CHAPTER

Surveillance

22

Surveillance is nowadays used very widely in transport and civil centers for monitoring traffic and people and is increasingly being carried out by computer. The motivation is largely to locate instances of undesirable behavior—theft, loitering with intent, speeding, and so on. What is special about surveillance is the rate at which pictorial information is delivered, and the fact that many of the objects being monitored are in motion. To cope with this, there is considerable emphasis on identification and elimination of the background and effective tracking of moving objects. At the same time, algorithms need to be fast in operation, though some help can be obtained with fast-dedicated hardware systems.

Look out for:

- The geometry of surveillance
- The need to separate foreground from background
- The basics of particle filters and their use for tracking
- The use of color histograms for tracking
- Chamfer matching and its use for identification and tracking
- How multiple cameras are used to obtain coverage over wide areas
- Systems for monitoring traffic flow
- Identification of the ground plane as an early stage in the analysis of many types of scene incorporating motion
- The need for "occlusion reasoning" when objects repeatedly pass behind one another and then reemerge
- The importance of the Kalman filter in motion applications
- License plate location
- How studies of the motions of complex objects may have to take into account 3D articulated models of linked parts
- Basic concepts of human gait analysis
- Animal tracking.

While this chapter covers the situation of static cameras being used to monitor moving objects, the following chapter covers the more complex case of in-vehicle vision systems, where moving cameras are used to monitor both stationary and moving objects.

"T is Cinna; I do know him by his gait" (William Shakespeare, 1599)

Computer Vision. DOI: http://dx.doi.org/10.1016/B978-0-12-809284-2.00022-8

22.1 INTRODUCTION

Visual surveillance is a long-standing area of computer vision, and one of its main early uses was to obtain information on military activities—whether from high flying aircraft or from satellites. However, with the advent of ever cheaper video cameras, it subsequently became widely used for monitoring road traffic, and most recently it has become ubiquitous for monitoring pedestrians. In fact, its application has actually become much wider than this, the aim being to locate criminals or people acting suspiciously—for example, wandering around car-parks with the potential purpose of theft. However, by far, the majority of visual surveillance cameras are connected to video recorders and gather miles of video-tape, most of which will never be looked at—though, following criminal or other activity, some hours of videotape may be scanned for relevant events. Further cameras will be attached to closed circuit television monitors where human operators may be able to extract some fraction of the events displayed, though human attentiveness and reliability when overseeing a dozen or so screens will not be high. Clearly, it would be far better if video cameras could be connected to automatic computer vision monitoring systems, which would call human operators' attention to potential hazards or misdemeanors of various types. Even if this were not carried out in real time in specific applications, it would be useful if it could be achieved at high speed with selected videotapes: This could save huge amounts of police time in locating and identifying perpetrators of crime.

Surveillance can cover other useful activities, including riot control, monitoring of crowds on football pitches, checking for overcrowding on underground stations, and generally helping with safety as well as crime. To some extent, human privacy must suffer when surveillance is called into play, and there is clearly a tradeoff between privacy and security: Suffice it to say here that many would be happier to have increased levels of security, a small loss of privacy being a welcome price to pay to achieve it.

In fact, there are many difficulties to be solved before the "people tracking" aspects of surveillance are fully solved. First, in comparison to cars, people are articulated objects that change shape markedly as they move: That their motion is often largely periodic can help visual analysis, though the irregularities in human motion may be considerable—especially if obstacles have to be avoided. Second, human motions are partly self-occluding, one leg regularly disappearing behind another, while arms can similarly disappear from view. Third, people vary in size and apparent shape, having a variety of clothes that can disguise their outlines. Fourth, when pedestrians are observed on a pavement, or on the underground, there is some possibility of losing track when one person passes behind another, as the two outlines tend to coalesce before reemerging from the combined object shape.

It could be said that all these problems have been solved. However, many of the algorithms that have been applied to these tasks have limited intelligence: Indeed, some employ rather simplified algorithms, as the need to operate continuously in real time generally overrides the need for absolute accuracy. In any case, given the visual data that the computer actually receives, it is doubtful whether a human operator could always guarantee making correct interpretations: For example, there are occasions when humans turn around in their tracks because they have forgotten something, and this could cause confusion when trying to track every person in a complex scene. Further complexities can be caused by varying illumination, fixed shadows from buildings, moving shadows from clouds or vehicles, and so on.

In the following sections, we cover two main areas of surveillance—those in which people or pedestrians are the prime targets, and those in which vehicles are the prime targets. Of course, there are many transport scenarios where both would be observed by the same systems. In addition, similar techniques and considerations would apply in both cases. The next section, on the geometry underlying camera positioning, talks mainly about pedestrians: This is done for definiteness, though most of the considerations apply equally when vehicles are the prime targets, as happens on motorways, for example.

22.2 SURVEILLANCE—THE BASIC GEOMETRY

Perhaps the most obvious way of monitoring pedestrians is indicated in Fig. 22.1A. As we have seen in Chapter 16, The Three-Dimensional World, this leads to the following relations between real-world (X, Y, Z) and image coordinates (x, y):

$$x = fX/Z \tag{22.1}$$

$$y = fY/Z \tag{22.2}$$

Here, Z is the (horizontal) depth in the scene, X represents lateral position, Y represents vertical position (downward from the camera axis), and f is the focal length of the camera lens. This method of observation is useful in providing undistorted profiles of pedestrians from which they may be recognized. However, it provides virtually no information about depth in the scene beyond what can be deduced from knowledge of the pedestrian's size; and as size may be one of the key parameters to be determined by the vision system, this is an unsatisfactory situation. Note also that this view of the scene is subject to gross occlusion of one pedestrian by another.

To overcome these problems, an overhead view would be better. However, it is difficult to obtain views from directly overhead; in any case, any one view would give a highly restricted range, and again pedestrian height could not be measured. An alternative approach is to place the camera in Fig. 22.1A higher up, as shown in Fig. 22.1B, so that the positions of the feet of any pedestrian on the ground plane can be seen: This makes it possible to obtain a reasonable estimate of depth in the scene. In fact, if the camera is at a height H_c above the ground plane, Eq. (22.2) gives the depth Z as

$$Z = fH_c/y \tag{22.3}$$

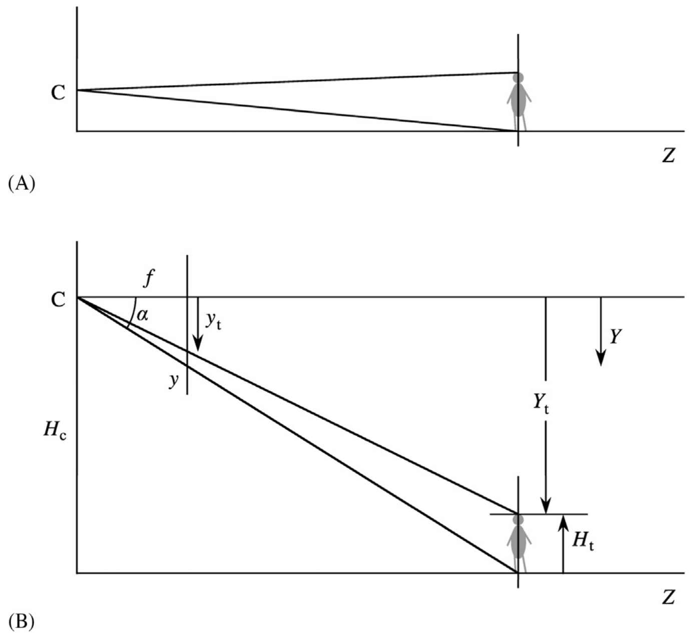

FIGURE 22.1

3D monitoring: camera axis horizontal. (A) Camera axis mounted at eye level. (B) Camera mounted higher up to obtain a less restricted view.

while the modified value of y at the top of the pedestrian is given by

$$y_t = fY_t/Z = yY_t/H_c \tag{22.4}$$

The height of the pedestrian H_t can now be estimated from the following equation:

$$H_t = H_c - Y_t = H_c(1 - y_t/y) \tag{22.5}$$

Notice that to achieve this, H_c must be known from prior on-site measurements, or alternatively by camera calibration using test objects.

In practice, it is better to modify the above scheme by tilting the optical axis of the camera slightly downward (see Fig. 22.2), as this allows the range of observation to be increased, and particularly for nearby pedestrians to be kept in view. However, the geometry of the situation becomes somewhat more complicated, leading to the following basic formulae:

$$\tan \alpha = H_c/Z \tag{22.6}$$

$$\tan(\alpha - \delta) = y/f \tag{22.7}$$

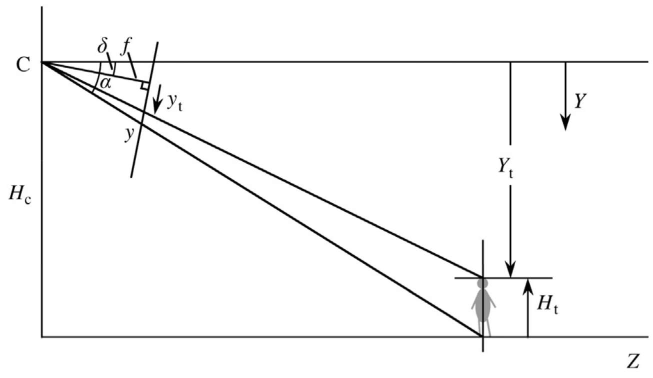

FIGURE 22.2

3D monitoring: camera tilted downward. δ is the angle of declination of the camera optical axis.

where δ is the angle of declination of the camera. Substituting for tan $(\alpha - \delta)$ using the formula:

$$\tan(\alpha - \delta) = (\tan\alpha - \tan\delta)/(1 + \tan\alpha\tan\delta) \tag{22.8}$$

and using the above equations to eliminate α, we obtain the following formula for Z in terms of y:

$$Z = H_c(f - y\tan\delta)/(y + f\tan\delta) \tag{22.9}$$

So far, we have not allowed for the heights of any objects but have only considered points on the ground plane. To estimate the heights of pedestrians, we need to bring in the additional equation:

$$Z = Y_t(f - y_t\tan\delta)/(y_t + f\tan\delta) \tag{22.10}$$

which is simply derived by substituting Y_t for H_c and y_t for y in Eq. (22.9). Eliminating Z between these two equations now allows us to find Y_t:

$$Y_t = H_c(f - y\tan\delta)(y_t + f\tan\delta)/(y + f\tan\delta)(f - y_t\tan\delta) \tag{22.11}$$

thereby permitting $H_t = H_c - Y_t$ to be calculated in this case too.

Next, we consider the optimum value for the angle of declination δ of the camera optical axis. We assume that the viewing range of the camera has to vary from a near point given by Z_n to a far point given by Z_f, corresponding to respective values of α, α_n, and α_f (Fig. 22.3). We also assume that the overall vertical field of view (FOV) of the camera is 2γ. This immediately results in the following formulae:

$$H_c/Z_n = \tan\alpha_n = \tan(\delta + \gamma) \tag{22.12}$$

$$H_c/Z_f = \tan\alpha_f = \tan(\delta - \gamma) \tag{22.13}$$

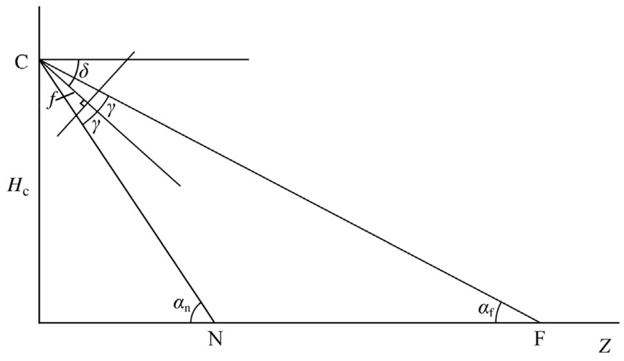

FIGURE 22.3

Geometry for considering optimum camera tilt. δ is the angle of declination of the camera optical axis. 2γ is the overall vertical field of view of the camera.

Taking the ratio between these equations now shows that

$$\eta = Z_n/Z_f = \tan(\delta - \gamma)/\tan(\delta + \gamma) \tag{22.14}$$

so specifying either Z_n or Z_f immediately gives the alternate value. In the case that Z_f is taken to be infinity, Eq. (22.13) shows that δ has to be equal to γ, in which case Eq. (22.12) leads to the relation $Z_n = H_c \cot 2\gamma$. Note that $\delta = \gamma = 45°$ is a limiting case that covers all points on the ground plane, that is, $Z_n = 0$ and $Z_f = \infty$. For smaller values of γ, values of Z_n and Z_f are determined by δ: e.g., for $\gamma = 30°$, the optimum value of η (namely, 0) occurs both at $\delta = 30°$ and at $\delta = 60°$, and the worst case ($\eta \approx 0.072$) occurs at $\delta = 45°$.

Finally, it is instructive to consider the minimum separation Z_s that is needed between pedestrians if they are not to occlude each other at all. By equating $\tan\alpha$ to both H_t/Z_s and H_c/Z [see Eq. (22.6)], we find:

$$Z_s = H_t Z/H_c \tag{22.15}$$

As might have been expected, this varies inversely with camera height, but note that it is also proportional to Z.

Overall, we have seen that placing the camera high up permits both depth and height to be estimated and the incidence of occlusion to be considerably reduced. In addition, tilting the camera downward permits the maximum range to be achieved. Importantly, two cameras placed at the far ends of a courtyard should be able to cover it quite well. Pedestrians can be identified as having a particular position on the ground plane, though they could then be recognized pictorially from knowledge of their size, shape, and coloring. The formulae that are involved reflect all the complications of perspective projection, and some are quite complex. Notice that even in the simple case of Fig. 22.1B, the inverse relation between y and Z is highly nonlinear [see Eq. (22.3)], and equal intervals in the Z

direction by no means correspond to equal vertical intervals in the image plane: See Section 18.8 for further theory underpinning this point.

22.3 FOREGROUND—BACKGROUND SEPARATION

One of the first problems of surveillance is to locate the targets that are to be placed under observation. In principle, we could follow all the recognition methods of earlier chapters and just proceed to recognize the targets individually. However, there are two reasons why we should approach this differently. First, cars moving along a road, or pedestrians in a precinct, are highly variegated, unlike the situation for products appearing on a product line. Second, there is usually a significant real-time problem, especially when vehicles are moving at up to 100 mph on a highway, and cameras typically deliver 30 frames per second under highly variable conditions. Thus, it pays to capitalize on the motion of the targets and performs motion-based segmentation.

In these circumstances, it is natural to think of frame differencing and optical flow. Indeed, frame differencing has been applied to this task, but it is prone to noise problems and consequent unreliability. In any case, when applied between adjacent frames, it locates only limited sections of target outlines—in accordance with the $-\nabla I.\mathbf{v}$ formula of Chapter 20, Motion. The simplest way out of this difficulty is that of background modeling.

22.3.1 BACKGROUND MODELING

The idea of background modeling is to create an idealized background image that can be subtracted from any frame to yield the target or foreground image. To achieve this, the simplest strategy is to take a frame when there are known to be no targets present and use that as the background model. In addition, to eliminate noise, it is useful to average a number of frames prior to making observations of targets. The problems with this strategy are (1) how to know when there are no targets present so that frames represent true background and (2) how to cope with the usual outdoor situation of illumination that varies with the weather and the time of day.

To solve the latter problem, only the most recent frames can reasonably be used, and if this path is followed, it is difficult to tackle the former problem (in any case, on highways with a lot of traffic, or precincts with a continuous mêlée of people, there may seldom be a chance of obtaining a clear background frame). One compromise solution is to take an average of many background frames over the most recent period Δt, whether or not targets are present. If targets are reasonably rare, most of the frames will be clear and a good approximation to an ideal background model will be built: Of course, any targets will not be eliminated so much as averaged in, and the result will sometimes be visible "tails" in the model.

To optimize the model, Δt can be increased, thereby minimizing problem (1), or decreased, thereby minimizing problem (2). Clearly, there is a tradeoff between the two difficulties: while it can be adjusted to suit the time of day and prevailing weather and illumination levels, this approach is limited.

Part of the problem is due to the "averaging in" mentioned above, and this can be partially eliminated by using a temporal median filter. Note that this means applying a median filter to the **I** (intensity and color) values of each pixel, over the sequence of frames arising during the most recent period Δt. This is a computation intensive process but is considerably better than taking a raw average, as mentioned above. It is effective to take the median because it eliminates outliers, but ultimately it will still lead to biased estimates. In particular, if we suppose that vehicles are on the whole darker than the road, then the temporal median will also tend to end up darker than the road. To overcome this problem, a temporal mode filter can be used, and hopefully, the intensity distribution will have separate modes—one from the road and one from the vehicles, so the former can be used and could probably be identified even if it became a minor mode when there were a lot of vehicles. However, there is no guarantee that there would only be one mode for vehicles, or even that any such modes would be clearly separated from the one corresponding to the road, and bias would again be the most likely result. Figs. 22.4–22.6 in Section 22.3.2 illustrate some of the problems.

In fact, there are significant further problems with background modeling. In many situations, the background objects are themselves subject to motion. In particular, shadows will move with time and their crispness will change with the weather; while leaves, branches of trees, and flags will flutter and sway in the wind, with highly variable frequencies, even the camera may sway, especially if it is mounted on a pole, but we defer that type of problem until Chapter 23, In-Vehicle Vision Systems. Motions of small animals and birds may also have to be considered. At this point, we shall concentrate on fluttering vegetation, which is often prevalent in outdoor scenarios, even within cities.

