A Unified Stochastic Model for Detecting and Tracking Faces

Sachin Gangaputra
Dept. of Electrical and Computer Engineering
The Johns Hopkins University
Baltimore, MD 21218
sachin@jhu.edu

Donald Geman
Dept. of Applied Mathematics and Statistics
The Johns Hopkins University
Baltimore, MD 21218
geman@jhu.edu

Abstract

We propose merging face detection and face tracking into a single probabilistic framework. The motivation stems from a broader project in algorithmic modeling, centered on the design and analysis of the online computational process in visual recognition. Detection is represented as a tree-structured graphical network in which likelihoods are assigned to each history or "trace" of processing, thereby introducing a new probabilistic component into coarse-to-fine search strategies. When embedded within a temporal Markov framework, the resulting tracking system yields encouraging results.

1. Introduction

Detecting objects in an image and tracking them through a video sequence has been a fundamental problem in computer vision and has inspired significant amount of research in the last few decades. The particular problem of detecting and tracking a human face continues to attract interest due to applications in human-computer interaction, visual surveillance and vision-based control. At present, there exists no solution comparable to human performance in either precision or speed.

The main challenge is achieving invariance to facial appearance and motion. Variations result from many factors, including environmental conditions (lighting, clutter), imaging sensors (frame rate, quantization, noise) and inherent non-rigid variations in the pose, shape, structure and motion of a face. In view of the diversity and magnitude of such variations, most existing approaches are designed to operate most effectively within a restricted domain of variability.

Face detection refers to determining the presence and the location of faces in an image, in particular, distinguishing faces from all other patterns in the scene. Standard methods apply a face vs. background classifier at every image location and at several scales (and perhaps rotations). Base classifiers such as neural networks [14], support vector machines [12], Gaussian models [17] and naive Bayesian models [15] have all been used. Recent work has focused on serially combining multiple classifiers to yield faster and more powerful detectors [6, 5, 18].

The purpose of face tracking is to follow one or more faces through a video sequence. Most approaches exploit the temporal correlation between successive frames in order to refine the localization of the target. Whereas manual initialization is common, in some cases an independent face detector is used to automatically start the process. Tracking methods can broadly be classified as region-based [7], color-based [16], shape-based [3] or model-based [4]. Fairly exhaustive surveys on face detection and tracking can be found in [8, 19]. Traditionally, work in face detection and face tracking have progressed independently of one another and only a few approaches have attempted to merge them into a single framework [11].

The work in this paper stems from a broader project on *algorithmic* or *computational* modeling for semantic scene interpretation. It is motivated in part by the limitations of pure predictive learning and based on explicit modeling of the computational process. Single-frame detection is based on the coarse-to-fine (CTF) framework proposed in [6], where a series of linear classifiers is used to gradually reject non-face patterns and focus computation on ambiguous regions. The resulting distribution of processing is highly skewed and face detection is rapid at the expense of a few false alarms. Here, we expand this approach by introducing the concept of the *trace* of processing in the sense of encoding the computational history – the family of performed tests, together with their outcomes, during coarse-to-fine search. When the set of traces is endowed with a probability distribution, the resulting stochastic network induces a likelihood on each candidate detection.

These frame-based probability measures are naturally integrated into a spatial-temporal model for the joint distribution of the time-varying pose parameters and the trace

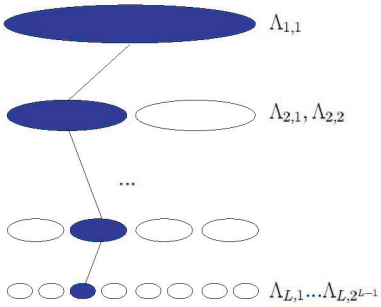

Figure 1. Hierarchical pose decomposition. Each $\Lambda_{l,k}$ represents a subset of geometric poses. An alarm is identified with a fine (leaf) cell Λ if the classifiers for every coarser cell (i.e containing Λ) respond positively.

of processing within each individual frame. A very primitive model for frame-to-frame pose transition probabilities leads to coherent tracking over the entire video sequence. Unlike existing approaches, the motion model is not used to restrict the search domain of the tracker but rather only to link detections from differing frames. The resulting algorithm unites detection and tracking in one probabilistic framework.

