Hindawi Publishing Corporation
International Journal of Digital Multimedia Broadcasting
Volume 2010, Article ID 864123, 9 pages
doi:10.1155/2010/864123

Research Article

An Optimized Dynamic Scene Change Detection Algorithm for H.264/AVC Encoded Video Sequences

Giorgio Rascioni, Susanna Spinsante, and Ennio Gambi

Università Politecnica delle Marche, Italy

Correspondence should be addressed to Susanna Spinsante, s.spinsante@univpm.it

Received 1 September 2009; Accepted 28 December 2009

Academic Editor: Ling Shao

Scene change detection plays an important role in a number of video applications, including video indexing, semantic features extraction, and, in general, pre- and post-processing operations. This paper deals with the design and performance evaluation of a dynamic scene change detector optimized for H.264/AVC encoded video sequences. The detector is based on a dynamic threshold that adaptively tracks different features of the video sequence, to increase the whole scheme accuracy in correctly locating true scene changes. The solution has been tested on suitable video sequences resembling real-world videos thanks to a number of different motion features, and has provided good performance without requiring an increase in decoder complexity. This is a valuable issue, considering the possible application of the proposed algorithm in post-processing operations, such as error concealment for video decoding in typical error prone video transmission environments, such as wireless networks.

1. Introduction

Scene change detection is an issue easy to solve for humans, but it becomes really complicated when it has to be performed automatically by a device, which usually requires complex algorithms and computations, involving a huge amount of operations. The process of scene change detection becomes more and more complex when other constraints and specific limitations, due to the peculiar environment of application, may be present. A scene in a movie, and, in general, in a video sequence, can be defined as a succession of individual shots semantically related, where a shot is intended as an uninterrupted segment of the video sequence, with static frames or continuous camera motion.

In the field of video processing, scene change detection can be applied either in preprocessing and postprocessing operations, according to the purposes that the detection phase has to achieve, and with different features and performance. As an example, in H.264/AVC video coding applications, scene change detection can be used in preprocessing as a decisional algorithm, in order to force Intraframe encoding (I) instead of temporal prediction (P), when a scene change occurs, and to confirm predicted or bi-predicted (B) coding for the remaining frames. As discussed in [1], a dynamic threshold model for real time scene change detection among consecutive frames may serve as a criterion for the selection of the compression method, as well as for the temporal prediction; it may also help to optimize rate control mechanisms at the encoder.

In lossy video transmission environments, the effects of the errors on the video presentation quality depend on the coding scheme and the possible error resilience strategy adopted at the encoder, on the network congestion status, and on the error concealment scheme eventually present at the decoder. In order to improve error resilience of the transmitted video signal, and stop error propagation during the decoding phase, Intra-picture refresh is usually adopted at the encoder, even if it is an expensive process, in terms of bit rate, as the temporal correlation among frames may not be exploited. In such conditions, a predictive picture refresh based on scene context reference picture, selected through a scene change detector, may ensure bit rate savings, while optimizing the choice of the refresh frames [2]. Scene cut detection may be also exploited to improve video coding,

attention based adaptive bit allocation, as presented in [3]. Scene cut detection is applied to extract frames in the vicinities of abrupt scene changes; visual saliency analysis on those frames and a visual attention-based adaptive bit allocation scheme are used to assign more bits to visually salient blocks, and fewer bits to visually less important blocks, thus improving the efficiency of the encoding process and the final quality of the compressed video sequence.

As previously introduced, besides being adopted in preprocessing operations, scene change detection may be usefully exploited in video postprocessing algorithms, such as in the context of error concealment of decoded video sequences affected by errors and losses. It is reasonable to expect that scene change detection at the decoder will have to face different conditions, with respect to scene change detection applied at the encoder. As an example, the input video sequence for a decoder could be the result of a video editing process, originating an encoded video stream with a lot of independent scene changes, as frequently happens in advertising videos. Moreover, the detector has to perform decisions and computations based on the available data, that may be missing or erroneous. H.264/AVC compressed video information is very sensitive to channel errors appearing during transmission. The adoption of Variable Length Coding (VLC) at the encoder side, together with more complex techniques like Motion Compensation, can lead to dramatic error propagation effects during decoding. Additionally, lost or damaged data cannot be retransmitted in real-time applications. As already discussed, error resilience techniques for compression, enhancing the robustness of the bitstream at the source coder, can be employed. They basically rely on adding extra parameters or more synchronization points at the encoder; however, this solution requires to change the encoding scheme and in some situations this is not possible, or not compatible with existing systems. Moreover, even if the bitstream is resilient to errors, errors may still occur. Hence, error concealment solutions at the decoder are usually preferred in most practical applications. When residual bit errors remain, error concealment approaches can conceal the error blocks by exploiting spatial and/or temporal correlation [4] of the correctly received data. Scene change detection algorithms may improve the performance of error concealment solutions, by allowing the selection of the proper spatial or temporal strategy. At the same time, the integration of a scene change detector in a real time concealment solution at the decoder poses strict constraints on complexity and computational time requirements. In a real world video, it is reasonable to expect that errors occurring at scene changes are less frequent than errors occurring in other pictures of the video sequence, basically because the number of scene change events is necessarily smaller than the total amount of frames in the sequence. However, besides being catastrophic for the decoding mechanism, errors at scene cuts can be really annoying to the viewer: the temporal correlation among two frames in a scene change is so insignificant that Intererror concealment also generates very poor results, and macroblocks that do not fit with the content of a frame can appear on the scene, disturbing the viewer's experience. As a consequence, errors at scene cuts should be avoided or compensated somehow, but conventional temporal error concealment strategies are inadequate for this case. Therefore, a suited scene change detector designed for real-time decoding of video signals can contribute to mitigate the effects of data losses at scene cuts, and to improve the final quality of the video sequence.

