Video Object Tracking in the Compressed Domain Using Spatio-Temporal Markov Random Fields

Article *in* IEEE Transactions on Image Processing · August 2012

DOI: 10.1109/TIP.2012.2214049 · Source: PubMed

2 authors:

Hossein Khatoonabadi
Simon Fraser University
13 PUBLICATIONS **150** CITATIONS

SEE PROFILE

Ivan V. Bajic
Simon Fraser University
195 PUBLICATIONS **1,572** CITATIONS

SEE PROFILE

Video Object Tracking in the Compressed Domain Using Spatio-Temporal Markov Random Fields

Sayed Hossein Khatoonabadi and Ivan V. Bajić, *Senior Member, IEEE*

***Abstract*— Despite the recent progress in both pixel-domain and compressed-domain video object tracking, the need for a tracking framework with both reasonable accuracy and reasonable complexity still exists. This paper presents a method for tracking moving objects in H.264/AVC-compressed video sequences using a spatio-temporal Markov random field (ST-MRF) model. An ST-MRF model naturally integrates the spatial and temporal aspects of the object's motion. Built upon such a model, the proposed method works in the compressed domain and uses only the motion vectors (MVs) and block coding modes from the compressed bitstream to perform tracking. First, the MVs are preprocessed through intracoded block motion approximation and global motion compensation. At each frame, the decision of whether a particular block belongs to the object being tracked is made with the help of the ST-MRF model, which is updated from frame to frame in order to follow the changes in the object's motion. The proposed method is tested on a number of standard sequences, and the results demonstrate its advantages over some of the recent state-of-the-art methods.**

***Index Terms*— Compressed-domain video object tracking, H.264/AVC, spatio-temporal Markov random field (ST-MRF).**

I. Introduction

VIDEO-BASED object tracking is one of the challenging problems with a variety of applications, such as video surveillance, video indexing and retrieval, video editing, communication, compression, etc. There are two major groups of approaches to segment and track moving objects in a video sequence, distinguished by the domain in which they operate: pixel domain and compressed domain. The former have the potential for higher accuracy, but also require higher computational complexity [1]–[10]. In addition, since most video content nowadays is only available in the compressed form, decoding is required in order to generate pixel-domain information. On the other hand, "compressed-domain" approaches make use of the data from the compressed video bitstream, such as motion vectors (MVs), block coding modes, motion-compensated prediction residuals or their transform coefficients, etc. In practice, some, but not necessarily all, of the information from the bitstream needs to be decoded. In the literature, however, even methods that fully decode the bitstream are sometimes referred to as "compressed-domain" methods, so long as pixel values are not recovered completely for all the frames. The lack of full pixel information often leads to lower accuracy, but the main advantage of compressed-domain methods in practical applications is their generally lower computational cost. This is due to the fact that part of decoding can be avoided, a smaller amount of data needs to be processed compared to pixel-domain methods, and some of the information produced during encoding (e.g., MVs) can be reused. Therefore, compressed-domain methods are thought to be more suitable for real-time applications [11]–[20], although some of them are still characterized by high complexity [21]–[25]. This paper presents a compressed-domain object tracking method that uses only MVs and block coding mode information to perform fast and fairly accurate tracking. The method assumes a video bitstream coded using H.264/AVC [26], but the approach is applicable to older coding standards as well.

Manuscript received January 15, 2012; revised June 14, 2012; accepted July 19, 2012. Date of publication August 17, 2012; date of current version December 20, 2012. This work was supported in part by the Natural Sciences and Engineering Research Council of Canada under Grant RGPIN 327249. The associate editor coordinating the review of this manuscript and approving it for publication was Prof. Gang Hua.

The authors are with the School of Engineering Science, Simon Fraser University, Burnaby, BC V5A 1S6, Canada (e-mail: skhatoon@sfu.ca; ibajic@ensc.sfu.ca).

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.2012.2214049

A. Prior Work on Compressed-Domain Segmentation and Tracking

An iterative scheme that combines Global Motion Estimation (GME) and macroblock (MB) rejection is exploited in [14] to identify moving object blocks, which are then tracked via MB-level tracking. This scheme, however, is not able to segment and track the moving objects whose motion is not sufficiently distinct from the background motion. Käs and Nicolas [15] estimate the trajectories of moving objects from H.264-AVC/SVC MVs. Foreground objects are identified by applying the background subtraction technique followed by temporal filtering to remove the noise. Afterwards, motion segmentation is performed by Timed Motion History Images approach, and finally, the trajectory is estimated by object correspondence processing. Mean shift clustering is used in [13] to segment moving objects from MVs and partition size in H.264 bitstream. After obtaining salient MVs by applying spatial-temporal median filter and Global Motion Compensation (GMC), this method applies spatial-range mean shift to find motion-homogenous regions, and then smoothes the regions by temporal-range mean shift. You *et al.* [24] presented an algorithm to track multiple moving objects in H.264/AVC compressed video based on probabilistic spatiotemporal MB filtering and partial decoding. Their work assumes stationary background and relatively slow-moving objects. Liu *et al.* [11], [12] perform iterative backward projection for accumulating

MVs over time in order to obtain the salient MVs. The spatial-homogenous moving regions are formed by statistical region growing. In [11], the segmented regions are further classified temporally using the block residuals of GMC.

Another line of research [16]–[22] addresses segmentation and tracking problems using Markov Random Field (MRF) models, which provide a meaningful framework for imposing spatial constraints. Treetasanatavorn *et al.* [16] proposed an algorithm for motion segmentation and tracking from MV fields through the Gibbs-Markov random field theory and Bayesian estimation framework. The segmentation of the first frame is carried out by using the stochastic motion coherence model [18]. For the subsequent frames, the algorithm predicts a partition hypothesis by projecting previous partitions to the current frame based on the affine displacement model, and then relaxes the partition labels [17]. The algorithm in [16] also initializes local partition hypothesis and employs the resulting incongruities of two hypotheses to finalize the segmentation. The optimal configuration of this partition is found in the relaxation process so that it minimizes the *maximum a posteriori* (MAP) cost, described by hypothesis incongruity and the motion coherence model. Zeng *et al.* [19] apply MRF classification to segment and track moving objects in H.264/AVC compressed video. First, the MVs are classified into multiple types, such as background, edge, foreground, and noise. In the second stage, moving blocks are extracted by maximizing MAP probability. MRF-based segmentation is also used in [20]–[22]. In these methods, moving regions are coarsely segmented from MV fields prior to boundary refinement, which uses color and edge information. The authors employ GMC and MV quantization to extract a preliminary segmentation map, which is used later in initializing their spatial MRF model.

B. Overview of the Proposed Method

Most existing MRF-based schemes use a spatial (i.e., 2-D) MRF field model to impose spatial constraints. Our proposed spatio-temporal MRF (ST-MRF) takes advantage of one additional dimension by incorporating temporal dependence into the model. The resulting ST-MRF model helps optimize object tracking by referring to motion coherence and spatial compactness, as well as temporal continuity of the object's motion. The proposed method evaluates the labeling dependence of consecutive frames through MVs as well as the similarity of context and neighboring MVs within the frame, and then assigns MVs into the most probable class (object vs. non-object), as measured by the posterior probability. Treetasanatavorn *et al*. [16], [17] also employ temporal dependence in their Bayesian estimation framework to initialize partition hypotheses for subsequent frames. However, unlike [16], [17], where temporal dependence plays a role in projecting previous segmentation result onto the current frame, in our ST-MRF, spatial and temporal dependence are jointly taken into account in the distribution itself.

ST-MRF model has been effectively used in pixel-domain applications such as moving object detection [6], object segmentation [7]–[9], Unmanned Aerial Vehicles (UAV) [10], etc. The characteristics of ST-MRF models vary by application. For example, Yin and Collins [6] used belief propagation and a 3D MRF model with 6-way spatial and temporal neighborhood connectivity to solve the problem of deciding motion or no-motion at each pixel. Here, each hidden state, which is the likelihood of the pixel's motion, is estimated by message passing, where the observed data are binary motion decisions resulting from inter-frame differencing. Some of the methods [7]–[10] define various energy functions for applying spatial and temporal constraints. The spatial energy function is usually defined using pixel-domain information such as intensity, texture, shape, etc., under a Gaussian assumption. In contrast to these pixel-domain methods, the only cue used in our compressed-domain approach is MV information, which usually doesn't follow a Gaussian distribution. As a consequence, we propose a new and efficient spatial energy function that does not rely on Gaussianity and is robust to outliers.

In some cases, in order to incorporate temporal constraints into the model, the states of the positions in the neighboring frames with the same coordinates have been used [6], [9], [10]. However, for moving objects, and even for background with camera motion, the real corresponding position in a neighboring frame is not at the same coordinates as in the current frame. Huang *et al.* [8] compute the optical flow for each node in two directions (between two consecutive frames) and use this correspondence only if the two flows match. Therefore, there is no guarantee that the temporal constraints will be incorporated into all the nodes in the model. In [7], the motion vector field, as well as object segmentation, are estimated according to a Bayesian network and the MAP estimation criterion. Although this work offers good performance, the computational cost is very high. In contrast to these methods, in our model, the corresponding position in the neighboring frame is obtained by making use of the MV information already available in the video bitstream, resulting in much lower complexity.

To the best of our knowledge, the most similar approach to our method is that of Zeng *et al.* [19]. The general ideas are similar, but some important details are different, which leads to different performance as will be illustrated in the results section. One important difference between [19] and the proposed method is the calculation of motion coherence in which [19] first classifies MVs to multiple types under the assumption of static camera and then estimates motion coherence of a block according to the MV classification. On the contrary, our proposed method directly uses the MV observation in computing motion coherence. The common way to compute motion coherence from MV field is based on the assumption of Gaussian distribution noise over MV field; nevertheless, our experience shows that this assumption frequently fails in the compressed domain. In this respect, we propose a novel method, which specifies best the coherence between a MV and a set of MVs belonging to the object, even in the presence of outliers.

