Viewpoint-Invariant Learning and Detection of Human Heads

M. Weber[†] W. Einhäuser[§] M. Welling[‡] P. Perona[†‡]

California Institute of Technology
[†]Dept. of Computation and Neural Systems
[‡]Dept. of Electrical Engineering
Caltech, MC 136-93
Pasadena, CA 91125, U.S.A.
{mweber, welling, perona}@vision.caltech.edu

[§]Universität Heidelberg
Kirchhoff-Institut für Physik
Schröderstr. 90
D-69120 Heidelberg, Germany
weinhaeu@ix.urz.uni-heidelberg.de

Abstract

We present a method to learn models of human heads for the purpose of detection from different viewing angles. We focus on a model where objects are represented as constellations of rigid features (parts). Variability is represented by a joint probability density function (pdf) on the shape of the constellation. In a first stage, the method automatically identifies distinctive features in the training set using an interest operator followed by vector quantization. The set of model parameters, including the shape pdf, is then learned using expectation maximization. Experiments show good generalization performance to novel viewpoints and unseen faces. Performance is above 90% *correct with less than* 1*s computation time per image.*

1 Introduction and Related Work

The human head and face are the most valuable objects that a computer vision system may detect, track and recognize. Amongst these tasks, detection is perhaps the most challenging; while recognition and tracking have registered considerable progress during the last decade, detection has so far eluded the efforts of computer vision researchers. The main source of hope comes from observing the human visual system; it can detect reliably and quickly human heads in clutter: independently of scale, viewpoint, in a large variety of lighting conditions, and robustly with respect to occlusion.

A number of studies have addressed detection in simplified scenarios: Sung and Poggio [8], Baluja, Rowley and Kanade [5], have proposed neural-network approaches to detecting unoccluded frontal views of the face. Schneiderman and Kanade [7] have proposed an approach based on histograms of feature detectors to address the same problem. Burl, Leung, Weber and Perona [1] additionally address the issue of occlusion. All these systems use supervised training where the training examples are normalized and aligned by hand. In the system by Burl et al. an operator indicates the main features of the faces by clicking on them with a mouse.

It is difficult to assess and compare the performance of these systems since no common benchmark exists. Baluja et al. made their system available for testing on the internet [6]. On realistic datasets the detection rates of these systems is above 80% of frontally viewed faces, with false alarm rates in the range of $0 - 10$ in clutter. The computational time is implementation-dependent and ranges from 1 to 100 seconds per image. Training on large datasets ranging from 10^2 to 10^4 images is required.

We address here the problem of detecting human heads in clutter idependently of their orientation around the vertical axis. We seek to achieve this invariance while preserving the robustness to occlusion of Burl et al. and the mild training supervision required by the neural-network approaches. Our starting point is the scheme proposed by Burl et al. which we extend along two directions: (a) we obtain viewpoint invariance using two distinct methods: training a single detector on multiple views, and combining the output of multiple detectors trained on different views, (b) we weaken the training paradigm requiring no image normalization, registration or manual detection of features: We develop ideas for automatic feature selection and shape training.

2 Overview of the Approach

We model object classes following the work of Burl et al. [1]. An object is composed of *parts* and *shape*, where *parts* are image patches which may be detected and characterized by appropriate detectors, and *shape* describes the geometry of the mutual position of the parts in a way that