The fluttering of vegetation can be more serious than might at first be imagined. It can result in the **I** values of some pixels oscillating between those of leaves, branches, and sky (or ground, buildings, etc.). Thus, the distributions of intensities and colors for any pixel may best be regarded as the superposition of several distributions corresponding to two or three component sources. Here, what is important is that each of the component distributions could be quite narrow and well defined. This means that if each is known from ongoing training, any current intensity **I** can be checked to determine whether it is likely to correspond to background. If not, it has to correspond to a new foreground object.

Models formed from multiple component distributions are commonly called mixture models: In practice, the component distributions are approximated by Gaussians, because the odd shape of the overall distributions is largely attributable to the existence of the separate components. Thus, we arrive at the terms Gaussian mixture models (GMMs) and mixtures of Gaussians. Notice that the number of components at any pixel is initially unknown: Indeed, a large

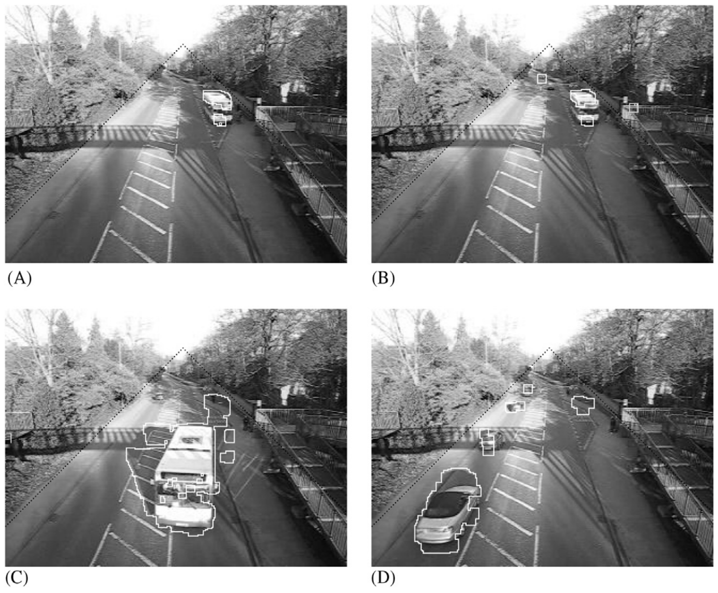

FIGURE 22.4

Background subtraction using a temporal median filter. The lines of black graphics dots demarcate the relevant road region: Almost all of the fluttering vegetation lies outside this region [it is indicated by fainter boundaries than for the foreground objects: see (B)]. Note the plethora of stationary shadows that are completely eliminated during the process of background subtraction. The stationary bus is progressively eaten away in (A) and (B), while in (C) and (D), the ghost of the bus appears and then starts to merge back into the background. These problems are largely eliminated in Fig. 22.5, which includes the same four frames.

proportion of pixels will have only a single component, and it may seem unlikely that the number would be much larger than three in practice. However, the fact that every pixel will have to be analyzed to determine its GMM is computationally burdensome, while the analysis can be unstable if the component distributions are not as tidy as suggested above. These factors mean that a computation intensive algorithm, the expectation maximization (EM) algorithm, has to be used to analyze the situation. In fact, while it is usual to use this rigorous approach to *initialize* the background generation process, many workers use simpler more efficient techniques for updating it so that the ongoing process can proceed in real time. The GMM method determines for itself the number of component distributions to use, the judgment being based on a threshold value for the fraction of the total weight given to the background model.

Unfortunately, the GMM approach fails when the background has very high frequency variations. Essentially, this is because the algorithm has to cope with

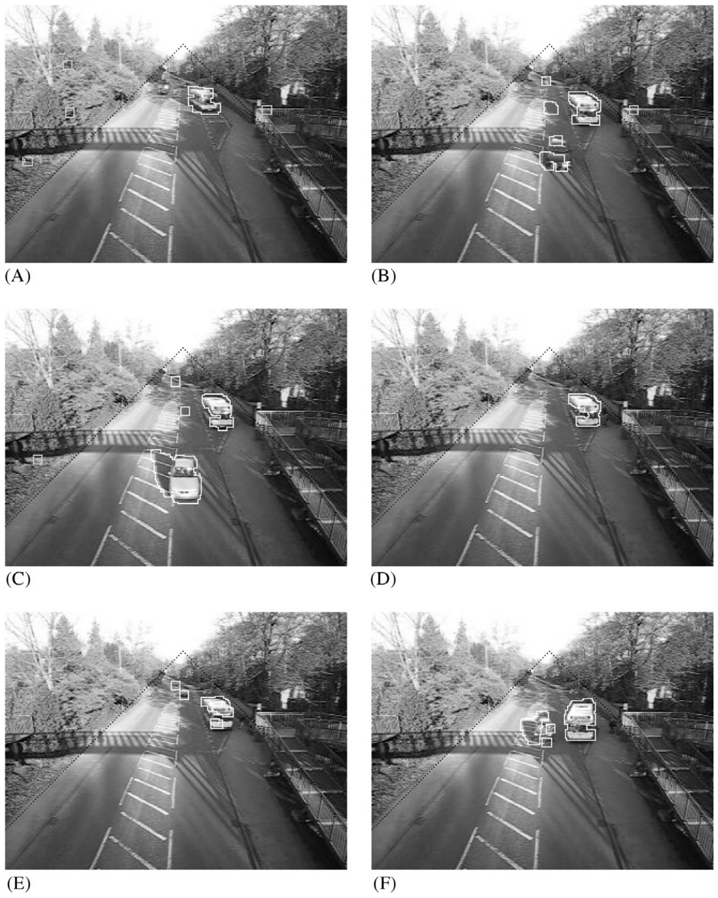

FIGURE 22.5

Background subtraction using a restrained temporal median filter. This figure shows a much more comprehensive set of frames than Fig. 22.4 because the method is more accurate. In particular, its responses (D, E, G, H) to the bus problems of Fig. 22.4 are vastly improved. Fluttering vegetation problems are indicated by fainter boundaries than for vehicles but are entirely absent from the road region. In all frames, the stationary shadows are completely eliminated by background subtraction: even the prominent bridge shadow is ignored; neither does it have much effect on the integrity of foreground objects. Note the low false-negative rate for vehicles, and the fact that they only tend to be joined together in the distance. Overall, foreground object fragmentation and false shapes (including the effects of moving shadows) are the worst problems.

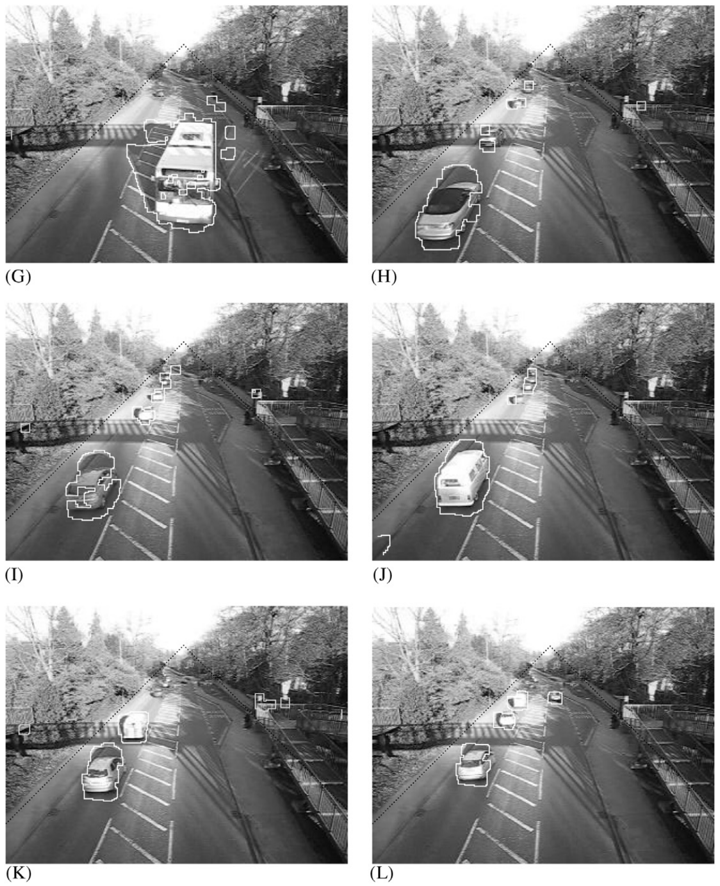

FIGURE 22.5

(continued)

rapidly varying distributions, which change dramatically over very short periods of time, so the statistics become too poorly defined. To tackle this problem, Elgammal et al. (2000) moved away from the parametric approach of the GMM (the latter essentially finds the weights and variances of the component distributions, and thus is parametric). Their nonparametric method involves taking a kernel smoothing function (typically a Gaussian) and for each pixel, applying it to the N samples of $\mathbf{I}$ for frames appearing during the period Δt prior to the current time t. This approach is able to rapidly adapt to jumps from one intensity value to another, while at the same time obtaining the local variances at each pixel. Thus, its value lies in its capability to quickly forget old intensities and to reflect local variances rather than random intensity jumps. In addition, it is a probabilistic approach but has no need of the EM algorithm, and this enables it to run highly

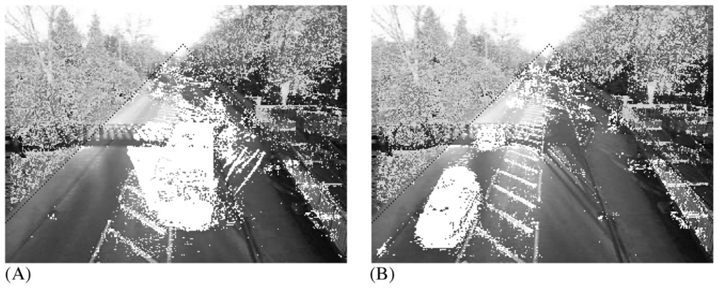

FIGURE 22.6

Problems arising immediately after background subtraction. These two frames show clearly the noise problems that arise during background subtraction: the white pixels indicate where the current image fails to closely match the background model. Most of the noise effects occur for fluttering vegetation outside the road region. Morphological operations (see text) are used to largely eliminate the noise and to integrate the vehicle shapes as far as possible, as shown in Figs. 22.4 and 22.5.

efficiently in real time. In addition, it is capable of sensitive detection of foreground objects coupled with low false alarm rates. To achieve all this, it incorporates two further features:

1. It assumes independence between three different color channels, each having its own kernel bandwidth (variance). Together with the adoption of a Gaussian kernel function, this leads to a probability estimate given by

$$P(\mathbf{I}) = \frac{1}{N}\sum_{i=1}^{N}\prod_{j=1}^{C}\frac{1}{\left(2\pi\sigma_j^2\right)^{1/2}}e^{-\left(I_j - I_{j,i}\right)^2/2\sigma_j^2} \tag{22.16}$$

   where i runs over the N samples taken in the time period Δt and j runs over the C color channels; this function is simple to calculate, though computation is further speeded up by use of precalculated kernel function lookup tables.
2. It uses chromaticity coordinates for suppressing shadows. As these coordinates are independent of the level of illumination, and shadows can be regarded as poorly lit background, this means that they should largely merge back into the background. Hence, the foreground is much less likely to have shadows accompanying it after background subtraction. The chromaticity coordinates r, g, b are derived from the usual R, G, B coordinates by the equations $r = R/(R + G + B)$, etc., with $r + g + b = 1$.

In fact, shadows can be particularly problematic: Not only do they distort the apparent shapes of foreground objects after background subtraction, but also they can connect separate foreground objects, and thus cause undersegmentation. The problem is reviewed by Prati et al. (2003), while Xu et al. (2005) have proposed a hybrid shadow removal method that makes use of morphology: See also Guan (2010).

Whatever method is used for background modeling leading to background subtraction and hence to foreground detection, the various blobs will need to be clustered and labeled using connected components analysis. Frame-to-frame tracking is then carried out by making correspondences between the blobs in the different frames. As in the case of Xu et al. (2005), morphology can be used to help with this process. Nevertheless, false positives tend to arise because of shadows and illumination effects, while false negatives can arise from color similarities between foreground and background.

Overall, failures arise in two categories: (1) *the stationary background problem*, in which the shape of the foreground object is not defined accurately enough; and (2) *the transient background problem*, in which the start and stop of the foreground object aren't found quickly enough. If the accuracy or reactivity of the background model is inadequate, background subtraction will lead to the detection of false objects: These are called "ghosts" by Cucchiara et al. (2003). In addition, as indicated above, shadows tend to compound these problems.

22.3.2 PRACTICAL EXAMPLES OF BACKGROUND MODELING

To add concreteness to the above discussion, a traffic surveillance video was taken and submitted to some of the algorithms mentioned above. For illustrative purposes, the algorithms were kept as simple as possible. The raw data consisted of an AVI video from a digital camera (Canon Ixus 850 IS), which was decompiled into individual JPG frames, and though the JPG artifacts were fairly severe, no specific attempt was made to eliminate them. The frame size was 320×240 pixels in RGB color, but only the 8-bit lightness component was used for the main tests. While the video was taken at 15 frames per second, only every 10th frame was used for the test, which comprised 113 frames. Of these, the first 10 can be regarded as initialization training material and these are not considered further. During the test, a bus arrived and was stationary at a bus stop for some time. The overall sequence is illustrated in Fig. 22.5: However, for reasons of space, the only frames included in the figures are those that illustrate the problems well. Note that the video was taken on a sunny day, and that there are a great many shadows, which over the minute or so of the video did not change markedly. On the other hand, some camera motion is detectable, possibly due to movement of the bridge. Thus, there are many ways in which the raw data were not ideal: These therefore impose exacting conditions on the success of any algorithm.

Fig. 22.4 shows some of the results obtained by applying a temporal median. Parts (A) and (B) show the bus stationary at the bus stop and being progressively eaten up as it starts to merge with the background. Part (C) shows the bus moving away from the bus stop, leaving a large "ghost" behind it. Part (D) shows that the ghost remains for some time and is a substantial factor to be taken into account by any foreground interpretation procedure.

To overcome these problems, the median was restrained so that it could only take into account pixel intensities within a limited number of gray levels of the current median value: In this way, it took on something of the characteristics of a mode filter (a temporal mode filter per se would lock on to the current value too inflexibly and not adapt well to the changing intensity distribution). The results are shown in Fig. 22.5. It is clear that the restrained median largely eliminates the two problems mentioned above (viz., the observed vehicle being eaten away while stationary and leaving a ghost behind it when moving on). For this reason, the remainder of the tests used only the restrained median. Problems that are seen in Fig. 22.5 include the following:

1. Eating away of foreground objects, leaving unusual shapes [e.g., (D), (I)].
2. Fragmentation of foreground objects [e.g., (B), (F)].
3. Shadows accompanying the moving foreground objects [e.g., (C), (G), (J)].
4. Joining of foreground objects that should appear separated [e.g., (I), (J)].
5. Signals from fluttering vegetation [e.g., (A), (K)].

Item 2 can be considered as an extreme case of item 1. Item 3 is bound to arise as the shadows are moving at the same speed as the vehicles that give rise to them, and straightforward background suppression or alternatively moving object detection alone will not eliminate them: in general, unless color interpretation will help (we return to this possibility below), high-level interpretation is needed to achieve satisfactory elimination. Item 4 is due partly to the effects of vehicle shadows, which tend to connect vehicles, especially when seen in the distance. The morphological operations that were applied (see below) also tended to make vehicles become joined. Item 5 is never manifest in the road region, i.e., between the lines of black graphics dots shown in the frames. This is because, in this case, the vegetation is high up, away from the road region. In addition, it is largely eliminated by morphological operations. In fact, Fig. 22.6 shows the results obtained immediately after background subtraction. It is clear that there is a serious noise problem, caused largely by (1) camera noise, (2) the effects of JPG artifacts, (3) fluttering vegetation, and (4) the effects of slight camera motion. Interestingly, two applications of a single pixel erosion operation were sufficient to eliminate the noise almost completely, these being followed by four applications of a single dilation operation to help restore vehicle shapes. (Overall, this corresponds to a 2-pixel opening operation followed by a 2-pixel dilation operation.) These morphological operations were selected to give roughly optimal results—in particular, low probability of failing to capture foreground objects in any individual frame, coupled with pressure to maintain object shapes as far as reasonable, and not to join vehicles together unavoidably. The point is that background subtraction must aim to pass on sufficient useful information to subsequent foreground object identification, tracking, and interpretation stages.

One of the remarkable aspects of the results is the total elimination of stationary shadows and lack of problems arising from this. On the other hand, two other sorts of shadows are manifest—those arising from moving objects, and those falling on moving objects (the latter arise in the video both from the bridge and from the other causes of ground shadows): neither of these sorts of shadows are eliminated. Other problems are those of reflections, particularly from the windows of the bus (see the frame in Fig. 22.5G), and secondary illumination from moving vehicles.

Lastly, it was felt worthwhile to attempt utilizing the original color images and augmenting the background model by using the chromaticity coordinates as outlined in the previous section. In fact, while in some respects improvements occurred, these were more than canceled out by increased numbers of false negatives and fragmented shapes of the foreground objects. No results are shown here, but whereas Elgammal et al. (2000) were able to show excellent results obtained in this way, with the video used here no improvement seemed to be achievable by this approach. This requires some explanation. High up amongst the reasons is the effect of the highly variegated colors and intensities of the many different vehicles. In particular, some vehicles turned out to have body intensities close to those of shadows, whereas others had windows or transparent roofs of similar intensity. These had the effect of eliminating large portions of vehicles together with the shadows, thereby increasing the incidence of false-negative and false-shape information. However, even Elgammal et al. (2000) point out that intensities have to be used carefully in a way that will bolster up the shadow-removing capabilities of the chromaticity information, and here there appeared to be no way this could be achieved. Overall, all the color and grayscale information has to be taken into account in a more considered and strategic way, and this demands a thoroughgoing statistical pattern recognition approach in which objects are identified one by one in a high-level schema rather than by relatively chancy ad hoc methods: the latter definitely have their place, but their use must not be pushed beyond what is reasonable. One example of the use of object-by-object recognition is the identification of road markings, which need to be, and could easily be, identified whether or not vehicle shadows cover them. This could then lead to a much more viable strategy for identifying, tracking, and eliminating vehicle shadows. Meanwhile, in situations where the road has almost no color content, as in the traffic surveillance trials described above, it is difficult to remove shadows effectively merely by using chromaticity information.