The aim of this paper is to provide a framework for tracking moving objects in compressed domain based on MVs and associated block coding modes alone. The object of interest is selected by the user in the first frame, and then tracked through

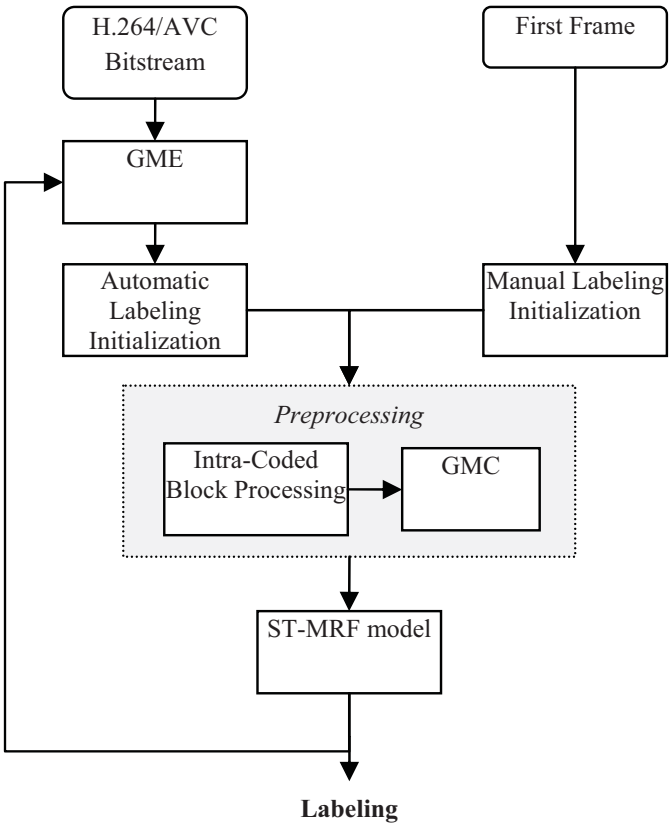

Fig. 1. Flowchart of the proposed ST-MRF-based tracking.

subsequent frames as shown in Fig. 1. In each frame, the proposed method first approximates MVs of intra-coded blocks, estimates global motion (GM) parameters, and removes GM from the MV field. The estimated GM parameters are used to initialize a rough position of the object by projecting its previous position into the current frame. Eventually, the procedure of Iterated Conditional Modes (ICM) updates and refines the predicted position according to spatial and temporal coherence under the MAP criterion, defined by the ST-MRF model.

This work has been performed in a reproducible research manner [27]. The MATLAB implementation of the proposed method and the data used in this study are available online.[1]

The paper is organized as follows. Section II presents the ST-MRF model for compressed-domain video object tracking. Details of preprocessing steps for obtaining a more reliable MV field are discussed in Section III. The results of experimental evaluation carried out on ground truth segmented sequences, and comparisons with other methods, are reported in Section IV. Finally, the conclusions are drawn in Section V.

II. ST-MRF-Based Tracking

A moving rigid object is generally characterized by spatial compactness (i.e., not dispersed across different parts of the frame), relative similarity of motion within the region occupied by the object, and a continuous motion trajectory. Motion of flexible objects is much more difficult to characterize but, at least in principle, these objects can be treated in a divide-and-conquer manner as a collection of smaller sufficiently rigid objects. Therefore, our ST-MRF model is based on rigid object motion characteristics. We treat moving object tracking as a problem of inferring the MAP solution of the ST-MRF model. More specifically, we consider the frame to be divided into small blocks (4×4 in our experiments). Object blocks will be labeled 1, non object blocks 0. We want to infer the block labels $\omega^t \in \{0, 1\}$ in frame t, given the labels ω^{t-1} in frame $t-1$, and the observed motion information $\kappa^t = (\nu^t, o^t)$. Here, the motion information κ^t consists of the MVs from the compressed bitstream, denoted $\nu^t(\mathbf{n})$, and the block coding mode and partition size $o^t(\mathbf{n})$, where $\mathbf{n} = (x, y)$ indicates the position of the block within the frame. The criterion for choosing "the best" labeling ω^t is that it should maximize the posterior probability $P(\omega^t|\omega^{t-1}, \kappa^t)$. This problem is treated in a Bayesian framework, where we express the posterior probability in terms of the inter-frame likelihood $P(\omega^{t-1}|\omega^t, \kappa^t)$, the intra-frame likelihood $P(\kappa^t|\omega^t)$, and the *a priori* probability $P(\omega^t)$ as follows

$$P\left(\omega^t|\omega^{t-1}, \kappa^t\right) = \frac{P\left(\omega^{t-1}|\omega^t, \kappa^t\right) \cdot P\left(\kappa^t|\omega^t\right) \cdot P(\omega^t)}{P(\omega^{t-1}, \kappa^t)}. \quad (1)$$

Since the denominator does not depend on ω^t, the MAP solution for ω^t is found by maximizing the numerator

$$\omega^t = \underset{\psi \in \Omega}{\operatorname{argmax}} \left\{ P\left(\omega^{t-1}|\psi, \kappa^t\right) \cdot P\left(\kappa^t|\psi\right) \cdot P(\psi) \right\} \quad (2)$$

where Ω denotes the set of all possible labeling configurations for frame t. By the monotonicity of the logarithm, the solution to the maximization problem in (2) is the same as the solution to the following minimization problem

$$\omega^t = \underset{\psi \in \Omega}{\operatorname{argmin}} \left\{ -\log P\left(\omega^{t-1}|\psi, \kappa^t\right) - \log P\left(\kappa^t|\psi\right) - \log P(\psi) \right\}. \quad (3)$$

According to the Hammersley-Clifford theorem [28], the probabilities in (3) can be expressed as Gibbs distributions of the form $e^{-E(x)}/Z$ for some energy function $E(x)$ and normalizing constant Z. Hence, we write

$$P\left(\omega^{t-1}|\psi, \kappa^t\right) = \frac{1}{Z_1} \exp\left\{ -\frac{1}{\lambda_\Gamma} E_\Gamma\left(\psi; \omega^{t-1}, \kappa^t\right) \right\} \quad (4)$$

$$P\left(\kappa^t|\psi\right) = \frac{1}{Z_2} \exp\left\{ -\frac{1}{\lambda_\Lambda} E_\Lambda\left(\psi; \kappa^t\right) \right\} \quad (5)$$

$$P(\psi) = \frac{1}{Z_3} \exp\left\{ -\frac{1}{\lambda_\Phi} E_\Phi\left(\psi\right) \right\}. \quad (6)$$

In equations (4)–(6), the three energy functions E_Γ, E_Λ, and E_Φ represent the degree of inconsistency in temporal continuity, spatial context coherence, and compactness, respectively. The parameters λ_Γ, λ_Λ and λ_Φ are scaling constants. In our model, each of the three energy functions E_i is expressed as the summation of the corresponding block-wise energy terms ε_i over the object blocks, so that the optimization problem

[1] http://www.sfu.ca/~ibajic/software.html.

becomes as shown in eq. (7)

$$\omega^t = \underset{\psi\in\Omega}{\operatorname{argmin}} \left\{ \frac{1}{\lambda_\Gamma} \cdot \sum_{\mathbf{n}:\psi(\mathbf{n})=1} \varepsilon_\Gamma\left(\mathbf{n}; \omega^{t-1}, \kappa^t\right) + \frac{1}{\lambda_\Lambda} \cdot \sum_{\mathbf{n}:\psi(\mathbf{n})=1} \varepsilon_\Lambda\left(\mathbf{n}; \kappa^t\right) + \frac{1}{\lambda_\Phi} \cdot \sum_{\mathbf{n}:\psi(\mathbf{n})=1} \varepsilon_\Phi(\mathbf{n}) \right\}. \tag{7}$$

The first term measures the temporal discontinuity of labeling between consecutive frames - the larger the difference between the labeling ω^{t-1} and the backwards-projected candidate labeling ψ, the larger this term will be. The second term represents spatial incoherence among object's MVs - the larger the difference among the MVs within the region labeled as the object under the candidate labeling ψ, the larger this term will be. The compactness of object's shape is accounted for in the last term. All three terms will be defined more precisely in the following sections. Finally, the minimization problem (7) will be solved by the method of ICM [29].

A. Temporal Continuity

Temporal continuity is measured by the overlap between the labeling of the previous frame, ω^{t-1}, and the backwards-projected candidate labeling ψ for the current frame. Consider a block $\mathbf{n}$ in the current frame that is assigned to the object by the candidate labeling ψ, i.e. $\psi(\mathbf{n}) = 1$. The block is projected backwards into the previous frame along its MV, $v^t(\mathbf{n}) = (v_x(\mathbf{n}), v_y(\mathbf{n}))$, and degree of overlap $D(\mathbf{n} + v^t(\mathbf{n}); \omega^{t-1})$ is computed as the fraction of pixels within the projected block in the previous frame that carry the label 1 under the labeling ω^{t-1}. The energy term for block $\mathbf{n}$ is taken to be

$$\varepsilon_\Gamma\left(\mathbf{n}; \omega^{t-1}, \kappa^t\right) = -D\left(\mathbf{n} + v^t(\mathbf{n}); \omega^{t-1}\right). \tag{8}$$

Our experiments suggest that temporal continuity is a powerful cue for object tracking, and by itself is able to provide relatively accurate tracking in many cases. It is also relatively simple to compute. However, its performance may suffer due to noisy or inaccurate MVs, especially near object boundaries. Hence, a robust tracking algorithm should not rely exclusively on temporal continuity, and should incorporate some spatial properties of the MV field. Two such concepts, corresponding to the second and third terms in (7), are discussed next.

B. Context Coherence

One of the characteristics of rigid object motion in natural video is the relative consistency (coherence) of the MVs belonging to the object. This is true even if the motion is not pure translation, so long as the frame rate is not too low. Such motion coherence has frequently been used in compressed-domain segmentation [11], [12], [16]–[22]. A popular approach [20]–[22] is to model the MVs within an object with an independent bivariate Gaussian distribution whose parameters are estimated from the decoded MVs. Once the parameter estimates are obtained, one can adjust the segmentation by testing how well the MVs fit into the assumed model. A particular problem with parameter estimation in this context is the presence of outliers - incorrect or noisy MVs that are often found in flat-texture regions or near object boundaries. For small objects, even a few outliers can lead to large estimation errors. Sample variance is especially sensitive to outliers [30]. To resolve the problem we employ robust statistics methods [30], which tend to be more resistant to the effect of outliers compared to classical statistics.

A number of robust statistics methods have been proposed to estimate central tendency of the data in the presence of outliers, e.g., Median, Trimmed Mean, Median Absolute Deviation, and Inter Quartile Range [30]–[32]. We have tested these methods in the context of our problem, and finally settled on a Modified Trimmed Mean method, as it gave the best results. The main difference between our proposed Modified Trimmed Mean and the conventional Trimmed Mean [30] is that instead of truncating a fixed percentage of the data set from one or both ends of the distribution, in our method the outliers are adaptively recognized and truncated, as explained below.

