A Statistical Method for 3D Object Detection Applied to Faces and Cars

Henry Schneiderman and Takeo Kanade
Robotics Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

In this paper, we describe a statistical method for 3D object detection. We represent the statistics of both object appearance and "non-object" appearance using a product of histograms. Each histogram represents the joint statistics of a subset of wavelet coefficients and their position on the object. Our approach is to use many such histograms representing a wide variety of visual attributes. Using this method, we have developed the first algorithm that can reliably detect human faces with out-of-plane rotation and the first algorithm that can reliably detect passenger cars over a wide range of viewpoints.

1. Introduction

The main challenge in object detection is the amount of variation in visual appearance. For example, cars vary in shape, size, coloring, and in small details such as the headlights, grill, and tires. Visual appearance also depends on the surrounding environment. Light sources will vary in their intensity, color, and location with respect to the object. Nearby objects may cast shadows on the object or reflect additional light on the object. The appearance of the object also depends on its pose; that is, its position and orientation with respect to the camera. For example, a side view of a human face will look much different than a frontal view. An object detector much accommodate all this variation and still distinguish the object from any other pattern that may occur in the visual world.

To cope with all this variation, we use a two-part strategy for object detection. To cope with variation in pose, we use a view-based approach with multiple detectors that are each specialized to a specific orientation of the object as described in Section 2. We then use statistical modeling within each of these detectors to account for the remaining variation. This statistical modeling is the focus of this paper. In Section 3 we derive the functional form we use in all view-based detectors. In Section 4 we describe how we collect the statistics for these detectors from representative sets of training images. In Section 5 we discuss our implementation and in Sections 6 and 7 we describe the accuracy of the face and car detectors.

2. View-Based Detectors

We develop separate detectors that are each specialized to a specific orientation of the object. For example, we have one detector specialized to right profile views of faces and one that is specialized to frontal views. We apply these view-based detectors in parallel and then combine their results. If there are multiple detections at the same or adjacent locations, our method chooses the strongest detection.

We empirically determined the number of orientations to model for each object. For faces we use two view-based detectors: frontal and right profile, as shown below. To detect left-profile faces, we apply the right profile detector to a mirror-reversed input images. For cars we use eight detectors as shown below. Again, we detect left side views by running the seven right-side detectors on mirror reversed images.

Figure 1. Examples of training images for each face orientation

Figure 2. Examples of training images for each car orientation

Each of these detector is not only specialized in orientation, but is trained to find the object only at a specified size within a rectangular image window. Therefore, to be able to detect the object at any position within an image, we re-apply the detectors for all possible positions of this rectangular window. Then to be able to detect the object at any size we iteratively resize the input image and re-apply the detectors in the same fashion to each resized image.

3. Functional Form of Decision Rule

For each view-based detector we use statistical modeling to account for the remaining forms of variation. Each of the detectors uses the same underlying form for the statistical decision rule. They differ only in that they use statistics gathered from different sets of images.

There are two statistical distributions we model for each view-based detector. We model the statistics of the given object, $P(image \mid \text{object})$ and the statistics of the rest of the visual world, which we call the "non-object" class, $P(image \mid \text{non-object})$. We then compute our detection decision using the likelihood ratio test:

$$\frac{P(image \mid \text{object})}{P(image \mid \text{non-object})} > \lambda \qquad \left(\lambda = \frac{P(\text{non-object})}{P(\text{object})}\right) \tag{1}$$

If the likelihood ratio (the left side) is greater than the right side, we decide the object is present.

The likelihood ratio test is equivalent to Bayes decision rule (MAP decision rule) and will be optimal if our representations for $P(image \mid \text{object})$ and $P(image \mid \text{non-object})$ are accurate. The rest of this section focuses on the functional forms we choose for these distributions.

3.1. Representation of Statistics Using Products of Histograms

The difficulty in modeling $P(image \mid \text{object})$ and $P(image \mid \text{non-object})$ is that we do not know the true statistical characteristics of appearance either for the object or for the rest of the world. For example, we do not know if the true distributions are Gaussian, Poisson, or multimodal. These properties are unknown since it is not tractable to analyze the joint statistics of large numbers of pixels.

