Identification and analysis of video subsequence using bipartite graph matching

Silvio Jamil F. Guimarães
PUC Minas – Belo Horizonte – Brazil
sjamil@pucminas.br

Zenilton Kleber G. do Patrocínio Jr.
PUC Minas – Belo Horizonte – Brazil
zenilton@pucminas.br

ABSTRACT

Subsequence identification consists in identifying real positions of a specific video clip in a video stream together with the operations that may be used to transform the former into a subsequence from the latter. To cope with this problem, we propose a new approach considering a bipartite graph matching to measure video clip similarity with a target video stream which has not been preprocessed. We show that our approach locates edited video clips allowing insertion, removal and replacement operations. Experimental results demonstrate that our method performance achieve 93% recall with 93% precision, though it has a low computational cost since its classifications step is extremely simple.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Search process—*multimedia and video streams*; H.5.1 [**Multimedia Information Systems**]: Video

General Terms

Algorithms, Experimentation, Performance

Keywords

Video retrieval, Graph bipartite, Video clip localization

1. INTRODUCTION

Traditionally, multimedia information has been analogically stored and manually indexed. Due to advances in multimedia technology, techniques to video retrieval are increasing. Unfortunately, the recall and precision of these systems depend on the similarity measure that are used to retrieve information. Nowadays, due to improvements on digitalization and compression technologies, database systems are used to store images and videos, together with their metadata and associated taxonomy. Thus, there is an increasing search for efficient systems to process and index image, audio and video information, mainly for the purposes of information retrieval.

The task of automatic segmentation, indexing, and retrieval of large amount of video data has important applications in archive management, entertainment, media production, rights control, surveillance, and many more.

The complex task of video segmenting (mainly in presence of gradual transitions and motion) and indexing faces the challenge of coping with the exponential growth of the Internet, that has resulted in a massive publication and sharing of video content and an increase in the number of duplicated documents; and the distribution across communication channels, like TV, resulting in thousands of hours of streaming broadcast media. According to [5, 6, 7], one important application of video content management is broadcast monitoring for the purpose of market analysis. The video clip localization, as it will be referred during this paper, has arisen in the domain of broadcast television, and consists of identifying the real locations of a specific video clip in a target video stream. The main issues that must be considered during video clip localization are: (i) the definition of the dissimilarity measures of video clips; (ii) the processing time of the algorithms due to the huge amount of information that must be analyzed; (iii) the insertion of intentional and non-intentional distortions; (iv) different frame rates; and (v) edition of the videos. The selection of the feature used to compute dissimilarity measure has an important role in content-based image retrieval and has been largely explored [19]. [4] showed that the performance of the features is task dependent, and that it is hard to select the best feature for an specific task without empirical studies. Low-complexity features and matching algorithms can work together to increase matching performance.

Current methods for solving the video retrieval/localization problem can be grouped in two main approaches: (i) computation of video signatures after temporal video segmentation, as described in [6, 11, 14]; and (ii) use of matching algorithms after transformation of the video frame content into a feature vector, as described in [1, 8, 12, 16, 20]. When video signatures are used, methods for temporal video segmentation must be applied before signature calculation [2]. Although temporal video segmentation is a widely studied problem, it represents an important issue that has to be considered, as it increases complexity of the algorithms and affects matching performance. For methods based on string matching algorithms, the efficiency of the these algorithms must be taken into account, when compared to image/video identification algorithms. [1] and [12] successfully applied

Table 1: Comparison of some approaches for video clip localization (adapted from [18]).

	Sliding window [3]	Temporal order [10, 21]	Vstring edit [1]	Multi-level [13]	Graph approach [18]	Dense [20]	BMH [8]	Our proposed method
Shot/Frame Matching	shot	shot	shot	shot	shot	frame	frame	frame
Temporal order	no	yes	yes	yes	yes	possible	yes	possible
Clip filtering	no	no	no	no	yes	no	no	no
Online Clip Segment.	yes	no	no	no	yes	yes	no	no
Preprocessing	yes	yes	yes	yes	yes	yes	no	no
Video edition	no	no	no	no	no	yes	no	yes

the longest common substring (LCS) algorithm to deal with the problem. However, it requires a $O(mn)$ space and time cost, in which m and n represent the size of the query and target video clips, respectively. In [8], it is proposed a modified version of the fastest algorithm for exact string matching, the Boyer-Moore-Horspool (BMH) [9, 15], to deal with the problem of video location and counting. In [16] we proposed a new approach to cope with the problem of video clip localization using the maximum cardinality matching of a bipartite graph, however this approach copes only with exact video matching. [20] also used bipartite graph matching, however the first step of the algorithm is related to analyze the video query and video target to identify frame similarities and possible locations of the matching.

In the present paper, we present a modified version of our previous approach [16], which is able to deal with general case in video clip localization problem, allowing insertion, removal and replacement operations on the video clip. Our method not only solves the approximate video clip (or subsequence) localization problem but also gives a precise description of the operation set that are necessary to transform the query video into a clip of the target video stream. Our approach copes with the problem of video clip localization using the maximum cardinality matching of a bipartite graph. It is important to note that we use this information to shift the video query on video target in order to identify the probable location of the matching. For a set of frames from a query video clip and from a target video a graph is constructed based on a similarity measure between each pair of frames (illustrated in Fig. 1(a)). The size of the maximum cardinality matching of the graph defines a video similarity measure that is used for video identification. This information together with the maximum distance between any two frames of the target video which belong to the maximum cardinality matching are used to analyze hit occurrences and to identify precisely which operations are necessary to transform the query video into the target video content. Table 1 presents a comparison between some approaches found in the literature. The first difference between our proposed approach and the others is associated with the matching used to establish the video similarity. Most of the works consider that the target video has been preprocessed and online/offline segmented into video clips which are used by the search procedure, while ours can be applied directly to a target video stream without any preprocessing since it uses frame-based similarity measures. With the exponential growth of the Internet, the storage of segmented videos may become an intractable problem. Our approach allows us to perform video localization over a streaming media downloaded directly from the Internet, while the others need to download, segment and store segmented video clips before starting video clip localization.

