Facial Recognition

Project Members: *Haider Syed & Travis Peters*
Project Advisor: *Lorenzo Torresani*
Dartmouth College Computer Science – Computer Vision (CS 183)
November 21st, 2013

1. Introduction

1.1 Background

This project was inspired by work being done in the mHealth Security & Privacy lab to address authentication with a mobile medical device. Some of the members of the lab were working with students from another school to construct a low-cost, mobile device known as a Spirometer with a camera installed on it. The intention would then be to take a picture at the time of use and verify that that person is a known user for the device; properly identifying a person would also record any session data into their medical record.

This background information helps to provide insight into the motivation for the problem but it should be clear that our project aims to address *only* the problem of face recognition from an image.

Details about the aforementioned device have yet to be finalized; therefore our project doesn't have a representative dataset to validate our methods. In light of this, we will not try to consider hypothetical scenarios from a potential dataset and we will focus on addressing four main problems in the realm of facial recognition with the dataset discussed in section 3 of this paper – the primary goal of our methods address four well-known problems that hinder the performance of face recognition algorithms:

1. Complex backgrounds
2. Varying subject pose
3. Varying subject scale
4. Varying illumination conditions

In this paper we present the current state of our implementation and demonstrate how, at this point, we have begun to address the issues mentioned above. We also propose a vision of future modifications that could improve the performance of our implementation,

and a justification for why we believe these modifications will improve the accuracy of our facial recognition system given the tests we have performed thus far.

1.2 Assumptions

The following assumptions were made in order to control the scope of our project and keep our efforts as focused on the problem of recognition in the context of a system that would be used by people using the mobile medical device indicated above:

- Each subject has 4 representative images.
- The input image represents someone who is in the database.
- We only account for pose variations up to 60 degrees (this is a limitation of the SIFT algorithm which our recognition system utilizes).
- Images do not contain multiple faces that could be identified as a primary face (in the next section we discuss how we segment the face from its background; our implementation relies on the fact that the largest skin region in an image is the face of the primary subject).
- Images are minimally occluded or not occluded at all. While the final system we hope to produce will need to be robust to the presence of occlusions, we did not address this issue in the scope of this project.

2. Methodology

In this section we present our current implementation. We approach the problem of face recognition from an image by breaking it down into general sub-problems:

(1) Identifying the face and segmenting it from the background, (2) extracting feature descriptors, and, (3) using a nearest-neighbor classifier to perform identification.

2.1 Identifying Skin/Non-Skin Regions

The first problem we aim to address is the issue of having the face we are trying to recognize amongst a complex background. If we simply try to extract feature descriptors from an image with a complex background, there is a high likelihood of picking up features in the background, which are obviously not representative of the subject we are trying to recognize. So, we've chosen to utilize Gaussian Models to perform skin detection based on research that suggests that skin colors (pixels) cluster in a specific part of the color space even for people of different races. [7] Using this information, it is possible to compute a probability that a pixel in a given image is a skin or non-skin pixel. Thus, over the course of this project, we have implemented two different Gaussian Models that can be trained to determine if a given pixel in an input image is more likely to be a "skin pixel" or a "non-skin pixel."

To clarify, we have broken the training process into two parts: part (1) allows a user to open an image and interactively define contour regions that consist of skin, and part (2) opens the same image but allows the user to now define non-skin regions.

This is all the information needed to construct our Gaussian Models! Once the model is constructed, we can process an image pixel-by-pixel and determine the probability that some pixel is a skin/non-skin pixel – thresholding on a ratio of these probabilities (maximum-likelihood theorem) allows us to generate a binary map representing the detected skin pixels.

Currently, the training process is only conducted once, initially – and we have set up the test framework in such a way that the training can be done using multiple images in order to better describe the skin/non-skin color spaces, respectively.