The rest of the paper is organized as follows: Section 2 provides an overview of the CTF face detection algorithm. In Section 3, the trace model is introduced, along with a description of how it is learned from training data. The tracking framework is presented in Section 4 and experimental results are presented in Section 5, followed by a brief discussion in Section 6.

2. Coarse-to-Fine Face Detection

Algorithmic modeling offers a new approach to pattern classification. The object of analysis is the computational process itself rather than probability distributions (Bayesian inference) or decision boundaries (predictive learning). The formulation is motivated by attempting to unite stored representations and online processing, and by applications to scene interpretation in which there are many possible explanations for the data, and one of them, "background," is statistically dominant. Computation should then by concentrated on ambiguous regions of the image.

Theoretical work [1] has shown that under certain assumptions about the tradeoffs among cost, power and invariance, a testing strategy which is coarse-to-fine (CTF) in both the representation and exploration of hypothesis yields an optimal computational design. The entire set of hypotheses (e.g., class/pose pairings) is represented as a hierarchy of nested partitions. Each cell of the hierarchy corresponds to a subset of hypotheses and is included in exactly one of the cells in the preceding, coarser partition (see Fig. 1). The partitioning is recursive and provides a CTF representation of the space of hypotheses. In order to explore the hierarchy, a test or classifier (binary random variable) is associated with each cell and is designed to respond positively to all hypotheses represented by the cell. CTF search proceeds from tests near the root which accommodate many hypotheses to those that are more dedicated and discriminative near the leaves.

The result of processing a scene results in a list of alarms (detections). More precisely, this set is the union of all fine cells $\Lambda_{L,k}$ with the property that all the tests which correspond to ancestor cells $\Lambda \supseteq \Lambda_{L,k}$ respond positively. Here L represents the last level of the hierarchy. This can be visualized as a *chain of positive responses* in the hierarchy of cells (see Fig. 1). The computation of this list of (complete) chains can be performed efficiently by evaluating a test at cell Λ if and only if all the tests corresponding to cells containing Λ (hence more invariant) have already been evaluated and responded positively. Areas of the scene rejected by coarse tests are then rapidly processed, whereas ambiguous areas are not labeled until some fine tests have been evaluated.

The face detector used in this work is based on this CTF technique where the space of hypotheses is the set of poses of a face; all the details can be found in [6]. Briefly, each test is constructed from oriented edges and is trained on a specific subset of face subimages which satisfy certain pose restrictions. In principle, the test at the root of the hierarchy would apply to *all* possible face poses simultaneously and could, for example, be based on color or motion. In our experiments, however, this first test is only virtual, assumed always positive, and the hierarchical search begins with the second level at which the position of the face (say the midpoint between the eyes) is partitioned into (non-overlapping) 8×8 blocks. More specifically, the coarsest test is designed to detect faces with tilts in the range $-20^\circ < \theta < 20^\circ$, scales (pixels between the eyes) in the range $8 < s < 16$, and location restricted to an 8×8 window. The finer cells localize faces to a 2×2 region with $\Delta\theta = 10^\circ$ and $\Delta s = 2$. In order to find larger faces, the hierarchy of classifiers is applied to every non-overlapping 8×8 image block at three different scales. The criterion for detection at a given pose is the existence of a chain in the hierarchy, from root to leaf, whose corresponding classifiers all respond positively.

The result of the detection scheme is a binary decision labeling each 8×8 image patch (at several resolutions) as face or background and providing an estimate of the pose. Although this scheme yields low error rates, it does not assign any numeric confidence measure to each detection or account for resolving competing interpretations, i.e., for

