FACIAL FEATURE DETECTION USING HAAR CLASSIFIERS*

Phillip Ian Wilson
Texas A&M University – Corpus Christi
6300 Ocean Dr., Corpus Christi, TX 78412
361-877-9062
pwilson@sci.tamucc.edu

Dr. John Fernandez
Texas A&M University – Corpus Christi
6300 Ocean Dr. CI334, Corpus Christi, TX 78412
361-825-3622
jfernand@sci.tamucc.edu

ABSTRACT

Viola and Jones [9] introduced a method to accurately and rapidly detect faces within an image. This technique can be adapted to accurately detect facial features. However, the area of the image being analyzed for a facial feature needs to be regionalized to the location with the highest probability of containing the feature. By regionalizing the detection area, false positives are eliminated and the speed of detection is increased due to the reduction of the area examined.

INTRODUCTION

The human face poses even more problems than other objects since the human face is a dynamic object that comes in many forms and colors [7]. However, facial detection and tracking provides many benefits. Facial recognition is not possible if the face is not isolated from the background. Human Computer Interaction (HCI) could greatly be improved by using emotion, pose, and gesture recognition, all of which require face and facial feature detection and tracking [2].

Although many different algorithms exist to perform face detection, each has its own weaknesses and strengths. Some use flesh tones, some use contours, and other are even more complex involving templates, neural networks, or filters. These algorithms suffer from the same problem; they are computationally expensive [2]. An image is only a collection of color and/or light intensity values. Analyzing these pixels for face detection is time consuming and difficult to accomplish because of the wide variations of shape and

pigmentation within a human face. Pixels often require reanalysis for scaling and precision. Viola and Jones devised an algorithm, called Haar Classifiers, to rapidly detect any object, including human faces, using AdaBoost classifier cascades that are based on Haar-like features and not pixels [9].

HAAR CASCADE CLASSIFIERS

The core basis for Haar classifier object detection is the Haar-like features. These features, rather than using the intensity values of a pixel, use the change in contrast values between adjacent rectangular groups of pixels. The contrast variances between the pixel groups are used to determine relative light and dark areas. Two or three adjacent groups with a relative contrast variance form a Haar-like feature. Haar-like features, as shown in Figure 1 are used to detect an image [8]. Haar features can easily be scaled by increasing or decreasing the size of the pixel group being examined. This allows features to be used to detect objects of various sizes.

Integral Image

The simple rectangular features of an image are calculated using an intermediate representation of an image, called the integral image [9]. The integral image is an array containing the sums of the pixels' intensity values located directly to the left of a pixel and directly above the pixel at location (x, y) inclusive. So if A[x,y] is the original image and AI[x,y] is the integral image then the integral image is computed as shown in equation 1 and illustrated in Figure 2.

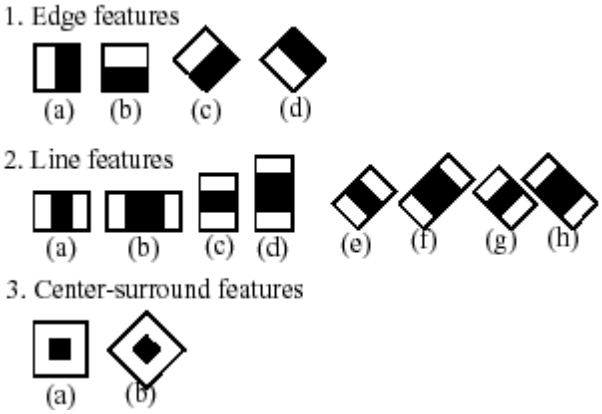

Figure 1 Common Haar Features

$$AI[x,y] = \sum_{x' \le x, y' \le y} A(x', y') \qquad (1)$$

The features rotated by forty-five degrees, like the line feature shown in Figure 1 2(e), as introduced by Lienhart and Maydt, require another intermediate representation called the rotated integral image or rotated sum auxiliary image [5]. The rotated integral image is calculated by finding the sum of the pixels' intensity values that are located at a forty five degree angle to the left and above for the x value and below for the y value. So if A[x,y] is the original image and AR[x,y] is the rotated integral image then the integral image is computed as shown in equation 2 an illustrated in Figure 3.