Moreover, our approach can be applied without considering temporal order constraints, which allows us to locate the position of the query video even if the video has been edited and its frames reordered (illustrated in Fig. 1(b)). Current version of our algorithm also deals with insertion, removal and replacement of frames/shots, and it also allows changes in temporal order of query video clip frames/shots. However, our approach can be applied to the traditional (exact) video clip localization problem using dynamic programming and temporal order similarity. On the other hand, clip editing and reordering has become a desired feature on the new context of online video delivery. As mentioned in [17], users expect to be able to manipulate video content based on choices such as desired portions of video, ordering and "crop/stitch" of clips. New coding schemes that consider this novel scenario have been included in most recent standards such as MPEG-7 and MPEG-21. Nevertheless, since our approach is based on frame similarity measures, it may present an efficiency problem. This issue has been addressed by employing a *shift strategy* based on the size of the maximum cardinality matching.

This paper is organized as follows. In Section 2, the problem of video subsequence identification is described, together with some formal definitions of the field. In Section 3, we present a methodology to identify the locations of a query video clip using bipartite graph matching. In Section 4, we discuss about the experiments and the setting of algorithm parameters. Finally, in Section 5, we give some conclusions and suggest future works.

2. PROBLEM DEFINITION

Let $\mathbb{A} \subset \mathbb{N}^2$, $\mathbb{A} = \{0, \ldots, \mathrm{W}-1\} \times \{0, \ldots, \mathrm{H}-1\}$, where W and H are the width and height of each frame, respectively, and, $\mathbb{T} \subset \mathbb{N}$, $\mathbb{T} = \{0, \ldots, N-1\}$, where N is the length of a video. Frame, video, and video clip can be defined as follows.

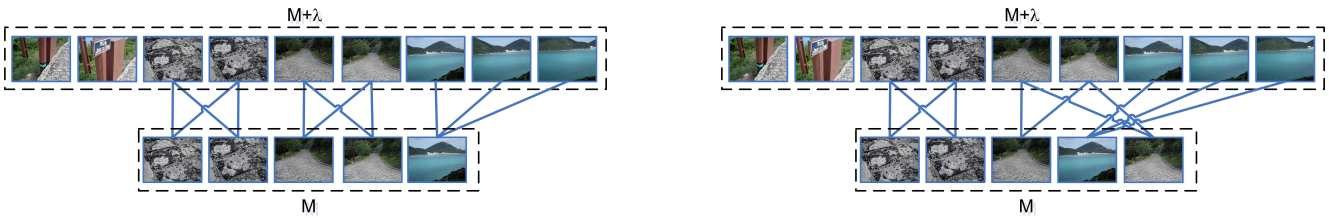
Definition 1 (Frame) *A frame f is a function from $\mathbb{A}$ to $\mathbb{Z}^3$, where for each spatial position (x, y) in $\mathbb{A}$, $f(x, y)$ represents a color value[1] at pixel location (x, y).*

Definition 2 (Video) *A video V_N, in domain $2\mathbb{D} \times \mathbb{T}$, can be seen as a temporally ordered sequence of frames f. It is described by*

$$\mathrm{V}_N = (\, f \,)_{\, t \in \mathbb{T}} \tag{1}$$

where N is the number of frames contained in the video.

[1] without loss of generality, a frame can be defined by a function from $\mathbb{A}$ to $\mathbb{Z}$ to represent a grayscale value at location (x, y).



(a) query video without edition

(b) edited (frame-reordered) query video

Figure 1: Frame similarity graph.

Definition 3 (Video clip) *Let V_N be a video. A j-sized video clip $\mathrm{C}_{k,j}$ is a temporally ordered subsequence of frames from V_N which starts at frame k with j frames. It can be described by*

$$\mathrm{C}_{k,j} = (\, f_t \mid f_t \in \mathrm{V}_N)_{t\in[k,k+j-1]}. \tag{2}$$

in which $k \leq N - j$.

It is obvious that $\mathrm{C}_{k,0} = \emptyset, \forall k$, and $\mathrm{C}_{0,N} = \mathrm{V}_N$. Moreover, from two distinct videos one can produce distinct video clips, so a superscript will be used to indicated which video is the frame source. Therefore video clip $\mathrm{C}^X_{\cdot,\cdot}$ contains frames f^X_t from video V^X_N, and, $\mathrm{C}^X_{0,N} = \mathrm{V}^X_N$.

A dataset may contain an altered version of a specified video clip, in which some frames may have been added, removed, or even replaced by others. Consequently, we can define a distance between two video clips based only on the number of frame insertions, removals, and replacements, as follows. It is important to note that frame reordering is permitted without changing the video edit distance.

Definition 4 (Video clip edit distance – $\mathrm{d}(\mathrm{C}^{X_1}_{k_1,i}, \mathrm{C}^{X_2}_{k_2,j})$) *Let $\mathrm{C}^{X_1}_{k_1,i}$ and $\mathrm{C}^{X_2}_{k_2,j}$ be two video clips of size i and j from videos V^{X_1} and V^{X_2}, respectively. Then the edit distance between $\mathrm{C}^{X_1}_{k_1,i}$ and $\mathrm{C}^{X_2}_{k_2,j}$ is equal to the minimum number of operations (insertions, removals and replacements) needed to transform $\mathrm{C}^{X_1}_{k_1,i}$ into $\mathrm{C}^{X_2}_{k_2,j}$, and vice-versa, regardless of frame temporal ordering.*

Let $\mathrm{V}^{X_1}_4$ be the original video Y_1 presented by Fig. 2(a), so $\mathrm{C}^{X_1}_{0,4} = \mathrm{V}^{X_1}_4$. Let $Y_2 = \mathrm{V}^{X_2}_5$, $Y_3 = \mathrm{V}^{X_3}_3$, $Y_4 = \mathrm{V}^{X_4}_4$, and $Y_5 = \mathrm{V}^{X_5}_5$ be the altered versions presented by Fig. 2(b), 2(c), 2(d), and 2(e), respectively.

