Songklanakarin J. Sci. Technol.
31 (2), 157-165, Mar. - Apr. 2009

Songklanakarin Journal
of Science and Technology
http://rdo.psu.ac.th/sjst

*Original Article*

# Natural-pose hand detection in low-resolution images

Nyan Bo Bo[1] , Matthew N. Dailey[2] and Bunyarit Uyyanonvara[1]

[1] *Information Technology Program, School of Information and Computer Technology, Sirindhorn International Institute of Technology, Thammasat University Bangkadi, Muang, Pathum Thani, 12000 Thailand*

[2] *Computer Science and Information Management, School of Engineering and Technology, Asian Institute of Technology, Khlong Luang, Pathum Thani, 12120 Thailand*

Received 4 July 2007; Accepted 25 December 2008

**Abstract**

Robust real-time hand detection and tracking in video sequences would enable many applications in areas as diverse as human-computer interaction, robotics, security and surveillance, and sign language-based systems. In this paper, we introduce a new approach for detecting human hands that works on single, cluttered, low-resolution images. Our prototype system, which is primarily intended for security applications in which the images are noisy and low-resolution, is able to detect hands as small as 24×24 pixels in cluttered scenes. The system uses grayscale appearance information to classify image sub-windows as either containing or not containing a human hand very rapidly at the cost of a high false positive rate. To improve on the false positive rate of the main classifier without affecting its detection rate, we introduce a post-processor system that utilizes the geometric properties of skin color blobs. When we test our detector on a test image set containing 106 hands, 92 of those hands are detected (86.8% detection rate), with an average false positive rate of 1.19 false positive detections per image. The rapid detection speed, the high detection rate of 86.8%, and the low false positive rate together ensure that our system is useable as the main detector in a diverse variety of applications requiring robust hand detection and tracking in low-resolution, cluttered scenes.

**Keywords:** hand detection and tracking, boosted classifier cascade, skin detection, Mahalanobis classifier, artificial intelligence, computer vision and image processing

## 1. Introduction

If it were possible to detect and track human hands in video sequences, a variety of useful applications would be possible. These applications include human-computer interaction, human-robot interaction, gesture and sign-language recognition, intelligent security systems and more. Over the last 15 years, the problem of hand tracking has become an attractive area for research in the field of computer vision. Many early hand tracking systems relied on uncluttered static backgrounds, high resolution imagery, and manual initialization. Most of the modern hand tracking systems are oriented towards sign language recognition, human-computer interaction, and human-robot interaction. In these applications, it is possible to make the very useful assumption that only hands are moving while the rest of the scene is stationary. The problem can be further simplified by assuming that there will be only two hands, since there should be only one person performing sign language or gestures in the scene. Nowadays, the systems are becoming more robust, but they generally still require high resolution imagery.

We are primarily interested in hand detection because monitoring person's hand is a key to predict what that person is doing. In security applications, it would be very useful to detect and track hands of people in the scene and perform

*Corresponding author.
Email address: nyanbobo@gmail.com

automated analysis of their actions, e.g., by determining if they are walking, running, punching someone, or even identifying any object they are holding. Detecting and tracking hands in security applications is more challenging than in human-computer interaction because most surveillance cameras provide noisy images, with human figures quite far away and therefore appearing at a fairly low resolution. The resolution of hands in those images may be as small as 24×24 pixels; detecting such small hands in static images is a very challenging task. Another difficulty is that motion information is less useful since there may be many people in the scene, and their entire bodies may be moving from frame to frame as they move in front of the security camera.

Some early hand tracking systems like Pfinder proposed by Wren *et al.* (1997) attempt to follow the way humans look for the hand in images. Instead of directly detecting hands in an image, Pfinder looks for human bodies first and then easily segments out hands from the rest of the body by using skin color. However, since detecting humans in a cluttered video sequence is itself a very difficult problem, and the human body could easily be partially occluded in the scene, we try to bypass the human detection problem in our work by finding hands directly, without any attempt to find the entire human body first.

