Machine Vision and Applications (2003) 15: 29–45
Digital Object Identifier (DOI) 10.1007/s00138-003-0121-6

DETER: Detection of events for threat evaluation and recognition

V. Morellas[1], **I. Pavlidis**[2], **P. Tsiamyrtzis**[3]

[1] Honeywell Laboratories, 3660 Technology Dr., Minneapolis, MN 55418, USA
[2] Dept. of Computer Science, University of Houston, 501 Philip G. Hoffman Hall, Houston, TX 77204, USA
[3] School of Statistics, University of Minnesota, 313 Ford Hall, Minneapolis, MN 55455, USA

Received: 6 July 2001 / Accepted: 12 November 2002
Published online: 23 July 2003 – © Springer-Verlag 2003

Abstract. The current security infrastructure can be summarized as follows: (1) Security systems act locally and do not cooperate in an effective manner, (2) Very valuable assets are protected inadequately by antiquated technology systems and (3) Security systems rely on intensive human concentration to detect and assess threats.

In this paper we present DETER (Detection of Events for Threat Evaluation and Recognition), a research and development (R&D) project aimed to develop a high-end automated security system. DETER can be seen as an attempt to bridge the gap between current systems reporting isolated events and an automated cooperating network capable of inferring and reporting threats, a function currently being performed by humans.

The prototype DETER system is installed at the parking lot of Honeywell Laboratories (HL) in Minneapolis. The computer vision module of DETER reliably tracks pedestrians and vehicles and reports their annotated trajectories to the threat assessment module for evaluation. DETER features a systematic optical and system design that sets it apart from "toy" surveillance systems. It employs a powerful Normal mixture model at the pixel level supported by an expectation-maximization (EM) initialization, the Jeffreys divergence measure, and the method of moments. It also features a practical and accurate multicamera calibration method. The threat assessment module utilizes the computer vision information and can provide alerts for behaviors as complicated as the "hopping" of potential vehicle thieves from vehicle spot to vehicle spot.

Extensive experimental results measured during actual field operations support DETER's exceptional characteristics. DETER has recently been successfully productized. The product-grade version of DETER monitors movements across the length of a new oil pipeline.

Key words: Object tracking – Multicamera calibration – Threat assessment – Surveillance system – Security system

Correspondence to: V. Morellas
(e-mail: vassilios.morellas@honeywell.com)

1 Introduction

The Video Surveillance and Monitoring (VSAM) program funded by DARPA in 1997–1999 [1–3] pushed the state of the art to a point where future commercial application of the technology is no longer unthinkable. No large-scale R&D effort has been undertaken since then in the area of surveillance. Although substantial achievements have been demonstrated during the VSAM program, many R&D challenges remain. Our work addresses a number of these challenges and makes headway toward the commercialization of an intelligent monitoring system.

We are interested in monitoring large open spaces like parking lots, plazas, crossroads, and perimeters of large industrial structures. We are particularly interested in monitoring human and vehicular traffic patterns. These patterns are very informative and valuable from the point of view of security. For example, an M-type pedestrian trajectory in a parking lot may signify a potential vehicle break-in activity. This is a "stroll mode" favorite with vehicle thieves as they look into the interior of parked vehicles trying to single out the most appropriate target. Even simpler patterns like an overspeeding vehicle in a slow-speed parking lot area may be indicative of a getaway effort. Traffic statistics around a commercial or government building are also valuable from the point of view of building operations. They may provide an insight into parking lot utilization during different times and days. This insight can support a functional redesign of the open space to better facilitate transportation and safety needs.

A comprehensive video surveillance and monitoring system depends primarily on two different technologies: computer vision and threat assessment. Computer vision technology consists of the optical and system design, object tracking, and multicamera fusion stages. Threat assessment technology consists of the feature assembly, offline modeling, and threat classification stages (see Fig. 1). We will give a brief overview of each stage and compare our solutions to others proposed in the literature.

Our system is probably the only one that features a formal optical and system design stage. Most of the efforts reported in the literature had as their main objective to demonstrate the feasibility of a novel idea and paid no attention to the practical aspects of fielding a surveillance system. There are a number of

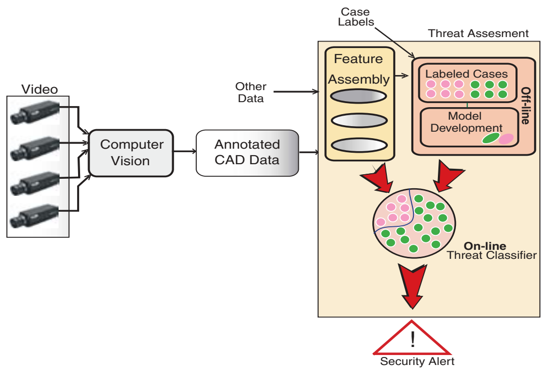


Fig. 1. Architecture of the DETER system

requirements that a surveillance system needs to fulfill to function properly and be commercially viable. It should ensure full coverage of the open space, or blind spots may pause the threat of a security breach. Grimson et al. [4,5] have proposed the concept of a "forest of sensors" (FoS) to address this problem without delving into the implementation details of the idea. The argument is predicated on the fact that video sensors and computational power are getting cheaper and therefore can be employed in mass to provide coverage for any open space. The grand vision of FoS is to drop dozens of cheap small cameras at random in a zone. FoS, then, would work out the position of every camera, build up a three-dimensional representation of the entire area, and begin monitoring activity.

FoS may have some merits under certain military scenarios, but it is inappropriate for commercial applications. Most of the cheap video sensors still do not have the required resolution to accommodate high-quality object tracking. Both cheap and expensive cameras also need to become weatherproof for outdoor use, which increases their cost substantially. Next is the issue of installation cost, which includes the provision of power and the transmission of video signals, sometimes at significant distances from the building. The installation cost for each camera is usually a figure many times its original value. Even if there were no cost considerations, cameras cannot be used arbitrarily in public places. There are restrictions due to the topography of the area (e.g., streets and tree lines) and to city and building ordinances (e.g., aesthetics). All these considerations severely limit the allowable number and positions of cameras for an urban surveillance system.

In addition to optical considerations, there are also system design considerations including the type of computational resources, computer network bandwidth, and display capabilities. In Sect. 2, we provide a detailed account of our method of addressing all the relevant optical and system design issues.

The computer vision component of DETER consists of a moving-object segmenter, a tracker, and a multicamera fusion module. A variety of moving-object segmenters have been reported in the literature. There are two conventional approaches to moving-object segmentation with respect to a static camera: temporal differencing [6] and background subtraction [7]. Temporal differencing is very adaptive to dynamic environments but generally does a poor job of extracting all the relevant object pixels. Background subtraction provides the most complete object data but is extremely sensitive to dynamic scene changes due to lighting and extraneous events. Most researchers have abandoned nonadaptive methods of backgrounding, which are useful only in highly supervised, short-term tracking applications without significant changes in the scene. More recent adaptive backgrounding methods [8] can cope much better with environmental dynamism. However, they still cannot handle bimodal backgrounds and have problems in scenes with many moving objects. Grimson et al. [9] have proposed a more advanced object-detection method whereby each pixel's color channel (R, G, B) is modeled by a mixture of Gaussians. This method features a far better adaptability and can handle even bimodal backgrounds (e.g., swaying tree branches). The secret is in the powerful representation scheme. Each Normal reflects the expectation that samples of the same scene point are likely to display Gaussian noise distributions. The mixture of Normals reflects the expectation that more than one process may be observed over time.

Our method is similar to Grimson's method in the sense that we also use a multi-Normal representation at the pixel level. However, this is where the similarity ends. We use an expectation-maximization (EM) algorithm to initialize our models. In contrast to the K-means approximation or to random initialization, our EM algorithm provides strong initial statistical support that facilitates fast convergence and stable performance of the segmentation operation. We use the Jeffreys (J) divergence measure as the matching criterion between Normals of incoming pixels and existing model Normals. This is a far superior measure to the fixed value (two standard deviations) used in Stauffer and Grimson [9]. Finally, when a match is found, the model update is performed using the method of moments. When a match is not found, the update is performed in a way that guarantees the inclusion of the incoming distribution in the foreground set.

The method described above allows us to identify foreground pixels in each new frame while updating the description of each pixel's mixture model. The labeled foreground pixels can then be assembled into objects using a connected components algorithm [10]. Establishing correspondence of objects between frames (tracking) is accomplished using a linearly predictive multiple hypothesis tracking algorithm that incorporates both position and size. We describe our object-tracking method in detail in Sect. 3.

No single camera is able to cover large open spaces, like parking lots, in their entirety. Therefore, we need to fuse the fields of view (FOVs) of the various cameras into a coherent superpicture to maintain global awareness. We fuse (calibrate) multiple cameras by computing the respective homography matrices. The computation is based on the identification of several landmark points in the common FOV between camera pairs. The landmark points are physically marked on the scene and sampled through the user interface. The calibration thus achieved is very accurate, much more accurate than the one achieved through the matching and fitting of tracked objects [11]. The trade-off is that setting up and sampling landmark points is a rather laborious process. Since, however, it is done only during the installation of the system, it is a perfectly viable practical solution. A short description of our fusion method is given in Sect. 4.

The threat assessment portion of DETER consists of a feature assembly module followed by a threat classifier. Feature assembly extracts various security-relevant statistics from object tracks and groups of tracks. The threat classifier decides in real time whether a particular point in feature space constitutes a threat. The classifier is assisted by an offline threat-modeling component (see Fig. 1). Section 5 provides details about our threat assessment methodology.

In Sect. 6 we provide experimental results that demonstrate the performance of the DETER system. Finally, we conclude the paper in Sect. 7.

2 Optical and system design

The optical and overall system design for DETER includes the specification of a camera set arrangement that optimally covers the HL Minneapolis parking lot. It also includes the specification of the computational resources necessary to run the DETER algorithms in real time. Finally, it includes the specification of the display hardware and software.

The optical design effort, in particular, has the following objectives:

1. Specify the camera model
2. Specify the camera lens
3. Specify the number of cameras
4. Specify the camera locations

We decided to employ dual-channel camera systems. These systems utilize a medium-resolution color camera during the day and a high-resolution grayscale camera during the night. Switching from day to night operation is controlled automatically through a photosensor. The dual-channel technology capitalizes on the fact that color information in the low light conditions at night is lost. Therefore, there is no reason to use color cameras during nighttime conditions. Instead we can employ cheaper and higher resolution grayscale cameras to compensate for the loss of color information. We have selected the DSE DS-5000 dual-channel system as the camera model. The color day camera has a resolution of $H_D = 480$ lines/frame. The grayscale night camera has a resolution of $H_N = 570$ lines/frame. The DSE DS-5000 camera system has a 2.8 to 6-mm $f/1.4$ varifocal auto iris lens for both day and night cameras. This permits us to vary the FOV of the cameras from $FOV = 44.4 - 82.4°$.

We seek an optimal solution that provides coverage to the entire parking lot area with the minimum number of cameras and installation cost. The topography of the area under surveillance imposes practical constraints. For example, we cannot place a camera pole in the middle of the road, existing poles and rooftops should be utilized to the extent possible to reduce the installation cost, and city codes regarding the aesthetics must be obeyed. Taking into account all these considerations we can delineate in the computer aided design (CAD) of the parking lot the possible installation sites. These are usually only a small fraction of the entire open area, and therefore our search space is drastically reduced.

