INVITED PAPER

Object Detection, Tracking and Recognition for Multiple Smart Cameras

Efficient distributed algorithms defined for small networks of fixed cameras may be adaptable to larger networks with mobile, steerable cameras.

By Aswin C. Sankaranarayanan, *Student Member IEEE*, Ashok Veeraraghavan, *Student Member IEEE*, and Rama Chellappa, *Fellow IEEE*

ABSTRACT | Video cameras are among the most commonly used sensors in a large number of applications, ranging from surveillance to smart rooms for videoconferencing. There is a need to develop algorithms for tasks such as detection, tracking, and recognition of objects, specifically using distributed networks of cameras. The projective nature of imaging sensors provides ample challenges for data association across cameras. We first discuss the nature of these challenges in the context of visual sensor networks. Then, we show how real-world constraints can be favorably exploited in order to tackle these challenges. Examples of real-world constraints are a) the presence of a world plane, b) the presence of a three-dimiensional scene model, c) consistency of motion across cameras, and d) color and texture properties. In this regard, the main focus of this paper is towards highlighting the efficient use of the geometric constraints induced by the imaging devices to derive distributed algorithms for target detection, tracking, and recognition. Our discussions are supported by several examples drawn from real applications. Lastly, we also describe several potential research problems that remain to be addressed.

KEYWORDS | Detection; distributed sensing; geometric constraints; multiview geometry; recognition; smart cameras; tracking

I. INTRODUCTION

Video cameras are fast becoming ubiquitous for a wide range of applications including surveillance, smart video conferencing, markerless human motion capture, animation transfer, and even some critical tasks such as assisted surgery. Some of the challenges in building applications for single video cameras have been studied for more than a decade, and reliable algorithms for many tasks have been realized. Looking ahead, the challenge is to make these algorithms and applications robust in the presence of several cameras (possibly even several hundreds) that are networked.

Distributed sensing using a host of networked *smart* cameras raises challenges that can be broadly clustered in two main areas.

- *Distributed Sensing.* The image obtained by each camera depends on its position and orientation, and hence, in general, is different from that of other cameras observing the same scene. This raises the need for designing algorithms that can efficiently fuse the evidence available at each of these cameras into a consistent and robust estimate. Specifically, since we are interested in object detection, it is important to determine whether an object is present. If an object is present within the field of view, it is also of interest to estimate its pose, appearance, and identity. Typical objects of interest include humans and vehicles.
- *Smart Cameras.* Constraints in communication make it inappropriate to transmit all of the video data collected at each node across the network. However, the availability of processing power at each camera enables the transmission of processed low-bandwidth information that is *sufficient* for the task at hand. This raises two important issues. What is the nature of the information that needs to be extracted in each individual "smart camera" node? How does one fuse the information extracted at the nodes in order to solve detection, tracking, and recognition tasks in visual sensor

Manuscript received December 3, 2007; revised May 29, 2008. First published October 17, 2008; current version published October 31, 2008. This work was supported in part by the National Science Foundation under ITR Grant IIS 03-25119.
The authors are with the Department of Electrical and Computer Engineering, University of Maryland, College Park, MD 20742 USA (e-mail: aswch@umiacs.umd.edu; vashok@umiacs.umd.edu; rama@umiacs.umd.edu).
Digital Object Identifier: 10.1109/JPROC.2008.928758

networks? Power and energy optimization in distributed smart cameras is another important challenge that other papers in this issue tackle in detail.

There are numerous applications of distributed visual sensing algorithms. We concentrate on the problems of distributed detection, tracking, and recognition. These form three integral subsystems of any robust distributed visual sensing network. The number of cameras connected in the distributed network can vary greatly: from a few (less than ten) cameras for household monitoring of the elderly to tens of cameras to monitor a building and all the way to hundreds of cameras connected in a traffic monitoring network. The specific challenges encountered in each of these applications vary with the number of cameras that are connected in the network. Nevertheless, some of the basic principles of algorithm design and optimization remain the same in all these scenarios.

A. Outline of This Paper

In Section II, we describe the geometric constraints that are involved in multicamera problems with special emphasis on those constraints that have a direct impact on algorithm design for detection, tracking, and recognition of objects. In Section III, we describe some of the challenges in object detection from uncalibrated cameras and describe how the constraint that the scene has a dominant plane can be exploited to develop distributed detection algorithms for cameras with overlapping fields of view. In Section IV, we provide a formal description of the problem of distributed tracking in a camera network and describe an optimal multiview fusion algorithm that can combine evidence from multiple camera views to obtain a robust estimate of the objects' location in a three-dimensional (3-D) world coordinate system. In Section V, we study the problem of recognition of humans and vehicles. In particular, we show how to perform object verification across nonoverlapping views using novel views synthesized from 3-D models built and propagated across the network of cameras. We conclude by highlighting several problems that remain to be solved in the area of visual sensor networks.

II. GEOMETRIC CONSTRAINTS OF MULTIPLE CAMERAS

In this section, we introduce the basics of projective geometry and discuss some of the concepts that are extensively used for detection and tracking. We do note that this is not an exhaustive coverage of this topic. An in-depth discussion of projective geometry can be found in [1] and [2]. The projective nature of imaging introduces unique challenges in distributed camera networks. In the context of detection, tracking, and recognition algorithms, it becomes important to understand the nature of such constraints and their impact on these problems.

A. A Note on Notation and Homogeneous Coordinates

In the rest of this paper, we use bold to denote vectors and capital letters to denote matrices. Further, we use x, y, and z to denote quantities in world coordinates and u and v for image plane coordinates. In addition to this, the concept of homogeneous coordinates is important. We use a tilde to represent entities in homogeneous coordinates. Given a d-dimensional vector $\mathbf{u} \in \mathbb{R}^d$, its homogeneous representation is given as a $(d+1)$-dimensional vector $\tilde{\mathbf{u}} \backsim [\mathbf{u}, 1]^T$, where the operator $\backsim$ denotes equality up to scale. In other words, $\tilde{\mathbf{u}} \backsim \tilde{\mathbf{x}} \leftrightarrow \tilde{\mathbf{u}} = \lambda \tilde{\mathbf{x}}$, $\lambda \neq 0$. In simpler terms, when we deal with homogeneous quantities, we allow for a scale ambiguity in our representation. The representation mainly allows for elegant representations of the basic imaging equations that we discuss next.

B. Central Projection

Central projection is the fundamental principle behind imaging with a pinhole camera and serves as a good approximation for lens-based imaging for the applications considered here. In the pinhole camera model, rays (or photons) from the scene are projected onto a planar screen after passing through a pinhole. The screen is typically called the image plane of the camera. Consider a camera with its pinhole at the origin and the image plane aligned with the plane $z = f$. Under this setup, a 3-D point $\mathbf{x} = (x, y, z)^T$ projects onto the image plane point $\mathbf{u} = (u, v)^T$, such that

$$u = f\frac{x}{z}, \quad v = f\frac{y}{z}. \tag{1}$$

This can be elegantly written in homogeneous terms as

$$\tilde{\mathbf{u}} \backsim \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} fx/z \\ fy/z \\ 1 \end{pmatrix} \backsim \begin{pmatrix} fx \\ fy \\ z \end{pmatrix} = \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{x}. \tag{2}$$

A more general model of the pinhole camera allows for the pinhole to be at an arbitrary position and the image plane oriented arbitrarily. However, we can use a simple Euclidean coordinate transformation to map this as an instance of the previous one. Finally, the camera might have nonsquare pixels with image plane skew. This leads us to a general camera model whose basic imaging equation is given as

$$\tilde{\mathbf{u}} \backsim K[R \ \mathbf{t}]\tilde{\mathbf{x}} = P\tilde{\mathbf{x}} \tag{3}$$

where P is the 3×4 matrix encoding both the internal parameters of the camera K (its focal length, principal

point, etc.) and the external parameters (its orientation R and position $\mathbf{t}$ in a world coordinate system). $\tilde{\mathbf{u}}$ and $\tilde{\mathbf{x}}$ are the homogeneous coordinate representations of the pixel in the image plane and the point being imaged in the real world, respectively. Although central projection is inherently nonlinear, it can be written as a linear transformation of the homogeneous coordinates. Finally, (2) can be obtained from (3) with $R = \mathbb{I}_3$ (the identity matrix), $\mathbf{t} = \mathbf{0}$, and $K = \mathrm{diag}(f, f, 1)$.

It is noteworthy that the projection equation of (3) is not invertible in general. Intuitively, the pinhole camera maps a 3-D world onto a two-dimensional (2-D) plane, and hence, the mapping is many-to-one and noninvertible. All points that lie on a line passing through the pinhole map onto the same image plane point. This can also be independently verified by the scale ambiguity in (3). Given a point on the image plane $\mathbf{u}$, its *preimage* is defined as the set of all scene points that map onto $\mathbf{u}$ under central projection. It is easily seen that the preimage of a point is a line in the real world. Without additional knowledge of the scene and/or additional constraints, it is not possible to identify the scene point that projects onto $\mathbf{u}$. This lack of invertibility leads to some of the classical problems in computer vision, the most fundamental being establishing correspondence across views.

C. Epipolar Geometry

Consider two images (or central projections) of a 3-D world. Given a point $\mathbf{u}_A$ on the first image of a world point $\mathbf{x}$, we know that its preimage is a line passing through the point $\mathbf{u}_A$ and C_A, the pinhole of the camera (see Fig. 1). Hence, given information about $\mathbf{u}_A$ on the first image, all we can establish is that the corresponding projection of the point $\mathbf{x}$ on the second image plane $\mathbf{u}_B$ lies on the projection of the preimage of $\mathbf{u}_A$ onto the second image plane. Since the preimage of $\mathbf{u}_A$ is a line, the projection of this line onto view B gives the line $L(\mathbf{u}_A)$, the *epipolar line* associated with $\mathbf{u}_A$. Thus, the epipolar geometry constrains corresponding points to lie on conjugate pairs of epipolar lines.

In the context of multiview localization problems, the epipolar constraint can be used to associate objects across multiple views [3]. Once we obtain reliable correspondence across multiple views, we can triangulate to localize objects in the real world. However, correspondences based on epipolar constraint alone tends to be insufficient, as the constraint does not map points uniquely across views. In general, all points lying on the epipolar line are potential candidates for correspondence.

