Compressed-Domain Video Object Tracking using Markov Random Fields with Graph Cuts Optimization

Fernando Bombardelli(✉), Serhan Gül, and Cornelius Hellge

Dept. of Video Coding & Analytics
Fraunhofer Heinrich Hertz Institute, Berlin, Germany
{fernando.bombardelli,serhan.guel,cornelius.hellge}@hhi.fraunhofer.de

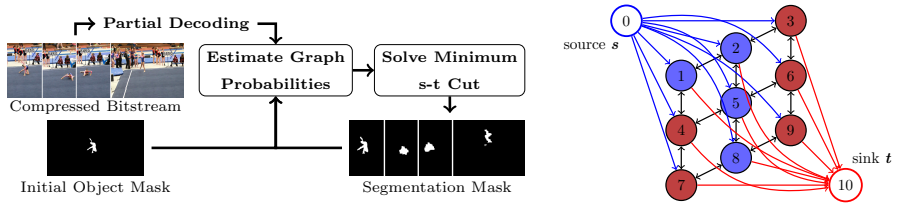
Abstract. We propose a method for tracking objects in H.264/AVC compressed videos using a Markov Random Field model. Given an initial segmentation of the target object in the first frame, our algorithm applies a graph-cuts-based optimization to output a binary segmentation map for the next frame. Our model uses only the motion vectors and block coding modes from the compressed bitstream. Thus, complexity and storage requirements are significantly reduced compared to pixel-domain algorithms. We evaluate our method over two datasets and compare its performance to a state-of-the-art compressed-domain algorithm. Results show that we achieve better results in more challenging sequences.

1 Introduction

Visual object tracking is an important task in many applications of computer vision such as human-computer interaction, motion-based recognition, video surveillance, traffic control, and vehicle navigation [25]. Although some efficient algorithms have been proposed recently [11, 7], many high-accuracy visual tracking algorithms consist of multistage, complex solutions that require large amounts of processing. Such high computational complexity can be a limiting factor in environments with limited processing resources, such as embedded systems, and in scenarios that require efficient processing of several video streams in parallel. A typical example of such an application is the large-scale analysis of videos from surveillance cameras deployed in a smart city [4].

An alternative to tracking algorithms that operate on decoded and reconstructed pixels are the so-called *compressed-domain* algorithms that directly operate on compressed video data. These algorithms operate on features extracted from the video bitstream such as prediction block types, motion vectors (MVs) and transform coefficients. They do not require a full decoding of the video bitstream and have a reduced amount of storage and processing requirements (due to compression) with significantly lower computational complexity.

In this paper, we present a compressed-domain object tracking method that uses only the extracted MVs and block coding modes from a H.264/AVC video



(a) Processing pipeline of the proposed approach. Given an H.264/AVC compressed video bitstream and an initial object mask, for each frame we (i) extract its motion vectors, (ii) estimate the probabilities in our Markov Random Field model, (iii) generate its corresponding graph, and (iv) infer the new object's mask by finding an exact solution of a minimum s-t cut optimization problem.

(b) Example of a constructed graph. Supposing a minimum s-t cut that yields the partition depicted by the blue and red-colored nodes, then setting $\omega_1^t, \omega_2^t, \omega_5^t, \omega_8^t = 1$ and $\omega_3^t, \omega_4^t, \omega_6^t, \omega_7^t, \omega_9^t = 0$ determines an optimal solution to the problem.

Fig. 1. Overview of our method

bitstream. Our approach is also applicable to the newer H.265/HEVC standard and other block-based video compression schemes. However, we chose H.264/AVC since it is still the most widely used standard, especially for surveillance cameras and embedded systems. We define the problem as tracking-by-segmentation as follows: H and W denote the height and the width of the pictures, then, given a sequence of frames $\{F_t\}_{t\in\mathbb{N}}$ and a binary matrix $M_1 \in \{0,1\}^{\mathrm{H}\times\mathrm{W}}$ labeling the object in the frame F_1, the objective is to infer for every $t > 1$ the matrix $M_t \in \{0,1\}^{\mathrm{H}\times\mathrm{W}}$, which segments the pixels of the frame F_t into the classes *object* and *non-object*. An overview of our approach is shown in Fig. 1 (a).

Our main contributions are: (i) formulation of the compressed-domain tracking task as an optimization problem over a Markov Random Field (MRF) by defining the appropriate energy (potential) functions; (ii) application of a graph-cuts-based solution to perform binary segmentation relying only on MVs from the compressed bitstream.

2 Related Work

Various compressed-domain tracking and moving object detection methods were proposed in literature with recent work focusing on tracking rigid objects in H.264/AVC and H.265/HEVC video bitstreams [3]. Some works on moving object detection solely rely on macroblock (MB) types [18, 16, 17]. Poppe *et al.* [18] analyze the bitstreams on a MB level and introduced a syntax level algorithm based on MB sizes in bits. Their algorithm learns a background model in an initial training phase and uses this model as a benchmark in the following steps to determine MBs corresponding to moving objects. Laumer *et al.* [16] define weights for all possible MB types considering the decision process of the encoder while selecting the MB types. Assigned weights indicate the probability that a MB belongs to a moving object.

