Video Segmentation and Camera Motion Characterization Using Compressed Data

Ruggero Milanese Frédéric Deguillaume Alain Jacot-Descombes

Computer Science Department
University of Geneva, Switzerland

ABSTRACT

We address the problem of automatically extracting visual indexes from videos, in order to provide sophisticated access methods to the contents of a video server. We focus on two tasks, namely the decomposition of a video clip into uniform segments (shots), and the characterization of each shot by camera motion parameters. For the first task we use a Bayesian classification approach to detecting scene cuts by analyzing motion vectors. For the second task a least-squares fitting procedure determines the pan/tilt/zoom camera parameters. In order to guarantee the highest processing speed, all techniques process and analyze directly MPEG-1 motion vectors, without need for video decompression. Experimental results are reported for a database of news video clips.

Keywords: Shot detection, Camera motion, Video archival, Content-based retrieval, Bayesian classification, MPEG

1. INTRODUCTION

A clear trend in the video production and television broadcasting market concerns the use of digital video, both for production (aquisition) and postproduction (editing, archival/retrieval). In order to cope with the huge amount of data of a typical studio, compression schemes must be considered. Although proprietary compression algorithms have been proposed by some vendors, the use of standards is highly desirable. According to perceptual quality requirements, high-bitrate MPEG-2 compression (delivering of the order of 50 Mb/s) is appropriate for broadcasting, and a few commercial products including video servers and compression boards have already appeared on the market. However, much lower bitrates appear sufficient for a number of postproduction operations, including indexing, browsing, previewing, and low-cost editing. One solution would be to transcode on demand the high-bitrate video into a lower one. However, the delay, network cost, and quality loss introduced by this approach justify the use and storage of a separate video stream, compressed using the MPEG-1 standard at approximately 1.5 Mb/s.

In this paper we describe a number of techniques for processing MPEG-1 compressed video clips in order to facilitate their use and access through a database. In particular, we aim at extracting a number of indexes that enable a user to compose user-friendly queries to a database, to rapidly display the matching terms, and reuse the desired clips for editing or for referencing the corresponding high-bitrate version for direct broadcasting.

A number of operations must be performed to achieve the above goals. First, a clip must be segmented into smaller intervals, having uniform visual and/or motion characteristics (shots). For each shot, a few images (*key frames*) must be selected to characterize its visual content. These two operations greatly facilitate a documentalist's activity, by displaying a condensed view of a clip, and by automatically providing footage boundaries for textual descriptions. In addition to semantic, textual descriptions, it is possible to extract a representation of each key frame, describing its visual properties, such as composition, color, and texture. In this way, a database can be queried using example images. Another set of indexes can be automatically extracted from each shot, by analyzing its motion vectors. A classification can be made between stationary vs. moving shots. Shots characterized by an overall camera motion can also be detected and represented through pan/tilt/zoom parameters. These types of indexes are particularly important for video editing, in order to select a succession of segments that convey a coherent sense of self-motion throughout the whole clip. As a by-product of this motion vector analysis, a finer shot segmentation can be obtained, by detecting discontinuities in the camera motion properties.

Other author information: send correspondence to R.M., Computer Science Department,University of Geneva, 24 rue Général-Dufour, 1211 Geneva 4, Switzerland. Email: Ruggero.Milanese@cui.unige.ch. Supported by the European Union Project AC089.

Several algorithms have recently been described in order to extract the above information (cf. [5] for a review), including a few commercial systems and several publicly-available prototypes. A system by IBM [4] achieves shot segmentation by comparing consecutive frames using a combination of histograms, direct image differencing, and dominant motion estimation. Similar results are achieved by another commercial system by Virage Inc. [13]. A system called Jacobs enables the user to specify the dominant directions of motion in four quadrants of the image, which are matched to the optical flow vectors [8]. However, since these systems are based on uncompressed data, an MPEG decompression stage is necessary. Furthermore, operations such as optical flow estimation or frame/histogram differencing are very time consuming. In order to overcome these problems the use of motion vectors available from the compressed stream should be considered. A few systems that do not require decompression have been reported. WebClip [9] uses the ratio between forward and backward MPEG motion vectors at each frame to determine scene cuts. Camera motion is computed for P and B frames only, using an affine motion estimation method. Patel and Sethi [11] use a complementary approach, by computing histograms from decompressed I frames, thereby providing cut accuracy only at the *group of pictures* level. Finally, Zhang, Smoliar, Furht and colleagues [5] [14] use statistical moments of vector orientation histograms both for cut detection and camera motion indexing.

The methods proposed in the present paper are related to the latter class of approaches, dealing with compressed data. The main contributions consist of: (i) a model for computing an approximation of the optical flow from the available motion vectors; (ii) a rigorous approach to cut detection by adaptive learning of classification rules; (iii) a shot segmentation refinement method capable of detecting intervals with different motion characteristics; and (iv) a method for recovering camera motion parameters and their qualitative representation for content-based retrieval.

These aspects are described in detail in the following three sections. In section 5 results are reported for the experimental validation of the proposed methods.

2. VIDEO MODEL

An MPEG-1 stream consists of a sequence of groups of pictures (GOP), each of which is composed of three types of frames [6] [7]. Figure 1.a shows a typical sequence of such frames. The first type is called I-frame, and is compressed as a still picture (*intra*-coded), using the JPEG algorithm. A second type is denoted P-frame (from *predicted*), and is compressed through forward motion prediction using the corresponding macroblock from the previous I-frame. This yields a number of motion vectors (one for each macroblock) and a set residual macroblocks. A third type, called B-frame is obtained by bi-directional forward and backward motion prediction using the two closest previous and following non-B frames.