After one of the aforementioned Gaussian Models is constructed & applied to detect skin/non-skin regions, the resulting image may "miss" some parts of the face due to the fact that skin color can be affected by illumination conditions and the skin/non-skin models are not trained to pick up the mouth and eyes as skin. Therefore, the identified face region will have "holes" and may not be contiguous. In order to properly identify the face we apply some post-processing steps, which aim to get a contiguous face region and to get rid of false matches in the background. To achieve this, we first erode the image with a small circular kernel to get rid of small regions of false matches in the image (noise that gets picked up as skin in the image background). We then fill holes in the image using morphological reconstruction so that the eyes and mouth in the face get filled in. Finally, we isolate the largest contiguous segmented region by area and construct a bounding box around the identified face region.

Brief descriptions of the two Gaussian Models alluded to above, are presented in sections 2.1.1 & 2.1.2.

2.1.1 Single Gaussian Model (SGM)

For the milestone – and for the purpose of creating a baseline to compare more sophisticated methods to in the future – we implemented a Single Gaussian Model in order to generate two separate representations: one for skin pixels and one for non-skin pixels. The probability distributions used in the Single Gaussian Model is computed as follows:

$$p(x \mid \mu, \Sigma) = \frac{1}{(2\pi)^{\frac{d}{2}} (\det(\Sigma))^{\frac{1}{2}}} \times \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu) \right)$$

where $\mu = \frac{1}{n}\sum_{k=1}^{n} x_k$ is the estimated mean vector, $\Sigma = \frac{1}{n}\sum_{k=1}^{n}(x_k - \mu)(x_k - \mu)^T$ is the estimated covariance matrix, n is the number of observations in the dataset, and x_k is the k^{th} observation in the dataset. [7]

2.1.2 Gaussian Mixture Models (GMM)

As has already been addressed, a Gaussian distribution can be used to model the probability that a pixel is skin and non-skin, respectively. However, using multiple Gaussians to model the probability distribution is able to better account for inter-class

variance and for multiple sources of variance (such as illumination in the image) [7]. The probability distributions used in the Gaussian Mixture Model are computed as follows:

$$p(x \mid \lambda) = \sum_{i=1}^{M} w_i \cdot g(x \mid \mu_i, \sum_i)$$

where

$$g(x \mid \mu_i, \Sigma_i) = \frac{1}{(2\pi)^{\frac{d}{2}} (\det(\Sigma_i))^{\frac{1}{2}}} \times \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1} (x-\mu_i)\right)$$

Note that $\lambda = \{w_i, \mu_i, \Sigma_i\}$ refers to the parameters for the mixture model and $i = 1, \ldots, M$, where *M* is the number of mixture components for the model. It is also worth noting that, given *T* data points (pixels), the model parameters are computed as follows:

$$w_i = \frac{1}{T} \sum_{t=1}^{T} \Pr(i \mid x_t, \lambda)$$

$$\mu_i = \sum_{t=1}^{T} \Pr(i \mid x_t, \lambda) \cdot x_t \Big/ \sum_{t=1}^{T} \Pr(i \mid x_t, \lambda)$$

$$\Sigma_i = \left(\sum_{t=1}^{T} \Pr(i \mid x_t, \lambda) \cdot x_t^2 \Big/ \sum_{t=1}^{T} \Pr(i \mid x_t, \lambda) - \mu_i^2 \right)$$

We implemented an Expectation-Maximization algorithm to estimate the parameters for the mixture model. Further details about the theoretical aspects of the Gaussian Mixture Model can be found in *Performance evaluation of single and multiple-Gaussian models for skin-color modeling* [7] or, more directly, in our code, which referenced the *Expectation Maximization and Mixture of Gaussians*, slides to guide our implementation. [9]

2.2 GrabCut

We setup GrabCut so that it would use the bounding box from our segmented SGM and GMM images as input. However, the version of GrabCut we obtained did not have the final matting step which smoothes out the jagged segmented image from GrabCut. Therefore, the image that is returned has jagged corners. Using this image, we found the refined bounding box from GrabCut to not be representative. Furthermore, our SGM and GMM would give results which were usually too restrictive… i.e. did not identify enough of the face. Therefore, using GrabCut would have meant that we would capture even less of the face so we did not use it in our final implementation.

