Compressed Domain Video Object Segmentation

Fatih Porikli, Huifang Sun

TR2005-040 May 2005

Abstract

We propose a compressed domain video object segmentation method for MPEG or MPEG-like encoded videos. Computational superiority is the main advantage of the compressed domain processing. In addition to computational advantage, the compressed domain video process possesses two important features, which are very attractive for object analysis. First, the texture characteristics are provided by the DCT coefficiens with the need of only partial decoding. Second, the motion information is readily available without incurring cost of complicated motion estimation process for not intra only MPEG encoded videos. In the proposed method, we first exploit the macro-block structure of the MPEG encoded video to decrease the spatial resolution of the processed data, which exponentially reduces the computational load. Further reduction of complexity is achieved by temporal grouping of the intra-coded and estimated frames into a single feature layer. The video segmentation is achieved by using the combination of DCt coefficients for I-frames and block motion veactors for P-frames. A frequency-temporal data structure is constructed. Starting from the blocks where the AC-coefficient energy and local inter-bloack DC-coefficient variance is small, the homogeneous volumes are enlarged by evaluating the distance of candidate vectors to the volume characteristics. Affine motion models are fit to volumes. Finally, a hierarchical clustering stage iteratively merges the most similar parts to generate an object partition tree as an output. The experimental results have shown that the proposed compressed domain video segmentation method provides the similar results as by using spatial domain process with much less computational complexity.

IEEE Transactions on Circuits and Systems for Video Technology

Compressed Domain Video Object Segmentation

Fatih Porikli, *Senior Member, IEEE,* and Huifang Sun, *Fellow, IEEE,*

Abstract

We propose a compressed domain video object segmentation method for MPEG or MPEG-like encoded videos. Computational superiority is the main advantage of the compressed domain processing. In addition to computational advantage, the compressed domain video process possesses two important features, which are very attractive for object analysis. First, the texture characteristics are provided by the DCT coefficients with the need of only partial decoding. Second, the motion information is readily available without incurring cost of complicated motion estimation process for not intra only MPEG encoded videos. In the proposed method, we first exploit the macro-block structure of the MPEG encoded video to decrease the spatial resolution of the processed data, which exponentially reduces the computational load. Further reduction of complexity is achieved by temporal grouping of the intra-coded and estimated frames into a single feature layer. The video segmentation is achieved by using the combination of DCT coefficients for I-frames and block motion vectors for P-frames. A frequency-temporal data structure is constructed. Starting from the blocks where the AC-coefficient energy and local inter-block DC-coefficient variance is small, the homogeneous volumes are enlarged by evaluating the distance of candidate vectors to the volume characteristics. Affine motion models are fit to volumes. Finally, a hierarchical clustering stage iteratively merges the most similar parts to generate an object partition tree as an output. The experimental results have shown that the proposed compressed domain video segmentation method provides the similar results as by using spatial domain process with much less computational complexity.

Index Terms

MPEG Video, Segmentation, Volume Growing

I. Introduction

Video object segmentation is one of the challenging tasks in video processing since it requires fusion of different modalities such as color, texture, motion, etc., and bridging the semantic gap between the numerical information embedded in the video and the human perception. Nevertheless, it has several important applications from the general video compression standards (MPEG-4) to specific event detection tasks. The segmentation information also enables indexing and content retrieval in video databases.

Conventionally, object segmentation is performed in the spatial-color domain, so called as *raw* data, using a pair of consecutive images at a time [1]-[5]. Since no motion information is available in the raw data, a pixel-wise motion field is often estimated by the optical flow or block-matching based approaches with an additional computational cost, which can be very high depending on the desired accuracy. We should also note that most commercial video systems makes heavy use of the compressed data especially when the data needs to be transmitted and stored. This becomes a constraint especially for the large scale systems. In such systems, the raw domain segmentation approaches cannot be performed until the compressed video has been decompressed first. Since the video is already encoded, it is computationally more convenient to process data in the compressed domain rather than decomposing it. Furthermore, the block structure of the compressed domain data drastically condenses the amount of data to be processed, and speeds up the processing time. In addition to the reduction of the computational complexity, there are several other advantages of imposing object segmentation in the compressed domain. For instance, the compressed video contains information about the spatial energy distribution within the image blocks as a result of the frequency decomposition. Therefore, several image attributes such as texture and gradient may be estimated easily without a tedious analysis. Most importantly, a block-wise translational motion information is embedded within the encoded sequence for the motion compensated frames, which can be used directly or as a prior information to guide more sophisticated motion estimation stages. The compressed domain analysis serve as an initial stage that steers the following uncompressed domain segmentation by providing fundamental information such as motion parameters and color properties to decrease the computational load of the further processing.

It has to be noted that, compressed domain analysis has limitations as well. The Discrete Cosine Transform (DCT) removes the spatial correlation among the pixels within a block, thus the precision of the segmentation degrades by the block dimension. Since the goal of motion compensation is to provide a good prediction but not to find the correct optical flow, the the motion vectors (MV) are often contaminated with mismatching and quantization errors. On top of that, the motion fields in MPEG streams are quite prone to quantization errors.

In contrast to the immense amount of work performed over uncompressed video (a review is given in [9]), only a few researchers have proposed the object segmentation algorithms in the compressed domain [6]-[13]. Some algorithms are even restricted in the DCT coefficients. For instance, Wang [13] proposed an algorithm to automatically detect faces where he uses skin-tone statistics, shape constraints, and energy distribution of the luminance DCT coefficients to locate the face position. Similarly, De Queiroz [8] proposed and algorithm to segment JPEG images into specific regions such as those containing halftones, text, and continuous-tone using the encoding-cost-map based on DCT coefficients. In a related work, Sukmarg and Rao [12] proposed a region segmentation and clustering based algorithm to detect objects in MPEG compressed video. Their segmentation algorithm consists of four main stages; initial segmentation using sequential leader and adaptive k-means clustering, region merging based on spatiotemporal similarities, foreground-background classification, and object detail extraction. However, this algorithm does not have a mechanism to handle the motion vectors of multiple P-frames. It requires several preset thresholds and the value of the sequential clustering threshold is crucial to determine the number of objects. Besides, k-means clustering method needs appropriate weights for the block coordinates, DCT coefficients, and motion information. A confidence measure based moving object extraction system was proposed by Zhang [14]. They proposed several confidence measures to improve motion layer separation. Their algorithm detects objects after a global motion compensation. Ji and Park [10] segmented dynamic regions based on the DCT coefficient similarity and true/false motion block classification. However, this method requires tracking of individual regions.

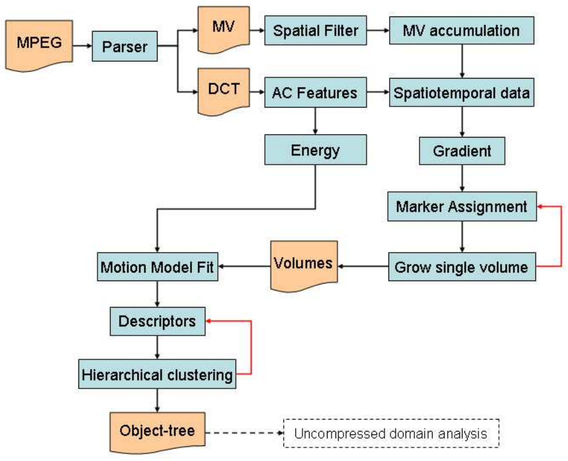

Fig. 1. Detailed flow diagram of the compressed domain segmentation algorithm. An hierarchical object tree with an associated quality assessment score is generated after the segmentation. This information may be used to guide the possible proceeding tasks, including the transcoding of video into MPEG-4.

Babu and Ramakrishnan [6], on the other hand, used only aggregated motion vectors.

Based on the above observation about the previous algorithms of compressed domain video segmentation, we develop a fast, automatic, compressed domain segmentation algorithm that fuses motion and frequency information. A flow diagram is shown in Fig.1. After parsing an MPEG video into the DCT coefficients and motion vectors, we construct a 3D frequency-temporal data structure using multiple Group of Pictures (GOP)'s that contains I and P frames between two scene-cuts. Each GOP is represented by a layer of vectors that correspond to blocks in an I-frame. Each vector consists of a number of selected DCT coefficients and a set of accumulated forward-pointing MV's within the GOP. Then, we grow volumes within the 3D data structure by starting from selected seed points. The volume growing gives the connected parts of video that have consistent DCT coefficients and motion parameters. The seed points are assigned as the blocks that have minimum texture and gradient in their local neighborhood to improve the likelihood of generating coherent volumes. For each volume, we determine a set of volume descriptors, including trajectorial motion, affine motion, color, representative DCT coefficients. As a final stage, we iteratively merge the similar volumes using their descriptors to obtain a hierarchical object-partition tree. We compute a validity score to assess the quality of the current segmentation results at each level of the object tree. The validity score indicates the optimal segmentation. As shown in the presented results, our method is robust towards the similarity threshold perturbations. It is macro-block accurate and computationally simple at the same time.

