Multi-Scale People Detection and Motion Analysis for Video Surveillance

YingLi Tian [1], Rogerio Feris [2], Lisa Brown[2], Daniel Vaquero [3], Yun Zhai [2], Arun Hampapur [2]

[1]*Department of Electrical Engineering*
City College, City University of New York, New York, NY
ytian@ccny.cuny.edu

[2] *IBM T. J. Watson Research Center, Hawthorne, NY*
{rsferis,lisabr,yunzhai,arunh}@us.ibm.com

[3] *Department of Computer Science*
University of California, Santa Barbara
daniel@cs.ucsb.edu

Visual processing of people, including detection, tracking, recognition, and behavior interpretation, is a key component of intelligent video surveillance systems. Computer vision algorithms with the capability of "looking at people" at multiple scales can be applied in different surveillance scenarios, such as far-field people detection for wide-area perimeter protection, mid-field people detection for retail/banking applications or parking lot monitoring, and near-field people/face detection for facility security and access. In this chapter, we address the people detection problem in different scales as well as human tracking and motion analysis for real video surveillance applications including people search, retail loss prevention, people counting, and display effectiveness.

Keywords: Video surveillance, object detection, face tracking, human tracking, motion analysis, color classification, people search, multiple scales.

1. INTRODUCTION

As the number of cameras deployed for surveillance increases, the challenge of effectively extracting useful information from the torrent of camera data becomes formidable. The inability of human vigilance to effectively monitor surveillance cameras is well recognized in the scientific community [Green 1999]. Additionally, the cost of employing security staff to monitor hundreds of cameras by manually watching videos is prohibitive.

Intelligent (smart) surveillance systems, which are now "watching the video" and providing alerts and content-based search capabilities, make the video monitoring and investigation process

scalable and effective. The software algorithms that analyze the video and provide alerts are commonly referred to as video analytics. These are responsible for turning video cameras from a mere data gathering tool into smart surveillance systems for proactive security. Advances in computer vision, video analysis, pattern recognition, and multimedia indexing technologies have enabled smart surveillance systems over the past decade.

People detection, tracking, recognition, and behavior interpretation play very important roles in video surveillance. For different surveillance scenarios, different algorithms are employed to detect people in distinct scales, such as far-field people detection for wide-area perimeter protection, mid-field people detection for retail/banking applications or parking lot monitoring, and near-field people/face detection for facility security and access. People detection and tracking has been an active area of research. The approaches for people detection can be classified as either model-based or learning-based. The latter can use different kinds of features such as edge templates [Gavrila 2000], Haar features [Viola *et al.* 2001, 2003], histogram-of-oriented-gradients descriptors [Dalal & Triggs 2005, Han *et al.* 2006], shapelet features [Sabzmeydani 2007], etc. To deal with occlusions, some approaches use part-based detectors [Wu & Nevatia 2005, Leibe 2005].

In our system, learning-based methods are employed to detect humans at different scales. For each person entering and leaving the field of view of a surveillance camera, our goal is to detect the person and to store in a database a key frame containing the image of the person, associated with a corresponding video. This allows the user to perform queries such as "Show me all people who entered the facility yesterday from 1pm to 5pm." The retrieved key frames can then be used for recognition, either manually or by an automatic face recognition system (if the face image is available). To achieve this goal, we developed 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 [Munder & Gavrila 2006], they offer advantages over global features such as Principal Component Analysis [Zhang *et al.* 2004] or Fisher Discriminant Analysis [Wang & Ji 2005], which tend to smooth out important details.

In order to detect trajectory anomalies, our system tracks faces and people, analyzes the paths of tracked people, learns a set of repeated patterns that occur frequently, and detects when a person moves in a way inconsistent with these normal patterns. We implement two types of tracking methods: person-detection-based and moving-object-based. The person-detection-based tracking method is used to track faces and people in near-field scenarios. In far-field scenarios, the moving-object-based tracking method is employed because faces are too small to be accurately detected. The moving objects are first detected by an adaptive background subtraction method, and are then tracked by using a tracking method based on appearance. An object classifier further labels each tracked object as a car, person, group of people, animal, etc. To build the model of motion patterns, the trajectories of all tracks with a given start/end location labeling are resampled and clustered together. This gives an average or "prototypical" track along with standard deviations. Most tracks from a given entry location to a given exit will lie close to the prototypical track, with typical normal variation indicated by the length of the crossbars. Tracks that wander outside this normal area can be labeled as anomalous and may warrant further investigation. The principal components of the cluster indicate typical modes of variation or "eigentracks", providing a more accurate model of normal vs. abnormal.

The algorithms for people detection and motion analysis can then be used in several higher-level surveillance applications. Starting from a detected person, we perform clothing color classification based on two body parts (torso and legs), which are segmented from the human silhouette. This enables searching for people based on the color of their clothes. We also present

applications of our system in retail loss prevention, people counting and display effectiveness. These have been successfully deployed in commercial establishments.

The rest of this chapter is organized as follows: Section 2 reviews the IBM Smart Surveillance System (SSS). Section 3 presents our learning framework for selecting and combining multiple types of visual features for object detection. The application of this framework to human detection in multiple scales is discussed in Section 4. Section 5 covers human tracking and motion analysis. Several case studies in video surveillance including people search by clothes color, retail loss prevention, people counting, and display effectiveness are demonstrated in Section 6. We conclude our work in Section 7.

2. THE IBM SMART SURVEILLANCE SYSTEM

The IBM Smart Surveillance System (SSS) employs a number of distinct and highly specialized techniques and algorithms [Hampapur *et al.* 2005], which can be summarized as follows:

- Plug and Play Analytics Frameworks: Video cameras capture a wide range of information about people, vehicles and events. The type of information captured is dependent on a number of parameters like camera type, angle, field of view, resolution, etc. Automatically detecting each type of information requires specialized sets of algorithms. For example, automatically reading license plates requires specialized OCR algorithms; capturing face images requires face detection algorithms and recognizing behaviors; finding abandoned packages requires detection and tracking algorithms. A smart surveillance system needs to support all of these algorithms, typically through a plug and play framework;

- Object Detection and Tracking: One of the core capabilities of smart surveillance systems is the ability to detect and track moving objects. Object detection algorithms are typically statistical learning algorithms that dynamically learn the scene background model and use the reference model to determine which parts of the scene correspond to moving objects [Tian *et al.* 2005]. Tracking algorithms associate the movement of objects over time generating a trajectory [Senior *et al.* 2001]. These two algorithms together take a video stream and decompose it into objects and events, effectively creating a parse tree for the surveillance video;

- Object and Color Classification: Object classification algorithms classify objects into different classes (such as people, vehicles, animals, etc.), using training data and calibration schemes. Color classification algorithms classify the dominant color of the object into one of the standard colors (red, green, blue, yellow, black and white). These attributes become part of the searchable index, allowing users to query for "red vehicles" or "people wearing blue clothes" [Brown 2008, Chen *et al.* 2008].

- Alert Definition and Detection: Typical smart surveillance systems support a variety of user-defined behavior detection capabilities such as detecting motion within a defined zone, detecting objects that cross a user-defined virtual boundary, detecting abandoned objects, etc. The user uses graphical user interface tools to specify zones of interest, object sizes and other parameters that define the behavior. When the behavior of interest occurs within a camera's field of view, the system automatically generates an alert message that can be transmitted to a workstation, PDA or email reader, depending on the users' preference [Tian *et al.* 2008, Zhai *et al.* 2008].

- Database Event Indexing: The events detected by the video analysis algorithms are indexed by content and stored in a database. This allows events to be cross-referenced across multiple spatially distributed cameras and creates a historical archive of events. The event index information typically includes time of occurrence, camera identifier, event type, object type, object appearance attributes and an index into the video repository, which allows the user to "play back the relevant video at the touch of a button".

- Search and Retrieval: Users can use a variety of GUI tools to define complex search criteria to retrieve specific events. Events are typically presented as shown in the middle section of Figure 1. Search criteria include object size, color, location in the scene, velocity, time of occurrence, and several other parameters. The results of a search can also be rendered in a variety of summary views, one of which (called track summary) is shown in the right section of Figure 1.

(a) *(b)* *(c)*

Figure 1: (a) The home page, with 1) cameras (bottom right) running a variety of video analysis capabilities, such as license plate recognition, face capture and behavior analysis; 2) Real-time alert panel (top right); 3) Map of the area (top left); 4) Video player (bottom left). (b) The results of searching for a red car. (c) A summary view of all activity in a camera over a selected period, represented as object tracks.

3. FEATURE ADAPTATION PRIOR TO LEARNING FOR OBJECT DETECTION

In this section, we describe a novel learning framework for selecting and combining multiple types of visual features for object detection [Feris *et al.* 2008]. The application of this framework to human detection in multiple scales is covered in Section 4.

