



(51) International Patent Classification:
H04N 5/93 (2006.01)

(21) International Application Number:
PCT/AU2010/000020

(22) International Filing Date:
12 January 2010 (12.01.2010)

(25) Filing Language: English

(26) Publication Language: English

(30) Priority Data:
2009900085 12 January 2009 (12.01.2009) AU

(71) Applicant (for all designated States except US): **THE UNIVERSITY OF QUEENSLAND** [AU/AU]; Sir Fred Schonell Drive, St Lucia, Brisbane, QLD 4067 (AU).

(72) Inventors; and

(75) Inventors/Applicants (for US only): **SHEN, Heng, Tao** [AU/AU]; Unit 1, 1 Brasted Steet, Taringa, QLD 4068 (AU). **HUANG, Zi** [AU/AU]; Unit 1, 1 Brasted Steet, Taringa, QLD 4068 (AU). **ZHOU, Ziaofang** [AU/AU]; Unit 1, 1 Brasted Steet, Taringa, QLD 4068 (AU).

(74) Agent: **MICHAEL BUCK IP**; PO Box 4361, St Lucia, QLD 4067 (AU).

(81) Designated States (unless otherwise indicated, for every kind of national protection available): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IS, JP, KE, KG, KM, KN, KP, KR, KZ, LA, LC, LK, LR, LS, LT, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PE, PG, PH, PL, PT, RO, RS, RU, SC, SD, SE, SG, SK, SL, SM, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, ZA, ZM, ZW.

(84) Designated States (unless otherwise indicated, for every kind of regional protection available): ARIPO (BW, GH, GM, KE, LS, MW, MZ, NA, SD, SL, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

Published:

— with international search report (Art. 21(3))

(54) Title: A SYSTEM FOR REAL TIME NEAR-DUPLICATE VIDEO DETECTION

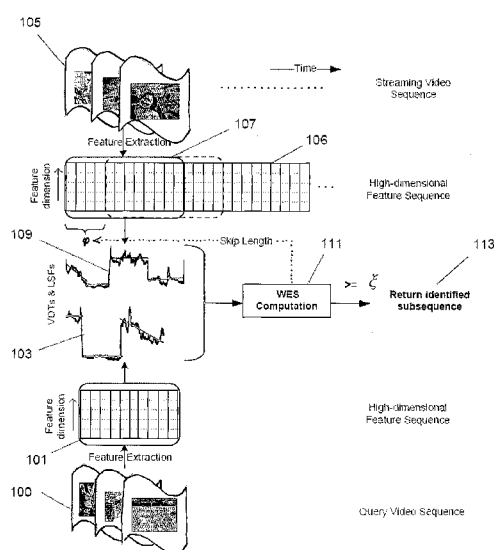


Fig. 9

(57) Abstract: In one embodiment a method of testing for a near-duplicate relationship between a test video clip and a query video clip includes determining linear smoothing functions (LSFs) corresponding to segments of the test video clip from Video Distance Trajectories of the segments. The LSFs of the test video clip are then processed with the LSFs of the query video clip to generate a similarity measure. If the similarity measure is greater than or equal to a predetermined similarity threshold value ϵ , then the segments are deemed to be matched. Sequences of LSFs for query and test video clip sequences, and their similarity measures, are processed to recursively generate a Weighted Edit Similarity (WES) measure. A near duplicate relationship between the test and query clips is deemed to be present upon the WES exceeding a predetermined threshold value ζ .

A SYSTEM FOR REAL TIME NEAR-DUPLICATE VIDEO DETECTION

TECHNICAL FIELD

- 5 The present invention relates to techniques for detecting near-duplicate video sequences in real time.

BACKGROUND

- 10 The discussion of prior art that follows does not constitute a representation or admission that any such prior art comprises common general knowledge.

- With recent advances of video processing technologies in both hardware (e.g., wide availability of webcam) and software (e.g., video editing software), the amount of video data has grown rapidly in many fields, such as broadcasting, advertising, filming, personal video archives, and scientific video repositories. In addition, as massive Internet audience are emerging rapidly and technologies become more video friendly (e.g., increasing bandwidth and near-ubiquitous Internet access), video publishing and sharing in various forms, such as social networking websites, blogs, IPTV, mobile TV, and so on, have become popular.
- 15
- 20 Consequently, online video content is growing aggressively. According to comScore [1], a leader in measuring the digital world, nearly 134 million Americans viewed more than 9 billion online videos in July 2007 alone. The average online video viewer consumed 68 videos during the month and the number is expected to keep growing. Clearly, the mainstream media is moving to the Web, which is forging many novel applications that produce and consume videos in creative ways.
- 25

- The open nature of online video publishing and sharing as well as its huge popularity give rise to the existence of a sizeable percentage of near-duplicate videos which have tolerable content variations introduced during video acquisition, transformation, and edition [20], [17]. Such a phenomenon imposes heavy demands on online near-duplicate detection for many novel applications, such as online copyright enforcement, broadcast monitoring/filtering, Web search
- 30

results cleansing, etc. For example, online near-duplicate detection can raise an alert to possible copyright infringement immediately for intellectual property protection while the video is streaming. In the broadcast domain, advertising clients can monitor TV channels or Web streaming videos to check whether their commercials are actually broadcasted as contracted in the right time slots at the right frequencies. Offending content can also be filtered based on near-duplicate detection.

Recently, several methods have been proposed to retrieve video copies from video databases, where a copy is derived from the origin by standard transformations on the whole video sequence [8], [14], [27], [6]. Different from a copy, a near-duplicate allows for a greater diversity of variations introduced during capturing time (view point and setting, background, foreground, etc) and editing operations (frame insertion, deletion, swap and modification), in addition to standard video transformations (format, resize, shift, crop, etc) [20], [17]. For example, commercial clips of the same product in the same company with different shown prices or contact numbers broadcasted in different countries are regarded as near-duplicates rather than copies. Existing methods fall short as they either cannot identify near-duplicates with moderate variations, or cannot achieve realtime detection due to high time complexity and lack of advanced query processing techniques. Furthermore, online detection over continuous video streams has not been addressed. Preliminary approaches dealing with sophisticated near-duplicates study issues to do with retrieval rather than online detection [20], [17].

Due to the high complexity of video features, keyframe representation is widely used. Paper [2] identifies keyframes by selecting the keyfeature vectors to minimize the Hausdorff distance between them and the original sequence.

Paper [26] proposes a hierarchical video summarization strategy that explores video content structure to select keyframes. Besides keyframe representation, [5] introduces a randomized summarization method to approximate a video into a small set of sampled frames called the Video Signature (ViSig), where each

sampled frame is associated with a Voronoi cell. Video similarity is estimated by the volume of an intersected Voronoi cell. However, the selection of sampled frames may sample non-similar frames from two almost-identical videos. Paper [19] summarizes a video into a sets of clusters each of which is represented by a Video Triplet (ViTri). ViTri models a cluster as a tightly bounded hypersphere described by its position, radius, and density. The ViTri similarity is measured by the volume of intersection between two hyperspheres multiplying the minimal density. When the temporal order is considered, distance functions widely used in time series can be extended.

Mean distance [15] normalizes the sum of pairwise frame distances by sequence length without allowing frame alignment or gap. Dynamic Time Warping (DTW) distance [7], [11], [6] is used to address frame alignment by repeating some frames as many times as necessary without considering noise or gap. Longest Common Subsequence (LCSS) [21] is proposed to address possible noises, but ignores the effect of video lengths. Edit Distance and its variants such as Edit Distance with Real Sequence (EDR) [3] can handle temporal order, frame alignment and gap by matching two sequences with the smallest number of added, deleted or substituted elements. However, all these measures need to compare most, if not all, high-dimensional frames pairwise.

In content-based video detection, efficiency becomes more pressing. Recently, several methods particularly designed for copy detection have been proposed. In [8], [13], local interest points identified from keyframes are used for detection. Each keyframe typically has hundreds of interest points, each of which is represented by a high-dimensional feature vector. However, matching a large number of local interest points between two keyframes is computationally expensive. Pairwise comparisons of high-dimensional vectors are often involved. Recently, [27] has introduced several compact video signatures which are a sequence of numbers, thus string matching technique could be used to rank the results.

The shot-length method summarizes a video clip into a sequence of numbers each of which represents a shot length. This method is efficient but loses the content information completely. Blurry shot boundaries or a very limited number of shots may lead the accuracy to deteriorate greatly.

5

The Color-shift method uses color distributions to produce a signature that represents the inter-frame change in color in the video over time.

Centroid-based signature represents the inter-frame change in spatial movement of the lightest and darkest pixels in each frame over time. Color-shift and centroid-based signatures can also be combined together. All the changes are formalized into numbers so that each video is summarized into a sequence of numbers. However, both color-shift and centroid-based methods only indicate the consecutive inter-frame distance within each individual sequence without carrying content information. Furthermore, their time complexity remains quadratic to video length. More recently, [24] uses a min-hash method to construct sketches for video streams to achieve continuous detection. However, it is based on the Discrete Cosine features extracted from keyframes which may not reflect the visual content as sufficiently as popular features (e.g., color) do. Meanwhile, transitive information between two keyframe is also lost.

In [20] and [23], the concept of near-duplicate video is raised. Particularly, [20] proposes a statistical method for short near-duplicate clip retrieval which captures the clip's content and its changing trends in a single representation. Paper [23] describes the use of local interest points for near-duplicate detection to eliminate similar results from Web search. Both papers focus on short video clips.

It is an object of the invention to provide a near duplicate video clip detection method and apparatus which is an improvement, or at least a useful alternative, to those hitherto known in the prior art.

SUMMARY OF THE INVENTION

According to a first aspect of the present invention there is provided a method of operating an electronic apparatus to indicate a near-duplicate relationship between a test video clip and a query video clip, the method including the steps of:

determining test smoothing functions corresponding to segments of the test video clip;

processing the test smoothing functions and query smoothing functions for the query video clip to determine matched segments of the test video clip and the query video clip;

flagging the near-duplicate relationship based on the determination of the matched segments.

Preferably the step of determining test smoothing functions includes processing Video Distance Trajectories (VDTs) of the segments to determine parameters representing linear smoothing functions (LSFs).

The step of processing the test smoothing functions and query smoothing functions to determine matched segments may include calculating a segment similarity measure.

The segment similarity measure is preferably calculated with operations corresponding to;

shifting

rotating; and

scaling;

one of said linear smoothing functions to map onto to the other.

Preferably the step of processing the test smoothing functions and query smoothing functions to determine matched segments of the test video clip and the query video clip includes comparing a similarity measure to a predetermined segment matching threshold value.

In a preferred embodiment the step of flagging the near duplicate relationship includes computing a Weighted Edit Similarity (WES) measure.

5 The step of calculating the WES may include computing the similarity between a query sequence Q of the query video clip and a comparison window sequence V of the test video clip as the maximal weight of matched LSFs in Q .

10 Preferably the step of flagging the near-duplicate relationship based on the determination of the matched segments includes comparing the WES to a predefined sequence matching threshold value.

15 The method may include shifting the comparison window of the test video clip relative to the query video clip by skipping a number of segments calculated with reference to the WES between the query sequence and the window sequence.

In one embodiment the method includes shifting a comparison window of the test video clip relative to the query video clip by skipping a number of segments of the comparison window preceding a leading segment of said window.

20 Preferably the method also includes processing the test video clip to detect the clip segments, although this could have been done previously in a separate operation.

25 The step of processing the test video clip to detect segments of said clip preferably includes producing color vector signals representing color vectors corresponding to frames of said clip.

30 The method may include processing the color vector signals to produce a color distance signal indicating a color distance between proximal, or consecutive, frames of said clip.

Preferably the method includes generating a cut signal to indicate a segment cut upon the color distance complying with a predetermined condition.

The method may include pre-processing the query video clip to produce the query smoothing functions.

- 5 In one embodiment the query video clip is pre-processed to produce the query smoothing functions at a remote site.

The method may include distributing software to the remote site for producing the query smoothing functions from the query video clip.

10

According to a further aspect of the present invention there is provided a computer software product stored on a program storage device, readable by machine, tangibly embodying a program of instructions executable by the machine to cause the machine to perform the method of claim 1 to indicate the
15 near-duplicate relationship between the test video clip and the query video clip.

20

According to another aspect of the invention there is provided a computational device including at least one electronic processor responsive to an electronic video signal and in electrical communication with a an electronic memory for detecting near-duplicate videos, the electronic processor being arranged to execute instructions stored in the electronic memory, said instructions including:

instructions to process the video signal to flag the near duplicate relationship in accordance with the previously described method.

25

In a further aspect of the invention there is provided a near-duplicate video detection apparatus arranged to determine if a test video clip is a near duplicate of a query video clip and to generate a near-duplicate alert, the video detection apparatus comprising:

30

a cut detector to produce cut signals indicating segments of the test video clip;

a feature extractor unit arranged to produce a plurality of feature signals for each frame of the test video, said feature signals corresponding to feature vectors for frames of the test video;

a video distance trajectory processor arranged to generate distance signals corresponding to each plurality of feature signals;

a linear smoothing processor responsive to the distance signals and to the cut signals and arranged to process the distance signals to calculate smoothed test video line signals for each of the segments;

a similarity calculator arranged to process the smoothed test video line signals and smoothed query video line signals to produce segment similarity signals

a weighted edit similarity processor arranged to process the segment similarity signals and to produce a sequence similarity signal indicating similarity between the query video clip and the test video clip; and

an alert module responsive to the similarity signals to produce the near-duplicate alert upon the similarity signals indicating a similarity measure closer than a predefined threshold.

Preferably the apparatus includes a color vector generator arranged to produce color vector signals representing color vectors corresponding to frames of the test video.

The apparatus may include an inter-frame distance calculator responsive to the color vector generator and arranged to produce a color distance signal indicating a color distance between proximal, or consecutive, frames of the test video clip.

Preferably the cut detector includes a threshold comparator responsive to the inter-frame distance calculator and arranged to generate a cut signal upon the color distance signal complying with a predetermined condition.

The predetermined condition is that the color distance signal exceeds a predetermined threshold value.

According to a further embodiment of the present invention there is provided a networked near duplicate video clip detection method comprising:

making video clip summarization software available to a number of subscribers to the system for said subscribers to produce video clip query summarizations;

receiving subscriber video clip query summarizations from the subscribers
5 via a computer network;

performing near duplicate video clip detection for each subscriber video clip query summarization; and

alerting subscribers upon detecting a near duplicate video clip in respect of their video clip query summarizations.

10 Preferably the step of performing near duplicate video clip detection for each subscriber video query summarization comprises a method as previously described.

15 BRIEF DESCRIPTION OF THE DRAWINGS

Preferred features, embodiments and variations of the invention may be discerned from the following Detailed Description which provides sufficient information for those skilled in the art to perform the invention. The Detailed
20 Description is not to be regarded as limiting the scope of the preceding Summary of the Invention in any way. The Detailed Description will make reference to a number of drawings as follows:

Figure 1 depicts two near duplicate video clips, not being copies of each
25 other.

Figure 2 is a graph of video distance trajectories (VDTs) for two near duplicate video clips.

Figure 3 is a graph of two VDTs for video clips to which different transformations have been applied.

30 Figure 4 is a graph of two VDTs corresponding to video clips that have had shots swapped.

Figure 5 is a graph of linear smoothing functions (LSFs) for one of the video clips.

Figure 6 graphically depicts LSF similarity measures.

Figure 7 illustrates an example of a Weighted Edit Similarity computation.

Figure 8 is a block diagram of a computer system according to an embodiment of the present invention for implementing the method illustrated in
5 Figure 8.

Figure 9 illustrates a method of online near duplicate video detection according to an embodiment of the present invention.

Figure 10 illustrates a window skipping method according to a preferred embodiment of the invention.