For the purpose of MV coherence analysis, the object's motion is represented by a single representative MV $\hat{\mathbf{v}}$. This vector is computed based on the Polar Vector Median (cf. Section III-A) of MVs that are assigned to the object by the candidate labeling ψ, i.e. $\psi(\mathbf{n}) = 1$. At this stage, we are using preprocessed MVs, $\mathbf{v}'(\mathbf{n})$ (c.f section III). The deviation of a MV of the block $\mathbf{n}$ from the object's representative vector $\hat{\mathbf{v}}$ is computed as the Euclidian distance between $\mathbf{v}'(\mathbf{n})$ and $\hat{\mathbf{v}}$:

$$d(\mathbf{n}) = \left\|\mathbf{v}'(\mathbf{n}) - \hat{\mathbf{v}}\right\|_2. \tag{9}$$

It is observed that the deviation $d(\mathbf{n})$ for the blocks belonging to the object can be modeled reasonably well by the rectified (non-negative) Gaussian distribution, except for the outliers. Thus, we identify the outliers in the data by checking whether $d(\mathbf{n}) > \theta$, where θ is computed as

$$\theta = \max(2 \cdot \sigma_d, 1) \tag{10}$$

and σ_d is the square root of the second moment of $d(\mathbf{n})$. The distribution of $d(\mathbf{n})$ for the rotating ball object in frame #4 of the *Mobile Calendar* sequence is shown in Fig. 2(a). In this example, the candidate labeling ψ is initially predicted by projecting the previous frame labeling, i.e. ω^3, via current GM parameters. Two MVs with $d(\mathbf{n}) \cong 24.7$ and one with $d(\mathbf{n}) \cong 29$ are recognized as the outliers based on the threshold defined in (10). The fitted rectified Gaussian distributions before and after outlier removal are also shown. It can be seen that outliers significantly increase the sample variance and that after their removal, the rectified Gaussian can be used as a reasonable model for $d(\mathbf{n})$.

After removing the outliers, the sample standard deviation σ_d is recalculated. In certain cases, when the object's MVs are close to identical, the recalculated σ_d is close to zero. We clip σ_d from below to 0.5 in order to avoid problems with subsequent computations. After that, the MV deviation is normalized to the interval $[-1, 1]$ as follows

$$d'(\mathbf{n}) = \min\left\{\frac{d(\mathbf{n})/\sigma_d - 2}{2}, 1\right\}. \tag{11}$$

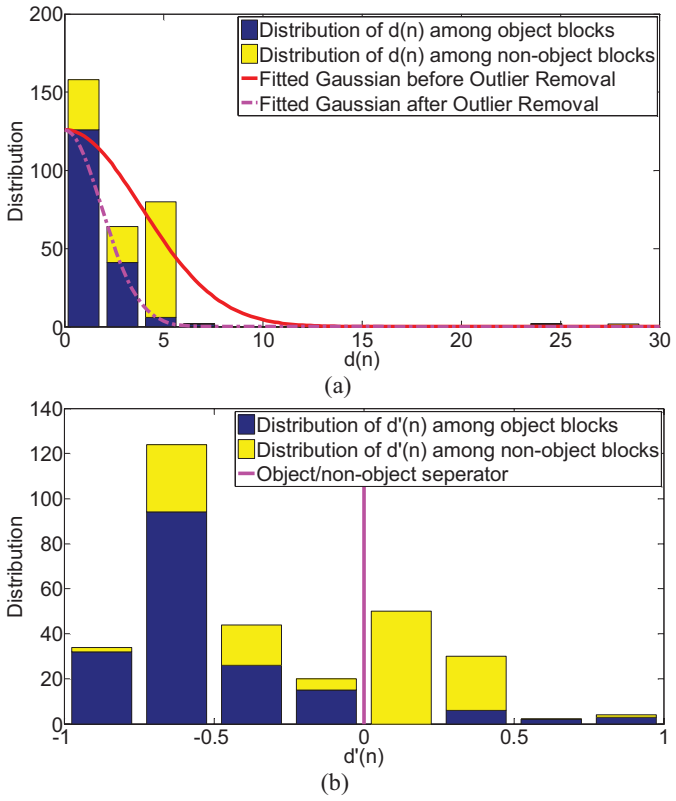

Fig. 2. Distribution of $d(\mathbf{n})$ and $d'(\mathbf{n})$ for the ball in frame #4 of the *Mobile Calendar* sequence.

The higher the value of the normalized deviation $d'(\mathbf{n})$, the less similar is $\mathbf{v}'(\mathbf{n})$ to $\hat{\mathbf{v}}$, so the less likely is block $\mathbf{n}$ to belong to the object, as far as MV similarity goes. Fig. 2(b) shows the distribution of $d'(\mathbf{n})$.

The effect of GM is taken into account by dividing $d'(\mathbf{n})$ with a GM parameter ρ defined as

$$\begin{aligned} \rho &= 2 - \exp\left(-c_1 \cdot (\mathrm{A} + \mathrm{B})\right) \\ \mathrm{A} &= c_2 \cdot (|m_1| + |m_4|)^{c_3} \\ \mathrm{B} &= |1 - m_2| + |m_3| + |m_5| + |1 - m_6| \end{aligned} \tag{12}$$

where $m_i, i = 1, 2, \ldots, 6$, are the six affine GM parameters (c.f. Section III-B) and c_1, c_2 and c_3 are constant values defined in Section IV. The logic behind eq. (12) is as follows: the larger the camera motion, the closer the value of ρ is to 2; the smaller the camera motion, the closer its value is to 1. Parameters m_1 and m_4 represent translation (value 0 means no translation), m_2 and m_6 represent zoom (value 1 means no zoom), and m_3 and m_5 represent rotation (value 0 means no rotation). Hence, as any component of the affine motion increases, the exponent in (12) becomes more negative, and the value of ρ gets closer to 2, which is the highest it can be. In the case of fast camera motion, MVs are less reliable, and the influence of context coherence on tracking decisions should be reduced. Hence, the block-wise context coherence energy term in (7) is computed by dividing the normalized MV deviation in (11) by ρ, that is

$$\varepsilon_{\Lambda}\left(\mathbf{n}; \kappa^t\right) = d'\left(\mathbf{n}\right)/\rho. \tag{13}$$

C. Compactness

While there are certainly counterexamples to this observation, most rigid objects in natural video tend to have compact shape, meaning that the chance of a block belonging to the object is increased if many of its neighbors are known to belong to the object. We take this observation into account through the last term in eq. (7). We employ the 8-adjacency neighborhood with different weights for the first-order and second-order neighbors. The block-wise energy term for compactness is computed by the weighted sum of labels in the neighborhood of the current block:

$$\varepsilon_{\Phi}(\mathbf{n}) = -\alpha \sum_{\mathbf{k} \in Y_{+}(\mathbf{n})} \psi(\mathbf{k}) - \beta \sum_{\mathbf{k} \in Y_{\times}(\mathbf{n})} \psi(\mathbf{k}) \tag{14}$$

where $\Upsilon_{+}(\mathbf{n})$ and $\Upsilon_{\times}(\mathbf{n})$ are, respectively, the first-order (North, South, East, and West) and the second-order (North-East, North-West, South-East, and South-West) neighborhoods of block $\mathbf{n}$. Recall from (7) that $\psi(\mathbf{n}) = 1$. A neighboring block with label 0 will not change the value of (14), while a neighboring block with label 1 will make (14) more negative. Hence, the higher the number of neighboring blocks that also belong to the object (label 1), the lower the value of the energy term in (14), meaning that the more likely it is that block $\mathbf{n}$ also belongs to the object. We set $\alpha = 1/6$ and $\beta = 1/12$ in our experiments to give higher weight to first-order neighbors.

D. ST-MRF Optimization

Now that each term in (7) is defined, the optimization problem needs to be solved. Two popular algorithms are often used for this purpose: Stochastic Relaxation (SR) [33], [34] and ICM [29]. SR has been reported to have some advantage in accuracy compared to ICM, but at a higher computational cost [35]. In this work we use ICM to solve (7), mainly due to its simplicity. At the beginning, the label of each block is initialized by projecting the previous frame labeling ω^{t-1} into the current frame using the current GM parameters. After that, each block is relabeled with the label (0 or 1) that leads to the largest reduction in the energy function. This relabeling procedure is iterated until no further energy reduction is achieved. Usually, six iterations are enough to reach the local minimum. It is worth mentioning that because the ICM procedure is prone to being trapped in a local minimum, results are dependent on the initial labeling.

III. Preprocessing

Our tracking algorithm makes use of two types of information from the H.264/AVC-compressed bitstream: block coding mode (partition) information and MVs. Texture data does not need to be decoded in the proposed method. H.264/AVC defines four basic MB modes: 16 × 16, 16 × 8, 8 × 16, and 8 × 8, where the 8 × 8 mode can be further split into 8 × 4, 4 × 8, and 4 × 4 modes. Since the smallest coding mode (partition) in H.264/AVC is 4 × 4, in order to have a uniformly sampled MV field, we map all MVs to 4 × 4 blocks. This is straightforward in inter-coded blocks, as well as SKIP blocks where the MV is simply set to zero. However, interpreting the motion in the intra-coded blocks is more involved.

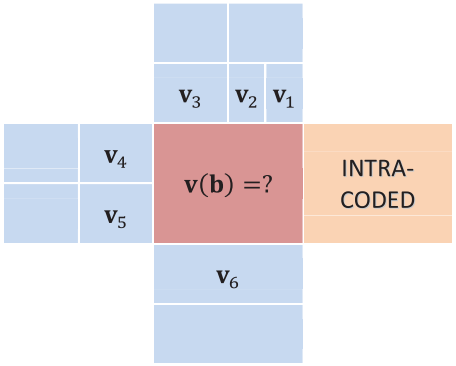

Fig. 3. MV assignment for an intracoded MB. One of the first-order neighboring MBs is also intracoded, and the remaining neighbors have MVs assigned to variably sized blocks.

In this section we describe two preprocessing steps that are employed before the actual ST-MRF optimization discussed in the previous section. These steps are the management of intra-coded blocks and eliminating GM.