22.3.3 DIRECT DETECTION OF THE FOREGROUND

In the previous subsection, it has been seen that background modeling followed by background subtraction constitutes a powerful strategy for the location of moving targets in image sequences. Nevertheless, for all sorts of reasons, it is limited in what it can achieve. While these reasons devolve into problems such as changes in ambient illumination, effects of shadows, irrelevant motions such as fluttering leaves, and color similarities between foreground and background, there is one whole tranche of information that is absent: Specifically, there is a total lack of information on the nature of the target objects, including size, shape, location, orientation, color, speed, and probability of occurrence. If this sort of information were obtainable, there would be some chance of incorporating it into a complete target-detection system and achieving close to perfect detection capability. Indeed, it seems possible that in some cases, ignoring the background and attempting direct detection of the foreground might be a better first approximation. Such a procedure might well be both effective and efficient for the case of face detection, for example. In what follows, we consider how direct foreground detection might be achieved.

Direct foreground detection is only possible if a suitable foreground model is available or can be constructed. It would seem that this requires a specialization to each particular application, such as pedestrian detection or vehicle detection. However, some workers (e.g., Khan and Shah, 2000) have managed to achieve it more generically by a bootstrapping process. They start with background modeling and background subtraction, locate foreground objects by an "exception to background" procedure, and thus create initial foreground models. In subsequent frames, these are enhanced, mostly using Gaussian-based models: GMMs and nonparametric models have been employed for this. However, a difference relative to background modeling is that the latter applies continuously (with updates) for the same camera, whereas each foreground object must have its own individual model which is learnt anew for that object. So background modeling is only applied to initially locate the foreground object: thereafter, the foreground model is built and tracked, albeit in a similar way to what happens with background modeling.

More recently, in a new class of algorithm, Yu et al. (2007) use a GMM for simultaneously modeling both foreground and background. In this way, a tension is built between foreground and background that potentially leads to higher segmentation accuracy, and this does seem to have been achieved in practice. Against this, the algorithm has to be initialized by marking areas of definite foreground and background, and then it continues to track autonomously. However, there seems to be no reason why initialization should not also be carried out autonomously with the help of an initial stage of background modeling.

22.4 PARTICLE FILTERS

When trying to track foreground objects, independent detection in each frame, followed by appropriate linking, does not make best use of available information; neither does it achieve optimum sensitivity: this will be obvious when noting that averaging slowly moving objects over a number of frames can boost signal-to-noise ratio. In addition, over time—and sometimes over very few frames—objects can change radically in appearance, so tracking is needed in order to ensure continued capture. Nowhere is this more obvious than in the case of a guided missile approaching a target over several miles, as during the time of flight, the size, scale, and resolution will increase dramatically. But even in cases where a person is being tracked, rotation of the head will present if anything even more dramatic changes in appearance. With the radically changing backgrounds arising with moving and rotating objects, sensitive robust tracking is clearly of fundamental importance. To achieve this, optimal methods are needed. In particular, in the face of radical change, we need to know what is the most *likely* position of an object that is being tracked. Optimal estimation of likelihood implies the need for Bayesian filtering.

To achieve this, we start by considering the observations $\mathbf{z}_1$ to $\mathbf{z}_k$ of an object in successive frames, and the corresponding deduced states of the object $\mathbf{x}_0$ to $\mathbf{x}_k$ (there is no zeroth value of $\mathbf{z}$ because it takes at least two frames to estimate the velocity $\mathbf{v}_k$, which forms part of the state information). At each stage, we need to estimate the most probable state of the object, and Bayes rule gives us the a posteriori probability density shown in Eq. (22.17) below. (Note that it is much easier to see the relation to Bayes rule if conditional dependence on $\mathbf{z}_{1:\mathrm{k}}$ is eliminated; once this is done, all remaining subscripts are equal to $k + 1$, and suppressing them, Eqs. (22.17) and (22.18) become standard Bayes rule. Reinstating dependence on $\mathbf{z}_{1:k}$ is of course necessary when dealing with tracking over $k + 1$ frames involving previous observations $\mathbf{z}_{1:k}$.)

$$p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k+1}) = \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k})}{p(\mathbf{z}_{k+1}|\mathbf{z}_{1:k})} \tag{22.17}$$

where the normalizing constant is

$$p(\mathbf{z}_{k+1}|\mathbf{z}_{1:k}) = \int p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1})p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k})\,\mathrm{d}\mathbf{x}_{k+1} \tag{22.18}$$

The prior density is obtained from the previous time-step:

$$p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k}) = \int p(\mathbf{x}_{k+1}|\mathbf{x}_k)p(\mathbf{x}_k|\mathbf{z}_{1:k})\,\mathrm{d}\mathbf{x}_k \tag{22.19}$$

but note that this is only valid because of the Markov process (of order one) assumption commonly taken to simplify Bayesian analysis, which leads to

$$p(\mathbf{x}_{k+1}|\mathbf{x}_k, \mathbf{z}_{1:k}) = p(\mathbf{x}_{k+1}|\mathbf{x}_k) \tag{22.20}$$

In other words, the transition probability for the update $\mathbf{x}_k \rightarrow \mathbf{x}_{k+1}$ depends only indirectly on $\mathbf{z}_{1:k}$, via previous updates.

General solutions of this set of equations—in particular, Eqs. (22.17) and (22.19)—do not exist. However, restricted solutions are possible, as in the case of the Kalman filter (see Chapter 20: Motion), which assumes that all posterior densities are Gaussian. In addition, particle filters can be used to approximate the optimal Bayesian solution when Gaussian constraints are inapplicable.

The particle filter, also known as sequential importance sampling (SIS), the sequential Monte Carlo approach, bootstrap filtering, and condensation, is a recursive (iteratively applied) Bayesian approach that at each stage employs a set of

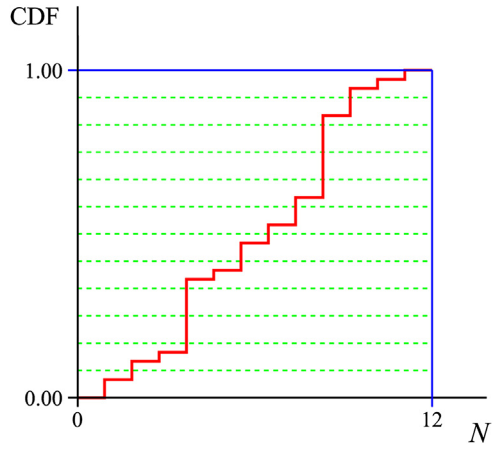

FIGURE 22.7

Use of the cumulative distribution function (CDF) to perform systematic resampling. Applying the regularly spaced horizontal sampling lines shows the cuts needed to find appropriate indexes (*N*) for the new samples. The cuts tend to ignore the small steps in the CDF and to accentuate the large steps by duplicating samples.

samples of the posterior density function (see Appendix D for basic concepts on sampling from distributions). It is an attractive concept because in the limit of large numbers of samples (or "particles"), the filter is known to approach the optimal Bayesian estimate (Arulampalam et al., 2002).

To apply this method, the posterior density is reformulated as a sum of delta function samples:

$$p(\mathbf{x}_k|\mathbf{z}_{1:k}) \approx \sum_{i=1}^{N} w_k^i \delta\left(\mathbf{x}_k - \mathbf{x}_k^i\right) \tag{22.21}$$

where the weights are normalized by

$$\sum_{i=1}^{N} w_k^i = 1 \tag{22.22}$$

Substituting into Eqs. (22.17)–(22.19), we obtain the posterior as follows:

$$p(\mathbf{x}_{k+1}|\mathbf{z}_{1:k+1}) \propto p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}) \sum_{i=1}^{N} w_k^i\, p\left(\mathbf{x}_{k+1}|\mathbf{x}_k^i\right) \tag{22.23}$$

where the prior now takes the form of a mixture of *N* components.

In principle, this gives us a discrete weighted approximation to the true posterior density. In fact, it is often difficult to sample directly from the posterior density: this problem is normally solved by SIS from a suitable "proposal" density

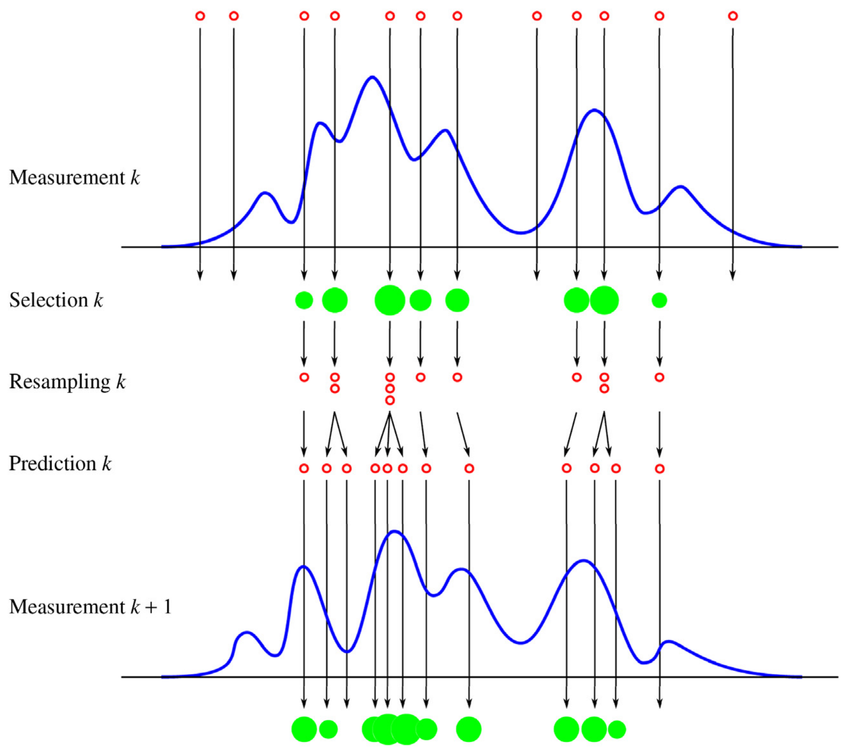

FIGURE 22.8

Perspective on the processes involved in particle filtering. Notice, how the filter cycles repeatedly through the same basic sequence.

function $q(\mathbf{x}_{0:k} \mid \mathbf{z}_{1:k})$. It is useful to take an importance density function that can be factorized:

$$q(\mathbf{x}_{0:k+1}|\mathbf{z}_{1:k+1}) = q(\mathbf{x}_{k+1}|\mathbf{x}_{0:k}\mathbf{z}_{1:k+1})q(\mathbf{x}_{0:k}|\mathbf{z}_{1:k}) \tag{22.24}$$

following which, the weight update equation can be obtained (Arulampalam et al., 2002) in the form:

$$\begin{aligned} w_{k+1}^i &= w_k^i \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^i)p(\mathbf{x}_{k+1}^i|\mathbf{x}_k^i)}{q(\mathbf{x}_{k+1}^i|\mathbf{x}_{0:k}^i, z_{1:k+1})} \\ &= w_k^i \frac{p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^i)p(\mathbf{x}_{k+1}^i|\mathbf{x}_k^i)}{q(\mathbf{x}_{k+1}^i|\mathbf{x}_k^i, \mathbf{z}_{k+1})} \end{aligned} \tag{22.25}$$

where the path $\mathbf{x}_{0:k}^i$ and history of observations $\mathbf{z}_{1:k}$ have been eliminated—as is necessary if the particle filter is to be able to track recursively in a manageable way.

In fact, pure SIS has the largely unavoidable problem that all but one particle will have negligible weight after a few iterations. More precisely, the variance of the importance weights is only able to increase over time, leading ineluctably to this degeneracy problem. However, one simple means of limiting the problem is to resample particles so that those with small weights are eliminated, while those with large weights are enhanced by duplication. Duplication can be implemented relatively easily, but it also leads to so-called sample impoverishment, i.e., it still results in some loss of diversity among the particles, which is itself a form of degeneracy. Nevertheless, if there is sufficient process noise, the result may prove to be adequate.

One basic algorithm for performing the resampling is "systematic resampling" and involves taking the cumulative discrete probability distribution (in which the original delta function samples are integrated into a series of steps) and subjecting it to uniform cuts over the range 0 to 1 to find appropriate indexes for the new samples. As will be seen from Fig. 22.7, this leads to small samples being eliminated and strong samples being duplicated, possibly several times. The result is called sampling importance resampling (Sir) and is a useful first step on the way to producing stable sets of samples. With this particular approach, the importance density is chosen to be the prior density:

$$q(\mathbf{x}_{k+1}|\mathbf{x}_k^i, \mathbf{z}_{k+1}) = p(\mathbf{x}_{k+1}|\mathbf{x}_k^i) \tag{22.26}$$

Appealing to Eqs. (22.25) shows that the weight update equation becomes enormously simplified to

$$w_{k+1}^i = w_k^i \, p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^i) \tag{22.27}$$

Moreover, as resampling is applied at every time index, previous weights w_k^i are all given the value $1/N$, so we can simplify this equation to

$$w_{k+1}^i \propto p(\mathbf{z}_{k+1}|\mathbf{x}_{k+1}^i) \tag{22.28}$$

As can be seen in Eq. (22.26), the importance density is taken to be independent of measurement $\mathbf{z}_{k+1}$, so the algorithm is restricted with regard to observational evidence, and this is one cause of the loss of particle diversity mentioned earlier.

The Condensation method of Isard and Blake (1996) goes some way to eliminating these problems by following the resampling with a prediction phase during which a diffusion process separates any duplicated samples, thereby helping to maintain sample diversity. This is achieved by applying a stochastic dynamical model that has been trained on sample object motions. Fig. 22.8 gives an overall perspective on the approach and includes all the sampling and other processes that have been discussed above.

The concept is taken further in the ICondensation approach (Isard and Blake, 1998) by using a mixture of samples, some using standard Sir and some using an importance function depending on the most recent measurement $\mathbf{z}_{k+1}$ but ignoring the dynamics. Thus, this complex method reflects the need to ensure continued sample diversity: It also aims to combine low- and high-level approaches to tracking by noting that model switching may be necessary when handling real-world tasks such as tracking human hands.

Similar ideas and motivation were employed by Pitt and Shephard (1999) in their auxiliary particle filter (APF). This generates particles from an importance distribution depending on the most recent observations, and then samples the posterior using this importance density. The algorithm involves an additional likelihood computation for each particle, but overall, the computational efficiency is improved because fewer particles are needed. Nevertheless, Nait-Charif and McKenna (2004) found that the method gave only limited improvement relative to Sir. They went on to make a comparison with the iterated likelihood weighting (ILW) scheme. In this approach, after an initial iteration of Sir, the sample set is split randomly into two sets of equal size; one of these is migrated to regions of high likelihood and the other is handled normally. The purpose is to cope on the one hand with situations where the prior is sound and on the other hand, with situations where it is not and regions of high likelihood need to be explored. When tracking human heads, the method proved to be a significantly more robust tracker than either Sir or the APF. Perhaps oddly, the ILW is designed to reduce approximation error rather than to give unbiased estimates of a posterior. This means that it is not based completely on probabilistic methods. On the other hand, as for the Isard and Blake ICondensation approach mentioned above, it is intended to match a variety of scenarios that can be found when tracking under real-world conditions, where it is difficult to model all probabilities accurately.

Many more particle filter methods have been developed over the past decade or so. Several incorporate the Kalman filter and its extended and "unscented" versions, in attempts to optimize likelihoods when sample diversity turns out to be insufficient. More recently "regularized" and "kernel" particle filters (Schmidt et al., 2006) have been developed to tackle the problem of sample impoverishment. These perform resampling using a continuous approximation to the posterior density, typically using the Epanechnikov kernel (Comaniciu and Meer, 2002). The mean shift approach is in this category. Essentially, the mean shift algorithm is a means of climbing density gradients to identify underlying modes in sparse distributions and involves moving a sampling sphere around the space being searched. This makes it a good iterative search technique, though it is only suitable for locating one mode at a time. It complements the particle filter formalism well, as it can be used to refine the accuracy with which objects may be found, and works well even if a limited number of particles are employed. It has recently been applied by Chang and Lin (2010) for tracking various parts of the moving human body.

At this stage, it is starting to become apparent that different tracking applications will demand different types of particle filter. This will depend on a variety of factors including how jerky the motion is, whether rotations will be involved, whether occlusions will occur, and if so for how long—and of course on the appearance and variability of the objects being tracked. It should be noted that all the theory and most of the ideas presented above reflect abstract situations and the concentration is on relatively small, well-localized objects that are considered locally, i.e., the filters themselves will not have a global understanding of the situation. Thus, they must be categorized as low or intermediate level vision. In contrast, the human eye is an excellent tracker by virtue of its capability for thinking about what objects are present and which ones have moved where, including passing behind other objects or even temporarily out of the scene. Clearly, we must not expect too much from particle filters just because they are based on probabilistic models.

An important advantage of particle filters is that they can be used to track multiple objects in an image sequence. This is because there is no record of which object is being tracked by which particles. However, this possibility arises only because no restrictions are placed on the posterior densities: In particular, they are not assumed to be Gaussian as in the case of Kalman filters. Indeed, if Kalman filters are used for tracking, each object must be tracked by its own Kalman filter.

