Multi-camera scheduling for video production

Fahad Daniyal and Andrea Cavallaro

Queen Mary University of London
Mile End Road, E1 4NS London, United Kingdom
Email: {fahad.daniyal, andrea.cavallaro}@eecs.qmul.ac.uk

Abstract

We present a novel algorithm for automated video production based on content ranking. The proposed algorithm generates videos by performing camera selection while minimizing the number of inter-camera switches. We model the problem as a finite horizon Partially Observable Markov Decision Process over temporal windows and we use a multivariate Gaussian distribution to represent the content-quality score for each camera. The performance of the proposed approach is demonstrated on a multi-camera setup of fixed cameras with partially overlapping fields of view. Subjective experiments based on the Turing test confirmed the quality of the automatically produced videos. The proposed approach is also compared with recent methods based on Recursive Decision and on Dynamic Bayesian Networks and its results outperform both methods.

Keywords: Best-view selection; Feature analysis; Content ranking; Autonomous video production; Camera scheduling.

1 Introduction

The selection of the camera that best describes a dynamic scene is an important problem in multi-camera networks. This selection is useful for automated video production and for highlights generation. The main challenges in addressing this problem are the effective analysis of the video content and the identification of the best view to satisfy the objectives of the task at hand. These objectives can be selecting the camera capturing the maximum number of people, or offering the best view of a specific person or an activity.

Camera selection can be seen as an optimization [10] or as a scheduling [14] problem, where the goal is to maximize the visibility of the features or events of interest while minimizing inter-camera switching. Camera selection methods involve a content analysis stage that assigns a score to each camera based on its *Quality of View* (QoV). Scoring can be based on a single feature [15] or a combination of features [9]. Naive methods for best-view selection based on QoV usually perform poorly, as they generally produce frequent view changes [10]. To mitigate this problem, reward and cost functions associated to camera switching have been introduced [6, 11]. Reward can be expressed in terms of feature observability and smoothness of the final output video and a cost is incurred whenever the selected view is switched. In general, best-view selection requires knowledge of the selected camera and the QoV over a finite time horizon [15]. Moreover the camera selection strategy should be able to predict the time intervals during which features or objects of interest would be most visible [12]. For this reason, an efficient camera selection strategy should take into account past as well as future (or predicted) information.

The works in [2, 15] use a *scheduling interval* to observe targets for a certain minimal duration. This approach does not scale to large camera networks where multiple cameras may be simultaneously competing for the best view. In [2], Time Dependent Orienteering (TDO), target motion, position, target birth and deadline are used to trigger a pan-tilt-zoom camera that captures targets in the scene. The cost of the system is associated to the number of targets not captured. The scheduling strategy is the Kinetic Traveling Salesperson Problem with deadlines. A schedule to observe targets is chosen that minimizes the path cost in terms of TDO. This work does not consider target occlusions and does not predict the best time intervals to capture images. In [10] a cost function is proposed that depends on QoV using features such as object size, pose and orientation. While this approach minimizes frequent switches, it tends not to select the best-view for highly dynamic scenes, as demonstrated in Sec. 3.

Scheduling strategies based on *queue processing* techniques have also been used for camera selection [19, 14]. In this case views compete to be selected and are assigned priorities based on their features. In [19] when more than one person is in the view of a fixed camera an active camera focuses on the closest target. The performance of the system does not scale with the number of objects as the camera switches from target to target. The work in [14] uses a Weighted Round Robin technique for scheduling each target that enters the monitored area, but no penalties are assigned to frequent camera switching. *Greedy scheduling policies* have also been used for camera scheduling [4]. In these methods targets are treated as network packets and routing approaches based on techniques such as First Come First Served (FCFS), Earliest Deadline First (EDF) and Current Minloss Throughput Optimal (CMTO) are used. These approaches do not include the transition cost for the camera that is associated with target swaps. Moreover, all these approaches assume that the dynamics of the observed site remain constant when a certain person is viewed and,

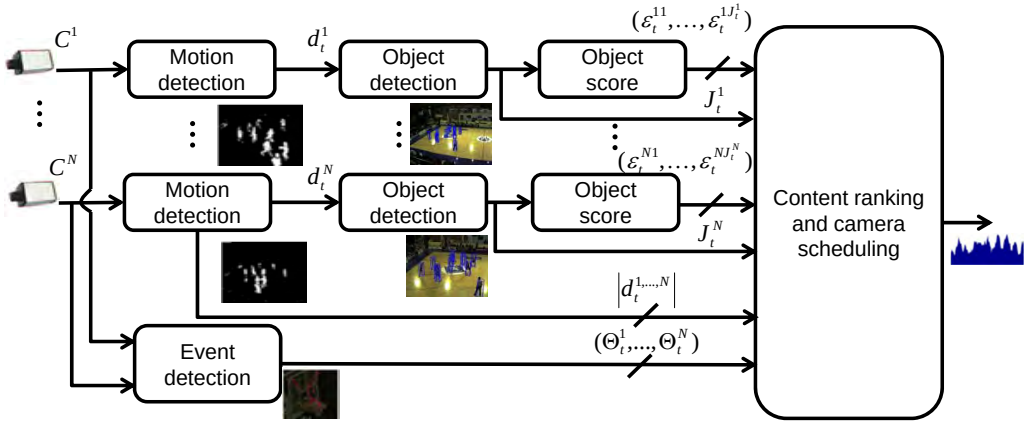


Figure 1: Block diagram of the proposed approach.

while minimizing the switches, they do not quantify the loss of information in the views that are not selected.

In this paper we model the view-selection problem as a decision process during which information is only partially visible. In particular we use a Partially Observable Markov Decision Process (POMDP), where the process measured by the cameras (e.g., object size, location or scene activity) is a Markov process and the sensor scheduling is based on recursively estimating and updating the belief state, the sensor-scheduling actions, and the posterior distribution of the process given the history of the sensor measurements. We represent the process dynamics and measurements as linear Gaussian state-space models and track the belief using the Bayes Rule. The reward for camera selection is modeled as a function on a content-quality score and the related camera switching. This reward modeling allows the proposed approach to control the number of camera switches, thus enabling the generation of pleasant videos. The proposed approach is tested using both objective and subjective evaluations on a real multi-camera setup with partially overlapping fields of view.

The paper is organized as follows. In Sec. 2 we present the proposed approach for feature-based camera selection. Experimental results on a real multi-camera basketball dataset are evaluated and discussed in Sec. 3. Finally conclusions are drawn in Sec. 4.

