# Capturing People in Surveillance Video

Rogerio Feris, Ying-Li Tian, and Arun Hampapur
IBM T.J. Watson Research Center
PO BOX 704, Yorktown Heights, NY 10598
{rsferis,yltian,arunh}@us.ibm.com

## Abstract

*This paper presents reliable techniques for detecting, tracking, and storing keyframes of people in surveillance video. The first component of our system is a novel face detector algorithm, which is based on first learning local adaptive features for each training image, and then using Adaboost learning to select the most general features for detection. This method provides a powerful mechanism for combining multiple features, allowing faster training time and better detection rates. The second component is a face tracking algorithm that interleaves multiple view-based classifiers along the temporal domain in a video sequence. This interleaving technique, combined with a correlation-based tracker, enables fast and robust face tracking over time. Finally, the third component of our system is a keyframe selection method that combines a person classifier with a face classifier. The basic idea is to generate a person keyframe in case the face is not visible, in order to reduce the number of false negatives. We performed quantitatively evaluation of our techniques on standard datasets and on surveillance videos captured by a camera over several days.*

## 1. Introduction

Visual processing of people, including detection, tracking, recognition, and behaviour interpretation, is a key component of smart surveillance systems. Computer vision algorithms with the capability of "looking at people" can be applied in different surveillance scenarios, such as the detection of people entering restricted areas, or even the detection of suspicious or aggressive behaviour.

Our work is particularly focused on the analysis of human faces. For each person entering and leaving the field of view of a surveillance camera, our goal is to store in a database a keyframe containing the face image of the person, associated with a corresponding video. This allows the user to query the system like "Show me all people who entered the facility yesterday from 1pm to 5pm". The retrieved face keyframes can then be used for recognition either by a person or by an automatic face recognition system.

To achieve this goal, reliable modules for face detection, tracking, and keyframe storage are required. Although significant progress has been made in this area [11, 1, 12], current systems still suffer from slow training time and limited accuracy in challenging conditions like face pose and lighting changes.

Our first contribution is a novel face detector algorithm that uses local feature adaptation prior to Adaboost learning. Local features have been widely used in learning-based object detection systems. As noted by Munder and Gavrila [6], they offer advantages over global features such as Principal Component Analysis [14] or Fisher Discriminant Analysis [12], which tend to smooth out important details.

The novelty of our technique is to use local feature adaptation before applying the traditional Adaboost learning for feature selection. Local feature adaptation is carried out by a non-linear optimization method that determines feature parameters (such as position, orientation and scale) that match the geometric structure of each face image. In a second stage, Adaboost learning is applied to the pool of adaptive features in order to obtain general features, which encode common characteristics of all training face images and thus are suitable for detection. Compared to other techniques, our method offers faster learning time and improved detection rates.

The second contribution of our paper is a face tracking algorithm that interleaves multiple view-based face detectors along the temporal domain in a video sequence. This poses a considerable advantage over methods that run view-based detectors (e.g., frontal and profile detectors) in each frame. By interleaving the classifiers along the video sequence, a much faster frame rate is obtained, leading to less inter-frame variation and more robust tracking. We also combine this technique with a correlation-based tracker in order to follow the face when face detection fails.

Finally, we propose a keyframe selection technique that combines a face detector with a person detector. In general, due to face occlusion or false negatives in face detection,

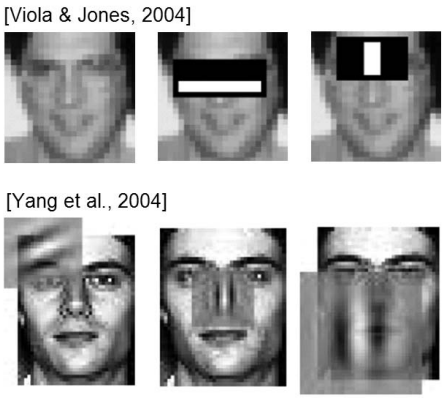


Figure 1. Selected features from Adaboost in the context of face detection (top) and face recognition (bottom). Note that the features tend to adapt to the local face structure.

important events of people entering and leaving the scene might be missed. Our technique basically outputs a face keyframe in case the face is detected and a person keyframe otherwise.

