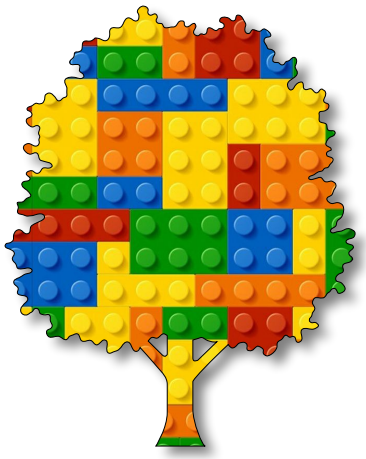


Table of Contents

 Module G Unit #2 handPose emoji

This is using the same pre-trained model but instead of points we want to emit an emoji as a particle. This makes it a little bit more complicated but not much.

Sketch G1.1

As before make sure you have the ml5.js library line of code included. You will also not that we (at time of writing) are on version 2.2.3 of p5.js.

```
<!DOCTYPE html>
<html lang="en"><head>
  <script src="https://cdn.jsdelivr.net/npm/p5@2.2.3/lib/p5.js"></script>
  <script src="https://unpkg.com/ml5@1/dist/ml5.min.js"></script>
    <link rel="stylesheet" type="text/css" href="style.css">
    <meta charset="utf-8">
  </head>
  <body>
    <main>
    </main>
    <script src="sketch.js"></script>
</body></html>
```

Sketch G1.2

Pulling the key code from the previous unit, we have the starting point. If you want to see (check it is working) then comment out the background and uncomment the image() function.

```
let video
let handPose
let hands = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
}

function gotHands(results)
{
  hands = results
}
```

Sketch G1.3 hand of hands

We want to get all the data points of either (or both) hands. To do that we create a separate variable called hand and copy the array called hands which is full of all the datapoints (keypoints). We start with our conditional expression to see if there are any keypoints detected. Then loop through all the keypoints in the hand array.

```
let video
let handPose
let hands = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        console.log(hand.keypoints.length)
      }
    }
  }
}

function gotHands(results)
```

```
{
  hands = results
}
```

Notes

This does not do anything yet, it is just adding another piece to the jigsaw. The console log will check to see how many datapoints (keypoints it has detected. There should be 21.

Sketch G1.4 emitters

We have our hands array, we need to make those datapoints become emitters rather than just a point or circle. We introduce an emitter class along with an array. In other words we want the origins of the emitters where the keypoints of your hand are.

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
        emitter.origin.y = keypoint.y
```

```
        console.log(emitter.origin.x)
      }
    }
  }
}

function gotHands(results)
{
  hands = results
}
```

```
class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
  }
}
```

Notes

We initially fill the emitter array with all the x, y values set to width/2, height/2, it doesn't really matter, we could just set them all to zero. There is nothing to see except the x values in the console. After this you can get rid of the console log.

Sketch G1.5 particles

We need some particles to play with. So we need a particle class and a particle. The particle will be an emoji which you can easily get from most modern keyboards, otherwise just use point() or circle(). If have used a flower from apple.

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
        emitter.origin.y = keypoint.y
```

```
        // console.log(emitter.origin.x)
      }
    }
  }
}

function gotHands(results)
{
  hands = results
}

class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
  }
}
```

```
class Particle
{
  constructor(x, y)
  {
    this.position = createVector(x, y)
    this.icon = '🌼'
  }

  show()
  {
    textSize(24)
    textAlign(CENTER)
    noStroke()
    text(this.icon, this.position.x, this.position.y)
  }
}
```

Notes

Nothing to see yet except a nice red canvas.

Sketch G1.6 movement

As with all emitters there is movement, you want the particles to go somewhere. They could float off, fall to the ground, fly in all directions. We are going to do the later, so we need to give those particles a velocity and acceleration, we have already given them a location (position).