10 Figure 11 further illustrates a window skipping method according to a preferred embodiment of the invention.

Figure 12 is a first sheet of a block diagram of a near duplicate video detection apparatus according to an embodiment of the present invention.

15 Figure 13 is a second sheet of the block diagram of the near duplicate video detection apparatus.

Figure 14 is a graph showing the relationship between precision and recall for a number of segment representation schemes.

Figure 15 graphically compares the recall time of four different segments for each of four segment representation schemes.

20 Figures 16 is a graph showing the effect of changing the value of the segment matching threshold ε during operation of a method according to the present invention.

Figure 17 is a graph demonstrating the effect of window skipping, according to a preferred embodiment of the invention.

25 Figure 18 is a graph corresponding to Figure 17 and showing the effect on corresponding sequence matching time as precision increases.

Figure 19 is a graph showing the average sequence matching time for each query using a number of different methods including LSF according to an embodiment of the present invention.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

A paradigm for online near-duplicate video detection over video streams may be described as follows. Given a query video clip, its signature is first generated in an offline process. For a video stream being monitored, its signature is continuously generated and temporarily stored in a small buffer for comparison with the query signature. A near-duplicate relationship is detected if the similarity between the query and the subsequence in the buffer is greater than a threshold. Efficiency of detection is critical.

10 DEFINITIONS

A. Near-duplicate vs. A copy of a video clip based on the notion of tolerable transformation [8] can be defined as below:

Definition 1 (Copy): A video V_i is a copy of another video V_j if V_i is generated after some tolerable transformations on V_j which is called the original. Such a definition has been widely accepted [6], [13], [12]. Typical transformations include changing frame format, resizing, shifting, rotating, mirroring, cropping, coloring (contrast, brightness, etc) and various effects (aging, shearing, sharpening, etc). They are generally applied on the whole video.

20

Near-duplicate image retrieval has also been studied recently [17], [9]. A near-duplicate video clip may be defined as follows:

Definition 2 (Near-duplicate): A video V_i is in a near-duplicate relationship with another video V_j , (i.e. V_i is a near duplicate of V_j), if both are almost the same as each other, but appear differently due to various changes introduced during acquisitions, transformations, and editing. The definition of near-duplicate video is not completely unambiguous. The judgement of near-duplicate is, to a degree, subjective. For example, the decision on two videos taken from the same venue but at different times could be ambiguous since the background and foreground scenes could change greatly. Meanwhile, it is also possible that a video becomes more visually similar to a non-duplicate than its near-duplicate after substantial video transformations and editing. However, in many applications,

30

such as TV commercials and news videos, ambiguity can be reduced largely. For example, TV commercials for the same product (or breaking news on the same event) often show strong visual similarity. Figure 1 shows two different versions of a TV commercial, where the acquisition foregrounds are different and contact numbers are also different. They are near-duplicates, but not copies. In practice, there is always a subjective limit to the tolerable difference between two video clips. The content difference of two near-duplicate videos can be caused by various reasons which can be generalized into the following categories:

- 1) Acquisition with different camera setting, view point, light condition, background, foreground, etc;
- 2) Transformation at sequence level with different encoding parameters, such as frame format, frame rate, resize, shift, flip, crop, color (gamma, contrast, brightness, situation, etc), effect (blur, age, sharpen, etc);
- 3) Editing at frame level with insertion, deletion, swap and content modification;
- 4) Mixture of (1), (2) and (3).

Category (1) introduces near-duplicate of different acquisitions/takes. In Category (2), a transformation typically happens at video sequence level, i.e., a change is applied to the whole video. For example, it introduces different encoding schemes which are usually applied at sequence level uniformly. Category (3) usually occurs at frame level, e.g., delete some frames from a TV commercial to fit into required time frame and modify the contact number on related frames for the same TV commercial to be broadcasted in other countries. Category (4) combines all three categories and can populate very large and diverse collections of near-duplicates.

As defined, copies must be transformed from the same origin. Therefore, two videos with different capturing conditions are not copies. For example, breaking news on the same scene and event taken by different broadcasting corporations are not copies, but near-duplicates without much controversy. Furthermore, existing work mainly focuses on transformations at the video level [14], [6], [13],

[12], i.e., category (2). The effect of editing operations (e.g., frame deletion for a shorter version for a TV commercial or news) at frame level has not been tested. As a result, copies may be regarded as a type of near-duplicate. The detection methods described herein are tested on various near-duplicates with much greater diversity of variations.

B. Online Detection vs. Database Retrieval While most existing work addresses the detection problem from the retrieval point of view (i.e., retrieve relevant results from an existing database) [8], [20], [6], [13], [12], online detection has, but is not limited to, the following main distinctive characteristics:

First, online video streams usually come continuously and are not maintained in the database. Only those detected subsequences are maintained for future analysis. However, retrieval is performed on an existing database.

Second, online detection requires video signatures to be generated on-the-fly and compared in realtime over continuous video streams. However, all video signatures in a database can be pre-generated and indexed for retrieval.

Third, detection is a binary problem which generates a "1" or "0" i.e. a "true" or "false". That is, a near-duplicate is identified if the similarity exceeds a predetermined tolerance value. However, retrieval often involves ranking to return the top-K results. Clearly, online detection addresses more on realtime sequence matching, where compact signature, efficient measure and pruning technique play most important roles.

The inventors have conceived a new similarity measure for video segments that integrates independent evidences by Compound Probability, from which a Weighted Edit Similarity (WES) is extended to compute the maximal similarity between two segment sequences as the video similarity.

The remainder of this description of an embodiment of the invention addresses five areas as follows:

1) A compact video signature is described called Video Distance Trajectory (VDT) which is robust to various changes for near-duplicate detection. It represents a high-dimensional video stream as a one-dimensional distance trajectory monitoring the continuous changes of consecutive frames with respect to a reference point.

2) A segment representation model is described called Linear Smoothing Function (LSF). By performing online segmentation on VDT, LSFs can be generated with guarantee of minimal fitting errors. Each LSF is modeled as a vector in the axis system by preserving three video factors including content, content changing direction and length for effective representation. Its similarity measure integrates three corresponding components to match one LSF to the other by shift, rotate and scaling operations. Given the independence of three components, they are combined by Compound Probability. A particular set of LSFs may be considered as a summarization of the corresponding video clip.

3) A sequence similarity measure called Weighted Edit Similarity (WES) is developed by extending Edit distance and taking segment weight into consideration. Based on constant LSF similarity measure, WES avoids high-dimensional comparisons and reduces the time complexity largely.

4) An online near-duplicate detection method is described which embeds effective skipping methods to speed up detection process by avoiding unnecessary segmentations and WES computations.

5) The effectiveness and efficiency of the online near-duplicate video detection system is demonstrated by extensive performance comparisons with existing methods conducted on diverse near-duplicates.

1. VIDEO DISTANCE TRAJECTORY

In this section, Video Distance Trajectory (VDT) is defined and the mapping of a high-dimensional video sequence into a one-dimensional VDT and representation of VDT by Linear Smoothing Function (LSF), will be explained.

5

Since the video sequence is continuously streamed and the memory size is limited, the video has to be processed upon its receipt for online detection. The visual features (and compact signatures) for the video stream are continuously generated. Typically, a small buffer containing the current video subsequence is served as an intermediary for matching. Note that online detection is done on-the-fly. The received video subsequences that are not identified as near-duplicates are simply discarded. In the following discussion, an assumption is made that the memory is large enough to contain buffers and queries.

10

Applying image feature extraction methods transforms an original video sequence into a sequence of high-dimensional feature vectors, where each frame is represented by a high-dimensional feature vector such as color histogram and texture. Given a video defined as a high-dimensional sequence in a buffer of size w , a summarization of the buffered sequence into its VDT, is defined below. Online query processing with continuous buffer management for video streams will be discussed later.

20

Definition 3 (Video Distance Trajectory): Given a video sequence $V = \{f_1, f_2, \dots, f_w\}$ where f_i is a D-dimensional image feature vector and w is the video length, its Video Distance Trajectory (VDT) is defined as: $VDT = \{d(f_1, o), d(f_2, o), \dots, d(f_w, o)\}$, where o is the reference point and d is the applied distance function.

25

VDT transforms a high-dimensional sequence into a one-dimensional sequence which monitors the consecutive changes of frame distances with respect to a reference point. Figure 2 shows the generated VDTs for the two near-duplicates depicted in Figure 1, where x-axis indicates the length of sequence, and y-axis indicates the frame distance (i.e., content difference) to the reference point. It will

30

be noted that, both VDTs are very similar, except for slight horizontal and vertical shifts.

Obviously, transforming a high-dimensional sequence into a one-dimensional VDT constitutes an aggressive reduction of frame content. Different frames may have similar distances to the reference point. Given a query frame, searching based on frames' one-dimensional distance values may introduce many false positives. However, the sequence information is retained in VDT. Although many false positives could be introduced for a single frame, whole sequence matching potentially reduces the false positives of a query sequence significantly. That is, given two irrelevant video sequences, it is possible that some frames in one video sequence match some frames in another video sequence based on their one-dimensional distance values. However, it is unlikely to have two whole sequences matched. Sequence information can be used to largely offset the loss of frame content largely. It is expected that a longer sequence leads to more accurate sequence matching. Comparing to existing color-shift signature, centroid-based signature and their combination [27] that preserve the inter-frame content difference within a single video sequence only, VDT is also able to indicate the inter-frame content difference among different video sequences via frame distances to the reference point. Thus the content difference among different videos can be reflected. The main advantages of VDT include:

- 1) Robust to various acquisitions, transformations and editions. Slightly different acquisitions of near-duplicates preserve the overall VDT appearance. As shown in Figure 2, the VDTs for two near-duplicates with different foregrounds are highly consistent. Clearly, transformations such as resize, shift, flip and mirror do not change the distance between a frame and the reference point. That is, VDT is invariant to the above transformations. Other transformations within a tolerance on brightness, contrast, crop, etc, do not change the trend of VDT. Figure 3 shows the VDTs for various transformations on version 1 commercial. It may be observed that the signal is slightly shifted up or down, left or right. However, the signal trend is preserved in VDT. Video can also be edited by

modifying, deleting/inserting, or swapping partial frames. Figure 4 shows the VDT for swapping the first few shots to the end of the sequence, where individual shots can be preserved.

5 2) Efficient for online detection. It maps a high-dimensional video sequence to a one-dimensional signal which highly reduces the data complexity. Quick online segmentation can be performed on one-dimensional VDT to generate segments that typically represent shots. High level segment representation can be introduced to further reduce the time complexity of
10 sequence matching to achieve online detection.

3) Capable of fast subsequence detection. Obtained segments can be efficiently matched and integrated for subsequence detection of various
15 lengths.

15 One important issue in generating VDT is the selection of reference point O , which affects the distribution of one-dimensional distance values of frames. An optimal reference point distinguishes frames at maximal degree, i.e., inter-frame distances in the original high-dimensional space are maximally preserved. In
20 Paper [19] it was proved that the optimal reference point lies on the direction identified by the first principal component of the data set and out of the data range. For online near-duplicate video detection, it is very interesting to notice the role change between query sequences and the streaming sequences. Multiple query sequences are often used for simultaneous detection and
25 maintained in the memory. The streaming sequence is continuously processed to detect any near-duplicates of query sequences. Query sequences turn to act as a static database and the streaming sequence acts as continuous queries. By choosing an optimal reference point for all frames in query sequences, query VDTs can be best distinguished from each other so that the chance of
30 introducing false positives for a query frame is minimized. Correspondingly, the probability of introducing irrelevant results is also reduced.

As previously mentioned, a video sequence may be abstracted into a VDT, which is a one-dimensional sequence. Due to the high quadratic time complexity of sequence matching (i.e., quadratic to sequence length), many higher level representations have been proposed in time series literature to improve search on long sequences, including globalized models (Fourier Transform, Wavelet Transform, etc) and localized models (APCA [1], PLA [4], etc). Obviously, global methods are not suitable for continuous video streams. For localized models, the general idea is to first perform segmentation on the sequence and represent each segment by a compact representation. The sequence is then approximated as a sequence of small number of segments.

Different from time series, video sequence has its own characteristics. First, the frame content within a single shot, the basic unit in video, often changes gradually. Therefore, line segments in a VDT may often represent shots. Second, the transit from one shot to another often leads to a big change of frame content. Sudden changes in a VDT may often indicate the boundaries of shots. There could be also some fade in/out effect (like . or . shape in a VDT) during shot transit. Therefore, it is natural to model a VDT as a sequence of segments, each of which typically represents a shot. Particularly to video shot factors, according to, each segment is modeled as a Linear Smoothing Function (LSF).

2. Linear Smoothing Function

A smoothing function is a line used to approximate or “summarize” a segment. A smoothing function in the form of a Linear Smoothing Function (LSF) is used to model each segment, which preserves important information of the segment such as video content, content changing direction, and length. More importantly, the similarity of LSF is measured by taking important video shot factors into consideration. Where the smoothing function is associated with a segment of a video clip to be tested to determine if a near duplicate relationship to a query video clip exists, it may be called a “test smoothing function”. Similarly, where the smoothing function is associated with a segment of a video clip that is being used as a query then it may be called a “query smoothing function”.

Definition 4 (Linear Smoothing Function):

Given a segment $S = \{d(f_i, o), d(f_{i+l}, o), \dots, d(f_{i'}, o)\}$ in a VDT where $i' > i$ or $S = \{d_1, d_2, \dots, d_l\}$ in short, its LSF is defined as a triplet of (α, β, l) , where α and β are parameters representing the line, i.e. the LSF, which fits the segment, for example so as to produce a minimal Sum Square Errors (SSE) and $l = i' - i + 1$ indicates the number of frames over which the segment spans.

For an LSF in the axis system, α is the intercept of the y-axis indicating the initial value of the line. It can be regarded as a very coarse one-dimensional abstraction of frame content from the original feature space. β is the slope of the line reflecting the abstracted content changing trend over consecutive frames. Based on LSF, each element in a segment can be reconstructed by the following formula:

15

$$d'_j = \alpha + \beta \times j$$

Sum Square Error (SSE), i.e., $\sum (d'_j - d_j)^2$, is used to indicate the information lost for LSF representation. Meanwhile, the length of a segment is also maintained. To compute α and β for minimal SSE the following line fitting formulae can be used:

20

$$\beta = \frac{\sum_{j=1}^l j \times d_j - l \times \bar{j} \times \bar{d}}{\sum_{j=1}^l j^2 - l \times \bar{j}^2} \text{ and } \alpha = \bar{d} - \beta \times \bar{j}$$

$$\text{Where } \bar{d} = \frac{\sum_{j=1}^l d_j}{l} \text{ and } \bar{j} = \frac{\sum_{j=1}^l j}{l}$$

25

The inventors have comprehended that representing a segment by an LSF which achieves minimal representation error reduces the data complexity greatly and

speeds up the comparison. For long sequence matching, segmentation is preferable due to quadratic time complexity of (dis)similarity measure. A method for performing online segmentations according to an embodiment of the present invention will now be discussed.

5

Given a $VDT = \{d(f_1, o), d(f_2, o), \dots, d(f_w, o)\}$ in the w -sized buffer, online segmentation is required to generate segments which are then represented by compact LSFs for fast detection. Since the number of segments in a VDT is unknown and time is critical for online detection, a fast cut detection algorithm is used to segment the video sequence or "clip" into shots, i.e. inter-cut sequences of the video sequence. Both query videos and the streaming video are processed with the same cut-detection algorithm to generate segments (or shots) represented by LSFs.

10

15 The cut may be detected in a rather simple way by using the 32-dimensional RGB color feature in the uncompressed domain. Of course other fast cut detection methods are also known. For example a 2D Gabor filtering, method for detecting cuts is described in [28].