The installation search space is reduced even further when we consider the constraints imposed by the computer vision algorithms. Specifically:

1. An urban surveillance system such as DETER monitors two kind of objects – vehicles and people. In terms of size, people are the smallest objects under surveillance. Therefore, their footprint should drive the requirements for the limiting range of the cameras. In turn, the determination of the limiting range will help us to verify if there is any space in the parking lot that is not covered under any given camera configuration.
2. Each camera should have an overlapping FOV with at least one more camera. The overlapping arrangement should be done in such a way that we can transition from one camera to another through indexing of the overlapped areas and manage to visit all the cameras in a unidirectional trip without encountering any discontinuity.
3. The overlap in the FOVs should be between 25% and 50% for the multicamera calibration algorithm to perform reliably. This requirement stems from the need to get several well spread landmark points in the common FOV for accurate homography. Usually a portion of the overlapping area cannot be used for landmarking because it is covered by nonplanar structures like tree lines. Therefore, at times the common area between two cameras may be required to cover as much as half of the individual FOVs.

As mentioned earlier, the DSE DS-5000 cameras feature a varifocal lens with a FOV that can range between 44.4 and 82.4°. We chose the intermediate value of $FOV = 60°$ as the basis of our calculations. To satisfy the overlapping constraints, we may need to increase or decrease the FOV of some of the cameras from this average value. The camera placement algorithm proceeds as follows:

1. In one of the allowed installation sites, place a camera in such a way that its FOV borders the outer edge of the parking lot.
2. Continue adding cameras around the initial camera until you reach the next outer edge of the parking lot. Make sure there is at least 25% overlap in neighboring FOVs.
3. Compute the limiting range of the installed cameras. By knowing the FOV and the limiting range we can know the full useful coverage area for each camera.
4. Continue with the next installation site just outside of the already covered area. Make sure that at least one of the new cameras overlaps at least 25% with one of the previous cameras.
5. Repeat the previous three steps until the entire parking lot area is covered.
6. Make some postprocessing adjustments. These usually involve the increase or reduction of the FOV for some of the cameras. This FOV adjustment is meant to either trim excessive overlap or add extra overlap in areas where there is little planar space (lots of trees).

Of particular interest is the computation of a camera's limiting range R_c. It is computed from the equation

$$R_c = \frac{P_f}{\tan(IFOV)}$$

where P_f is the smallest acceptable pixel footprint of a human and $IFOV$ is the instantaneous field of view. Based on our experimental experience, the signature of the human body should not be smaller than a $w \times h = 3 \times 9 = 27$ pixel rectangle on the focal plane array (FPA). Clusters with fewer than 27 pixels are likely to be below the noise level. If we assume that the width of an average person is about $W_p = 24$ in., then the pixel footprint $P_f = 24/3 = 8$. The $IFOV$ is computed from the following formula:

$$IFOV = \frac{FOV}{L_{FPA}}$$

where L_{FPA} is the resolution for the camera. For $FOV = 60°$ and $L_{FPA} = 480$ pixels (color day camera), the limiting range is $R_c = 305$ ft. For $FOV = 60°$ and $L_{FPA} = 570$ pixels (grayscale night camera), the limiting range is $R_c = 362$ ft. In other words, between two cameras with the same FOV the higher-resolution camera has the larger useful range. Conversely, if two cameras have the same resolution, the one with the smaller FOV has the larger useful range. During postprocessing we needed to reduce the FOV ($FOV = 52°$) in some of the lower-resolution day camera channels to increase their effective range limit. Extended tree lines in the HL parking lot necessitate larger overlapping areas than the anticipated minimum.

A good optical design is essential to the success of an urban surveillance system, and many computer vision projects often ignore this aspect altogether. The principles, algorithms, and computations we used for the DETER optical design can be codified and automate the optical design of future similar security systems in any parking lot or open area.

Our study concluded that seven cameras in the configuration shown in Fig. 2 is the recommended arrangement for our parking lot. We have assigned one standard PC for the processing requirements of each camera. One of the seven PCs is designated as the server, and this is where the fusion of information from all seven cameras takes place. The fused information is displayed in a 44-in. flat panel display along with all the necessary annotation. This comprehensive high-quality picture allows the security operator to maintain instant awareness without the distraction of multiple fragmented views.

In a later stage, four more cameras were added to monitor movement in the vicinity of the building's air intakes (see Fig. 3). The placement and type of the air-intake cameras was determined by the same algorithm we used for the parking lot scenario. The air-intake cluster of cameras is disjointed from the parking lot cameras and controlled by the product and not the R&D version of DETER. It is meant to demonstrate the application of video-based detection technology in a very specific and highly lethal threat scenario: that of chem-bio attack through the air intakes of commercial buildings. We will provide more information about this application of DETER in Sect. 6.

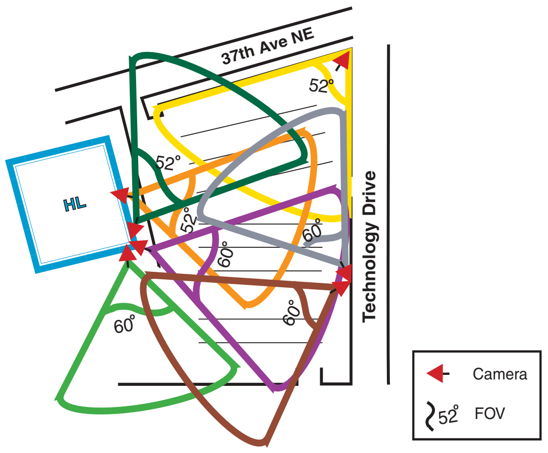


Fig. 2. Camera configuration scheme for DETER in the HL parking lot. This figure depicts the FOVs for the cameras' day channels. The FOVs for the night channels are somewhat different

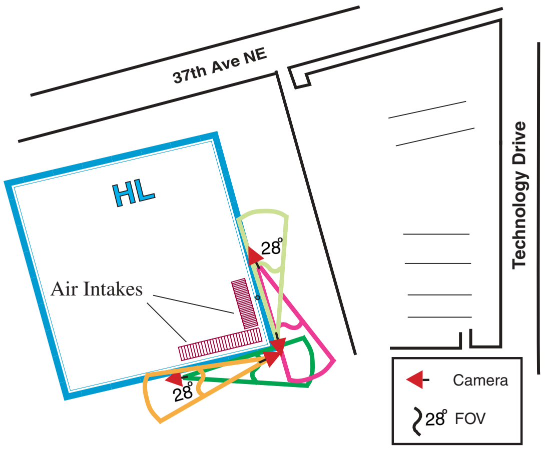


Fig. 3. Camera configuration scheme around the air intakes of the HL building. This cluster of cameras is controlled by the product version of DETER

3 Object tracking

3.1 Initialization

The goal of the initialization phase is to provide statistically valid values for the pixels corresponding to the scene. These values are then used as starting points for the dynamic process of foreground and background awareness. Initialization happens only once, and there are no strict real-time processing requirements for this phase. We accumulate a certain number of frames N ($N = 70$) and then process them offline.

Each pixel $\mathbf{X}$ is considered to be a mixture of three time-varying trivariate normal distributions:

$$\mathbf{X} \sim \sum_{i=1}^{3} \pi_i \mathbf{N}_3 \left(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\right)$$

where

$$\pi_i \geq 0, \quad i = 1, \ldots, 3 \quad \text{and} \quad \sum_{i=1}^{3} \pi_i = 1$$

are the mixing proportions (weights) and $\mathbf{N}_3 \left(\boldsymbol{\mu}, \boldsymbol{\Sigma}\right)$ denotes a trivariate Normal distribution with vector mean $\boldsymbol{\mu}$ and variance-covariance matrix $\boldsymbol{\Sigma}$. The distributions are trivariate to account for the three component colors (red, green, and blue) of each pixel in the general case of a color camera.

Other similar methods reported in the literature initialize the pixel values either with random numbers or with the K-means algorithm. Random initialization results in slow learning during the dynamic mixture model update phase. Sometimes it even results in instability. Initialization with the K-means method gives significantly better, but not optimal, results. K-means is not an optimal method because it commits each incoming data point to a particular distribution in the mixture model. Ideally, each data point should be partially committed to all of the existing distributions. The level of its commitment would be described by appropriate weighting factors. This is how the expectation-maximization (EM) algorithm [12] works. We use the EM algorithm to estimate the parameters of the initial distribution: $\pi_i, \boldsymbol{\mu}_i$, and $\boldsymbol{\Sigma}_i, i = 1, \ldots, 3$ for every pixel $\mathbf{X}$ in the scene. Since the EM algorithm is applied offline over N frames, that means that for each pixel we have at our disposal N data points in time. These data points $\mathbf{x}_j, j = 1, \ldots, N$ are triplets:

$$\mathbf{x}_j = \begin{pmatrix} x_j^R \\ x_j^G \\ x_j^B \end{pmatrix}$$

where x_j^R, x_j^G, and x_j^B stand for the measurement we received from, respectively, the red, green, and blue channels of the camera for the specific pixel at time j. These data $\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N$ are assumed to be sampled from a mixture of three trivariate Normals:

$$\mathbf{x}_j \sim \sum_{i=1}^{3} \pi_i \mathbf{N}_3 \left[\begin{pmatrix} \mu_i^R \\ \mu_i^G \\ \mu_i^B \end{pmatrix}, \sigma_i^2 I \right]$$

where the variance-covariance matrix is assumed to be diagonal with x_j^R, x_j^G, x_j^B, having identical variance within each Normal component but not across all components (i.e., $\sigma_k^2 \neq \sigma_l^2$ for $k \neq l$ components).

Originally we provide the algorithm with some crude estimates of the parameters of interest: $\pi_i^{(0)}$, $\boldsymbol{\mu}_i^{(0)}$, and $\left(\sigma_i^{(0)}\right)^2$. We obtain these estimates using the K-means method. Then we apply the loop described in Appendix 1. The EM process is applied for every pixel in the camera's focal plane array (FPA). The result is a mixture model of three Normal distributions per pixel. These Normal distributions represent three potentially different states for each pixel. Some of these states could be background states and some could be transient foreground states. The EM algorithm is computationally intensive, but since the initialization phase takes place offline, this is a nonissue. In our experiments, EM has proved a superior initialization method that caters to fast learning and provides exceptional stability to the subsequent main stage of object segmentation. This is especially true when initialization happens during challenging weather conditions like fast moving clouds or other cases of multimodal background.

It is worth noting that during both initialization and regular processing we chose to represent each color component of a pixel as a mixture of three Normals. We arrived at this choice after experimenting with a whole set of different mixture schemes. Specifically, we ran versions of DETER featuring mixtures of two, three, four, and five Normals simultaneously on the benchmarking video streams (see Sect. 7). The DETER version featuring three Normals performed better than any other version. It appears that three Normals per color channel is a sufficiently rich representation scheme to capture natural motion and phenomena. Adding Normals beyond that simply increases the computational load without improving the quality of foreground-background segmentation.

3.2 Segmentation of moving objects

The initial mixture model is updated dynamically thereafter. The update mechanism is based on the incoming evidence (new camera frames). Several things could change during an update cycle:

1. The form of some of the distributions could change (weight π_i, mean $\boldsymbol{\mu}_i$, and variance σ_i^2).
2. Some of the foreground states could revert to background and vice versa.
3. One of the existing distributions could be dropped and replaced by a new distribution.