Most of the modern hand tracking systems fall into one of two main categories. The first approach uses skin color information to segment hands from the background and then tracks segmented hands between frames using a tracking algorithm. The face and hand tracking system for sign language recognition proposed by Soontranon *et al.* (2004) first segments the image into skin and non-skin regions using an elliptical model for skin pixels in CbCr space. Then face detection is used to locate the face skin blob ideally leaving only the skin blobs of hands. The system constructs a template for each hand then in subsequent frames, finds the region best matching that template using a minimum mean-squared error cost function. A similar approach is used by Wachs *et al.* (2005) to detect and track hands for human-robot interaction. The system proposed by Varona *et al.* (2004) also takes this approach but their system is extended to track hands and faces in 3D for a virtual reality application. The hand tracker of Shamaie and Sutherland (2005) does not use skin color information so it works on monochrome video sequences. Hands are extracted from the background using a blob analysis algorithm then tracked using a dynamic model from control theory. Unfortunately, these approaches relying on tracking are not suitable when the goal is to extract hands from single images.

However, in a second approach, a detection window is scanned over the image and each of the scanned image patches are classified as hand or non-hand. In contrast to the first approach, this approach can be used to detect hands in static images. Barreto *et al.* (2004) applied improved version of Viola and Jones (2001) face detector by Lienhart and Maydt (2002), to detect hands for a human-robot interaction application. Their hand detection system works quite well at various scales and with different backgrounds under various illumination conditions. The hand tracking system proposed by Ong and Bowden (2004) uses a similar approach, but they construct a tree-structured classifier, instead of a linear cascade, not only to detect hands but also to classify hand posture. Both of these systems require high-resolution imagery. The hand detector by Caglar and Lobo (2006) also detects hands in high resolution static images, in this case making use of the geometric properties of the hand without the use of skin color or motion information. Their proposed system is robust to the size and the orientation of hands with the limitation that one or more fingers must be visible.

The goal of our proposed system is to detect and track multiple hands in arbitrary postures in relatively low-resolution video sequences. Our approach uses grayscale appearance information to reject most of the non-hand image regions very rapidly and then uses the shape of skin color regions to reject most of the remaining non-hand image patches. We conducted a thorough evaluation on our proposed system and found that its detection rate was 86.8% and that its false positive rate was 1.19 false detections per image on average. The system's speed and accuracy will enable many useful applications that are based on hand detection and tracking.

## 2. Materials and Methods

### 2.1 Hand detector

A block diagram of our hand detection system is shown in Figure 1. A scan window is scanned over the input image at different scales and each of the resulting image patches is fed into a classifier cascade which rapidly determines whether the image patch is a hand. Our classifier cascade eliminates

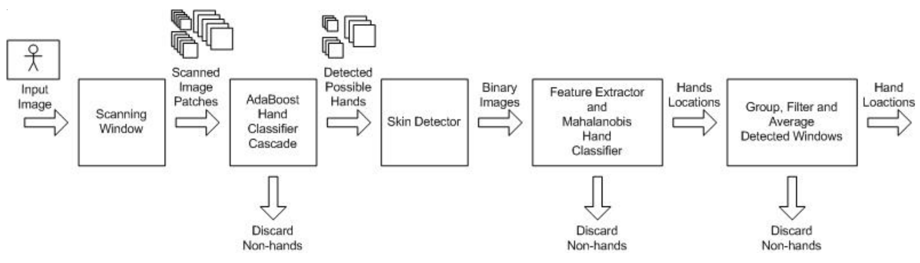


Figure 1. Hand detection system architecture.

more than 95% of the non-hand regions in a given image. However, due to the large number of candidate regions in one image, to be practical, the false positive rate must be further reduced. To serve this need, we add a postprocessor to the system in order to further reduce false positive detections. The postprocessor takes advantage of a priori knowledge of hand's color and geometry. Skin detection, feature extraction, and Mahalanobis classification are the essential building blocks of our postprocessor.

## 2.2 Boosted classifier cascade

Viola and Jones (2001, 2004) originally proposed the cascade of boosted classifiers as a real-time general object detector and applied it to face detection. They showed that the system can robustly detect faces in static images independent of the background. The system runs in real-time since the feature detector is limited to a class of Haar-like filters that can be computed in constant time with the help of integral images, regardless of the spatial extent of the filters. The speed of the system is increased even further by arranging the classifiers in a cascaded fashion, so that the early stages reject most of the image patches unlikely to contain the object of interest. The cascade therefore only spends significant compute time on the image patches most likely to contain the object of interest.

