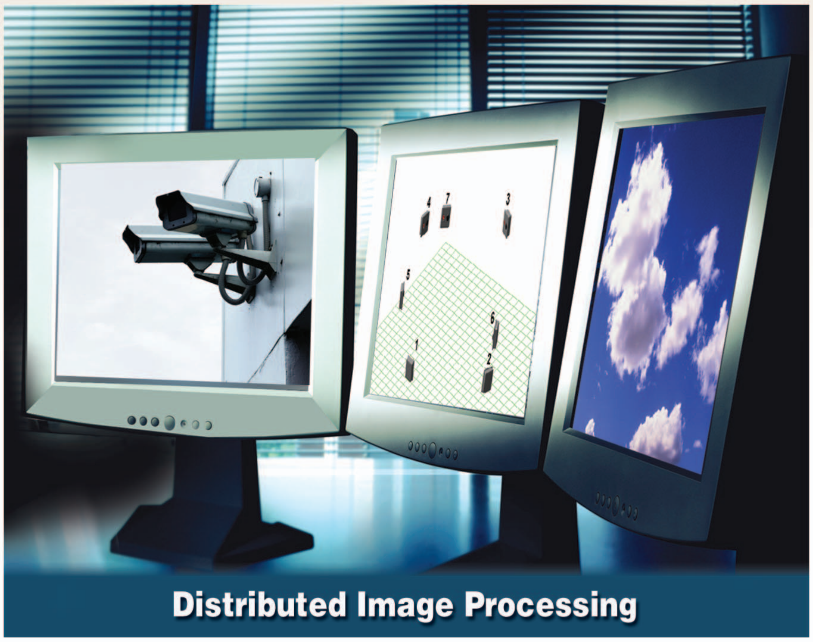
[Bi Song, Chong Ding, Ahmed T. Kamal, Jay A. Farrell, and Amit K. Roy-Chowdhury]

Distributed Camera Networks

[Integrated sensing and analysis for wide-area scene understanding]



Over the past decade, large-scale camera networks have become increasingly prevalent in a wide range of applications, such as security and surveillance, disaster response, and environmental modeling. In many applications, bandwidth constraints, security concerns, and difficulty in storing and analyzing large amounts of data centrally at a single location necessitate the development of distributed camera network architectures. Thus, the development of distributed scene-analysis algorithms has received much attention lately. However, the performance of these algorithms often suffers because of the inability to effectively acquire the desired images, especially when the targets are dispersed over a wide field of view (FOV). In this article, we show how to develop an end-to-end framework for integrated sensing and analysis in a distributed camera network so as to maximize various scene-understanding performance criteria (e.g., tracking accuracy, best shot, and image resolution). We show how the existing work in autonomous multiagent systems can be leveraged for this purpose, more specifically, game theory-based distributed optimization algorithms for dynamic camera network reconfiguration and consensus algorithms for scene analysis. An experimental test bed for evaluating such work is described, and the comparisons against other approaches are provided. The results from a real-life camera network are also presented.

Digital Object Identifier 10.1109/MSP.2011.940441
Date of publication: 19 April 2011

INTRODUCTION

Networks of video cameras are being installed in many applications. Currently, most of the data collected by such networks is manually analyzed, a task that is extremely tedious. This reduces the potential of the installed networks. Therefore, it is essential to develop tools for automatically analyzing the data collected from these cameras and summarizing the results in a manner that is meaningful to the end user. In existing camera networks consisting of fixed cameras covering a large area, it may often be the case that some targets are not covered at the desired resolution or viewpoint, thus making the analysis of the video difficult. A possible solution is to integrate the analysis and sensing tasks more closely. This can be achieved by controlling the parameters of a pan–tilt–zoom (PTZ) camera network to fulfill the analysis requirements, thus allowing for maximal utilization of the network, providing greater flexibility while requiring less hardware and being cost efficient.

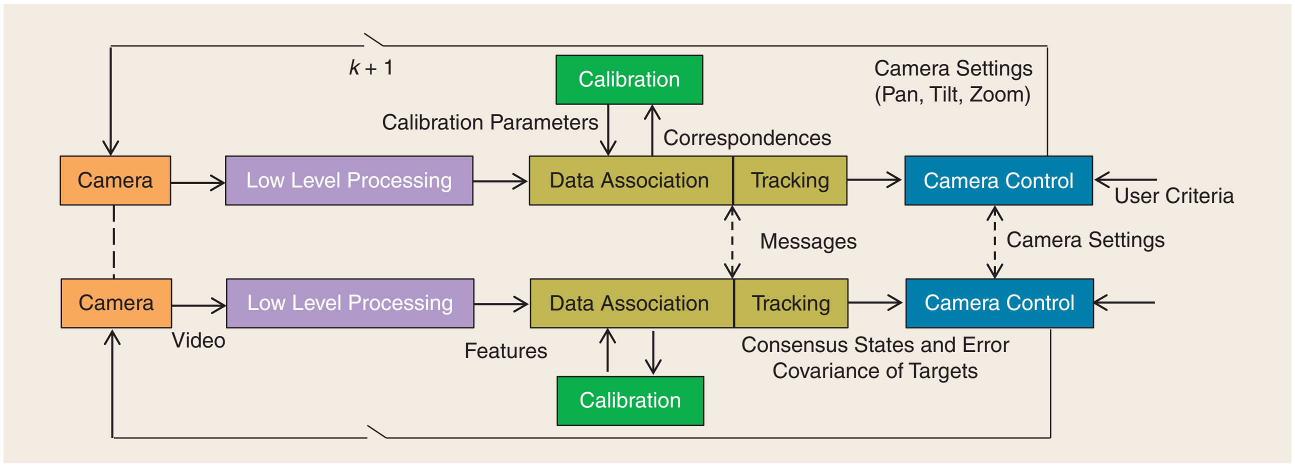
[THE DEVELOPMENT OF DISTRIBUTED SCENE-ANALYSIS ALGORITHMS HAS RECEIVED MUCH ATTENTION LATELY.]

Moreover, in many applications, it is desirable that the video analysis tasks be decentralized for various reasons. For example, there may be bandwidth constraints (e.g., mobile networks), security issues, and difficulties in analyzing a large amount of data centrally. In such situations, the cameras would have to act as autonomous agents, making decisions in a decentralized manner. At the same time, however, the decisions of the cameras need to be coordinated so that there is a consensus about the task (e.g., tracking, camera parameter assignment, and activity recognition) even if each camera is an autonomous agent. Thus, the cameras, acting as autonomous agents, analyze the raw data locally, exchange only distilled information that is relevant to the collaboration, and reach a shared, global analysis of the scene and the selection of camera parameters for the next image.

Although there are a number of methods in video analysis that deal with multiple cameras, and even camera networks (see [1] and the references therein), distributed and integrated processing and sensing in camera networks has received very little attention. On the other hand, distributed algorithms have been extensively studied in the multiagent systems and cooperative control literature (e.g., see [2]–[4]). Methods have been developed for reaching a consensus on a state independently observed by multiple sensors. In this article, we show how to build upon these results for integrated sensing and analysis in distributed camera networks for various wide-area scene-understanding problems.

AN INTEGRATED SENSING AND ANALYSIS FRAMEWORK

We show how a closed-loop framework for dynamic scene analysis in a reconfigurable, distributed PTZ camera network can be developed by integrating a number of component parts that have been studied more or less separately. The goal is to optimize the performance of various dynamic scene-analysis criteria, for example, minimizing the tracking error, getting a shot at the desired resolution and pose, getting a shot of the face for identification, or a combination of the above. To achieve this, it is necessary that the camera parameters be selected in an intelligent and collaborative manner. Each of the cameras in such a network will have its own embedded target detection and association module, a distributed tracker that provides an estimate on the state of each target in the scene, and finally, a distributed camera-control mechanism. More complex functionalities like behavior recognition can also be included based on the application requirements. Neighboring cameras can communicate with each other so as to fulfill each of these tasks. Some major camera-specific issues need to be addressed, including the fact that cameras are directional sensors, targets are dynamic, and we need to simultaneously solve for the PTZ parameter settings of the cameras and the states of the targets. The integrated system structure is shown in Figure 1.



[FIG1] Overall system diagram depicting a framework for integrated sensing and analysis in a reconfigurable, distributed camera network. The user criteria can define what performance metrics the network will optimize. The user criteria could include covering the entire area at a desired resolution, obtaining facial shots, maximizing image resolution, and so on.