D. Triangulation

In many detection and tracking applications, once we have correspondences between object locations across views, we are interested in localization of these objects in scene coordinates. Let us assume that the same object has been detected in two views (A and B) with camera center C_A and C_B at image place locations $\mathbf{u}_A$ and $\mathbf{u}_B$, as shown in Fig. 2. In this case, the basics of projective imaging constrains the object to lie on the preimage of the point $\mathbf{u}_A$ (the line connecting C_A and $\mathbf{u}_A$). Similarly, the object must also lie on the preimage of $\mathbf{u}_B$ in view B. Therefore, estimating the true location of the object amounts to estimating the point of intersection of these two lines. In a general scenario with several cameras, each camera gives

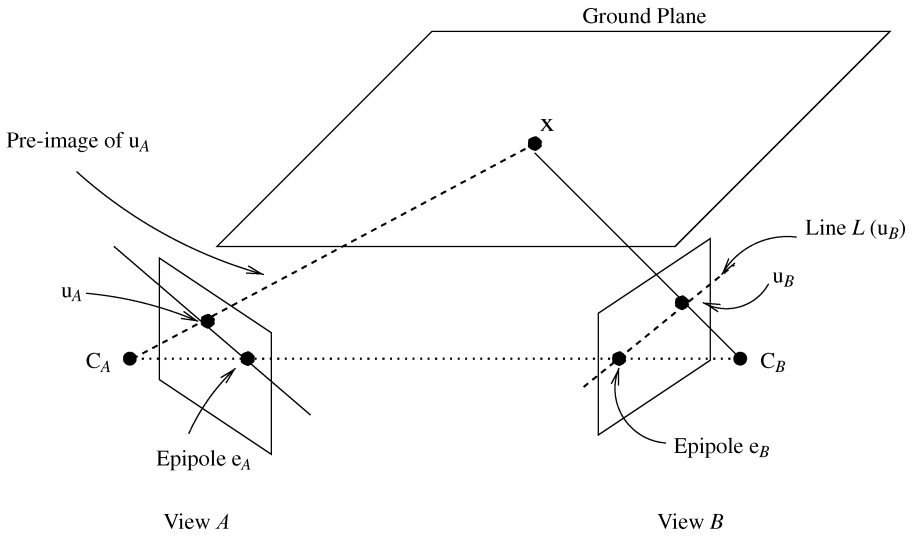


Fig. 1. *Consider views A and B (camera centers C_A and C_B) of a scene with a point x imaged as u_A and u_B on the two views. Without any additional assumptions, given u_A, we can only constrain u_B to lie along the image of the pre-image of u_A (a line). However, if the world were planar (and we knew the relevant calibration information), then we can uniquely invert u_A to obtain x and reproject x to obtain u_B.*

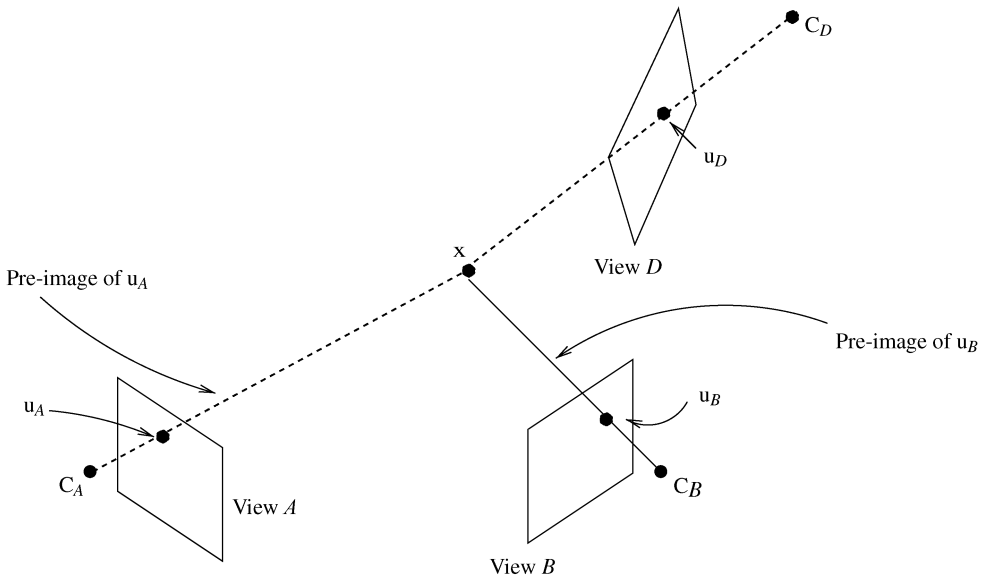


Fig. 2. *Consider views A, B, and D of a scene with a point x imaged as u_A, u_B, and u_D on the views. We can estimate the location of x by triangulating the image plane points as shown in the figure. At each view, we draw the preimage of the point, which is the line joining the image plane point and the associated camera center. The properties of projective imaging ensure that the world point x lies on this preimage. Hence, when we have multiple preimages (one from each view), the intersection of these lines gives the point x.*

rise to a line and estimating the object's location involves computing the point of intersection of these lines. In the presence of noisy measurements, these lines do not intersect at a single point and error measures such as sum of squares are used to obtain a robust estimate of the location of the object. This is called *triangulation* [4]. The drawback of the triangulation approach is that it requires correspondence information across cameras, which is difficult to obtain.

E. Planar Scenes and Homography

There is one special scenario when the imaging equation becomes invertible, and that is when the world is planar. Most urban scenarios form a good fit as the majority of actions in the world occur over the ground plane. This makes it a valid assumption for a host of visual sensing applications. The invertibility can also be efficiently exploited by algorithms for various purposes. As an example, consider the correspondence problem that we mentioned earlier. Under a planar world assumption, the preimage of a point becomes a point (in most cases) being the intersection of the world plane and the preimage line. This implies that by projecting this world point back onto the second image plane, we can almost trivially find correspondence between points on the two image planes. This property induced by the world plane, that seemingly allows for finding correspondences across image planes, is referred to as the *homography* induced by the plane.

Consider two views of a planar scene labeled view A and view B. We can define a local coordinate system at each view. The same scene point denoted as $\mathbf{x}_A$ and $\mathbf{x}_B$ on the two coordinate systems is related by a Euclidean transformation

$$\mathbf{x}_B = R\mathbf{x}_A + \mathbf{t}. \tag{4}$$

Here, R (a rotation matrix) and $\mathbf{t}$ (a 3-D translation vector) define the coordinate transformation from A to B. Let us assume that the world plane has an equation $\mathbf{n}^T\mathbf{x}_A = d$, with $d \neq 0$.[1] For points that lie on the plane, we can rewrite (4) as

$$\begin{aligned}\mathbf{x}_B &= R\mathbf{x}_A + \mathbf{t}\frac{\mathbf{n}^T\mathbf{x}_A}{d} \\ &= \left(R + \frac{1}{d}\mathbf{t}\mathbf{n}^T\right)\mathbf{x}_A.\end{aligned} \tag{5}$$

In each local camera coordinate system, we know that $\tilde{\mathbf{u}} \backsim K[R\ \mathbf{t}]\tilde{\mathbf{x}}$ [see (3)] with $R = \mathbb{I}_3$ and $\mathbf{t} = \mathbf{0}$. Therefore, $\tilde{\mathbf{u}}_B \backsim K_B\mathbf{x}_B$ and $\tilde{\mathbf{u}}_A \backsim K_A\mathbf{x}_A$, which gives us

$$\begin{aligned}K_B^{-1}\tilde{\mathbf{u}}_B &\backsim \left(R + \frac{1}{d}\mathbf{t}\mathbf{n}^T\right)K_A^{-1}\tilde{\mathbf{u}}_A \\ \tilde{\mathbf{u}}_B &\backsim H\tilde{\mathbf{u}}_A, \text{ where } H = K_B\left(R + \frac{1}{d}\mathbf{t}\mathbf{n}^T\right)K_A^{-1}.\end{aligned} \tag{6}$$

[1]When $d = 0$, the plane passes through the pinhole at A, thereby making the imaging noninvertible.

This implies that a point in view A, $\mathbf{u}_A$ maps to the point $\mathbf{u}_B$ on view B as defined by the relationship in (6). The 3×3 matrix H [in (6)] is called the homography matrix or just the homography. Also, note that H (like P) is a homogeneous matrix, and the transformation defined by it is unchanged when H is scaled. Further, H is invertible when the world plane does not pass through pinholes at either of the two views. This is easily verified, as our derivation is symmetric in its assumptions regarding the two views.

Finally, the premise of the induced homography critically depends on the fact that the preimage of a point on the image plane is a unique point on the world plane. Suppose we use a local 2-D coordinate system over the world plane; the image plane to world plane transformation (from their respective 2-D coordinate systems) can be shown to be a projective transformation, which, like before, can be encoded as a 3×3 homogeneous matrix, say, H_π. This transformation is useful when we want to estimate metric quantities, or quantities in a Euclidean setting. The most common example of this is when we need to localize the target in the scene coordinates.

Computing the image plane to world plane transformation H_π is a challenging problem that is typically done by exploiting properties of parallel and perpendicular lines on the planes. Typically, this requires manual inputs such as identifying straight lines segments that are parallel. While this is not always possible, many urban scenes (such as parking lots, roads, buildings) contain such lines, which makes it easier to estimate the transformation H_π, at least in a semisupervised way. Computing H_π, as it turns out, is identical to a metric rectification of the image plane. Many such techniques are illustrated in [1].

III. DETECTION

The first and foremost task in distributed visual sensing is to detect objects of interest as they appear in the individual camera views [5]–[7]. In general, this is a very challenging task since objects belonging to the same class (say, humans, for instance) can have significantly different appearances in the images because of factors such as clothing, illumination, pose, and camera parameters. Object detection in images and video may be achieved using one of two major approaches—a static feature-based characterization of the objects of interest or object motion as a cue to detect objects. Several recent approaches have been developed for object detection and recognition in images and videos, and these approaches typically involve maintaining a model for the objects of interest in the form of a set of static features and possibly a model for the spatial relationship between the various features. Given a test image, object detection is then decomposed into two steps—finding features in the test image and then validating whether the set of visible features in the test image suitably explains the presence of the object in the image. One problem with adopting any such approach for video is that these approaches are computationally intensive and it would be inefficient especially when the number of objects is large. Secondly, these approaches also require a training set of images in which the objects of interest have been labeled. This would mean that objects not previously modeled would not be detected in a test sequence. Therefore, we will not discuss these methods in the rest of this paper.

In typical visual sensing scenarios, the objects of interest are those that are moving. Detection of moving objects is a much easier task, because object motion typically leads to changes in the observed intensity at the corresponding pixel locations, and this change in intensity can be used to detect moving objects. The challenge in a single camera setup is to associate groups of coherently moving nearby pixels to a single object. In multicamera networks, it also becomes necessary to associate detected objects across camera views.

A. Background Modeling for Moving Object Detection

Detection of moving objects is typically performed by modeling the static background and looking for regions in the image that violate this model. The simplest model is that of a single template image representing the static background. A test image can then be subtracted from the template and pixels with large absolute difference can be marked as *moving*. This simple model introduces the idea of background subtraction—essentially the process of removing static background pixels from an image.

B. Background Subtraction

Traditionally, background subtraction is posed as a hypothesis test [8] at each pixel, where the *null* hypothesis H_0 is that the pixel belongs to the background model B_t and the *alternate* hypothesis H_1 is that the pixel *does not* belong to B_t. Here, the subscript t is used to denote time, and hence B_t represents the background model at time t, and I_t the image at time t.

Given the hypothesis test defined as

$$\begin{aligned} H_0 &: I_t^i \in B_t^i \text{ (pixel is background)} \\ H_1 &: I_t^i \notin B_t^i \text{ (pixel is NOT background)} \end{aligned} \tag{7}$$

the likelihood ratio associated with the hypothesis test is defined as

$$\frac{\Pr\left(I_t^i | B_t^i\right)}{1 - \Pr\left(I_t^i | B_t^i\right)} \gtrless_{H_1}^{H_0} \tau. \tag{8}$$

I_t^i and B_t^i correspond to the ith pixel of the image and background model, respectively, and τ defines the threshold whose value is decided based on a desired false alarm or misdetection rate.