Since we do not know the true structure of these distributions, the safest approach is to choose models that are flexible and can accommodate a wide range of structure. One class of flexible models are non-parametric memory-based models such as Parzen windows or nearest neighbor. The disadvantage of these models is that to compute a probability for a given input we may have to compare that input to all the training data. Such a computation will be extremely time consuming.

An alternative is to use a flexible parametric model that is capable of representing multimodal distributions, such as a multilayer perceptron neural network or a mixture model. However, there are no closed-form solutions for fitting these models to a set of training examples. All estimation methods for these models are susceptible to local minima.

Instead of these approaches, we choose to use histograms. Histograms are almost as flexible as memory-based methods but use a more compact representation whereby we retrieve probability by table look-up. Estimation of a histogram simply involves counting how often each attribute value occurs in the training data. The resulting estimates are statistically optimal. They are unbiased, consistent, and satisfy the Cramer-Rao lower bound.

The main drawback of a histogram is that we can only use a relatively small number of discrete values to describe appearance. To overcome this limitation, we use multiple histograms where each histogram, $P_k(pattern \mid \text{object})$, represents the probability of appearance over some specified **visual attribute**, $pattern_k$; that is, $pattern_k$ is a random variable describing some chosen visual characteristic such as low frequency content. We will soon specify how we partition appearance into different visual attributes. However, in order to do so, we need to first understand the issues in combining probabilities from different attributes.

To combine probabilities from different attributes, we will take the following product where we approximate each class-conditional probability function as a product of histograms:

$$\begin{aligned} P(image \mid \text{object}) &\approx \prod_k P_k(pattern_k \mid \text{object}) \\ P(image \mid \text{non-object}) &\approx \prod_k P_k(pattern_k \mid \text{non-object}) \end{aligned} \tag{2}$$

In forming these representations for $P(image \mid \text{object})$ and $P(image \mid \text{non-object})$ we implicitly assume that the attributes ($pattern_k$) are statistically independent for both the object and the non-object. However, it can be shown that we can relax this assumption since our goal is accurate classification not accurate probabilistic modeling [2]. For example, let us consider a classification example based on two random variables, A and B. Let's assume that A is a deterministic function of B, $A = f(B)$, and is therefore fully dependent on A, $P(A{=}f(B) \mid B) = 1$. The optimal classifier becomes:

$$\frac{P(A, B \mid \text{object})}{P(A, B \mid \text{non-object})} = \frac{P(A \mid B, \text{object})P(B \mid \text{object})}{P(A \mid B, \text{non-object})P(B \mid \text{non-object})}$$

$$= \frac{P(B \mid \text{object})}{P(B \mid \text{non-object})} > \lambda \tag{3}$$

If we wrongly assume statistical independence between A and B, then the classifier becomes:

$$\begin{aligned} \frac{P(A, B \mid \text{object})}{P(A, B \mid \text{non-object})} &\approx \frac{P(A \mid \text{object})P(B \mid \text{object})}{P(A \mid \text{non-object})P(B \mid \text{non-object})} \\ &= \left(\frac{P(B \mid \text{object})}{P(B \mid \text{non-object})}\right)^2 > \gamma \end{aligned} \tag{4}$$

This case illustrates that we can achieve accurate classification (by choosing $\gamma = \lambda^2$) even though we have violated the statistical independence assumption.

In the general case, when we do not know the relationship between A and B, performance will depend on how well the ratio $P(A \mid \text{object}) / P(A \mid \text{non-object})$ approximates $P(A \mid B, \text{object}) / P(A \mid B, \text{non-object})$. However, it is probably better to model A and B as statistically independent than to only model one variable. For example, if we only model B our classifier would become:

$$\frac{P(A, B \mid \text{object})}{P(A, B \mid \text{non-object})} \approx (1)\frac{P(B \mid \text{object})}{P(B \mid \text{non-object})} \tag{5}$$

In this formulation, we would be implicitly approximating $P(A \mid B, \text{object}) / P(A \mid B, \text{non-object})$, by 1. Chances are that $P(A \mid \text{object}) / P(A \mid \text{non-object})$ is a better approximation.