3.1 Motivation

Current machine learning methods for object detection based on local image features suffer from a scalability problem in the feature selection process. For a specific *feature type* (*e.g.*, Gabor filters), most methods include many feature configurations in a *feature pool* (*e.g.*, Gabor filters uniformly sampled at different positions, orientations and scales), and then use a learning algorithm, such as Adaboost or SVM, to select the features that best discriminate object images from images that do not contain the object. Therefore, as new feature types are considered, the feature pool size increases dramatically, leading to computational problems. This scalability issue has several implications. First, the training time can be excessively long due to the large feature

pool and the brute-force strategy for feature selection. Most methods consider a pool with hundreds of thousands of local features, and training can take weeks on conventional machines. Second, the detection/recognition accuracy can be significantly affected, as important feature configurations may not be included in the feature pool due to the sampling process, whereas many features that are less meaningful for discrimination may be present in the pool.

In order to overcome the limitations discussed above, we propose a novel framework to combine and select multiple types of visual features in the learning process. Our approach relies on the observation that the local features selected for discrimination tend to match the local structure of the object. Figure 2 shows the first features selected by Adaboost in the context of face detection [Viola & Jones 2001] and recognition [Yang *et al.* 2004]. In this example, the selected Haar filters capture the local image contrast. In the middle image of the top row, the dark part of the filter coincides with the dark image region (the eyes), while the bright part of the filter matches the bright image region under the eyes (the cheek and nose). Similarly, in the bottom row, Gabor wavelets capture local structures of the face. In fact, Liu and Shum [Liu & Shum 2003], in their Kullback-Leibler boosting framework, argued that features should resemble the face semantics, matching the face structure either locally or globally.

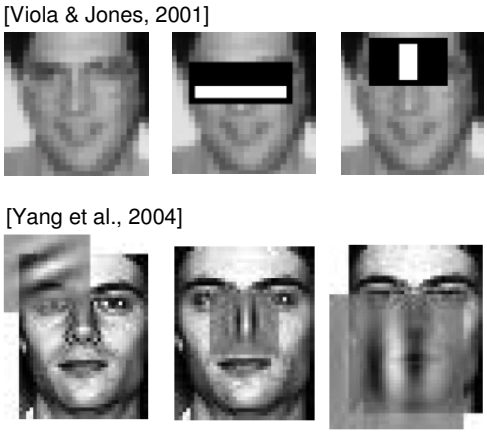


Figure 2: Features selected by Adaboost in the context of face detection (top) and face recognition (bottom). Note that the features tend to adapt to the local face structure.

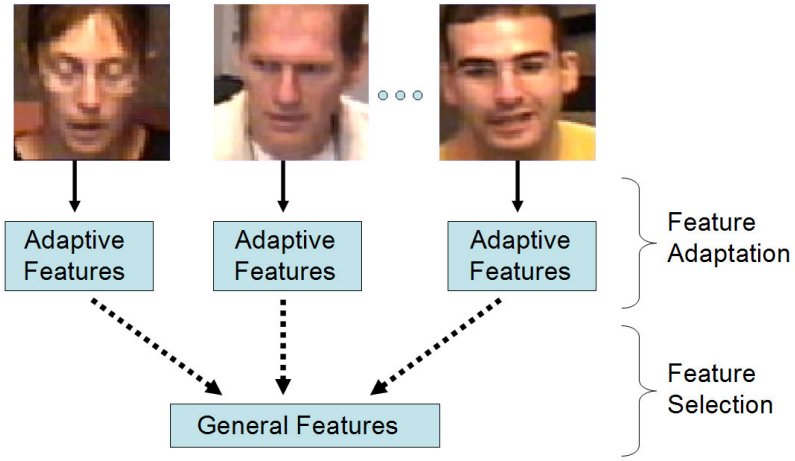


Figure 3: Our approach has two stages: first, we compute adaptive features for each training sample; then, we use a feature selection mechanism (such as Adaboost) to obtain general features.

Based on this observation, we present an approach that pre-selects local image filters based on how well they fit the local image structures of an object, and then use traditional learning techniques such as Adaboost or SVMs to select the final set of features. Our technique can be summarized in two stages. In the first stage, features that adapt to each particular sample in the training set are determined. This is carried out by a non-linear optimization method that determines local image filter parameters (such as position, orientation and scale) that match the geometric structure of each object sample. By combining adaptive features of different types from multiple training samples, a compact and diversified feature pool is generated. Thus, the computational cost of learning is reduced, since it is proportional to the feature pool size. The efficiency and the accuracy of the detector are also improved, as the use of adaptive features allows for the design of classifiers composed of fewer features, which better describe the structure of the objects to be detected. In the second stage, Adaboost feature selection is applied to the pool of adaptive features in order to select the final set of discriminative features. As the pool contains features adapted to *individual* object samples, this process selects features which encode common characteristics of *all* training samples, and thus are suitable for detection. Figure 3 illustrates this process for the particular case when the objects to be detected are faces. Throughout the remainder of this chapter, we use the term *adaptive features* to describe features that match the geometric structure of an object in a particular training image and *general features* to describe the final set of discriminative features that encode common characteristics of all training images.

3.2 Learning Using Locally Adapted Features

We now describe our framework to incorporate local feature adaptation in the training process. We begin by presenting the feature adaptation algorithm, which should be applied to each individual training image containing the target object. We then show how to use this adaptation method to create a meaningful feature pool containing multiple types of wavelet features. Lastly, the adapted feature pool is used in Adaboost learning to design a classifier to detect the object present in the training images. Although we have used wavelet filters in our work, the technique is general in the sense that other local image filters could also have been applied in the same settings.

3.2.1 Feature Adaptation

In this section, we address the problem of generating a set of local adaptive features for a given image of the object that we would like to detect. In other words, our goal is to learn the parameters of wavelet features, including position, scale, and orientation, such that the wavelet features match the local structures of the object. This is motivated by the wavelet networks proposed by Zhang [Zhang 1997] and introduced in computer vision by Krueger [Krueger 2001].

Consider a family $\Psi = \{\psi_{n_1}, \ldots, \psi_{n_N}\}$ of N two-dimensional wavelet functions, where $\psi_{n_i}(x, y)$ is a particular mother wavelet (*e.g.*, Haar or Gabor) with parameters $n_i = (c_x, c_y, \theta, s_{x,} s_y)^T$. Here, c_x and c_y denote the translation of the wavelet, s_x and s_y

denote the dilation, and θ denotes the orientation. The choice of *N* depends on the desired degree of precision for the representation.

Let *I* be an input training image, which contains an instance of the object for which a detector is to be designed. First, we initialize the set of wavelets Ψ along the image in a grid, with the wavelets having random orientations, and scales initialized to a unified value that is related to the density with which the wavelets are distributed. Figure 4(a) illustrates this process, for the specific case when the objects to be detected are faces. Then, assuming *I* is dc-free, without loss of generality, we minimize the energy function

$$E = \min_{n_i, w_i \forall i} \| I - \sum_i w_i \psi_{n_i} \|^2$$

with respect to the wavelet parameter vectors n_i and their corresponding weights w_i. Figures 4(b-e) show 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 the mother wavelet, and we used the Levenberg-Marquardt method to solve the optimization problem.

Figure 4: Learning adaptive features for a particular training image. (a) Input training image with wavelets initialized as a grid along the object region, with random orientations and scales. (b-e) Wavelet features being selected one by one, with parameters (position, scale, and orientation) optimized to match the local structure of the input image.

Differently from most existing discrete approaches, the parameters n_i are optimized in the *continuous* domain and the wavelets are positioned with sub-pixel accuracy. This assures that a maximum of the image information can be encoded with a small number of wavelets.

Using the optimal wavelets ψ_{n_i} and weights w_i, the image *I* can be closely reconstructed by a linear combination of the weighted wavelets:

$$\hat{I} = \sum_{i=1}^{N} w_i \psi_{n_i} .$$

There is an alternative procedure to directly compute the wavelet weights w_i once the wavelet parameters n_i have been optimized. This solution is faster and more accurate than using Levenberg-Marquardt optimization. If the wavelet functions are orthogonal, the weights can be calculated by computing the inner products of the image *I* with each wavelet filter, *i.e.*, $w_i = < I, \psi_{n_i} >$. In the more general cases where the wavelet functions may not be orthogonal, a

family of dual wavelets $\tilde{\Psi} = \{\tilde{\psi}_{n_1}, \ldots, \tilde{\psi}_{n_N}\}$ has to be considered. Recall that the wavelet $\tilde{\psi}_{n_j}$ is the dual wavelet of ψ_{n_i} if it satisfies the bi-orthogonality condition: $<\psi_{n_i}, \tilde{\psi}_{n_j}> = \delta_{i,j}$, where $\delta_{i,j}$ is the Kronecker delta function. Given an image *I* and a set of wavelets $\Psi = \{\psi_{n_1}, \ldots, \psi_{n_N}\}$, the optimal weights are given by $w_i = < I, \tilde{\psi}_{n_i} >$. It can be shown that $\tilde{\psi}_{n_i} = \sum_j (A_{i,j}^{-1}) \psi_{n_j}$, where $A_{i,j} = < \psi_{n_i}, \psi_{n_j} >$.