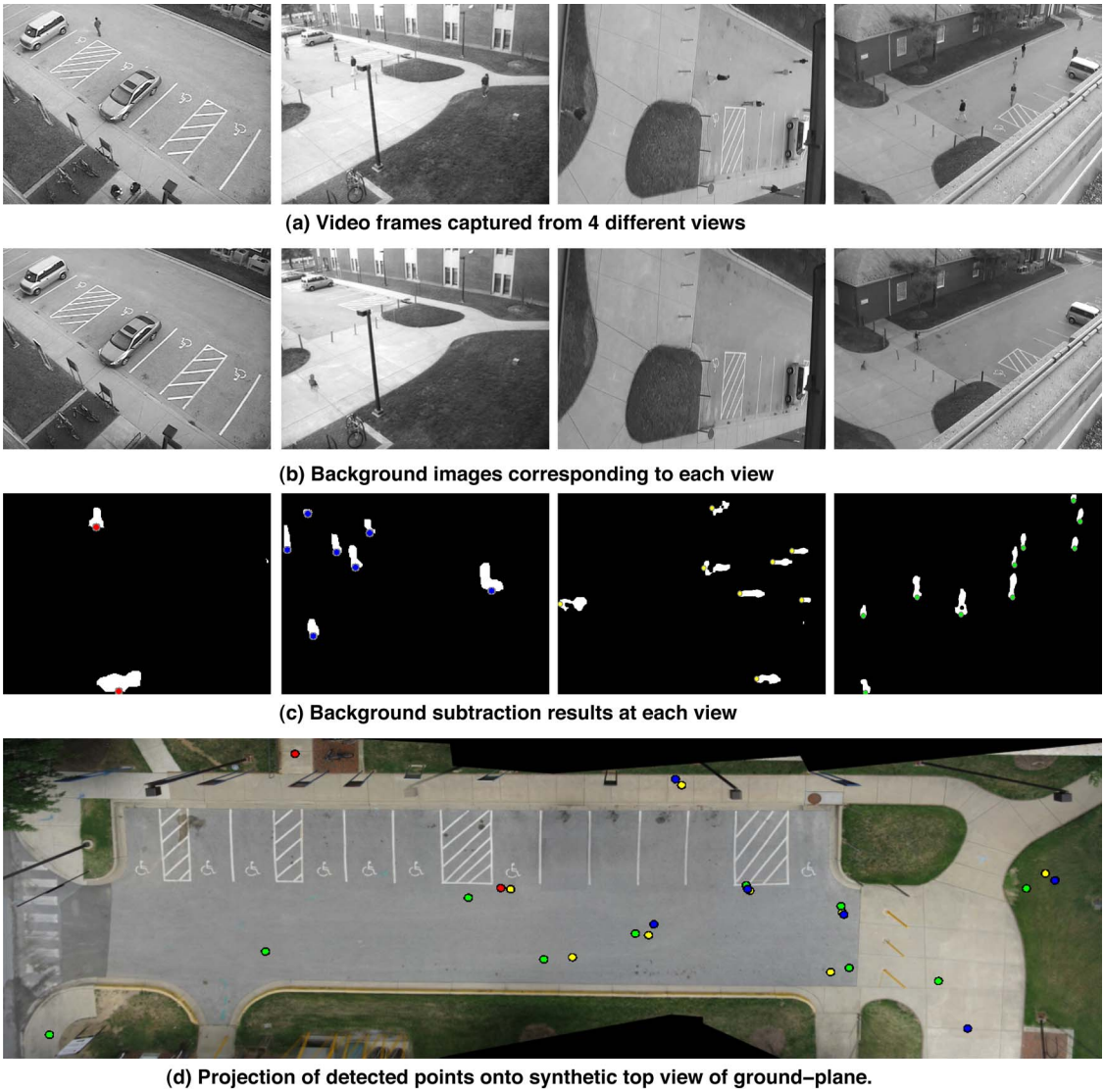


Fig. 3. *Use of geometry in multiview detection. (a) Snapshot from each view. (b) Object-free background image. (c) Background subtraction results. (d) Synthetically generated top view of the ground plane. The bottom point (feet) of each blob is mapped to the ground plane using the image-plane to ground-plane homography. Each color represents a blob detected in a different camera view. Points of different colors very close together on the ground plane probably correspond to the same subject seen via different camera views.*

The likelihood ratio test defined in (8) is equivalent to

$$\Pr\left(I_t^i | B_t^i\right) \gtrless_{H_1}^{H_0} \eta = \frac{\tau}{1+\tau}. \tag{9}$$

As an example, consider a simple background model, where $B_t = B_0$ is an object-free static background image. For common models used in the hypothesis test, the likelihood ratio test takes the form

$$\left|I_t^i - B_0^i\right| \lessgtr_{H_1}^{H_0} \eta'. \tag{10}$$

This intuition behind this test is that the error term $|I_t^i - B_0^i|$ will be very small at pixels that correspond to static objects, while this term will be large for pixels corresponding to moving objects. Fig. 3 shows an observed image, the corresponding background model, and their difference. As seen, this difference is very small except in locations corresponding to the moving person and the moving car. This difference image can be used to estimate the set of pixels that correspond to moving objects.

C. Common Background Models

However, a simple background model such as a fixed template $(B_t = B_0)$ would be susceptible to global changes in the environment due to lighting, time of the day,

weather effects, etc. Ideally, we would like the following from our detection algorithm:

- adaptive to global changes in the scene, such as illumination or camera jitter;
- resilient to periodic disturbances (such as tree movement due to wind or rain).

There exist a host of algorithms that work very well in many scenarios [5]–[7]. One simple extension of the static model is by modeling each pixel as Gaussian distributed with mean B_t^i and variance $\sigma_{i,t}^2$.

The background model is defined as

$$\Pr\left(I_t^i|B_t\right) \propto \exp\left(-\frac{\left(I_t^i - B_t^i\right)^2}{2\sigma_{i,t}^2}\right). \tag{11}$$

The model defined in (11) extends the simple static background model by allowing for a variance term $\sigma_{i,t}^2$ at each pixel. Effectively, this corresponds to using a different threshold at each pixel, one that depends both on τ and on the variance $\sigma_{i,t}^2$. Such models are useful in handling dynamic scenes with significant clutter. The means B_t^i and variances $\sigma_{i,t}^2$ are typically updated in a manner that attempts to keep the background model object-free [6].

Another background model that robustly handles periodic background disturbances is the mixture of Gaussians (MoG) model [7]. This model adaptively learns an MoG distribution at each pixel. The multimodality of the underlying distribution gives the ability to capture repetitive actions as part of the background. An adaptive learning method can be used to keep track of global changes in the scene. As before, given a new image, we can compute the likelihood image and threshold it to obtain pixels that violate the background model.

D. Using Homography for Multiview Localization

At each individual camera view, background subtraction results in a binary image that labels each individual pixel either as belonging to the background or as belonging to the moving foreground object. Due to changing texture and illumination conditions in the scene, the pixels belonging to a single object may lead to disconnected blobs. Simple heuristics and connected component analysis is performed on the binary background subtracted image to label the binary image into a set of objects with each object occupying a connected set of pixels (blob) in the image. The location and the various characteristics of each detected object blob are stored.

If we assume that the objects are all moving on the ground plane, then the detected blobs in each camera view can be transformed to the world plane using the image-plane to world-plane transformation. Let us assume that H_i represents the 3×3 homogeneous matrix that relates coordinates in the image plane of camera i to that of a local 2-D coordinate system on the world plane. Also, let $H_{(i,j)}$ denote the homography relating the image plane of camera i to that of camera j. Therefore, $H_{(i,j)} = H_j^{-1}H_i$. A single object moving on the world plane will produce corresponding blobs in each of the cameras that are able to view the object.

Existing multiview localization algorithms project features from the detected blob on each of the image planes to the world plane. These features are typically representative points [9], lines [10], or the whole blob itself [11]. The individual choice of the feature depends heavily on the performance of the background subtraction algorithms on the underlying data. Consensus across views is achieved by appropriately fusing these transformed features in the world plane. This is schematically shown in Fig. 3.

E. Relaxing the Homography Constraint: Epipolarity

In many cases, we need to study multiview detection algorithms for more generic scenarios, those in which the assumption of planar scene is violated. In the absence of planarity constraint, the image to scene inversion is no longer unique. In the presence of multiview inputs, it is possible to triangulate and solve for the intersection of these line provided the *necessary correspondence information associating points across views is available*. Such correspondences are in general hard to solve for, given the weakness of the epipolar constraint, which usually generates multiple hypotheses for point correspondences. Typically, these hypotheses are resolved using additional constraints (such as the curvature of the trajectory or the appearance of the object). Fig. 2 and Section II-D illustrate the concept of triangulation for multiview detection.

IV. DISTRIBUTED MULTIVIEW TRACKING

Once the objects of interest have been detected in each of the individual cameras, the next task is to track each object using multiview inputs. Most algorithms maintain an appearance model for the detected objects and use this appearance model in conjunction with a motion model for the object to estimate the object position at each individual camera. Such tracking can be achieved using deterministic approaches that pose tracking as an optimization problem [12], [13] or using stochastic approaches that estimate the posterior distribution of the object location using Kalman filters [14] or more commonly particle filters [15]–[19]. For surveys on visual tracking of objects, we refer the interested readers to [20], [21].

There exist many application domains that benefit immensely from multiview inputs. The presence of multiview inputs allows for the robust estimation of pose and limb geometry (markerless motion capture) [22]–[24]. When targets are at lower resolution, position tracking in scene coordinates provides information that is useful for higher

level reasoning of the scene. As an example, [25] uses multiple cameras to track football players on a ground plane. Similarly, [9], [10], [26], and [27] consider the problem of multiview tracking in the context of a ground plane, with the intent of localization of target position on the plane.

In the case of multicamera systems, another important challenge is the association of objects across the camera views. For cameras that have nonoverlapping fields of views, object association can be achieved by learning the relationship between patterns of entry and exit in various camera views [28]. For cameras with overlapping fields of views, an important issue that arises is fusion of object location estimates. This requires the use of epipolar geometry and triangulation in the most general case. As a special case, in the presence of ground-plane constraint, it is possible to derive efficient estimators for fusing multiview information. We discuss this next.

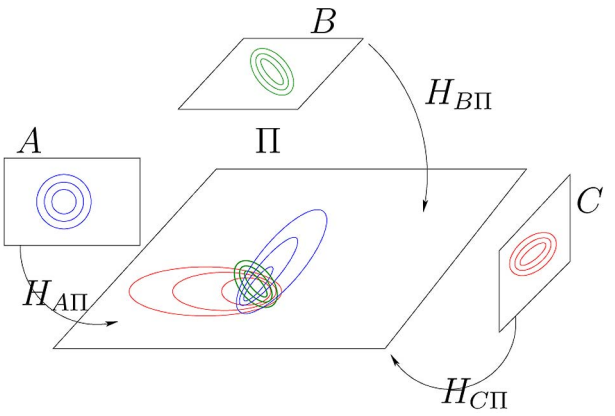


Fig. 4. *A schematic showing densities of the image planes of cameras and their transformations to the ground plane.*

A. Multiview Tracking: Planar World

Multicamera tracking in the presence of ground-plane constraint has been the focus of many recent papers [9]–[11], [29]. The key concepts in many of the algorithms proposed are the following.

- *Association of data across views by exploiting the homography constraint*. This can be done by projecting various features associated with the silhouette. The vertical axis from each segmented human is used as the feature by Kim and Davis [10], while points features [9], [29] or even the whole silhouette [11] form alternate choices. These feature(s) are projected onto a reference view using the homography transformation, and consensus between features is used to associate data across views.
- *Temporal continuity of object motion to track*. Typically, a particle filter [9], [10] is used to temporally filter the data after association. Alternatively, [11] builds a temporal consistency graph and uses graph-cuts [30] to segment tracks.