Both backward and forward motion vectors $\mathbf{v}^b_{ij}(t)$, $\mathbf{v}^f_{ij}(t)$ of an MPEG-1 stream are generally computed by block-matching over a search window around each macroblock (i,j). Therefore, they can be seen as an approximation of the optical flow $\mathbf{v}_{ij}(t)$. However, these vectors are not defined over a continuous time scale, since they correspond to displacements from the *closest* I/P frames, with bi-directional predictions and the possibility of skipping several frames. Furthermore, no motion vectors are provided for I frames. An approximation $\hat{\mathbf{v}}_{ij}(t)$ of step-by-step forward motion vectors must thus be computed using the available ones. Let us first consider the case of a frame of type P or B at time t preceded by either a frame I or P. In this case, the forward vectors are retained for all macroblocks, while the backward vectors are discarded, i.e. $\hat{\mathbf{v}}_{ij}(t) = \mathbf{v}^f_{ij}(t)$. For a frame of type I, no vectors are available, although the opposite of the backward vectors for the previous B frame can be considered, i.e. $\hat{\mathbf{v}}_{ij}(t) = -\mathbf{v}^b_{ij}(t-1)$. For the case of a P frame, forward vectors are available. However, they are based on a reference I frame that may have occurred much earlier in the stream. For this reason, the same strategy as above is employed. Finally, we consider a B frame at time t preceded by another B frame. In this case two options are possible: either we compute the difference between consecutive backward vectors, i.e. $\hat{\mathbf{v}}_{ij}(t) = \mathbf{v}^b_{ij}(t) - \mathbf{v}^b_{ij}(t-1)$, or consecutive forward vectors, in which case $\hat{\mathbf{v}}_{ij}(t) = \mathbf{v}^f_{ij}(t) - \mathbf{v}^f_{ij}(t-1)$. Figure 1.b illustrates the latter situation. A choice between these two options can be made for each macroblock independently, according to the one providing the smallest vector magnitude.

Clearly, the resulting motion vector field $\{\hat{\mathbf{v}}_{ij}(t)\}$ is only an approximation of the true one, in particular for frames of the latter case (which we call *rank 2* frames, as opposed to *rank-1* frames, obtained from MPEG vectors of only one frame). Indeed, the vectors from two consecutive frames (e.g. $\mathbf{v}^f_{ij}(t), \mathbf{v}^f_{ij}(t-1)$) do not necessarily describe a translation of the same entity. Nevertheless, the algorithms described below for shot segmentation and camera motion estimation appear to be robust to the error introduced by this type of approximation.

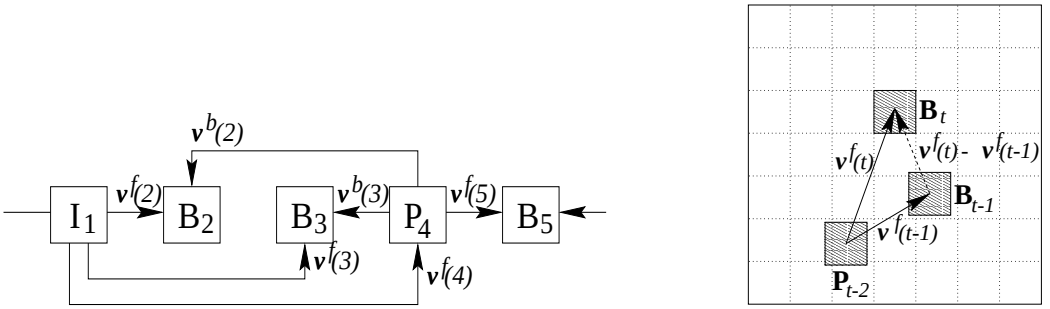


Figure 1. (a) Structure of a typical Group of Pictures in an MPEG stream. (b) Reconstruction of step-wise motion vectors.

3. SHOT SEGMENTATION

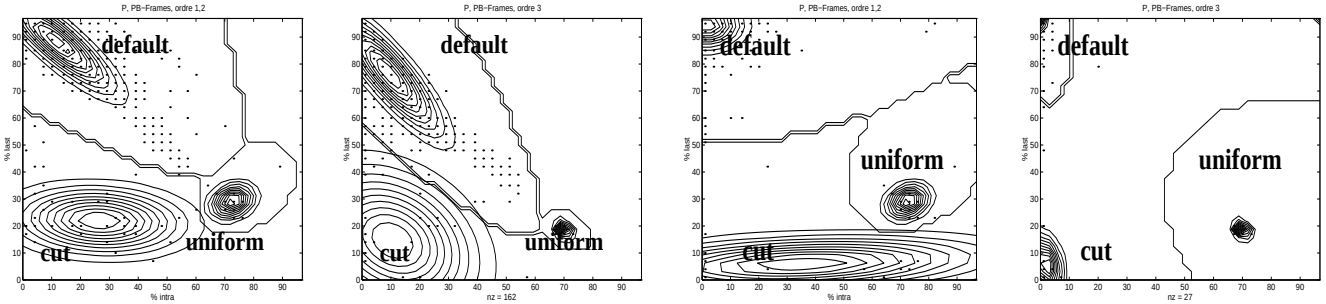
Several processes can originate discontinuities in the visual content of a clip, and should be detected in order to decompose it into smaller units. The first one is a scene cut. A cut generally separates shots of different visual content, since they have been recorded at different times. The second one is a discontinuity in the camera motion parameters, for instance a camera starts a smooth panning movement after having captured a stationary scene for some time. Finally, discontinuities may be introduced at editing time, by special effects such as fade, dissolve, etc. In this section we focus on the first case, i.e. scene cuts. An algorithm for further decomposing shots by motion analysis is described in section 4.