The edit distance between Y_1 and Y_2 is equal to 1 since both videos contain the same frames except by $f^{X_2}_2$, which is not present in the original video Y_1, i.e., $\mathrm{C}^{X_1}_{0,2} = \mathrm{C}^{X_2}_{0,2}$ and $\mathrm{C}^{X_1}_{2,2} = \mathrm{C}^{X_2}_{3,2}$. Therefore, $\mathrm{d}(\mathrm{C}^{X_1}_{0,4}, \mathrm{C}^{X_2}_{0,5}) = 1$ because only one insertion is need to transform Y_1 into Y_2. Analogously, the edit distance between Y_1 and Y_3 is also equal to 1 since both videos contain the same frames except by $f^{X_1}_0$, which is not present in altered video Y_3, i.e., $\mathrm{C}^{X_1}_{1,3} = \mathrm{C}^{X_3}_{0,3}$. So, $\mathrm{d}(\mathrm{C}^{X_1}_{0,4}, \mathrm{C}^{X_3}_{0,3}) = 1$ because only one removal is need to transform Y_1 into Y_3. The edit distance between Y_1 and Y_4 is equal to 1 since frame $f^{X_1}_1$ has been replaced by $f^{X_4}_1$, i.e., $\mathrm{C}^{X_1}_{0,1} = \mathrm{C}^{X_4}_{0,1}$ and $\mathrm{C}^{X_1}_{2,2} = \mathrm{C}^{X_4}_{2,2}$. Therefore, $\mathrm{d}(\mathrm{C}^{X_1}_{0,4}, \mathrm{C}^{X_4}_{0,4}) = 1$ because one replacement is need to transform Y_1 into Y_4. Finally, the edit distance between Y_1 and Y_5 is equal to 0 since there is only a temporal reordering of frames without any insertion, removal or replacement. Therefore, $\mathrm{d}(\mathrm{C}^{X_1}_{0,4}, \mathrm{C}^{X_5}_{0,4}) = 0$ because no insertion, removal and replacement is need to transform Y_1 into Y_5.

In the previous example, two frames were considered equal if they were exactly alike, but this may not be always true due to minor differences in vector quantization used in digital representation for both videos. In order to establish if two distinct frames of two distinct video clips are similar, we define *frame similarity* as follows.

Definition 5 (Frame similarity) *Let $f^{X_1}_{t_1}$ and $f^{X_2}_{t_2}$ be two video frames at locations t_1 and t_2 from video clips $\mathrm{C}^{X_1}_{k_1,i}$ and $\mathrm{C}^{X_2}_{k_2,j}$, respectively. Two frames are similar if a distance measure $\mathcal{D}(f^{X_1}_{t_1}, f^{X_2}_{t_2})$ between them is smaller than a specified threshold δ. The frame similarity is defined as*

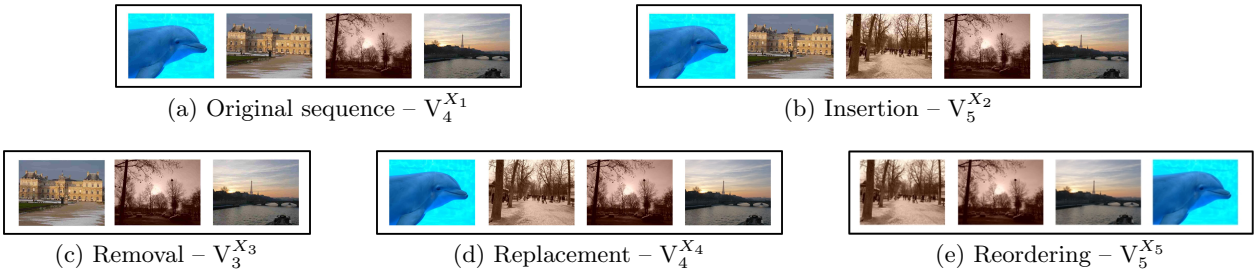
$$\mathrm{FS}(f^{X_1}_{t_1}, f^{X_2}_{t_2}, \delta) = \begin{cases} 1, & \text{if } \mathcal{D}(f^{X_1}_{t_1}, f^{X_2}_{t_2}) \leq \delta; \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

There are several choices for $\mathcal{D}(f^{X_1}_{t_1}, f^{X_2}_{t_2})$, i.e., the distance measure between two frames, e.g. histogram/frame difference, histogram intersection, difference of histograms means, and others. In our experiments, we have adopted histogram intersection as the distance measure between frames. After selecting one, it is possible to construct a frame similarity graph based on a query video V^Q_M and a $(M+\lambda)$-sized video clip of target video $C^T_{k,M+\lambda}$, in which λ is the maximum allowed edit distance, as follows.

Definition 6 (Frame similarity graph – $\mathrm{G}^\delta_{k,\lambda}$) *Let V^Q_M and V^T_N be a query video with M frames and a target video with N frames, respectively, and let $\mathrm{C}^T_{k,M+\lambda}$ be a $(M+\lambda)$-sized video clip which starts at frame k of target video with $(M+\lambda)$ frames. A frame similarity graph $\mathrm{G}^\delta_{k,\lambda} = (\mathrm{N}^Q \cup \mathrm{N}^T_{k,\lambda}, \mathrm{E}^\delta_{k,\lambda})$ is a bipartite graph. Each node $v^Q_{t_1} \in \mathrm{N}^Q$ represents a frame from the query video $f^Q_{t_1} \in \mathrm{C}^Q_{0,M}$ and each node $v^T_{t_2} \in \mathrm{N}^T_{k,\lambda}$ represents a frame from the target video clip $f^T_{t_2} \in \mathrm{C}^T_{k,M+\lambda}$. There is an edge $e \in \mathrm{E}^\delta_{k,\lambda}$ between $v^Q_{t_1}$ and $v^T_{t_2}$ if frame similarity of associated frames is equal to 1, i.e.,*

$$\mathrm{E}^\delta_{k,\lambda} = \{\, (\, v^Q_{t_1}\, ,\, v^T_{t_2}\,) \mid v^Q_{t_1} \in \mathrm{N}^Q, v^T_{t_2} \in \mathrm{N}^T_{k,\lambda}, \mathrm{FS}(f^Q_{t_1}, f^T_{t_2}, \delta) = 1\}. \tag{4}$$

As illustrated in Fig. 1, in order to allow up to λ operations of insertion into the query video clip, we match the query



(a) Original sequence – $V_4^{X_1}$ (b) Insertion – $V_5^{X_2}$

(c) Removal – $V_3^{X_3}$ (d) Replacement – $V_4^{X_4}$ (e) Reordering – $V_5^{X_5}$

Figure 2: Video clip modifications.

video to a video clip of the target video stream whose size (number of frames) is equal to the query video size plus the maximum allowed edit distance (λ). Conversely, in order to allow up to λ operations of removal from the query video clip, we match the query video to a video clip of the target video stream whose size (number of frames) is equal at least to the query video size minus the maximum allowed edit distance (λ). Thus, the size of the target video clip must vary in range $[M-\lambda, M+\lambda]$. And, in order to ensure that we consider all insertions, removals and replacements allowed by the maximum edit distance, we need to match the query video clip (of size M) to a target video clip (of size $M+\lambda$).

