A Boosted Classifier Tree for Hand Shape Detection

Eng-Jon Ong and Richard Bowden
Centre for Vision, Speech and Signal Processing
University of Surrey, Guildford, GU2 7XH
e-mail: r.bowden,e.ong@eim.surrey.ac.uk

Abstract

The ability to detect a persons unconstrained hand in a natural video sequence has applications in sign language, gesture recognition and HCI. This paper presents a novel, unsupervised approach to training an efficient and robust detector which is capable of not only detecting the presence of human hands within an image but classifying the hand shape. A database of images is first clustered using a k-mediod clustering algorithm with a distance metric based upon shape context. From this, a tree structure of boosted cascades is constructed. The head of the tree provides a general hand detector while the individual branches of the tree classify a valid shape as belong to one of the predetermined clusters exemplified by an indicative hand shape. Preliminary experiments carried out showed that the approach boasts a promising 99.8% success rate on hand detection and 97.4% success at classification. Although we demonstrate the approach within the domain of hand shape it is equally applicable to other problems where both detection and classification are required for objects that display high variability in appearance.

1. Introduction

This paper presents a framework capable of both detecting a hand and its corresponding shape efficiently and robustly without constraints upon either user or environment. This has long been an area of interest due to its obvious uses in areas such as sign and gesture recognition to name but just two. The hand is a complex object. In addition to its rigid transformation it has 14 joints which means that the number of possible configurations, and therefore the variability of its 2D projection onto the image, make its detection and/or classification a non-trivial task. To this end many researchers have constrained the problem through the use of fixed or known backgrounds [11], coloured gloves [8], limited movement or markers [4]. To avoid such constraints many researchers use colour to find coherent skin regions. The approach relies on the fact that human skin is relatively uniform and can often be modelled as simply as with a single gaussian [11]. Although more sophisticated colour models can be used [12], colour alone generates ambiguity between the hands and other, similarly coloured objects, most notably the face. Gong et al overcame this through the fusion of colour and motion in a Bayesian network [7] with promising results. However, our approach to detecting the location of the hands uses a boosted cascade of classifiers [9] to detect shape alone in grey scale images. The obvious addition of colour may benefit this approach but with detection rates as high 99.8% it is unclear what these benefits would be.

Our detector is based upon a tree structure of boosted cascades of weak classifiers. The head of the tree forms the general hand detector and its sole purpose is to find all possible hand hypotheses in the image. Successful hypotheses are then passed onto the branches of the tree where specific cascades designed only to detect hands of a specific shape are used to determine the exact pose of the hand in the image. To build shape specific detectors the data set must be broken up or clustered into similar shapes that are specific, yet contain sufficient variation in shape to allow the classifier to generalise. To do this we perform a k-mediod clustering on the training data using shape context as a distance metric to asses shape similarity.

The remainder of this paper is structured thus. In Section 2 we propose the use of boosting methods for learning a tree of hand detectors. Section 3 discusses how unsupervised learning is performed on a database of hand images to cluster them into distinct groups based on their shape appearance. Following this, we briefly describe in Section 4 how the final classifier is used to efficiently and robustly detect a range of different hand shapes. We then provide some results in Section 5 before drawing our conclusions in Section 6.

2. Learning through Boosting

Boosting is a general method that can be used for improving the accuracy of a given learning algorithm [5].

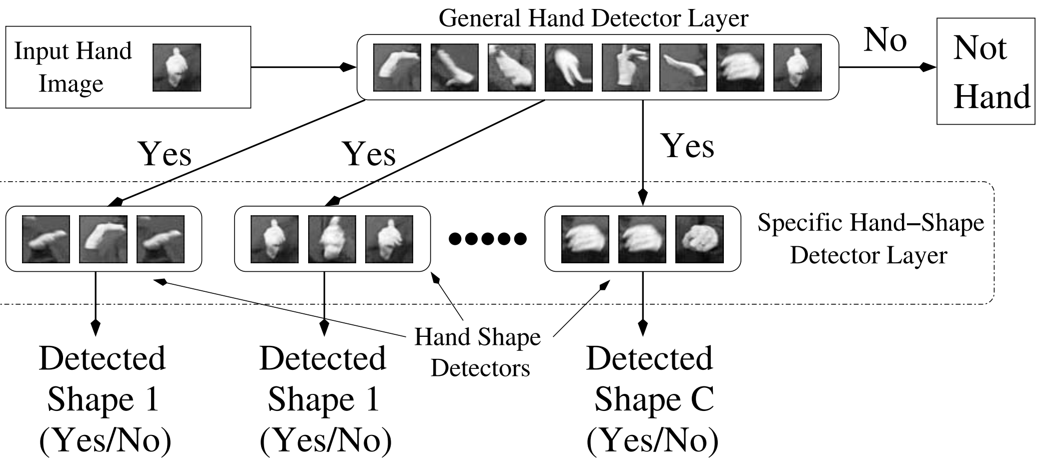
General Hand Detector Layer
Input Hand Image
No
Not Hand
Yes
Yes
Yes
Specific Hand–Shape Detector Layer
Hand Shape Detectors
Detected Shape 1 (Yes/No)
Detected Shape 1 (Yes/No)
Detected Shape C (Yes/No)