At every point in time the distribution with the strongest evidence is considered to represent the pixel's most probable background state. Figure 4 presents a visualization of the mixture of Normals model, while Fig. 5 depicts the update mechanism for the mixture model.

The update cycle for each pixel proceeds as follows:

1. First, the existing distributions are arranged in descending order based on their weight values.

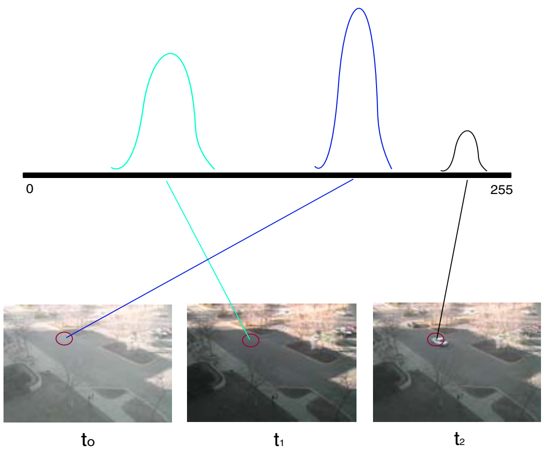


Fig. 4. Visualization of the mixture of Normals model at the pixel level. For the sake of simplicity the Normals of only one of the three color channels are depicted

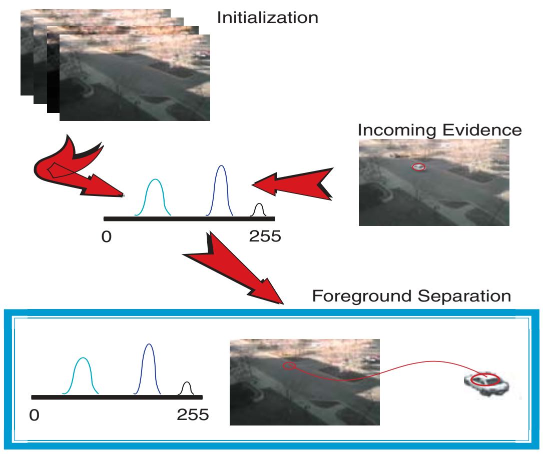


Fig. 5. Visualization of the mixture model update mechanism. For the sake of simplicity the Normals of only one of the three color channels are depicted. The *red ellipse* marks the pixel area being monitored

2. Second, the algorithm selects the first B distributions that account for a predefined fraction of the evidence T:

$$B = \arg\min_b \left\{ \sum_{i=1}^{b} w_i > T \right\}$$

where w_i, $i = 1, \ldots, b$ are the respective distribution weights. These B distributions are considered background distributions, while the remaining $3 - B$ distributions are considered foreground distributions. We have experimentally established that the optimal value for threshold T is $T = 0.80$ (see Sect. 7 for more details).
3. Third, the algorithm checks if the incoming pixel value can be ascribed to any of the existing Normal distributions. The matching criterion we use is the Jeffreys (J) divergence measure and is a key differentiator of our approach from other similar approaches.
4. Fourth, the algorithm updates the mixture of distributions and their parameters. The nature of the update depends on the outcome of the matching operation. If a match is found, the update is performed using the method of moments. This is also a key differentiator of our approach. If a match is not found, the weakest distribution is replaced with a new distribution. The update performed in this case guarantees the inclusion of the new distribution in the foreground set, which is another novelty of our method.

The matching and model update operations are quite involved [13] and are described in detail in the next three subsections.

3.2.1 The matching operation

The Kullback-Leibler (KL) information number between two distributions f and g is defined as:

$$K(f,g) = E_f \left[log \left(\frac{f}{g} \right) \right] = \int log \left(\frac{f(x)}{g(x)} \right) f(x) \ \ dx$$

A formal interpretation of the use of the KL information number given by Fergusson [14] is "... whether the likelihood ratio can discriminate between f and g when f is the true distribution."

For the purpose of our algorithm we need to define some *divergence* measure between two distributions so that if the divergence measure between the new distribution and one of the existing distributions is "too small," we will pool these two together (i.e., the new data point will be attached to one of the existing distributions). For a divergence measure $d(f, g)$ it is necessary to satisfy (at least) the following three axioms:

$$\begin{aligned} &(a)\ d(f,f) = 0 \\ &(b)\ d(f,g) \geq 0 \\ &(c)\ d(f,g) = d(g,f) \end{aligned}$$

The KL information number between two distributions f and g does not satisfy (c), since:

$$K(f,g) = E_f \left[log \left(\frac{f}{g} \right) \right] \neq E_g \left[log \left(\frac{g}{f} \right) \right] = K(g,f)$$

i.e., the KL information number is not symmetric around its arguments and thus cannot be considered as a divergence measure.

Jeffreys [15] proposed as divergence measure between two distributions f and g the following:

$$J(f,g) = \int [f(x) - g(x)]\, log \left(\frac{f(x)}{g(x)} \right) dx$$

This divergence measure is closely related to the KL information number, as the following Lemma indicates.

Lemma 1.

$$J(f,g) = K(f,g) + K(g,f)$$

Proof:

$$\begin{aligned} J(f,g) &= \int [f(x) - g(x)]\, log \left(\frac{f(x)}{g(x)} \right) dx \\ &= \int f(x) log \left(\frac{f(x)}{g(x)} \right) dx + \\ &\quad \int g(x) log \left(\frac{g(x)}{f(x)} \right) dx \\ &= K(f,g) + K(g,f) \end{aligned}$$

The $J(f, g)$ is now symmetric around its arguments since

$$J(f,g) = K(f,g)+K(g,f) = K(g,f)+K(f,g) = J(g,f)$$

and also satisfies axioms (a) and (b). Thus, $J(f, g)$ is a divergence measure between f and g.

We will use $J(f, g)$ to determine whether or not the new distribution matches one of the three existing distributions. The three existing Normal distributions are: $f_i \sim \mathbf{N}_3(\boldsymbol{\mu}_i, \sigma_i^2 I)$, $i = 1, \ldots, 3$. The incoming distribution is $g \sim \mathbf{N}_3(\boldsymbol{\mu}_g, \sigma_g^2 I)$. We assume that

$$\boldsymbol{\mu}_g = \mathbf{x}_t \qquad \text{and} \qquad \sigma_g^2 = 9$$

where $\mathbf{x}_t$ is the incoming data point. The choice of $\sigma_g^2 = 9$ is the result of experimental observation about the typical spread of successive pixel values in small time windows. The three divergence measures between g and f_i, $i = 1, \ldots, 3$ will be given by the following formula (proof in the Appendix):

$$\begin{aligned} J(f_i, g) = &\frac{3}{2} \left(\frac{\sigma_i}{\sigma_g} - \frac{\sigma_g}{\sigma_i} \right)^2 \\ &+ \frac{1}{2} \left(\frac{1}{\sigma_i^2} + \frac{1}{\sigma_g^2} \right) (\boldsymbol{\mu}_g - \boldsymbol{\mu}_i)'(\boldsymbol{\mu}_g - \boldsymbol{\mu}_i) \end{aligned}$$

Once the three divergence measures have been calculated we find the distribution f_j $(1 \leq j \leq 3)$ for which

$$J(f_j, g) = \min_{1 \leq i \leq 3} \{J(f_i, g)\}$$

and we have a match between f_j and g if and only if

$$J(f_j, g) \leq K^*$$

where K^* is a prespecified cutoff value. In the case where $J(f_j, g) > K^*$, the new distribution g cannot be matched to any of the existing distributions.

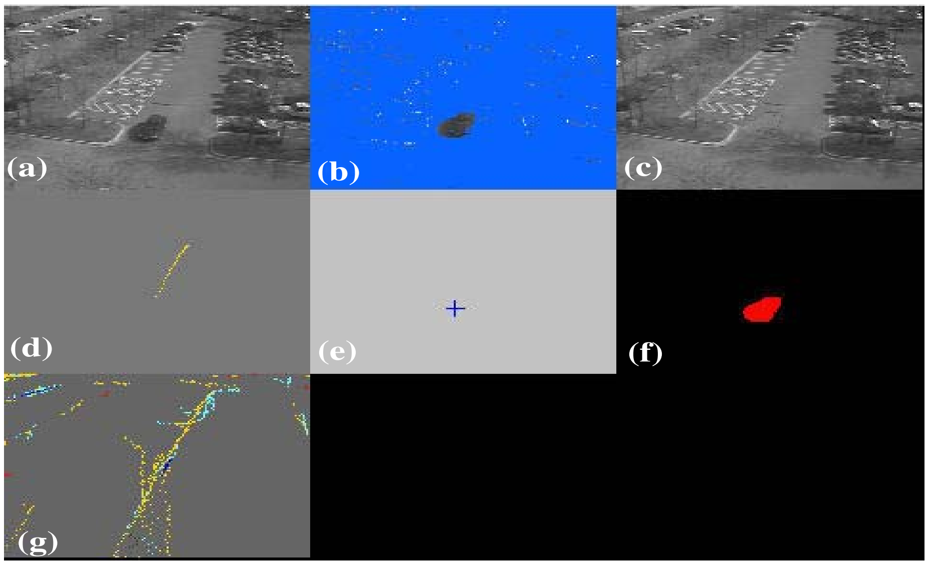


Fig. 6. Visualization of the computer vision operation of DETER. The snapshot was taken "live" on April 15, 2002. **a** Live video feed. **b** Segmented moving object. **c** Dynamically updated background. **d** Trajectories of the current moving objects. **e** Centroids of the moving objects. **f** Results of the blob analysis. **g** Cumulative trajectory visualization of human and vehicle traffic for the past hour

3.2.2 Model update when a match is found

If the incoming distribution matches one of the existing distributions, we pool them together to a new Normal distribution. This new Normal distribution is considered to represent the current state of the pixel. The state is labeled either background or foreground depending on the position of the matched distribution in the ordered list of distributions. The next issue requiring clarification is how we update the parameters of the mixture. We use the *method of moments*. First, we introduce some learning parameter α, which weighs on the weights of the existing distributions. So we subtract $100\alpha\%$ weight from each of the three existing weights and assign it to the incoming distribution's weight. In other words, the incoming distribution has weight α since

$$\sum_{i=1}^{3} \alpha\pi_i = \alpha \sum_{i=1}^{3} \pi_i = \alpha$$

and the three existing distributions have weights $\pi_i(1-\alpha)$, $i = 1, \ldots, 3$.

Obviously for α we need to have $0 < \alpha < 1$. The choice of α depends mainly on the choice of K^*. The two quantities are inversely related. The smaller the value of K^*, the higher the value of α and vice versa. The values of K^* and α are also affected by how much noise we have in the monitoring area. So if, for example, we were monitoring an outside region and had a lot of noise due to environmental conditions (rain, snow, etc.), then we would need a "high" value of K^* and thus a "small" value of α since nonmatch to one of the distributions is very likely to be caused by background noise. On the other hand, if we were recording indoors where the noise is almost nonexistent, we would prefer a "small" value of K^* and thus a "higher" value of α because any time we do not get a match to one of the existing three distributions is very likely to occur due to some foreground movement (since the background has almost no noise at all).