Since the early work of Smith and Cheeseman [31], researchers have known that one has to account for the effect of varying covariances on parameter fusion and estimation accuracy, especially under highly asymmetric placement of cameras. Consider, for example, the problem of location estimation of a point object moving on a plane. At each camera, background subtraction provides an estimate of where the object lies. We can now project the image plane locations to arrive at individual estimates of the world plane location of the tracked point. In an ideal noise-free condition, the estimates arising from each of the cameras would be identical. However, in the presence of noise corrupting the image plane observations, errors in calibration, and inaccuracies in modeling, the world plane location estimates will no longer be identical. We now need a strategy to fuse these estimates. However, to do so in a systematic fashion, we need to characterize the statistical properties of these estimates. Let us suppose that we have a characterization of the location of the object in the individual image planes as a random variable. We can project these random variables to the ground plane using the projective transformation linking the individual image planes and the ground plane.

Fig. 4 shows an example of three cameras A, B, and C looking at a plane Π, with the image plane of B parallel to Π. In contrast, the image planes of A and C are perpendicular to Π. Also shown on the image planes of the cameras are iso-error contours representing the image plane distribution at each camera. The homographies between the cameras and the plane Π are $H_{A\Pi}$, $H_{B\Pi}$, and $H_{C\Pi}$, respectively. In this setup, $H_{B\Pi}$ is not fully projective, defining only an affine transformation, as opposed to $H_{A\Pi}$ and $H_{C\Pi}$, which induce strong projective distortion. We would expect the density on B to retain its original form (similar error isocontours) when projected on the plane.

The projective mapping is in general a nonlinear transformation involving ratios. The statistics of random variables, when transformed under such a ratio transformation, change significantly. Given that the projective transformations linking different views of the same scene are different, one can expect that the statistics of random variables on the world plane arising from different views will necessarily be different, even when the original random variables are identically distributed.

Given M cameras, and the homography matrices H_i, $i = 1, \ldots, M$, between the camera views and the ground plane, one can derive an algorithm for fusing location estimates. Let $\mathbf{Z}_u^i$ be the random variable modeling the object location on the image plane of the ith camera. Let us assume that the random variables $\{\mathbf{Z}_u^i\}_{i=1}^M$ are statistically independent. Now, each of these random variables can be projected to the world plane to obtain $\mathbf{Z}_x^i$, such that $\tilde{\mathbf{Z}}_x^i \backsim H_i \tilde{\mathbf{Z}}_U^i$, $i = 1, \ldots, M$.

Let us consider the distribution of $\mathbf{Z}_x^i$ under the assumption that the $\mathbf{Z}_u^i$ are Gaussian. Specifically, when

Fig. 5. *Variance ellipses are shown for the individual image planes. The corresponding color-coded ellipse on the ground plane shows the covariance when transformed to the ground plane. The ellipse in black (on the ground plane) depicts the variance of the minimum variance estimator. Note that this estimate performs better than the individual estimates.*

certain geometric properties are satisfied,[2] we can show that the distribution of Z_x^i is closely approximated by a Gaussian distribution [1], [9]. Further, we can relate the mean and the covariance matrix of the transformed random variable to the statistics of the original random variable and the parameters characterizing the projective transformation. This result is useful for designing strategies to fuse $\{\mathbf{Z}_x^i, i = 1, \ldots, M\}$ in an optimal sense. In the case of multiview localization, if the covariances of the estimates $\mathbf{Z}_x^i$ is Σ_i, then the minimum variance estimate $\mathbf{Z}_{mv}$ is computed as

$$\mathbf{Z}_{mv} = \sum_{i=1}^{M} \Sigma_i^{-1} \left(\sum_{j=1}^{M} \Sigma_j^{-1} \right)^{-1} \mathbf{Z}_x^i. \tag{12}$$

The covariance of $\mathbf{Z}_{mv}$, Σ_{mv} is given as

$$\Sigma_{mv} = \left(\sum_{j=1}^{M} \Sigma_j^{-1} \right)^{-1}. \tag{13}$$

We refer the reader to the early works of Smith and Cheeseman [31], [32], Kanatani [33], and, more recently, Sankaranarayanan and Chellappa [9].

[2]The required geometric properties reduce to the region of interest that is being imaged to be far away from the line of infinity in each of the views (for details, refer to [9]).

Hence, given a true object location on the ground plane, Σ_{mv} provides an estimate of the maximum accuracy (or minimum error) with which we can localize the object on the ground plane given modeling assumptions on the image plane (see Fig. 5).

Finally, we can embed the concept used in constructing the minimum variance estimators in formulating dynamical systems that can be used to track objects using multiview inputs. As before, we *efficiently* fuse estimates arising from different views by appropriately determining the accuracy of the estimate characterized by its covariance matrix. Fig. 6 shows tracking outputs from processing video data acquired from six cameras. Each object is tracked using a particle filter, and object-to-data associations are maintained using joint probability data association [34].

This algorithm can be easily implemented in a distributed sensor network. Each camera transmits the blobs extracted from the background subtraction algorithm to other nodes in the network. For the purposes of tracking, it is adequate even if we approximate the blob with an enclosing bounding box. Each camera maintains a multiobject tracker filtering the outputs received from all the other nodes (along with its own output). Further, the data association problem between the tracker and the data is solved at each node separately, and the association with maximum likelihood is transmitted along with data to other nodes.

B. Relaxing the Planar Constraint

There exist many scenarios when the objects' motion is not restricted to the plane or when the scene deviates significantly from a plane. In [35], we observe and track

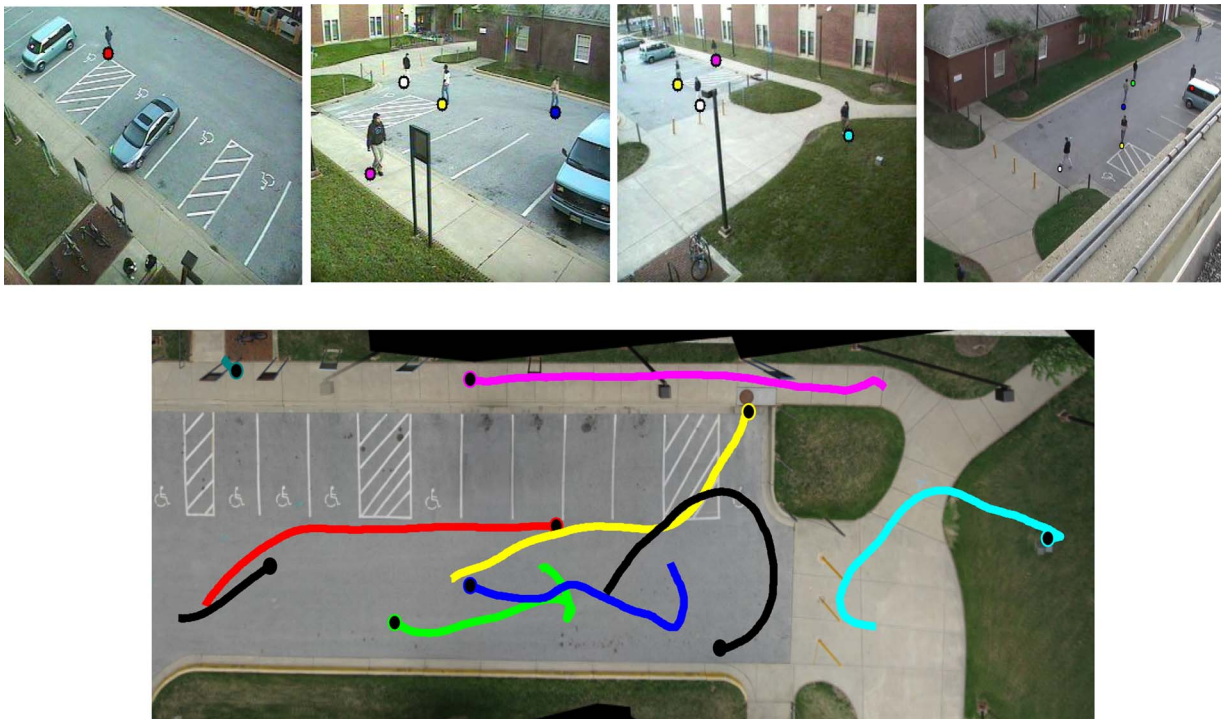


Fig. 6. *Output from the multiobject tracking algorithm working with input from six camera views. (Top row) Four camera views of a scene with several humans walking. Each camera independently detects/tracks the humans using a simple background subtraction scheme. The center location of the feet of each human is indicated with color-coded circles in each view. These estimates are then fused together, taking into account the relationship between each view and the ground plane. (Bottom row) Fused tracks overlaid on a top-down view of the ground plane.*

objects (bees) flying freely in an enclosed space, using inputs from two cameras. The bees are all similar in appearance and image onto small areas (5–10 pixels). We cannot perform association using appearance information in this scenario. Finally, the internal and the external calibration parameters of the camera might not be available.

In [35], we exploit the property that *critical points of trajectories are invariant to changes in view* [36] for associating objects across camera views. We consider the cameras to be independent and first perform background subtraction as shown in Fig. 3. At each camera, we track bees by associating the background subtracted blobs temporally, generating a set of object trajectories in each camera view. We need to now perform object association across camera views, i.e., associate each trajectory in the view of one camera to a unique trajectory in the other camera view. We use the fact that instants of maximal curvature in the original 3-D space map to instants of maximal curvature in the respective image spaces irrespective of the specific camera view [36]. Therefore, we first compute the instants of maximal curvature in the 2-D trajectories observed in each view and associate trajectories using the time instants of maximal curvature. Fig. 7 shows the spatiotemporal curvature of a flight path as seen in two different camera views. Since the points of maximal curvature match irrespective of the view, one can use this in order to associate targets across camera views. We can use the correspondences that the trajectory associations provide in order to obtain most of the required calibration parameters and then perform triangulation to obtain the actual 3-D location of the object in each frame. Shown in the last row of Fig. 7 are the 3-D flight trajectories of five different bees flying from a bee hive to a sugar bowl.

V. RECOGNITION

Having detected and tracked objects using multiple cameras, we are now in a position to recognize the objects. Object recognition from images and videos is a long-standing research problem, and there have been several competing approaches. In general, algorithms for object recognition can be divided into two major divisions—local feature-based approaches and global approaches. Feature-based approaches detect several points of interest in each image of an object and describe the object using descriptors of these local feature points.