$$AR[x,y] = \sum_{x' \le x, x' \le x - |y - y'|} A(x', y') \qquad (2)$$

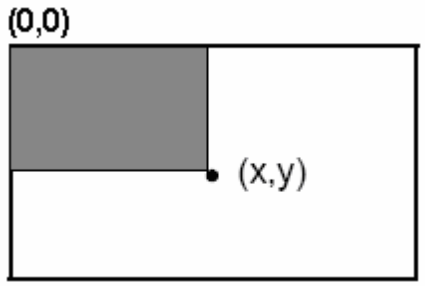

Figure 2 Summed area of integral image

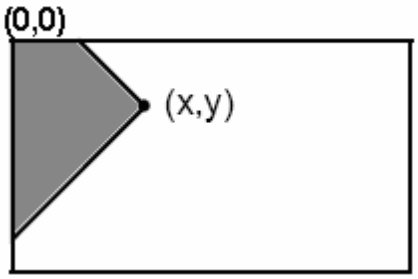

Figure 3 Summed area of rotated integral image

It only takes two passes to compute both integral image arrays, one for each array. Using the appropriate integral image and taking the difference between six to eight array elements forming two or three connected rectangles, a feature of any scale can be computed. Thus calculating a feature is extremely fast and efficient. It also means calculating features of various sizes requires the same effort as a feature of only two or three pixels. The detection of various sizes of the same object requires the same amount of effort and time as objects of similar sizes since scaling requires no additional effort [9].

Classifiers Cascaded

Although calculating a feature is extremely efficient and fast, calculating all 180,000 features contained within a 24 × 24 sub-image is impractical [Viola 2001, Wilson 2005]. Fortunately, only a tiny fraction of those features are needed to determine if a sub-image potentially contains the desired object [6]. In order to eliminate as many sub-images as possible, only a few of the features that define an object are used when analyzing sub-images. The goal is to eliminate a substantial amount, around 50%, of the sub-images that do not contain the object. This process continues, increasing the number of features used to analyze the sub-image at each stage.

The cascading of the classifiers allows only the sub-images with the highest probability to be analyzed for all Haar-features that distinguish an object. It also allows one to vary the accuracy of a classifier. One can increase both the false alarm rate and positive hit rate by decreasing the number of stages. The inverse of this is also true. Viola and Jones were able to achieve a 95% accuracy rate for the detection of a human face using only 200 simple features [9]. Using a 2 GHz computer, a Haar classifier cascade could detect human faces at a rate of at least five frames per second [5].

TRAINING CLASSIFIERS FOR FACIAL FEATURES

Detecting human facial features, such as the mouth, eyes, and nose require that Haar classifier cascades first be trained. In order to train the classifiers, this gentle AdaBoost algorithm and Haar feature algorithms must be implemented. Fortunately, Intel

developed an open source library devoted to easing the implementation of computer vision related programs called Open Computer Vision Library (OpenCV). The OpenCV library is designed to be used in conjunction with applications that pertain to the field of HCI, robotics, biometrics, image processing, and other areas where visualization is important and includes an implementation of Haar classifier detection and training [8].

To train the classifiers, two set of images are needed. One set contains an image or scene that does not contain the object, in this case a facial feature, which is going to be detected. This set of images is referred to as the negative images. The other set of images, the positive images, contain one or more instances of the object. The location of the objects within the positive images is specified by: image name, the upper left pixel and the height, and width of the object [1]. For training facial features 5,000 negative images with at least a mega-pixel resolution were used for training. These images consisted of everyday objects, like paperclips, and of natural scenery, like photographs of forests and mountains.