2 Proposed approach

Let a site be monitored by a set of cameras $\mathbf{C} = \{C^1,\ldots,C^N\}$, with $N \geq 2$. Dynamic best-view selection can be regarded as a three-stage problem (see Fig. 1).

The *first stage* focuses on the extraction over time t for each view i (1) of a feature vector for each object j within the view, ψ_t^{ij}, and (2) of features associated to the entire camera view, ψ_t^i. The selection of the features depends on the task at hand. In the *second stage*, a QoV score, ρ_t^i, is computed for each camera C^i at each time t based on the object features $\psi_t^{i,o}$ (i.e., all $\psi_t^{ij} : j = 1,\ldots,J_t^i$ where J_t^i is the number of objects in the view of C^i at time t) and the camera features ψ_t^i. ρ_t^i can then be represented as a measure of feature visibility:

$$\rho_t^i = \mathscr{M}(\psi_t^{i,o}, \psi_t^i), \tag{1}$$

where $\mathscr{M}(.)$ generates the QoV ρ_t^i given the two feature vectors. In the *third stage*, a best-camera selection mechanism is constructed as a function of time t such that the best trade off between the best-camera selection and number of switches is found.

2.1 Camera selection

Let the selected camera at time t be represented by an N dimensional vector $\Omega_t^{i^*} = (c_t^1,\ldots,c_t^N)$ which has 1 only in the index i^* and is 0 elsewhere. The best view can be selected for each t as

$$i^* = \underset{i=1,\ldots,N}{\operatorname{argmax}}\,(\rho_t^1,\rho_t^2,\ldots\rho_t^N). \tag{2}$$

However in such selection the number of switches are not constrained, thus generating unpleasant videos[1]. To solve this problem, fixed constraints such as a minimum scheduling period Δ can be introduced [2]. However in realistic scenarios such a constraint may cause loss of information as sudden and important changes in video content from one view will not be catered for. To this end it is preferable to set $\Delta \to 1$, such that the selected camera is the best camera most of the time [10]. To constrain the number of inter-camera switching, past information [10] as well as future information [12] can be used. Moreover, the best-camera selection process should take into account the currently selected view. This dynamic modeling of the multi-camera system can be done by modeling the state of each camera using a random variable at each point of time. The instantaneous *snapshot* of such random variables, at each t, describes the state of our multi-camera system at t. To this end we model the camera selection problem as a Markovian Decision Process (MDP), with partially observable states (POMDP) where each decision (action) takes our system

[1] An example can be seen at http://www.eecs.qmul.ac.uk/~andrea/view-selection.html

to the next state. POMDP implies control over the states while having partial observability i.e. no information about the future states [18]. Within the POMDP framework, for best-view selection we first map the observation and the selected camera to associate a utility to the system; then an optimal policy is defined that maximizes this utility. The maximization of the utility relates to the best-view selection.

A POMDP can be defined by the influence diagrams shown in Fig. 2. Let the state of a camera C^i at time t be represented as $s_t^i \in \mathbb{R}^+$, where the state space for the POMDP is $S = [0,1]$. Thus the state for the multi-camera system at time t can be expressed as

$$s_t = (\rho_t^1, \ldots, \rho_t^N) \in (\mathbb{R}^+)^N. \tag{3}$$

Let the action space be represented by $\mathbf{C}$ and the action at any time be represented by the camera transition as $c_t^j \rightarrow c_{t+1}^i$. Then the reward $u(s_t, c_t^i)$ of selecting a camera $c_t^i \in \mathbf{C}$, given the state s_t can be represented by the one-step reward function

$$u(s_t, c_t^i) = \alpha \rho_t^i + (1-\alpha)\vartheta_t^i, \tag{4}$$

where $\alpha \in [0,1]$ is a scaling factor and $\vartheta \in \{0,1\}$ is defined based on the previously selected camera

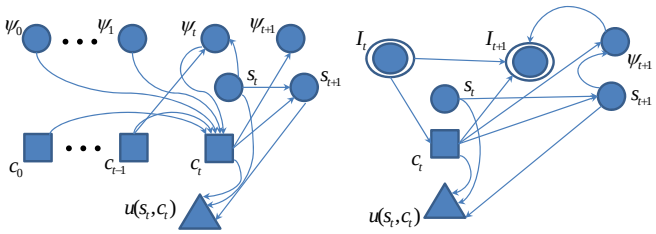
$$\vartheta_t^i = \begin{cases} 1 & \text{if } c_{t-1}^i = 1 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

It should be noted that if $\alpha = 1$, $u(s_t, c_t^n) = \rho_t^n$ thus converting this utility into the quality score (Eq. 10). Hence the system will select only the best camera over a temporal window without introducing any smoothing. The one-step cost function described in Eq. 4 is an integrated metric that accounts for both camera switching and the observability of features given by the accumulated quality score at each time k. The state space of a POMDP is continuous and estimating the current state from such a large state space is computationally intractable. Thus approximate solutions for state estimation are formulated [16]. These solutions assume that the space S is quantized with a factor g such that the quantized state is represented as $s_t^d = g.s_t$, where $g = (g_1, g_2, ..., g_S)$ and $g_{k+1} > g_k$, with $k \in [1, S]$. For clarity, we will drop the superscript d from s_t^d and refer to this discrete state as s_t.

The solution to the POMDP is a policy that can be represented as $\pi = \{\mu(p(s_t|I_t))\}$ such, that for each t, $\mu(p(s_t|I_t))$ is a state feedback map that specifies an action $c_t^i \rightarrow c_{t+1}^j$ on $\mathbf{C}$ depending on the belief state probability $p(s_t|I_t)$. A graphical representation is shown in Fig. 3 where the posterior probability distribution of the state s_t is conditioned on the observable history I_t such that

$$I_t := \begin{cases} p_0 & \text{if } t = 0 \\ (p_0, \varpi_0, \ldots, \varpi_t) & \text{otherwise} \end{cases} \tag{6}$$

Here $\varpi_t = (\Omega_t^i, (\psi_t^1, \ldots, \psi_t^N))$, where p_0 is the initial probability distribution and $\psi_t^i \in \Psi$ is the observation from C^i,



(a) (b)

Figure 2: Influence diagram describing a POMDP model. Rectangles correspond to decision nodes (actions), circles to random variables (states) and triangles to reward nodes. Links represent the dependencies among the components. s_t, Ω_t^i, ψ_t, and $u(.)$ denote the state, action, observation and reward at time t. Information states (I_t and I_{t+1}) are represented by double-circled nodes. (a) Note that an action at time t depends on past observations and actions, not on the states. (b) An action choice (rectangle) depends on the current information state only.

