Matthew Weiler - mweiler2

Josh Cordell - jcordell

Connor Barrett - cbarrett3

Maya Subramanian - msubramania6

10/31/2018

**Midterm Report**

**Our Problem & It’s Importance:**

Personal image is a very important means of expressing oneself in modern society. People enjoy making changes to their appearance in order to feel comfortable and unique. However, it can be daunting to make changes to one’s appearance without first knowing if the alteration will make one look and feel good about oneself.

With our application, we intend to assist people in making changes to their image by providing them with an accurate and realistic picture of what they would look like with various changes to their facial features. Facial enhancements have been trending among millennials, and we feel that an application such as this would be greatly appreciated now more than ever. Our application can create more confidence in users interested in making changes to their image, in addition to potentially saving them from making a bad decision or wasting money.

**Current State-of-the-Art and its Issues [4]:**

There is currently an iOS mobile application that tackles a similar problem that we intend to solve, called “FaceApp-AI Face Editor.” While FaceApp has been praised for having many realistic face filters, the algorithm behind these filters has met criticism due to various issues.

FaceApp uses neural networks to analyze thousands of images to identify various facial traits. However, the app has been criticized for its original “Hot” feature, where it attempted to make people appear more attractive but in reality white-washed their features. This is due to the fact that their database of images featured primarily Caucasian people. Attractiveness is also a very subjective trait, so this feature in an app would be nearly impossible to implement without inaccuracy and a level of inappropriateness. We do not intend to implement a feature such as this one.

This application has also been criticized for some features that appear very fake, such as the filter on eyes, where many user’s eyes have appeared glassy and unrealistic. We intend to implement subtler changes to our user’s images in sensitive features such as eye color in order to have these filters look as realistic as possible.

**Our Solution:**

**Why We Chose Python/Dlib vs. MATLAB [3]:**

Originally, we had planned to use solely MATLAB for our project. After considering multiple routes to accomplish our goal with this project, we have decided that we will be using dlib in tandem with Python as our main programming tool. We will explain this technology in detail and why we are using this in the following paragraphs.

Dlib is a software library that contains deep learning-based algorithms for face localization. We plan to use this technology as the baseline of our application and will create our own facial filters based on dlib’s detection capabilities.

Our decision to use dlib for facial detection was also partly because our group prefers to use Python, as we are all more familiar with it than we are with MATLAB.  Our familiarity will render us more efficient in finding ways to optimize code to improve runtime. Second, Python provides a powerful environment for scientific computing and machine learning, along with the ability to use a web server, Flask, to do on-request image processing if we choose to host our application on the web. We also like the capabilities that we have with Python’s user interface toolkit, PyQt in the case that we do not choose to build a web app to interact with users.

**Post-Proposal Changes:**

As we continue to explore the capabilities of dlib to make changes to one’s image, we are getting a good idea of what features are realistic to implement and what features are not. The software does not appear to recognize any hair except for eyebrows on images, so we may need to remove any features that have to do with altering hairstyle and hair color. However, we will try to implement beard and mustache features as we can calculate the area between the nose and lips of a person and the area between the lips and jaw of a person using dlib’s facial detection software and the x-y coordinates applied to each image. Dlib’s facial detection software also does not recognize a person’s ears. However, we will attempt to add ear piercings by using the jaw detection as a threshold to find the ears in an image.

We will have to utilize the x-y coordinates on each image to experiment with more complicated features. Other than this potential challenge, we feel like we are on pace to implement the following features:

* Lip color
* Eye color
* Adding on accessories (i.e. glasses, masks, etc.)
* Facial piercings
* Eyebrow color
* Tattoos

“Challenge”/potential features:

* Beard/mustache alterations
* Ear piercings

**Evaluating Performance:**

**Recognizing Faces [2]:**

Detecting faces in an image is a crucial piece of this project. As we considered how we were going to go about detecting faces in images, we knew that there were many great resources out there for us to explore. As mentioned previously, there exists a Python library called “dlib” that we are using to detect facial landmarks in images. Before using this library, we did research to understand how it is working, and if it is indeed going to be what we need. We have learned that dlib will help us recognize the (x,y) coordinates which are associated with specific facial landmarks. Furthermore, this facial landmark detector is implemented within ‘dlib’ by training a shape detector, which translates into a facial feature detector, on a labeled dataset. In the current state of our project, we are able to recognize mouths, right eyebrows, left eyebrows, right eyes, left eyes, noses, and the lower jawline in faces.

As of now, we have successfully been able to get the software setup to recognize faces from a variety of angles. It recognizes the mouth, eyebrows, jawline, eyes, and nose. A picture of the application recognizing our faces can be found in **Figure A [1]:**

**Figure A**



**How We Will Add New Features:**

With the capability of detecting the (x,y) points of the facial features that we need, we will be able to add the features that we want to. The process will include looping over the subset of points for the facial landmark that we intend to change, and drawing our specific feature in this area. For facial regions that we would like to alter that are not precisely recognized by our facial detector, we will have to redefine the subset of points that we care about before drawing over the region to make the change.

**Current Timeline of Progress:**

|  |  |  |  |
| --- | --- | --- | --- |
| **Initial Planned Date** | **Current Planned Date** | **Task** | **Status** |
| 10/16/2018 | 10/16/2018 | Photo uploading / facial checking feature | Completed |
| 10/31/2018  (Midterm report due) | 10/31/2018 | Add 2-3 features that can be edited:  Lip, eye, and eyebrow color | In Progress |
| 11/15/2018 | 11/15/2018 | Add 2-3 more features that can be edited:  Adding accessories to the face  Work on GUI | In Progress |
| 12/03/2018 (Presentations Start) | 12/03/2018 | Finish Presentation/add 2-3 more features:  Facial piercings and tattoos  Potentially beard and mustache alterations and ear piercings  Finalize GUI | Not Started |
| 12/12/2018  (Website due) | 12/12/2018 | Finish Website/GUI for application | In Progress |

**Sources:**

[1] <http://dlib.net/face_detector.py.html>

[2] <https://www.pyimagesearch.com/2017/04/10/detect-eyes-nose-lips-jaw-dlib-opencv-python/>

[3] <https://www.learnopencv.com/opencv-c-vs-python-vs-matlab-for-computer-vision/>

[4] <https://gizmodo.com/why-faceapps-selfie-filters-work-so-well-and-why-they-d-1795012961>