In choosing how to decompose visual appearance into different attributes we face the question of what image measurements to model jointly and what to model independently.

Obviously, if the joint relationship between two variables, such as *A* and *B* seems to distinguish the object from the rest of the world, we should try to model them jointly. If we are unsure, it is still probably better to model them independently than not to model one at all.

Ideally, we would like to follow a systematic method for determining which visual qualities distinguish the object from the rest of the visual world. Unfortunately, the only known way of doing this is to try all possibilities and compare their classification accuracy. Such an approach is not computationally tractable. Ultimately, we have to make educated guesses about what visual qualities distinguish faces and cars from the rest of the world. We describe these guesses below.

3.2. Decomposition of Appearance in Space, Frequency, and Orientation

For both faces and cars, our approach is to jointly model visual information that is localized in space, frequency, and orientation. To do so, we decompose visual appearance along these dimensions. Below we explain this decomposition and in the next section we specify our visual attributes based on this decomposition.

First, we decompose the appearance of the object into "parts" whereby each visual attribute describes a spatially localized region on the object. By doing so we concentrate the limited modeling power of each histogram over a smaller amount of visual information.

We would like these parts to be suited to the size of the features on each object. However, since important cues for faces and cars occur at many sizes, we need multiple attributes over a range of scales. We will define such attributes by making a joint decomposition in both space and frequency. Since low frequencies exist only over large areas and high frequencies can exist over small areas, we define attributes with large spatial extents to describe low frequencies and attributes with small spatial extents to describe high frequencies. The attributes that cover small spatial extents will be able to do so at high resolution. These attributes will capture small distinctive areas such as the eyes, nose, and mouth on a face and the grill, headlights, and tires on a car. Attributes defined over larger areas at lower resolution will be able to capture other important cues. On a face, the forehead is brighter than the eye sockets. On a car, various surfaces such as the hood, windshield, and fenders may differ in intensity.

We also decompose some attributes in orientation content. For example, an attribute that is specialized to horizontal features can devote greater representational power to horizontal features than if it also had to describe vertical features.

Finally, by decomposing the object spatially, we do not want to discard all relationships between the various parts. We believe that the spatial relationships of the parts is an important cue for detection. For example, on a human face, the eyes nose, and mouth appear in a fixed geometric configuration. To model these geometric relationships, we represent the positions of each attribute sample with respect to a coordinate frame affixed to the object. This representation captures each sample's relative position with respect to all the others. With this representation, each histogram now becomes a joint distribution of attribute and attribute position, $P_k(pattern_k(x,y), x, y \mid \text{object})$ and $P_k(pattern_k(x,y), x, y \mid \text{non-object})$, where attribute position, x, y, is measured with respect to rectangular image window we our classifying (see section 2). However, we do not represent attribute position at the original resolution of the image. Instead, we represent position at a coarser resolution to save on modeling cost and to implicitly accommodate small variations in the geometric arrangements of parts.

3.3. Representation of Visual Attributes by Subsets of Quantized Wavelet Coefficients

To create visual attributes that are localized in space, frequency, and orientation, we need to be able to easily select information that is localized along these dimensions. In particular, we would like to transform the image into a representation that is jointly localized in space, frequency, and orientation. To do so, we perform a wavelet transform of the image.

The wavelet transform is not the only possible decomposition in space, frequency, and orientation. Both the short-term Fourier transform and pyramid algorithms can create such representations. Wavelets, however, produce no redundancy. Unlike these other transforms, we can perfectly reconstruct the image from its transform where the number of transform coefficients is equal to the original number of pixels.