In the following sections, we will give more details about the three main components of our system: face detection, face tracking, and keyframe selection.

## 2. Face Detection

Despite the progress made on face detection techniques over the last years [7, 11, 9, 2], existing systems still have limitations. First, they require thousands of samples to learn a robust classifier. Levi and Weiss [4] recently showed that the choice of features plays a major role in learning object detection from a small number of examples.

Another problem is the slow training time due to a large feature pool and brute-force feature selection. Most methods have hundreds of thousands of features in the pool and take order of weeks for training in conventional machines. Also, they can not be easily applied for other types of objects. For example, Haar wavelet features have shown excellent results for frontal face detection, but have limited discrimination power in other domains, even in profile view face detection [12].

Finally, existing methods do not offer support for integration of multiple features. In fact, integrating different types of features such as Haar, Gabor, and other wavelet filters, with multiple frequencies and orientations, would imply an extremely large set of possible feature configurations in the pool, which is not practical.

We propose a novel framework to overcome the above limitations by combining and selecting multiple types of visual features in object detection. Our approach relies on the observation that selected local features tend to match the local structure of the object. Figure 1 shows the first selected

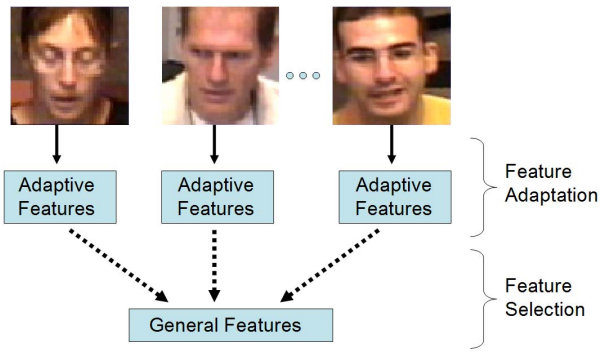


Figure 2. Our approach has two stages: first we compute adaptive features for each training sample and then use a feature selection mechanism to obtain general features.

features by Adaboost in the context of face detection [11] and recognition [13]. Note how the features encode that the eye region is darker than the nose region, for example. In fact, Liu and Shum [5], in their KullBack-Leibler boosting framework, argue that features should resemble the face semantics, matching the local or global face structure.

Based on this observation, we propose a two-stage technique that learns adaptive and general features for face detection. In the first stage, we learn adaptive features that adapt to each particular sample in the training set. This is carried out by a non-linear optimization method that determines feature parameters (such as position, orientation and scale) that match the geometric structure of each object sample. In the second stage, Adaboost feature selection is applied to the pool of adaptive features in order to obtain general features, which encode common characteristics of all object samples and thus are suitable for detection (see Figure 2).

### 2.1. Learning Adaptive Features

Given a particular object image, our goal is to learn the parameters of wavelet features (position, scale, and orientation) so that they match the local object structure. This is similar in spirit to the idea of wavelet networks proposed by Zhang [15] and introduced in computer vision by Krueger [3].

We start by taking a family of $N$ two-dimensional wavelet functions $\Psi = \{\psi_{n_1}, \ldots, \psi_{n_N}\}$, where $= \psi_{n_i}(x, y)$ is a particular mother wavelet (e.g., Haar, Gabor, etc.) with parameters $n_i = (c_x, c_y, \theta, s_x, s_y)^T$. Here, $c_x$, $c_y$ denote the translation of the wavelet, $s_x$, $s_y$ denote the dilation, and $\theta$ denotes the orientation. The choice of $N$ is arbitrary and related to the degree of desired representation precision.

Let $I$ be an input object image. First, we initialize the set of wavelets $\Psi$ along the image in a grid, as shown in Figure 3a, with the wavelets having random orientations, and


Figure 3. Learning adaptive features for a particular training image. (a) Input training image with wavelets initialized as a grid along the face region, with random orientations and scales. (b)-(e) Wavelet features being optimized one by one, with parameters (position, scale, and orientation) selected to match the local structure of the input face image.

scales initialized to a constant value that is related to the density with which the wavelets are distributed. Then, assuming $I$ is dc-free, without loss of generality, we minimize the following energy functional:

$$E = \min_{n_i, w_i \text{ for all } i} \|I - \sum_i w_i \psi_{n_i}\|^2 \quad (1)$$

with respect to the weights $w_i$ and the wavelet parameter vectors $n_i$. We used the Levenberg-Marquard method for the optimization process. Figure 3b-d shows the wavelet features being optimized one by one to match the local image structure of the object. In this example, a Gabor wavelet was adopted as mother wavelet.