Let us assume that we have a match between the new distribution g and one of the existing distributions f_j where $1 \le j \le 3$. Then we update the weights of the mixture model as follows:

$$\pi_{i,t} = (1-\alpha)\pi_{i,t-1} \qquad i = 1, \ldots, 3 \quad \text{and} \quad i \neq j$$
$$\pi_{j,t} = (1-\alpha)\pi_{j,t-1} + \alpha$$

We also update the mean vectors and the variances. If we call w_1 as $(1-\alpha)\pi_{j,t-1}$, i.e., w_1 is the weight of the jth component (which is the winner in the match) before pooling it with the new distribution g, and if we call $w_2 = \alpha$, i.e., the weight of the new observation, then define

$$\rho = \frac{w_2}{w_1 + w_2} = \frac{\alpha}{(1-\alpha)\pi_{j,t-1} + \alpha}$$

using the method of moments [16] we get

$$\boldsymbol{\mu}_{j,t} = (1-\rho)\boldsymbol{\mu}_{j,t-1} + \rho\boldsymbol{\mu}_g$$

$$\sigma^2_{j,t} = (1-\rho)\sigma^2_{j,t-1} + \rho\sigma^2_g + \rho(1-\rho)(\mathbf{x}_t - \boldsymbol{\mu}_{j,t-1})'(\mathbf{x}_t - \boldsymbol{\mu}_{j,t-1})$$

while the other two (unmatched) distributions keep the *same* mean and variance that they had at time $t-1$.

3.2.3 Model update when a match is not found

In the case where a match is not found (i.e., $\min_{1 \le i \le 3} K(f_i, g) > K^*$), we commit the current pixel state to be foreground and we replace the last distribution in the ordered list with a new one. The parameters of the new distribution are computed as follows:

1. The mean vector $\boldsymbol{\mu}_3$ is replaced with the incoming pixel value.

2. The variance σ_3^2 is replaced with the minimum variance from the list of distributions. This is in the context of our philosophy that considers incoming data points as means of narrow distributions. We have experimentally verified that in short time windows the distribution of color values regarding a specific pixel in the scene do not vary greatly. If the initial distributions prove persistent (i.e., background), they spread out over time as changes in illumination conditions take effect.
3. The weight of the new distribution is computed as follows:

$$w_{3,t+1} = \frac{1-T}{2}$$

where T is the background threshold index. This formula guarantees the classification of the current pixel state as foreground. The weights of the two remaining distributions are updated according to the following formula:

$$w_{i,t+1} = w_{i,t} + \frac{w_{3,t} - (1-T)/2}{2}$$

3.3 Multiple hypothesis predictive tracking

In the previous section, we described a statistical procedure for performing online segmentation of *foreground pixels* corresponding to moving objects of interest, i.e., people and vehicles. In this section, we describe how to form trajectories traced by the various moving objects. Figure 6 shows a snapshot of the output from the various computer vision modules of DETER. The basic requirement for forming object trajectories is the calculation of blob centroids (corresponding to moving objects). Blobs are formed after we apply a standard eight-connected component analysis algorithm to the foreground pixels. The connected component algorithm filters out blobs with area less than $A = 3 \times 9 = 27$ pixels as noise. According to our optical computation in Sect. 2, this is the minimal pixel footprint of the smallest object of interest (human) in the cameras' FOV.

A multiple hypothesis tracking (MHT) algorithm is then used that groups the blob centroids of foreground objects into distinct trajectories. MHT is considered to be the best approach to multitarget tracking applications. It is a recursive Bayesian probabilistic procedure that maximizes the probability of correctly associating input data with tracks. Its superiority against other tracking algorithms stems from the fact that it does not commit early to a trajectory. Early commitment usually leads to mistakes. MHT groups the input data into trajectories only after enough information has been collected and processed. In this context, it forms a number of candidate hypotheses regarding the association of input data with existing trajectories. MHT has shown to be the method of choice for applications with heavy *clutter* and dense *traffic*. In difficult multitarget tracking problems with crossed trajectories, MHT performs very well as compared with other tracking procedures such as the nearest neighbor correlation and the joint probabilistic data association [17].

Figure 7 depicts the architecture of our MHT algorithm. In the following subsections we will describe the function of each module .

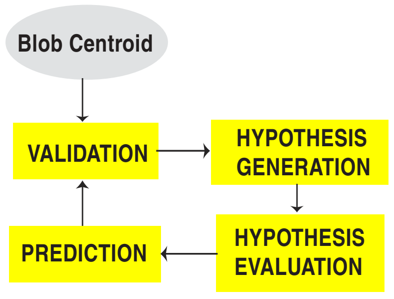


Fig. 7. Architecture of the MHT algorithm

3.3.1 Prediction

An integral part of any tracking system is the prediction module. Prediction provides estimates of moving objects' states and in the DETER system is implemented as a Kalman filter. Kalman filter predictions are made based on a priori models for target dynamics and measurement noise.

The state vector describing the motion of a foreground object (blob) consists of the position and velocity of its centroid expressed in pixel coordinates, i.e.,

$$\mathbf{x}_k = \begin{pmatrix} x_k \; \dot{x}_k \; y_k \; \dot{y}_k \end{pmatrix}^T$$

The state space model is a constant velocity model given by:

$$\mathbf{x}_{k+1} = \mathbf{F}_k \mathbf{x}_k + \mathbf{u}_k$$

with transition matrix $\mathbf{F}_k$:

$$\mathbf{F}_k = \begin{pmatrix} 1 & dt & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & dt \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

The process noise is white noise with zero mean and covariance matrix:

$$\mathbf{Q}_k = E[\mathbf{u}_k \mathbf{u}_k^T] = \begin{pmatrix} \frac{dt^3}{3} & \frac{dt^2}{2} & 0 & 0 \\ \frac{dt^2}{2} & dt & 0 & 0 \\ 0 & 0 & \frac{dt^3}{3} & \frac{dt^2}{2} \\ 0 & 0 & \frac{dt^2}{2} & dt \end{pmatrix} q$$

where q is the process variance. The measurement model describes how measurements are made and is defined by

$$\mathbf{z}_k = \mathbf{H}\mathbf{x}_k + \mathbf{v}_k$$

with

$$\mathbf{H} = \begin{pmatrix} 1 \; 0 \; 0 \; 0 \\ 0 \; 0 \; 1 \; 0 \end{pmatrix}$$

and a constant 2×2 covariance matrix of measurement noise given by

$$\mathbf{R}_k = E[\mathbf{v}_k \mathbf{v}_k^T]$$

Based on the above assumptions, the Kalman filter provides minimum mean squared estimates $\widehat{\mathbf{x}}_{k|k}$ of the state vector according to the following equations:

$$\begin{aligned}
\mathbf{K}_k &= \mathbf{P}_{k|k-1}\mathbf{H}^T[\mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R}_k]^{-1} \\
\mathbf{P}_{k|k} &= [\mathbf{I} - \mathbf{K}_k\mathbf{H}]\mathbf{P}_{k|k-1} \\
\mathbf{P}_{k+1|k} &= \mathbf{F}_k\mathbf{P}_{k|k}\mathbf{F}_k^T + \mathbf{Q}_k \\
\widehat{\mathbf{x}}_{k|k} &= \widehat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k[\mathbf{z}_k - \mathbf{H}\widehat{\mathbf{x}}_{k|k-1}] \\
\widehat{\mathbf{x}}_{k+1|k} &= \mathbf{F}_k\widehat{\mathbf{x}}_{k|k}
\end{aligned}$$

3.3.2 Validation

Validation is a process that precedes the generation of hypotheses regarding associations between input data (blob centroids) and the current set of trajectories (tracks). Its function is to exclude, early on, associations that are unlikely to happen, thereby limiting the number of possible hypotheses to be generated. The vector difference between measured and predicted states $\boldsymbol{\nu}_k$ is a random variable characterized by the covariance matrix $\mathbf{S}_k$:

$$\begin{aligned}
\boldsymbol{\nu}_k &= \mathbf{z}_k - \mathbf{H}\widehat{\mathbf{x}}_{k|k-1} \\
\mathbf{S}_k &= \mathbf{H}\mathbf{P}_{k|k-1}\mathbf{H}^T + \mathbf{R}_k
\end{aligned}$$

For every track from the list of current tracks there exists an associated *gate*. A gate can be visualized as an area surrounding a track's predicted location (next move). In our case, a gate is an elliptical shape defined by the squared *Mahalanobis distance*:

$$d^2 = \boldsymbol{\nu}_k^T \mathbf{S}_k^{-1} \boldsymbol{\nu}_k$$

An incoming measurement (blob centroid) is associated with a track only when it falls within the gate of the respective track. Mathematically this is expressed by

$$d^2 \le D_{threshold}$$

The result of validating a new set of blob centroids takes the form of an *ambiguity matrix*. An example of an ambiguity matrix corresponding to a hypothetical situation of an existing set of two tracks (T_1 and T_2) and a current set of three measurements ($z_1(k)$, $z_2(k)$, and $z_3(k)$) is given in Eq. 1.

$$\Omega = \begin{array}{cc} \begin{array}{cccc} T_F & T_1 & T_2 & T_N \end{array} & \\ \left(\begin{array}{cccc} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right) & \begin{array}{c} z_1(k) \\ z_2(k) \\ z_3(k) \end{array} \end{array} \tag{1}$$

The columns of the ambiguity matrix denote the current set of tracks, with the first and last columns being reserved for false alarms (T_F) and new tracks (T_N), respectively. The rows correspond to the particular measurements of blob centroids made on the current frame. Nonzero elements of the ambiguity matrix signal that the respective measurements are contained in the validation region of the associated track. It is important to note that we further constrain the assignments in the ambiguity matrix by allowing each measurement in the current scan to be associated with only one track. Furthermore, we assume that a track is paired with at most one measurement per iteration. We therefore limit the number of nonzero elements in any row or column (barring the first and last columns) to one. We thus make the ambiguity matrix a cost matrix as it is defined in linear assignment problems [18]. This formulation makes the ambiguity matrix a representation of a new set of hypotheses about blob centroid-track pairings.

3.3.3 Hypothesis generation

Central to the implementation of the MHT algorithm is the generation and representation of track hypotheses. Tracks are generated based on the assumption that a new measurement may:

1. belong to an existing track,
2. be the start of a new track, or
3. be a false alarm.

Assumptions are validated through the validation process described in Sect. 3 before they are incorporated into the hypothesis structure. The complete set of track hypotheses can be represented by a hypothesis matrix as shown in Table 1. The hypothetical situation in Table 1 corresponds to a set of two scans of two and one measurements made, respectively, on frame $k = 1$ and $k + 1 = 2$. Some notation clarification is in order. A measurement $z_j(k)$ is the jth observation (blob centroid) made on frame k. In addition, a false alarm is denoted by 0, while the formation of a new track (T_{newID}) generated from an old track (T_{oldID}) is shown as $T_{newID}(T_{oldID})$. The first column in this table is the hypothesis index. In our example case, we have a total of four hypotheses generated during scan 1, and eight more are generated during scan 2. The last column lists the tracks that the particular hypothesis contains (e.g., hypothesis H_8 contains track nos. 1 and 4). The row cells in the hypothesis table denote the tracks to which the particular measurement $z_j(k)$ belongs (e.g., under hypothesis H_{10} the measurement $z_1(2)$ belongs to track no. 5). A hypothesis matrix is represented computationally by a tree structure as it is schematically shown in Fig. 8. The branches of the tree are in essence the hypotheses about measurement-track associations.

