Senior Project:

Computing Countenance

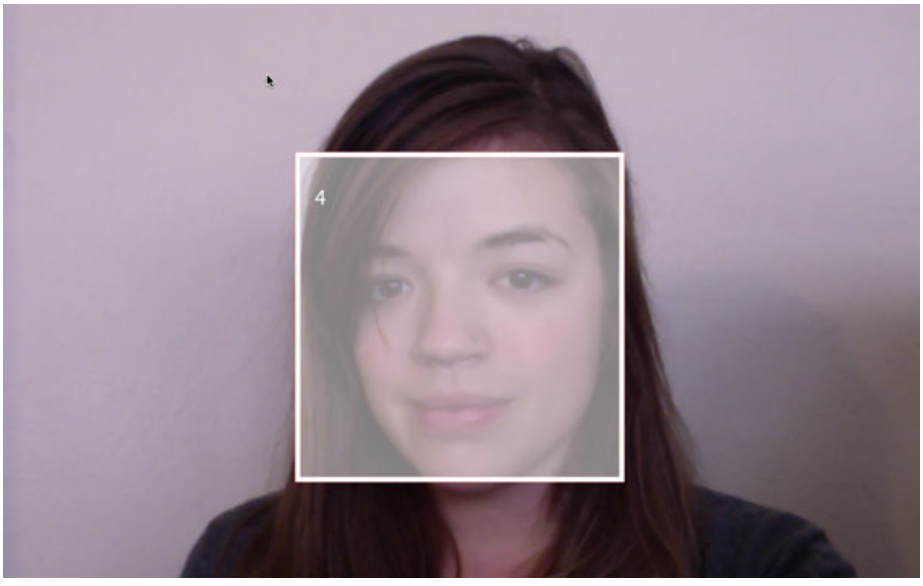
Liz Klarecki

Overview

For my senior project, I created a generative data visualization written in Processing - programming language and development environment - that interprets the data from captured images of its viewers. Using OpenCV face detection libraries in Processing, an iMac camera is running during the Devos Art Museum hours. Whenever it detects a face, an image is captured, and the image is then parsed through the Rekognition API to gather data – age, gender, smile, and emotion. The image and data are stored in a JSON file and in a MySQL database. The Processing program creates graphics based on each set of data to be added to a visualization projected on the gallery wall above. There is a supplementary website on the iMac screen that explains the project and displays all the images captured with the respective data collected. Below is a detailed description of all components of my project, followed by a diagram.

Processing + OpenCV Face Detection

The first element of the project is the face detection component. Written in Processing with the assistance of the OpenCV face detection library, the program - or as Processing refers to it, the sketch - opens the webcam and loads the Frontal Face library. The Face class creates a Face object every time a new face is detected and adds it to the array list in the main program of current faces. When a face is detected a conditional checks the array list so the detection persists rather than continually detecting the same face (and saving hundreds of repetitive images). This is explained in more depth in the code sample below. There is a timer variable that gives time for the face to disappear before it deletes the face from the array list. There will be a demo of how the face detection part functions during my presentation, however below is a screenshot from testing.



Face Detection Code Sample

When OpenCV detection is called, it gives a new array of Rectangle objects for each frame. Meaning if an image were to be captured every time a face was detected, there would be hundreds of repetitive images. Below is the solution to this issue – using an arraylist to store Face objects and allowing the detection to persist.

```
Rectangle[] faces = opencv.detect();
```

If there is nothing in the ArrayList, a new Face object is created for the Rectangles in the face array created by OpenCV.

```
if (faceList.isEmpty()) {
  for (int i = 0; i < faces.length; i++) {
    faceList.add(new Face(faces[i].x,faces[i].y,faces[i].width,faces[i].height));
  }
}
```

If there are less Face objects in the arraylist than Rectangles detected from OpenCV, a new Face object is created for the residual Rectangles. Booleans keep track of which Rectangles have been matched to prevent two Face objects from accounting for the same face.

```
else if (faceList.size() <= faces.length) {
  boolean[] used = new boolean[faces.length];
  for (Face f : faceList) {
    float record = 50000;
    int index = -1;
    for (int i = 0; i < faces.length; i++) {
      float d = dist(faces[i].x,faces[i].y,f.r.x,f.r.y);
      if (d < record && !used[i]) {
        record = d;
        index = i;
      }
    }
    used[index] = true;
    f.update(faces[index]);
  }
  for (int i = 0; i < faces.length; i++) {
    if (!used[i]) {
      faceList.add(new Face(faces[i].x,faces[i].y,faces[i].width,faces[i].height));
    }
  }
}
```

