Toward a Complete Face Recognition System

Chong Hock Kelvin Goh
School of Computer Science
Carnegie Mellon University
E-mail: chongk@andrew.cmu.edu

Advisor
Professor Takeo Kanade
Robotics Institute
Carnegie Mellon University
E-mail: Takeo.Kanade@cs.cmu.edu

Abstract

This research is to develop a computer program for human face recognition. A user sits in front of a computer equipped with a web camera. The program then captures the photo of the user, and through various modules, processes the photo and identifies the user within the database.

The system includes many modules that perform separate functions: image capturing module, face detector, face component localizer, face feature calculation, and classification algorithm. The current face classification module uses SSD (sum of squared differences) values of various face regions as features. I have investigated other features to be used for classification, in particular the use of wavelets.

The result of this research provides a basis for future development of many face recognition applications, such as surveillance, digital personal assistant, and camera-equipped cell-phones.

Toward a Complete Face Recognition System

Chong Hock Kelvin Goh, Professor Takeo Kanade (Advisor)
Carnegie Mellon University
E-mail: chongk@andrew.cmu.edu, Takeo.Kanade@cs.cmu.edu

Abstract

This research is to develop a computer program for human face recognition. A user sits in front of a computer equipped with a web camera. The program then captures the photo of the user, and through various modules, processes the photo and identifies the user within the database.

The system includes many modules that perform separate functions: image capturing module, face detector, face component localizer, face feature calculation, and classification algorithm. The current face classification module uses SSD (sum of squared differences) values of various face regions as features. I have investigated other features to be used for classification, in particular the use of wavelets.

The result of this research provides a basis for future development of many face recognition applications, such as surveillance, digital personal assistant, and camera-equipped cell-phones.

1 Introduction

We as human beings, perform face recognition from a very young age. It is a basic reflex to observe faces in order to recognize someone. Yet, currently, face recognition performed by a computer is not as instantaneous.

For a computer, the given image has to be processed by passing through a face detector module, identifying regions in the picture that contains a human face. Within this region, two traditional classes of techniques can be applied to the recognition of faces under normalized (roughly constant) illumination [1]. Feature based matching allows a face to be recognized even when details of individual features such as eyes, nose and mouth are not resolved. The key idea is to extract information such as relative position and other parameters of distinctive features such as eyes, nose and mouth. An example of a method to extract such features is the use of edge detection. The other technique is template matching. In the simplest version of template matching, the image, which is represented as a two dimensional array of values (typically illumination intensity) is compared with a single template representing the whole face. The comparison typically uses a suitable metric such as euclidean distance. In order to tackle the problem of pose differences between faces, a more complex approach is to use a single template together with a qualitative prior model of how a generic face transforms under a change of viewpoint. The deformation model is then heuristically built into the metric used by the matching measure; this is the idea underlying the technique of elastic templates and elastic bunch graphs.

Our approach is categorically that of appearance-based template matching. In template matching, if we use the whole face region for comparison, it is not easy to take into account changes in appearance due to pose differences, because the appearance in a different part of a face changes in a different manner due to its complicated three-dimensional shape. Instead, one can compare several subregions of the face separately, such as the eyes, nose and mouth [1, 2]. Hence, given a probe image (the image with face(s) to be recognized), the subregions of the face will be compared to the corresponding subregions of other faces stored in the database.

There are several different measures that can be used to compare such subregions. The current method is to use sum of squared differences (SSD). To compare the two, the SSD similarity value for each subregion after image normalization (so effectively the same as normalized correlation), is computed after finer alignment is done in order to compensate for the potential error in registration and the local deformation due to pose and other variations. The total similarity value between the probe face and the gallery face is then obtained by combining the similarity values of all subregions. Besides SSD, we have investigated the use of wavelets as a measure.

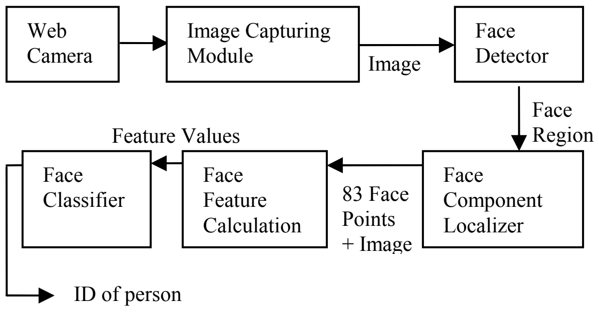
Many face recognition algorithms have been developed and some have been commercialized for applications such as access control and surveillance. Several studies have been reported in recent years [3, 4, 5] that compare those algorithms and evaluate the state-of-the-art of face recognition technology. These studies show that current algorithms are not robust against changes in illumination, pose, facial expression and occlusion.

Of these, pose change is one of the most important and difficult issues for the practical use of automatic face recognition. Our face feature calculation, and classification algorithm, developed by T. Kanade [7] and T. Sawao [8], is able to tackle pose differences well. It was shown that our algorithm outperformed a baseline algorithm (PCA) and a commercial product for face recognition [9]. However, unfortunately, the face detector employed poses as a limitation. The face detector used for this system is only able to detect frontal pose. Hence the overall system is only limited to frontal pose.

Because of this limitation, we have to modify the face feature calculation, and classification algorithm to accept only one pose. Also, the algorithm had to be modified such that it is able to use a dynamic database, instead of the static CMU PIE database.