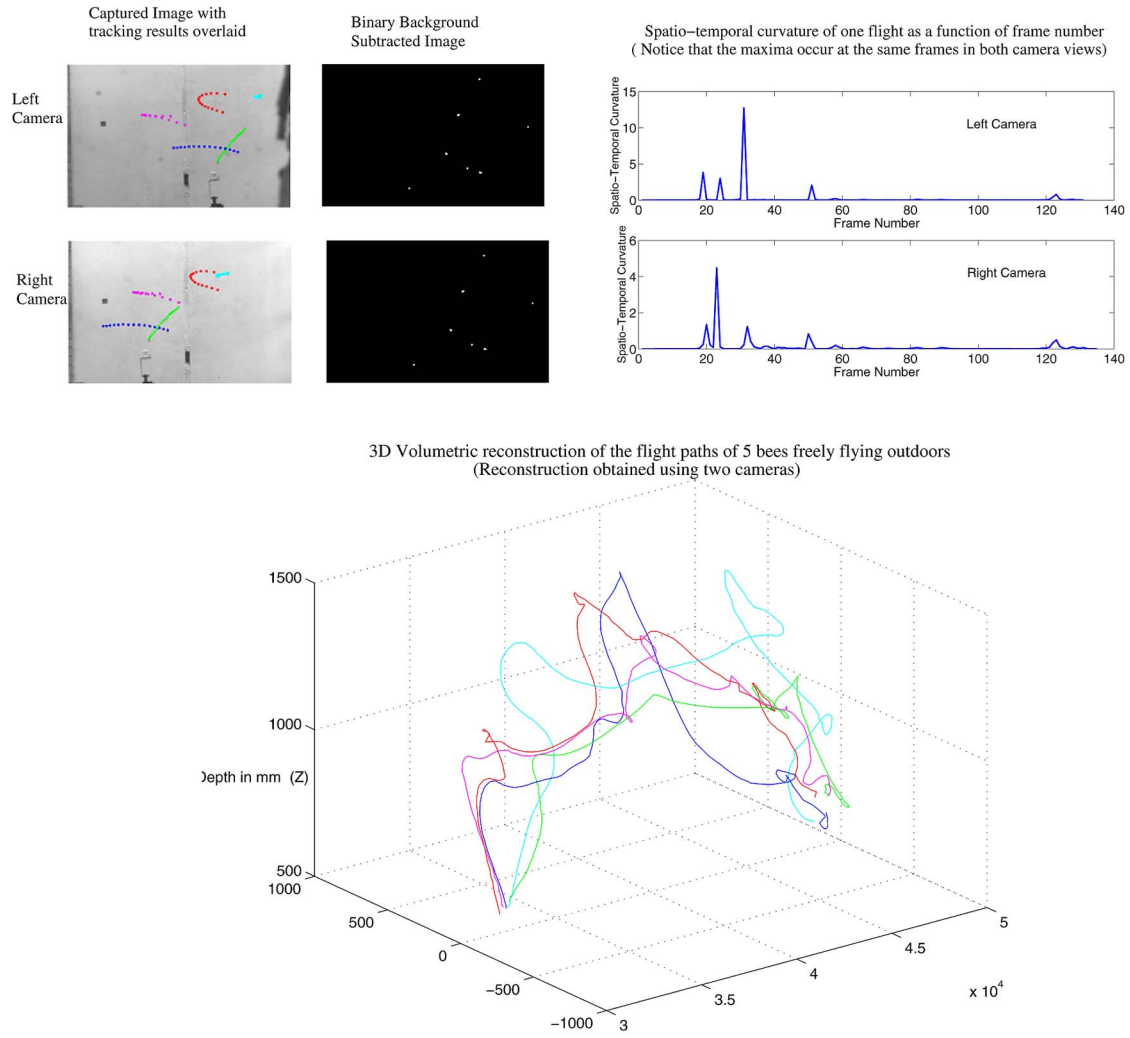


Fig. 7. ***Two camera views viewing a scene in which there are several bees freely flying around. The second column shows the background subtracted images. Note that each object occupies only a few pixels. The spatiotemporal curvature as observed at the two camera views is also shown. Note how the maxima of the spatiotemporal curvature match irrespective of the camera view. The reconstructed 3-D flight paths are shown below. (Image courtesy of [35].)***

Global approaches describe the object using their global properties such as shape, silhouette, texture, color, appearance, or any combination of such descriptors.

A. Global Approaches for Object Recognition

Global methods for object recognition involve the use of some global property of the object such as color, texture, shape, etc., for recognition. Such approaches are inherently sensitive to the effect of external conditions such as lighting, pose, viewpoint, etc. The influence of such external conditions on the global properties of the object is usually complex and very difficult to model. Therefore, one needs additional assumptions about either the 3-D structure of the objects or the viewpoint of the camera in order for these methods to be successful.

1) 2-D Appearance Models for Recognition: A simple feature for classification is to build 2-D appearance models for each class and use these 2-D appearance models for classification. Such 2-D appearance models are a natural choice, specifically while modeling and recognizing planar or near-planar objects (ignoring effects of self-occlusion) since the effect of viewpoint on these appearance models is easily accounted for. In particular, small viewpoint changes produce affine deformation on the 2-D appearance models. Thus, affine-invariant 2-D appearance models are common and effective representations for recognizing planar and near-planar objects. Nevertheless, the problem with using a single 2-D appearance model is that when the pose of a 3-D object changes, a simple 2-D appearance model cannot account for this change in pose.

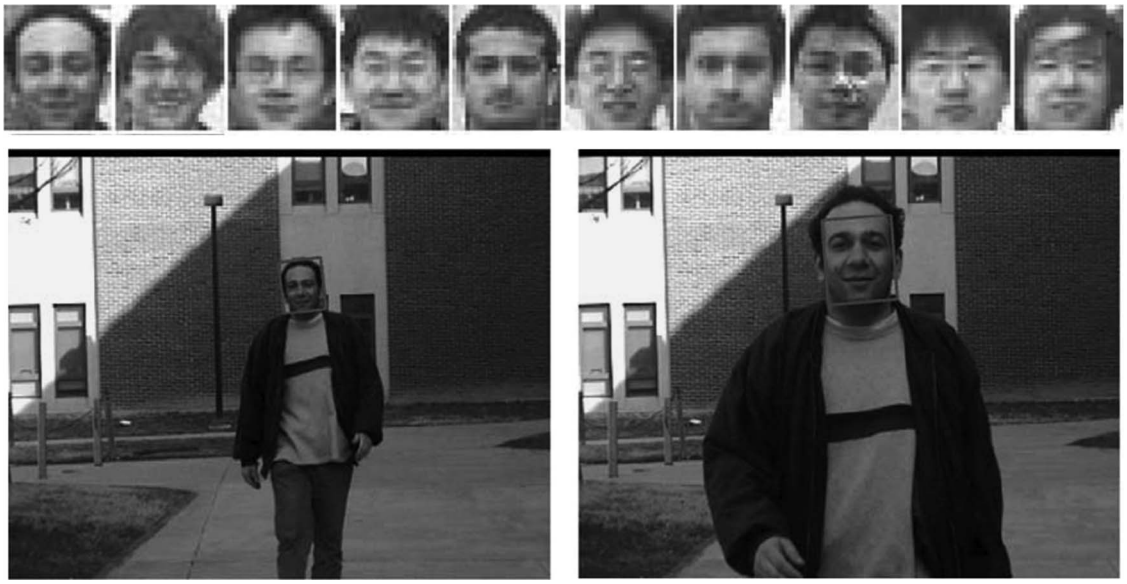


Fig. 8. *(Top row) 2-D appearance models for the individuals in the gallery. (Bottom row) Two images from a video sequence in which a person is walking. The target's face is being tracked, and the image within the bounding box of the tracked face is matched with the 2-D appearance models in the gallery in order to perform recognition.*

There are several approaches that use 2-D affine invariant appearance models for face tracking and recognition [37]–[42]. As an example, let us consider the simultaneous face tracking and recognition framework presented in [41]. A 2-D appearance-based tracker is used in order to track objects of interest. For each person in the gallery, a simple 2-D appearance template is stored, which is an image of the person's face under uniform illumination conditions. A particle filter is then used to simultaneously estimate both the position of the target's face and the identity of the individual. The 2-D appearance of the individual is modeled as a mixture of Gaussians, and the parameters of the mixture density are estimated from a training gallery. Each camera first estimates its confidence with regard to whether the face appears frontal in its view. This can be achieved using a simple correlation-based detector with a generic frontal face appearance or other view-selection methodologies [43]. Assuming that at least one of the camera views is frontal, this camera then compares the observed appearance with those stored in the gallery in order to recognize the individual. The top row of Fig. 8 shows the stored 2-D appearance templates for the individuals in the gallery. In the bottom are two images from a test sequence with the bounding box showing the location of the target's face. The image within the bounding box is matched with the stored 2-D appearance models in the gallery in order to perform recognition.

2) 3-D Face Tracking With Geometric Face Models: The problem with 2-D appearance models is that it does not adequately account for the inherent changes in the feature that occur due to large pose changes in the video (especially for nonplanar objects). For applications such as face tracking and recognition (where the perspective effects cannot be ignored due to proximity between the face and the camera), it becomes extremely important to account for pose changes that occur throughout the video so that continuous recognition is possible even when there are few cameras viewing the individual and none of these is able to obtain a frontal view. One way to account for changes in pose is to model the face as a 3-D object with a certain structure and a corresponding texture. Since the variations in face structure across individuals is at best modest, one can assume a generic 3-D model for the face with the texture varying according to the individuals. The texture forms the cue for identity, while the 3-D generic face model allows recognition to be performed irrespective of the pose of the face in the video. There are several competing approaches for fitting 3-D models to a face in order to perform recognition. In [44]–[46], a statistical model of 3-D faces is learnt from a population of 3-D scans, and recognition is performed after morphing the acquired image to the 3-D model. Unfortunately, moving from a planar model to complicated 3-D models also introduces significant issues in registration between an acquired 2-D image and the 3-D model. As the number of parameters in the 3-D model becomes large, this registration task becomes difficult. Therefore, several approaches have adopted simple parameterized 3-D models to model the face, thus keeping the registration between a 2-D image and a 3-D model simple.

A simple but effective model for the generic 3-D model of a face is that of a cylinder [47], [48]. The advantage of

using such a simple model is that occlusion analysis becomes extremely simple, allowing efficient inference algorithms for estimating the pose of the model at each frame. Once the pose of the model at each frame is estimated/tracked at each camera independently, the image intensities in the original image frame are then mapped onto the generic 3-D model to obtain the texture map of the face being tracked. The texture mapped models obtained at each individual camera node can all be fused to obtain a complete 3-D model of the face. This texture mapped model is then compared with the stored texture maps of all the 3-D models in the gallery in order to perform face-based person recognition. Another point to be noted is that since the face is assumed to be cylindrical, once the pose of the face is estimated, the surface normals at each of the points on the face are known. This allows us to extract texture features that are moderately insensitive to illumination conditions. Therefore, modeling the 3-D structure of the face in order to perform simultaneous tracking and recognition allows us to design recognition algorithms that are robust to both changes in facial pose and illumination conditions. Fig. 9 shows some of the results [48] of 3-D facial pose tracking and recognition. Notice that the pose of the face is accurately estimated in spite of the significant variations in lighting, pose and also significant occlusions. The graphical rendering in the last column shows the cylindrical face model at the pose estimated from the images in the third column. An implementation of this algorithm suitable for smart cameras is discussed in [49].

Another significant advantage of using 3-D face models for tracking and recognition is that such models can be easily extended for multicamera applications. Each camera can independently track faces using its own 3-D face model while the individual pose estimates can then be appropriately fused either at a central node or in a completely distributed manner using just local communications. A joint estimate of the pose can be obtained as a Euclidean mean of these individual estimates, and this mean can be efficiently estimated in a distributed network using just local communications [50]. Unfortunately, the 3-D pose does not lie in Euclidean space, and therefore the averaging procedure needs to account for the non-Euclidean nature of this space. Many methods to average rotation matrices can be found in [51] and [52]. However, the convergence properties of such estimation methods, when used in a decentralized computation framework (such as the one described in [50]), need to be studied.