drawn from the observation space Ψ, given by the observation equation as

$$\psi_t^i = h(s_t^i, w_t), \tag{7}$$

$$\psi_t = (\psi_t^1, \ldots \psi_t^N), \tag{8}$$

where h represents the observation map and w_t represent the randomness in the observations at time t. We assume that w_t is an independent and identically distributed (iid) random variable with zero-mean Gaussian distribution. Then the sequence of states within the POMDP are generated such that at time $t = 0$ the system starts at an initial unobservable state s_0 with the given initial distribution p_0. If at any time t, the system is in state $s_t \in S$, and taking an action $c_{t-1}^j \rightarrow c_t^i$ (selecting the camera C^i given that the camera C^j was selected at the previous time instance $t-1$) takes the system to the next state $s_{t+1} \in S$ and an immediate reward $u(s_{t+1}, \Omega_t^i)$ is achieved. This state transition is governed by the state transition equation

$$s_{t+1} = f(s_t, \Omega_t^i, v_t), \tag{9}$$

where f and v_t represent the state dynamics and randomness in the state transitions, respectively. Since the state equation s_t is composed of two segments, the state dynamics (Eq. 9) can be decomposed as $f(s_t, \Omega_t^i, v_t) = [f^s(s_t, v_t), f^c(\Omega_t^i)]$. All the components of $f^c(\Omega_t^i)$ are 0 but the i^{th} component that corresponds to the selected camera C^i. The specific form of f^s represents the model for the QoV evolution which we approximate with a Gaussian distribution [9] as

$$\rho_t^i = \mathcal{N}(\mu^i, \Sigma^i, \psi_t^i), \tag{10}$$

where μ^i and Σ^i are the mean and the covariance of the Gaussian model $\mathcal{N}$ for C^i.

Please note that the belief state probability $p(s_t|I_t)$, i.e., the probability of being in state s_t, is the posterior probability

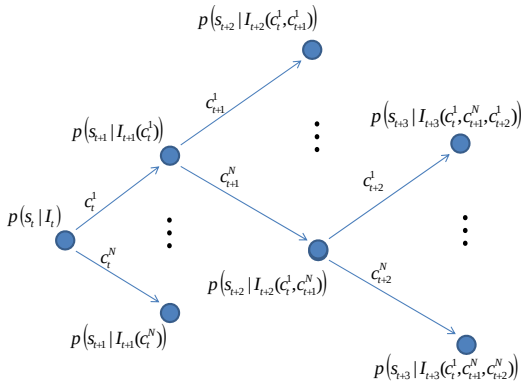


Figure 3: Belief state distribution for three consecutive time steps. Please note that $I_t(c_t^n, c_{t+1}^m)$ signifies the observable history I_t given $c_t^n = 1$ and $c_{t+1}^m = 1$.

distribution of state s_t conditioned on the observable history I_t. Then the *estimated* belief state probability $\bar{s}_{t+1}$, given s_t after selecting camera C^i, and observing ψ_t is given by the Bayes' rule as

$$\bar{s}_{t+1} = \eta\, p(\psi_t|s_t, c_t^i) \sum_{s_t \in S} p(s_t|\psi_t, c_t^i) p(s_t|I_t), \tag{11}$$

where $\eta^{-1} = p(\psi_t|p(s_t|I_t), c_t^i)$ is a normalizing constant.

The next step calculates the optimal value $\mu^*(p(s_t|I_t))$ and the optimal policy π^* that constructs the value to action mapping

$$\pi^* : \mu^*(p(s_t|I_t)) \rightarrow \mathbf{C} \tag{12}$$

These can be estimated using the Bellman equations [1]:

$$\mu^*(p(s_t|I_t)) = \tag{13}$$
$$= \max_{c_t^i \in \mathbf{C}} \left[\begin{array}{l} \sum_{s_t \in S} u(s_t, \Omega_t^i) p(s_t|I_t) + \\ +\gamma \sum_{\psi_t \in \Psi} \left(\begin{array}{l} p(\psi_t|p(s_t|I_t), c_t^i) \\ \mu^*(p(s_{t+1}|I_{t+1})) \end{array} \right) \end{array} \right],$$

where $\gamma \in [0,1]$ is a discount factor and the corresponding optimal policy selects the value-maximizing action as

$$\pi^*(p(s_t|I_t)) = \tag{14}$$
$$= \operatorname*{argmax}_{c_t^i \in \mathbf{C}} \left[\begin{array}{l} \sum_{s_t \in S} u(s_t, \Omega_t^i) p(s_t|I_t) + \\ +\gamma \sum_{\psi_t \in \Psi} \left(\begin{array}{l} p(\psi_t|p(s_t|I_t), c_t^i) \\ \mu^*(p(s_{t+1}|I_{t+1})) \end{array} \right) \end{array} \right].$$

The optimal value function in Eq. 14 or its approximation can be computed using the value iteration algorithm [3]. As demonstrated in [8], the optimal value function μ can be determined within a finite horizon by performing a sequence of value-iteration steps assuming that the sequence of estimates converges to the unique fixed-point solution. To this end we need to rewrite Eq. 11 in the value-function mapping form. Let the real-valued bounded functions μ^* be such that value function mapping H for all information states can be written $\pi^* = H\mu^*$ and the value mapping function H can be written as

$$(H\mu)(p(s_t|I_t)) = \max_{C^i \in \mathbf{C}} h(p(s_t|I_t), \Omega_t^i, \mu_t), \tag{15}$$

where H is an isotone mapping and such that value-functions are estimated per each iteration as:

$$h(p(s_t|I_t), \Omega_t^i, \mu_t) = \tag{16}$$
$$= \begin{array}{l} \sum_{s_t \in S} u(s_t, \Omega_t^i) p(s_t|I_t) + \\ +\gamma \sum_{\psi_t \in \Psi} \sum_{s_t \in S} \left(\begin{array}{l} p(\psi_t|p(s_t|I_t), c_t^i) \\ \mu^* p(s_{t+1}|I_{t+1}) \end{array} \right). \end{array}$$

The error in the belief state is estimated using the error in the estimated and observed belief state

$$g(s_t, \Omega_t^i) = E[\|s_t - \bar{s}_t\|^2] + (1 - u(s_t, \Omega_t^i)). \tag{17}$$

Ideally this should continue until $g(s_t, \Omega_t^i) = 0$. However, in practice, we stop the iteration well before it reaches the limit solution (10^{-5}). Finally, camera selection is performed $\forall\ t$ using the belief-to-action mapping of Eq. 14.