Figure 1. The framework of a tree of hand detectors.

More specifically, it is based on the principal that a highly accurate or "strong" classifier can be produced through the linear combination of many inaccurate or "weak" classifiers. In general, the performance of an individual weak classifier may be only slightly better than random. In the context of this paper, a weak classifier is a simple detector based on basic image block differences efficiently calculated using an integral image, as proposed in [9](for details, see Appendix B). Formally, given an image ($\mathbf{x}$), a strong classifer ($H_M(\mathbf{x})$) can be defined as a signed linear combination of the outputs of a number (M) of weak classifiers (h_m):

$$H_M(\mathbf{x}) = sign(\sum_{m=1}^{M} h_m(\mathbf{x})) \tag{1}$$

Here, we require the output of H_M to be postive for a hand detection and zero or negative for a non-hand detection.

2.1. AdaBoost

The AdaBoost algorithm that was introduced by Fruend and Schapire [3] providing a learning solution for finding suitable collections of weak classifiers. In order to perform training, an exponential loss function modelling the *upper bound* of the training error is used:

$$J(H_M) = \sum_{i}^{N_H} e^{-y_i H_M(\mathbf{h}_i)} \tag{2}$$

For us, we have a training set ($\mathbf{h}_i$) of (N_H) images that consists of hands and non-hands. The hands and non-hand images are associated with labels (defined as y_i)+1 and -1 respectively. In this algorithm, weak classifiers are added sequentially into an existing set (H_M) of (M) other weak classifiers such this upper bound is decreased. It is known that this lowering is possible if weak classifiers of the form shown in Appendix B is chosen[2, 6].

AdaBoost was applied to the area of face detection and recently pedestrian detection by Viola *et al*[9, 10] with impressive results. However, this method may result in an excessive number of weak classifiers in each strong classifier layer in the final detector cascade. It does not consider the removal of existing weak classifiers that no longer contribute to the detection process.

2.2. FloatBoost

To address the issue above, the FloatBoost algorithm was proposed by Li *et al*[13]. This algorithm essentially adds an additional step into the original AdaBoost which removes an existing weak classifier from a strong classifier if it no longer contributes positively to the training error. This removal of redundancy results in a smaller and therefore more efficient set of weak classifiers in each strong classifier layer. The FloatBoost algorithm itself is given in Appendix A.

2.3. Tree of Detectors

Classification can be made more feasible by dividing the final detector into a *cascade* of strong classifier layers[9]. Taking this idea further, in attempting to detect a hand which has greater variability in appearance, a tree structure of detectors can be used as initially proposed in [13] for face detection. The tree structure consists of a general detector sitting at the first layer, with branch nodes of increasingly more appearance-specific detectors at deeper layers (see Figure 1). These deeper layer nodes are trained with a smaller and more specific set of images. However, this raises the issue of the need for a labelled database of images. For faces, the pose can be extracted and images grouped accordingly. However, it is less straightforward for grouping hand images.

An automatic method for performing such a grouping would be highly advantegous as as boosting approaches typically deal with databases of thousands of images. Performing such a task manually would be prohibitively time consuming, since the hand is capable of assuming many different shapes. We therefore use unsupervised clustering to group similar hand shapes using shape context and the k-mediod algorithm as described in Section 3.

3. Grouping Hand Shapes

A training set of 5013 hand images such as those shown in Figure 3 were automatically segmented from various video sequences. They were extracted by modelling skin tones with a single Gaussian and extracted by finding regions of high skin probability. Due to this scheme, the hand images in our database contain a certain amount of variability in translation, scale and occasionally other "noisy features" arise (e.g. the face).