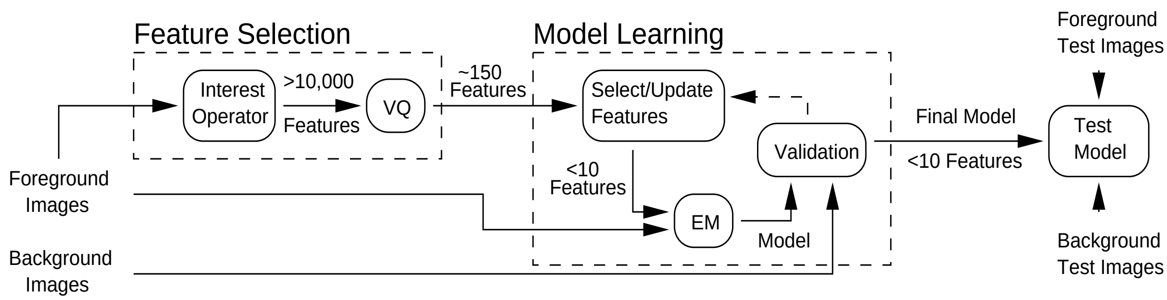

Figure 1. Overview of our method

is invariant with respect to rigid and, possibly, affine transformations [4]. A joint probability density on part appearance and shape models the object class. Object detection is performed by first running part detectors on the image, thus obtaining a set of candidate part locations. The second stage consists of forming likely object hypotheses, i.e. constellations of appropriate features (e.g. eyes, nose, mouth, ears); both complete and partial constellations are considered, in order to allow for partial occlusion. The third stage consists of using the object's joint probability density for either calculating the likelihood that any hypothesis arises from an object (object detection), or the likelihood that one specific hypothesis arises from an object (object localization). In order to train a model we need to decide on the key parts of the object, identify those parts in the training images and estimate the joint probability density function on part appearance and shape. The method we present here performs all three steps automatically. A block diagram is shown in Fig. 1.

Our technique for selecting potentially informative features/regions is composed of two steps: first highly textured regions are detected in the training images by means of a standard *interest operator* or keypoint detector. The number of those potential features is then reduced in an unsupervised clustering step. Appropriate feature detectors may be trained using the resulting clusters.

The second step of our proposed model learning algorithm simultaneously estimates which ones of the features actually are the most informative, and what is the probabilistic description of the constellation that they tend to form when they are associated to an object of interest. This process requires iterating four operations: (a) choosing a tentative *model structure*, i.e. the collection of features (or parts) that are associated to the object, (b) establishing a correspondence between homologous parts across the training set, and simultaneously labelling as 'background' or 'noise' all parts that are not put in such correspondence, (c) estimating the joint model probability density from such a labelled training set, (d) assessing the performance of such a model. In our method, operations (b) and (c) are performed only implicitly in a "soft" way, using *expectation maximization*.

The best performing model generated in such fashion is in the end selected as "the model".

Outline of the Paper

In Section 3 we discuss our feature selection procedure based on vector quantization. Section 4 introduces the probabilistic model, explains MAP classification and discusses how the EM algorithm is used to learn the model parameters from example images. In Section 5 we report on the results of experiments that test the ability of the model to generalize to unseen viewing angles. The appendix completes the theory with some detailed formulas of the learning algorithm.

3 Automatic Feature Selection

The problem of selecting distinctive and well localizeable features is intimately related to the method chosen to detect these features. Since we need to evaluate a large number of potential features and thus, detectors, we settled on normalized correlation as feature detection method. Furthermore, extensive experiments lead us to believe that this method offers comparable performance over many more elaborate detection methods.

With correlation based detection, every pattern in a small neighborhood in the training images defines a feature detector. The purpose of the procedure described here is to reduce this potentially huge set of features to a reasonable number, such that the model learning algorithm described in the next section can then select a few most useful features. We use a two-step procedure to accomplish this.

In the first step, we identify points of interest in the training images (see Fig. 2). This is done using the interest operator proposed by Förstner in [3], which is capable of detecting corner points, intersections of two or more lines, as well as center points of circular patterns. These pattern are

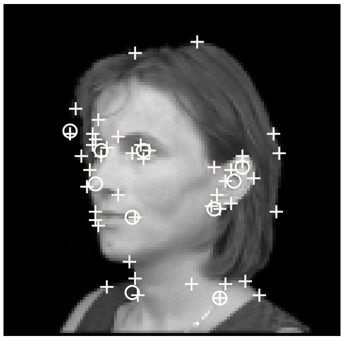

Figure 2. Points of interest identified on a human head using Förstner's method. Crosses denote corner-type patterns while circles mark circle-type patterns.

