Algorithmic Art

The Joy of Coding

Module B

Unit #1

pretrained handPose

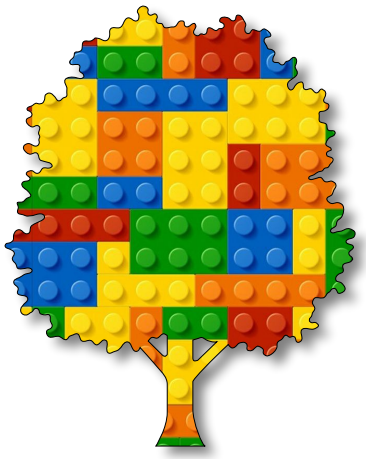


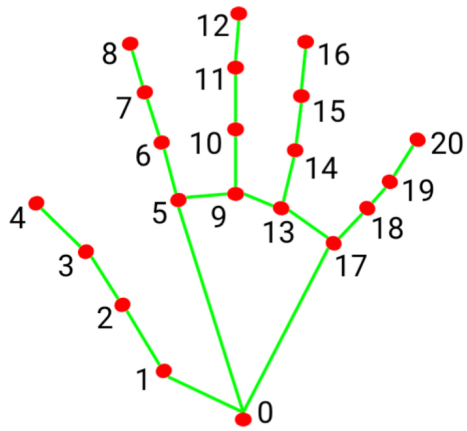
Table of Contents

Module G Unit #1 handPose pretrained AI

The handPose model is another pre-trained model that is part of the ml5.js library. We can get all the data points of your hands and use some of them to control the size and position of a circle.

The handPose model can detect 21 keypoints on a hand (see figure 1 below). It can detect both hands at the same time and determine the left from the right hand. Also, the keypoints are given as 3D co-ordinates, but we will focus on the 2D co-ordinates for now.

Figure 1: all 21 keypoints for handPose



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

The index.html file

Adding the ml5.js file to the index.html file.

```
<!DOCTYPE html>
<html lang="en"><head>
    <script src="https://cdn.jsdelivr.net/npm/p5@2.2.2/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 G1.1 our starting sketch

The process is similar to the previous unit, but we will still go through it step by step.

```
function setup()
{
  createCanvas(640, 480)
}

function draw()
{
  background(220)
}
```

Sketch G1.2 video

Getting the video feed from the webcam.

```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
}

function draw()
{
  background(220)
}
```

Notes

You should get a grey canvas and underneath it, a larger video image of you.

Sketch G1.3 putting you on the canvas

We hide the video stream and use the `image()` function. Notice that we include the width and height, not entirely necessary in this case, but if you wanted to match the canvas dimensions, this is the way to go.

```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO)
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}
```

Notes

Now you should have a video image of yourself where the canvas was, but if you lift up your hand, it is not mirrored.

Sketch G1.4 flipping the video

We can flip the video using a function in p5.js.

```
let video

function setup()
{
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
}

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}
```

Notes

You will now have a mirror image from the camera.

Sketch G1.5 the handPose model

There is a pre-trained model within ml5.js, and all we need to do is call it. We will preload it before we try to use it; otherwise, we may get error messages.

```
let video
let handPose

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

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}
```

Notes

Nothing will happen when you hold your hand up to the camera because we haven't connected the two together.

Sketch G1.6 start detecting

We use a function called `detectStart()`, and this will connect the model to the video feed.

```
let video
let handPose

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

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}
```

Notes

❗ If you run this now, you will get an error message.
It will tell you that you need a callback function. This callback function will hold the data.

Sketch G1.7 callback

Let's create a callback function called `gotHands()` and `console.log()` the `results` as we did before.