B. Feature-Based Methods for Object Recognition

In recent years, feature-based methods for object recognition have been gaining in popularity. This is

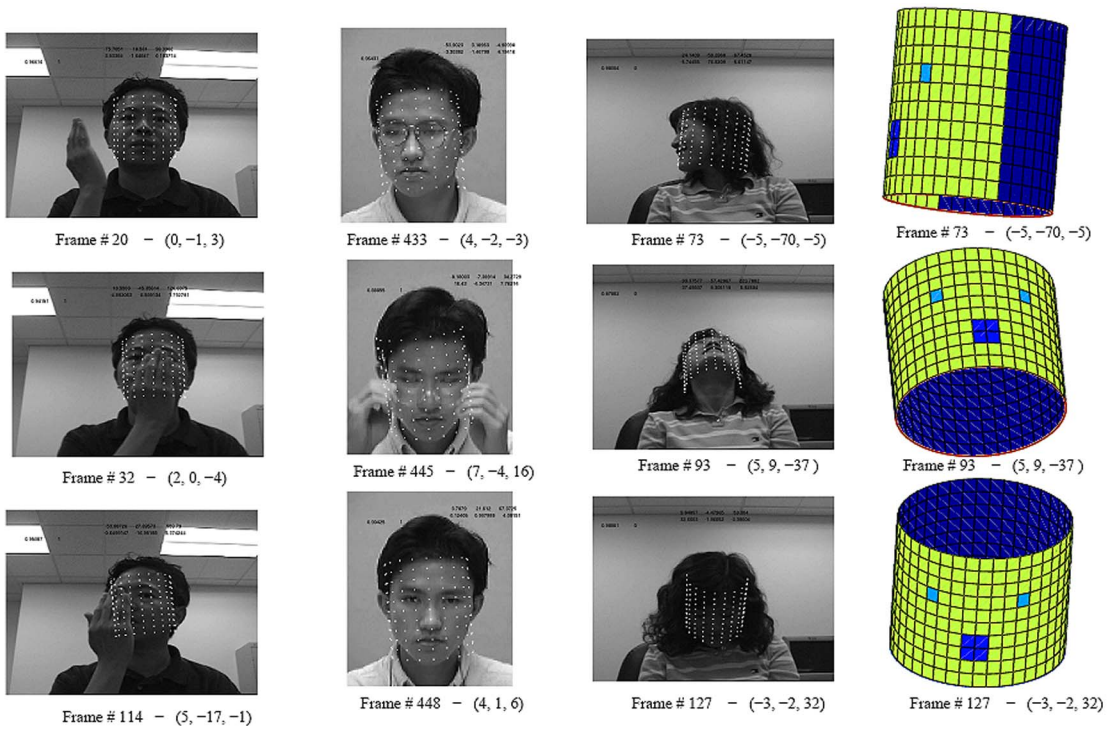


Fig. 9. ***Tracking results under severe occlusion, extreme poses, and different illumination conditions. The cylindrical grid is overlaid on the image plane to display the results. The three-tuple shows the estimated orientation (roll, yaw, pitch) in degrees. The second column shows a cylindrical model in the pose estimated for the sequence in the third column. (Courtesy of [48].)***

because while it is extremely difficult to model the deformation of global features due to changes in viewpoint and illumination conditions, it is a much simpler task to model the local deformations due to these structured changes. Feature-based object recognition can usually be divided into three stages of processing. First, discriminative points of interest are detected in each image frame. These interest points may be chosen using the simple Harris corner detector [53], or by selecting the most discriminative features [54], or by using the scale-invariant feature transform (SIFT) interest points that are invariant to scale and orientation [55]. Next, a descriptor is computed for each of these chosen feature locations. This descriptor is typically chosen such that it is invariant to some local deformations, so that pose and lighting changes do not affect these local descriptors significantly. Examples of such descriptors that are popular include SIFT [55], [56] or the deformation invariant feature proposed in [57]. Once such descriptors are computed for each feature point, then the object is described using a bag of features model [58], [59], in which the geometrical relationship between the feature points is completely lost. Instead, some approaches use a coarse representation of this geometrical relationship between feature points in order to improve discrimination between object categories [60]. The essential advantage of feature-based approaches for object recognition is the fact that since these local feature descriptors are very robust to changes in viewpoint and illumination, these approaches are consequently robust even under extreme view changes and occlusions.

1) Feature Based Tracking and Recognition Across a Sparse Camera Network: In a smart camera network, local feature-based methods allow for object recognition simultaneously and independently to be performed on each of the smart cameras locally. Moreover, even when the fields of view of the cameras do not overlap, such feature-based approaches may be used to maintain the target's identity across the smart camera network. As an example, consider the scenario where a sparse collection of video cameras is monitoring a large area. Target association across cameras needs to be performed only using the appearance information of these targets since the fields of views of these cameras might not overlap. Unfortunately, global appearance models such as a 2-D affine template image are not sufficient since the pose of the target will be very different when it reappears in another camera view. Moreover, since the targets may have very different 3-D structures, it is not possible to use a generic 3-D model, as was the case for face tracking. In such a scenario, local feature-based methods in combination with structure from motion techniques provide an effective alternative. In addition, structure from motion-based methods allows for target-specific 3-D models to be built online as the target moves within the field of view of each camera.

Consider a sparse distribution of cameras (see Fig. 10) covering a large area with blind regions in the coverage of the vehicle movement. As a potential application, let us suppose a white SUV is seen approaching a camera. Suppose, a list of *authorized* vehicles is available with appropriate descriptions; then we could verify if the approaching vehicle is in the authorized list. Verification of vehicle identity across nonoverlapping views presents two important challenges: pose and illumination. The models built for each vehicle need to account for possible changes in both.

In [61], we address the problem of establishing identity of vehicles across nonoverlapping views when the vehicle moves on a plane. We use the 3-D structure of the vehicle, along with statistical appearance models as the *fingerprint* representing the vehicles. Estimation of the 3-D structure of the vehicle is performed using an approach specifically suited to vehicles exhibiting planar motion [62]. The ability to estimate 3-D structure allows us to explicitly address the changes in pose (and hence, view). The estimated 3-D structure and its texture are used to generate synthetic views of the object. These synthetic views are then compared with the view of the vehicle in other cameras in order to perform recognition across nonoverlapping cameras. We formulate the problem as one of simultaneous tracking and verification [18] of vehicle identity using the structural and appearance models in the fingerprint. In traditional tracking, the state-space over which filtering is performed contains the position of the object being tracked, while in a simultaneous tracking and verification framework, this state-space is augmented with the identity of the vehicle. Thus simultaneous tracking and verification formally amounts to recursively estimating the position and the identity of the vehicle. Such an approach has several advantages over traditional methods since a) accurate tracking results improve recognition performance, b) improving recognition performance improves tracking since the gallery can contain addition individual specific information, and c) recursive filtering in a video enables the algorithm to be robust to occlusions, slow illumination, and pose changes. Estimating the pose of the object in every frame enables recognition to be performed irrespective of the viewpoint of the camera. Fig. 10 summarizes results from [61].

VI. CONCLUSIONS AND DISCUSSIONS

In this paper, we discussed the basic challenges in detection, tracking, and classification using multiview inputs. In particular, we discussed the role of the geometry induced by imaging with a camera in estimating target characteristics. In detection and tracking, we show that the presence of a ground plane can be used as a strong constraint for designing efficient and robust estimators for target location. We also discuss how one can go beyond the planar constraint for overlapping network of cameras. We

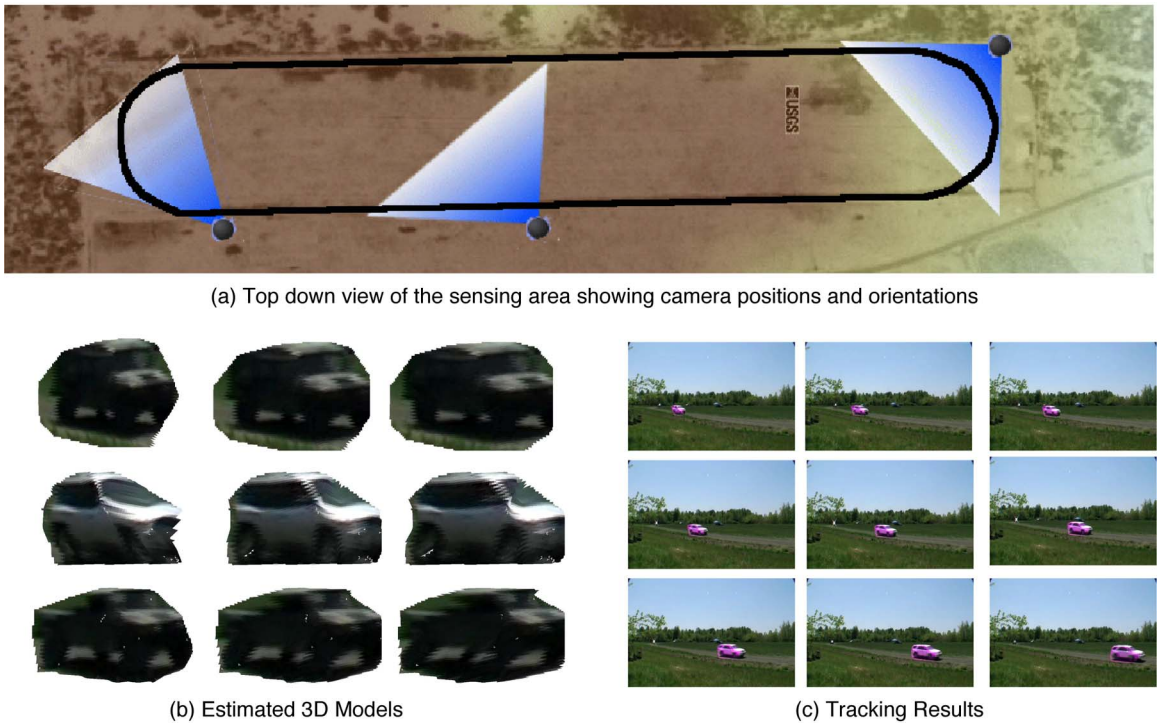


Fig. 10. ***Tracking and verification across nonoverlapping views. (a) A schematic top view of the sensing area with fields of view of three cameras shown. (b) 3-D structures (with texture maps overlaid) of three vehicles, as estimated from one of the views. (c) Tracking results with the output inlaid in magenta.***

demonstrated how 2-D appearance models and 3-D shape and texture models can be used for recognition of objects.

The algorithms presented here are optimized for sensor networks that contain a small number of cameras. Therefore, the solutions presented typically consisted of distributed sensing using a set of cameras but central and coordinated processing. In the near future, we will need to adapt some of the algorithms presented here in order to tackle the same detection, tracking, and recognition problems in camera networks containing possibly hundreds of cameras. In such cases, it becomes essential to consider not only distributed sensing but also distributed processing, efficient transmission of sufficient data across the network, and optimize over power and energy consumption of the network. In order to tackle these challenges, we need a more integrated approach that exploits the rich theory for distributed estimation in sensor networks to solve the detection, tracking, and classification problems using a network of video sensors. Several papers in this Special Issue discuss algorithms for such distributed computations while simultaneously optimizing for power, energy, and/or bandwidth constraints. Several upcoming fields of research including distributed function estimation, distributed processing, mobile camera control, and manipulation will also allow us to tackle many of these challenges. Lastly, the issue of smart visual surveillance in the context of additional modalities has immense significance for practical deployable systems [63]–[65]. We briefly discuss some of these issues here.

