Face Detection, Extraction, and Swapping on Mobile Devices

Andrew Chou
Department of Computer Science
Stanford University
Stanford, CA 94305

Yang Hong
Department of Electrical Engineering
Stanford University
Stanford, CA 94305

***Abstract*—The Face Swap algorithm uses Viola-Jones face detection, Active Shape Model fitting, and Laplacian Pyramids among other methods to locate faces in an image and swap them while maintaining a natural and realistic look. The main application of this this software is preventing unwanted online identification of people in photos, such as those on Google and Bing Maps.**

I. Introduction

In recent years the internet has given rise to a number of public forums in which it has become very easy to connect collect personal information about complete strangers. Such forums have raised many privacy concerns. The Face Swap algorithm presented in this paper aims to address privacy in applications such as Google Maps street view. Past implementations have blurred faces or tried to remove people entirely. Both of these solutions may look somewhat unnatural since there will be busy intersections with no people in them, or only people with blurred faces. We aim to allow faces to be swapped out for other random faces, or a pre-approved white-list of faces, thus protecting the privacy of innocent bystanders on the street while maintaining the natural quality of the original photos. Our Android implementation aims to make this technology available on mobile devices for ease of testing and for general enjoyment.

The Face Swap algorithm consists of five main steps: Viola-Jones face detection using Haar-like features [1], Active Shape Model fitting [4], face rotation, skin-tone matching, and smoothing using Laplacian Pyramids [2]. The Viola-Jones face detection uses an OpenCV library [5] to detect faces from a frontal view. This type of classifier sacrifices accuracy for speed, so it sometimes misses faces or classifies inanimate objects as faces. This trade-off is worthwhile for the Android application, but is probably not acceptable in a production quality application such as Google Maps.

II. Related Work

A. Viola-Jones Face Detector

Viola-Jones face detection uses Haar-like features for image classification. These features are computed using very simple masks over small regions of a gray-scale image. The crude quality of the features allows for an extremely fast classifier for detecting faces, yet also a mildly inaccurate one. [1]

B. Active Shape Models

Active Shape Models are used to quickly find the outlines of distinctive shapes such as faces or hands. It is a supervised model that uses labelled training data to determine what strong edges or other features various model key-points are usually close to an image. When a trained model is initially overlaid on a test image the algorithm searches in the directions normal to the the currently approximated face contour to find a better fit for the key-points. This process continues for a fixed number of iterations or until convergence. [3]

C. Laplacian Pyramids

The main technique used for stitching the swapped face smoothly into the original head involves Laplacian pyramids. The general algorithm is as follows: If A and B are square matrices of the two selected faces to swap, with A being face 1 (the face to mask and swap in) and B being face 2 (the face to replace), let M be a binary square matrix, mask 1, corresponding to the detected face 1 output from applying the ASM. With A, B, and M having the same dimensions, the Gaussian pyramids (GM) are constructed for M and the Laplacian (of Gaussian) pyramids are constructed for A and B (LA and LB respectively). For GM, the pyramid at each level is constructed by Gaussian filtering followed by 2:1 sub-sampling of the previous levels pyramid. For LA and LB, the LoG pyramid at each level is constructed by first resizing each pyramid to have dimensions 2N-1 (where N is the size of the next smaller pyramid), then taking the difference between the pyramid at that level constructed via blurring and sub-sampling starting from the original (largest) and the pyramid at that level constructed via sharpening and up-sampling starting from the smallest level. The general formula for masking at each level is then (LA)(GM) + (LB)(1-GM). [2]

III. The Face Swap Algorithm

A. Android Phone

The Face Swap Android application has a long, but straightforward data flow. After a picture is taken with the camera on the Android phone the phone uploads the image to a server and waits while the server processes the image.

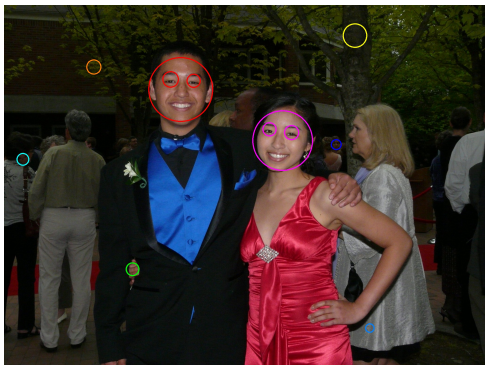

Fig. 1. Image with three real faces and five spurious faces.