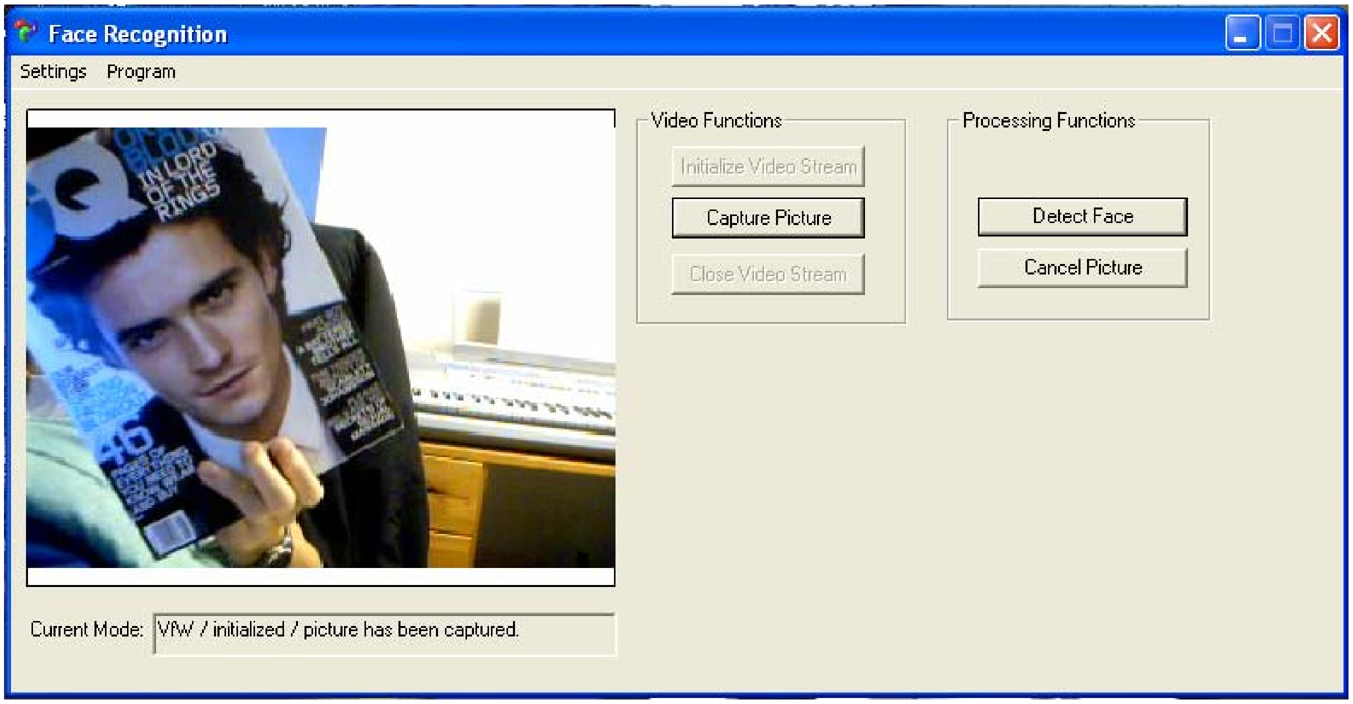
In addition to the face detector, face component localizer module developed by L. Gu [10], and the modified face feature calculation, and classification module, a image capturing module was built to allow a user to sit in front of a web camera and capture images of his/her own face. With these modules, we can build a complete face recognition system that accepts images from a web camera.

2 System Overview



The system connects to a web camera and allows the user to freeze a frame (capture image). The image will then be passed to the Face Detector module which will detect a face region in the image. If a face region is not automatically detected, then the user may manually specify a face region. The rectangular face region (4 coordinates) is passed on to the Face Component Localizer, which will produce 83 points that correspond to particular points of a face. These coordinates will then be passed on to the Face Feature Calculation module. It then calculates the feature values and passes this to the Face Classifier module. Based on a non-static database of images, the Face Classifier will output the ID of the

Fig 2.3 Screenshot of user interface for face detection



Current Mode: VfW / initialized / picture has been captured.

person. The user may add the image to the database if he wishes to do so.

The remaining sections of the paper will discuss the components of the system. Section 3 will discuss the Image Capturing module; Section 4 will discuss the Face Detector module and how it allows faces to be detected very quickly; Section 5 will discuss the Face Component Localizer module and how it produces the 83 facial points; Section 6 will discuss the Face Feature Calculation and Section 7 will discuss the Face Classifier module. Lastly, Section 8 will discuss the results we have.

3 Image Capturing Module

The purpose of this module is to allow the user to capture an image frame from a web camera. The program will show the user what the web camera is viewing. When the user is satisfied with the position of his face, the user would click on the Capture Picture button to freeze the frame. Since the face detector module can only detect frontal pose, it is best that the user positions his face facing directly to the web camera. However, the user is allowed to tilt his head left or right, as long as he is still directly facing the camera.

Fig 2.1 An example of a bad pose – User should be looking directly at web camera.

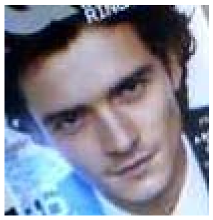


Fig 2.2 An example of a good pose – Even if the user's face is tilted, it is acceptable as long as the user is looking straight at the camera.

The main purpose for the user interface is to allow users to perform face recognition seamlessly. Hence the image capture module would pass the captured image frame to the face detector module, without the user's intervention. Once the Detect Face button is pressed, the face detector module would attempt to detect a face in the captured image frame. If a face has been automatically detected, the user is shown the face region generated by the face detector module (Refer to Appendix, Fig. 11.1). Then, 83 points of the face, which is generated by the face component localizer module, will be displayed. These 83 points correspond to particular features of the face. For example, point #5 would correspond to the left corner of the left eye. If a face was not detected by the face detector module, then the user will be prompted to manually select a rectangular region where the face lies. Then the face component localizer module will attempt to generate the 83 points based on the user-specified region. (Refer to Appendix, Fig. 11.2)

4 Face Detector

The face detector module uses the robust real-time object detection algorithm developed by P.Viola and M. Jones [11]. The frontal face detection system firstly varies the size and position of a detection window in the image. It then calculates the values of rectangle features within the detection window. For example, the value of a two-rectangle feature is the difference between the sum of the pixels within two rectangular regions. The regions have the same size and shape and are horizontally or vertically adjacent (see Fig. 2.4). A three-rectangle feature computes the sum within two outside rectangles subtracted from the sum in a center rectangle. Finally a four-rectangle feature computers the difference between diagonal pairs of rectangles.

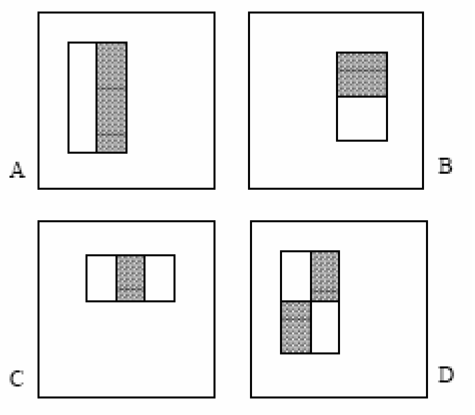


Figure 2.4 Example rectangle features shown relative to an enclosing detection window. The sum of the pixels which lie within the white rectangles are subtracted from the sum of pixels in the grey rectangles. Two-rectangle features are shown in (A) and (B). Figure (C) shows a three-rectangle feature, and (D) a four-rectangle feature.