The low-level processing module computes the image plane positions for targets. Through calibration, we know about the transformation between the image and ground plane coordinates, which can be used for data association. An overview of calibration and distributed data-association strategies is presented in the next section. These features, along with their measurement error, are then passed on to the distributed tracker associated with each target, which then combines the information from all cameras to come to a consensus regarding the estimated state and associated error covariance of each target. An overview of the distributed tracking approaches is presented in the "Distributed Tracking in Camera Networks" section.

[THE LOW-LEVEL PROCESSING MODULE COMPUTES THE IMAGE PLANE POSITIONS FOR TARGETS.]

The camera-control module attempts to optimize the scene-analysis performance at the next imaging time instant by selecting the desired values for the camera parameters, resulting in measurements that optimize criteria that are specified by the user, such as minimizing the estimated error covariance of the tracker, maximizing the resolution of the targets, and minimizing the risk of failing to obtain an image of the target. The issues involved in representing these criteria mathematically and the strategies for their distributed optimization are described in the "Reconfiguration of Distributed Camera Networks" section. We show how a game-theoretic distributed-optimization approach to target assignment can be adapted for this purpose. We will also describe a test bed for evaluating such systems, show results in a specific real-life scenario, and provide a comparative performance analysis.

DATA ASSOCIATION AND CALIBRATION IN DISTRIBUTED CAMERA NETWORKS

In the initial work on scene analysis in camera networks, particular interest was focused on learning a network topology, i.e., learning the traffic patterns between the entry/exit points in different views of the cameras. This is critical for establishing associations between multiple views. The authors in [5] used the location and velocity of objects moving across multiple nonoverlapping cameras to estimate the calibration parameters of the cameras and the targets' trajectories. In [6], the links between camera views were identified by exploiting the statistical consistency of the observation data. A framework for handoff in environments that are covered by multiple cameras by finding the limits of the FOV of a camera in other cameras was described in [7]. This method was adopted in [8] with consideration of power and bandwidth constraints with wireless embedded smart cameras. These methods were designed for a network composed of static cameras, and the dynamics of active camera networks was not taken into account.

Much effort has been devoted to the study of data associations in multitarget tracking, but many challenges remain in their applicability to camera networks, especially in distributed environments. Multihypothesis tracking (MHT) [9] and joint probabilistic data-association filters (JPDAFs) [10] are two representative methods. To overcome the large computational cost of MHT and JPDAFs, various optimization algorithms such as linear programming [11] and the Hungarian algorithm [12] are used for data association. In [13], the authors proposed a min-cost flow framework for global optimal data association. There are some papers on applying these techniques to camera networks, with a consideration of the geometrical relationship between cameras. In [14], the data association across cameras was achieved by extending the min-cost flow algorithm to camera networks. However, it is not straightforward to apply the approach to distributed camera networks. An intercamera matching method is presented in [15] by exploiting geometrical independence properties. The authors also considered the communication efficiency aspect through compressive sensing, which has the potential to be used in a distributed manner. The problem of distributed data association has been addressed in [16] and [17]. Below, we briefly describe these two approaches.

In distributed architectures, usually a network topology is considered where a camera node can only communicate with a node directly connected to it (neighboring node). The goal of a distributed approach for data association in camera networks is that the cameras must come to a consensus only by communicating with their network neighbors and without sending all the information to each other or to a centralized server.

In [16], a probabilistic data-association technique called JPDA was used. The system was initialized with the correct number of targets. Then at each time step, each of the cameras updated the estimated targets' positions with a probabilistic fusion of all its own observations. Next, they exchanged information with their neighboring cameras and associated the closest tracks to each other. After this data-association step, the associated tracked entities from neighboring cameras were fused together using a Kalman consensus framework. In [17], a graphical method is used to solve the data-association problem in a distributed framework. The virtual nodes were assumed for each camera, target, and nonoverlapping region, allowing distributed algorithms for graphical models, such as message-passing algorithm, to be used to solve the data-association problem. In reality, these virtual nodes may be available in a centralized server or may be distributed among the cameras. The prior spatial distribution of the targets was assumed to be known by the cameras covering each particular region. Then by sharing the data association confidences with each other, all the nodes come to an agreement about the data association. The camera calibration was assumed to be known in both of these distributed data-association papers.

The field of distributed calibration of camera networks has also garnered some interest in recent years. Average consensus-based methods as in [18] and graphical methods as in [19] have been used to design distributed estimation algorithms for calibration parameters. However, distributed solution of the calibration parameters requires reliable data association to be available at each of the nodes. From the above discussion, we can see that distributed data association and calibration are closely interlinked. We are not yet familiar with any work on distributed camera networks that looks at these two problems jointly, and this field is becoming a very interesting topic for future research. For the experimental analysis in this article, we implement the distributed data association as a special case of [16], such that the measurements are assigned to the targets based on the nearest neighbor criterion. The same criterion is also applied for target association across cameras.

[COOPERATION MEANS GIVING CONSENT TO PROVIDING ONE'S STATE AND FOLLOWING A COMMON PROTOCOL THAT SERVES THE GROUP'S OBJECTIVE.]

DISTRIBUTED TRACKING IN CAMERA NETWORKS

In this section, we address the aspect of tracking in distributed camera networks. We start with a review of the existing literature and then present a method for consensus-based tracking in a distributed environment. One of the earliest methods on tracking in a camera network is given in [20], where the correspondences of tracks between different camera views are constructed using a bipartite graph-matching strategy. In [21], a multiobjective optimization framework was presented for tracking in a camera network. There has also been recent work on tracking people in a multicamera setup [22], [23] by exploiting the principal axes of the targets or using a planar homography constraint. However, none of these methods address the issue of distributed processing.

In [24], a partially distributed target-tracking approach using a cluster-based Kalman filter was proposed. Here, a camera is selected as a cluster head that aggregates all the measurements of a target to estimate its position using a Kalman filter and sends that estimate to a central base station. Because of the presence of cluster heads and a central station, this is not a completely distributed approach. A related work that deals with tracking targets in a camera network with PTZ cameras is given in [25]. Here, the authors proposed a mixture between a distributed and centralized scheme using both static and PTZ cameras in a virtual camera network environment.

In the multiagent systems community, completely distributed estimation methods have been proposed using consensus-based ideas. In a network of agents, consensus can be defined as reaching an agreement through cooperation regarding a certain quantity of interest that depends on the information available through measurements from all agents. An interaction rule that specifies the information exchange between an agent and all of its neighbors in the network and the method by which the information is fused is called a consensus algorithm (or protocol). Cooperation means giving consent to providing one's state and following a common protocol that serves the group's objective. The interaction topology of a network of sensors is represented using a graph $G=(V,E)$, with the set of nodes $V=\{1,2,\ldots,n\}$ and edges $E\subseteq V\times V$. Each sensor node $i=1,\ldots,n$ maintains an estimate $\hat{\mathbf{x}}_i\in\mathbb{R}^m$ of a quantity $\mathbf{x}_i\in\mathbb{R}^m$. Consensus is achieved when $\hat{\mathbf{x}}_1=\hat{\mathbf{x}}_2=\cdots=\hat{\mathbf{x}}_n$, which is an n-dimensional subspace of $\mathbb{R}^{mn}$. A thorough review of consensus in networked multiagent systems can be found in [26].

A distributed Kalman-consensus filter, and subsequent variants, was proposed in [2], [27], and [28]. This was a completely distributed solution for estimating the dynamic state of a moving target. However, there are some major issues in applying the method to camera networks due to the nature of video sensors. Cameras are directional sensors, with each having a limited view of the entire theater of action and the complexity of the data requires high bandwidth to implement full communication. We will next show how the basic approaches on consensus for distributed estimation in the multiagent system literature can be applied for designing a consensus-based tracking algorithm in camera networks.

KALMAN-CONSENSUS TRACKING IN CAMERA NETWORKS

Below is a brief overview of the Kalman consensus tracker with an emphasis on the modifications needed for it to be applied to a camera network. Details on the approach can be found in [29].