2.2 Quality of View

The Quality of View (QoV) is computed at each time t using information related to the *amount of activity*, the *number of visible objects*, the visible *events* and the *accumulated object score*. The frame score is dependent on the application at hand. While keeping most of the descriptions generic, we will focus in this section on the coverage of team sport events and, in particular, on basketball.

Let d_t^i represent the binary mask for camera C^i encoding the position of the pixels that changed their intensity due to motion. In this implementation we use the color based change detector presented in [17]. The amount of activity, $|d_t^i|$, observed in a view at time t is thus based on the amount of non-zero motion observed in a frame.

The observation vector is $\psi_t = (\psi_t^1, \ldots, \psi_t^N)$, where each ψ_t^i is constructed as

$$\psi_t^i = (J_t^i, |d_t^i|, E_t^i, \Theta_t^i), \tag{18}$$

where J_t^i is the number of objects in the view of a camera C^i, $E_t^i = \sum_{j=1}^{J_t^i} \varepsilon_t^{ij}$ is the sum of individual object scores $\varepsilon_t^{ij} : j = 1, \ldots, J_t^i$ and $\Theta_i(t)$ is the total event score in the view of the i^{th} camera at time t.

The number of objects J_t^i is computed based on a multi-level homography of d_t^i, that projects the moving objects from each view to a common ground plane [7]. The homography is constructed by labeling associated set of points across camera views and on a virtual top view. By construction, all the points from the d_t^i that are labeled as 1 because of the presence of a target in a particular plane project to the corresponding top view position. Each object is assigned an object score ε_t^{ij}, which is indicative of the importance of an object within the scene and it is based on its size, location and proximity to calculate the object score.

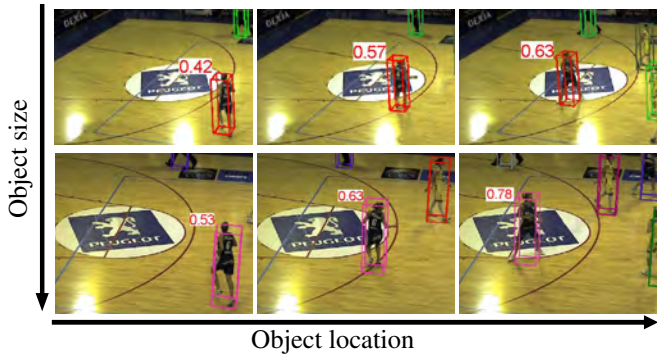


Figure 4: Effect of size and location on ε_t^{ij}: change in an object score when it moves from a region of lower interest to a region of higher interest (left to right); change in object score due to size (top to bottom) .

The *size score* s_t^{ij} of the j^{th} object in the i^{th} camera is calculated as

$$s_t^{ij} = \frac{1}{A^i} \frac{w_t^{ij}.h_t^{ij}}{w_t^{ij} + h_t^{ij}}, \tag{19}$$

where A^i is the imaging area of camera C^i, w_t^{ij} and h_t^{ij} are the width and height of the j^{th} object, respectively, at time t when viewed from C^i.

The *proximity* of the targets to certain objects or areas in the scene is also taken into account for calculating the object score ε_t^{ij}. Authors in [5, 14] consider the distance between the current location of the target being observed and the exit points located in the scene, to estimate the deadline (i.e., the approximate time before the target leaves the scene). Similarly in sports scenarios, the ball is the object of attention and needs to be in the selected view most of the time. Moreover, objects near the ball are of higher interest. The distance of each j^{th} object x_t^{ij} in each camera C^i at time t from the point of interest is calculated and is used as a proximity R_t^{ij}. The lower the value of R_t^{ij}, the more significant the object in the scene.

For generating the *location score*, the site is divided into K non-overlapping regions and each region is assigned a region score $\gamma_k \in [0,1]$, where $\gamma_k \to 1$ represents the region of maximum significance. In basketball scenarios, these could be the regions near the basket (see Fig. 7, where regions of high interest are shown in green). Based on its location in the image plane, each object is assigned a region score γ_t^{ij} at time t.

The object observation vector for the j^{th} object in C^i at time t is then constructed as

$$\psi_t^{ij} = (s_t^{ij}, \gamma_t^{ij}, R_t^{ij}), \tag{20}$$

and the object score ε_t^{ij} is then modeled as a multivariate Gaussian distribution

$$\varepsilon_t^{ij} = \mathcal{N}(\mu_i^o, \Sigma_i^o, \psi_t^{ij}), \tag{21}$$

with mean μ_i^o and covariance Σ_i^o. The motivation for using a multivariate Gaussian as opposed to a linear fusion of features [5] is to normalize each feature individually in its own feature space. Moreover, such fusion of features allows the extension of the feature vector to suit a specific task, for instance, visibility of faces, team information or object associated event score. These local features provide information about the interesting objects inside each camera view. Figure 4 shows the values for ε_t^{ij} for an object according to its distance from the region of interest (marked in green in Fig. 7) in a camera view that observes the object from a distance (top row) as compared to a closeup view of the same object (bottom row) in another camera view. When object moves closer to the region of interest there is an increase in the significance of the object from 0.42 to 0.57 and then to 0.63 (left to right) when the object is approaching the point of interest (the basket). For the camera with a larger version of the same object, ε_t^{ij} goes from 0.53 to 0.63 and then to 0.78. Because of this, the object instance in the bottom-right part of the Fig. 4 will have a higher score (larger size and within the area of interest) and the object instance in the top-left will have the lowest score (smaller size and outside the area of interest).

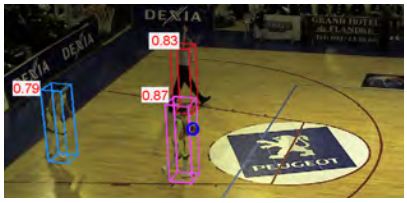


Figure 5: Example of object scores based on the proximity of players to the object of interest (i.e., the ball shown with the blue circle).

The effect of *proximity* is shown in Fig. 5. As mentioned earlier the objects closer to the object of interest are more important than other objects, hence the object closest to the ball (shown with a blue circle) will have higher score (0.87) as compared to other objects (0.83 and 0.79).

The visibility of *events* occurring in the site causes certain views to be more significant than others. Let us assume that there are L possible events which can happen for a multi-camera setup. Based on the significance of each event, it is manually assigned a score θ_l indexed by $l = 1, \ldots, L$. The total event score Θ_t^i for C^i at time t is given as

$$\Theta_t^i = \sum_{l=1}^{L} \theta_l. \tag{22}$$

The events which are not observed by the camera at time t are assigned a 0 score.