Other approaches rely either solely on the extracted MVs or a combination of MB types and MVs [24, 13, 26]. Käs and Nicolas [13] use global motion estimation to filter out outlier MVs and use the filtered MVs to estimate the trajectory of moving objects in the scene. Wojaczek *et al.* [24] utilize MB types at a preprocessing step for fast evaluation of video streams. If an object is detected, a more precise pixel-domain object detector is employed to obtain a finer segmentation.

Most successful tracking results in compressed domain were obtained by using statistical inference models, specifically MRFs. Zeng *et al.* [27] published one of the earliest works that employed an MRF model for MV-based tracking. Their approach merges similar MVs into moving objects by minimizing the MRF energy and defining different MV types. Their model considers the spatial continuity and temporal consistency of MVs to identify groups of MVs belonging to the tracking object. However, it is only suitable for static scenes since the method does not take into account the camera motion. Khatoonabadi and Bajić [14] treat the tracking problem in a Bayesian framework and use MVs to compute a motion coherence metric. Furthermore, they employ global motion compensation to deal with dynamic scenes and propose an interpolation technique to assign motion information to intra-coded blocks based on the MVs of their neighboring blocks. Enhancing the objective function of [14], Gül *et al.* [10] use the color information from the first intra-coded frame (I frame) to create a Gaussian Mixture Model (GMM) based on the color distributions of the object and the background, provided by initial labeling, and update the solution at every I frame. Their approach increases the tracking accuracy with a negligible loss in terms of computational speed caused by the sparse sampling of color information from I frames.

3 Proposed Method

The video object tracking problem can be considered as finding an optimal segmentation of the current video frame into *object* and *non-object* classes based on the binary segmentation from the previous one. We assume that pixelwise ground truth annotation of the tracking object is given for the first frame as initial input. We then treat the tracking problem as an inference problem on an MRF model (Section 3.1) and use minimum s-t cut (*i.e.* graph cuts) method [19] to solve the optimization problem (Section 3.2). Our method operates on a uniformly sampled motion vector field. Since H.264/AVC has variable prediction block sizes and intra-coded blocks without MVs [23], we apply some preprocessing steps to obtain such a uniform MV field (Section 3.4).

3.1 Probabilistic Framework

Our approach is based on the work of Khatoonabadi and Bajić [14] who model the object tracking task as an inference problem using an MRF. Their idea is to have for each frame t one random variable ω^t representing the binary

segmentation of the respective frame. So, the problem is to find the most likely labeling ω^t given the previous one ω^{t-1} and the motion information.

In contrast to them, we assign every block a random variable which may assume the values 1, for *object*, or 0, for *non-object*. In order to model the relations between adjacent blocks in a uniform manner, we construct a uniformly sampled MV field by preprocessing, and obtain a grid with N blocks. In our probabilistic framework, for each block $n = 1, \ldots, N$, the corresponding observed MV $\mathbf{x}_n$ is conditionally dependent on the respective block state $\omega_n^t \in \{0, 1\}$, and each block state is conditionally dependent on its four-connected neighbors and its state ω_n^{t-1} in the previous frame.

The inference problem can then be stated as a maximum-a-posteriori optimization problem, that is,

$$\max_{\omega_1^t,\ldots,\omega_N^t} P(\omega_1^t, \ldots, \omega_N^t \,|\, \mathbf{x}_1, \ldots, \mathbf{x}_N, \omega_1^{t-1}, \ldots, \omega_N^{t-1}). \tag{1}$$

For convenience we denote $\omega_1^t, ..., \omega_N^t$ as $\boldsymbol{\omega}^t$; $\omega_1^{t-1}, ..., \omega_N^{t-1}$ as $\boldsymbol{\omega}^{t-1}$; and the objective function of (1) as f. From Bayes' theorem and the conditional independences implied by the MRF it follows that

$$f(\boldsymbol{\omega}^t) = \frac{P(\mathbf{x}_1,...,\mathbf{x}_N|\boldsymbol{\omega}^t, \boldsymbol{\omega}^{t-1})P(\boldsymbol{\omega}^t, \boldsymbol{\omega}^{t-1})}{P(\mathbf{x}_1, ..., \mathbf{x}_N, \boldsymbol{\omega}^{t-1})} = \left[\prod_{n=1}^{N} P(\mathbf{x}_n|\omega_n^t)\right] \frac{P(\boldsymbol{\omega}^t, \boldsymbol{\omega}^{t-1})}{P(\mathbf{x}_1,...,\mathbf{x}_N, \boldsymbol{\omega}^{t-1})}. \tag{2}$$

