A Low Complexity Motion Segmentation Based on Semantic Representation of Encoded Video Streams

Maurizio Abbate, Ciro D'Elia, and Paola Mariano

Università di Cassino
Via G. Di Biasio, 43 02043 Cassino (FR) – Italy
{m.abbate,delia,p.mariano}@unicas.it

Abstract. Video streaming is characterized by a deep heterogeneity due to the availability of many different video standards such as H.262, H.263, MPEG-4/H.264, H.261 and others. In this situation two approaches to motion segmentation are possible: the first needs to decode each stream before processing it, with a high computational complexity, while the second is based on video processing in the coded domain, with the disadvantage of coupling between implementation and the coded stream. In this paper a motion segmentation based on a "generic encoded video model" is proposed. It aims at building applications in the encoded domain independently by target codec. This can be done by a video stream representation based on a semantic abstraction of the video syntax. This model joins the advantages of the two previous approaches by making it possible working in real time, with low complexity, and with small latency. The effectiveness of the proposed representation is evaluated on a low complexity video segmentation of moving objects.

1 Introduction

In many applications the video stream is available at the source only in compressed form because it is included in some system protocols such as 3G-324M, DVB-H, SIP/RTP and others, or because the NetCams, nowadays used in many applications, send data on RTP/RTSP in MPEG-4, H.264, H.263 and others. In this varied context, a direct approach to implement video processing algorithms consist in a work in the pixel domain after video decoding. This choice has the disadvantage of an high complexity, because of decoding and pixel domain processing. Moreover the encoded stream contains useful information for many applications, such as motion vectors. Hence a processing using such information is able to fully exploit the work of the encoder. For these reasons the proposed approach detects moving regions in the compressed domain without decompression. This allows us to work in real time, with low complexity, and with a small latency because, in principle, the representation could be constructed on the head of a video picture while its tail is still being received. The proposed representation addresses also the problem that in video applications,

G. Maino and G.L. Foresti (Eds.): ICIAP 2011, Part II, LNCS 6979, pp. 209–218, 2011.

as stated before, there are many encoded stream standards, thus implying some problems such as the dependence between the implementation and the codec, or in other words the leak of generality of the implemented algorithm. Indeed our proposal is to represent the video using a generic model of encoded video streams, in order to develop codec independent algorithms. In this paper we discuss a low complexity motion segmentation, based on a semantic and unified representation of encoded video streams. Motion segmentation is a process that decomposes a video scene into moving objects or regions. Moving objects or regions extraction is a necessary preprocessing for many application such as scene interpretation, analysis and other. Moreover, the motion segmentation is also worthwhile in cooperation with real-time video trans-coding, object recognition, video surveillance applications and source encoding with content based video standards. Many segmentation methods proposed in the literature, regard the extraction of motion information as a separate process without using any information derived from video encoding process as [9,6]. Other methods use the information derived from encoded bitstream, like DCT coefficients [14,5,10,7], motion vectors [2,12,4,3,8], or both [11,13,1]. These are efficient for real-time applications because they work on encoded domain, however they have the disadvantage of being codec dependent. The novelty of our method is to build a video segmentation, independently from the codec syntax by the use of a semantic model of the encoded stream. The motion segmentation presented in this work is block-based and computed through some basic features contained on the semantic representation, like motion vectors, intra/inter block information and others. The direct use of these features is not sufficient for our aims, indeed encoders operate given a bit budget in order to minimize the source distortion. This implies that some features contained in the bitstream, like motion vectors, are present not only on moving regions but also on similar areas in order to efficiently represent some local changes, using information of neighborhood areas. In other circumstances on flat areas those features are absent even on moving objects. The approach has been evaluated on some standard sequences, but due to space constraints, we present results on two of the most used test videos, "Hall Monitor" and "Container", in order to make easy the comparison in literature. The results show that real-time motion segmentation can be achieved by using our approach. The paper is organized as follows. Section 2 presents the video bitstream model, Section 3 shows details of the proposed motion segmentation method. Experiments, conclusions and future works are in Section 4 and Section 5.