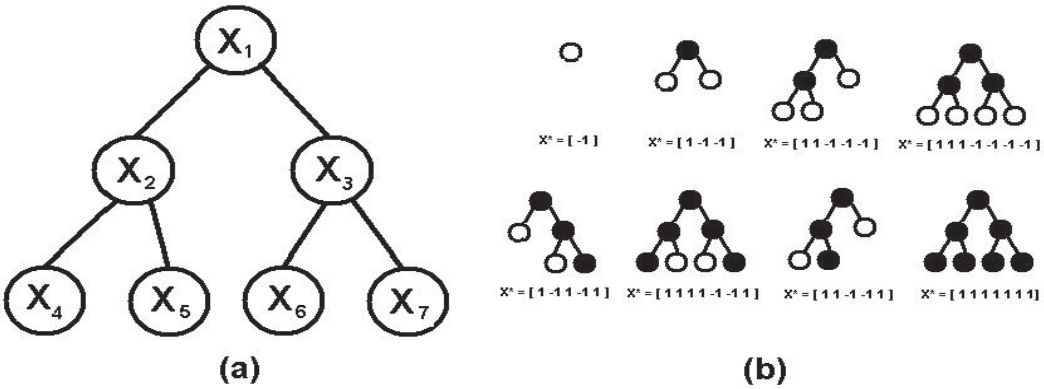

Figure 2. (a) A tree structure with 7 nodes representing a hierarchical set of classifiers used to detect an object at different poses. (b) The result of coarse-to-fine search is a labeled subtree where dark circles indicate a positive test and light circles a negative test. Eight of the 26 possible "traces" are depicted together with the outcomes of the tests performed.

spatial context. In this work, the computational process defined by CTF traversal of the hierarchy is modeled as a tree-structured stochastic network. The nature of CTF search imposes major restrictions on the possible configurations (realizations of this stochastic process) that can be observed, which in turn lead to a simple model that characterizes possible search histories or "traces." This provides a likelihood-ratio test for weeding out false detections.

3. Trace-Based Image Representation

In order to facilitate the exposition, we first describe (§3.1-§3.3) the trace model for a general hierarchy and probability distribution on the set of traces; then, in §3.4, we specialize to the case of a pose hierarchy and probability distributions conditional on pose.

3.1. Tree-Structured Networks

Let T denote the tree graph underlying the type of hierarchy described in §2 and let $\{X_\eta, \eta \in T\}$ be the corresponding set of binary tests or classifiers. We write $X_\eta = 1$ to indicate a positive test and $X_\eta = -1$ to indicate a negative test. Of course a (rooted) tree is a special case of a *directed acyclic graph* (DAG) in which the nodes have an implicit ordering and the edges have natural orientations, towards or away from the root. The set of parent nodes of η is denoted $\mathcal{A}_\eta$. A joint distribution P on configurations $\{-1, 1\}^T$ is determined by imposing the splitting property of DAGs [13] and well-know conditional independence assumptions. Specifically, the distribution of $\{X_\eta, \eta \in T\}$ is given by

$$P(\mathbf{X}) = P(X_\eta, \eta \in T) = \prod_{\eta \in T} P(X_\eta | X_\xi, \xi \in \mathcal{A}_\eta). \quad (1)$$

We refer to equation (1) as the "full-tree model" as there are no restrictions on *which particular tests are actually performed in any given realization.* Learning the full model, and computing realizations at many image locations and resolutions, can be difficult for large T since the number of nodes, as well as the number of parameters determining each conditional probability, increases exponentially with $|T|$.

The situation is illustrated in Fig. 2(a) for a simplified hierarchy T with seven nodes and corresponding binary classifiers $X_1, ..., X_7$. We can imagine the four leaf cells represent four possible fine pose cells. Clearly there are $2^7 = 128$ possible test realizations. However, the nature of CTF search results in certain restrictions on the subset of classifiers which are actually performed and the values they may obtain; some examples are illustrated in Fig. 2(b) (as explained below in §3.2).