3.2.2 Integrating Multiple Features

We have described how to obtain adaptive features for a single object training image. Now we proceed to generate a pool of adaptive features obtained from multiple training images.

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 Ψ_i, using the optimization method described in the previous section.

It is possible to integrate multiple feature types by using different wavelet settings for each training image. More specifically, each set Ψ_i is learned with different parameters, including:

- *Number of wavelets.* The number of wavelets indicates how many wavelet functions will be optimized for a particular object image. From this set, we can further select a subset of functions that have the largest weights, as wavelet filters with larger associated weights in general tend to coincide with more significant local image variations;
- *Wavelet type*. In our system, we used only Haar and Gabor wavelets for the wavelet type parameter, but other feature types could also be considered;
- *Wavelet frequency*. The frequency parameter controls the number of oscillations for the wavelet filters;
- *Group of features treated as a single feature*. We also allow a group of wavelet functions to be treated as a single feature, which is important to encode global object information.

Those parameters can be randomly initialized for each training image in order to allow a variety of different features to be in the pool. This initialization process is fully automatic and allows the creation of a compact and diversified feature pool.

All generated adaptive features for all object images are then put together in a single pool of features Ω, defined as

$$\Omega = \bigcup_{i=1}^{M} \Psi_i .$$

As an example, Figure 5 shows some adaptive features (Haar and Gabor wavelets with different frequencies, orientations and aspect ratios) learned from a dataset of frontal face images. In the resulting feature pool, different types of local wavelet filters and global features, which are obtained by grouping individual wavelet functions, are present.

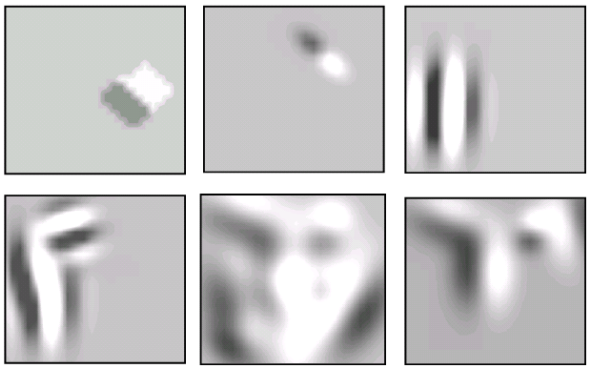


Figure 5: Some examples of features present in the pool of learned adaptive features for a frontal face dataset. A large variety of different wavelet filters are considered. The top row shows local wavelet functions, whereas the bottom row shows global features generated by combining a set of local filters.

3.2.3 Learning General Features for Detection

In Sections 3.2.1 and 3.2.2, we have described a method to generate a pool of adaptive features from a set of training images. Those features are selected according to how well they match each individual training example. Now, in order to design an object detector, the goal is to select general features, *i.e.*, features from the pool that encode common characteristics to *all* object samples.

We use Adaboost learning for both selecting general features and designing a classifier [Viola & Jones 2001]. A large set of negative examples (*i.e.*, images that do not contain the object to be detected) is used in addition to the previously mentioned training images (which contain instances of the object of interest). The general features are those that best separate the whole set of object samples from non-object (negative) samples during classification. We refer the reader to [Viola & Jones 2001] for more details about the Adaboost classifier and the feature selection mechanism. It is important to notice that other boosting techniques might be used in this step, such as GentleBoost [Friedman *et al.* 2000], Real Adaboost [Schapire & Singer 1999], and Vector Boosting [Huang *et al.* 2005]. In a more interesting way, our method could be integrated into the learning method recently proposed by Pham and Cham [Pham & Cham 2007], which achieves extremely fast learning time in comparison to previous methods. We believe that this method would have an even larger reduction in computational learning time if locally adapted features are used.

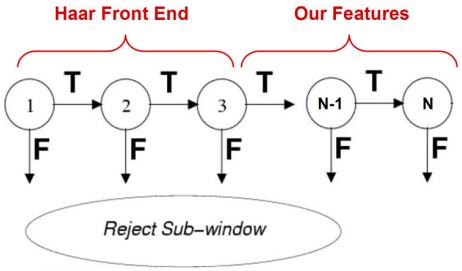


Figure 6: We use a Haar filter in the first levels of the cascade detector in order to achieve real-time performance.

3.2.4 Applying the Classifier and Efficiency Considerations

Once the cascade classifier is designed, we would like to apply it to images in order to find the object of interest. A sliding window is moved pixel by pixel at different image scales. Starting with the original scale, the image is rescaled by a given factor (typically 1.1 or 1.2) in each iteration, and the detector is applied to every window. Overlapping detection results can be merged to produce a single result for each location and scale. However, even using a cascade classifier, real-time performance (25/30Hz) can not be achieved due to the time required to compute our features. We addressed 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 rejected 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. Figure 6 depicts this architecture.

4. MULTI-SCALE HUMAN DETECTION IN SURVEILLANCE VIDEOS

In the previous section, we presented a learning framework to design object detectors based on adaptive local features. This framework can be applied to design people detectors in near-field, mid-field, and far-field surveillance scenarios, which deal with images with different levels of detail. In order to account for these differences, for each scenario we designed a person detector in a scale specifically tailored to the available resolution. We now describe in detail our implementation of the framework for near-field person detection and discuss its advantages. The same concepts could be similarly applied to improve our detectors in the other scenarios (mid-field and far-field), as these detectors are based on local image features as well.

4.1 Near-field Person Detection

In near-field surveillance videos, resolution is sufficient to make facial features of people clearly visible. We developed a face detector and a tracking system using the learning method described above to detect people in near-field scenes. To design the face detector, we used a frontal face dataset containing 4000 face images for training purposes. Each training image was cropped and rescaled to a 24x24 patch size. A pool of adaptive features was generated by running the optimization process described in Section 3.2.1, with different wavelet settings (wavelet type, frequency, etc.) for each sample. As a result, a pool of 80000 adaptive features was generated, containing a large variety of wavelet filters. It takes less than a second to create hundreds of adaptive features for a particular 24x24 sample in a conventional 3GHz desktop computer.

For the second step of the algorithm (learning general features), we used an additional database of about 1000 background (non-face) 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 [Rowley *et al.* 1998]. 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 consisted of 24 levels. A Haar filter corresponding to the first 18 levels of the cascade was used in our experiments, in order to achieve real-time performance.

Figure 7 shows the first three general features selected by Adaboost. The first selected feature gives more importance to the eyes region. The second selected feature is a local coarse-scale

Gabor wavelet with three oscillations, which align with the eyes, nose, and mouth regions. The third feature is a global feature that encodes the rounded face shape.

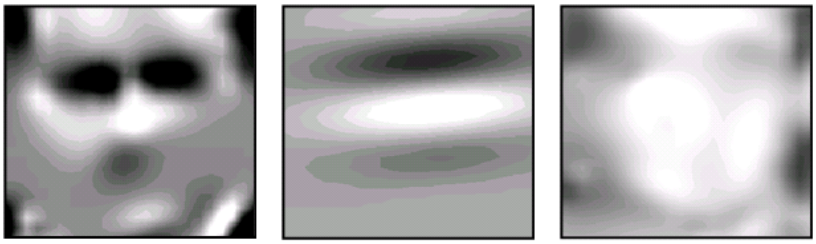


Figure 7: The first three general features selected while designing the face detector.

The CMU+MIT frontal face test set, containing 130 gray-scale 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 the width (*i.e.*, size) of the detected face box is within ±70% of the width of the ground-truth box. Figure 8 shows the detected faces in one of the images from the CMU+MIT dataset, and four video frames taken from our surveillance system.

Figure 8: Face detection results in one of the CMU+MIT dataset images, and four video frames taken from our surveillance system.

In order to show the effectiveness of feature adaptation prior to learning, we compared our face detector to a classifier learned from a similar feature pool, containing the same number and type of features (Haar, Gabor, etc.) sampled uniformly from the parameter space (at discrete positions, orientations, and scales), rather than adapted to the local structure of the training samples. Figure 9(a) shows a plot of the Receiver Operating Characteristic (ROC) curves for this comparison, demonstrating the superior performance of our method. In addition to achieving improved detection accuracy, the number of weak classifiers needed for each strong classifier is significantly smaller in our method (see Figure 9(b)). This has a direct impact in both training and testing computational costs. We observed a reduction of about 50%.