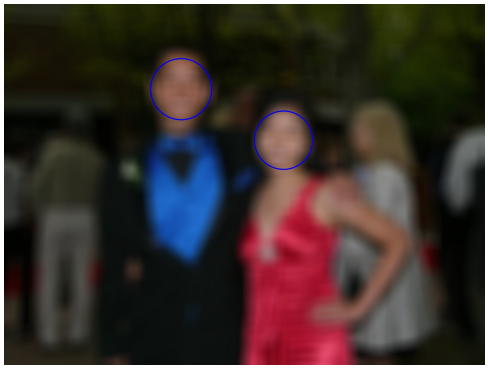

Fig. 2. Blurred image has two real faces and no spurious faces.

B. Face and Eye Detection

Upon receiving the image the server runs Viola-Jones face detection, which outputs the locations of faces and eyes in the image. The input is converted to gray-scale before being sent as input to the face detector along with trained face and eye classifiers. Since there is no user interface to select which faces to choose in the image, the face detection algorithm must automatically select faces that have the potential to be easily swapped. If fewer than two faces are detected then the server outputs an error. If exactly two faces are detected then those faces are outputted (the output is simply a bounding box roughly around the head) along with any eyes that were found on the faces. However, if there are an excess of faces to choose from (see figure 1) then extra precautions are taken to make sure that the selected faces are legitimate faces. First eye detection, also using Viola-Jones and Haar-like features, is run on each face. If at least two faces have at least one eye detected then it is assumed that these are not spurious faces and are in fact inanimate objects. Therefore the faces with the most eyes are outputted.

If one or none of the faces have eyes that are detected then further testing is done to make sure that none of the faces are spurious. At this point the image is low pass filtered by a square kernel of ones (normalized by the area of the kernel) whose side length is one fourth the length of the detected face. This low pass filtering is done based on the assumption that spurious faces are due to random background noise with low magnitude and will not be detected again after low pass filtering (see figure 2). Face detection is then run again and faces that are found in both the original image and the low pass filtered image are written to the output. If fewer than two faces were found in both images then all of the other faces are output as well.

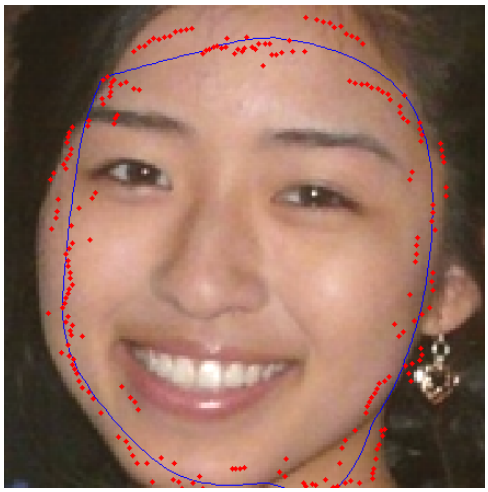

Fig. 3. ASM in the process of fitting a face.

C. Face Outlining

Next the Active Shape Model (ASM) is fit to the faces in the image. The ASM was trained with 60 labelled training images. Thirty of these were male faces and thirty were female, representing a diverse array of ethnic backgrounds. All training is done beforehand so the ASM fitting algorithm only needs to focus on the faces found during Viola-Jones fact detection. The face detection software found at least two faces and by default the largest two faces are chosen to be passed into the ASM algorithm. The initial location of the model overlaid on each face image is simply the center of the image, and the faces are scaled to the size of the trained model (the model itself was trained on 256x256 pixel images). The ASM fitting algorithm then runs for 15 iterations on each face; five iterations at 2:1 sub-sampling to help the algorithm converge more quickly, then ten iterations at the full size of the model (see figure 3). The algorithm search searches 15 pixels (equivalent to 30 pixels in the original image) on either side of the current outline during each iteration on the sub-sampled image, then searches 10 pixels on either side of the outline during each iteration on the full image. When the ASM algorithm has finished it outputs a binary mask of each face.

