Collaborative Sensing in a Distributed PTZ Camera Network

Chong Ding, Bi Song, Akshay Morye, Jay A. Farrell, and Amit K. Roy-Chowdhury

***Abstract*—The performance of dynamic scene algorithms often suffers because of the inability to effectively acquire features on the targets, particularly when they are distributed over a wide field of view. In this paper, we propose an integrated analysis and control framework for a pan, tilt, zoom (PTZ) camera network in order to maximize various scene understanding performance criteria (e.g., tracking accuracy, best shot, and image resolution) through dynamic camera-to-target assignment and efficient feature acquisition. Moreover, we consider the situation where processing is distributed across the network since it is often unrealistic to have all the image data at a central location. In such situations, the cameras, although autonomous, must collaborate among themselves because each camera's PTZ parameter entails constraints on the others. Motivated by recent work in cooperative control of sensor networks, we propose a distributed optimization strategy, which can be modeled as a game involving the cameras and targets. The cameras gain by reducing the error covariance of the tracked targets or through higher resolution feature acquisition, which, however, comes at the risk of losing the dynamic target. Through the optimization of this reward-versus-risk tradeoff, we are able to control the PTZ parameters of the cameras and assign them to targets dynamically. The tracks, upon which the control algorithm is dependent, are obtained through a consensus estimation algorithm whereby cameras can arrive at a consensus on the state of each target through a negotiation strategy. We analyze the performance of this collaborative sensing strategy in active camera networks in a simulation environment, as well as a real-life camera network.**

***Index Terms*—Camera networks, cooperative camera control, distributed estimation, game theory, video analysis.**

I. Introduction

NETWORKS of video cameras are being installed for a variety of applications, e.g., surveillance and security, environmental monitoring, and disaster response. Existing camera networks often consist of fixed cameras covering a large area. This results in situations where targets are often not covered at the desired resolutions or viewpoints, thus making the analysis of the video difficult, particularly when there are specified requirements associated with the targets, such as tracking precision, face shots for identification, and so on. Since the total number of cameras is usually dictated by various factors (e.g., cost and ease of deployment) beyond video acquisition fidelity, a possible solution is to integrate the analysis and sensing tasks more closely. This can be achieved by dynamically controlling the parameters of a pan, tilt, zoom (PTZ) camera network to fulfill the analysis requirements. Dynamic target assignment will allow for maximal utilization of the network, allowing the cameras to differentially focus on multiple regions of interest leading to efficient feature acquisition. Such a setup will provide greater flexibility while requiring less hardware and being less costly. For this to successfully happen, the cameras must work in a collaborative manner to achieve the same overall goal.

In this paper, we focus on the problem of controlling the PTZ parameters of the cameras in a wide-area distributed camera network in order to optimize the performance of various solutions to dynamic scene analysis problems, for example, minimizing the tracking error, getting a shot at the desired resolution and pose, getting a shot of the face for identification, or the combination of the above. In order to achieve this, it is necessary that the camera parameters, which yield the camera-to-target assignments, be selected in an intelligent and collaborative manner. Since the targets are moving, the cameras need to be dynamically controlled so that the imaging requirements can be met.

It is desirable that the control mechanism be distributed for a number of reasons. For example, there may be constraints of bandwidth, secure transmission facilities, and difficulty in installing a camera network with central processing facilities. In such situations, the cameras would have to act as autonomous agents and decisions would have to be taken in a decentralized manner. Because the settings of the cameras jointly determine the value of the resulting imagery, the decentralized camera agents must collaboratively work toward achieving the same optimal result as could be achieved by a centralized approach. Although there are a number of methods in video analysis that deal with multiple cameras, and even camera networks, *distributed* processing in camera networks has received very little attention. On the other hand, distributed processing has been extensively studied in the multiagent systems and cooperative control literature [25]. In some recent work [3], [16], the distributed optimization problem in cooperative control is viewed using the language of learning in games. Game theory is concerned with the interactions between multiple decision-makers, and hence, it is clearly relevant to cooperative control. The effectiveness of utilizing game-theoretic approaches for controlling multiagent systems was demonstrated in [16]. However, there is a very little study on the applicability of these methods in camera networks.

Manuscript received October 12, 2010; revised January 14, 2012; accepted January 30, 2012. Date of publication February 23, 2012; date of current version June 13, 2012. This work was supported in part by the National Science Foundation under Grant 0622176, by the Office of Naval Research under Grant N00014-09-1-0666, and by the Army Research Office under Grant W911NF-07-1-0485. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Mary Comer.

C. Ding, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury are with the University of California at Riverside, Riverside, CA 92521 USA (e-mail: amitrc@ee.ucr.edu).

B. Song was with University of California at Riverside, Riverside, CA 92521 USA. She is now with Sony Research USA, San Jose, CA 95101 USA.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TIP.2012.2188806

In this paper, we leverage the research on distributed optimization built upon game theoretic concepts in the cooperative control literature [14]. However, there are some major differences based on the nature of cameras that will require us to focus specifically on this domain. For example, in the vehicle-to-target (V–T) assignment literature [3], each vehicle is assigned to one target (one-to-one mapping), whereas in the camera network, each camera can observe multiple targets and a target can be observed by multiple cameras (many-to-many mapping). Traditional V–T assignment is a one-time process. Camera tracking for image acquisition is an ongoing tracking and feedback process where the PTZ parameters affect the acquired imagery and that imagery affects the tracking accuracy, which in turn affects the PTZ selection. Moreover, cameras are directional sensors, and thus, geographically neighboring cameras may be viewing very different portions of the scene or have distinct aspect angles on targets. On the other hand, cameras that are geographically far away may be observing the same target. In addition, camera control schemes must take into account the uncertainty in the target tracks arising from inherent challenges in video analysis and the unknown trajectories.

We formulate the multicamera control problem in the setting of a multiplayer game. Specifically, we employ a framework in which the optimization of local sensor utility functions leads to an optimal value for a global utility. We will show how to design the general utility functions, and we provide examples for several representative applications under this general framework. The optimal camera parameters in the sensor network are dynamically determined according to these utility functions and negotiation mechanisms between cameras. To employ suitable negotiation mechanisms between the different sensors is of great importance since the cameras have to take strategic decisions according to the perceived or anticipated actions of the other cameras. All this requires tracks that are obtained through a distributed tracking algorithm using the Kalman-Consensus filter [21], [27].

The remainder of this paper is organized as follows. Section II presents a rationale for the need of a decentralized collaborative camera network. Section III states the problem with its solution in game theoretic terms and we study several representative application scenarios. Extensive experimental results are presented in Section IV. Finally, we summarize this paper in Section V.

II. Technical Rationale

A. Necessity of Collaboration in a PTZ Camera Network

We start by motivating the necessity of a cooperative strategy in an intelligent camera network. The two questions we address are the following: 1) why do we need active cameras (as opposed to having a network of cameras with a fixed set of parameters) and 2) why does the control strategy need to be cooperative?

The main reason for having a dynamically self-configurable network is that it would be prohibitively expensive to have a static setup that would cater to all possible situations. For example, suppose we need: 1) to track one person (possibly noncooperative) as he walks around an airport terminal; 2) to obtain a high-resolution image of him or his specific feature (e.g., face); and 3) to observe other activities going on in the terminal. To achieve this, we will either need to dynamically change the parameters of the cameras where this person is visible or have a setup whereby it would be possible to capture high-resolution imagery irrespective of where the person is in the terminal. The second option would be very expensive and a huge waste of resources, both technical and economic. Therefore, we need a way to control the cameras based on the sensed data.

The control strategy must be necessarily cooperative because each camera's parameter setting entails certain constraints on other cameras. For example, if a camera zooms in to focus on the face of one particular person, thus narrowing its field of view (FOV), it *risks* losing much of that person and the surroundings, therefore being less useful for target tracking. Another camera can compensate for this by adjusting its parameters. This requires analysis of the video data in a *collaborative network-centric manner*, leading to a cost-effective method to obtain high-resolution images for features at dynamically changing locations.

B. Necessity of a Decentralized Strategy

As the problem complexity increases, it may be difficult to analyze all the data in a centralized manner and come up with an effective strategy for persistently observing the dynamic scene. There may not be enough bandwidth and transmission power available to send all the data to a central station. Furthermore, security of the transmission and interception by a hostile opponent may be a factor in some applications. A centralized scheme may be challenging in many field operations as it is often difficult to setup the required infrastructure in the environments where they are deployed (e.g., monitoring of remote regions and hostile areas). In the proposed framework, each camera must take its own decisions based on the analysis of its own sensed data and negotiation mechanisms with other sensors.

C. Related Work

Our review of scene analysis algorithms will be limited to those directly related to the application domain of camera networks.

There have been a few papers in the recent past that deal with networks of video sensors. In [32], the authors presented preliminary work in camera networks using independent cameras (i.e., no coordination between the cameras). Particular interest has been focused on learning a network topology [17], [34], i.e., configuring connections between cameras and entry/exit points in their view. Some of the existing methods on tracking over the network include [12], [23], and [28]. Other interesting problems in camera networks, such as object/behavior detection and matching across cameras, camera handoff, and camera network configuration, have been addressed in [1], [9], [11], [15], [31], and [40]. In [8], 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. There has been recent work also on tracking people in a multicamera setup [7], [13]. However, these methods did not address the issue of distributed processing.

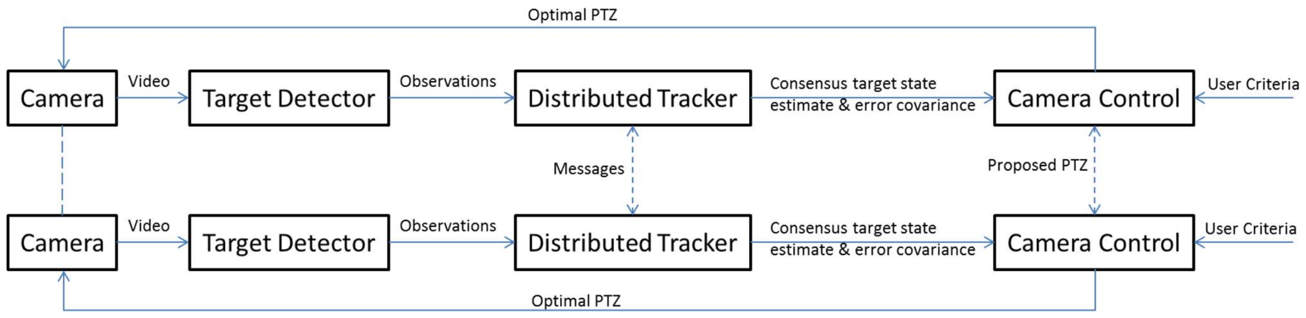


Fig. 1. Overall system diagram depicting our proposed framework for collaborative sensing in an active distributive camera network. The user criteria can define what performance metrics the network will optimize. The user criteria could include covering the entire area at a desired resolution, obtaining facial shots, maximizing image resolution, and so on. If the user does not specify any additional criteria, the tracking error will be minimized.

