A Histogram-Based Method for Detection of 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 that vary from frontal view to full profile view and the first algorithm that can reliably detect 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. For example, a Lamborghini looks much different from a Ford Pinto. Visual appearance also depends on the surrounding environment. Light sources will vary in their location with respect to the object, their intensity, and their color. 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 human face will look much different when viewed from the side than viewed frontally. 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 2D view-based approach with multiple detectors that are each specialized to a specific orientation of the object. We then use statistical modeling within each of the detectors to account for the remaining variation. We collect these statistical models from representative sets of training images.

2. View-Based Detectors

We develop separate 2D 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 of human faces. We apply these view-based detectors independently and then combine their results. If there are multiple detections at the same or adjacent locations, our method chooses the most confident detection.

We empirically determined the number of orientations to model for each object. For faces we use two view-based detectors 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

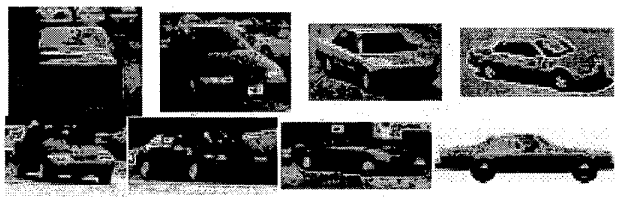


Figure 2. Examples of training images for each car orientation

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 image and re-apply the detectors in the same fashion to each resized image.

3. Statistical Form of Detector

Within each view-based detector we use statistical modeling to account for the remaining forms of variation. Each of these detectors share the same underlying statistical form. They differ only in that their models use statistics gathered from different sets of images.

There are two statistical distributions we model within 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,

0-7803-6297-7/00/$10.00

P(*image* | non-object). We then compute our detection decision using the likelihood ratio test:

$$\frac{P(image|\text{object})}{P(image|\text{non-object})} > \frac{P(\text{non-object})}{P(\text{object})} \quad (1)$$

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

This is equivalent to Bayes decision rule or MAP decision rule and will be optimal if our representations for *P*(*image* | object) and *P*(*image* | non-object) are accurate. The rest of this section focuses on our representation of these distributions.

3.1. Representation of Statistics using Histograms

The difficulty in modeling *P*(*image* | object) and *P*(*image* | 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 statistical properties over a large number 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 the input to all the training data. Such a computation will be extremely time consuming. An alternative is to use a flexible parametric model 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 and may be sub-optimal.

We choose to use histograms as a compromise between the above models. Histograms are almost as flexible as memory-based methods but use a more compact representation whereby probability is retrieved by table look-up. Estimation of a histogram is also trivial. It simply involves counting how often each attribute value occurs in the training data.

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*, $pattern_k$. We will soon specify these different attributes. But to do so, we need understand how we would combine probabilities from different histograms and the consequences of our chosen method of combination.

To combine these histograms, we approximate the class-conditional probabilities as products of histograms:

$$P(image|\text{object}) \approx \prod_k P_k(pattern_k|\text{object})$$
$$P(image|\text{non-object}) \approx \prod_k P_k(pattern_k|\text{non-object}) \quad (2)$$

In forming these representations for *P*(*image* | object) and *P*(*image* | non-object) there is an implicit assumption that the $pattern_k$s are statistically independent for both classes. However, we can relax this assumption somewhat because our goal is accurate classification not accurate probabilistic modeling. 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*) | *B*) = 1. The optimal classifier becomes:

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

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

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

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

$$= \left(\frac{P(B|\text{object})}{P(B|\text{non-object})}\right)^2 > \gamma \quad (4)$$

Since the likelihood ratio of the statistically independent classifier is the square of that for the optimal classifier, it can achieve the same performance if we choose $\gamma = \lambda^2$.

This case illustrates that we can achieve accurate classification even when we violate the statistical independence assumption.

In the general case, 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. In the next section we describe the qualities we model jointly for face and car detection.