If there are more Face objects in the arraylist than face Rectangles found with OpenCV, the Face objects get matched and leftover ones are marked to be deleted.

```
else {
   for (Face f : faceList) {
     f.available = true;
   }
   for (int i = 0; i < faces.length; i++) {
     float record = 50000;
     int index = -1;
     for (int j = 0; j < faceList.size(); j++) {
       Face f = faceList.get(j);
       float d = dist(faces[i].x,faces[i].y,f.r.x,f.r.y);
       if (d < record && f.available) {
         record = d;
         index = j;

       }
     }
     Face f = faceList.get(index);
     f.available = false;
     f.update(faces[i]);
   }
```

Rekognition + JSON + MySQL

During the creation of each Face object, the program saves an image of the face and sends a call via a HTTP query string to the insert PHP script. The insert PHP script takes the image and parses it through the Rekognition API which returns a JSON Object with the following data: id, age, gender, emotion, smile, day (with a timestamp), and overall confidence. The JSON Object gets saved to a JSON Array in an external file and the script parses that file and then calls an SQL insert operation to save the datasets into a MySQL table. Only images with a high confidence level and sufficient data get inserted to the table to prevent blurry images or non-face images from being included in the visualization.

The data for emotion is a confidence level for each emotion: happy, sad, surprised, angry, calm, and confused. The PHP script compares the values for each emotion category and stores the one with the max value to be used in the visualization. Below is a short sample of the JSON Array saved for each detected face and a screenshot of the MySQL database schema (with data from a testing session).

tablenum	id	filename	confidence	age	smile	glasses	eyes	mouth	sex	emotion	happy	sad	confused	angry	surprised	calm	disgust	time	day	hour
1	0	day18-img0.jpg	1	25.78	0.83	0.08	0.38	0.04	1	happy	0.59	0.05	0	0.5	0	0	0	2014-11-18 12:49:48	18	12
2	1	day18-img1.jpg	0.96	30.92	0.05	0	0.99	0.01	1	sad	0	0.37	0.04	0.03	0	0	0	2014-11-18 12:50:26	18	12
3	2	day18-img2.jpg	0.96	24.78	0.61	0	0.52	0	1	surprised	0	0.14	0	0	0.18	0.17	0	2014-11-18 12:50:41	18	12
4	3	day18-img3.jpg	0.99	30.41	0.94	0	0.66	0	1	calm	0.1	0.06	0	0	0	0.13	0	2014-11-18 12:50:42	18	12

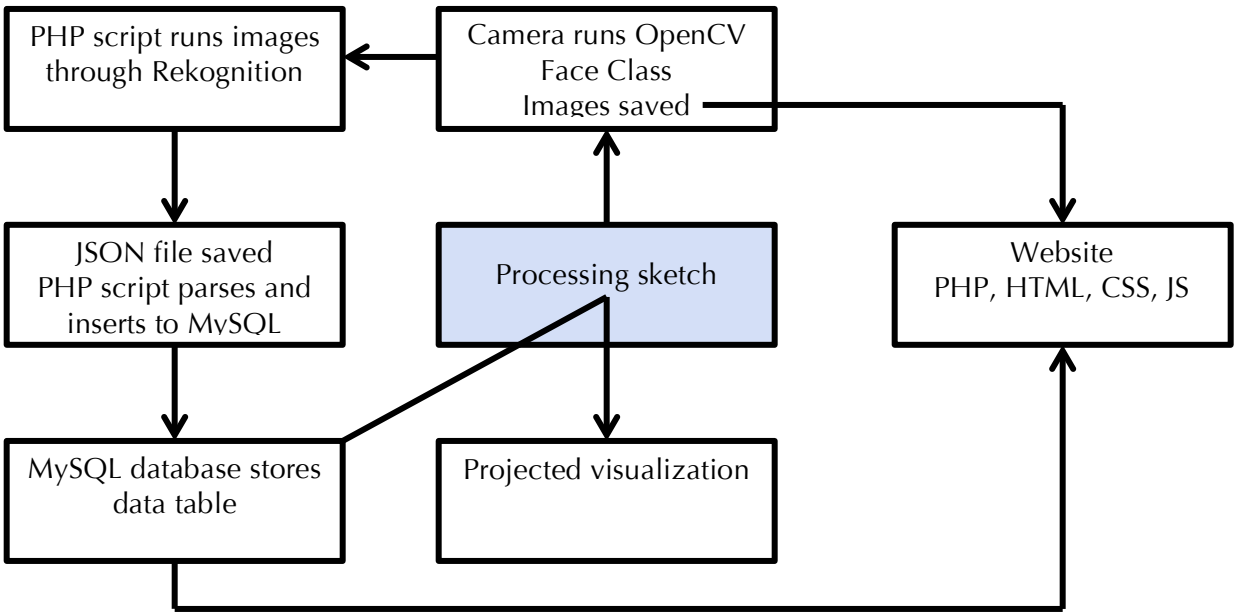
```
[{"face_detection":
          [
           "confidence":0.92,
           "eye_left":
            {
             "x":91.5,
             "y":72.7
            },
           "eye_right":
            {
             "x":113.2,
             "y":70.6
            },
           "nose":
            {
             "x":104.6,
             "y":79.6
            },
            ...
          ]
        }]
```


Processing sketch + visualization

The Processing sketch then loads the data from the MySQL table using HTTP query strings to save the data as variables for the visual. The datasets are saved into an array for each id. The visualization is written with the Processing language using basic visual structures. Each object in the visualization represents a face that was detected. The shape of the object is based on the gender – circles are females and squares are males. The color derives from the dominant emotion detected. The size is determined from the age. The speed of the object is determined by the smile rating. The objects animate in circular motion in rings that represent the days of the exhibition - December 1st being the innermost ring. The visualization will continue to grow each day until there are 12 rings at the closing reception on December 12th.