20 Given a video frame sequence represented by 32-dimensional RGB feature vectors, the consecutive inter-frame Euclidean distance is computed. If an inter-frame distance exceeding a predefined threshold, ζ , is found, a cut is identified. The subsequence between two consecutive cuts forms a segment. Although errors are still made in the cut-detection method, it is fast and reliable since cut
25 detection scans the sequence once only and is insensitive to changes in color, intensity, bit rate, and resolution [27]. To select a proper ζ value so that the obtained segments closely match the real shots in the video, a preliminary experiment was performed on TRECVID 2007 Sound and Vision video data, where the shot information is provided. The results indicated that $\zeta = 0.05$
30 achieves good segmentation performance based on the provided shot information. Therefore, in experiments, a setting of $\zeta = 0.05$ for online VDT segmentation is used.

So finally, after online segmentation, a one-dimensional VDT will be further summarized into a segment sequence, where each segment is represented by an LSF. Figure 5 shows the generated segments and their LSFs for the VDT (version 1 commercial) depicted in Figure 2. A method for computing the similarity between two LSFs, which is further utilized to calculate sequence level similarity according to a preferred embodiment of the invention, is now described.

3. SIMILARITY MEASURES

As discussed previously, a high-dimensional video sequence is first processed to generate its VDT which is further represented by a sequence of LSFs when each LSF comprises a triplet (α, β, λ) of values corresponding to the gradient, y -intercept and number of frames spanned by the segment. LSF comparison typically measures segment level similarity, which is further aggregated for sequence level similarity. A similar LSF with respect to a query LSF flags the potential occurrence of a near-duplicate. A LSF similarity measure will now be explained.

Each LSF can be viewed as a vector in an axis system whose x -axis indicates the segment length and y -axis indicates the distance to the reference point (i.e., content information). In time series, the distance between two lines is often measured by extended Euclidean distances [10], [4]. However, such distance functions are not applicable for measuring video similarity since neither video segment length nor segment changing trend is fully considered. For example, it tends to give large distance to long similar segments and small distance to short dissimilar ones. Segments of very different lengths may also be measured with small distance.

A new LSF similarity function is now introduced which takes video factors into consideration. Each LSF is modelled as a vector starting from y -axis in the axis system. The similarity is derived from the cost to match one LSF from the other. Typically, three operations are needed to match two LSFs:

- Shift: the operation to shift one LSF to have the same starting point (i.e., α) as the other. Since LSFs all start from y-axis of the system, shift operation is vertically done along the y-axis.
- Rotate: the operation to rotate one LSF to have the same slope (i.e., β) as the other.
- Scaling: the operation to stretch the shorter LSF to have the same segment length (i.e., l) as the other.

Correspondingly, the cost of matching two LSFs comes from three individual components:

- $d_{\updownarrow} = |\alpha_i - \alpha_j|$ the vertical distance along y-axis to shift one LSF to match α of the other. This indicates the difference of video content between two LSFs.
- $d_{\Delta} = |\beta_i - \beta_j|$ the angular distance to rotate one LSF to the other. This indicates the difference of video content changing directions between two LSFs.
- $d_{\leftrightarrow} = |l_i - l_j|$: the horizontal distance along x-axis to stretch the shorter LSF to have the same segment length as the other. This indicates the difference of segment lengths between two LSFs.

20

Figure 6 depicts a graphical interpretation for three components. From the understanding of LSF, the vertical distance $d_{\updownarrow}$ approximates the difference of abstracted video content and the angular distance d_{Δ} is designed to reflect the difference caused by changing directions. The horizontal distance $d_{\leftrightarrow}$ indicates the difference of segment lengths.

However, these three components are independent. They are heterogenous and not directly combinable for video similarity computation. Therefore they are normalized into relative values by their ranges. Meanwhile, for easy illustration, distance value can be mapped into similarity value as below:

30

$$sim_{\updownarrow} = 1 - \frac{d_{\updownarrow}}{\alpha_{\max} - \alpha_{\min}}; \quad sim_{\Delta} = 1 - \frac{d_{\Delta}}{\pi}; \quad sim_{\leftrightarrow} = 1 - \frac{d_{\leftrightarrow}}{\max(l_i, l_j)}$$

With these three similarity components in relative values, it is possible to define an LSF similarity measure. Given the fact that they are independent evidences from different sources, linear combination is not suitable. This is because near-duplicates may have slight variations on content and length. Change of content may also lead to minor variation of changing direction. Clearly, if linear combination is used, some video sequences with overlarge distances in one aspect may also be identified as near-duplicates. Instead, Compound Probability in statistics assumes that evidences are independent. A large Compound Probability value requires that all individual similarities must be large. This clearly better matches the definition of near-duplicate. Compound Probability is adopted to integrate them into a segment similarity measure as follows:

$$\text{sim}(LSF_i, LSF_j) = \text{sim}_{\uparrow} \times \text{sim}_{\triangleleft} \times \text{sim}_{\leftrightarrow}$$

Two segments are *matched* if their LSF segment similarity measure is greater than or equal to ε , a predetermined segment matching threshold value. Clearly, LSF similarity measure considers more video shot factors and combines them by Compound Probability to eliminate more false positives in segment matching. Next the computation of VDT similarity, based on LSF similarity measure to approximate video similarity, is discussed.

Many existing copy detection methods assume that two matching sequences have the same lengths [12], [13]. Obviously, this does not allow for any possible temporal tolerance which widely exists in near-duplicates. Given a query sequence Q and a subsequence of the streaming video V in the intermediate buffer (or window) called window sequence, each is abstracted as a sequence of LSFs. A new similarity measure is proposed to determine if V is a near-duplicate of Q . This similarity measure is called Weighted Edit Similarity (WES) and is derived by extending standard Edit distance and taking the weights of matched segments into consideration, where the weight is measured by the segment length over query sequence length.

Definition 5 (Weighted Edit Similarity):

Given a query sequence $Q = \{LSF_1^Q, LSF_2^Q, \dots, LSF_m^Q\}$ and a comparison window sequence $V = \{LSF_1^V, LSF_2^V, \dots, LSF_n^V\}$ where m and n are the number of segments in Q and V respectively, the Weighted Edit Similarity (WES) between Q and V is defined in a recursive manner as below.

$$WES(V, Q) = \max \begin{cases} WES(head(V), Q) + 0, \\ WES(head(Q), V) + 0, \\ WES(head(V), head(Q)) + \lambda \end{cases}$$

$$\lambda = \begin{cases} \frac{\|LSF_i^Q\|}{\|Q\|} & , \text{ if } sim(LSF_i^Q, LSF_j^V) \geq \varepsilon \\ 0, & \text{ otherwise} \end{cases}$$

where $head(V)$ and $head(Q)$ return the subsequence by removing the first LSF in Q (i.e., LSF_1^Q) and V (i.e., LSF_1^V) respectively, $\|Q\|$ is the query length and $\|LSF_i^Q\|$ is the length of the i^{th} segment in the query. The function terminates when one sequence becomes empty (i.e., one sequence has been completely processed). The initial value of $WES(V, Q)$ is 0 since if there is no match, 0 is summed. It should also be noted that $WES(V, Q) = 0$ if the length of V or Q is 0.

20

In the above definition, each LSF acts as a basic unit for comparison. WES computes the similarity between Q and the comparison window sequence V measured by the maximal weight of matched LSFs in Q . Similar to standard Edit distance, WES considers temporal information and alignment. However, WES is differentiated from standard Edit distance that WES computes the real similarity measured as the maximal percentage of matched query length. The time complexity of WES is $O(mn)$. Comparing to the original video length in number of frames, m (or n) is much smaller. Therefore, the time complexity of sequence

25

comparison is reduced from being quadratic to the number of frames to being quadratic to the number of segments. Furthermore, the time complexity of the LSF similarity measure is constant and much smaller than that of high-dimensional frame comparison.

5

Interestingly, WES computation can also identify the subsequence in V which maximally matches Q by the corresponding matching relationship between LSF_i^Q and LSF_j^V . During WES computation, a bipartite graph can be established where the link indicates the matching relationship between LSF_i^Q and LSF_j^V . The subsequence in V from the first matching LSF to the last matching LSF maximally matches Q . To have a clearer picture, Figure 7 shows an example where each character represents an LSF and two LSFs are matched if they have the same character. For simplicity, assume each LSF has equal length of 1 (i.e., equal weight too), the window size $w = 6$ and $\|Q\| = 5$. From WES computation,

10 $WES(V, Q) = 2/5$ and the identified subsequence in the window which maximally matches Q is **BFC**.

15

When $WES(V, Q) \geq \xi$, a predefined sequence matching threshold, the identified subsequence in V which maximally matches Q is then returned as a near-

20 duplicate of Q . The smaller the value of ξ the more closely the two match.

Referring now to Figure 8, there is depicted a computational device in the form of a personal computer system 52. The computer system will preferably equal or exceed the performance of Dell Precision M4300 2Duo CPU (2.39 GHz) 3.5G

25 RAM. Computer system 52 operates as a real-time near duplicate video clip detection system according to a preferred embodiment of the present invention while executing a computer program which will be described shortly. Personal Computer system 52 includes data entry devices in the form of pointing device 60 and keyboard 58 and a data output device in the form of display 56. The data

30 entry and output devices are coupled to a mainboard 68 which interfaces to various modules including at least one central processing unit 70. The mainboard and associated components are housed in case 54.

Central processing unit (CPU) 70 interfaces with storage devices that are readable by machine and which tangibly embody programs of instructions that are executable by the CPU. These storage devices include electronic memories in the form of RAM 62, ROM 64 and secondary storage devices i.e. a magnetic
5 hard disk 66 and optical disk reader 48, via mainboard 68. The personal computer system also includes a network port 50 so that it can exchange data over a network. For example, the personal computer system may receive query video clips over computer networks such as the Internet 75 from remote parties such as subscribers 77a,...77n, search for the identity of one or more near
10 duplicate video clips and return the search results over the Internet. Computer system 52 also includes a video capture card 74, which includes a TV tuner in order that the system can monitor and record cable television, or radio frequency broadcast television via antenna 76. Furthermore, the computer system may be configured to receive LSF data, i.e. query data, for query video clips rather than
15 the video clips themselves, from remote destinations via the network port 50. In that event it is envisaged that a software program, or dedicated hardware be distributed to the subscribers, for pre-processing their query video clips into query electronic data defining straight line segments (e.g. LSFs corresponding to segments of the query video clip) at their remote sites.

20 It will be realized that the video capture card 74 and network port 50 constitute examples of video clip access assemblies for presenting video clip files in electronic format to the remainder of the computer system 52. Other video clip access assemblies are possible, for example a digital video camera.

25 Secondary storage device 66 is a magnetic data storage medium that bears tangible instructions, for execution by central processor 70. These instructions will typically have been installed from an installation disk such as a program storage device in the form of an optical disk 46 although they might also be
30 provided in a memory integrated circuit or via a computer network from a remote server installation. The program storage device tangibly embodies a program of instructions constituting a computer software product 72 executable by processor 70 to cause computer system 52 to operate as a Near Duplicate Video Clip

Detection (NDVCD) system and in particular to implement a NDVCD method that will be described shortly with reference to Figure 9.

It will be realized that program storage devices such as magnetic and optical disks, and computer memory integrated circuits, all of which are readable by machine, tangibly embody a program of instructions executable by the machine, to cause the machine to perform the NDVCD method. For example, an optical disk includes an arrangement of pits representing bits that an optical disk reader can discern to assemble the program from the instructions.

It will be realized by those skilled in the art that the programming of software product 72 is straightforward in light of the method of the present invention, a preferred embodiment of which is described. In the execution of the method various electrical signals representing variables are manipulated. It will be further realized that during operation of computer system 52 to implement the method corresponding registers of CPU 70 will be incremented and data written to and retrieved from secondary storage 66 and RAM 62 by virtue of electrical signals travelling along conductive busses etched on mainboard 68. Consequently, physical effects and transformations occur within computer system 52 as it executes software 72 to implement the methods.

Figure 9 illustrates an online near-duplicate video detection method according to a preferred embodiment of the invention for implementation by a computer system such as that shown in Figure 8. As previously explained, the method is programmed as software product 72 (Figure 8).

Referring again to Figure 9, The query video 100 is preprocessed offline to extract its high-dimensional feature sequence 101, followed by its VDT and LSF generation 103. For the incoming video stream 105, its high-dimensional feature sequence 106 is continuously extracted and applied with a buffer (or window) 107. Within the window, the feature sequence is transformed into a VDT represented by a sequence of LSFs 109. WES between the query sequence and the window sequence (i.e. the streaming subsequence in the window) is

then computed 111. When $WES(V,Q) \geq \xi$, a near-duplicate is detected and the identified subsequence within the window is returned 113. This is preferably done by flagging a near-duplicate alert such as flashing a "near duplicate detected" message on display 56 (Figure 8). The window keeps shifting forward and the detection continues. A window size (i.e. w) is set to be 20% larger than the query length to allow 20% temporal tolerance. Depending on applications, different window sizes may be set.

In the online detection paradigm, the window over the video stream needs to shift forward to include new coming frames. The naïve method is to shift the windows by removing the front LSF (i.e., front segment) every time and refill the window by adding new coming frames to the end. After each shift, the new coming subsequence is mapped into its VDT and the whole VDT in the window is then segmented and represented as a sequence of LSFs. The deficiency of the naive method is mainly caused by two factors. First, the segmentation cost after each window shift is high since the window size could be large. Second, the number of shifts is large, since the window skips only one segment every time.

As previously mentioned a networked near duplicate video clip detection method may be provided that uses a central server computer arrangement such as computer 52 of Figure 8. Video clip summarization software, for generating the LSFs for a video clip, according to an embodiment of the present invention is made available to a number of subscribers 77a,...,77n to the system. These subscribers will typically be owners of copyright material that are able to connected the central server, for example by the Internet. The subscribers 77a,...,77n then produce video clip query summarizations with the video clip summarization software and transmit them back to the central server 52 by a computer network such as the Internet 75. The central server 52 receives the subscriber video clip query summarizations from the subscribers and performs near duplicate video clip detection for each subscriber video clip query summarization. If a near-duplicate is found then the central server alerts the relevant subscriber. As previously mentioned, an advantage of this approach is

that only the compact summarizations are transmitted across the computer network, e.g. the Internet, rather than the video clip.

4. Detection by Skipping

5

To speed up the detection process, Detection by Skipping may be employed in a preferred embodiment of the invention. Detection by Skipping is composed of segment skipping and window skipping to avoid a large portion of redundant computations.

10

1) Segment Skipping: Given a window sequence $V = \{LSF_1^V, LSF_2^V, \dots, LSF_n^V\}$, assume the next window is shifted by k segment, i.e., the first k segments are removed and $\sum_{j=1}^k \|LSF_j^V\|$ new coming frames are inserted to the end of the

window, followed by the VDT generation of new frames. To generate LSFs for the new VDT in the next window, the online segmentation method needs to be applied. Instead of segmenting the whole VDT, segment skipping is based on the following three observations.

15

First, VDTs between two consecutive windows preserve partial continuity since the ending segments in the current window, i.e., $(k+1)^{th}, \dots, n^{th}$ segments, become the beginning segments in the next window.

20

Second, the last segment could be broken since the window provides a hard cut to the video stream.

25

Third, a segment typically represents a shot, and appending more frames to the sequence does not affect the precedent shots, except the last one.

Therefore, it is not necessary to segment the whole VDT in the next window.

30

Instead, segmentation is performed on the partial VDT constructed by the last segment (i.e., LSF_n^V) and the new coming frames. Consequently, the segmentation cost is reduced by skipping the precedent segments (i.e. $LSF_{k+1}^V, \dots$,

LSF_{n-1}^V) preceding the leading LSF_k^V segment. The inventors' preliminary experiments have indicated that the performance difference between using the whole VDT segmentation and segment skipping is negligible.