Once a suitable approach to particle filtering has been arrived at, it is necessary to determine how to implement it. The basic means of achieving this is via appearance models. In particular, color and shape models are frequently used for this purpose. However, before delving into this topic, it will be useful to find what can be achieved by color analysis using what is by now quite an old approach—that of color indexing via color histogram matching.

22.5 USE OF COLOR HISTOGRAMS FOR TRACKING

One of the most useful tools that is available for object tracking was developed as early as 1991 by Swain and Ballard (1991) in a paper called "Color indexing." The aim of that work was to index from a color image into a large database of models. In a sense that idea is the inverse of the tracking problem, as its purpose was to search for the model with the best match to a given image rather than to search for instances of a given model in frames from an image sequence. Actually, database searches fall in the realm of classification, whereas the process of tracking assumes implicitly that the object in question has already been identified. However, apart from one important difference which will be discussed below, this is a rather minor point.

The main idea behind the color indexing approach is that of matching color histograms rather than the images themselves. There is an obvious validity in this approach, in that if the images match, so will their color histograms. What is more, as the histograms have no memory of where in the image a particular color originated, histograms are invariant to translation and rotation about the viewing axis (so-called "in-plane rotation"). Also relevant is the fact that a planar object that is subject to out-of-plane rotation will still have the same color histogram, although it will involve different numbers of pixels, so normalization will be required. The same applies for objects that are at different depths in the scene: The histogram profile will be unchanged, but it will have to be normalized to allow for the different numbers of pixels that are involved. Finally, a spherical or cylindrical object with the same set of colors distributed similarly over its surface will again have the same histogram. While exact adherence to this scenario might be relatively rare, it would apply almost perfectly for a ball of wool or a football, and with varying degrees of exactness for a shaven human head or torso. In fact, the main problem with use of histograms for recognition is the possible ambiguity it could bring, but when tracking a known object that has moved only a small distance between frames, this problem should be a minor one.

The above explanation demonstrates the potential power of the histogram approach but raises an important question about what happens when the object moves in such a way as to be larger or smaller than the model, whether through depth scaling or through out-of-plane rotation. In particular, if it becomes smaller, this will mean that the model will be matched partly against the object background. Swain and Ballard sought to minimize this effect by taking the following intersection measure rather than any sort of correlation between the image I and model M histograms:

$$\sum_{i=1}^{n} \min(I_i, M_i) \tag{22.29}$$

This would have the effect of discounting any pixels of a given color in excess of those expected in the model histogram (including both those whose colors are simply not represented in the model and those that have limited representation). The above expression was then normalized by the number of pixels in the model histogram. However, here, we follow Birchfield (1998) in normalizing by the number of pixels in the image histogram, to reflect the point made earlier that we are searching for the best image match rather than the best model match:

$$H_{\mathrm{N}}(I, M) = \frac{\sum_{i=1}^{n} \min(I_i, M_i)}{\sum_{i=1}^{n} I_i} \tag{22.30}$$

At first sight, this formula might appear wrong, in that a match over fewer pixels would be normalized out, still representing perfect agreement and giving a normalized intersection of unity. Note, for example, that in shape matching, it is common to use the formula $(A \cap B)/(A \cup B)$, where A and B are sets representing object areas, which would give a value less than unity in the case $A \supset B$. However, Eq. (22.30) is designed to cope well with partial occlusions in the image, which will lead to the intersection with M being reduced, yielding the I values, which would then cancel with the denominator, giving the answer 1.

The overall effect of using the normalized intersection of Eq. (22.30) is that the method has the twin advantages of minimizing the effects of background and canceling the effects of occlusion, while also coping well (and in some cases exactly) with varying viewpoints. The problem of varying scale remains, but this can be countered by preliminary segmentation of the object and scaling its histogram to the size of the model histogram.

There remains one further important consideration when matching an image against a model—that the model might have become out of date under varying levels of illumination. To a large extent, the latter can be considered as varying levels of *luminance*, with the *chrominance* parameters remaining more or less unchanged. This problem can be addressed by changing to different color representations. For example, we can move from the RGB representation to the HSI (hue, saturation, intensity) representation (see Appendix C), and then use the hue (H) and saturation (S) parameters. However, more protection will be available by color normalization (dividing by I)—though it is far easier and less computation intensive to normalize the RGB parameters directly:

$$r = R/(R + G + B) \tag{22.31}$$

$$g = G/(R + G + B) \tag{22.32}$$

$$b = B/(R + G + B) \tag{22.33}$$

but because $r + g + b = 1$, we should ignore one of the parameters, e.g., b.

While the above arguments suggest that luminance should be totally ignored, this is inadvisable, as colors that are close to the black–white line in color space (where saturation $S \approx 0$) would be indistinguishable. Indeed, Birchfield (1998) cites the "dangerous" case of dark-brown hair looking similar to a white wall if luminance is ignored. For these reasons, most workers use different sizes and numbers of histogram bins for luminance and chrominance information. Here, we have to remember that a full-sized color histogram with 256 bins in each of the color dimensions would be both large and clumsy and would not easily be searchable in real time—an especially important factor in tracking applications. In addition, such a histogram would not be well populated and would lead to very noisy statistics. For this reason, 16–40 bins per color dimension are much more typical. In particular, $16 \times 16 \times 8$ bins are widely used, 8 being the number in the luminance channel. Note that these numbers correspond, respectively, to 16, 16, and 32 levels per channel, and that a 512×512 image would lead to an average occupation number of 128 per bin: However, a 256×256 image would give an average occupation of just 32 per bin, which is distinctly low and liable to be inaccurate (though this would depend very much on the type of data).

Birchfield (1998) reported that when used for head tracking, the color histogram method was able to follow a head reliably, though it became "unstable" when the head was in front of a white board whose color was quite close to that of skin. This behavior is understandable, as it has already been noted that the histogram approach is invariant to translation (hence, as long as the head is somewhere within the image, the histogram tracker will be unlikely to lose it). These points show that ultimately the histogram tracker approach is limited, and needs to be enhanced by other means, in particular some means of detecting object outlines. To achieve this, Fieguth and Terzopoulos (1997) used (1) simple M-ary hypothesis testing of position around the previous position, the displacements merely being those at the $M = 9$ points in a 3×3 window; (2) a highly nonlinear velocity prediction scheme involving step incremental corrections for acceleration, deceleration, and damping to avoid oscillations; and (3) color histograms bins based exclusively on chrominance. The reason for these simplifications was to achieve real-time operation for full frames (640×480 pixels) at 30 frames per second—at which rate, object displacements become much smaller and easier to track.

Birchfield (1998) developed a more sophisticated approach, based on approximating the shape of the human head by a vertical ellipse with a fixed aspect ratio of 1:2. Then, in common with previous contour trackers, he measured the goodness of match by computing the normalized sum of gradient magnitudes around the boundary of the ellipse, though (1) he summed the gradient values at all points on the boundary rather than just at selected points (in fact, this is unusual: most workers sample at a set of 100 or so points on the boundary), and (2) he took the component of the gradient along the perpendicular to the boundary. This led to a shape model $\mathbf{s}(x, y, \sigma)$ with three parameters—x, y denoting the ellipse location and σ denoting its semiminor axis—and the following goodness of fit parameter:

$$\psi(\mathbf{s}) = \frac{1}{N_\sigma} \sum_{i=1}^{N_\sigma} \left| \mathbf{n}_\sigma(i) \cdot \mathbf{g}_\mathbf{s}(i) \right| \tag{22.34}$$

Here, N_σ is the number of pixels on the boundary of an ellipse with semiminor axis σ, $\mathbf{n}_\sigma(i)$ is the unit vector normal to the ellipse at pixel i, and $\mathbf{g}_\mathrm{s}(i)$ is the local intensity gradient vector. Normalized goodness of fit parameters for boundary shape (ψ_b) and color (ψ_c) is added and used to obtain an optimum fit:

$$\mathbf{s}_\mathrm{opt} = \arg \max_{\mathbf{s}_i} \{\psi_\mathrm{b}(\mathbf{s}_i) + \psi_\mathrm{c}(\mathbf{s}_i)\} \tag{22.35}$$

As discussed earlier, when the color module was tested on its own, it performed well, but somewhat unstably when the background color was close to that of skin. All this was corrected by adding the gradient module. However, the gradient module on its own performed less well than the color module on its own. As time progressed, the gradient model tended to became distracted by the background, not having any inbuilt design features to counteract this. Moreover, in a cluttered background, it behaved even less well; and it was not able to handle large accelerations adequately because of its limited ability to probe for high gradient regions, and its consequent propensity for attaching itself to the wrong ones. Fortunately, the two modules were able to complement each other's capabilities: in particular, the color module helped the gradient module by its ability to ignore background clutter, and by providing a larger region of attraction. Finally, when the human subject turned around, so that only his hair was visible, the gradient module was able to take over and handle rescaling correctly as the subject moved; and it was able to prevent the color tracker from slipping down the subject's neck, which had a similar color histogram. All this signals that two or more strategies for tracking can be useful in real-world situations where enough information needs to be brought to bear to provide correct tracking interpretations on an ongoing basis. It also signals that the color histogram type of tracking module is exceptionally powerful and tends to need only minor tweaks to keep it properly on lock. Overall, however, the outstanding factor that needs further detailed attention and development is the handling of occlusion: this needs to be arranged by design rather than by tweaks, as we shall see in a later section.

22.6 IMPLEMENTATION OF PARTICLE FILTERS

The particle filter formalism is a very powerful one, mediated by generic probability-based optimization, yet needing to be taken further to achieve its promise. To arrange this, it needs to be applied to real objects and thus appearance models have to be taken into account—though in the present context, it will be more accurate and general to refer to them as observation models. Our particle filter formalism already embodies these in the form of conditional densities $p(\mathbf{z}_k|\mathbf{x}_k)$: See Eq. (22.28).

At this stage, we need to specialize the observations: here, we illustrate the process by considering the color and assumed elliptical shape of a human head: these can be thought of as region-based (r) and boundary-based (b) properties, each with their own likelihoods. Taking the latter to be conditionally independent, we can factorize $p(\mathbf{z}_k|\mathbf{x}_k)$ as follows:

$$p(\mathbf{z}_k|\mathbf{x}_k) = p(\mathbf{z}_k^{\mathrm{r}}|\mathbf{x}_k)p(\mathbf{z}_k^{\mathrm{b}}|\mathbf{x}_k) \tag{22.36}$$

Clearly, the region-based likelihood will depend not only on the color but also on the shape of the region it is in. Nevertheless, the conditional independence assumption will be valid, as we are really interested in the colors within the boundary and the gradient values along it.

To proceed further, we assume that color histograms I and M have been obtained for the image and the target model, within the current region r. Following Nummiaro et al. (2003) and many other workers—and at this point, abandoning the Swain and Ballard (1991) normalized intersection formalism—we normalize them to unity as p^{I}, p^{M}, respectively. To compare these distributions, it is convenient to use the Bhattacharyya coefficient (here, expressed as a sum rather than an integral) which expresses the similarity between the distributions:

$$\rho\left(p^{\mathrm{I}}, p^{\mathrm{M}}\right) = \sum_{i=1}^{m} \sqrt{p_i^{\mathrm{I}} p_i^{\mathrm{M}}} \tag{22.37}$$

To display the *distance* between the distributions, we simply apply the measure

$$d = \sqrt{1 - \rho(p^{\mathrm{I}}, p^{\mathrm{M}})} \tag{22.38}$$

Ideally, the color distribution will be close to the target distribution, so these should differ only as a Gaussian error function. Remembering that p^{I} is actually a function of $\mathbf{x}_k$, we now find the region (and color) conditional likelihood:

$$p\left(\mathbf{z}_k^{\mathrm{r}}|\mathbf{x}_k\right) = \frac{1}{(2\pi\sigma_{\mathrm{r}}{}^2)^{1/2}} e^{-\frac{d^2}{2\sigma_{\mathrm{r}}{}^2}} = \frac{1}{(2\pi\sigma_{\mathrm{r}}{}^2)^{1/2}} e^{-\frac{1-\rho\left(p^{\mathrm{I}}(\mathbf{x}_k), p^{\mathrm{M}}\right)}{2\sigma_{\mathrm{r}}{}^2}} \tag{22.39}$$

Making a similar assumption that the estimated gradient positions in the image I will differ from those in the target model M by a Gaussian error function, we find the boundary conditional likelihood as follows:

$$p\left(\mathbf{z}_k^{\mathrm{b}}|\mathbf{x}_k\right) = \frac{1}{(2\pi\sigma_{\mathrm{b}}{}^2)^{1/2}} e^{-\frac{G^2}{2\sigma_{\mathrm{b}}{}^2}} \tag{22.40}$$

where G represents the sum of the gradient magnitude values perpendicular to the local boundary positions.

Combining the last two equations, as specified by Eq. (22.36), now provides the required estimate of $p(\mathbf{z}_k|\mathbf{x}_k)$, which in turn leads via the particle filter formulation to an estimate of $p(\mathbf{x}_k|\mathbf{z}_{1:k})$. This essentially completes the long series of arguments and calculations comprising the particle filter scenario.

In fact, there are several further aspects to consider. The first is that it is natural to weight the contributions made by the various pixels to the color histograms. In particular, the pixels nearest to the centers of the ellipses should be weighted higher than those near their boundaries, so that any inaccuracies in the center locations will be minimized. For example, Nummiaro et al. (2003) used the weighting function as follows:

$$k(r) = \begin{cases} 1 - \dfrac{r^2}{r_0{}^2}: & r < r_0 \\ 0: & r \geq r_0 \end{cases} \tag{22.41}$$

with $r_0 = \sqrt{a^2 + b^2}$, a and b being the semimajor and semiminor axes of the ellipses. In fact, Nummiaro et al. (2003) placed such reliance on this weighting that their particle filter did not use a separate boundary likelihood $p(\mathbf{z}_k^{\mathrm{b}}|\mathbf{x}_k)$. In contrast, Zhang et al. (2006) used both, almost exactly as described above, albeit with an APF incorporating mean shift filtering.

Another important aspect not so far mentioned is the need to adapt the target model M to keep it up to date, e.g., with regard to the size and orientation of the real-world target. Nummiaro et al. (2003) achieved this using the commonly applied "learning/forgetting" operation:

$$p_{k+1,i}^{\mathrm{M}} = \alpha p_{k,i}^{\mathrm{M}} + (1-\alpha) p_{k,i}^{\mathrm{I}} \qquad i = 1, 2, \ldots, m \tag{22.42}$$

which mixes in a little of the recent image data while forgetting a correspondingly small amount of the old model data. During this process, care is taken to avoid mixing in outlier data, such as when an object is partly occluded. Even with this precaution, it should be borne in mind that use of an adaptive model is potentially dangerous: while it helps by valid adaptation to appearance changes, it gives an increased sensitivity to extended occlusions and loss of target.

While heads are typically tracked using 2-D position (x, y) and ellipse shape parameters (a, b), it can normally be assumed that the ellipse is vertically aligned. However, when viewed from overhead, in-plane orientation (θ) is also an important parameter. Sometimes, similar models are used for individual human limbs, though rectangles have also been employed. However, ellipses provide a simple, easily parameterized shape and can be specified with as few as three parameters (x, y, b); they can even be used to track whole human figures using three or four parameters (Nummiaro et al., 2003). On the other hand, when torsos or hands are being tracked, closed curves may not be appropriate, and it is common to use parametric spline curves.

With the type of particle filter design outlined above, performance in the event of occlusions is a vexed question. Indeed, in this area, many claims and counter-claims about relative effectiveness of tracking and occlusion handling capabilities are made in various papers. As the claims are often made on different datasets, it is difficult to know the true position. However, the particle filter has quite a high level of intrinsic robustness. This is because "less likely object states have a chance to temporarily remain in the tracking process, (so) particle filters can deal with short-lived occlusions" (Nummiaro et al., 2003). Hence, minor propping up in a judicious way using other modules can often boost performance significantly. In principle, if a significant change such as a strong partial occlusion occurs, the simple artifice of putting the tracker on hold is often sufficient to allow it to recover and continue tracking. However, to be surer of recovery, the tracker might have to wait for a background subtraction routine to signal that the object is again present (Nait-Charif and McKenna, 2006). In any case, a background subtraction module is useful for signaling when a totally new object has entered the scene. Finally, when objects leave the scene, some memory of their appearance and position is useful in case they reenter the scene after a short time in the same or other location (when humans appear indoors, there are usually a limited number of entry and exit points, and reentry via the same one will generally be the most likely possibility). However, there is a danger of instituting a rather ad hoc set of algorithms to solve such problems, when what is needed is a more absolute object recognition module to positively identify individuals, or at least to search for the most likely identifications, together with sets of probabilities. A particular example of this type of situation is when two pedestrians walking in opposite directions (1) pass each other without interacting, but with the one momentarily occluding the other; or (2) stop, shake hands, and then proceed; or (3) stop, shake hands, and then retrace their steps. Scenario (3) involves merging of profiles and can be as difficult to handle as occlusion: In any case, temporary partial occlusion involves merging of the figures;

only seldom does complete occlusion and total disappearance of one figure occur. It ought to be stressed that scenario (1) is handled well by use of a Kalman filter module, which uses continuity of velocity to aid interpretation; scenario (2) is handled badly or not at all by such a module, depending on the time delay; while scenario (3) is not handled at all by such a module. (These points about use of Kalman filters are well illustrated by Nummiaro et al. (2003) in relation to a quite different scenario—that of a bouncing ball.) In general, human interactions have to use a Kalman filter at most tentatively, to throw up possible hypotheses about the motion: this being so, it is possible to incorporate Kalman filters usefully into a particle filter (van der Merwe et al., 2000); equally, they can be incorporated into supervisory programs that oversee the whole tracking process, as indicated above (see also Comaniciu et al., 2003).