The wavelet transform organizes the image into subbands that are localized in orientation and frequency. Within each subband, each coefficient is spatially localized. We use a wavelet transform based on 3 level decomposition using a 5/3 linear phase filter-bank [1] producing 10 subbands as shown below in Figure 3. Each level in the transform represents a higher octave

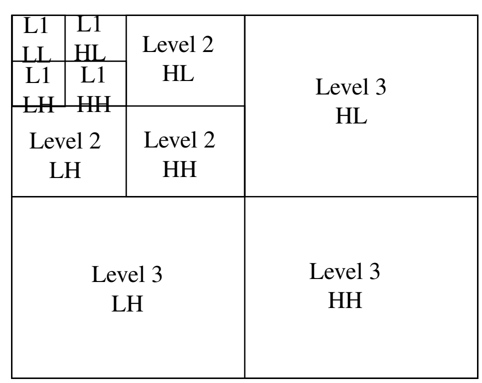


Figure 3. Wavelet representation of an image

of frequencies. A coefficient in level 1 describes 4 times the area of a coefficient in level 2, which describes 4 times the area of a coefficient in level 3. In terms of orientation, LH denotes low-pass filtering in the horizontal direction and high pass filtering in the vertical direction, that is horizontal features. Similarly, HL represents vertical features.

We use this representation as a basis for specifying visual attributes. Each attribute will be defined to sample a moving window of transform coefficients. For example, one attribute could be defined to represent a 3x3 window of coefficients in level 3 LH band. This attribute would capture high frequency horizontal patterns over a small extent in the original image. Another pattern set could represent spatially registered 2x2 blocks in the LH and HL bands of the 2nd level. This would represent an intermediate frequency band over a larger spatial extent in the image.

Since each attribute must only take on a finite number of values, we will have to compute a vector quantization of its sampled wavelet coefficients. To keep histogram size under 1,000,000 bins, we would like to express each attribute by no more than 10,000 discrete values since *x,y* (position) will together take on about 100 discrete values. To stay within this limit, each visual attribute will be defined to sample 8 wavelet coefficients at a time and will quantize each coefficient to 3 levels. This quantization scheme gives 3^8=6,561 discrete values for each visual attribute.

Overall, we use 17 attributes that sample the wavelet transform in groups of 8 coefficients in one of the following ways (These definitions are taken from [3]):

1. Intra-subband - All the coefficients come from the same subband. These visual attributes are the most localized in frequency and orientation. We define 7 of these attributes for the following subbands: level 1 LL, level 1 LH, level 1 HL, level 2 LH, level 2 HL, level 3 LH, level 3 HL.

2. Inter-frequency- Coefficients come from the same orientation but multiple frequency bands. These attributes represent visual cues that span a range of frequencies such as edges. We define 6 such attributes using the following subband pairs: level 1 LL - level l HL, level 1 LL-level l LH, level 1 LH - level 2 LH, level 1 HL - level 2 HL, level 2 LH - level 3 LH, level 2 HL - level 3 HL.

3. Inter-orientation - Coefficients come from the same frequency band but multiple orientation bands. These attributes can represent cues that have both horizontal and vertical components such as corners. We define 3 such attributes using the following subband pairs: level 1 LH - level 1 HL, level 2 LH - level 2 HL, level 3 LH - level 3 HL.

4. Inter-frequency / inter-orientation - This combination is designed to represent cues that span a range of frequencies and orientations. We define one such attribute combining coefficients from the following subbands: level 1 LL, level l LH, level 1 HL, level 2 LH, level 2 HL.

In terms of spatial-frequency decomposition, attributes that use level 1 coefficients describe large spatial extents over a small range of low frequencies. Attributes that use level 2 coefficients describe mid-sized spatial extents over a mid-range of frequencies, and attributes that use level 3 coefficients describe small spatial extents over a large range of high frequencies.

3.4. Final Form of Detector

Finally, our approach is to sample each attribute at regular intervals over the full extent of the object, allowing samples to partially overlap. Our philosophy in doing so is to use as much information as possible in making a detection decision. For example, salient features such as the eyes and nose will be very important for face detection, however, other areas such as the cheeks and chin will also help, but perhaps to a lesser extent.

Thus, the final form of the detector is given by:

$$\frac{\prod_{x,\,y\,\in\,\text{region}}\ \prod_{k\,=\,1}^{17} P_k(pattern_k(x,y),\,x,\,y\,|\,\text{object})}{\prod_{x,\,y\,\in\,\text{region}}\ \prod_{k\,=\,1}^{17} P_k(pattern_k(x,y),\,x,\,y\,|\,\text{non-object})} > \lambda \qquad (6)$$

where “region” is the image window (see Section 2) we are classifying.