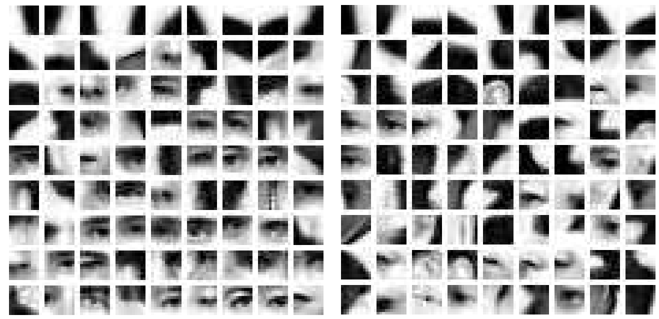

Figure 3. An example of a set of patterns obtained using k-means clustering (vector quantization) of small image patches. The left set was obtianed from frontal views, the right one from semi-profile views.

easy to localize, since the interest operator assures that they contain points of large image gradients in more than one direction. This step produces about 50 feature candidates per training image.

A significant reduction of the number of features can be achieved by the second step of the selection process, which consists in performing vector quantization on the patterns. To this end, we use a standard k-means clustering algorithm [2], which we tuned to produce a set of about 100–150 patterns.

In order to further eliminate redundancies, we remove patterns which are simlilar to others after a shift of up to two pixels in an arbitrary direction.

Due to the restriction to points of interest and the fact that the training images contain only human heads, the pattern set exhibits interesting structure, as can be seen in Figure 3. The patterns shown represent the centers of the respective clusters and are obtained as averages of about 20–500 original patterns. Since we segment the heads from the training images and overlay them on white and black backgrounds, the most common patterns are simple corners which stem mostly from the silhouette of the heads. But one can also readily identify true facial features such as eyes, nostrils, earlobes, mouth corners and so forth.

4 Model Training

In order to train an object model we need to solve two problems. Firstly, we need to decide on a small subset of the selected features, to be used as parts in the model, i.e. define the *model configuration*. Secondly, we need to learn the parameters of the statistical part of the model. The first problem is solved in an outer loop throughout which different promising configurations are evaluated. The model parameters are estimated within the loop using *expectation maximization* (EM). After each iteration, the detection performance of the model is evaluated using a validation data set (disjoint from the test set). Based on the performance, a feature in the configuration might be exchanged against a more promising one.

In the remainder of this section we discuss the problem of estimating the parameters of the statistical object class model, given a fixed model configuration.

We assume that we have at our disposal T training images, identified by subscripts i . We first apply all feature detectors of the given configuration to the training images and retain only the positions at which the detectors have maximal response (locally). The only training data extracted from the images are these *candidate locations*. In order to achieve a high recognition performance, we then optimize for the *maximum likelihood* fit of our object model to the training data, using the EM algorithm.

4.1 Notation

We assume that a model configuration, comprised of F features, has been chosen.[1] As a simplification, we derive the learning algorithm for a Gaussian density of part positions in the image. The necessary changes to obtain the *translation invariant* version used in the experiments are minor.[2]

All information extracted from a training image, I_i, is

[1] Although an object could, in principle, exhibit several features of the same type, we assume for now, that every detector is included in the model at most once, to avoid further complication of the notation. The extension to multiple features of the same type is straightforward.

[2] This is due to the fact that switching to a representation where feature positions are described relative to a reference feature involves only a linear transformation and there is thus no need to depart from the class of Gaussian pdf's.

represented in the following "matrix" of feature candidate positions,

$$X_i^o = \left(\begin{array}{c} x_{11} x_{12}, \ldots, x_{1N_1} \\ x_{21} x_{22}, \ldots, x_{2N_2} \\ \vdots \\ x_{F1} x_{F2}, \ldots, x_{FN_F} \end{array} \right.,$$

where the superscript 'o' indicates that these positions are observed in the image, as opposed to being unobserved or *missing*, which will be indicated by 'm'. Thus, the f^{th} row contains the (two-dimensional) locations of detections of feature type f.