- 5 2) Window Skipping: Each shift corresponds to a WES computation between the query sequence and the window sequence. Shifting the window by one segment a time is obviously less than optimally efficient. Observing that the WES computation between the query sequence and the window sequence also shows partial continuity, that can be used to advantage by computing an upper bound of
- 10 WES as a function of the skip length to skip all intermediate segments until the upper bound exceeds the WES threshold - ξ .

Lemma 1 (WES Upper Bound):

- 15 Given a query sequence

$Q = \{LSF_1^Q, LSF_2^Q, \dots, LSF_m^Q\}$ and a window sequence

$V = \{LSF_1^V, LSF_2^V, \dots, LSF_n^V\}$ where m and n are the number of segments in Q and V respectively, the upper bound on $WES(Q, V')$ where V' is the window sequence after shifting V by k segments, is:

20

$$WES^u(Q, V') = WES(Q, V) + \frac{\sum_{j=1}^k \|LSF_j^V\| + \|LSF_n^V\|}{\|Q\|}$$

Proof.

- V' comprises two consecutive subsequences V'_1 and V'_2 where V'_1 consists of
- 25 $LSF_{k+1}^V, \dots, LSF_{n-1}^V$ and V'_2 includes LSF_n^V and the new coming frames into a new segment. The number of new coming frames is equal to the total number of frames skipped, i.e. $\sum_{j=1}^k \|LSF_j^V\|$. Since WES computes the maximal similarity between Q and V and V'_1 is a subsequence of V , it follows that:

$$WES^u(Q, V') = WES(Q, V)$$

The maximal WES between Q and V'_2 is achieved when V'_2 is fully matched with Q , i.e.

$$WES^u(Q, V'_2) \leq \frac{\sum_{j=1}^k \|LSF_j^V\| + \|LSF_n^V\|}{\|Q\|}$$

5 Since

$$WES(Q, V') \leq WES(Q, V'_1) + WES(Q, V'_2),$$

the upper bound $WES^u(Q, V')$ holds.

10 If $WES(Q, V) \geq \xi$, the identified subsequence in the window is returned as near duplicate and skipped. Otherwise, based on Lemma 1, it follows that the skip length φ for the window is as follows:

$$\varphi = \lfloor \|Q\| \times (\xi - WES(Q, V)) \rfloor - \|LSF_n^V\| + 1$$

Note that φ indicates the number of frames the window can skip with guarantee that no subsequence whose WES is greater than or equal to ξ will be missed. In order to skip the window by segment it is necessary to map φ into the number of segments k to be skipped. Simply, $k = 1$ when $\|LSF_1^V\| \geq \varphi$. Otherwise keep k increasing until $\sum_{j=1}^k \|LSF_j^V\| > \varphi$ then $k \leftarrow k - 1$. Based on the derived k value, the window can be quickly shifted by skipping k segments. This accelerates the detection process with guarantee.

20

Recall that during WES computation, the matching relationship between the query LSFs and the window LSFs can be established. This information can be used to further skip the window, based on the following Lemma 2.

25 *Lemma 2:*

If $WES(Q, V) \geq \xi$, then at least one segment among the first m' segments in Q is matched, where m' is the smallest integer satisfying:

$$\sum_{j=1}^{m'} \|LSF_j^Q\| > \lceil \|Q\| \times (1 - \xi) \rceil.$$

The value of k is derived, based on Lemma 1, a further check can be performed to determine if LSF_{k+1}^V matches any of the first m' segments in Q . Based on

5 Lemma 2, if LSF_{k+1}^V does not match any of the first m' segments in Q , then LSF_{k+1}^V does not contribute to WES computation and thus can be safely skipped.

The next segment in the window is checked until the segment which matches one of the first m' segments in Q is encountered or the last segment in the windows is reached. All its precedent segments can be skipped with the

10 guarantee that no true positive is missed.

Following on from the example shown in Figure 7 where $\|LSF\| = 1$, window size $w = 6$ and $\|Q\| = 5$. Firstly, set $\xi = 4/5$. Since $WES(Q, V) = 2/5$, based on Lemma 1, it follows that $k = \varphi = \lceil 5 \times (0.8 - 0.4) \rceil - 1 + 1 = 2$. As shown in Figure 10, the

15 window can skip $k = 2$ LSFs. Then based on Lemma 2, it follows that $m' = 2$, which is the smallest integer greater than $\lceil 5 \times (1 - 0.8) \rceil = 1$. The next LSF in the window which matches any one of the first $m' = 2$ LSFs in Q is then simply found. As an example shown in Figure 11, the next LSF in the window which matches any of the first 2 LSFs in Q is **A**, which is also the last LSF in the

20 window. The window can skip 5 segments until the first LSF of Q is matched. Until then, a WES computation is consumed between the query and the window.

Clearly, by window skipping based on Lemma 1 and 2, a portion of WES computations can be saved. In the best case, the window can skip $\|Q\|$ frames

25 every time. Depending on the query and video stream, the portion of saved WES computations may vary.

Referring now to Figures 12 and 13, there is shown a block diagram of a dedicated hardware near duplicate video detector (NDVD) 2 according to a

30 further embodiment of the present invention. The NDVD 2 includes a video stream data bus 4 for conveying electrical signals representing the frames

$F_1, \dots, F_w$ of a portion of video clip 6. The video clip 6 may have been transmitted over the Internet in a compressed format, for example in accordance with a compression protocol such as MP4. However, the portion of video clip shown in the Figure will have been decompressed. The video clip frame sequence is received into a video stream buffer 8 that comprises an integrated circuit digital memory chip. The video stream buffer 8, along with most of the other hardware components that comprise the NDVD 2, are under synchronous control from a central clock (not shown). An output side of the digital memory buffer 8 communicates with both a cut detector, in the form of Cut Detection Module 10 and a Feature Extractor Unit 16. The Cut Detection Module 10 includes a RGB Color Vector Generator 11 that processes each data frame from buffer 8 to load an internal digital memory with signals corresponding to a 32-dimensional RGB color feature vector for the frame. Interframe Distance Calculator 12 is coupled to the RGB Vector Generator 11 and is arranged to process the output from the RGB Vector Generator to produce a color difference signal indicative of the magnitude of the RGB vector difference between consecutive frames. (Alternative embodiments of the invention may produce a difference signal indicative of the magnitude of the RGB vector difference between proximal, rather than consecutive, frames). A threshold comparator 14, coupled to the Interframe Distance Calculator 12, determines if the difference signal is greater than a predetermined threshold value. If the difference signal is indeed greater than the predetermined threshold value then a cut, i.e. a change of scene or shot is deemed to have occurred in the video and a corresponding cut signal is generated so that data identifying the frame, or another corresponding timing parameter, is then stored in the segment memory buffer.

Feature extractor unit 16 processes the output of the video stream memory buffer. The feature extractor is arranged to perform video feature extraction to thereby generate a stream of digital vector data signals that represent feature vectors $f_1, \dots, f_w$. The digital vector data signals are stored in a vector data memory chip 20, i.e. a digital memory buffer. A Video Distance Trajectory (VDT) Processor 22 is coupled to an output side of the the Vector Data Memory Buffer 20. The VDT Processor 22 is processes the signals that represent each feature

vector to produce corresponding distance signals that indicate distances $d(f_1, 0), \dots, d(f_w, 0)$ between the feature vector and a predetermined origin. The distance signals for each frame are stored in a VDT Digital Memory Buffer 24.

- 5 A Linear Smoothing Processor 26 is coupled to an output side of the VDT Digital Memory Buffer 24 and to an output side of the Segment Memory Buffer 18. The Linear Smoothing Processor 26 is arranged to process the distance signals from the VDT Digital Memory Buffer 24 to calculate a number of straight line segments of best fit for each segment as defined by the segment signals from the Segment
- 10 Memory Buffer. As previously described, this will preferably be performed according to a least squares line fitting method.

The Linear Smoothing Processor 26 is arranged to generate test electronic data signals defining each of the straight line segments. These signals represent the

15 previously described (α, β, l) , triplets for each straight line segment, i.e. Linear Smoothing Function (LSF) and are stored in a Line Parameter Buffer 28.

A Query Video Line Parameter Buffer 32 stores query electronic data straight line segment signals representing (α, β, l) , triplets corresponding to a query video.

20 These query video line signals, which represent LSFs, for each segment will typically have been generated, i.e. pre-processed, using the same hardware arrangement and/or software, as previously described.

A Weighted Edit Similarity Processor 30 receives signals from the Line

25 Parameter Buffer 28 and from the Query Video Line Parameter Buffer 32. The Weighted Edit Similarity Processor 30 is arranged to generate similarity electronic data that represents the similarity between the line signals for the query video, stored in buffer 32, and the test video line signals for segments of the incoming video clip, stored in buffer 28. The Weighted Edit Similarity

30 Processor 30 communicates with a similarity calculator 34 during this processing. The similarity calculator 34 is arranged to compare corresponding (α, β, l) triplets of the query video and the incoming video to produce a similarity measure signal

for each segment. The similarity measure signals are processed by the Weighted Edit Similarity Processor 30 to produce the similarity electronic data.

In the event that the Weighted Edit Similarity Processor 30 determines that the similarity between the test electronic data in the line parameter buffer 28 and the query electronic data in the video buffer 32 complies with a predetermined condition, i.e. that it exceeds a predetermined sequence matching threshold then a Near-Duplicate relationship is flagged. The Weighted Edit Similarity Processor 30 produces an alert signal which is passed to an external alert module 36. The external alert module may be a visual or aural indicator to alert an operator to the fact that the query video and the incoming video have been deemed to be near duplicates. If desired, the external alert module may include a computer network port in order that it can send an alert to a remote destination, for example by means of the Internet.

Flagging of the Near-Duplicate status may also take other forms, for example in other embodiments a variable may be flagged in a memory register of the apparatus to indicate that a Near-Duplicate status has been detected. A software routine, executed by a processor in communication with the memory register causes the processor to repeatedly check if the variable has been flagged and if so to take appropriate action, that action might include logging the event or emailing the owner of the query video clip in question, for example.

A Skipping Controller 38 receives signals from the Weighted Edit Similarity Processor 30, Line Parameter Buffer 28 and Query Video Buffer 32. On the basis of those signals the Skipping Controller calculates the number of line segments, represented by the signals stored in buffers 28 and 32 that may be skipped, without introducing near duplicate detection errors, before the Weighted Edit Similarity Processor 30 commences generation of the next similarity measure signal. Skipping Controller 38 also calculates the number of frames of incoming video corresponding to the skipped line segments and transmits control signals to the various registers 8, 18, 20, 22, 24, 28, 32 in order to synchronise corresponding skipping of frame and line segment skipping. The Skipping

Controller ensures that the Weighted Edit Similarity Processor does not have to process signals which are unnecessary to determining whether or not the incoming video is a near duplicate of the query video.

5. Effectiveness & Efficiency – Experimental Results

The Sound and Vision data in TRECVID 2007 was used to simulate a base video stream. It consists of news magazine, science news, news reports, documentaries, educational programming, and archival video. A portion of 10 hours video was used. Twenty video clips were used as queries, which included commercials, movie trailers, news, etc. Their lengths varied from seconds to minutes with an average of about 60 seconds. For each query, 10 different near-duplicates were generated in all categories as previously discussed.

ID	Category	Description
1	1	Different take
2	2	Contrast – 20%
3	2	Brightness – 20%
4	2	Crop by – 20%
5	3	Frame rate – 20%
6	3	Add logo 20% of size
7	4	1 & 2
8	4	2 & 5
9	4	3 & 6
10	4	4 & 6

Table 1

Table I shows the list of near-duplicates. Particularly, the first near-duplicate (i.e., Category (1)) of each query is manually judged without ambiguity from the real-life video collection among a group of 5 users. That is, the Category (1) in Table I is the near-duplicate that naturally occurs in the video collection with different

acquisition conditions, such as examples shown in Figure 1. The remaining near-duplicates are generated by various transformations (i.e., Category (2)), editions (i.e., Category (3)) and mixture of the above (i.e., Category (4)). Clearly, the near-duplicates are produced with a greater diversity than copies [8], [24]. For the generated near-duplicates, the inventors randomly inserted them into the base video to form a longer video stream of about 13 hours, which was monitored continuously for near-duplicate detection for all queries.

Standard precision and recall measures were used for effectiveness evaluation. Precision was defined as the ratio of near-duplicates detected divided by the total number of subsequence returned and recall was defined as the ratio of near-duplicates detected divided by the total number of near-duplicates in the video stream. Figure 14 is a graph showing the relationship between precision and recall values for a number of segment representation schemes including LSF. It will be noted that LSF gave the highest precision values for a given level of recall.

Computational time was measured to evaluate efficiency. The programs were written in Matlab and C++ by running them on Dell Precision M4300 2Duo CPU (2.39GHz) with 3.5G RAM. The visual features extracted from video frames of 192x144 pixels resolution in AVI format are 32-dimensional RGB feature vectors, based on which the VDT is be generated.

The effectiveness of the previously described methods will now be discussed. Firstly, there are two parameters to be tuned:

1) Parameter Tuning: ε suggests the maximal tolerance allowed for two segments, while ζ indicates the minimal percentage of query length to be matched for near-duplicates. One can affect the other. If ε is large, few segments will be matched and ζ needs to be small in order to achieve high recall. On the other hand, if ε is small, many segments can be matched and ζ needs to be large in order to achieve high precision. Figure 15 shows the effect of different ε values on precision and recall, where ζ is adjusted for each value to construct its

precision-recall line. As can be seen, when recall is low, three values achieve similar precisions. However, as recall increases (i.e., ξ decreases to include more results), $\varepsilon = 0.7$ performs best. The precision for $\xi = 0.8$ drops quickly and becomes worst. This is because a large ε value may potentially filter out some near-duplicates. On the other hand, for $\varepsilon = 0.6$, many segments are matched, which may potentially bring in more false near-duplicates in the results. Therefore, $\varepsilon = 0.7$ is set for the experiments. From its precision-recall line, a proper ξ value can be determined based on the requirement.

2) A Comparison Study: As can be seen in the next subsection, performing sequence matching on full video length is too expensive for online detection, even on one-dimensional sequence. The preferred way is partition a sequence into segments each of which is represented by a compact representation. Many methods have been proposed to represent segments. In high-dimensional video sequence, a segment is often represented by a high-dimensional keyframe [2] or simply by the shot length [27]. In one-dimensional sequence, several segment representations have been proposed, such as APCA [1] and PLA [4], where their similarity is measured by extended Euclidean distance. Figure 16 graphically compares the recall time of four different segments C1, C2, C3, C4 for each of five different segment representation schemes LSF, APCA, Keyframe, PLA, and Shot-Length. It will be noted that the recall value for LSF is greatest for each of the different segments. The effectiveness of the LSF model and its measure with the above segment representation methods and their similarity measures are now compared.

The detection process includes the time from two main components: feature extraction and sequence matching. Given the extracted frame features, sequence matching usually further summarizes them into more compact form in order to speed up detection. Note that the present work is not concerned with improving feature extraction time. Rather, the aim is to reduce the sequence matching time, which is mainly determined by the time complexity of sequence similarity measure and query processing techniques. Initially the performance of the query processing technique window skipping, in saving unnecessary WES

computations, is examined. The reported sequence matching time is the average over all queries.

Without window skipping the window has to shift segment by segment and every shift invokes a WES computation. Figures 17 and 18 show the effect of window skipping as precision increases. Note that ξ can be adjusted to get different precisions for the fixed ε . A larger precision (i.e., fewer results) corresponds to a larger ξ . From Figure 17, window skipping is able to prune a large percentage of segments starting from a small precision value. As precision increases, the pruning power of window skipping grows with respect to larger ξ values. Figure 14b shows the corresponding sequence matching time as precision goes up. Precision changes do not affect the sequence matching time for the segment by segment method. For window skipping, as ξ increases, its pruning power becomes stronger (from Lemma 1) and thus sequence matching time drops. When precision=0.8, window skipping improves the segment by segment method by nearly 4 times. This experiment confirms the effectiveness of the window skipping technique.