In a basketball scenario, *attempt on basket* is identified using motion vectors and the contextual information associated to a view. We consider the region in the vicinity of the basket and when the overall magnitude of vectors in this basket region is larger than a pre-learned threshold, this is considered to be an

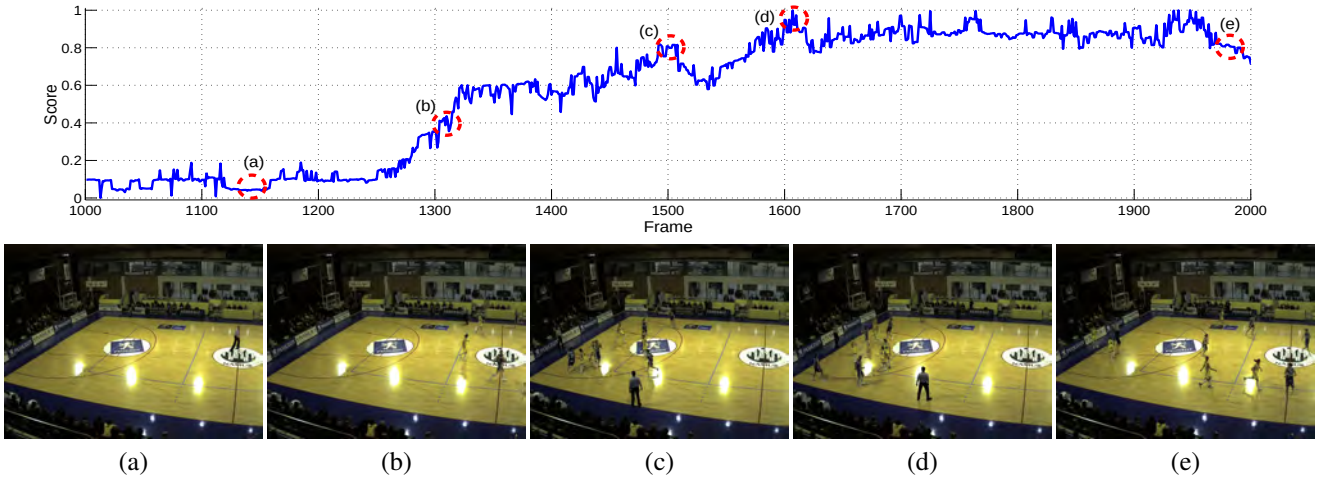


Figure 6: Quality score based on the Gaussian distribution model for C^1 from frame 1000 to frame 2000. Sample images at frame (a)** 1140**, (b)** 1321**, (c)** 1500**, (d)** 1621**, and (e)** 1736**.

event.

A sample output for the QoV score of camera C^1 is shown in Fig. 6. Starting from a near zero score for an almost empty frame (Fig. 6 (a)), as players start entering the view of the camera, the score starts increasing (Fig. 6 (b)) and reaches a higher value when multiple players are within the view (Fig. 6 (c)). The score reaches its maximum when an *attempt-on-basket* event is detected (Fig. 6 (d)). After the *attempt on basket*, players start moving outside the field of view thus leading to a decrease in the score (Fig. 6 (e)).

3 Results

3.1 Experimental setup

We test the performance of the proposed method on a basketball match monitored by 5 cameras with partially overlapping fields of view (Fig. 7). The data consist of approximately 15 minutes (22475 frames at 25 fps) of recording for each camera. We used 500 frames per camera for training the system. The regions of interest are defined as the areas bounded by the *three-point-line* (Fig. 7, shown in green). The value of α was selected to be 0.75. The point of interest is the ball and its location was marked manually. Alternative approaches exist to detect the ball automatically [13], however they work only on portions of a scene and are not reliable for the entire match. The camera selection reference videos (Υ_{gt}) was generated by 11 non-professional users for approximately 4 minutes of the video and the *mode* at each time was taken as the reference ground truth for the selected camera.

To evaluate the effectiveness of the proposed camera selection strategy, here referred to as Υ_{util}, we compare it with three alternative scheduling strategies: the maximum-rank Υ_{max} (selecting the camera with the highest value of ρ_t^i), the recursive decision on a group-of-frames Υ_{gof}, implemented via Dynamic Programming (DP) [10], and the Dynamic

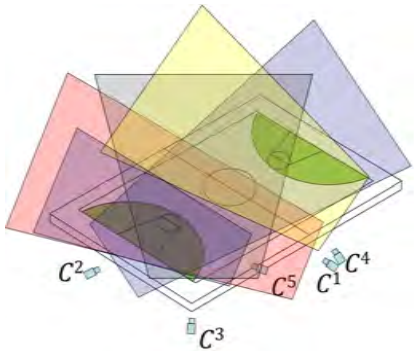


Figure 7: Camera layout with the field of view of each camera C^i highlighted. The regions of high importance are shown in solid (green) color.

Bayesian Network (DBN) based approach Υ_{dbn}. Please note that Υ_{dbn} [6] uses only past information and does not take into account the future (predicted) values, that humans use when watching a video. The automatically generated video using the proposed approach and its comparison of various camera selection strategies can be seen at http://www.eecs.qmul.ac.uk/~andrea/view-selection.html

3.2 Analysis

The performance of each method for camera selection is compared to Υ_{gt}. The overlap between Υ_{gt} and the results of the methods under analysis is shown in Table 1 as a function of the selected features. It can be seen that the choice of features has an impact on the overlap score. For instance, when we use only the number of objects, J_t^i, in the view of each camera (F_1), there is a very small overlap. Such values are due to the camera layout (Fig. 7) as C^2 observes the whole basketball court and thus it is selected most of the time. Here, Υ_{max} outperforms the three other methods that penalize camera switching. In comparison F_2, which only uses the amount of motion $|d_t^i|$, has

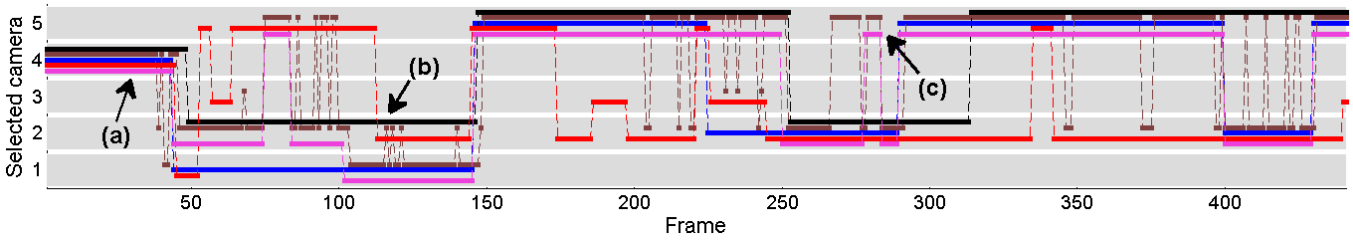


Figure 8: Selection output for different methods for camera selection. (Key. Black: Υ_{gof}; brown: Υ_{max}; blue: Υ_{gt}; red: Υ_{dbn}; pink: Υ_{util})

