Face Detection

EE 368 Final Project

Spring 2003

Group 5

Cheryl Lam

Yu-Kuan Lin

David Mahashin

Introduction

The goal of the project is to identify faces in a group photograph. From a color image, we use a number of image processing techniques to detect the locations of faces within the image.

There are two main steps in our implementation: mask creation and face correlation. Mask creation involves detecting candidate face regions within the original image. This preliminary step selects all possible face region candidates, and rejects obvious non-face regions. These candidate face regions are then analyzed and correlated with an "average" face template, derived from a training image. The end result is a set of (x,y) image coordinates, identifying all face locations within the original image.

Mask Creation - Implementation Details

The mask creation portion of our implementation relies heavily on color segmentation and morphological image processing. The resulting mask, which identifies possible face regions within the original image, is then correlated against the average template face (derived prior to image analysis from a single training image). A number of statistics and decision criteria are then used to interpret the correlation output for each region and determine the number and location(s) of faces within each region. These regions are then mapped back to the original image in order to give the locations of faces within the original image.

YIQ Color Segmentation

The first step in creating the ìmaskî is the rejection of non-facial components, which include clothing and background objects. We decided to search for skin-color components in the image using the YIQ color space. This approach is motivated by the fact that luminance is a notoriously fickle determinant of skin tones. The YIQ space describes an image in terms of its luminance (Y), hue (I), and saturation (Q). Using the training images as test cases, all the faces within these images are extracted and their

luminance and hue distributions are observed. Because the face pixels occupy a small portion of the I and Q spaces, we are able to filter out non-facial components with relatively success. Several drawbacks include the inclusion of hands and arms in the mask, as well as other elements with color tones similar to human skin.