22.7 CHAMFER MATCHING, TRACKING, AND OCCLUSION

As we have seen, one of the perennial problems of matching and tracking is that of occlusion of objects within the FOV. A variety of measures can be applied to make single camera systems as robust as possible against overlap. Leibe et al. (2005) have devised methods based on chamfer matching and segmentation, together with a minimum description length procedure for hypothesis verification. The latter evaluates hypotheses in terms of the savings that can be made by explaining part of the image by the hypotheses. Here, we concentrate on the concept of chamfer matching, as it has achieved considerable use for matching pedestrians, notably by Gavrila (e.g., Gavrila, 1998, 2000).

The basic idea behind chamfer matching relates to the process of matching objects to templates via their boundaries—a strategy that should be much less computation intensive than matching via whole object regions. However, since this would not give much indication of a potential match until very close to the match position, some means is required of making the approach to a match far smoother. This should also permit substantial speedup of the process by employing a hierarchical coarse-to-fine search. To achieve a smoother transition, edge points in the image are first located, and then a distance function image is generated, starting with the edge points, which are initialized to zero distance values. Application of the template, also in the form of edge points, will ideally yield a zero sum (of image distance function values) along the template points: This will rise to a higher value when the template is misplaced or the shape of the object is distorted, corresponding to the sum of distances of each image point from the ideal position. Taking the distance function as $DF_{\mathrm{I}}(i)$, we can express the degree of match by the average "chamfer" distance, i.e., the average distance from each edge point to the nearest edge point in the template T:

$$D_{\mathrm{chamfer}}(\mathrm{T},\mathrm{I}) = \frac{1}{N_{\mathrm{T}}}\sum_{i=1}^{N_{\mathrm{T}}} DF_{\mathrm{I}}(i) \tag{22.43}$$

where N_T is the number of edge points within the template. $D_{\text{chamfer}}(\mathrm{T}, \mathrm{I})$ is actually a dissimilarity measure, having a value of zero for a perfect match.

In fact, there is no necessity to take edge points for the image and the template: corner points or other feature points can be utilized, and the method is quite general. However, the method works best when the point set is sparse, so that (1) accurate location is achieved and (2) computation is reduced. On the other hand, reducing the number of points too far will result in lack of sensitivity and robustness as parts of the image and template will not be adequately represented.

As it stands, this approach is limited because any outliers (caused by occlusion or segmentation errors, for example) will lead to substantial matching problems. To limit this problem, Leibe et al. (2005) used a truncated distance for matching

$$D_{\text{chamfer}}(\mathrm{T}, \mathrm{I}) = \frac{1}{N_{\mathrm{T}}} \sum_{i=1}^{N_{\mathrm{T}}} \min(DF_{\mathrm{I}}(i), d) \tag{22.44}$$

with a suitable empirical value of d. On the other hand, Gavrila (Gavrila, 1998) applied an order-based method for limiting the number of interfering distance function values, taking the kth of the ordered values (1 to N_{T}) as the solution value:

$$D_{\text{chamfer}}(\mathrm{T}, \mathrm{I}) = \arg\ \text{order}_k^{i=1:N_{\mathrm{T}}}\ DF_{\mathrm{I}}(i) \tag{22.45}$$

When applying this formula, it may seem attractive to use the median value, for which $k = \frac{1}{2}(N_{\mathrm{T}} + 1)$. However, it can easily happen that a large proportion of the template area is obscured, so we need to take a smaller value of k (e.g., $0.25N_{\mathrm{T}}$) that reflects this. In fact, this will reduce accuracy when none of the template is obscured, so, in the end, Eq. (22.44) might give a more useful result. We take this discussion no further here, as a lot depends on the type of data that is involved. In passing, it is worth observing that when $k = N_{\mathrm{T}}$, Eq. (22.45) gives the well-known Hausdorff distance (Huttenlocher et al., 1993):

$$D_{\text{chamfer}}(\mathrm{T}, \mathrm{I}) = \max_{i=1:N_{\mathrm{T}}} DF_{\mathrm{I}}(i) \tag{22.46}$$

This formula for the Hausdorff distance may appear different from the usual one, which involves a max–min operation: However, as computation of a distance function involves taking local minima of possible distances (see Chapter 8: Binary Shape Analysis), there is concurrence in the two formulations.

Note that, in the foregoing discussion, the distance function of the image is used rather than that of the template. This is because in practical situations, many templates will have to be applied in order to cover expected variations in the objects being detected. For example, if the method is being applied for pedestrian detection, various sizes, poses, positions of limbs, and types of clothing will have to be allowed for, as well as variations in the background and possible overlaps. In these circumstances, it is far more efficient to use DF_{I} than DF_{T}. Gavrila (1998) showed with considerable success, how all the variations listed above can be dealt with and how the method can be made to work well to detect pedestrians.

Finally, returning to the work of Leibe et al. (2005), limitations of the chamfer matching technique were compensated by using segmentation information. This meant obtaining a similarity function from the chamfer distance (which is a dissimilarity measure), and then combining with a Bhattacharyya coefficient representing overlap with the hypothesized segmentation $\mathrm{Seg_I}(i)$ to produce an overall similarity measure:

$$S = a\left[1 - \frac{1}{b}D_{\mathrm{chamfer}}(\mathrm{T},\mathrm{I})\right] + (1-a)\sum_i \sqrt{Seg_{\mathrm{I}}(i)R_{\mathrm{T}}(i)} \tag{22.47}$$

Here, $R_{\mathrm{T}}(i)$ is the region within T, and the sum covers the pixels in this region. In addition, a somewhat arbitrary but nonetheless reasonable pair of weights is applied to balance the two similarity measures: a is the proportion of the overall similarity allotted to chamfer matching, and b is a weight expressing the fact that chamfer matching is applied over a significant boundary distance; in the work of Leibe et al. (2005), a and b were taken to be 0.45 and 50, respectively. The overall effect was to produce much improved solutions in respect of placement accuracy and elimination of false positives, relative to the chamfer distance method taken on its own (Eq. (22.44)).

22.8 COMBINING VIEWS FROM MULTIPLE CAMERAS

Over the past decade or so, there has been a surge of interest in multicamera surveillance systems. Multiple cameras are clearly necessary if, for example, long stretches of motorway are to be monitored, or if pedestrians are to be tracked around cities or shopping precincts. The FOV of a single camera is quite restricted and the resolution available for viewing in the distance will almost certainly be inadequate for detailed observation. Another reason for the use of several cameras is that of viewing in stereo and obtaining sufficient depth information. A further reason is that pedestrians in a precinct will frequently be partially or wholly occluded by architectural features such as statues or other pedestrians, but the chance of missing a pedestrian will be less, much less if the scene is viewed by multiple cameras; this sort of situation will also apply on roads, where many other possibilities for occlusion exist.

On roads, cameras are often mounted on overhead gantries, and maintaining observation over long distances will require many cameras. This raises the question of whether the observation should be unbroken, i.e., whether the cameras will have overlapping, contiguous, or nonoverlapping views. On motorways, cameras may be separated by several miles, and can usefully be sited at junctions, so it will be possible to keep track of all vehicles without too much expense, though breakdowns at intermediate locations may not be observed. On the other hand, in a shopping precinct, if pedestrians are to be monitored closely enough for attacks or terrorist activities to be detected, contiguous or overlapping views will be mandatory. In fact, there will be a problem in ensuring that all pedestrians are positively identified as they progress from one FOV to the next: To facilitate this, and for ease of setting up the system, overlapping views are normally required.

Next, we consider the layout of a multicamera system. To do this, we must examine the area of the ground plane that lies within the FOV of the camera. First, note that the optical axis of the camera passes through the center of the image plane and that the latter has a rectangular shape given by the minimum and maximum values of x and y, $\pm x_{\mathrm{m}}$ and $\pm y_{\mathrm{m}}$. The FOV is therefore limited by four planes, at horizontal and vertical angles $\pm\alpha$ and $\pm\beta$, where $\tan\alpha = x_{\mathrm{m}}/f$ and $\tan\beta = y_{\mathrm{m}}/f$, f being the focal length of the camera lens. Each plane will intersect with the ground plane in a line, and for a camera with a horizontal x-axis, the viewed area on the ground plane will be a symmetrical trapezium (Fig. 22.9). However, following on from the discussion in Section 22.2, if the camera is not inclined slightly downward, the distant side of the trapezium will not be visible. Since this would not make the most of the camera FOV, we will assume that it has been arranged for the distant side to fall on the ground plane.

When an adjacent camera views an adjacent section of the ground plane, there are two possibilities: (1) it will view the next stretch in the same direction, as on a motorway and (2) it will not be restricted to lie, or point, in the same direction, but just to overlap in some convenient way. For example, in a precinct or park, a typical placement would be that shown in Fig. 22.10A, where two opposite sides of the common viewing area would arise from the FOV of the first camera and the other two from that of the second camera—thereby forming a quadrilateral rather than a trapezium. However, other situations are possible, as shown in Fig. 22.10B, where the trapezia of the two cameras overlap in a more complex way, and the common viewing area is not a quadrilateral.

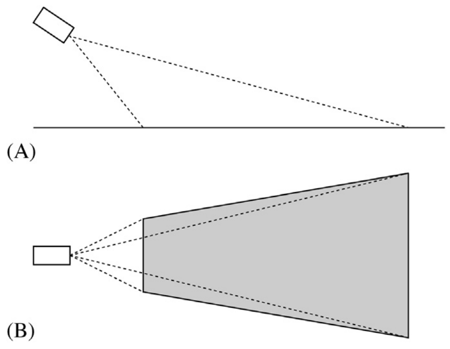

FIGURE 22.9

Area on the ground plane viewed by a camera. (A) Side view with the camera canted slightly downward. (B) Plan view of the symmetrical trapezium seen by the camera on the ground plane.

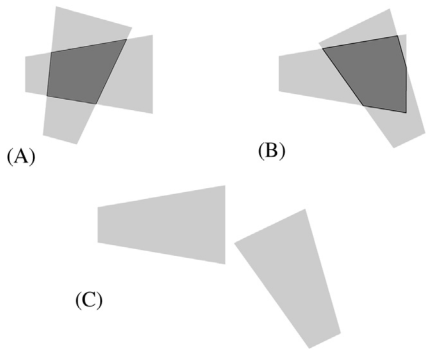

FIGURE 22.10

Areas on the ground plane viewed by multiple cameras. (A) Overlapping trapezia forming a quadrilateral. (B) Overlapping trapezia forming another shape—here, a pentagon. (C) Trapezia that do not overlap, though tracking across the gap can in some cases be achieved by making spatial and temporal correspondences (see text).

No matter which of the reasons for using a multicamera system apply, there is a need to relate the views from the separate cameras in order to obtain a consistent labeling of the objects passing between them. The obvious means of achieving this is by appearance, i.e., to apply recognition algorithms to establish that the same person or vehicle is being tracked across the various camera fields of view. Unfortunately, while this correspondence problem can normally be solved straightforwardly in binocular vision, when the two cameras are close together and pointing in a similar direction, this is by no means true for wide baseline cases such as those shown in Fig. 22.10. This is so for two reasons: (1) A person seen in two disparate views may have an altogether different appearance: e.g., the face may be visible in one and the back of the head in the other, or the back of a shirt may have a different design or color from the front and (2) the illumination may be quite different for each of the views, and this will make it even more difficult to confirm the person's identity from the other camera.

The obvious solution to this problem is to confirm identity not by appearance but by position and time. If we know that person P is at position **X** in the scene at time t, this must be the case in all views. So, all that has to be done is to relate the common areas of the ground plane uniquely between cameras. Following the widely used and usually sufficiently accurate assumption that everything is happening on the same flat ground plane, we only need to set up a homography between the two cameras to arrange for the same correct interpretation from any view. Under perspective projection, it requires a minimum of four common feature points to set up a homography (the number is as small as this because of the planar constraint, as is made clear in Table 16.1), though more points can be used to improve accuracy; note also that at least one more point is needed to validate the homography.

In the work of Calderara et al. (2008), greater accuracy was achieved by finding the straight lines bounding the common quadrilateral and using its corners as highly accurate points by which to define the homography. While this might seem trivial, in fact, the common quadrilateral has to be located by experiment. This can be achieved most easily when the scene is empty (e.g., overnight), and one individual can be sent to walk repeatedly around the site until a sufficient number of boundary points—as determined by the individual entering or leaving one of the fields of view—have been measured in both views. Note that to ensure that this gives sound results, temporal synchronization of the two camera systems is crucial. Once all this has been carried out, methods such as Hough transforms or RANSAC are applied to collate the boundary points into the straight lines bounding the quadrilateral: and because of the averaging inherent in this process, the straight lines will be known accurately; therefore, the corner positions will be known accurately, so there will be no need to use more points to establish an accurate homography.

Interestingly, Khan and Shah (2003) consider this approach an overkill to solve the consistent labeling problem. They assert that there is no need to determine the homography in this numerical sort of way: rather, it should be done by finding the FOV boundary lines and then merely noting when a pedestrian passes over one of these lines and making the identity at that point in time. That is, if an individual crosses a line at time t, this will be detected at the same time t in each camera and the person's identity can be passed across at that moment. This process is commonly called camera "handoff" (whereas it might appear to be more natural to call it "handover," there is a subtlety in that the latter term would tend to imply that the fields of view are contiguous rather than overlapping). However, if a group of people all cross the line together, this could obviously give rise to difficulties. Indeed, the whole problem of tracking groups of individuals is a difficult one and becomes almost insuperable in dense crowd situations.

While finding FOV boundary lines can be carried out when no crowds are present, and ideally when a single individual walks around, there are limits to the performance of the trained system. This is because a homography relates to a plane, and the simplest way of defining and using a plane is to use the foot locations to provide the plane contact points. (In principle, this is easily done by taking the lowest point on the individual.) However, when the calibrated system is used, the feet of one individual will often be obscured by another individual—a situation that will be virtually unavoidable in crowds. Consequently, there has been a fair amount of attention to recognizing and locating individuals from the tops of their heads (e.g., Eshel and Moses, 2008, 2010). Clearly, tops of heads are much less likely than feet to be occluded. Hence, even in crowd conditions, as long as cameras are quite high up and canted down at quite high angles (say 40°), all but the shortest individuals should be identifiable. Interestingly, apart from orientation, tops of heads may actually look similar in different views. As the camera cant angle will be known, altered head orientation can be allowed for and recognition and cross identification between cameras can proceed. With fully calibrated cameras (see Chapter 19: Image Transformations and Camera Calibration), tops of heads can be located in 3D space, and the positions of feet and heights of individuals can be deduced. Unfortunately, full camera calibration is a tedious process and may need frequent updating, so it is better not to rely on that approach in "informal" (and therefore changeable) surveillance situations such as shopping centers. Instead, camera views can be related using the fundamental matrix formulation (Chapter 19: Image Transformations and Camera Calibration), which requires only that epipoles should be known so that epipolar lines can be determined; however, finding them requires considerable computation, though this can be done offline prior to actual use (Calderara et al., 2008).

An intriguing approach to top-of-head location is to try various homographies differing only in the parameter H signifying distance from the floor. When a homography is found that indicates the same value of H, the foot locations can be calculated for each camera view, even though the feet themselves are obscured. However, to achieve this, a somewhat complex and subtle process is required (Eshel and Moses, 2008, 2010). Four vertical poles are set up at the corners of each viewing quadrilateral (or other convenient location), each pole having three bright lights along it (e.g., at the top, bottom, and middle of the pole). Then standard homographies are set up for each of these, so that at any location in an image, three heights can be deduced. Finally, a height that is to be measured can be related to the three known ones for that location, a cross ratio calculated along a vertical line, and the actual height deduced; at the same time, the foot position in each camera view can be identified unambiguously.

Overall, the simplest and most powerful approach is that of prior training by getting someone to walk around the site and thus demarcate the boundaries of each common viewing zone. Then, applying the fundamental matrix for pairs of cameras will permit homographies to be set up relating all the mutually viewable regions of ground planes. The paper by Calderara et al. (2008) contains a number of other subtleties, but space prevents them from being described in detail here. Finally, if heights and exact locations of people are to be found from top-of-head positions, elegant though fairly complex methods using several homographies have to be used, but in some applications, such as observation of crowds, the additional complexity may well be justified. However, segmentation of crowd views and identification of all individuals remains a research topic, especially when the people are tightly packed—as can easily happen in metro stations and football matches.

22.8.1 THE CASE OF NONOVERLAPPING FIELDS OF VIEW

Next, we move on to the case of nonoverlapping fields of view. Here, there seems to be no basis for homographies or for reliable camera handoff. However, some degree of similarity in appearance will still be detectable between views; in addition, there will be strong correlations between the time of leaving one FOV and arriving in another. The situation will often be helped if there is some restriction of access, such as would occur if there is a single adjoining door. (On a motorway, there is anyway such a restriction, and temporal correlations can be strong.) Pflugfelder and Bischof (2008) have obtained significant success in this sort of situation and make no assumptions about appearance. In particular, they have found how to relate the camera calibration matrices when overlapping views are not available. While this seems intrinsically impossible because no common image points can be found and hence no equations can be obtained linking the parameters (recall that the 8-point algorithm requires eight points in order to obtain a sufficient number of equations), they have found that if velocities are assumed to be more or less constant across the intervening space, this provides the continuity needed to permit enough equations to be found. Thus, a minimum of two positions per view for each trajectory is sufficient, these being immediately before and after camera handoff. Strict temporal correspondences are required, as are data on relative camera orientations, but a common ground plane is not assumed. Under these conditions, tracking across gaps of up to 4 m was achieved (Fig. 22.10C). The method works because Rother and Carlsson's (2001) 2-point technique shows how to determine the relative positions of two cameras with overlapping views, and the new method simulates this situation by utilizing a separate two points in the second nonoverlapping view in order to emulate and replace the two points that would ideally have been present in an overlapping view.