Recently, some problems in video analysis have been addressed in a distributed manner. A distributed algorithm for the calibration of a camera network was presented in [6]. Object detection, tracking, recognition, and pose estimation in a distributed camera network were considered in [24] and [38]. Distributed estimation for various computer vision problems was studied in [33], [35], and [36]. These methods were designed for a network composed of static cameras; the dynamics of active camera networks are not taken into account.

An overview of some main video processing techniques and current trends for video analysis in PTZ camera networks can be found in [19] and [26]. Cooperation in a network consisting of static and PT cameras was considered in [18]. The parameters of PT cameras were determined by centering the desired targets in the FOV, and cooperation is only between the static camera set and the PT camera set, i.e., the static cameras check if a PT camera is tracking a correct target (unlike our case where all PTZ cameras in the network cooperate). A related work that deals with control of a camera network with PTZ cameras is [22]. Here, a virtual camera network environment was used to demonstrate a camera control scheme that is a mixture between a distributed and a centralized scheme using both passive and active PTZ cameras. Their work focused on how to group cameras that are relevant to the same task in a centralized manner while maintaining the individual groups decentralized. In our method, we consider a completely distributed solution using a game-theoretic framework for camera parameter control and implicit target assignment. A game-theoretic framework for V–T assignment was proposed in [3] in the context of distributed sensor networks. However, in that work, the targets were stationary and each vehicle was assigned to one target. That work did not consider the constraints imposed by video cameras. A camera handoff approach using game theory was presented in [15]. That method, however, considers only a set of static cameras and does not deal with the problem of persistently observing targets with varying resolutions over a large geographic area using a dynamic camera network with overlapping and nonoverlapping FOVs.

A game-theoretic approach to camera control was presented in [29] but limited only to the area coverage problem. This was expanded on in [30] to a distributed tracking and control approach. It required the camera control and tracking to independently run and in parallel. The camera control used game theory to assign camera settings that provided coverage over regions of interest while maintaining a high-resolution shot of a target. Concurrently, a Kalman-Consensus filter provided tracks of each target on the ground plane. In this paper, we propose a distributed strategy for controlling the parameters of the cameras that is integrated with a distributed tracking algorithm. The camera control algorithm is designed to maximize different performance criteria for scene analysis (e.g., minimize tracking error, obtain a facial shot, and maximize image resolution), which was not considered in [30]. The method presented in [30] can be shown as a special case of the framework proposed herein.

The research presented in this paper is also related to active vision [2], [4], [5]. However, active vision in a camera network is a relatively unexplored area that would involve cooperation and coordination between many cameras.

D. *Overview of Solution Strategy*

Each of the cameras in our network has its own embedded target detection module, a distributed tracker that provides an estimate on the state of each target in the scene [21], [27] and finally a distributed camera control mechanism. Given the target detection module, this paper provides a distributed solution for the tracking and control problems. Targets are tracked using measurements from multiple cameras that may not have direct communication between each other. Neighboring cameras communicate with each other to come to a consensus about the estimated position of each target. Similarly, camera parameters are assigned based on information exchanged with neighboring cameras within a game-theoretic framework. Our proposed system structure is shown in Fig. 1.

The target detection module takes the image plane measurements and returns the image plane positions associated with specific targets. Through calibration, we know the transformation between image and ground plane coordinates, which can be used for data association. 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 neighboring cameras at each imagery time instant to come to a consensus regarding the estimated state and estimated error covariance [21] of each target. This results in each camera having a consensus estimate of the state and error covariance of each target in the region of interest. In our implementation, an Extended Kalman-Consensus filter is developed for distributed tracking. Since the Kalman-Consensus filter has been presented in [27], we only provide an outline of a particular implementation of its extension, i.e., the Extended Kalman-Consensus filter in Appendix A.

The camera control module attempts to optimize the scene analysis performance at the next imaging time instant by selecting the camera parameters expected to result in measurement optimizing 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. These are represented in a reward-versus-risk tradeoff function. We propose a distributed optimization solution strategy to this problem, which is described in Section III. Following recent trends in cooperative control work, this can be represented in a game-theoretic framework.

III. Distributed Camera Network Control Framework

Our goal is to develop a distributed strategy for coordinated control that relies on local decision-making at the camera nodes while being aligned with the suitable global criteria for scene analysis that are specified by the user. For this purpose, we propose a distributed optimization framework, which can be represented in terms of a cooperative *game* that relies on *multiplayer learning and negotiation mechanisms* [10], [14]. The result is 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. Our contribution in this paper is to show how the distributed optimization framework can be developed for the camera network control problem, including design of specific utility functions, optimization strategies, and performance analysis.

A. Problem Statement and Notation

Consider a region R over which there are $\mathcal{T} = \{T_1, \ldots, T_{N_T}\}$ targets independently moving while being observed by a network of cameras $\mathcal{C} = \{C_1, \ldots, C_{N_C}\}$. The behavior of these camera networks will be determined by a set of criteria specified by the user. Each target T_j, for $j \in \{1, \ldots, N_T\}$, may have different criteria that must be satisfied by the camera network, e.g., we may want to identify T_1 using facial recognition and determine what T_2 is doing using action recognition. In another scenario, we may be interested in observing the scene of a disaster zone to discover and monitor the state of people. Here, we would likely place priority on those who are wounded or in need of aid. This scenario would require targets in R to have varying values of importance and varying requirements over time. The question is whether we can create a generalized framework for distributed camera networks in which many of the scenarios may be modeled and solved.

In our formulation, the state of the jth target at time step k is defined as $\mathbf{x}^j(k) = (x^j(k), y^j(k), \dot{x}^j(k), \dot{y}^j(k))^T$, where $(x^j(k), y^j(k))$ and $(\dot{x}^j(k), \dot{y}^j(k))$ are the position and velocity of target T_j on the ground plane, respectively. By considering a linear discrete-time dynamical model

$$\mathbf{x}^j(k+1) = \mathbf{A}^j(k)\mathbf{x}^j(k) + \mathbf{B}^j(k)\mathbf{w}^j(k) \tag{1}$$

and a nonlinear observation model for each camera C_i

$$\mathbf{z}_i^j(k) = h_i\left(\mathbf{x}^j(k)\right) + \mathbf{v}_i^j(k) \tag{2}$$

the consensus state estimate $\bar{\mathbf{x}}_i^j$ and the error covariance matrices $\mathbf{P}_i^j$ computed by the Extended Kalman-Consensus Filter (see Appendix A and [27]) are the inputs to the control module. Since the Kalman-Consensus tracking approach has been presented in detail in [21] and [27], we only include a summary of it in the Appendix for the sake of completeness, with focus on its extension to deal with a nonlinear observation equation.

For camera control, we will design a strategy such that each camera is a rational decision-maker, optimizing its own utility function that indirectly translates to the optimization of a global utility function. Camera $C_i \in \mathcal{C}$ will select its own set of parameters $\mathbf{a}_i \in \mathcal{A}_i$, where $\mathcal{A}_i$ is the parameter profile that C_i can select from, to optimize its own utility function $U_{C_i}(\mathbf{a}_i, \mathbf{a}_{-i})$. Our problem is to design these utility functions and appropriate negotiation procedures that lead to mutually agreeable parameter settings of the cameras meeting the global criterion.

The game-theoretic interpretation of the distributed optimization in the cooperative control work of [3] and [16] allows for performance analysis in terms of game theoretic results. 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 sets of parameters $\mathbf{a}^* = (\mathbf{a}_1^*, \ldots, \mathbf{a}_{N_C}^*)$ such that no sensor could improve its utility further by deviating from $\mathbf{a}^*$. Obviously, $\mathbf{a}^*$ is a function of time since the targets are dynamic and the cameras could be also mobile and capable of panning, tilting, and zooming. For our problem, a Nash equilibrium will be reached, at each instant of time, when there is no advantage for a particular camera to choose some other parameter set.

Mathematically, if $\mathbf{a}_{-i}$ denotes the collection of parameter settings for all cameras *except* camera C_i, then $\mathbf{a}^*$ is a *pure Nash equilibrium* if

$$U_{C_i}\left(\mathbf{a}_i^*, \mathbf{a}_{-i}^*\right) = \max_{\mathbf{a}_i \in \mathcal{A}_i} U_{C_i}\left(\mathbf{a}_i, \mathbf{a}_{-i}^*\right), \ \forall C_i \in \mathcal{C}. \tag{3}$$

B. Choice of Utility Functions

In our framework, the set of criteria that the camera network must satisfy is modeled using global utility function $U_G(\mathbf{a})$. In almost every task, the specified criterion is directly associated with the targets, e.g., getting shots at a desired resolution and pose. We can assign a target utility $U_{T_j}(\mathbf{a})$ that quantifies the satisfaction of the criterion, given parameter settings $\mathbf{a}$, for a target. Maximization of the utility functions across all targets, constrained by the parameter sets of the camera network, results in the set of parameters that best satisfies the criteria at each point in time.

1) Target Utility: Assuming there are L criteria that the camera network needs to meet, the satisfaction of those criteria on a target T_j is measured by a set of metrics $\{M_1(\mathbf{a}, T_j), \ldots, M_L(\mathbf{a}, T_j)\}$, which are functions of parameter settings for all cameras. The target utility describes the set of criteria for target T_j, which can be represented as a function of the set of metrics on T_j, i.e.,

$$U_{T_j}(\mathbf{a}) = \mathcal{F}\left(M_1(\mathbf{a}, T_j), \ldots, M_L(\mathbf{a}, T_j)\right). \tag{4}$$

There could be many choices of $\mathcal{F}(\cdot)$, one possible form of $\mathcal{F}$ being weighted summation, i.e.,

$$U_{T_j}(\mathbf{a}) = w_1 \cdot M_1(\mathbf{a}, T_j) + \cdots + w_L \cdot M_L(\mathbf{a}, T_j). \quad (5)$$

The weights preceding each metric define the relevance/importance of each criterion and can be dynamically modified to allow for fine grain control of the behavior of the camera network toward each target T_j. The higher the weight, the greater the emphasis of the corresponding criterion. A sample set of possible metrics for different criteria is listed below.