The training images must now be grouped into sets of hand images that share common appearance. Lockton and Fitzgibbon [4] grouped perfectly aligned hands using binary images, a greedy clustering algorithm and a distance metric based upon binary correlation. However, this approach will not work here due to the aforementioned variability in alignment. We therefore represent the hand with a set of shape context features[1]. These features have the advantage that they are robust to noise and alignment.

With the above shape context features, we employ K-mediod to group hand images according to their shape. However, one issue with this method is its sensitivity to the initial settings of the cluster centres. To address this, we propose an initialisation method that aims to extract initial cluster centres that are separated as far from each other as possible in our metric shape context space.

3.1. Shape Context

We have chosen to represent hands images with shape context features proposed initially by Belongie *et al.*[1]. In shape context analysis, a number (N_P) of samples are extracted from the contour or edges of a shape. There are no restrictions to what type of contour these points belong to, nor do they need to be regularly spaced.

Using this set of points, we can construct a set of compact shape descriptors in the form of histograms based on a log-polar coordinate system (Figure 2a). Each histogram is centred on these points (Figure 2b). A visualisation of the entire set of log-polar histograms for two hands can be seen

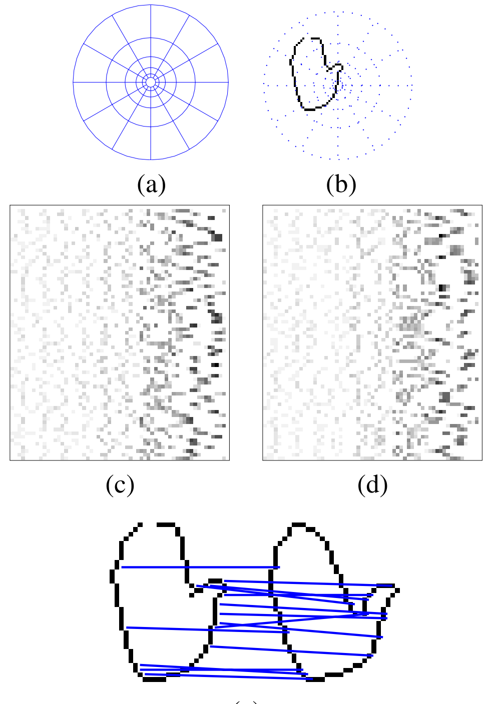

Figure 2. Shape context components: (a) the log polar histogram used, (b) shows it centered on a point on a hand contour. (c) and (d) visualisations of the set of log polar histograms for two hand contours. (e) some correspondances between points on two hand contours using the shape context metric.

in Figure 2c and 2d. In this image, each row is an "unrolled" normalised log-polar histogram. Histogram values closer to 1 are dark, whilst those closer to 0 are brighter. The distance measure or *cost* between two K-bin normalised histograms, $e(k)$ and $f(k)$, is obtained using the χ^2 distance:

$$\chi^2(e(k), f(k)) = \frac{1}{2} \sum_{k=1}^{K} \frac{(e(k) - f(k))^2}{e(k) + f(k)} \tag{3}$$

We can now build a cost matrix (E) for two shapes, where the elements ($E_{i,j}$) is the cost between the log-polar histograms of the i^{th} and j^{th} point of the first and second shapes respectively. Using this cost matrix, we can then calculate the optimal association between every point in the first shape to that of the second shape using the hungarian method. This results in a permutation $\pi(i)$, where the sum $d = \sum_k C_{k,\pi(k)}$ is minimum. The value d will also be used as a matching cost between two shapes. An example of some correspondences given by this permutation can be seen in Figure 2e.

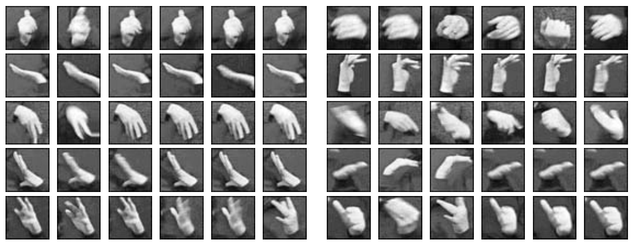

Figure 3. The result of the hand shape clustering algorithm given in Section 3. 6 samples from 10 hand image clusters are shown.