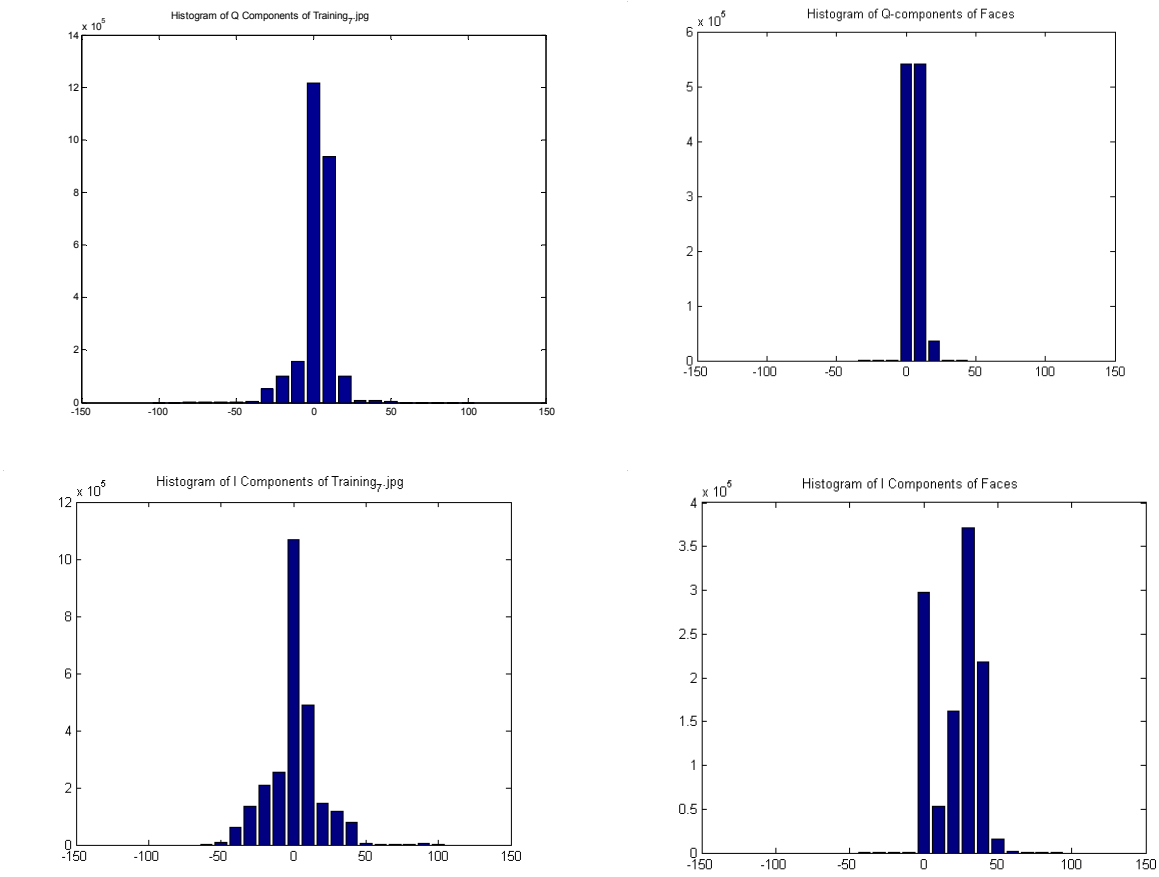

Fig. 1, a-d. Histograms in YIQ Space

Small Objects Rejection

The second step aims to filter out objects that are deemed too small to be faces. The threshold is determined dynamically by observing the distribution of all the non-zero regions in the mask. The threshold is set such that the probability of rejecting the smallest faces is very small. After small object rejection, we are able to virtually eliminate non-facial components. The problem that remains is the joining of multiple faces into a singular region. Ideally we would like each region to contain a single face, thus relegating the face correlator to decide

whether the region represents one face or a non-face.

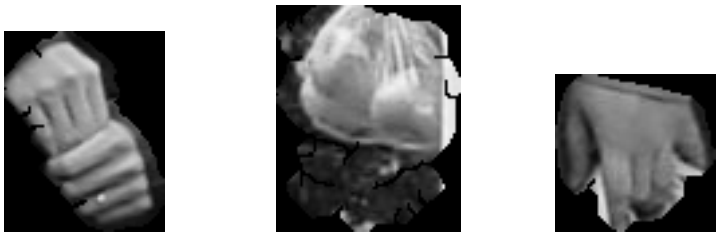

Fig. 2. Small objects successfully rejected

Joined Faces Separation

A significant challenge in creating the mask is the separation of joined faces. In a group photograph, people who stand close to one another will have their faces merged in the preliminary mask. The following is an example of three faces joined together:

Fig. 3. Joined faces within a single region

We considered many approaches to solve this problem, each with varying points of success and failure. A simple approach would be to repeatedly apply an erosion operation to the entire mask. This is successful only if the faces overlap only slightly.

In a more involved attempt, we attempt to classify joined-face regions by calculating and comparing the size of a blob to the median blob size. Regions of size greater than one standard deviation of the median are candidates for separation. To separate the blobs, we employ several morphological operations. Simple erosion of the mask works well for faces that overlap over a small region. However, because the training images contain a region of up to 8 overlapping faces, we have to devise a more involved strategy. The process is divided into two general phases. In the first phase, the candidate region undergoes aggressive erosion to break up the region into smaller pieces. At this point, we evaluate the size of each piece; if it is greater than a certain threshold, the piece is removed and is given to the template matcher as a possible face. It is tempting to reject the smaller pieces as non-faces. However, experiments show that erosion often causes the smaller face to break in half. To compensate, we dilate the smaller pieces with the intention of recovering faces that are unintentionally broken down. In most cases, this process of selective dilation is effective in reconstructing faces. There are, however, false alarms, when regions such as the neck are falsely reconstructed. We ultimately decide that given our edge correlator (described in the following) that this aggressive separation of the blobs is unnecessary. In fact, repeated application of erosion, dilation, and thickening give unnaturally jagged edges that are not beneficial to our edge correlator.