Tracking Criterion: The purpose of the tracking utility is to quantify how well the tracking module of camera C_i is tracking target T_j given the settings $\mathbf{a}_i$. Assuming an Extended Kalman-Consensus tracker, we can define this utility using the error covariance matrices $\mathbf{P}_i^j$ computed by the Extended Kalman-Consensus Filter (see Appendix A and [27]) and the measurement Jacobian matrix $\mathbf{H}_i^j$, i.e.,

$$M_{Tr_j}(\mathbf{a}) = \exp\left\{-\text{Trace}\left(\mathbf{P}_i^{j+}\right)\right\} \quad (6)$$

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

$$\left(\mathbf{P}_i^{j+}\right)^{-1} = \left(\mathbf{P}_i^{j}\right)^{-1} + \sum_{l \in \left(C_i \cup \mathcal{C}_i^n\right)} \left(\mathbf{H}_l^j(\mathbf{a})\right)^T \mathbf{R}_l^j(\mathbf{a})^{-1} \mathbf{H}_l^j(\mathbf{a}) \quad (7)$$

where $\mathbf{R}$ is the measurement error covariance using $\mathbf{a}$ and $\mathcal{C}_i^n$ is the neighborhood of C_i, i.e., the cameras that can directly communicate with C_i. Note that $\mathbf{P}_i^{j+}$ is the information we expect to get from the next image set given $\mathbf{H}_l^j(\mathbf{a})$ and $\mathbf{R}_l^j(\mathbf{a})$, which are computed based on $\mathbf{a}$ and $\hat{\mathbf{x}}_i^j$; hence, the right-hand side of (6) is a function of $\mathbf{a}$.

Using this estimated error covariance update equation, we can predict covariance $\mathbf{P}_i^{j+}$ given some settings profile $\mathbf{a}$. By choosing a settings profile that maximizes the tracking utility, we are selecting the set of measurements that will minimize the estimated error covariance within the Extended Kalman-Consensus tracker.

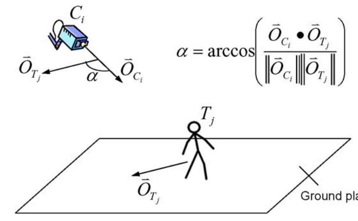
View Criterion: The view utility $M_{V_j}(\mathbf{a})$ quantifies the extent to which the resolution and/or pose requirement of target T_j is satisfied by the camera network using settings profile $\mathbf{a}$. Let p_{ij} be the probability that target T_j is viewed at the desired resolution by camera C_i. Then, $M_{V_j}(\mathbf{a})$ is defined here for the resolution requirement as

$$M_{V_j}(\mathbf{a}) = 1 - \prod_i (1 - p_{ij}) \quad (8)$$

where

$$p_{ij} = \begin{cases} 1 - e^{-\lambda \frac{r_i^j}{r_{\max}}}, & \text{if } r_{\min} \le r_i^j \le r_{\max} \\ 0, & \text{if otherwise.} \end{cases} \quad (9)$$

$r_{\min}$ and $r_{\max}$ are the minimum and maximum height requirements, respectively, of target T_j in the camera image plane and r_i^j is the resolution at which T_j is being viewed at by C_i. 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.



C_i
$\vec{O}_{T_j}$
α
$\vec{O}_{C_i}$
$\alpha = \arccos\left(\frac{\vec{O}_{C_i} \bullet \vec{O}_{T_j}}{\|\vec{O}_{C_i}\|\|\vec{O}_{T_j}\|}\right)$
T_j
$\vec{O}_{T_j}$
Ground plane

Fig. 2. Illustration of the view angle factor defined in (10).

This utility could be also 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

$$\gamma(\mathbf{a}, \bar{\mathbf{x}}_j) = \cos\left(\theta_{T_j} - \arccos\left(\frac{\vec{O}_{T_j} \cdot \vec{O}_{C_i}}{\|\vec{O}_{T_j}\|\|\vec{O}_{C_i}\|}\right)\right) \quad (10)$$

where θ_{T_j} is the desired view angle of T_j, $\vec{O}_{T_j}$ is the orientation of T_j, and $\vec{O}_{C_i}$ is the orientation of camera C_i. An illustration of the view angle factor is shown in Fig. 2. Thus, the view utility becomes

$$M_{V_j}(\mathbf{a}) = \gamma(\mathbf{a}, \bar{\mathbf{x}}_j)\left[1 - \prod_i (1 - p_{ij})\right]. \quad (11)$$

Risk of Losing Target: Risk $M_{R_j}(\mathbf{a})$ determines the probability that the current settings profile $\mathbf{a}$ will not cover target T_j at its probable locations. This is dependent on consensus position estimate $\bar{\mathbf{p}}_i^j$, a subvector of consensus state estimate $\bar{\mathbf{x}}_i^j$, PTZ settings $\mathbf{a}$, and covariance $\mathbf{P}_i^j$. It is defined as

$$M_{R_j}(\mathbf{a}) = 1 - \iint_D \frac{1}{2\pi |\mathbf{E}_i^j|^{\frac{1}{2}}} \cdot \exp\left(-\frac{1}{2}\left(\mathbf{b} - \bar{\mathbf{p}}_i^j\right)^T \mathbf{E}_i^j \left(\mathbf{b} - \bar{\mathbf{p}}_i^j\right)\right) d\mathbf{b} \quad (12)$$

where $\mathbf{E}_i^j$ is the submatrix of $\mathbf{P}_i^j$ containing the covariance of the position and D is the area covered by cameras.

2) Global Utility: Global utility U_G describes the desirability of the settings profile $\mathbf{a}$, given the criteria that must be satisfied by the camera network. This is represented as a function of the importance/priority of each target T_j and its related target utility U_T

$$U_G(\mathbf{a}) = v_1 \cdot U_{T_1}(\mathbf{a}) + \cdots + v_{N_T} \cdot U_{T_{N_T}}(\mathbf{a}) \quad (13)$$

where v_j denotes the importance of target T_j.

By maximizing the global utility, we are choosing the settings profile that best satisfies the criteria specified for the camera network.

3) Camera Utility: The global utilities must be converted now into local utilities in order for them to be solved in a distributed fashion. Convergence proofs in game theory [20] require that the local utilities are aligned with the global utility, i.e., a change in the local utility affects the global utility similarly. We achieve this by making the utility of our camera equivalent to its contribution to the global utility, i.e.,

$$U_{C_i}(\mathbf{a}) = U_G(\mathbf{a}) - U_G(\mathbf{a}_{-i}) \quad (14)$$

where set $\mathbf{a}_{-i}$ is the set of settings profiles excluding the profile for camera i. It can be shown that this leads to a potential game using the global utility as the potential function [20], [37] (the definition of potential game and its convergence are stated in Appendix B). This allows us to maximize the global utility through the maximization of the utility of each camera.

C. Negotiation Mechanisms

The dynamic nature of the targets in the region being observed requires that our cameras communicate with each other in order to decide the set of parameters that will result in the optimal global utility. Each camera negotiates with its neighboring cameras to accurately predict the actions of the other cameras before deciding its own action. The overall idea of the proposed negotiation strategy is to use *learning algorithms for multiplayer games* [20]. A particularly appealing strategy for this problem is spatial adaptive play (SAP) [39]. This is because it can be implemented with a low computational burden on each camera and leads to an optimal assignment of targets with arbitrarily high probabilities for the camera utility described above. Iteration stops if a Nash equilibrium is attained or if the available operation time expires.

Application of SAP Negotiation Mechanism: At any step of SAP negotiations, a camera C_i is randomly chosen from the pool of cameras in the network according to a uniform distribution over the cameras, and only this camera is given the chance to update its proposed parameter settings. At negotiation step k, C_i proposes a parameter setting according to the following probability distribution based on other cameras' parameters at the previous step:

$$p_i(k) = \sigma\left(\frac{1}{\tau}\begin{bmatrix} U_{C_i}\left(A_i^1, \mathbf{a}_{-i}(k-1)\right) \\ \vdots \\ U_{C_i}\left(A_i^{|\mathcal{A}_i|}, \mathbf{a}_{-i}(k-1)\right)\end{bmatrix}\right) \tag{15}$$

for some $\tau > 0$, where $a(k-1)$ denotes the profile of the proposed parameter settings at step $k-1$, $\mathcal{A}_i = \{A_i^1, \ldots, A_i^{|\mathcal{A}_i|}\}$ is the enumeration of all possible parameter settings of camera C_i, and $|\mathcal{A}_i|$ is the number of elements in $\mathcal{A}_i$. $\sigma(.)$ is the logistic or soft-max function, and its mth element is defined as

$$(\sigma(x))_m = \frac{e^{x_m}}{e^{x_1} + \cdots + e^{x_n}}.$$

Constant τ determines how likely C_i is to select a parameter setting. If $\tau \to \infty$, C_i will select any setting $a_i \in \mathcal{A}_i$ with equal probability. As $\tau \to 0$, C_i will select a setting from its best response set

$$\left\{a_i \in \mathcal{A}_i : U_{C_i}\left(a_i, \mathbf{a}_{-i}(t-1)\right) = \max_{a_i' \in \mathcal{A}_i} U_{C_i}\left(a_i', \mathbf{a}_{-i}(t-1)\right)\right\}$$

with arbitrarily high probability. After C_i updates its settings, it broadcasts its parameters (i.e., PTZ) to all neighboring cameras. The other cameras can then use that information to update their parameters after being chosen at any negotiation step until a consensus is reached. This occurs when the cameras have reached a Nash equilibrium, i.e., when no camera can increase the global utility by changing its parameters.

The utility functions combined with the negotiation strategy 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 as defined by the global utility. In practice, we will not need to do this at every time step. Optimization can be done whenever the utility falls below a certain threshold.

D. Example Scenarios

Consider a network of N_C PTZ smart cameras, each of which contains a person detector, a Kalman-Consensus tracker, and some high-level processing tasks (as needed by the application). In this section, we will study several specific cases of achieving different system goals for scene analysis. We will discuss in detail the design of the utility functions.

1) Example 1—Cover Entire Area With at Least Some Minimum Resolution While Focusing on Some Targets at Higher Resolution: To cover the entire area, the area can be divided into grids in a way similar to [8] to make the problem more tractable. Then, each grid of the area is treated as a virtual target.

Since the goal is covering the entire area at a required resolution, there is no risk of losing a target if the goal is achieved. Thus, only the view utility needs to be considered in this case. For each virtual target T_j, the utility is

$$U_{T_j^v}(\mathbf{a}) = M_{V_j}(\mathbf{a}) \quad \text{with } r_{\min} = r_v, \quad \text{for } j = 1, \ldots, N_g \tag{16}$$

where N_g is the total number of grid cells, M_{V_j} is defined in (8), $r_{\min}$ is the minimum resolution requirement as defined in (9), and r_v is the minimum requirement for covering the entire area.