We will use the following random variables, which represent either observed or hidden information in image i,

$$\mathcal{D}_i = \{X_i^o, \mathbf{x}_i^m, \mathbf{n}_i, \mathbf{h}_i, \mathbf{b}_i\}.$$

The entire set, X_i^o, of feature candidates can be divided into candidates which are the *true* features of the object (the *foreground*) and noise features coming from the *background*. This non-observeable fact is conveyed by the random variable $\mathbf{h}$, a set of indices, which is also called *a hypothesis*—for reasons to become evident later. Thus, $h_i = j,\ j > 0$, means that point x_{ij} is a foreground point. If the true object features has been missed altogether by the corresponding detector, the correct hypothesis will have a zero-entry at this position. We denote by $\mathbf{b}$ a binary vector which has entry $b_f = 1$ if $h_f > 0$ and zero otherwise, indicating whether the relevant feature is hypothesized to be detected (1) or not (0). The positions of the missed (or occluded) foreground features are represented by a separate vector $\mathbf{x}^m$. The dimension of $\mathbf{x}^m$ varies between 0 and F depending on the number of unobserved features. Finally $\mathbf{n}$ denotes the number of background detections.[3]

All variables, except X^o, are hidden from direct observation.

4.2 Classification

For the experiments in this paper our objective is to classify images into the classes "head present" (class $\mathcal{C}_1$) and "head absent" (class $\mathcal{C}_0$). Given a probability density for the observed data, $p(X^o)$, the optimal decision—minimizing the total error probability—is made by choosing the class which has the maximum a posteriori probability (MAP approach, see e.g. [2]). It is therefore convenient to consider the following ratio,

$$\frac{p(\mathcal{C}_1|X^o)}{p(\mathcal{C}_0|X^o)} \propto \frac{\sum_{\mathbf{h}} p(X^o, \mathbf{h}|\mathcal{C}_1)}{p(X^o, \mathbf{h}_0|\mathcal{C}_0)}, \qquad (1)$$

[3] Although this representation is redundant ($\mathbf{b}$ is entirely determined by $\mathbf{h}$ while $\mathbf{n}$ is obtained from $\mathbf{h}$ and the total number of detections, $\mathbf{N}$), it allows us to put the parts of the probabilistic model into correspondence with the underlying physical processes, while accurately reflecting the natural dependencies between the random variables.

where $\mathbf{h}_0$ denotes the *null hypothesis* which explains all features as background noise. For convenience, we omitted the variables $\mathbf{b} = sign(\mathbf{h})$ (with $sign(0) = 0$) and $\mathbf{n} = \mathbf{N} - \mathbf{b}$, which are functions of $\mathbf{h}$. Notice that the ratio $\frac{p(\mathcal{C}_1)}{p(\mathcal{C}_0)}$ can be absorbed into a decision threshold. The sum in the numerator includes all hypothesis, including the null hypothesis, since the object could be present but remain undetected by any feature detector. In the denominator, the only consistent hypothesis to explain "object absent" is, of course, the null hypothesis.

4.3 The Model

For a given training image, I_i, we can write the probability density function modeling the data as:

$$p(X_i^o, \mathbf{x}_i^m, \mathbf{h}_i, \mathbf{n}_i, \mathbf{b}_i) =$$
$$p(X_i^o, \mathbf{x}_i^m | \mathbf{h}_i, \mathbf{n}_i, \mathbf{b}_i)\, p(\mathbf{h}_i|\mathbf{n}_i, \mathbf{b}_i)\, p(\mathbf{n}_i)\, p(\mathbf{b}_i).$$

The probability density over the number of background detections can be modeled by a Poisson[4] distribution,

$$p(\mathbf{n}) = \prod_{f=1}^{F} \frac{1}{n_f!} (M_f)^{n_f} e^{-M_f},$$

where M_f is the average number of background detections per image. Admitting a different M_f for every feature allows us to model different detector statistics.

The binary vector $\mathbf{b}$ encodes information about which features have been detected and which have been missed or occluded. The corresponding probability, $p(\mathbf{b})$, is modeled explicitely by a table of size 2^F which equals the number of possible binary vectors of length F. If F is large, the explicit probability table might become unreasonably large. In this case we can assume independence between the feature detectors and model $p(\mathbf{b})$ by a product of independent densities, $p(\mathbf{b}) = \prod_{f=1}^{F} p(b_f)$. The number of parameters is reduced in that case from 2^F to F.