D. Face Transformation and Centering

The masks in the ASM output are used to compute the centroid of each face by computing the centroid of the masked region. Then the rotation of each face mask is computed using the locations of the eyes detected by the Viola-Jones classifier. This is done using the arc-tangent of the slope from one eye to the other. If there were fewer than two eyes detected for a face then the rotation of the ASM model fitted to the face is used as

Fig. 4. Left: Direct swapping. Right: Gaussian blurring.

Fig. 5. Gaussian blurring of masked boundary with two different kernels. Left: size=10 and sigma=4. Right: size=10 and sigma=2.

Fig. 6. From left to right: mask of face, convex hull of face, convex boundary of face.

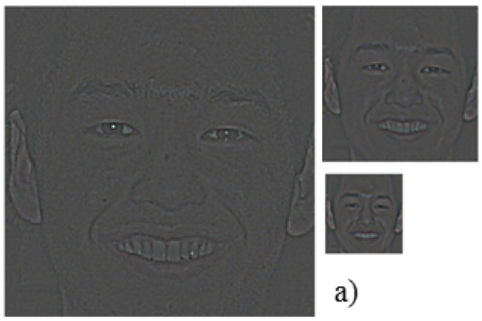

Fig. 7. Laplacian Pyramid for face 2.

a fallback. Finally, the average intensity values are computed for each RGB color and for each face, and the intensity values for each channel and each face are scaled to match the old values for the other face. When these three steps are done the faces are ready to be masked out, rotated, and moved to the centroid of the other face.

E. Smoothing using Laplacian Pyramids

The last stage of the algorithm uses Laplacian Pyramids to swap the faces using the previously calculated rotation angle and centroid inputs while smoothing around the mask boundaries where the each face was swapped in. Four pyramid levels are used (four was emperically determined to be the best number of levels). The level 1 pyramid is the size of the image itself, and each successive level is Gaussian filtered and then sub-sampled with a 2:1 ratio. The output pyramids are constructed by masking each level individually and then

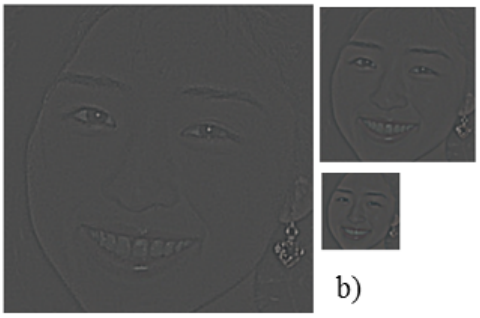

Fig. 8. Laplacian Pyramid for face 1.

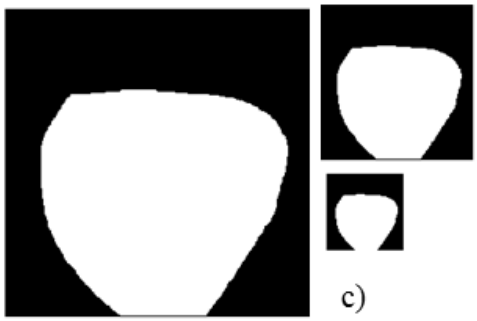

Fig. 9. Gaussian Pyramid for mask 1.

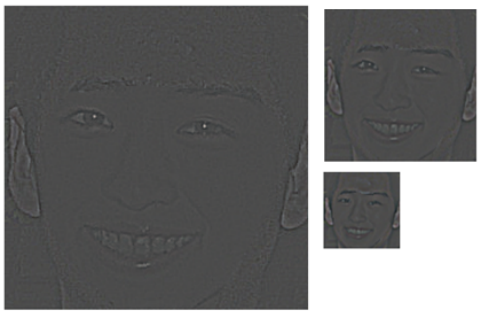

Fig. 10. Laplacian Pyramid after Swapping

Fig. 11. Final Collapsed Pyramid

TABLE I
HIGHER EYE DETECTION RATES USING MULTIPLE CLASSIFIERS

Image #	# Faces detected	# Faces with two eyes using one Viola-Jones eye detector	# Faces with two eyes using five Viola-Jones eye detectors
1	3	3	3
2	3	2	3
3	2	2	2
4	2	0	0
5	6	0	2
6	1	0	1
7	3	0	2
8	8	2	8
9	2	0	2
10	2	2	2
11	2	0	2
12	3	1	2
13	3	1	3
14	2	0	1
15	2	2	2
16	9	0	0
17	2	1	2
18	2	0	0
total	55	16	37

Fig. 12. Image No. 8 using just one eye detector.

collapsing the levels into one base pyramid. Then we blur the edge of the pyramid image into the whole image using a simple low pass filter (a normalized column or row of ones depending upon which edge is being blurred). This final image is then sent back to the Android phone to be displayed.