```
let video
let handPose

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

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}

function gotHands(results)
{
  console.log(results)
}
```

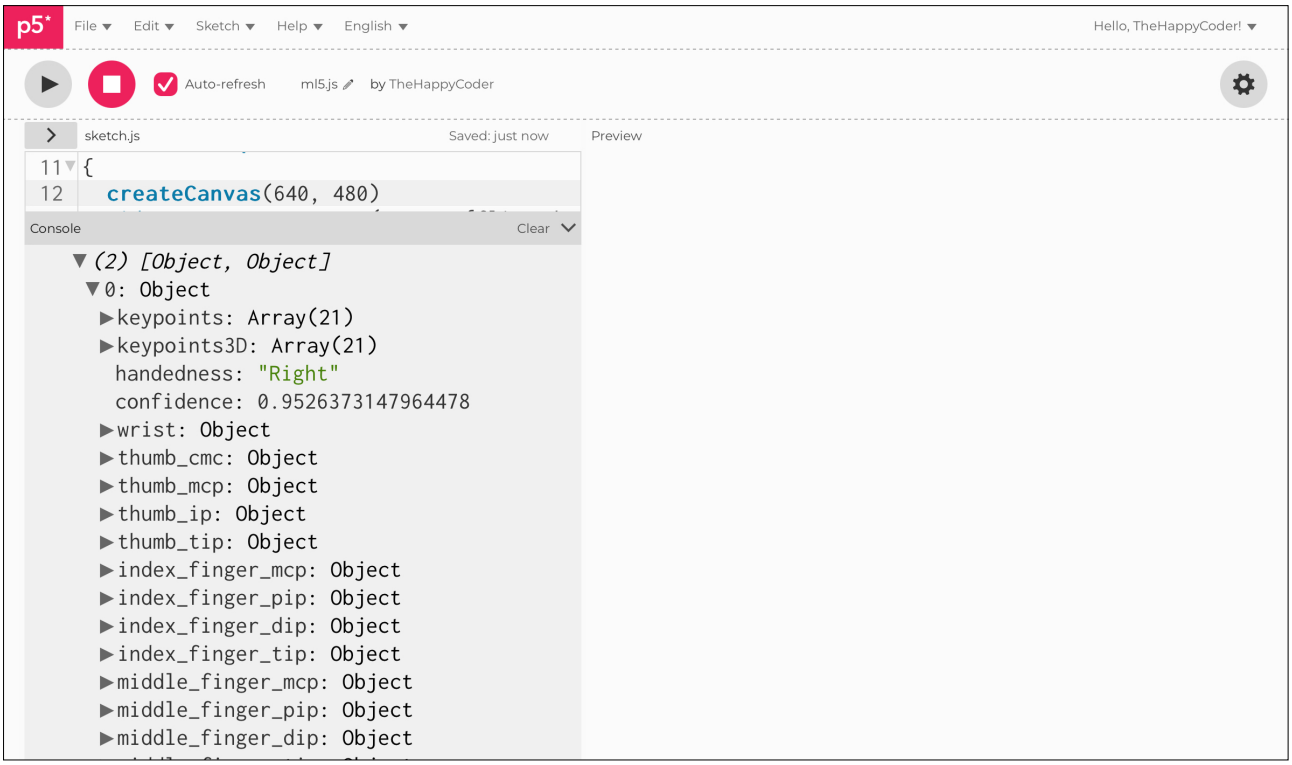
Notes

You will get an empty array when it doesn't see any hands, one array object when you hold up one hand, and two array objects when it can see two hands. If you pause the programme, you can examine the content of the arrays. You will notice how many points there are and the labelling of your fingers, plus other parts of your hand. You have 21 keypoints for each hand.

Challenge

Highly recommend that you explore the results in the console.

Figure G1.7


p5*
File
Edit
Sketch
Help
English
Hello, TheHappyCoder!
Auto-refresh
ml5.js
by TheHappyCoder
sketch.js
Saved: just now
Preview
11 {
12 createCanvas(640, 480)
Console
Clear
▼ (2) [Object, Object]
▼ 0: Object
▶ keypoints: Array(21)
▶ keypoints3D: Array(21)
handedness: "Right"
confidence: 0.9526373147964478
▶ wrist: Object
▶ thumb_cmc: Object
▶ thumb_mcp: Object
▶ thumb_ip: Object
▶ thumb_tip: Object
▶ index_finger_mcp: Object