The density $p(\mathbf{h}|\mathbf{n}, \mathbf{b})$ is modeled by,

$$p(\mathbf{h}|\mathbf{n}, \mathbf{b}) = \begin{cases} \frac{1}{\prod_{f=1}^{F} N_f^{b_f}} & \mathbf{h} \in \mathcal{H}_{\mathbf{b}} \\ 0 & \text{other } \mathbf{h} \end{cases}$$

where $\mathcal{H}_{\mathbf{b}}$ denotes the set of all hypotheses consistent with $\mathbf{b}$ and $\mathbf{n}$, and N_f denotes the total number of detections of feature f. This expresses the fact that all consistent hypotheses, the number of which is $\prod_{f=1}^{F} N_f^{b_f}$, are equally likely in the absence of information on the feature locations.

Finally, we use

$$p(X^o, \mathbf{x}^m | \mathbf{h}, \mathbf{n}) = p_{\mathbf{fg}}(\mathbf{x}^o, \mathbf{x}^m)\, p_{\mathbf{bg}}(\mathbf{x}_{bg}),$$

[4] Given that we are dealing with a discrete set of pixel locations, a binomial distribution might seem more natural. However, since the Gaussian foreground density is defined over a continuum of part positions, the Poisson distribution is the natural counterpart for the background process.

where we defined $[\mathbf{x}^o\, \mathbf{x}^m]$ as the coordinates of the *hypothesized* foreground detections (observed and missing) and $\mathbf{x}_{bg}$ as the coordinates of the background detections. The density $p_{\mathbf{fg}}(\mathbf{x}^o, \mathbf{x}^m)$ is modeled as a joint Gaussian with mean μ and covariance Σ. The positions of the background detections are modeled by a uniform density,

$$p_{\mathbf{bg}}(\mathbf{x}_{bg}) = \prod_{f=1}^{F} \frac{1}{A^{n_f}},$$

where A is the area covered by the image. This also conveys our assumption that individual background detections are independent of each other and of the foreground detections.

4.4 Learning

To estimate the parameters of the generative model, $\theta = \{\mu,\ \Sigma,\ p(\mathbf{b}),\ \mathbf{M}\}$, we will use the expectation maximization (EM) algorithm to find their maximum likelihood (ML) solutions. The EM algorithm is particularly suited for our problem since the variables $\mathbf{n}, \mathbf{b}, \mathbf{h}$ and $\mathbf{x}^m$ are unobserved and must be inferred from the observed data X^o. In the following we will omit reference to the variables $\mathbf{n}$ and $\mathbf{b}$ because they are simple functions of $\mathbf{h}$ (see previous section).

In standard EM fashion, we attempt to maximize the log-likelihood of the observed data, which is given as

$$L(X^o|\theta) = \sum_{i=1}^{N} \log \sum_{\mathbf{h}_i} \int p(X_i^o, \mathbf{x}_i^m, \mathbf{h}_i|\theta)\, d\mathbf{x}_i^m.$$

Since maximizing sums and integrals of a logarithm is difficult in practice, we choose to iteratively optimize a sequence of cost functions—again in standard EM fashion:

$$Q(\theta_k|\theta_{k-1}) = \sum_{i=1}^{N} E_{k-1}[\log p(X_i^o, \mathbf{x}_i^m, \mathbf{h}_i|\theta_k)],$$

where k counts iterations and $E_{k-1}[.]$ denotes expectation with respect to the posterior density $p(\mathbf{h}_i, \mathbf{x}_i^m|X_i^o, \theta_{k-1})$. Formally, the E-step amounts to the calculation of this posterior density (or certain sufficient statistics thereof), while the M-step maximizes $Q(\theta_k|\theta_{k-1})$ over θ_k, given this posterior density with parameters from the previous iteration, θ_{k-1}. It can be shown that the EM algorithm converges to a local maximum of the log-likelihood. The respective M- and E-steps for our model are included in the appendix.