IV. EXPERIMENTAL RESULTS

A. Image Quality

The algorithm was tested on many different types of people and faces (see figures 14-17 for a few examples). These included people with different hairstyles, skin-tones, and in different lighting conditions. It doesn't make too much sense to statistically evaluate the results of the algorithm since the realism of a face swap is mostly subjective. However, there are a few things that clearly went well and went wrong in our algorithm.

In general, the face detection and ASM algorithms are most robust against different lighting conditions (except for sharp shadows on the face from external sources), and the skin-tone matching does well matching skin-tones from

Fig. 13. Image No. 8 using five eye detectors.

different lighting conditions. Difficulties include determining the rotation of faces covered by long hair when one or no eyes are detected, and matching very different skin-tones. Thus results have ranged from near perfect (observers are unable to tell that the face has been swapped even when informed that a face has been swapped) to amateur (the face rotation is completely wrong) to reasonably good (the face looks fine but it is clear that some editing has been done). The biggest problem is consistently determining the degree of rotation of faces. This can be done using a more reliable eye detector. The solution to this problem is to use a more robust eye detector since originally just 16 out of 55 (29%) of the faces detected had exactly two eyes detected in them. The current Face Swap implementation now uses five duplicate Viola-Jones eye detectors based on different training sets to help find more eyes. Duplicate eyes are then removed and there are much better overall detection rates: 37 out of 55 faces (67%) had exactly two eyes detected in them (see table 1 and figures 12-13).

B. Face Stitching

To stitch and smooth the swapped faces, we first tried a direct method of detecting the convex boundary of the swapped faces binary mask (computed from the convex hull of the mask) and applying a Gaussian filter along this boundary (see figure 4 and 5). However, in practice this approach yielded a blurred boundary artifact as a result of the sliding convolution kernel (see figure 6), which led us to use Laplacian pyramids. We wrote and tested two implementations of Laplacian pyramids. First, we constructed pyramids from the face images themselves (bounding box of faces being outputs from OpenCVs face detection) such that Laplacian pyramids LA and LB are the two faces to be swapped, which yielded smooth results but had a slight artifact around the square bounding box of the face image when fit back into the original full image. Then we constructed pyramids such that LA was the face image to be swapped out and LB was the full image, which yielded varying results, although in general the swapped-in face tended to be more blurred due to the dramatically differing sizes of LA and LB. Finally, we chose the first implementation of Laplacian pyramids and also

Fig. 14. Before

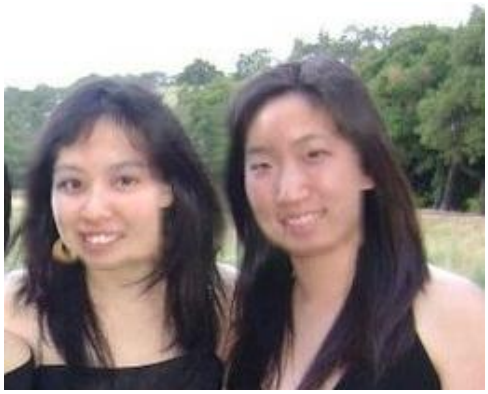

Fig. 15. After

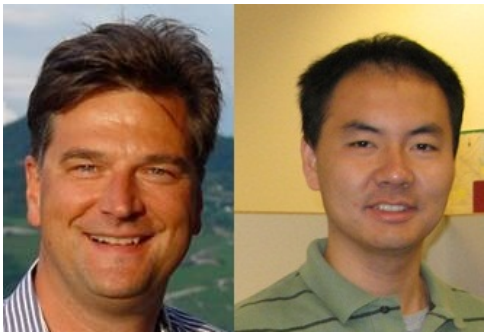

Fig. 16. Before

Fig. 17. After

incorporated the calculations of rotation, centroid matching, and skin tone matching into the masking in at every pyramid level (see figures 7 - 11). This was very effective in smoothing the swapped in face to the masked out face. After resizing and fitting the swapped face image back into the original full image, we added post-processing to blur the bounding box of the face image and smooth it into the final image.