In the next section, we explain the MPEG parser. In section 3, we introduce the frequency-temporal data structure. In section 4, we give details of the volume growing. In section 5 and 6, we present the motion parameter estimation and the hierarchical volume clustering algorithms.

II. MPEG Parser

To obtain the DCT coefficients and motion vectors without fully decoding the input video, the parser retrace the encoding stages. We give brief explanation of the mechanisms of MPEG below.

The basic idea behind the MPEG video compression is to remove spatial redundancy within a video frame and temporal redundancy between video frames. DCT-based compression is used to reduce spatial redundancy. Motion-compensation is used to remove the temporal redundancy. The MPEG compression scheme converts a bitstream in terms of I (intra-compressed), P (forward predicted), and B (bi-directional predicted) frames. An I-frame is encoded as a single image, with no reference to any past or future frames. It stores the DCT information of the original frame. The P and B frames store the motion information and residues after motion compensation. Although I-frame provides no motion information, still color and texture information can be propagated to the P-frames by inverse motion compensation. A P-frame is encoded relative to the past reference frame. A reference frame is either a P or I-frame. All I-frames are divided into 16x16 pixel macroblocks. Each macroblock consists of four 8×8 luminance (Y) blocks and two 8x8 chrominance (U,V) blocks.

The block is first transformed from the spatial domain into a frequency domain using the DCT, which separates the signal into independent frequency bands. The DCT coefficients are correlated with spatial frequencies, thus, given that the different components have different importance, it can be used to remove the redundancy. The DCT is often used in signal and image processing due to its strong energy compaction property. The signal information tends to be concentrated in a few low-frequency components of the DCT, approaching the optimal Karhunen-Love transform for signals based on certain limits of Markov processes. In DCT, coefficient corresponding to the zero frequency decomposition is called as the dc and the remaining coefficients as ac parameters. The dc parameter indicate the average color within the macroblock for the given color channel. After the DCT transform, the data is quantized for further reduction. The quantization process can be regarded as dismissing the lower-order bits. The resulting data is then run-length encoded in a zigzag ordering to increase coding efficiency. The goal of motion compensation is to provide an approximate prediction for the macroblock. Motion-compensated prediction assumes that the current picture can be locally modeled as a translation of the pictures of some previous time. Each macroblock in a P-frame can be encoded either as an I-macroblock or as a P-macroblock. An I-macroblock is encoded just like a macroblock in an I-frame. A P-macroblock is encoded as a 16x16 area of the past reference frame, plus an error term. In the macroblocks where prediction is applied, the DCT is performed to the prediction errors instead of to the image samples and more the prediction errors are low and more the entropy coding is effective. The MPEG specifies how to represent the motion information for each macroblock of P-frames. It does not, however, specify how such vectors are to be computed. Due to the block-based motion representation, many implementations use block-matching techniques, where the motion vector is obtained by minimizing a cost function measuring the mismatch between the reference and the current block. Thus, the MPEG motion vectors does not necessarily correspond to the true motion but the best matching of macroblocks. The sequence of different frame types is called the Group of Pictures (GOP) structure. There are many possible I, P, B frame arrangements, often

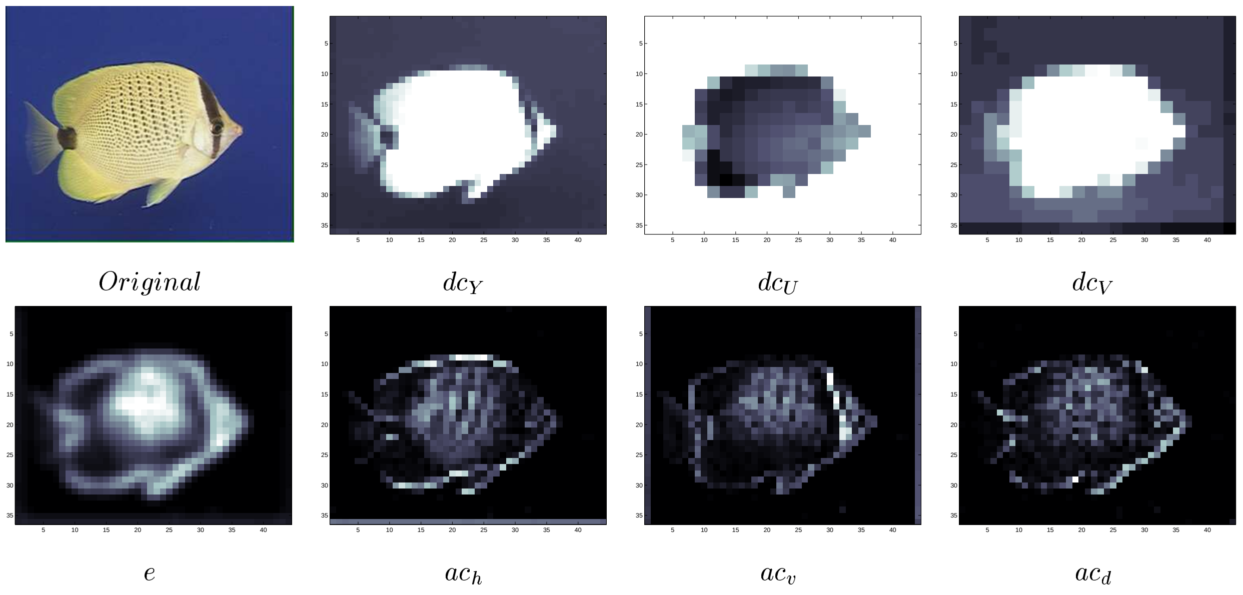

Fig. 2. A decoded I-frame and its corresponding layers in the frequency-temporal data FT.

ranging from 12 to 15 frames. The ratio of different type of frames in the GOP is determined by the nature of the video stream and the bandwidth requirement of the output stream.

To parse an MPEG video, we first chop the binary bitstream into bytes. At this point, all the DCT coefficients are in the quantized format. Thus, we apply an inverse quantization to find the integer valued DCT coefficients. We obtain the motion vectors after variable length decoding. We reconstruct the scan lines of macroblocks by indexing the DCT coefficients and motion vectors. The parsing process is computationally much simpler than the full decoding of MPEG video, which requires application of inverse DCT and motion compensation stages. On average, the parsing takes $3 \sim 10\%$ of the decoding time [7]-[11] for a GOP.

After we parse the data, we assemble the DCT coefficients and motion vectors of I-frame and P-frames of a GOP into a frequency-temporal data, which will be used in the segmentation.

III. Frequency-Temporal Data Structure

The frequency-temporal (FT) data structure contains the DCT coefficients and motion vectors of the corresponding GOP or a set of multiple consecutive GOP's within a video shot, which is a part of the video between two scene-cuts. Therefore, the content of the video shot, i.e. the number of objects and their properties, are consistent for the set of GOP's. The FT data has three dimensions; spatial horizontal and vertical, and time. Each element of the FT corresponds to a feature vector, $f(t, m, n)$, that represents the attributes of an $M \times M$ macroblock, where t is the index of the GOP's, and m, n are the indices of macroblocks. The vectors belong to the same GOP constitute a temporal layer.

The feature vector components consists of the dc parameters of the I-frame (for all Y,U,V channels), a reduced set of the ac parameters (for Y-channel only), a spatial energy term e, and the accompanying forward-predicted motion vectors obtained from the P-frames. The dc and ac components only exist for the I-frame in the GOP. The dc parameters represent the average color of the block, thus they can be considered as a subsampled I-frame by a factor of 8 as shown in Fig. 2. However, not all the color channels are encoded in the same precision. It is often preferred in the compression of the color data for MPEG that the chrominance channels have half the resolution of the luminance channel, basically due to the fact that human visual perception is more sensitive to the luminance variance than the chrominance.