4. Collection of Statistics

So far we have only specified the form of the detector. We now need to do collect the actual histograms for $P_k(pattern_k(x,y), x, y\,|\,\text{object})$ and $P_k(pattern_k(x,y), x, y\,|\,\text{non-object})$.

In gathering statistics, one of the immediate problems we face is choosing training examples for the class “non-object.” Conceptually, this class represents the visual appearance of everything in the world excluding the object we want to classify. To represent this class accurately, we would need to collect an extremely large set of images. However, since our goal is classification and not accurate representation, we do not necessarily need a representative set of “non-object” images. In order to achieve accurate classification it is more important to use non-object samples that are most likely to be mistaken for the object [4]. (This concept is similar to the way support vector machines work by selecting samples near the decision boundary [5].) To determine such samples we use a method called bootstrapping. In bootstrapping, we train a preliminary detector by estimating $P_k(pattern_k(x,y), x, y\,|\,\text{non-object})$ using randomly drawn samples from a set of non-object images. We then run this preliminary detector over a set of about 2,500 images that do not contain the object and select additional samples at those locations that gave high response.

We collect $P_k(pattern_k(x,y), x, y\,|\,\text{object})$ from images of the object. For each face viewpoint we use about 2,000 original images and for each car viewpoint we use between 300 and 500 original images. For each original image we generate around 400 synthetic variations by altering background scenery and making small changes in aspect ratio, orientation, frequency content, and position.

We can collect statistics for these training examples using several approaches. In the first approach, we simply give all the training examples equal weight and estimate each histogram separately. We used this method for estimating the car detectors. However, the disadvantage of this approach is that it does not explicitly minimize classification error on the training set. For the face detectors, we minimize classification error over the training set, by using the AdaBoost[9][10] algorithm. AdaBoost works in an iterative fashion. First, we train a detector by assigning the same weight to all training examples. Then we iteratively retrain the detector where at each iteration more weight is given to training examples that were incorrectly classified by the detector trained in the previous iteration. It can be shown that through this process, the classification error can be decreased[9][10].

5. Implementation of Detectors: Coarse to Fine Search Strategy

As we mentioned in the introduction, we search the image exhaustively in position and scale to find objects. A direct implementation of this search will take a long time to compute. We use a heuristic coarse-to-fine strategy to speed up this process. We first partially evaluate the likelihood ratio for each possible object location using low resolution visual attributes, i.e., the ones that use level 1 coefficients. We then only continue evaluation at higher resolution for those object candidates that are promising, i.e., are above a minimum threshold for the partial evaluation. Currently, using this strategy to search a 320x240 image over 4 octaves of candidate size takes about 1 minute for faces and 5 minutes for cars.

We also optimized an earlier frontal face detector we developed [8] using this same strategy and reduced its execution time to about 5 seconds for a 320x240 image.

6. Accuracy of Face Detection with Out-of-Plane Rotation

We have developed the first successful algorithm for detection of faces with out-of-plane rotation. To test its accuracy, we collected a test set of 208 images with 441 faces of which 347 are in profile view. We gathered these images from various sites on the World Wide Web. We show the accuracy of the detector in Table 1 for different sensitivities below, where the last row is our minimum error performance:

Table 1. Face detection with out-of-plane rotation

γ	Detection (all faces)	Detection (profiles)	False Detections
0.0	92.7%	92.8%	700
1.5	85.5%	86.4%	91
2.5	75.2%	78.6%	12

where γ is a linear scaling of λ in equation (6). In figure 4, we show our results on some of these images at $\gamma = 1.5$.

In face detection experiments, we noticed some differences in performance between the detector described in this paper and an improved version of the detector we described in [8]. Both detectors use similar probabilistic structures but differ mainly in that the detector in [8] uses visual attributes based on localized eigenvectors rather than wavelet coefficients. The wavelet based detector described in this paper performs much better for profile faces. However, the eigenvector detector performs slightly better on frontal faces. Below in Table 2 we compare these detectors with others on the combined test sets of Sung and Poggio[6] and Rowley, Baluja, and Kanade[7].