Face Correlator - Implementation Details

Once the color segmentation and morphological operations have occurred, we are left with shapeless ìblob,î containing both the image of interest as well as clutter and noise. One common method to distinguish desired image (i.e., faces in our case), and clutter, is through correlating the unknown images with a template, and then matching the image based on the correlation level. Methods for generating a template include averaging sample faces from training sets or

constructing a template from eigenfacial deconstructions. Template matching is a simple, yet robust

In our project, we first approach toward constructing the face template is first by constructing an average face from all the training sets, using ground truth masks for extraction. All faces are equally weighted as follows:

$$H = \frac{I}{N}\sum_{i=1}^{N} x_i$$

Where xi represents each facial image, scaled to the median width and height of the training image sets using bicubic interpolation. We chose the median width and height to maximize the correlation with images of different sizes. Our first template looks like this:

Fig. 4. First template face

We then used calculated the 2-D cross-correlation to determine the goodness of match. We were expecting to see good separation between faces and clutter. Instead we discovered that the average template does not perform very well as image identifier.

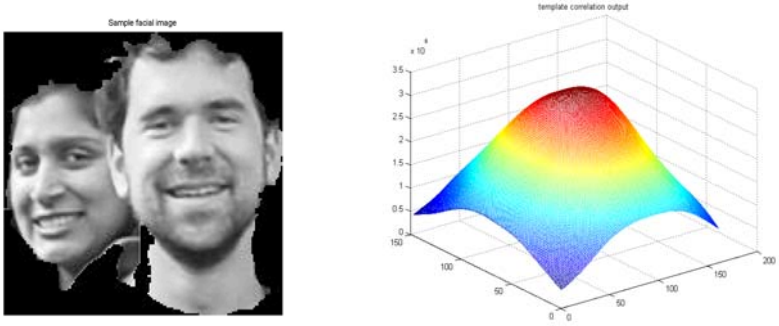

Fig. 5. a) 2-face region. b) Cross-correlation with first template face

As one can see, this average face does not generate high levels of separation (thus one cannot identify the two pictures from one single peak), which would be necessary for distinguishing faces. Worse, this face template was prone to identify not only faces, but skin-color regions in general, whether it be body parts, khakis, or the natural environment. This does not serve our detection purposes well. One possible explanation for this fact could be that human skin color all happen to have similar hue, saturation, and intensity. Furthermore, the averaging operation blends and reduces the unique and distinguishing features of all images, thus the separation between faces is quite poor.

We then resorted to a different method of face template. Realizing that much of what distinguishes a face lies within its feature, we decided to convert all grayscale images to black and white, and then use edge extraction to extract the important features, then combine them to form a template face. The following face was constructed from a sampling of 25 faces taken from the training set.

Fig. 6. Edge extraction template face

We extracted the edges first by converting the images into binary images, then using the Canny filter to extract edges. Furthermore, we normalize this image by subtracting its mean from itself, thus removing the elements common to most pixels. The remaining components highlight the differences in features, as well as reduce an upward bias in the correlation output. Another advantage of this approach is that since an edge template is a sparse matrix with mostly zeros, it

has low correlation except when it is exactly aligned with the template. The following figure illustrates the large separation between non-correlated and correlated output, and the large spike helps to distinguish the locations of desired faces.

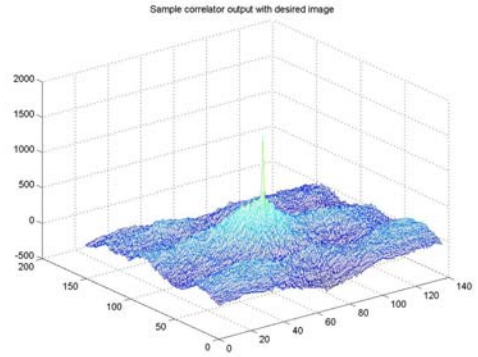

Fig. 7. Edge extraction template correlation

Another advantage of using edges for detection is that they form clearly identiable regions, which can then be counted and used to help distinguish unknown images extracted from the segmentation mask. For example, the following two images represent the edge representation of a manís face and hand. If we were to attempt to process the color images, because their size and color content are similar, their correlation output would probably not be too different, thus introducing the risk of false positives or missing hits. However, if we were to process them in the edge domain, then we can utilize the number of enclosed regions for identification.

Fig. 8 a&b. Edges and enclosed regions

Using **bwlabel**, we can label and count the regions, and it shows that the face has 43 connected regions, while the hand only has 14. Since faces are in general more complex than hands, clothing, or other clutter, by using the region statistics of these edges images, it can further supplement our algorithm.

