Artificial Intelligence Nanodegree Alexandru Korotcov

Project: Mimic Me July 29th, 2017

Mimic Me Game

Overview

This project is an implementation of a game “Mimic Me”. A face is detected and tracked in a video captured by computer camera. The facial expressions are identified using Affectiva’s Emotion-as-a-Service API. Each face displays key point to identify emotion with an appropriate emoji next to it. In this game the player needs to mimic a random emoji displayed by the computer!

Display Feature Points

f*unction drawFeaturePoints(canvas, img, face)* displays the feature points on top of the webcam image that are returned along with the metrics. It accepts three parameters: *canvas* (HTML DOM element to draw on), *img* (image frame that was processed), and *face* (an object with all the detected feature points and metrics for a face). The stroke style is set to white and all feature points were drawn as hollow cycles with 2 pixels radius.

```
150  // Draw the detected facial feature points on the image
151  function drawFeaturePoints(canvas, img, face) {
152    // Obtain a 2D context object to draw on the canvas
153    var ctx = canvas.getContext('2d');
154
155    // TODO: Set the stroke and/or fill style you want for each feature point marker
156    // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D#Fill_and_stroke_styles
157    ctx.strokeStyle = 'white';
158
159    // Loop over each feature point in the face
160    for (var id in face.featurePoints) {
161      var featurePoint = face.featurePoints[id];
162
163      // TODO: Draw feature point, e.g. as a circle using ctx.arc()
164      // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/arc
165      ctx.beginPath();
166      ctx.arc(featurePoint.x, featurePoint.y, 2, 0, 2 * Math.PI);
167      ctx.stroke();
168    }
169  }
```

Show Dominant Emoji

In addition to feature points and metrics that capture facial expressions and emotions, the Affectiva API also reports back what emoji best represents the current emotional state of a face. This is referred to as the *dominant emoji. function drawEmoji(canvas, img, face)* accepts the same input as *drawFeaturePoints()* and draws the *dominant emoji* on the image next to the face. The size of this emoji is set to 80px and the feature point #10 is used as an anchor, thus the emoji sticks to the face. In addition, the emoji is slightly shifted to the right so it is not overlapping with the face.

```
171  // Draw the dominant emoji on the image
172  function drawEmoji(canvas, img, face) {
173    // Obtain a 2D context object to draw on the canvas
174    var ctx = canvas.getContext('2d');
175
176    // TODO: Set the font and style you want for the emoji
177    ctx.font = '80px serif';
178    // TODO: Draw it using ctx.strokeText() or fillText()
179    // See: https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/fillText
180    // TIP: Pick a particular feature point as an anchor so that the emoji sticks to your face
181    var anchor = face.featurePoints[10];
182    dominantEmoji = face.emojis.dominantEmoji; // use this dominant Emoji to compare with targer Emoji in the game
183    ctx.fillText(dominantEmoji, anchor.x+25, anchor.y-50);
184  }
```

Mimic Me implementation

In this game, the computer displays an emoji at random, and the goal of the human player is to mimic that emoji as best as they can. Affectiva's SDK can recognize 13 different emojis. The computer continually monitor the player's face, and as soon as they are able to mimic the face the game updates score and moves on to the next random emoji. If human are not able to mimic emoji within 6 seconds, the game updates only total score and moves to the next random emoji.

Buttons functionality. The *"Start"* button is initializing the game by setting all scores to 0, displaying facial features point, the dominant emoji, and the target emoji to mimic. *"Stop"* button stops the game and reset all scores and replaces target emoji with the question mark sign. *"Reset"* button resets all scores and continue the game.

Helper *function wait(ms)* stops code execution for a set time *ms* in milliseconds, and used in the game implementation to delay some functions start.

```
208  // this function will stop code execution for a ms time in milliseconds
209  function wait(ms){
210     var start = new Date().getTime();
211     var end = start;
212     while(end < start + ms) {
213       end = new Date().getTime();
214    }
215  }
```

Helper *function gameInit()* is used to initialize correctly mimicked emoji score, total attempts to mimic, displays initial scores, waits for a second, and then call a function to display a target emoji to mimic. This function executed on "Start" and "Reset" button press.

```
199  // initialization-reset function will set all scores to 0 and restart target Emoji with 1 seconds delay
200  function gameInit(){
201       score = 0; // initialize correctly mimicked emoji score
202       scoreTotal = 0; // total attempts to mimic
203       setScore(score, scoreTotal); // update scores
204       wait(1000) // call wait for 1000ms before show the target Emoji
205       showEmoji(); // displays a target emoji to mimic
206  }
```

Helper *function showEmoji()* randomly picks up an emoji from the list of 13 emoji's, set it as a target emoji, and displays it in the right corner of the screen, thus player can start mimic it.

```
217  // this functions displays a target Emoji
218  function showEmoji(){
219      var idTargetEmoji = Math.floor(Math.random() * emojis.length); // randomly pick up an index for traget Emoji
220      targetEmoji = emojis[idTargetEmoji]
221      setTargetEmoji(targetEmoji); // Display the target Emoji
222  }
```

Helper *function updateScoreEmoji()* update scores, wait for 2 seconds, and then displays new target Emoji to mimic.

```
224  // this function will update scores and displays new target Emoji
225  function updateScoreEmoji() {
226       setScore(score, scoreTotal); // udate game scores
227       wait(2000); // wait for 2000 ms before showing the next tagret Emoji
228       timeStampInit = timeStampCurrent++;
229       showEmoji (); // update target emoji
230  }
```

Main *function mimicMeMain()* executes the main logic of the game. It is continuously monitoring if the dominant Emoji matching, the target Emoji. In case of dominant emoji is matching the target emoji within 7 seconds the winning and total scores are increased by 1, the "DETECTOR LOG MSGS" massage is updated with the following messages "Good job! You did it!" and "Your dominant Emoji was: " followed by the dominant emoji picture. The code execution pauses for 2 seconds before randomly updating the target emoji. If player is not able to mimic the target emoji

within 7 seconds, the system updated only total attempts score and return the following messages: "You run out of time! Try again! " and “Your dominant Emoji was: “ followed by the dominant emoji picture. As soon as messages are displayed the code execution pauses for 2 seconds before randomly updating the target emoji.

```
// main function is continuously monitoring if the dominant Emoji matching, the target Emoji.
// if they match within 7 seconds the scores are increased by 1 target Emoji updates
// otherwise it updates only total score and tagret Emoji
function mimicMeMain(){
    if (toUnicode(dominantEmoji) == targetEmoji) {
        scoreTotal++; // Score total plus one
        score++; // Score plus one if the dominant Emoji matched the target Emoji
        $("#logs").html("Good job! You did it! ");
        log('#logs', "Your dominant Emoji was: "+dominantEmoji);
        updateScoreEmoji(); // Updates scores and target Emoji
    }
    if (((timeStampCurrent - timeStampInit + 1) % 8) === 0){
        scoreTotal++; // Score total plus one

       /* log('#logs', "Time stamp " + timeStampCurrent);
        log('#logs', "Time stamp modulo of 7 " + ((timeStampCurrent - timeStampInit) % 7));
        log('#logs', "Time stamp initial " + timeStampInit);  */
        $("#logs").html("You run out of time! Try again! ");
        log('#logs', "Your dominant Emoji was: "+dominantEmoji);
        updateScoreEmoji(); // Updates scores and tagret Emoji
    }
}
```

The entire repository is locates at:

https://github.com/korotulea/Artificial-Intelligence-NanoDegree-Udacity/tree/master/AIND-CV-Mimic

Enjoy the game!