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
```

```
        emitter.origin.y = keypoint.y
        emitter.addParticle()
      }
    }
  }
}

function gotHands(results)
{
  hands = results
}

class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
    this.particles = []
  }

  addParticle()
  {
    this.particles.push(new Particle(this.origin.x, this.origin.y))
  }
}

class Particle
{
  constructor(x, y)
  {
    this.position = createVector(x, y)
    this.acceleration = createVector(0, 0)
    this.velocity = p5.Vector.random2D()
    this.icon = '🌼'
  }

  show()
  {
```

```
    textSize(24)
    textAlign(CENTER)
    noStroke()
    text(this.icon, this.position.x, this.position.y)
  }

  move()
  {
    this.velocity.add(this.acceleration)
    this.position.add(this.velocity)
    this.acceleration.mult(0)
  }
}
```

Notes

There is so much to unpack here, although nothing is seen yet. For a deeper understanding of vectors, movement etc it will be worth digging into the physics workbooks.

Challenge

If you comment out a line of code such as (later on): `this.acceleration.mult(0)` You will see what difference it makes and why we multiply the acceleration by zero on each iteration.

Sketch G1.7 let's go running

Most of the pieces are in place. We now want to run everything so we will be creating functions inside both classes called run(). This will produce a flurry of particles when it sees your hand.

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
        emitter.origin.y = keypoint.y
```

```
        emitter.addParticle()
      }
    }
  }
  for (let emitter of emitters)
  {
    emitter.run()
  }
}

function gotHands(results)
{
  hands = results
}

class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
    this.particles = []
  }

  addParticle()
  {
    this.particles.push(new Particle(this.origin.x, this.origin.y))
  }

  run()
  {
    for (let i = this.particles.length - 1; i > 0; i--)
    {
      this.particles[i].run()
    }
  }
}

class Particle
```

```
{
  constructor(x, y)
  {
    this.position = createVector(x, y)
    this.acceleration = createVector(0, 0)
    this.velocity = p5.Vector.random2D()
    this.icon = '🌼'
  }

  run()
  {
    this.show()
    this.move()
  }

  show()
  {
    textSize(24)
    textAlign(CENTER)
    noStroke()
    text(this.icon, this.position.x, this.position.y)
  }

  move()
  {
    this.velocity.add(this.acceleration)
    this.position.add(this.velocity)
    this.acceleration.mult(0)
  }
}
```

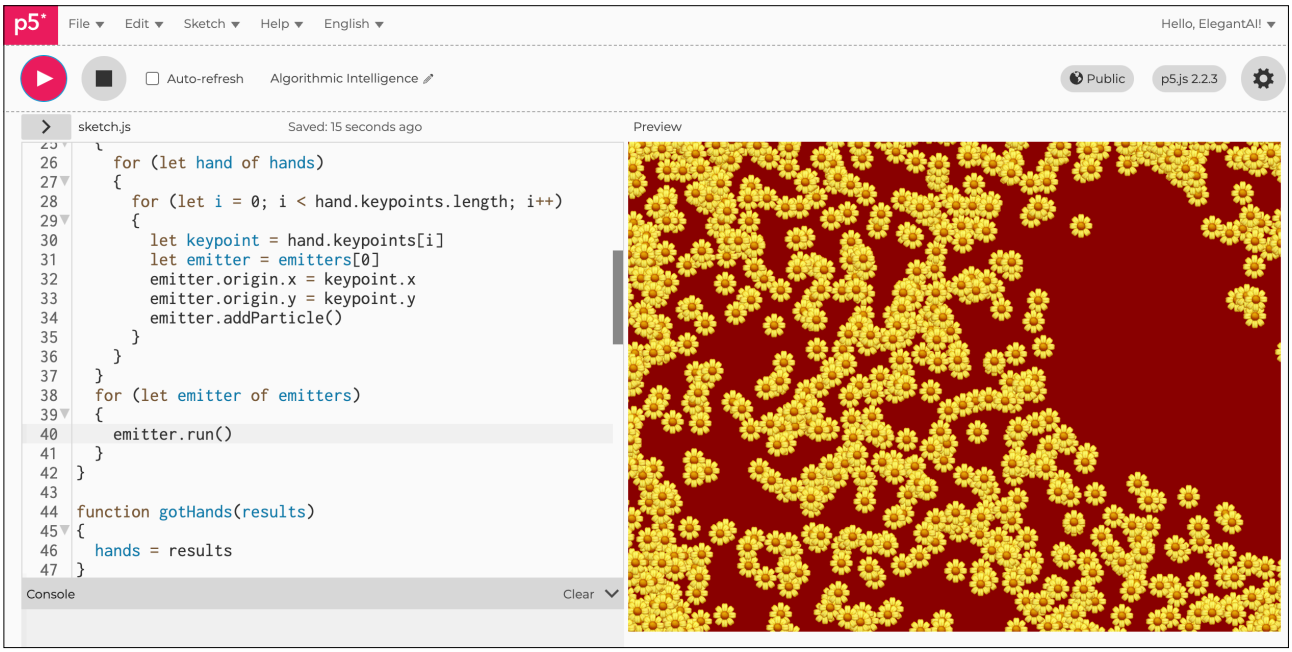
Notes

In the particle class we do something which might seem a bit mystifying, we have a for() loop that works backwards through the particles array. There is a reason for this which will become apparent shortly.