In this paper, video subsequence identification problem with (or without) any changes in the video content (including changes in its temporal ordering) will be addressed using *maximum cardinality matching*. To do so, we define *matching* and *maximum cardinality matching* as follows.

Definition 7 (Matching – $\mathrm{M}_{k,\lambda}^{\delta}$) *Let $\mathrm{G}_{k,\lambda}^{\delta} = (\mathrm{N}^{Q} \cup \mathrm{N}_{k,\lambda}^{T}, \mathrm{E}_{k,\lambda}^{\delta})$ be a frame similarity graph. A subset $\mathrm{M}_{k,\lambda}^{\delta} \subseteq \mathrm{E}_{k,\lambda}^{\delta}$ is a match if any two edges in $\mathrm{M}_{k,\lambda}^{\delta}$ are not adjacent. The size of matching $\mathrm{M}_{k,\lambda}^{\delta}$ is the number of edges in $\mathrm{M}_{k,\lambda}^{\delta}$, written as $|\mathrm{M}_{k,\lambda}^{\delta}|$.*

Definition 8 (Maximum cardinality matching – $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$) *Let $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$ be a matching in a frame similarity graph $\mathrm{G}_{k,\lambda}^{\delta}$. So, $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$ is the maximum cardinality matching (MCM) if there is no other matching $\mathrm{M}_{k,\lambda}^{\delta}$ in $\mathrm{G}_{k,\lambda}^{\delta}$ such that $|\mathrm{M}_{k,\lambda}^{\delta}| > |\overline{\mathrm{M}_{k,\lambda}^{\delta}}|$.*

During the search, a hit can be associated with the size of the MCM, i.e., if $M-\lambda \leq |\overline{\mathrm{M}_{k,\lambda}^{\delta}}| \leq M$ then a hit may have been found. However, in order to prevent false positives, we need to evaluate the largest distance between any two frames of the target video clip which belong to the MCM found. This evaluation is necessary to cope with insertions and replacements since the size of MCM may be smaller than the video query size M.

Let $\mathcal{F}$ be the set of frames associated with the MCM $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$ and it contains frames from the query and the target video clips, i.e., $\mathcal{F} = \mathcal{F}^{Q} \cup \mathcal{F}_{k,\lambda}^{T}$, in which $\mathcal{F}^{Q}$ and $\mathcal{F}_{k,\lambda}^{T}$ are frame sets from the query and the target video clips, respectively. So, $\mathcal{F}^{Q} = (\, f_t^{Q} \mid f_t^{Q} \in \mathrm{C}_{0,M}^{Q}, t \in [0, M-1])$, while $\mathcal{F}_{k,\lambda}^{T} = (\, f_t^{T} \mid f_t^{T} \in \mathrm{C}_{k,M+\lambda}^{T}, t \in [k, k+M+\lambda-1])$. It is also easy to verify that $|\overline{\mathrm{M}_{k,\lambda}^{\delta}}| = |\mathcal{F}^{Q}| = |\mathcal{F}_{k,\lambda}^{T}|$. So, the maximum distance between any two frames of the target video clip which belong to the MCM found could be defined as follows.

Definition 9 (Maximum MCM distance – $D(\overline{\mathrm{M}_{k,\lambda}^{\delta}})$) *Let $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$ be the MCM in a frame similarity graph $\mathrm{G}_{k,\lambda}^{\delta}$ and $\mathcal{F} = \mathcal{F}^{Q} \cup \mathcal{F}_{k,\lambda}^{T}$ be the set of frames associated with the MCM. Let t_f and t_l represent the locations of the first and the last frames of the target video clip that belong to the MCM $\overline{\mathrm{M}_{k,\lambda}^{\delta}}$, so $t_f = \min\{\, t \mid f_t^{T} \in \mathcal{F}_{k,\lambda}^{T}\}$ and $t_l = \max\{\, t \mid f_t^{T} \in \mathcal{F}_{k,\lambda}^{T}\}$. The maximum distance between any two frames of the target video clip which belong to the MCM is defined as*

$$D(\overline{\mathrm{M}_{k,\lambda}^{\delta}}) = t_l - t_f + 1. \tag{5}$$

Using the maximum MCM distance together with the size of the MCM, we can identify not only a hit occurrence but also its type. The type of a hit occurrence represents which operations that are necessary to transform the query video into the target video clip. Therefore, we can define 06 (six) types of hit occurrence as follows:

- **T1 – Exact hit (ExHit):** represents an exact match between the query video and the target video clip, so $|\overline{\mathrm{M}_{k,\lambda}^{\delta}}| = M$ in order to guarantee that every frame of the query video appears in the target[2], and $D(\overline{\mathrm{M}_{k,\lambda}^{\delta}}) = |\overline{\mathrm{M}_{k,\lambda}^{\delta}}|$ to prevent that any additional frame has been inserted among the target frames;
- **T2 – Insertion hit (InsHit):** represents a match between the query video and the target video clip in which additional frames have been inserted into the target, so $|\overline{\mathrm{M}_{k,\lambda}^{\delta}}| = M$ in order to guarantee that every frame of the query video appears in the target video clip, and $|\overline{\mathrm{M}_{k,\lambda}^{\delta}}| < D(\overline{\mathrm{M}_{k,\lambda}^{\delta}}) \leq M+\lambda$ to ensure that no more than λ additional frames have been inserted among the target frames;
- **T3 – Removal hit (RemHit):** represents a match between the query video and the target video clip in which some frames have been removed, so $M-\lambda \leq |\overline{\mathrm{M}_{k,\lambda}^{\delta}}| < M$ in order to guarantee that some frames of the query video do not appear in the target, and $D(\overline{\mathrm{M}_{k,\lambda}^{\delta}}) = |\overline{\mathrm{M}_{k,\lambda}^{\delta}}|$ to prevent that any additional frame has been inserted among the target frames;

[2]i.e., every frame of the query video is similar to at least one distinct frame of the target video clip.

