Fast Face Detection with Precise Pose Estimation

François Fleuret*

IMEDIA project,
INRIA
Domaine de Voluceau
78150 Le Chesnay
FRANCE

Donald Geman

Center for Imaging Science
Johns Hopkins University
301 Clark Hall
3400 N. Charles St.
Baltimore, MD 21218-2686
USA

Abstract

We present a fast algorithm for face detection and precise pose estimation. Our detection scheme is based on a tree-structured hierarchy of face vs. background classifiers combined with a lazy evaluation strategy which concentrates computation on ambiguous areas of the image. The hierarchy corresponds to successive partitions of the pose parameter space, and thereby provides, at no additional cost, a fine estimate of the 2D pose of detected faces.

1 Introduction

Standard methods for face detection apply roughly the same, computationally intensive, strategy: a face vs. background classifier is applied at every image location and at several scales. Different base classifiers have been used, such as artificial neural networks (Rowley, Baluja & Kanade 1998), support vector machines (Osuna, Freund & Girosi 1997) and Gaussian models (Sung & Poggio 1998).

Some recent work (Amit & Geman 1999, Fleuret & Geman 2001, Viola & Jones 2001) has focused on hierarchical representations combining several classifiers at different levels of invariance and discrimination, and based on primitive Boolean features. In these schemes, the various classifiers reject different types of non-face structures, and can result in lower false positive rates. Also, due to the hierarchical architecture, when such detectors are implemented by a coarse-to-fine ("lazy") procedure, the overall computation is concentrated on the portions of the scene that cannot be easily classified as non-face.

We present in this paper an extension of (Fleuret & Geman 2001). The core idea in this approach is to (manually) design a hierarchy of subsets (or "cells") of pose space (see Figure 2) and to build a corresponding family of very simple binary classifiers, one for each cell. Each classifier is then dedicated to the face instantiations with poses in a particular cell; it responds positively if and only if it finds in the picture a minimum number of edges among a subset of edges dedicated to the pose cell and determined during an off-line training phase.

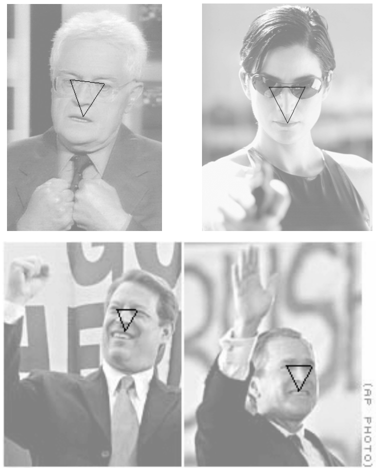

Figure 1: Each detection is displayed with an equilateral triangle. Two of the vertices should correspond to the eyes and the third roughly estimates the location of the mouth. Note that all faces have been rescaled for printing, yet originally appeared at different sizes: From upper-left to lower-right, the distances between the eyes were 60, 70, 15 and 18 pixels respectively.

The family of classifiers is combined into one detector, very tolerant (insensitive) to both scale and location, which is then applied at one location per 8 × 8 block in the scene, and at four different scales (1, $\frac{1}{2}$, $\frac{1}{4}$

[*] Corresponding author: francois.fleuret@inria.fr

Level	Δx	Δy	Δs	$\Delta\theta$
Λ_1^0	8	8	8	40°
$\Lambda_1^1, \ldots, \Lambda_4^1$	4	4	8	40°
$\Lambda_1^2, \ldots, \Lambda_{16}^2$	2	2	8	40°
$\Lambda_1^3, \ldots, \Lambda_{32}^3$	2	2	4	40°
$\Lambda_1^4, \ldots, \Lambda_{64}^4$	2	2	4	20°
$\Lambda_1^5, \ldots, \Lambda_{128}^5$	2	2	2	20°
$\Lambda_1^6, \ldots, \Lambda_{256}^6$	2	2	2	10°

Table 1: The tolerance in pixels and degrees of the pose parameters for each cell in the pose hierarchy. First the position is increasingly constrained, then alternately the scale and tilt.

and $\frac{1}{8}$). The criterion for detection at a given rough location and rough scale is the existence of a chain in the hierarchy, from root to leaf, whose corresponding classifiers all respond positively (see Figure 2.)