To speed up calculation of rectangle features, the algorithm computes integral images. The integral image at location *x,y* contains the sum of the pixels above and to the left of *x,y*, inclusive. We store the integral image value for each *x,y* coordinate into an 2-D array. Hence using the integral image, any rectangular sum can be computed in four array references. Since the two-rectangle features defined above involve adjacent rectangular sums they can be computed in six array references.

Steerable filters, and their relatives, are excellent for the detailed analysis of boundaries, image compression, and texture analysis. In contrast rectangle features, while sensitive to the presence of edges, bars, and other simple image structures, are quite coarse and somewhat primitive. However, the extreme computational efficiency of rectangle features provides ample compensation for their limited flexibility.

The detector scans the input image at many scales; starting at the base scale in which objects are detected at a size of 24x24 pixels, the image is scanned at 11 scales each a factor of 1.25 larger than the last. The computation requires many iterations, hence we use these computationally efficient rectangle features.

These features are combined to form a classifier. Within any image sub-window the total number of Harr-like features is very large, far larger than the number of pixels. In order to ensure fast classification, the learning process must exclude a large majority of the available features, and focus on a small set of critical features. Motivated by the work of Tieu and Viola, feature selection is achieved through a simple modification of the AdaBoost procedure: the weak learner is constrained so that each weak classifier returned can depend on only a single feature. As a result each stage of the boosting process, which selects a new weak classifier, can be viewed as a feature selection process.

Next, the algorithm uses a method to combine successively more complex classifiers in a cascade structure which dramatically increases the speed of the detector by focusing attention on promising regions of the image. More complex processing is reserved only for these promising regions. In the domain of face detection it is possible to achieve fewer than 1% false negatives and 40% false positives using a classifier which can be evaluated in 20 simple operations. The effect of this filter is to reduce by over one half the number of locations where the final detector must be evaluated.

Hence the face detector module is extremely fast, and is able to detect frontal pose faces in both black & white, and color images.

5 Face Component Localizer

Given the face region, the face component localizer module will produce 83 points that correspond to parts of the face.

For this purpose people generally use a model-based method. For example, a face model is defined by a set of key feature points on face, and the connectivity of the points. Given an initial guess of the position, the algorithm attempts to find the best matching of the model to the image. Hence, based on the given face region, a template of 83 face points are generated. Then, these points will be adjusted to better match the face in the image.

The matching result includes two components. The first is a shape component which interprets the variation of the structure of the object. The second is a pose component including a set of geometrical parameters which transforms the shape in the image to standard pose.

Given an input image, we want to recover both shape parameters and pose parameters simultaneously.

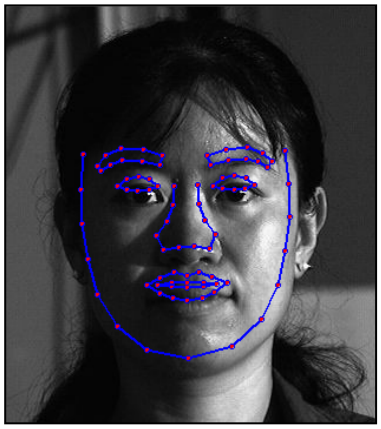


Fig 5.1 Starting Approximation

Firstly, based on the face region given by the Face Detector module, we generate a starting approximation based on a model template. We then implement the matching algorithm.

The matching algorithm is generally a 2 step iteration process.

Given a starting approximation or previous estimation of the model parameters, the first step is to independently update each feature point in a local window. In this example, for each point we search in a line segment along the normal direction, find the besting matching, and we get a shape like this, which is jaggy. We call it the observed shape.

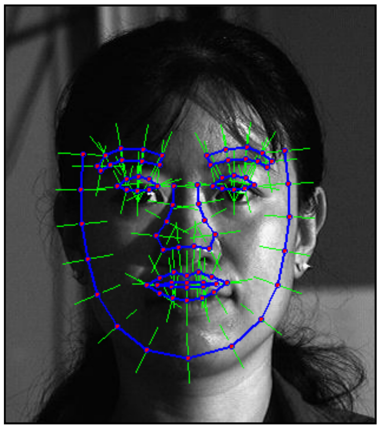


Fig 5.2 Observation

This local searching step generally encourages the landmark points move to the position with high feature response. For example, because of this illumination effects, the points on the left side of the nose contour are moved to some high contrast position. So from a global view, the observed shape does not looks like a reasonable face.

Therefore, the next step is to regularize the observation by some prior knowledge, and hope the results will be more accurate than the previous estimation.

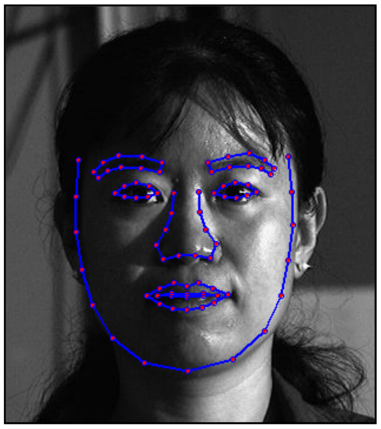


Fig 5.3 Regularization

Because of variability in images, local feature matching is always unreliable. We would say that the regularization step, or in other words, the way to incorporate the prior knowledge with the observation, is the most important part in shape registration [12]. This naturally leads us to a Bayesian treatment of shape registration problem.

Since these 83 points correspond to a certain part of the face, these points serve as a basis for comparison of certain face regions of the probe image, with the other images stored in the database. For example, the region around the left corner of the left eye of the probe image can be compared to the corresponding region of each image in the database.