5 Experiments

We performed three experiments in order to evaluate the performance of the method and, in particular, the ability of the face detectors to generalize to unseen viewing directions. In the first experiment, we trained three models, each for only one particular viewing direction (frontal, profile and semi-profile). In the second experiment, three models were provided with training images from segments of width 30°. Finally, we trained a single model on the entire quadrant from frontal to profile view.

Performance was measured in a classification task where images had to be labeled as either containing a face or not containing a face.

5.1 Training and Test Images

In order to produce a large set of images with different but known head orientations, a sufficient number of subjects, as well as different, cluttered backgrounds, we resorted to a synthetic blending of head images with background scenes. Subjects were photographed in front of a blue background facing 9 different viewing directions $(-15°, 0°(=\text{frontal}), 15°, \ldots, 90°(=\text{profile}), 105°)$. The background was then subtracted from the images which were converted to grayscale; and the background was replaced with entirely white or black regions to produce training images (see Fig. 4 top), as well as with random scenes to produce test images (see Fig. 4 bottom). From the same set of background scenes images were selected at random to serve as negative examples in the classification task. Overall, we used images of 22 subjects. This set was divided into two non-overlapping, equally large sets for training and testing. Four pictures were taken of each subject at every viewing direction.

The set of background scenes contained 150 pictures of landscapes, outdoor scenes of buildings and cars, as well as indoor scenes of office and laboratory environments. This set was divided into two sets of 75 pictures for training and testing.

The resolution of the training images was such that the distance from top of the head to chin spanned about 80 pixels.

5.2 Automatic Feature Selection

Features were automatically selected according to the procedure described in Sec. 3. The Förstner interest operator was applied to training images with black and white background, taken from the same viewing direction(s) as were used to subsequently train the model.

The idea underlying the choice of pure black and white backgrounds is that points on the silhouette of the face might be useful features, if there is a reasonable chance that the face is seen, at detection time, against a fairly uniform background which is either darker or brighter than the head itself. Since we used normalized correlation, it is only necessary that the background be *slightly* different in brightness from the head, since any difference in a local region will be amplified through the normalization.

We performed vector quantization on grayscale patches of size 11×11 pixels, cut out at the points of interest. A different code book was produced for every experiment, and samples are shown in Figure 3.

Figure 4. Examples from the image database. Training images (top), background test images (center) and faces synthetically blended into different backgrounds (bottom) are shown. The latter were used for testing and validation.

5.3 Model Training

The experiments described here were carried out using a translation invariant version of the EM learning algorithm. The algorithm could have been used as described above, but care would have had to be taken to register the training images.

We initialized the model configuration with a small randomly drawn set and estimated the model parameters by running EM on the data from images with black and white backgrounds (22 images per viewing direction were used). Aside from the fact that black and white backgrounds were used in the feature selection step, they are used here again, in order to avoid biasing the model towards backgrounds darker or brighter than the foreground.

The number of features in all models was limited to three since we found that, due the limited number of training images, the learning algorithm was overfitting the training data when four or more features were used. Even with three feature models, the training error was often close to zero, while the test error was significantly larger. This indicates that some overfitting remained and that the amount of training data was not sufficient to estimate all degrees of freedom of the model. Reducing the degrees of freedom by allowing only diagonal covariance matrices in the statistical shape model did not solve this problem. For a larger set of training images, we expect both errors to approach a common limit, somewhere between the observed test and training errors.

We found the EM algorithm to converge in 10-100 iterations. One iteration took about 0.2 seconds using a Matlab implementation. The entire training process took about three hours on a state of the art PC and, on average, about 250 model configurations were tried in the case of a three feature model.

5.4 Performance Evaluation

Rather than classifying every image by applying a fixed decision threshold according to Equation 1, we computed receiver operating characteristics (ROCs) based on the ratio of posterior probabilities. We determined the point on the ROC curve corresponding to an equal fraction of misclassified foreground and background images and used this error rate as a performance measure. Error rates are shown for different viewing directions in the tuning curves in Figure 5.