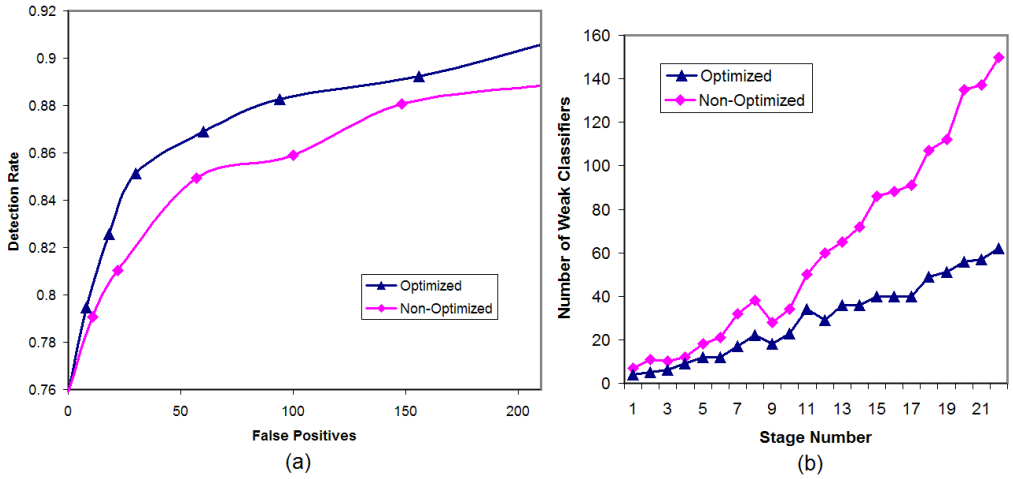


Figure 9: (a) ROC Curve comparing classifiers learned from adaptive (optimized) and non-adaptive (non-optimized) features in the CMU+MIT dataset. (b) Number of classifiers for each level of the cascade in both methods. Our approach offers advantages in terms of detection accuracy and reduced computational costs over traditional methods that use local features uniformly sampled from the parameter space.

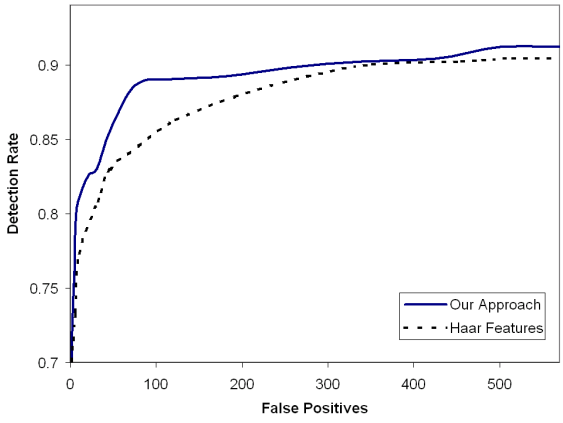


Figure 10: ROC Curves for our approach and traditional Haar features in the CMU+MIT dataset. We used only half of the number of features in the feature pool compared to Haar features and still get superior performance.

Figure 10 shows the comparison between our approach and traditional Haar-like/rectangle features. The cascade detector based on Haar features also consisted of 24 levels, and was learned using the same training set. The feature pool, however, was twice as large as the one used by our approach, containing about 160000 features. With half of the features in the pool, we achieve superior accuracy and a faster learning time. Table 1 confirms this. Although our optimized features can cause overfitting when a small number of training samples are considered, this issue does not arise when thousands of faces are used for learning.

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

Feature Pool	Number of Features	Learning Time
Haar Features	160000	About 5 days
Our Approach	80000	About 3 days

4.2 Mid-field Person Detection

In mid-field scenes, facial features may not be visible due to poor resolution. However, the lines that delimit the head and shoulders of an individual are still informative cues to find people in images. For these scenes, we developed a system for tracking and detection which locates people by scanning a window through the image and applying a head and shoulders detector at every position and scale. This detector is designed according to the same learning framework from [Viola & Jones 2001] (as we implemented this classifier prior to our research on feature optimization), *i.e.*, it is a cascade classifier based on Adaboost learning and Haar features. Similarly to the face detector, a training set of 4000 images containing the head and shoulders region was used for training. As this classifier is based on feature selection from a pool of local features, it is part of our current work to apply the learning framework from Section 3 to first create a pool of adaptive local features and then select the most discriminative features using Adaboost. Figure 11 illustrates the detection results from the head and shoulders detector in our system for mid-field scenes.

Figure 11: Sample detections in mid-field scenes, using a head and shoulders detector based on Haar features.

4.3 Far-field Person Detection

In far-field imagery, pedestrians may appear as small as 30-pixels tall. In our scenario, the camera is known to be in a fixed position, making it feasible to use background modeling techniques to segment moving objects. In [Chen *et al.* 2008], we described how our far-field surveillance system classifies blobs obtained from background subtraction into one of three classes: cars, people and groups of people. In [Viola *et al.* 2003], a system for pedestrian detection in videos which extends the technique from [Viola & Jones 2001] was proposed. It augments the feature space by including Haar features computed from differences between pairs of subsequent frames. Our locally adaptive feature learning framework could also be applied in this case, in order to pre-select the features that best adapt to pedestrians in individual frames or to differences in subsequent frames due to movement. Then, the resulting detector could be combined with our classifier based on foreground measurements in order to confirm that the blobs classified as people in fact do correspond to people.

5. HUMAN TRACKING AND MOTION ANALYSIS

Human tracking and motion analysis are key components of video surveillance systems, and very active research areas. Recently, researchers focused on combining detection and tracking into unified frameworks. Andriluka *et al.* [Andriluka *et al.* 2008] proposed a unified framework that combines both pedestrian detection and tracking techniques. Detection results of human articulations using a hierarchical Gaussian process latent variable model are further applied in a Hidden Markov Model to perform pedestrian tracking. Okuma *et al.* [Okuma *et al.* 2004] proposed a target detection and tracking framework by combining two very popular techniques: mixture particle filters for multi-target tracking and Adaboost for object detection. Another Adaboost-based method is presented in [Avidan 2005] by Avidan. In his work, weak classifiers are combined to distinguish foreground objects from the background. The mean-shift algorithm is applied to track the objects using the confidence map generated by the Adaboost method. Leibe *et al.* [Leibe *et al.* 2005, 2007] used a top-down segmentation approach to localize pedestrians in the image using both local and global cues. An implicit human shape model is built to detect pedestrian candidates that are further refined by the segmentation process, using Chamfer matching on the silhouettes. Ramanan *et al.* [Ramanan *et al.* 2007] proposed a "tracking by model-building and detection" framework for tracking people in videos. Predefined human models are combined with candidates detected in the video to form actual human clusters (models). These models are then used to detect the person in subsequent images and perform tracking. Other co-training based approaches are [Javed *et al.* 2005, Grabner *et al.* 2006]. They proposed detecting objects and further using them for online updating of the object classifiers. Object appearance features derived from PCA are iteratively updated using the samples with high detection confidence.

For motion analysis, Stauffer *et al.* [Stauffer *et al.* 2000] proposed a motion tracking framework. Each pixel in the image is modeled by a mixture of Gaussian distributions that represents its color statistics. Object tracks are formed by correlating motion segments across frames using a co-occurrence matrix. Buzan *et al.* [Buzan *et al.* 2004] proposed a clustering technique to group three-dimensional trajectories of tracked objects in videos. A novel measure called the Longest Common Subsequence is employed to match trajectory projections on the coordinate axes. Junejo and Foroosh [Junejo & Foroosh 2008] proposed a trajectory grouping method based on normalized cuts. The matching criteria include spatial proximity, motion characteristics, curvature, and absolute world velocity, which are based on automated camera calibration.

The IBM SSS provides functions for tracking people and detecting trajectory anomalies. The system tracks faces and people, analyzes the paths of tracked people, learns a set of repeated patterns that occur frequently, and detects when a person moves in a way inconsistent with these normal patterns.

5.1 Face Tracking

Once a face is detected in a particular video frame, it is necessary to track it in order to analyze the trajectory of the person and identify a single key frame of the face, to be stored in a database. Our face tracking method is based on applying the face detector to every frame of the video sequence. In order to maintain the track of 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, the track is updated with the window given by the correlation tracker. Otherwise, if the face detector reports a window result with a close position and size to the current tracking window, then this result is used to update the track. This mechanism is important to avoid drifting.

In order to improve the efficiency of our detector and enable real-time face tracking (25/30Hz) on conventional desktop computers, we use 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) combines view-based detectors and tracking.

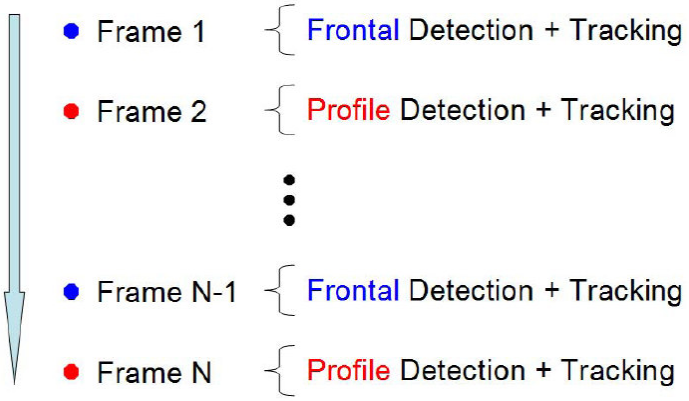


Figure 12: Our surveillance system interleaves view-based detectors to save frame rate for tracking.