Table 2: Examples of hit occurrence with $\lambda = 2$.

Query	Target	Hit type	$\lvert\overline{M^{\delta}_{k,\lambda}}\rvert$	$D(.)$
1 2 3 4	... 1 2 3 4 - - ...	ExHit	4	4
1 2 3 4	... 1 2 3 5 4 - ...	InsHit	4	5
1 2 3 4	... 1 2 3 5 5 4 ...	InsHit	4	6
1 2 3 4	... 1 2 - - - - ...	RemHit	2	2
1 2 3 4	... 1 2 3 - - - ...	RemHit	3	3
1 2 3 4	... 1 2 5 4 - - ...	RepHit	3	4
1 2 3 4	... 1 5 5 4 - - ...	RepHit	2	4
1 2 3 4	... 1 5 4 - - - ...	RepRHit	2	3
1 2 3 4	... 1 2 5 5 4 - ...	RepIHit	3	5
1 2 3 4	... 1 2 5 5 5 4 ...	No hit	3	6

- **T4 – Replacement hit (RepHit):** represents a match between the query video and the target video clip in which some frames have been replaced, so $M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M$ in order to guarantee that some frames of the query video do not appear in the target, and $D(\overline{M^{\delta}_{k,\lambda}}) = M$ to ensure that every missing target frame has been replaced by another one;

- **T5 – Replacement and Removal hit (RepRHit):** represents a match between the query video and the target video clip in which some frames have been replaced and others have been removed, so $M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M$ in order to guarantee that some frames of the query video do not appear in the target, and $\lvert\overline{M^{\delta}_{k,\lambda}}\rvert < D(\overline{M^{\delta}_{k,\lambda}}) < M$ to ensure that some of the frames have been removed since $D(\overline{M^{\delta}_{k,\lambda}}) < M$ (it is not only a replacement) but also that some frames have been replaced since $D(\overline{M^{\delta}_{k,\lambda}}) > \lvert\overline{M^{\delta}_{k,\lambda}}\rvert$ (it is not only a removal);

- **T6 – Replacement and Insertion hit (RepIHit):** represents a match between the query video and the target video clip in which some frames have been replaced and others have been inserted, so $M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M$ in order to guarantee that some frames of the query video do not appear in the target (they have been replaced), and $M < D(\overline{M^{\delta}_{k,\lambda}}) \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert + \lambda$ to ensure that additional frames have been inserted among the target frames since $D(\overline{M^{\delta}_{k,\lambda}}) > M$ (it is not only a replacement) but also to limit the edit distance to a maximum of λ operations, i.e., $D(\overline{M^{\delta}_{k,\lambda}}) - \lvert\overline{M^{\delta}_{k,\lambda}}\rvert \leq \lambda$.

Table 2 presents examples of hit occurrences with a maximum edit distance $\lambda = 2$. Without loss of generality, each frame is represented only by a single feature value (an integer one) and two frames are similar if their features are exactly the same. First column shows the sequence of frames from a query video clip, while the second column presents the sequence of frames from the target video associated with a hit occurrence. Hit type is presented at the third column, followed by MCM size and maximum MCM distance at forth and fifth columns, respectively. Therefore, using MCM size and maximum MCM distance, a function to detect a hit occurrence can be defined as follows.

Definition 10 (Hit function – $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}})$) *Let $\overline{M^{\delta}_{k,\lambda}}$ be the MCM in a frame similarity graph $G^{\delta}_{k,\lambda}$. So a function $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}})$ to detect a hit occurrence can be defined as*

$$\mathrm{H}(\overline{M^{\delta}_{k,\lambda}}) = \begin{cases} 1, & \text{if } \lvert\overline{M^{\delta}_{k,\lambda}}\rvert = M \text{ and } D(\overline{M^{\delta}_{k,\lambda}}) = \lvert\overline{M^{\delta}_{k,\lambda}}\rvert; \\ 2, & \text{if } \lvert\overline{M^{\delta}_{k,\lambda}}\rvert = M \text{ and } \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < D(\overline{M^{\delta}_{k,\lambda}}) \leq M + \lambda; \\ 3, & \text{if } M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M \text{ and } D(\overline{M^{\delta}_{k,\lambda}}) = \lvert\overline{M^{\delta}_{k,\lambda}}\rvert; \\ 4, & \text{if } M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M \text{ and } D(\overline{M^{\delta}_{k,\lambda}}) = M; \\ 5, & \text{if } M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M \text{ and } \lvert\overline{M^{\delta}_{k,\lambda}}\rvert \leq D(\overline{M^{\delta}_{k,\lambda}}) < M; \\ 6, & \text{if } M - \lambda \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert < M \text{ and } M < D(\overline{M^{\delta}_{k,\lambda}}) \leq \lvert\overline{M^{\delta}_{k,\lambda}}\rvert + \lambda; \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

The hit function returns a value greater than zero for a hit occurrence, while zero corresponds to a miss. Moreover, the hit function value corresponds to the type of the hit occurrence, i.e., $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}}) = 1$ for a **T1 hit** (or ExHit), $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}}) = 2$ for a **T2 hit** (or InsHit), and so on. As described before, the proposed hit function ignores the temporal reordering of frames since we are interested in identifying video subsequence with similar content.

Finally, video subsequence identification problem can be stated.

Definition 11 (Video subsequence identification) *Let $\overline{M^{\delta}_{k,\lambda}}$ be a MCM of a frame similarity graph $G^{\delta}_{k,\lambda}$ with a maximum edit distance λ and a threshold δ. The video subsequence identification (VSI) problem corresponds to identify the locations of a query video V^{Q}_{M} at the target video V^{T}_{N} together with the operations necessary to transform the former into a subsequence from the latter. A hit at location k is found if there is a $(M + \lambda)$-sized video clip $C^{T}_{k,M+\lambda}$ of V^{T}_{N} that matches with $C^{Q}_{0,M}$ according to the hit function $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}})$. Thus, this problem can be stated as*

$$VSI(V^{Q}_{M}, V^{T}_{N}, \delta, \lambda) = \{\, k \in [0, N-1] \mid \mathrm{H}(\overline{M^{\delta}_{k,\lambda}}) \neq 0 \,\}. \quad (7)$$

3. METHODOLOGY

As described before, the main goal of the video subsequence identification problem is to identify occurrences of a query video in a video target stream together with the operations necessary to transform the former into a subsequence from the latter, see Fig. 3. One of the key steps of the process is feature extraction. Choosing an appropriate feature that enhances performance of a matching algorithm is not a trivial task. Therefore, empirical studies are the best way to get insights of which feature should be used for each case.