Several solutions may be implemented to provide scene change detection, differentiated on the basis of the target application, and the corresponding computational requirements. In the context of video storage and retrieval, it is reasonable to assume the possibility of performing an offline processing of the video sequence, that may allow for increased complexity and accuracy; in real time environments, strict requirements on available time and computing resources must be satisfied, thus determining the need for low-complexity solutions, anyway able to provide acceptable performance.

The remainder of this paper is structured as follows: a review of some previous works on real time scene change detection algorithms is provided in Section 2. The proposed detector is presented in Section 3, and its performance is discussed, by means of experimental evaluations, in Section 4. Finally, conclusions are given in Section 5.

2. Previous Work

Several solutions for scene change detection have been proposed in the literature, to be applied either at the video encoder or at the decoder.

Sastre et al. presented a low-complexity shot detection method for real time and low-bit rate video coding, in [5]. As clearly stated by the authors, the method is basically aimed at compression efficiency more than frame indexing or other purposes. The algorithm is based on Intra/Inter decision for each macroblock, during the encoding process, and on the use of two thresholds, a fixed one and an adaptive one. If the algorithm detects the first frame of a scene change, based on the fact that the number of Intra macroblocks used when encoding the frame as a P-frame overcomes the thresholds, the algorithm stops the encoding as a P-frame, and forces the Intraframe encoding. Shot changes represent the best choice to insert key frames in the video sequence: the next frames of the new shot may be then encoded via motion compensation and prediction, based on the first I-frame. Inserting the key frames in suited positions of the bitstream allows to obtain the best quality in the decoded stream, and to optimize the output bit rate.

The proposed algorithm relies on two basic thresholds expressed as a percentage of the total number of macroblocks in a coded picture. The fixed threshold is set to a high value, to ensure that any frame in which the number of Intra macroblocks (I-MB) exceeds the threshold is coded as an I-frame, independently of the rest of the algorithm's conditions. The second, adaptive threshold depends on the average number of I-MBs of all the pictures encoded since the last I-frame, and forces a frame to be coded as an I-frame if the number of I-MBs in it exceeds the average number of intra MBs of the previous frames, in a given

quantity. Several ideas implemented within this algorithm have been exploited also by the one proposed in this paper, that is described in the next Section. First of all, we use two different thresholds, a fixed one, expressed in terms of absolute number of macroblocks, and an adaptive one, used to sharpen the shot detection. A limit is also placed on the adaptive threshold, below the fixed one, to prevent a frame from being encoded as a P-frame when almost all of its macroblocks are Intracoded. A smoothing algorithm with memory is used to determine the average of Intra macroblocks of the previous frames within the shot, in order to avoid tracking the number of Intra macroblocks too fast, and to provide a stable value for the desired average. Finally, a span parameter is used to avoid shot detections too close in time: the span establishes a period of time after a shot detection, during which only the fixed threshold is active, and the adaptive threshold cannot cause the insertion of a key frame in the bitstream.

In [6], a pixel based-algorithm for abrupt scene change detection is presented. The algorithm requires a two-stage processing of the frames, before passing them to the H.264/AVC encoder. In the first stage, subsequent frames are tested against a dissimilarity metric, the Mean Absolute Frame Difference (MAFD):

$$\mathrm{MAFD}_n = \frac{1}{MN} \sum_{i=0}^{M-1N-1} \sum_{j=0} \left| f_n(i,j) - f_{n-1}(i,j) \right|, \quad (1)$$