For a differently motivated probabilistic strategy tackling this problem, based on transition probabilities between nonoverlapping views, see Makris et al. (2004): What is special about this approach is that it is quite general as it is entirely unsupervised and has no direct knowledge of camera placement or camera characteristics.

22.9 APPLICATIONS TO THE MONITORING OF TRAFFIC FLOW

22.9.1 THE SYSTEM OF BASCLE *ET AL.*

One important area of surveillance is the visual analysis of traffic flow. In an early study (Bascle *et al.*, 1994), it was found that the complexity of the analysis was reduced because vehicles run on the roadway and because their motions are generally smooth. Nevertheless, the methods that had to be used to make scene interpretation reliable and robust were nontrivial.

First, motion-based segmentation is used to initialize the interpretation of the sequence of scenes. The motion image is used to obtain a rough mask of the object, and then the object outline is refined by classical edge detection and linking. B-splines are used to obtain a smoother version of the outline, which is fed to a snake-based tracking algorithm. The latter updates the fit of the object outline and proceeds to repeat this for each incoming image.

However, snake-based segmentation concentrates on isolation of the object boundary and therefore ignores motion information from the main region of the object. It is therefore more reliable to perform motion-based segmentation of the entire region bounded by the snake, and to use this information to refine the description of the motion and to predict the position of the object in the next image. The overall process is thus to feed the output of the snake boundary estimator into a motion-based segmenter and position predictor which reinitializes the snake for the next image—so both constituent algorithms perform the operations they are best adapted to. It is especially relevant that the snake has a good starting approximation in each frame, both to help eliminate ambiguities and to save on computation. The motion-based region segmenter operates principally by analysis of optical flow, though in practice, the increments between frames are not especially small: this means that while true derivatives are not obtained, the result is not as bedevilled by noise as it might otherwise be.

Various refinements were incorporated into the basic procedure:

- B-splines are used to smooth the outlines.
- The motion predictions are carried out using an affine motion model that works on a point-by-point basis. (The affine model is sufficiently accurate for this purpose if perspective is weak so that motion can be approximated locally by a set of linear equations.)
- A multiresolution procedure is invoked to perform a more reliable analysis of the motion parameters.
- Temporal filtering of the motion is performed over several image frames.
- The overall trajectories of the boundary points are smoothed by a Kalman filter (a basic treatment of Kalman filters is given in Section 20.8).

Before proceeding to set up an affine motion model, recall that an affine transformation is one which is linear in the coordinates employed. This type of transformation includes the following geometric transformations: translation, rotation, scaling, and skewing (see Chapters 6 and 19). Hence, the relevant affine motion model involves six parameters:

$$\begin{bmatrix} x(t+1) \\ y(t+1) \end{bmatrix} = \begin{bmatrix} a_{11}(t) & a_{12}(t) \\ a_{21}(t) & a_{22}(t) \end{bmatrix} \begin{bmatrix} x(t) \\ y(t) \end{bmatrix} + \begin{bmatrix} b_1(t) \\ b_2(t) \end{bmatrix} \tag{22.48}$$

This leads to an affine model of image velocities, also with six parameters:

$$\begin{bmatrix} u(t+1) \\ v(t+1) \end{bmatrix} = \begin{bmatrix} m_{11}(t) & m_{12}(t) \\ m_{21}(t) & m_{22}(t) \end{bmatrix} \begin{bmatrix} u(t) \\ v(t) \end{bmatrix} + \begin{bmatrix} c_1(t) \\ c_2(t) \end{bmatrix} \tag{22.49}$$

Once the motion parameters have been found from the optical flow field, it is straightforward to estimate the following snake position.

An important factor in the application of this type of algorithm is the degree of robustness it permits. In this case, both the snake algorithm and the motion-based region segmentation scheme are claimed to be relatively robust to partial occlusions: the abundance of available motion information for each object, the insistence on consistent motion, and the recursive application of smoothing procedures including a Kalman filter, all help to achieve this end. However, no specific nonlinear outlier rejection process is mentioned, which could help if two vehicles merged together and became separated later on or if total occlusion occurred.

Finally, the initial motion segmentation scheme locates the vehicles with their shadows since these are also moving (see Fig. 22.11); subsequent analysis seems able to eliminate the shadows and arrive at smooth vehicle boundaries.

22.9.2 THE SYSTEM OF KOLLER *ET AL.*

Another scheme for automatic traffic scene analysis was described by Koller *et al.* (1994). This contrasts with the system described above by placing heavy reliance on high-level scene interpretation through use of belief networks. The basic system incorporates a low-level vision system employing optical flow, intensity gradient, and temporal derivatives. These provide feature extraction and lead to snake approximations to contours; since convex polygons would be difficult to track from image to image (because the control points would tend to move randomly), the boundaries are smoothed by closed cubic splines having 12 control points: tracking is then achieved using Kalman filters. The motion is again approximated by an affine model, though in this case, only three parameters are used, one being a scale parameter and the other two being velocity parameters:

$$\Delta\mathbf{x} = s(\mathbf{x} - \mathbf{x}_{\mathrm{m}}) + \Delta\mathbf{x}_{\mathrm{m}} \tag{22.50}$$

Here, the second term gives the basic velocity component of the center of a vehicle region, and the first term gives the relative velocity for other points in the

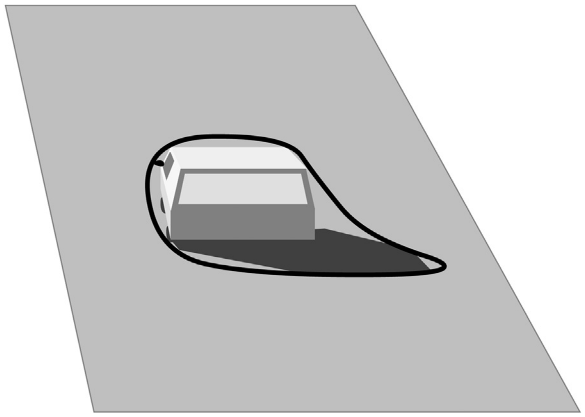

FIGURE 22.11

Vehicles located with their shadows. In many practical situations, shadows move with the objects that cause them, and simple motion segmentation procedures produce composite objects that include the shadows. Here, a snake tracker envelops the car and its shadow.

region, s being the change in scale of the vehicle ($s = 0$ if there is no change in scale). The rationale for this is that vehicles are constrained to move on the roadway and rotations will be small. In addition, motion with a component toward the camera will result in an increase in size of the object and a corresponding increase in its apparent speed of motion.

Occlusion reasoning is achieved by assuming that the vehicles are moving along the roadway, and are proceeding in a definite order, so that later vehicles (when viewed from behind) may partly or wholly obscure earlier ones. This depth ordering defines the order in which vehicles are able to occlude each other and appears to be the minimum necessary to rigorously overcome problems of occlusion.

As stated above, belief networks are employed in this system to distinguish between various possible interpretations of the image sequence. Belief networks are directed acyclic graphs in which the nodes represent random variables and arcs between them represent causal connections. In fact, each node has an associated list of the conditional probabilities of its various states corresponding to assumed states of its parents (i.e., the previous nodes on the directed network). Thus, observed states for subsets of nodes permit deductions to be made about the probabilities of the states of other nodes. The reason for using such networks is to permit rigorous analysis of probabilities of different outcomes when a limited amount of knowledge is available about the system. Likewise, once various outcomes are known with certainty (e.g., a particular vehicle has passed beneath a bridge), parts of the network will become redundant and can be removed: however, before removal, their influence must be "rolled up" by updating the probabilities for the remainder of the network. Clearly, when applied to traffic, the belief network has to be updated in a manner appropriate to the vehicles that are currently being observed; indeed, each vehicle will have its own belief network that will contribute a complete description of the entire traffic scene. However, one vehicle will have some influence on other vehicles, and special note will have to be taken of stalled vehicles or those making lane changes. In addition, one vehicle slowing down will have some influence on the decisions made by drivers in following vehicles. All these factors can be encoded into the belief network and can aid in arriving at globally correct interpretations. General road and weather conditions can also be taken into account.

Further work was planned to enable the vision part of the system to deal with shadows, brake lights and other signals, and a wide enough variety of weather conditions. Overall, the system was designed in a very similar manner to that of Bascle *et al.* (1994), though its use of belief networks made it rather more sophisticated.

In a later version of the system (Coifman et al., 1998), it was decided that a greater degree of robustness with regard to partial occlusion was required. Hence, the idea of tracking objects as a whole was abandoned and corner features were used for detection. This led to a different problem—that of grouping corner features to infer the presence of the vehicles, a process that was simplified by using a common motion constraint, so that features that were seen to be rigidly moving together were grouped together. The new version of the system also applied a homography between the image plane and the ground plane. The reason for this was to generate world parameters so that ground-based positions, trajectories, velocities, and densities could be established. Note for example that a vehicle traveling at constant speed on the road would have variable speed when viewed in an image. In addition, the right information could more easily be brought to bear when problems of partial or total occlusion are being investigated.

When designing a much later system, Magee (2004) made several interesting observations: (1) corner features are unreliable because of the small size of the objects of interest, (2) connected components analysis is a poor tool for combining parts of vehicles because of fragmentation and similarity of some object foreground points to background, and (3) particle filter trackers have high computational cost that does not scale linearly with the number of objects present—a serious matter when 30 or more vehicles in close proximity are being tracked simultaneously. He found that a sound way to track vehicles was to dynamically model vehicle invariants such as size, color, and speed: in other words, object appearance and recognition were important to systematic and accurate tracking; and the only way they could be achieved was by establishing a homography between the image and the ground plane. In that way, vehicle parameters properly became invariants as required. The homography is expressible as a nonlinear perspective transformation (or "inverse perspective mapping"), and some care is required in setting it up. (Note that such a mapping is mathematically valid only for points known to lie on the ground plane. When points not lying on the ground plane are back-projected to it, they give rise to weird, nonsensical effects, such as buildings that appear to lean backwards.) In fact, if the camera x-axis is horizontal, the homography only requires a rotation through an angle θ about the image x-axis, together with a scaling, in order to relate the image coordinates to the ground plane coordinates. Ignoring the scaling, there is only one parameter (θ) to be determined. Magee adopted the simple strategy of estimating θ as the angle required to make the roadway appear to have constant width, a procedure that proved to be adequate in his particular application (Fig. 22.12). The calculation was made sufficiently accurate by approximating the road centerlines and outlines by three polynomials and performing a fit by iteratively adjusting θ. The reason for adopting this procedure is that the roadway has no absolute predefined shape, so a heuristic approach seemed appropriate. Ideally, however, the ground truth for the road centerlines and outlines would be known and the value of θ could be adjusted to fit the ground truth without having to assume that the roadway has constant width.

22.10 LICENSE PLATE LOCATION

Over the past decade, there has been intensive effort to identify vehicles automatically by their license plates. Although license plates were introduced many years ago for the purpose of checking ownership and detecting stolen vehicles, nowadays two other important reasons for automatically identifying vehicles are (1) for

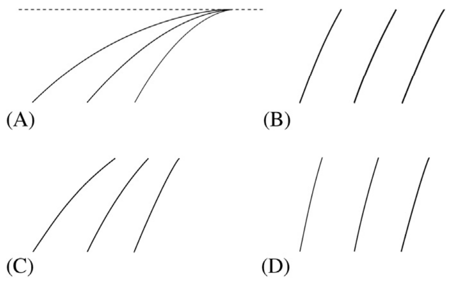

FIGURE 22.12

Adjusting the inverse perspective mapping of the roadway (A) shows the roadway as observed by the camera, (B) shows an inverse perspective mapping with the roadway adjusted for constant width, and (C) and (D) show cases of incorrect adjustment of the mapping.

taxation within tolling zones and (2) for exacting fines in the case of parking offenses—because considerable sums of money can be obtained in these ways with very little human intervention. Also, considering all the possible applications of computer vision in surveillance, identification of license plates represents a potentially straightforward application of current methodology. Nevertheless, there are many problems, not least because of the different styles of license plate from different countries.

Identification of license plates progresses through three main stages: (1) location and segmentation of the license plate, (2) segmentation of the individual characters, and (3) recognition of the individual characters. Here, we concentrate on the first of these stages, as the other two are more specialized and less generic, considering the different styles, fonts, and character sets in use in different countries. In any case, the first stage is probably the most difficult to engineer.

A priori, it might be thought that the best way of locating license plates would be via their colors, which are generally well specified for each country. However, many problems arise from variations in ambient lighting, particularly with the seasons, the weather and time of day, while shadows are also a source of difficulty. In this milieu, one of the best starting points has been found to be use of a simple Sobel or other vertical edge-detection operator, in conjunction with horizontal nonmaximum suppression and thresholding. This has been found to locate not only the vertical lines at the ends of the number plates but also the vertical lines at the sides of the characters (Zheng et al., 2005). This generally gives a relatively dense set of vertical edges within the region of the license plate. To proceed further, long background edges and short noise edges are eliminated. Finally, moving a rectangle of license plate size over the image and counting the edge pixels within it turns out to be a highly reliable way of locating license plates (in fact, this process is a form of correlation). The whole process is shown in Fig. 22.13, with the difference that in the case shown, the final stages are

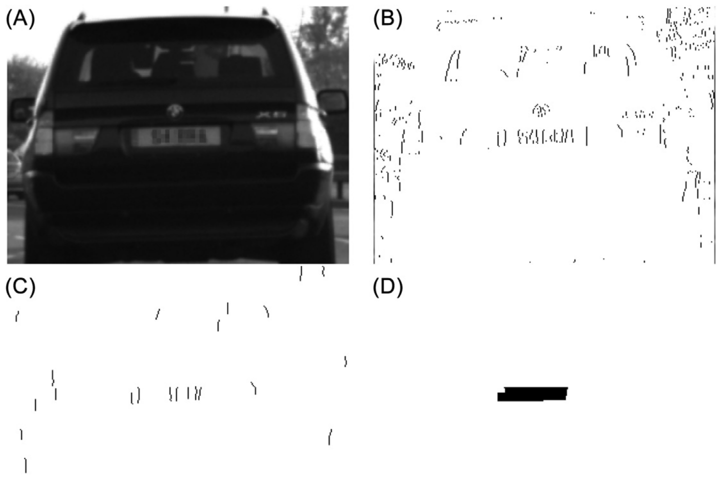

FIGURE 22.13

Simple procedure for locating license plates. (A) Original image with license plate pixelated to prevent identification. (B) Vertical edges of original image. (C) Vertical edges selected for length. (D) Region of license plate located by horizontal closing followed by horizontal opening, each over a substantial distance (in this case 16 pixels).

carried out solely using morphological operations (horizontal closing followed by horizontal opening, in each case by 16 pixels).

This method has been developed considerably further by Abolghasemi and Ahmadyfard (2009) using color and texture cues. They found that a particular advantage of color object analysis is robustness to viewpoint changes. They also used morphological closing to link all the vertical edge points and followed this by opening to eliminate the effects of isolated noise points.

Before characters can be segmented and recognized, another stage is needed—that of license plate distortion correction. This arises because license plates may not be observed from the most ideal viewpoint. This is something that requires careful attention. If vehicles are too far away from the camera, the resolution will be too low to permit vertical edges to be found; likewise, accurate identification of the characters will not be possible. If the license plate is viewed obliquely, it will appear misorientated and will not even appear rectangular. However, if license plates were always viewed at a particular distance and location, a standard perspective transformation could be applied to correct such distortions. While it is acknowledged in the literature (e.g., Chang et al., 2004) that adding such a step would improve the performance of license number recognition, few systems seem to incorporate such a step. The reason is probably that OCR (optical character recognition) systems are already very accurate even when characters are slightly sheared and rotated.

22.11 OCCLUSION CLASSIFICATION FOR TRACKING

It will be clear from the many remarks made about occlusion on the preceding pages that this is a serious problem that needs in-depth analysis and careful algorithm design, particularly with regard to people tracking. To this end, Vezzani and Cucchiara (2008) and Vezzani et al. (2011) have made a careful analysis of the means by which occlusion can arise, starting with the definition of *nonvisible regions* as the parts of objects that are not visible in the current frame. They proceeded to classify these as "dynamic," "scene," or "apparent" occlusions:

1. *Dynamic occlusions* are due to moving objects that are readily identified.
2. *Scene occlusions* are due to static objects that are part of the background, but which can nevertheless be in front of moving objects.
3. *Apparent occlusions* are sets of pixels that arise from shape variations of objects being tracked.

Here, it is important to note the distinction between background and foreground. To the layman, "background" merely means a backdrop in front of which the actors perform: it is regarded as static, while the foreground is considered to consist of more interesting moving objects. However, in computer vision, we have to consider the background as static wherever it is, with the moving "foreground" objects ranged at different distances from the camera and sometimes moving *behind* background objects (Fig. 22.14). Note that the background that is identified by background modeling algorithms is the static part of the scene. Of course, a complication that can disrupt this tidy situation is that the background may be composed partly of objects that have come to rest, either permanently or temporarily, and it will be up to the vision algorithm to consider the available evidence from watching the scene and to assess the various possibilities and probabilities.