3.2. Trace Configurations

The result of CTF search is a labeled subtree which encodes the set of tests performed and the values observed. As the hierarchy is traversed breadth-first CTF, certain nodes $\eta \in T$ are visited and their corresponding classifiers $X_\eta \in \{-1, 1\}$ are evaluated. The set $T^* \subset T$ of visited nodes is actually a *random subtree* as it depends on the values of the tests performed. By definition, the *trace*

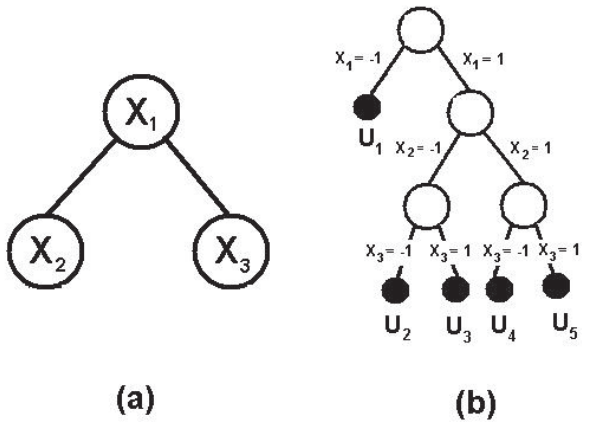

Figure 3. (a) A CTF hierarchy with 3 classifiers. (b) A decision tree representation of CTF search.

is $\mathbf{X}^* = \{X_\eta, \eta \in T^*\}$. For any trace $\mathbf{X}^*$, certain constraints result from the fact that a test X_η is performed if and only if all ancestor tests $\{X_\xi, \xi \in \mathcal{A}_\eta\}$ are performed *and* each one is positive. In particular, with ∂T^* and ∂T denoting the leaf nodes of the subtree and the full-tree respectively, we have

$$\begin{aligned} \eta \notin \partial T^* &\Rightarrow X_\eta = 1 \\ \eta \in \partial T^* \setminus \partial T &\Rightarrow X_\eta = -1 \\ \eta \in \partial T^* \cap \partial T &\Rightarrow X_\eta = \pm 1. \end{aligned}$$

(Note that an internal node of T maybe a leaf of T^* depending on the realization of tests; the test at such a node is necessarily negative.) For instance, for the hierarchy in Fig. 2(a), there are 26 distinct traces (labeled subtrees), eight of which are depicted in Fig. 2(b).

The set of all possible traces partitions the full configuration space $\{-1, 1\}^T$. By taking into account the *order* in which the tests in the hierarchy are performed, there is a natural identification with a decision tree in which each node corresponds to a particular test and each edge to one of the two possible answers. There is then a one-to-one correspondence between traces and complete paths in the decision tree. This correspondence should not lead to confusing CTF hierarchies and decision trees. Among the differences, a CTF hierarchy is a recursive partitioning of the space of hypothesis, not of the feature space, and many branches in the hierarchy may be traversed simultaneously.

The example in Figure 3 illustrates a CTF hierarchy with three classifiers and the decision tree representation of CTF search; the leaf nodes are labeled $U_1, ..., U_5$. Since every test realization $\mathbf{X}$ lands in exactly one leaf, the decision tree representation results in a mapping $\mathbf{X} \rightarrow \{U_i\}$. Of course the U_i's are disjoint and span all configurations. For any probability distribution on $\mathbf{X}$ it then follows that

$$\sum_{\mathbf{x}^*} P(\mathbf{X}^* = \mathbf{x}^*) = \sum_{U_i} P(U_i) = 1$$

where $\{\mathbf{x}^*\}$ is the set of possible traces.

3.3. Learning Trace Models

A Bayesian network model of the form (1) then induces a very simple probability distribution on traces. As above, let $\mathbf{X}^*$ be the random trace (labeled subtree) and let $\mathbf{x}^* = \{x_\eta, \eta \in T^*\}$ denote possible value. It follows from standard calculations of graphical models that:

$$\begin{aligned} P(\mathbf{X}^* = \mathbf{x}^*) &= \prod_{\eta \in T^*} P(X_\eta = x_\eta | X_\xi = x_\xi, \xi \in \mathcal{A}_\eta) \\ &= \prod_{\eta \in T^*} P(X_\eta = x_\eta | X_\xi = 1, \xi \in \mathcal{A}_\eta) \\ &= \prod_{\eta \in T^*} P_\eta(x_\eta) \end{aligned} \quad (2)$$

where $P_\eta(x_\eta) = P(X_\eta = x_\eta | X_\xi = 1, \xi \in \mathcal{A}_\eta)$. The conditional probabilities in the full-model are reduced to binomial terms $P_\eta(x_\eta)$ since all the conditional events are "positive histories." *Consequently, specifying a single parameter $P_\eta(1)$ for every node $\eta \in T$ yields a global and consistent probability model on traces.* In contrast, in the full model, there are 2^k parameters required to specify each conditional probability for a history of length k.