3.1 Identification procedure

Fig. 4 presents our identification procedure. It scans over target video stream, looking for a video clip that matches the query video, i.e., one that generates a frame similarity graph (*line 3*) which has a MCM that corresponds to a hit occurrence according to the hit function $\mathrm{H}(\overline{M^{\delta}_{k,\lambda}})$ (*lines 4-6*). If a hit is found, its location, size and type are saved (*lines 7-11*).

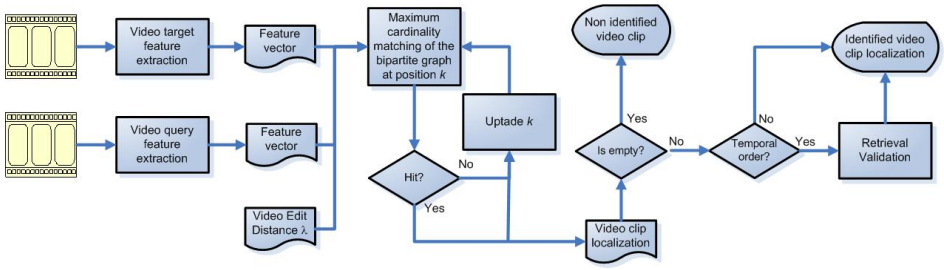


Figure 3: Workflow for subsequence identification.

It also important to describe the *shift strategy* adopted (at *line 12*, *line 16* and *line 18* of algorithm described in Fig. 4). After locating a hit occurrence, the procedure ensures a jump to the location after the last frame of the target video clip that belongs to the MCM (*line 12*) since one should not expect to find the query video inside itself. In this case, minimum shift value is M because it corresponds to the minimum value of maximum MCM distance for a **T1** or a **T2 hit**. Moreover, $t_l+1 = D(\overline{\mathrm{M}^{\delta}_{k,\lambda}})+t_f$ and t_f is equal at least to the previous value of k. Therefore, *line 12* could also be written as $k = k + D(\overline{\mathrm{M}^{\delta}_{k,\lambda}})$. This not only contributes to accelerate the search but it also helps reducing the number of false positives.

In case of a mismatch, shift value depends on MCM size. If MCM size is equal to zero, i.e., there is no match between query and target frames, so the a maximum shift value ($M+\lambda$) is employed (*line 16*). But, if MCM size is greater than zero, the shift value is set to the difference between $M-\lambda$ and the size of the MCM, i.e., the number of unmatched frames necessary to detect a hit occurrence (*line 18*). In order to ensure positive shift value, the value of k is set to t_f at least. In spite of being a conservative approach, this setting allows our search procedure to perform better than the naive (brute force) algorithm and it could result in a great improvement depending on query content and size, e.g., the search would be faster for videos that are more dissimilar and/or for lower values of edit distance (λ) – see experiment results for more about that. It is also important that the search procedure does not miss a *hit* position.

Table 3 presents an example of subsequence identification procedure for a maximum edit distance $\lambda = 1$ ($\equiv 30\%$). In this example, three hit occurrences were found with an average shift value of 2 ($\equiv 66\%$).

Generation of frame similarity graph (*line 3*) and calculation of the MCM (*line 4*) are the most time consuming steps of algorithm in Fig. 4. Thus, our search procedure has a time complexity of $O(NM^2)$ since it is dominated by total time spent on the graph generation step. A detail analysis is omitted due to length limitations for the manuscript.

3.2 Retrieval validation

After query location candidates are selected, they may be validated to ensure other assumptions, like temporal ordering. To verify this assumption, we can use the dynamic programming (DP), as proposed by [18]. We define a temporal order similarity as follows.

Definition 12 (Temporal order similarity – TS^{δ}_{k}) *Let i be i-th frame of the query video V^{Q}_{M},and j be j-th frame of a $\overline{M}$-sized video clip $\mathrm{C}^{T}_{k,\overline{M}}$ of the target video, in which $\overline{M} = D(\overline{\mathrm{M}^{\delta}_{k,\lambda}})$. Temporal order similarity $\mathrm{TS}^{\delta}_{k}(\mathrm{V}^{Q}_{M}, \mathrm{C}^{T}_{k,\overline{M}})$ between the query video and a $\overline{M}$-sized target video clip is equal to $\mathrm{T}^{\delta}(M, \overline{M}+k)$ which is calculated using DP and a specified frame similarity threshold δ as follows*

$$\mathrm{T}^{\delta}(i,j) = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0; \\ \mathrm{T}^{\delta}(i-1, j-1) + 1 & \text{if } \mathrm{FS}(i,j,\delta) = 1; \\ \max\{\mathrm{T}^{\delta}(i, j-1), & \\ \qquad \mathrm{T}^{\delta}(i-1, j)\} & \text{otherwise.} \end{cases} \quad (8)$$

Using the temporal order similarity TS^{δ}_{k}, we can validate location candidates by ensuring that its value is greater than a threshold, i.e., $\mathrm{TS}^{\delta}_{k}(\mathrm{V}^{Q}_{M}, \mathrm{C}^{T}_{k,\overline{M}}) \geq \Delta$. Δ represents the minimum number of similar frames in correct temporal order needed to accept a location candidate. No temporal order changes are allowed, if $\Delta = |\overline{\mathrm{M}^{\delta}_{k,\lambda}}|$ (i.e., the MCM size).