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 [Tian *et al.* 2005], to prune sub-windows that do not lie in motion regions. A skin color detector could also be used to speed up the processing.

A problem faced by most existing systems is the computational time required to run a set of view-based detectors 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 12. In each frame, rather than running both frontal and profile detectors, we run just 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 allows us to improve the frame rate by 50%, while facilitating tracking due to less inter-frame variation.

Tracking is terminated when there are 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.

5.2 Human Tracking and Motion Analysis

5.2.1 Human Tracking

Tracking can be seen as a problem of assigning consistent identities to visible objects. We obtain a number of observations of objects (detections by the background subtraction algorithm) over time, and we need to label these so that all observations of a given person are given the same label. When one object passes in front of another, partial or total occlusion takes place, and the background subtraction algorithm detects a single moving region. By handling occlusions, we hope to be able to segment this region, appropriately labeling each part and still maintaining the correct labels when the objects separate. In more complex scenes, occlusions between many objects must be handled [Senior *et al.* 2001].

When objects are widely separated, a simple bounding box tracker is sufficient to associate a track identity with each foreground region. Bounding box tracking works by measuring the distance between each foreground region in the current frame and each object that was tracked in the previous frame. If the object overlaps with the region or lies very close to it, then a match is declared.

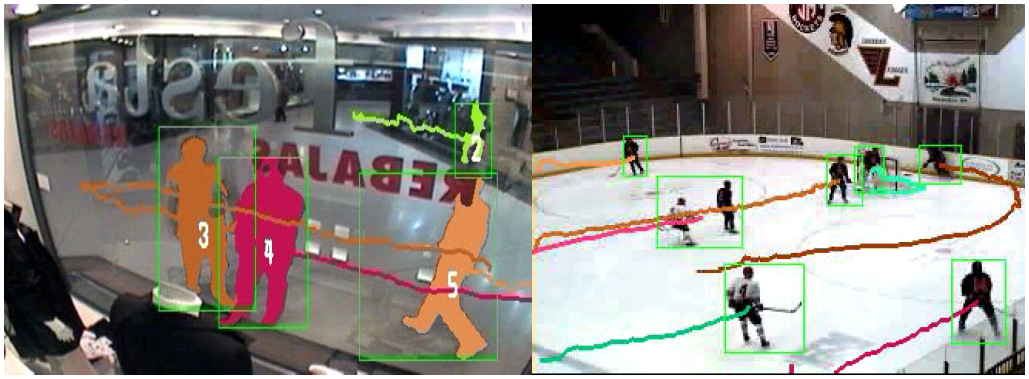
If the foreground regions and tracks form a one-to-one mapping, then tracking is complete and the tracks are extended to include the regions in the new frame using this association. If a foreground region is not matched by any track, then a new track is created. If a track does not match any foreground regions, then it continues at a constant velocity, but it is considered to have left the scene once it fails to match any regions for a few frames.

Occasionally, a single track may be associated with two regions. For a few frames, this is assumed to be a failure of background subtraction and both regions are associated with the track. If there are consistently two or more foreground regions, then the track is split into two, to model cases as when a group of people separate, a person leaves a vehicle, or an object is deposited by a person. Figure 13 shows some people tracking results.

5.2.2 Motion Analysis

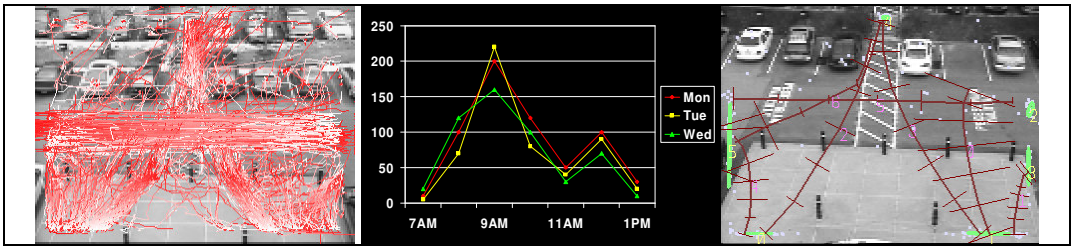
In order to perform motion analysis, as shown in Figure 14, the system begins by detecting the locations where objects enter and exit the scene. The start and end points of tracks are clustered to

find regions where tracks often begin or end. These points tend to be where paths or roads reach the edge of the camera's field of view. Having clustered these locations, we classify the trajectories by labeling a track with its start and end location (or as an anomaly when it starts or ends in an unusual location, such as a person walking through the bushes). For example, when we cluster trajectories for the camera placed at the entrance to our building, trajectories are classified into one of 5 classes – entering/exiting to the left side (from the road on the left or from the center), entering/exiting to the right side (from the road on the right or from the center), or moving horizontally across the road. We then apply a secondary clustering scheme to further detect anomalous behavior. This scheme operates as follows: the trajectories of all tracks with a given start/end location labeling are resampled and clustered together. This gives an average or "prototypical" track together with standard deviations. Most tracks going from a given entry location to a given exit will lie close to the prototypical track, with typical normal variation indicated by the length of the crossbars. Tracks that wander outside this normal area can be labeled as anomalous and may warrant further investigation. The principal components of the cluster indicate typical modes of variation or "eigentracks" , providing a more accurate model of normal vs. abnormal. Figure 15 shows some examples of abnormal activities (people loitering) in front of the IBM Hawthorne building.



(a) (b)

Figure 13: Examples of people tracking results. (a) People tracking in a shopping mall; (b) tracking of hockey players.



(a) (b) (c)

Figure 14: (a) Summary view showing the retrieval of the trajectories of all events that occurred in the parking lot over a 24 hour period. The trajectory colors are coded: starting points are shown in white and ending points are displayed in red. (b) Activity distribution over an extended time period, where the time is shown on the x-axis and the number of people in the area is shown on the y-axis. Each line represents a different day of the week. (c) Unsupervised behavior

analysis. Object entrance/departure zones (green ellipses) and prototypical tracks (brown curves) with typical variation (crossbars) are displayed.

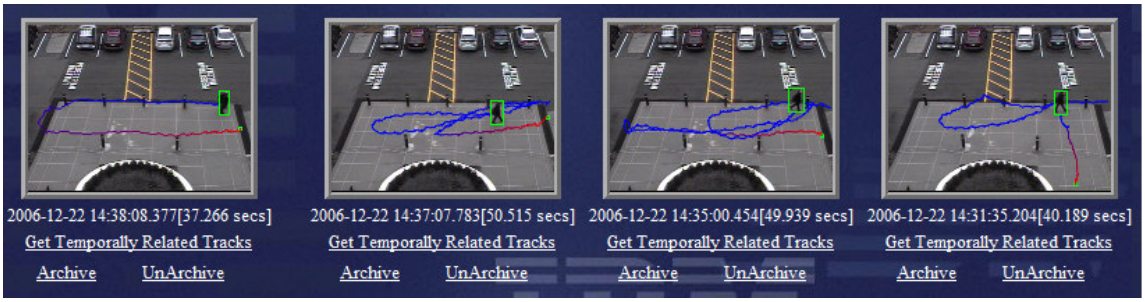


Figure 15: Abnormal activity analysis (people loitering in front of the IBM Hawthorne building).

6. CASE STUDY FOR REAL VIDEO SURVEILLANCE APPLICATIONS

There are many applications for people detection, tracking, and motion analysis in video surveillance. In this section, we present several case studies including people search, retail loss prevention, people counting, and display effectiveness.

6.1 People Search by Clothing Color

An important aspect of human identification for finding and/or matching a person to another person or description, in the short term (*e.g.*, the same day), is based on clothing. Color is one of the most prominent cues for describing clothing. Here, we present our methodology for categorizing the color of people's clothes for the purposes of people search.

We perform clothing color classification based on two body parts segmented from the human silhouette: the torso and the legs. These are determined using the normative spatial relationship to the face location, as detected in Section 4.1. The torso or upper body region represents what is primarily the shirt or jacket, while the legs or lower body region represents the pants or skirt of the tracked human extracted from the camera.

Three issues are critical for successful color classification. The first is the issue of color constancy. People perceive an object to be of the same color across a wide range of illumination conditions. However, the actual pixels of an object, which are perceived by a human to be of the same color, may have values (when sensed by a camera) which range across the color spectrum depending on the lighting conditions. Secondly, moving objects extracted from video are not perfectly segmented from the background. Shadows are often part of the object and errors exist in the segmentation due to the similarity of the object and the background model. Lastly, complex objects (torsos and legs, in this case) may have more than one color.