▶ index_finger_pip: Object
▶ index_finger_dip: Object
▶ index_finger_tip: Object
▶ middle_finger_mcp: Object
▶ middle_finger_pip: Object
▶ middle_finger_dip: Object

Sketch G1.8 an array of hands

We need to store this data and access the array of your hands.

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

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

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
}

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

Notes

We create an empty array to put the data in and fill it with the results in the callback function.

Sketch G1.9 the keypoints

We want to cycle through all the keypoints in the hands array and draw them. For this, we use a nested loop.

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

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

function draw()
{
  background(220)
  image(video, 0, 0, width, height)
  for (let i = 0; i < hands.length; i++)
  {
    for (let j = 0; j < hands[i].keypoints.length; j++)
    {
      let keypoint = hands[i].keypoints[j]
      fill('yellow')
      circle(keypoint.x, keypoint.y, 10)
    }
  }
}

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

Notes

Notice that the yellow dots are not mirrored!

Sketch G1.10 mirroring the yellow dots

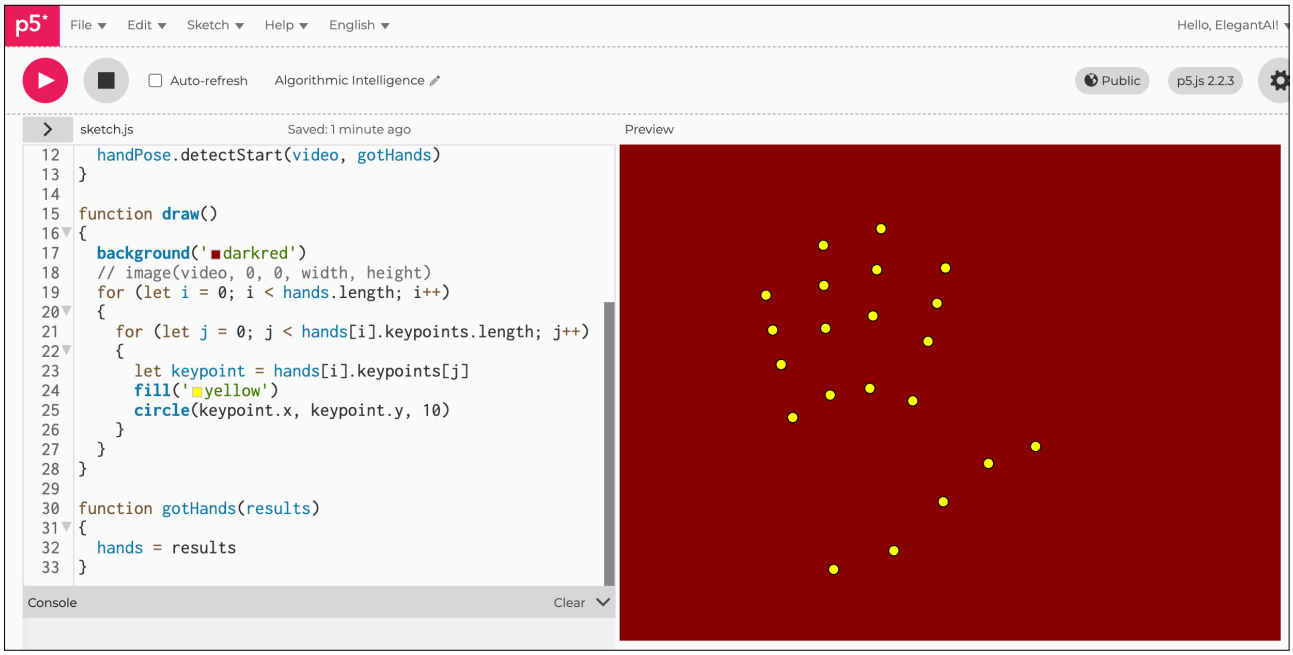
We can add the `flipped` is `true`. Also we will remove the video of you and replace with a darkred background.

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

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

function draw()
{
  background('darkred')
  // image(video, 0, 0, width, height)
  for (let i = 0; i < hands.length; i++)
  {
    for (let j = 0; j < hands[i].keypoints.length; j++)
    {
      let keypoint = hands[i].keypoints[j]
      fill('yellow')
      circle(keypoint.x, keypoint.y, 10)
    }
  }
}

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

Notes

The points are mirrored on both hands.

Figure G1.10



Sketch G1.11 reference keypoints

Instead of points, let's get the index reference for each point, because then we can use specific keypoints.

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

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

function draw()
{