It is important to note that the parameters $n_i$ are chosen from the *continuous* domain and the wavelets are positioned with sub-pixel accuracy, contrasting with most existing discrete approaches. This assures that a maximum of the image information can be encoded with only a small number of wavelets.

## 2.2. Learning General Features

Let $\chi = \{I_1, \ldots, I_M\}$ be a set of object training images. For each image $I_i$, we generate a set of adaptive features $\Psi_i$, using the optimization method described in Section 2.1.

All the generated adaptive features for all the object images are then put together in a single pool of features $\Omega$, defined as:

$$\Omega = \bigcup_{i=1}^{M} \Psi_i \quad (2)$$

For selection of general features, we used Adaboost learning in the same way of Viola and Jones [11] to both select features and project a classifier. A large set of non-object (background) images is used in addition to the object training images. The general features are those which best separate the whole set of object samples from non-object samples during classification. We refer the reader to [11] for more details about the Adaboost classifier and the feature selection mechanism.

The object detector classifier is applied in all possible image locations and scales. We used a cascade technique [11] to improve the efficiency of this operation. However, even using a cascade classifier, real-time performance can not be achieved due to the time required to compute our features. We solved this problem by using traditional Haar-like/rectangle features in the first levels of the cascade. This allows for efficient rejection of background patches during classification. The image patches that are not filtered by the Haar cascade are then fed into a cascade of classifiers using our features. The choice of which cascade level should be used to switch from Haar features to our features is application dependent. Switching in lower levels allows for more accuracy, but switching in higher levels allows for more efficiency.

## 2.3. Implementation

In our experiments, we used a frontal face dataset containing 4000 faces for training purposes. Each training image was rescaled and cropped to a 24x24 patch size. A pool of adaptive features was generated by running the optimization process described in Section 2.1 for each sample. We used different types of wavelets for different training samples, including Haar and Gabor filters with different frequencies. The total number of features in the pool was 80000, as we generated 20 adaptive wavelet features per sample. It takes less than a second to generate features for a particular 24x24 sample in a conventional 3GHz desktop computer.

For learning general features, we used a database of about 1000 background (non-faces) images from which 24x24 patches are sampled. A cascade classifier was trained by considering 4000 faces and 4000 non-faces at each level, where the non-face samples were obtained through bootstrap [7]. Each level in the cascade was trained to reject about half of the negative patterns, while correctly accepting 99.9% of the face patterns. A fully trained cascade con-

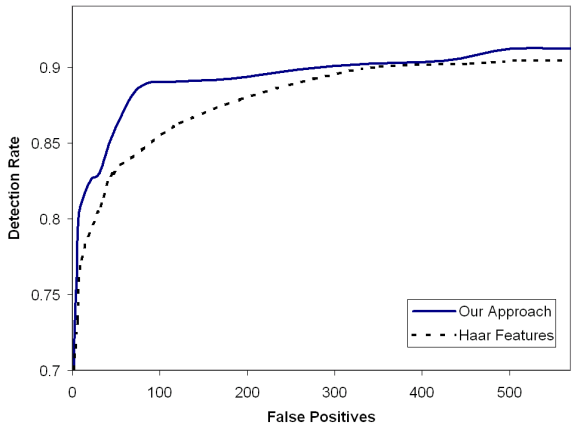


Figure 4. ROC Curves for our approach and traditional Haar features in a frontal face test dataset. We used only half of the number of features in the feature pool compared to Haar features and still get superior performance.

sisted of 24 levels.

During detection, a sliding window was moved pixel by pixel at different image scales. Starting with the original scale, the image was rescaled by a factor of 1.2 in each iteration. Multiple overlapping detection results were merged to produce a single result for each location and scale.

The CMU+MIT frontal face test set, containing 130 grayscale images with 511 faces was used for evaluation. A face is considered to be correctly detected if the Euclidean distance between the center of the detected box and the ground-truth is less than 50% of the width of the ground-truth box, and also if the width (i.e., size) of the detected face box is within $\pm 70\%$ of the width of the ground-truth box.