A. Polar Vector Median for Intracoded Blocks

Intra-coded blocks have no associated MVs. However, for the purpose of running the ST-MRF optimization in the previous section, it is useful to assign MVs to these blocks. We propose to do this based on the neighboring MVs using a new method called Polar Vector Median (PVM). For this purpose, we employ MVs of the first-order neighboring MBs (North, West, South, and East) that are not intra-coded. Fig. 3 shows a sample intra-coded MB along with its first-order neighboring MBs. In this example, the goal is to find $\mathbf{v}(\mathbf{b})$ for all blocks $\mathbf{b}$ in the intra-coded MB. We collect MVs of the 4×4 blocks from the neighboring MBs that are closest to the current intra-coded MB and store them in the list V. For this example, the list of MVs is

$$V = (\mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_3, \mathbf{v}_3, \mathbf{v}_4, \mathbf{v}_4, \mathbf{v}_5, \mathbf{v}_5, \mathbf{v}_6, \mathbf{v}_6, \mathbf{v}_6, \mathbf{v}_6)\,.$$

Note that $\mathbf{v}_6$ appears four times in the list, because it is assigned to four 4×4 blocks along the southern boundary of the intra-coded MB. For the same reason, $\mathbf{v}_3$, $\mathbf{v}_4$, and $\mathbf{v}_5$ appear twice, while $\mathbf{v}_1$ and $\mathbf{v}_2$ appear only once in the list. The list will contain at most 16 vectors, which happens when all first-order neighboring MBs are inter-coded. In the above example, one of the neighboring MBs is intra-coded, so the list contains only 12 vectors.

The next step is to assign a representative vector from this collection of vectors. The standard vector median [36] would be the vector from the list with the minimum total distance to other vectors in the list. The existence of outliers, however, has an adverse effect on the vector median. We therefore propose another method called PVM, which has proved less cost demanding and more robust in our study.

In PVM, the representative vector is computed in polar coordinates as follows. Let $V = (\mathbf{v}_i)_{i=1:n}$ be the list of n input vectors, sorted according to their angle (in radians) from $-\pi$ to $+\pi$. Then, a collection of $m = \lfloor (n+1)/2 \rfloor$ vectors is selected as $\hat{V} = (\mathbf{v}_i)_{i=I:I+m-1}$, where index I is found as

$$I = \underset{j}{\operatorname{argmin}} \sum_{i=j}^{j+m-2} \theta_i \tag{15}$$

and θ_i denotes the angle between vectors $\mathbf{v}_i$ and $\mathbf{v}_{i+1}$ (let $\mathbf{v}_1 \equiv \mathbf{v}_{n+1}$). The new list $\hat{V}$ contains approximately half the original number of vectors in V chosen such that the sum of the angles between them is minimum. Hence, they are clustered in a narrow beam. The PVM $\hat{\mathbf{v}}$ is constructed as follows: its angle is chosen to be the median of angles of the vectors in $\hat{V}$, while its magnitude is set to the median of magnitudes of the vectors in V, that is

$$\angle\hat{\mathbf{v}} = \text{median } (\angle\mathbf{v}_i)_{i=I:I+m-1} \tag{16}$$

$$\|\hat{\mathbf{v}}\|_2 = \text{median } (\|\mathbf{v}_i\|_2)_{i=1:n} \tag{17}$$

Fig. 4 shows an example of computing the PVM. It should be mentioned that zero-vectors are excluded from angle calculations, since their angle is indeterminate. Once PVM $\hat{\mathbf{v}}$ is computed, it is assigned to all 4×4 blocks within the intra-coded MB (e.g., $\mathbf{v}(\mathbf{b})$ in Fig. 3).

Fig. 5 shows the effect of assigning PVM of neighboring blocks to intra-coded blocks on frame #35 of the *Hall Monitor* sequence. The intra-coded blocks are indicated in Fig. 5(a), the labeling with zero MV assignment to intra-coded MBs is shown in Fig. 5(b), while the labeling with PVM assignment to intra-coded MBs is shown in Fig. 5(c). When the block labeling is computed as discussed in Section II, all pixels in a given block are assigned the label of that block. By comparing with the manually segmented ground truth[2], one can identify correctly labeled object pixels (true positives - TP), non-object pixels incorrectly labeled as object pixels (false positives - FP), and missed object pixels that are labeled as non-object pixels (false negatives - FN). The numbers are TP = 3340, FP = 580, FN = 199 for Fig. 5(c), where PVM is used, against TP = 3313, FP = 591, FN = 226 in Fig. 5(b), where zero-vector is used instead of the PVM. It is easy to see that detection is improved by PVM assignment around the man's feet in the bottom parts of Fig. 5(b) and Fig. 5(c).

B. Global Motion Compensation

Global motion (GM), caused by camera movement, affects all pixels in the frame. Since GM adds to the object's native motion, for accurate tracking, it is important to remove GM from the MV field prior to further processing. In this work we use the 6-parameter affine model [37] to represent GM. Although less flexible than the 8-parameter perspective model, the affine model has been widely used in the literature to represent GM, in part because there are fewer parameters to be estimated, which often leads to higher estimation accuracy. Given the parameters of the affine model, $\mathbf{m} = [m_1, \ldots, m_6]$, a block centered at (x, y) in the current frame will be transformed to a quadrangle centered at (x', y') in the reference frame, where (x', y') is given by

$$x' = m_1 + m_2 x + m_3 y, \qquad y' = m_4 + m_5 x + m_6 y \tag{18}$$

[2]http://www.sfu.ca/~ibajic/datasets.html.

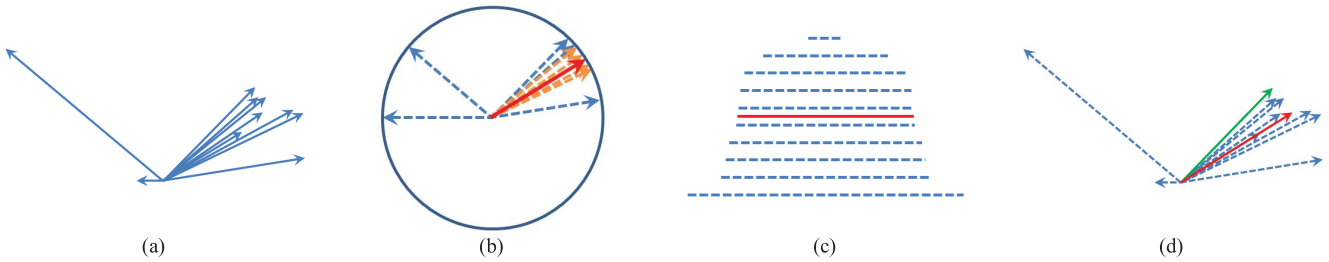

Fig. 4. Polar vector median. (a) Original vectors. (b) Angles of vectors. Dotted brown vectors: candidate vectors for computing representative angle. Red vector: representative angle. (c) Lengths of vectors. Red line: representative length. (d) Result of polar vector median. Green vector: standard vector median [36]. Red vector: polar vector median.

The MV due to affine motion is given by

$$\mathbf{v}(x, y) = \left(x' - x, y' - y\right) \tag{19}$$

To estimate GM parameters $\mathbf{m}$, we use a modified version of the M-Estimator introduced in [38] by Arvanitidou *et al.*, which is an extension of [39]. This method reduces the influence of outliers in a re-weighted iteration procedure based on the estimation error obtained using least squares estimation. This iterative procedure continues until convergence. Chen *et al*. [21] show that the performance of this approach depends on the tuning constant in weighting factor calculation. In [38], the weighting factor $w(\mathbf{n})$ for block $\mathbf{n}$, which imposes the strength of outlier suppression, is calculated as

$$w(\mathbf{n}) = \begin{cases} \left(1 - \frac{\varepsilon^2(\mathbf{n})}{c^2 \mu_\varepsilon^2}\right)^2 & \varepsilon(\mathbf{n}) < c\mu_\varepsilon \\ 0 & \varepsilon(\mathbf{n}) \geq c\mu_\varepsilon \end{cases} \tag{20}$$

where c is the tuning constant, $\varepsilon(\mathbf{n})$ is the estimation error calculated as the Manhattan norm between the observed MV and the MV obtained from the estimated model (18)–(19), and μ_ε is the average estimation error over all the blocks in the frame. In our slightly modified approach, to obtain the decision boundary for outlier suppression (which is $c\mu_\varepsilon$ in (20)), instead of using all the blocks in the frame, we choose only those blocks whose MVs are not currently declared as outliers. In particular, the weighting factor is calculated iteratively as

$$w(\mathbf{n}) = \begin{cases} \left(1 - \frac{\varepsilon^2(\mathbf{n})}{\left(\mu_{\varepsilon'} + 2\sigma_{\varepsilon'}\right)^2}\right)^2 & \varepsilon(\mathbf{n}) < \mu_{\varepsilon'} + 2\sigma_{\varepsilon'} \\ 0 & \varepsilon(\mathbf{n}) \geq \mu_{\varepsilon'} + 2\sigma_{\varepsilon'} \end{cases} \tag{21}$$

where ε' is the set of estimation errors for blocks with a non-zero weighting factor, while $\mu_{\varepsilon'}$ and $\sigma_{\varepsilon'}$ are, respectively, the average value and the standard deviation of the set ε'. Once the weighting factor for a block becomes zero, it will be discarded from the set ε' in the following iterations. In this approach, the decision boundary for outlier suppression uses both the average value and standard deviation of estimation errors over the set ε', which is more robust in comparison to the conventional approach where only the average value of estimation errors over all blocks is used.

Arvanitidou *et al.* [38] observe that large blocks (8 × 8 and above) are more likely to be part of the background, whereas small blocks, arising from splitting in the motion estimation procedure at the encoder, are more likely to belong to the moving objects. Hence, only large blocks (16 × 16, 16 × 8, 8 × 16, 8 × 8) are used in GME, while small blocks (8 × 4, 4 × 8, 4 × 4) and intra-coded blocks are discarded from this process. In our approach, for the purpose of GME, we also discard the MVs from the region that was occupied by the object in the previous frame. To get fast convergence to a stable solution and escape from being trapped in many local minima, we initialize the weighting factor for block $\mathbf{n}$ based on its dissimilarity to neighboring blocks, denoted by $\Delta(\mathbf{n})$, which is computed as the median of Manhattan differences between the MV of block $\mathbf{n}$ and MVs of its neighbors. Therefore, the initial weight for block $\mathbf{n}$ is computed by

$$w(\mathbf{n}) = \exp\left(-\Delta(\mathbf{n})\right). \tag{22}$$

Fig. 5. Effect of assigning PVM to intracoded blocks. (a) Intracoded blocks indicated by yellow squares. (b) Tracking result without PVM assignment. (c) Tracking result with PVM assignment. True positives (TPs) are shown as green, false positives (FPs) as blue, and false negatives (FNs) as red.