Each stage in the cascade is constructed from a set of simple Haar-like filters using Freund and Shapire's (1997) AdaBoost algorithm. AdaBoost builds a strong nonlinear classifier from multiple weak threshold classifiers, in this case each using a Haar-like filter, a threshold, and a weight, all of which are selected by AdaBoost to minimize the weighted error for the whole stage over the training set, while maintaining the desired detection rate. Viola and Jones (2001, 2004) used the four types of Haar-like filters shown in Figure 2 (a). The filters can take on arbitrary positions and sizes within an 24×24 image patch. The output of each filter

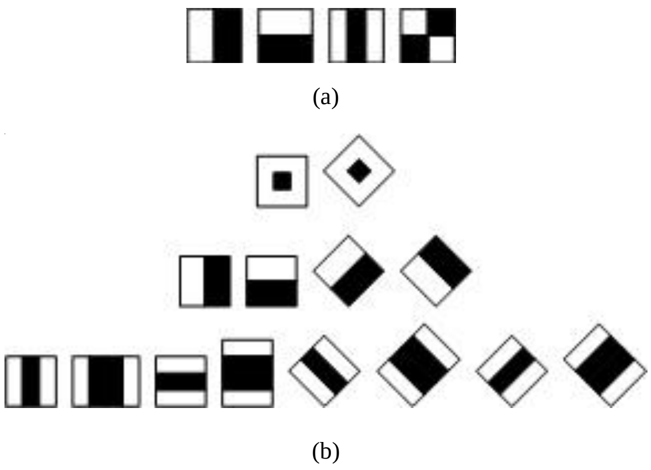


Figure 2. Haar-like features used to construct weak classifies in the boosted classifier cascade. (a) Features used by Viola and Jones. (b) Features used by Lienhart and colleagues.

is simply the difference between the average pixel value within the clear rectangular regions and the shaded rectangular regions.

Recently, Lienhart and Maydt (2002) modified Viola and Jones (2001, 2004) detector. Their system adds additional rotated Haar-like filter types, as shown in Figure 2 (b). On a particular test set, they found that their modified system gave 10% fewer false positives than the original system for certain detection rates. The empirical analysis of detection cascade of boosted classifier by Lienhart *et al.* (2002) compared DiscreteBoost (the algorithm used by Viola and Jones [2001]), RealBoost, and GentleBoost and found that classifiers trained with GentleBoost performed the best.

We apply Lienhart and colleagues' methods, as implemented in the OpenCV, Open Computer Vision Library (2006), to the hand detection problem, using all the filter types in Figure 2 (b), 24×24 image patches, and the GentleBoost learning algorithm.

Since boosting algorithms are supervised learning algorithms, a large number of labeled positive and negative examples must be input to the training process. Besides the examples, some learning parameters must be specified. The most important parameter is the desired true positive and false positive rate for each stage of the cascade. We train one stage at a time until that stage achieves the specified true positive and false positive rates. Then a new stage is begun, and the process continues until some stopping criterion is reached. Only the positive examples that are correctly classified by the previous stages and the negative examples that are incorrectly classified by the previous stages are used to train each new stage.

## 2.3 Skin detector

There are many approaches to segmenting regions with similar color and texture from other regions. To extract skin color blobs from images, color information is the obvious choice. The skin detector for our system need not be extremely robust but it should be fast.

The Bayesian maximum likelihood classifier based on color histograms, as presented by Zarit *et al.* (1999), meets all of these needs. Based on their results and our own follow-up study, we selected the HS (hue and saturation) color model. Histograms used in the skin detector have two dimensions, namely hue and saturation. Each axis of the plane is quantized into 16 bins, so that each histogram will have $16^2 = 256$ bins. We selected 16-bin quantization based on comparison experiments with different bins counts of 8, 16, 32, and 64. We found that 16 bins along each axis gave the best performance. The reasons for excluding the intensity component from the histogram are to eliminate the effect of non-uniform illumination and to save computational cost. We construct histograms for skin and non-skin pixels from a large training set. The histogram counts are used to construct a discrete class-conditional likelihood for a Bayesian maximum likelihood classifier which we then use to determine whether a given pixel is most likely skin or not skin.

Each image patch which is classified as a hand by the cascade is scaled to a standard size 24×24 pixels and then fed to the skin detector, which produces a binary image, in which the value 1 represents a putative skin pixel and the value 0 represents a non-skin pixel.

## 2.4 Features extractor and Mahalanobis classifier

The shape and relative size of the skin blob within the detection window give useful information for discriminating image patches containing hand from those not containing hand. We extract four simple features from the binary skin image that are surprisingly useful for accurate classification:

1. The area of the largest connected component of skin pixels.
2. The length of the perimeter of the largest connected component of skin pixels.
3. The eccentricity of the largest connected component of skin pixels.
4. The number of pixels on the boundary of the largest connected component of skin pixels that intersect the detection window boundary.