2) A Comparison Study: In this experiment, the efficiency of the LSF method supported by window skipping with other methods is compared. The time complexity of sequence similarity measure for each representation model is firstly analyzed.

Given a query sequence Q having m segments and a window sequence V having n segments, full length sequence matching between Q and V requires the time complexity of $O(\|Q\| \times \|V\|)$, while segment based sequence matching has the time complexity of $O(mn)$. Consider Q and V are one minute videos with 1500 frames and $m=n=10$. A single full length matching takes $\left(\frac{1500}{10}\right)^2$ times longer than a segment based matching. Clearly, full length matching is impractical for continuous online detection. Measuring sequence similarity based on a small number of segments is more robust.

Among different segment representations, keyframe, PLA and APCA involve Euclidean distance computations and shot-length involves several conditional switches in its categorical scoring function. There is no clear sequence skipping technique associated with the above representations to speed up sequence matching over continuous video streams.

Figure 15 shows the average sequence matching time for each query over the video stream of 13 hours, where LSF* represents LSF with window skipping for precision=0.8. As can be seen, LSF* achieves the best performance with an average sequence matching time of about 4.5 seconds, followed by LSF, shot-length, keyframe, APCA and PLA. Given the sequence matching time, it is possible to derive the maximal number of queries that can be simultaneously monitored over a video stream for online near-duplicate detection as $\frac{T_v - T_{fe}}{T_{sm}}$,

where T_v is the video time, T_{fe} is the feature extraction time, and T_{sm} is the sequence matching time. Feature extraction takes about one hour for 13 hours video.

Therefore, the system can support $\frac{12 \times 3600}{4.5} = 9,600$ queries of 60 seconds for accurate online near-duplicate detection over a continuous video stream, provided that features have been extracted and memory is large enough.

It will be realized that a preferred embodiment of the invention provides an efficient online system which monitors queries over continuous video streams to detect near-duplicate occurrences. To achieve this, a compact one-dimensional video signature VDT is used which is further represented by a sequence of LSFs considering more shot information including content, content changing direction and length for more effective measure. A sequence similarity measure called Weighted Edit Similarity is used to determine similarity by taking the segment weight into consideration. During the detection process, effective skipping methods are also introduced to speed up the process. An extensive performance study conducted on near-duplicates of great diversity shows the effectiveness and efficiency of methods according to embodiments of the present invention.

REFERENCES

The reference to the following prior art is not to be taken as any representation or admission that such art forms part of the common general knowledge. The disclosures of each of the following articles are hereby incorporated by reference in their entireties and for all purposes.

[1] K. Chakrabarti, E. J. Keogh, S. Mehrotra, and M. J. Pazzani. Locally adaptive dimensionality reduction for indexing large time series databases. *TODS*, 27(2):188–228, 2002.

[2] H. Chang, S. Sull, and S. Lee. Efficient video indexing scheme for content-based retrieval. *IEEE Transactions on Circuits and Systems for Video Technology*, 9(8):1269–1279, 1999.

[3] L. Chen, M. T. Ozsu, and V. Oria. Robust and fast similarity search for moving object trajectories. In *SIGMOD*, pages 491–502, 2005.

[4] Q. Chen, L. Chen, X. Lian, Y. Liu, and J. X. Yu. Indexable pla for efficient similarity search. In *VLDB*, pages 435–446, 2007.

[5] S. S. Cheung and A. Zakhori. Efficient video similarity measurement with video signature. *IEEE Transactions on Circuits and Systems for Video Technology*, 13(1):59–74, 2003.

[6] C.-Y. Chiu, C.-H. Li, H.-A. Wang, C.-S. Chen, and L.-F. Chien. A time warping based approach for video copy detection. In *ICPR*, pages 228–231, 2006.

[7] A. Fu, E. J. Keogh, L. Y. H. Lau, and C. A. Ratanamahatana. Scaling and time warping in time series querying. In *VLDB*, pages 649–660, 2005.

[8] A. Joly, O. Buisson, and C. Frilicot. Content-based copy retrieval using distortion-based probabilistic similarity search. *IEEE Transactions on Multimedia*, 9(2):293–306, 2007.

[9] Y. Ke, R. Suthankar, and L. Huston. Efficient near-duplicate detection and sub-image retrieval. In *ACM Multimedia*, pages 869–876, 2004.

[10] E. J. Keogh, S. Chu, D. Hart, and M. J. Pazzani. An online algorithm for segmenting time series. In *ICDM*, pages 289–296, 2001.

[11] E. J. Keogh and C. A. Ratanamahatana. Exact indexing of dynamic time warping. In VLDB, pages 406–417, 2002.

[12] C. Kim and B. Vasudev. Spatiotemporal sequence matching for efficient video copy detection. IEEE Transactions on Circuits and Systems for Video Technology, 15(1):127–132, 2005.

[13] J. Law-To, O. Buisson, V. Gouet-Brunetand, and N. Boujemaa. Robust voting algorithm based on labels of behavior for video copy detection. In ACM Multimedia, pages 835–844, 2006.

[14] J. Law-To, L. Chen, A. Joly, Y. Laptev, O. Buisson, V. Gouet, N. Boujemaa, and F. Stentiford. Video copy detection: a comparative study. In CIVR, pages 371–378, 2007.

[15] S.-L. Lee, S.-J. Chun, D.-H. Kim, J.-H. Lee, and C.-W. Chung. Similarity search for multidimensional data sequences. In ICDE, pages 599–608, 2000. [16] M. Morse and J. Patel. An efficient and accurate method for evaluating time series similarity. In SIGMOD, pages 569–580, 2007.

[17] C.-W. Ngo, W. Zhao, and Y.-G. Jiang. Fast tracking of near-duplicate keyframes in broadcast domain with transitivity propagation. In ACM Multimedia, pages 845–854, 2006.

[18] S. Rasetic, J. Sander, J. Elding, and M. A. Nascimento. A trajectory splitting model for efficient spatio-temporal indexing. In VLDB, pages 934–945, 2005.

[19] H. T. Shen, B. C. Ooi, X. Zhou, and Z. Huang. Towards Effective Indexing for Very Large Video Sequence Database. In SIGMOD, pages 730–741, 2005.

[20] H. T. Shen, X. Zhou, Z. Huang, J. Shao, and E. Zhou. Uqlips: A real-time near-duplicate video clip detection system. In VLDB, pages 1374–1377, 2007. [21] M. Vlachos, M. Hadjieleftheriou, D. Gunopulos, and E. J. Keogh. Indexing multi-dimensional time-series with support for multiple distance measures. In SIGKDD, pages 216–225, 2003.

[22] H. Wu, B. Salzberg, and D. Zhang. Online event-driven subsequence matching over financial data streams. In SIGMOD, pages 23–34, 2004.

[23] X. Wu, A. G. Hauptmann, and C. W. Ngo. Practical elimination of near-duplicates from web video search. In ACM Multimedia, pages 218–227, 2007.

[24] Y. Yan, B. C. Ooi, and A. Zhou. Continuous content-based copy detection over streaming videos. In ICDE, 2008.

[25] B.-K. Yi and C. Faloutsos. Fast time sequence indexing for arbitrary Lp norms. In VLDB, pages 385–394, 2000.

5 [26] X. Zhu, X. Wu, J. Fan, A. K. Elmagarmid, and W. G. Aref. Exploring video content structure for hierarchical summarization. Multimedia System, 10(2):98–115, 2004.

[27] J. Zobel and T. C. Hoad. Detection of video sequences using compact signatures. TOIS, 24(1):1–50, 2006.

10 [28] Tudor Barbu. Novel automatic video cut detection technique using Gabor filtering: Computers & Electrical Engineering, Volume 35, Issue 5, September 2009, Pages 712-721

In compliance with the statute, the invention has been described in language
15 more or less specific to structural or methodical features. It is to be understood that the invention is not limited to specific features shown or described since the means herein described comprises preferred forms of putting the invention into effect. The invention is, therefore, claimed in any of its forms or modifications within the proper scope of the appended claims appropriately interpreted by
20 those skilled in the art.

The words “comprising”, “comprises” and their derivatives and variations are used throughout this specification and the appended claims in an inclusive sense and not to the exclusion of other features and integers.

CLAIMS

1. A method of operating an electronic apparatus to indicate a near-duplicate relationship between a test video clip and a query video clip, the method including the steps of:

determining test smoothing functions corresponding to segments of the test video clip;

processing the test smoothing functions and query smoothing functions for the query video clip to determine matched segments of the test video clip and the query video clip;

flagging the near-duplicate relationship based on the determination of the matched segments.

2. A method according to claim 1, wherein the step of determining test smoothing functions includes processing Video Distance Trajectories (VDTs) of the segments to determine parameters representing linear smoothing functions (LSFs).

3. A method according to claim 1 or claim 2, wherein the step of processing the test smoothing functions and query smoothing functions to determine matched segments includes calculating a segment similarity measure.

4. A method according to claim 3, wherein the segment similarity measure is calculated with operations corresponding to;

shifting

rotating; and

scaling;

one of said linear smoothing functions to map onto to the other.

5. A method according to claim 3 or claim 4, wherein the step of processing the test smoothing functions and query smoothing functions to determine matched segments of the test video clip and the query video clip includes

comparing a similarity measure to a predetermined segment matching threshold value.

6. A method according to any one of claims 2 to 5, wherein the step of flagging the near duplicate relationship includes computing a Weighted Edit Similarity (WES) measure.
7. A method according to claim 6, wherein the step of calculating the WES includes computing the similarity between a query sequence Q of the query video clip and a comparison window sequence V of the test video clip as the maximal weight of matched LSFs in Q .
8. A method according claim 7, wherein the step of flagging the near-duplicate relationship based on the determination of the matched segments includes comparing the WES to a predefined sequence matching threshold value.
9. A method according to any one of claims 7 and 8, including shifting the comparison window of the test video clip relative to the query video clip by skipping a number of segments calculated with reference to the WES between the query sequence and the window sequence.
10. A method according to any one of claims 7 to 9, including shifting a comparison window of the test video clip relative to the query video clip by skipping a number of segments of the comparison window preceding a leading segment of said window.
11. A method according to any one of the preceding claims including:
processing the test video clip to detect segments of said clip.
12. A method according to claim 11, wherein the step of processing the test video clip to detect segments of said clip includes producing color vector signals representing color vectors corresponding to frames of said clip.

13. A method according to claim 12, including processing the color vector signals to produce a color distance signal indicating a color distance between proximal, or consecutive, frames of said clip.
14. A method according to claim 13, including generating a cut signal to indicate a segment cut upon the color distance complying with a predetermined condition.
15. A method according to any one of claims 1 to 14, including pre-processing the query video clip to produce the query smoothing functions.
16. A method according to claim 16, wherein the query video clip is pre-processed to produce the query smoothing functions at a remote site.
17. A method according to claim 16 including distributing software to the remote site for producing the query smoothing functions from the query video clip.
18. A computer software product stored on a program storage device, readable by machine, tangibly embodying a program of instructions executable by the machine to cause the machine to perform the method of claim 1 to indicate the near-duplicate relationship between the test video clip and the query video clip.
19. A computational device including at least one electronic processor responsive to an electronic video signal and in electrical communication with an electronic memory for detecting near-duplicate videos, the electronic processor being arranged to execute instructions stored in the electronic memory, said instructions including:
 - instructions to process the video signal to flag the near duplicate relationship in accordance with the method of claim 1.

20. A near-duplicate video detection apparatus arranged to determine if a test video clip is a near duplicate of a query video clip and to generate a near-duplicate alert, the video detection apparatus comprising:

- a cut detector to produce cut signals indicating segments of the test video clip;

- a feature extractor unit arranged to produce a plurality of feature signals for each frame of the test video, said feature signals corresponding to feature vectors for frames of the test video;

- a video distance trajectory processor arranged to generate distance signals corresponding to each plurality of feature signals;

- a linear smoothing processor responsive to the distance signals and to the cut signals and arranged to process the distance signals to calculate smoothed test video line signals for each of the segments;

- a similarity calculator arranged to process the smoothed test video line signals and smoothed query video line signals to produce segment similarity signals

- a weighted edit similarity processor arranged to process the segment similarity signals and to produce a sequence similarity signal indicating similarity between the query video clip and the test video clip; and

- an alert module responsive to the similarity signals to produce the near-duplicate alert upon the similarity signals indicating a similarity measure closer than a predefined threshold.

21. An apparatus according to claim 20, including a color vector generator arranged to produce color vector signals representing color vectors corresponding to frames of the test video.

22. An apparatus according to claim 21, including an inter-frame distance calculator responsive to the color vector generator and arranged to produce a color distance signal indicating a color distance between proximal, or consecutive, frames of the test video clip.

23. An apparatus according to claim 22, wherein the cut detector includes a threshold comparator responsive to the inter-frame distance calculator and arranged to generate a cut signal upon the color distance signal complying with a predetermined condition.

24. An apparatus according to claim 23, wherein the predetermined condition is that the color distance signal exceeds a predetermined threshold value.

25. A method for operating a networked near duplicate video clip detection system, the method comprising:

- making video clip summarization software available to a number of subscribers to the system for said subscribers to produce video clip query summarizations;

- receiving subscriber video clip query summarizations from the subscribers via a computer network;

- performing near duplicate video clip detection for each subscriber video clip query summarization; and

- alerting subscribers upon detecting a near duplicate video clip in respect of their video clip query summarizations.

26. A networked near duplicate video clip detection method according to claim 25, wherein the step of performing near duplicate video clip detection for each subscriber video query summarization comprises a method according to any one of claims 1 to 17.