Example of jagged edges recovered by using GrabCut

2.3 SIFT Feature Descriptor Extraction

Once we have an image that is segmented from the background, the next step is to generate feature descriptors for that subject that will be used to actually perform recognition - for this, we use SIFT feature descriptors which allows us to extract scale invariant features from an image. For our project, we generate about 100-300 sift descriptors for each image in our database. The threshold used by SIFT for rejecting keypoints is set in a way that we get about 100-300 descriptors for the image in general. For SGM and GMM, the threshold used was about 0.08 while the threshold used for sift on the original images was about 1.2.

We attempted to implement our own version of SIFT based on the description of the algorithm found in David Lowe's paper [3] and Andrea Vedaldi's notes [4], but have had problems with the correctness of computing the scale spaces. Therefore, we have utilized Vedaldi's implementation of SIFT that was programmed in *MATLAB* [4].

2.4 K-Nearest Neighbors (kNN) Classifier

Feature classification, though not really a Computer Vision problem, is critical for our overall facial recognition system. Upon recommendation from Lorenzo Torresani, we've decided to use the k-Nearest Neighbor classifier as it yields good results so long as there are meaningful feature descriptors. [2]

A brief description of our kNN classifier implementation follows:

Each subject in our database has multiple representative images (for now we assume 4 in an effort to keep the training set small). For each subject, we compute the feature descriptors for each of the images and concatenate those into a single matrix – we call this the feature set, $fset_i$, for subject i - this matrix now represents all of the features identified in all images for that subject; we do this for all subjects in our database and then construct a single matrix, $\boldsymbol{F}$, that contains each $fset_i$ where $1 \leq i \leq N$.

We then compare each feature in the *probe* image, *p*, to every feature in ***F***, by computing the *L2* norm between each of features. For each feature in the probe image, we find the best matching feature (the nearest neighbor) in ***F***. To identify the subject in the probe image we simply determine the feature set that was most representative of the probe image (i.e. the feature set containing the most similar features to those found in the probe image).

We have been informally calling this algorithm "*consolidated kNN" (ckNN)*.

3. Results

In this section we present information about the dataset that we are testing on, how we chose images from this dataset to actually use for testing, results from testing various components of our face recognition system, and our basic approach to testing the system as a whole.

3.1 Dataset

Currently we are testing the performance of our face recognition system/its various components on images from the *Faces in the Wild* database [5]. We chose images from *Faces in the Wild* because this dataset is great for testing the issues that we laid out in the introduction – presenting a variety of images with complex backgrounds, illumination conditions, and variable scale and pose of the subject.

Even though *Faces in the Wild* consists of more than 13,000 images in total, there are only about 1,680 subjects that have two or more representative images. Of that subset, we manually selected 100 subjects with the following constraints:

- The subject had 4 available images,
- There were minimal/no occlusions of the subject's face,
- The variation in the subjects pose was roughly within 60 degrees, and
- No other faces were within close proximity to the primary subject that we are trying to recognize.

3.2 A Comparison of Gaussian Models

Before we tested our face recognition system as a whole, we wished to understand how the two different Gaussian Models (Single & Mixture) compared in their ability to accurately detect skin regions in an image. To achieve this, we used the SGM and GMM to attempt to detect the largest skin region in *each* of the 400 images (100 subjects – 4 images each). The threshold values for determining if a given pixel in an image was a skin pixel/non-skin pixel were 3 and 2 for the SGM and GMM, respectively. Once the skin had been detected and the post-processing had been applied, a bounding box was constructed and used to "trim" the image (this was done so that subsequent use of the

image would have the complex background removed). At this stage in the process, we created a script to examine the dimensions of the resulting images – if any of the resulting images were less than 65 pixels in either dimension (size based on general observation across observed images), the SGM/GMM was deemed to have "failed."
For segmented images that were larger than 65 x 65, we manually verified the accuracy of the SGM/GMM in identifying the skin region for each of the remaining images.

Details about the training and our observations follow.

3.2.1 Training

The training process described in section 2.1 was carried out to train the skin/non-skin models for both the Single Gaussian Models and the Gaussian Mixture Models. Samples of training images and the skin/non-skin contour regions are shown below:

Figure 2: (Row1) Non-skin training and **(Row 2)** skin training, for the SGM and GMM skin models

The complete training process was based on 10 images.