PHP + HTML5 + CSS3 + Canvas website

The supplementary website is built in PHP using HTML semantics and CSS3 animations and transitions. The background is a Canvas animation. It loads the data from the MySQL table and uses basic SQL operations like AVG, COUNT, and SUM in data section to display the collected in a traditional infographic. The infographic shows the data from the entire exhibit but can also break down by day to see how the statistics changed each day.



Inspiration

As a graphic designer and computer programmer, I am captivated by data and the way it can be interpreted. In my work, I use code to organize quantities of information, and design to present that data in a cohesive way. Technology has encouraged innovative methods that capture the infinite information surrounding us and computational techniques are providing new insights into large quantities of data. From my project, I'm hoping to gain insight into the relationship between the order in statistical data and the variability in human expression.

My recent research into face detection from the previous semester special topics course in Web Programming specifically inspired me to explore this emerging technology and use face detection as the data source for my visualization. Face detection is creating many new possibilities in technology; it is assisting in security applications, identifying people to tag in Facebook photos, and helping people with Prosopagnosia (face blindness) identify people with face recognizing glass technology.

The emotion detection component adds an entire new level of possibilities as well. Although it is an estimated calculation by the API, the emotion data is collected without any external interference. In this small experiment, I will be able to analyze how people generally feel while observing the Senior Exhibition and how it fluctuates from day to day, hour to hour. This is a more beneficial strategy than any survey where many factors play a part in the end result and allow for skewed data. In my project, the images are captured before the viewer even has time to realize what my project is about.

Technology

The main technology used in this project was Processing (processing.org). Processing is a programming language and development environment that was created by software Casey Reas and Ben Fry to encourage software in visual arts. Its focus is on visual, interactive media insight. I was introduced to Processing as a freshman at NMU and have been experimenting with it over the years. I utilized the OpenCV face detection library in Processing after learning about it during my CS Web Programming class in Winter 2014 (opencv.org). From my research into face detection and recognition for this project I found the Rekoginition API and was very interested in the emotion detection component (rekognition.com). Through my testing I found decent accuracy for emotion, smile, and gender detection. The age detection seems to be the most inaccurate one in my opinion, but I was impressed with this API overall. The Rekogition parser returns a JSON Object that is saved but also parsed to insert the data into a MySQL database. PHP, Chart.js, HTML5, and CSS3 are also used to create my website. The website runs on the localhost using MAMP, and I learned about technical details like displaying two monitors at full screen and using Google Chrome in Kiosk mode for the exhibition.

Features

While working through the semester on this project many factors changed. I executed as many of the proposed features as possible. The focus changed a lot when I started researching emotion detection. In the end, I spent more time on learning the Rekognition API and improving the website experience for within the gallery instead of working with NodeJS or a Web Application.

Proposed Features	*Executed Features*
OpenCV face detection application	OpenCV face detection application
Image database	Image database
Processing JavaScript application	Processing JavaScript application
Data visualization with 4 unique data points	Data visualization with 6 unique data points
Live-time updating	Live-time updating
NodeJS server	Rekognition API

Supplementary website on process utilizing PHP and MySQL database	Supplementary website on process utilizing PHP, MySQL database, Chart.js, Canvas animation, CSS3
Web application	

Reflection

From working on this project, I learned how to integrate multiple languages, systems, and ideologies into one cohesive piece. From the planning, to coding, to installing – I worked to coordinate and troubleshoot. I had little familiarity with Processing, OpenCV, and using APIs before this project and I have improved my knowledge of PHP, MySQL, Canvas, and JSON greatly. Upon the completion of this project, I have better understanding of libraries in Processing and API SDKs, greater practice with SQL functions and parsing JSON, and knowledge about passing variables between Processing and PHP. I also improved my web knowledge and practice of HTML and CSS by using data attributes, Canvas, CSS animations/transforms, and CSS before and after tags for the first time.

Beyond the technology aspect of this project, I learned a lot about preparing for an exhibition. I spent a lot of time testing my piece, editing, and reorganizing all aspects of my project. I worked between the CS and Art departments to coordinate the details for my piece and gained practical real life experience for executing an interactive data art piece.

*I would like to thank the Computer Science Department for providing my with the necessary equipment for me to execute this piece for my Senior Exhibition.