  background(‘darkred)
  // image(video, 0, 0, width, height)
  for (let i = 0; i < hands.length; i++)
  {
    for (let j = 0; j < hands[i].keypoints.length; j++)
    {
      let keypoint = hands[i].keypoints[j]
      fill('yellow')
      // circle(keypoint.x, keypoint.y, 10)
      text(i, keypoint.x -10, keypoint.y)
      text(j, keypoint.x, keypoint.y)
    }
  }
}

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

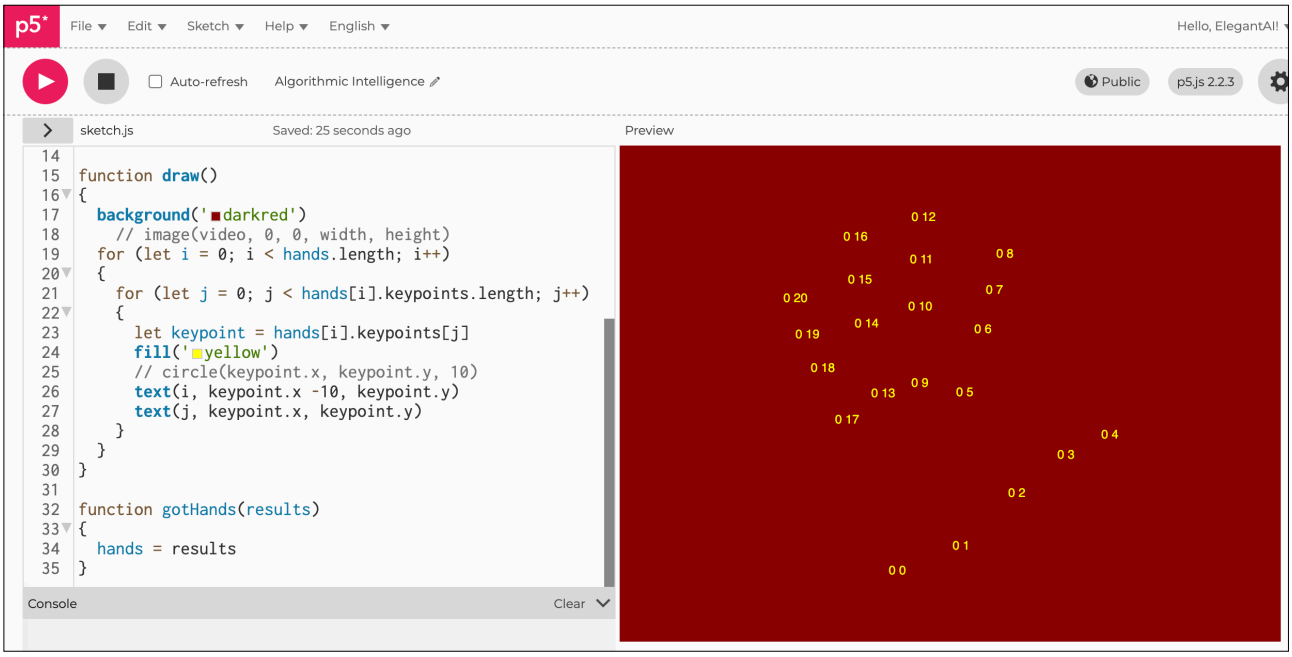
```
}
```

Notes

You will notice that when you hold up two hands, you get a 1 for the left hand followed by the reference number on that hand, and a 0 for the right hand followed by the corresponding reference number on the other hand.

We want the index finger and thumb (below are the reference numbers):
Index finger: keypoint 8
Thumb: keypoint 4

Figure G1.11



Sketch G1.12 index and thumb

Just to prove we have the right keypoint reference, we can write the words onto the canvas. We are only interested in one hand at a time. We can remove the commented-out nested loops.

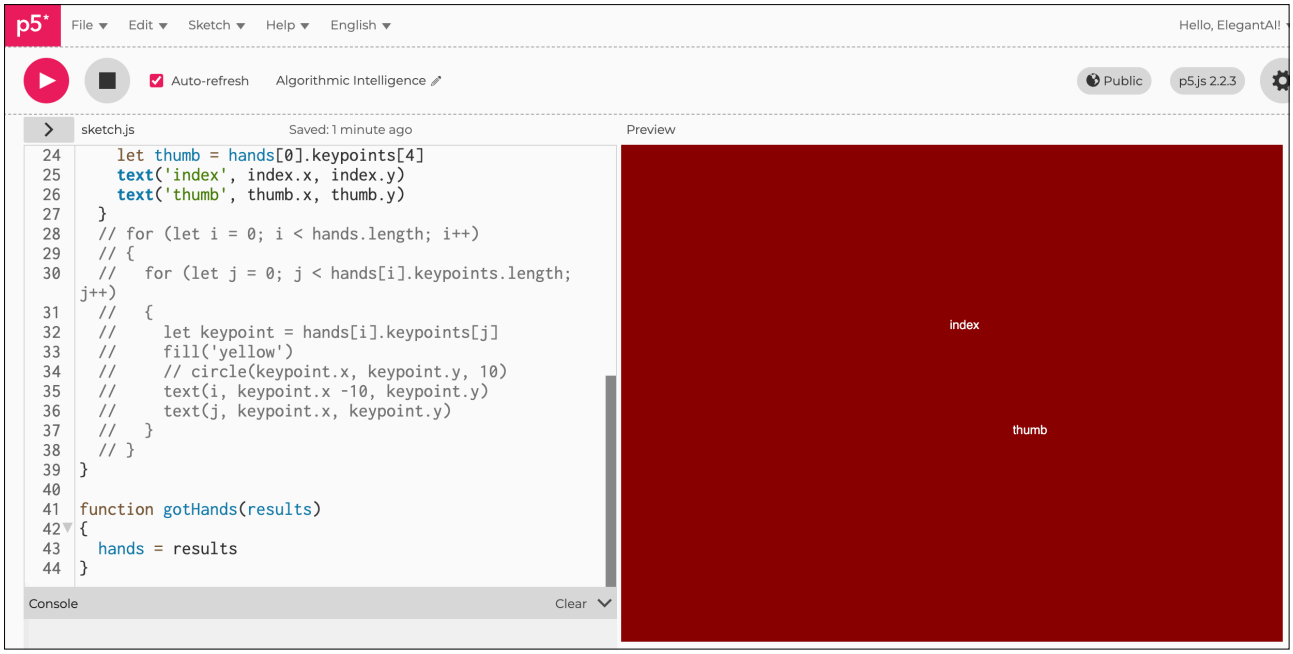
```
let video
let handPose
let hands = []