Scene cuts can in principle be detected by frame comparison. Comparison can be done either after decompression in the spatial domain, for instance by Euclidean distance, or directly in the transformed domain, using a selection of both frames' most meaningful DCT coefficients (if available). However, the computational cost of both approaches is quite high, and their results are very sensitive to camera motion. We propose an alternative approach, where global statistics are computed for each frame, using the available information from the compressed stream and from the reconstructed stepwise motion vectors. These statistics are used to characterize each frame with a feature vector $\mathbf{x}(t)$ suitable for classification.

Let us first consider the case of P and B frames. If one such frame is at the interior of a uniform shot, then its macroblocks should be predicted well from the previous and/or subsequent frames. Conversely, if such a frame is the beginning of a new shot, we can expect a bad prediction from previous macroblocks. In this case, the compression algorithm has likely decided to encode a large fraction of such macroblocks anew, using the DCT without prediction in order to achieve a higher compression ratio (*intra-coded* macroblocks). The number of intra-coded macroblocks $n_I(t)$ is therefore a good feature to characterize scene cuts. A similar reasoning can be made on the stepwise reconstructed motion vectors $\hat{\mathbf{v}}_{ij}(t)$ (cf. section 2). Although the number $n_R(t)$ of such vectors is clearly related to $n_I(t)$, they are not completely redundant, since the former expresses a continuous measure of uniformity with respect to the *previous* frame. For P and B frames the feature vector consists of $\mathbf{x}(t) = [n_I, n_R]$. For I frames, however, only n_R is available, and the feature vector becomes 1D.

Before using the above features for cut detection, some normalizations need to be performed. In particular, the number of intra-coded macroblocks must be divided by the number of available macroblocks $n_V = N - n_S(t)$, where N is the total number of macroblocks ($N = 396$ for SIF-PAL), and $n_S(t)$ denotes the number of skipped macroblocks. Similarly, the number of reconstructed stepwise vectors must be divided by the number of valid macroblocks, i.e. $n_V = N - n_S(t)$ or by $n_V = N - n_S(t-1)$, for the case of rank-1 frames, and by $n_V = N - \max(n_S(t), n_S(t-1))$ for the case of rank-2 frames.

Cut detection is formalized as the classification of a normalized feature vector $\mathbf{x}$ into a set of classes $\Omega = \{\text{cut}, \text{default}, \text{uniform}\}$. The third class is meant to represent non-cut frames that contain large uniform areas. Although functionally equivalent to the default class, these frames tend to have quite different statistics. Indeed,



(a) (b) (c) (d)

Figure 2. Classification masks for default, cut, and uniform frames (P and B frames only). The horizontal axes represent the value of the feature $n_I \times 100$, whereas the vertical ones indicate the value of the feature $n_{rec} \times 100$. Dots indicate the bins of the discretized feature space with the highest probability density. Curves within each class indicate 10 uniformly-spaced level curves for the estimated Gaussian distributions. (a) - (b) Masks learnt from a database of clips compressed using non-professional equipment, for rank-1 and rank-2 frames, respectively. (c) - (d) Same as above, for a set of clips compressed using professional equipment.

the compression scheme often finds it more convenient to intra-code the corresponding macroblocks, since the AC components are more compact to represent.

The cut detection task is thus formalized as a classification problem of a feature vector $\mathbf{x}$ into a class of Ω. To this end, a discriminant functions approach is employed, in the framework of Bayes decision theory [3]. For each class ω, a discriminant function $g_\omega(\mathbf{x})$ is constructed, equal to the a-posteri probability $p(\omega \mid \mathbf{x})$. The decision rule consists of assigning $\mathbf{x}$ to the class $\tilde{\omega} = argmax_\omega \; g_\omega(\mathbf{x})$. Given Bayes' theorem, this is equivalent to maximizing the discriminant function $g_\omega(\mathbf{x}) = p(\mathbf{x} \mid \omega) \cdot \mathrm{Prob}(\omega)$, where $p(\mathbf{x} \mid \omega)$ is the class-conditional probability density and $\mathrm{Prob}(\omega)$ is the prior probability of each class. The value of $\mathrm{Prob}(\omega)$ can simply be estimated through the frequency of each class over a training set. In order to estimate $p(\mathbf{x} \mid \omega)$ from a limited-size training set, a parametric model of the underlying density function is used. This model corresponds to a multivariate Gaussian, i.e.

$$p(\mathbf{x} \mid \omega) = \frac{\exp -\frac{1}{2}(\mathbf{x} - \mu_\omega)^t \Sigma_\omega^{-1} (\mathbf{x} - \mu_\omega)}{\sqrt{(2\pi)^p \mid \Sigma_\omega \mid}} \; ,$$

where $\mu_\omega = \mathrm{E}[\mathbf{x}]$ is the estimated mean of the distribution, and $\Sigma_\omega = \mathrm{E}[(\mathbf{x}-\mu)(\mathbf{x}-\mu)^t]$ is its estimated covariance matrix, both computed over vectors belonging to the class ω.

In order to speed up the classification, we discretize the feature space into a multidimensional matrix, and for each entry of the matrix we store the value of ω with the highest frequency, over the whole bin. In this way classification is done in a single step, by using such a classification mask as a look-up table, given a quantized input $\mathbf{x}$.