The method described here is based on acquiring a normalized cumulative color histogram for each tracked object in the bi-conic HSL (hue, saturation, luminance) space. The method is designed with mechanisms to divide this space via parameters that can be set by a user or by active color measurements of the scene. The key idea is to intelligently quantize the color space based on the relationships between hue, saturation and luminance. As color information is limited by both lack of saturation and intensity, it is necessary to separate the chromatic from the

achromatic space along surfaces defined by a function of saturation and intensity in the bi-conic space.

In particular, for each tracked body part (torso or legs), the color classifier will create a histogram of a small number of colors. We usually use six colors: black, white, red, blue, green and yellow. However, for clothing from people detected by the cameras used in this study (our facility), we use the following colors based on the empirical distribution and our ability to discern: red, green, blue, yellow, orange, purple, black, grey, brown, and beige. The histogram is created as follows. Each frame of the tracked object that is used for histogram accumulation (every frame the body parts detector finds) is first converted from RGB to HSL color space. Next, the HSL space is quantized into a small number of colors.

In order to quantize this space into a small number of colors, we determine the angular cutoffs between colors. When we use six colors (black, white, yellow, green, blue, red), we need only four cutoffs between the hues: yellow/green, green/blue, blue/red and red/yellow. However, variations due to lighting conditions, object textures and object-to-camera viewpoint lead to differences in brightness and color saturation. Therefore, we also need to specify lightness and saturation cutoffs. Here, it is interesting to note that saturation and intensity are related. Both properties can make the hue of a pixel indiscernible. For intensity, this occurs when the light is too bright or too dark. For saturation, it happens when there is insufficient saturation. However, as the brightness gets too low or too high, the necessary saturation increases. In general, as intensity increases from 0 up to halfway (the central horizontal cross-section of the bi-conic) or decreases from the maximum (white) down to halfway, the range of pixels with visible or discernable hue increases.

In summary, we first quantize the HSL space based on hue. We subsequently re-label pixels as either white or black depending on whether they lie outside the lightness/saturation curve above or below the horizontal mid-plane. This is related to earlier work in color segmentation performed by Tseng and Chang [Tseng & Chang 1994].

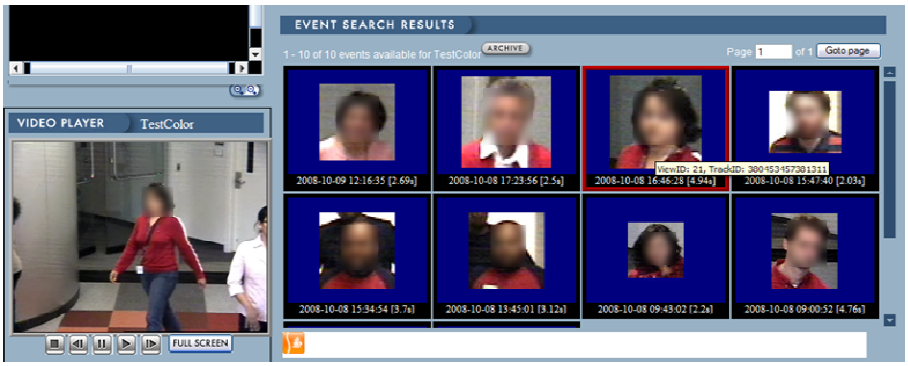


Figure 16: Results of search for people with red torso (shirts).

Using this color classification scheme, the system records the colors of torso and legs for each person for whom the detection of the face and the body parts is successful. This information is sent to the database for retrieval. Users may search for people with specific colors of torso and legs. Figure 16 shows the results of a search for people with red torsos (shirts). Figure 17 shows the results of a search for people with white torso and red legs. The final figure (Figure 18) in this section shows the results of a search for people with yellow torso and black legs. In each picture, the key frames for each event (person) are shown on the right. The user may select any key frame

to watch the associated video clip of the person as they walk through the scene. A single frame of this video is shown at the left of each figure.

Figure 17: Results of search for people with white torso (shirt) and red legs (pants/skirt).

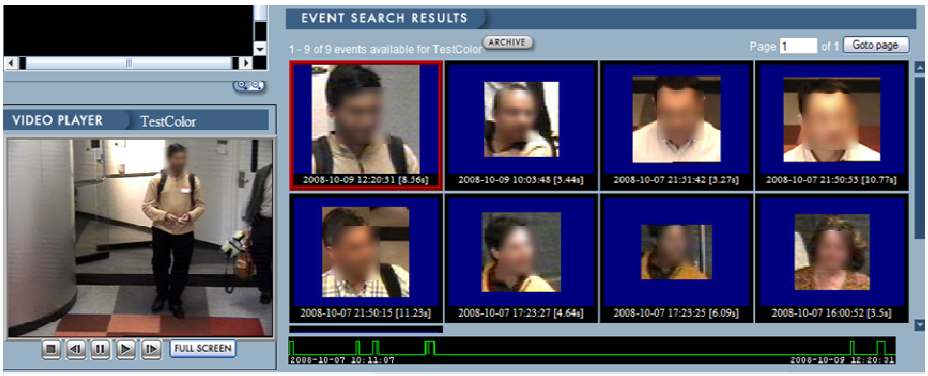


Figure 18: Results of search for people with yellow torso (shirts) and black legs (pants/skirt).

6.2 Retail Loss Prevention

There are many practical applications for the people search capabilities described in this chapter. Many of these applications involve matching people across cameras, with identification information, or with queries. Matching people across cameras is useful to "track" a person across cameras, thereby linking a person in one camera to their movements and actions in another. In this way, a person can be tracked across non-overlapping cameras. Many important security tasks involve this capability. For example, a person who breaks through a security checkpoint in an airport could be tracked to a different part of the building.

A person can also be "matched" against identification information. This information may be associated with a badge or other type of ID, and may be used to corroborate the legitimacy of their access to a building or a cash register. Lastly, a person can be matched against a query to locate them across several cameras and time periods. We have built an application to match

people to improve the detection of returns fraud. In this application, a user is looking to match a person at the returns counter in a department store with people entering the store.

Returns fraud can take one of a number of forms. Our retail customer was particularly interested in the case of returning items that were just picked up in the store, but were never bought. The return could be done either without a receipt (in stores that have liberal return policies), or using a receipt from a previously purchased (and kept) item.

Our approach, described in more detail in a previous paper [Senior, 2007], allows loss prevention staff to quickly determine whether a person returning an item entered the store carrying that item. The system works by detecting and tracking customers at entrances and customer service desks and associating the two events. This solution only requires cameras at the store entrances and the return counters, being simpler than approaches that rely on tracking customers throughout the store (requiring many cameras and very reliable camera hand-off algorithms) and must be continuously monitored to determine whether items are picked up.

Two cameras at the customer service desk record activity there, including the appearance of customers returning items. A separate set of cameras points at the doors and captures all activity of people entering and leaving the store. Figure 19 shows the fields of view of two such cameras. Our approach automatically segments events in each of these cameras, filters them and then provides a user interface which allows the association of each returns event with the door entrance event that corresponds to when the person came into the store. At the customer service desk, the face tracking algorithm tracks customers' faces, generating a single event per customer. Customers at the doors are tracked with our appearance-based tracker [Senior 2001].

The returns fraud interface provides intuitive selection and browsing of the events, summarized by presentation of key frames (at both scales), timestamps and original video clips (from DVR or media server). Search typically begins by selecting a return event from the TLOG (see Figure 20). In response to this, the interface displays the people found at the customer service counter near that time. Selecting one of these then displays people entering the store shortly before the selected event. The user can then browse through the entrance events, using full-frame and zoomed-in key frames as well as original video, and, when a match is found, determine whether a fraud has taken place.

The fundamental indexing attribute of the database is time. All devices are synchronized and events are time-stamped. Temporal constraints from real world conditions are exploited to limit the events displayed. In our implementation, a user may also use color information (Section 6.1) to assist in finding likely matches between the people entering and the person at the returns counter.

Figure 19: Views from the customer service desk (left) and a door (right). The "region of uninterest" in which detections are ignored to reduce false positives is outlined in blue. Alert tripwires (enter and leave) are drawn on the door view.

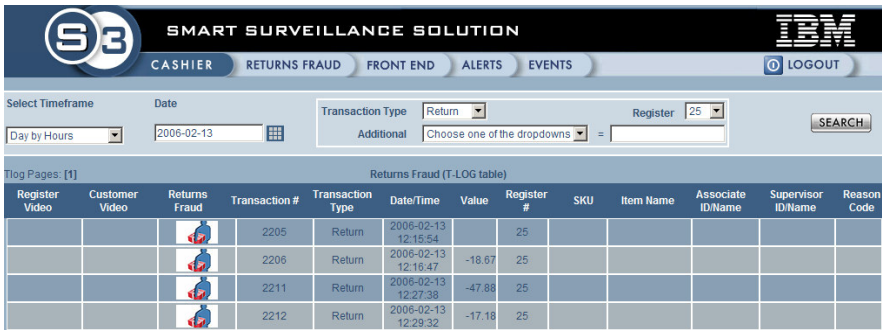


Figure 20: Interface for browsing and searching the TLOG (transaction log), showing a table of return transactions.