For some user-specified real targets that need to be viewed with higher resolution, although a tracking module is run on them to provide the estimates of their positions, we do not factor the tracking performance into the optimization function in the control module because the entire area is required to be covered. We refer to this case as the ***Area Coverage*** problem. The utilities for those specified real targets are

$$U_{T_j^h}(\mathbf{a}) = M_{V_j}(\mathbf{a}) \quad \text{with } r_{\min} = r_h, \quad \text{for } j = N_g+1, \ldots, N_g+N_T \tag{17}$$

where r_h is the resolution requirement for those specified targets.

Now we can define the global utility as

$$U_G(\mathbf{a}) = v^h \sum_{j=N_g+1}^{N_g+N_T} U_{T_j^h}(\mathbf{a}) + v^v \sum_{j=1}^{N_g} U_{T_j^v}(\mathbf{a}) \tag{18}$$

where N_T is the number of user-specified targets and the importance of those high-resolution targets v^h should be set to be higher than that of virtual targets v^v.

2) Example 2—Cover Entrances of the Area While Optimizing Tracking of Targets Within the Area: The purpose of the system here is to minimize the tracking error of all targets within the region under surveillance and obtain the best possible shots (in terms of resolution) for each target. The choice of camera parameters determines the expected assignment of targets to cameras, as well as the expected tracking accuracy and risk. Given that the targets are moving, when the entire area is not covered, there is a tradeoff between resolution gain and tracking risk. Each camera may view only a portion of the area under surveillance. Here, we also consider the issue that the importance of every location within the observed space may not be equal. Monitoring entrances and exits to the area under consideration will allow us to identify and subsequently track all targets, whereas monitoring empty space is of little use.

To achieve the system goal, the camera control is aware of the state of the Kalman-Consensus filter and actively seeks to provide it with the most informative measurements for state estimation, which makes the architecture of the system in this case different from the Area Coverage one in Example 1. Furthermore, the estimated error covariance is considered in addition to the estimated state of each target. This allows the optimization to gauge the risk of failing to capture a feature when attempting high-resolution shots. We refer to this case as the ***Target Coverage*** problem.

From the tracking, view, and risk metrics defined in (6), (8), and (12), we can define the target utility as the metrics generated by viewing target T_j, i.e., the metric generated by tracking T_j, minus the risk, or

$$U_{T_j}(\mathbf{a}) = w_1^j M_{Tr_j}(\mathbf{a}) + w_2^j M_{V_j}(\mathbf{a}) - w_3^j M_{R_j}(\mathbf{a}) \qquad (19)$$

where w_1^j, w_2^j, and w_3^j are the weights corresponding to the tracking, view, and risk metrics for target T_j, respectively. We can then determine those settings profiles that maximize the global utility based on the analysis of $U_G(\mathbf{a})$.

Setting a high value for w_1^j and w_2^j and a low value for w_3^j causes the system to prioritize tracking gain with high resolution at the risk of losing the target. With such a setting and a poor estimate of the target state, it is possible that the chosen settings profile fails to capture a high-resolution image of some target and may not image the target. Recovery from such an event is discussed below. By setting a higher value for w_2^j (compared with w_1^j and w_3^j), the camera network will attempt to acquire a high-resolution shot of the target j. This has the added effect of reducing overlap within the FOVs of the network. Conversely, if w_3^j for all targets T_j is set high, the system would choose the view containing all targets since this system setting minimizes the risk that targets are uncovered at any time instant.

Fault tolerance is handled through the interaction of the tracking utility and risk. The estimated error covariance of a target grows the longer a target is not observed. The larger the estimated error covariance, the larger the area that needs to be covered to decrease the risk. Thus, the system will prefer a setting whereby a larger area will be covered so that the lost target can be imaged, although this camera will be at low resolution. We show this through a specific example in Fig. 9.

3) Example 3—Cover Entire Area With at Least Some Minimum Resolution While Optimizing Tracking of Targets Within the Area: This case is a combination of the Area Coverage and Target Coverage problems. From Example 1 and Example 2, we can easily define the global utility as

$$U_G(\mathbf{a}) = \sum_{j=N_g+1}^{N_g+N_T} v_j^h \left(w_1^k M_{Tr_j}(\mathbf{a}) + w_2^k M_{V_j}(\mathbf{a})\right) + v^v \sum_{j=1}^{N_g} M_{V_j}(\mathbf{a}). \qquad (20)$$

The risk is not included in this because the entire area is covered, and hence, there is no risk of losing a target.

IV. Experimental Results

To evaluate the proposed camera control algorithm, we have performed extensive experiments and analysis. First, we test our approach on a simulated network of cameras. Next, we show results on a real-life camera network by considering two example performance criteria for scene analysis, i.e., Area Coverage and Target Coverage (see Section III-D). Finally, we quantitatively analyze the performance of the proposed method with changing numbers of targets.

A. Simulation Data

The goal of the simulation was to test the algorithm under a variety of conditions that are not possible with real data due to the physical limitations of the experimental setup. In addition, we can compare performance with ground truth. Our simulation setup was very close to the real-life experimental framework. For this simulation, we consider the special case of monitoring the entire area while focusing on a few targets at high resolution.

The area under surveillance is set up as a rectangular region of 20 m × 30 m. In our specific implementation presented here, we divide the area into grids in a way similar to [8] to make the problem more tractable. Thereafter, we treat each grid of the area as a virtual target (since we are dealing with the problem of covering the entire area). In this simulation, the grids are of size 1 m × 1 m. The sensor network consists of eight PTZ cameras with a resolution of 320 × 240 pixels spread around the perimeter of the area, as shown in Fig. 3(a). Initially, the camera parameters (i.e., PTZ) are arbitrarily assigned so that the area under surveillance is not entirely covered by the cameras' FOVs.

Initialization: The first goal is to determine an initial set of camera parameters to cover the entire area at an acceptable resolution. To achieve this, we use the game-theoretic approach described above (see Section III-D-1). Each grid in the area is treated as a virtual target. At each negotiation step, the cameras can update their parameter settings, given the settings of the other cameras at the previous time step, to affect which of these virtual targets they are observing. At each negotiation step, after a camera updates its PTZ parameters, these parameters are transmitted to the other cameras in the network. The other cameras can therefore calculate the new area that this camera is now covering. This process is repeated at each negotiation step in which a camera is chosen in turn to update its parameters. In this simulated setting, the negotiation converges to a completely covered area at an acceptable resolution after typically 17 negotiation steps. The result after initialization is shown in Fig. 3(a). Areas in different shades of gray are covered. Any white area is not covered. The shading gets darker for a block as more cameras cover it. Successful convergence is shown, as there is no white area in Fig. 3(a).

Zooming in one target: After the entire region is covered at an acceptable resolution, it follows that every moving targets in that area is also being viewed at an acceptable resolution given the constraint of covering the whole area. However, a human operator could chose to observe a feature (e.g., face) of a specific moving target at a higher resolution for a specific application

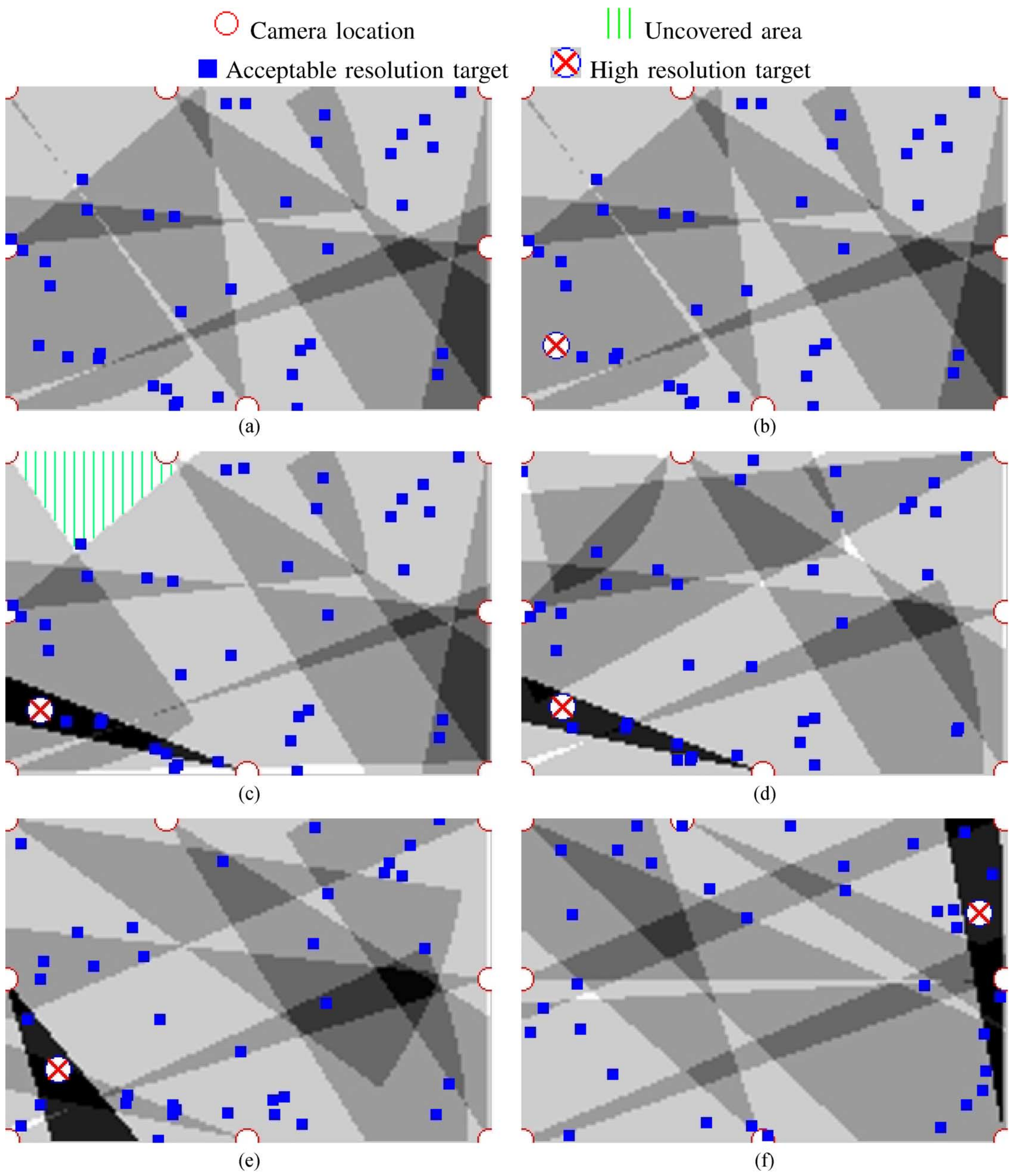


Fig. 3. Simulated area under surveillance with several targets visible. White indicates uncovered area. Shades of gray indicate coverage. (a) Result after initial convergence shows the area of interest being completely covered by the cameras at an acceptable resolution r_v; therefore, all the targets in the area are being viewed at least r_v. (b) Initial condition when a target (T_j^h, r_h) has been selected for observation at a high resolution r_h. In (c), C_h, the camera with the dark FOV, decided that it could observe (T_j^h, r_h) and adjusted its FOV accordingly. Note now that there is a part of the area uncovered due to C_h's change of parameters. As shown in (d), some of the other cameras readjusted their parameters through the negotiation mechanisms to maximize their utilities covering again the entire area under surveillance. At a later time, in (e), when C_h is not able to keep (T_j^h, r_h) in its FOV anymore, based on the knowledge of T_j^h, the other cameras adjust their parameters to maintain both area coverage and (T_k^h, r_h). The entire process above is repeated until (T_j^h, r_h) exits the area, as shown in (f).