2 Encoded Video Stream Model

The proposed encoded video stream model is based on the abstraction of syntax elements of the most significant video standards. Looking at the standards and at their evolution, it's easy to notice that they share some common ideas. Indeed video standards syntax evolution moves from simple to more and more complex scene models, or on more complex tools for representing it. Video standard syntax could be seen as a set of encoding tools that, if appropriately used,

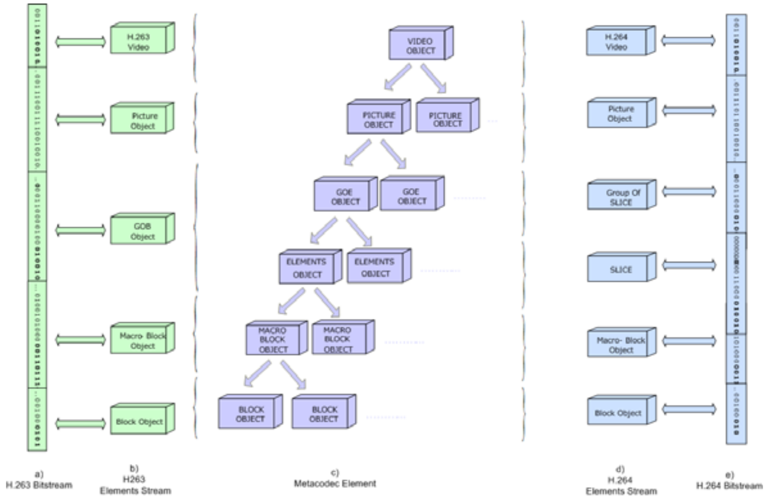


Fig. 1. Assuming in a) and e) respectively an H.263 and a H.264 bitstreams, the semantic representations specific for H.263 and H.264 bitstreams are shown respectively in b) and d), in which the hierarchical dependencies between elements defined by this standards are maintained; in c) the Semantic Representation of Encoded Video Stream is shown, which is able of generalizing different video standards

let the encoder to achieve "good" compression. The aim of our model is to catch common elements of the main standards or to generalize the novelty introduced by some of them. For this reason, introducing the proposed model is useful to reconsider some concepts about video compression and some encoding tools provided by various standards. As it is well known, video itself is basically a three-dimensional array of color pixels of which two dimensions lies in the spatial directions and the last dimension represents the time evolution of the scene. Video data contain spatial, temporal and chromatic redundancy, hence, in order to achieve video compression, standard codecs provide different tools to exploit redundancies. For the spacial redundancies the use of DCT transform tools is quite common. There are many syntactic differences, which could be viewed as a different scene model or different tools to represent the same model. More in details the syntax tools provided in ITU-H.261 are adequate on a scene model with a main object on a background, for instance a speaker on a background. Indeed H.261 introduces encoding of "future frame" by P-frame encoding tool. ITU-H.263 is also suited on the same scene model, but introduces more complex tools for representing scene evolution like PB-frames (Bidirectional Prediction), and many others. The ISO-IEC MPEG-2/ITU-H.262 has encoding tools to predict with high efficiency the redundant information by P-frames and B-frames and also introduces some tools to encode interlaced analog signals. The ISO-IEC MPEG-4 part2 introduces, instead, more complex scene models, made up by several visual object on the scene, explicitly encoded in the bitstream but quite complex to exploit. Finally ISO-IEC MPEG-4 part10/ITU-H.264 uses