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

function draw()
{
  background('darkred')
  // image(video, 0, 0, width, height)
  if (hands.length > 0)
  {
    fill('white')
    noStroke()
    let index = hands[0].keypoints[8]
    let thumb = hands[0].keypoints[4]
    text('index', index.x, index.y)
    text('thumb', thumb.x, thumb.y)
  }
  // for (let i = 0; i < hands.length; i++)
  // {
  //   for (let j = 0; j < hands[i].keypoints.length; j++)
  //   {
  //     let keypoint = hands[i].keypoints[j]
  //     fill('yellow')
```

```
  //     // circle(keypoint.x, keypoint.y, 10)
  //     text(i, keypoint.x -10, keypoint.y)
  //     text(j, keypoint.x, keypoint.y)
  //   }
  // }
}

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

Notes

You should have the index finger and thumb correctly identified using either hand.

Figure G1.12



Sketch G1.13 circle size

Instead of writing text, we will draw a circle that changes depending on the distance (d) between those two fingers.

❗ We have completely removed the nested loops as we don't need that information anymore.

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

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

function draw()
{
  background('darkred')
  // image(video, 0, 0, width, height)
  if (hands.length > 0)
  {
    // fill('black')
    noFill()
    strokeWeight(5)
    stroke('yellow')
    let index = hands[0].keypoints[8]
    let thumb = hands[0].keypoints[4]
    let d = dist(index.x, index.y, thumb.x, thumb.y)
    circle(index.x, index.y, d)
    // text('index', index.x, index.y)
    // text('thumb', thumb.x, thumb.y)
  }
```

```
}


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

Notes

What you get is the circle changing size as you open and close your thumb and index finger. However, I have left the centre on the index finger for now. We want it halfway between the index finger and the thumb. You may not really be able to see it with the image commented out but the circle is entered around the index finger.

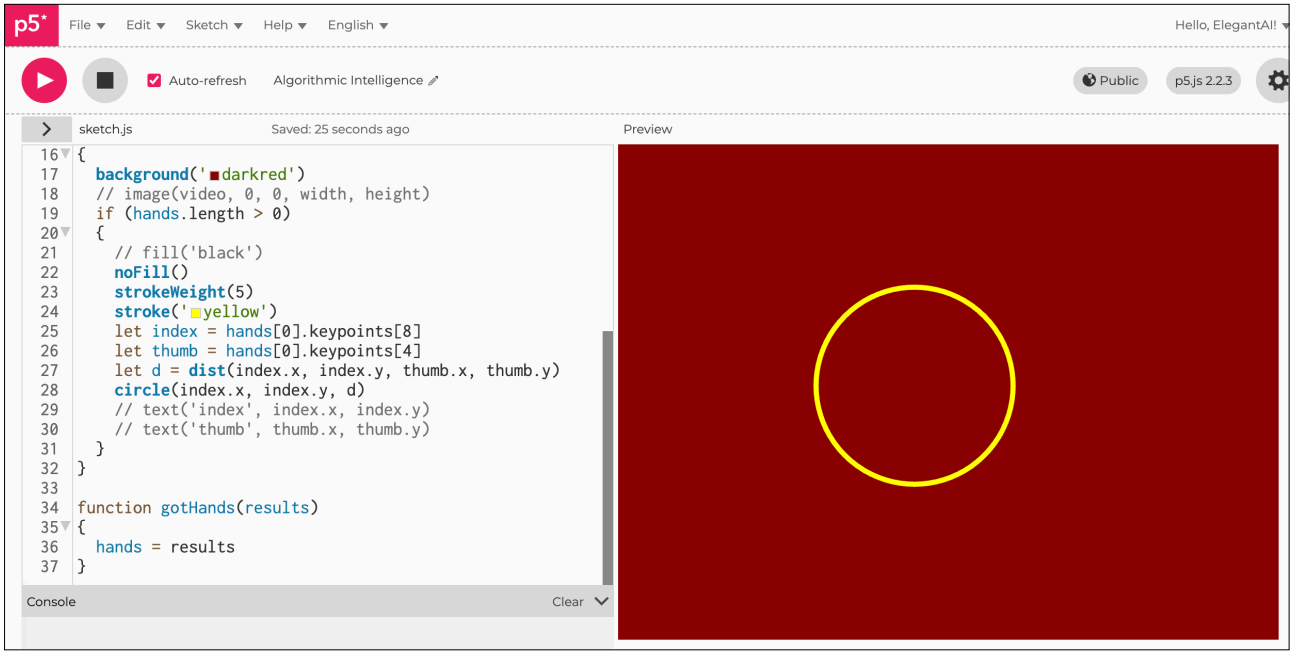
Challenges

1. Uncomment the line of code to show the video feed
2. You could try other fingers.

Code Explanation

`let d = dist(index.x, index.y, thumb.x, thumb.y)`	Measures the distance between the x, y co-ordinates of the index finger with the x, y co-ordinates of the thumb