Detection Algorithm

Three main criteria were used in our detection algorithm. We used:

- Correlation ñ Using the edge template as a measure of how closely it matches our average facial features
- Dimensions ñ Using width, height and area, whether the image is large enough to contain a face, observing aspect ratios, etc.
- Region counting ñ Using the region number statistics to help differentiate faces from clutter.

The precise algorithm is described in our Matlab implementation, but we basically used the correlation level on the first pass, eliminating objects that are clearly unlikely to be faces. Then we used heuristics such as dimensions and region counting to help determine whether zero, one, or multiple faces reside in a given region. If an image does not qualify as a face, we move on to the next object. If a single face exists, we return the location of the highest correlation peak as the faceís estimated location. If, however, we suspected an image formed by one or more partially or wholly joined faces, we have to then use a multiple facial detection routine.

Detecting Multiple Faces within a Single Face Region

For the case of multiple faces within a single region, we cannot rely solely on the single correlation peak detection used for single faces. Once an estimate of the expected number of faces within a region has been made, we attempt to find that number of peaks (and corresponding face locations) in the correlation output for the given region.

We begin by finding the largest peak (largest correlation output value) within the region. We count this as the first face location. We then exclude the subregion of "average" face radius around the first peak location, and continue searching for the next largest peak outside of this sub-region. We continue this process of subregion exclusion and peak detection until the desired number of peak locations have been found.

Fig. 9. Image region with 3 faces

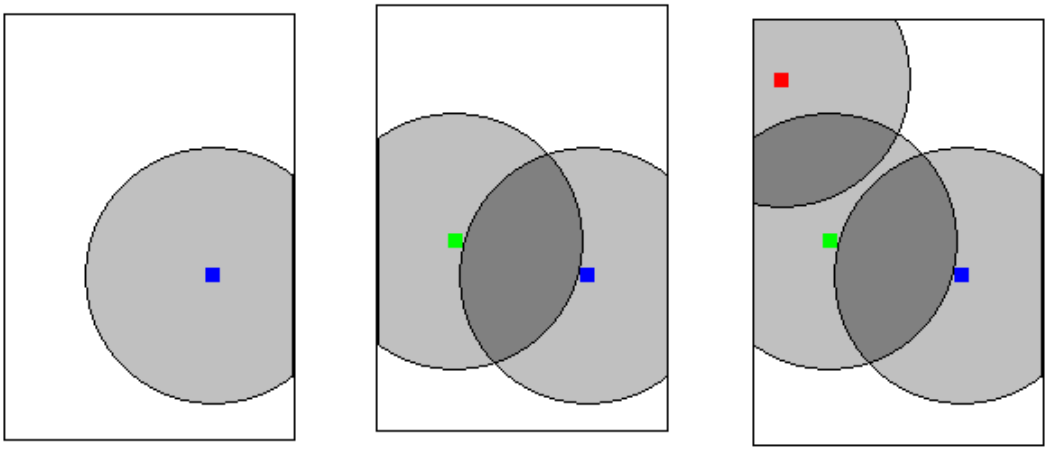

Fig. 10 a-c. a) Begin by detecting largest peak (denoted by blue square) and exclude circle around blue peak from futher peak search. b) Find next largest peak outside of circle, and exclude circle around green peak. c) Continue peak search and exclusion to find third peak

The "average" face radius is determined dynamically for each region, based on the size of the region and number of expected faces. For any given region, the average face radius is the mean of the x- and y-dimensions of the region, divided by the number of faces within the region. We have also included a "fudge factor" (which can be manually set) which is added to the average face radius in order to improve the performance. The nominal value of this fudge factor was determined by trial and error with different image face regions, derived from the training set.

Conclusion

On the whole, our algorithm performed with moderate to high success when used to detect faces in the training images. In some cases, we still miss ìobviousî faces within the image, while in others we detect false positives. Since the detection algorithm is highly dependant on face size and edges, it has particular difficulty with partially obstructed faces and faces which are very close to each other.

There is still room for algorithm refinement, particularly in the separation of closely-spaced faces and detecting multiple faces within a single candidate region. The general implementation, however, of color segmentation followed by edge image template matching, appears to give reasonable success in correct face detection.