TABLE I
PARAMETER VALUES USED IN OUR EXPERIMENTS

Parameter	λ_Φ	λ_Λ	λ_Γ	c_1	c_2	c_3
Value	1	2/3	0.25	1/128	1	2

IV. RESULTS

A number of standard test sequences were used to evaluate the performance of our proposed approach. Sequences were in the YUV 4:2:0 format, at two resolutions, CIF (352 × 288 pixels) and SIF (352 × 240 pixels), all at the frame rate of 30 fps. All sequences were encoded using the H.264/AVC JM v.18.0 encoder [40], at various bitrates, with the GOP structure IPPP, i.e., the first frame is coded as intra (I), and the subsequent frames are coded predictively (P). Motion and partition information were extracted from the compressed

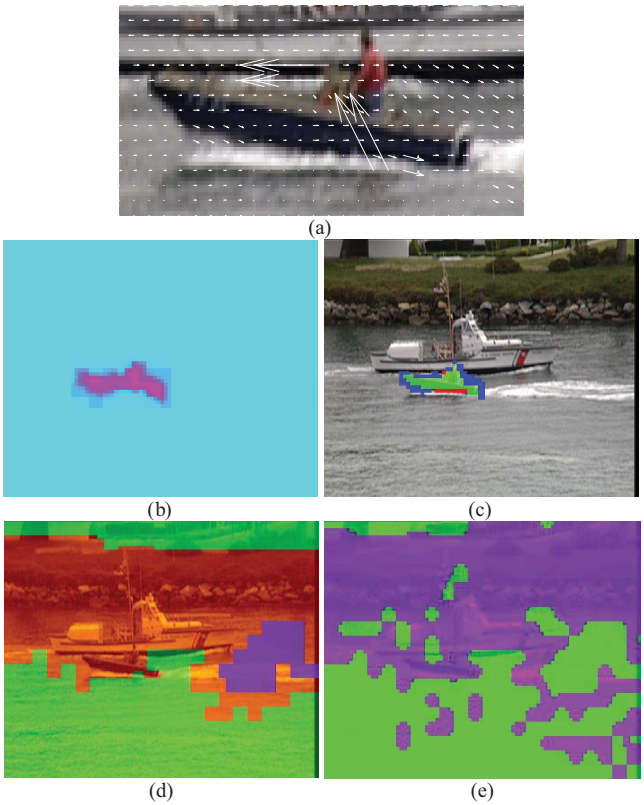

Fig. 6. Object detection during ST-MRF-based tracking. (a) Target at frame #70 of *Coastguard* superimposed by scaled MV field after GMC. (b) MRF energy value—the darker the color, the higher the energy. (c) Tracking results by the proposed method. (d) and (e) Segmentation result from [21] and [19], respectively.

bitstream, and MVs were remapped to 4×4 blocks, as explained in Section III.

Some of the characteristics of the proposed approach are its robustness and stability. To show this, we use the *same* parameters throughout all the experiments, as listed in Table I. We found that the average performance does not change much if some of these parameter values are changed, especially the parameters c_1, c_2 and c_3 that represent the effect of camera motion on energy function, and only affect a few frames. Fig. 6 illustrates a few intermediate results from the tracking process for a sample frame from *Coastguard*. As seen from Fig. 6(a), the MV field around the target (small boat) is somewhat erratic, due to fast camera motion in this part of the sequence. The proposed ST-MRF-based tracking algorithm computes the energy function for the chosen MRF model, which is shown in Fig. 6(b). Here, the darker the value, the smaller the energy, hence the higher the posterior probability. Therefore, despite the erratic MV field, the target seems to be localized reasonably well. Fig. 6(c) shows the detected target region after the tracking process has been completed. Pixels shaded with different colors indicate TP, FP and FN, as explained in the caption of Fig. 5. As seen here, some erroneous decisions are made around the boundary of the target object, but the object interior is detected well. For comparison purposes, the segmentation result from two other methods, [21] and [19], are illustrated in Fig. 6(d) and Fig. 6(e), respectively. Clearly, the proposed method has produced a much better result compared

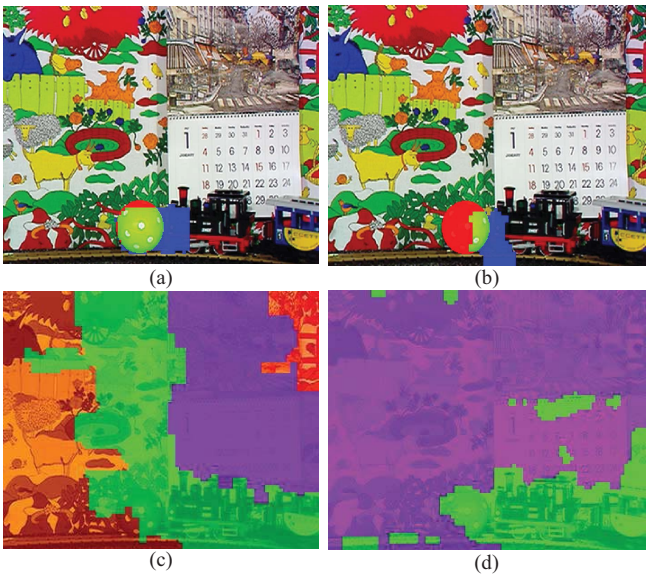

Fig. 7. Object detection/tracking by (a) proposed method and (b) method from [11] for frame #97 of *Mobile Calendar*. (c) and (d) Segmentation results from [21] and [19], respectively.

to these two methods in the face of sudden and fast camera movement.

In Fig. 7 our proposed method is compared with the methods of Liu *et al.* [11], Chen *et al.* [21], and Zeng *et al.* [19] in terms of visual segmentation results on frame #97 of *Mobile Calendar*. The tracking target is the ball. In frame #97, which corresponds to the moment when the ball touches the train and changes its direction, the blocks within the ball and the train have almost equal MVs. This may cause confusion in the tracking or segmentation task. As a consequence, the proposed method declares some parts of the train as the ball; nonetheless, the ball itself is detected correctly (Fig. 7(a)). The method from [11], on the other hand, misses a large part of the ball (Fig. 7(b)). It only detects a small part of the ball and misclassifies some parts of the train and the background as the target. Segmentation results of the methods from [21] (Fig. 7(c)) and [19] (Fig. 7(d)) methods show that the ball is not separated from the train, which is not surprising, since these methods rely on spatial segmentation of MVs. Another example that illustrates the robustness of the proposed method is the *Coastguard* sequence. Here, our method is able to track the small boat throughout the sequence, even during the fast camera movement, as shown by the trajectory in Fig. 11(b). By comparison, none of the other three methods were able to segment and/or track the small boat in the frames #68 to #76, where the camera motion overwhelms the object motion.

In Fig. 8 we compare the proposed method with the methods of [11], [19], and [21] in terms of *Precision, Recall*, and *F-measure* on *Mobile Calendar*, where the tracking target is again the ball. *Precision* is defined as the number of TP divided by the total number of labeled pixels, i.e., the sum of TP and FP (eq. (23)). *Recall* is defined as the number of TP divided by the total number of ground truth labels, i.e., the sum of TP and FN (eq. (24)). F-measure is the harmonic mean of

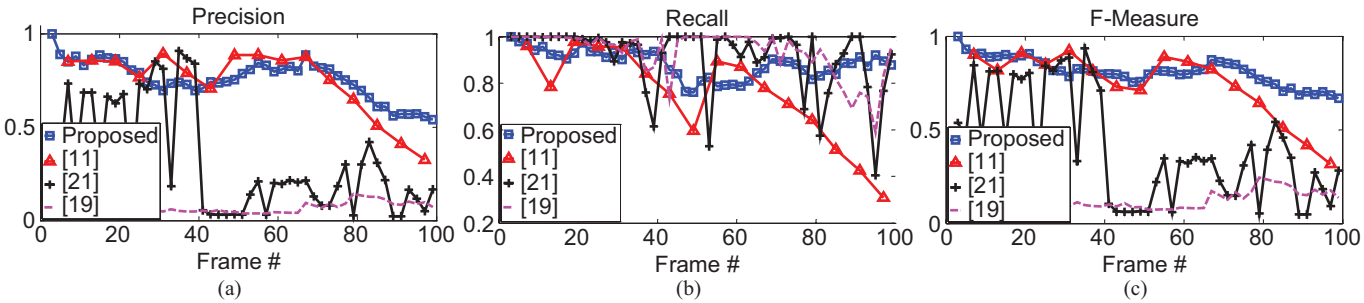

Fig. 8. Comparison of several methods in terms of (a) *Precision*, (b) *Recall*, and (c) *F-measure* for *Mobile Calendar*, where the tracking target is the ball.

TABLE II

COMPARISON OF SEVERAL METHODS IN TERMS OF THE AVERAGE *Precision*, *Recall*, AND *F-Measure* (IN PERCENTAGE) FOR THE *Mobile Calendar* SEQUENCE, WHERE THE TRACKING TARGET IS THE BALL

Method	*Precision*	*Recall*	*F-Measure*
Proposed	75.9	88.4	81.2
[11]	74.1	74.8	74.1
[21]	31.8	91.6	40.5
[19]	6.6	93.3	12.1

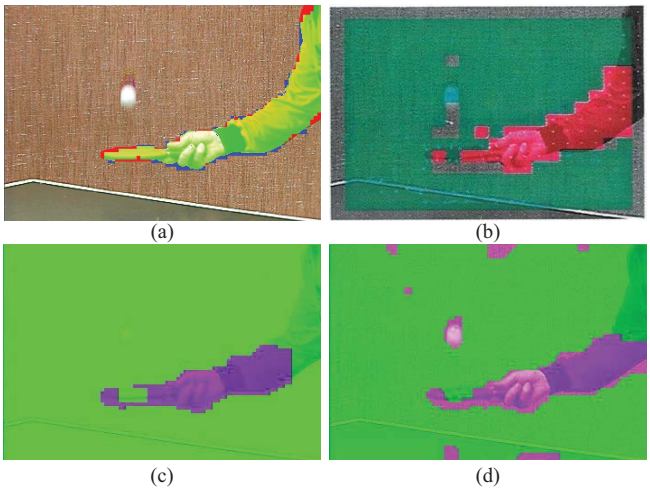

Fig. 9. Object detection and tracking by (a) proposed method for frame #10 of *Table Tennis*. (b)–(d) Segmentation results from [17], [21], and [19], respectively.

precision and recall as shown in eq. (25)

$$Precision = \frac{TP}{TP + FP} \tag{23}$$

$$Recall = \frac{TP}{TP + FN} \tag{24}$$

$$F - measure = \frac{2 \cdot Precision \cdot Recall}{Precision + Recall}. \tag{25}$$

Note that the method from [11], due to its motion accumulation, gives results only for every three frames (frames #1, #4, #7, …), while the ground truth is available only for odd-indexed frames. Therefore, for this method, only the results for frames #1, #7, #13, …, are shown, while for other methods, the results for all odd-indexed frames are shown. Fig. 8 shows that, although the method from [11] yields good performance in the initial part of the sequence, it is not able to track the ball accurately in the latter part of the sequence, whereas the proposed method has a good tracking performance throughout the sequence. The other two methods have worse performance. Table II summarizes the average *Precision*, *Recall* and *F-measure* from Fig. 8. The results show that the proposed method has the overall best tracking performance among the four tested methods.