Figure G1.13


p5*
File
Edit
Sketch
Help
English
Hello, ElegantAI!
Auto-refresh
Algorithmic Intelligence
Public
p5.js 2.2.3
sketch.js
Saved: 25 seconds ago
Preview
{
background('darkred')
// image(video, 0, 0, width, height)
if (hands.length > 0)
{
// fill('black')
noFill()
strokeWeight(5)
stroke('yellow')
let index = hands[0].keypoints[8]
let thumb = hands[0].keypoints[4]
let d = dist(index.x, index.y, thumb.x, thumb.y)
circle(index.x, index.y, d)
// text('index', index.x, index.y)
// text('thumb', thumb.x, thumb.y)
}
}
function gotHands(results)
{
hands = results
}
Console
Clear

Sketch G1.14 adding some coloured color

Now we have the centre of the circle between our thumb and index finger.

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

async function setup()
{
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  colour = color(255, 255, 0)
}

function draw()
{
  background(220)
  if (hands.length > 0)
  {
    noFill()
    strokeWeight(5)
    stroke(colour)
    let index = hands[0].keypoints[8]
    let thumb = hands[0].keypoints[4]
    let d = dist(index.x, index.y, thumb.x, thumb.y)
    circle(index.x, index.y, d)
  }
}

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

Notes

Just make a few tweaks which doesn't do much as of yet. The colour variable holds the color() function values. We can change these at will.

Challenge

How about drawing a square, rectangle, or ellipse?

Figure G1.14

Sketch G1.15 checking the distance

Now that we have distance between thumb and index finger we can change the colour of the circle to a random colour. Every time you close your index finger and thumb the circle should change colour.

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

async function setup()
{
  ml5.setBackend("webgl")
  handPose = await ml5.handPose({flipped: true})
  createCanvas(640, 480)
  video = createCapture(VIDEO, {flipped: true})
  video.hide()
  handPose.detectStart(video, gotHands)
  colour = color(255, 255, 0)
}

function draw()
{
  background(220)
  if (hands.length > 0)
  {
    noFill()
    strokeWeight(5)
    stroke(colour)
    let index = hands[0].keypoints[8]
    let thumb = hands[0].keypoints[4]
    let d = dist(index.x, index.y, thumb.x, thumb.y)
    circle(index.x, index.y, d)
    if (d < 25)
    {
      colour = color(random(255), random(255), random(255))
    }
    else
```