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

For both faces and cars we use the same underlying representation. Our approach is to partition visual appearance along the dimensions of space, frequency, and orientation in

forming visual attributes, $pattern_k$.

First, we decompose the appearance of the object spatially where each visual attribute describes a localized region on the object. By doing so we concentrate the limited modeling power of each histogram over a smaller area. To represent parts of different sizes, we use multiple attributes that differ in spatial extent. Some attributes will represent small spatial extents. These attributes will capture distinctive areas such as the eyes, nose, and mouth at a high resolution. We also define attributes over larger spatial extents to capture cues where, for example, the forehead is brighter than the eye sockets on a face.

In combination with this decomposition in size we also decompose appearance in frequency content. Since low frequencies exist only over large areas and high frequencies can exist over small areas, the most natural decomposition is to use attributes with large spatial extents to describe low frequencies and attributes with small spatial extents to describe a broad range of high frequencies.

We also specialize 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.

Our approach is to sample these attributes 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.

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 these various 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 and implicitly captures many geometric properties. To capture this representation statistically, each histogram now becomes a joint distribution of attribute and attribute position, $P_k(pattern(x,y), x, y \mid \text{object})$ and $P_k(pattern(x,y), x, y \mid \text{non-object})$, where attribute position, x, y, is measured with respect to rectangular image window we our classifying (see section 2).

3.3. Implementation of Visual Attributes

To create visual attributes that are localized in space, frequency, and orientation, we define each attribute to represent a moving window of quantized wavelet coefficients. In other words, each variable $pattern_k$ is a scalar number representing a conglomerate of spatially localized and quantized wavelet coefficients. We use a wavelet transform based on 3 level decomposition using a 5/3 linear phase filter-bank.

Overall, we use 17 attributes that sample the wavelet transform in one of the following ways [1]. Each of these represents a spatially localized set of wavelet coefficients:

1. Intra-subband (7 attributes) - All the coefficients come from the same subband. These visual attributes are the most localized in frequency and orientation.

2. Inter-frequency (6 attributes)- Coefficients come from the same orientation but different frequency bands. These attributes represent visual cues that span a range of frequencies such as edges.

3. Inter-orientation (3 attributes) - Coefficients come from the same frequency band but different orientation bands. These attributes can represent cues that have both horizontal and vertical components such as corners.

4. Inter-frequency / inter-orientation (1 attribute) - This attribute is designed to represent cues that span a range of frequencies and orientations.

Each coefficient in the wavelet transform is represented as part of several different attributes, usually including one of each of the 4 types mentioned above.

With this representation, attributes that use level 1 coefficients will describe large spatial extents over a small range of low frequencies. Those that use level 2 coefficients will describe mid-sized extents over a mid-range of frequencies, and those that use level 3 coefficients will describe small extents over a large range of high frequencies.

3.4. Final Form of Detector

The final form of the detector is given by:

$$\frac{\prod_{x,y \in \text{region}} \prod_{k=1}^{17} P_k(pattern_k(x,y), x, y \mid \text{object})}{\prod_{x,y \in \text{region}} \prod_{k=1}^{17} P_k(pattern_k(x,y), x, y \mid \text{non-object})} > \lambda \quad (5)$$

where "region" is the image window we are classifying.

4. Collection of Statistics

So far we have only specified the functional form of the detector. We now need to do collect the actual histograms for $P_k(pattern, x, y \mid \text{object})$ and $P_k(pattern, x, y \mid \text{non-object})$. We collect $P_k(pattern, x, y \mid \text{object})$ 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 1,000 synthetic variations by altering background scenery and making small changes in aspect ratio,

orientation, frequency content, and position.

For the non-object class we use about 2,500 images that do not contain faces or cars. We use bootstrapping to select training samples from these images. In bootstrapping we first train a preliminary detector using random samples drawn from these images. We then run this detector on the non-object images and select additional samples from sites that give high response.