The DCT transform of an $M \times M$ image block is defined as

$$dct(u,v) = \frac{2}{M} \sum_{x=1}^{M} \sum_{y=1}^{M} I(x,y) \cos \frac{\pi u(2x+1)}{2M} \cos \frac{\pi v(2y+1)}{2M} \tag{1}$$

where u and v are the horizontal and vertical frequencies $(u, v = 1, .., M)$, and $I(x, y)$ is a pixel. In regular MPEG syntax, the block size is often $M = 8$. For a block in which the spatial texture is smooth, most of the higher indexed DCT coefficients have lower values and they reduce to zero after the quantization stage. Another key observation is that the higher coefficients of the DCT is sensitive to the pattern shifts unlike the magnitude of the Discrete Fourier Transform, e.g. the spatially shifted versions of the same pattern will have different higher order coefficients. Since object movement will cause the shifted versions of the same pattern between the I-frames, the higher order coefficients, in fact, will be different. Moreover, the ordinary image noise, such as salt-and-pepper, causes changes in the higher order terms. Thus, it only make sense to include a set of certain number of lower indexed DCT coefficients into the feature vector. In our simulations, we chose 3 main such sets of horizontal, vertical, and diagonal ac parameters;

$$\begin{aligned} ac_h &= \frac{1}{K-1} \sum_{i=2}^{K} dct(i,0) \qquad (2)\\ ac_v &= \frac{1}{K-1} \sum_{i=2}^{K} dct(0,i) \\ ac_d &= \frac{1}{K-1} \sum_{i=2}^{K} dct(i,i) \end{aligned}$$

where $K < M$. We also define an energy term e to represent the amount of the spatial variance within the macroblock

$$e = \frac{1}{M^2} \sum_{u=1}^{M} \sum_{v=1}^{M} dct(u,v) \tag{3}$$

There is a strong correlation between the energy term and the accuracy of the motion estimation. A block-matching based motion estimation of a macroblock that contains smooth texture thus having a lower energy term, may not be as accurate as estimation for a high energy block since block matching may fail for smooth image regions. The original MPEG motion vectors are prone to errors due to the block-matching and quantization. Therefore, we apply a spatial filter to prune the extremities of motion vector field. We first estimate motion vectors of the intra-coded macroblocks of the P-frames, for which no motion vectors are assigned in the compressed data. We compute the mean of the motion vectors within a local window and set it as the motion vector of the intra-coded macroblock.

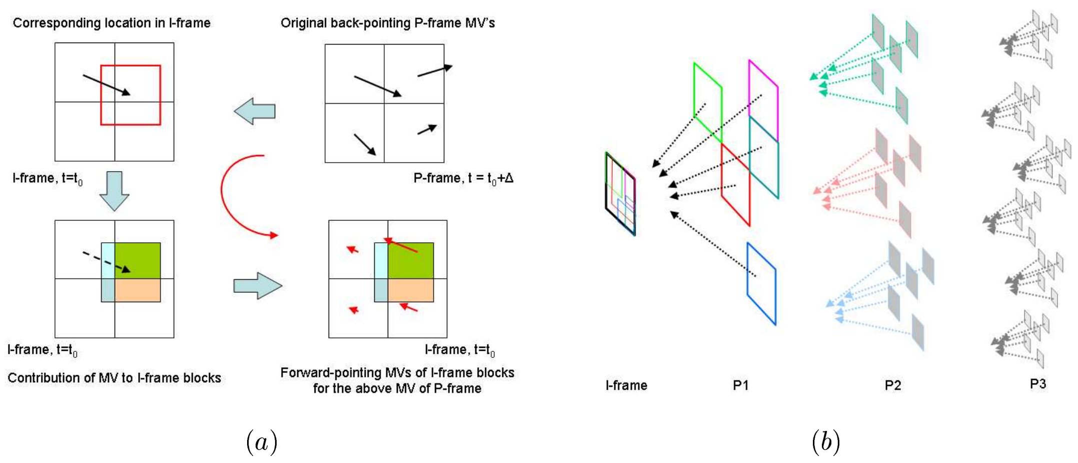
Corresponding location in I-frame
Original back-pointing P-frame MV's
I-frame, t=t0
P-frame, t = t0+Δ
I-frame, t=t0
I-frame, t=t0
Contribution of MV to I-frame blocks
Forward-pointing MVs of I-frame blocks for the above MV of P-frame
I-frame
P1
P2
P3
(a)
(b)

Fig. 3. (a) One P-frame motion vector contributes at most four I-frame macroblocks. (b) Forward motion projection exponentially branches out for more than one P-frames.

We then convolve the refined motion vectors with a Gaussian template to minimize the singularities. There is a trade of between filtering and boundary accuracy; although an application of Gaussian filter may smear the object boundaries, not removing the singularities causes more significant segmentation errors. We observed that a 3×3 window gives the best results for the various sequences that we tested. The feature vector at each point of the FT then defined as

$$f(t, m, n) : [dc_Y \; dc_U \; dc_V \; ac_h \; ac_v \; ac_d \; e \; mv_x \; mv_y]^T_{t,m,n}. \tag{4}$$

Note that, 15 frames of 352×288 spatial resolution color video needs $352 \times 288 \times 15 \times 3$ pixels in the raw data domain, however, the corresponding frequency-temporal data has only $44 \times 36 \times 9$ components, which is equal to a reduction of 320:1 in the data size.

One problem of integrating the motion information into the feature vector is that the motion vectors of P-frames are back-predicted. In other words, for an I and P frame pair, only the blocks in the first P-frame have their motion vectors pointing the most similar placements in the I-frame. In other words, motion vector for an I-frame block does not exists. We convert the motion vectors of the P-frame for the I-frame blocks as illustrated in Fig. 3-a. We find a motion vector for an I-frame block that points the matching region in the following P-frame. After forward projecting each P-frame block to the I-frame, we compute the overlapping areas between the original and projected blocks. We update the I-frame motion vectors of the overlapped I-frame blocks with respect to the ratio of the overlapping area to the covered area of this block after all the vectors of P-frame are projected. It is obvious that for an I-frame block that is entirely covered by the projected P-frame blocks, the motion vector prediction is more accurate than another frame that is partially covered.

The above forward projection is only applied to the immediate adjoint P-frame due to th fact that the accuracy of the motion prediction exponentially degrades as illustrated in the Fig. 3-b. Besides, the accuracy of the forward projected motion vector is limited to the accuracy of the original vectors, which may be inaccurate at the beginning.

One way to obtain the forward motion vectors is to partially decode both I and P frames and compute optical flow or motion field between them, which is computationally prohibitive. The generation of the FT data takes $1 \sim 2$msec. on average for a GOP.

IV. Volume Growing

We grow 3D volumes within the frequency-temporal data starting from the seed points. These volumes may extend between multiple GOP's. A temporal slice of a volume gives the corresponding region in a GOP. Volume growing associates the feature-wise similar FT points, macroblocks, into a coherent segment. Since we start grouping of similar points from a seed point, it is required that the initial seed point is a proper representative of its 3D local neighborhood. The points that have lower variances in their feature values are candidates. The energy term gives valuable information about the 2D variance. To enforce the continuity on the temporal dimension, we choose points that have minimum energy in both frequency and temporal dimensions instead of only frequency $(\arg\min f(m,n,t)\{e\})$ as

$$f(t,m,n)_{seed} = \arg\min_{f(m,n,t)} \sum_{i,j=-1}^{1} \sum_{j=-1}^{1} f(t+k, m+i, n+j)\{e\} \tag{5}$$

After we select a seed point, we initialize a volume using the feature vector of the seed point, that is, the components of the seed point becomes the components of the new volume. We define an active boundary of points that keeps account of the newly added points to the volume. At the beginning the active boundary has only the seed point inside. Then, we check the immediate neighboring points of the active boundary points. We evaluate 6-neighboring points in all 3 directions. Evaluating the points in temporal dimension enables imposing the assumption that the regions belongs to an object overlap between the consecutive GOP's. However, in case of a GOP contains a large number of P-frames, the time distance between the I-frames of two consecutive GOP's will increase accordingly, and as a result the overlapping assumption may be weaken. Favorably, most MPEG GOP's usually consist of 2-15 P-frames. We compare the feature vectors f of the neighboring points with the volume feature vector v. A distance metric is given as

$$\delta(v,f) = \omega_{dc}\delta_{dct}(v,f) + \omega_{ac}\delta_{ac}(v,f) + \omega_{mv}\delta_{mv}(v,f) \tag{6}$$

where ω_{dc}, ω_{ac} and ω_{ac} regulate the contribution of the DCT coefficients and motion vector distances, which are obtained as

$$\delta_{dc}(v,f) = |v\{dc_Y\} - f\{dc_Y\}| + |v\{dc_U\} - f\{dc_U\}| + |v\{dc_V\} - f\{dc_V\}|, \tag{7}$$

$$\delta_{ac}(v,f) = |v\{ac_h\} - f\{ac_h\}| + |v\{ac_v\} - f\{ac_v\}| + |v\{ac_d\} - f\{ac_d\}|, \tag{8}$$

$$\delta_{mv}(v,f) = \sqrt{(v\{mv_x\} - f\{mv_y\})^2 + (v\{mv_x\} - f\{mv_y\})^2}. \tag{9}$$

The above weights can be adapted to the given content. We observed that the higher values of the DCT distance provides more accurate segmentation in case of the motion is insignificant (e.g. for *Akiyo* and other head-and-shoulder sequences). In case there exists fast moving objects with multiple featured regions, the motion distance carries more discriminating information of objects.