The area feature is simply the number of pixels in the largest connected skin component; it is normalized by the total number of skin pixels (24×24 = 576) in the image patch. It is very obvious that the given image patch is unlikely to contain a hand if the area feature is very large or very small. The perimeter feature is the total number of pixels on the perimeter of the largest connected skin component; it is normalized in the same way as the area feature. The eccentricity feature is the eccentricity of the ellipse having the same second moments as the largest connected skin component, i.e., the ratio of the distance between the foci of the ellipse and its major axis length. The eccentricity is between 0 and 1, with 0 indicating a circle and 1 indicating a line segment. This feature helps to discriminate face skin regions, which tend to be quite round, from true hand skin regions, which tend to be more eccentric. Finally, the boundary feature helps to discriminate between arm skin regions, which tend to intersect the boundary of the detection window in two places, from true hand skin regions, which only intersect the detection window at the wrist. The boundary feature provides information about how wrist-like the boundary is.

No matter how good those four features are, they will not be efficiently utilized for classification without a suitable classifier. We prefer classifiers that are simple with few parameters to tune. We find that a simple classifier based on Mahalanobis distance is a reasonable choice. Each image patch can be represented by a feature vector consisting of the area, perimeter, eccentricity, and boundary features. To classify a given feature vector  as a true hand or not a hand, we calculate the Mahalanobis distance

$$d(x) = (x - \mu)^T \Sigma^{-1} (x - \mu)$$

between the feature vector  and the mean feature vector $x$, then we classify  as a hand if $d(x)$ is less than some threshold $\theta$. Here the mean hand feature vector $\mu$, the covariance matrix $\Sigma$, and the distance threshold $\theta$ are estimated from the training set.

Once classification for each possible detection windows is done, the positively detected hands are fed to the final module, the grouping, filtering, and averaging module. Further reduction of false positives is done there.

## 2.5 Grouping, filtering, and averaging module

Our Mahalanobis classifier produces a few very sparsely distributed false positives and densely distributed true detections around the actual targets. Since it produces several true detections around each of the actual detections, grouping and averaging is necessary to ensure only one detection for each target. A group which contains less than some number of detections can be disposed of on the assumption it is a false positive. We use the existing implementation of this technique in the OpenCV. The positively detected hands output from this module could then be forwarded to another component in an integrated application, for example a gesture recognition module. But, in this article we simply evaluate the performance and efficiency of the proposed algorithm on a series of video sequences. We now describe our experiments in detail.

## 2.6 Data acquisition

For the purpose of training, testing and evaluation of the proposed hand detection system, we captured 12 video sequences in a moderately cluttered laboratory environment. Four people volunteered to be models, and we captured three video sequences for each person. In the first sequence, each model walked away from the camera then came back to the starting position, in a direction parallel to the camera angle. In the second sequence, each model walked back and forth across the field of view in a direction perpendicular to the camera angle, at three different distances from the camera. In the last sequence, each model walked diagonally across the field of view, starting from a position to the right or left of the camera then returned to the start position, and repeated the procedure beginning from the other side of the camera.

We captured the video sequences at 15 frames per second with an inexpensive IEEE1394 Web camera at a resolution of 640×480 pixels. Each sequence lasted approximately 30 seconds. After video capture, all visible hands not smaller than the standard size of 24×24 pixels in every image of all 12 sequences were manually located. A total of 2246 hand locations were obtained. Our criteria for locating the selection window on the hand was that the hand should be roughly at the center of the window while taking up about 50% of the pixel area of the selection window. Some examples are shown in Figure 3.