In order to produce the most robust facial feature detection possible, the original positive set of images needs to be representative of the variance between different people, including, race, gender, and age. A good source for these images is National Institute of Standards and Technology's (NIST) Facial Recognition Technology (FERET) database. This database contains over 10,000 images of over 1,000 people under different lighting conditions, poses, and angles [10]. In training each facial feature, 1,500 images were used. These images were taken at angles ranging from zero to forty five degrees from a frontal view. This provides the needed variance required to allow detection if the head is turned slightly [1].

Three separate classifiers were trained, one for the eyes, one for the nose, and one for the mouth. Once the classifiers were trained, they were used to detect the facial features within another set of images from the FERET database. The accuracy of the classifier was then computed as shown in Table 1. With the exception of the mouth classifier, the classifiers have a high rate of detection. However, as implied by [3], the false positive rate is also quite high.

Facial Feature	Positive Hit Rate	Negative Hit Rate
Eyes	93%	23%
Nose	100%	29%
Mouth	67%	28%

Table 1 Accuracy of Classifiers

REGIONALIZED DETECTION

Since it is not possible to reduce the false positive rate of the classifier without reducing the positive hit rate, a method besides modifying the classifier training attribute

is needed to increase accuracy [3]. The method proposed to is to limit the region of the image that is analyzed for the facial features. By reducing the area analyzed, accuracy will increase since less area exists to produce false positives. It also increases efficiency since fewer features need to be computed and the area of the integral images is smaller.

In order to regionalize the image, one must first determine the likely area where a facial feature might exist. The simplest method is to perform facial detection on the image first. The area containing the face will also contain facial features. However, the facial feature cascades often detect other facial features as illustrated in Figure 4. The best method to eliminate extra feature detection is to further regionalize the area for facial feature detection. It can be assumed that the eyes will be located near the top of the head, the nose will be located in the center area and the mouth will be located near the bottom.

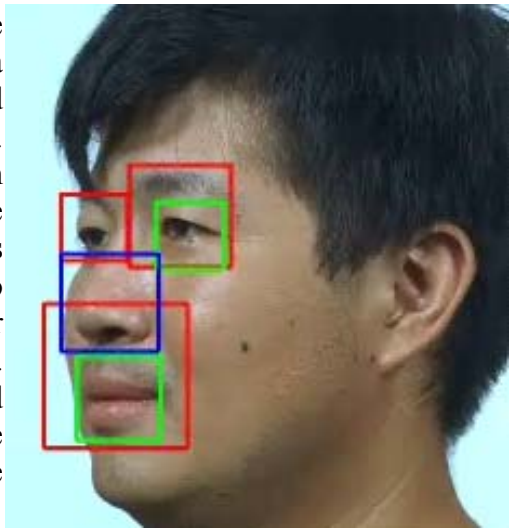

Figure 4 Inaccurate Detection: Eyes (red), Nose (blue), and Mouth (green), image from [4]

The upper 5/8 of the face is analyzed for the eyes. This area eliminates all other facial features while still allowing a wide variance in the tilt angle. The center of the face, an area that is 5/8 by 5/8 of the face, was used to for detection of the nose. This area eliminates all but the upper lip of the mouth and lower eyelid. The lower half of the facial image was used to detect the mouth. Since the facial detector used sometimes eliminates the lower lip the facial image was extended by an eighth for mouth detection only.

RESULTS

The first step in facial feature detection is detecting the face. This requires analyzing the entire image. The second step is using the isolated face(s) to detect each feature. The result is shown in Figure 5. Since each the portion of the image used to detect a feature is much smaller than that of the whole image, detection of all three facial features takes less time on average than detecting the face itself. Using a 1.2GHz AMD processor to analyze a 320 by 240 image, a frame rate of 3 frames per second was achieved. Since a frame rate of 5 frames per second was achieved in facial detection only by [5] using a much faster processor, regionalization provides a tremendous increase in efficiency in facial feature detection.