One can observe that the models trained under only one viewing direction have indeed narrow tuning characteristics. Also, these models are not superior to more broadly trained models within their designated viewing range. This suggests that a more diverse training set is generally beneficial. The models trained over $30°$ segments show the

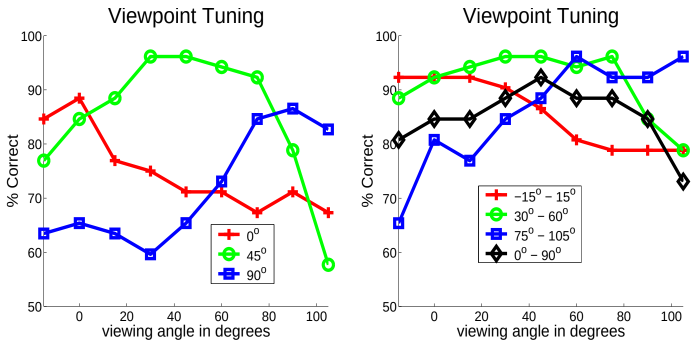

Figure 5. Tuning curves showing performance of models for Experiment 1 (left) and Experiments 2 and 3 (right).

best detection performance and a very good generalization capability. The likely reason for the somewhat disappointing performance of the model trained on the entire viewing range is that the learning algorithm is not able to identify consistent feature arrangements well in a very diverse data set. We have also observed a tendency of the configuration selection strategy to get stuck in local extrema in this case.

The computational requirements of our method for *detection* are rather small. The bulk of the processing time is used to filter the images with the feature templates, which takes about 0.2 seconds in Matlab for three-feature models and images of size 160×120 pixels.

Detection results are illustrated in Figure 6.

6 Discussion and Conclusion

The system we describe improves upon previous work on face and head detection on two counts. It is viewpoint invariant, rather than restricted to frontal views of the face. Furthermore, no direct supervision is required for training the system, unlike previous work where an operator had to align and normalize the training set and/or click on the most distinctive facial features of each training example. Moreover: it detects efficiently the head amongst clutter (around 0.3 seconds per image on a Pentium 400MHz) and it is robust with respect to occlusion.

Our experiments indicate that orientation invariance may be achieved both by combining the output of different models that were trained on specific views, and by training a single model on all views. Best performance is achieved by training on 30° viewpoint intervals, possibly achieving the optimum in a tradeoff between number of training examples and model specificity. Performance in that case is above 90% when training and testing on heads belonging to different people.

Figure 6. Examples of correctly and incorrectly classified images for models trained on $0°$ (left), $45°$ (center), and $90°$ (right) views. Models are comprised of three features. Feature candidates are labeled accordingly with three different markers. Markersize indicates the probability of a feature candidate to correspond to the foreground object. This information can be used to *locate* the heads. Classification errors result mostly from failure to detect enough features, due to large tilt/slant or scale difference.

Many aspects of the system are susceptible of improvement. The choice of the features is far from optimal: it may be noticed in Figure 2 that a number of features is edge-like rather than corner-like indicating that our implementation of the Förstner interest operator in connection with the clustering method needs to be improved. Another issue concerning feature selection: it should not be restricted to a single scale but rather features should be chosen at multiple scales of resolution in order to incorporate both fine details and coarse 'lowpass' aspects of the face. On model training: our greedy algorithm is not necessarily optimal. Lastly, it is likely that the performance of the system would further improve if more and more diverse training examples were used.

A M-step and E-step

Taking the derivative of $Q(\theta_k|\theta_{k-1})$ with respect to the parameters μ, Σ, $p(\mathbf{b})$, $\mathbf{M}$ and equating this to zero produces the following update rules,

$$\begin{aligned}
\mu_k &= \frac{1}{N}\sum_{i=1}^{N} E_{k-1}[\mathbf{z}_i], \\
\Sigma_k &= \frac{1}{N}\sum_{i=1}^{N} E_{k-1}[\mathbf{z}_i\mathbf{z}_i^T] - \mu_k\mu_k^T, \\
p_k(\bar{\mathbf{b}}) &= \frac{1}{N}\sum_{i=1}^{N} E_{k-1}[\delta_{\mathbf{b},\bar{\mathbf{b}}}], \\
\mathbf{M}_k &= \frac{1}{N}\sum_{i=1}^{N} E_{k-1}[\mathbf{n}_i],
\end{aligned}$$