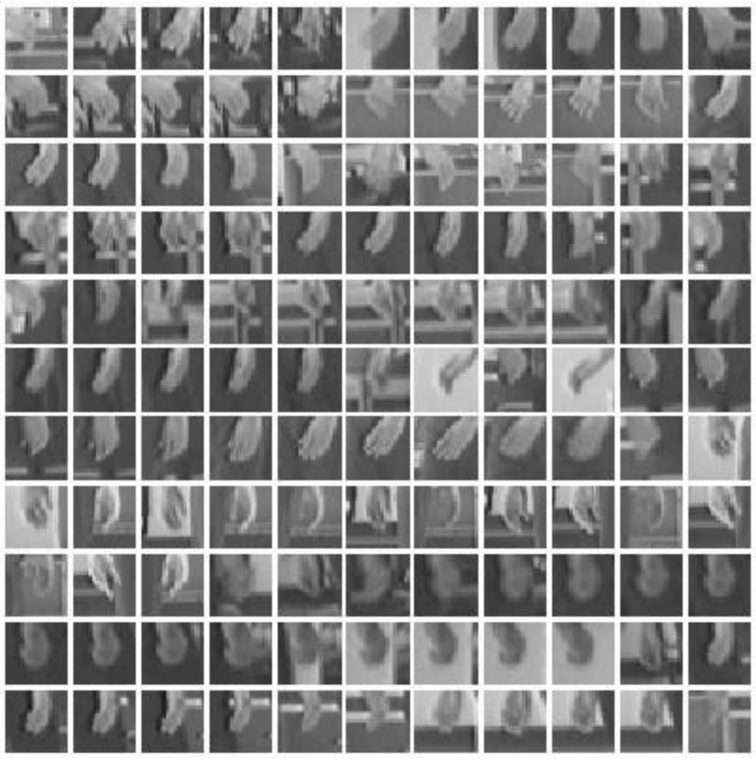


Figure 3. Example training images Scaled to 24×24.

Of the 12 video sequences, 11 were used to train the system and the remaining sequence was reserved for testing and evaluating the complete hand detection system. To train the boosted classifier cascade, we used 2000 hands as positive examples, and negative examples were automatically extracted from a set of background images. As background images, we used four randomly selected images from the video sequences that did not contain any human. We created an additional set of background images using six randomly selected images containing humans. From each image, we cut out two large regions that did not contain hands but did contain other body parts such as faces and arms. From the test image sequence, we selected 99 images, each containing at least one hand not smaller than 24×24 pixels. These 99 images contained a total of 106 proper hands. All of our test evaluation calculations are based on those 106 proper hands.

We also prepared a holdout set by randomly selecting 100 images from the 11 training video sequences. This holdout set was used to monitor system performance as well as to tune system parameters.

### 2.7 Boosted classifier training

To train the classifier cascade, we used Lienhart and colleagues' approach, implemented in OpenCV. We used the previously-described 2000 manually located hands from the eleven training video sequences as positive examples and the 16 previously-described background images.

The important parameters of the training process are the minimum hit rate (true positive rate) and maximum false alarm rate (false positive rate). Every stage in the cascade must satisfy these criteria on the training set. We used 100% for the hit rate and 60% for the false alarm rate. This means when adding a new stage to the classifier, the training system keeps adding additional weak classifiers to that stage until it correctly classifies all of the positive training examples with at most a 60% false alarm rate. Lienhart and colleagues' training system extracts the desired number of negative examples, 4000 for our experiment, by scanning a window with different scales over the background images. After training one stage of the classifier, the negative examples which are correctly classified are disposed of and the system extracts a sufficient number of new negative examples. We use the GentleBoost variant of AdaBoost and the full Haar-like feature set of Lienhart and Maydt (2002).

The performance of the cascade is tested on the hold-out set every time a new stage is constructed and added to the cascade. The results of the training process will be discussed in more detail in the Results and Discussion section.

### 2.8 Scanning window

When an image is presented to our hand detection system (Figure 1), a detection window is scanned over the image at multiple scales, and each resulting image patch is passed to the boosted classifier cascade. The scanning process is to begin with a 24×24 detection at the image's original scale. After every possible image patch at that scale has run through the classifier cascade, the image is scaled down by 90% and the process is repeated until a minimum image size (maximum detection window size) is reached.

### 2.9 Skin detector training

To train our skin detector, we selected 10 images containing one or more humans from a set of independent video sequences captured under various lighting conditions at several different locations. Skin pixels on those images were manually marked and the resulting 70,475 skin and 1,203,094 non-skin pixels were fed to the skin detector training process. The training process computes the hue (H) and saturation (S) for each pixel and quantizes each value into one of 16 bins. From the quantized values of skin pixels, one 2D histogram is constructed, and another is constructed from the quantized values of the non-skin pixels. Both histograms are constructed by simply counting the number of pixels which belong to same bin, and they are normalized by the total number of pixels used to construct the histogram.

### 2.10 Mahalanobis classifier training

The purpose of the Mahalanobis classifier is to eliminate the false detections made by the boosted classifier cascade while still maintaining a high detection rate. As the detection window is scanned over every image in the training set, the boosted classifier outputs both true positive and false positive image patches. We found 78,658 true positives on our training set then randomly selected 6,000 true positives for computing the mean feature vector $\mu$ and covariance matrix $\Sigma$ for the Mahalanobis classifier.