A. Mobile Cameras

Sensing with a mix of static and mobile cameras connected on a wireless network is becoming prevalent with the increasing use of unmanned air vehicles. The advantage of sensing with mobile cameras is that since the sensing resources may be allocated dynamically one, this may reduce the number of cameras required in order to monitor the same area. This results in significant power and energy savings while simultaneously increasing the number of pixels on the target by using the results of target tracking to zoom into these targets. A systematic and detailed study of both power/energy optimization versus algorithm performance [66], [67] for such mobile camera networks is necessary. Mobile platforms are also important, as they act as a synthetic aperture in capturing the scene. In [68], a homography-based view-synthesis algorithm is developed to generate novel views of observed objects of interest. Such modeling is used for verification of object identity in video sequences collected at a later time.

B. Steering PTZ Cameras for Improved Inference

The on-chip processing capabilities of smart cameras allow for the ability for local control of steerable cameras.

Pan-tilt-zoom (PTZ) cameras allow for steering of the camera view, keeping the camera center (pinhole) in place. Such a rotating (pan and tilt) and zooming camera produces views that satisfy elegant geometric properties [69]–[71]. The advantage of using PTZ cameras is that one may now control the specific settings of the PTZ cameras so as to improve tracking and recognition performance. Thus while one or few "master cameras" observe a wide sensing field of view, their tracking results in turn guide the PTZ controls for other slave cameras that can then zoom into objects of interest in order to obtain high-resolution imagery of these objects. In general, such a problem may be posed as an optimization of some desirable cost functional over the steering controls of the cameras. One example of such a cost functional would be the average tracking error over the whole scene. Here, we want to obtain new views (using the PTZ controls) that minimize the desired objective. We envision steering algorithms that operate some of the cameras in the slave mode, tethered with inputs from other cameras that sense the entire region of interest.

C. Novel Visualization of Multiview Inferences

The use of multiple cameras also necessitates novel technologies for visualization of the data streaming in from these cameras. There are many ways to present the processed information to the end user. These include simple map-based interfaces that give geometric context to the end user [72] and various user interfaces that display only scenes with persistent or interesting activity to the end user (like in the IBM Smart Surveillance System). There has been a significant amount of work in novel visualization tools that exploit these multiple camera views in order to render virtual views that best depict the information content in the scene [72]–[74]. Simultaneously, there has been the development of virtual reality tools in both 2-D [75] and 3-D [76] for immersive visualization of such data. We are currently building a testbed for novel visualization schemes in order to provide an end-user freedom in viewing the scene and the activities being performed from *arbitrary points of view*. Most view interpolation techniques require a dense array of sensors for reliable depiction. We overcome this requirement by precomputing the model of the static background. Dynamic foreground objects such as humans and vehicles are handled by estimating their location, pose, and other characteristics (such as clothing and gaze direction). Virtual view rendering is then performed by appropriately fusing the precomputed static background model with the dynamic foreground object characteristics imposed on a synthetic virtual actor. One potential application of this technology is in scene monitoring, where the security personnel can freely move around the scene without having to watch a fixed set of CCTV screens (where the spatial coherence between views and the activities are lost). Further, this can be combined with algorithms that alert the personnel when events of interest occur.

D. Distributed Particle Filtering

The algorithms for tracking and recognition presented here were both based on online inference using particle-filtering. Therefore, in order to make these algorithms truly distributed and enable their implementation on huge camera networks containing hundreds of cameras, one needs to pay attention to methods that enable these particle-filter-based estimates to be performed in a distributed manner. This can be achieved using either *synchronized particle filtering* or the more general means of *distributed function estimation*.

1) Synchronized Particle Filtering: One way to *decentralize* the filter operations is to replicate it identically at each node. For particle filtering, this can be done easily if the random number generators are made identical across nodes. Such a scheme is referred to as synchronized particle filtering [77]. By initializing the random number generator with the same seed, all nodes can be made to generate the same particles, which in turn makes fusion of the associated weights simpler. The communication costs are then limited to the transmission of the associated weights across the whole network.

The immense flexibility of this approach allows for it to be effective in any particle-filtering algorithm. However, this freedom in generality comes with associated drawbacks. For one, the stability of the algorithm depends critically on the requirement of synchronized random numbers, which requires that the hardware at each node be the same. Further, this particular way of decentralization does not efficiently use the processing power of the nodes, as in the end the same computations are performed *identically* at each node.

2) Distributed Function Estimation: However, we can relax the need to make our distributed inference algorithm *identical* to the centralized one. There are a host of methods that allow for the computation of average mean through explicit global communication or through local consensus [50], [78].

An alternative to the concept of synchronous filtering can be by approximating the inference at each camera with a Gaussian mixture model [79] or, in general, any parametric density family. The parameters can then be transmitted to all nodes in the sensor network, each of which locally updates their densities.

In addition to distributed inference and filtering, efficient implementations of particle filters form an important direction for future research, especially in the context of decentralized computing and sensing. Existing approaches to this problem [80], [81] are limited to node-level algorithms, although under a distributed architecture of computing.

E. The Future of Distributed Smart Cameras

The increasing need and reliance on distributed array of visual sensors for automated monitoring has several important wide-ranging applications, from surveillance

and homeland security to traffic management and commercial access control to medical applications such as automated monitoring of the elderly. Such applications and their prevalence will only increase in the future. The need to come up with a concerted and integrated approach for solving the interdisciplinary problems is paramount, and recent research efforts in the direction have made some significant progress. ■

REFERENCES

[1] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[2] Y. Ma, S. Soatto, J. Kosecka, and S. Sastry, *An Invitation to 3D Vision, From Images to Models*. Berlin, Germany: Springer-Verlag, 2003.

[3] G. Qian, R. Chellappa, and Q. Zheng, "Spatial self-calibration of distributed cameras," in *Proc. Collab. Technol. Alliances Conf.—Sensors*, 2003.

[4] R. I. Hartley and P. Sturm, "Triangulation," *Comput. Vision Image Understand.*, vol. 68, no. 2, pp. 146–157, 1997.

[5] T. Lv, B. Ozer, and W. Wolf, "A real-time background subtraction method with camera motion compensation," in *Proc. IEEE Int. Conf. Multimedia Expo*, 2004, vol. 1.

[6] S. Joo and Q. Zheng, "A temporal variance-based moving target detector," in *Proc. IEEE Int. Workshop Perform. Eval. Track. Surveill. (PETS)*, Jan. 2005.

[7] A. Elgammal, R. Duraiswami, D. Harwood, and L. S. Davis, "Background and foreground modeling using nonparametric kernel density estimation for visual surveillance," *Proc. IEEE*, vol. 90, no. 7, pp. 1151–1163, 2002.

[8] G. Casella and R. L. Berger, *Statistical Inference*. Belmont, CA: Duxbury, 1990.

[9] A. C. Sankaranarayanan and R. Chellappa, "Optimal multi-view fusion of object locations," in *Proc. IEEE Workshop Motion Video Comput. (WMVC)*, Jan. 2008.

[10] K. Kim and L. S. Davis, "Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering," in *Proc. Eur. Conf. Comput. Vision*, 2006, vol. 3, pp. 98–109.

[11] S. M. Khan and M. Shah, "A multi-view approach to tracking people in crowded scenes using a planar homography constraint," in *Proc. Eur. Conf. Comput. Vision*, 2006, vol. 4, pp. 133–146.

[12] G. D. Hager and P. N. Belhumeur, "Efficient region tracking with parametric models of geometry and illumination," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, pp. 1025–1039, 1998.

[13] D. Comaniciu, V. Ramesh, and P. Meer, "Real-time tracking of non-rigid objects using mean-shift," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2000, vol. 2, pp. 142–149.

[14] T. J. Broida, S. Chandrashekhar, and R. Chellappa, "Recursive techniques for the estimation of 3-d translation and rotation parameters from noisy image sequences," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 26, pp. 639–656, 1990.

[15] A. Doucet, N. D. Freitas, and N. Gordon, *Sequential Monte Carlo Methods in Practice*. New York: Springer-Verlag, 2001.

[16] M. Isard and A. Blake, "Icondensation: Unifying low-level and high-level tracking in a stochastic framework," in *Proc. Eur. Conf. Comput. Vision*, 1998, vol. 1, pp. 767–781.

[17] J. S. Liu and R. Chen, "Sequential monte carlo for dynamical systems," *J. Amer. Statist. Assoc.*, vol. 93, pp. 1031–1041, 1998.

[18] S. K. Zhou, R. Chellappa, and B. Moghaddam, "Visual tracking and recognition using appearance-adaptive models in particle filters," *IEEE Trans. Image Process.*, vol. 11, pp. 1434–1456, 2004.

[19] Z. Khan, T. Balch, and F. Dellaert, "An MCMC-based particle filter for tracking multiple interacting targets," in *Proc. Eur. Conf. Comput. Vision*, 2004.

[20] W. Hu, T. Tan, L. Wang, and S. Maybank, "A survey on visual surveillance of object motion and behaviors," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 34, no. 3, pp. 334–352, 2004.

[21] A. Yilmaz, O. Javed, and M. Shah, "Object tracking: A survey," *ACM Comput. Surv.*, vol. 38, no. 4, 2006.

[22] D. M. Gavrila and L. S. Davis, "3-D model-based tracking of humans in action: A multi-view approach," in *Proc. 1996 IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR'96)*, 1996, pp. 73–80.

[23] A. Sundaresan and R. Chellappa, "Model driven segmentation and registration of articulating humans in laplacian eigenspace," *IEEE Trans. Pattern Anal. Mach. Intell.*, to be published.

[24] Q. Delamarre and O. Faugeras, "3D articulated models and multi-view tracking with silhouettes," in *Proc. ICCV*, 1999, vol. 99, pp. 716–721.

[25] M. Xu, J. Orwell, and G. Jones, "Tracking football players with multiple cameras," in *Proc. 2004 Int. Conf. Image Process. (ICIP'04)*, 2004, vol. 5.

[26] J. Black, T. Ellis, and P. Rosin, "Multi view image surveillance and tracking," in *Proc. Workshop Motion Video Comput. 2002*, 2002, pp. 169–174.

[27] A. Mittal and L. S. Davis, "M 2 tracker: A multi-view approach to segmenting and tracking people in a cluttered scene," *Int. J. Comput. Vision*, vol. 51, no. 3, pp. 189–203, 2003.

[28] D. Makris, T. Ellis, and J. Black, "Bridging the gaps between cameras," in *Proc. Conf. Comput. Vision Pattern Recognit.*, 2004, vol. 2.

[29] F. Fleuret, J. Berclaz, and R. Lengagne, Multi-camera people tracking with a probabilistic occupancy map, Tech. Rep. EPFL/CVLAB2006.07, Jul. 2006.

[30] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, pp. 888–905, 2000.

[31] R. C. Smith and P. Cheeseman, "On the representation and estimation of spatial uncertainty," *Int. J. Robot. Res.*, vol. 5, no. 4, p. 56, 1986.

[32] R. Smith, M. Self, and P. Cheeseman, "Estimating uncertain spatial relationships in robotics," *Auton. Robot Vehicles*, vol. 1, pp. 167–193, 1990.

[33] K. Kanatani, *Statistical Optimization for Geometric Computation: Theory and Practice*. New York: Elsevier Science, 1996.

[34] Y. Bar-Shalom and T. Fortmann, *Tracking and Data Association*. New York: Academic, 1988.

[35] A. Veeraraghavan, M. Srinivasan, R. Chellappa, E. Baird, and R. Lamont, "Motion based correspondence for 3d tracking of multiple dim objects," in *Proc. Int. Conf. Acoust., Speech Signal Process.*, 2006, vol. 2.