3.4. Pose Hierarchy

We now return to the case outlined in §2, namely a recursive partitioning of the set of positions, tilts and scales of a face. (As before, we fix the scale in the range $8 - 16$ pixels and detect larger faces by repeating the search on progressively downsampled images. This could easily be accommodated within the existing framework by adding another level at the top of the hierarchy.)

The decomposition in (2) still holds for an observed trace conditional on the pose θ. The probability of observing a trace $\mathbf{x}^*$ given the true pose θ is

$$P(\mathbf{X}^* = \mathbf{x}^* | \theta) = \prod_{\eta \in T^*} P_\eta(x_\eta | \theta). \quad (3)$$

The entire trace model is used to make inferences about the pose θ. Given an image and its corresponding trace $\mathbf{X}^*$, likelihoods of individual poses are based on the *posterior* probability $P(\theta | \mathbf{X}^*)$. As usual, analyzing $P(\theta | \mathbf{X}^*)$ depends only on the "data model" $P(\mathbf{X}^* | \theta)$ and the "prior" model $P(\theta)$, which, for a single frame, can be taken as uniform and henceforth disregarded. (This of course will change when time is incorporated.)

Recall that the test at the root is virtual and the first level of the hierarchy corresponds to partitioning the image pixels into non-overlapping 8×8 blocks, say $[W_1, ..., W_n]$, where n is the total number of blocks. Let $\mathbf{X}^*(i)$ denote the trace corresponding to block i. Given a pose θ, we will write $i(\theta)$ for the (unique) block containing the position in θ. Writing $\mathbf{X}^* = [\mathbf{X}^*(1), ..., \mathbf{X}^*(n)]$, we make several simplifying assumptions about the conditional joint distribution of these components: 1) The trace components are conditionally independent given θ; 2) The trace components are *identically* distributed in the sense that $P(\mathbf{X}^*(i(\theta))|\theta)$ does not depend on the block $i(\theta)$; 3) For $i \neq i(\theta)$, the distribution of $\mathbf{X}^*(i)$ given θ follows a universal "background law", denoted $P(\mathbf{X}^*(i)|B)$. Consequently, the overall likelihood is decomposed as follows:

$$\begin{aligned} P(\mathbf{X}^*|\theta) &= \prod_{i=1}^{n} P(\mathbf{X}^*(i)|\theta) \\ &= P(\mathbf{X}^*(i(\theta))|\theta) \prod_{i \neq i(\theta)} P(\mathbf{X}^*(i)|\theta) \\ &= P(\mathbf{X}^*(i(\theta))|\theta) \prod_{i \neq i(\theta)} P(\mathbf{X}^*(i)|B) \\ &= \frac{P(\mathbf{X}^*(i(\theta))|\theta)}{P(\mathbf{X}^*(i(\theta))|B)} \prod_{i=1}^{n} P(\mathbf{X}^*(i)|B) \\ &= \frac{P(\mathbf{X}^*(i(\theta))|\theta)}{P(\mathbf{X}^*(i(\theta))|B)} \times C(\mathbf{X}^*) \end{aligned} \quad (4)$$

where $C = C(\mathbf{X}^*)$ does not depend on θ and hence can be disregarded when making inferences about the pose. Given θ, the full likelihood requires an evaluation of the likelihood of one 8×8 block under both the "object model" and the background model. These assumptions simplify the modeling process as we only consider learning responses of classifiers for all poses contained within a single, reference block.

The parameters $P_\eta(1)$ are learned for the object model by accumulating the results of classification tests over a standard face database. For each pose, we synthesize $N_t = 10000$ training instances which satisfy the given pose requirement. Each training instance is then processed by the entire classification hierarchy. For every training instance k and node η, a record is maintained of whether a node test was performed, denoted $\pi(\eta, k) \in \{0, 1\}$ and whether the node test succeeded, denoted $\alpha(\eta, k) \in \{0, 1\}$. Parameters are then estimated by collecting simple counts over the entire set of training instances as follows:

$$\hat{P}_\eta(1) = \frac{\sum_{k=1}^{N_t} \alpha(\eta, k)}{\sum_{k=1}^{N_t} \pi(\eta, k)}$$

The background model is learned in a similar way with training instances from a non-face database.