(e.g., face recognition). The zoom factor to view the targets at the desired resolution is determined by the number of pixels a target (assumed here to have a height of 170 cm on the ground plane) occupies on the image plane. This chosen target is the marked target in Fig. 3(b). A human operator is simulated by tasking the camera in the global configuration that yields the highest resolution of the target with obtaining high-resolution images and removing it from the area coverage optimization. The results of the change can be seen in the dark FOV in Fig. 3(c).

Since the parameter change in camera C_h may leave parts of the area uncovered, the other cameras must change their parameters during the negotiation to again cover the whole area at an acceptable resolution. This parameter change of the cameras is evident in Fig. 3(d). As the target moves out of the range of the camera, a different camera must be chosen to continue the high-resolution image capture. In this simulation, C_h predicts the position of the target at time t. If C_h determines that it will not be able to observe the high-resolution target after a time instance t_{out}, it will handoff the task to the camera with the best resolution and rejoin the pool of cameras optimizing area coverage. This can be seen in Fig. 3(d), where the high-resolution target is about to exit the camera's FOV, and in Fig. 3(e), where

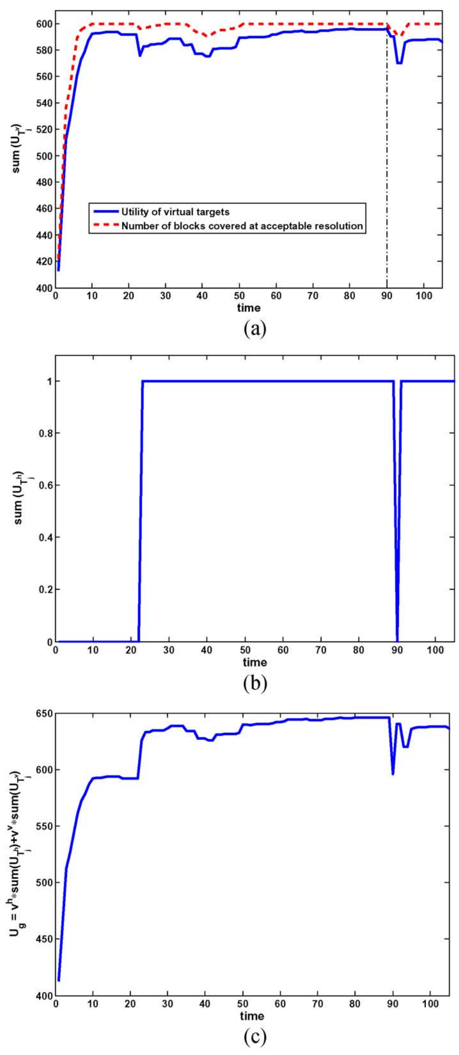


Fig. 4. Typical plots of utilities for the simulation in Section IV-A. (a) Coverage of the entire area. The solid line represents the summation of utilities for the virtual targets, i.e., $\sum_{j=1}^{N_g} U_{T_j^v}$, and the dashed line represents the number of area blocks being covered by the camera network at an acceptable resolution. The maximum number of block coverage is 600, i.e., the entire area is being covered. The utility for the high-resolution target is shown in (b). The utility of the high-resolution target is 0 for a very short time period around 90th second as there is a handoff between cameras; hence, no camera views the target at high resolution in this period, but the target is still viewed at an acceptable resolution (can be inferred from (a) as entire area is being covered in that period). At all other times, the high resolution is maintained on the real target. In (c), global utility is plotted; here, the importance values of real and virtual targets are set to be 50 and 1, respectively.

another camera that was participating in the negotiation decided that it could view the target at the highest resolution at t_{out} and took over observation of the target. Since a new camera has taken charge of observing the high-resolution target, the other cameras adjust their parameters in order to cover the entire area at an acceptable resolution. This process is repeated until the target exits the area of interest, as shown in Fig. 3(f). Once the target has left the area under surveillance completely, all the cameras continue with the negotiation, maximizing their utilities to keep the entire area covered at an acceptable resolution.

Fig. 4 shows typical plots of utilities versus time for this simulation. Utilities for real and virtual targets are shown in Fig. 4(a) and (b). Fig. 4(c) shows the global utility. As shown, when a camera starts observing a target at a high resolution, some area is left uncovered and the utility for virtual targets decreases. Note that the real targets are randomly simulated; actual change of utilities depends on the real scenario and should not be the same for different scenarios.

B. Real-Life Data

1) Example 1—Cover Entire Area While Observing Some Targets at Higher Resolution: The sensor network consists of nine PTZ network IP cameras with a resolution of 320 × 240 pixels and 12× optical zoom. In this experiment, for each camera, the pan and tilt parameters are roughly separated by 15° with 2 settings for zoom. We use this setting to show the performance of our approach with an increasing number of cameras removed from the area coverage optimization.

Fig. 5 demonstrates the ability of the game-theoretic approach to self-organize to maintain coverage while allowing some cameras to zoom in for high-resolution images. Fig. 5(a) shows the initial convergence result that covers the entire area at an acceptable resolution. The coverage is shown on the bottom-right (blue areas are covered, white areas are not; the darker the color of a block, the greater the number of cameras that is assigned there). Fig. 5(b) shows that, when one camera zooms in (camera C_8, bounded with red lines), the other cameras automatically adjust their settings (camera C_5 zooms out) to keep the area covered. Fig. 5(c)–(e) shows the effect of increasing the number of cameras zooming in from 2 to 4. It is shown that, as the number of zoomed-in cameras increases, more areas are left uncovered. Fig. 5(f) shows the reinitialization result when we reset the cameras with none of them zooming in, i.e., the network of cameras can again keep the entire area covered. By comparing Fig. 5(f) with (a), it can be noticed that the parameter settings in (f) are different with those in (a), although both of them could satisfy the coverage requirements. This illustrates that the Nash equilibrium is not unique.

2) Example 2—Cover Entrances of the Area While Optimizing Tracks of Targets Within the Area: First, we show results on a real-life camera network and then compare with the design for Area Coverage of Example 1. For the following experiments, the tracking and view utilities were weighted high with risk weighted low.

Our camera network is composed of five PTZ IP cameras surrounding a 600-m^2 courtyard. For this experiment, the number of possible PTZ settings for each camera was increased and quantized such that each camera had 4 zoom settings, with about 5° separation between pan and tilt settings at the highest zoom factor and about 15° at the lowest. The total number of configurations is determined by the PTZ binning procedure. The PTZ binning procedure then determines the required processing power and total region viewable at high resolution. This region is further reduced by obstacles. Tracked targets were assumed to have a height of 1.8 m. Each camera acquires images of resolution 640 × 480 pixels. Thus, r_i^j is the largest number of pixels in the vertical direction occupied by T_j in the image plane of some cameras in the network. 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

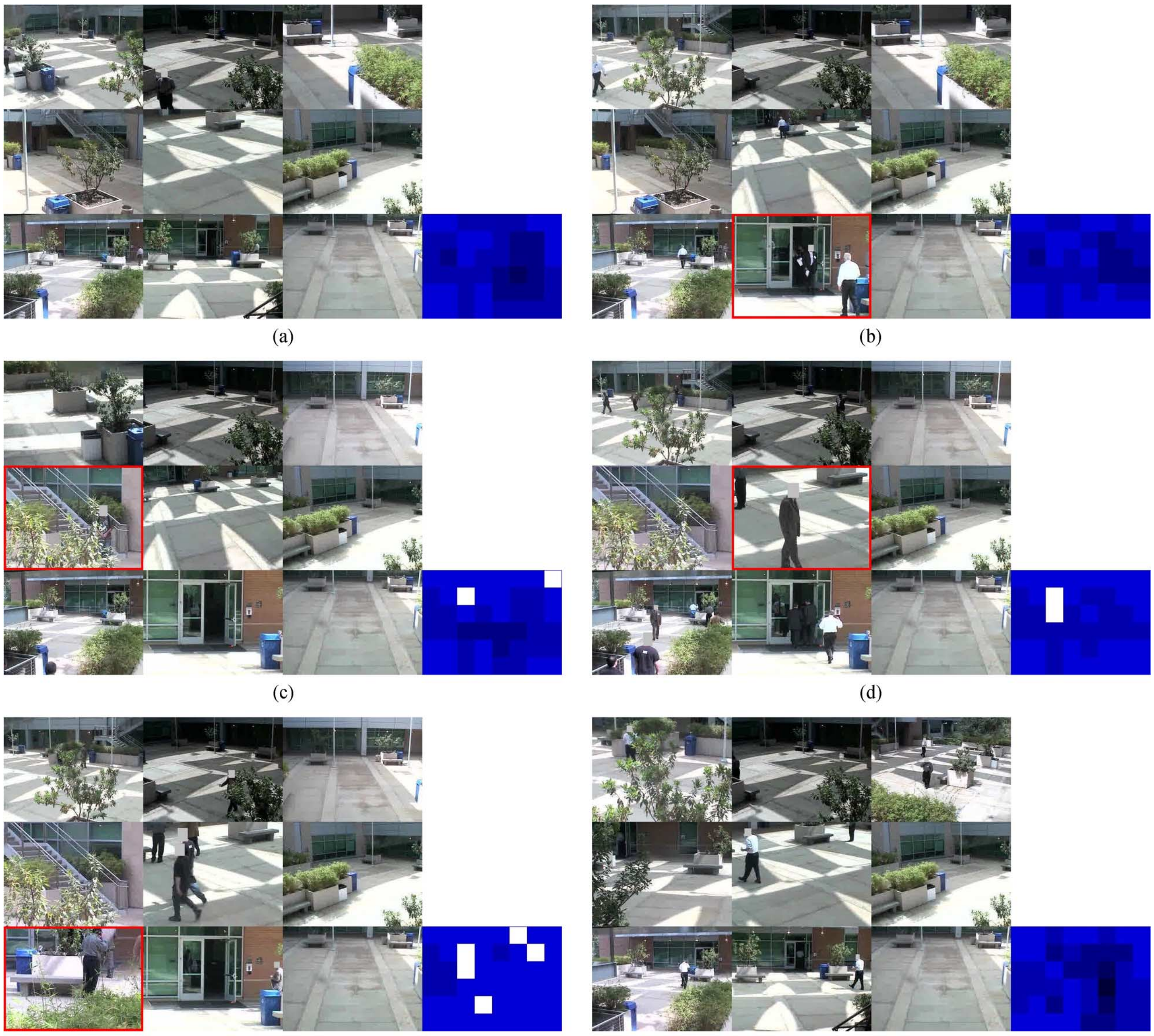
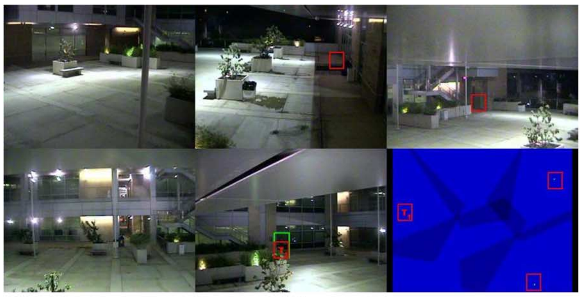