With the cost value given above, we define the N_HxN_H *database cost matrix* (D) that will contain the costs of every example to every other example. Therefore, the elements of D can be defined as

$$D_{i,j} = d(\mathbf{h}_i, \mathbf{h}_j) | i = \{1, ..., N_H\}, j = \{1, ..., N_H\} \quad (4)$$

3.2. Initialising of K-mediod Cluster Centres

We define the number of clusters as N_C. Each image cluster ($\mathbf{c}_i$) is represented as a set of C_i number of *indices* to the database images; ($\mathbf{c}_i = \{c_{i,1}, ..., c_{i,C_i}\}$). The cluster centre is represented as its first element ($c_{i,1}$), or the image $\mathbf{h}_{c_{i,1}}$. This has the effect of restricting the cluster centres to only lie on the training images available. We will address any issues that may arise because of this in the next section. We define the entire set of clusters as $\mathbf{C} = \{\mathbf{c}_1, ..., \mathbf{c}_{N_C}\}$.

We make use of the database cost matrix D to perform the initialisation. However, since we wish to preserve its original values, we make a copy $E = D$. For the following algorithm, we denote the i^{th} row vector of the matrix E as $E(:, i)$. The initialisation algorithm is as follows:

```
1: $c_{1,1} = 1$
2: $C_1 = 1$
3: $E(:, k) = 0, \forall k | E(c_{1,1}, k) < \beta$ {Remove any examples that look like the first image, alikeness is defined by value $\beta$, which is defined heuristically.}
4: for i = 2 to $N_C$ do
5:     $c_{i,1} = \max_j E(c_{i-1,1}, j)$
6:     $C_i = 1$
7:     $E(:, k) = 0 | E(c_{i,1}, k) < \beta$ {Again, remove any examples that resemble the image with index $c_{i,1}$}
8: end for
```

3.3. K-medoid Clustering

The k-medoid clustering algorithm is similar in nature to the K-means clustering algorithm in that it will associate to each cluster the set of images in the training hand database that most closely resembles the cluster centre. The main difference being it allows clustering within our metric space. Here, the distance metric between the cluster centre and the images is given by the shape context cost. At every iteration of the algorithm, cluster centres are updated by taking the example whose cost to all the other members of the set is the smallest.

Some results of applying the clustering algorithm to our hand image training database of 5013 images can be seen in Figure 3. There are times where there exists dissimilar images in a cluster, as can be seen at the bottommost right set of hands in Figure 3. This is mainly due to the shortcomings of solely using the log polar histograms for determining the difference between two hand images. However, more often than not, as can be seen, images that are relatively similar are grouped into the same clusters. Having separated the hand images into their own distinctive shape groups, we now attempt to learn classifiers that will both detect and recognise them.

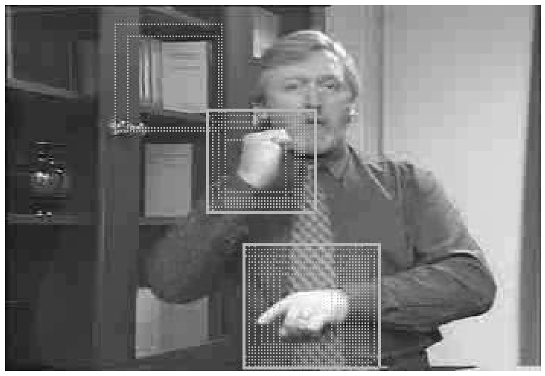

Figure 4. The result of the hand detector and the subsequent connected components analysis as described in Section 4.

4. Hand Shape Detection

In order to detect hands in an image, we first perform an exhaustive detection across all possible positions and scales. While this may sound very computationally taxing, we note that a majority of the positions and scales would not contain hands. The structure of the detector cascades means many parameterisations will be rejected in the first few layers of the top strong classifier, which require only a very small amount of computation.

This results in a significant amount of detections in the image area where hands lie, shown as dashed boxes in Figure 4. We exploit this to build a rough cumulative image that highlights areas that have many detections. The cumulative image is then thresholded to remove weak erroneous detections in the background. Finally, a connected components analysis is performed on the thresholded image to detect the size and positions of the hands. The final result can be seen as boxes with solid lines in Figure 4. We also note that the false detections in the background were rejected by this method.

In order to detect the hand shape, the sub-images in the areas where the hand was detected are given to the set of hand shape detectors on the second layer. We choose the shape corresponding to the detector that has the highest output. The results of the shape detection can be seen in the next section.