3.2.2 Comparison Results

The results obtained using the SGM and GMM models are shown below (including the intermediate processing steps):

SGM


GMM
SGM
GMM
SGM
GMM
SGM
GMM
SGM

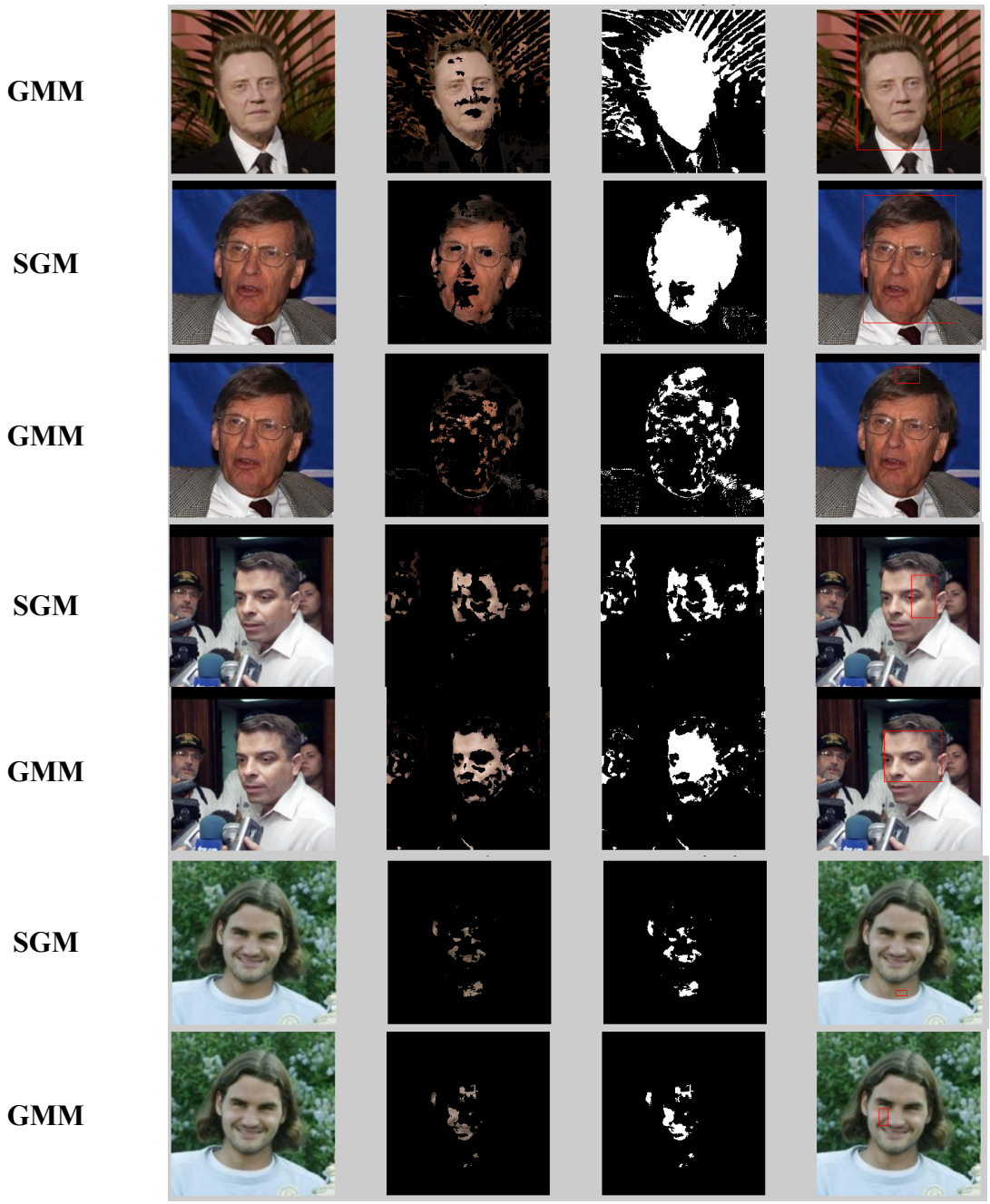
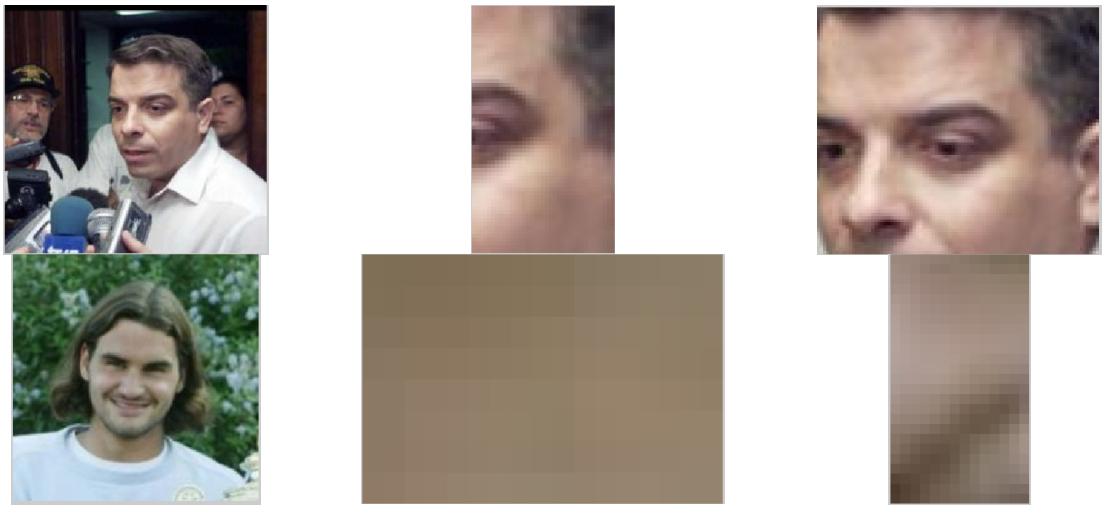