Consider the undirected graph relating the random variables $\boldsymbol{\omega}^t$ and $\boldsymbol{\omega}^{t-1}$, and note that every maximal clique in it contains exactly two vertices. Moreover, the set of its maximal cliques $\mathcal{C}$ can be partitioned into three subsets: the set $\mathcal{C}_1 = \{\{\omega_n^t, \omega_n^{t-1}\} \,|\, n = 1, \ldots, N\}$ of pair of vertices corresponding to the same blocks in different frames; the set $\mathcal{C}_2 = \{\{\omega_i^t, \omega_j^t\} \,|\, \{i, j\} \in \mathcal{L}\}$ of neighboring blocks in the frame t; and the set $\mathcal{C}_3 = \{\{\omega_i^{t-1}, \omega_j^{t-1}\} \,|\, \{i, j\} \in \mathcal{L}\}$ of neighboring blocks in the frame $t-1$. ($\mathcal{L}$ denotes the set of indices of the adjacent blocks in a same frame.) Then, its joint probability can be factorized by the Hammersley-Clifford theorem [5] as

$$P(\boldsymbol{\omega}^t,\boldsymbol{\omega}^{t-1}) = \frac{1}{Z} \exp\left(-\sum_{W_n \in \mathcal{C}_1} \Psi_n^1(W_n) - \sum_{W_{\{ij\}} \in \mathcal{C}_2} \Psi_{\{ij\}}^2(W_{\{ij\}}) - \sum_{W_{\{ij\}} \in \mathcal{C}_3} \Psi_{\{ij\}}^3(W_{\{ij\}})\right), \tag{3}$$

where Z is a normalization constant, and Ψ^k, $k = 1, 2, 3$, are the potential functions. Eliminating the terms that do not depend on the $\boldsymbol{\omega}^t$ (since they do not affect the result of the optimization), we obtain the following from (2) and (3)

$$f(\boldsymbol{\omega}^t) \propto \left[\prod_{n=1}^{N} P(\mathbf{x}_n|\omega_n^t)\right] \exp\left(-\sum_{n=1}^{N} \Psi_n^1(\omega_n^t, \omega_n^{t-1}) \; - \sum_{\{i,j\} \in \mathcal{L}} \Psi_{\{ij\}}^2(\{\omega_i^t, \omega_j^t\})\right). \tag{4}$$

Finally, the optimization problem can be solved by minimizing the negative logarithm of f, that is,

$$\min_{\omega_1^t,\ldots,\omega_N^t} E(\boldsymbol{\omega}^t) := \sum_{n=1}^{N} \Psi_n^1(\omega_n^t, \omega_n^{t-1}) - \ln P(\mathbf{x}_n|\omega_n^t) \; + \sum_{\{i,j\} \in \mathcal{L}} \Psi_{\{ij\}}^2(\{\omega_i^t, \omega_j^t\}). \tag{5}$$

3.2 Solving through Minimum s-t Cut

In order to solve (5) through minimum s-t cut, we construct a graph $G = (V, A)$ from the current frame. For that, let N be the number of blocks in the frame, and define the set V of vertices of G as $V = \{0, 1, \ldots, N, N+1\}$. In this set, 0 corresponds to the *source* vertex, $N+1$ to the *sink* vertex, and for all $n=1, \ldots, N$ the vertex n corresponds to the random variable ω_n^t. Furthermore, define the set A of arcs of G as the union of three subsets A_1, A_2, A_3, where $A_1 := \{(0,n) | n=1, ..., N\}$ is the set of arcs going from the source to every other vertex (except the sink); $A_2 := \{(n, N+1) | n=1, \ldots, N\}$ is the set of arcs going from every vertex (except the source) to the sink; and A_3 is the set of arcs linking adjacent blocks in the frame in both directions. Fig. 1 (b) shows an example of the constructed graph.

To determine the arc capacities, we first split the terms of the objective function (5) and define $U_n(\omega_n^t) := \Psi_n^1(\omega_n^t, \omega_n^{t-1}) - \ln P(\mathbf{x}_n | \omega_n^t)$ for all $n = 1, \ldots, N$, and $B_{ij}(\omega_i^t, \omega_j^t) := \Psi_{\{ij\}}^2(\{\omega_i^t, \omega_j^t\})$ for all arcs $(i, j) \in A_3$. Then, we set for all $n=1, \ldots, N$, the arc capacities $u_{0,n} = U_n(0)$, and $u_{n,N+1} = U_n(1)$. And for all arcs $(i, j) \in A_3$ we set $u_{i,j} = B_{ij}(0, 1)$. Note that $B_{ij}(0,1) = B_{ij}(1,0)$ and $B_{ij} \equiv B_{ji}$.

Once we have found a minimum s-t cut in G which partitions the set V into S and T with the *source* $0 \in S$ and the *sink* $N+1 \in T$, we determine the optimal solution $\boldsymbol{\omega}^t$ to (5) by setting, for all $n \in \{1, \ldots, N\}$, $\omega_n^t = 1$ if $n \in S$, and $\omega_n^t = 0$ otherwise.

Among the several algorithms for solving the minimum s-t cut problem [6], we have chosen Excesses Incremental Breadth-First Search (EIBFS) [8, 9], which finds global minima in polynomial time. Moreover, this algorithm allows the optimal solution of a given frame to be used as an initial, near-optimal solution in the next one, substantially reducing the computational effort.