In consideration of the differences between rank-1 and rank-2 frames outlined in the previous section, it is important that independent classification masks be computed for each rank. Figure 2.a-b shows the decision boundaries of the resulting two masks. A total of 2,875 frames from various newsclips broadcasted by the Euronews TV channel were used for the learning stage, accounting for approximately 21% of our database. It can be noticed that just one feature would have not been sufficient to discriminate the classes. Furthermore, the class introduced for the uniform frames is indeed necessary to avoid confusion with the cut class, and to guarantee a good estimate of the default class given the unimodal parametric distributions $p(\mathbf{x} \mid \omega)$.

One advantage of the above procedure over algorithms that required fixed thresholds is its capability to adapt to the data. Indeed, we found that compression equipment of different quality may provide quite different results. Figure 2.c-d shows the results obtained on a different learning set of 2,353 frames, from clips that have been compressed with professional equipment. It can be noticed that in this case, much fewer intra-coded macroblocks can be found, and the only relevant feature left for classification becomes n_R.

To conclude, a method has been proposed for segmenting clips into shots at scene cuts. For each shot, it is then possible to extract a number of key frames, which can be used for displaying a condensed view of the clip (cf. Figure 6 left), as well as for determining the visual content of each shot, for content-based retrieval. A straightforward solution to selecting such key frames is to choose the first I frame following the start of each shot. This guarantees the highest quality for the display, as well as for the indexes that can be computed from it.

4. ANALYSIS OF CAMERA MOTION

Besides key frames, one further possibility to indexing a shot is through its motion properties. This creates a content-based retrieval capability that is particularly valuable to postproduction activities, such as video editing. For instance, an editor may want to select among a set of shots already retrieved after an example image, those that have been recorded with a steady camera, or those obtained with a panning. This access possibility facilitates the production of final clips with a nice visual impact, complying with a number of rules for the concatenation of shots (e.g. "never insert a zoom-in after a zoom-out").

The analysis of camera motion is performed in various steps. First, the reconstructed motion vectors are filtered in the spatial and temporal domain using median filters. The goal is to remove outlier vectors, to reduce the amount of *noise*, and to overcome some instabilities for B frames that are preceded by other B frames. Furthermore this also leads to a reduction of skipped macroblocks. These two filters have a window of size $3 \times 3 \times 1$ and $1 \times 1 \times 3$ for the (x, y, t) domains respectively.

In the next step, *stationary*, or motionless frames are detected by computing global statistics on the filtered, reconstructed vectors. The goal is to suppress from further motion analysis frames that have likely been recorded with a fixed camera over a static scene. Let $n_0(t)$ denote the number of reconstructed and median-filtered vectors at time t, whose magnitude is below a fixed threshold. Let $n_V(t)$ denote the number of valid vectors at time t. We use the ratio $\frac{n_0(t)}{n_V(t)}$ as the likelihood of a *stationary* frame: if this value is above a threshold, the vectors of the corresponding frame are not considered for further analysis. For the remaining frames, a further classification is performed, into *regular* and *irregular* motion. Regular motion is meant to capture the case of uniform camera motion over a largely static background, although it may be indistinguishable from the case of a static camera portraying a large moving object. Irregular motion occurs when a global camera motion model fails to account for the motion vectors, indicating that multiple objects in the scene have different types of independent motion. The following subsections describe the analysis for discriminating between these two cases, as well as the method for extracting camera motion indexes.

4.1. Extraction of camera motion parameters

The camera motion model used in the present simulations includes pan, tilt and zoom, which account for global translations and scaling. The suitability of such model to describe the distribution of the reconstructed motion vectors can be determined by minimizing a least squares criterion. This provides both a key to discriminating between regular and irregular motion, and also yields the optimal values of the camera parameters for the frames classified as regular motion.

Error-minimization approaches to analyzing the optical field with parametric models have been adopted by several researchers [10], and a few have been directly applied to compressed data [9]. However, the latter approaches do not take into account the unsuitability of raw MPEG-1 motion vectors to represent the optical flow. The step-wise vector reconstruction and the non-linear filtering described above are two ways to improve the density and reliability of such vectors in the spatial and temporal domain. Nevertheless, one needs to account for further problems related to these vectors, which are typically produced by block-matching algorithms. First, the density of vectors is quite limited due to the use of non-overlapping macroblocks. Second, vectors often present large deviations from their neighbors (cf. figure 3.a-b). Third, the use of fixed-size blocks tends to produce macroblocks with very small magnitudes wherever the image contains large uniform regions, such as the sky (cf. figure 3.c-d). Finally, articulated moving objects (e.g. persons) tend to produce very irregular and incoherent motion vectors.

In order to overcome these problems we use a bucketing approach on the reconstructed step-wise vectors. The vector field is split into $5 \times 4 = 20$ *super-blocks* B_{mn}, each referring to a rectangular portion of the image containing from 16 up to 25 vectors, according to its eccentricity from the image center. The goal of these super-blocks

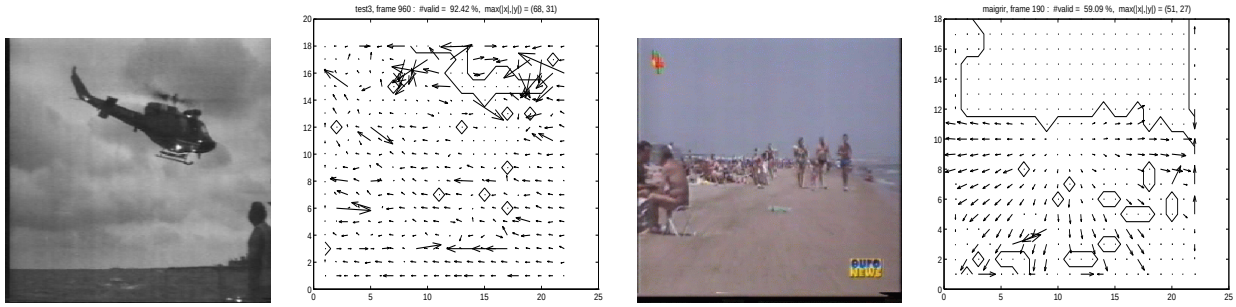


Figure 3. Examples of images associated to noisy motion vector fields. (a) - (b) Frame producing a large number of noisy, irregular vectors due to texture in the clouds. (c) - (d) Frame from a zoom-in shot, with large regions in the vector field containing low-magnitude or null vectors.