We improve the algorithm in (Fleuret & Geman 2001) in three ways: 1/ The classifiers count the number of edge fragments and flat areas instead of counting larger conjunctions (special pairs, triples, etc.) of edge fragments; 2/ A pose is associated with each alarm (detected chain) at no additional cost; 3/ An "average pose" is computed for each set of alarms which seems to be more precise than in any previously reported results on face detection.

This paper is written from an algorithmic perspective. In §2, we describe the search strategy behind the main detector D. In §3, we explain the structure of the component classifiers and how these are induced from a training database. In order to accurately estimate the pose, we give in §4 a simple heuristic to suppress trivial alarms and group the others. Finally, in §5 and §6, we provide experimental results, and summarize and evaluate our approach.

2 Detection Algorithm

The global procedure is to parse the scene at various scales, and at a sampling of locations, with a window of size 64×64, in each case applying a detector D which computes the list of the faces present in the window, and estimates their poses.

The pose of a face is defined in the image plane: The *center* (x, y) is the middle of the segment joining the two eyes, the *scale* s is the distance (in pixels) between the eyes, and the *tilt* θ is the angle between the vertical and the line passing through the center and the mouth.

The detector D is intended to find all faces verifying $(x, y) \in [28, 36]^2$, $-20 \leq \theta \leq 20$ and $8 \leq s \leq 16$. We call Λ_1^0 the set of all four-tuples (x, y, s, θ) verifying these constraints.

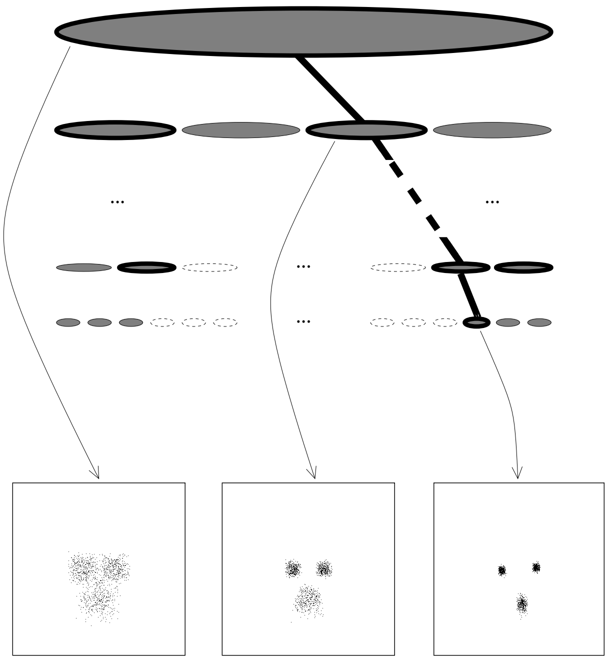

Figure 2: The parameter space of poses is hierarchically decomposed into cells corresponding to successively stronger constraints on the positions of the eyes and mouth in a 64×64 reference frame (cf. the three squares at the bottom). An alarm is identified with a fine (leaf) cell Λ if the classifier for every coarser cell (i.e., containing Λ) responds positively. Such responses are indicated by bold outlines and complete *chains of positive responses*, from root to leaf, are joined by bold lines.

We recursively partition Λ_1^0 into a sequence of nested partitions; each cell Λ of each partition is a subset of poses which is included in exactly one of the cells in the preceding, coarser partition. The first partition of Λ_1^0 is $\Lambda_1^1 \cup \Lambda_2^1 \cup \Lambda_3^1 \cup \Lambda_4^1$. In each of the four cells $\Lambda_i^1, i = 1, ..., 4$, the center (x, y) is constrained to one of the four sub-squares of size 4×4 of the initial 8×8 square. The second partition further constrains (x, y) to 2×2 squares, and contains 16 cells. This is the strongest constraint on location in the hierarchy. The next four partitions involve splitting parent cells into two pieces, and correspond to alternately constraining the scale and the tilt. The set of partitions is summarized in Table 1 and the hierarchy is illustrated in Figure 2.

A binary classifier C_k^m is constructed for each cell Λ_k^m. The classifier is a function of the image data in a 64×64 sub-image I and is intended to respond to faces appearing in I whose pose belongs to Λ_k^m. The classifiers are designed to have a null false negative error rate, i.e., no missed detections, at the expense of false positive errors.