MATHEMATICAL FRAMEWORK

Let $\mathcal{C}$ be the set of all cameras in the network. We can then define the subset of all cameras viewing target T_l as $\mathcal{C}_l^v\subset\mathcal{C}$ and the rest of the cameras as $\mathcal{C}_l^{v-}\subset\mathcal{C}$. Each camera C_i will also have its set of neighboring cameras $\mathcal{C}_i^n\subset\mathcal{C}$. Based on the communication constraints due to bandwidth limitations and network connections, we define the set $\mathcal{C}_i^n$ as all the cameras with which C_i is able to communicate directly. In other words, C_i can assume that no cameras other than its neighbors $\mathcal{C}_i^n$ exist, as no information flows directly from the nonneighboring cameras to C_i. Note that the set of neighbors need not be geographical neighbors. We also define the set of overlapping cameras of C_i as $\mathcal{C}_i^o\subset\mathcal{C}$; since all the cameras can change their PTZ parameters and therefore have several possible fields of view, we define the set $\mathcal{C}_i^o$ as all the cameras with which C_i can potentially have an overlapping FOV. By definition, it becomes clear that, for each $C_i\in\mathcal{C}_l^v$, it is true that $\mathcal{C}_l^v\subset\{\mathcal{C}_i^o\bigcup C_i\}$. We define $\mathcal{C}_i^c\subset\mathcal{C}$ as the connected component that C_i is in. We assume $\mathcal{C}_i^o\subset\mathcal{C}_i^c$, that is to say, C_i is able to exchange information with its overlapping cameras directly or via other cameras.

We consider the situation where targets are moving on a ground plane, and a homography between each camera's image plane and the ground plane is known. We will show how state vector estimation for each target by each camera (i.e., each camera's estimates based on its individual measurements) can

ALGORITHM 1

Distributed Kalman-consensus tracking algorithm performed by every C_i at discrete time step k. The state estimate of T_l by C_i is represented by $\boldsymbol{x}_i^l$ with an error-covariance matrix $\mathbf{P}_i^l$.

```
Input: $\bar{\mathbf{x}}_i^l$ and $\mathbf{P}_i^l$ valid at $k$ using measurements from time step $k-1$
for each $T_l$ that is being viewed by $\{\mathcal{C}_l^c \cup C_i\}$ do
    Obtain measurement $\mathbf{z}_i^l$ with covariance $\mathbf{R}_i^l$
    Compute information vector and matrix
        $\mathbf{u}_i^l = \mathbf{F}_i^{l^T}(\mathbf{R}_i^l)^{-1}\mathbf{z}_i^l,\ \ \mathbf{U}_i^l = \mathbf{F}_i^{l^T}(\mathbf{R}_i^l)^{-1}\mathbf{F}_i^l$
    Send messages $\mathbf{m}_i^l = (\mathbf{u}_i^l, \mathbf{U}_i^l, \bar{\mathbf{x}}_i^l)$ to neighboring cameras $\mathcal{C}_i^n$
    Receive messages $\mathbf{m}_j = (\mathbf{u}_j^l, \mathbf{U}_j^l, \bar{\mathbf{x}}_j^l)$ from all cameras $C_j \in \mathcal{C}_i^n$
    Fuse information matrices and vectors
        $\mathbf{y}_i^l = \sum_{j \in (C_i \cup \mathcal{C}_i^n)} \mathbf{u}_j^l,\ \ \mathbf{S}_i^l = \sum_{j \in (C_i \cup \mathcal{C}_i^n)} \mathbf{U}_j^l$    (S1)
    Compute the Kalman-consensus state estimate
        $\mathbf{M}_i^l = ((\mathbf{P}_i^l)^{-1} + \mathbf{S}_i^l)^{-1}$
        $\hat{\mathbf{x}}_i^l = \bar{\mathbf{x}}_i^l + \mathbf{M}_i^l(\mathbf{y}_i^l - \mathbf{S}_i^l\bar{\mathbf{x}}_i^l) + \gamma\mathbf{M}_i^l \sum_{j \in \mathcal{C}_i^n}(\bar{\mathbf{x}}_j^l - \bar{\mathbf{x}}_i^l)$
        $\gamma = 1/(\|\mathbf{M}_i^l\| + 1),\ \|\mathbf{X}\| = ((tr(\mathbf{X}^T\mathbf{X}))^{\frac{1}{2}}$    (S2)
    Propagate the state and error-covariance matrix from time step $k$ to $k+1$
        $\mathbf{P}_i^l \leftarrow \mathbf{A}^l\mathbf{M}_i^l\mathbf{A}^{l^T} + \mathbf{B}^l\mathbf{Q}^l\mathbf{B}^{l^T},\ \bar{\mathbf{x}}_i^l \leftarrow \mathbf{A}^l\hat{\mathbf{x}}_i^l$    (S3)
end for
```

be combined together through a consensus scheme. This method is independent of the tracking scheme employed in each camera. If the network of cameras is connected, then a consensus is achieved across the entire network.

The first step in the process is to model the motion of a target T_l on the ground plane, as observed by camera C_i. Assuming a linear dynamical system with time propagation and observation models, we have

$$\mathbf{x}^l(k+1) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \mathbf{x}^l(k) + \mathbf{w}^l(k); \quad \mathbf{x}^l(0) \tag{1}$$

$$\mathbf{z}_i^l(k) = \mathbf{F}_i^l(k)\mathbf{x}^l(k) + \mathbf{v}_i^l(k), \tag{2}$$

where $\mathbf{w}^l(k)$ and $\mathbf{v}_i^l(k)$ are zero mean, white, Gaussian noise ($\mathbf{w}^l(k) \sim \mathcal{N}(0, \mathbf{Q}^l)$, $\mathbf{v}_i(k) \sim \mathcal{N}(0, \mathbf{R}_i^l)$ and $\mathbf{x}^l(0) \sim \mathcal{N}(\mathbf{x}_0^l, \mathbf{P}_0)$ is the initial state of the target. $\mathbf{Q}$ and $\mathbf{R}$ are the covariance matrices of Gaussian noise. We define the state of the target at time step k as $\mathbf{x}^l(k) = (x^l(k), y^l(k), \dot{x}^l(k), \dot{y}^l(k))^T$, where $(x^l(k), y^l(k))$ and $(\dot{x}^l(k), \dot{y}^l(k))$ are the position and velocity of target T_l in the x and y directions, respectively. The vector $\mathbf{x}_i^l$ is the state of target T_l by C_i based on the measurements in C_i only. The vector $\mathbf{z}_i^l(k)$ is the noisy measurement of target T_l, i.e., the sensed position $(x_i^l(k), y_i^l(k))$, at camera C_i. $\mathbf{z}_i^l(k)$ can be measured on either ground or image planes. If $\mathbf{z}_i^l(k)$ is measured on the ground plane, then $\mathbf{F}_i^l = [\mathbf{I}_2\ \ \mathbf{0}_2]$, where $\mathbf{I}_2$ is a 2×2 identity matrix and $\mathbf{0}_2$ is a 2×2 matrix with zero entities. If $\mathbf{z}_i^l(k)$ is measured on the image plane, then $\mathbf{F}_i^l = [\widetilde{F}_i\ \ \mathbf{0}_2]$ where $\widetilde{F}_i$ denotes the mapping from ground to the image plane of C_i, which can be derived from the homography. Between each camera's image and ground planes, the homography can be estimated by marking corresponding pairs of feature points on the image and ground planes. In our implementation, the camera parameter setting is discretized, i.e., each camera has a finite number of candidate parameter settings. The homography is precalculated for each possible setting.

ALGORITHM DESCRIPTION

The Kalman-consensus distributed tracking algorithm is presented in "Algorithm 1," which is similar to the one in [27], but considers the different nature of camera sensors. The formal analysis of Algorithm 1 can be found in [28]. We describe it for the general system model of (1) and (2) and is applicable for the two special cases described above. This algorithm is performed in a distributed fashion by each camera node C_i.

