Grace Street

mstreet@rollins.edu

Sierra Sarver

ssarver@rollins.edu

V. Summit

CMS 495

11/ 28/17

# Project 3 Write Up

The objective of this project is to use facial detection to apply snap chat filters to live videos and images. We wanted to include many different options for filters, some easy as posting pictures on top of features, but others, like Face Swap, are more difficult. The idea is that we want the filters to move dynamically with the subject.

To achieve this, we implemented and downloaded the OpenCV functions for Haar Cascades with which we drew rectangles to follow our features. We used the cascades for the default face and the eyes; We attempted to use the one for smiles, but it made the set up too sensitive. For the nose and mouth, we had to manually draw rectangles at the estimated points on a normal person's face (Image A). The nose is almost at the center of the face and the mouth is about 2/3 of the way down the face.

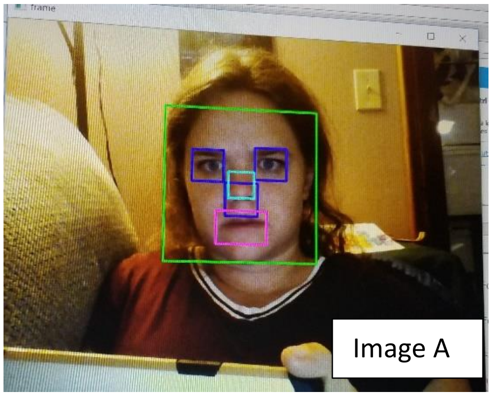


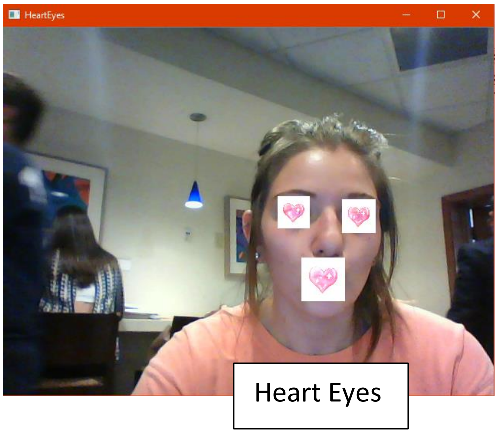
Image A

OpenCV also has functions that allow for video capture. It allows the user to open the webcam or any camera, and any adjustment we perform in the loop that keeps the camera open, will be applied to the video.

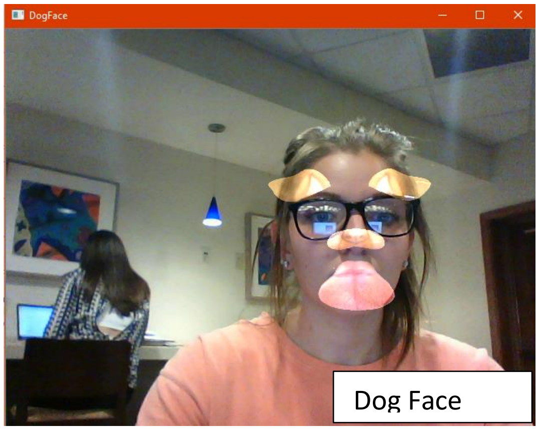
There were some very simple filters to create, like the dog face filter which was a matter of putting the images of the dog's features on the person's face. (Dog Face). We also have a moustache(Moustache) and hat filter(Hat), which act the same. For the heart eyes filter, which targets the eyes and puts hearts over them. However, we could not get the white background to disappear. (Heart Eyes) The

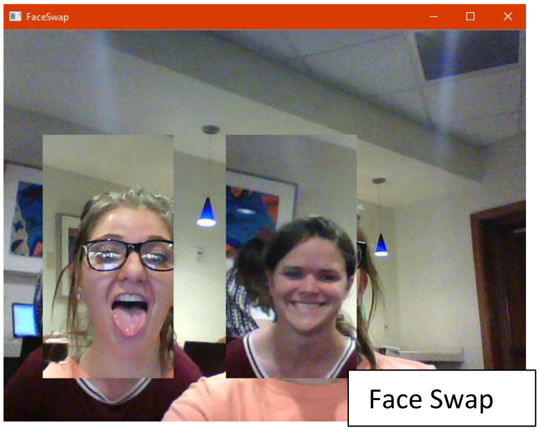
However, there were more complicated filters like face swap, which captures video from a selected region and swaps it with another region. (Face Swap) We were unable to find a way to target the pixels to face swap with both faces using feature detection, so the next best thing was to draw static rectangles in which to swap the videos in.

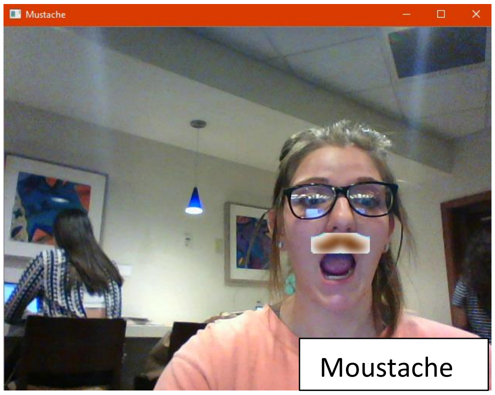
The fact that we were working with live video made it difficult to work with images, as it often interfered with the appearance, such as with the heart eyes which maintained their background despite being transparent.

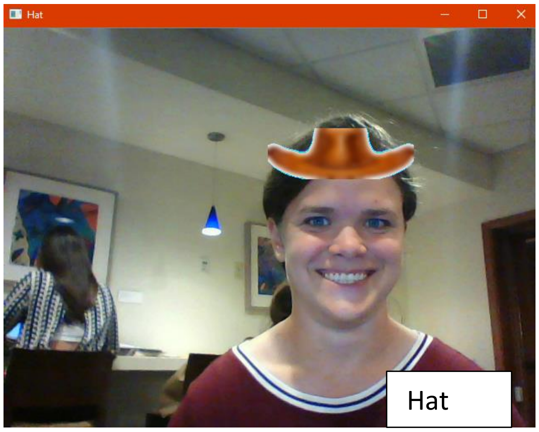


Heart Eyes









## Sources

- https://docs.opencv.org/3.0-beta/doc/py_tutorials/py_gui/py_video_display/py_video_display.html#display-
- https://realpython.com/blog/python/`aface-detection-in-python-using-a-webcam/
- https://www.youtube.com/watch?v=88HdqNDQsEk&t=586s
- https://github.com/kunalgupta777/OpenCV-Face-Filters/blob/master/filters.py
- https://github.com/opencv/opencv/tree/master/data/haarcascades