[36] J. M. Rubin and W. A. Richards, *Boundaries of Visual Motion*. Cambridge, MA: MIT Press, 1985.

[37] W. Zhao, R. Chellappa, P. J. Phillips, and A. Rosenfelf, "Face recognition: A literature survey," *ACM Comput. Surv.*, vol. 35, no. 4, pp. 399–458, 2003.

[38] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 1991, pp. 586–591.

[39] P. N. Belhumeur, J. P. Hespanha, D. J. Kriegman *et al.*, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 711–720, 1997.

[40] G. Aggarwal, A. K. R. Chowdhury, and R. Chellappa, "A system identification approach for video-based face recognition," in *Proc. 17th Int. Conf. Pattern Recognit. (ICPR 2004)*, 2004, vol. 4.

[41] S. Zhou, V. Krueger, and R. Chellappa, "Probabilistic recognition of human faces from video," *Comput. Vision Image Understand.*, vol. 91, pp. 214–245, 2003.

[42] B. Moghaddam and A. Pentland, "Probabilistic visual learning for object representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 19, no. 7, pp. 696–710, 1997.

[43] F. Mokhtarian and S. Abbasi, "Automatic selection of optimal views in multi-view

object recognition," in *Proc. Br. Mach. Vision Conf. (BMVC'00)*, 2000, pp. 272–281.

[44] V. Blanz and T. Vetter, "Face recognition based on fitting a 3 D morphable model," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 9, pp. 1063–1074, 2003.

[45] G. J. Edwards, T. F. Cootes, and C. J. Taylor, "Face recognition using active appearance models," in *Proc. Eur. Conf. Comput. Vision*, 1998, vol. 2, pp. 581–695.

[46] T. S. Jebara and A. Pentland, "Parametrized structure from motion for 3D adaptive feedback tracking of faces," in *Proc. Comput. Vision Pattern Recognit.*, 1997, vol. 761, p. 762.

[47] M. La Cascia, S. Sclaroff, and V. Athitsos, "Fast, reliable head tracking under varying illumination: An approach based on registration of texture-mapped 3 D models," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 4, pp. 322–336, 2000.

[48] G. Aggarwal, A. Veeraraghavan, and R. Chellappa, "3D facial pose tracking in uncalibrated videos," in *Proc. IEEE Int. Conf. Pattern Recognit. Mach. Intell.*, 2005.

[49] S. Saha, C. C. Shen, C. J. Hsu, G. Aggarwal, A. Veeraraghavan, A. Sussman, and S. S. Bhattacharyya, "Model-based OpenMP implementation of a 3D facial pose tracking system," in *Proc. Int. Conf. Workshops Parallel Process.*, 2006, pp. 66–73.

[50] L. Xiao, S. Boyd, and S. J. Kim, "Distributed average consensus with least-mean-square deviation," *J. Parallel Distrib. Comput.*, vol. 67, no. 1, pp. 33–46, 2007.

[51] M. Moakher, "Means and averaging in the group of rotations," *SIAM J. Matrix Anal. Applicat.*, vol. 24, pp. 1–16, 2002.

[52] S. R. Buss and J. P. Fillmore, "Spherical averages and applications to spherical splines and interpolation," *ACM Trans. Graph.*, vol. 20, no. 2, pp. 95–126, 2001.

[53] C. Harris and M. Stephens, "A combined corner and edge detector," in *Alvey Vision Conf.*, 1988, vol. 15, p. 50.

[54] J. Shi and C. Tomasi, "Good features to track," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR'94)*, 1994, pp. 593–600.

[55] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[56] Y. Ke and R. Sukthankar, "PCA-SIFT: A more distinctive representation for local image descriptors," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2004, vol. 2, pp. 506–513.

[57] H. Ling and D. W. Jacobs, "Deformation invariant image matching," in *Proc. ICCV*, 2005, pp. 1466–1473.

[58] J. Sivic, B. C. Russell, A. Efros, A. Zisserman, and W. T. Freeman, "Discovering object categories in image collections," in *Proc. ICCV*, 2005, vol. 1, p. 65.

[59] L. Fei-Fei and P. Perona, "A Bayesian hierarchical model for learning natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vision Pattern Recognit. (CVPR)*, 2005, vol. 5.

[60] P. F. Felzenszwalb and D. P. Huttenlocher, "Pictorial structures for object recognition," *Int. J. Comput. Vision*, vol. 61, no. 1, pp. 55–79, 2005.

[61] A. C. Sankaranarayanan, J. Li, and R. Chellappa, "Fingerprinting vehicles for tracking across non-overlapping views," in *Proc. Army Sci. Conf.*, 2006.

[62] J. Li and R. Chellappa, "Structure from planar motion," *IEEE Trans. Image Process.*, vol. 15, no. 11, pp. 3466–3477, 2006.

[63] A. Hampapur, L. Brown, J. Connell, A. Ekin, N. Haas, M. Lu, H. Merkl, S. Pankanti, A. Senior, C. F. Shu *et al.*, "Smart video surveillance," *IEEE Signal Process. Mag.*, pp. 38–51, 2005.

[64] L. Marcenaro, F. Oberti, G. L. Foresti, and C. S. Regazzoni, "Distributed architectures and logical-task decomposition in multimedia surveillance systems," *Proc. IEEE*, vol. 89, no. 10, pp. 1419–1440, 2001.

[65] V. Cevher, A. C. Sankaranarayanan, J. H. McClellan, and R. Chellappa, "Target tracking using a joint acoustic video system," *IEEE Trans. Multimedia*, vol. 9, no. 4, pp. 715–727, Jun. 2007.

[66] C. Shen, W. Plishker, S. S. Bhattacharyya, and N. Goldsman, "An energy-driven design methodology for distributing dsp applications across wireless sensor networks," in *Proc. IEEE Real-Time Systems Symp.*, Tucson, AZ, Dec. 2007.

[67] J. Schlessman and W. Wolf, "Leakage power considerations for processor array-based vision systems," in *Proc. Workshop Synth. Syst. Integr. Mixed Inf. Technol.*, 2004.

[68] Z. Yue, S. K. Zhou, and R. Chellappa, "Robust two-camera tracking using homography," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP'04)*, 2004, vol. 3.

[69] A. W. Senior, A. Hampapur, and M. Lu, "Acquiring multi-scale images by pan-tilt-zoom control and automatic multi-camera calibration," in *Proc. 7th IEEE Workshop Applicat. Comput. Vision (WACV/MOTION'05)*, Washington, DC, 2005, vol. 1, pp. 433–438.

[70] C. Stauffer and K. Tieu, "Automated multi-camera planar tracking correspondence modeling," in *Proc. CVPR*, 2003, vol. 1, p. 259.

[71] S. N. Sinha and M. Pollefeys, "Pan-tilt-zoom camera calibration and high-resolution mosaic generation," *Comput. Vision Image Understand.*, vol. 103, no. 3, pp. 170–183, 2006.

[72] R. T. Collins, A. J. Lipton, H. Fujiyoshi, and T. Kanade, "Algorithms for cooperative multisensor surveillance," *Proc. IEEE*, vol. 89, no. 10, pp. 1456–1477, 2001.

[73] I. O. Sebe, J. Hu, S. You, and U. Neumann, "3D video surveillance with augmented virtual environments," in *Proc. Int. Multimedia Conf.*, 2003, pp. 107–112.

[74] S. Fleck, F. Busch, P. Biber, and W. Straßer, "3D surveillance—A distributed network of smart cameras for real-time tracking and its visualization in 3D," in *Proc. 2006 Conf. Comput. Vision Pattern Recognit. Workshop*, 2006, p. 118.

[75] K. Cauble, "The openscene modstealth," in *Proc. Simul. Interoper. Workshop*, 1997.

[76] M. D. Petty, "Computer generated forces in distributed interactive simulation," in *Proc. Distrib. Interact. Simul. Syst. Simul. Train. Aerosp. Environ.*, 1995, pp. 251–280.

[77] M. Coates, "Distributed particle filters for sensor networks," in *Proc. 3rd Int. Symp. Inf. Process. Sensor Netw.*, 2004, pp. 99–107.

[78] L. Xiao and S. Boyd, "Fast linear iterations for distributed averaging," *Syst. Contr. Lett.*, vol. 53, no. 1, pp. 65–78, 2004.

[79] X. Sheng, Y. H. Hu, and P. Ramanathan, "Distributed particle filter with GMM approximation for multiple targets localization and tracking in wireless sensor network," in *Proc. 4th Int. Symp. Inf. Process. Sensor Netw.*, 2005.

[80] A. C. Sankaranarayanan, A. Srivastava, and R. Chellappa, "Algorithmic and architectural optimizations for computationally efficient particle filtering," *IEEE Trans. Image Process.*, vol. 17, no. 5, pp. 737–748, 2008.

[81] M. D. Bolic and P. M. S. Hong, "Resampling algorithms and architectures for distributed particle filters," *IEEE Trans. Signal Process.*, vol. 53, no. 7, pp. 2442–2450, 2005.

ABOUT THE AUTHORS

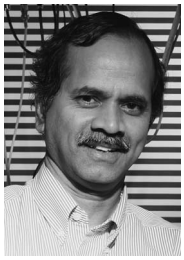
Aswin C. Sankaranarayanan (Student Member, IEEE) received the B.Tech. degree from the Indian Institute of Technology, Madras, in 2003. He is currently pursuing the Ph.D degree from the Department of Electrical and Computer Engineering, University of Maryland, College Park.

In spring 2007, he became a Future Faculty Fellow in A. J. Clark School of Engineering, University of Maryland. He participated in IBM's Emerging Leaders In Multimedia Workshop, T. J. Watson Research Center, in October 2007. His research interests are in computer vision, statistics, geometry, and signal processing.

Ashok Veeraraghavan (Student Member, IEEE) received the B.Tech. degree in electrical engineering from the Indian Institute of Technology, Madras, in 2002 and the M.S. degree from the Department of Electrical and Computer Engineering, University of Maryland, College Park, in 2004, where he is currently pursuing the Ph.D. degree.

His research interests are in signal, image, and video processing, computer vision, pattern recognition, and computational photography.

Mr. Veeraraghavan received the Distinguished Dissertation Fellowship from the Department of Electrical and Computer Engineering, University of Maryland.



Rama Chellappa (Fellow, IEEE) received the Ph.D. degree in electrical engineering from Purdue University, West Lafayette, IN, in 1981.

He is the Minta Martin Professor of Engineering at the University of Maryland, College Park. He directs the Center for Automation Research and is a permanent Member of the Institute for Advanced Computer Studies. During 1981–1991, he was a Faculty Member of the University of Southern California (USC), Los Angeles.

Dr. Chellappa was Editor-in-Chief of the IEEE TRANSACTIONS ON PATTERN ANALYSIS AND MACHINE INTELLIGENCE, a member of the IEEE Signal Processing Society Board of Governors, and its Vice President of Awards. He has received several awards, including the National Science Foundation Presidential Young Investigator Award, three IBM Faculty Development Awards, the Excellence in Teaching Award from the School of Engineering, USC, and a Technical Achievement Award from IEEE Signal Processing Society. He was a Distinguished Faculty Research Fellow and a Distinguished Scholar-Teacher. He received the 2007 A. J. Clark School of Engineering Faculty Outstanding Research Award. He is serving a Distinguished Lecturer of the IEEE Signal Processing Society and received the society's 2008 Meritorious Service Award. He received a Meritorious Service Award and will receive a 2008 Technical Achievement Award from the IEEE Computer Society.