If C_i is viewing a target T_l, it obtains T_l's measurements $\mathbf{z}_i^l$ and computes the corresponding information vector $\mathbf{u}_i^l$ and matrix $\mathbf{U}_i^l$. $\mathbf{u}_i^l$ and $\mathbf{U}_i^l$ are the sensor data and inverse-covariance matrix on T_l of camera C_i. The information fusion of multiple cameras can be achieved by summing $\mathbf{u}_i^l$ and $\mathbf{U}_i^l$ over all cameras. Similar to [27] for those cameras that cannot observe a certain target, the information vectors and matrices it sends to other cameras for that target are defined to be $\mathbf{0}$, by assuming that their output matrices on T_l are zero, i.e., $\mathbf{F}_i^l = \mathbf{0}$. If $C_i \in \mathcal{C}_l^v$ and the communication graph for $\mathcal{C}_l^v$ is fully connected, such that C_i can receive information from all the other cameras viewing the same target, the local state estimation at C_i by fusing information vectors and matrixes is the same as the central estimation, wherein all the cameras send their observations to a central processor, and tracking is centrally preformed. However, in a more typical situation, the neighbors of each camera are different; therefore, at each time instant, the information each camera receives to fuse may also be different. There is no guarantee that the state estimates at different cameras remain cohesive. Thus, a consensus step is implemented as part of the estimation step [see (S2) in "Algorithm 1"]. It is proved in [28] that all estimators asymptotically reach an unbiased consensus, i.e., $\hat{\mathbf{x}}_1 = \cdots = \hat{\mathbf{x}}_n = \mathbf{x}$, with time k growing to infinity.

HANDOFF IN CONSENSUS-TRACKING ALGORITHMS

For wide-area tracking algorithms, it is necessary to develop suitable handoff strategies between the cameras. Through the Kalman-consensus algorithm, each C_i has a consensus-based ground plane state estimate of each target that is being viewed by the cameras with which C_i can exchange information directly or indirectly, even if C_i has never seen some of the targets. For the case of overlapping cameras, a target T_l will move from one camera C_i's FOV to the FOV of an overlapping camera $C_j \in \mathcal{C}_i^o$. Moreover, C_i can exchange information with its overlapping cameras, $\mathcal{C}_i^o$, directly or via other cameras. Therefore, C_j can take over the tracking of T_l and find the target correspondence in a seamless way, since the cameras had the knowledge of T_l's ground plane position through the consensus tracking before it even entered its FOV. If the target moves from one camera to another that is nonoverlapping, the distributed data-association strategies outlined above can be employed to find the correspondence and maintain the continuity of the track.

[FOR WIDE-AREA TRACKING ALGORITHMS, IT IS NECESSARY TO DEVELOP SUITABLE HANDOFF STRATEGIES BETWEEN THE CAMERAS.]

Another advantage of the fact that cameras have knowledge of all the targets in their neighborhood is that, in the event of a sudden failure of camera node C_i, the targets that were viewed by C_i are not suddenly lost by the camera network. Also, a camera may take a short amount of time to change its parameters to a new position in a nonstatic camera network. If no camera is viewing the target for the short amount of time it takes for the cameras to come to a new set of parameters, the target state estimate and covariance continue to propagate according to (S3). This does not translate to a significant decrease in tracking performance as seen in our experiments.

RECONFIGURATION OF DISTRIBUTED CAMERA NETWORKS

In the integrated sensing and analysis framework described earlier, the image-acquisition strategies should be informed by the output of the analysis modules. In this section, we will describe how to reconfigure a PTZ network in real time to achieve this.

Many modern vision networks consist of a mixture of static and PTZ cameras. The placement of these networks is determined at the moment of deployment. Optimal camera placement strategies were proposed in [30] and solved by using a camera-placement metric that captures occlusion in three-dimensional (3-D) environments using binary integer programming. In [31], a solution to the problem of optimal camera placement, given some coverage constraints, was presented and can be used to come up with an initial camera configuration. The large area covered by these networks results in many situations where the observed targets are often not imaged at desirable resolutions. The ability to actively control the PTZ cameras in such a network raises many interesting research issues that have not yet been widely addressed.

The path-planning inspired approach proposed in [32] used a mixed network of cameras. Static cameras were used to track all targets in a virtual environment, while PTZ cameras were assigned to obtain high-resolution video from the targets. This approach showed that, given the predicted tracks of all the targets, a plan of one-to-one mappings between cameras and targets can be formed to acquire high-resolution videos. A method for determining good sensor configurations that would maximize performance measures was introduced in [33]. The configuration framework is based on the presence of random occluding objects, and two techniques are proposed to analyze the visibility of the objects. These methods address the camera network reconfiguration problem in a centralized manner, while a distributed solution is more desirable in many application domains.

A recent distributed approach in [34] uses the expectation-maximization (EM) algorithm to find the optimal configuration of PTZ cameras, given a map of activities. The value of each discretized ground coordinate is determined using the map. This approach, upon convergence of the EM algorithm, provides the PTZ settings to optimally cover an area. A framework for distributed control and target assignment in camera networks was presented in [35], in which the cooperative network-control ideas based on multiplayer learning in games [36] were used. The result was a decision-making process that aims to optimize a certain global criterion based on individual decisions by each component (sensor) and the decisions of other interconnected components. The proposed method was related to the vehicle–target assignment problem using game theory, as was presented in [3] where a group of vehicles is expected to optimally assign themselves to a set of targets. However, in that work, the targets were not dynamic, and each vehicle was assigned to one target. In a camera network, each camera can observe multiple targets, and multiple cameras can observe each target (many-to-many mapping).

In keeping with the overall theme of this article, we now provide an overview of how certain network control ideas can be used for camera network reconfiguration, the final module in the integrated sensing and analysis framework of Figure 1.

GAME-THEORETIC FRAMEWORK FOR CAMERA NETWORK RECONFIGURATION

The objective of the control module in Figure 1 is to develop a decentralized strategy for determining camera parameters that relies on local decision making at the camera nodes while being aligned with a suitable global criterion, e.g., persistently observing multiple targets at specified resolutions. We design each sensor to be a rational decision maker and formulate our problem as a multiplayer game, where each camera is a player and interested in optimizing its own utility. By designing the camera utility functions to be aligned with the global utility function, the game is a potential game with the global utility function being the potential function [37]. Then the agreeable settings of cameras, i.e., the settings at which there is no incentive for any camera to unilaterally deviate (Nash equilibria), should lead to high, ideally maximal, global utility.

The first step is to find suitable local utility functions such that the objectives of each camera are localized to that camera, yet aligned with the global utility function. The second step is to develop an appropriate negotiation mechanism between cameras to ensure convergence of the distributed solution towards the global solution. The actual computation of these functions depends upon the analysis of the sensed video, e.g., the tracks obtained by the tracking modules. Below, we provide an overview of how such utility functions can be designed for various scene-analysis tasks.

Let us consider N_t targets in the entire area of deployment and N_c sensors that need to be assigned to these targets. In the camera network setup, each target will be represented by a location vector and a resolution parameter. Each target has an associated resolution parameter $r_l, l = 1, \ldots, N_t$. Camera $C_i \in \mathcal{C}$ will select its own set of parameters $a_i \subseteq \mathcal{A}_i$, where $\mathcal{A}_i$ is the parameter profile that C_i can select from, to optimize its own utility function $U_{C_i}(a_i)$. Our problem is to design these utility functions and choose appropriate negotiation procedures that lead to a mutually agreeable parameter settings of the cameras resulting in meeting the global criterion [37].

MANY MODERN VISION NETWORKS CONSIST OF A MIXTURE OF STATIC AND PTZ CAMERAS.

A well-known concept in game theory is the notion of Nash equilibrium. In the context of our image-network problem, it will be defined as a choice of parameter settings $a^* = (a_1^*, \ldots, a_{N_c}^*)$ such that no sensor could improve its utility further by deviating from a^*, i.e., by choosing a different set of parameters, the utility functions of all cameras cannot be improved further as expressed in (3). Obviously, this is a function of time since the targets are dynamic, and the cameras could also be mobile or capable of panning, tilting, and zooming. For example, for the problem of tracking all targets, a Nash equilibrium will be reached at a particular instant of time when all the cameras are collectively observing all the targets in the deployment region at an acceptable resolution and there is no advantage for a particular camera to choose some other target to observe. Mathematically, if a_{-i} denotes the collection of parameter settings of all cameras except camera C_i, then a^* is a pure Nash equilibrium if

$$U_{C_i}(a_i^*, a_{-i}^*) = \max_{a_i \in \mathcal{A}_i} U_{C_i}(a_i, a_{-i}^*), \forall C_i \in \mathcal{C}. \quad (3)$$

DESIGNING UTILITY FUNCTIONS

We outline below how the different utility functions can be designed for various scene-analysis tasks.

TARGET UTILITY

The target utility $U_{T_l}(\mathcal{A})$ refers to the utility gained in observing a particular target under the constraints specified by the application. Some examples are provided below.