Fig. 5. Results of game-theoretic camera control with decreasing number of cameras available to the area coverage optimization. The coverage of the entire space is shown on the bottom-right of each subfigure, where blue areas are covered white areas are not. The darker the color of a block, the greater the number of cameras that is assigned there. The results with multiple cameras zooming in are shown in (b)–(e). The number of cameras that zooms in is increasing from 1 to 4. From (b) to (e), the view of new zooming in cameras is bounded by red lines. It is shown that, as the number of zoomed-in cameras increases, more areas are left uncovered. (f) Reinitialization results when we reset the cameras with none of them zooming in.

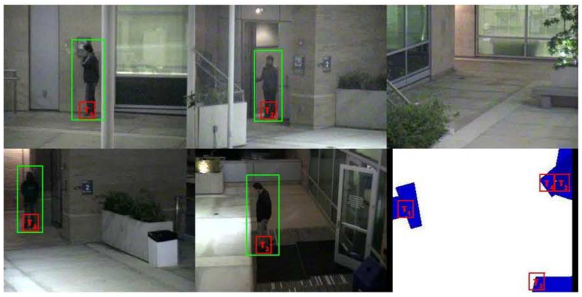
and exits. Since the entrances and exits must be monitored always, we treated them as static virtual targets, leading to a total of seven targets. Each camera in our setup is an independent entity with full network connectivity through a wireless network, with the entire system running in real time.

At initialization, all of the cameras apply the utility function defined in Section III-D-1 for the Area Coverage to cover the entire region under surveillance 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 Extended Kalman-Consensus filter and is processed along with the information from the filters running on neighboring cameras as described in Appendix A. The resulting consensus state estimate is then used by each camera to determine the best PTZ settings available according to the negotiation mechanism in Section III-C.

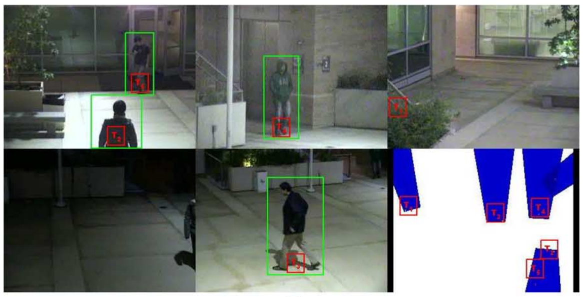
We compare two scenarios, i.e., Area Coverage and Target Coverage, as was explained in Section III-D. In the Area Coverage problem, the camera networks have to cover the entire area and take shots at lower resolutions, resulting in increased tracking error. We show the results for both the Area Coverage and Target Coverage approaches (where only targets and entrances are covered), and we clearly show the advantages of optimizing the tracking within the control module. The targets followed the same path through the courtyard during the collection of data for both cases. Fig. 6 shows the images captured by the actively controlled camera network at different time steps. Fig. 6(a) shows the result for the Area Coverage as the initialization. Fig. 6(b)–(d) shows the results for the Target Coverage. Since targets are free to move about the region under surveillance, the cameras in the network are required to adjust their parameters dynamically



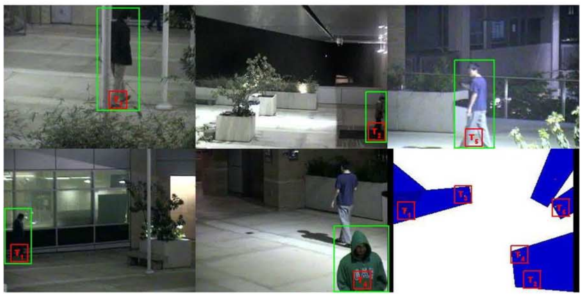
(a)



(b)

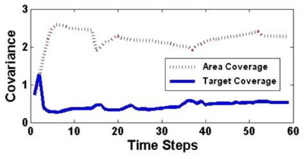


(c)

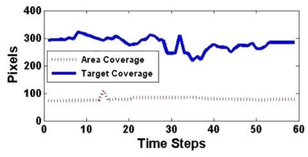


(d)

Fig. 6. Dynamic camera control images. Blue regions mark the FOV; darker regions identify camera overlap. Targets are marked in green with a red label. This figure is best viewed on a computer monitor. The video is available as a supplementary material. (a) Time step $k = 0$. (b) Time step $k = 2$. (c) Time step $k = 19$. (d) Time step $k = 36$.

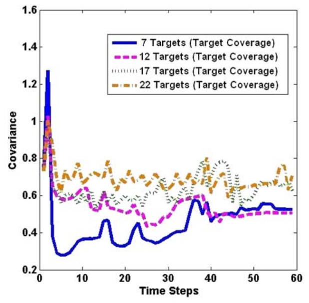


(a)

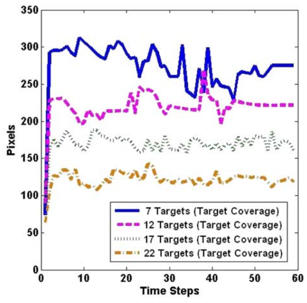


(b)

Fig. 7. Comparison of the average tracker covariance [i.e., $\mathbf{P}_i^j$, as shown in (6)] and average resolution [i.e., r_i^j, as shown in (8)] of all targets being actively tracked by a system using Target Coverage versus Area Coverage. (a) Average trace of tracker covariance of targets. (b) Average resolution of targets over time.

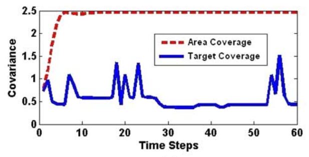


(a)

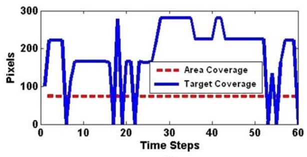


(b)

Fig. 8. Effects of increasing the number of targets on tracking covariance and image resolution of the targets. (a) Average trace of tracker covariance of targets. (b) Average resolution of targets over time.



(a)



(b)

Fig. 9. Tracking covariance and resolution of a single target $T_j \in N_t$ for $|N_t| = 22$. (a) Average trace of tracker covariance of targets. (b) Average resolution of targets over time.

to maintain shots of each target that optimize the utility functions presented in Section III-D2. To acquire these shots, the camera network concedes a large unobserved area. We can see in Fig. 7 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 proposed in this paper). This is because, at each time step, the camera network will choose the set of parameters that optimizes the utility, which is dependent on the error covariance of the Extended Kalman-Consensus filter.

C. *Performance Analysis*

Through another simulation, we analyze the performance of the proposed system in the cases of Example 1 and Example 2 (which are two common modes of operation). The goal of the simulation here is to show the effect of the number of targets on the performance of the proposed method. The simulation was modeled very similarly to our real-world setup, allowing us to do experiments that would be otherwise prohibitive due to the setup and maintenance costs involved with active camera network experiments. New simulated targets were evenly distributed around the available area. We show below the effects of increasing the number of targets within the surveillance region and discuss the dynamics of our system.

Fig. 8(a) and (b) shows that, as the number of targets increases, the average tracking covariance does not significantly change, in contrast to the change in average resolution. Note that the resolutions are still at a much better value compared with the system for Area Coverage. An example of one of the targets is shown in Fig. 9. The reason for the small effect on tracking covariance increase is partly due to the low weight to the risk in our experiments. This allows the system to choose to leave some targets unobserved at any given time step in favor of improving poorly tracked or poorly resolved shots of targets. We can see in Fig. 9(a) and (b) that at some time instants the target is unobserved. At these instants, the number of pixels imaged is zero and the tracking covariance increases. It is also clear that, as the tracking covariance increases, the greater the incentive for the system to image the target at a setting that minimizes the increasing error. Thus, the system will zoom out to get an image of the target. This is shown in the plots in Fig. 9 where the target is reacquired.

V. Conclusion

We have presented in this paper an approach to optimize various scene analysis performance criteria through distributed control of a dynamic camera network. An Extended Kalman-Consensus filtering approach was used to track multiple targets in a dynamic network. A camera control framework using a distributed optimization approach allowed selection of parameters to acquire images that best optimized the user-specified requirements. We also considered the uncertainty in state estimation when deciding upon camera settings and the effect of different utility function weights. We also demonstrated the effectiveness of our system by showing results from a real-life camera network in addition to analysis of simulation results. This paper can lay the foundations for autonomous systems with embedded video processors that are capable of opportunistic sensing, efficient navigation, and scene analysis.

Appendix A
Extended Kalman-Consensus Filter

The Extended Kalman-Consensus filter allows us to track targets on the ground plane using multiple measurements in the image plane taken from various cameras. This allows each camera C_i to have at any time step k a consensus state estimate $\bar{\mathbf{x}}_i^j$ and estimate error covariance $\mathbf{P}_i^j$ for each target T_j. Let C_i^j be the set of cameras viewing target T_j. To model the motion of a target T_j on the ground plane, we use (1) with the nonlinear observation model of (2) for each camera C_i, where $h_i(.)$ is the mapping from the ground to the image plane for camera C_i, $\mathbf{w}^j(k)$ and $\mathbf{v}_i^j(k)$ are zero-mean white Gaussian noise (i.e., $\mathbf{w}^j(k) \sim \mathcal{N}(0, \mathbf{Q}^j)$ and $\mathbf{v}_i(k) \sim \mathcal{N}(0, \mathbf{R}_i^j)$, respectively), and $\mathbf{x}^j(0)$ is the initial state of the target. The estimate state $\hat{\mathbf{x}}_i^j$ of T_j is based on the observations by the cameras viewing T_j. The noisy measurement $\mathbf{z}_i^j(k)$ at camera C_i is the sensed target position $({}^{im}x_i^j(k), {}^{im}y_i^j(k))$ on C_i's image plane.