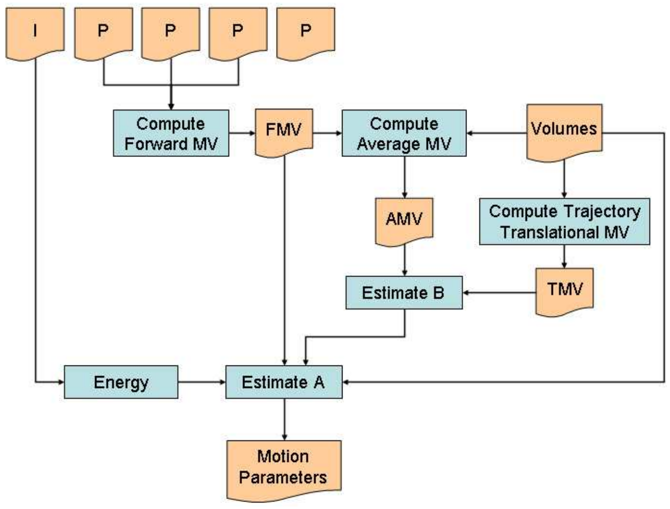

Fig. 4. Affine motion parameters are fitted using translational motion information.

We apply a threshold ϵ to the above color distance. The threshold determines the precision of the segmentation process, thus the average size of the volumes. Automatic adaptation of the distance threshold require evaluation of the segmentation results. We overcome this problem by fusing the similar volumes in the hierarchical clustering stage. Therefore, any threshold value that prevents from under-segmentation would suffice for our purpose.

In case the distance is less than the threshold $\delta(v, f) < \epsilon$, the vector f is included in the volume The volume feature vector is updated by the averaged means of the corresponding components. The new point is assigned as an active boundary. After a volume is grown, all the vectors of the volume is removed from the FT. The seed selection and volume growing process is iterated until no more point remains in the FT. As a post-processing stage, the volumes that have negligible size (e.g. 1-4 blocks) are removed and the remaining volumes are inflated to fill up the empty space.

The seed selection is a relatively intensive task since it involves a search for the minimum. One way to speed it up is to separate minimum search in the layers, i.e. the local minimum in the current 2D layer is searched and a volume in 3D data is grown, with a twist that the next seed is searched not in the previous 2D layer but in the following layer. The seed selection and volume growing take $0.8 \sim 1.5 msec$ for a GOP on average.

V. Motion Parameters

After volume growing, we have the parts of the FT that are consistent in terms of their DCT coefficients and translational motion distributions. The next task is to fit a motion model to each volume. We accomplish this by estimating the affine motion parameters of the regions of a volume in the corresponding temporal layers then averaging the set of individual parameters over all of the layers. Thus, we solve the notorious region of support problem of motion segmentation by using the segmented regions of the temporal layers. We model the layer-wise

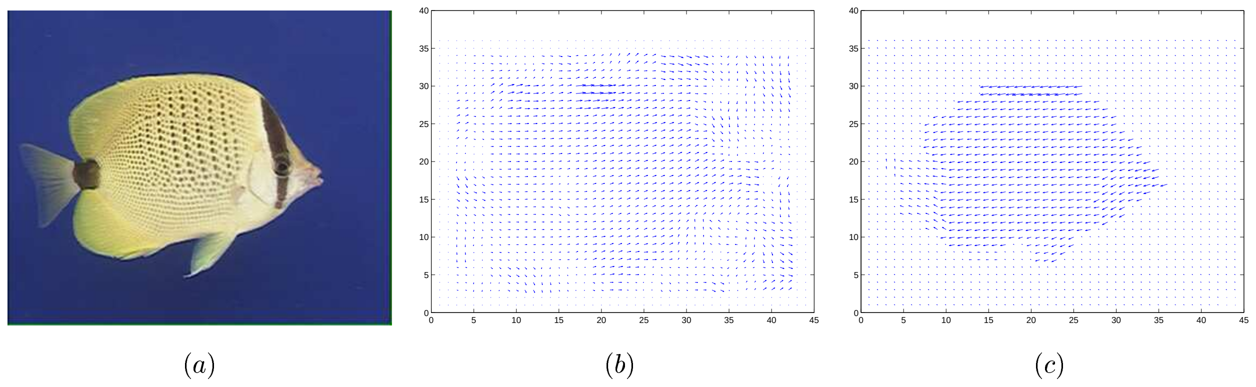

Fig. 5. (a) A frame from *Bream*, (b) original MPEG motion vectors interpolated for 8×8 blocks, (c) motion vectors after parameter estimation.

motion by a set of affine motion parameters A, B

$$8\begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} mv_x \\ mv_y \end{bmatrix} = Ax + B = \begin{bmatrix} a_1 & a_2 \\ a_3 & a_4 \end{bmatrix}\begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \end{bmatrix} \tag{10}$$

where $[m, n]^T$ is the block indices. The constant multiplier 8 converts the block indices to spatial coordinates in which the original motion vectors are measured.

We estimate translational motion $b = [b_1, b_2]^T$ for each volume at every temporal layer. We have two translational motion information sources; one is the average of the motion vectors within the region and the second is the trajectory displacement. Trajectory of a volume is defined as the set of layer-wise representative coordinates that can be chosen as the center-of-mass of the corresponding regions. Trajectory is calculated by averaging the coordinates of the points belong to the volume in a layer. After finding the trajectory coordinates, we take difference to determine trajectory displacement. We assign the translational motion b as the mean of the average motion and the trajectory displacement vectors. For a region that consists of K blocks in a temporal layer, we accumulate the motion vectors $[mv_x^k, mv_y^k]^T$ and its originating coordinates $[m_i, n_i]^T$ as

$$8\left[\begin{array}{c|c|c} m^1 & ... & m^K \\ n^1 & ... & n^K \end{array}\right] + \left[\begin{array}{c|c|c} mv_x^1 & ... & mv_x^K \\ mv_y^1 & ... & mv_y^K \end{array}\right] - \left[\begin{array}{c|c|c} b_1 & ... & b_1 \\ b_2 & ... & b_2 \end{array}\right] = A\left[\begin{array}{c|c|c} m^1 & ... & m^K \\ n^1 & ... & n^K \end{array}\right]$$

$$Y_{(2\times K)} = A_{(2\times 2)}X_{(2\times K)} \tag{11}$$

where only unknown is the matrix A since we already estimated B, and the left hand side of the above equation Y is known. X is an $2 \times K$ matrix, therefore $A = Y/X$ is the solution in the least squares sense to the overdetermined system of equations $Y = AX$. The effective rank R is determined from the QR decomposition with pivoting. A solution A is computed which has at most R nonzero components per column. We compute the affine motion parameters A, B for every frame of the volume. Motion parameter estimation enables to refine motion field as illustrated in Fig. 5. This process takes $8 \sim 10ms$ for a GOP.

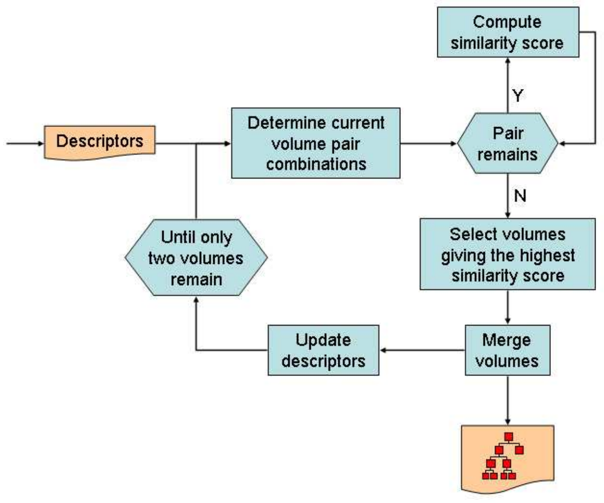
Descriptors
Determine current volume pair combinations
Pair remains
Y
N
Compute similarity score
Select volumes giving the highest similarity score
Merge volumes
Update descriptors
Until only two volumes remain

Fig. 6. At each iteration of hierarchical clustering, two most similar volumes are merged.

VI. Hierarchical Clustering

The segmentation algorithm generates volumes, their attributes, and information about how these volumes can be merged. Since human is the ultimate decision maker in analyzing the results of video segmentation, it is necessary to provide the segmentation results in an appropriate format to user or for further analysis and application specific merging. The object tree generated as a result of the hierarchical clustering satisfies this requirement.