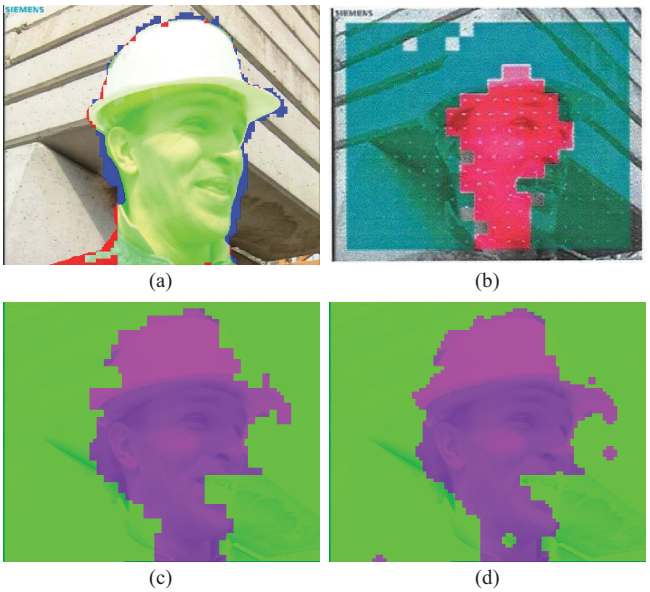

Fig. 10. Object detection and tracking by (a) proposed method for frame #10 of *Foreman*. (b)–(d) Segmentation results from [17], [21], and [19], respectively.

The method from [11] first accumulates MVs to smooth out object motion and suppress noise. This process, however, may result in wrong MVs in certain cases, for example the motion of boats between frames #68 and #76 of *Coastguard*. The authors use residual magnitude along with region growing to obtain spatial segmentation. Then the blocks are classified into "certain blocks" and "uncertain blocks" based on the temporal projection of previously detected objects to the current frame. The correlation between spatial segmentation and temporal-based classification is further exploited for final segmentation. During the spatial segmentation of *Mobile Calendar*, the region around the ball, which contains noisy MVs, is segmented as part of the ball. This false segmentation propagates into subsequent frames, as indicated by the results in Fig. 8. Meanwhile, our method has the advantage of employing

TABLE III

TOTAL AVERAGE OF *Precision*, *Recall*, AND *F-Measure* (IN PERCENTAGE) FOR DIFFERENT METHODS

Method	Measure	*Mobile Calendar*	*Coastguard*	*Stefan* (CIF)	*Stefan* (SIF)	*Hall Monitor*	*Flower Garden*	*Table Tennis*	*City*	*Foreman*	Avg.
Proposed	***Precision***	75.9	64.3	84.2	84.7	72.8	82.9	94.1	92.9	92.3	**82.7**
	Recall	88.4	89.4	68.3	67.8	84.4	95.8	88.0	96.5	90.4	**85.5**
	F-Measure	81.2	74.4	74.1	74.3	78.1	88.8	90.8	94.6	91.2	**83.0**
[21]	***Precision***	31.8	4.8	18.5	18.4	27.9	53.2	76.3	86.8	85.8	**44.8**
	Recall	91.6	86.3	86.0	88.5	91.9	99.0	72.9	96.9	64.3	**86.4**
	F-Measure	40.5	8.1	25.5	24.4	37.3	68.8	69.9	91.5	69.9	**48.4**
[19]	***Precision***	6.6	3.0	10.7	10.5	15.6	34.6	48.9	77.8	81.2	**32.1**
	Recall	93.3	95.9	87.8	88.4	90.1	99.2	76.2	97.0	65.3	**88.1**
	F-Measure	12.1	5.8	18.3	17.9	22.9	50.7	52.2	84.2	69.5	**37.1**

both spatial and temporal information at the same time through the ST-MRF model, and manages to avoid some of the problems faced by the method from [11]. This is demonstrated well in *Mobile Calendar*, which comprises camera motion, dynamic background, and partial occlusion, as well as in *Coastguard*, which involves strong camera motion between frames #68 and #76, as well as occlusion. Two more examples of segmentation and tracking result are shown in Fig. 9 and Fig. 10, where the proposed method is compared with the methods of [17], [19], and [21]. The average values of *Precision*, *Recall* and *F-measure* for all sequences where ground truth is available are shown in Table III for [19], [21], and the proposed method. Please note that the number of frames originally used in [21] (the first 50 frames of each sequence) is different from the number of frames used here (up to 100 frames for some sequences). Unfortunately, for the method from [11], we were only able to obtain object masks for *Mobile Calendar* from the authors, which is why the *Mobile Calendar* results are shown separately in Table II. As seen in the tables, the proposed method has the highest *Precision* and *F-measure* across all sequences, averaging almost a two-fold improvement in both of these metrics. The reason why its *Recall* is sometimes lower than that of the other two methods is that it tracks the boundary of the target object fairly closely in a block-based fashion, and therefore excludes from the object those pixels that fall into boundary blocks that get declared as the background, resulting in higher FN. By comparison, the other two methods often include the object and a large portion of the background into the segmented region, resulting in a smaller FN.

The example shown in Fig. 11 shows the tracked trajectory produced by the proposed method for several standard sequences, superimposed onto the last frame in the sequence. Trajectory is obtained by connecting the center of gravity of the tracked target through the frames. Each blue asterisk represents the center of gravity produced by the proposed method in a particular frame, while yellow squares show the centers of gravity of the ground truth. The trajectory result of ball tracking in *Mobile Calendar* is shown in Fig. 11(a). Although the ball has unreliable MVs due to its rotational movement and relatively small size, and even changes its direction twice; the proposed algorithm is able to track it reasonably well. In *Coastguard* in Fig. 11(b), the small boat moves from right to left; the camera at first follows the small boat until frame #66, then moves upwards quickly, and after that it follows the big boat starting at frame #75. During the first 66 frames, because the camera motion is in line with the movement of the small boat, the location of the small boat relative to the frame is fairly static. During the frames #67 to #74, when the camera moves upwards quickly, the MVs are rather erratic. The method of [11] has problems in this part of the sequence and fails to track the small boat, as noted in [11]. Similarly, the methods of [19] and [21] fail here as well, since the camera motion completely overwhelms object

Fig. 11. Trajectory results of (a) *Mobile Calendar*, (b) *Coastguard*, (c) *Stefan* CIF, (d) *Stefan* SIF, (e) *Hall Monitor*, and (f) *Flower Garden*, Blue asterisk: proposed algorithm. Yellow square: ground truth.

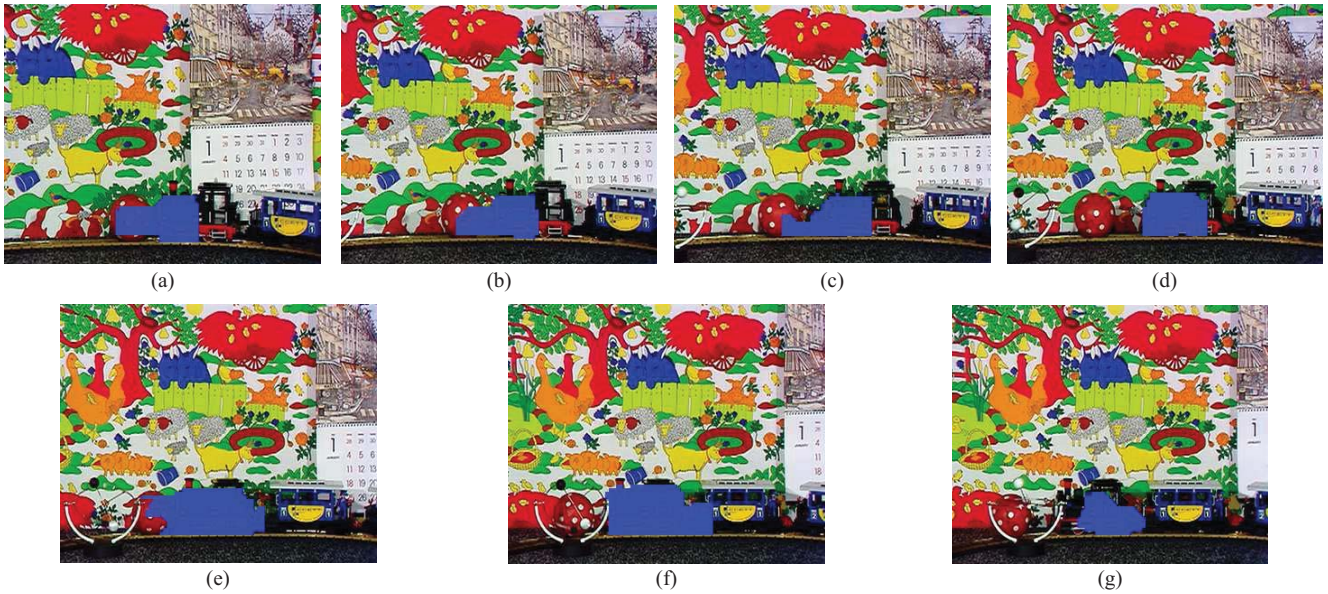

Fig. 12. Illustration of a tracking problem for objects with similar motion in *Mobile Calendar*. The method initially tracks the ball, but after the ball touches the train, the method starts to track the train. (a) Frame #130. (b) Frame #150. (c) Frame #165. (d) Frame #190. (e) Frame #240. (f) Frame #275. (g) Frame #285.