```
    {
      stroke(colour)
    }
  }
}

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

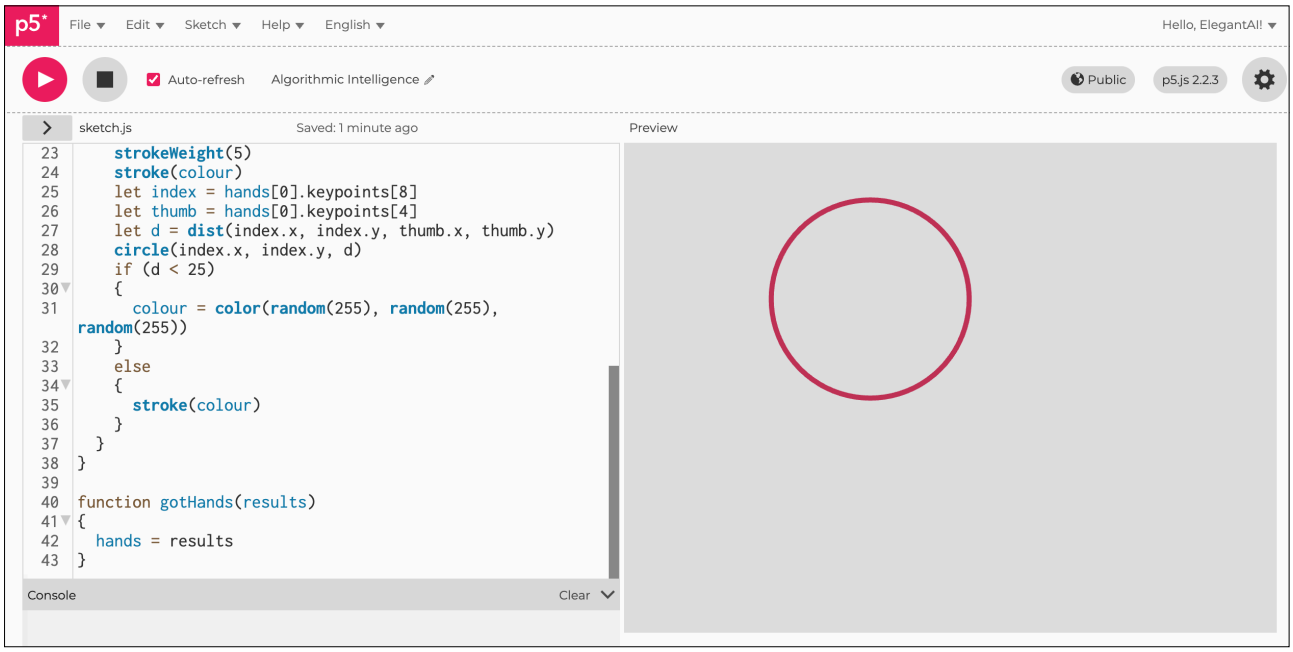
Notes

The reason why we have an `else{ }` condition is so that it settles on one random colour otherwise it would cycle through all the random colours, flickering continuously.

Challenge

Change the circle fill colour.

Figure G1.15



Sketch G1.16 slimmed down sketch

! Get rid of everything except what is left below and change the highlighted code.

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

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

function draw()
{
  background('darkred')
  if (hands.length > 0)
  {
    noFill()
    strokeWeight(2)
    stroke('yellow')
    let index = hands[0].keypoints[8]
    let thumb = hands[0].keypoints[4]
  }
}

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

Notes

We will see nothing yet. Just check for errors.

Sketch G1.17 many hands make light work

We need to make sure we detect both hands for this. One hand is hands[0] and the other hand is hands[1].

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

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

function draw()
{
  background('darkred')
  if (hands[0] && hands[1])
  {
    noFill()
    strokeWeight(2)
    stroke('yellow')
    let indexL = hands[0].keypoints[8]
    let thumbL = hands[0].keypoints[4]
    let indexR = hands[1].keypoints[8]
    let thumbR = hands[1].keypoints[4]
  }
}

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

Notes

We are getting the thumb and index finger of each hand. Still nothing to see here.

Code Explanation

`if (hands[0] && hands[1])`	Checks to see if it can see both hands

Sketch G1.18 the vertex

Now we can draw the vertex points for our irregular shape. When it detects both hands it joins up all the points. Move your fingers around to see the effect. Just keep both hands in view.

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

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

function draw()
{
  background('darkred')
  if (hands[0] && hands[1])
  {
    noFill()
    strokeWeight(2)
    stroke('yellow')
    let indexL = hands[0].keypoints[8]
    let thumbL = hands[0].keypoints[4]
    let indexR = hands[1].keypoints[8]
    let thumbR = hands[1].keypoints[4]
    beginShape()
    vertex(indexL.x, indexL.y)
    vertex(indexR.x, indexR.y)
    vertex(thumbR.x, thumbR.y)
    vertex(thumbL.x, thumbL.y)
    endShape(CLOSE)
  }
```

```
}


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

Challenges

1. What shapes could you create with all the fingers?
2. Change the fill colour when you bring your fingers together

Figure G1.18