The list of alarms computed by D correspodns to the set of fine cells Λ_k^M, where M is the last level ($M = 6$ in Table 1), such that all the classifiers C corresponding to cells $\Lambda \supseteq \Lambda_k^M$ respond positively, including C_k^M itself. This can be visualized as *chains of positive responses* in the hierarchy of cells (see Figure 2). Each such chain is identified with an average pose in the finest cell, which serves as a good approximation of the pose of the detected face.

The computation of this list of (complete) chains can be performed efficiently by computing the response of a classifier C_k^m if and only if all the classifiers corresponding to cells containing Λ_k^m (hence more invariant to pose) have already been evaluated and responded positively. Areas of the scene rejected by coarse classifiers are then rapidly processed, whereas ambiguous areas, e.g., those containing faces or face-like structures, will not be labeled until some of the fine classifiers have been evaluated, and hence require more computation (see Figure 4). Calculations of the *mean* computation under certain models for the cost and statistics of the classifiers C_k^m appear in (Fleuret & Geman 2001).

3 Cell Classifiers

Each classifier C_k^m checks for a certain number of distinguished edge fragments and a certain number of distinguished flat areas, and hence is defined by two lists (of size $N = 250$) and two thresholds, all determined during training. Consequently, the classifiers are simply counting operators. Checking for an edge fragment means evaluating a binary local feature indexed by a direction ϕ, a location (x, y) in the 64×64 reference window, and a "tolerance" parameter σ which controls the degree of invariance (see Figure 3); details about the edge detector are given in (Fleuret & Geman 2001). Similarly, for each cell, there is a list of locations (x, y) expected to be *flat*; this means that the absolute difference between the intensity at pixel (x, y) and each of its four neighbors is smaller than 70% of the neighbor differences in the 64×64 window.

We emphasize that the algorithm does not check for *conjunctions* of edges as in (Fleuret & Geman 2001), but only for individual edges and flat areas.

All the classifiers in the hierarchy are built with the same learning algorithm; the lists differ due to varying training sets, corresponding to varying constraints on the set of poses. Our experiments are based on the ORL database[1], which contains 400 grayscale face pictures of size 112×96 pixels. We have marked (by hand) the locations of the eyes and the mouth on each training image. (Of course the detection algorithm does not explicitly involve searching for such facial attributes.) For each cell, we create a synthetic training set of 1600 face images whose poses are in Λ_k^m. This is accomplished by randomly choosing four poses in Λ_k^m for each original picture and translating, rotating and scaling it to each of these four poses.

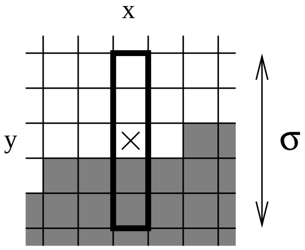

Figure 3: A local feature is present at (x, y) if an edge of a given direction (here horizontal) is detected anywhere in a strip of length σ pixels around (x, y). The orientation of the strip is perpendicular to the edge direction.

The list of edges is determined as follows. First, for each location (x, y) in the 64×64 window, and for each direction ϕ, we compute the minimal tolerance σ such that the probability of an edge somewhere in the strip is at least $\frac{1}{2}$, as estimated by the corresponding fraction of training pictures. We reject the location and direction if this can not be achieved for $\sigma \leq 8$. Then, we subsample N of the remaining local features (x, y, ϕ, σ) to maximize the minimum number of positive responses over the training set. This subsampling is iterative: at each step select a new feature which occurs on the training picture with the least number of positive responses to the features already selected. The procedure is the same for selecting the N flat-area detectors, which are indexed by only a location (x, y). The thresholds are the largest ones achieving a null false negative rate for the resulting classifier on the training set.

The complete training time is about two hours on a standard 1Ghz desktop PC.

4 Pose Estimation

Due to the built-in invariances, the same face will induce multiple alarms (detections). Therefore we post-process the list of alarms in order to group them and to eliminate isolated ones. Indeed, due to the

[1] http://www.cam-orl.co.uk/facedatabase.html

rich and redundant information in the set of complete chains provided by the detector, this is relatively easy to accomplish and could be done in numerous ways. For example, consider the distance

$$d((x, y, \theta, s),\ (x', y', \theta', s')) = \frac{\sqrt{(x-x')^2 + (y-y')^2}}{\min(s, s')}$$

An iterative process begins by assigning a mass of 1 to each alarm. Then, if there exists a pair of alarms closer than $\frac{1}{3}$, also verifying $|s - s'| \leq \frac{1}{2}\ \min(s, s')$, we replace the two closest ones by a weighted average, with the sum of the two masses. When this loop terminates, we eliminate all alarms with small mass (say less than 5). This procedure yields estimates of the pose which are more accurate than the resolution of the finest cells of the hierarchy (see Figures 1 and 4).