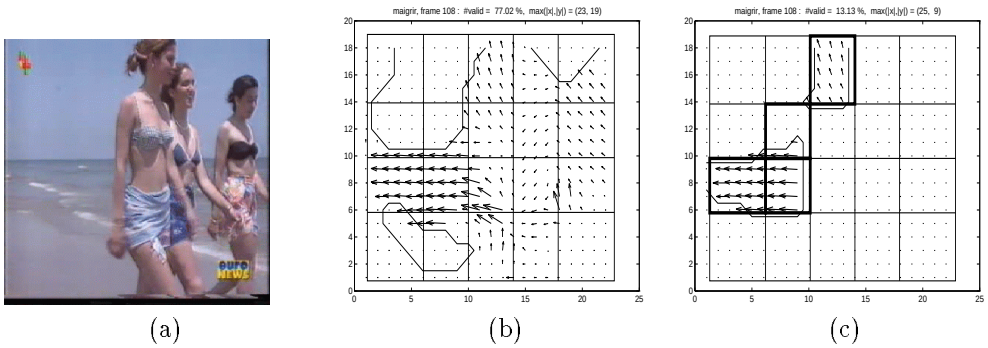


Figure 4. Suppression of noisy motion vectors by super-block analysis. (a) Source image, with a left pan to follow persons walking. (b) Corresponding filtered motion vectors field, and super-blocks. (c) Retained super-blocks, and retained vectors in each of them.

is to isolate parts of the image containing the major sources of noise, such as articulated moving objects and uniform areas. In order to detect such noisy super-blocks and to prevent them from corrupting the analysis of the remaining ones, a measure of uniformity in their motion vectors is computed. For each super-block at coordinates m, n, the x and y components of the mean vector $\mathrm{E}[\hat{\mathbf{v}}_{mn}] = \frac{1}{K}\sum_{ij \in B_{mn}} \hat{\mathbf{v}}_{ij}$ and the variance vector $\mathrm{E}[(\hat{\mathbf{v}}_{mn})^2] = \frac{1}{K-1}\sum_{ij \in B_{mn}} (\hat{\mathbf{v}}_{ij} - \mathrm{E}[\hat{\mathbf{v}}_{mn}])^2$ are estimated, where K is the number of valid vectors of this superblock. We then compute the norm of the variance vector, and compare it to a positive threshold. If its value is below the threshold, the super-block is deemed uniform, otherwise it is suppressed from further analysis, since it is subject to multiple types of motion. For each uniform super-block we then perform a similar type of variance analysis in the temporal domain, in order to validate its regularity through time. In this case however, only the direction of motion is considered, and no constraint is imposed on the vector magnitudes. Eventually, only super-blocks with a uniform spatio-temporal distribution will be conserved. Figure 4 shows the regular super-blocks that have been retained for a typical segment presenting both uniform regions and multiple moving objects.

The least squares motion model fitting procedure can now be applied to the valid and retained vectors belonging to the retained superblocks. The motion model is a 3-parameters one, consisting of horizontal translation Δx (pan), vertical translation Δy (tilt), and zooming α ($\alpha > 0$ indicates a zoom in, $\alpha < 0$ indicates a zoom out). The fitting procedure is the least squares, and uses the iterative Levenberg-Marquardt technique [12]. One immediate criterion to evaluate the goodness of fit is to check the χ^2 value returned by this method, after normalization by the sum of squared vector norms. If this exceeds a threshold, the whole frame is classified as *irregular*. Otherwise, we use

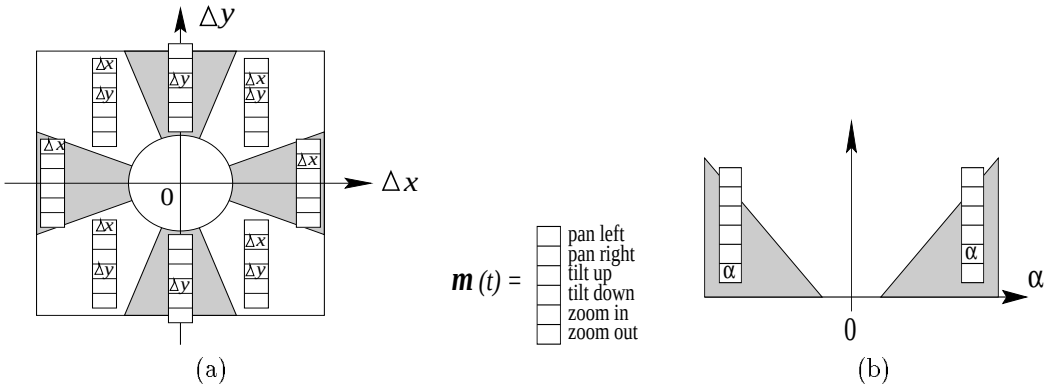


Figure 5. (a) Classification of the camera translation parameters $\Delta x, \Delta y$ into 4 components of the 6D motion vector $\mathbf{m}(t)$. (b) Classification of the camera zoom parameter α into the remaining two components of $\mathbf{m}(t)$.