Figure 3: Examples of resulting images from SGM/GMM pre-processing. [Column 1] The original image; [Column 2] Skin recovered using the Gaussian Model (SGM or GMM); [Column 3] The result of eroding the image and filling in holes (using morphological reconstruction) [Column 4] The original image with a bounding box around the segmented face.

Original Image | **SGM** | **GMM**



The images above should show that at times the SGM outperforms the GMM, whereas in other cases, the GMM outperforms the SGM. However, as can be seen in Table 1, the GMM fails completely (i.e. recovers a patch of skin which is smaller than 65 pixels in either width or height) almost twice as much as the SGM.

Rates	Single Gaussian Model	Gaussian Mixture Model
Failures*	29/400 = 7.25%	57/400 = 14.25%
True-Positive	297/371 = 80.05%	274/343 = 79.88%
False-Positive	74/371 = 19.95 %	69/343 = 20.12%

Table 1: Performance results from qualitative comparison of Single Gaussian Model vs. Gaussian Mixture Model in skin region detection.

**Failure defined as an identified patch smaller than 65x65 in any dimension.*

Before examining the results too closely it should be clarified that a *False-Positive* was a resulting image that (1) clearly shows that the model in question was "confused" about what skin was in the image (i.e. the identified region consisted of a large portion of the image, including irrelevant background), or (2) clearly shows that the model in question did not identify the largest skin region (i.e. a small portion of the face). It should go without saying that a *True-Positive* was a resulting image where the face/largest skin region was clearly identified.

The results from comparing the performance of the two Gaussian Models suggest that their ability to identify regions of images containing skin/non-skin are more similar than suggested in the literature. In fact, under our testing scenario it appears that the Single Gaussian Model actually outperforms the Gaussian Mixture Model. We don't believe, however, that our tests actually ended up being entirely indicative of the true capabilities of these models; relevant literature suggests that the Single Gaussian Model is a less accurate model of the color space which actually defines the variance observed in skin color, thus, we believe the Gaussian Mixture Model really is the better choice for modeling this color space. The fact that the GMM performs worse than the SGM in our

results is likely a result of 2 factors: (1) the amount of training data used for the skin/non-skin Gaussian Mixture Models, (2) the number of mixture components (clusters) used for the GMM. Given more time we would have liked to experiment with these parameters more and continue comparing its performance to the Single Gaussian Model.