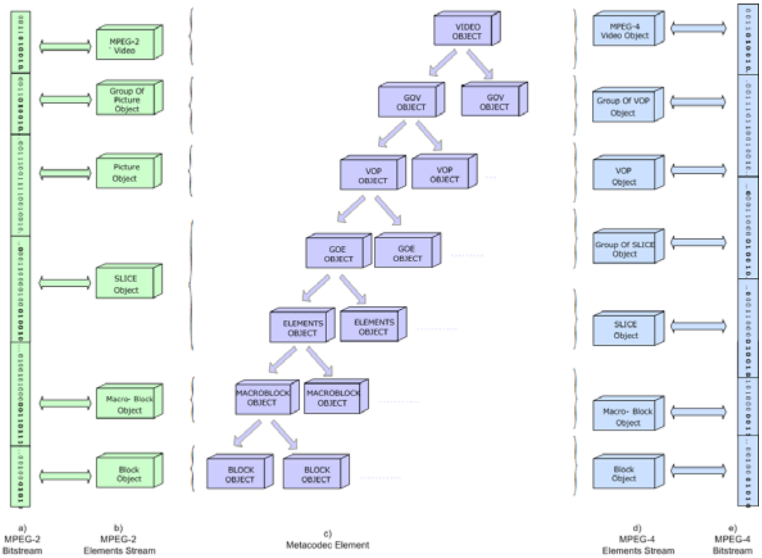


Fig. 2. Assuming in a) and e) respectively an MPEG-2 and a MPEG-4 bitstreams, the semantic representations specific for of MPEG-2and MPEG-4 bitstreams are shown respectively in b) and d), in which the hierarchical dependencies between elements defined by this standards are maintained; in c) the Semantic Representation of Encoded Video Stream is shown, which is able of generalizing different video standards

advanced but easy to implement, tools to represent the evolution of the scene. These tools give it a big advantage if compared with other standards in terms of performance. Even with this short survey of same characteristics of the main codec standards, it is possible to understand that, to obtain a unified semantic representation of the "encoder tools' syntax", is useful to build an abstraction of the different implementations of the encoding tools themselves such as cited P/B-frames. For instance the structure shown in Fig. 1b, represents the semantic description of the H.263 bitstream (Fig. 1a), where the hierarchical dependencies between elements defined by the H.263 standard are maintained. This type of representation makes video more easily manageable, but it's valid only for the H.263 bitstream. Indeed an H.264 bitstream cannot be represented by the same structure. To this aim we should define a specific representation in which groups of slices and slices appear (see Fig. 1d). In Fig. 1c) the generic model proposed to represent an encoded video stream is shown, where the sought level of generalization / abstraction is obtained through each of the semantic elements which compose the model. Each element of the model generalizes one or more syntactical elements of different video standards.

The model consists of Picture Objects containing information about the frame format (CIF, QCIF, etc.), the coding type used (Intra, Inter, etc.) and the quantization step size. *GroupOfElements* Object handles the case where the frame is divided into *SliceGroup* (H.264, MPEG-2). *Elements* Object generalizes GOB

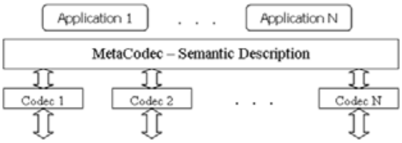


Fig. 3. Framework layers: the low layer is specialized on the supported codecs, the middle layer performs the semantic representation and finally on the top the application layer

(H.261, H.263) and Slices (H.263+, MPEG-2, H.264). The *Macroblock* Object contains the coding type used in the macroblock, the motion vector value and the blocks transmitted (Coded Block Pattern). Finally the *Block* Object contains luminance (Y) and chrominance (Cr-Cb) DCT coefficients. Though the proposed abstraction model is valid only for H.26X encoding, it can be extended to MPEG encoding streams by providing the semantic structure with a cap consisting of elements generalizing the Video Object Planes (VOP) and groups of VOPs, or groups of Pictures, which are typical syntactic elements of MPEG standards. Therefore we propose a model which provides a generic representation of the video bitstream at a higher level of abstraction, i.e. a metacodec (see Fig. 3). The framework is organized in three main layers: the low layer specialized on the supported codecs, which should map the specific codec syntax into the semantic representation of the middle layer, called “Metacodec”. On the top, at the third level, there is the application layer, like segmentation, that is based on “Metacodec” layer, which, being codec independent, allows the development of codec independent applications/algorithms. Hence the video segmentation, discussed in details in Section 3, places itself at the third level of Fig. 3. In order to easily evaluate the results of the motion segmentation algorithm, we have implemented a visualizer algorithm, called Halo, based on the semantic representation of the encoded stream, therefore codec independent. Experimental results show the effectiveness of the abstraction model.