which measures the degree of dissimilarity at every frame transition, with M and N being the width and height of the frames, $f_n(i,j)$ the pixel intensity at position (i,j) of the nth frame, and $f_{n-1}(i,j)$ the pixel intensity at the same position of frame $n-1$. Considering that most of the frames in a video sequence do not belong to scene changes, a quick frame skimming can be performed by such a metric. As a matter of fact, abrupt scene transitions produce a peak value in MAFD within a period of time, in contrast with normal motion of objects and camera in the scene, that usually causes a large MAFD signal over a period of time. In the second stage, the set of frames not previously discarded are normalized via a histogram equalization process, through a progressive refinement based on MAFD and other three metrics, applied on the normalized pixel values. The algorithm does not perform motion estimation but it only works on frame pixel values, thus avoiding high-computational costs. For this reason, it may be suitable for real-time video segmentation applications, and rate control. Experimental tests discussed by the authors show that the algorithm is efficient and robust in presence of abrupt scene changes, whereas it shows some limitations when gradual changes (such as dissolve and fade) or luminance variations (flickers) affect the video sequence. A combination of different metrics should be applied in those cases, in order to improve and refine the algorithm's detection capabilities.

A prominent reference for the scene change detector proposed in this paper is the scheme presented in [1], by Dimou et al.. The fundamental result is the definition of a Dynamic Threshold Model (DTM) that can efficiently trace scene changes, based on the use of an adaptive and dynamic threshold which reacts to the sequence features, and does not need to be calculated before the detection, and after the whole sequence is obtained. The method is based on the extraction of the Sum of Absolute Differences (SAD) between consecutive frames from the H.264 codec, that is then used to select the compression method and the temporal prediction to apply. The SAD defines a random variable, whose local statistical properties, such as mean value and standard deviation, are used to define a continuously updating automated threshold. Statistical properties are extracted over a sliding window, whose size is defined in terms of the number of frames over which the random variable is observed. The algorithm also applies a function-based lowering of the detection threshold, in order to avoid false detections immediately after a scene change. As a matter of fact, each time a scene change is detected, the SAD value of this frame is assigned to the threshold; for the following K frames, the threshold value is set according to an exponentially decaying law, with a suitably chosen parameter to control the speed of decaying.

Scene changes generate high SAD values that make them detectable. Given the classical SAD definition for the n-th frame,

$$\mathrm{SAD}_n = \sum_{i=0}^{M-1N-1} \sum_{j=0} \left| f_n(i,j) - f_{n-1}(i,j) \right|, \quad (2)$$

a random variable X_i is defined, which models the SAD value in frame i. A sliding window of length K, with respect to the nth frame, is defined as the subset of frames whose index lies in $[n-(K+1), n-1]$. Over the sliding window, the empirical mean value m_n and the standard deviation σ_n of X_i are computed as follows:

$$m_n = \frac{1}{K} \cdot \sum_{i=n-K-1}^{n-1} X_i, \quad (3)$$

$$\sigma_n = \sqrt{\frac{1}{K-1} \cdot \sum_{i=n-K-1}^{n-1} (X_i - m_n)^2}. \quad (4)$$

Both (3) and (4), together with X_{n-1}, are used to define threshold $T(n)$ as follows:

$$T(n) = a \cdot X_{n-1} + b \cdot m_n + c \cdot \sigma_n, \quad (5)$$

where n denotes the current frame, and a, b,,and c are constant coefficients. The algorithm's performance is strongly related to the proper selection of the values assigned to constants a, b, and c: not only may they determine better or worse detection rates, but they must also be tailored to the application context, which means they will have different values if used at the encoding or decoding stage. Constant a rules the way threshold $T(n)$ follows the evolution of the random variable X_i: it is suggested to keep the value of a small, as many factors different from true scene changes can cause the rapid variation of X_i, and could consequently affect the correct detection. Constant b, on its turn, gives different weight to the average SAD computed over the

sliding window: if b takes high values, the threshold becomes more rigid and does not approach the X_i sequence. This avoids wrong change detection in presence of intense motion scenes, but, on the other hand, can also cause some missed detections, in presence of difficult scene changes featuring low SAD values. As σ_n is the standard deviation of variable X_i, high values of constant c prevent detecting intense motion events as scene changes. From this brief discussion, it is evident that the selection of a, b, and c is a hot point, and only a good tradeoff according to the target application can ensure proper functioning of the whole algorithm. Once a scene change has been detected in the pth frame, threshold $T(n)$ assumes the value of the SAD computed over the last frame. In order to avoid false detections immediately after a scene change, the threshold to use for the successive frames is forced to decay exponentially, according to the following law:

$$T_e(n) = X_{n-1} \cdot \exp^{-s(n-p)}, \tag{6}$$

where parameter s controls the speed of decaying. In experimental tests reported by the authors, constants a, b, and c were empirically chosen; the sliding window size was set to 20 frames, and the decaying parameter equal to 0.02. Remarkable improvements can be obtained by the algorithm, when compared to a scene change detector based on an optimal fixed threshold, chosen after having computed the SAD over all the frames, and *manually* identified the true scene changes.