After we obtain $\mu$ and $\Sigma$ for the Mahalanobis classifier, we need to find the optimum threshold. To do so, we scanned a detection window over every image in the holdout set and separated the detected image patches into false positives and true positives using the known hand locations for the holdout set. We extracted the Mahalanobis classifier's four features from each detected image patches and calculated the Mahalanobis distance between the feature vector of each image patch and the mean feature vector $\mu$. As the class for each image patch is known, we plotted the ROC curve as shown in Figure 4. At this point, a detection rate of less than 100% is acceptable because the classifier cascade typically produces multiple true detections around each hand. Examining the ROC curve, we found that a Mahalanobis distance of 2.9 is a reasonable threshold since this threshold gives a very low false positive rate (6%) while giving an acceptable true positive rate (60%) on the image patches output by the classifier cascade.

### 2.11 Parameter tuning for the system

Once all the required building blocks for hand detection are in place, we need to specify one last parameter, i.e., the minimum number of nearby positive image patches required for the Group, Filter, and Average block. In practice,

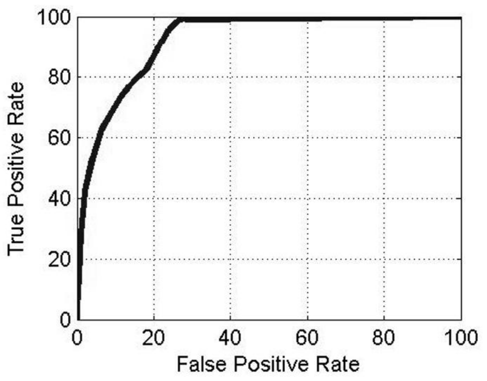


Figure 4. ROC curve between true positive and false positive for different thresholds of Mahalanobis distance. True positive and false positive rate are calculated based on number of true detection and false detection input to the Mahalanobis Classifier.

this parameter must be tuned to achieve a good detection rate. To tune this parameter, we assembled all of the building blocks into a complete system then tested it on the holdout set with various values for each parameter. We found that rejecting detection with less than 4 neighboring patches per group produced the optimal result: 81.8% of the hands in the holdout set were detected and the false positive rate was also relatively low, an average of 1.55 false positives per image.

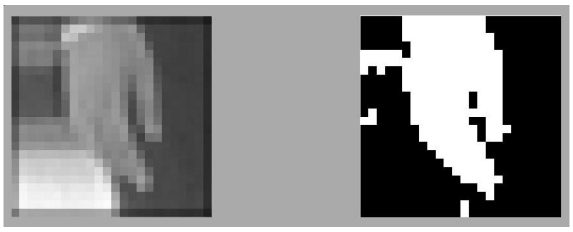
### 2.12 Testing the complete system

We tested the complete hand detection system on the test set that was never used in any part of the training process. As previously described we used 99 images containing 106 hands in known locations. The detailed results of the test are discussed in the next section.

## 3. Results and Discussion

During the training process, we monitored the performance of the cascade and found that 12 stages of strong classifiers gave the optimum performance. The 12-stage classifier had a 97.5% detection rate on the holdout set, while having a reasonably low false positive rate of about 0.3% on the holdout set. A false positive rate of 0.3% may seem quite low but in fact this means we had an average of 1,000 false positive detections per image because one image contains more than 300,000 possible image patches. Clearly, these results indicate that a post processor is necessary to further eliminate false positives if the system is to be usable in practical applications.

When we tested our system on the test set, we found that the boosted classifier cascade frequently detected incorrect body parts such as arms, as shown in the left half of Figure 5 (b). However, the skin detection images shown in the right halves of Figures 5 (a) and 5 (b) show that the Mahalanobis classifier's boundary feature can distinguish


(a) Properly detected hand.

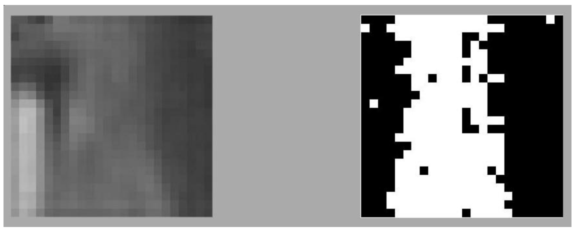
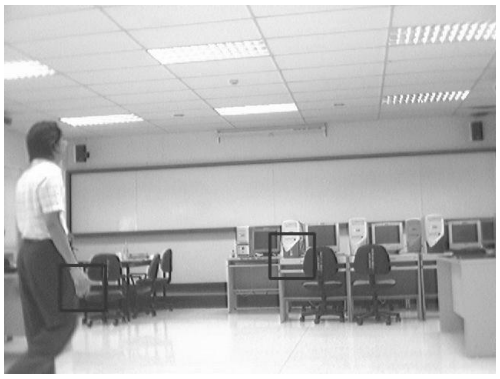

(b) Non-hand body part detected as hand.