Table 1. Complete set of track hypotheses with the associated sets of tracks

Hypothesis	$z_1(1)$	$z_2(1)$	$z_1(2)$	Track no.
H_1	0	0	–	0
H_2	1	0	–	1
H_3	0	2	–	2
H_4	1	2	–	1,2
H_5	1	0	3(1)	3
H_6	1	2	3(1)	2,3
H_7	0	2	4(2)	4
H_8	1	2	4(2)	1,4
H_9	0	0	5	5
H_{10}	1	0	5	1,5
H_{11}	0	2	5	2,5
H_{12}	1	2	5	1,2,5

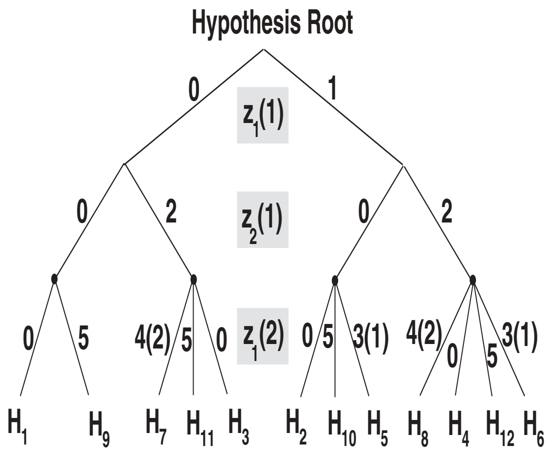


Fig. 8. Formation of a hypothesis tree

3.3.4 Hypothesis evaluation

As is evident from the above example, the hypothesis tree can grow exponentially with the number of measurements. We apply two measures to reduce the number of hypotheses. Our first measure is to cluster the hypotheses into disjoint sets [19]. In this sense, tracks that do not compete for the same measurements compose disjoint sets, which in turn are associated with disjoint hypothesis trees. Our second measure is to assign probabilities on every branch of hypothesis trees. The set of branches with the N_{hypo} highest probabilities are only considered. The derivation of hypothesis probabilities is beyond the scope of this paper. However, the interested reader is referred to [19] and [20]. Suffice it to say simply that a recursive Bayesian methodology is followed for calculating hypothesis probabilities from frame to frame.

4 Multicamera fusion

Monitoring of large sites (such as parking lots) can be accomplished only through the coordinated use of multiple cameras. In DETER, we need to have seamless tracking of humans and vehicles across the whole geographical area covered by all cameras. We produce a panoramic view of the HL parking lot by fusing the individual camera FOVs. Then object motion is registered against a global coordinate system. In turn, this global coordinate system is mapped to the CAD design of the parking lot. The CAD mapping enriches the inference capabilities of the system since it provides awareness about parking stalls and vehicle and pedestrian pathways.

We achieve multicamera registration (fusion) by computing the homography transformation between pairs of cameras. Our homography computation procedure takes advantage of the overlapping that exists between pairs of camera FOVs. We use the pixel coordinates of more than four points to calculate the homography transformation matrix. These points are projections of physical ground plane points that fall in the overlapping area between the two camera FOVs. We select and physically mark these points on the ground with paint during the installation phase. We then sample the corresponding projected image points through the DETER graphical user interface (GUI). This process happens only in the beginning, and once the camera cross registration is complete it is never repeated. In DETER, in order to achieve optimal coverage with the minimum number of sensors, we place the cameras far apart from each other and at varying angles. Therefore, we had to develop a sophisticated warping algorithm to accommodate the large distortions produced by the highly nonlinear homography transformations.

4.1 Homography computation

The homography computation is challenging primarily for two reasons:

- It is an underconstrained problem that is usually based on a small number of matching points.
- It introduces inaccuracies in specialized transformations (e.g., pure rotation or translation).

A very popular and relatively simple method for the computation of the homography matrices is the so-called *least squares* method [11]. This method may provide a poor solution to the underconstrained system of equations due to biased estimation. It also cannot effectively specialize the general homography computation in exceptional cases.

We have adopted the algorithm by Kanatani [21] to compute the homography matrices. The algorithm is based on a statistical optimization theory for geometric computer vision [22] and cures the deficiencies exhibited by the least squares method. The basic premise is that the *epipolar constraint* may be violated by various noise sources due to the statistical nature of the imaging problem (see Fig. 9).

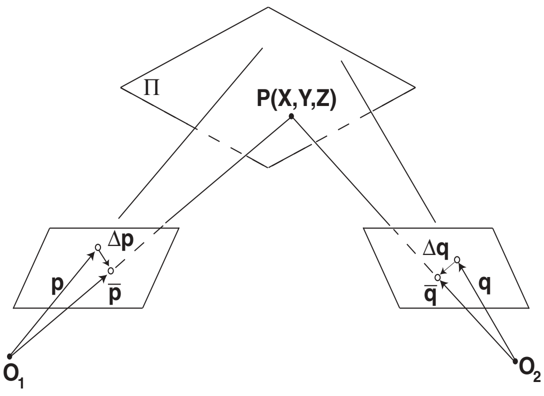


Fig. 9. The statistical nature of the imaging problem affects the epipolar constraint. O_1 and O_2 are the optical centers of the corresponding cameras. $P(X,Y,Z)$ is a point in the scene that falls in the common area between the two camera FOVs. Ideally, the vectors $\overrightarrow{O_1\bar{p}}$, $\overrightarrow{O_2\bar{q}}$, $\overrightarrow{O_1O_2}$ are coplanar. Due to the noisy imaging process, however, the actual vectors $\overrightarrow{O_1p}$, $\overrightarrow{O_2q}$, $\overrightarrow{O_1O_2}$ may not be coplanar

In particular, for every camera pair we compute a 3×3 homography matrix H such that for a number of world points $\mathbf{P}_\alpha(\mathbf{X}_\alpha, \mathbf{Y}_\alpha, \mathbf{Z}_\alpha), \alpha = 1, 2, \cdots, N$ and $N \geq 4$, projected into the image points $\mathbf{p}_\alpha$, and $\mathbf{q}_\alpha$ the following equation holds:

$$\vec{p}_\alpha \times \vec{H}\vec{q}_\alpha = 0, \qquad \alpha = 1, 2, \cdots, N \tag{2}$$

Notice that the symbol $(\times)$ denotes the exterior product and also that the above equation does not hold for the actual image points $\vec{p}_\alpha$ and $\vec{q}_\alpha$ but for the corresponding ideal image points $\bar{\vec{p}}_\alpha$ and $\bar{\vec{q}}_\alpha$ for which the epipolar constraint is satisfied (see Fig. 9). Equivalently, Eq. 2 can be written as:

$$(\vec{X}_\alpha^{(k)}; \vec{H}) = 0, \quad k = 1, 2, 3, \tag{3}$$

with:

$$\vec{X}_\alpha^{(k)} = \vec{e}^{(k)} \times \bar{\vec{p}}_\alpha \bar{\vec{q}}_\alpha^T, \qquad \alpha = 1, 2, \cdots, N, \tag{4}$$

where for any two matrices $\vec{A}$ and $\vec{B}$ $(\vec{A}; \vec{B}) = tr(\vec{A}^T \vec{B})$ and $\vec{e}^{(1)} = (1,0,0)^T$, $\vec{e}^{(2)} = (0,1,0)^T$, $\vec{e}^{(3)} = (0,0,1)^T$. In the statistical framework introduced by Kanatani, homography estimation is equivalent to minimizing the sum of the following squared Mahalanobis distances:

$$J[\vec{H}] = \frac{1}{N} \sum_{\alpha=1}^{N} \sum_{k,l=1}^{3} (\Delta \vec{X}_\alpha^{(k)}, \mathcal{V}(\vec{X}_\alpha^{(k)}, \vec{X}_\alpha^{(l)}) \Delta \vec{X}_\alpha^{(l)})$$

under the constraints described by Eq. 3. Note that the covariant tensor of the matrices $\Delta \vec{X}_\alpha^{(k)}$ and $\Delta \vec{X}_\alpha^{(l)}$ is denoted by

$$\mathcal{V}(\vec{X}_\alpha^{(k)}, \vec{X}_\alpha^{(l)}) = \mathbf{E}[\Delta \vec{X}_\alpha^{(k)} \otimes \Delta \vec{X}_\alpha^{(l)}]$$

where $\Delta \vec{X}_\alpha^{(k)} = \vec{X}_\alpha^{(k)} - \bar{\vec{X}}_\alpha^{(k)}$. The symbol $\otimes$ denotes tensor product. If one uses Lagrange multipliers, estimation of the homography matrix $\vec{H}$ reduces to the optimization of the following functional $J[\vec{H}]$:

$$J[\vec{H}] = \frac{1}{N} \sum_{\alpha=1}^{N} \sum_{k,l=1}^{3} (\mathbf{W}_\alpha^{(kl)}(\vec{H})(\vec{X}_\alpha^{(k)}; \vec{H})(\vec{X}_\alpha^{(l)}; \vec{H})) \tag{5}$$

The (3×3) weight matrix $\mathbf{W}_\alpha(\vec{H})$ is expressed as

$$\vec{W}_\alpha(\vec{H}) = (\vec{p}_\alpha \times \vec{H}\vec{V}[\vec{q}_\alpha]\vec{H}^T \times \vec{p}_\alpha + (\vec{H}\vec{q}_\alpha) \times \vec{V}[\vec{p}_\alpha](\vec{H}\vec{q}_\alpha))_2^- \tag{6}$$

The symbol $(\cdot)_r^-$ symbolizes the generalized inverse of a matrix $(N \times N)$ computed by replacing the smallest $(N - r)$ eigenvalues by zeroes. The computation process for the optimization of the functional in Eq. 5 proceeds as follows:

1. Initialization begins by setting the parameter $c = 0$ and the weights $\vec{W}_\alpha = \mathbf{I}$ for $\alpha = 1, \cdots, N$.
2. We proceed by computing the following matrices:

$$\mathbf{M} = \frac{1}{N} \sum_{\alpha=1}^{N} \sum_{k,l=1}^{3} \vec{W}_\alpha^{(kl)} \vec{X}_\alpha^{(k)} \otimes \vec{X}_\alpha^{(l)}$$

$$\mathbf{N} = \frac{1}{N} \sum_{\alpha=1}^{N} \sum_{k,l=1}^{3} \vec{W}_\alpha^{(kl)} \mathcal{V}(\vec{X}_\alpha^{(k)}, \vec{X}_\alpha^{(l)})$$

3. We next calculate the smallest eigenvalue λ_{min} of $\widehat{\mathbf{M}} = \mathbf{M} - c\mathbf{N}$ and the associated eigenvector $\vec{H}_{min}$.
4. If $\lambda_{min} \to 0$, then the estimated homography matrix $\widehat{\vec{H}} = \vec{H}_{min}$ is returned and the program exits.
   Otherwise, the weights $\vec{W}_\alpha$ are updated according to Eq. 6 and the value of c_{old} is updated according to

$$c_{old} = c_{old} + \frac{\lambda_{min}}{(\vec{H}_{min}; \mathbf{N}\vec{H}_{min})}$$

   In this latter case, the computation continues by looping back through step 2.