Due to the nonlinear nature of the observation model, the linear Kalman-Consensus filter proposed in [21] cannot be applied as is. An extension to deal with the nonlinearity of the observation model is required. Taking into account the nonlinear nature of our dynamical model, we utilize an Extended Kalman-Consensus distributed tracking algorithm. The following are our basic Extended Kalman filter iterations, as implemented in each camera, in the information form:[1]

- Prediction

$$\begin{aligned}\mathbf{P}(k+1) &= \mathbf{A}(k)\mathbf{M}(k)\mathbf{A}(k)^T + \mathbf{B}(k)\mathbf{Q}(k)\mathbf{B}(k)^T\\ \bar{\mathbf{x}}(k+1) &= \mathbf{A}(k)\hat{\mathbf{x}}(k)\end{aligned} \tag{21}$$

- Correction

$$\begin{aligned}\mathbf{M}(k)^{-1} &= \mathbf{P}(k)^{-1} + \mathbf{H}(k)^T\mathbf{R}(k)^{-1}\mathbf{H}(k)\\ \mathbf{K}(k) &= \mathbf{M}(k)\mathbf{H}(k)^T\mathbf{R}(k)^{-1}\\ \hat{\mathbf{x}}(k) &= \bar{\mathbf{x}}(k) + \mathbf{K}(k)\left(\mathbf{z}(k) - h\left(\bar{\mathbf{x}}(k)\right)\right)\end{aligned} \tag{22}$$

[1]The subscript/superscript that indicate camera/target is dropped here for ease of reading.

where $\mathbf{P}$ and $\mathbf{M}$ are the *a priori* and *a posteriori* estimate error covariance matrices, respectively, and $\mathbf{H}$ is the Jacobian matrix of partial derivatives of h with respect to $\mathbf{x}$, i.e.,

$$\mathbf{H}_{[m,n]} = \left.\frac{\partial h_{[m]}}{\partial \mathbf{x}_{[n]}}\right|_{\mathbf{x}=\bar{\mathbf{x}}(k)}. \tag{23}$$

This algorithm is distributedly performed in each camera node C_i. At each time step k and for each target T_j, we assume to be given the estimated *a priori* target state $\bar{\mathbf{x}}_i^j(k)$ and the error covariance matrix $\mathbf{P}_i^j$. At time step $k = 0$, the Extended Kalman-Consensus filter is initialized with $\mathbf{P}_i^j = \mathbf{P}_0$ and $\bar{\mathbf{x}}_i^j = E\langle \mathbf{x}_i^j(0)\rangle$. The consensus algorithm is shown in Algorithm 1.

Algorithm 1 Distributed Extended Kalman-Consensus tracking algorithm performed by every C_i at discrete time step k. The state of T_j is represented by $\mathbf{x}_i^j$ with error covariance matrix $\mathbf{P}_i^j$

```
*Input:* $\bar{\mathbf{x}}_i^j$ and $\mathbf{P}_i^j$ from time step $k-1$
**for** each $T_j$ that is being viewed by $\{\mathcal{C}_i^n \cup C_i\}$ **do**
    Obtain measurement $\mathbf{z}_i^j$ with covariance $\mathbf{R}_i^j$
    Calculate Jacobian matrix $\mathbf{H}_i^j$ with respect to the observation model
        $(\mathbf{H}_i^j)_{[m,n]} = \left.\frac{\partial (h_i)_{[m]}}{\partial \mathbf{x}_{[n]}}\right|_{\mathbf{x}=\bar{\mathbf{x}}_i^j}$
    Compute information vector and matrix
        $\mathbf{u}_i^j = {\mathbf{H}_i^j}^T {\mathbf{R}_i^j}^{-1} \mathbf{z}_i^j$ and $\mathbf{U}_i^j = {\mathbf{H}_i^j}^T {\mathbf{R}_i^j}^{-1} \mathbf{H}_i^j$
    Compute the predicted measurement
        $\mathbf{g}_i^j = {\mathbf{H}_i^j}^T {\mathbf{R}_i^j}^{-1} h_i\left(\bar{\mathbf{x}}_i^j\right)$
    Compute the residue
        $\mathbf{r}_i^j = \mathbf{u}_i^j - \mathbf{g}_i^j$
    Send message $\mathbf{m}_i^j = (\mathbf{r}_i^j, \mathbf{U}_i^j, \bar{\mathbf{x}}_i^j)$ to neighboring cameras $\mathcal{C}_i^n$
    Receive messages $\mathbf{m}_l = (\mathbf{r}_l^j, \mathbf{U}_l^j, \bar{\mathbf{x}}_l^j)$ from all cameras $C_l \in \mathcal{C}_i^n$
    Fuse information
        $\mathbf{y}_i^j = \sum_{l \in (C_i \cup \mathcal{C}_i^n)} \mathbf{r}_l^j, \quad \mathbf{S}_i^j = \sum_{l \in (C_i \cup \mathcal{C}_i^n)} \mathbf{U}_l^j$
    Compute the Extended Kalman-Consensus state estimate
        $\mathbf{M}_i^j = \left((\mathbf{P}_i^j)^{-1} + \mathbf{S}_i^j\right)^{-1}$
        $\hat{\mathbf{x}}_i^j = \bar{\mathbf{x}}_i^j + \mathbf{M}_i^j \mathbf{y}_i^j + \gamma \mathbf{M}_i^j \sum_{l \in \mathcal{C}_i^n} \left(\bar{\mathbf{x}}_l^j - \bar{\mathbf{x}}_i^j\right)$
        $\gamma = 1 \Big/ \left(\|\mathbf{M}_i^j\| + 1\right), \|\mathbf{x}\| = \left(tr(\mathbf{x}^T\mathbf{x})\right)^{\frac{1}{2}}$
    Update the state and error covariance matrix for time step $k$
        $\mathbf{P}_i^j \leftarrow \mathbf{A}^j \mathbf{M}_i^j {\mathbf{A}^j}^T + \mathbf{B}^j \mathbf{Q}^j {\mathbf{B}^j}^T$
        $\bar{\mathbf{x}}_i^j \leftarrow \mathbf{A}^j \hat{\mathbf{x}}_i^j$
**end for**
```

We now describe the consensus algorithm, as shown in Algorithm 1. It is performed at each C_i distributedly for each T_j that is viewed by $\{\mathcal{C}_i^n \cup C_i\}$, where $\mathcal{C}_i^n$ is the neighboring camera set of C_i and defined as all cameras with which C_i can directly communicate. If C_i is viewing a target T_j, it obtains T_j's position on its image plane $\mathbf{z}_i^j$ and calculates the Jacobian matrix $\mathbf{H}_i^j$ of its observation model and consensus state estimate $\bar{\mathbf{x}}_i^j$. After this, the corresponding information vector and matrix are computed with the given measurement covariance matrix $\mathbf{R}_i^j$ and $\mathbf{H}_i^j$. C_i then sends a message $\mathbf{m}_i^j$ to its neighbors, which includes the computed information vector and matrix, and its estimated target state $\bar{\mathbf{x}}_i^j$ at previous time step $(k-1)$. C_i then receives similar messages $\mathbf{m}_l$ only from the cameras in its neighborhood that are also viewing T_l. The information matrices and vectors received from these messages, and its own information matrix and vector, are then fused, and the Extended Kalman-Consensus state estimate is computed. Finally, ground plane state $\bar{\mathbf{x}}_i^j$ and error covariance matrix $\mathbf{P}_i^j$ are updated according to the assumed linear dynamical system.

Appendix B
Game Theory Fundamentals for Camera Networks

Below, we summarize some known results in game theory-based distributed optimization as they pertain to the camera network problem in this paper [20].

Definition B.1: A game is an **exact potential game** if there exists potential function ϕ: $\mathcal{A} \to \mathbb{R}$ such that for every camera, $C_i \in \mathcal{C}$, for every $\mathbf{a}_{-i} \in \mathcal{A}_{-i}$ and for every $a_i, a_i' \in \mathcal{A}_i$

$$U_{C_i}(a_i, \mathbf{a}_{-i}) - U_{C_i}(a_i', \mathbf{a}_{-i}) = \phi(a_i, \mathbf{a}_{-i}) - \phi(a_i', \mathbf{a}_{-i})$$

where $\mathcal{A} = \mathcal{A}_1 \times \mathcal{A}_2 \times \ldots \times \mathcal{A}_{N_c}$ is the strategy space and $\mathcal{A}_{-i} = \mathcal{A}_1 \times \ldots \times \mathcal{A}_{i-1} \times \mathcal{A}_{i+1} \times \ldots \times \mathcal{A}_{N_c}$.

Definition B.2: A game is an **ordinal potential game** if there exists ordinal potential function ϕ: $\mathcal{A} \to \mathbb{R}$ such that for every camera, $C_i \in \mathcal{C}$, for every $\mathbf{a}_{-i} \in \mathcal{A}_{-i}$ and for every $a_i, a_i' \in \mathcal{A}_i$

$$U_{C_i}(a_i, \mathbf{a}_{-i}) - U_{C_i}(a_i', \mathbf{a}_{-i}) > 0 \Leftrightarrow \phi(a_i, \mathbf{a}_{-i}) - \phi(a_i', \mathbf{a}_{-i}) > 0.$$

Result (Existence of Pure Nash Equilibrium): If a game involving the set of cameras $\mathcal{C}$ has a continuous ordinal potential function ϕ, then the game has at least one Nash Equilibrium.

Proof: Let $\mathbf{a} \in \mathcal{A}$ be the set of assigned camera parameters. Let $\phi : \mathcal{A} \to \mathbb{R}$ be an ordinal potential function. Consider the case where a single camera changes parameters from a_i to $b_i (a_i, b_i \in \mathcal{A}_i)$.