the optimal motion parameters to determine the ideal distribution of motion vectors $\{\mathbf{w}_{ij}\}$. The relative distance $\frac{\|\hat{\mathbf{v}}_{ij}-\mathbf{w}_{ij}\|}{\|\hat{\mathbf{v}}_{ij}\|+\|\mathbf{w}_{ij}\|}$ between each valid vector and the corresponding ideal one is then computed. The percentage of vectors that can be appropriately described by the model can thus be computed. A low percentage indicates that the scene contains multiple motions that cannot be described by a unique camera motion, and again the frame is classified as irregular. Only frames presenting a significative fraction of well fitted vectors are eventually classified as regular.

4.2. Temporal filtering of motion classes

The classification of frames into stationary, regular, and irregular motion categories can be used to refine the shot segmentation method described in section 2, by simply detecting discontinuities in the temporally-ordered sequence of classes. This further segmentation can thus decompose a long sequence presenting a succession of camera motions (e.g. a stationary subsequence, followed by a zoom, over the same scene).

However, it should be recalled that the main goal of this process is to help post-production activities, such as video annotation and editing. Therefore, it is important to avoid oversegmentation, which would produce segments too short to deserve specific annotation, and would not be reusable into a new clip. To this end some motion class discontinuities need to be suppressed. This can be done by non-linear filtering the sequence of frame classes with a temporal window. The class of each frame is redefined according to a voting rule across the neighboring frames. After this filtering, a sequence of cuts is hypothesized at all frames exhibiting motion class discontinuities. In order to validate these cuts, the resulting sub-shots are sorted by increasing length. For the i-th sub-shot, if its length is below a minimum threshold equal to 12 frames the sub-shot is removed, and split into two halves, which are assigned the same class as their neighbors. The validation process stops as soon as the length of the current sub-shot exceeds the minimum-length threshold.

4.3. Temporal filtering of camera motion parameters

Once the shot segmentation has identified sub-shots of different motion classes, it is possible to isolate those of the *regular* class, and to fully characterize their camera motion. To this end, the values $\Delta x, \Delta y, \alpha$ provided by the least-squares fitting procedure described above, are analyzed. The analysis method and the type of indexes to be computed depend on the type of queries that should be envisaged for the database. Clearly, a user (most often a video editing professional) will generally not be able to fully specify the desired motion parameters numerically. Search criteria should rather be the presence or absence of a pan, tilt, zoom, as well as their direction. The three numerical values provided by the least-squares fitting should thus be mapped into a representation more suitable for "linguistic"-type queries.

The proposed method maps the parameters vector $[\Delta x(t), \Delta y(t), \alpha(t)]$ into a 6-parameters feature vector $\mathbf{m}(t)$, whose components indicate, respectively, a left pan, a right pan, an upward tilt, a downward tilt, a zoom in, and a zoom out. Querying can then be done by specifying one or more desired components of motion. The mapping into

these six components is performed by thresholding on the directions of the translation vector, and on the value of the zoom. Figure 5 indicates the strategy for assigning the vector $\mathbf{m}(t)$.

Given motion feature vectors $\{\mathbf{m}(t)\}$ describing all frames of a shot, it is now necessary to filter some components that may have occurred from camera vibrations or irregular motion. To this end, we use a sliding window of length W on the sequence of $\mathbf{m}(t)$ and count the number of frames contributing to each component. If this exceeds a threshold set as $\frac{2W}{3}$, the corresponding component of a binary vector $\mathbf{z}(t)$ is set to 1, otherwise it is set to 0. This binary vector indicates the most significant motion components throughout the window centered at each frame, and suppresses those introduced by short-lasting camera motion, such as those produced by hand-operated cameras.

Given the binary vectors $\mathbf{z}(t)$, discontinuities can now be determined more reliably between subintervals of different camera motions. Indeed, variations in the magnitude of motion (which may be due to a manually-operated camera) can be disregarded, while focusing on changes in the *components* of motion. Let us consider, for instance, the case of a left pan $\mathbf{z}(t) = [1, 0, ..., 0]$, which then changes direction within the same shot: $\mathbf{z}(t+1) = [0, 1, ..., 0]$. In this case, a cut should be introduced, in order to create two further units, and to not mix their parameters. However, a vector $\mathbf{z}(t)$ may present multiple non-zero components. For instance a shot may start with a concurrent left pan and zoom in, and then the pan may stop while the zoom continues for a few more frames. In this case, the whole shot should not be split, since there is continuity within a subset of its motion components. This result can be obtained by introducing a cut only if the following property is satisfied:

$$\mathbf{z}(t-1) \wedge \mathbf{z}(t) = \mathbf{0} \quad ,$$

where "$\wedge$" indicates the bit-wise logical *and* operation between the components of two binary vectors. After the above rule has been applied to all frames of the shot, a final sequence of sub-shots is obtained. For each sub-shot, a unique vector of camera motion parameters can then be computed, representing the whole sub-shot. This is done by computing the average value $\tilde{\mathbf{z}} = \mathrm{E}[\mathbf{z}(t)]$, representing the frequency of each type of motion throughout the sub-shot. Queries can therefore be made by setting all desired motion components into a binary query vector $\mathbf{q}$. The degree of match of a stored vector $\tilde{\mathbf{z}}$ can thus be computed through the normalized inner product:

$$\frac{\mathbf{q}^t \tilde{\mathbf{z}}}{||\mathbf{q}||_1} \quad ,$$

where $||\mathbf{x}||_1 = \frac{1}{N} \sum_{i=1}^{N} \mid x_i \mid$ is the L_1 norm of a N-D vector.