1) *View Criterion*: The view utility $M_{V_l}(\mathcal{A})$ determines whether the resolution and/or pose requirement for target T_l is satisfied by the camera network using a particular parameter profile $\mathcal{A} = \{a_1, a_2, \ldots, a_{N_c}\}$. Let p_{il} be the probability that target T_l is viewed at the desired resolution by camera C_i. Then, $M_{V_l}(\mathcal{A})$ can be defined for the resolution requirement as

$$M_{V_l}(\mathcal{A}) = 1 - \prod_i (1 - \mathrm{p}_{il}), \quad (4)$$

where p_{il} is defined as

$$p_{il} = \begin{cases} 1 - e^{-\lambda \frac{r_{il}}{r_{\max}}} & \text{if } \mathrm{r}_{il} > r_0, \\ 0 & \text{otherwise,} \end{cases} \quad (5)$$

r_0 is the minimum acceptable resolution in terms of target pixel height at which the targets should be viewed and $r_{\max}$ is the height in pixels of C_i's image plane. r_{il} is the resolution at which T_l is being viewed by C_i. The term λ can be changed according to how well the single-view tracking algorithm performs as the height of the target on the image plane increases or decreases. It can be seen from (5) that the higher the resolution at which T_l is being viewed, the higher the probability acquired by T_l.

This utility could also be modified to prioritize certain poses or facial shots by factoring in pose and facial-view angle or resolution. For example, if a facial shot is required for identification, which is view dependent, a view angle factor can be defined as $\gamma = \cos(\theta_{T_l} - \arccos((\vec{O}_{T_l} \cdot \vec{O}_{C_i})/(\|\vec{O}_{T_l}\| \, \|\vec{O}_{C_i}\|))$ where θ_{T_l} is the desired view angle of T_l, $\vec{O}_{T_l}$ is the orientation of T_l and $\vec{O}_{C_i}$ is the orientation of camera C_i. Thus, the view utility becomes $M_{V_l}(\mathcal{A}) = \gamma[1 - \prod_i (1 - \mathrm{p}_{il})]$.

2) *Tracking Criterion*: The purpose of the tracking utility is to quantify how well the tracking module in camera C_i is tracking target T_l, given the settings a_i. We can define this utility using the error-covariance matrices $\mathbf{P}_i^l$ computed by the Kalman-consensus filter and the measurement matrix $\mathbf{F}_i^l$, i.e.,

$$M_{Tr_l}(\mathcal{A}) = \exp\{-\mathrm{Trace}(\mathbf{P}_i^{l+})\}, \quad (6)$$

where $\mathbf{P}_i^{l+}$ is defined as

$$(\mathbf{P}_i^{l+})^{-1} = (\mathbf{P}_i^l)^{-1} + \sum_{j \in (C_i \cup \mathcal{C}_i^n)} (\mathbf{F}_j^l(\mathcal{A}))^T \mathbf{R}_j^l(\mathcal{A})^{-1} \mathbf{F}_j^l(\mathcal{A}). \quad (7)$$

$\mathbf{R}$ is the measurement-error covariance and $\mathcal{C}_i^n$ is the neighborhood of C_i, i.e., the cameras that can directly communicate with C_i. Note that $(\mathbf{P}_i^{l+})^{-1}$ is the information we expect to get from the next image set based on the camera settings $\mathcal{A}$; hence, the right-hand side of (6) is a function of $\mathcal{A}$. Using this estimated error-covariance update equation, we can predict the covariance $\mathbf{P}_i^{l+}$, given some settings profile $\mathcal{A}$. By choosing a settings profile that maximizes the tracking utility, we are selecting a set of measurements that will minimize the estimated error covariance within the Kalman-consensus tracker.

GLOBAL UTILITY

The global utility U_G describes the desirability of the settings profile **a**, given the criteria that must be satisfied by the entire camera network. From the target utility functions, we can now define the global utility function as the sum of the utilities generated by observing all the targets weighted by their importance, i.e., $U_g(\mathcal{A}) = \sum_l V_l \cdot U_{T_l}(\mathcal{A})$, where V_l denotes the importance of target T_l. V_l can be set based on the user's input. For example, if a user specifies a target to view, then the importance of this target should be set much higher than the others.

CAMERA UTILITY

We define the utility of a camera observing a particular target as its marginal contribution to the global utility as a result of this action, i.e., the camera utility is the change in the global utility as a result of that sensor observing that particular target as opposed to not observing it. Since each camera C_i in our distributed camera network can influence the target assignments of its neighbors, we will define the camera utility depending on the assignments of its neighboring cameras, i.e.,

$$U_{C_i}(\mathcal{A}) = U_g(\mathcal{A}) - U_g(a_{-i}) = \sum_l V_l(U_{T_l}(\mathcal{A}) - U_{T_l}(a_{-i})), \quad (8)$$

where $a_{-i} = \mathcal{A} - a_i$ is the parameter profile of all the cameras except C_i.

OPTIMIZATION STRATEGY

At each control time instant, all the cameras in the network will team up to play a potential game. At any step of the negotiation, a camera C_i is randomly chosen from a pool of cameras in the network according to a uniform distribution, and only this camera is given the chance to update its proposed parameter settings. At negotiation step m, camera C_i searches for a set of parameters that maximizes its camera utility based on other camera parameters from the previous negotiation step and broadcasts its choice to all its neighboring cameras (more details on the negotiation mechanism can be found in [38]). This occurs until no camera can increase its own utility by changing its parameters. This is called the Nash equilibrium of the game. The camera utility functions being aligned with the global utility, combined with appropriate negotiation strategies, guarantees that, at each time step, the set of parameters chosen for the network of cameras is an optimal solution for the system's goals.

This distributed optimization strategy is similar to the EM approach used in [34]. The prediction of the global utility is comparable to the expectation step, and the maximization of the camera utility can be related to the maximization step in the EM approach. Compared with the other camera reconfiguration strategies described above [32], [33], this is a completely distributed system. It is a more general framework than that in [34], since it explicitly shows how the analysis and sensing phases can be linked together in a closed-loop system (see the design of the utility functions, whose parameters are dependent on the tracker outputs).

PERFORMANCE ANALYSIS AND DISCUSSION

We provide below an experimental evaluation of the integrated sensing and analysis framework and a comparison with other approaches.

EXPERIMENTAL SETUP

Our camera network is composed of five PTZ cameras looking over an outdoor area of approximately 600 m^2. The area was divided into a number of grids, each of size 1 cm^2. Tracked targets were assumed to have a height of 1.80 m. Each camera acquired images of resolution 640×480 pixels. The cameras were arranged such that four of the cameras were located on the same side of the courtyard, with one camera on the opposite side. In the region of interest, there were five targets in addition to two entrances and exits. Since the entrances and exits must be always monitored, we treated them as static virtual targets, leading to a total of seven targets. Each camera in our setup is an independent entity connected through a wireless network, with the entire system running in real time.