Another factor to consider is whether occlusions are partial or total. For many static scenes and static situations, total occlusion is an eventuality that is normally disregarded; hence, all occlusions are taken to be partial, and they are simply referred to as "occlusions." However, when tracking objects, total occlusion is a possibility that has to be borne in mind indefinitely (though in practical situations a time limit may have to be set).

So, when viewing image sequences containing motion, objects may temporarily be totally occluded, *or* they may be partially occluded—in which case, they may be broken into several sections. And when the objects reemerge later on, these sections need to be reassembled into whole objects: this scenario arises when a person passes behind a table, for example. In addition, when a person passes behind a low fence, and the lower body is temporarily invisible, it is necessary for the model of the complete person to be remembered; for if the model becomes adapted to the changed situation, it may not be able to cope properly when the whole person reemerges and it will continue to track only the top of the

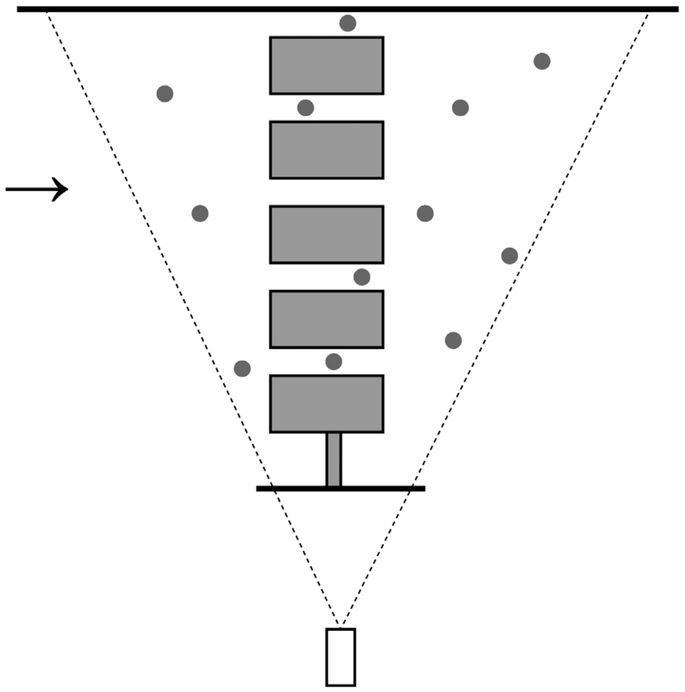

FIGURE 22.14

Typical situation of occlusion. This illustrates the case of turnstiles leading to underground trains, viewed from the side. The dots represent people (moving in the direction of the arrow) ranged at different distances from the camera.

body. Clearly, to cope successfully with such situations (Fig. 22.15), the computer needs to have the means to deal with them holistically. Similarly, when two people merge into a larger blob when walking together, the computer will need to have the means to recall that two people were involved so that their identities will be preserved and reinstated when they separate again. Thus, we require substantial intelligence to be incorporated into tracking algorithms.

The various components that have to be incorporated into the algorithm would appear to be the following: (1) the usual background extraction capability, (2) the usual blob tracking capability, (3) full appearance and identity recall, (4) merge capability, (5) split capability, and (6) probabilistic analysis of interpretations. Indeed, (6) will probably have to be the unifying force that drives the whole algorithm.

All these aspects are included in the work of Vezzani and Cucchiara (2008) and Vezzani et al. (2011). In particular, they employ an appearance-based formalism that integrates the possible shape variations of each object and represents them by probabilistic maps. This means that when part of an object is obscured,

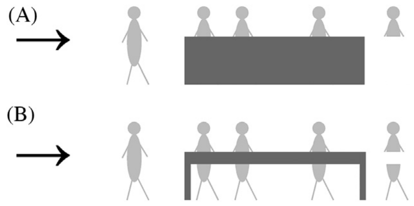

FIGURE 22.15

Further examples of occlusion. (A) Case of people walking behind a fence or barrier, potentially resulting in only the head and shoulders being tracked afterward. (B) Case of people walking behind a table, potentially resulting in two parts of the body being tracked independently afterward.

its shape model hallucinates the whole of the object in the probabilistic shape it ought to have, so that when it reemerges, it is automatically reintegrated virtually instantaneously into its natural form.

So far, we have not examined item 3 (see the beginning of this section), which mentioned apparent partial occlusions due to changes in shape. These arise because as a body rotates or bends slightly, or otherwise deforms, new parts will become visible though other parts will become invisible. While these could be regarded as arising through self-occlusion, this may not be the only possibility, e.g., if stretching is involved. We shall not delve further into this point, but merely underline that apparent partial occlusions are not caused by any other objects. This is quite an innovative observation relating to occlusion and may be part of the reason why progress with occlusion has been drawn out over many years. Suffice it to say that the work of Vezzani et al. represents a sound and impressive advance through its cognizance of the many aspects of occlusion in tracking scenarios.

The overall system is very robust and fast and is well able to cope with upward of 40 people in videos from the PETS2006 dataset. Nevertheless, it gives rise to some failures, which devolve into the following categories: (1) identity change of one person, (2) split head/feet, (3) incorrect splitting of groups containing two or three people, and (4) identity change of luggage. In fact, it appears that these are failures not of the *overall* system, including the handling of objects and occlusions, but of the part of the system handling appearance, which is arguably the part to which relatively little design effort has been devoted. Furthermore, it is not clear from the two papers whether a human observer could have performed better using the same video input. Nevertheless, the way forward, including the ability to handle problems (2) and (3) above, is probably to enhance the system using stick-figure-based models of humans, which can take proper account of limb articulation constraints (see the following section): the performance of a system that looks at the body as a whole and models it as a holistic probabilistic shape profile must in the end be limited without suitable enhancement.

22.12 DISTINGUISHING PEDESTRIANS BY THEIR GAIT

This section outlines a method for distinguishing pedestrians by their gait. Clearly, unlike many other moving objects such as vehicles, pedestrians have cyclical motions, and it is actually possible to recognize individual people by their gait. However, here, we consider only the methodology needed to locate pedestrians in image sequences.

The basis of the approach is to perform spatiotemporal differencing operations, in which spatiotemporal averaging is followed by temporal differencing. This "motion distillation" method (Sugrue and Davies, 2008) is implemented as a Haar wavelet and leads to a nonbinary motion map of the video at each time-step, according to the equation:

$$W = \sum_{t=t_0}^{t}\sum_{i}\sum_{j} x_{tij} - \sum_{t=t_1}^{t}\sum_{i}\sum_{j} x_{tij} \tag{22.51}$$

where x_{tij} represents the video pixel data at the point (t, i, j) in spatiotemporal space.

In this method, undesirable contrast dependence is removed by normalizing W values across the detected object: the process involves taking the ratio R of positive (W_+) to negative (W_-) filter outputs:

$$R = \frac{\sum |W_+|}{\sum |W_-|} \tag{22.52}$$

For a rigid object that retains its orientation relative to the camera, R will remain approximately constant over time. On the other hand, pedestrians deform as they move and can quickly be detected by testing for changes, and particularly oscillations in the "rigidity parameter" R.

Fig. 22.17A compares the motion signals R of a typical vehicle and a pedestrian (see Fig. 22.16 for typical frames from the original videos). While the vehicle signal changes only gradually as a result of slight rotation, changing perspective, and noise, the pedestrian signal is highly variable and oscillatory because of gait motion. Note that over the period shown in Fig. 22.17A, the area of the vehicle changes by a factor ~ 10, while the area of the pedestrian changes by only a few percent. This makes it all the more significant that R is so constant for the vehicle, demonstrating that it is a useful invariant of the motion.

After detection, a pedestrian's motion field can be further analyzed for a variety of behavior patterns. Normal behavior can be modeled by fitting a rectangular box to the subject's motion field. The rectangle is the full height of the figure with a width typically set at half the height (see below). The total motion area A is calculated as

$$A = \sum |W_+| + \sum |W_-| \tag{22.53}$$

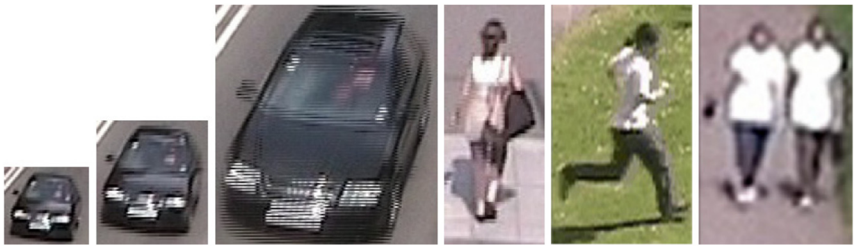

FIGURE 22.16

Portions of frames extracted from video sequences by motion detector. Left: Three frames of moving vehicle. Right: Respective frames of pedestrian, runner and group of walkers.

© IET 2007.

where the sums are taken over the whole object; in addition, the corresponding area A_{ex} is calculated for the region outside the box. The box parameter η is then defined as the ratio of the two areas:

$$\eta = \frac{A_{ex}}{A} \tag{22.54}$$

This parameter should also be an invariant both for rigid motion and for "compact" motion where A_{ex} is small, giving a measure of the type of behavior: this is because η is dimensionless and compares like with like but still contrasts two motions (viz., exterior to the box and overall). If the pedestrian is walking normally, the η value will be low at all times (see, e.g., the bottom trace in Fig. 22.17A): the higher values typical of runners are demonstrated clearly in Fig. 22.17B. In addition, individual sudden actions such as waving and jumping will result in spikes in η.

A third type of invariant R_{ex} has also been developed to help discriminate other more complex cases. This has the same definition as for R, except that it applies only for the part of the object external to the box. It provides additional useful information helping to discriminate runners from groups of walkers (see Fig. 22.16). Both of these categories have been found to have values of η around 0.5, so R_{ex} is useful in enabling them to be discriminated. (Specifically, $R_{ex} \approx 1.5R$ for runners and $R_{ex} \approx R$ for groups of walkers, though additionally R and R_{ex} are only well synchronized for runners.) While the R_{ex} information cannot be described as being specific to groups, it is nevertheless valuable, though ultimately, only detailed analysis leading for example to stick-figure models of people may provide the information that is required in a particular application (see the following section).

Because of the importance of box size in determining the values of η and R_{ex}, a careful study was made to optimize discrimination between single walkers and runners. This gave the optimum box width/height ratio as close to 0.5, for which the walker–runner threshold was best set at about 0.1 (see Fig. 22.18).

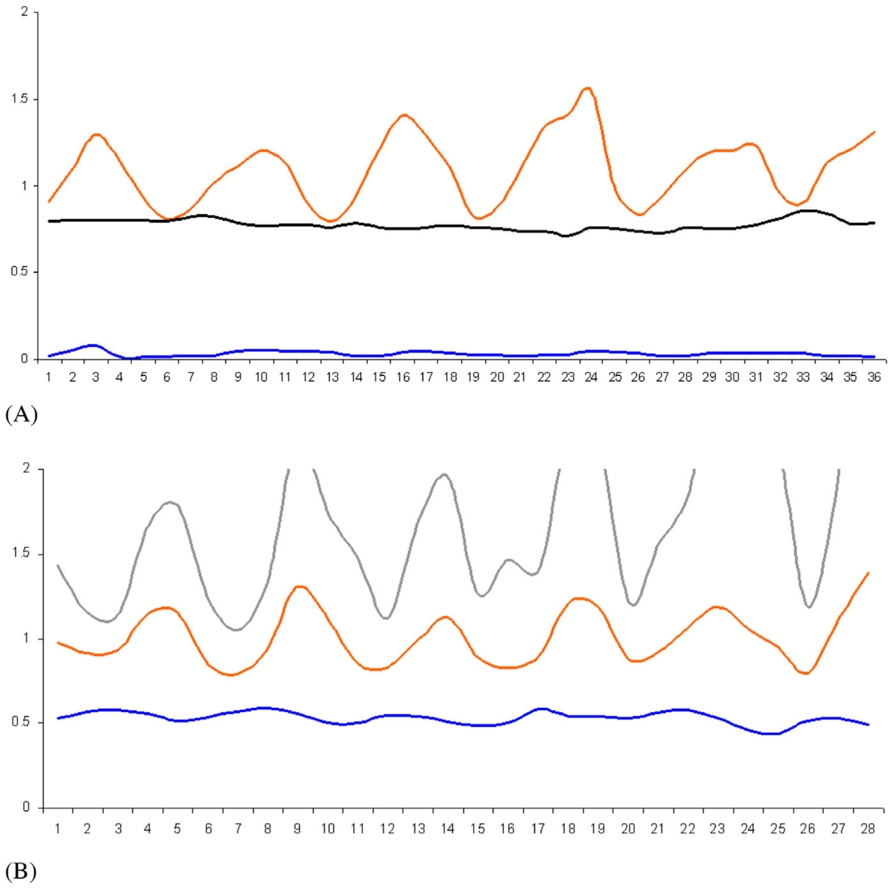

FIGURE 22.17

Motion analysis using rigidity and box parameters. (A) Top to bottom: result of applying the rigidity parameter R for the pedestrian; and result of applying R for the vehicle; result of applying box parameter η for the pedestrian. The horizontal scales indicate video frames. (B) Top to bottom: result of applying the respective parameters R_{ex}, R, η for the runner. In all cases the originals are shown in Fig. 22.16.

© IET 2007.

Overall, the methods described here have been found to distinguish motions of rigid and nonrigid objects with ~97% accuracy. They are also able to classify single walkers with ~95% accuracy, and runners and groups of walkers with ~87% accuracy; in addition, they give useful indications of “extravagant” activities such as waving and jumping. Interestingly, all this was achieved via use of specially designed invariants, which save complexity and computation while being straightforward to set up and adjust.

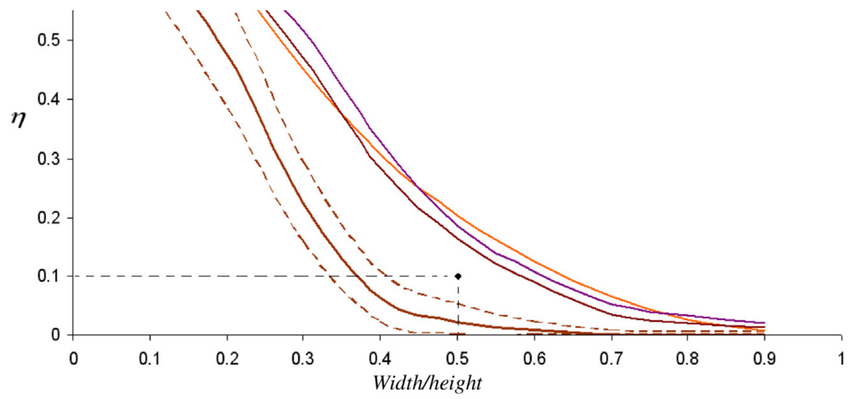

FIGURE 22.18

Discrimination permitted by box parameter η. The lower solid line represents the mean value of a sample of walkers (the broken lines indicate $\pm\sigma$ error bars), and the upper solid lines record runners. To discriminate between walkers and runners, the best operating point is close to (0.5, 0.1), as shown.

© IET 2007.

22.13 HUMAN GAIT ANALYSIS

For several decades, human motion has been studied using conventional cinematography. Often, the aim of this work has been to analyze human movements in the context of various sports—in particular, tracking the swing of a golf club and thus helping the player to improve his game. To make the actions clearer, stroboscopic analysis coupled with bright markers attached to the body have been employed and have resulted in highly effective action displays. In the 1990s, machine vision was applied to the same task. At this point, the studies became much more serious and there was increased focus on accuracy. The reason for this was a widening of the area of application not only to other sports but also to medical diagnosis and to animation for modern types of film containing artificial sequences.

Because high accuracy is needed for many of these purposes—not least measuring limps or other imperfections of human gait—analysis of the motion of the whole human body in normally lit scenes proved insufficient, and body markers remained important. Typically, two are needed per limb, so that the 3D orientation of each limb is deducible. Some work has been done to analyze human motions using single cameras, but the majority of the work employs two or more cameras: multiple cameras are valuable because of the occlusion that occurs when one limb passes behind another, or behind the body.

To proceed with the analysis, a kinematic model of the human body is required. In general, such models assume that limbs are rigid links between a limited number of ball and socket joints, which can be approximated as point junctions between stick limbs. For example, one such model (Ringer and Lazenby, 2000) employs two rotation parameters at the point where the hips join the backbone, three for the joint where the thigh bone joins the hips, plus one for the knee, and another for the ankle. Thus, each leg has seven degrees of freedom, two of these being common (at the backbone): this leads to a total of 12 parameters covering leg movements (Fig. 22.19). Clearly, it is part of the nature of the skeleton that the joints are basically rotational, though there is some slack in the system, especially in the shoulders, while the knees have some lateral freedom. Finally, the whole situation is made more complex by constraints such as the inability of the knee to extend the lower leg too far forward.

Once a kinematic model has been established, tracking can be undertaken. It is relatively straightforward to identify the markers on the body with reasonable accuracy. The next problem is to distinguish one marker from another and to label them. Considering the huge number of combinations of labels that are possible, and the frequency with which occlusions of parts of a leg or arm are bound to take place, special association algorithms are required for the purpose. These include the Kalman filter that helps to predict how unseen markers will move until they come back into view. Such models can be improved by including acceleration parameters as well as position and velocity parameters (Dockstader and Tekalp, 2002). Their model is not merely theoretically deduced: it has to be trained, typically on sequences of 2500 images each separated by 1/30 seconds. In

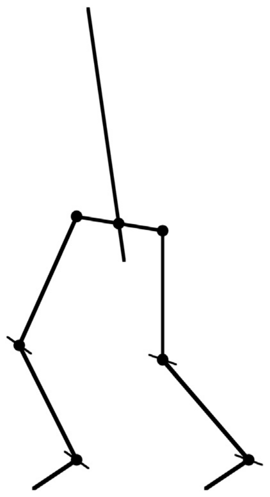

FIGURE 22.19

Stick-skeleton model of the lower human body. This model takes the main joints on the skeleton as being universal ball-and-socket joints, which can be approximated by point junctions—albeit with additional constraints on the possible motions (see text). Here, a thin line through a joint indicates the single rotational axis of that joint.

addition, the stick model of each human subject has to be initialized manually. Considerable training is necessary to overcome the slight inaccuracies of measurement and to build up the statistics sufficiently for practical application when testing. Errors are greatest when measuring hand and arm movements, because of the frequent occlusions they are subject to.