Require: Videos (V^{T}_{N},V^{Q}_{M}), threshold (δ), distance (λ)
{pos = hit locations at the target}
{size = hit occurrence size}
{type = hit types at the target}
1: count = 0; k = 0;
2: **while** (k < $N - M$) **do**
3: "Construct $\mathrm{G}^{\delta}_{k,\lambda}$";
4: "Calculate $\overline{\mathrm{M}^{\delta}_{k,\lambda}}$ and $\mathcal{F}$ for $\mathrm{G}^{\delta}_{k,\lambda}$";
5: "Calculate t_f and t_l for $\mathcal{F}$";
6: **if** $\mathrm{H}(\overline{\mathrm{M}^{\delta}_{k,\lambda}}) \neq 0$ **then**
7: "Query video was found at location k"
8: pos[count] = t_f;
9: size[count] = $D(\overline{\mathrm{M}^{\delta}_{k,\lambda}})$;
10: type[count] = $\mathrm{H}(\overline{\mathrm{M}^{\delta}_{k,\lambda}})$;
11: count = count + 1;
12: k = $t_l + 1$;
13: **else**
14: "Query video was not found at location k"
15: **if** $|\overline{\mathrm{M}^{\delta}_{k,\lambda}}| = 0$ **then**
16: k = k + $(M + \lambda)$;
17: **else**
18: k = max{k + $M - (|\overline{\mathrm{M}^{\delta}_{k,\lambda}}| + \lambda), t_f$};
19: **end if**
20: **end if**
21: **end while**
22: **return** pos, size, type

Figure 4: Identification procedure.

Table 3: Example of subsequence identification procedure.

Video Information																	$\lvert M^{\delta}_{k,\lambda} \rvert$	$D(.)$	Shift	Iteration	Hit
Target video	1	5	6	2	4	2	1	3	5	1	2	3	7	6	1	4	2	-	-	-	-	
$\lambda = 1$	1	2	3	-														2	4	1	1	no
		1	2	3	-													1	1	1	2	no
			1	2	3	-												1	1	1	3	no
				1	2	3	-											2	4	1	4	no
					1	2	3	-										3	3	4	5	ExHit
									1	2	3	-						3	3	4	6	ExHit
													1	2	3	-		1	1	2	7	no
															1	2	3	2	3	-	8	RepHit
Average																		1.87	-	2.00	-	

4. EXPERIMENTS

In this section, we present experiments with a video corpora that consist of TV broadcast, recorded directly and continuously from a brazilian cable TV channel, and Internet retrieved video. Table 4 shows some information about the dataset (including video length and the number of queries). The experiments searched for 86 occurrences of video clips in our dataset (54 for TV broadcast and 32 for Internet retrieved video).

Table 4: Video corpora

Video dataset	Time length	Video queries	Frame rate
TV Broadcast 1	1h 00m 04s	8	30 fps
TV Broadcast 2	35m 02s	2	30 fps
TV Broadcast 3	31m 50s	3	30 fps
TV Broadcast 4	33m 13s	5	30 fps
TV Broadcast 5	30m 27s	8	30 fps
Internet Retrieved Video	19m 52s	17	25 fps
Total	3h 30m 29s	43	-

In order to evaluate the results, it is necessary to define some measures. We denote by $\#Occurrences$ the number of query video occurrences, by $\#Video\ clip\ identified$ the number of query video occurrences that are properly identified and by $\#Falses$ the number of video occurrences that do not represent a correct identification. So, we consider the following quality measures.

Definition 13 (Recall and precision rates) *The recall rate represents the ratio of correct and the precision value relates correct to false detections. They are given by*

$$\mathcal{R} = \frac{\#Video\ clip\ identified}{\#Occurrences};\ (recall) \quad (9)$$

$$\mathcal{P} = \frac{\#Video\ clip\ identified}{\#Falses + \#Video\ clip\ identified}.\ (precision) \quad (10)$$

Definition 14 (F measure) *The F measure is a weight harmonic mean of recall and rate, and it is given by*

$$F = \frac{2 \times \mathcal{P} \times \mathcal{R}}{\mathcal{P} + \mathcal{R}}. \quad (11)$$

Table 5 shows the quality measures related to precision and recall rates, together with F measures for different values of edit distance (λ) and threshold (δ). Fig. 5 illustrates the precision-recall curves. Best results are associated with $\delta = 20\%$ and our method achieves 93% recall with 93% precision (see Table 5 for $\delta = 20\%$ and $\lambda = 30\%$), which is similar to (and even better than) the one proposed by [20] with low computational cost without preprocessing of target video. One of the features of the algorithm that contributes to those results is the size of the MCM. Using the size of the MCM prevents the algorithm from finding versions of query video that have additional frames/shots, or target video clips with

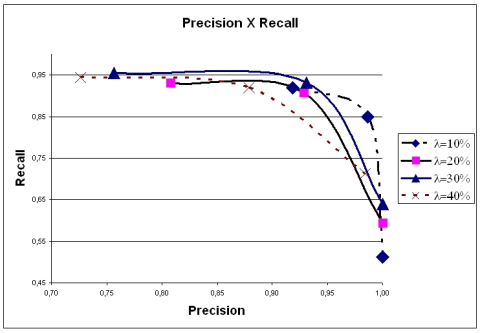


Figure 5: Precision-recall curves.

suppressed parts (frames/shots) of the query video, when we search for exact video, i.e., $\lambda = 0$. A relaxation of the edit distance may imply in finding edited video clips, but it may also contribute in rising the number of false positives. Another parameter that is related to high precision detection is the threshold (δ), which was kept very low.

Table 5: Precision and recall rates

Video edit distance (λ)	Threshold value (δ)								
	10%			20%			30%		
	$\mathcal{R}$	$\mathcal{P}$	F	$\mathcal{R}$	$\mathcal{P}$	F	$\mathcal{R}$	$\mathcal{P}$	F
10%	51%	100%	68%	85%	99%	91%	92%	92%	92%
20%	59%	100%	74%	91%	93%	92%	93%	81%	86%
30%	64%	100%	78%	93%	93%	93%	95%	76%	84%
40%	71%	98%	83%	92%	88%	90%	94%	73%	82%

The recall and precision rates are directly influenced by video edit distance. As showed in Table 5, recall rate increases together with high distance values. This should be expected since a relaxation of the edit distance may imply in finding a great number of query video occurrences (with or without editions). Conversely, precision rate decreases for high edit distance values since more false positives may be identified. With respect to the shift value (see Table 6), for high edit distance values, the shift value decreases since we need less frames to find a hit occurrence. The recall and precision are also directly influenced by threshold value. As showed in Table 5, recall rate decreases as threshold value becomes lower. That should be expected since we impose more restrictions on the similarity between frames. Conversely, precision rate increases for low threshold values since less false positives are identified. With respect to the shift value (see Table 6), for low threshold values, the shift value increases since we need more frames to find a hit occurrence.