Figure 4 shows a plot of Receiver Operating Characteristic (ROC) curves, comparing our approach with traditional Haar-like/rectangle features in the CMU+MIT test set. The Haar detector was also trained with a cascade consisting of 24 levels in the same training set. The feature pool, however, was twice as large than our approach, containing about 160000 features. With half of the features in the pool, we achieve superior performance and a faster learning time (see table 1). A Haar filter corresponding to the first 18 levels of the cascade was used in our approach for generating the ROC curve.

We have also done preliminary training on profile face detection. Similarly to frontal face training, we used a database of 4000 faces and 80000 adaptive features. Only full profile faces were considered. In order to report results on standard CMU profile datasets, we still need to train additional detectors for half-profile and profile faces with different in-plane rotations.

| Feature Pool | Number of Features | Learning Time |
|---|---|---|
| Haar Features | 160000 | around 5 days |
| Our Approach | 80000 | around 3 days |

Table 1. By learning adaptive and general features, we can use a smaller feature pool, which allows reduced training time, while still keeping superior performance in detection rate, when compared to a traditional pool of Haar features.

## 2.4. Discussion

Although we have quantitatively compared our method only with Haar features, we believe that our technique has advantages over other methods that use more complex features. As recently demonstrated by Munder and Gavrila [6], local adaptive features have more discriminative power than global features, such as those obtained by Principal Component Analysis or Fisher Discriminant Analysis. Global features tend to smooth out details, which are captured by local adaptive features. In addition, our method provides a principled mechanism to create a dictionary with a large variety of meaningful, optimized features. We believe this is an important contribution on the feature level for learning-based object detection. Our approach could be combined with state of the art techniques on the learning algorithm level [1, 8] to obtain even better results.

Although adaptive features are projected to reconstruct an object image, they play an important role in discrimination, as motivated in Section 2. An alternative approach would be to bypass the learning of adaptive features and determine the optimal local feature parameters (position, scale, and orientation) to maximize a discrimination criterium between object and non-object samples directly. However, this would not be feasible computationally, especially if we are targeting subpixel precision.

Current feature selection methods sample the feature space at discrete positions, scales and orientations. The pool of features therefore contains many configurations that are not useful for classification. In addition, important feature configurations may not be included in the feature pool, due to the sampling process. This becomes more problematic when multiple features of different types are considered. In our method, only meaningful features that match the local structure of the object samples, with subpixel precision, are included in the pool.

Our approach is related to the work of Krueger [3], who uses adaptive Gabor wavelets as adaptive features for object representation. His work is applicable for object recognition, but not for object detection, since only a single training image is considered and no general features are learned.

A key advantage of our method is the integration of multiple types of features to describe the appearance of an object. In addition to wavelet filters, other types of features,

such as local edge orientation histograms [4], could be optimized in a similar way. An interesting observation is that the wavelet filters with large coefficients in fact tend to align with the orientation of local intensity edges in the image. Hence, edge-based classifiers could be projected based only on the spatial support of the wavelet features.

The fact that the features adapt to the training set makes our method suitable to detect different objects other than faces. This contrasts with traditional Haar-like/rectangle features, which are more appropriated for symmetric objects like frontal faces.

## 3. Face Tracking

The second component of our system is a face tracking algorithm. After the face is detected in a particular video frame, tracking the face is required to analyze the trajectory of the person and enable just a single keyframe of the face to be stored in a database.

Our face tracking method is based on applying face detection in every frame of the video sequence. In order to keep tracking the face even when the face detector fails, we also use a simple correlation-based tracker. More specifically, when a face is detected, the correlation-based tracker is triggered. For the subsequent frame, if the face detection fails, tracking is updated with the window given by the correlation tracker. Otherwise, if the face detector reports a window result with close position and size to the current tracking window, then this face detection window result is used to update tracking. This mechanism is important to avoid drifting.

In order to improve the efficiency of our detector and allow real-time face tracking performance (25/30Hz) in conventional desktop computers, we apply the following techniques:

- We only apply the detector at specific scales provided by the user and at motion regions detected by background subtraction.
- An interleaving technique (explained below) is applied to combine view-based detectors and tracking.