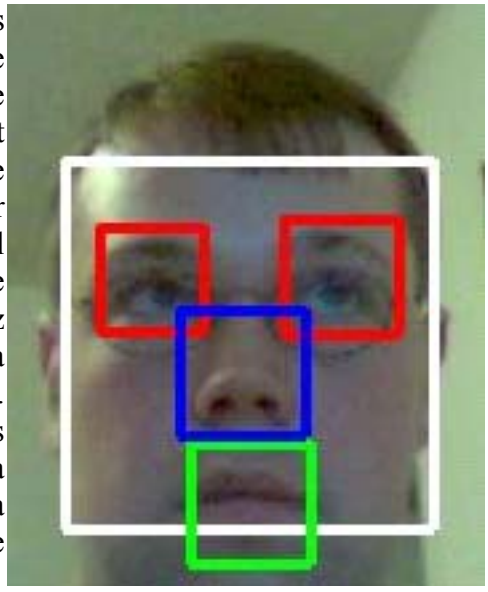

Figure 5 Detected Objects: Face (white), Eyes (red), Nose (blue), and Mouth (green)

Regionalization also greatly increased the accuracy of the detection. All false positives were eliminated, giving a detection rate of around 95% for the eyes and nose. The mouth detection has a lower rate due to the minimum size required for detection. By changing the height and width parameter to more accurately represent the dimensions of the mouth and retraining the classifier the accuracy should increase the accuracy to that of the other features.

FUTURE PLANS

With the successful detection of facial features, the next goal is to research the ability for more precise details, like individual points, of the facial features to be gathered. These points will be use to differentiate general human emotions, like happiness and sadness. Recognition of human emotion would require detection and analysis of the various elements of a human face, like the brow and the mouth, to determine an individual's current expression. The expression can then be compared to what is considered to be the basic signs of an emotion. This research will be used in the field human-computer interaction to analyze the emotions one exhibits while interacting with a user interface.

ACKNOWLEDGEMENT

This work was partially funded by NSF Minority Institutions Infrastructure Program grant #EIA-0330822.

REFERENCES

1. Adolf, F. How-to build a cascade of boosted classifiers based on Haar-like features. http://robotik.inflomatik.info/other/opencv/OpenCV_ObjectDetection_HowTo.pdf, June 20 2003.

2. Bradski, G. Computer vision face tracking for use in a perceptual user interface. *Intel Technology Journal*, 2nd Quarter, 1998.

3. Cristinacce, D. and Cootes, T. Facial feature detection using AdaBoost with shape constraints. *British Machine Vision Conference*, 2003.

4. The Facial Recognition Technology (FERET) Database. *National Institute of Standards and Technology*, 2003. http://www.itl.nist.gov/iad/humanid/feret/

5. Lienhart, R. and Maydt, J. An extended set of Haar-like features for rapid object detection. *IEEE ICIP 2002*, Vol. 1, pp. 900-903, Sep. 2002..

6. Menezes, P., Barreto, J.C. and Dias, J. Face tracking based on Haar-like features and eigenfaces. *5th IFAC Symposium on Intelligent Autonomous Vehicles*, Lisbon, Portugal, July 5-7, 2004.

7. Muller, N., Magaia, L. and Herbst B.M. Singular value decomposition, eigenfaces, and 3D reconstructions. *SIAM Review*, Vol. 46 Issue 3, pp. 518–545. Dec. 2004.

8. Open Computer Vision Library Reference Manual. *Intel Corporation*, USA, 2001.

9. Viola, P. and Jones, M. Rapid object detection using boosted cascade of simple features. *IEEE Conference on Computer Vision and Pattern Recognition*, 2001.

10. Wilson, P and Fernandez, J. Establishing a face recognition research environment using open source software. *ASEE Gulf-Southwest Annual Conference*, March, 2005.