5. EXPERIMENTAL RESULTS

In this section we provide the experimental validation of the cut detection and camera motion characterization methods described above. The data set consists of 7 clips broadcasted by the Euronews and Megachannel TV channels, for a total of 13,389 frames. Approximately 36% of these clips were digitized and compressed using professional material, the remaining ones being compressed with home-PC equipment.

We first describe the results obtained for the shot segmentation task introduced in section 3. In this case, approximately 40% of the data set was used to train the Bayesian classifiers, while the remaining data was used to estimate the error rate. Training and testing was achieved by hand-labeling all available frames into $\Omega = \{\mathrm{cut}, \mathrm{default}, \mathrm{uniform}\}$. Table 1 shows how frames from each class were actually classified by the algorithm. Since the goal was cut detection, the overall percentage of false positives amounts to 0.3%, while the amount of false negatives is 2.2%, representing an improvement with respect to previous systems. One further advantage is the complete frame-accurateness of cuts, which can occur for all types of frames.

Next we describe the results of the camera motion characterization algorithm introduced in section 4. This task is achieved in several steps. The first step refines scene-cut-based shot segmentation by motion class analysis. All frames of the "default" class are thus mapped into the set $\Lambda = \{\mathrm{stationary}, \mathrm{regular}, \mathrm{irregular}\}$, and segmented into uniform sub-intervals. In order to evaluate the performance of this sub-shot segmentation, all frames have been hand-labeled over the set Λ. Table 2 shows how the classification provided by the algorithm given the ground-truth labels provided by a human. In this case no learning rule was necessary and the whole dataset of 13,389 frames was

Table 1. Results of the scene-cut-based shot segmentation method on the "validation" subset of the database (8,033 frames). The left column indicates the ground-truth labels introduced by the supervisor. Numbers in brackets indicate the frequency of each class in the database.

Ground truth	Default	Uniform	Cut
Default (99.0%)	99.7%	0%	0.3%
Uniform (0.3%)	0%	100%	0%
Cut (0.7%)	1.1%	1.1%	97.8%

Table 2. Results of shot segmentation refinement, based on classification by motion type, on the full dataset of 13,389 frames. The left column indicates the ground-truth labels introduced by the supervisor.

Ground truth	Stationary	Regular	Irregular
Stationary (66.2%)	91.97 %	0.64 %	7.39 %
Regular (18.7%)	4.15 %	81.31 %	14.53 %
Irregular (15.1%)	3.00 %	15.67 %	81.33 %

used. It should be noticed that the largest error occurs for the non-stationary motion types. Given their relative scarcity in the dataset, the overall successful classification rate (sum of weighted diagonal entries) attains 88.3%.

Next, we provide the results of the camera motion labelling procedure, on all shots s that have been classified as "regular" (denoted by the set S). In this case, each frame t within such shots was hand-labeled with a binary vector $\mathbf{z}_s(t)$ representing the ground truth. For all frames in the recovered "regular" class, an estimated vector $\hat{\mathbf{z}}(t)$ was computed. The actual and estimated vectors were then compared and the number of mismatching 1s at each frame was determined. A false positive was counted whenever a 1 at coordinate i was matched by a 0, and a false negative in the opposite case. For each frame, the number of mismatches was then normalized by the number of "ideal" 1s. Finally, the mismatch measure was averaged over the full length of a shot and over all shots:

$$\begin{aligned} \text{False positives} &= \frac{1}{\sum_{s\in S} l(s)} \sum_{s\in S} \sum_t \frac{\|\neg \mathbf{z}_s(t) \wedge \hat{\mathbf{z}}_s(t)\|_1}{\|\hat{\mathbf{z}}_s(t)\|_1} \times 100 = \quad 5.74\% \\ \text{False negatives} &= \frac{1}{\sum_{s\in S} l(s)} \sum_{s\in S} \sum_t \frac{\|\mathbf{z}_s(t) \wedge \neg \hat{\mathbf{z}}_s(t)\|_1}{\|\mathbf{z}_s(t)\|_1} \times 100 = \quad 4.89\% \quad , \end{aligned} \tag{1}$$

where the operator "$\wedge$" is the bit-wise logical *and* between two vectors, "$\neg$" denotes the bit-wise complement of a vector, and $l(s)$ is the length of a "regular" motion shot. These values represent the performance over all components of motion. In table 3, a finer distinction is made for each component independently.

Table 3. Results of camera motion characterization for all correctly-classified frames of the "regular" class (2,049 frames). Numbers in brackets indicate the frequency of each motion type in such a dataset. Note that their sum exceeds 100, since approximately 12% of the frames contained a combination of different motion types.