These 83 points produced by the face component localizer will be passed on to the face feature calculation module. Although we have 83 points available, the face feature calculation module and face classifier module do not require that many points. Since a region surround each point is being compared, and these points are usually close to each other hence there will be many overlapping regions if all points were used. For example, point #1 is the right corner of the left eye, and point #5 is the left corner of the left eye, and points #2 to #4 are points located on the upper arc of the left eye.

6 Face Feature Calculation

In order to perform comparisons on fair ground, the image captured from the web camera has to be normalized. The target image that contains just the face (background removed) will be a 128x128 grayscale image. The image is reduced from RGB format as given by the web camera to grayscale, because the face classifier module is unable to process RGB images. Also, the normalization of RGB images is more complicated and this might increase recognition error rate, since 2 images containing the same face might not be recognized as being the same face due to different lighting that gives rise to more variation in illumination in color mode. For example, if a yellow light is shone, then the user might appear to have a yellower skin as compared to white light being shone.

The first step in the normalization process is to ensure that the face is vertically upright. Since the face captured by the web camera can be tilted, we perform affine transformation on the face by using the 83 points, so that 3 points of the face will always be transformed to the same coordinates. The left corner of the left eye of any face will always be at coordinate (32, 16) of the final 128x128 image. Likewise, the center of the nose will always be at coordinate (72, 64). In order to ensure that the face is vertically straight, based on the 83 points, the intersection point of the line that connects the left corner of the left eye to the right corner of the right eye (line1), and the line that connects the middle of the eyebrows to the middle of the mouth (line2), will always be at coordinate (32, 64). The affine transformation will hence result in line1 being horizontal and line2 being vertical. In this process, interpolation has to be used otherwise the transformed image will look jagged. This will result in a 128x128 image.

21 points are then selected from the 83 points to be used for comparison. Some of these 21 points are calculated by using 2 or more points from the original set of 83 points. Therefore even though we only use 21 points for comparison, this allows us to use more information, as compared to the case where we only take exactly 21 points from the original set of 83 points). For example, one of the 21 points is the middle of the left cheek. This would be the midpoint of the line that connects a point at the left border of the face, and a point at the left side of the nose.

Using these 21 points, a 9x15 subregion with the point as its center is created. Hence the face region is divided into a set of small subregions, and each subregion is compared with the corresponding subregion of the face in the gallery. To compare the two, a similarity value for each subregion, defined by the sum of squared difference (SSD), is computed after finer alignment is done in order to compensate for the potential error in registration and the local deformation due to pose and other variations. The total similarity value between the probe face and the gallery face is then obtained by combining the similarity values of all subregions. For each subregion the intensity values are normalized to have zero mean and a unit variance.

As the similarity measure, the SSD (sum of squared differences) values s_j between corresponding j–th subregions for all the pair of images $\boldsymbol{I}_k = (i_k, \phi_k)$ vs. $\boldsymbol{I}_m = (i_m, \phi_m)$ in the training dataset were calculated. Note that since we compute the SSD after image normalization for each subregion, it contains effectively the same information as normalized correlation.

In addition, we investigate the use of gabor wavelets instead of SSD as a feature. Using gabor wavelets, we can get texture characteristic locally in spatial frequency domain. In our implementation, we use 40 gabor wavelets for each location. We vary spatial frequency and orientation to yield 40 gabor wavelets. In particular, we have 5 spatial frequencies: ($\pi/2$, $\pi/4$, $\pi/8$, $\pi/16$, $\pi/32$), and 8 Orientations: (from 0 to π, differing $\pi/8$). We also vary the kernel size of the wavelet. It is important to choose a kernel size such that the wavelet does not exceed the image area (zeros padded on non-image region). If the kernel size is too big, the wavelet will include unintended areas such as the background, instead of the areas which we are interested in (e.g. left eye).

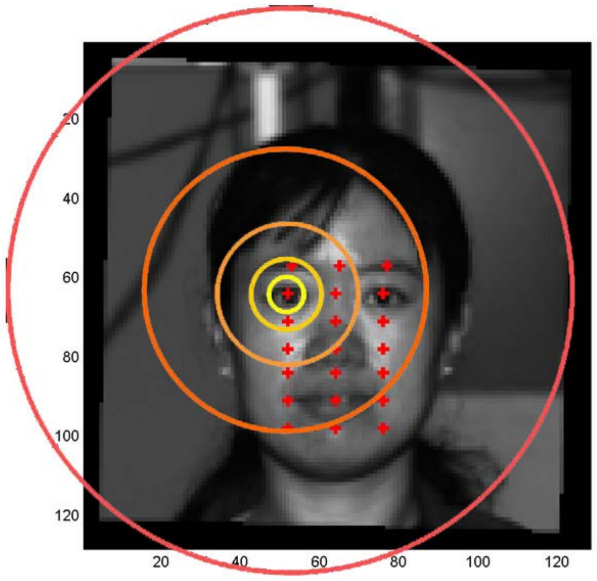


Fig 6.1 Coverage of gabor wavelets with different filter sizes

Initial results reveal that wavelets, on the whole, is not as a good as a feature as compared to SSD. However, for certain features such as the eye, it might be a better (more discriminative) feature than SSD. With SSD, the graph of same ID probability and the graph of different ID probability overlaps by around 10%. On the other hand, with gabor wavelets, the overlapping area is around 35%. A good feature with high discriminating factor will have a small overlapping area, since this means that it can clearly tell apart same ID images from different ID images.

7 Face Classifier

The previous Face Classifier module was built with the intention to recognize the ID of faces with different poses. This poses a problem because for our system, we can only accept images with frontal pose.

In the old module, for the training phase, we create $P(s_j|same, \phi_k, \phi_m)$, the conditional probability density of the *j*-th SSD similarity value s_j given that the images are of the class *same* identity and the poses of the two images are is ϕ_k and ϕ_m, respectively, from the histograms of SSD values of each region of the images. Likewise we also create $P(s_j|diff, \phi_k, \phi_m)$ for the class of *different.* We approximate these distributions by a Gaussian distribution. Accordingly,

$$P(s_j \mid same, \phi_k, \phi_m) = \frac{1}{\sqrt{2\pi}\sigma_j^{same}} \exp\left[-\frac{1}{2} \left(\frac{s_j - \mu_j^{same}}{\sigma_j^{same}} \right)^2 \right]$$

$$P(s_j \mid diff, \phi_k, \phi_m) = \frac{1}{\sqrt{2\pi}\sigma_j^{diff}} \exp\left[-\frac{1}{2} \left(\frac{s_j - \mu_j^{diff}}{\sigma_j^{diff}} \right)^2 \right]$$

where μ_j^{same} and μ_j^{diff}, σ_j^{same} and σ_j^{dif} are the means and standard deviations of class *same* and *diff*, respectively, which are obtained from the histograms. These distributions are then used during classification.