3.3 Probability Models

In our approach, the probabilities $P(\mathbf{x}_n | \omega_n^t = 0)$ and $P(\mathbf{x}_n | \omega_n^t = 1)$ are estimated by computing the two-dimensional histogram of vectors from the background and from the object, respectively. Although the exact regions of the MVs belonging to *object* and *non-object* classes are unknown at the current frame, the previously computed object's mask M_{t-1} gives the best estimate of these regions at the previous frame. Therefore, assuming that the tracking object moves smoothly, we can use the vectors of the current frame in the area occupied by M_{t-1} as object samples, and the vectors outside this area as non-object samples. In order to obtain a more robust sample of MVs for the *object* class, we apply an erosion morphological transformation with a predefined kernel size C_1 on M_{t-1}, which removes its borders.

Moreover, we compute these probabilities using additive smoothing with parameter $\alpha > 0$. In other words, given the histogram of a class ω_n^t with a total of B bins and for which K vectors were used to create it, if k is the number of samples inside the bin corresponding to a given MV $\mathbf{x}_n$, then its conditional probability is given by

$$P(\mathbf{x}_n | \omega_n^t) = \frac{k + \alpha}{K + \alpha B}. \tag{6}$$

This procedure not only smooths the histogram, but also guarantees non-zero probabilities, which is necessary to calculate their logarithms in Eq. (5).

For modeling the potential functions Ψ_n^1, note that they should indicate a temporal relation between the current frame and the previous one. We, therefore, follow the proposal of Khatoonabadi and Bajić [14] of using a measure of temporal continuity of the blocks. In summary, to determine the temporal continuity $D_n \in [0,1]$ of a block $n \in \{1,\ldots,N\}$, one firstly projects it backwards onto the previous frame according to its MV, and then determines how much of its area lies on a block labeled as *object*. On the matrix yielded by calculating this measure on the whole frame we apply a Gaussian filter with predefined standard deviation σ in order to make it more robust to noisy MVs. Then, for all $n = 1,\ldots,N$, we define Ψ_n^1 parametrized by $C_2 > 0$ as follows:

$$\Psi_n^1(\omega_n^t, \omega_n^{t-1}) = \begin{cases} C_2\, D_n & \text{if } \omega_n^t = 0, \\ C_2\,(1 - D_n) & \text{if } \omega_n^t = 1. \end{cases} \tag{7}$$

The potential function $\Psi_{\{ij\}}^2$ models the relation between adjacent blocks i and j in the current frame t as a cost, which is positive if the labels are different and zero otherwise. We empirically define it as being inversely proportional to the distance between the respective MVs $\mathbf{x}_i$ and $\mathbf{x}_j$, as follows:

$$\Psi_{\{ij\}}^2(\{\omega_i^t, \omega_j^t\}) = \begin{cases} \gamma\,(\|\mathbf{x}_i - \mathbf{x}_j\|^2 + \gamma^2)^{-1.5} & \text{if } \omega_i^t \neq \omega_j^t, \\ 0 & \text{if } \omega_i^t = \omega_j^t, \end{cases} \tag{8}$$

where $\gamma > 0$ is a parameter of the function.

3.4 Preprocessing

Our proposed tracking algorithm uses two types of information from the encoded video stream: MVs and block coding modes (macroblock partitioning). H.264/AVC allows for four block partitioning modes: 16×16, 8×16, 16×8, and 8×8 where the 8×8 blocks can further be partitioned into 8×4, 4×8, and 4×4 sub-blocks. In order to fulfill the input assumptions of the proposed model, a uniformly sampled MV field needs to be constructed. Since the smallest partition in H.264/AVC is 4×4, we map all MVs to 4×4 blocks. Specifically, we apply the following preprocessing steps: (i) partially decode the video stream (as in [12]), yielding a list of prediction blocks and respective MVs. (ii) Arrange these MVs into a two-channel matrix in order to produce a uniform grid of vectors, where every entry corresponds to a 4×4 block. (iii) For intra-predicted blocks, which have no associated MVs, we compute the Polar Vector Median as defined in [14] by employing the MVs of the not intra-coded neighboring blocks.

Although these preprocessing steps may be sufficient for the case of static camera, the method could fail in the presence of camera motion. Therefore, we estimate global motion with the method introduced by Smolić *et al.* in [21] and

improved by Arvanitidou *et al.* in [2]. In this model, six parameters $a_1, \ldots, a_6$ are used to obtain an approximation $\tilde{\mathbf{v}}_{x,y}(\mathbf{a})$ of the MV at the position (x, y) by

$$\tilde{\mathbf{v}}_{x,y}(\mathbf{a}) = \begin{bmatrix} a_1 x + a_2 y + a_3 - x \\ a_4 x + a_5 y + a_6 - y \end{bmatrix}. \tag{9}$$

Then, given M MVs $\mathbf{v}_{x^1,y^1}, ..., \mathbf{v}_{x^M,y^M}$, the goal is to find the global motion parameter $\mathbf{a} \in \mathbb{R}^6$ which minimizes the error $\sum_{k=1}^{M} \|\tilde{\mathbf{v}}_{x^k,y^k}(\mathbf{a}) - \mathbf{v}_{x^k,y^k}\|^2$. The optimal solution $\mathbf{a}^*$ is determined by solving this least squares problem using an M-estimator. As in [2], we only use MVs of prediction blocks larger than or equal to 8×8 pixels since they are less likely to belong to moving foreground. Finally, in order to compensate the camera motion, we subtract from the MV field the estimates $\tilde{\mathbf{v}}_{x,y}(\mathbf{a}^*)$ since they indicate the effect of global motion at the respective coordinates.