Figure 4. Tracking results from different video sequences.

4. Markov Tracking Model

The trace model provides a likelihood function $P(observation|state)$ and hence could be used in conjunction with various probabilistic-based tracking approaches, e.g., the Kalman filter and the condensation filter [10]. In this section a simple Markov-based setting is chosen. We write $\mathbf{I}_{0:t-1}$ and $\theta_{0:t-1}$ to denote the set of observed image frames and the set of observed poses from time 0 to $t-1$. The trace of frame $\mathbf{I}_t$ is denoted by $\mathbf{X}_t^*$. The tracking problem is formulated by estimating the pose of a face at time t given

- The new trace $\mathbf{X}_t^*$;
- The previously recorded set of traces $\mathbf{X}_{0:t-1}^*$;
- The previously estimated poses $\theta_{0:t-1}$.

The estimate of the pose $\hat{\theta}_t$ is given by the MAP estimator

$$\begin{aligned} \hat{\theta}_t &= \arg\max_{\theta_t \in \Theta} P(\theta_t | \mathbf{X}_{0:t}^*, \theta_{0:t-1}) \\ &= \arg\max_{\theta_t \in \Theta} \frac{P(\mathbf{X}_{0:t}^*, \theta_{0:t})}{P(\mathbf{X}_{0:t}^*, \theta_{0:t-1})} \\ &= \arg\max_{\theta_t \in \Theta} P(\mathbf{X}_{0:t}^*, \theta_{0:t}) \\ &= \arg\max_{\theta_t \in \Theta} P(\mathbf{X}_t^*, \theta_t | \mathbf{X}_{0:t-1}^*, \theta_{0:t-1}) \end{aligned}$$

where at every step we have dropped the terms which are independent of θ_t and we have assumed that the trace/pose process $(\mathbf{X}_t^*, \theta_t), t \geq 0$ is jointly Markov. To further simplify the computations, we assume that i) Given the current pose θ_t, the current trace $\mathbf{X}_t^*$ is independent of the previous trace/pose pair and ii) Given the previous pose θ_{t-1}, the current pose θ_t is independent of the previous trace $\mathbf{X}_{t-1}^*$. This

Figure 5. Top row: The result of our tracker in four different frames. Bottom row: The raw results of pure detection in the same four frames.

results in the following baseline tracker:

$$\begin{aligned}\hat{\theta}_t &= \arg\max_{\theta_t \in \Theta} P(\mathbf{X}_t^*, \theta_t | \mathbf{X}_{t-1}^*, \theta_{t-1}) \\ &= \arg\max_{\theta_t \in \Theta} P(\mathbf{X}_t^* | \mathbf{X}_{t-1}^*, \theta_t, \theta_{t-1}) P(\theta_t | \mathbf{X}_{t-1}^*, \theta_{t-1}) \\ &= \arg\max_{\theta_t \in \Theta} P(\mathbf{X}_t^* | \theta_t) P(\theta_t | \theta_{t-1}). \end{aligned} \quad (5)$$

The likelihood $P(\mathbf{X}_t^*|\theta_t)$ is evaluated according to (4) in the previous section with $\mathbf{X}^*(i)$ replaced by $\mathbf{X}_t^*(i)$ and θ replaced by θ_t. The transition probability $P(\theta_t|\theta_{t-1})$ is assumed stationary and captures our prior knowledge about how the pose moves from one frame to another. In our experiments, this transition model is learned from data.

In practice, we do not search over all possible poses at each time t. Instead, the search space is restricted to a limited set of poses constructed from the union of two sets. One is the set of poses which are consistent with the estimated pose in the previous frame (in the sense that the transition probability is above some threshold). The other set consists of the full set of alarms (complete chains in the hierarchy) which are produced by the CTF detection scheme. Of course these alarms include both true detections and false positives, but evaluating these poses allows for correcting mistakes and accommodating the appearance of new faces or the re-appearance of occluded faces. The poses in the union of these two sets are then sorted by the likelihood function in (5). Tracking can be further accelerated by limiting the search space to regions that satisfy certain color and motion constraints.