where $\mathbf{z}^T = (\mathbf{x}^o\ \mathbf{x}^m)$ and $E_{k-1}[.]$ denotes taking the expectation with respect to the posterior density $p(\mathbf{h}_i, \mathbf{x}_i^m|X_i^o, \theta_{k-1})$. These update rules constitute the M-step. The E-step amounts to the calulation of the sufficient statistics $E[\mathbf{z}_i]$, $E[\mathbf{z}_i\mathbf{z}_i]$, $E[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$ and $E[\mathbf{n}_i]$. The posterior density is given by,

$$p(\mathbf{h}_i, \mathbf{x}_i^m|X_i^o, \theta) = \frac{p(\mathbf{h}_i, \mathbf{x}_i^m, X_i^o|\theta)}{\sum_{\mathbf{h}}\int p(\mathbf{h}_i, \mathbf{x}_i^m, X_i^o|\theta)\, d\mathbf{x}_i^m},$$

where we omitted again the dependence on $\mathbf{b}(\mathbf{h})$ and $\mathbf{n}(\mathbf{h})$. The denominator in the expression above, $p(X_i^o)$, is calculated as follows. Choose a hypothesis consistent with the observed data. Integrate out the missing data in that hyposthesis[5]. Calculate $\mathbf{b}(\mathbf{h})$ and $\mathbf{n}(\mathbf{h})$ and insert them into the joint density. Now repeat this operation, summing over all possible hypothesis. The expectations of the statistics are calculated in a similar fashion. $E[\delta_{\mathbf{b},\bar{\mathbf{b}}}]$ is calculated by summing only over those hypotheses consistent with $\bar{\mathbf{b}}$ in the numerator and dividing by $p(X_i^o)$. Similarly, $E[\mathbf{n}_i]$ is calculated by averaging $\mathbf{n}(\mathbf{h})$ over all hypotheses. For $E[\mathbf{z}] = (\mathbf{x}^o\ E[\mathbf{x}^m])$ we need

$$\int \mathbf{x}^m\ G(\mathbf{z}|\mu,\sigma)\ d\mathbf{x}^m = \mu^m + \Sigma^{mo}\Sigma^{oo\,-1}(\mathbf{x}^o - \mu^o),$$

where we defined $\mu = (\mu^o\ \mu^m)$ and a similar decomposition for Σ. For the calculation of $E[\mathbf{z}\mathbf{z}^T]$ we need in addition to the above equation the following result

$$E[\mathbf{x}^m\mathbf{x}^{m\,T}] = \Sigma^{mm} - \Sigma^{mo}\Sigma^{oo\,-1}\Sigma^{mo\,T} + E[\mathbf{x}^m]E[\mathbf{x}^m]^T.$$

[5] Integrating out dimensions of a Gaussian is simply done by deleting the means and covariances of those dimensions

References

[1] M. Burl, M. Weber, and P. Perona. A probabilistic approach to object recognition using local photometry and global geometry. In *Proc. 5th Europ. Conf. Comput. Vision, H. Burkhardt and B. Neumann (Eds.), LNCS-Series Vol. 1406–1407, Springer-Verlag*, pages 628–641, 1998.

[2] R. Duda and P. Hart. *Pattern Classification and Scene Analysis*. John Wiley and Sons, Inc., 1973.

[3] R. Haralick and L. Shapiro. *Computer and Robot Vision II*. Addison-Wesley, 1993.

[4] T. Leung, M. Burl, and P. Perona. Probabilistic affine invariants for recognition. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 678–684, 1998.

[5] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1):23–38, Jan 1998.

[6] H. A. Rowley, S. Baluja, and T. Kanade. Face detection demo at http://www.ius.cs.cmu.edu/demos/facedemo.html.

[7] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. In *Proc. IEEE Comput. Soc. Conf. Comput. Vision and Pattern Recogn.*, pages 45–51, Santa Barbara, CA., 1998.

[8] K.-K. Sung and T. Poggio. Example-based learning for view-based human face detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(1):39–51, Jan 1998.