*

*

*

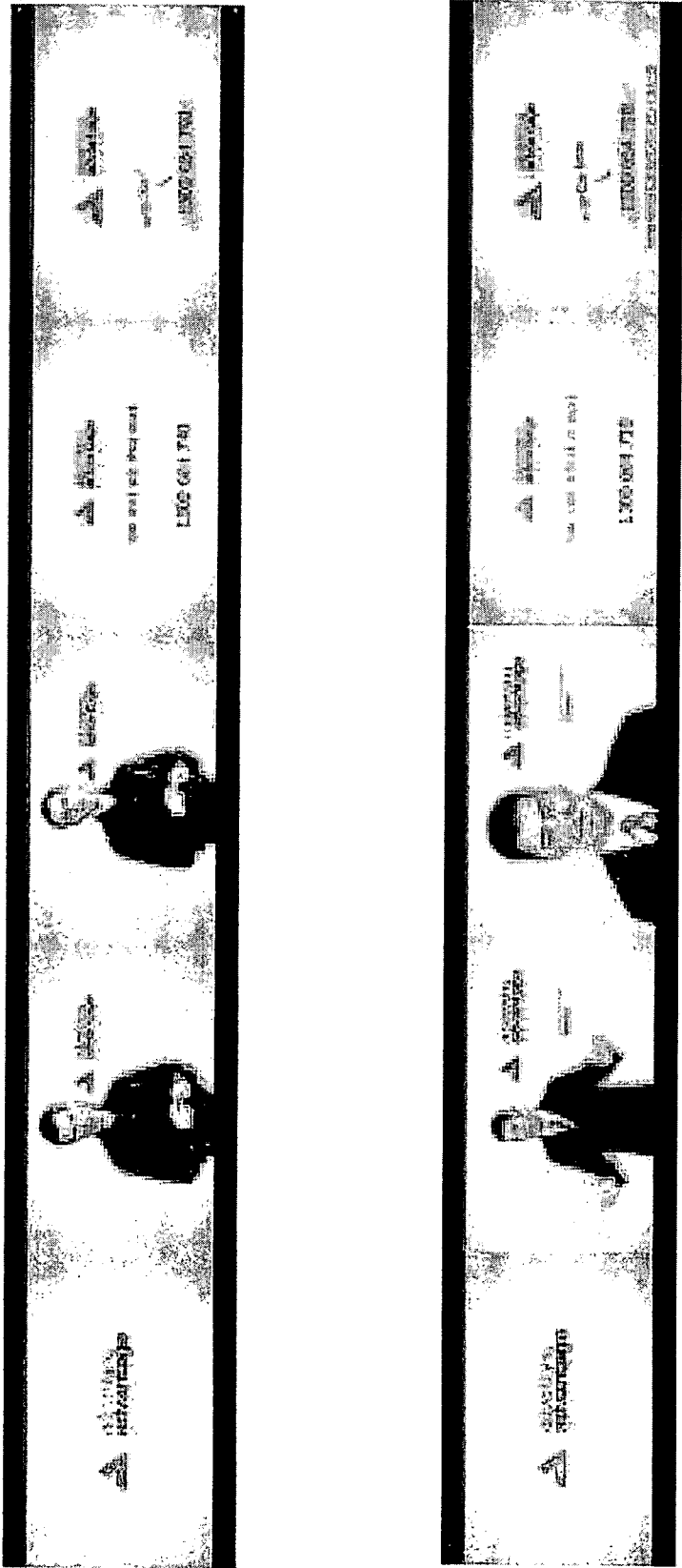


Fig. 1

2/17

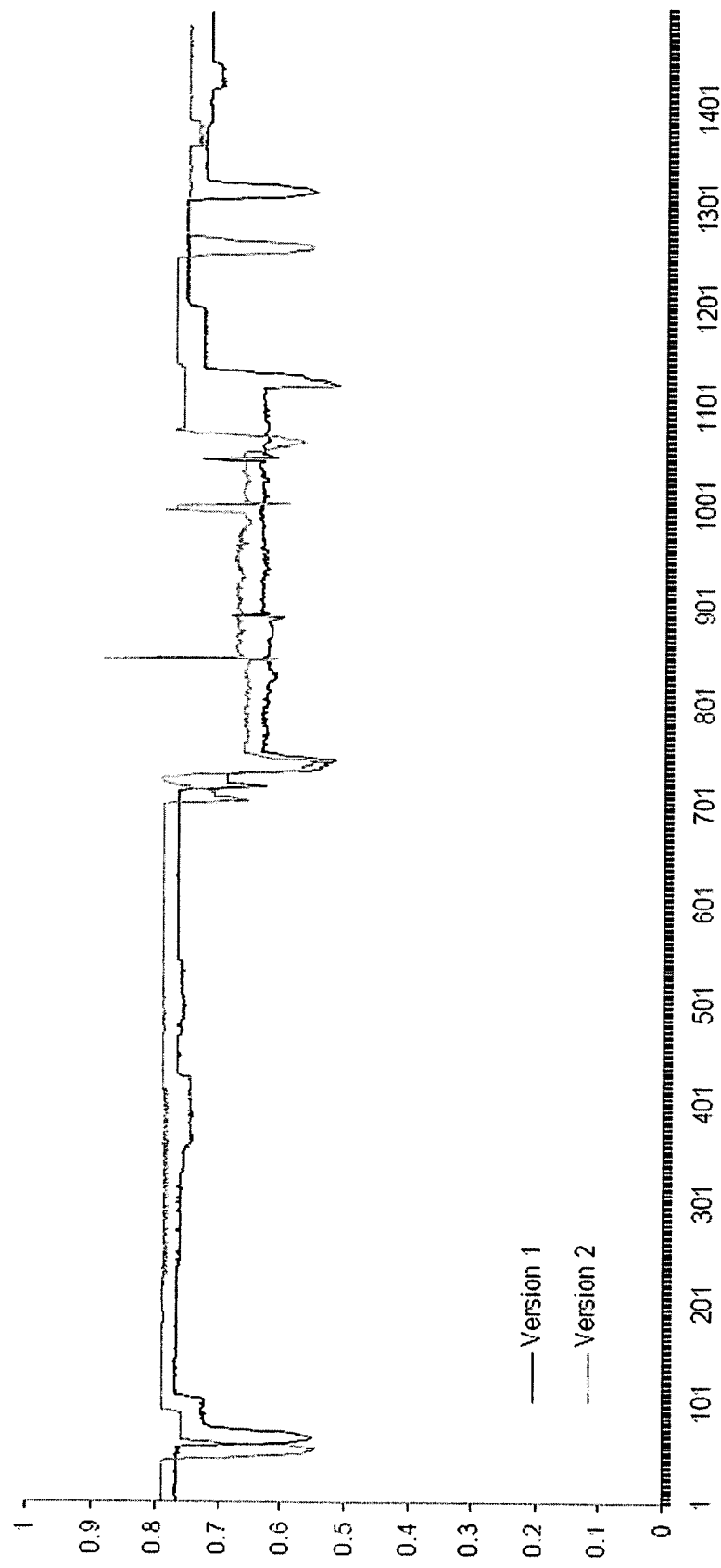


Fig. 2

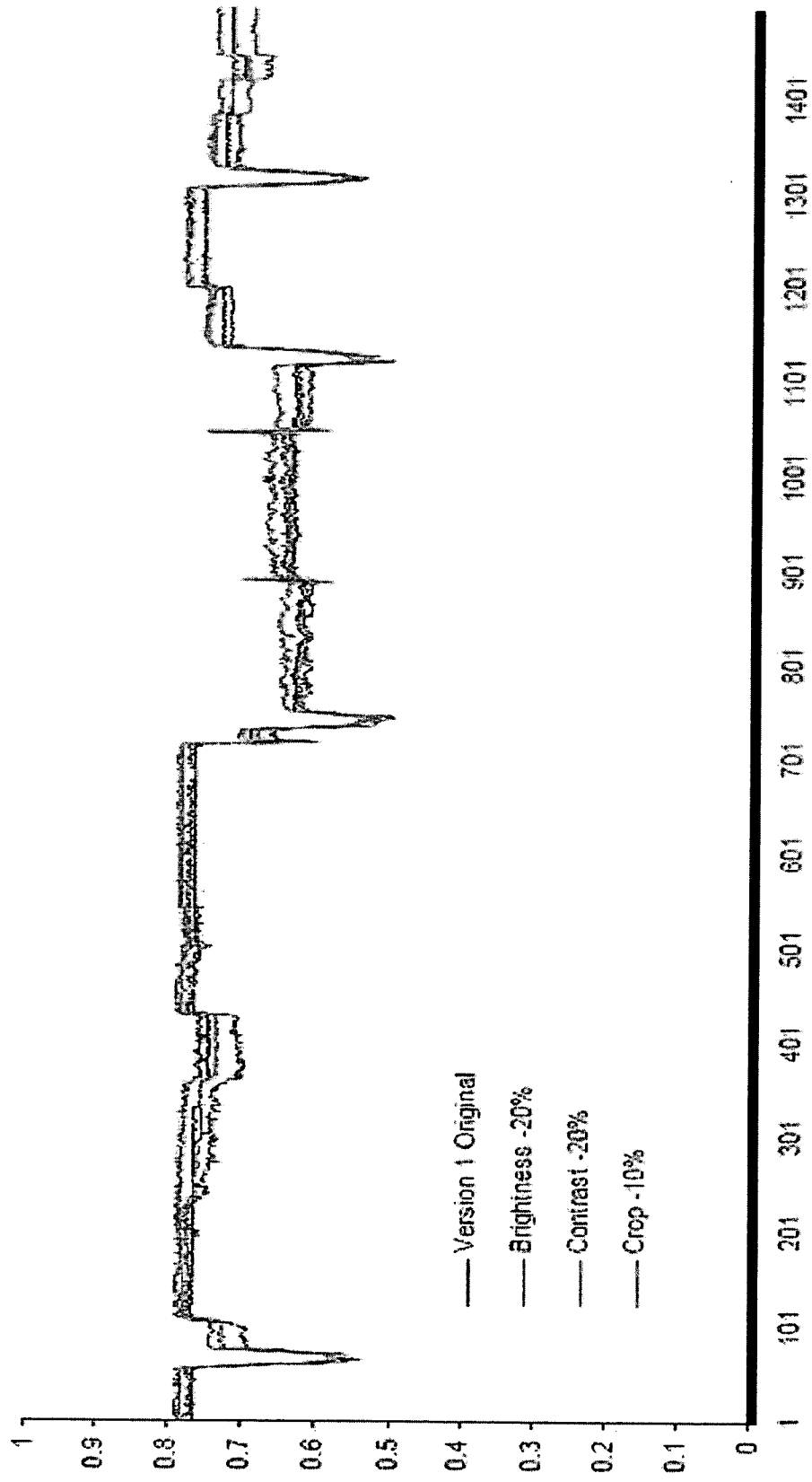


Fig. 3

4/17

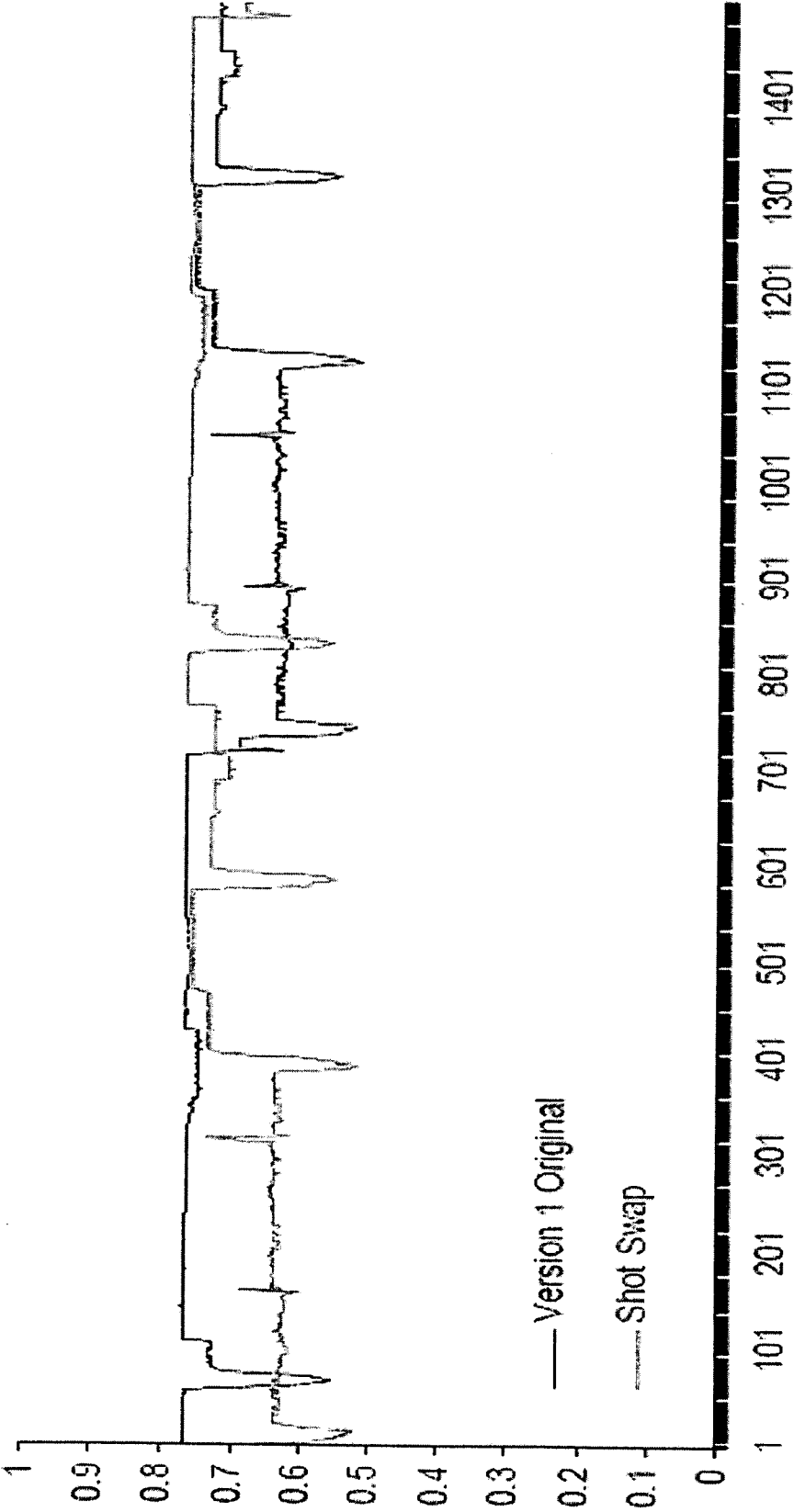
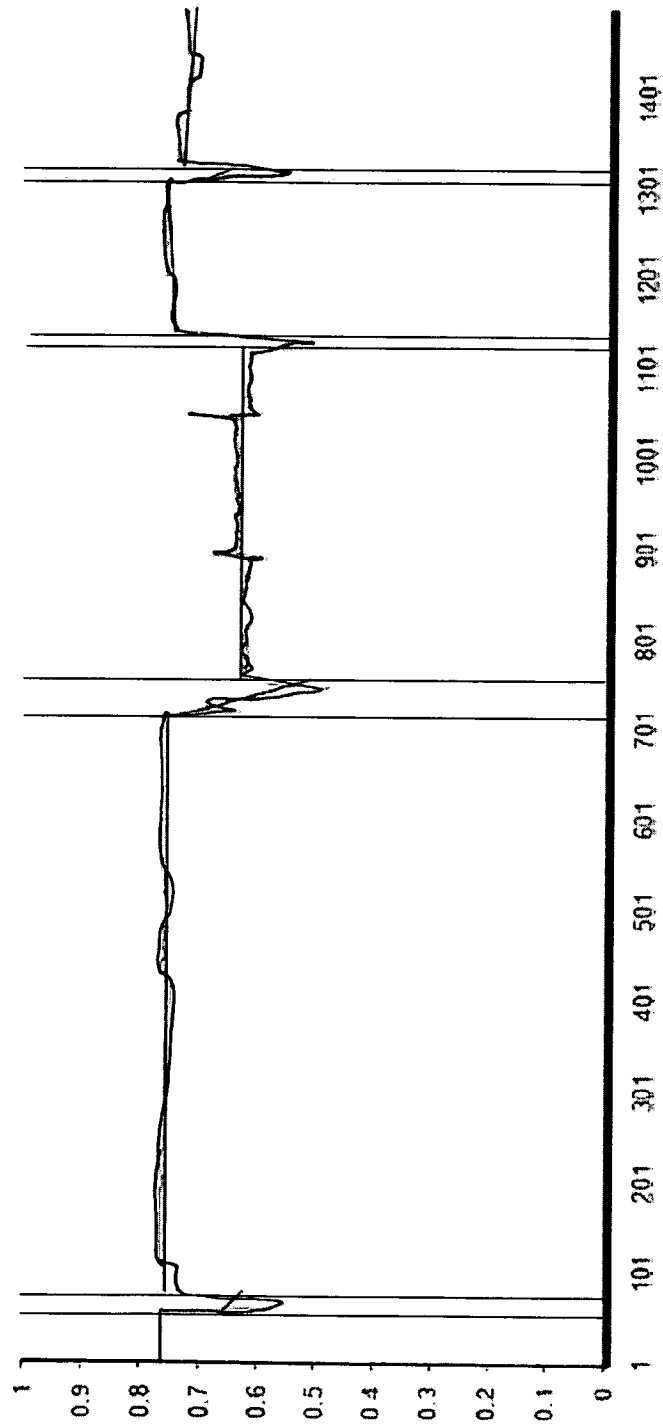