3 Motion Segmentation Based on Semantic Description

In the context of video analysis, processing and coding, the segmentation is a very common operation and of particular interest. Video segmentation aims at partitioning a video sequence into several regions according to a given criterion. In the context of video analysis, processing and coding, the segmentation is a very common operation and of particular interest. As said before segmentation is at the third level of Fig. 3, but the segmentation itself could be seen as a further semantic layer, between applications and MetaCodec, in particular as a low level layer for the scene content representation. The presented motion segmentation algorithm starts from the basic idea that motion is a special change. It works in two steps: Spatial Segmentation and Temporal Segmentation. The *Spatial Segmentation* step labels a macroblock (MB) of a frame according to the MB spatial changes, then the *Temporal Segmentation* labels a MB according to its temporal behavior.

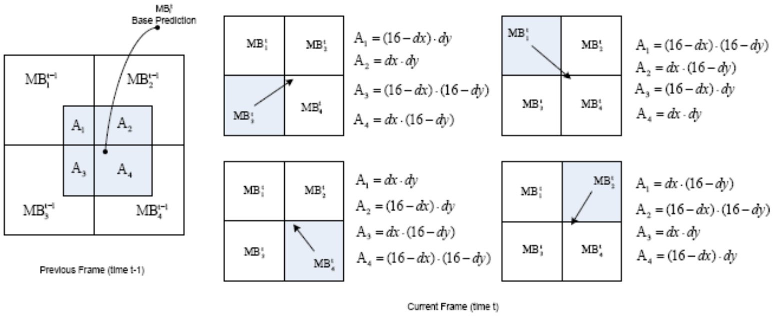

MB_i^t
Base Prediction
MB_1^{t-1}
MB_2^{t-1}
MB_3^{t-1}
MB_4^{t-1}
A_1
A_2
A_3
A_4
Previous Frame (time t-1)
MB_1^t
MB_2^t
MB_3^t
MB_4^t
$A_1 = (16 - dx) \cdot dy$
$A_2 = dx \cdot dy$
$A_3 = (16 - dx) \cdot (16 - dy)$
$A_4 = dx \cdot (16 - dy)$
$A_1 = (16 - dx) \cdot (16 - dy)$
$A_2 = dx \cdot (16 - dy)$
$A_3 = (16 - dx) \cdot dy$
$A_4 = dx \cdot dy$
$A_1 = dx \cdot dy$
$A_2 = (16 - dx) \cdot dy$
$A_3 = dx \cdot (16 - dy)$
$A_4 = (16 - dx) \cdot (16 - dy)$
$A_1 = dx \cdot (16 - dy)$
$A_2 = (16 - dx) \cdot (16 - dy)$
$A_3 = dx \cdot dy$
$A_4 = (16 - dx) \cdot dy$
Current Frame (time t)

Fig. 4. Temporal evolution of a MB "active", where dx and dy indicate the horizontal and vertical components of MV_i^t (MV_x, MV_y) associated to MB_i^t

3.1 Spatial Segmentation

In the first segmentation step we take into consideration the MBs of each frame. A MB that represents a moving object or local changes in the background is tagged as 1 ("active"), while a MB that is not changing is tagged as 0 ("static"), according to the following equation:

$$ACT_i^t = \begin{cases} 1, \begin{cases} \text{if } MV_i^t \neq 0. \\ \text{if } MB_i^t \text{ is INTRA in INTER frame.} \\ \text{if } MV_i^t = 0 \text{ and neighborhood is} \\ \text{``mainly'' in one of the previous cases.} \end{cases} \\ \\ 0, \text{ otherwise.} \end{cases} \tag{1}$$

where ACT_i^t, MB_i^t and MV_i^t, are respectively the action tag, the macroblock and the motion vector of the $i-th$ block at time t. More in deep, if the motion vector is zero, we consider the eight adjacent MBs of the current MB; if the neighborhood consists of a number of inter or intra MBs greater than a threshold Th_1, the MB considered is labeled as "active", otherwise "static".