TABLE IV

AVERAGE AND STANDARD DEVIATION OF THE EUCLIDIAN DISTANCE BETWEEN THE OBJECT'S GROUND TRUTH GRAVITY CENTER AND THE OBTAINED GRAVITY CENTER IN PIXELS

	Mobile Calendar	*Coastguard*	*Stefan* (CIF)	*Stefan* (SIF)	*Hall Monitor*	*Flower Garden*	*Table Tennis*	*City*	*Foreman*	**Avg.**
Average	7.9	3.3	16	14	6.9	11	5.6	3.2	9.8	**8.6**
Std.	4.72	1.61	6.49	5.99	2.46	2.98	3	1.87	7.32	**4**

motion. However, the proposed method is able to track the small boat fairly accurately even through this challenging part of the sequence. Fig. 11(c) and Fig. 11(d) (*Stefan* CIF and SIF) are good examples of having both rapid movement of the target object (the player) and the camera. Since our algorithm is not able to detect the player's feet as a part of the target, the center of gravity produced by the proposed algorithm is above the ground truth center of gravity. Other than that, the proposed method correctly tracks the player even in this challenging MV field involving fast camera motion and rapid player's movement. Fig. 11(e) shows the trajectory results for *Hall Monitor*, where the camera is fixed and the man is moving from the door to the end of the hall. In Fig. 11(f) the camera motion is toward the right, therefore the position of the target (tree trunk) changes from right to left within the frame. The center of gravity goes upwards in the latter part because the bottom of the tree trunk disappears and the center of gravity rises as the result.

Table IV summarizes quantitative results of target trajectory estimation in terms of the average Euclidean distance between our results and ground truth data, as well as its standard deviation. The results indicate a fairly good tracking performance in various sequences and scenarios. It can be noticed that *Stefan* CIF, *Stefan* SIF, and *Foreman* have the worst results among the tested sequences. Certain characteristics of these sequences make successful tracking difficult. The tennis player's feet are hard to detect as part of the target due to their small size, which leads to inaccurate MVs, since the blocks covering the feet also cover a large part of the ground. As a result, the estimated center of gravity is above the ground truth center of gravity. Similarly, the foreman's neck and shoulders are not included in the target by the proposed method due to zero MVs in this area, while the segmented ground truth for *Foreman* includes neck and shoulders. Hence, in this sequence as well, the estimated center of gravity is above the ground truth one, although it could be argued that the ground truth here is not the most reasonable, as it perhaps should not include neck and shoulders.

One of the limitations of this framework is in dealing with two or more nearby objects with very similar MVs. For example, in the *Mobile Calendar* sequence (see Fig. 12), after the ball touches the train, starting at frame #97, both the ball and the train move in the same direction and have almost equal MVs. Therefore, our method starts to track a part of the train in addition to the ball. This problem continues until a large part of the train is declared as the target object (see frame #130). Although the ball then starts to separate from the train, the method tracks both the ball and the train as the same object (see frame #150). When the ball is far enough from the train, the largest part of the tracked region belongs to the train (see frame #165). As a result, the method stops tracking the ball and instead starts tracking only the train (see frame #190) until

TABLE V
Precision (IN PERCENTAGE) FOR VARIOUS QP VALUES

QP	*Mobile Calendar*	*Coastguard*	*Stefan* (CIF)	*Stefan* (SIF)	*Hall Monitor*	*Flower Garden*	*Table Tennis*	*City*	*Foreman*	Avg.
16	80.2	62.2	82.6	83.6	55.2	85.1	95.6	93.6	95.4	**81.5**
20	84.6	62.7	83.1	84.5	58.7	85.7	94.8	93.7	94.8	**82.5**
24	78.5	61.0	82.6	84.9	65.8	85.9	94.8	93.2	93.2	**82.2**
28	75.9	64.3	84.2	84.7	72.8	82.9	94.1	92.9	92.3	**82.7**
32	77.0	57.9	83.6	85.2	71.6	79.2	92.8	91.8	90.1	**81.0**
36	73.8	49.9	82.5	82.1	71.6	66.2	89.2	90.2	86.3	**76.9**
40	75.9	46.1	80.5	77.3	71.9	61.7	74.3	88.5	85.9	**73.6**

TABLE VI
Recall (IN PERCENTAGE) FOR VARIOUS QP VALUES

QP	*Mobile Calendar*	*Coastguard*	*Stefan* (CIF)	*Stefan* (SIF)	*Hall Monitor*	*Flower Garden*	*Table Tennis*	*City*	*Foreman*	Avg.
16	85.0	87.4	66.9	68.2	86.6	95.7	86.6	95.8	86.6	**84.3**
20	81.9	86.8	68.9	67.5	89.1	95.2	88.4	95.9	88.4	**84.7**
24	88.3	90.5	67.7	67.4	86.4	95.2	87.9	96.2	89.3	**85.4**
28	88.4	89.4	68.3	67.8	84.4	95.8	88.0	96.5	90.4	**85.5**
32	91.0	89.4	65.6	67.4	83.5	97.0	91.5	96.4	91.6	**85.9**
36	86.9	92.6	66.4	67.3	82.0	98.1	89.0	95.9	90.5	**85.4**
40	78.6	92.3	67.9	69.6	81.5	98.1	90.2	93.6	87.7	**84.4**

TABLE VII
F-Measure (IN PERCENTAGE) FOR VARIOUS QP VALUES

QP	*Mobile Calendar*	*Coastguard*	*Stefan* (CIF)	*Stefan* (SIF)	*Hall Monitor*	*Flower Garden*	*Table Tennis*	*City*	*Foreman*	Avg.
16	81.8	71.9	72.7	74.0	66.8	89.9	90.7	94.7	90.5	**81.4**
20	82.5	72.2	74.2	73.7	70.3	90.1	91.3	94.7	91.3	**82.3**
24	82.4	72.0	72.9	73.8	74.4	90.2	91.0	94.7	91.0	**82.5**
28	81.2	74.4	74.1	74.3	78.1	88.8	90.8	94.6	91.2	**83.0**
32	82.9	69.4	72.2	74.1	77.0	86.9	92.0	94.0	90.8	**82.1**
36	79.3	63.3	72.5	73.1	76.4	77.8	89.0	92.9	88.2	**79.2**
40	75.3	59.0	72.5	72.0	76.4	74.4	81.5	90.9	86.6	**76.5**

the ball touches the train again. Then, after a few frames, a part of the ball is once more detected as the target object (see frame #240). The same situation happens again and only the train is tracked in the subsequent frames (see frame #275). The train is being tracked until it almost stops at frame #280. From this frame on, the detected object region start to shrink because the tracked object is no longer moving, and therefore, there is no motion information for the object to be tracked (see frame #285). Finally, the algorithm detects no objects at frame #293. This type of problem could possibly be resolved by decoding the texture and looking at pixel-level information such as color, edges, etc. Neighboring objects that have the same MVs could probably be distinguished this way, but our goal in this work was to examine how well we could track using only the MV information.

In the experiments described thus far, the QP value for H.264/AVC encoding was set to 28. In Tables V–VII we show the quantitative performance of the proposed method for various values of QP between 16 and 40. The average *Precision*, *Recall*, and *F-measure* for different QP values are also shown in Fig. 13. As seen in these results, the tracking performance is reasonably consistent for QP values of 28 and below, and starts to drop as the QP (and with it, the compression ratio) increases. This is due to the less accurate MVs at higher compression ratios.

The average processing time for the main functions of the proposed tracking method is listed in Table VIII. These results were obtained on a Dell Optiplex system with 2.83 GHz Intel Core 2 Quad CPU and 8 GB RAM, using the MATLAB implementation of the proposed tracking method. The MV decoding time is not included in these results, since it is highly decoder dependent, and not really part of this work. Due to the availability of hardware-based decoding solutions, the MV decoding time is expected to be very small in practice. As seen in Table VIII, the average time for ST-MRF processing is 8 ms per frame, while the average time for pre-processing is 120 ms per frame, mostly due to GME and GMC. Hence, more efficient pre-processing seems to be the way to reduce the complexity of the proposed tracking method. Note that in cases where the camera is known to be static (e.g., certain surveillance situations), GME and GMC may be omitted, resulting in the tracking time of only 9 ms per frame, which is well within the real-time requirements even using the current MATLAB implementation. In any case, it can be expected that

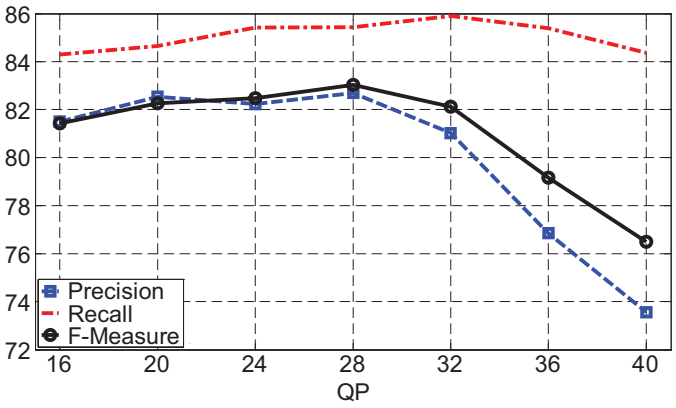

Fig. 13. Average *Precision*, *Recall*, and *F-measure* (in percentage) versus QP value.

TABLE VIII
AVERAGE PROCESSING TIME PER FRAME

Function	Processing time
MV Approximation for intracoded blocks	1 ms
GME and GMC	119 ms
Labeling initialization and ST-MRF model	8 ms
Total	128 ms

the processing time can be significantly reduced in a C/C++ implementation.

The processing time needed to run the algorithm of [21], including GME, initial clustering and MRF classification, is almost one second on average per frame on the same platform as our method with MATLAB implementation. The processing time for the MATLAB re-implementation of [19] is almost 250 ms on average per frame. Note that the abovementioned processing times are for segmenting a whole frame, while the processing time in our algorithm (128 ms on average per frame) is for tracking a single target object. However, the majority of this time (120 ms) is assigned for performing preprocessing, which is done for the whole frame, and only a fraction of this time is spent on ST-MRF based tracking (8 ms).