Fig. 4

5/17

Fig. 5

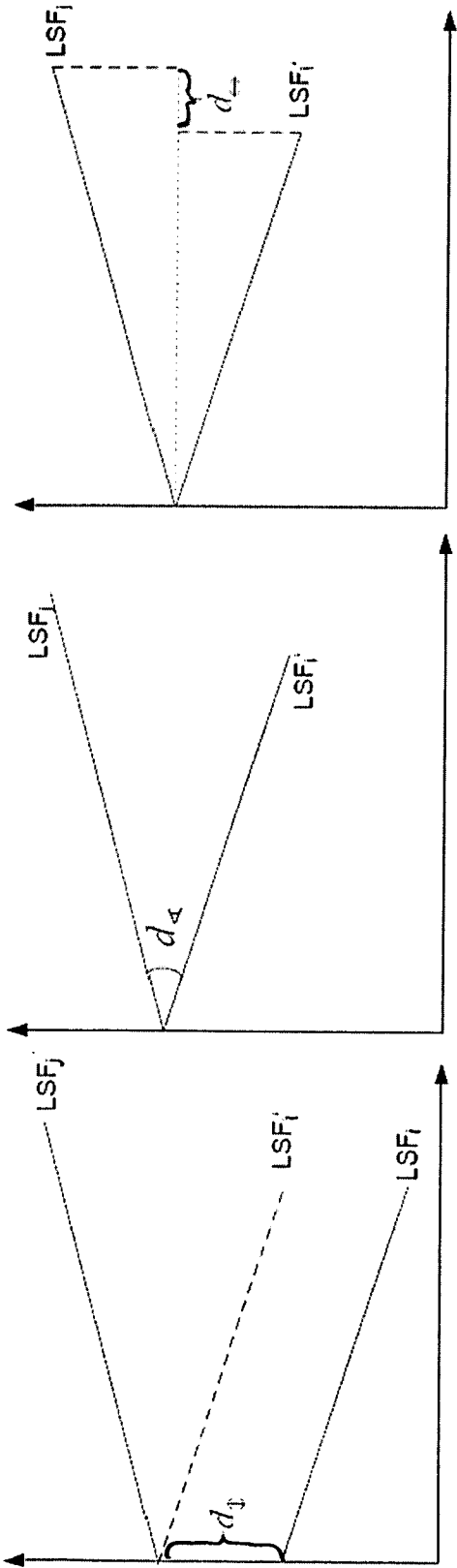


Fig. 6

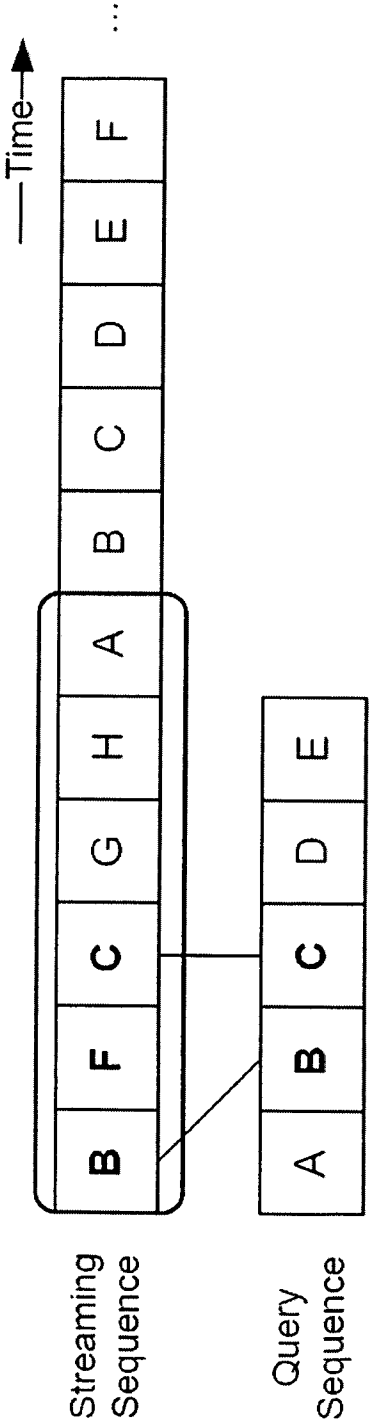


Fig. 7

7/17

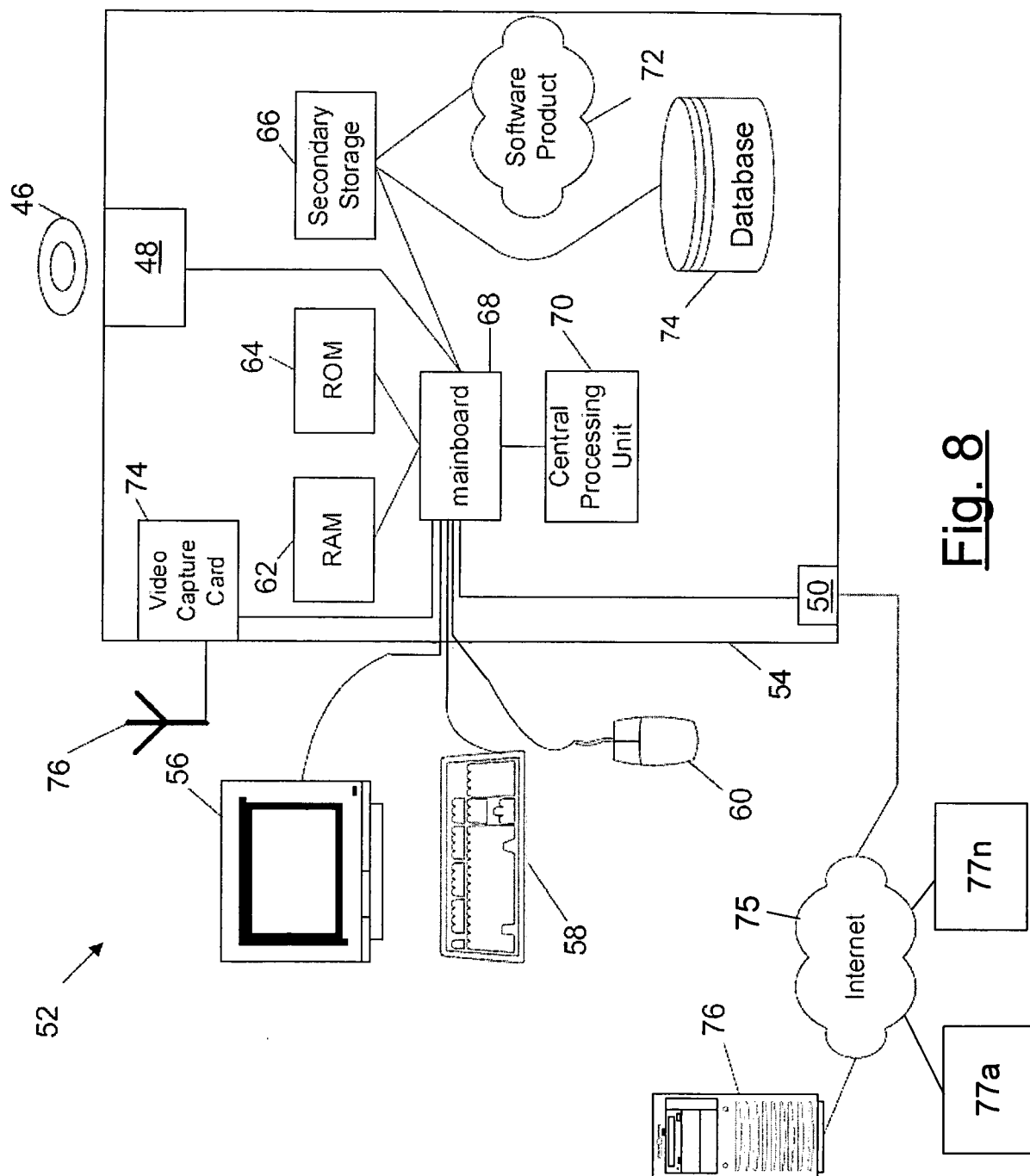
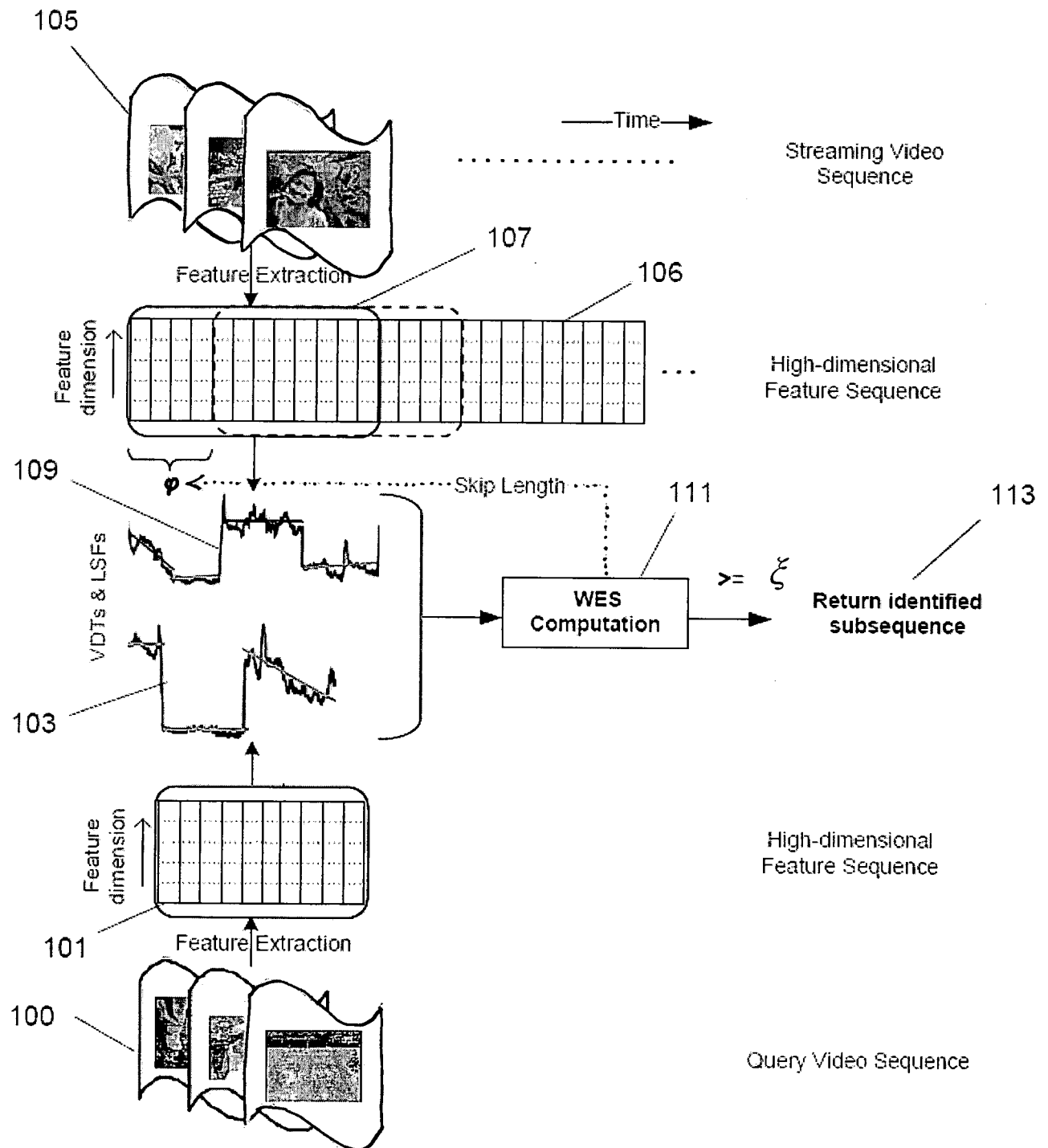
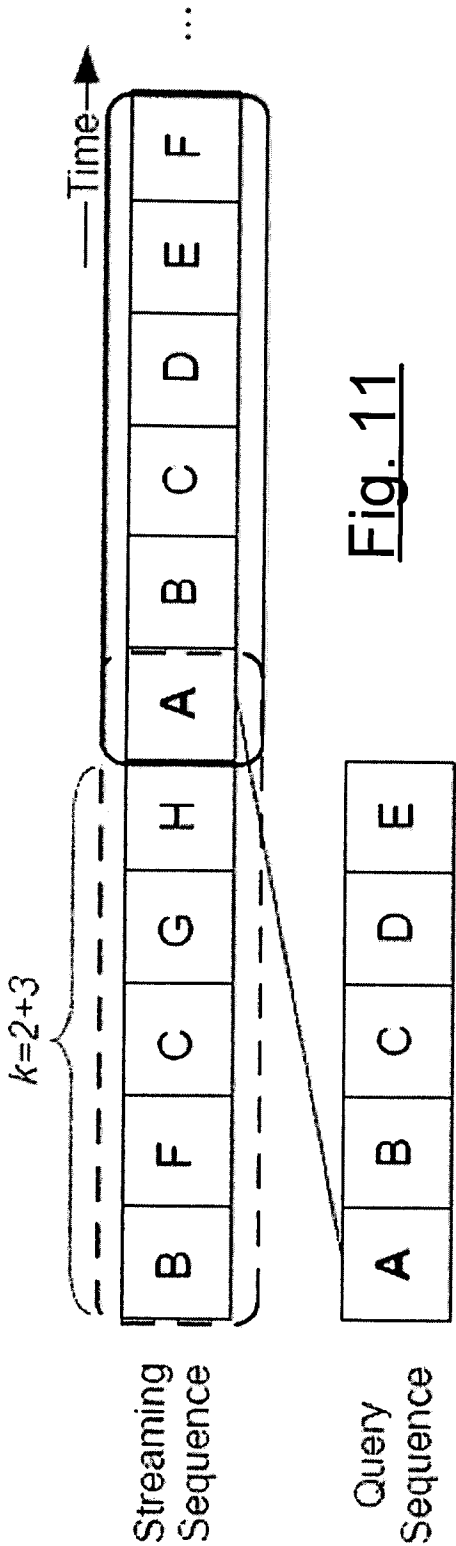
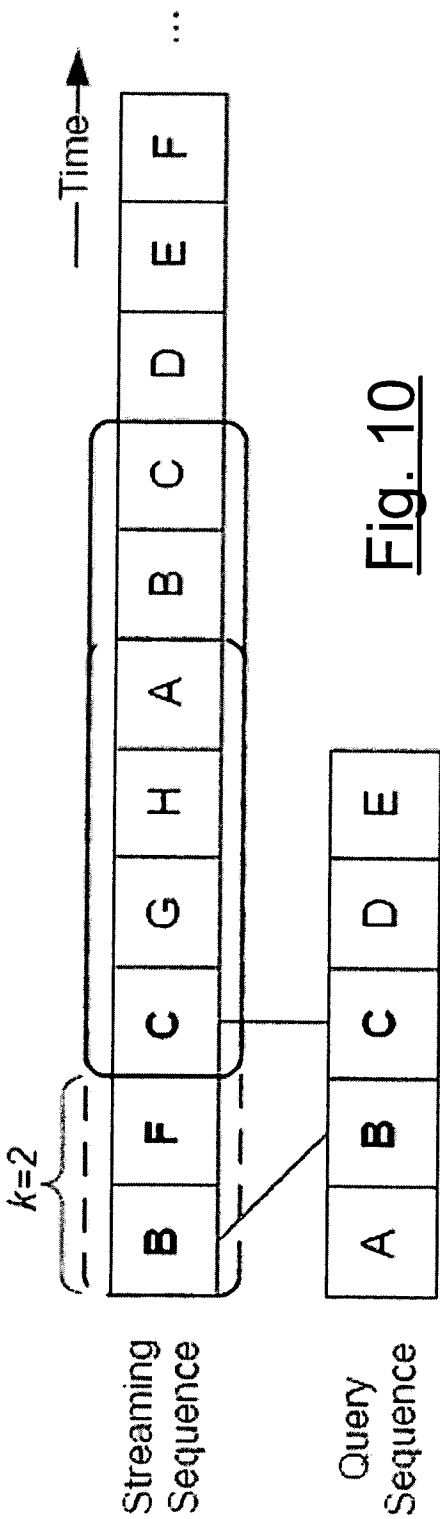
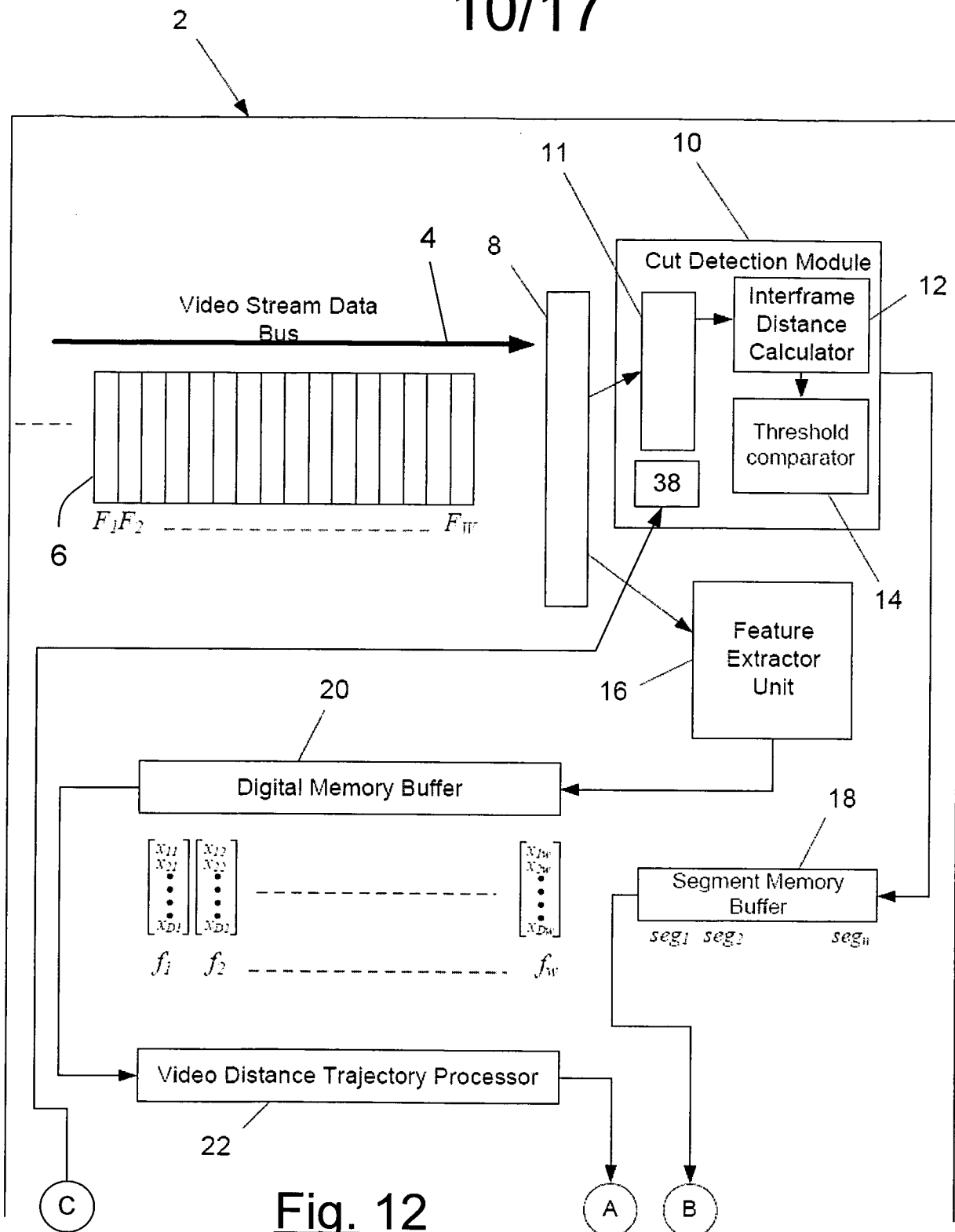