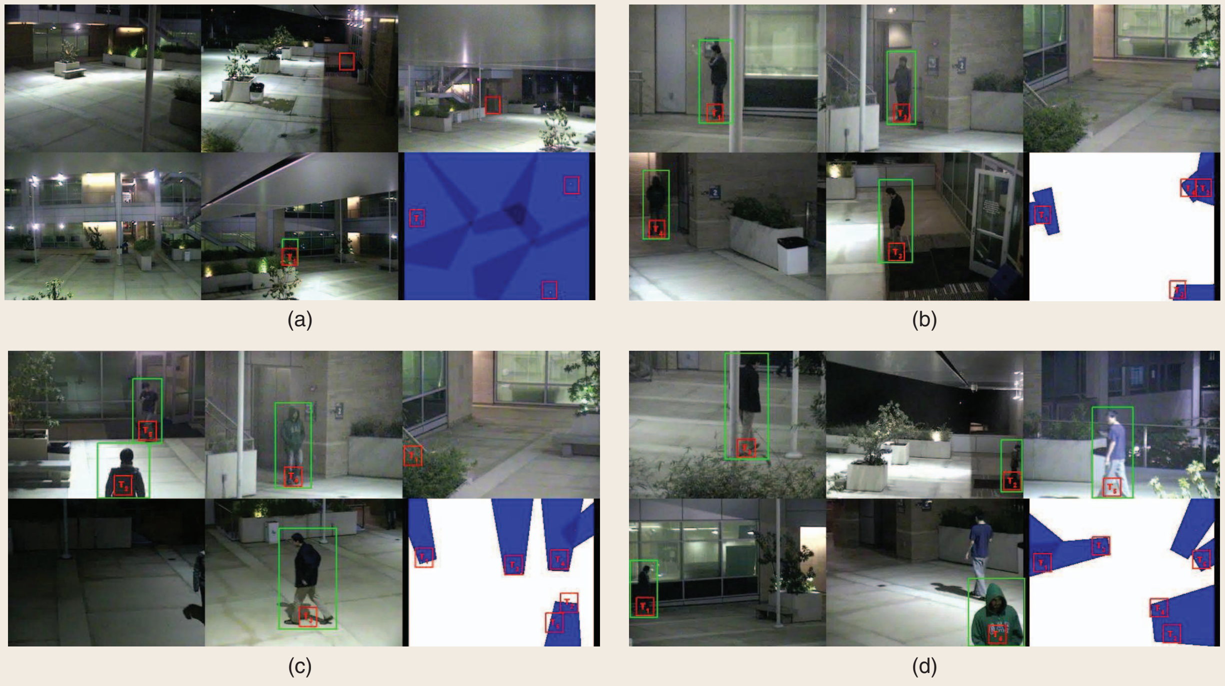
[AT EACH CONTROL TIME INSTANT, ALL THE CAMERAS IN THE NETWORK WILL TEAM UP TO PLAY A POTENTIAL GAME.]

We considered two tasks. The first task was to cover the entire area (area coverage), while the second one was to cover only the targets and the entry/exit regions (target coverage). We defined the target utility for viewing target T_l from the tracking and view utility functions defined in (4) and (6), i.e., $U_{T_l}(\mathcal{A}) = w_1^l M_{Tr_l}(\mathcal{A}) + w_2^l M_{V_l}(\mathcal{A})$, where w_1^l and w_2^l are the weights that can be set by the user and were set to one for this experiment. This can be easily adapted to the case of covering the entire area by treating every grid point as a virtual target (possibly viewed at a lower resolution) and the tracking utility for them is set to be zero. The acceptable resolution in terms of target pixel height [r_0 as in (5)] is set to be 70 here. We can then determine the settings profile that maximize the global utility based on analysis of $U_G(\mathcal{A})$ for both these cases.

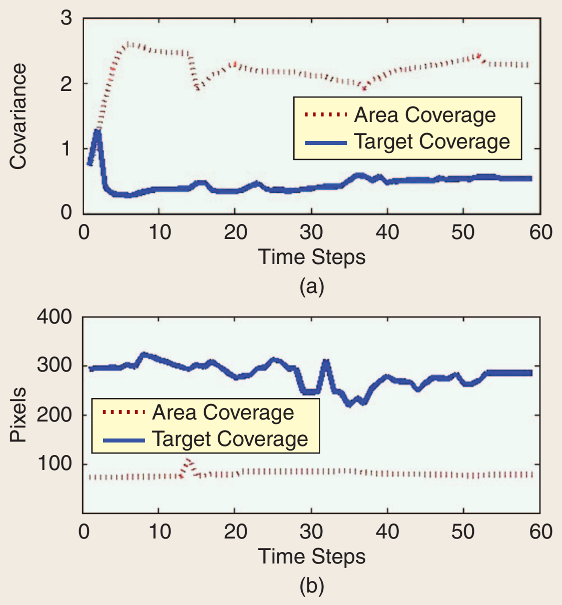
PERFORMANCE ANALYSIS

At initialization, the entire region under surveillance is divided into grids, and each grid is treated as a virtual target. All of the cameras apply the utility function to cover the entire region and to detect targets already in the region. The target-detection modules in each camera determine the image plane position of each target in its FOV. This information is then passed along to the Kalman-consensus filter and is processed along with the information from the filters running on neighboring cameras.

We compared the two scenarios: area coverage and target coverage. The targets followed the same path through the courtyard during the collection of data for both cases. Figure 2



[FIG2] Dynamic camera-control images. Blue regions mark the FOVs. The targets are marked in green with a red label. This figure is best viewed on a computer monitor. (The video as well as the experimental results and example code are available at http://www.ee.ucr.edu/ amitrc/CameraNetworks.php.) Time steps are as follows: (a) $k = 0$, (b) $k = 2$, (c) $k = 19$, and (d) $k = 36$.



[FIG3] Comparison of the average tracker covariance and resolution of all targets being actively tracked by a system for target coverage versus the one for area coverage.

shows the images captured by the actively controlled camera network at different time steps. Figure 2(a) shows the result for the area coverage as the initialization. Figure 2(b)–(d) shows the results for the target coverage. Since the targets are free to move about the region under surveillance, the cameras in the network are required to adjust their parameters dynamically to maintain shots of each target that optimize the utility functions. To acquire these shots, the camera network concedes a large unobserved area. We can see in Figure 3 that, as time progresses, the average trace of the covariance and the resolution of all targets settle at a significantly better value (compared to the area coverage) when the tracking and control modules are integrated together (as described in this article). This is because, at each time step, the camera network will choose a set of parameters that optimizes the utility, which is dependent on the error covariance of the Kalman-consensus filter. In the area-coverage problem, the camera network has to cover the entire area and take shots at lower resolutions, resulting in increased tracking error.

DISCUSSION

We address some performance issues related to distributed camera networks.

SCALABILITY

The data that need to be exchanged between cameras include information vector $\mathbf{u}$, information matrix $\mathbf{U}$, state estimate $\bar{\mathbf{x}}$ as shown in Algorithm 1, and PTZ parameters of cameras. As $\mathbf{U}$ depends on the settings of cameras, in practice, it does not need to be transmitted at every step as long as the cameras keep their current settings. $\mathbf{u}$ is a 4×1 vector, $\mathbf{U}$ is a 4×4 matrix, and $\bar{\mathbf{x}}$ is a 4×1 vector. The bandwidth capacity determines the precision of these values. Since these values are quantized, a quantization error is introduced into the tracking results if the bandwidth is reduced. For camera parameters, in practice, we discretize their parameter space (i.e., PTZ parameters) so each camera only has a finite number of possible settings to choose from. The PTZ parameters are sampled over their allowed ranges to reflect possible selections, e.g., the FOV of that specific setting should overlap with the region under surveillance. As a typical example, we select nine settings for pan, three settings for tilt, and two settings for zoom, which result in totally 54 candidate-parameter settings of a camera. The cameras only need to transmit the indices of their PTZ settings. Assuming there are N_T targets being viewed by a network of N_C cameras, each camera has N_p possible parameter settings, the frame rate is fr, and the precision of the values contained in $\mathbf{u}$, $\mathbf{U}$, $\bar{\mathbf{x}}$ and is of the type double (64 b each; the precision is more than enough for tracking), the upper bound of the bandwidth requirement (if every camera can view all the targets) is $[(4+16+4)\cdot 64\cdot N_T\cdot N_C+\log_2(N_p)\cdot N_C]\cdot fr$ b/s.

THE DATA ASSOCIATION ACROSS CAMERAS CAN ALSO BE ACHIEVED THROUGH THE BIPARTITE GRAPH MATCHING, BASED ON THE INFORMATION OF THE STATES OF TARGETS EXCHANGED BETWEEN CAMERAS.

LATENCY
The cameras used in our experiments can adjust their parameters very fast compared with the speed of targets; for example, the pan speed is 180 °/s and tilt speed is 140 °/s. So the latency of the cameras does not have any effect on the experimental results. For example, assume that the distance of a person to the optical center of a camera is 4 m and the person is walking parallel to that camera. Assuming the average human walking speed is about 5 km/h, that camera only needs to pan at the speed of around 20 °/s to keep the person in the center of its view, which is much lower than the moving speed of our camera. If for a moment a camera cannot obtain measurements for targets because of the latency, the information vectors and matrices it sends to other cameras are defined to be $\mathbf{0}$. It can be seen from (S1) and (S2) that the Kalman-consensus state estimate is achieved based on the other cameras' measurements. In addition, if no camera can stably observe the targets due to parameter adjustment, the state estimate is equal to $\bar{\mathbf{x}}$, the predicted states, and the error covariance increases because of the lack of measurements; however, it will drop again once the new measurements become available.