5 Threat assessment

Automation is clearly necessary to allow limited and fallible human attention to monitor a large protected space. The primary objective of DETER is to alert security personnel to just those activities that require their scrutiny while ignoring innocuous use. DETER achieves its objective by processing the computer vision information through its threat assessment module. All threat assessment analysis is done after converting the pixel coordinates of the object tracks into a world coordinate system set by the CAD drawing of the facility (see Fig. 1). Thus, we can use well-known landmarks to provide content for evaluating intent. Such landmarks include individual parking spots, lot perimeter, power poles, and tree lines. The coordinate transformation is achieved through the use of the optical computation package *CODE V*.

The feature assembly uses the trajectory information provided by the computer vision module to compute relevant higher level features on a per vehicle/pedestrian basis. The features are designed to capture "common sense" beliefs about innocuous, law-abiding trajectories, and the known or supposed patterns of intruders. In the current prototype, the features calculated include the following:

- object shape number (computed from blob analysis)
- number of sample points
- starting position (x,y)
- ending position (x,y)
- path length
- distance covered (straight line)
- distance ratio (path length / distance covered)
- start time (local wall clock)
- end time (local wall clock)
- duration
- average speed
- maximum speed
- speed ratio (average / maximum)
- total turn angles (radians)
- average turn angles (radians)
- number of M crossings

Most of these are self explanatory, but a few are not so obvious. The object shape number in combination with speed help in differentiating between people and vehicles. The wall clock is relevant since activities on some paths are automatically suspect at certain times of day – late night and early morning in particular.

The turn angles and distance ratio features capture aspects of how circuitous the path followed was. The legitimate users of the facility tend to follow the most direct paths permitted by

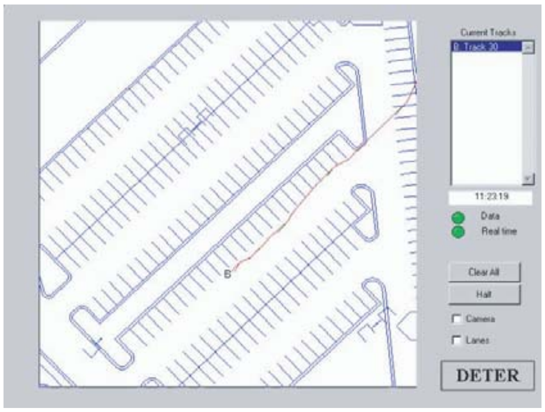


Fig. 10. The trajectory of a vehicle as it was recorded by DETER on the parking lot's CAD. The snapshot was taken in real time. As is evident, the vehicle was driven very close to the parking stalls and went over the pedestrian pavement. The CART classifier flagged this as an alarming event and issued an alert. For alarming events the color of the offending trajectory turns red on the CAD display

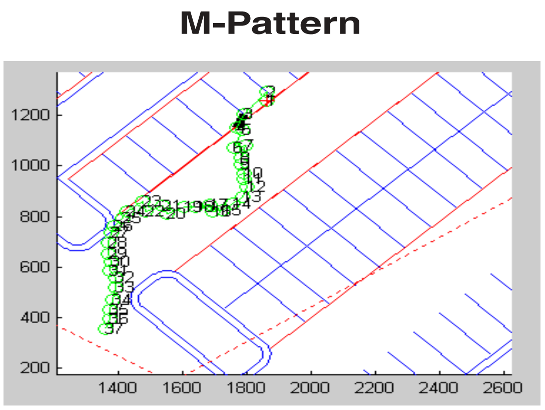


Fig. 11. An M-pattern traced by DETER. The snapshot was re-created offline from the alarm database. The purpose of the offline re-creation was to replace the continuous red line of the alarming trajectory with the centroids of the vehicle across the timeline. The fine spacing of the centroids reveals the high quality of tracking that DETER is capable of achieving even in nontrivial trajectories such as this

the lanes. “Browsers” may take a more serpentine course. For example, the M crossings feature attempts to monitor a well-known tendency of vehicle thieves to systematically check multiple parking stalls along a lane, looping repeatedly back to the vehicle doors for a good look or lock check (two loops yielding a letter M profile). This can be monitored by keeping reference lines for the parking stalls and counting the number of traversals into stalls. An M-type pedestrian crossing captured by DETER is illustrated in Fig. 11.

The output of the feature assembly module for trajectories recorded from the site over some period of time is fed into the offline training module. The goal of offline training is to produce threat models based on a database of features. In the current system, we have gathered data by running DETER over a period of several hours. During this period we staged several suspicious events (like M-type strolls) to enrich our data collection. We then manually labeled the individual object trajectories as either innocuous (*OK*) or suspicious (*THREAT*). In the future, a clustering algorithm will assist in the production of more parsimonious descriptions of object behavior. The complete training data consist of the labeled trajectories and the corresponding feature vectors. They are all processed together by a classification tree induction algorithm based on CART [23]. The trained classifier is then used online to classify incoming live data as either innocuous or suspicious.

Currently DETER can successfully identify the following alarming events:

Overspeeding vehicles. This is considered an activity that puts in danger the safety of people and vehicles in the parking lot. It may also indicate a getaway effort. Any vehicle with average speed over 20 mph raises an alarm.

Overrun of pedestrian pavement. This is by definition an unsafe behavior that requires the immediate attention of the guard (see Fig. 10).

Running pedestrians. This could be either an innocuous (jogger) or suspicious (getaway effort) activity. DETER cannot clearly differentiate intent in this case and just issues the alarm, leaving the interpretation to the guard. Any pedestrian with average speed over 3 mph raises an alarm.

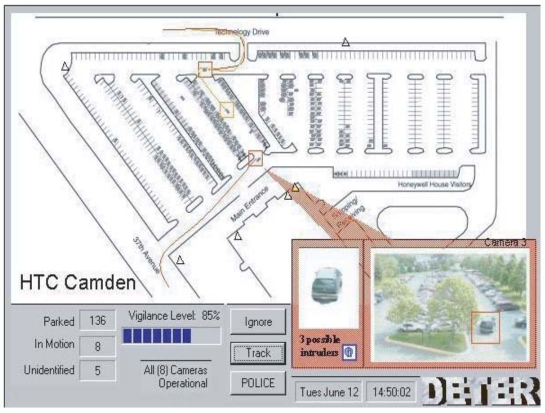


Fig. 12. Live capture of a staged synchronous vehicle attack. The picture also depicts the overall graphical user interface of DETER that is used to convey the threats perceived by the CART classifier

M pattern. This refers to pedestrians following a serpentine course. It is more often than not a suspicious pattern associated with vehicle break-ins (see Fig. 11).

Multivehicle attack. Vehicles entering simultaneously from different entries in the parking lot and converging on a common parking lot lane. This is a pattern often used by gangs to block access to a parking lot lane while a coconspirator is breaking into the in-between vehicles. Figure 12 shows the capture of such a staged event by three vehicles. The figure also shows the panoramic CAD picture where all seven cameras are calibrated against.

Time zones. Movement of pedestrians and vehicles at certain times and days. Specifically, we have set as the time zone the interval from 12:30 am – 5:30 am every day.

Hot zones. Movement of pedestrians and vehicles at certain designated areas within the perimeter of the building. We call these areas hot zones. In our case, we have designated as hot zone the vicinity to the building's air intakes. (Actually, after 9/11 the facilities management of our building bought a product-grade version of DETER to monitor movements around the air intakes.) The product version of DETER runs

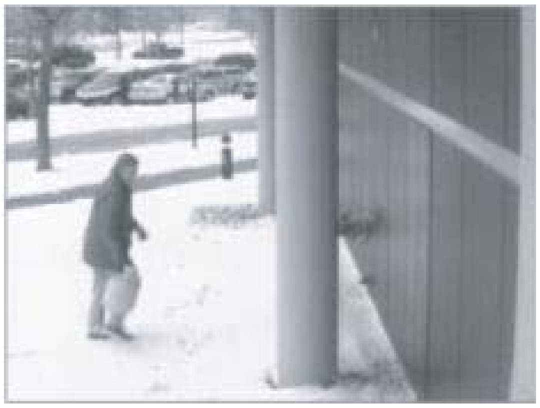


Fig. 13. Snapshot of a staged chemical attack as recorded automatically by the productized version of DETER. The suspect is carrying a gas tank near the air intakes. DETER does not clearly understand the person's intent. It simply detects an individual close to a hot zone and relays the information automatically to the guard. It is up to the guard to further interpret the information and act upon it if necessary

independently of the R&D parking lot version for security reasons. Figure 13 displays a snapshot of a staged chemical attack as it was recorded automatically by the productized DETER.
Combinations. Combinations of the above scenarios. A byproduct of the threat assessment module is a number of ancillary statistical functions. For example, the system is aware of the capacity and spatial utilization of the parking lot (Fig. 12). This is a direct result of the mapping of the fused camera coordinate system to the world (CAD) coordinate system and the ability of the system to track moving objects and differentiate between vehicles and people. When the motion of a vehicle ends in one of the empty parking stalls, this stall is considered occupied. The stall is labeled as empty again when a vehicle motion is detected out of its area.

6 Experimental results

As of this writing (April 2002) the DETER system has been operating for almost 2 years. During this time incremental improvements have been made at the algorithmic and software levels. We use the experience of the building's guards as the primary feedback mechanism. This feedback is mostly qualitative but is very important since this is how products are evaluated in the security marketplace.

In addition to the qualitative testing performed by the actual users, we also performed quantitative testing for benchmarking purposes. We ran our latest benchmark on a data set of 16-hr video data. The video data were recorded in half-hour increments over a period of 6 months (December 2001 – April 2002). The recording was simultaneous from all seven parking lot cameras. We selected this data set to fulfill certain requirements.

1. Sizeable duration (several hours)
2. Scenarios with significant traffic and others predominantly inactive. Typical busy times that were captured were in the morning when people arrive at work and in mid-afternoon when they leave for the day. Typical inactive times were latenight hours and weekends
3. Inclusion of some alarming events. We have staged most of these events ourselves in the absence of significant criminal or threatening activity during the performance period.
4. Challenging weather conditions. We have included a day with strong winds (0.5 hr), where the swaying of tree branches posed a strong challenge to the system. We have included two cloudy days (1.0 hr), three snowy days (1.5 hrs), and four rainy days (2.0 hrs)

We have measured two aspects of DETER's performance: quality of trajectory tracking and classification of threats. The first relates to the performance of the computer vision module, while the second relates to the performance of the threat assessment module. Obviously the performance of the threat classifier partially depends on the performance of the object tracker. Table 2 shows the results of DETER's tracking performance. The ground truth was done by indexing back the actual events on the video clip to the output of DETER. Parking lot activity included walking and running of a single individual, simultaneous walking of a number of individuals (following crossing or parallel paths), driving of a single and multiple vehicles, and finally a combination of vehicles and humans in motion.

DETER perfectly detected and tracked all vehicle tracks. It also detected and tracked 768 out of 800 pedestrian trajectories (96% success rate). All of the 32 missed pedestrian trajectories were associated with groups of people moving very close to each other (party of two or more). DETER correctly detected and tracked the motion but as a single object. This is partially a camera resolution problem and partially a problem with our blob analysis algorithm. If we had covered less area with each camera, the resolution would have been better and the segmentation of closely spaced moving objects more accurate. Also, our blob analysis algorithm tended to compound the problem by merging blobs that appeared to be very close to each other. The loss of detailed tracking information in multipeople groups is not important to DETER's current threat assessment function. This loss of information would have been important only if we had been interested in monitoring human interaction.