Table 2. Frontal face detection

	Detection rate	False detections
Schneiderman and Kanade* eigenvector	94.4% (95.8%)	65
Roth, Yang, Ahuja [11]*	(94.8%)	78
Schneiderman and Kanade* wavelet	90.2% (91.8%)	110
Rowley, Baluja, Kanade[7]	86.0%	31

* indicates that 5 images of line drawn faces were excluded leaving 125 images with 483 labeled faces. However, there are at least 10 additional human faces that are not labeled. The numbers not in parentheses indicate results on just the 483 labeled faces. To be consistent with [11], we also indicate, in parentheses, the ratio between the total number of faces found by computer and 483.

7. Accuracy of Car Detection

We have also developed the first algorithm that can reliably detect passenger cars over a range of viewpoints. To test its accuracy, we collected a test set of 104 images that contain 213 cars which span a wide variety of models, sizes, orientations, background scenery, lighting conditions, and include some partial occlusion using several cameras and from the internet. Overall our performance was as follows:

Table 3. Car detection

γ	Detections	False Detections
1.05	83%	7
1.0	86%	10
0.9	92%	71

In figure 4, we show our results on some typical images from this set evaluated at $\gamma = 1.0$.

References

[1] G. Strang and T. Nguyen. *Wavelets and Filter Banks*. Wellesley-Cambridge Press. 1997.

[2] P. Domingos and M. Pazzani. “On the Optimality of the Simple Bayesian Classifier under Zero-One Loss.” Machine Learning, 29, pp. 103-130. 1997.

[3]P.C. Cosman, R.M. Gray, M. Vetterli. “Vector Quantization of Image Subbands: A Survey.” IEEE Trans. on Image Processing. 5:2. pp. 202-225. Feb., 1996.

[4]B. D. Ripley. *Pattern Recognition and Neural Networks*. Cambridge University Press. 1996.

[5]V. N. Vapnik. *The Nature of Statistical Learning Theory.* Springer, 1995.

[6]. K-K Sung, T. Poggio. “Example-based Learning of View-Based Human Face Detection.” ACCV ‘95 and AI Memo #1521, 1572, MIT.

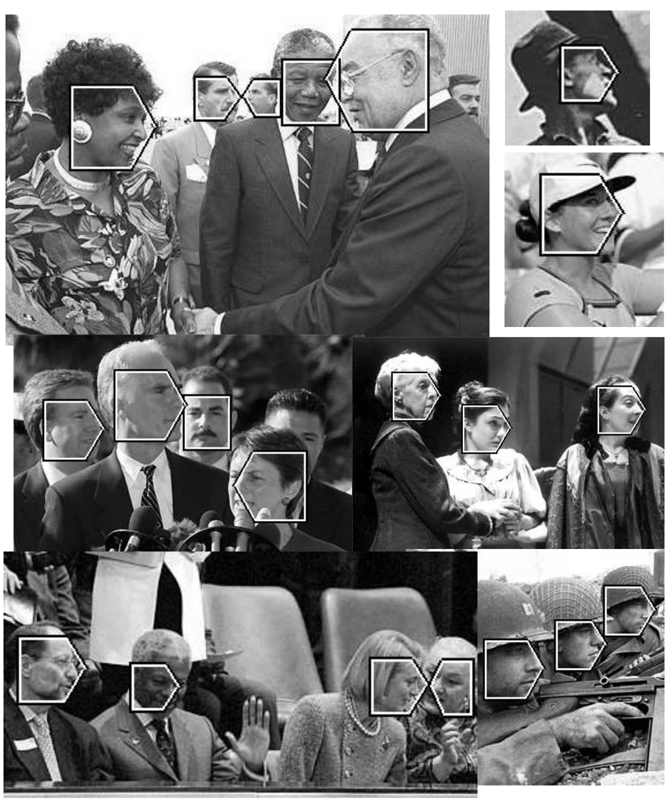
[7]. H. Rowley, S. Baluja, T. Kanade. “Neural Network-Based Face Detection.” PAMI 20(1), January, 1998.