ACCURACY
In a single camera, the measurements are assigned to targets based on appearance (color) and position information (this is only for our experiments; the framework can use any other feature). The association could be achieved through an $O(n^{2.5})$ algorithm for maximum matchings in bipartite graphs, as used in [20]. Noisy measurements may introduce errors in association so as to affect the tracking accuracy. In such a case, a multiobjective optimization framework [21] can be used to correct the tracking error caused by the mismatch in data association. The data association across cameras can also be achieved through the bipartite graph matching, based on the information of the states of targets exchanged between cameras. There may exist errors in data association, i.e., different targets are wrongly associated across cameras, especially when they are very close in their state space. Due to the characteristics of the consensus algorithm, i.e., seeking the agreement of most participants, the tracking inaccuracy caused by incorrect intercamera data association can be minimized by integrating information from multiple cameras. The exact nature of how this will affect the performance is very much a function of the targets, their movements, the camera positions, and the area under surveillance.

COMPARATIVE ANALYSIS
In Table 1, we provide a comparison of the different methods in the literature that have looked into the problem of camera network reconfiguration. We note that they have different objective functions, architectures, and performance criteria. The area of distributed analysis in camera networks is very much evolving, and standard comparison metrics or experimental frameworks are not available. This should be an area of attention as this research matures.

The method proposed in [33] uses a simulated annealing approach to evaluate a globally optimal configuration for

[TABLE 1] COMPARISON BETWEEN CAMERA NETWORK RECONFIGURATION STRATEGIES.

APPROACH	OBJECTIVE	ARCHITECTURE	OUTCOMES
MITTAL AND DAVIS [33]	STATIC CAMERA PLACEMENT	CENTRALIZED	GLOBAL MAXIMA OF AREA COVERED WHILE CONSIDERING OCCLUSION
SOTO ET AL. [35]	AREA COVERAGE	DISTRIBUTED	LOCAL MAXIMA OF TOTAL AREA COVERED
PICIARELLI ET AL. [34]	WEIGHTED AREA COVERAGE BASED ON PRIOR ACTIVITY MAP	DISTRIBUTED	LOCAL MAXIMA OF WEIGHTED AREA COVERED
QURESHI AND TERZOPOULOS [32]	CAMERA-TO-TARGET ASSIGNMENT	CENTRALIZED	TRACK-BASED ONE-TO-ONE MAPPING (BETWEEN CAMERAS AND TARGETS) AND HANDOFF
INTEGRATED APPROACH	SATISFIES MULTIPLE CRITERIA	DISTRIBUTED	TRACK-BASED MANY-TO-MANY MAPPING (BETWEEN CAMERAS AND TARGETS)

static camera networks. As shown in the area-coverage case, our framework can also handle this situation with some minor changes in the view utility to model occlusion.

The approaches proposed in [32], [34], and [35] have an independent tracker running in the background. The authors in [34] use it to generate the prior (activity map) and then optimize camera settings on the weighed area. In [32], the authors use tracks to predict the paths of targets to do assignments requiring minimal assignment switching but break when targets mingle due to unreliable predictions. The work in [35], while using a distributed tracker, does not tie together the control and tracking modules. The integrated sensing and analysis approach described here is a generalization of all of these and provides a framework for optimizing the image-acquisition capabilities based on how well the system objectives are being met. This leads to an optimal allocation of resources and provides an overall efficiency to the system. This can be seen from Figure 3, where, in the case where the entire area needs to be covered, it leads to a situation in which the resources are being used to cover empty space most of the time (this would be the case using either [34] or [35]).

[THE APPROACH FOR INTEGRATED SENSING AND VIDEO ANALYSIS CAN LAY THE GROUNDWORK FOR SOLVING MANY PROBLEMS IN WIDE-AREA SCENE UNDERSTANDING IN CAMERA NETWORKS.]

CONCLUSIONS AND FUTURE WORK

In this article we discussed a framework to optimize various scene-analysis performance criteria through the distributed control of a reconfigurable camera network and studied its relation to the existing literature. We addressed three fundamental tasks: distributed data association, dynamic camera control, and distributed tracking using a Kalman-consensus filtering approach. We also provided a comparative analysis of work in this area. The approach for integrated sensing and video analysis can lay the groundwork for solving many problems in wide-area scene understanding in camera networks.

Research in camera networks as a collection of autonomous agents capable of sensing and reasoning about the environment is very much in its infancy. There are numerous problems with an interdisciplinary flavor. These include analysis of the effects of communication constraints such as power and bandwidth; path planning, and routing of mobile agents; robustness of the networks to partial outage due to malfunction or adversarial attacks; distributed data storage and retrieval; learning semantic models for complex tasks like behavior recognition; performance analysis of complex distributed systems; and visualization tools for dynamic high-dimensional data.

ACKNOWLEDGMENTS

This work was partially supported by National Science Foundation grant 0622176, Office of Naval Research grant N00014-09-1-0666, and Army Research Office grant W911NF-07-1-0485.

AUTHORS

Bi Song (bsong@ee.ucr.edu) received her B.S. and M.S. degrees in electronic engineering and information science from the University of Science and Technology of China, Hefei, in 2001 and 2004, respectively, and her Ph.D. degree in 2009 from the Department of Electrical Engineering at the University of California, Riverside. She is currently a postdoctoral scholar at the University of California, Riverside. Her main research interests include computer vision, image processing, and analysis.

Chong Ding (cding@cs.ucr.edu) received his B.S. degree in computer science from the University of California, Riverside, in 2008. He is currently a Ph.D. candidate in the Department of Computer Science at the same university. His research interests include intelligent camera networks, wide-area scene analysis, and distributed and real-time systems.

Ahmed T. Kamal (akamal@ee.ucr.edu) received his B.S. degree in electrical and electronic engineering from the Bangladesh University of Engineering and Technology, Dhaka, in 2008. He received his M.S. degree in electrical engineering from the University of California, Riverside, in 2010. He is currently a Ph.D. candidate in the Department of Electrical Engineering at the same university. His research interests include intelligent camera networks, wide-area scene analysis, activity recognition, and search and distributed information fusion.

Jay A. Farrell (farrell@ee.ucr.edu) received his B.S. degree in physics and electrical engineering from Iowa State University, in 1986 and his M.S. and Ph.D. degrees in electrical engineering from the University of Notre Dame in 1988 and 1989, respectively. He was a principal investigator at Charles Stark Draper Lab (1989–1994). He received the engineering vice president's Best Technical Publication Award in 1990 and recognition awards for outstanding performance and achievement in 1991 and 1993, respectively. He is a professor and former chair of the Department of Electrical Engineering at the University of California, Riverside. He was the vice president for Finance and vice president for Technical Activities for IEEE Control Systems Society (CSS) and is general chair of the 2012 IEEE Conference on Decision and Control. He is a Fellow of the IEEE and AAAS, a distinguished member of CSS, and author of more than 170 technical publications. He is author of the book *Aided Navigation: GPS with High Rate Sensors* (McGraw-Hill, 2008). He is also the coauthor of *The Global Positioning System and Inertial Navigation* (McGraw-Hill, 1998) and Adaptive *Approximation Based Control: Unifying Neural, Fuzzy and Traditional Adaptive Approximation Approaches* (Wiley, 2006).

Amit K. Roy-Chowdhury (amitrc@ee.ucr.edu) is an associate professor of electrical engineering and a cooperating faculty in the Department of Computer Science at the University of California, Riverside. He received his bachelor's degree in electrical engineering from Jadavpur University, Calcutta, India, his master's degree in systems science and automation from the Indian Institute of Science, Bangalore, India, and his Ph.D. degree in electrical engineering from the University of Maryland, College Park. His research interests are in the areas of image processing and analysis, computer vision, video communications, and statistical methods for signal analysis. His current research projects include intelligent camera networks, wide-area scene analysis, motion analysis in video, activity recognition and search, video-based biometrics (face and gait), biological video analysis, and distributed video compression. The work is supported by NSF, ONR, ARO, DARPA, and private industries. He has published about 100 papers in peer-reviewed journals, conferences, and edited books.

REFERENCES

[1] C. Micheloni, B. Rinner, and G. L. Foresti, "Video analysis in pan-tilt-zoom camera networks," *IEEE Signal Processing Mag*., vol. 27, no. 5, pp. 78–90, Sept. 2010.