3. The Proposed Scene Change Detection Algorithm

The object of this paper is to present a robust scene change detector, based on an improved version of the DTM discussed in the previous Section, but aimed at being applied in the different context of postprocessing applications, as in the case of an error concealment framework for H.264/AVC decoders. As a consequence, besides the strict requirements on low-complexity and real-time capability, the algorithm should be able to detect incorrelation between consecutive frames, that is, scene changes, even when applied to a corrupted bitstream, where the information needed to reveal a scene cut may be missing or not complete, due to errors and losses happened during video transmission. Besides that, the algorithm cannot rely on information about future frames to locate scene change events (as, on the contrary, it may happen in applications addressing the encoder side), and, considering the target application context of concealment at the decoder, it is important to design a detector able to locate changes affecting parts of the frame content, and not only the whole scene.

In encoded video streams of the YUV color space, the SAD computation may be performed on each single color component. However, considering a YUV 4:2:0 stream, it is obvious that the luminance component Y carries the greatest amount of information, so that SAD computation can be executed on the Y component only. Besides that, the luminace component is the one the human visual system is most sensitive to. In the proposed detector, a random variable X_i is defined as the average number of pixels per MB for which the SAD value is greater than 30. It is important to note that, being the random variable defined over frames affected by errors and losses, the average number of pixels is computed with respect to all the correctly received MBs shared (i.e. co-located) between consecutive frames. The threshold value of 30 has been set empirically, by observing that in case of a scene change, it is highly probable that colocated pixels have an absolute difference value greater than the threshold chosen. The dissimilarity measure provided by X_i seems more reasonable than a *pure* SAD metric in a context of possibly missing information; however, misbehaviours may still be present and are to be faced by proper adjustments.

As a first condition to consider, given the fact that the dissimilarity metric adopted is defined on the basis of the Y component only, it is clear that it will show a marked sensitivity to rapid variations in the luminance content of the scene, even if not due to a real scene cut. Looking at Figure 1, flashing lights produce a rapid increase in the luminance level of consecutive frames, even if no scene change has happened at all. In these situations, the metric previously defined could reveal a false scene cut, so that a proper correcting action is to be applied.

In order to avoid false scene cut detection, during the decoding phase, and before computing the dissimilarity metric value, a second parameter is computed, named ΔY, defined as the difference between the average value of the MB luminance of two consecutive frames. The MBs included in the computation may be not the co-located ones in the two frames, given the possible losses during video signal transmission. The positive or negative ΔY value is subtracted to the dissimilarity value obtained by the SAD-based computation, to get the final metric. The curves reported in Figure 2 show how this simple modification may improve the reliability of the dissimilarity metric: peaks in the average Y value per MB curve ($\langle Y \rangle$) correspond to flashing light events in the video sequence and are obviously revealed by associated couples of peaks in the value assumed by X_i (there are 2 peaks in X_i for each peak in $\langle Y \rangle$, as a flashing light event affects two consecutive frames). The modified metric curve maintains the correct location of peak couples, but avoids a false scene cut detection, by properly lowering the resulting dissimilarity measure, with respect to the unmodified metric curve.

A second modification to the original scene cut detector inspiring this work is motivated by the target application context of error concealment solutions at the decoder. In view of concealment operations possibly performed on the same frames analyzed by the scene cut detector, it may be useful to collect, through the application of the detector, information related not only to global changes affecting the whole frames, but also referred to parts of the frame, on a local scale. This *granular* information may be possibly exploited to identify parts of the frame where Intraconcealment could be more suitable than Inter, because of local changes, even if the frame under processing is temporally related to the previous one, and so could claim for a global Inter concealment. An example of this possible situation is

Frame ($n-1$) (a)

Frame (n) (b)

Figure 1: Luminance variation between consecutive frames due to flashing lights, with no scene change event.

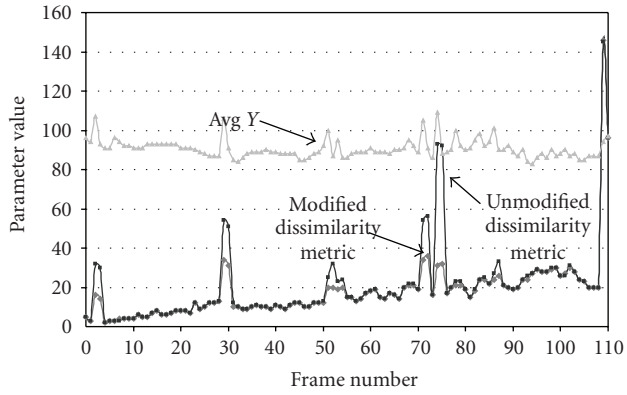


Figure 2: Variations of the average Y per MB, unmodified dissimilarity metric, and modified dissimilarity metric by ΔY parameter, for a test video sequence, due to flashing light events.

shown in Figure 3: comparing the two consecutive frames, it is clear that a scene cut does not take place, however, part of the background changes substantially. If several MBs get lost in the background area evidenced on the figure, an Inter concealment algorithm based on temporal correlation would fail in properly restoring the scene, whereas an Intra, spatial-based concealment, could be effective. Availability of such an information about local changes in the frame could enable an adaptive concealment strategy, based on differentiating the recovery technique on a group-of-MBs level.