6.3 People Counting

As a direct application of person detection and search, people counting has significant importance in various scenarios. In the retail sector, accurate people counts provide a reliable base for high-level store operation analysis and improvements, such as traffic load monitoring, staffing assignment, conversion rate estimation, etc.

In our solution, we incorporate a people counting framework which effectively detects and tracks moving people in bi-directional traffic flows, denoted as "entries" and "exits." Counting results are indexed into the logical relational database for generating future statistical reports. An example of a report is shown in Figure 21. We demonstrate the people counting application in the cafeteria traffic scenario at our institution. In addition to the statistical reports, individual counting events are also available for investigation by exhibiting corresponding key frames. Disjoint search intervals enable the system's capability of performing cross-time traffic comparison. For instance, in Figure 22, the average traffic counts between three time intervals on five days of a week are compared. For morning times (8:00AM-11:00AM), Monday has the fewest count. This is due to the fact that employees tend to have a busier morning after the weekend, and thus do not visit the cafeteria as often as on other weekdays. For lunch time (11:00AM-2:00PM) and afternoon tea time (2:00PM-5:00PM), Friday has the least traffic volume. This is due to the fact that some employees leave work earlier, and thus do not have lunch or afternoon tea/coffee at the company.

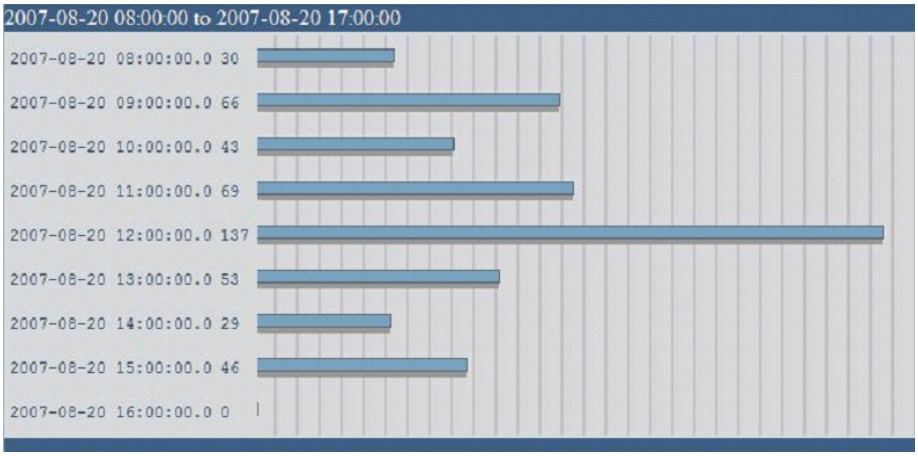


Figure 21: People counting statistical report.

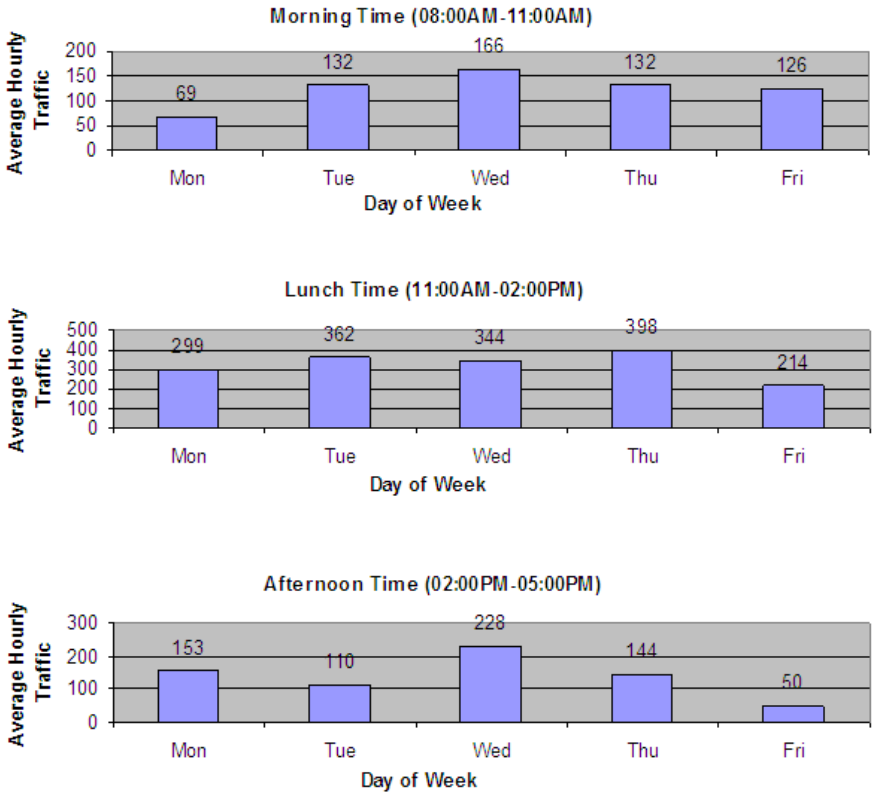


Figure 22: Weekly comparison of the average people counts for three time intervals at the IBM Hawthorne cafeteria.

People counting can also be associated with non-visual sensor data. One application is badge use enforcement, where security wants to ensure that employees always carry identification badges. In certain working environments, employees need to swipe their badges to access critical areas. In order to detect people entering the critical areas without using their badges (*e.g.*, by tailgating other people who properly swiped their badges), the number of entering people is estimated by applying a person detector (Section 4.1). The people count is then matched with the input signals obtained from the badge reader. If the number obtained from the people counting algorithm is greater than the one indicated by the badge read signals, then an alert is generated.

6.4 Display Effectiveness

In retail stores, a critical problem is to effectively display merchandise at certain places to achieve maximum customer attention and optimal revenue. For instance, the store management is very interested in the average number of people who have stopped in front of a particular item each hour. If the number of people looking at this item is consistently far less than the number of people who have stopped by its neighboring items, then the store management could conclude that this particular item does not attract enough customers. Thus, it brings less value to the store revenue, and may be removed from the display or relocated to a less valued region. This analysis can be accomplished by constructing a "heat map" of the store plan, which represents the traffic density in the store. One such "heat map" is demonstrated in Figure 23, where warmer values represent higher number of people passing through and stopping by the corresponding location, and colder values represent lower traffic. Based on this analysis, the duration of the stay can also

be derived. The management can obtain a better understanding on whether the customers are actually looking at a certain item, or are just passing by.

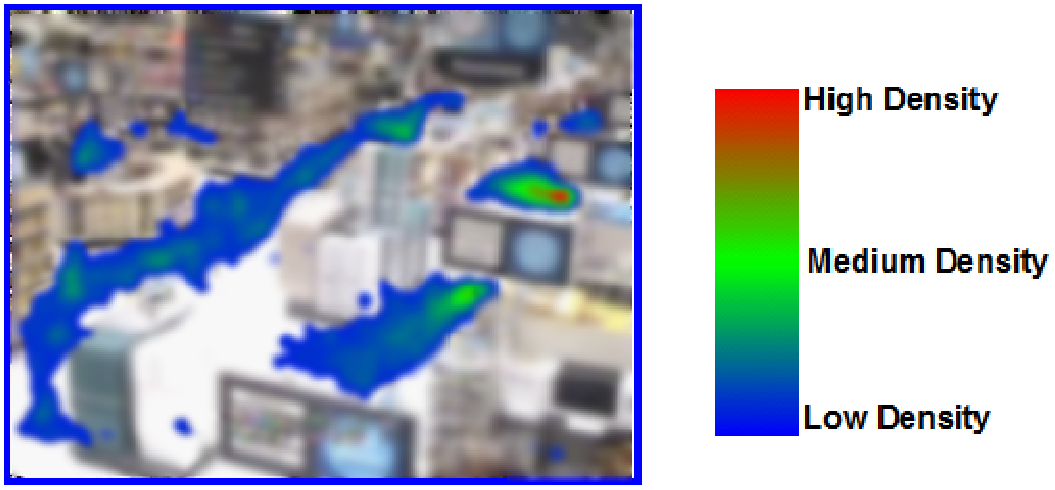


Figure 23: A heat map of the store's traffic density, where warmer values represent a larger number of people passing through and stopping by the corresponding location, and colder values represent lower traffic.