[2] R. Olfati-Saber, "Distributed Kalman filtering for sensor networks," in *Proc. 46th IEEE Conf. Decision and Control*, New Orleans, LA, Dec. 2007, pp. 5492–5498.

[3] G. Arslan, J. Marden, and J. Shamma, "Autonomous vehicle-target assignment: A game-theoretical formulation," *ASME J. Dyn. Syst. Meas.* Control, vol. 129, no. 5, pp. 584–596, Sept. 2007.

[4] P. N. J. Hespanha and Y. Xu, "A survey of recent results in networked control systems," *Proc. IEEE (Special Issue on Technology of Networked Control Systems)*, vol. 95, no. 1, pp. 138–162, Jan. 2007.

[5] A. Rahimi and T. Darrell, "Simultaneous calibration and tracking with a network of non-overlapping sensors," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Washington, DC, July 2004, vol. 1, pp. 187–194.

[6] D. Markis, T. Ellis, and J. Black, "Bridging the gap between cameras," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Washington, DC, July 2004, vol. 2, pp. 205–210.

[7] S. Khan, O. Javed, Z. Rasheed, and M. Shah, "Camera handoff: Tracking in multiple uncalibrated stationary cameras," in *Proc. IEEE Workshop Human Motion, 2000,* p. 113.

[8] Y. Wang, S. Velipasalar, and M. Casares, "Cooperative object tracking and composite event detection with wireless embedded smart cameras," *IEEE Trans. Image Processing,* vol. 19, no. 10, pp. 2595–2613, Oct. 2010.

[9] D. Reid, "An algorithm for tracking multiple targets," *IEEE Trans. Automat*. Contr., vol. 24, no. 6, pp. 843–854, 1979.

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

[11] H. Jiang, S. Fels, and J. Little, "A linear programming approach for multiple object tracking," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Minneapolis, MN, June 2007, pp. 1–8.

[12] A. Perera, C. Srinivas, A. Hoogs, G. Brooksby, and W. Hu, "Multi-object tracking through simultaneous long occlusions and split-merge conditions," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, New York, NY, June 2006, vol. 1, pp. 666–673.

[13] L. Zhang, Y. Li, and R. Nevatia, "Global data association for multi-object tracking using network flows," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, Anchorage, AL, June 2008, pp. 1–8.

[14] M. Dixon, N. Jacobs, and R. Pless, "An efficient system for vehicle tracking in multi-camera networks," in *Proc. IEEE/ACM Int. Conf. Distributed Smart Cameras*, Como, Italy, Aug. 2009, pp. 1–8.

[15] E. Ermis, P. Clarot, P.-M. Jodoin, and V. Saligrama, "Activity based matching in distributed camera networks," *IEEE Trans. Image Processing*, vol. 19, no. 10, pp. 2564–2579, Oct. 2010.

[16] N. Sandell and R. Olfati-Saber, "Distributed data association for multi-target tracking in sensor networks," in *Proc. IEEE Conf. Decision and Control*, 2008, pp. 1085–1090.

[17] L. Chen, M. Cetin, and A. Willsky, "Distributed data association for multi-target tracking in sensor networks," in *Proc. Int. Conf. Information Fusion*, Philadelphia, PA, July 2005, pp. 9–16.

[18] E. Elhamifar and R. Vidal, "Distributed calibration of camera sensor networks," in *Proc. IEEE/ACM Int. Conf. Distributed Smart Cameras*, Como, Italy, Aug. 2009, pp. 1–8.

[19] D. Devarajan and R. Radke, "Calibrating distributed camera networks using belief propagation," *EURASIP J. Appl. Signal Process*., vol. 2007, no. 1, pp. 1–10, Jan. 2007.

[20] O. Javed, Z. Rasheed, K. Shafique, and M. Shah, "Tracking across multiple cameras with disjoint views," in *Proc. IEEE Int. Conf.* Computer Vision, Nice, France, Oct. 2003, vol. 2, pp. 952–957.

[21] B. Song and A. Roy-Chowdhury, "Robust tracking in a camera network: A multi-objective optimization framework," *IEEE J. Select. Topics Signal Processing (Special Issue on Distributed Processing in Vision Networks)*, vol. 2, no. 4, pp. 582–596, Aug. 2008.

[22] W. Du and J. Piater, "Multi-camera people tracking by collaborative particle filters and principal axis-based integration," in *Proc. Asian Conf. Computer Vision*, Tokyo, Japan, Nov. 2007, vol. 1, pp. 365–374.

[23] S. Khan and M. Shah, "A multiview approach to tracking people in crowded scenes using a planar homography constraint," in *Proc. Euro. Conf.* Computer Vision, Graz, Austria, May 2006, pp. 133–146.

[24] H. Medeiros, J. Park, and A. Kak, "Distributed object tracking using a cluster-based Kalman filter in wireless camera networks," *IEEE J. Select Topics Signal Processing*, vol. 2, no. 4, pp. 448–463, Aug. 2008.

[25] F. Qureshi and D. Terzopoulos, "Surveillance in virtual reality: System design and multi-camera control," in *Proc. IEEE Conf. Computer Vision and Pattern Recogniton*, Minneapolis, MN, June 2007, pp. 1–8.

[26] R. Olfati-Saber, J. Fax, and R. Murray, "Consensus and cooperation in networked multi-agent systems," *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, Jan. 2007.

[27] R. Olfati-Saber and N. F. Sandell, "Distributed tracking in sensor networks with limited sensing range," in *Proc. American Control Conf.*, Seattle, WA, June 2008, pp. 3157–3162.

[28] R. Olfati-Saber, "Kalman-consensus filter: Optimality, stability, and performance," in *Proc. 48th IEEE Conf. Decision and Control and the 28th Chinese Control Conf.*, Shanghai, China, Dec. 2009, pp. 7036–7042.

[29] B. Song, A. Kamal, C. Soto, C. Ding, J. Farrell, and A. Roy-Chowdhury, "Tracking and activity recognition through consensus in distribution camera network," *IEEE Trans. Image Processing*, vol. 19, no. 10, pp. 2564–2579, Oct. 2010.

[30] J. Zhao, S. C. Cheung, and T. Nguyen, "Optimal camera network configurations for visual tagging," *IEEE J. Select. Topics Signal Processing (Special Issue on Distributed Processing in Vision Networks)*, vol. 2, no. 4, pp. 464–479, Aug. 2008.

[31] U. M. Erdem and S. Sclaroff, "Automated camera layout to satisfy task-specific and floor plan-specific coverage requirements," *Comput. Vis. Image Understand*., vol. 103, no. 3, pp. 156–169, 2006.

[32] F. Qureshi and D. Terzopoulos, "Planning ahead for PTZ camera assignment and handoff," in *Proc. IEEE/ACM Int. Conf. Distributed Smart Cameras,* Como, Italy, Aug.-Sept. 2009, pp. 1–8.

[33] A. Mittal and L. Davis, "A general method for sensor planning in multi-sensor systems: Extension to random occlusion," *Int. J. Comput. Vis.,* vol. 76, no. 1, pp. 31–52, 2008.

[34] C. Piciarelli, C. Micheloni, and G. Foresti, "PTZ camera network reconfiguration," in *Proc. IEEE/ACM Int. Conf. Distributed Smart Cameras*, Como, Italy, Aug. 2009, pp. 1–8.

[35] C. Soto, B. Song, and A. Roy-Chowdhury, "Distributed multi-target tracking in a self-configuring camera network," in *Proc. IEEE Conf. Computer Vision and Pattern Recogniton*, Miami, FL, June 2009, pp. 1486–1493.

[36] D. Fudenberg and D. K. Levine, *The Theory of Learning in Games* (Series on Economic Learning and Social Evolution). Cambridge, MA: MIT Press, 1998.

[37] D. Monderer and L. Shapley. (1996). "Potential games," *Games Economic Behav.* [Online]. *14(1)*, pp. 124–143. Available: http://www.sciencedirect.com/science/article/B6WFW-45MH08H-2C/2/69a63e2471795b08d937785bb923e9*ea*

[38] B. Song, C. Ding, A. Roy-Chowdhury, and J. Farrell, "Persistent observation of dynamic scenes in an active camera network," in *Distributed Video Sensor Networks,* B. Bhanu, C. V. Ravishankar, A. K. Roy-Chowdhury, H. Aghajan, and D. Terzopoulos, Eds. London: Springer-Verlag, 2011, pp. 259–271. [SP]