In order to collect local scale information about the frame content, each frame has been virtually divided into macro areas, the number of which depends on the frame format; for CIF frames, 20 areas are located. By this way, each area includes 4×4 MBs, with the exception of the edge areas where the number of MBs may be 4×5 or 5×5, as shown in Figure 4. By such a virtual chessboard pattern, it is possible to track useful local information about the frame, even if not on a pixel basis, which would require unacceptable storage resources. At the decoder, a memory buffer is defined, whose elements are indexed according with the label associated to each macro area; each buffer element, in its turn, stores the average dissimilarity value evaluated over the specific macro area identified by the element index.

Besides being useful in the case of subsequent concealment operations, the virtual frame partition may help in correctly revealing a true scene cut, with respect to variations in the content which could affect most of the frame, without anyway representing a true scene change. As a matter of fact, if a true scene cut takes place, evident variations in the dissimilarity value will affect all the macro areas, and not only a limited subset of them. According to such a reasoning, a further decision step is included in the detector: once having computed the average and median dissimilarity values over the virtual partition buffer, if they both result in greater than an empirically set value of 100, the dissimilarity measure is increased by 20%; otherwise, if both the values are lower than 80, the dissimilarity metric X_i is reduced by the same percent value. Figure 5 highlights the effects of such a modification on the behaviour of the dissimilarity metric X_i: possible true scene cuts are emphasized by the modified metric, thus permitting their correct detection, whereas possible false cuts are minimized, to reduce the probability of an erroneous detection.

Information collected by the virtual frame partition process, as said before, may be exploited to analyze the local dynamic evolution of a frame. As shown in Figure 6(b), the average dissimilarity values for each macro area denote a change in the central part of the frame, which, however, is not due to a true scene cut, as many of the edge areas show a zero value. In the specific case reported, the algorithm provides an average value of 30.2, and a median value of 17: consequently, by lowering the dissimilarity value by a 20% amount, the risk of false cut detection is avoided.

The last modification added to the dynamic detector is conceived to face the case of high-motion scenes, as, for example, in the case of panning effects of the video camera. These situations show a typical effect over the set of frames included in the observation window (i.e., the sliding window cited in previous section), which spreads over 5 frames in the proposed scheme: given the threshold definition in (5),

Frame $(n-1)$ (a) Frame (n) (b)

Figure 3: Two sample consecutive frames with local changes in the background, but no scene change.

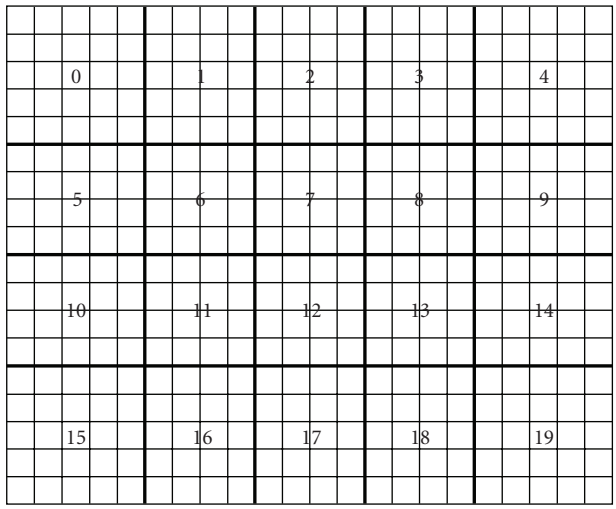


Figure 4: Virtual frame partition.

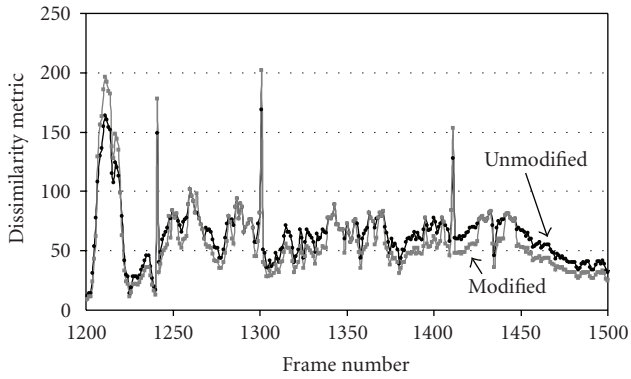


Figure 5: Performance of unmodified and modified dissimilarity metric exploiting local scale information provided by virtual frame partition.