Figure 5. Original (left) image patches detected as hand by boosted hand classifier and binary images patches (right) after skin detection.


(a)

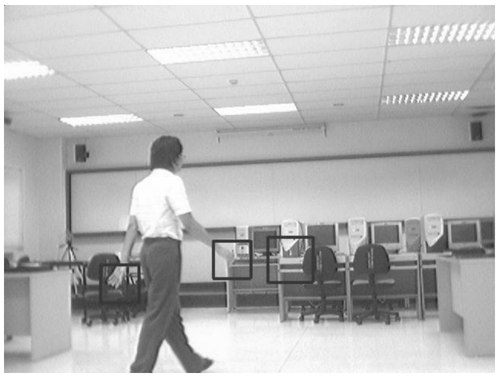

(b)

Figure 6. Example detection results of our proposed hand detection system.

between these cases. We found that most of the remaining false positives contained either too few skin pixels or sparsely distributed skin pixels. These cases are easily eliminated by the Mahalanobis classifier's area feature since it operates on the largest connected skin component.

Our final hand detector detects 92 hands (86.8%) of the 106 hands in the test set, with an acceptable false positive rate of 1.19 false detections per image on average. Table 1 compares the performance of the original boosted classifier cascade of Lienhart and Maydt (2002) to that of our hand detector system on six consecutive frames from the test set. The average false positive rate drops from about 1,000 per image to 1.19 per image at the cost of a drop in true positive rate from 99.1% to 86.8%. Post-processing by the Mahalanobis classifier transforms a completely unusable classifier into a high-precision hand detector. A detection rate of 86.8% will enable many applications based on hand detection, such as human action recognition systems for security.

Images (a) and (b) in Figure 6 illustrate example detections by our complete system, and all detected hands in the test set are shown in Figure 7. In the example, all hands were detected in both images, and only one false detection occurred in each image. The false detection of the desktop computer in the middle of the image is present in almost every image because the computer's color and texture are in fact similar to that of a hand. This kind of false positive

Table 1. Performance comparison between a system with only the boosted classifier cascade and our complete hand detector system in terms of the number of false positive detections per image. Six consecutive frames from the test set were selected for the comparison.

| Image name | Number of false positive from system with only boosted classifier cascade | Number of false positive from our system (boosted classifier cascade + post processor) |
|---|---|---|
| Image 1 | 749 | 1 |
| Image 2 | 762 | 3 |
| Image 3 | 811 | 1 |
| Image 4 | 933 | 1 |
| Image 5 | 1047 | 1 |
| Image 6 | 978 | 1 |

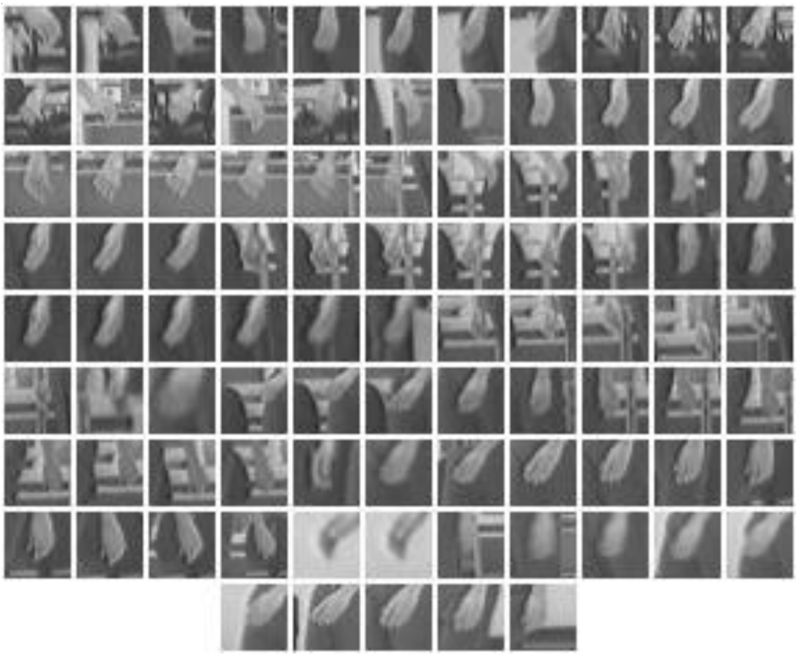


Figure 7. Hands detected by our complete hand detector system. All detections are scaled down to standard size 24x24 pixels for easy visualization.

detection on a stationary object will be eliminated if we add motion information between two consecutive frames in the video sequence.