4 Experiments

We evaluated our method on two different datasets and compared its tracking performance with the ST-MRF method in [14]. For evaluation, we use the commonly employed metrics in tracking literature: Precision, Recall and F-Measure [25]. These three measures are computer at each video frame and their values are averaged for each sequence.

4.1 Experimental Setup

The first dataset (referred to as *derf*), also used in [14], comprises nine standard test sequences with formats CIF (352×288 pixels) and SIF (352×240 pixels) encoded by the H.264/AVC JM 18.0 Reference Software [22]. Their group-of-pictures (GOP) structure is IPPP..., *i.e.*, the first frame is an intra-predicted picture (predicted from previously decoded data from the same picture), and each frame after the first one is a forward-predicted picture (predicted from a picture that has been previously coded and transmitted) containing MVs pointing to the previous reference pictures [20].

Since the ground truth segmentations of the *derf* dataset are not available over the entire duration of the sequences (but only on a relatively small portion), performance of the tracking algorithms can only be evaluated over a limited portion of each sequence. Moreover, *derf* contains test sequences that were originally collected for evaluation of video codecs. As a result, the sequences do not thoroughly address the challenging situations (*e.g.* complex camera motion, multiple similar objects) that a tracking algorithm may encounter. To address these issues, we performed further experiments on a second, larger dataset created for Visual Object Tracking (VOT) Challenge 2016 [15]. The dataset has 60 sequences with resolutions varying from 320×240 to 1280×720 pixels. VOT 2016 challenge aims at comparing single-object visual trackers most of which were published at major computer vision conferences and journals in recent years. Therefore, the dataset

Table 1. Evaluation results in percentages for both datasets. Columns *Prec.*, *Rec.* and *F-Me.* show averaged precision, recall and F-measure, respectively. Numbers in bold face correspond to the method which obtained the highest F-measure for the respective video sequence.

Derf	ST-MRF [14]			Proposed		
Sequence	Prec.	Rec.	F-Me.	Prec.	Rec.	F-Me.
Mobile Calendar	**75.9**	**88.4**	**81.2**	70.8	95.9	80.8
Coastguard	**64.3**	**89.4**	**74.4**	43.0	98.7	57.8
Stefan (CIF)	**84.2**	**68.3**	**74.1**	75.8	72.2	73.6
Stefan (SIF)	**84.7**	**67.8**	**74.3**	69.5	79.0	73.2
Hall Monitor	72.8	84.4	78.1	**71.7**	**87.4**	**78.6**
Flower Garden	**82.9**	**95.8**	**88.8**	70.6	98.4	81.5
Table Tennis	**94.1**	**88.0**	**90.8**	89.0	92.1	90.5
City	**92.9**	**96.5**	**94.6**	91.3	97.4	94.2
Foreman	**92.3**	**90.4**	**91.2**	98.0	76.1	85.2
Average	**82.7**	**85.4**	**83.1**	75.5	88.6	79.5

VOT2016	ST-MRF [14]			Proposed		
Sequence	Prec.	Rec.	F-Me.	Prec.	Rec.	F-Me.
bag	17.9	90.7	28.3	**34.8**	**79.0**	**47.1**
birds2	8.7	5.9	6.7	**31.5**	**47.1**	**37.5**
blanket	20.8	30.9	22.3	**40.3**	**69.1**	**49.6**
bmx	**55.4**	**77.1**	**62.8**	60.4	41.3	48.1
crossing	**36.1**	**92.9**	**49.4**	22.2	85.7	33.9
dinosaur	39.7	88.0	52.7	**45.9**	**87.3**	**58.6**
fernando	51.5	81.5	53.2	**55.4**	**74.1**	**59.8**