Code Explanation

`for (let i = this.particles.length - 1; i > 0; i--)`	This starts with the end of the array and works backwards through it

Figure G1.7


p5*
File
Edit
Sketch
Help
English
Hello, ElegantAI!
Auto-refresh
Algorithmic Intelligence
Public
p5.js 2.2.3
sketch.js
Saved: 15 seconds ago
Preview
for (let hand of hands)
{
for (let i = 0; i < hand.keypoints.length; i++)
{
let keypoint = hand.keypoints[i]
let emitter = emitters[0]
emitter.origin.x = keypoint.x
emitter.origin.y = keypoint.y
emitter.addParticle()
}
}
}
for (let emitter of emitters)
{
emitter.run()
}
}
function gotHands(results)
{
hands = results
}
Console
Clear

Sketch G1.8 is it alive?

As you can see it produces many particles and if they aren't culled they will slow down the sketch very quickly. So we need a life span for the particle so that it doesn't hang around forever. Hence, the reverse for() loop in the particle class. We need a function and a variable to handle this.

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
```

```
        emitter.origin.y = keypoint.y
        emitter.addParticle()
      }
    }
  }
  for (let emitter of emitters)
  {
    emitter.run()
  }
}

function gotHands(results)
{
  hands = results
}

class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
    this.particles = []
  }

  addParticle()
  {
    this.particles.push(new Particle(this.origin.x, this.origin.y))
  }

  run()
  {
    for (let i = this.particles.length - 1; i > 0; i--)
    {
      this.particles[i].run()
      if (this.particles[i].expired())
      {
        this.particles.splice(i, 1)
      }
```

```
    }
  }
}

class Particle
{
  constructor(x, y)
  {
    this.position = createVector(x, y)
    this.acceleration = createVector(0, 0)
    this.velocity = p5.Vector.random2D()
    this.icon = '🌼'
    this.lifespan = 255
  }

  run()
  {
    this.show()
    this.move()
  }

  show()
  {
    textSize(24)
    textAlign(CENTER)
    noStroke()
    text(this.icon, this.position.x, this.position.y)
  }

  move()
  {
    this.velocity.add(this.acceleration)
    this.position.add(this.velocity)
    this.acceleration.mult(0)
    this.lifespan -= 12
  }