## 4. Conclusion

From the literature, we know that hand detectors incorporating AdaBoost and Haar-like features perform quite well in applications like sign-language recognition, in which images are relatively high resolution with less cluttered background and constrained hand gesture. These approaches suffer from high false positive rates and low detection rates when applied to detect less constrained hands in low resolution and cluttered images. However, we find that these limitations can be overcome with the help of a simple but efficient post processing system – in our experiments, the prototype hand detection system achieved excellent performance on its test set. One important limitation of this work is that both the training and testing image sequences were captured in the same environment. This means that the performance of our current system is likely background dependent; if so, the reported performance is optimistic. However, the current results are encouraging, and in future work we plan to explore integrating our system with gesture recognition and human action recognition systems.

## Acknowledgement

We thank the member of the Image and Vision Computing Laboratory at the Sirindhorn International Institute of Technology for the participation in our data collection efforts. This research was partially supported by Thailand Research Fund grant MRG4780209 to Matthew N. Dailey.

## References

Barreto, J., Menezes, P. and Dias, J. 2004. Human-robot interaction based on haar-like features and eigenfaces. Proceedings of the 2004 IEEE Conference on Robotics and Automation, 2004, 1888-1893.

Caglar, M.B. and Lobo, N. 2006. Open Hand Detection in a Cluttered Single Image using Finger Primitives. Proceeding of the 2006 Computer Vision and Pattern Recognition Workshop, June 17-22, 2006.

Freund, Y. and Shapire, R.E. 1997. A decision-theoretic generalization of online learning and an application to boosting. Journal of Computer and System Sciences 5(1), 119-139.

Intel Corporation. Open Computer Vision Library (software). 2006. Open source software available at http://sourceforge.net/projects/opencv/.

Lienhart, R. and Maydt, J. 2002. An extended set of Haarlike features for rapid object detection. Proceedings of the IEEE International Conference on Image Processing, 2002, 900-903.

Lienhart, R., Kuranov, A. and Pisarevsky, V. 2002. Empirical analysis of detection cascades of boosted classifiers for rapid object detection. Technical report, Microprocessor Research Lab, Intel Labs.

Ong, E. and Bowden, R. 2004. A boosted classifier tree for hand shape detection. Proceedings of the Sixth IEEE International Conference on Automatic Face and

Gesture Recognition, 2004, 889-894.

Shamaie, A. and Sutherland, A. 2005. Hand tracking in bimanual movements. Image and Vision Computing 23(13), 1131-1149.

Soontranon, N., Aramvith, S. and Chalidabhongse, T.H. 2004. Face and hands localization and tracking for sign language recognition. International Symposium on Communications and Information Technologies, 2004, 1246-1251.

Varona, J., Buades, J.M. and Perales, F.J. 2005. Hands and face tracking for VR applications, Computers & Graphics. 29(2), 179-187.

Viola, P.A. and Jones, M.J. 2004. Robust real-time face detection. International Journal of Computer Vision 57(2), 137-154.

Viola, P.A. and Jones, M.J. 2001. Rapid object detection using a boosted cascade of simple features. IEEE Conference on Computer Vision and Pattern Recognition, 2001, 511-518.

Wachs, J., Stern, H., Edan, Y., et al. 2005. A real-time hand gesture system based on evolutionary search. Genetic and Evolutionary Computation Conference, 2005.

Wren, C.R., Azarbayejani, A., Darrell, T. and Pentland, A. 1997. Pfinder: Real-time tracking of the human body. IEEE Transactions on Pattern Analysis and Machine Intelligence 19(7), 780-785.

Zarit, B.D., Super, B.J. and Quek, F.K.H. 1999. Comparison of five color models in skin pixel classification. International Workshop on Recognition, Analysis and Tracking of Faces and Gestures in Real-Time Systems, 1999, 58-63.