[8] H. Schneiderman and T. Kanade. “Probabilistic Modeling of Local Appearance and Spatial Relationships for Object Recognition.” CVPR ‘98. pp. 45-51.

[9]Y. Freund, R. E. Shapire. “A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting.” Journal of Computer and System Sciences. 55:1, pp. 119-139. 1997.

[10]R. E. Shapire, Y. Singer. “Improving Boosting Algorithms Using Confidence-rated Predictions.” Machine Learning 37:3, pp. 297-336. December, 1999.

[11] D. Roth, M-H. Yang, N. Ahuja. “A SNoW-Based Face Detector.” NPPS ‘99.



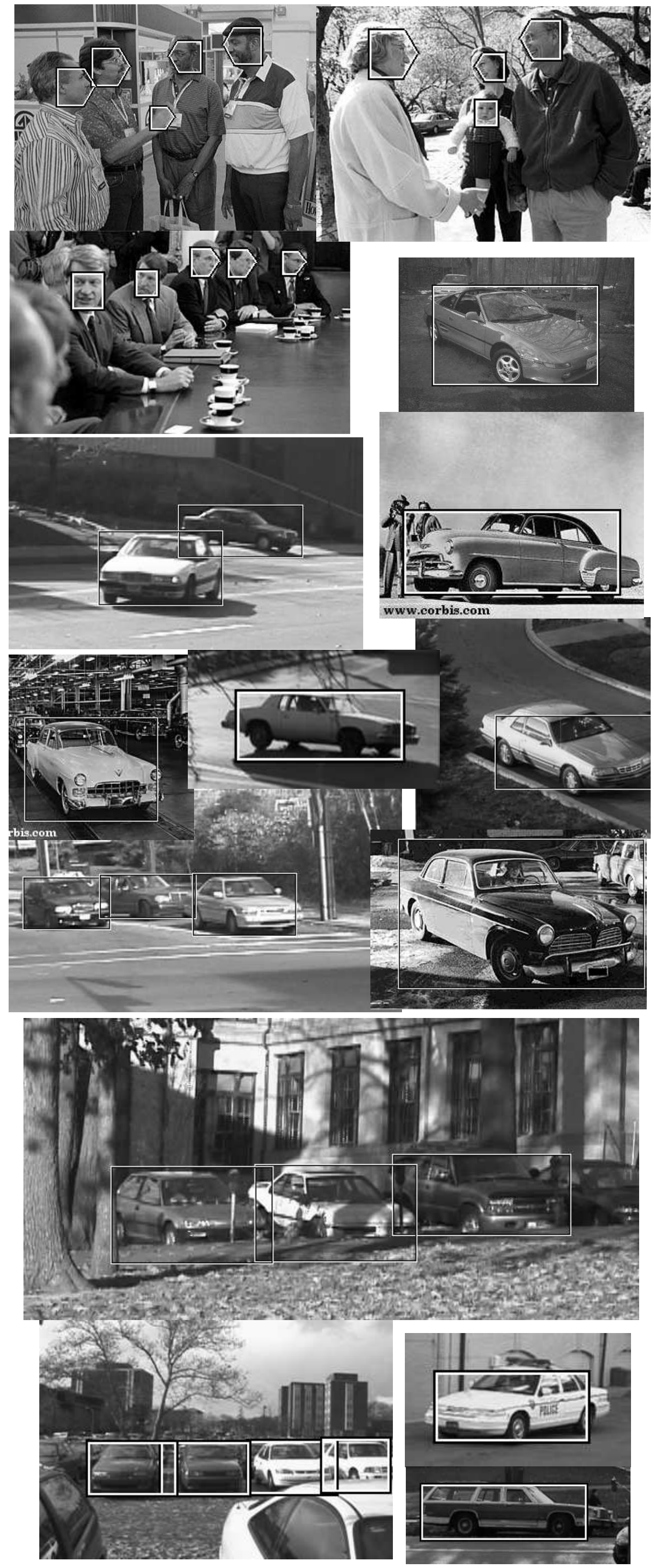


Figure 4. Face and car detection examples