Overall, articulated motion analysis involves complex processing and a lot of training data. It is a key area of computer vision and the subject is evolving rapidly. It has already reached the stage of producing useful output, but accuracy will improve over the next few years and this will set the scene for practical medical monitoring and diagnosis, completely natural animation, detailed help with sports activities at affordable costs, not to mention recognition of criminals by their characteristic gaits. Certain requirements—such as multiple cameras—will probably remain, though the trend to markerless monitoring can be expected to continue. For further information, the reader could start by referring to the monograph by Nixon et al. (2006).

22.14 MODEL-BASED TRACKING OF ANIMALS

This section is concerned with the care of farm animals. Good stockmen notice many aspects of the behavior of the animals and learn to respond to them. Fighting, bullying, tail biting, activity, resting behavior, and posture are useful indicators of states of health, potential lameness, or heat stress, while group behavior may indicate the presence of predators or human intruders. In addition, feeding behavior is all important, as is the incidence of animals giving birth or breaking away from the confinement of the pen. In all these aspects, automatic observation of animals by computer vision systems is potentially useful.

Some animals such as pigs and sheep are lighter than their usual backgrounds of soil and grass, and thus they can in principle be located by thresholding. However, the backgrounds may be cluttered with other objects such as fences, pen walls, drinking troughs, and so on—all of which can complicate interpretation. Thus, straightforward thresholding will rarely work well in normal farm scenes. McFarlane and Schofield (1995) tackled this problem by background subtraction. They used a background image obtained by temporal median filtering for a whole range of images taken over a fair period: during this process, care was taken to mask out regions where piglets were known to be resting. Their algorithm modeled piglets as simple ellipses and achieved fair success in its task of monitoring the animals.

We next examine the more rigorous modeling approach adopted by Marchant and Onyango (1995) and developed further by Onyango and Marchant (1996) and Tillett *et al.* (1997). These workers aimed to track movements of pigs within a pen by viewing them from overhead under not very uniform lighting conditions. The main aim of the work at this early stage was tracking the animals, though, as indicated above, it was intended to lead on to behavioral analysis in later work. To find the animals, some form of template matching is required. Shape matching is an attractive concept, but with live animals such as pigs, the shapes are highly variable: specifically, animals which are standing up or walking around will bend from side to side and may also bend their necks sideways or up and down as they feed. It is insufficient to use a small number of template masks to match the shapes, as there is an infinity of shapes related by various values of the shape parameters mentioned. These parameters are additional to the obvious ones of position, orientation, and size.

Careful trials showed that matching with all these parameters is insufficient, as the model is quite likely to be shifted laterally by variations in illumination: If one side of a pig is closer to the source of illumination, it will be brighter, and hence the final template used for matching will also shift in that direction. The resulting fit could be so poor that many possible "goodness of fit" criteria will deny the presence of a pig. These factors mean that possible variations in lighting have to be taken into account in fitting the animal's intensity profile.

A rigorous approach involves principal components analysis (PCA). The deviation in position and intensity between the training objects and the model at a series of carefully chosen points is fed to a PCA system: the highest energy eigenvalues indicate the main modes of variation to be expected; then any specific test example is fitted to the model and amplitudes for each of these modes of variation are extracted, together with an overall parameter representing the goodness of fit. Unfortunately, this sort of approach is highly computation intensive because of the large number of free parameters; in addition, the position and intensity parameters are disparate measures that require quite different scale factors to be used to coax the schema into working. This means that some means is required for decoupling the position and intensity information. This is achieved by performing two independent PCAs in sequence—first on the position coordinates and then on the intensity values.

When this procedure is carried out, three significant shape parameters are found, the first being lateral bending of the pig's back, accounting for 78% of the variance from the mean; and the second being nodding of the pig's head: as the latter corresponded to only $\sim$20% of the total variance, it was ignored in later analysis. In addition, the gray-level distribution model had three modes of variation amounting to a total of 77% of the intensity variance: the first two modes corresponded to (1) a general amplitude variation in which the distribution is symmetrical about the backbone, and (2) a more complex variation in which the intensity distribution is laterally shifted relative to the backbone (this arises largely from lateral illumination of the animal)—see Fig. 22.20.

While PCAs yield the important modes of variation in shape and intensity, in any given case the animal's profile has still to be fitted using the requisite number of parameters—one for shape and two for intensity. The Simplex algorithm (Press *et al.*, 1992) proved effective for this purpose. The objective function to be minimized to optimize the fit takes account of (1) the average difference in intensity

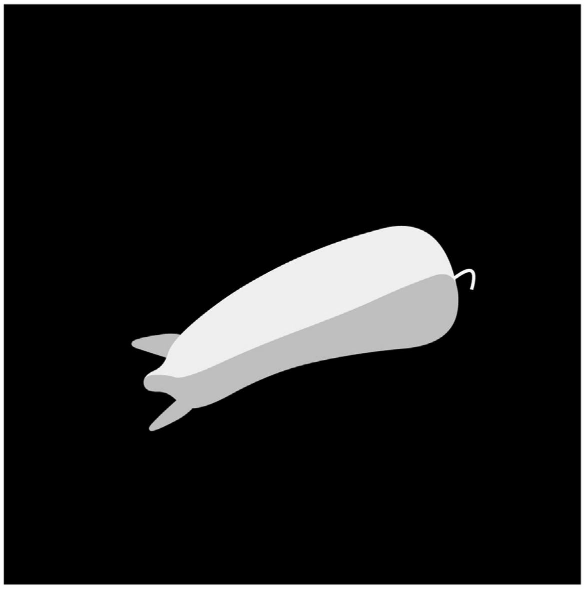

FIGURE 22.20

Effect of one mode of intensity variation found by PCA. This mode clearly arises from lateral illumination of the pig.

© World Scientific 2000.

between the rendered (gray-level) model and the image over the region of the model, and (2) the negative of the local intensity gradient in the image normal to the model boundary averaged along the model boundary (the local intensity gradient will be a maximum right around the animal if this is correctly outlined by the model).

One crucial factor has been skirted around in the preceding discussion: that the positioning and alignment of the model to the animal must be highly accurate (Cootes *et al.*, 1992). This applies both for the initial PCA and later when fitting individual animals to the model is in progress. Here, we concentrate on the PCA task. When using PCA, it should be borne in mind that it is a method of characterizing deviations: this means that the deviations must already be minimized by referring all variations to the mean of the distribution. Thus, it is very important when setting up the data to bring all objects to a common position, orientation, and scale before attempting PCA. In the present context, the PCA relates to shape analysis, and it is assumed that prior normalizations of position, orientation, and scale have already been carried out. (Note that in more general cases, scaling may be included within PCA if required. However, PCA is a computation intensive task, and it is best to encumber it as little as possible with unnecessary parameters.)

Overall, the achievements outlined above are notable, particularly in the effective method for decoupling shape and intensity analysis. In addition, the work holds significant promise for application in animal husbandry, demonstrating that animal monitoring and ultimately behavioral analysis should be attainable with the aid of computer vision.

22.15 CONCLUDING REMARKS

This chapter has shown something of the purpose of surveillance—which is largely to do with monitoring the behavior patterns of people and vehicles on roads and precincts. It has also shown a number of the principles and methods by which surveillance may be implemented: these include identification and elimination of background, detection and tracking of moving objects, identification of the ground plane, occlusion reasoning, Kalman and particle filtering, capability for modeling complex motions including those of articulated objects, and use of multiple cameras for widening coverage in time and space.

Over time, some specialized application areas have appeared, such as location and identification of license plates, identification of vehicles exceeding the speed limit, human gait analysis, and even animal tracking: the chapter has aimed to indicate how all these can be achieved. Early methods included Kalman filters and chamfer matching, and later ones included particle filters, which rely on a probabilistic approach to tracking. Particle filters have come a long way, but it is doubtful whether they can go much further if based on probability assessment alone, as it is clear that humans bring to bear huge databases of relevant information when tracking moving objects.

An important lesson is that detection and tracking are distinct, complementary functions, and there is no reason why the same algorithms will be optimal for both. As indicated in Section 22.3.3, foreground detection requires the application of a suitable foreground model, or else a bootstrapping process involving an "exception to background" procedure. But once detection has been achieved, tracking can in principle proceed as a much simpler, more blinkered process. It remains to be seen whether future work will find ways of streamlining the detection + tracking model. Most likely, it will be found lacking, because objects such as people radically alter in appearance as they walk by; hence, it is more natural to have both processes working in parallel (not necessarily at constant rates) all the time, rather than being applied serially. The same applies when a guided missile approaches a tank but can be misled into tracking a different object in the background as the scale of the target radically changes by several orders of magnitude: again the tracking algorithm needs to be monitored by a continuously acting detection algorithm. These points are labored because detection and tracking are at the core of surveillance, whatever the application area, and are thus very much the generic backcloth to this chapter.

While the chapter has covered the situation of static cameras being used to monitor moving objects, the following chapter covers the intrinsically more complex case of in-vehicle vision systems, where moving cameras are used to monitor both stationary and moving objects. This will call for a radical rethink of vision system strategy, because all parts of the scene will be eternally shifting and changing, and it will generally not be possible to rely on relatively trivial preliminary identification of a stationary background.

This chapter has shown that surveillance is largely about the detection and tracking of moving objects, and that different types of algorithm will often be needed to achieve each of these functions. In most cases, locating the ground plane is a necessary first step in the analysis, while occlusion reasoning, Kalman filtering, capability for modeling complex motions, and multiple cameras will often be needed to achieve the ultimate aim of analyzing developing behavior patterns.

22.16 BIBLIOGRAPHICAL AND HISTORICAL NOTES

As we have seen, surveillance involves many factors, from 3D to motion, but paramount amongst these is the tracking of moving objects—in particular, vehicles and people. For some years, tracking meant the use of Kalman filters, but the deficiencies of this approach led in the 1990s to the development of particle filters, including particularly the work of Isard and Blake (1996, 1998), Pitt and Shepherd (1999), van der Merwe et al. (2000), Nummiaro et al. (2003), Nait-Charif and McKenna (2004, 2006), Schmidt et al. (2006), and many others. Much of the early work is summarized in a tutorial paper by Arulampalam et al. (2002), though Doucet and Johansen (2011) have justifiably felt it necessary to produce another. The first of these and a 2008 preprint of the second might better be called reviews than tutorials, as the going is difficult—partly because of the lack of explanatory figures—and often it is easier to appeal to the original works than to them.

In parallel with these developments, much work took place on background modeling using both parametric and nonparametric methods: See for example Elgammal et al. (2000). Cucchiara et al. (2003) helped by defining the stationary and transient background problems and by clarifying the problem of “ghosts.” Shadows have been a source of problems over the whole period, not least because they can be static or moving, and also because they can fall on static or moving objects: Elgammal et al. (2000) and Prati et al. (2003) carried out seminal work on this topic.

Khan and Shah (2000, 2003, 2009) were responsible for a thoroughgoing approach to the tracking of people both with single and with multiple cameras, this work being followed up by Eshel and Moses (2008, 2010) who found how to make good use of top-of-head tracking in crowd scenes. Pflugfelder and Bischof (2008, 2010) developed the approach to cover nonoverlapping views—a task that had previously (Makris et al., 2004) been solved with some degree of generality, but without knowledge of scene geometry, by learning transition probabilities for objects passing between views.

Vezzani and Cucchiara (2008) and Vezzani et al. (2011) made a careful analysis of the means by which occlusions can arise, and this enabled them to devise algorithms that cope better with temporary partial or full occlusions or temporary merging of moving objects—in the sense of not getting confused, and recovering faster from such events.

Work on traffic monitoring has stretched over many years (e.g., Fathy and Siyal, 1995; Kastrinaki *et al.*, 2003). Early work on the application of snakes to tracking was carried out by Delagnes *et al.* (1995); on the use of Kalman filters for tracking by Marslin *et al.* (1991); and on the recognition of vehicles on the ground plane by Tan *et al.* (1994). For details of belief networks see Pearl (1988). Note that corner detectors (Chapter 6: Corner, Interest Point, and Invariant Feature Detection) have also been widely used for tracking: See Tissainayagam and Suter (2004) for an assessment of performance.

A huge amount of work has been carried out on the analysis of human motions (Aggarwal and Cai, 1999; Gavrila, 1999; Collins *et al.*, 2000; Haritaoglu *et al.*, 2000; Siebel and Maybank, 2002; Maybank and Tan, 2004). See Sugrue and Davies (2007) for a simple method of distinguishing pedestrians. However, note that rigorous analysis of human motion involves studies of articulated motion (Ringer and Lazenby, 2000; Dockstader and Tekalp, 2001), one of the earliest enabling techniques being that of Wolfson (1991). As a result, a number of workers have been able to characterize or even recognize human gait patterns (Foster *et al.*, 2001; Dockstader and Tekalp, 2002; Vega and Sarkar, 2003): See Nixon et al. (2006) for a recent monograph on the subject. A particular purpose for this type of work has been the identification and avoidance of pedestrians from moving vehicles (Broggi *et al.*, 2000; Gavrila, 2000). Much of this work has its roots in the early farsighted paper by Hogg (1983), which was later followed up by crucial work on eigenshape and deformable models (Cootes *et al.*, 1992; Baumberg and Hogg, 1995; Shen and Hogg, 1995). Gavrila's work on pedestrian detection (Gavrila, 1998, 2000) used chamfer matching, while Leibe et al. (2005) developed the method further, albeit with the help of a minimum distance length top-down segmentation scheme capable of handling multiple hypotheses.

In our concentration on complex topics such as articulated motion and complications caused by occlusion, it is important not to lose sight of simple but elegant developments such as histograms of orientated gradients (HOGs), which have only appeared relatively recently (Dalal and Triggs, 2005). These were designed for, and are well-matched to, the detection of human shapes. Basically, they focus on the straight limbs of the human body, which have many edge points aligned along the same direction—though the latter will naturally change with walking or other motions. The basis of the method is to divide the image into "cells" (sets of pixels) and to produce orientation histograms for each of them. Voting into the orientation histogram bins takes place with weighting proportional to gradient magnitude. To provide strong illumination invariance, a robust normalization method is used. The cells are combined into larger overlapping blocks in several ways, with the result that some of the blocks end up with larger signals indicating the presence of human limbs. However, the result is curiously that the HOG detectors cue mainly on silhouette contours and emphasize the head, shoulders, and feet. In a later paper, Dalal, Triggs, and Schmid (2006) combined the HOG detector with motion detectors and were able to achieve even better results (motion detection improved the false alarm rate by a factor of 10 relative to the best appearance-based detector). An interesting feature of the HOG approach is that it outperforms wavelet analysis because the latter eliminates vital abrupt edge information by prematurely blurring the image data.

Overall, work on surveillance has stretched over many years but has vastly accelerated since the mid-1990s as workers have had access to more powerful computers that made it realistic to think of real-time implementation, both for experimentation and for on-road systems. Note that the past few years have seen developments in real-time systems involving FPGAs (field programmable gate arrays)—a trend that was already present circa 2000—and GPUs (graphics processing units)—a trend that is especially recent and has arisen as a result of natural interaction between the video games industry and computer vision.

22.16.1 MORE RECENT DEVELOPMENTS

Amongst the most recent works, Kim et al. (2010) have proposed a robust method for recognizing humans by their gait by using a hierarchical active shape model. The approach is novel in that it is prediction based and overcomes the drawbacks of existing methods by extracting a set of model parameters instead of directly analyzing the gait. Feature extraction proceeds by motion detection, object region detection, and Kalman prediction of the active shape model parameters. The method is able to alleviate tasks such as background generation, shadow removal, and obtaining high recognition rates. Ramanan (2006) has obtained good results by a new iterative parsing method for analyzing motions of articulated bodies ranging from humans playing games to horses frolicking and cantering. The approach has the advantage of being generic and does not depend on the location of skin or human faces. Lian et al. (2011) have obtained impressive performance when tracking pedestrians between camera view separations of more than 20 m—much greater than the separations ~ 4 m obtained by Pflugfelder and Bischof (2008, 2010).

Ulusoy and Yuruk (2011) have analyzed the problems of fusing data from visual and thermal images in order to make good use of their complementary properties to improve overall performance. They show that fusion should lead to a better recall rate (fewer false negatives) but, at the same time, result in a decrease in precision rate (more false positives); they also note that the infrared (thermal) domain always has higher precision (the underlying reasons for these observations are that thermal images effectively provide the *foreground* information containing the object pixels). In fact, it is only worth attempting fusion when an improved recall rate is required. This paper presents a more efficient method for fusing the data from the two domains and at the same time obtaining recall rates better than those previously obtained. The method was tested on outdoor images of human groups including those from a well-known database. The work in this paper leans heavily on the earlier work of Davis and Sharma (2007). Both papers refer to thermal imagery in spite of the title of the first which refers to "infrared" images.

22.17 PROBLEM

1. When an inverse perspective mapping onto the ground plane is carried out, points on the ground plane are well represented in the new representation. Explain why this does not apply for buildings or people, and why they always appear to lean backward when presented in this representation.