One thing that we found encouraging about this is that, even with fairly minimal training, the Gaussian Mixture Model does appear to perform very similarly to the Single Gaussian Model.

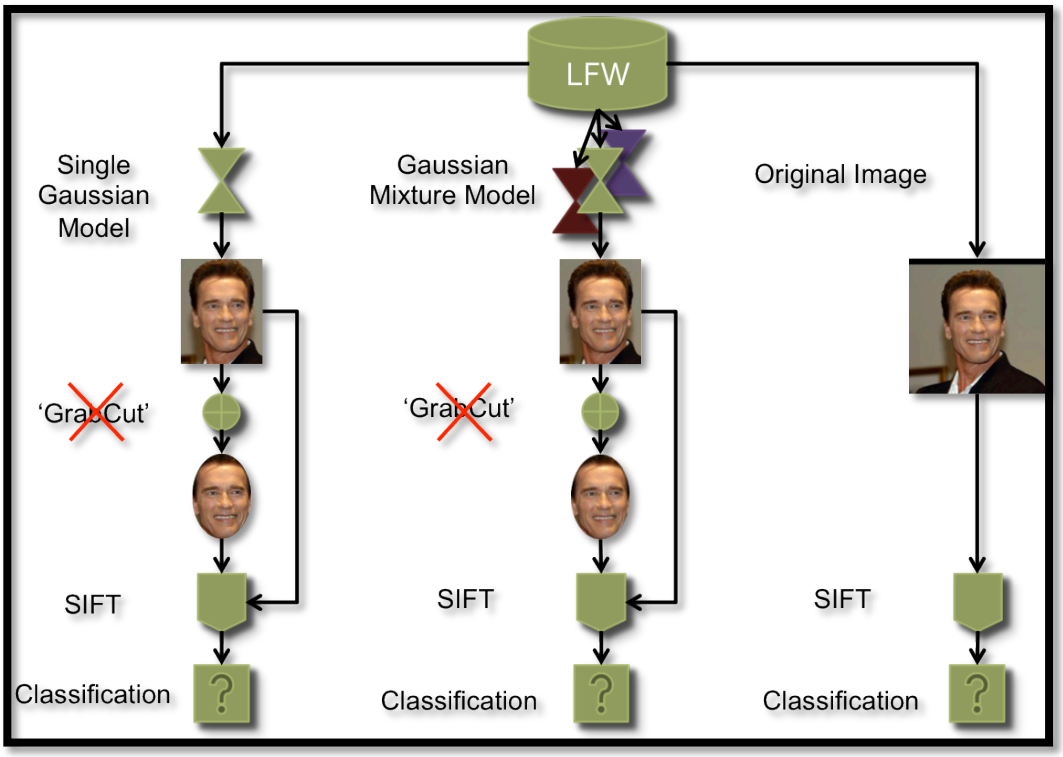


Figure 4: *Overview of end-to-end testing*

3.3 End-to-End Testing

Figure 4 provides a basic visual overview of our approach to testing the performance of our face recognition system as a whole. Due to the issues that we experienced with using *GrabCut* as an intermediate processing step for images put through the Single Gaussian Model/Gaussian Mixture Model, we ended up deciding to by-pass testing that component in the end result. Thus, you will see that in the end we generate feature descriptors for each of the 3 possible images – (1) the trimmed image resulting from using the SGM to perform skin detection, (2) the trimmed image resulting from using the GMM to perform skin detection, and (3) the original image (no pre-processing). Once the skin in images has been detected and their features have been extracted, our *ckNN* classifier is used to

match a user to some identity in our dataset. The results of this recognition procedure are shown in the table below:

Note: *As discussed in section 3.2, in some cases the SGM/GMM can "fail." In the case that either model fails to detect a "viable" skin region, the original image is used in its place in order to ensure that a reasonable part of the face is represented so that the system can still attempt to perform recognition.*

Rates	SGM/SIFT	GMM/SIFT	SIFT
% Correct Identification	35%	33%	36%

Table 2: Performance results from end-to-end testing.