C. Running Time

The Face Swap algorithm takes about 15 seconds to run on a typical 648x484 Android Image. The Viola-Jones face and eye detection takes 2-4 seconds, depending upon how many faces and eyes are initially found. The ASM fitting takes approximately 10 seconds, and the post-processing and smoothing take around 3 seconds total. This time could easily be cut down since the bulk of the time (ASM fitting and post-processing) is spent running Matlab code, which is much slower than C++ code, so language conversion would be a very easy optimization. Furthermore, there are potentially faster methods than ASM for outlining faces, so the total time could be cut down even further.

V. Future Work

A. Face Orientation and Rotation

The initial project proposal sought to deal with minor face rotations outside the plane of image. However, it proved difficult enough to robustly detect rotations of the face in the plane of image. One challenge is consistently detecting the eyes, which is the easiest way to determine the rotation of the face. The orientation of the Active Shape Models are unreliable because of the rough rotational symmetry of faces, which is further complicated by the presence of hair. Future work could possibly use Active Appearance Models to fit a more complete model of the face, not just an outline.

B. Skin-tone Matching

The skin-tone matching used in the current version of the Face Swap algorithm is very robust against lighting changes across the entire face. This is because it matches the average channel values for each of the RGB channels. However, vast differences in skin-tone are problematic under any lighting conditions. The main reason is that all pixels in the face are treated equally when computing the mean color intensities. Furthermore, all pixels are uniformly lightened or darkened. This causes teeth and eye pixels to be transformed in the same proportion that the skin pixels are transformed. Finally, any spurious pixels, such as glasses or hair, covering the face can cause the mean color intensities for the face to change dramatically, thus matching the final skin-tones to the wrong color.

C. Smoothing

Currently, there is a still a slight boundary artifact visible in some output images due to the Laplacian pyramid blending being performed on a modified square subset of each face.

Given more time, alternative smoothing methods of stitching the swapped faces together could also be tested and compared to Laplacian pyramid blending. Possible competitive techniques to consider are seamless cloning and/or Gradient Domain Image Stitching (GIST1) algorithms. In particular, experimenting with processing in the gradient domain can lead to improved overall robustness. For example, in pre-processing (before the image is passed to face detection) projection tensors could be used for shadow and variable lighting removal from a face and making local illumination changes such as toning down intensity on a noticeably reflective face region. In post-processing (when faces are ready to be swapped or after), stitching can be performed by minimizing gradient differences in overlapping boundary regions from one face to another.

D. Database

If the Face Swap algorithm were to be used to protect privacy on the internet it would be best to use a separate database of pre-approved faces to swap in so that no one with privacy concerns would have their face swapped onto another body. Each face to be swapped out could first be queried against the database using face recognition software. Then the closest match in the database could be used as a replacement. This approach would avoid tricky issues relating to matching vastly different skin tones, and it would be easier to make the faces look good in general.

ACKNOWLEDGMENT

The authors would like to thank Professor Bernd Girod, teaching assistants David Chen and Derek Pang, and their mentor Huizhong Chen for all the advice, instruction, and inspiration they provided throughout the Digital Image Processing class.

Andrew Chou worked on the Viola-Jones fact detection using OpenCV. He also did skin-tone matching, face rotation, and set up the server. Yang Hong worked on setting up the Android application and smoothing using Laplacian Pyramids. They jointly worked on using the Active Shape Models to outline the faces and swap the faces to the opposite centroid.

REFERENCES

[1] P. Viola and M. Jones, *Rapid Object Detection using a Boosted Cascade of Simple Features*, Conference on Computer Vision and Pattern Recognition (2001).

[2] P. Burt and E.H. Adelson, *The Laplacian Pyramid as a Compact Image Code*, IEEE Transactions on Communications, Vol. Com-31, No. 4 (1983).

[3] T.F. Cootes, C.J. Taylor, D.H. Cooper, and J. Graham, *Active shape models - their training and application*, Computer Vision and Image Understanding (61): 3859 (1995).

[4] D. Kroon, *Active Shape Models*, Matlab Central File Exchange (22 Mar. 2010).

[5] G. Bradski, *The OpenCV Library*, Dr. Dobb's Journal of Software Tools (2000).