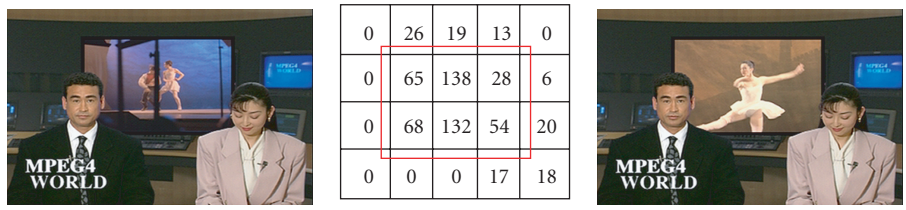
high values of the parameter m_n and low values of σ_n are jointly observed. In such situations, the probability of a false scene cut detection may be very high; consequently, the dissimilarity value is forced to decrease by a 20% amount, when $m_n > 80$ and $\sigma_n < 10$. These specific thresholds have been derived by extensive empirical tests over different video sequences. Figure 7 shows the behaviour of the modified dissimilarity metric in presence of a high-motion video sequence. It is important to notice that the motion degree of a video sequence, besides being an intrinsic property of the sequence itself, is also influenced by the frame rate set at the encoder. If a YUV sequence encoded at 25 ÷ 30 fps is decimated by a coefficient of 2 or 3, the final effect is to increase the motion degree of the decimated video sequence; this is an issue to take into account, as frame decimation is a typical operation performed on video sequences in order to reduce their bit rate and allow transmissions over limited bandwidth channels (i.e., wireless systems).

Table 1: Dynamic variation conditions for the threshold coefficients.

Condition	Value
Default	$a = -0.4$
	$b = 1.7$
	$c = 2$
$0 < \sigma_n \leq 5$	$c = 10$
$5 < \sigma_n \leq 10$	$c = 5$
$10 < \sigma_n < 30$	$c = 3$
$T(n) < 70$	$T(n) = 70$

Further tuning operations in the detection algorithm involve the a, b, and c coefficients defining the detection threshold (5). Imposing an adaptive and dynamic variation of these coefficients adds flexibility to the detection threshold, thus maintaining its effectiveness for a correct scene cut detection. Variations applied on coefficients a, b, and c, and extracted by empirical observations over many different video sequences are summarized in Table 1. Besides that, in order to avoid false detections, as soon as the dissimilarity value obtained for a true scene cut goes out from the sliding window, a correcting action, named *lowering condition*, is applied, by comparing the value of X_i to the value given by $(m_n + \sigma_n + X_i/2)$ and taking the lowest one, as the new value for scene cut detection.

Figure 8 shows the behaviour of the dissimilarity measure X_i, and of the threshold $T(n)$ used for scene cut detection,



(a) (b) (c)

Figure 6: Local frame dynamics evidenced by virtual frame partition.

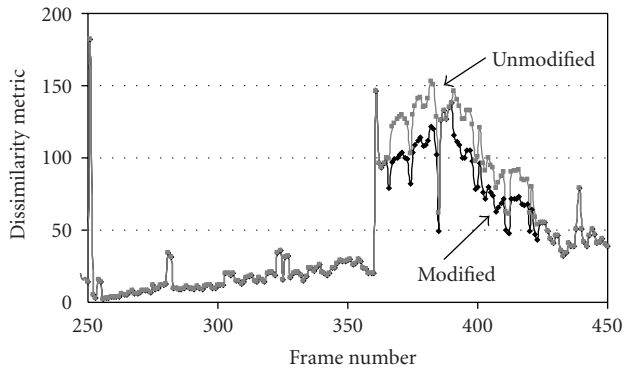


Figure 7: Improvement of the modified dissimilarity metric behaviour in the case of high-motion video sequences.

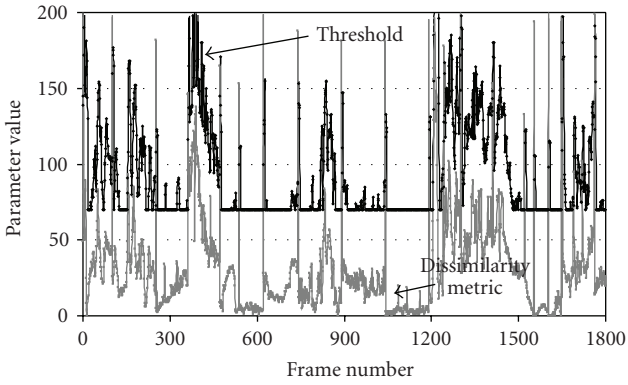


Figure 8: Threshold $T(n)$ and dissimilarity metric X_i variations over a whole 12.5 fps video sequence with no losses.

over a test video sequence of 12.5 fps frame rate, with no losses. The dynamics obtained by modifying the detection algorithm, according to the solutions described above, allow to adaptively change the detection threshold in order to increase the correct detection rate and reduce the false or missed detections.

Before moving to the experimental evaluation of the proposed detector, as discussed in the next Section, Figure 9 summarizes the detector main components and the data processing flow in a block diagram fashion.