3.2 Temporal Segmentation

After Spatial Segmentation, Temporal Segmentation has been performed in order to understand if an area is "active" due to a moving object or to a local change.Temporal Segmentation operates on matrices of labels ("static" or "active") previously computed. Starting from the consideration that a MB belonging to a moving object has a coherence of the action through some frames, at this stage our aim is to assess the temporal behavior of the "active" INTER MBs. We can say that it is possible to calculate roughly the Motion Coherence (MC) of the current MB at time t using the frames at time $t-1$ and $t-2$ through the weighted average of ACT coefficients influencing the current MB. More in deep,

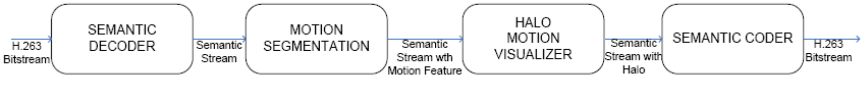


Fig. 5. The diagram blocks sequence of the experimental phase

$MC(1)_i^t$ and $MC(2)_i^t$ are defined as the weighted average of the ACT coefficients influencing MB_i^t block respectively at time $t-1$ and $t-2$, as in the following definitions:

$$MC(1)_i^t = \frac{\sum\limits_{k \in \eta(i, MV_i^t)} ACT_k^{t-1} \cdot A_k(i, MV_i^t)}{\sum_{k=1}^{4} A_k(i, MV_i^t)} \tag{2}$$

$$MC(2)_i^t = \frac{\sum\limits_{k \in \eta(i, MV_i^t)} MC(1)_i^{t-1} \cdot A_k(i, MV_i^t)}{\sum_{k=1}^{4} A_k(i, MV_i^t)} \tag{3}$$

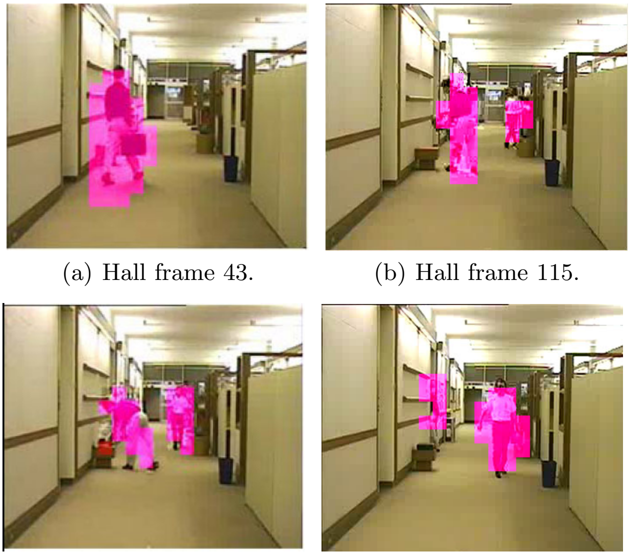
where $\eta(i, MV)$ is the set of indices of the MBs, in the previous frame, influencing the MB of index i by Motion Vector MV, depicted as $\{1, 2, 3, 4\}$ in Fig. 4 for sake of simplicity. In the same figure the current MB_i^t is predicted from parts of the four macroblocks MB_k^{t-1}, furthermore the coefficients $A_k(i, MV)$, representing the influence of each of the four macroblocks MB_k^{t-1} on the current macroblock, are calculated through the motion vector associated to the MB_i^t block, as shown by the formulas, where dx and dy are respectively the horizontal and vertical components of the motion vector. Finally, the temporal segmentation algorithm decides whether the MB_i^t belongs to a moving object depending on the fact that the following equation is greater or smaller than a threshold Th_2:

$$MOV_i^t = \alpha_0 \cdot ACT_i^t + \alpha_1 \cdot MC(1)_i^t + \alpha_2 \cdot MC(2)_i^t \tag{4}$$

where $\alpha_0 + \alpha_1 + \alpha_2 = 1$ and MOV is close to 0 if there is no movement and close to 1 if there is a coherent movement from previous MBs to the current one.