In most surveillance scenarios, human faces appear in images in a certain range of scales. In our system, the user can specify the minimum and maximum possible face sizes for a particular camera, so that face detection is applied only for sub-windows within this range of scales. We also apply background subtraction, using statistical mixture modeling [10], to prune sub-windows that do not lie in motion regions.

One problem faced by most existing systems is the computational time required to run a set of view-based detectors

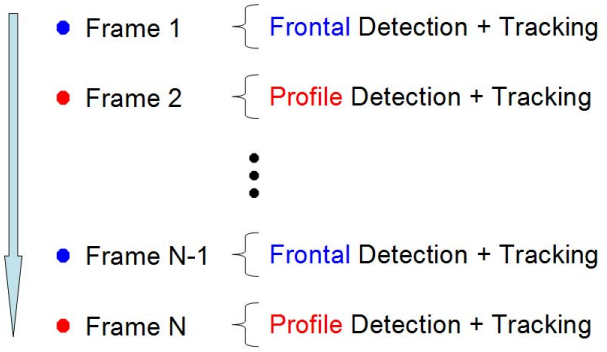


Figure 5. Our surveillance system interleaves view-based detectors to save frame rate for tracking.

in each frame. This causes large inter-frame image variation, posing a problem for tracking. We handled this issue by using an interleaving technique that alternates view-based detectors in each frame. This idea is illustrated for frontal and profile face detection in Figure 5. In each frame, rather than running both frontal and profile detectors, we just run one detector for a specific view (e.g., frontal view). The detector for the other view (profile view) is applied in the subsequent frame and so forth. This allow us to improve frame rate by 50%, while facilitating tracking due to less inter-frame variation.

Tracking is terminated when there is no foreground regions (obtained from the background subtraction module) near the current tracking window or when the face detector fails consecutively for a given time or number of frames specified by the user.

## 4. KeyFrame Selection

In addition to detecting and tracking human faces, we also store a keyframe for each captured face image in a database. In our system, the keyframe is an image which contains only the face of the person, not the full frame. A timestamp associated with each keyframe is also stored, allowing the system to answer questions like"Show me all people who entered the facility yesterday from 1pm to 5pm". The stored face keyframes can also be used for recognition either by a person or by an automatic face recognition system.

For each person that enters and leaves the scene, only one keyframe of its face needs to be stored in the database. Given the tracking information for this person, we select the face image with maximum resolution that was detected by the face detector. We give priority to frontal faces, meaning that a frontal face keyframe would be selected even if a higher resolution profile face image was present along the tracking.

Associated with each face keyframe, we also store a link

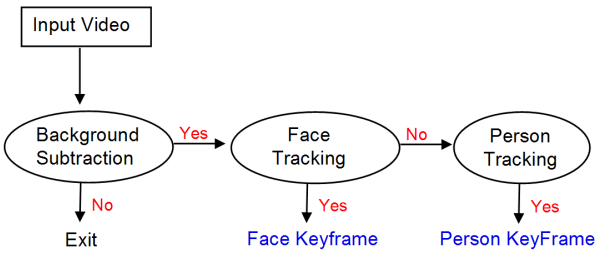


Figure 6. Our keyframe selection technique is used to ouput a person keyframe in case the face is not visible.

to a video sequence corresponding to the period in which the person was tracked. Then, the user can not only search for faces, but also click on a specific face keyframe to get a video showing what that particular person was doing.

### 4.1. Zero False Negatives

Ideally, for every person leaving the scene, a face keyframe would be generated and stored in the database. However, due to false negatives in face detection, important events might be missed. Moreover, in our surveillance scenario, depending on the direction that the person is walking, only a back view of the person may be captured. In this case, the face is not visible and no keyframes are generated.

We address this problem by using a keyframe selection technique that combines a face classifier with a person classifier (see Figure 6). If a face is detected and tracked in the video sequence, a face keyframe is stored in the database. Otherwise, a person keyframe is generated if a person is detected and tracked in the video.

A simple person classifier was used in our system. We segment motion blobs based on background subtraction and use measurements such as area and aspect ratio to classify the blob as a person. The goal is to have a system that does not miss any event.