DETER also produced a small number of false alarms. Twenty of these false alarms were produced on the snowy days included in the benchmarking data set. The culprit was accumulated iced snow that was hovering from the top covers of some of the cameras. The remaining twelve false alarms were produced by the shadows of pedestrians in exceptionally bright days. In the case of "thick" shadows, it appears that DETER multiplies one real pedestrian trajectory to many (real and shadow). This is the opposite of the problem we discussed earlier in which many pedestrian trajectories are reduced to one (group of people). Both of these problems do not have any bearing on DETER's current threat assessment capabilities. Nevertheless, we consider the shadow problem a weak point in our algorithm and are working toward its solution.

Table 2. Breakdown of tracking results for the 16-hr-long data set

Perfect Vehicle Tracks	Perfect People Tracks	False Alarms	Missed Vehicle Tracks	Missed People Tracks
576	768	32	0	32

Table 3. Breakdown of classification results per threatening activity

	Overspeeding Vehicles	Overrun of Pedestrian Pavement	Running Pedestrians	M Pattern	Multivehicle Attack	Time Zones	Hot Zones
Number of Events	16	2	10	2	2	45	54
Correct Classification	100%	100%	100%	100%	100%	100%	100%

Our data set included several staged and nonstaged threatening events. Table 3 shows the classification results for all the threatening events (staged and not) that were included in the benchmarking data set. The CART classifier earned a perfect score thanks in no small part to the excellent quality annotated trajectories provided by the computer vision module. In the case of overspeeding vehicles, all but one of the events were staged. In the case of overruning a pedestrian pavement, all the events were staged. In the case of running pedestrians, all but two of the events were staged. The M and multivehicle attack patterns were all staged. In contrast, all the events in the time zones occurred naturally. The events in the hot zone (air-intake area) were recorded by the productized version of DETER and do not belong to the 16-hr benchmarking data set. Half of these events occurred naturally and the other half were staged chemical and biological attacks by a single individual. The naturally occurred events had to do with facilities employees passing by for maintenance reasons. The air-intake data were gathered over a period of 3 months (January–March 2002).

We have also used the benchmarking data set to fine-tune two critical parameters of our algorithm: a) the background evidence T and b) the number of Normals in the mixture. The background evidence T is very important because it controls the sensitivity of the algorithm in its perception of foreground motion. Too low a T value and the algorithm becomes highly sensitive, detecting all motion but also triggering many false alarms. Too high a T value and the algorithm is desensitized to foreground motion. As a result, the number of missed detections increases, but at the same time the number of false alarms decreases. Therefore, the choice of a value for background evidence T is a trade-off between high false alarm rate and high missed detection rate.

We have measured the performance of DETER on the benchmarking data for different values of T starting with $T = 0.60$. Figure 14 shows how tracking rate and false alarm rate change with respect to T. The false alarm rate increases dramatically below $T = 0.80$, while at the same time all the tracks are perfectly captured. Above $T = 0.80$ the false alarm rate remains steady and close to zero. At the same time the tracking rate is reduced, particularly for vehicles. As the system becomes slow in its understanding of foreground motion it finds it difficult to perfectly track unparked vehicles that move backwards for a very short time, stop, and then move forward.

To fine-tune the number of Normals n we use in the mixture, we measured the performance of DETER for $n = [2, ..., 5]$. Figure 15 shows the results. Fewer Normals in the mixture provide less representation power but make for a more efficient system. It is evident from the figure that for $n = 2$ we had the highest false alarm rate and the lowest tracking performance. There is a significant increase for $n = 3$, a value that appears to make the system perform almost flawlessly on the experimental data. For higher values of n we notice a slight decrease in the performance of vehicle tracking. By adding more Normals to the mixture we make the system slower and therefore prone to lose track of the fastest moving vehicles in the scene. It appears that $n = 3$ Normals is the golden trade-off for the composition of the mixture. It provides one Normal for capturing foreground objects, one Normal for capturing the dominant background mode, and one more Normal for capturing an alternate background mode. From our experience during the last 2 years and from the measurable experimental results, a mixture of three Normals is more than sufficient to represent the vast majority of naturally occurring foreground-background events.

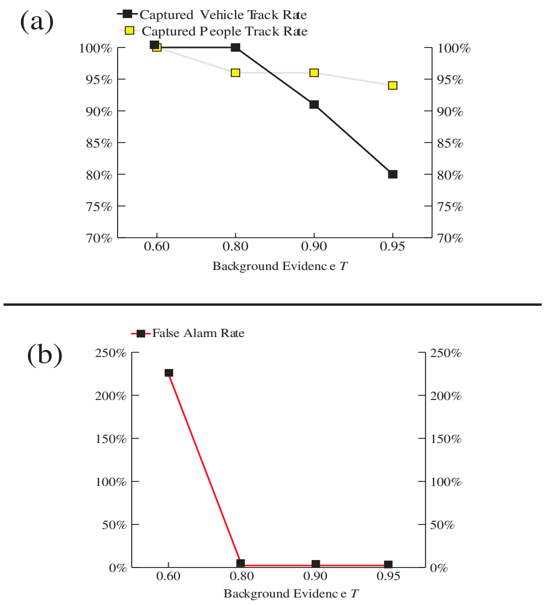
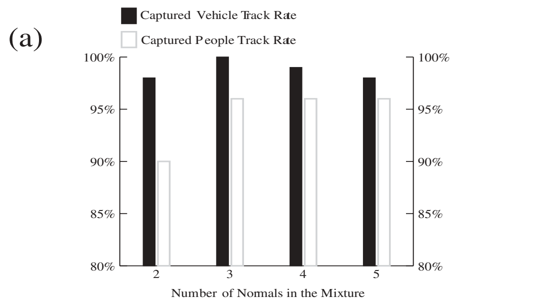


Fig. 14. a Effect of background evidence T on the tracking quality for vehicles and people. **b** Effect of background evidence T on the false alarm rate

7 Conclusions and future work

We have presented DETER, an experimental urban surveillance system for monitoring large open spaces. DETER reliably tracks humans and vehicles both day and night. We have adopted a tracking approach that is based on a multi-Normal mixture representation of the pixel processes and the Jeffreys divergence measure for matching to foreground or background states. This sophisticated matching criterion results in stellar dynamic segmentation performance. Tracks are formed using



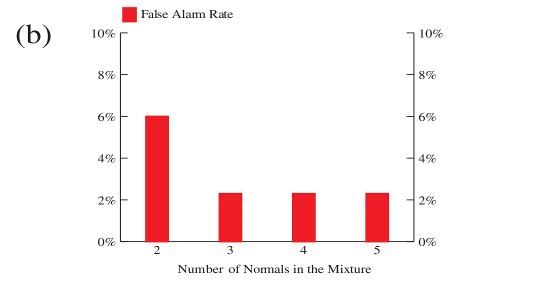


Fig. 15. **a** Effect of number of Normals on the tracking quality for vehicles and people. **b** Effect of number of Normals on the false alarm rate

a multiple hypothesis tracking (MHT) algorithm, and external multicamera calibration is achieved through the computation of homographies. A threat assessment module based on a tree induction algorithm reports suspicious patterns detected in the annotated trajectory data.

Our ongoing work focuses on the issue of shadow detection. Additionally, we are working toward improving the threat assessment module with the inclusion of a clustering algorithm. The clustering algorithm will help in the partial automation of the offline training, which is currently performed manually.

The performance of the prototype DETER system has been very successful. Over an evaluation period of almost 2 years (2000–2002) a dozen actual security officers that used the system gave it high marks and deemed it reliable. This led to the productization of DETER by Honeywell Australia. The first large-scale employment of the DETER product has been across the length of a new oil pipeline in Russia. Hundreds of cameras distributed across hundreds of miles send real-time video information on PC servers that run the DETER software. The primary function is to provide automatic alerts to the central command center about any movement by humans or vehicles anywhere across the oil pipeline.

Acknowledgements. DETER was funded by two major Research Initiative awards from the Honeywell Laboratories. We would like to thank a number of individuals for contributing to the success of this project including Steve Harp, Karen Haigh, Mike Bazakos, Peter Reutiman, and Peter Roeber. DVM, the project that productized DETER, was funded by Honeywell Australia.

References

1. Vsam home page, http://www-2.cs.cmu.edu/ vsam/.
2. Collins RT, Lipton AJ, Kanade T, Fujiyoshi H, Duggins D, Tsim Y, Tolliver D, Enomoto N, Hasegawa O, Burt P, Wixson L (2000) A system for video surveillance and monitoring: Vsam final report. Technical report CMU-RI-TR-00-12, Robotics Institute, Carnegie Mellon University, Pittsburgh
3. Collins RT, Lipton AJ, Kanade T (2000) Introduction to the special section on video surveillance. IEEE Transactions on pattern analysis and machine intelligence, 22(8):745–746
4. Grimson WEL, Stauffer C, Romano R, Lee L (1998) Using adaptive tracking to classify and monitor activities in a site. In: Proceedings of the 1998 IEEE conference on computer vision and pattern recognition, Santa Barbara, CA, 23–25 June 1998, pp 22–29
5. Stauffer C, Grimson WEL (2000) Learning patterns of activity using real-time tracking. IEEE Transactions on pattern analysis and machine intelligence, 22(8):747–757
6. Anderson CH, Burt PJ, Van Der Wal GS (1985) Change detection and tracking using pyramid transform techniques. In: Proceedings of SPIE – the international society for optical engineering, Cambridge, MA, 16–20 September 1985, 579:72–78
7. Haritaoglu I, Harwood D, Davis LS (2000) W/sup 4/s: real-time surveillance of people and their activities. IEEE Transactions on pattern analysis and machine intelligence, 22(8):809–830
8. Kanade T, Collins RT, Lipton AJ (1997) Advances in cooperative multi-sensor video surveillance. In: Proceedings of DARPA image understanding workshop, New Orleans, pp 3–10
9. Stauffer C, Grimson WEL (1999) Adaptive background mixture models for real-time tracking. In: Proceedings of the 1999 IEEE conference on computer vision and pattern recognition, Fort Collins, CO, 23–25 June 1999, 2:246–252
10. Horn BKP (1986) Robot vision. MIT Press, Cambridge, MA, pp 66–69
11. Stein G, Romano R, Lee L (2000) Monitoring activities from multiple video streams: establishing a common coordinate frame. IEEE Transactions on pattern analysis and machine intelligence, 22(8):758–767
12. Dempster AP, Laird NM, Rubin DB (1977) Maximum likelihood from incomplete data via the em algorithm (with discussion). J R Stat Soc B 39:1–38
13. Tsiamyrtzis P (2000) A Bayesian approach to quality control problems. Ph.D. thesis, Minneapolis
14. Ferguson ST (1996) A course in large sample theory. Chapman & Hall, New York
15. Jeffreys H (1948) Theory of probability. University Press, Oxford
16. McLachlan GJ, Basford KE (1988) Mixture models inference and applications to clustering. Marcel Dekker, New York
17. Blackman SS (1986) Multiple-target tracking with radar applications. Artech House, Norwood, MA
18. Murty KG (1968) An algorithm for ranking all the assignments in order of increasing cost. Oper Res 16:682-687
19. Reid DB (1979) An algorithm for tracking multiple targets. IEEE transactions on automatic control, AC-24:843–854
20. Cox IJ, Hingorani SL (1996) An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. IEEE Transactions on pattern analysis and machine intelligence, 18:138–150
21. Kanatani K (1998) Optimal homography computation with a reliability measure. In: Proceedings of the IAPR workshop on machine vision applications, Makuhari, Chiba, Japan, November 1998, pp 426–429
22. Kanatani K (1996) Statistical optimization for geometric computer vision: theory and practice. Elsevier Science, Amsterdam
23. Buntine W (1992) Learning classification trees. Stat Comput 2(2):63–73