4. Experimental Evaluation and Results

As a preliminary evaluation, the proposed algorithm has been compared to other scene cut detection solutions, by the application of the MSU Video Quality Measurement Tool [7], which is able to implement four different similarity metrics, defined as follows:

(1) *Pixel-Level Comparison*: the similarity measure of two frames is the SAD computed over the intensity values of corresponding pixels;

(2) *Global Histogram*: the histogram is obtained by counting the number of pixels in the frame, with specified luminance level. The difference between two histograms is then determined by calculating the SAD over the pixels having the same luminance level;

(3) *Block-Based Histogram*: each frame is divided into 16×16 pixels blocks. For each block, a luminance distribution histogram is constructed, the similarity measure for each block is obtained, and the average value of these measures is accepted as the frame similarity measure as the frame similarity measure;

(4) *Motion-Based Similarity Measure*: a Motion Estimation algorithm with block size 16×16 pixels is applied on adjacent frames. The average value of the Motion Vector errors is accepted as the similarity value.

The MSU tool has been applied offline, and comparisons with the proposed detector were performed on a 12.5 fps CIF video sequence in YUV format, not affected by losses, showing 38 true scene changes located by visual inspection. The test video sequence has been generated by composition of 29 subsequences collected from the Video Quality Expert Group repository, in order to include as many different effects as possible, such has low and high motion, panning, zooming, light variations, scene changes, and so on.

Results presented in Table 2 confirm the effectiveness of the proposed detector: besides being able to provide a local-scale information about the sequence dynamics, which is not provided by the MSU software tool, the proposed detector has been designed to process sequences affected by losses, as reported in the following discussion.

In order to evaluate the performance of the proposed scene cut detection algorithm in presence of losses, tests have been executed on H.264/AVC encoded video sequences, encapsulated according to the Real Time Protocol (RTP) packet format. Before applying the H.264/AVC reference

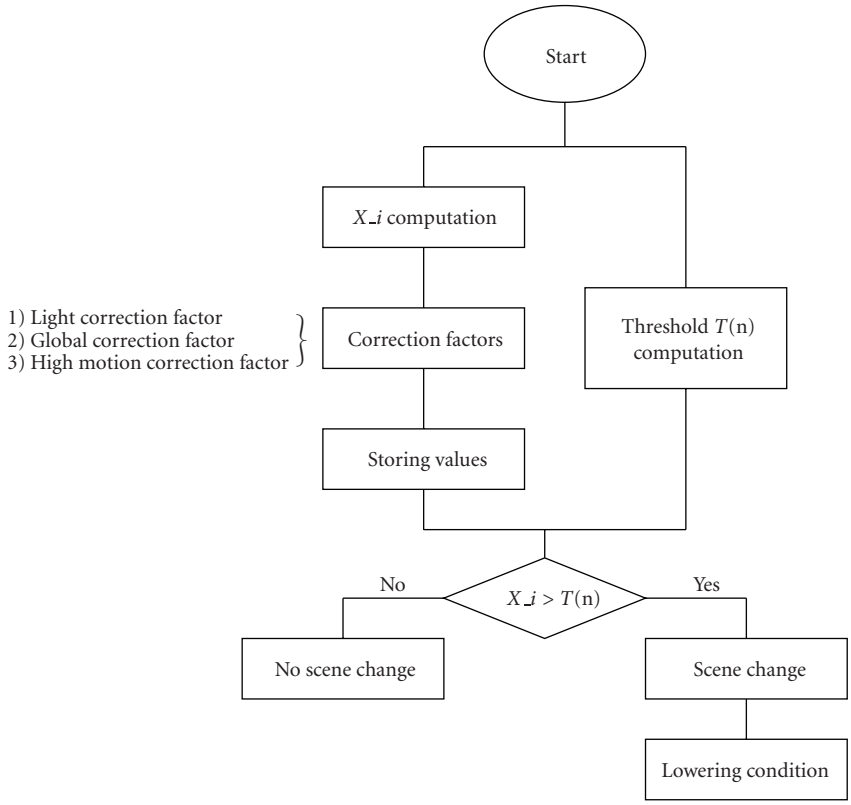


Figure 9: The scene change detector main components and processing flow.

Frame $(n-1)$ (a)

Frame (n) (b)

Figure 10: False scene change detection caused by lost MBs in the 12.5 fps sequence, for a 4% packet loss rate.

decoder properly modified to include the detector, H.264 encoded bitstreams have been subjected to a packet erasure process, in such a way as to simulate different packet loss rates, of 1%, 2%, 4%, and 10%, which may be considered representative of realistic environments, such as video transmission over packet-based wireless networks. For each packet loss rate value, 5 simulations over the same video bitstream have been executed, and the average result was considered, in order to account for different error patterns randomly generated. Simulations have been performed over sequences encoded at 25 fps, and over their decimated versions at 12.5 fps, in order to test the detector behaviour with respect to frame rate. Other main encoder parameters have been set as follows: the selected H.264 profile is Baseline, with a CIF, YUV 4 : 2 : 0 format, QPISlice = 28 and QPPSlice = 28.