  expired()
```

```
  {
    return this.lifespan < 0
  }
}
```

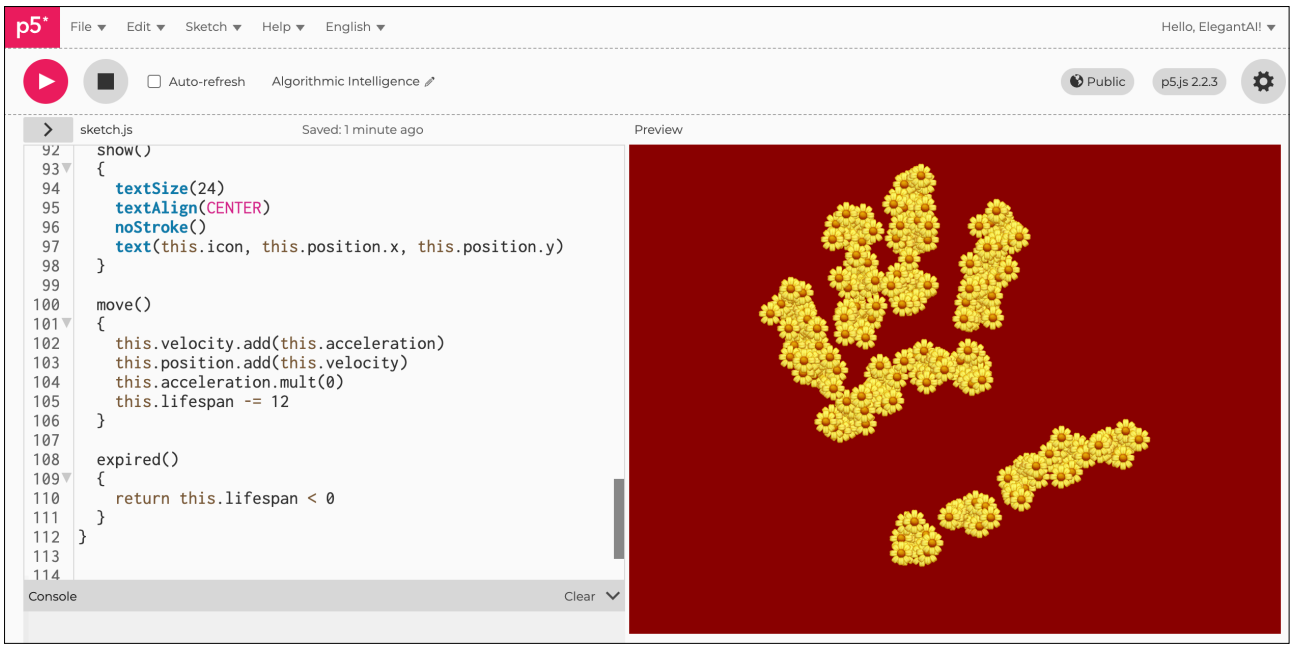
Notes

This improves the performance of the sketch no end.

Code Explanation

`if (this.particles[i].expired())`	Checks each particle to see if it has expired
`this.particles.splice(i, 1)`	When that particle has expired we remove it from the array
`this.lifespan = 255`	Our starting lifespan, the reason we use 255 is so that we can gently remove it rather than just suddenly eliminating it.
`this.lifespan -= 12`	Every iteration we reduce the lifespan by 12
`return this.lifespan < 0`	Think of this as an if statement, if the lifespan is less than zero, then return the expired() function as true for each particle.

Figure G1.8


p5*
File
Edit
Sketch
Help
English
Hello, ElegantAI!
Auto-refresh
Algorithmic Intelligence
Public
p5.js 2.2.3
sketch.js
Saved: 1 minute ago
Preview
show()
{
textSize(24)
textAlign(CENTER)
noStroke()
text(this.icon, this.position.x, this.position.y)
}
move()
{
this.velocity.add(this.acceleration)
this.position.add(this.velocity)
this.acceleration.mult(0)
this.lifespan -= 12
}
expired()
{
return this.lifespan < 0
}
}
Console
Clear

Sketch G1.9

One final tweak to make look a lot less sudden, we can reduce the alpha by how much lifespan it has before it is eliminated.

```
let video
let handPose
let hands = []
let emitters = []

async function setup()
{
  createCanvas(640, 480)
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  video = await createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  for (let i = 0; i < 21; i++)
  {
    emitters.push(new Emitter(width/2, height/2))
  }
}

function draw()
{
  background('darkred')
  // image(video, 0, 0)
  if (hands.length > 0)
  {
    for (let hand of hands)
    {
      for (let i = 0; i < hand.keypoints.length; i++)
      {
        let keypoint = hand.keypoints[i]
        let emitter = emitters[0]
        emitter.origin.x = keypoint.x
        emitter.origin.y = keypoint.y
        emitter.addParticle()
```

```
      }
    }
  }
  for (let emitter of emitters)
  {
    emitter.run()
  }
}

function gotHands(results)
{
  hands = results
}

class Emitter
{
  constructor(x, y)
  {
    this.origin = createVector(x, y)
    this.particles = []
  }

  addParticle()
  {
    this.particles.push(new Particle(this.origin.x, this.origin.y))
  }

  run()
  {
    for (let i = this.particles.length - 1; i > 0; i--)
    {
      this.particles[i].run()
      if (this.particles[i].expired())
      {
        this.particles.splice(i, 1)
      }
    }
  }
```

```
}

class Particle
{
  constructor(x, y)
  {
    this.position = createVector(x, y)
    this.acceleration = createVector(0, 0)
    this.velocity = p5.Vector.random2D()
    this.icon = '🌼'
    this.lifespan = 255
  }

  run()
  {
    this.show()
    this.move()
  }

  show()
  {
    textSize(24)
    textAlign(CENTER)
    noStroke()
    fill(255, this.lifespan)
    text(this.icon, this.position.x, this.position.y)
  }

  move()
  {
    this.velocity.add(this.acceleration)
    this.position.add(this.velocity)
    this.acceleration.mult(0)
    this.lifespan -= 12
  }

  expired()
  {
```

```
    return this.lifespan < 0
  }
}
```

Notes

A far more pleasing result I think you will agree.

Figure G2.9