4 Experimental Results

The proposed motion segmentation was tested on some video sequences recorded by a static camera. For briefness, only the results obtained for the "Hall Monitor" (300 frames) and "Container" (300 frames) CIF sequences will be discussed. Each sequence was encoded into a simple H.263 profile. The encoded video bitstream was decoded by the metacodec layer and then the result was used by our algorithm of motion segmentation to detect the moving objects of the scene from a largely static background (Section 3). Because the use of a basic H.263 codec, the resulting segmentation is related with the size of macroblock (16×16 pixel). It was found that, by applying 300 frames "Hall" sequence to the proposed motion segmentation, coherent motions are actually detected throughout the whole



(a) Hall frame 43.

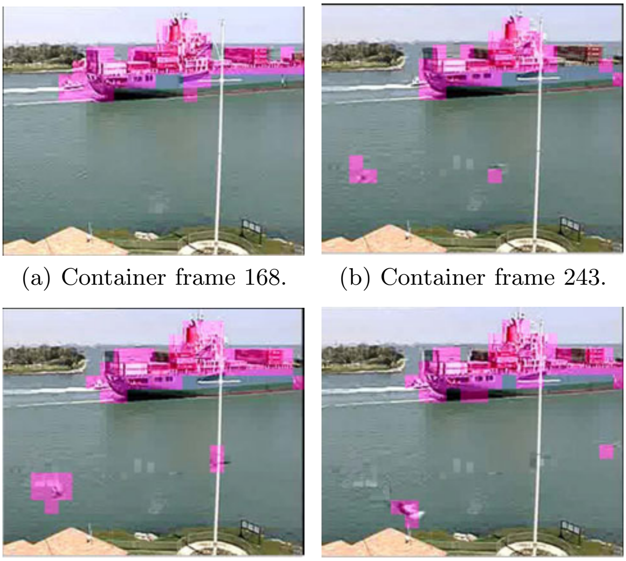
(b) Hall frame 115.

(c) Hall frame 150.

(d) Hall frame 241.

Fig. 6. Experimental Results on "Hall Monitor" Video Sequence ($Th_1 = 3$, $Th_2 = 0.6$, $\alpha_0 = 0.4$, $\alpha_1 = 0.3$, $\alpha_2 = 0.3$)

video sequences. The motion segmentation's results are visualized through the "Halo" algorithm, cited in Section 2, as the block diagram in Figure 5) shows. Figure 6 shows the segmentation results in the 43-th (man in black T-shirt), 115-th (entry of man in white T-shirt), 125-th (both men in the corridor), 241-th (output of man in black T-shirt) frames respectively; the segmentation algorithm detects the real movement of video scene, i.e. the two people moving in the corridor. Figure 7 shows the segmentation results for the "Container" video sequence in the 168-th (ship and speedboat), 243-th (passage of both gulls) and 268-th (exit of the first gulls) frames respectively. In the 168-th frame the segmentation is able to distinguish the two boats, as it is able to distinguish the two gulls in the 243-th frame. By examining all the experiments (not only those discussed here), we have observed that: 1) parameters α_0, α_1 and α_2 respectively set to 0.4, 0.3 and 0.3 are suitable for all tested video sequences; 2) the threshold Th_1 allows us to select the dimension of the moving objects we want to detect; 3) if we set the threshold Th_2 to 0.6, we can apply a majority rule to decide if a macroblock belongs to a moving object (i.e. if it is a moving macroblock in at least two of three frames). The experimental results show that our segmentation is able to identify motion areas using the limited information previously calculated by the codec (motion vectors and macroblock type) with very simple decision rules and a low computational complexity. However, it can wrongly decide like in the 168-th frame of Figure 7, where the algorithm detects motion of waves caused by the transit of the motorboat, so representing in this case the small changes of



(a) Container frame 168. (b) Container frame 243.

(c) Container frame 258. (d) Container frame 263.

