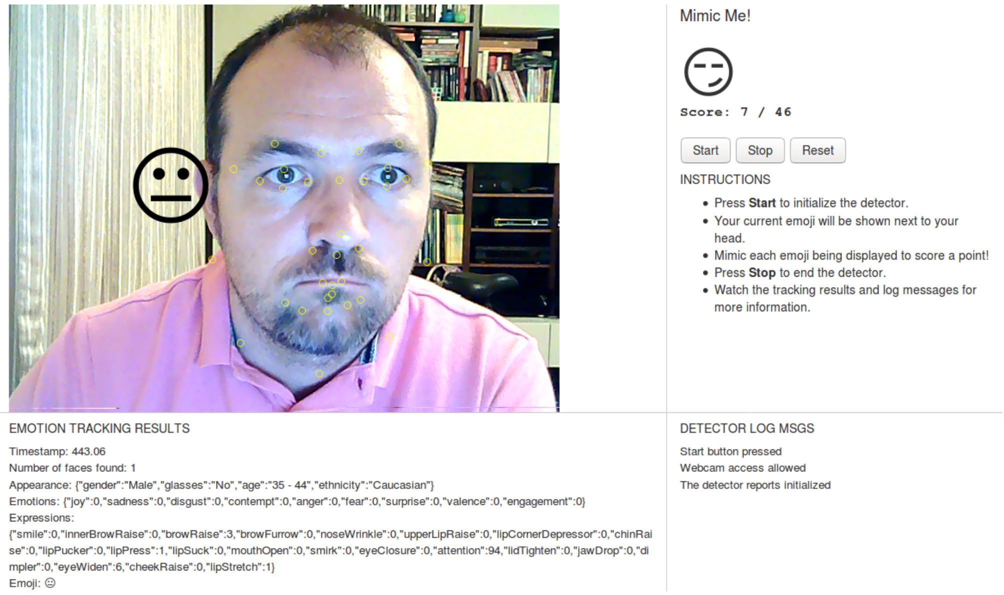
AIND-CV-Mimic Project

Report on mimic.js implementation

Student: Miodrag Vujkovic E-mail: vujkovic@gmail.com



Mimic Me!

😏

Score: 7 / 46

INSTRUCTIONS

- Press **Start** to initialize the detector.
- Your current emoji will be shown next to your head.
- Mimic each emoji being displayed to score a point!
- Press **Stop** to end the detector.
- Watch the tracking results and log messages for more information.

EMOTION TRACKING RESULTS

```
Timestamp: 443.06
Number of faces found: 1
Appearance: {"gender":"Male","glasses":"No","age":"35 - 44","ethnicity":"Caucasian"}
Emotions: {"joy":0,"sadness":0,"disgust":0,"contempt":0,"anger":0,"fear":0,"surprise":0,"valence":0,"engagement":0}
Expressions:
{"smile":0,"innerBrowRaise":0,"browRaise":3,"browFurrow":0,"noseWrinkle":0,"upperLipRaise":0,"lipCornerDepressor":0,"chinRaise":0,"lipPucker":0,"lipPress":1,"lipSuck":0,"mouthOpen":0,"smirk":0,"eyeClosure":0,"attention":94,"lidTighten":0,"jawDrop":0,"dimpler":0,"eyeWiden":6,"cheekRaise":0,"lipStretch":1}
Emoji: 😐
```

DETECTOR LOG MSGS

```
Start button pressed
Webcam access allowed
The detector reports initialized
```

Task 1. Display feature points

Feature points are displayed using drawFeaturePoints(canvas, img, face) function using yellow circles with 4 pixels radius. Yellow color was set using strokeStyle on canvas context and circles were drawn with arc function on the Path. Function code is presented below:

```
// Draw the detected facial feature points on the image
function drawFeaturePoints(canvas, img, face) {
// Obtain a 2D context object to draw on the canvas
var ctx = canvas.getContext('2d');

// TODO: Set the stroke and/or fill style you want for each feature point marker
// See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D#Fill_and_stroke_styles
// <your code here>
ctx.strokeStyle = 'yellow';
// Loop over each feature point in the face
for (var id in face.featurePoints) {
var featurePoint = face.featurePoints[id];

// TODO: Draw feature point, e.g. as a circle using ctx.arc()
// See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/arc
// <your code here>
ctx.beginPath();
ctx.arc(featurePoint.x, featurePoint.y, 4, 0, 2 * Math.PI);
ctx.stroke();
}
}
```

Task 2. Show dominant emoji

Dominant emoji was presented using drawEmoji(canvas, img, face) function with fill text.

```
// Draw the dominant emoji on the image
function drawEmoji(canvas, img, face) {
// Obtain a 2D context object to draw on the canvas
var ctx = canvas.getContext('2d');

// TODO: Set the font and style you want for the emoji
// <your code here>
ctx.font="100px serif";
ctx.fillStyle="black";
// TODO: Draw it using ctx.strokeText() or fillText()
// See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/fillText
// TIP: Pick a particular feature point as an anchor so that the emoji sticks to your face
// <your code here>
ctx.fillText(face.emojis.dominantEmoji, face.featurePoints[0].x-100, face.featurePoints[0].y-50);
}
```

Task 3. Implement the game Mimic Me!

The game implementation is simple. Random emoji is shown and we have 10 seconds to match it. If we match it, correct and total score is increased and we move to the next emoji. If we don't match the emoji in 10 seconds, number of total tries is increased and we move on to the next emoji. The game code is presented below:

```
// play the game
function play(faces) {
// get start time
var startTime = getTimeInSeconds();

// if we matched the emoji increase correct, total, show the score and move to the next emoji
if (target == toUnicode(faces[0].emojis.dominantEmoji)) {
correct++;
total++;
showRandomEmoji();
time = time + showInterval;
setScore(correct, total);
}

// if we did not match emoji in given time, increase the total and move to the next emoji
if (startTime >= time) {
showRandomEmoji();
total++;
time = time + showInterval;
setScore(correct, total);
}
}
```