Table 1: Comparison of camera selection approaches on different feature sets. The numbers represent the % of frames selected by a specific approach on a particular feature vector composition that overlaps with the reference selection Υ_{gt}. (Key. Υ_{max}: maximum-score-based; Υ_{gof} : DP-based [10]; Υ_{dbn}: DBN-based [6]; Υ_{util}: proposed method; $F_1 - F_{15}$: feature vector compositions; J_t^i: Number of objects; $|d_t^i|$: Amount of motion; E_t^i: Accumulated object score; Θ_t^i: Event score)

		F_1	F_2	F_3	F_4	F_5	F_6	F_7	F_8	F_9	F_{10}	F_{11}	F_{12}	F_{13}	F_{14}	F_{15}
Features	J_t^i	✓				✓	✓	✓				✓	✓		✓	✓
	$\|d_t^i\|$		✓			✓			✓	✓		✓	✓	✓		✓
	E_t^i			✓			✓		✓		✓	✓		✓	✓	✓
	Θ_t^i				✓			✓		✓	✓		✓	✓	✓	✓
Methods	Υ_{max}	16.24	68.14	30.65	28.14	83.17	41.38	36.16	73.65	84.17	48.43	81.65	83.17	86.23	49.52	88.13
	Υ_{gof}	4.49	70.17	24.26	17.89	78.13	40.52	27.26	83.72	74.46	47.93	83.72	82.14	74.59	45.97	83.80
	Υ_{dbn}	3.39	68.37	28.14	28.37	80.17	38.39	30.39	79.37	74.37	53.06	88.42	81.34	86.53	48.49	91.35
	Υ_{util}	4.39	70.21	29.46	32.31	81.31	48.23	30.07	78.80	78.29	44.09	88.97	83.85	90.27	51.73	95.42

a larger overlap for all methods as the amount of motion can be treated as an indicator of the significance of the content. However, as discussed earlier, it does not compensate for events that do not considerably affect the motion in the view (for instance the ball being thrown to the basket). Moreover, it is very sensitive to noise due to illumination changes and using it alone is not always appropriate. Similarly, the use of event information Θ_t^i alone, F_4, is not a reliable feature as events may be very sparse in time and can thus lead to decision points that are far apart in time. In F_3, the accumulated object score E_t^i, that includes the object size and location information has generally a larger overlap than F_1. However these features are local, depend on the objects only and do not take into account any event information.

From F_5 to F_{10}, we couple features. When $|d_t^i|$ is used (F_5, F_8 and F_9) a larger overlap is achieved. In comparison, F_7 has the smallest overlap as it only takes into account the number of objects and the event score. These features as described earlier, are either too sparse to be used alone (F_4) or misleading as they would favor the selection of the camera with the maximum number of objects (J_t^i). If we include the amount of motion along with these features as in F_{12}, the overlap for all the methods is significantly increased as compared to F_7 and F_2. However, when they are included with E_t^i (F_{14}), the increase in the overlap percentage is limited. The largest overlap is achieved when all the features are used together (F_{15}), where Υ_{util} has the largest overlap. The percentage overlap for Υ_{gof} is the smallest as it has the minimum number of switches

Table 2: Mean error in the number of switches per second of the automatically generated videos, compared to the ground truth. Note that Δ has no effect on Υ_{gof} as it operates on a temporal window and therefore the mean error 0.024 remains unaffected.

Δ	Method			
	Υ_{max}	Υ_{dbn}	Υ_{gof}	Υ_{util}
1	1.394	0.188	0.024	0.047
5	0.404	0.183	0.024	0.047
10	0.197	0.132	0.024	0.038
15	0.141	0.103	0.024	0.038
20	0.075	0.089	0.024	0.014
25	0.061	0.066	0.024	0.009

but does not always select the best-view (see Fig. 8, black). In comparison Υ_{max}, although presents more switches, still operates around the best view (see Fig. 8, brown).

Figure 9 shows sample results of the proposed method Υ_{util}, the ground truth Υ_{gt}, the state-of-the art methods Υ_{gof} and Υ_{dbn}, and the baseline method Υ_{max}. The three frames (from each camera) are indicated by an arrow in the graph shown in Fig. 8 as (a), (b) and (c). Starting from Fig. 9 (a), most of the players are seen by C^2, C^3 and C^4, whereas C^1 sees only one player and the view from C^5 is empty. However as most of the objects are on the right hand side of the court and when viewed from C^2 and C^3 have relatively smaller sizes, C^4 is selected by all the