Fig. 7. Experimental Results on “Container” Video Sequence ($Th_1 = 3$, $Th_2 = 0.6$, $\alpha_0 = 0.4$, $\alpha_1 = 0.3$, $\alpha_2 = 0.3$)

the background instead of the movements related to a real object. This kind of error is not due to a malfunctioning of the segmentation algorithm but simply to the fact that we are using only the information about the movement. Indeed the codec sometimes associates motion vectors to macroblocks that are not moving but are changed in texture; many of the errors due to this reason have been eliminated with the temporal segmentation, by recognizing as ”active” only those macroblocks whose motion remains consistent through three frames.

5 Conclusion and Future Developments

In this paper, a semantic model of encoded bitstreams has been proposed. Based on this model, a generic block-based motion segmentation method is introduced. The segmentation results from the Hall Monitor and Container sequences show that the proposed method can exploit the semantic description of an encoded video to perform motion segmentation. It also offers a practical approach to integrate the video encoding with the motion segmentation process, which indicates that the proposed segmentation method is suitable for real-time video applications like video surveillance and video transcoding. Future studies will be focused on detecting the video contents and their features through an efficient video segmentation algorithm based not only on motion but also on texture features computed from the semantic description.

References

1. Ahmad, A., Ahmad, B., Lee, S.: Fast and robust object detection framework in compressed domain. In: Proc. IEEE Sixth Int. Symposium on Multimedia Software Engineering, pp. 210–217 (December 2004)
2. Chung, R., Chin, F., Wong, K., Chow, K., Luo, T., Fung, H.: Efficient block-based motion segmentation method using motion vector consistency. In: MVA 2005 IAPR Conference on Machine Vision Applications, pp. 550–553 (May 2005)
3. Hong, W., Lee, T., Chang, P.: Real-time foreground segmentation for the moving camera based on h.264 video coding information. In: Proc. IEEE Int. Conf. on Future Generation Communication and Networking, pp. 385–390 (December 2007)
4. Hsieh, C., Lai, W., Chiang, A.: A real time spatial/temporal/motion integrated surveillance system in compressed domain. In: Proc. IEEE Int. Conf. on Intelligent Systems Design and Applications, pp. 658–665 (November 2008)
5. Ji, S., Park, H.: Region-based video segmentation using dct coefficients. In: Proc. IEEE Int. Con. Image Processing, vol. 2, pp. 150–154 (October 1999)
6. Karayiannis, Varughese, Tao, Frost, Wise, Mizrahi.: Quantifying motion in video recordings of neonatal seizures by regularized optical flow methods. IEEE Trans. Image Process.14(7), 890–903 (July)
7. Lee, S.W., Kim, Y.M., Choi, S.W.: Fast scene change detection using direct feature extraction from mpeg compressed video. IEEE Trans. Multimedia 2(4), 240–254 (2000)
8. Neri, A., Colonnese, S., Russo, G., Talone, P.: Automatic moving object and background separation. Signal Process.(Special Issue) (66), 219–232 (1998)
9. Nguyen, H., Worring, M., Dev, A.: Detection of moving objects in video using a robust motion similarity measure. IEEE Trans. Image Process. 1(9), 137–141 (2000)
10. Pons, J., Prades-Nebot, J., Albiol, A., Molina, J.: Fast motion detection in compressed domain for video surveillance. IEEE Electronics Letters 38(9), 409–411 (2002)
11. Porikli, F., Bashir, F., Sun, H.: Compressed domain video object segmentation. IEEE Trans. Image Process. 1(5297), 2–14 (2010)
12. Ritch, M., Canagarajah, N.: Motion-based video object tracking in the compressed domain. In: Proc. IEEE Int. Con. Image Processing, vol. 6, pp. 301–306 (2007)
13. Tao, K., Lin, S., Zhang, Y.: Compressed domain motion analysis for video semantic events detection. In: Proc. IEEE Int. Conf. on Information Engineering, pp. 201–204 (July 2009)
14. Zeng, W., Gao, W., Zhao, D.: Automatic moving object extraction in mpeg video. In: Proc. IEEE Int. Symposium on Circuits and Systems, vol. 2, pp. 524–527 (2003)