We cluster the segmented volumes into objects using their descriptors such as motion parameters and DCT coefficients. Clustering can be done either by hierarchical or partitional approaches. Hierarchical methods produce a nested series of partitions while a partitional clustering algorithm obtains a single partition of the data. We adapt a hierarchical clustering technique by merging the volumes in a fine-to-coarse manner.

Clustering starts with the volumes generated after the volume growing, thus the initial number of volumes varies from sequence of sequence. At each iteration of the hierarchical clustering, we merge the pair having the most similar parameters as shown in Fig. 6. We define a similarity criteria between volumes as

$$s(v_i, v_j) = 1 - \frac{1}{c_S} \sum_t \left(c_R |A_{i,t} - A_{j,t}| + c_T |B_{i,t} - B_{j,t}| \right) \tag{12}$$

where t is the temporal layers both volumes are visible, c_S is a normalization constant, i.e. maximum distance. The mixture constants are set as $c_T \gg c_R$ to take into account of the fact that a small change in the rotation/scaling parameters can lead to much larger difference in the modeled motion field than the translation parameters. We update the descriptors, i.e. the motion parameters of the volumes, accordingly after each merge. Hierarchical clustering is iterated until there are only two volumes remain.

At each level of the clustering algorithm, we evaluate whether the chosen volume pair is a valid merge. We keep track of the change in the similarity term. The sudden drops and small values of the similarity score indicate an

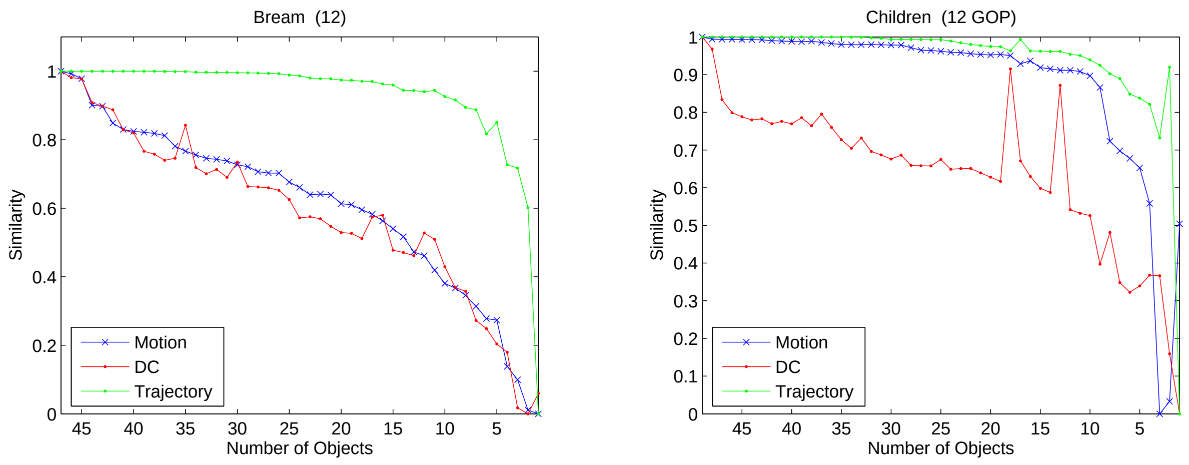

Fig. 7. Motion, trajectory, and DCT coefficient based similarity scores obtained of the merged pairs in the hierarchical clustering.

invalid merge. We should note that the consistency of this score depends the definition of the similarity. We observed that the motion parameter based similarity scores are robust, i.e. it gives a smoother and monotonicaly decreasing scores while providing accurate clustering results as shown in Fig. 7. The trajectory based motion similarity score is found to be not as consistent as the motion parameter metric score since it disregards the rotation and sensitive to the shape of the object, e.g. larger objects tend to have less descriptive trajectories due to the averaging of the positions of its member blocks. On the other hand, the dc coefficient based similarity causes wrong merges since it disregards the motion information.

Alternatively, we define a cluster validity score α_L to provide an answer to the basic question of clustering; ”what should be the optimum number of objects?” as

$$\alpha_L = \sum_{i=1}^{L} S(v_i) \sum_{f_{i,j} \in v_i} \left[\sigma^2(f_{i,j}\{mv_X\}) + \sigma^2(f_{i,j}\{mv_Y\})\right] \tag{13}$$

where $f_j\{mv_X\}, f_j\{mv_Y\}$ are the horizontal and vertical motion vector components for the points belong to the volume v_i. $S(v_i)$ is the total number of points in the volume v_i. We use $\sigma^2(.)$ as the variance operator. L is the current number of volumes at the hierarchical clustering level. In other words, the validity score corresponds to the total variance of the motion parameters within the all segments. The validity score gets lower values for the better fitted segments. Thus, by evaluating the minima of this score, we can determine the correct cluster number automatically. As shown in Fig. 8, the total segment variance suddenly increases in case of an invalid merge. For the *traffic* sequence, there are main 3 objects; vehicle on the left, vehicle on the right, and the stationary background. As visible, the validity score jumps when we merge 3 objects to get 2 objects, which means that the merge is incorrect and there should be only 3 objects, which is also presented in Fig. 13. Similarly, for the *table tennis* sequence, the validity score estimates the optimum number of clusters as 2; the ball and the background, since only the ball was moving for that segment of GOP's. In addition, the object tree includes the table and the arm as well as shown in Fig. 14.

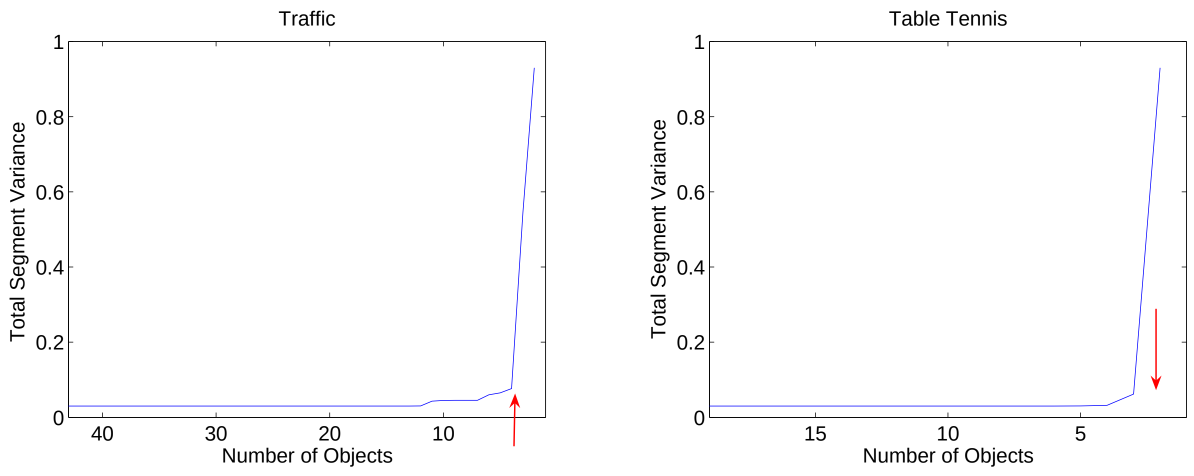

Fig. 8. Validity score α_L for two sequences, *Traffic* and *Table Tennis*. Arrows indicate invalid merges.

VII. Experiments

To test the proposed algorithm, we used different number of GOP's (2-12), which have various number of P-frames ranging up to 15. In case of using 12 GOP's with 1 I-frame, 4 P-frames, and 3 B-frames each, which is a common configuration, this corresponds to a synchronous segmentation of 96 frames of the original video. When we construct the frequency-temporal data structure, we employed only the first P-frame after the I-frame to obtain the forward-predicted motion vectors. We observed that although the inclusion of the following P-frames increases the computational load, it does not improve the segmentation.

We set the volume growing thresholds once and used the same thresholds for all test sequences. As we explained before, any threshold value that prevents from under-segmentation would suffice for our purpose since the clustering stage is designed to merge over-segmented volumes. The distance threshold assigned to a higher value for inter-layer vector difference than the intra-layer vector difference to exploit the inter-layer growing for the sequences that have fast motion.

We removed the volumes that are smaller than 4 points, which correspond to a 16$\times$16 raw image region, to prevent from having an excessive number of volumes. The *initial* images in Figures 9-14 show the segmentation results for an I-frame of the first GOP layer for each test sequence after the volume growing stage. As visible, the objects that have similar DCT coefficients and motion vectors are accurately detected even at the coarse block resolution, which shows the effectiveness of the compressed domain segmentation.

Figure 9 shows an I-frame, the initial segmentation and the object tree for the *akiyo* sequence, where both of the validity and motion parameter similarity based metrics indicate the optimum number of clusters is 2 (the head and the background since head had the most discriminating movement for that group of GOPs). Figure 10 presents the segmentation results for the *Lab* sequence. The validity score suddenly changes at the clustering level 2, which shows there should be two clusters, i.e. person and background. Figure 11 gives results for the *bream* sequence.