	Pan left (15.05%)	Pan right (52.77%)	Tilt up (17.13%)	Tilt down (17.65%)	Zoom in (9.17%)	Zoom out (0.00%)
False positives	8.89 %	4.81 %	7.48 %	0.00 %	0.00 %	0.00 %
False negatives	5.75 %	2.62 %	0.00 %	0.00 %	18.87 %	-

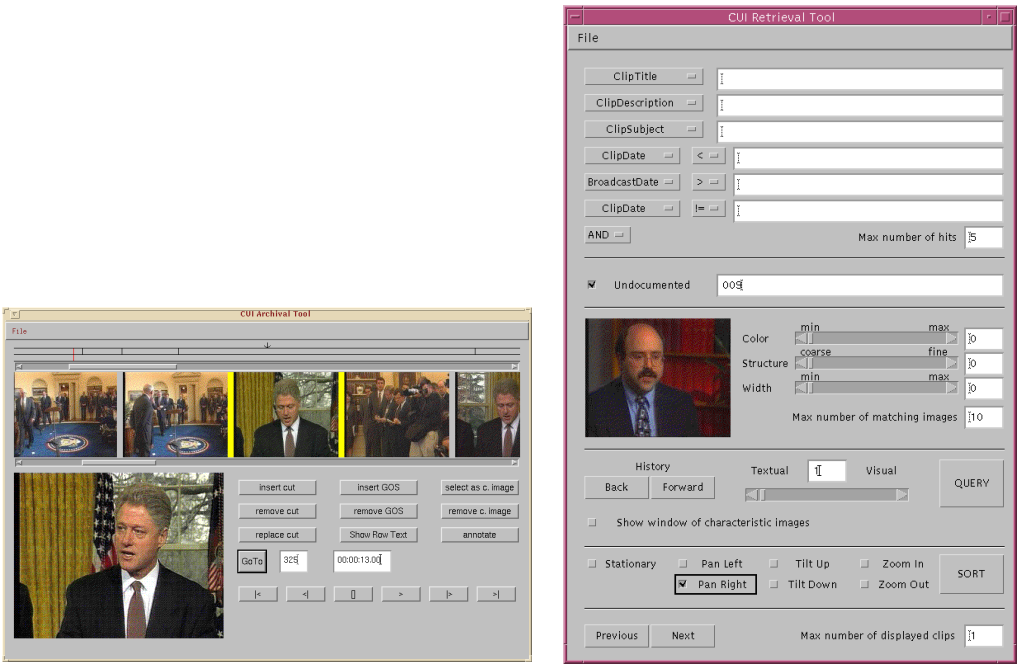


Figure 6. Graphical user interfaces for display and access to the results of shot segmentation and camera motion characterization algorithms. (Left) Archival tool; (right) Retrieval tool.

Finally, a few remarks about the implementation. All software is written in C, including a public-domain MPEG-1 parser [2]. Simulations were run on a standard Sun Sparc Ultra-2 Workstation. The processing speed achieved without any optimization is 59 frames/s, including video parsing and all processing steps described above.

6. CONCLUSIONS

In this paper we described two major algorithms for analyzing a video clip and extracting useful indexes for content-based retrieval. We first proposed a clip segmentation method based on the analysis of reconstructed motion vectors, capable of detecting scene cuts using a supervised Bayesian classification approach. We then introduced the problem of indexing shots by a representation of their camera motion. We showed that an analysis of the motion vectors can identify subintervals of different motion types, whose boundaries can be used to refine the previous segmentation process. Finally, we showed how qualitative descriptions of the camera motion can be computed by least-square fitting a simple motion model, and by binarizing the recovered parameters.

All the above algorithms are designed for processing MPEG-1 streams, without decompression. The total processing rate achieves 59 frames/s on a standard workstation. Experimental results have been reported from a database of 7 news clips, for a total of 13,389 frames. The error rates are comparable with state-of-the-art approaches requiring video decompression.

These algorithms are currently being used in the context of a content-based video archival and retrieval system designed for TV studio post-production within the European Union ACTS project "Distributed Video Production". Graphical user interfaces have been developed for two tools (cf. figure 6). One is the archival tool, which enables a documentalist to display the pre-segmented structure of a clip, together with its automatically-extracted key frames. The other one is the retrieval tool, which provides a user (journalist, video editor, etc.) access to the database using a combination of indexes provided by textual queries, example images, and camera motion descriptions.

REFERENCES

1. E. Ardizzone and M. La Cascia, Video Indexing Using Optical Flow Field. Proc. IEEE-ICIP'96, Sept. 16-19, 1996, Lausanne, Switzerland.
2. Berkeley Plateau Multimedia Research Group on MPEG-1: MPEG Stat, MPEG Play. Web Home Page: http://bmrc.berkeley.edu/projects/mpeg/.
3. R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis. John Wiley and Sons, 1973.
4. M. Flickner, et al., Query by image and video content: the QBIC system. IEEE Computer, Sept. 1995, 23-32.
5. B. Furht, S.W. Smoliar, H. Zhang, Video and Image Processing in Multimedia Systems. Kluwer Academic Publishers, 1995.
6. ISO/IEC/JTC1/SC29/WG11, MPEG Document AVC-400, Test Model 3.
7. ISO/IEC 13818 - 2 Committee Draft (MPEG-2).
8. JACOBS Web Home Page, http://wwwcsai.diepa.unipa.it/research/projects/jacob/jacob_demos.html
9. H.J. Meng and S.-F. Chang, CVEPS: A Compressed Video Parsing and Editing Systems. Proc. ACM Multimedia 96, Boston, Nov. 1996.
10. J.-I. Park, and C.W. Lee, Robust estimation of camera parameters from image sequence for video composition. Signal Processing: Image Communication, **9** (1996):43-53.
11. N.V. Patel and I.K. Sethi, Video shot detection and characterization for video databases. Pattern Recognition, April 1997, **30**(4):583-592.
12. W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, Numerical Recipes in C, 2nd edition, Cambridge University Press, 1995.
13. VIRAGE Web Home Page, http://www.virage.com.
14. H.J. Zhang, J. Wu, D. Zhong, and S.W. Smoliar, An integrated system for content-based video retrieval and browsing. Pattern Recognition, April 1997, **30**(4):643-658.