VOT2016	ST-MRF [14]			Proposed		
Sequence	Prec.	Rec.	F-Me.	Prec.	Rec.	F-Me.
fish1	19.7	21.0	19.2	**17.2**	**34.5**	**22.6**
fish2	**36.0**	**37.1**	**34.2**	25.9	45.8	31.5
fish3	24.9	25.7	21.2	**64.5**	**56.7**	**50.5**
fish4	8.0	8.0	7.5	**54.7**	**79.1**	**63.1**
graduate	5.1	1.7	2.3	**45.3**	**48.4**	**45.7**
gymnastics1	2.3	1.1	1.3	**20.3**	**29.2**	**21.2**
gymnastics2	42.1	25.7	27.4	**30.7**	**45.5**	**36.1**
gymnastics4	4.8	1.7	2.2	**63.0**	**66.5**	**63.3**
hand	10.3	20.7	11.6	**31.8**	**69.4**	**41.1**
handball2	**17.7**	**91.4**	**25.0**	9.5	61.4	15.9
helicopter	27.2	89.9	26.5	**21.5**	**52.1**	**28.7**
marching	**23.9**	**39.6**	**22.8**	3.3	5.3	3.9
matrix	9.4	5.6	6.3	**18.6**	**29.0**	**22.2**
nature	23.1	12.4	14.8	**26.9**	**21.1**	**22.0**
octopus	54.9	98.3	67.8	**56.5**	**88.0**	**68.2**
racing	31.7	84.7	40.5	**33.1**	**92.1**	**46.2**
shaking	17.0	89.0	24.0	**28.8**	**90.1**	**42.7**
sheep	2.8	1.5	1.8	**25.9**	**92.8**	**38.0**
singer1	**47.3**	**33.8**	**35.4**	23.4	82.2	34.4
singer2	13.7	89.0	22.4	**23.9**	**63.9**	**32.8**
singer3	**37.1**	**92.9**	**48.5**	17.9	33.0	19.5
soldier	**40.5**	**83.8**	**51.9**	32.8	74.5	42.7
sphere	26.1	84.7	37.7	**40.3**	**43.2**	**40.3**
tiger	10.3	27.6	13.7	**30.2**	**53.9**	**34.5**
traffic	**14.9**	**72.1**	**21.5**	1.7	12.4	2.8
Average	24.4	50.2	27.0	**32.4**	**57.9**	**37.6**

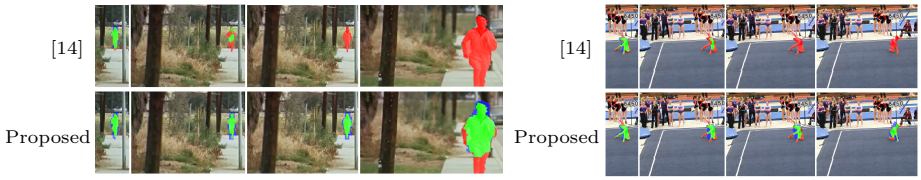
contains challenging sequences with strong camera motion and cluttered scenes with distracting objects. Since the sequences are originally available as JPEG images, we re-encoded them as H.264/AVC sequences with a GOP structure IPPP...IPPP... using the x264 encoder [1].

We implemented our tracker in Python[1] and integrated the minimum s-t cut solver (developed in C++ [9]) via Cython. The experiments were conducted on a Intel Xeon E5-v3 CPU with 32GB RAM running Linux. The parameters of the proposed model (*cf*. Section 3.3) were empirically determined as follows: $\alpha, \sigma = 1$; $\gamma = 0.5$; $C_1, C_2 = 6$.

4.2 Results and Discussion

Table 1 shows the results obtained for both trackers. At the *derf* dataset, our method performs slightly worse than ST-MRF [14] by 3.6% on average in terms of F-Measure. However, we believe that the high tracking performance over this dataset should be interpreted cautiously because the experiments were performed on a short portion of the videos (on average 73 frames) due to the limited availability of ground truth annotations (*cf*. Section 4.1). This implies that the tracker is less likely to lose the tracking object and also less likely to encounter significant scene changes. Another factor for the high performance over *derf* dataset can be attributed to the relative simplicity of the object motion patterns and relatively large object/picture size ratio which provides more MVs initially associated with the object, thus more reliable input data.

[1] Available online at https://github.com/bombardellif/hhi-stmrftracking

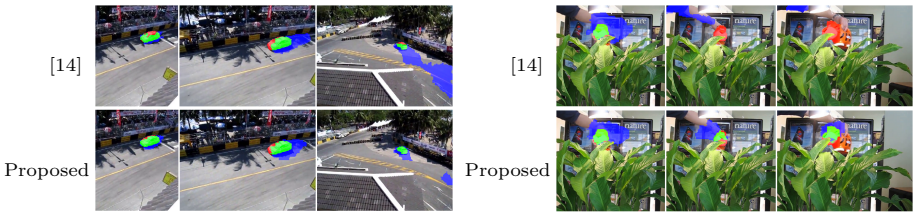


(a) Frames 2, 6, 12 and 455 of *graduate*. (b) Frames 2, 10, 17 and 70 of *gymnastics4*.

Fig. 2. Tracking results for videos with tracking objects with low motion. True positives, false positives and false negatives are denoted in green, blue and red, respectively. The proposed method can track objects with little movement more accurately and for a longer time compared to the ST-MRF method [14].