5. Experimental Results

Video sequences provided by [2], which are available at *http://www.cs.bu.edu/groups/ivc/HeadTracking/*, are used in the initial experiments. The sequences contain 200 frames with a resolution of 320×240 and contain free head motion of several subjects under varying conditions of illumination.

The pose transition model is learned from a set of similar pre-recorded video sequences. These training sequences are manually landmarked and provide ground truth for pose transitions. A histogram of the pose differences $\theta_t - \theta_{t-1}$ is generated for the entire training set and serves as a good estimate for the pose transition model $P(\theta_t|\theta_{t-1})$.

In Fig. 4, we show some of the results obtained with the face tracker. With a standard desktop PC and with no MMX optimizations, faces are tracked at around 10 frames per second. Since the evaluation of trace likelihoods is restricted to regions of interest, the speed of the tracker is mainly determined by the efficiency of detection. Real-time performance can be obtained by only executing the full-image detector every few frames or by incorporating global temporal information.

In Fig. 5, we illustrate the difference in the quality of single-frame detection between the dynamic tracking model and the static, frame-by-frame face detector (i.e., without the trace model, as implemented in the cited references). Tracking yields both a higher detection rate and a lower false positive rate. A higher detection rate is achieved because the tracker exploits the temporal information to occasionally estimate a pose which does not correspond to a de-

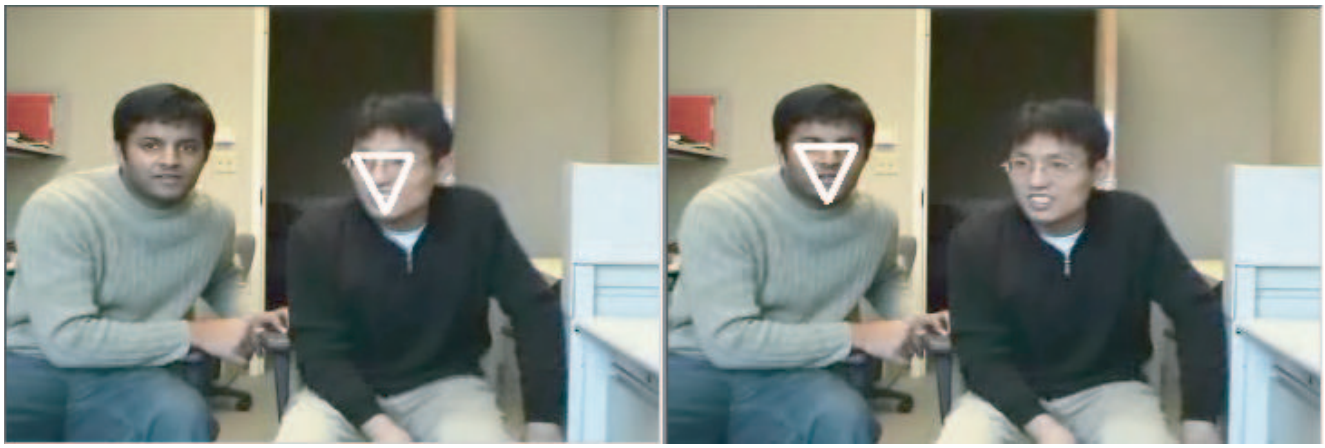

Figure 6. Tracking results in two consecutive frames on a sequence with two faces. Since the current tracking model assumes a single face, it occasionally jumps from one face to another when the (normalized) likelihood of a detection dominates in the MAP estimation of the pose.

Figure 7. Tracking results on a difficult sequence with high camera instability.

tected alarm. This phenomenon is mainly observed in cases in which the pose of the subject temporarily violates the a priori constraints (e.g., on the range of tilts) or in cases of temporary occlusion. In addition, the tracker filters out false positives resulting from high-frequency noise (since the tests are based on the presence of edges).