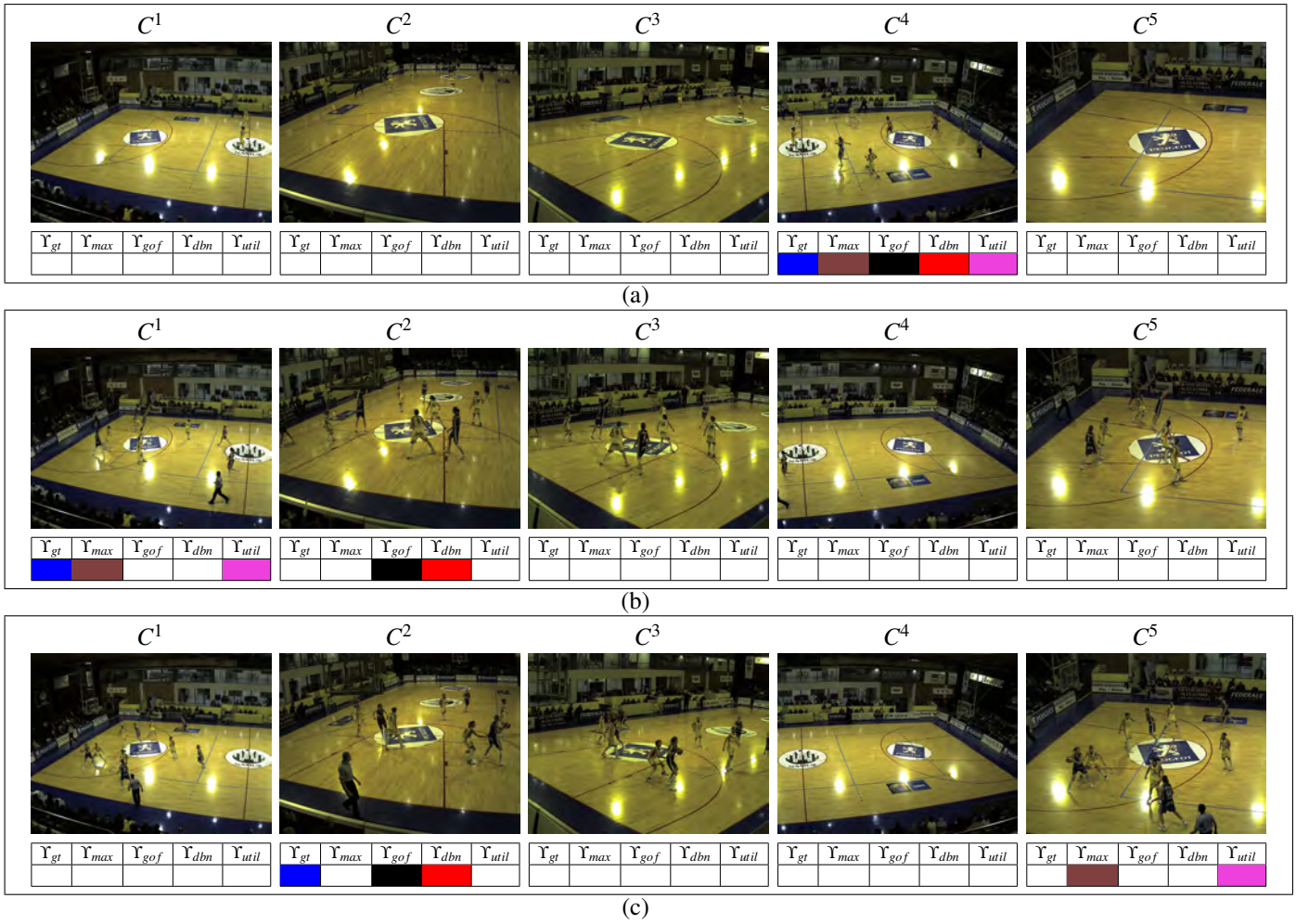


Figure 9: Comparison of selected cameras from different methods for the three time instances annotated as (a), (b) and, (c) in Fig. 8. The same color coding as Fig. 8 is used to highlight the selected camera.

selection methods and Υ_{gt}. When the players start moving from the left to right in Fig. 9 (b), C^1 is selected by Υ_{gt} for it shows a zoomed out version of the left side of the court allowing to see the placement of the players as they move in its field of view. Based on Υ_{max}, C^1 is indeed the best camera as it sees the maximum number of objects at a reasonable size (as compared to C^2). Υ_{util} is able to correctly select this camera, while Υ_{dbn}, bound by the transitions allowed in the adjacency matrix (see [6]) has to switch from C^4 to C^2 and then to C^1, selects C^2. Υ_{gof} selects C^2 as well, as it sees the entire basketball court from the side and has higher accumulation of the object score over time. Finally, in Fig. 9 (c), players have taken up positions in the left hand side of the court leaving C^4 empty. According to the Υ_{gt} the best camera is C^2, which is also selected by Υ_{dbn}. The best camera based on the QoV as selected by Υ_{max} is C^5. Our proposed method selects the best-camera C^5 while Υ_{gof} remains on the same camera C^2.

3.3 Comparison

To evaluate the effectiveness of smoothing introduced by the proposed approach we compare it with the other methods in terms of mean error in the average number of switches per second. In this experiment we introduce the selection interval Δ such that the decision is taken every Δ frames. This results in reducing the number of switches.

Table 2 shows the obtained result where the mean error in the average switches per second for Υ_{max} reduces from 1.394 to 0.061 as τ increases from 1 to 25. In the case of Υ_{util} the error decreases from 0.047 to 0.009 only. For Υ_{dbn} this mean error decreases from 0.188 to 0.066. The scheduling interval has no effect on Υ_{gof} as it operates on a temporal window and the mean error 0.024 remains unaffected. This shows that the proposed approach reduces the number of switches without the need of introducing an additional parameter which may need to be adjusted based on the dynamics of the scene.

Figure 10 shows the improvement achieved via temporal smoothing using the proposed approach from frame 517 to frame 560. Figure 10 (a-e) shows 6 switches between C^1 and C^5 for Υ_{max}. Using Υ_{dbn} and Υ_{util} there is only one switch. However in Υ_{dbn} this switch occurs after 59 frames, whereas in Υ_{util} this switch occurs at the 48^{th} frame. This is due to the fact

Figure 10: Camera selection comparison of the three approaches under analysis for** 2 **seconds of video. Row 1:** Υ_{max}**. Row 2:** Υ_{dbn}**. Row 3:** Υ_{util}**. (Frame numbers: (a) 517, (b) 521, (c) 530, (d) 548, and (e) 560).

that Υ_{util} is able to predict the next state and is able to switch to show the best view before any information is lost. However the ball is passed outside the view of the camera (Fig. 10(d) row 2) when using Υ_{dbn}.

3.4 Subjective testing

To evaluate the goodness of the automatically generated videos we performed a subjective test (a Turing test) using 7 videos of 5000 frames at 25 frames per seconds and 31 subjects. Out of these 7 videos, 3 videos ($M1$ - $M3$) were generated manually by different (non-professional) users, 4 videos were generated by using Υ_{max}, Υ_{gof}, Υ_{dbn} and the proposed method Υ_{util}. The manually generated videos were ranked such that the total number of switches in the end video were increasing ($M1$ (58), $M2$ (63), $M3$ (109)).

Each subject was asked to decide, for each video, *whether it was generated manually (by a human) or automatically (by an algorithm)*. The results of this subjective evaluation are shown in Tab. 3. It is possible to notice that 83.87% of the subjects misidentified the video automatically generated by Υ_{util} as manually generated. None of the subjects selected the video generated by Υ_{max} as manual, whereas 93.55%, 80.65% and 61.29% of the subjects were able to correctly identify $M1$, $M2$ and $M3$, respectively as manually generated video.