For the VOT2016 dataset, we only show the results for those video sequences in which at least one of the methods obtained an F-measure above 20%. From the table, we observe that the performance of both methods depends highly on the individual sequence and varies strongly within the dataset. In some sequences such as *birds2*, *sheep*, *graduate* and *gymnastics4*, we observed that the tracking object has (or appears to have) low motion for some seconds causing most MVs to become zero. Fig. 2 shows example frames from two such sequences: *graduate* and *gymnastics4*. In *graduate*, a person runs towards the camera and he appears to be motionless for multiple seconds (except for small arm movements) due to the scene perspective until he finally draws near to the camera and makes a right turn. In the shown portion of *gymnastics4*, the gymnast retains her pose for about two seconds as the (possibly hand-held) camera wiggles briefly. We observe that while ST-MRF loses the tracking object for these cases, our method is able to track them successfully (albeit with some false negatives). The reason why the proposed method handles such cases better than ST-MRF can be attributed to the arc weights of the graph, in which a minimum s-t cut is determined (*cf.* Section 3.2). This behavior can be explained by inspecting the individual terms of Eq. (5): if the probabilities $P(\mathbf{x}_n|\omega_n^t=0)$ and $P(\mathbf{x}_n|\omega_n^t=1)$ are nearly equal for some block n (*e.g.* background and object have very similar MVs), then the second term $\Psi_n^1(\omega_n^t, \omega_n^{t-1})$, which measures the temporal continuity, becomes more decisive in the objective function (5). This mechanism ensures that the resulting segmentation mask is similar to the one in the previous frame if the object is (or appears to be) motionless.

Similarly, our method also performs better in sequences with fast object motion, maintaining higher precision throughout the sequences. Fig. 3 shows example frames from two such sequences: *racing* and *tiger*. In *racing*, a car moves on a race track followed by the camera as it makes a turn. In *tiger*, we see a person's arm holding a toy tiger as the person moves the toy with quick, jerky movements behind a flowerpot placed closer to the camera. We observe for both cases that our method manages to track the fast moving object more successfully than ST-MRF. Specifically, our method produces much less false



(a) Frames 30, 40 and 150 of *racing*. (b) Frames 75, 150 and 300 of *tiger*.

Fig. 3. Tracking results for videos with fast motion. True positives, false positives and false negatives are denoted in green, blue and red, respectively. The proposed method yields more precise segmentation masks for fast moving targets than ST-MRF [14].

positives in *racing* and does not lose the target object in *tiger* unlike ST-MRF. The success of our method for sequences with fast motion can be explained again by inspecting Eq. (5): if the MVs of the tracking object are very different from the background MVs, then the probability $P(\mathbf{x}_n|\omega_n^t)$ in Eq. (5) becomes more decisive for the optimization. Thus, the importance of the previous mask is suppressed.

Nevertheless, our approach tends to fail in more challenging instances, such as *marching* and *traffic*. The former is a case of complete occlusion, where the marching band member being tracked is rapidly occluded by a person which passes running in front of the camera. The latter is an example of complex camera motion, where the tracking object is followed by a moving camera installed on the helmet of a motorcyclist. One reason for the failure of our method in such cases is that most MVs in the region occupied by the object either do not correspond to the tracking object (as in *marching*) or become hard to distinguish from the background (as in *traffic*), which impairs the segmentation by graph cuts.

In general, both methods have difficulties in performing reliable tracking especially for some of the more challenging sequences in the dataset VOT2016. For those, it is important to notice that compressed-domain approaches have fundamental limitations. One such case is when the object of interest does not occupy a sufficient amount of blocks in the frame, leading to a reduced number of vectors describing its motion. Another issue is the accumulation of the tracking error along the sequence. That is, since the outcome in a given frame depends on the outcome of the previous one, incorrect classifications are propagated and tend to grow with the length of the video. A possible solution to this problem — as proposed by Gül *et al.* [10] — is to periodically regularize the objective function using the color information sampled from the I frames of the video bitstream.

5 Conclusion

In this paper, we present a technique for tracking moving objects in H.264/AVC compressed videos. The proposed approach relies only on motion vectors and

block coding modes readily available in the compressed video bitstream. Thus, it requires low processing and storage compared to pixel-domain tracking algorithms. We compared the performance of our method with a state-of-the-art compressed-domain tracking algorithm. We observed that average tracking accuracy significantly depends on the complexity of the scene and motion of the tracking object. In videos with relatively simple motion, both approaches have relatively high tracking accuracy and their performances are comparable. In more challenging sequences, our algorithm achieves a higher average tracking performance for most sequences. Specifically, we observed that our algorithm performs better in sequences containing low motion and fast moving targets, respectively.

Acknowledgments. The research leading to these results has received funding from the German Federal Ministry for Economic Affairs and Energy under the VIRTUOSE-DE project.