However, the old classification module becomes a problem for our system because the variance σ_j^{same} is zero for same ID. We would then encounter a division-by-zero error. Previously, images with same ID had different poses, hence this variance is non-zero. Therefore, we have to develop another classification method.

We cannot simply look at a SSD value between a probe image p and a gallery image g, and determine if it is too high and classify the ID of probe p as not ID of g. We cannot use the raw SSD value because other factors such as illumination will affect the raw value. Instead, we should use the distribution of SSD values and calculate probabilities instead.

For each gallery image i, we plot SSD between image i and all other gallery images k. We then assume normal distribution. Then, given the value of SSD between probe image p and gallery image i, we calculate P(probe image is not gallery image i). This probability is the area under the probability distribution graph of SSD values between image i and all other gallery images k, given the value of SSD between probe image p and gallery image i.

We then take the minimum of each probability that the probe image is not gallery image i. In other words, if gallery image i has the lowest probability that probe image is not the ID of gallery image i, then the ID of the probe image is most likely to be the ID of gallery image i.

Lastly, we set a threshold to indicate if the probe image is a new image. This threshold is set arbitrarily and we note that we should set this value using a classifier. We should run our system using probe images with known IDs, and classify the probabilities accordingly as Yes (found in gallery) and No (probe image is not in gallery).

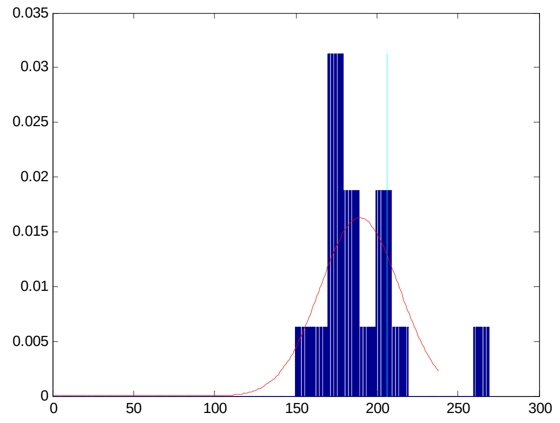


Fig 7.1 Blue bars: Histogram of SSD values between image i and all other gallery images k. Red line: Fitted normal distribution of histogram. Cyan line: SSD value of probe image p and gallery image i

8 Results

The system usually has problems trying to recognize the face captured via the web camera. This is usually because it is difficult for the user to adjust his/her face to the frontal pose expected by the system. Hence the user has to usually cancel the captured picture and readjust his/her face and retry.

Also, it takes a long time to classify / identify an image because SSD takes a long time to compute, especially because Matlab is inefficient with loops. However, to reduce inefficiency, we use a database of hashed SSD values. Hence we only need to compute SSD values of probe image with respect to the images in the gallery. We should not recomputed the SSD values between gallery images (which was done previously in the old face feature calculation module).

With regards to recognition accuracy, regrettably, we did not have time to formalize our results. However, we note that if there are faces in the gallery with the same ID as the probe face, the system will correctly identify the probe image most of the time. However, due to the arbitrarily set threshold, the system will identify a new probe image with an ID of a face from the gallery, instead of prompting the user that this is indeed a new face and ask if the user wishes to add the face to the database.

9 Conclusion

We have built a basic complete face recognition system, although it is still far from perfect. For example, the face detector module should be improved to accept faces with various poses. The next step will be to optimize the face feature calculation module and face classifer module so that face identification can be done in real time. If that is possible, then video surveillance can be done. The user interface can be altered so that continuous video feed (i.e. a sequence of image frames) is fed into the face recognition module, which then be used to compare the faces appearing on the web camera, with a database of known faces (such as known terrorists) for surveillance.

10 References

[1] R. Brunelli, T. Poggio. Face Recognition: Features versus Templates, IEEE Transactions On Pattern Analysis and Machine Intelligence, Vol. 15, No. 10, October 1993.

[2] A. Pentland, B. Moghaddam, T. Starner. View-based and Modular Eigenspaces for Face Recognition. M.I.T Media Laboratory Perceptual Computing Section Technical Report No.245.

[3] D. Blackburn, M. Bone, and P. Phillips. Facial Recognition Vendor Test 2000: Evaluation Report, 2000.

[4] P. J. Phillips, H. Moon, S. Rizvi, and P. Rauss. The FERET Evaluation Methodology for Face Recognition Algorithms: IEEE Trans. On PAMI, 22(10): 1090-1103, 2000.

[5] R. Gross, J. Shi, and J. Cohn. Quo Vadis Face Recognition?: Third Workshop on Empirical Evaluation Methods in Computer Vision, December, 2001.

[6] T. Kanade, A. Yamada. Multi-Subregion Based Probabilistic Approach Toward Pose-Invariant Face Recognition. IEEE International Symposium on Computational Intelligence in Robotics and Automation (CIRA2003), Kobe, Japan. August 2003.