For $k = 0, 1, \ldots$ calculate:

$$z_{ij}^{(k)} = \frac{\pi_i^{(k)} \left(\sigma_i^{(k)}\right)^{-3/2} exp\left\{-\frac{1}{2\left(\sigma_i^{(k)}\right)^2}\left(\mathbf{x}_j - \boldsymbol{\mu}_i^{(k)}\right)'\left(\mathbf{x}_j - \boldsymbol{\mu}_i^{(k)}\right)\right\}}{\sum_{t=1}^{3} \pi_t^{(k)} \left(\sigma_t^{(k)}\right)^{-3/2} exp\left\{-\frac{1}{2\left(\sigma_t^{(k)}\right)^2}\left(\mathbf{x}_j - \boldsymbol{\mu}_t^{(k)}\right)'\left(\mathbf{x}_j - \boldsymbol{\mu}_t^{(k)}\right)\right\}},$$

$$\pi_i^{(k+1)} = \frac{1}{N}\sum_{j=1}^{N} z_{ij}^{(k)},$$

$$\boldsymbol{\mu}_i^{(k+1)} = \frac{1}{N\pi_i^{(k+1)}}\sum_{j=1}^{N} z_{ij}^{(k)}\mathbf{x}_j,$$

$$\left(\sigma_i^{(k+1)}\right)^2 = \frac{1}{3N\pi_i^{(k+1)}}\sum_{j=1}^{N} z_{ij}^{(k)}\left(\mathbf{x}_j - \boldsymbol{\mu}_i^{(k+1)}\right)'\left(\mathbf{x}_j - \boldsymbol{\mu}_i^{(k+1)}\right),$$

for $i = 1, \ldots, 3$ and $j = 1, \ldots, N$. Then, set k=k+1 and repeat the loop.

Fig. 16. Optimization loop for the EM algorithm

Appendix I

The EM loop for initializing the mixture of Normals in the scene pixels is shown in Fig. 16. The condition for terminating the loop is $|\pi_i^{(k+1)} - \pi_i^{(k)}| < \varepsilon, \quad i = 1, \ldots, 3$ where ε is a "small" positive number (10^{-2}). $z_{ij}^{(k)}$ are the posterior probabilities that $\mathbf{x}_j$ belongs to the ith distribution and form a $3 \times N$ matrix at the kth step of the computation.

Appendix II

Lemma 2. *: If $f_0 \sim N_3(\boldsymbol{\mu}_0, \sigma_0^2 I)$ and $f_1 \sim N_3(\boldsymbol{\mu}_1, \sigma_1^2 I)$ then:*

$$J(f_0, f_1) = \frac{3}{2}\left(\frac{\sigma_o}{\sigma_1} - \frac{\sigma_1}{\sigma_0}\right)^2 + \frac{1}{2}\left(\frac{1}{\sigma_0^2} + \frac{1}{\sigma_1^2}\right)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)'(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)$$

Proof: We will start by calculating first $K(f_0, f_1)$. Since $log\,(f_0/f_1) =$

$$= log\left(\frac{\frac{1}{(2\pi\sigma_0^2)^{3/2}} exp\left\{-\frac{1}{2\sigma_0^2}(\mathbf{x} - \boldsymbol{\mu}_0)'(\mathbf{x} - \boldsymbol{\mu}_0)\right\}}{\frac{1}{(2\pi\sigma_1^2)^{3/2}} exp\left\{-\frac{1}{2\sigma_1^2}(\mathbf{x} - \boldsymbol{\mu}_1)'(\mathbf{x} - \boldsymbol{\mu}_1)\right\}}\right)$$
$$= 3log\left(\frac{\sigma_1}{\sigma_0}\right) + \frac{1}{2\sigma_1^2}(\mathbf{x} - \boldsymbol{\mu}_1)'(\mathbf{x} - \boldsymbol{\mu}_1) - \frac{1}{2\sigma_0^2}(\mathbf{x} - \boldsymbol{\mu}_0)'(\mathbf{x} - \boldsymbol{\mu}_0)$$

we have that

$$K(f_0, f_1) = E_{f_0}\left[log\left(\frac{f_0}{f_1}\right)\right] = 3log\left(\frac{\sigma_1}{\sigma_0}\right) + \frac{1}{2\sigma_1^2}E_{f_0}\left[(\mathbf{x} - \boldsymbol{\mu}_1)'(\mathbf{x} - \boldsymbol{\mu}_1)\right] - \frac{1}{2\sigma_0^2}E_{f_0}\left[(\mathbf{x} - \boldsymbol{\mu}_0)'(\mathbf{x} - \boldsymbol{\mu}_0)\right]$$

but

$$(\mathbf{x} - \boldsymbol{\mu}_1)'(\mathbf{x} - \boldsymbol{\mu}_1) = (x^R - \mu_1^R)^2 + (x^G - \mu_1^G)^2 + (x^B - \mu_1^B)^2,$$

so $E_{f_0}\left[(\mathbf{x} - \boldsymbol{\mu}_1)'(\mathbf{x} - \boldsymbol{\mu}_1)\right] =$

$$\begin{aligned}
&= E_{f_0}[(x^R - \mu_1^R)^2 + (x^G - \mu_1^G)^2 + (x^B - \mu_1^B)^2] \\
&= \sigma_0^2 + (\mu_0^R)^2 - 2\mu_1^R\mu_0^R + (\mu_1^R)^2 \\
&\quad + \sigma_0^2 + (\mu_0^G)^2 - 2\mu_1^G\mu_0^G + (\mu_1^G)^2 \\
&\quad + \sigma_0^2 + (\mu_0^B)^2 - 2\mu_1^B\mu_0^B + (\mu_1^B)^2 \\
&= 3\sigma_0^2 + (\mu_0^R - \mu_1^R)^2 + (\mu_0^G - \mu_1^G)^2 + (\mu_0^B - \mu_1^B)^2 \\
&= 3\sigma_0^2 + (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)'(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)
\end{aligned}$$

and $E_{f_0}\left[(\mathbf{x} - \boldsymbol{\mu}_0)'(\mathbf{x} - \boldsymbol{\mu}_0)\right] = 3\sigma_0^2$.
Therefore, for the KL information number $K(f_0, f_1)$ we will have $K(f_0, f_1) =$

$$3log\left(\frac{\sigma_1}{\sigma_0}\right) - \frac{3}{2}\left(1 - \frac{\sigma_0^2}{\sigma_1^2}\right) + \frac{1}{2\sigma_1^2}(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)'(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)$$

and similarly for the $K(f_1, f_0)$ we have: $K(f_1, f_0) =$

$$3log\left(\frac{\sigma_0}{\sigma_1}\right) - \frac{3}{2}\left(1 - \frac{\sigma_1^2}{\sigma_0^2}\right) + \frac{1}{2\sigma_0^2}(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)'(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0).$$

Thus, $J(f_0, f_1) =$

$$\begin{aligned}
&= K(f_0, f_1) + K(f_1, f_0) \\
&= \tfrac{3}{2}\left(\tfrac{\sigma_0^2}{\sigma_1^2} - 2 + \tfrac{\sigma_1^2}{\sigma_0^2}\right) + \tfrac{1}{2}\left(\tfrac{1}{\sigma_0^2} + \tfrac{1}{\sigma_1^2}\right)(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0)'(\boldsymbol{\mu}_1 - \boldsymbol{\mu}_0) \\
&= \tfrac{3}{2}\left(\tfrac{\sigma_o}{\sigma_1} - \tfrac{\sigma_1}{\sigma_0}\right)^2 + \tfrac{1}{2}\left(\tfrac{1}{\sigma_0^2} + \tfrac{1}{\sigma_1^2}\right)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)'(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1).
\end{aligned}$$

Dr. Morellas received his education at the National Technical University of Athens, Greece, Columbia University, New York, and the University of Minnesota, Minneapolis obtaining a Diploma, M.S.M.E., and Ph.D. respectively all in Mechanical Engineering. He is currently a senior principal research scientist at the Situation Assessment Technologies Laboratory in Honeywell's Automation and Control Systems business unit. His general research interests are in the areas of sensor integration and learning theories as they apply to system automation and machine intelligence. His current research focus is on the development of high-end video–based security and surveillance systems using advanced statistical imaging techniques. Currently, he is the Principal Investigator of a number of FAA and internally funded projects to enhance airport security through digital video integration. Prior to his tenure at Honeywell, he worked on Intelligent Transportation Systems research that led to the development of unique collision avoidance methodologies for improving vehicle safety. He is a member of the IEEE.

Dr. Pavlidis holds a Ph.D. and a M.S. degree in Computer Science from the University of Minnesota, a M.S. degree in Robotics from the Imperial College of the University of London, and a B.S. degree in Electrical Engineering from the Democritus University in Greece. He joined the Honeywell Laboratories immediately upon his graduation in January 1997. His expertise is in the areas of *Computer Vision Beyond the Visible Spectrum*, *Pattern Recognition of Highly Variable Patterns, and Software Engineering*. Dr. Pavlidis published extensively in these areas in major journals and refereed conference proceedings over the past several years. His publication record includes articles in the IEEE Proceedings, *New England Journal of Medicine, The Lancet,* and *Nature*. He is also the author of the book "Programming of Cameras and Pan-Tilts with DirectX and Java" by Morgan Kaufmann. Dr. Pavlidis is the Chair of the 2003 IEEE Conference in Advanced Video and Signal Based Surveillance, and has been the Chair of several other major IEEE Conferences in the past. He is also Associate Editor of the Pattern Analysis and Applications journal and serves as a Guest Editor for the Machine Vision and Applications as well as the Image and Vision Computing journals. Dr. Pavlidis' research work was cited extensively in the scientific literature and has been mentioned frequently by the international media. Dr. Pavlidis is a Fulbright Fellow, a Senior Member of IEEE, and a member of ACM. Dr. Pavlidis joined the Computer Science Department of the University of Houston in September 2002, where currently holds the position of Associate Professor.

Dr. Panagiotis Tsiamyrtzis received the B.S. degree in Mathematics from the Aristotle University of Thessaloniki, Greece and the M.Sc. and Ph.D. degrees in Statistics from the University of Minnesota. He served as a faculty member in the School of Statistics of the University of Minnesota in Fall 2000. He is currently a visiting lecturer at the Department of Statistics, Athens University of Economics and Business. His expertise is in the area of Quality Control and applications of the Bayesian Statistics. Dr. Tsiamyrtzis was the recipient of the best student paper award in 2000 from the Risk Analysis section of the American Statistical Association.