7. DISCUSSION AND CONCLUSION

Effective, efficient, and robust people detection and tracking are core components of video surveillance. In this chapter, we have presented reliable techniques for people detection, tracking, and motion analysis in the IBM Smart Surveillance System (SSS), a commercial video surveillance system. The people 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. We have described techniques for people detection at different scales. In order to meet the different requirements of video surveillance, we have also implemented two types (detector-based and appearance-based) of human tracking methods. Finally, the applicability and effectiveness of the proposed people detection and motion analysis techniques have been demonstrated in a variety of real surveillance applications, including people search based on clothing color, retail loss prevention, people counting, and display effectiveness.

While technologies like networked video surveillance, smart surveillance, IP cameras, and high density storage continue to improve the tools for surveillance, there are a number of processes, privacy and policy issues that are required for the success of operational security systems. Currently, most of our security agencies are largely geared toward responding to events and using video surveillance in a "reactive monitoring process." Technologies such as smart surveillance begin to enable a proactive monitoring process. We expect to develop more robust people detection algorithms that require a smaller number of examples and work across different environments without need for re-training. This is extremely important for scenarios where it is difficult to collect training examples. Tracking and recognizing people in more complex environments across multiple cameras will be continually investigated.

The adoption of a smart surveillance system introduces a new stream of video based alarms into the command center. To ensure successful deployment, customers and technology providers

should jointly address key issues such as the training of operators to use sophisticated technologies, evaluate alarm conditions, and determine appropriate responses to these events. The system must be designed, configured and tuned to minimize the impact of false alarms.

As the technology to monitor areas for purposes of law enforcement and homeland security evolves, such technologies typically raise the issues of citizen privacy in public spaces. These challenges can be addressed both at the technology and policy levels. Citizen privacy can be protected by enabling privacy preserving technologies in the surveillance systems [Senior *et al.* 2005]. The technical enablers of privacy then have to be put into practice by formulating, implementing and enforcing appropriate policies that govern the use of such systems.

REFERENCES

Andriluka, M., Roth, S., and Schiele, B. (2008) People-Tracking-by-Detection and People-Detection-by-Tracking, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'08).*

Avidan, S. (2005) "Ensemble Tracking", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05*

Beymer, D., and Konolige, K. (1999) Real-Time Tracking of Multiple People Using Continuous Detection, *IEEE International Conference on Computer Vision (ICCV'99).*

Brown, L. (2008). Color Retrieval in Video Surveillance, Fifth IEEE Int'l Conf. on Advanced Video and Signal Based Surveillance (AVSS), Santa Fe, NM.

Buzan, D., Sclaroff, S., and Kollios, G. (2004) "Extraction and Clustering of Motion Trajectories in Videos", *International Conference on Pattern Recognition (ICPR'04).*

Chen, L., Feris, R. S., Zhai, Y., Brown, L., and Hampapur, A. (2008). An integrated system for moving object classification in surveillance videos. *IEEE International Conference on Advanced Video and Signal-Based Surveillance*, Santa Fe, New Mexico.

Dalal, N. and Triggs. B. (2005) Histograms of Oriented Gradients for Human Detection. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05). Vol I (pp. 886-893).*

Feris, R., Tian, Y., Zhai, Y., and Hampapur, A. (2008). Facial image analysis using local feature adaptation prior to learning. *IEEE International Conference on Automatic Face and Gesture Recognition*. Amsterdam, Netherlands.

Friedman, J., Hastie, T., and Tibshirani , R. (2000). Additive logistic regression: a statistical view of boosting. *Annals of Statistics* 38(2) (pp. 337–374).

Gavrila, D. (2000) Pedestrian Detection from a Moving Vehicle. *Europe Conference on Computer Vision (ECCV'00).*

Green, M. (1999) The appropriate and effective use of security in schools, *US Department of Justice, Report NJC178265.*

Grabner H. and Bischof, H. (2006) "Online Boosting and Vision", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06)*

Hampapur, A., Brown, L., Connell, J., Ekin, A., Haas, N., Lu, M., Merkl, H., Pankanti, S., Senior, A., Shu, C., and Tian, Y. (2005) Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking, *IEEE Signal Processing Magazine*, Volume: 22, Issue: 2, (pp 38- 51).

Han, F., Shan, Y., Cekander, R., Sawhney, H., and Kumar, R. (2006) A Two-Stage Approach to People and Vehicle Detection With HOG-Based SVM, *Performance Metrics for Intelligent Systems Workshop*, National Institute of Standards and Technology.

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

Javed, O., Ali, S., and Shah, M. (2005) Online Detection and Classification of Moving objects Using Progressively Improving Detectors, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05).*

Junejo, I. and Foroosh, H. (2008) "Euclidean path modeling for video surveillance", IVC Vo. 26, No. 4, pp. 512-528.

Krueger, V. (2001). Gabor wavelet networks for object representation. PhD thesis, Christian-Albrecht University, Kiel, Germany.

Leibe, B., Seemann, E., and Schiele, B. (2005) Pedestrian Detection in Crowded Scenes. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05).*

Leibe, B., Schindler, K., and Van Gool, L. (2007) "Coupled Detection and Trajectory Estimation for Multi-Object Tracking", *International Conference on Computer Vision (ICCV'07).*

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

Miezianko, R., and Pokrajac, D. (2008) People detection in low resolution infrared videos. *IEEE International Workshop on Object Tracking and Classification in and Beyond the Visible Spectrum (OTCBVS'08).*

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

Okuma, K., Taleghani, A., Freitas, N. (2004) Little J., and Lowe, D. "A boosted particle filter: multi-target detection and tracking", *European Conference on Computer Vision (ECCV'04).*

Patil, R., Rybski, P., Veloso, M., and Kanade, T. (2004) People Detection and Tracking in High Resolution Panoramic Video Mosaic, *Proceedings of the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 1323-1328.

Pham, M., and Cham, T. (2007). Fast training and selection of haar features using statistics in boosting-based face detection. *IEEE International Conference on Computer Vision (ICCV'07)*, Rio de Janeiro, Brazil.

Ramanan, D., Forsyth, D. A., and Zisserman, A. (2007) Tracking people by learning their appearance, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29:65-81.

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

Sabzmeydani, P., and Mori, G. (2007) Detecting Pedestrians by Learning Shapelet Features. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'07).*

Senior, A., Brown, L., Shu, C., Tian, Y., Lu, M., Zhai, Y., and Hampapur, A. (2007). Visual person searches for retail loss detection. *International Conference on Vision Systems*.

Senior, A., Pankanti, S., Hampapur, A., Brown, L., Tian, Y., Ekin, A., Connell, J., Shu, C., and Lu, M. (2005) Enabling video privacy through computer vision, *IEEE Security & Privacy,Volume 3, Issue 3, pp. 50 – 57.*

Senior, A., Hampapur, A., Tian, Y., Brown, L., Pankanti, S. , and Bolle, R. (2001). Appearance models for occlusion handling. *International Workshop on Performance Evaluation of Tracking and Surveillance.*

Schapire, R., and Singer, Y. (1999). Improved boosting algorithms using confidence-rated predictions. *Machine Learning* 37 (pp. 297–336).

Stauffer, C., Eric, W. and Grimson, L. (2000) "Learning patterns of activity using real-time tracking", PAMI 2000.

Tian, Y., Feris, R., and Hampapur, A. (2008) Real-time Detection of Abandoned and Removed Objects in Complex Environments. *The 8th Int'l Workshop on Visual Surveillance (VS).*

Tian, Y., Lu, M., and Hampapur, A. (2005) Robust and Efficient Foreground Analysis for Real-time Video Surveillance, *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05)*. San Diego.

Tseng, D., and Chang, C. (1994). Color segmentation using UCS perceptual attributes, *Proc. Natl. Sci. Counc. ROC(A)*, Vol 18, No 3 (pp. 305-314).

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

Viola, P., Jones, M., and Snow, D. (2003). Detecting Pedestrians Using Patterns of Motion and Appearance. *IEEE International Conference on Computer Vision (ICCV'03)*, Vol. 2 (pp. 734-741).

Wang, P., and Ji. Q. (2005) Learning discriminant features for multiview face and eye detection. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'05).*

Wu, B., and Nevatia, R. (2005) Detection of Multiple, Partially Occluded Humans in a Single Image by Bayesian Combination of Edgelet Part Detectors. *IEEE International Conference on Computer Vision (ICCV'05) Vol I: (pp. 90-97).*

Wu, B. and Nevatia. R. (2006) "Tracking of Multiple, Partially Occluded Humans based Static Body Part Detection", *IEEE Conference on Computer Vision and Pattern Recognition (CVPR'06).*

Yang, P., Shan, S., Gao, W., Li, S., and Zhang, D. (2004). Face recognition using ada-boosted gabor features. *International Conference on Automatic Face and Gesture Recognition*, Seoul, Korea.

Zhai, Y., Tian, Y., and Hampapur, A. (2008) Composite Spatio-Temporal Event Detection in Multi-Camera Surveillance Networks. *Workshop on Multi-camera and Multi-modal Sensor Fusion Algorithms and Applications (M^2SFA^2).*

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

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