Our keyframe selection technique is a simple, but useful feature to be part of a surveillance system. We provide an interface which includes search based only on face keyframes (see Figure 7) and a more sensitive search which includes both person and face keyframes (see Figure 8). Search based only on face keyframes is useful to rapidly recognize people in surveillance video. The search based on face and person keyframes is useful to guarantee that no event was missed.

We captured data from a surveillance camera for ten consecutive days and analysed ten hours of this data, with each hour corresponding to the peak hour (i.e., the hour with most people entering the facility) in each day. Table 2 shows our results. Out of 445 people entering the facility (not walking back to the camera), we captured 351 faces, with only 7 false positives. The reason that some faces were missed is that sometimes people enter the door looking down, occluding the face from the camera, which is placed in the ceiling. By running our keyframe selection technique, we can capture all remaining 94 persons, as well as 40 persons walking back to the camera, with more 19 false positives.

| | |
|---|---|
| Captured Faces | 351 |
| Face False Positives | 7 |
| Detected Extra Persons | 134 |
| People Walking Back | 40 |
| Person False Positives | 19 |

Table 2. Results obtained from ten hours of surveillance video.

We plan to present a demo of our system at the conference, showing our techniques being applied to live surveillance video.

## 5. Conclusions

We have presented reliable techniques for face detection, tracking and keyframe selection in surveillance video. Our methods were incorporated in a smart surveillance system which is currently being commercialized. The face detection component is based on local feature adaptation prior to Adaboost learning. This technique offers better detection rates and faster training than traditional methods based on Haar features. It also allows the integration of a large dictionary of features in a principled way. Our face tracking component runs in real-time by using multiple interleaved classifiers along the temporal domain. Finally, our keyframe selection technique is useful to provide the user keyframes of persons when the face is not visible, so that no event is missed.

As future work, we plan to test our object detection and tracking approach with different objects and carry out a more extensive evaluation, analyzing parameters such as the size of the feature pool and the training set. We expect to have improved performance with larger feature pools and better ability to learn from a small number of examples.

## References

[1] C. Huang, H. Ai, Y. Li, and S. Lao. Vector boosting for rotation invariant multi-view face detection. In *IEEE International Conference on Computer Vision (ICCV'05)*, Beijing, China, October 2005. 1, 4

[2] C. Huang, H. Ai, Y. Li, and S. Lao. Learning sparse features in granular space for multi-view face detection. In *International Conference on Automatic Face and Gesture Recognition*, Southampton, UK, April 2006. 2


Figure 7. Example showing search results for face capture (faces intentionally blurred for privacy issues).


[3] V. Krueger. *Gabor wavelet networks for object representation*. PhD thesis, Christian-Albrecht University, Kiel, Germany, 2001. 2, 4

[4] K. Levi and Y. Weiss. Learning object detection from a small number of examples: the importance of good features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'04)*, Washington, DC, July 2004. 2, 5

[5] C. Liu and H. Shum. Kullback-leibler boosting. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'03)*, Madison, Wisconsin, June 2003. 2

[6] S. Munder and D. Gavrila. An experimental study on pedestrian classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(11):1863–1868, 2006. 1, 4

[7] H. Rowley, S. Baluja, and T. Kanade. Neural network-based face detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(1):23–38, 1998. 2, 3

[8] R. Schapire and Y. Singer. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37:297–336, 1999. 4

[9] H. Schneiderman and T. Kanade. A statistical approach to 3d object detection applied to faces and cars. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'00)*, Hilton Head Island, South Carolina, June 2000. 2

[10] Y. Tian, M. Lu, and A. Hampapur. Robust and efficient foreground analysis for real-time video surveillance. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, June 2005. 5

[11] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'01)*, Kauai, Hawaii, December 2001. 1, 2, 3

[12] P. Wang and Q. Ji. Learning discriminant features for multiview face and eye detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*, San Diego, CA, June 2005. 1, 2

[13] P. Yang, S. Shan, W. Gao, S. Li, and D. Zhang. Face recognition using ada-boosted gabor features. In *International Con-*