5. Experiments

5.1. Hand Detector

For our experiments, we gathered a total of 5013 hand images from various video sequences of different people signing. We selected 2504 examples for training and similarly 2509 hand images from different sequences were retained for testing. For the general hand detector, we then trained a cascade of 11 layers with a total of 634 weak classifiers. We have found that the detection error on the test database was 0.2% which is surprising due to the variability of the data. Some of the results of the hand detector ran over the test data sequences can be seen in Figure 5.

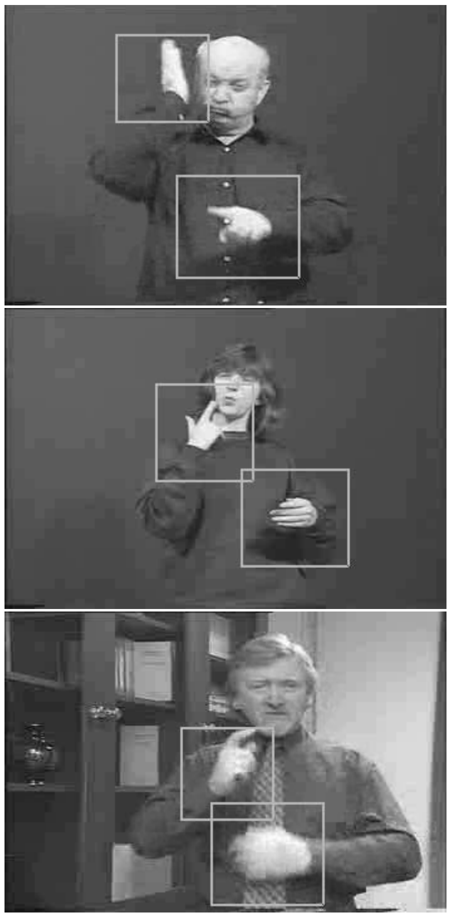
Figure 5. The results of the hand detector.

5.2. Shape Detector

To train the shape detector branches of the tree layer, we then combined both the training and test database together. This combined database was then split into 300 different shape clusters using the method described in Section 3. We then reserved a total of 900 unseen images across different clusters for our shape test database. A cascade of strong classifiers was then trained on the images of each cluster. The images in the remaining clusters provided the "non-hand-shape" images. We found that the average error for these detectors on the shape test database was 2.6%.

6. Conclusion and Future Work

In this paper, we have proposed a novel and unsupervised approach to training an efficient hand-shape detector in grey-level images. Apart from the image and the detectors, no other forms of information such as motion or background models were used. Firstly, a database of hand images was clustered into sets of similar looking hands using the k-mediod clustering algorithm that using a distance metric based on shape context. A tree of boosted hand detectors was then formed, consisting of two layers, the top layer for general hand detection, whilst branches in the second layer specialise in classifying the sets of hand shapes resulting from the unsupervised clustering method. We then tested the detector with an unseen database of 2509 images and the shape classifier with an unseen database of 900 labelled hand shapes. With this database, we found that the detector and classifier both had an unexpectedly high success rate of 99.8% and 97.4% respectively. However, we note that the hand images for both training and test databases have fairly simple and similar backgrounds. Future work will concentrate on evaluating the accuracy for detecting and classifying hand shapes in environments with more clutter and variability (e.g. outdoor). Additional future work includes the combination of these detectors with other forms of information, for example colour, motion or background models.

A. FloatBoost Algorithm

We merge hand and non-hand images into a database of N images, $\{\mathbf{h}_1, ..., \mathbf{h}_N\}$. Hand and non-hand images are associated with the label (y_i) +1 and -1 respectively. A weight

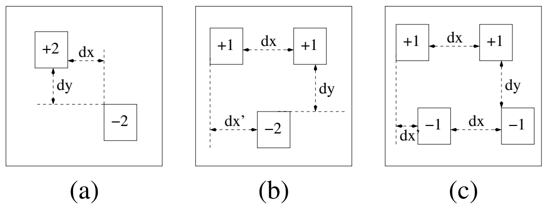

Figure 6. The three types of block differences that is used. The weights of the block differences are given by the numbers in the rectangles. The size for the rectangles that we have used is 4x3. The distances between these rectangles are dx, dy and dx'.