[7] T. Kanade. Homepage. http://www.ri.cmu.edu/people/kanade_takeo.html (15 Apr. 2004)

[8] T. Sawao. Homepage. http://www.ri.cmu.edu/people/sawao_takashi.html (8 Mar. 2004)

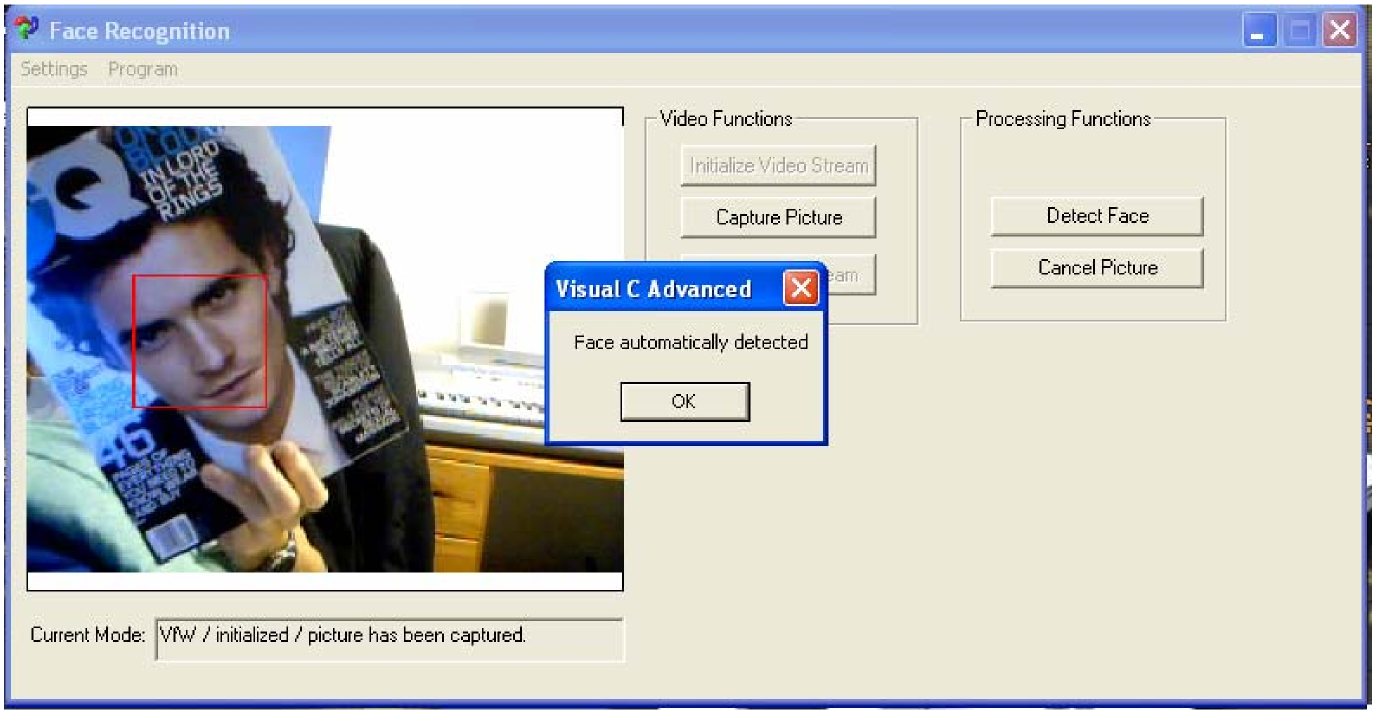
[9] P. S. Penev and J. J. Atick. Local Feature Analysis: A General Statistical Theory for Object Representation. Network: Computation in Neural Systems 7(3), 477-500.

[10] L. Gu. Homepage. http://www-2.cs.cmu.edu/~gu/ (15 Apr. 2004)

[11] P.Viola, M. Jones. Robust Real-time Object Detection. Second International Workshop on Statistical and Computational Theories of Vision – Modeling, Learning, Computing, and Sampling. Vancouver, Canada. July 13, 2001.

[12] Y.Zhou, L.Gu, H-J. Zhang. Bayesian Tangent Shape Model:Estimating Shape and Pose Parameters via Bayesian Inference. 2003.