5 Results

We have implemented our algorithm in C++ on a GNU/Linux$^{\mathrm{TM}}$system and tested it on 77 scenes taken from various news web sites, containing 103 frontal-view faces. These pictures are well-focused and contain faces of scales ranging from 8 pixels to 75 pixels on complex backgrounds.

The detection rate on these pictures is 92% (9 faces missed). On the average, for a picture of size 320×200, there are 1.45 false alarms and complete processing on a 1Ghz desktop PC requires 0.48 seconds.

We have estimated the error in the pose by marking by hand the real pose on every test face. For 90% of the alarms, the distance between the real location and the estimated one is within 12% of the real scale, the difference between the real scale and the estimated scale is within 21% of the real scale, and, finally, the estimated and real tilts are within 6.9 degrees.

6 Conclusion

We have presented a new way to represent objects and to organize the computation in searching for them in natural scenes. By imposing successively stronger constraints on the pose of a face, we progressively simplify the set of shapes to be modeled. Consequently, we can use very primitive classifiers and still obtain reasonable error rates, especially on high resolution images. Moreover, the search can be made very efficient.

The main weakness of the current implementation is the sensitivity of our edge detectors to degraded (e.g., blurred) images. Also, the database used for training is too small and biased. We intend to overcome these deficiencies, and thereby lower the error rates, while maintaining computational efficiency and precise pose estimation.

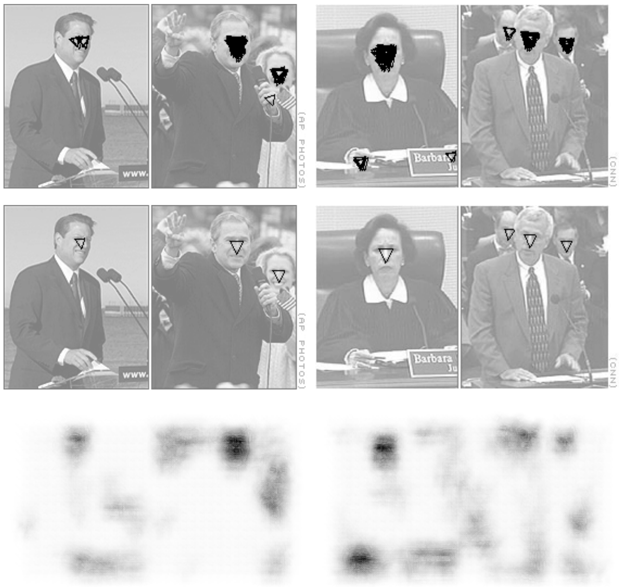

Figure 4: Top row: Detections before clustering. Middle row: Final result after pruning and pose averaging. Bottom row: Intensity of pixel usage (the graylevel shows the number of accesses to each pixel during detection); as expected, the computation is more intense on highly structured areas of the scene.

References

Amit, Y. & Geman, D. (1999), 'A computational model for visual selection', *Neural Computation* **11**, 1691–1715.

Fleuret, F. & Geman, D. (2001), 'Coarse-to-fine visual selection', *International Journal of Computer Vision* **41**(1/2), 85–107.

Osuna, E., Freund, R. & Girosi, F. (1997), Training support vector machines: an application to face detection, *in* 'Proceedings, CVPR', IEEE Computer Society Press, pp. 130–136.

Rowley, H. A., Baluja, S. & Kanade, T. (1998), 'Neural network-based face detection', *IEEE Trans. PAMI* **20**, 23–38.

Sung, K. K. & Poggio, T. (1998), 'Example-based learning for view-based face detection', *IEEE Trans. PAMI* **20**, 39–51.

Viola, P. & Jones, M. J. (2001), Robust real-time object detection, Technical Report CRL2001/01, COMPAQ Cambridge Research Laboratory.