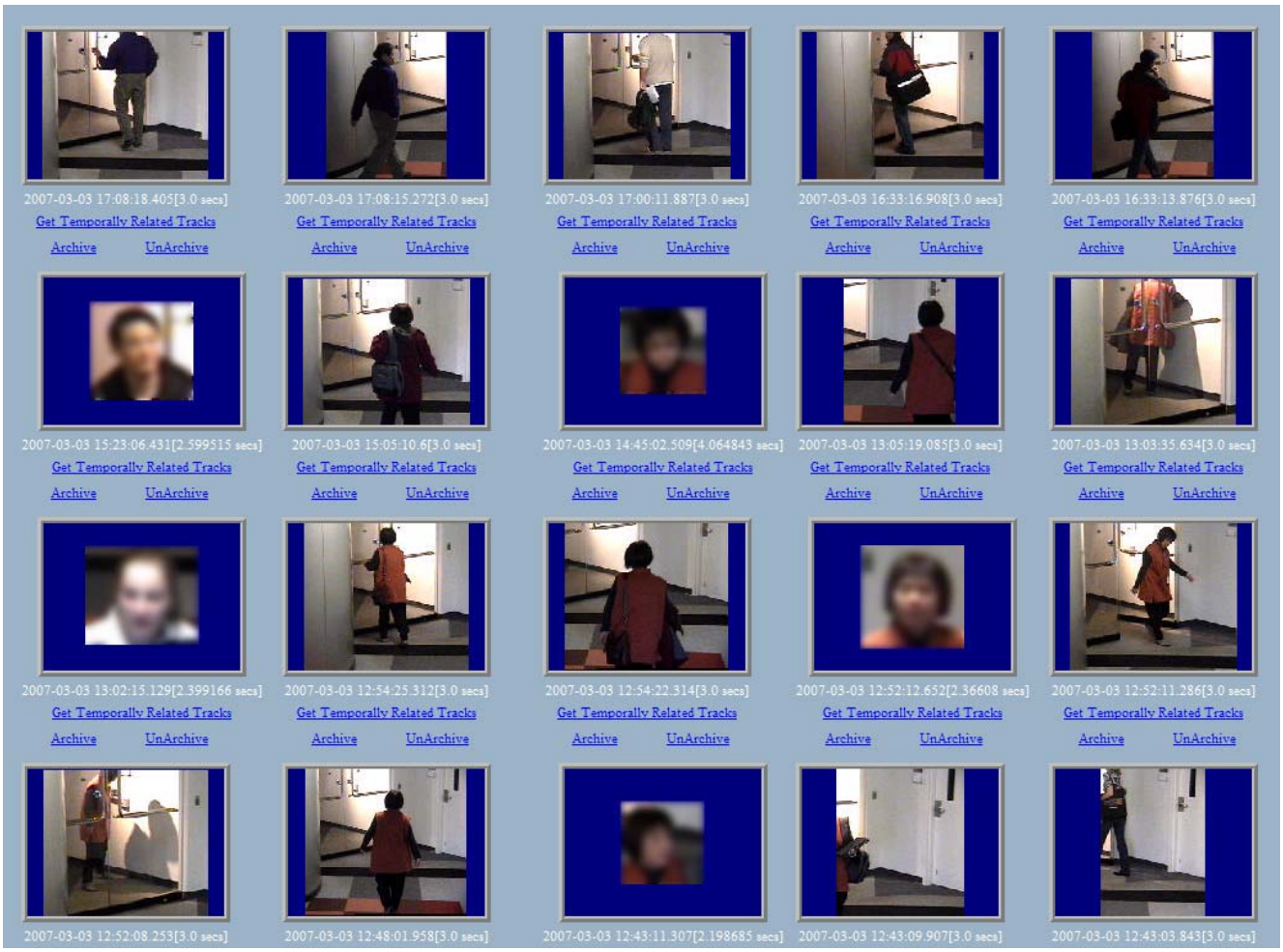


Figure 8. Example showing search results using our keyframe selection technique (faces intentionally blurred for privacy issues).

*ference on Automatic Face and Gesture Recognition*, Seoul, Korea, May 2004. 2

[14] D. Zhang, S. Li, and D. Gatica-Perez. Real-time face detection using boosting in hierarchical feature spaces. In *International Conference on Pattern Recognition (ICPR'04)*, Cambridge, UK, August 2004. 1

[15] Q. Zhang. Using wavelet network in nonparametric estimation. *IEEE Transactions on Neural Networks*, 8(2):227–236, 1997. 2