Finally we use an iterative method called AdaBoost[2][3] to assign weights to each sample, both for objects and non-objects. AdaBoost assigns these weights so as to minimize classification error on the training set.

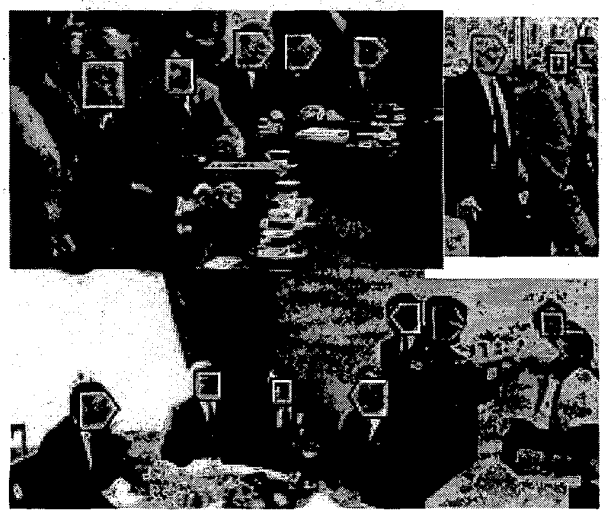
5. Face Detection with Out-of-Plane Rotation

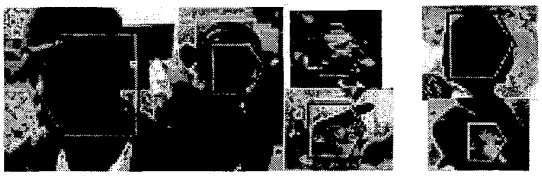
Currently, research in face detection has focused almost exclusively on frontal views. To the best of our knowledge, we have created the first robust algorithm for detecting faces with out-of-plane rotation. To evaluate its performance we collected a test set consisting of 208 images with 441 faces of which 347 were profile views from various news web sites. These image were not restricted in terms of subject matter or background scenery. Below we show our results. Each row indicates a different sensitivity, γ, of the composite face detector. We indicate our results on profile views in parentheses.

Table 1. Detection on faces with out-of-plane rotation

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

Below we show some representative results for $\gamma = 1.5$.





6. Car Detection with Variation in Viewpoint

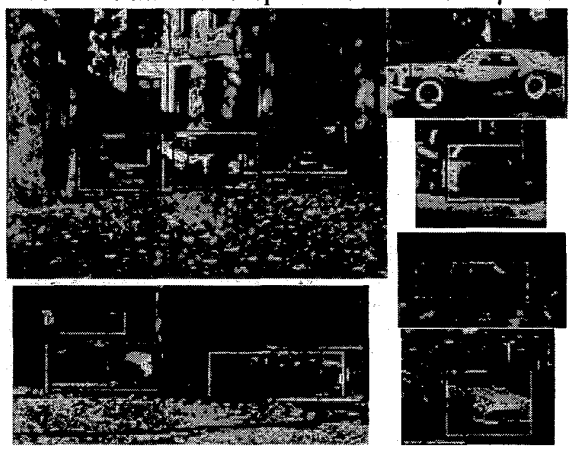
To the best of our knowledge, our method is the first successful car detection algorithm that works accurately over a wide range of viewpoints.

Below we show our results in passenger car detection (excluding trucks and vans) over cars that varied from side view to frontal view. This test set consists of 104 images with 213 cars. We gathered these images both from various web sites and with our own camera. Each row indicates a different sensitivity of the composite detector:

Table 2. Detection of cars

γ	Detection	False Detections
0.9	83%	7
1.0	86%	10
1.1	92%	71

Below we show some representative results for $\gamma = 1.0$.



7. References

[1] P. C. Cosman, R. M. Gray, M. Vetterli. "Vector Quantization of Image Subbands: A Survey." IEEE Transactions on Image Processing. 5:2 pp. 202-225. February, 1996.

[2]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.

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