TABLE I

COMPUTATIONAL LOAD OF A GOP (FOR A 352×288 VIDEO - 44×36 BLOCKS)

Parsing	$0.2 \sim 0.7$ msec
FT Generation	$0.5 \sim 1$ msec
Seed Selection & Volume Growing	$2 \sim 3$ msec
Motion Parameter Estimation	$8 \sim 10$ msec
Hierarchical Clustering	$2 \sim 4$ msec

We observed that the validity score gives the optimum at the clustering level 3 due to the fact that the upper fin of the fish has different movement. Ideally, the optimum number should be 2, however, not all regions of the fish has the same motion.

We show an I-frame from the *children* sequence and its object-tree in Figure 12. The computed validity scores changes suddenly after the clustering level 3, which indicates the optimum segments are the moving head of the boy on the left, the body of the boy on the right, and the background. The segmentation results for a GOP of the *traffic* is given in Figure 13. As shown in Figure 8, the validity score for this video jumped when we tried to merge 3 remaining objects. We observed that the validity score for the sequence increased at level 2. These results confirm with the motion existing in the scene, and proves effectiveness of the proposed algorithm even if the objects are small in comparison to the frame size.

As visible in these results, the motion parameter based similarity measure can detect the small motion variances. Although a fast moving single small object may invalidate the overlapping regions assumption and appear as separate objects in different layers, we observed that, for the moderate motion sequences, the trajectories are continuous and segmented region boundaries are accurate. We also concluded that the segmentation process is not sensitive to the minor threshold perturbations which gives additional flexibility. The proposed algorithm is faster than real time video playing speed. The total segmentation time including the MPEG parsing varies in the range of $10 \sim 20ms$ for a GOP on a P4 3Ghz platform depending on the number of initial objects after the volume growing as shown in Table I. Most computations are involved in motion parameter fitting stage. Favorably, the speed is not influenced by the complexity of the motion. Since the GOP's corresponded to the 8 frames of the original raw data in this computational analysis, the proposed algorithm achieves on average $0.9 \sim 2ms$ processing speeds per frame.

VIII. SUMMARY

We present a real-time object segmentation method for MPEG encoded video. Our method fuses the motion and frequency information. After parsing an MPEG video into the DCT coefficients and motion vectors, we construct a 3D frequency-temporal data structure and grow volumes by starting from selected seed points. The volume growing gives the connected parts of video that have consistent DCT coefficients and motion parameters. For each volume, we determine a set of volume descriptors, including trajectorial motion, affine motion, color, representative DCT coefficients. At the final stage, we iteratively merge the similar volumes using their descriptors to obtain a

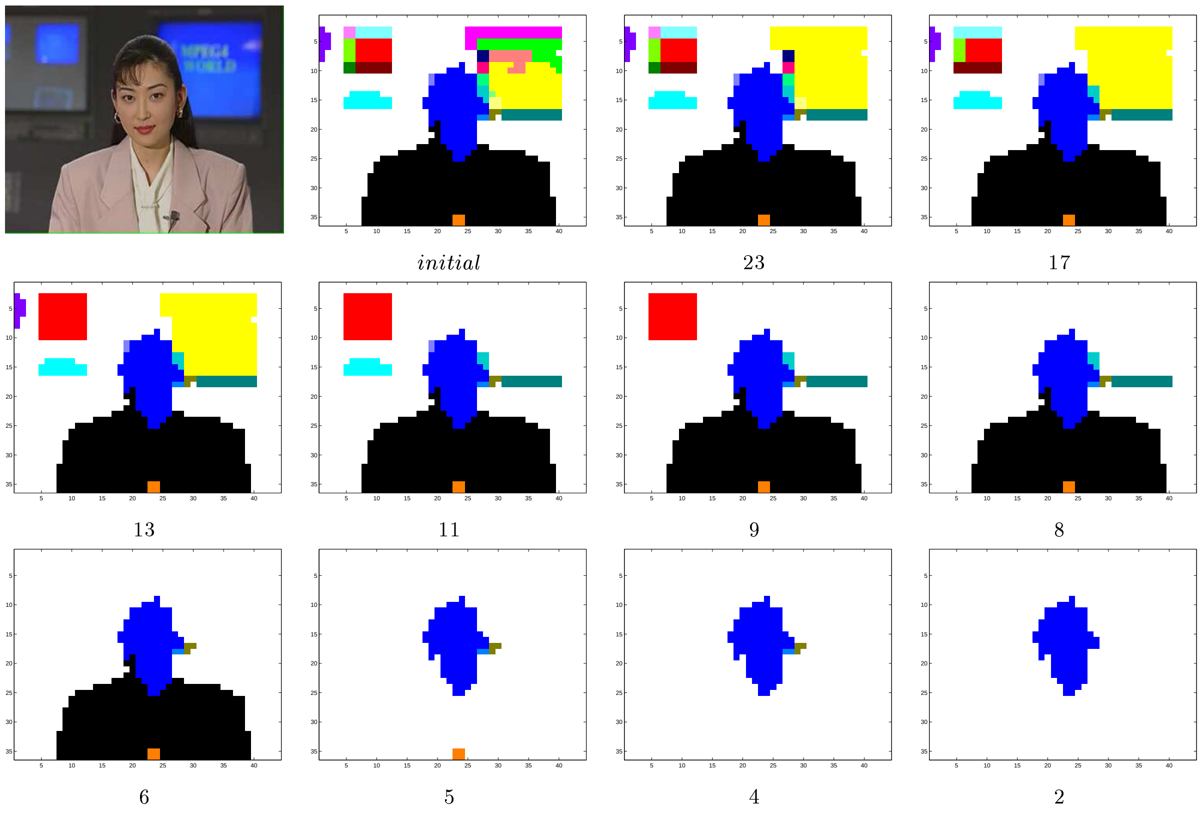

Fig. 9. An I-frame from *Akiyo*, and the segmentation results at the corresponding clustering levels including the background. Both validity and similarity based metrics indicate the optimum number of clusters is 2.

hierarchical object-partition tree. We compute a validity score to assess the quality of the current segmentation results at each level of the object tree. As shown in the presented results, our method gives accurate segmentation maps.

As future work, we plan to use the compressed domain processing as a precursor to improve the uncompressed domain segmentation.

References

[1] P. Bouthemy and E. Francois, "Motion segmentation and qualitative dynamic scene analysis from an image sequence", Int. Journal Computer Vision, no 10, 157-187, 1993

[2] F. Bremond and M Thonnat, "Tracking multiple nonrigid objects in video sequences", IEEE Trans. Circuit, Syst. Video Technol., no 8, 585-591, 1998

[3] T. Meier and K. N. Ngan, "Automatic segmentation of moving objects for video object plane generation", IEEE Trans. Circuit, Syst. Video Technol, no 8, 525-538, 1998

[4] F. Moscheni, S. Bhattacharjee, M. Kunt", "Spatiotemporal segmentation based on region merging", IEEE Trans. on PAMI, vol. 20, no 9, 897-915, 1998

[5] G. Wu and T. Reed, "Image sequence processing using spatiotemporal segmentation", IEEE Trans. on circuits and systems for video technology, vol. 9, 798-807, 1999