The algorithm performance is related to the shift applied during identification procedure. Table 6 shows the average shift values for different parameter settings. A number of 100% means that the shift value is equal to the query video length. It can be seen that, at lower values of δ the average

Table 6: Average shift value percentage.

Video edit distance (λ)	Threshold value (δ)		
	10%	**20%**	**30%**
10%	96%	83%	58%
20%	96%	79%	49%
30%	102%	71%	36%
40%	106%	71%	29%

shift value is higher for the same edit distance. This effect is expected since a higher value of δ increases the number of (mis)matched frames. Another observation is related to the shift and the edit distance. The shift is higher for lower edit distance. This effect is also expected due to conservative approach since to find a hit, we need only to shift enough frames to find it.

5. CONCLUSIONS

In this work, we present an approach which is able to cope with a general video clip localization problem, allowing insertion, removal and replacement operations on the video clip. Our method not only solves the approximate subsequence localization problem but also gives a precise description of the operation set that are necessary to transform the query video into the target content. In order to do that, it uses a bipartite graph matching to identify query location candidates. Main contributions of our work is the application of a simple and efficient distance to solve subsequence identification problem along with the definition of a hit function that identifies precisely which operations were used in query transformation.

According to experimental results, our method performance (93% recall with 93% precision) is similar to (and even better than) the one proposed by [20] with low computational cost and without preprocessing of target video. However, subsequence identification results can be highly dependent on the testing material, which is usually scarce and not especially representative. So, as a future work, we plan to apply our approach to a large and representative video database.

6. ACKNOWLEDGMENTS

The authors are grateful to PUC Minas (Pontificia Universidade Católica de Minas Gerais), CT-Info/MCT/CNPq (Project 551005/2007-6) and FAPEMIG (Project CEX PPM 126/08) for the financial support of this work.

7. REFERENCES

[1] D. A. Adjeroh, M.-C. Lee, and I. King. A distance measure for video sequences. *Computer Vision and Image Understanding*, 75(1-2):25–45, 1999.

[2] A. D. Bimbo. *Visual information retrieval*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1999.

[3] L. Chen and T.-S. Chua. A match and tiling approach to content-based video retrieval. In *ICME*. IEEE Computer Society, 2001.

[4] T. Deselaers, D. Keysers, and H. Ney. Features for image retrieval – a quantitative comparison. In *DAGM 2004, Pattern Recognition, 26th DAGM Symposium, Lecture Notes in Computer Science*, pages 228–236, Tübingen, Germany, September 2004.

[5] N. Diakopoulos and S. Volmer. Temporally tolerant video matching. In *Proc. of the ACM SIGIR Workshop on Multimedia Information Retrieval*, Toronto, Canada, August 2003.

[6] J. Gauch and A. Shivadas. Identification of new commercials using repeated video sequence detection. In *International Conference on Image Processing*, pages III: 1252–1255, 2005.

[7] J. M. Gauch and A. Shivadas. Finding and identifying unknown commercials using repeated video sequence detection. *Computer Vision and Image Understanding: CVIU*, 103(-):80–88, / 2006.

[8] S. J. F. Guimarães, R. Kelly, and A. Torres. Counting of video clip repetitions using a modified bmh algorithm: Preliminary results. In *Proc. of the IEEE ICME*, pages 1065–1068, Toronto, Canada, July 2006.

[9] R. N. Horspool. Practical fast searching in strings. *Softw., Pract. Exper.*, 10(6):501–506, 1980.

[10] A. K. Jain, A. Vailaya, and W. Xiong. Query by video clip. *Multimedia Syst.*, 7(5):369–384, 1999.

[11] A. Joly, C. Frelicot, and O. Buisson. Content-based video copy detection in large databases: A local fingerprints statistical similarity search approach. In *International Conference on Image Processing*, pages I: 505–508, 2005.

[12] Y. Kim and T. Chua. Retrieval of news video using video sequence matching. In *MMM*, pages 68–75, 2005.

[13] R. Lienhart, W. Effelsberg, and R. Jain. Visualgrep: A systematic method to compare and retrieve video sequences. *Multimedia Tools Appl.*, 10(1):47–72, 1999.

[14] X. Naturel and P. Gros. A fast shot matching strategy for detecting duplicate sequences in a television stream. In *Proceedings of the 2nd ACM SIGMOD International Workshop on Computer Vision meets DataBases*, 2005.

[15] G. Navarro. A guided tour to approximate string matching. *ACM Comput. Surv.*, 33(1):31–88, 2001.

[16] Z. K. G. Patrocínio Jr., S. J. F. Guimarães, and H. B. de Paula. Bipartite graph matching for video clip localization. In *SIBGRAPI*, pages 129–138, 2007.

[17] J. S. Pedro, N. Denis, and S. Domínguez. Video retrieval using an edl-based timeline. In J. S. Marques, N. P. de la Blanca, and P. Pina, editors, *IbPRIA (1)*, volume 3522 of *Lecture Notes in Computer Science*, pages 401–408. Springer, 2005.

[18] Y. Peng and C.-W. Ngo. Clip-based similarity measure for query-dependent clip retrieval and video summarization. *IEEE Trans. Circuits Syst. Video Techn.*, 16(5):612–627, 2006.

[19] Y. Rubner, J. Puzicha, C. Tomasi, and J. M. Buhmann. Empirical evaluation of dissimilarity measures for color and texture. *Computer Vision and Image Understanding*, 84(1):25–43, October 2001.

[20] H. T. Shen, J. Shao, Z. Huang, and X. Zhou. Effective and efficient query processing for video subsequence identification. *IEEE Trans. Knowl. Data Eng.*, 21(3):321–334, 2009.

[21] Y.-P. Tan, S. R. Kulkarni, and P. J. Ramadge. A framework for measuring video similarity and its application to video query by example. In *ICIP (2)*, pages 106–110, 1999.