The detector performance is defined with respect to two parameters, namely, the *Recall* (Re) and *Precision* (Pr) rates, that, in their turn, depend on the number of fake

Table 2: Performance comparison of the proposed detector and four different detection algorithms implemented by the MSU software tool, for a 12.5 fps CIF sequence with no losses.

Detection Algorithm	no. Correct Detections	no. False Detections
Proposed detector	38	0
MSU - 1	38	5
MSU - 2	37	13
MSU - 3	38	1
MSU - 4	38	3

Table 3: *Recall* and *Precision* average performance of the detector, for the same video sequence at 25 and 12.5 fps, and different packet loss rates.

Packet Loss Rate	25 fps		12.5 fps	
	Recall	Precision	Recall	Precision
No loss	1	1	1	1
1%	1	1	0.994	0.994
2%	0.994	1	0.976	1
4%	0.988	1	0.964	0.988
10%	0.982	0.994	0.952	0.966

detections (FD), the number of missed detections (MD), and the number of correct detections (CD) over a given sequence, as follows:

$$\text{Re} = \frac{\text{CD}}{\text{CD} + \text{MD}}, \qquad \text{Pr} = \frac{\text{CD}}{\text{CD} + \text{FD}}. \tag{7}$$

The test sequence adopted shows 38 true scene changes, revealed through visual inspection. Table 3 reports the *Recall* and *Precision* performance of the detector, for the same sequence at 25 fps and 12.5 fps, and for different packet loss rates; the values in the Table refer to average performance evaluated over 5 decoding iterations for each packet loss rate.

Results show a very satisfactory behaviour of the proposed detector, either at 25 and 12.5 fps, even if with a very small degradation in the latter case, with a *Recall* and a *Precision* figure always greater than 0.95. As reasonable and expected, performances degrade as the packet loss rate increases, according to the frame areas affected by data losses that may cause a false detection, or a missed one. Figure 10 shows a peculiar case for the 12.5 fps sequence at a 4% packet loss rate: missing MBs in the frame (represented as green MBs), due to packet losses, cause a variation in the dissimilarity metric which determines a false scene change detection. If losses do not occur, the detector correctly does not reveal any scene change, despite the evident variation of the frame in its bottom areas.

5. Conclusion

This paper presented an optimized scene change detector for H.264/AVC video sequences, based on a dynamic threshold model properly designed to be applied at the decoder side, even in presence of losses and errors in the received bitstreams. On the contrary, most of the detection algorithms presented in the previous literature are conceived for application at the encoder side, and cannot deal with data losses in the video bitstreams. The proposed detector, as discussed in the paper, besides performing better than the most popular detection solutions over error-free video sequences, also shows remarkable results when dealing with missing information. Given its effectiveness and joining its low-complexity and limited resource requirements, the proposed detector could be effectively included in error concealment strategies applied at the decoder, in order to improve the final video quality delivered to the user and compensate for quality degradation due to error-prone transmissions.

References

[1] A. Dimou, O. Nemethova, and M. Rupp, "Scene change detection for H.264 using dynamic threshold techniques," in *Proceedings of the 5th EURASIP Conference on Speech and Image Processing, Multimedia Communications and Service*, Smolenice, Slovak Republic, July 2005.

[2] Y.-H. Ai, W. Ye, S.-L. Feng, B. Hu, and M. Xie, "Predictive picture refresh based on scene-context reference picture for video transmission," in *Proceedings of the International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM '06)*, pp. 1–4, Wuhan, China, September 2007.

[3] Z. Chen, G. Qiu, Y. Lu, et al., "Improving video coding at scene cuts using attention based adaptive bit allocation," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '07)*, pp. 3634–3638, New Orleans, Calif, USA, May 2007.

[4] W.-Y. Kung, C.-S. Kim, and C.-C. J. Kuo, "Spatial and temporal error concealment techniques for video transmission over noisy channels," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 16, no. 7, pp. 789–802, 2006.

[5] J. Sastre, P. Usach, A. Moya, V. Naranjo, and J. M. Lopez, "Shot detection method for low bit-rate H.264 video coding," in *Proceedings of the 14th European Signal Processing Conference (EUSIPCO '06)*, Florence, Italy, September 2006.

[6] X. Yi and N. Ling, "Fast pixel-based video scene change detection," in *Proceedings of the IEEE International Symposium on Circuits and Systems (ISCAS '05)*, vol. 4, pp. 3443–3446, Kobe, Japan, May 2005.

[7] D. Vatolin, "The MSU Video Quality Measurement Tool," http://compression.ru/video/quality_measure/video_measurement_tool_en.html, August 2009.