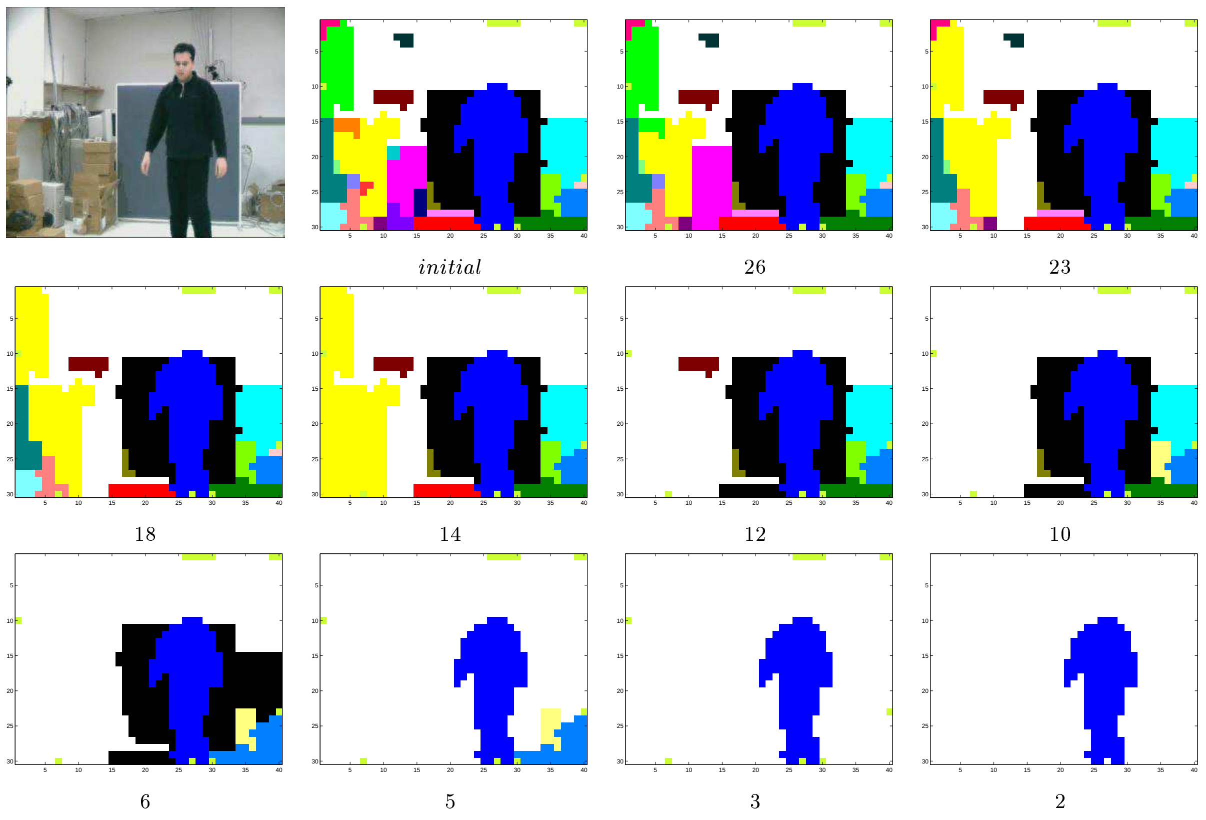

Fig. 10. An I-frame from *Lab*, and the segmentation results at the corresponding clustering levels. Validity metric jumped at level 2, which indicates there should be two clusters, i.e. person and background. Different volumes are randomly colored.

[6] R. Venkatesh Babu and K. Ramakrishnan, ”Compressed Domain Motion Segmentation for Video Object Extraction”, IEEE International Conference on Acoustics, Speech, and Signal Processing, Florida, 2002

[7] F. Cavalli, R. Cucchiara, M. Piccardi, and A. Prati, ”Performance Analysis Of Mpeg-4 Decoder And Encoder”, International Symposium on Video/Image Processing and Multimedia Communications, Croatia, 2002

[8] R. DeQueiroz, Z. Fan, and T. Tran, ”Optimizing Block Thresholding Segmentation for Multilayer Compression of Compound Images”, IEEE Transactions on Image Processing, 2000

[9] F. Porikli and Y. Wang, ”Automatic Video Object Segmentation Using Volume Growing And Hierarchical Clustering”, Journal of Applied Signal Processing, special issue on Object-Based and Semantic Image and Video Analysis, January 2004.

[10] S. Ji and H. W. Park, ”Image Segmentation Of Color Image Based On Region Coherency”, IEEE Transactions on Image Processing, 1998

[11] F. Kossentini and Y. Lee, ”Computation-Constrained Fast MPEG-2 Video Coding”, IEEE Signal Processing Letters, vol. 4, n. 8, 224-226, 1997

[12] O. Sukmarg and K. Rao, ”Fast algorithm to detect and segmentation in MPEG compressed domain”, Proceedings of IEEE Region 10 Technical Conference Malaysia, 2000

[13] H. Wang and S.F.Chang, ”Automatic face region detection in MPEG video sequences”, Electronic Imaging and Multimedia Systems, SPIE Photonics China, 1996

[14] R. Wang, H.J. Zhang, and Y.Q. Zhang, ”A Confidence Measure Based Moving Object Extraction System for Compressed Domain”, IEEE International Symposium on Circuits and Systems, Switzerland, 2000

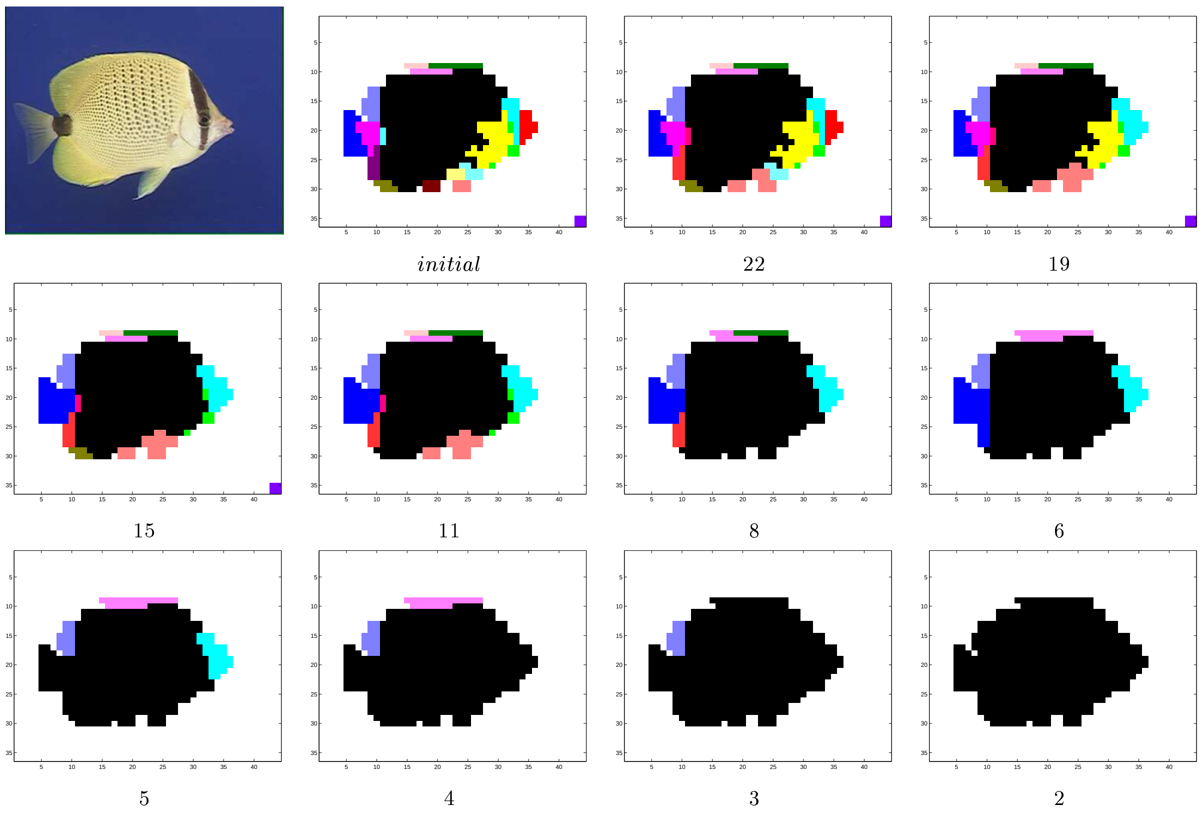

Fig. 11. An I-frame from *Bream*, and the segmentation results at the corresponding clustering levels. Validity score claims that the segmentation is optimum at the level 3 due to the fact that the upper fin of the fish has different movement.