In Fig. 6, we show two consecutive frames from a video sequence with multiple faces. As the current framework is based on MAP estimation of a single track, multiple faces cannot be tracked simultaneously. Extending the formulation to multiple faces is a subject of current research. Finally, in Fig. 7 we depict the result of tracking one individual (the singer) in a very challenging video sequence [9]. The face of the subject is successfully tracked despite heavy camera panning and unsteady focus. Unlike most tracking algorithms, the search is global and the influence of the detection model reduces the dependence on accurate motion estimation.

6. Summary and Conclusions

We have presented a new method to unite face detection and face tracking in a probabilistic framework. The online computational process of a CTF face detection algorithm is analyzed in the context of a graphical model for the history or "trace" of processing, thereby introducing a probabilistic component into the CTF face detection strategy. The resulting trace model can then be merged with pose dynamics within a single, coherent Bayesian framework to base tracking on both frame-by-frame detection and tem-

poral continuity, embedding detection in a filtering framework. Since the temporal model is extremely elementary, the encouraging experimental results can be seen to demonstrate the power of the trace model.

Unlike traditional tracking algorithms, there are no restrictions on the motion of a face. This is possible because CTF search makes detection very rapid, thereby allowing for a search for faces over each entire video frame. Conversely, this also renders detection more efficient by eliminating a significant number of hypotheses. Extending the formulation to tracking multiple faces is currently being investigated.

Acknowledgments

This work was supported in part by ONR contract N000140210053, by ARO grant DAAD19-02-1-0337 and by NSF grant DMS-0219016.

References

[1] G. Blanchard and D. Geman. Sequential testing designs for pattern recognition. *Annals of Statistics*, June 2005.

[2] M. Cascia, S. Sclaroff, and V. Athitos. Fast reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3d models. *IEEE. Trans. PAMI*, 21(6), 1999.

[3] D. Decarlo and D. Metaxas. Deformable model based face shape and motion estimation. *Proc. Int'l Conf. Auto. Face and Gesture Recognition*, 1996.

[4] C. Edwards, C. Taylor, and T. Cootes. Learning to identify and track faces in an image sequence. *Proc. Int'l Conf. Auto. Face and Gesture Recognition*, pages 260–265, 1998.

[5] C. Eveland, D. Socolinsky, C. Priebe, and D. Marchette. A hierarchical methodology for class detection problems with skewed priors. *Journal of classification*, 2005. inpress.

[6] F. Fleuret and D. Geman. Coarse-to-fine face detection. *International Journal of Computer Vision*, 41:85–107, 2001.

[7] G. Hager and K. Toyama. X vision: A portable substrate for real time vision applications. *Computer Vision and Image Understanding*, 69(1):23–37, 1998.

[8] E. Hjelmas and B. Low. Face detection: A survey. *Computer Vision and Image Understanding*, pages 236–274, 2001.

[9] http://www.madonnalicious.com/downloads.html.

[10] M. Isard and A. Blake. Condensation-conditional density propagation for visual tracking. *IJCV*, 29:5–28, 1998.

[11] B. Li and R. Chellappa. A generic approach to simultaneous tracking and verification in video. *IEEE Transactions Image Processing*, 11:530–544, 2002.

[12] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. *Proceedings IEEE CVPR*, pages 130–136, 1997.

[13] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1998.

[14] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions PAMI*, 20:23–38, 1998.

[15] H. Schneiderman and T. Kanade. Probabilistic modeling of local appearance and spatial relationships for object recognition. *IEEE Proceedings CVPR*, pages 45–51, 1998.

[16] K. Schwerdt and J. Crowley. Robust face tracking using colour. *Proc. Int'l Conf. Auto. Face and Gesture Recognition*, pages 90–95, 2000.

[17] K. Sung and T. Poggio. Example-based learning for view-based face detection. *IEEE Transactions PAMI*, 20:39–51, 1998.

[18] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. *Proceedings IEEE CVPR*, 2001.

[19] M. Yang, D. Kriegman, and N. Ahuja. Detecting faces in images: A survey. *IEEE Trans. PAMI*, pages 34–58, 2002.