V. CONCLUSION

In this paper, we have presented a novel approach to track a moving object in a H.264/AVC-compressed video. The only data from the compressed stream used in the proposed method are the motion vectors and block coding modes. As a result, the proposed method has a fairly low processing time, yet still provides high accuracy. After the preprocessing stage, which consists of intra-coded block motion approximation and global motion compensation, we employ Spatio-Temporal Markov Random Field model to detect and track a moving target. Using this model, an estimate of the labeling of the current frame is formed based on the previous frame labeling and current motion information. The results of experimental evaluations on ground truth video demonstrate superior functionality and accuracy of our approach against other state-of-the-art compressed-domain segmentation/tracking approaches. Although our algorithm works well even with fixed parameter values, possibly better performance may be obtained by adaptive tuning, although this would in general increase the complexity. Along these lines, dynamic parameter tuning as proposed by Kato *et al.* [5] would be worth investigating as future work.

REFERENCES

[1] M. G. Arvanitidou, M. Tok, A. Krutz, and T. Sikora, "Short-term motion-based object segmentation," in *Proc. IEEE Int. Conf. Multimedia Expo*, Barcelona, Spain, Jul. 2011, pp. 1–6.

[2] C. Aeschliman, J. Park, and A. C. Kak, "A probabilistic framework for joint segmentation and tracking," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, San Francisco, CA, Jun. 2010, pp. 1371–1378.

[3] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 25, no. 5, pp. 564–575, May 2003.

[4] T. Drummond and R. Cipolla, "Real-time visual tracking of complex structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 7, pp. 932–946, Jul. 2002.

[5] Z. Kato, T.-C. Pong, and J. C.-M. Lee, "Color image segmentation and parameter estimation in a Markovian framework," *Pattern Recognit. Lett.*, vol. 22, nos. 3–4, pp. 309–321, 2001.

[6] Z. Yin and R. Collins, "Belief propagation in a 3D spatio-temporal MRF for moving object detection," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, Jun. 2007, pp. 1–8.

[7] Y. Wang, K.-F. Loe, T. Tan, and J.-K. Wu, "Spatiotemporal video segmentation based on graphical models," *IEEE Trans. Image Process.*, vol. 14, no. 7, pp. 937–947, Jul. 2005.

[8] R. Huang, V. Pavlovic, and D. N. Metaxas, "A new spatio-temporal MRF framework for video-based object segmentation," in *Proc. MLVMA Conjunct. Eur. Conf. Comput. Vis.*, 2008, pp. 1–12.

[9] Y. Wang, "A dynamic conditional random field model for object segmentation in image sequences," in *Proc. IEEE Comput. Vis. Pattern Recognit.*, vol. 1. Jun. 2005, pp. 264–270.

[10] A. Cherian, J. Andersh, V. Morellas, N. Papanikolopoulos, and B. Mettler, "Autonomous altitude estimation of a UAV using a single onboard camera," in *Proc. IEEE Int. Conf. Intell. Robots Syst.*, Oct. 2009, pp. 3900–3905.

[11] Z. Liu, Y. Lu, and Z. Zhang, "Real-time spatiotemporal segmentation of video objects in the H.264 compressed domain," *J. Visual Commun. Image Represent.*, vol. 18, no. 3, pp. 275–290, 2007.

[12] Z. Liu, L. Shen, and Z. Zhang. "An efficient compressed domain moving object segmentation algorithm based on motion vector field," *J. Shanghai Univ. (English Ed.)*, vol. 12, no. 3, pp. 221–227, 2008.

[13] W. Fei and S. Zhu, "Mean shift clustering-based moving object segmentation in the H.264 compressed domain," *IET Image Process.*, vol. 4, no. 1, pp. 11–18, Feb. 2010.

[14] V. Mezaris, I. Kompatsiaris, N. V. Boulgouris, and M. G. Strintzis, "Real-time compressed-domain spatiotemporal segmentation and ontologies for video indexing and retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 14, no. 5, pp. 606–621, May 2004.

[15] C. Käs and H. Nicolas, "An approach to trajectory estimation of moving objects in the H.264 compressed domain," in *Proc. 3rd Pacific Rim Symp. Adv. Image Video Technol.*, LNCS 5414. 2009, pp. 318–329.

[16] S. Treetasanatavorn, U. Rauschenbach, J. Heuer, and A. Kaup, "Bayesian method for motion segmentation and tracking in compressed videos," in *Proc. 27th DAGM Conf. Pattern Recognit.*, LNCS 3663. 2005, pp. 277–284.

[17] S. Treetasanatavorn, U. Rauschenbach, J. Heuer, and A. Kaup, "Model based segmentation of motion fields in compressed video sequences using partition projection and relaxation," in *Proc. Visual Commun. Image Process.*, Beijing, China, Jul. 2005, pp. 111–120.

[18] S. Treetasanatavorn, U. Rauschenbach, J. Heuer, and A. Kaup, "Stochastic motion coherency analysis for motion vector field segmentation on compressed video sequences," in *Proc. IEEE Workshop Image Anal. Multimedia Interact. Services*, Apr. 2005, pp. 1–4.

[19] W. Zeng, J. Du, W. Gao, and Q. Huang, "Robust moving object segmentation on H.264/AVC compressed video using the block-based MRF model," *Real-Time Imaging*, vol. 11, no. 4, pp. 290–299, Aug. 2005.

[20] Y.-M. Chen and I. V. Bajic, "A joint approach to global motion estimation and motion segmentation from a coarsely sampled motion vector field," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 21, no. 9, pp. 1316–1328, Sep. 2011.

[21] Y.-M. Chen, I. V. Bajic, and P. Saeedi, "Moving region segmentation from compressed video using global motion estimation and Markov random fields," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 421–431, Jun. 2011.
[22] Y.-M. Chen, I. V. Bajic, and P. Saeedi, "Motion segmentation in compressed video using Markov random fields," in *Proc. IEEE Int. Conf. Multimedia Expo*, Singapore, Jul. 2010, pp. 760–765.
[23] W. You, M. S. H. Sabirin, and M. Kim, "Moving object tracking in H.264/AVC bitstream," in *Proc. Int. Workshop Multimedia Content Anal. Mining*, LNCS 4577. 2007, pp. 483–492.
[24] W. You, M. S. H. Sabirin, and M. Kim, "Real-time detection and tracking of multiple objects with partial decoding in H.264/AVC bitstream domain," *Proc. SPIE, Real-Time Image Video Process.*, vol. 7244, pp. 72440D-1–72440D-12, Mar. 2009.
[25] C. Niu and Y. Liu, "Moving object segmentation in the H.264 compressed domain," in *Proc. Asian Conf. Comput. Vis. Conf.*, LNCS 5995. 2009, pp. 645–654.
[26] T. Wiegand, G. J. Sullivan, G. Bjontegaard, and A. Luthra, "Overview of the H.264/AVC video coding standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 13, no. 7, pp. 560–576, Jul. 2003.
[27] P. Vandewalle, J. Kovacevic, and M. Vetterli, "Reproducible research in signal processing," *IEEE Signal Process. Mag.*, vol. 26, no. 3, pp. 37–47, May 2009.
[28] J. E. Besag, "Spatial interaction and the statistical analysis of lattice systems," *J. Royal Stat. Soc., Ser. B*, vol. 36, no. 2, pp. 192–236, 1974.
[29] J. Besag, "On the statistical analysis of dirty pictures," *J. Royal Stat. Soc. B*, vol. 48, no. 3, pp. 259–302, 1986.
[30] P. J. Rousseeuw and A. M. Leroy, *Robust Regression and Outlier Detection*. New York: Wiley, 2003.
[31] R. Venables and B. Ripley, *Modern Applied Statistics with S*. New York: Springer-Verlag, 2002, pp. 119–126.
[32] P. J. Rousseeuw and C. Croux, "Alternatives to the median absolute deviation," *J. Amer. Stat. Assoc.*, vol. 88, no. 424, pp. 1273–1283, 1993.
[33] Y. Hu and T. J. Dennis, "Simulated annealing and iterated conditional modes with selective and confidence enhanced update schemes," in *Proc. 5th Annu. IEEE Symp. Comput.-Based Med. Syst.*, Jun. 1992, pp. 257–264.
[34] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 6, no. 6, pp. 721–741, Nov. 1984.
[35] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for Markov random fields," in *Proc. Eur. Conf. Comput. Vis.*, LNCS 3952. 2006, pp. 16–29.
[36] J. Astola, P. Haavisto, and Y. Neuvo, "Vector median filters," *Proc. IEEE*, vol. 78, no. 4, pp. 678–689, Apr. 1990.
[37] R. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2004, pp. 39–44.
[38] M. G. Arvanitidou, A. Glantz, A. Krutz, T. Sikora, M. Mrak, and A. Kondoz, "Global motion estimation using variable block sizes and its application to object segmentation," in *Proc. IEEE Workshop Image Anal. Multimedia Interact. Services*, London, U.K., May 2009, pp. 173–176.
[39] A. Smolic, M. Hoeynck, and J.-R. Ohm, "Low-complexity global motion estimation from P-frame motion vectors for MPEG-7 applications," in *Proc. IEEE Int. Conf. Image Process.*, vol. 2. Vancouver, BC, Canada, Sep. 2000, pp. 271–274.
[40] *H.264/AVC Reference Software JM18.0*. (2012, Aug.) [Online]. Available: http://iphome.hhi.de/suehring/tml

Sayed Hossein Khatoonabadi received the B.Sc. degree in computer science from Kerman University, Kerman, Iran, and the M.S. degree in computer engineering from the Amirkabir University of Technology, Tehran, Iran, in 2002 and 2005, respectively. He is currently pursuing the Ph.D. degree with the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada.

He was involved in several research projects in biomedical signal processing, including automatic arrhythmia detection of ECG, from 2005 to 2010. His current research interests include video content analysis, machine learning and pattern recognition, and biomedical signal processing.

Ivan V. Bajić (S'99–M'04–SM'11) received the B.Sc.Eng. degree (*summa cum laude*) in electronic engineering from the University of Natal, Durban, South Africa, in 1998, and the M.S. degree in electrical engineering, the M.S. degree in mathematics, and the Ph.D. degree in electrical engineering from Rensselaer Polytechnic Institute, Troy, NY, in 2000, 2002, and 2003, respectively.

He was a Visiting Assistant Professor with the Department of Electrical and Computer Engineering, University of Miami, Coral Gables, FL, from 2003 to 2005. Since 2005, he has been with the School of Engineering Science, Simon Fraser University, Burnaby, BC, Canada, where he is currently an Associate Professor and the Co-Director of the Multimedia Communications Laboratory. He has authored over a dozen and co-authored over 60 publications. His current research interests include signal, image, and video processing and compression, perceptual aspects of visual information processing, and multimedia communication.