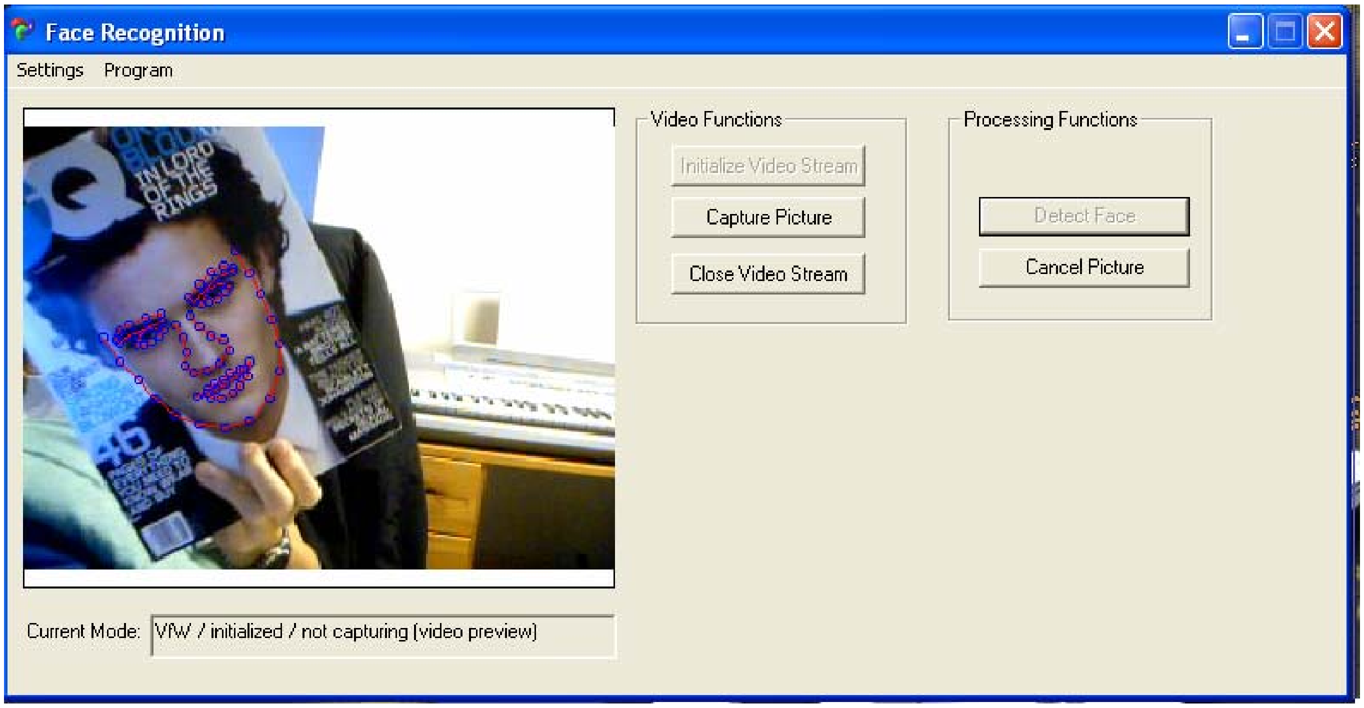
11 Appendix



Face automatically detected

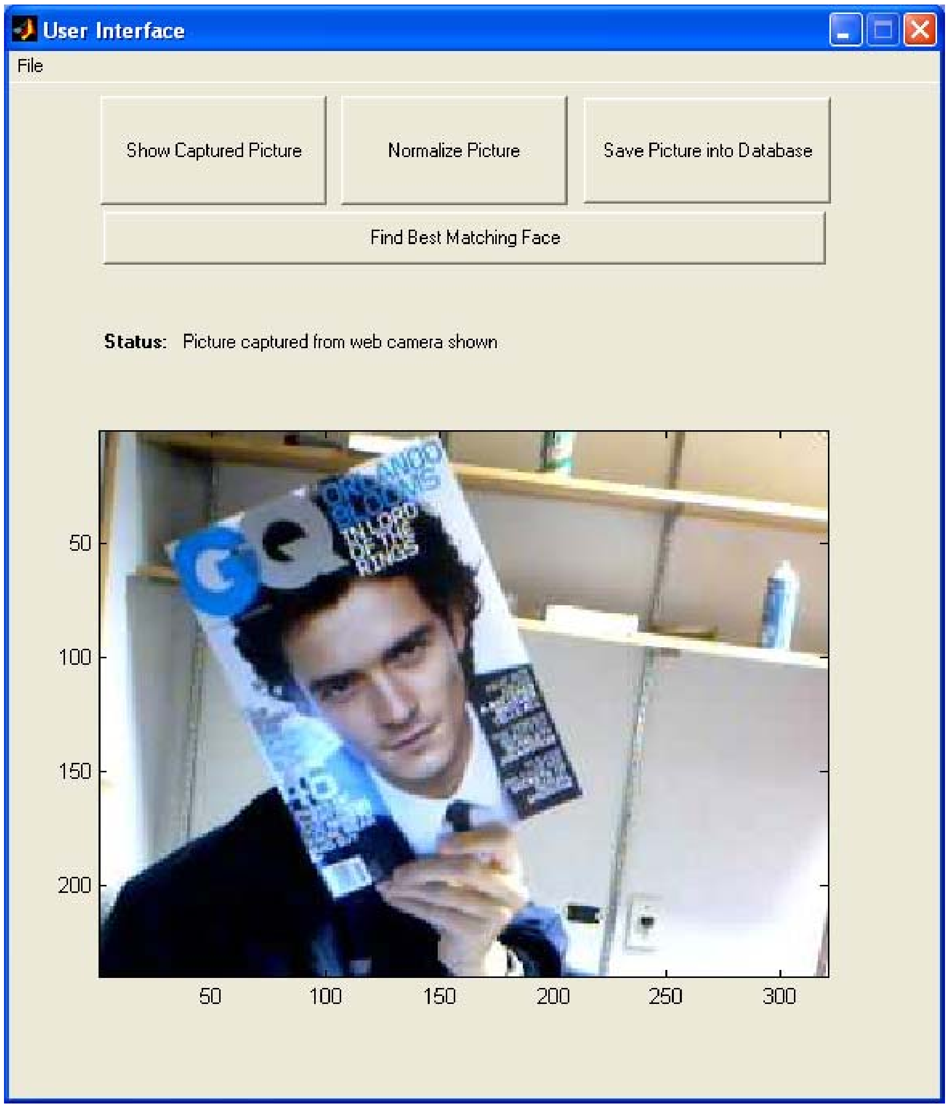
Current Mode: VfW / initialized / picture has been captured.

Fig 11.1 When the "Detect Face" button is clicked after a still picture has been captured, the Face Detection module will attempt to automatically detect a face region. The face region is shown with a red rectangle if a face can be automatically detected with the module.



Current Mode: VfW / initialized / not capturing (video preview)

Fig 11.2. Immediately after a face region is automatically detected, the Face Component Localizer module will generate 83 points that correspond to parts of the face.



Status: Picture captured from web camera shown

Fig 11.3. Screenshot of the user interface of the Face Recognizer. The program shows the captured image from the web camera.