(w_i) is attached to every training example $(\mathbf{h}_i)$. The FloatBoost algorithm for training a layer with at most M_{max} weak classifiers is as follows[13]:

```
1: Intialisation Step:
   (i)$M = 0$, $H_0 = \{\}$;
   (ii)Set $w_i = \frac{1}{2N_H}, \forall y_i = 1$ and $w_i = \frac{1}{2N_G}, \forall y_i = -1$;
   (iii)Initialise the minimum error list:
   $\epsilon_m^{min} = max\_number\_val(\forall m = 1, ..., M_{max})$.
2: Forward Inclusion (adding a new weak classifier)
   (i)$h_{M+1} = \arg\min_{h^{\oplus}} J(H_M(\mathbf{x}) + h^{\oplus}(\mathbf{x}))$;
   (ii)Update $w_i = \exp(H_M(-y_i\mathbf{h}_i))$ and normalise;
   (iii)$M = M + 1$; $H_M = H_{M-1} \cup \{h_M\}$;
   (iv)If $\epsilon_M^{min} > \epsilon(H_M)$, then $\epsilon_M^{min} = \epsilon(H_M)$
3: Conditional Exclusion (removing weak classifiers)
   (i) $h' = \arg\min h \in H_M \epsilon(H_M - h)$;
   (ii)If $\epsilon(H_M - h') < \epsilon_{M-1}^{min}$, then
       (iii)$\epsilon(H_M - h') = \epsilon_{M-1}^{min}$;
       (iv)$M = M - 1$;
       (v)$H_M = H_{M+1} - h'$; goto 3.(i)
   (vi)Else
       (vii)if $M = M_{max}$ or $J(H_M) < LowestJ$, goto 4
       (viii)Re-estimate weights (see Step 2.(ii)),goto 2.(i)
4: Strong Classifier Result
   (i)$H_M(\mathbf{x}) = sign[\sum_{h(x) \in H_M} h(\mathbf{x})]$
```

B. Weak Classifiers

To start, Harr wavelet like features[13] provided scalar hand image feature values. These take the form of disjointed image block differences, which can be computed efficiently using the integral image[9]. Three types of block differences are used (see Figure 6).

Two probability densities of the block difference values are then built to produce the weak classifier:

$$h(x) = 0.5(\frac{p(z|y = +1, w)}{p(x|y = -1, w)} - T) \qquad (5)$$

where w are the set of weights associated with each example at the time where this classifier was built, z is the value of the block differences, and T is the threshold that determines how easy it would be for the classifier to classify positive examples. which we heuristically set as 0.3. Both the probability distributions $p(z|y = +1, w)$ and $p(z|y = -1, w)$ are represented as normalised histograms for the different block difference values, built with the set of weight values of the examples, w.

References

[1] S. Belongie, J. Malik, and J. Puzicha. Shape context: A new descriptor for shape matching and object recognition. In *NIPS*, pages 831–837, 2000.

[2] J. Friedman, T. Hastie, and R. Tibshirani. Additive logistic regression: A statistical view of boosting. *The Annals of Statistics*, 28(2):337–374, 2000.

[3] Y. Fruend and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of Computer and System Sciences*, 55(1):119–139, 1997.

[4] R. Lockton and A. Fitzgibbon. Real-time gesture recognition using deterministic boosting. In *Proc. of British Machine Vision Conference*, 2002.

[5] R. Schapire. The boosting approach to machine learning: An overview. In *Proc. of MSRI Workshop on Nonlinear Estimation and Classification*, 2002.

[6] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. In *Proc. of the 11th Annual Conference on Computational Learning Theory*, pages 80–91, 1998.

[7] J. Sherrah and S. Gong. Continuous global evidence-based bayesian modality fusion for simultaneous tracking of multiple objects. In *Proc. IEEE International Conference on Computer Vision*, 2001.

[8] T. Starner and A. Pentland. Real-time american sign language recognition from video using hidden markov models. In *Proc. of SCV95*, 1995.

[9] P. Viola and M. Jones. Robust real-time object detection. In *Proc. of IEEE Workshop on Statistical and Computational Theories of Vision*, 2001.

[10] P. Viola, M. Jones, and D. Snow. Detecting pedestrians using patterns of motion and appearance. In *Proc. of IEEE International Conference on Computer Vision*, pages 734–741, 2003.

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

[12] S. M. Y. Raja and S. Gong. Tracking and segmenting people in varying lighting conditions using colour. In *Proc. IEEE International Conference on Automatic Face and Gesture Recognition*, 1998.

[13] Z. Zhang, M. Li, S. Li, and H. Zhang. Multi-view face detection with floatboost. *Proc. of the Sixth IEEE Workshop on Applications of Computer Vision*, 2002.