3.5 Computational cost

Figure 11 shows the computational cost of the proposed algorithm, broken down for each block outlined in Fig. 1, in terms of relative execution time. The time for each block was calculated on an Intel core i5 3.33 GHz Pentium dual core using a non-optimized serial Matlab implementation. The entire process took on average 0.937 seconds per frame. Multi-layer projections and object detection took 0.567 seconds in total with 44% of the time taken for multi-layer projections and 17% for the object detection module. Change detection (18%) and event detection using motion information (4%) cost on average 0.167 and 0.037 seconds per frame, respectively. The object- and frame-ranking (8% and 2%, respectively) and the camera selection (7%) takes only 0.159 seconds per frame.

4 Conclusions

We presented a technique for automated video production from multiple cameras, which is based on object- and frame-level feature ranking. The proposed approach estimates object visibility scores using a multivariate Gaussian distribution model and employs an optimal control policy for maximizing visibility over time, while minimizing the number of camera switches. The performance of the proposed approach was demonstrated on a multi-camera network with semi-overlapping fields of view from a real basketball match. An overlap of 95.42% with a manually generated ground truth is achieved for the best selected view at any given time. The effectiveness of the proposed approach was also validated through subjective testing with 31 people, of which 26 considered the automatically generated video via the proposed approach as good as a manually generated one.

The proposed camera selection framework can be adapted to work with other feature sets and it is therefore adaptable to different application scenarios. In the future we plan to use automatic ball detection [13] and to use multiple modalities, including audio, for automated video production.

Table 3: Summary of the subjective evaluation results based on the Turing test. "Turing %" represents the percentage of subjects who classifed the video as manually generated.

Method	Classified as manual	Classified as automatic	Turing %
$M1$	29	2	93.55
$M2$	25	6	80.65
$M3$	19	12	61.29
Υ_{max}	0	31	0.00
Υ_{gof}	16	15	51.61
Υ_{dbn}	24	7	77.42
Υ_{util}	26	5	83.87

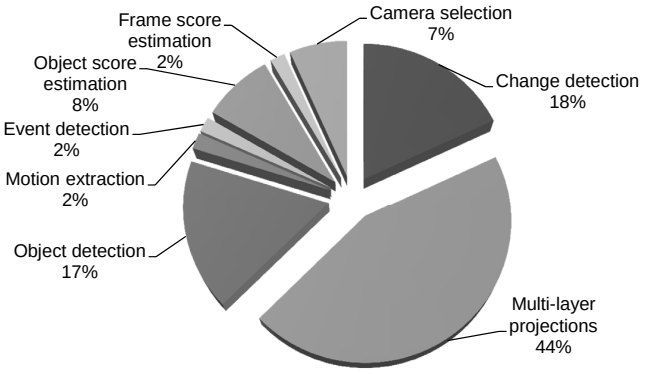


Figure 11: Relative execution time for each module of the proposed video production approach.

References

[1] R. Bellman. *Dynamic Programming*. Princeton University Press, Princeton, 1957.

[2] A. Del Bimbo and F. Pernici. Towards on-line saccade planning for high-resolution image sensing. *Pattern Recognition Letters*, 27(15):1826–1834, Nov. 2006.

[3] A. R. Cassandra. *Exact and approximate algorithms for partially observable Markov decision processes*. PhD thesis, Brown University, 1998.

[4] C. Costello, C. Diehl, A. Banerjee, and H. Fisher. Scheduling an active camera to observe people. In *Proc. of the ACM Int. Workshop on Video surveillance & sensornetworks*, pages 39–45, 15-16 Oct. 2004.

[5] F. Daniyal, M. Taj, and A. Cavallaro. Content-aware ranking of video segments. In *Proc. of ACM/IEEE Int. Conf. on Distributed Smart Cameras*, pages 1–9, 7-11 Sep. 2008.

[6] F. Daniyal, M. Taj, and A. Cavallaro. Content and task-based view selection from multiple video streams. *Multimedia Tools Appl.*, 46(2-3):235–258, Jan. 2010.

[7] D. Delannay, N. Danhier, and C. De Vleeschouwer. Detection and recognition of sports(wo)men from multiple views. In *Proc. of ACM/IEEE Int. Conf. on Distributed Smart Cameras*, pages 1–9, Sep. 2009.

[8] J. Denzler and C.M. Brown. Information theoretic sensor data selection for active object recognition and state estimation. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 24(2):145–157, Feb. 2002.

[9] Y. Fu, Y. Guo, Y. Zhu, F. Liu, C. Song, and Z. Zhou. Multi-view video summarization. *IEEE Trans. on Multimedia*, 12(7):717–729, Nov. 2010.

[10] H. Jiang, S. Fels, and J. J. Little. Optimizing multiple object tracking and best view video synthesis. *IEEE Trans. on Multimedia*, 10(6):997–1012, Oct 2008.

[11] V. Krishnamurthy and D.V. Djonin. Structured threshold policies for dynamic sensor scheduling-a partially observed markov decision process approach. *IEEE Trans. on Signal Processing*, 55(10):4938–4957, Oct. 2007.

[12] S.N. Lim, L.S. Davis, and A. Elgammal. Scalable image-based multi-camera visual surveillance system. In *Proc. of IEEE Int. Conf. on Advanced Video & Signal Based Surveillance*, pages 205–212, Jul. 2003.

[13] F. Poiesi, F. Daniyal, and A. Cavallaro. Detector-less ball localization using context and motion flow analysis. In *Proc. of IEEE Int. Conf. on Image Processing*, pages 3913–3916, 26-29 Sep. 2010.

[14] F. Z. Qureshi and D. Terzopoulos. Surveillance in virtual reality: System design and multi-camera control. In *Proc. of IEEE Int. Conf. on Computer Vision and Pattern Recognition*, pages 1–8, 17-22 Jun. 2007.

[15] M. Rezaeian. Sensor scheduling for optimal observability using estimation entropy. In *IEEE Int. Workshop on Pervasive Computing and Communications*, pages 307–312, 19-23 Mar. 2007.

[16] M. Spaan and P. Lima. A decision-theoretic approach to dynamic sensor selection in camera networks. In *Proc. of Int. Conf. on Automated Planning and Scheduling*, pages 1–8, 19–23 Sep. 2009.

[17] M. Taj, E. Maggio, and A. Cavallaro. Multi-feature graph-based object tracking. In *Proc. of Classification of Events, Activities and Relationships (CLEAR) Workshop*, pages 190–199, Apr. 2006.

[18] H.C. Tijms. *A first course in stochastic models*. John Wiley and Sons, Ltd, England, 2003.

[19] X. Zhou, R. Collins, T. Kanade, and P. Metes. A master-slave system to acquire biometric imagery of humans at distance. In *Proc. of ACM SIGMM Int. Workshop on Video surveillance*, pages 113–120, Nov. 2003.