Let ΔU_{C_i} be the change in utility caused by the change in parameters

$$\Delta U_{C_i} = U_{C_i}(b_i, \mathbf{a}_{-i}) - U_{C_i}(a_i, \mathbf{a}_{-i}) > 0.$$

Let $\Delta\phi$ be the change in potential caused by the change in parameters

$$\Delta\phi = \phi(b_i, \mathbf{a}_{-i}) - \phi(a_i, \mathbf{a}_{-i}) > 0.$$

Thus, we can conclude that, for a single camera's parameters, change we get

$$\Delta\phi > 0 \Rightarrow \Delta U_{C_i} > 0.$$

This means that we can start from an arbitrary $\mathbf{a}$, and at each step, one camera increases its utility. Likewise, at each step, ϕ is increased by $\Delta\phi > 0$. Since ϕ can accept only a finite number of values, it will eventually reach a local maxima. At this point, no camera can achieve improvement, and we reach a Nash Equilibrium. ■

References

[1] A. Alahi, D. Marimon, M. Bierlaire, and M. Kunt, "A master-slave approach for object detection and matching with fixed and mobile cameras," presented at the Int. Conf. Image Process., San Diego, CA, 2008.

[2] Y. Aloimonos, I. Weiss, and A. Bandyopadhyay, "Active vision," in *International Journal of Computer Vision*. New York: Springer-Verlag, 1988.

[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, Sep. 2007.

[4] R. Bajcsy, "Active perception," *Proc. IEEE*, vol. 76, no. 8, pp. 996–1005, Aug. 1988.

[5] D. Ballard, "Animate vision," *Art. Intell.*, vol. 48, no. 1, pp. 57–86, Feb. 1991.

[6] D. Devarajan, Z. Cheng, and R. Radke, "Calibrating distributed camera networks," *Proc. IEEE*, vol. 96, no. 10, pp. 1625–1639, Oct. 2008.

[7] W. Du and J. Piater, "Multi-camera people tracking by collaborative particle filters and principal axis-based integration," in *Proc. Asian Conf. Comput. Vis.*, 2007, pp. 365–374.

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

[9] E. Ermis, V. Saligrama, P. Jodoin, and J. Konrad, "Abnormal behavior detection and behavior matching for networked cameras," in *Proc. IEEE/ACM Int. Conf. Distrib. Smart Cameras*, 2008, pp. 1–10.

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

[11] S. Khan, O. Javed, Z. Rasheed, and M. Shah, "Camera handoff: Tracking in multiple uncalibrated stationary cameras," in *Proc. IEEE Workshop Human Motion*, 2000, pp. 113–118.

[12] S. Khan, O. Javed, Z. Rasheed, and M. Shah, "Human tracking in multiple cameras," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2001, vol. 1, pp. 331–336.

[13] S. Khan and M. Shah, "A multiview approach to tracking people in crowded scenes using a planar homography constraint," in *Proc. Eur. Conf. Comput. Vis.*, 2006, pp. 133–146.

[14] N. Li and J. Marden, "Designing games for distributed optimization," in *Proc. IEEE CDC-ECC*, Orlando, FL, Dec. 2011, pp. 2434–2440.

[15] Y. Li and B. Bhanu, "Utility-based dynamic camera assignment and hand-off in a video network," in *Proc. IEEE/ACM Int. Conf. Distrib. Smart Cameras*, 2008, pp. 1–9.

[16] J. Marden, G. Arslan, and J. Shamma, "Cooperative control and potential game," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 39, no. 6, pp. 1393–1407, Dec. 2009.

[17] D. Markis, T. Ellis, and J. Black, "Bridging the gap between cameras," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2004, pp. II-205–II-210.

[18] C. Micheloni, G. Foresti, and L. Snidaro, "A network of cooperative cameras for visual-surveillance," *Proc. Inst. Elect. Eng.—Vis., Image Signal Process.*, vol. 152, no. 2, pp. 205–212, Apr. 2005.

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

[20] D. Monderer and L. S. Shapley, "Potential Games," *Games Econom. Behav.*, vol. 14, no. 1, pp. 124–143, May 1996.

[21] R. Olfati-Saber and N. F. Sandell, "Distributed tracking in sensor networks with limited sensing range," in *Proc. Amer. Control Conf.*, Jun. 2008, pp. 3157–3162.

[22] F. Qureshi and D. Terzopoulos, "Surveillance in virtual reality: System design and multi-camera control," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2007, pp. 1–8.

[23] A. Rahimi and T. Darrell, "Simultaneous calibration and tracking with a network of non-overlapping sensors," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2004, pp. I-187–I-194.

[24] A. Sankaranarayanan, A. Veeraraghavan, and R. Chellappa, "Object detection, tracking and recognition for multiple smart cameras," *Proc. IEEE*, vol. 96, no. 10, pp. 1606–1624, Oct. 2008.

[25] *Cooperative Control of Distributed Multi-Agent Systems*, J. Shamma, Ed. Hoboken, NY: Wiley-Interscience, 2008.

[26] B. Song, C. Ding, A. Kamal, J. Farrell, and A. Roy-Chowdhury, "Distributed camera networks," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 20–31, May. 2011.
[27] B. Song, A. T. Kamal, C. Soto, C. Ding, J. A. Farrell, and A. K. Roy-Chowdhury, "Tracking and activity recognition through consensus in distributed camera networks," *IEEE Trans. Image Process.*, vol. 19, no. 10, pp. 2564–2579, Oct. 2010.
[28] B. Song and A. Roy-Chowdhury, "Stochastic adaptive tracking in a camera network," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2007, pp. 1–8.
[29] B. Song, C. Soto, A. K. Roy-Chowdhury, and J. A. Farrell, "Decentralized camera network control using game theory," in *Proc. Workshop ICDSC*, 2008, pp. 1–8.
[30] C. Soto, B. Song, and A. K. Roy-Chowdhury, "Distributed multi-target tracking in a self-configuring camera network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recog.*, 2009, pp. 1486–1493.
[31] B. Stancil, C. Zhang, and T. Chen, "Active multicamera networks: From rendering to surveillance," *IEEE J. Sel. Topics Signal Process.*, vol. 2, no. 4, pp. 597–605, Aug. 2008.
[32] S. Stillman and T. Tanawongsuwan, "Tracking multiple people with multiple cameras," in *Proc. Int. Conf. Audio- Video-Based Biometric Person Authentication*, 1999.
[33] M. Taj and A. Cavallaro, "Distributed and decentralized multicamera tracking," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 46–58, May 2011.
[34] K. Tieu, G. Dalley, and W. Grimson, "Inference of non-overlapping camera network topology by measuring statistical dependence," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2005, pp. 1842–1849.
[35] R. Tron and R. Vidal, "Distributed computer vision algorithms," *IEEE Signal Process. Mag.*, vol. 28, no. 3, pp. 32–45, May 2011.
[36] R. Tron and R. Vidal, "Distributed computer vision algorithms through distributed averaging," in *Proc. IEEE Conf. Comput. Vision Pattern Recog. (CVPR)*, 2011, pp. 57–63.
[37] D. Wolpert and K. Tumor, "A survey of collectives," in *Collectives and the Design of Complex Systems*, K. Tumer and D. Wolpert, Eds. New York: Springer-Verlag, 2004.
[38] C. Wu and H. Aghajan, "Real-time human pose estimation: A case study in algorithm design for smart camera networks," *Proc. IEEE*, vol. 96, no. 10, pp. 1715–1732, Oct. 2008.
[39] H. Young, *Individual Strategy and Social Structure: An Evolutionary Theory of Institutions*. Princeton, NJ: Princeton Univ. Press, 1998.
[40] J. Zhao, S. C. Cheung, and T. Nguyen, "Optimal camera network configurations for visual tagging," *IEEE J. Sel. Topics Signal Process.*, vol. 2, no. 4, pp. 464–479, Aug. 2008.

Chong Ding received the B.S. degree in computer science from the University of California, Riverside, in 2008, where he is currently pursuing the Ph.D. degree in the Department of Computer Science.

His main research interests include intelligent camera networks, wide-area scene analysis, and distributed and real-time systems.

Bi Song received the B.S. and M.S. degrees in electronic engineering and information science from the University of Science and Technology of China, Hefei, China, in 2001 and 2004, respectively, and the Ph.D. degree from the University of California, Riverside, in 2009.

She is currently a Sr. Applied Research Engineer at Sony Electronics Inc., San Jose, CA. She was a postdoctoral scholar at the University of California from 2009 to 2011. Her main research interests include distributed camera networks and motion analysis in video.

Akshay Morye received the Diploma in industrial electronics from Maharashtra State Board of Technical Education, Mumbai, India, in 2003, the B.E. degree in electronics engineering from the University of Mumbai, Mumbai, in 2006, and the M.S. degree in electrical engineering from the University of California, Riverside, in 2011, where he is currently pursuing the Ph.D. degree in the Department of Electrical Engineering.

His research interests include control and robotics, distributed constraint optimization, and algorithms for distributed camera sensor networks.

Jay A. Farrell received the B.S. degree in physics and electrical engineering from Iowa State University, Ames, in 1986 and the M.S. and Ph.D. degrees in electrical engineering from the University of Notre Dame, Notre Dame, IN, in 1988 and 1989, respectively.

From 1989 to 1994, at Charles Stark Draper Laboratory, he was a Principal Investigator on projects involving intelligent and learning control systems for autonomous vehicles. He is the current Chair of the Department of Electrical Engineering, University of California, Riverside, where he is currently a Professor. He is the author of the book *Aided Navigation: GPS with High Rate Sensors* (McGraw-Hill, 2008) and of more than 170 technical publications. He is also a coauthor of the books *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).

Dr. Farrell is a Fellow of the American Association for the Advancement of Science (2011) and a Distinguished Member of the IEEE Control Systems Society (CSS). He has served as Vice-President of Finance and Vice-President of Technical Activities for the IEEE CSS and as General Chair of the 2012 IEEE Conference on Decision and Control. He was a recipient of the Engineering Vice President's Best Technical Publication Award in 1990 and the Recognition Awards for Outstanding Performance and Achievement in 1991 and 1993.

Amit K. Roy-Chowdhury received the Bachelor's degree in electrical engineering from Jadavpur University, Calcutta, India, the Master's degree in systems science and automation from the Indian Institute of Science, Bangalore, India, and the Ph.D. degree in electrical engineering from the University of Maryland, College Park.

He is an Associate Professor of electrical engineering and a Cooperating Faculty in the Department of Computer Science, University of California, Riverside. His broad research interests include the areas of image processing and analysis, computer vision, and 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. He is a coauthor of two books *Camera Networks: The Acquisition and Analysis of Videos over Wide Areas* and *Recognition of Humans and Their Activities Using Video*. He is the editor of the book *Distributed Video Sensor Networks*. He has been on the organizing and program committees of multiple computer vision and image processing conferences and is serving on the editorial boards of IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS B, Machine Vision Applications, and Elsevier e-reference on Signal Processing.