These results are not altogether encouraging for the overall performance of our face recognition system; however, we have immediately identified specific aspects that can be addressed in the future to improve recognition accuracy – most notably, investing in conducting better training for the Gaussian Models.

It should be recognized that we fixate mostly on improving the Gaussian Mixture Model because we feel fairly confident in the accuracy of the *SIFT* components and our *ckNN* classifier as a result of testing that we performed for the milestone. At that point, in order to test the performance of *SIFT* coupled with the *ckNN* classifier we used the Aberdeen dataset. [6]

This dataset is a fairly "easy" dataset to perform recognition on due to the fact that images have no drastic changes in illumination, and the scale/pose of the subjects are relatively fixed. Our ground-truth database (images we tested against) consisted of 30 subjects, which have 3 representative images each. By selecting a single image to be the *probe* image for each of the 30 subjects and using the other 2 images for each subject as training, we set out to see how well *SIFT* and *ckNN* worked to accurately recognize a given subject.

We observed that by using *SIFT* (section 2.3) and our *ckNN* Classifier (section 2.4) for the 30 test images, we are able to correctly recognize 29 of the 30 subjects (~97%).

4. Future Work

Given the results that we had there are a number of things that we have identified that we would like to address in our future work on this project.

First of all, we would like to invest in training our Gaussian Models better and further experiment with different input parameters such as the threshold used when comparing the skin/non-skin probabilities.

Secondly, we want to explore the effectiveness of other feature extraction algorithms and/or working on determining better SIFT parameters.

We would also like to explore implementing the use of a method which uses spatial information about feature descriptors to facilitate better feature correspondence matching (i.e. correspondences in the left region of the face match correspondences in the left region of the face in other images).

Finally, we would like to explore the use of a "confidence factor" as a metric to avoid/reduce improper identification (though we never thought to address this during the project because one of our assumptions was to always assume that the person we are trying to recognize *is* in the database).

Acknowledgment

The members of this project would like to thank Aarati Prasad and the other member of the mHealth Security & Privacy Lab at Dartmouth College for their support in obtaining relevant project data as well as the idea for the project in the first place.

5. References

[1] A. Abdel-Hakim, and A. Farag. CSIFT: A SIFT Descriptor with Color Invariant Characteristics. In *Computer Vision and Pattern Recognition*. 2006.

[2] O. Boiman, E. Shechtman, and M. Irani. In Defense of the Nearest-Neighbor Based Image Classification. In Computer Vision and Pattern Recognition. 2008.

[3] D. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*. 60(2) 2004, pp. 91-110.

[4] A. Vedaldi, 2006. An implementation of SIFT detector and descriptor. *Unpublished - UCLA*. http://www.robots.ox.ac.uk/~vedaldi/assets/sift/sift.pdf

[5] Labeled Faces in the Wild database: http://vis-www.cs.umass.edu/lfw/

[6] Aberdeen: http://pics.psych.stir.ac.uk/2D_face_sets.htm

[7] T.S. Caetano, S.D. Olabarriaga, D.A.C. Barone, Performance evaluation of single and multiple-Gaussian models for skin-color modeling, SIBGRAPI02, 2002.

[8] A.E. Abdel-Hakim and A.A. Farag, CSIFT: A SIFT Descriptor with Color Invariant Characteristics, *Proc. IEEE Conf. Computer Vision and Pattern Recognition,* pp. 1978-1983, 2006.

[9] SlideShare. *Expectation Maximization and Mixture of Gaussians* Slide-Show Presentation. 06/2010. http://www.slideshare.net/petitegeek/expectation-maximization-and-gaussian-mixture-models

[10] Gupta, Mohit, and Krishnan Ramnath. *Interactive Segmentation Tool-Box.* http://www.cs.cmu.edu/~mohitg/segmentation.htm