References

1. Aimar, L., Merritt, L., Petit, E., Chen, M., Clay, J., Rullgrd, M., Heine, C., Izvorski, A.: x264, the best H.264/AVC encoder – VideoLAN. [Online] https://www.videolan.org/developers/x264.html (Aug 2017)
2. Arvanitidou, M.G., Glantz, A., Krutz, A., Sikora, T., Mrak, M., Kondoz, A.: Global motion estimation using variable block sizes and its application to object segmentation. In: Image Anal. for Multimedia Interactive Services, 10th Workshop on. pp. 173–176. IEEE (2009)
3. Babu, R.V., Tom, M., Wadekar, P.: A survey on compressed domain video analysis techniques. Multimedia Tools and Appl. **75**(2), 1043–1078 (2016)
4. Becker, D., Bombardelli, F., Gül, S., Schmidt, M., Hellge, C., Sawade, O., Radusch, I.: Visual object tracking in a parking garage using compressed domain analysis. In: Proc. of the 9th ACM Multimedia Syst. Conf. (2018)
5. Besag, J.: Spatial interaction and the statistical analysis of lattice systems. Journal of the Royal Statistical Society. Series B (Methodological) pp. 192–236 (1974)
6. Boykov, Y., Kolmogorov, V.: An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. IEEE Trans. Pattern Anal. Mach. Intell. **26**(9), 1124–1137 (2004)
7. Galoogahi, H.K., Fagg, A., Huang, C., Ramanan, D., Lucey, S.: Need for speed: A benchmark for higher frame rate object tracking. In: Computer Vision (ICCV), 2017 IEEE International Conference on. pp. 1134–1143. IEEE (2017)
8. Goldberg, A., Hed, S., Kaplan, H., Tarjan, R., Werneck, R.: Maximum flows by incremental breadth-first search. Algorithms–ESA 2011 pp. 457–468 (2011)
9. Goldberg, A.V., Hed, S., Kaplan, H., Kohli, P., Tarjan, R.E., Werneck, R.F.: Faster and more dynamic maximum flow by incremental breadth-first search. Algorithms-ESA 2015 pp. 619–630 (2015)
10. Gül, S., Meyer, J.T., Hellge, C., Schierl, T., Samek, W.: Hybrid video object tracking in H.265/HEVC video streams. In: Multimedia Signal Process., 2016 IEEE 18th Int. Workshop on. pp. 1–5 (2016)
11. Held, D., Thrun, S., Savarese, S.: Learning to track at 100 fps with deep regression networks. In: European Conference on Computer Vision. pp. 749–765. Springer (2016)

12. Kantorov, V., Laptev, I.: Efficient feature extraction, encoding and classification for action recognition. In: CVPR. pp. 2593–2600 (2014)
13. Käs, C., Nicolas, H.: An approach to trajectory estimation of moving objects in the H.264 compressed domain. Advances in Image and Video Technology pp. 318–329 (2009)
14. Khatoonabadi, S.H., Bajić, I.V.: Video object tracking in the compressed domain using spatio-temporal markov random fields. IEEE Trans. Image Process. **22**(1), 300–313 (2013)
15. Kristan, M., Leonardis, A., Matas, J., Felsberg, M., Pflugfelder, R., Čehovin, L., Vojir, T., Häger, G., Lukežič, A., Fernandez, G.: The visual object tracking VOT2016 challenge results. Springer (Oct 2016), http://www.springer.com/gp/book/9783319488806
16. Laumer, M., Amon, P., Hutter, A., Kaup, A.: Compressed domain moving object detection based on H.264/AVC macroblock types. In: VISAPP (1). pp. 219–228 (2013)
17. Laumer, M., Amon, P., Hutter, A., Kaup, A.: Moving object detection in the H.264/AVC compressed domain. APSIPA Trans. on Signal and Inform. Process. **5** (2016)
18. Poppe, C., De Bruyne, S., Paridaens, T., Lambert, P., Van de Walle, R.: Moving object detection in the H.264/AVC compressed domain for video surveillance applications. Journal of Visual Communi. and Image Representation **20**(6), 428–437 (2009)
19. Prince, S.J.: Computer vision: models, learning, and inference. Cambridge University Press (2012)
20. Richardson, I.E.: The H.264 advanced video compression standard. John Wiley & Sons (2011)
21. Smolić, A., Hoeynck, M., Ohm, J.R.: Low-complexity global motion estimation from P-frame motion vectors for MPEG-7 applications. In: Image Process., Int. Conf. on. vol. 2, pp. 271–274. IEEE (2000)
22. Sühring, K.: H.264/AVC JM reference software. [Online] http://iphome.hhi.de/suehring/tml/ (Aug 2017)
23. Wiegand, T., Sullivan, G.J., Bjøntegaard, G., Luthra, A.: Overview of the H.264/AVC video coding standard. IEEE Trans. Circuits Syst. Video Technol. **13**(7), 560–576 (2003)
24. Wojaczek, P., Laumer, M., Amon, P., Hutter, A., Kaup, A.: Hybrid person detection and tracking in H.264/AVC video streams. In: VISAPP (1). pp. 478–485 (2015)
25. Yilmaz, A., Javed, O., Shah, M.: Object tracking: A survey. Acm computing surveys (CSUR) **38**(4), 13 (2006)
26. You, W., Sabirin, M., Kim, M.: Real-time detection and tracking of multiple objects with partial decoding in H.264/AVC bitstream domain. In: Proc. of SPIE, the Int. Society for Optical Eng. (2009)
27. Zeng, W., Du, J., Gao, W., Huang, Q.: Robust moving object segmentation on H.264/AVC compressed video using the block-based MRF model. Real-Time Imaging **11**(4), 290–299 (2005)