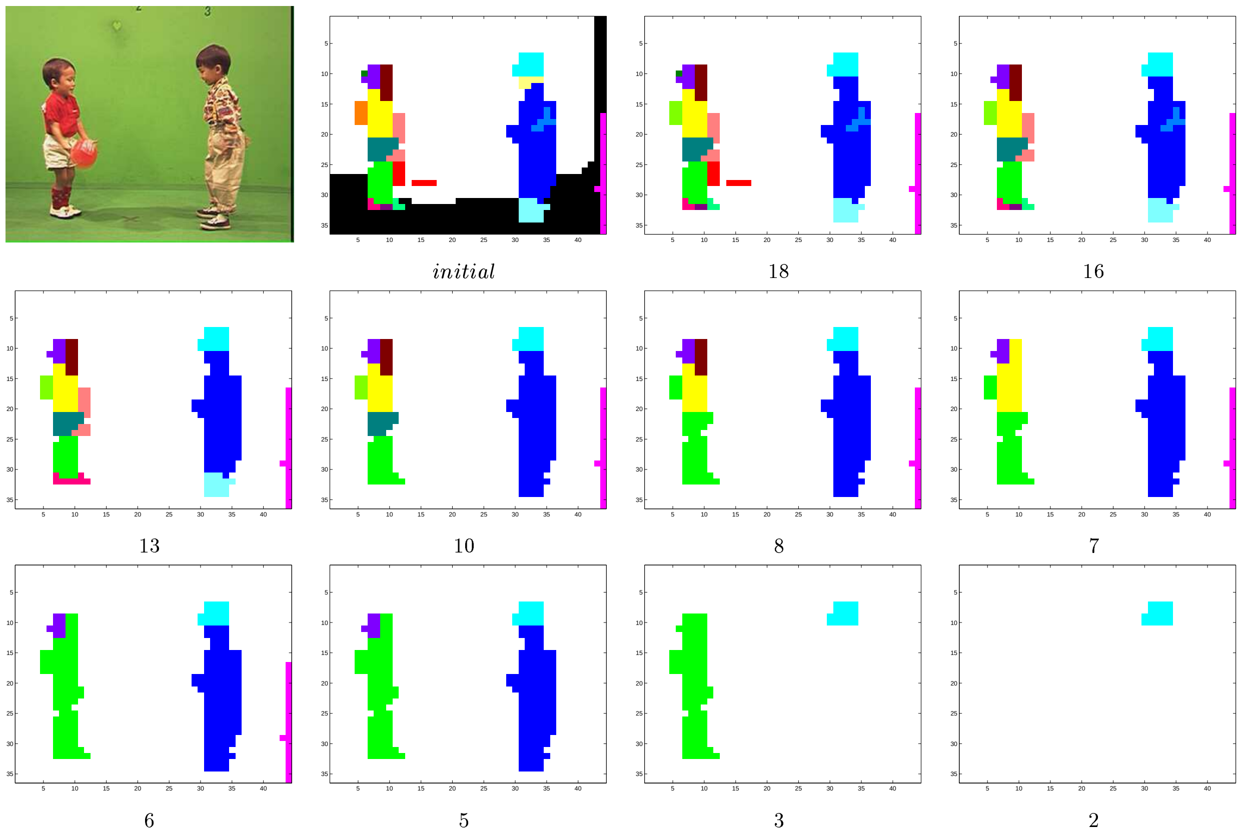

Fig. 12. An I-frame from *Children*, and the segmentation results at the corresponding clustering levels. Validity score indicates the optimum number of clusters should be 3; moving head of the boy on the left, body of the boy on the right, and background. However, our system enables the end user to choose any level in the object tree.

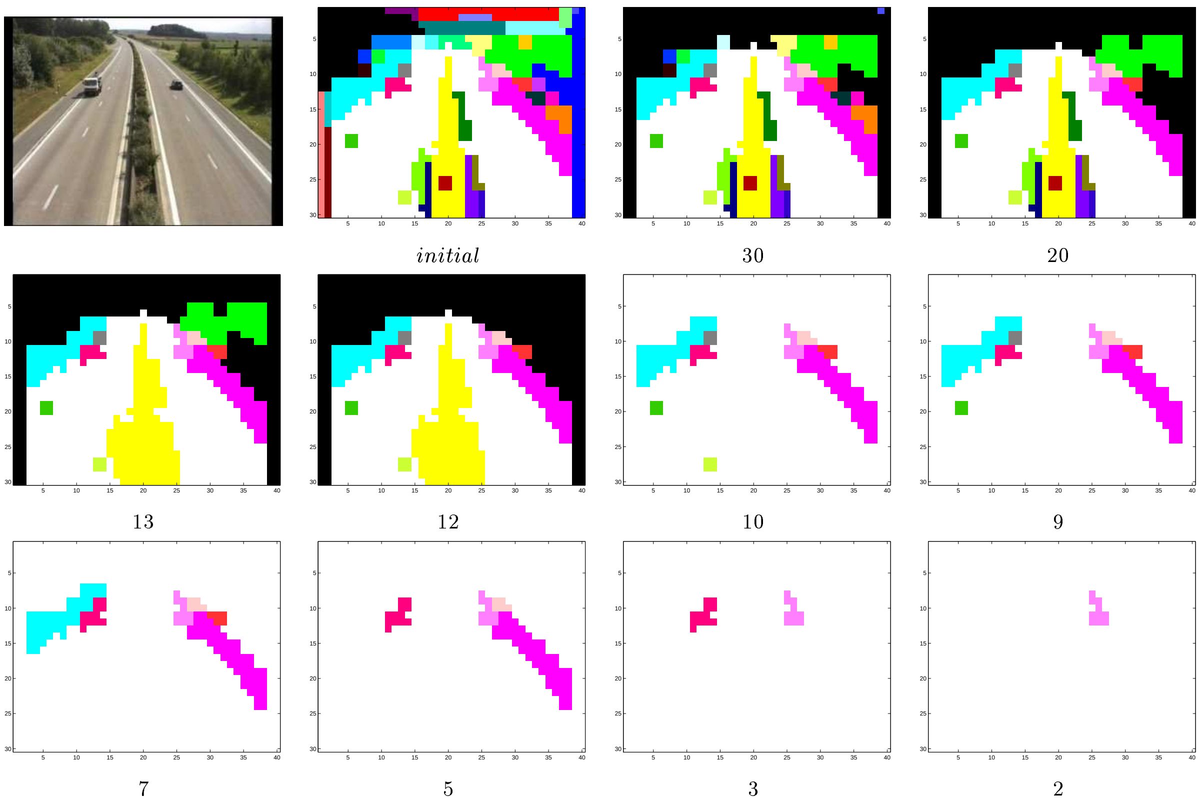

Fig. 13. An I-frame from *Traffic*, and the segmentation results at the corresponding clustering levels. The validity score indicates the optimum segment number should be 3 including the background.

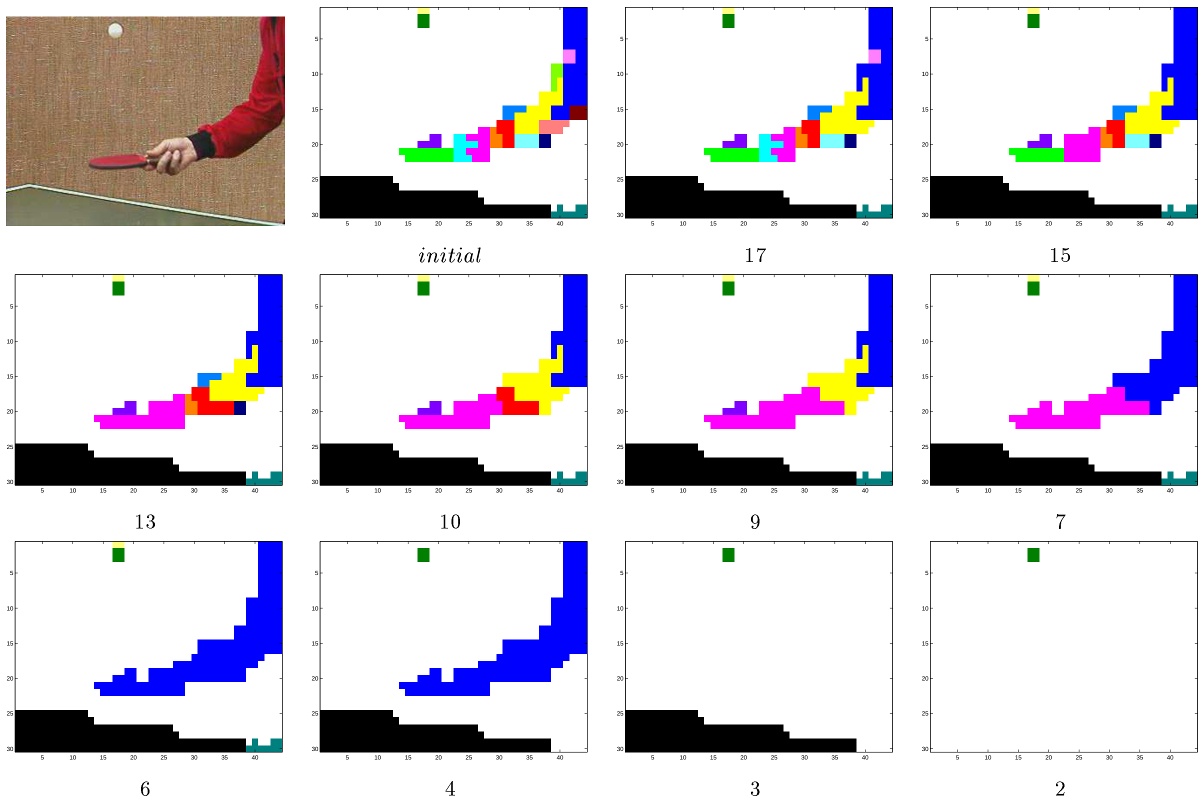

Fig. 14. An I-frame from *Table Tennis*, and the segmentation results at the corresponding clustering levels. The validity score for this sequence jumped at level 2.