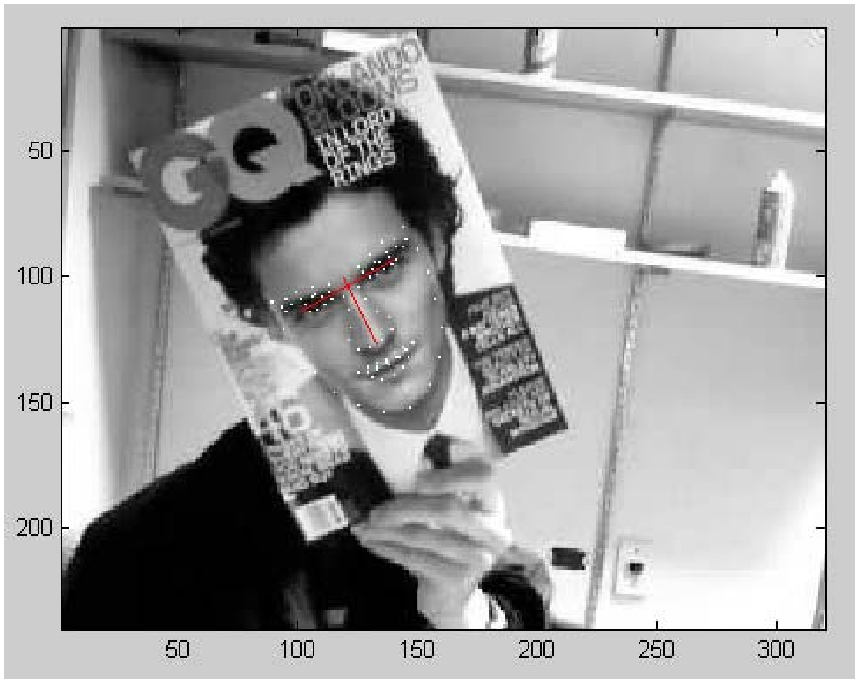
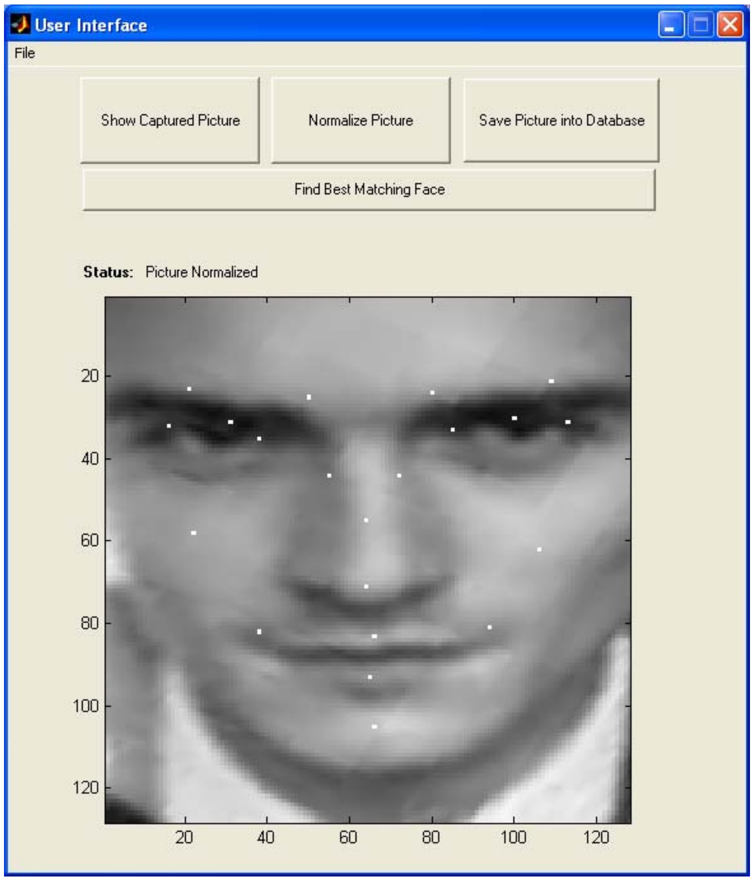
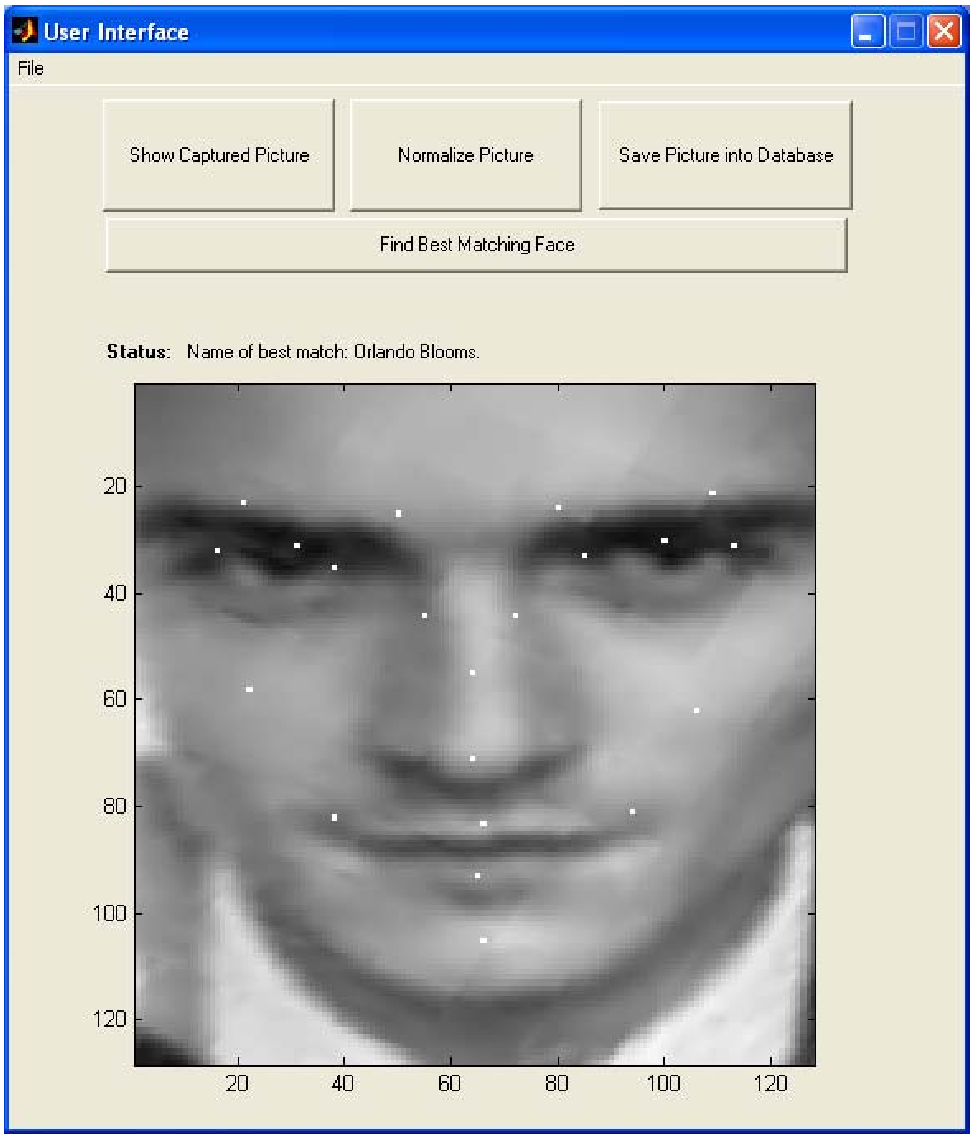


Fig 11.4. The user has to normalize the image before trying to recognize the probe image. During the normalization process, the program rotates and crops the image, according to the 83 points passed in from the Face Detector module. The normalized image is then displayed in the user interface.



Status: Picture Normalized



Status: Name of best match: Orlando Blooms.

Fig 11.5. Result of face recognition. The ID of the closest matching face in the gallery is shown. If the probe face is not found the database, the user will be prompted to enter face into gallery (if user wishes to).