Fig. 8

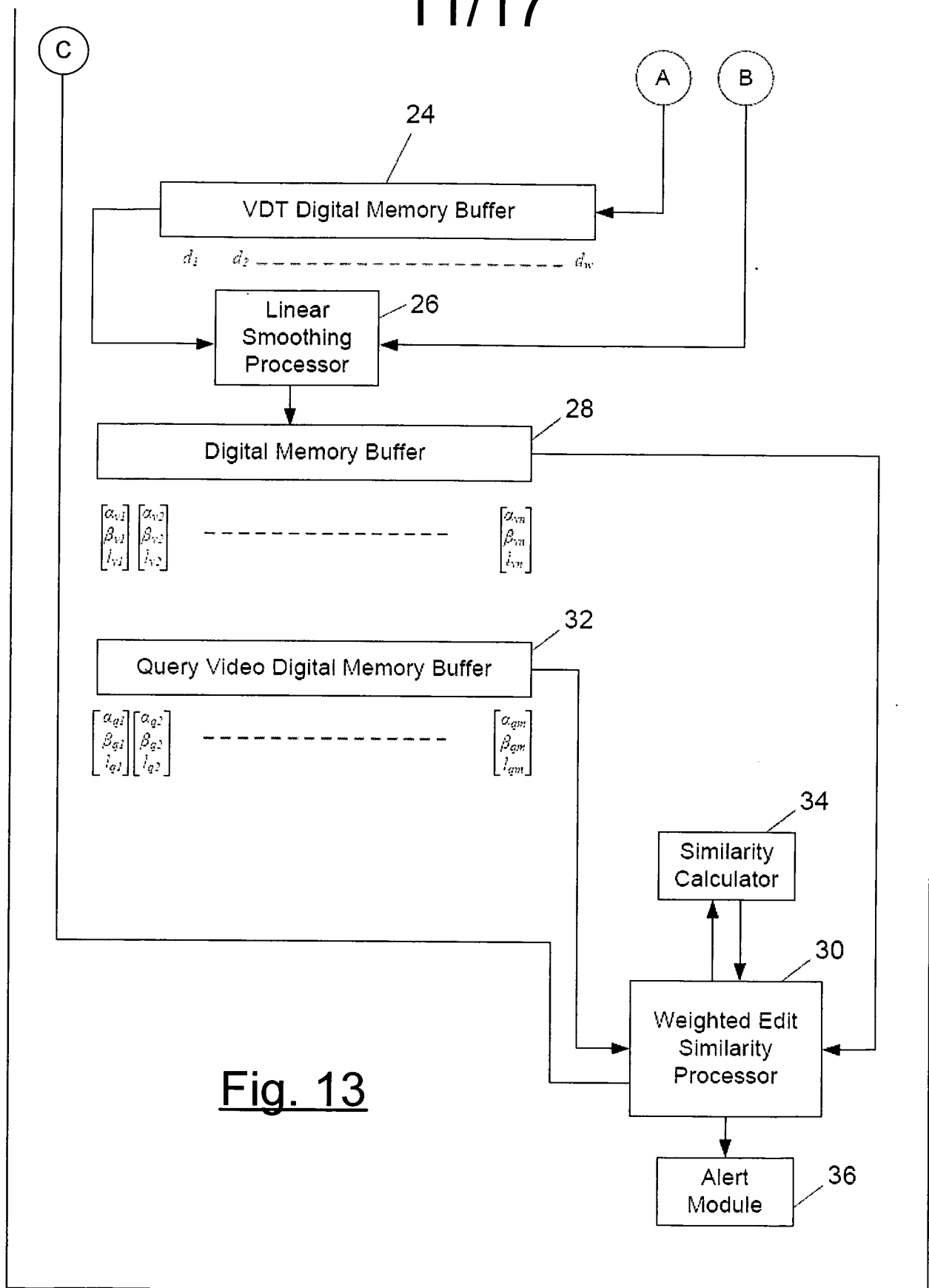
8/17

Fig. 9





11/17



12/17

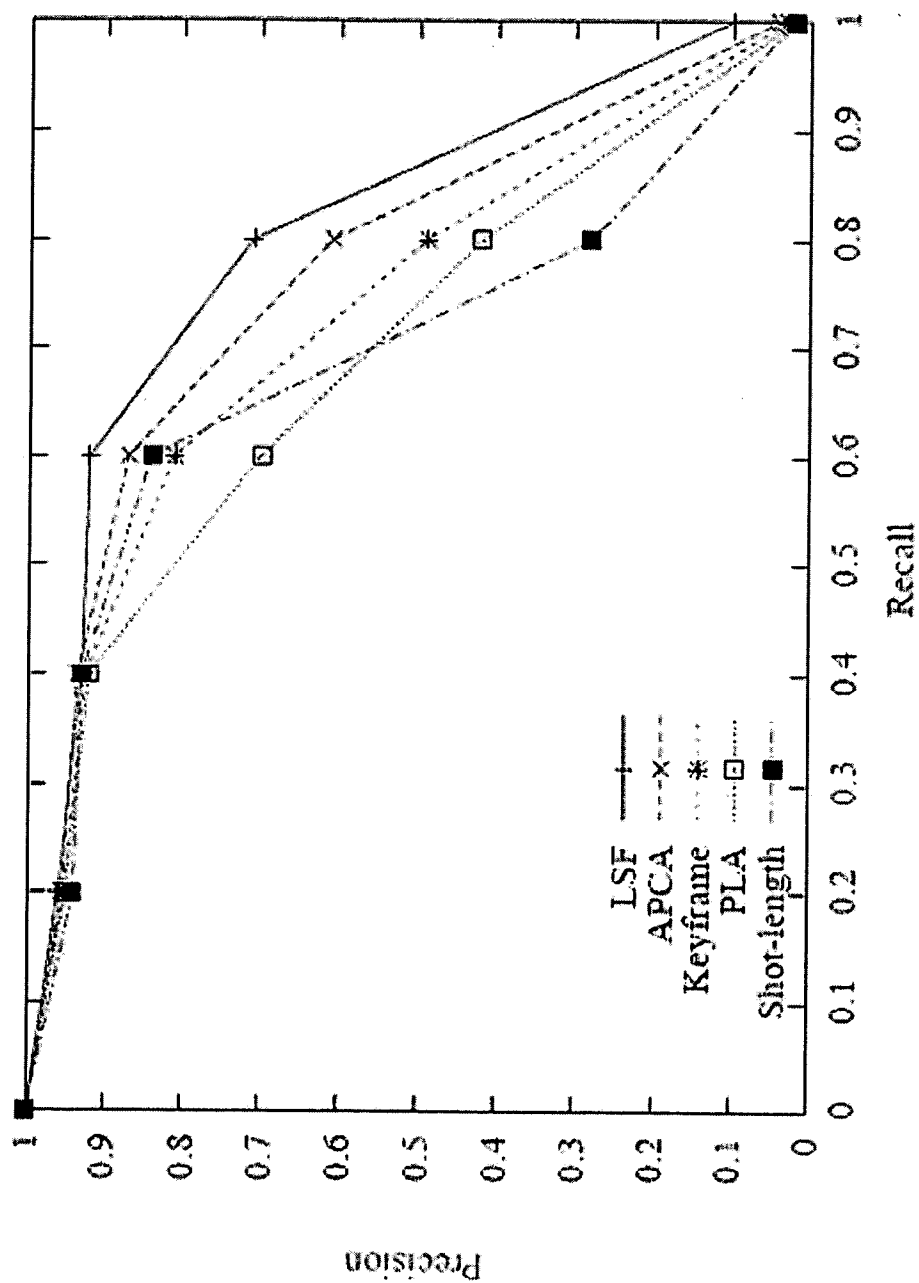


Fig. 14

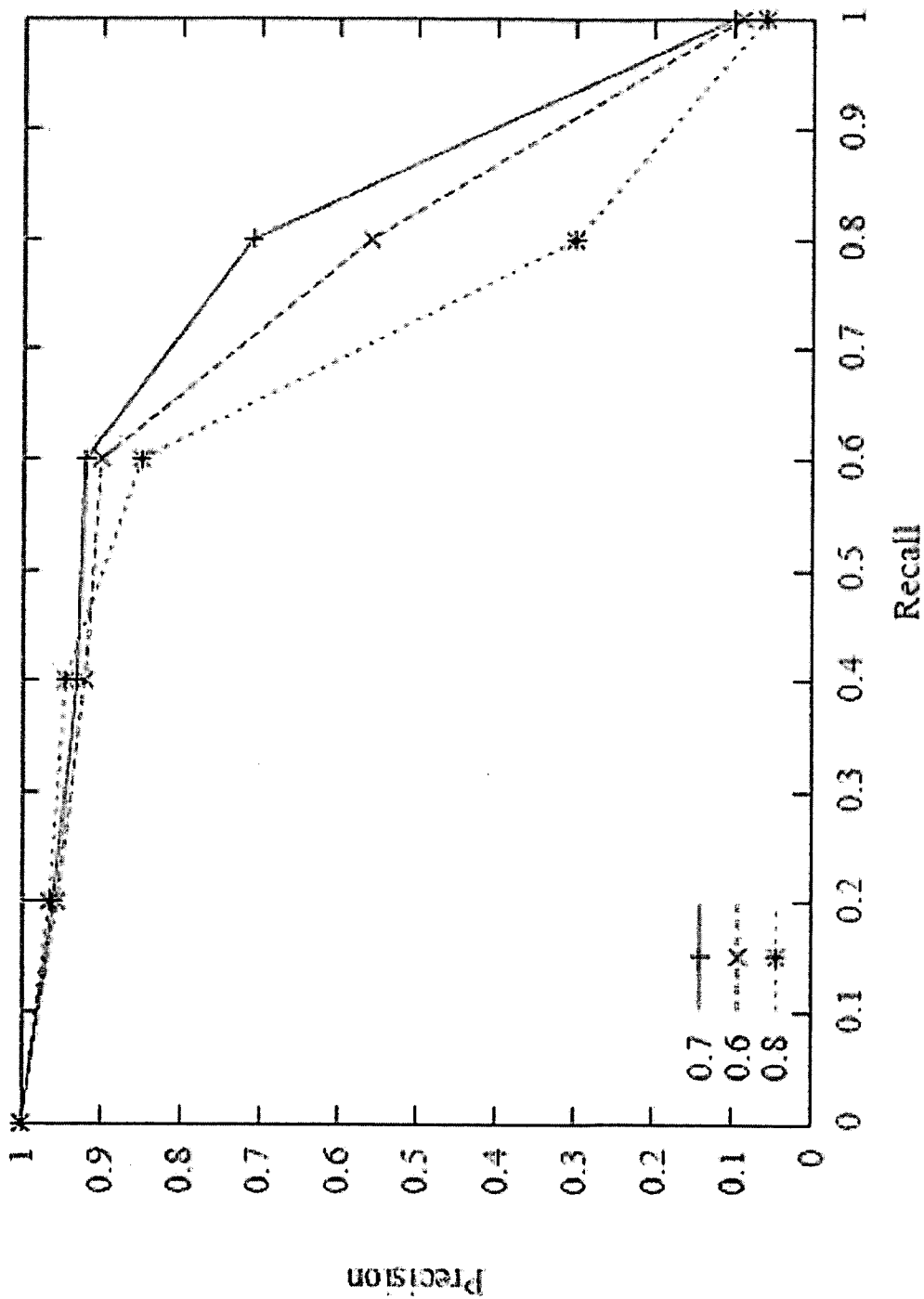


Fig. 15

14/17

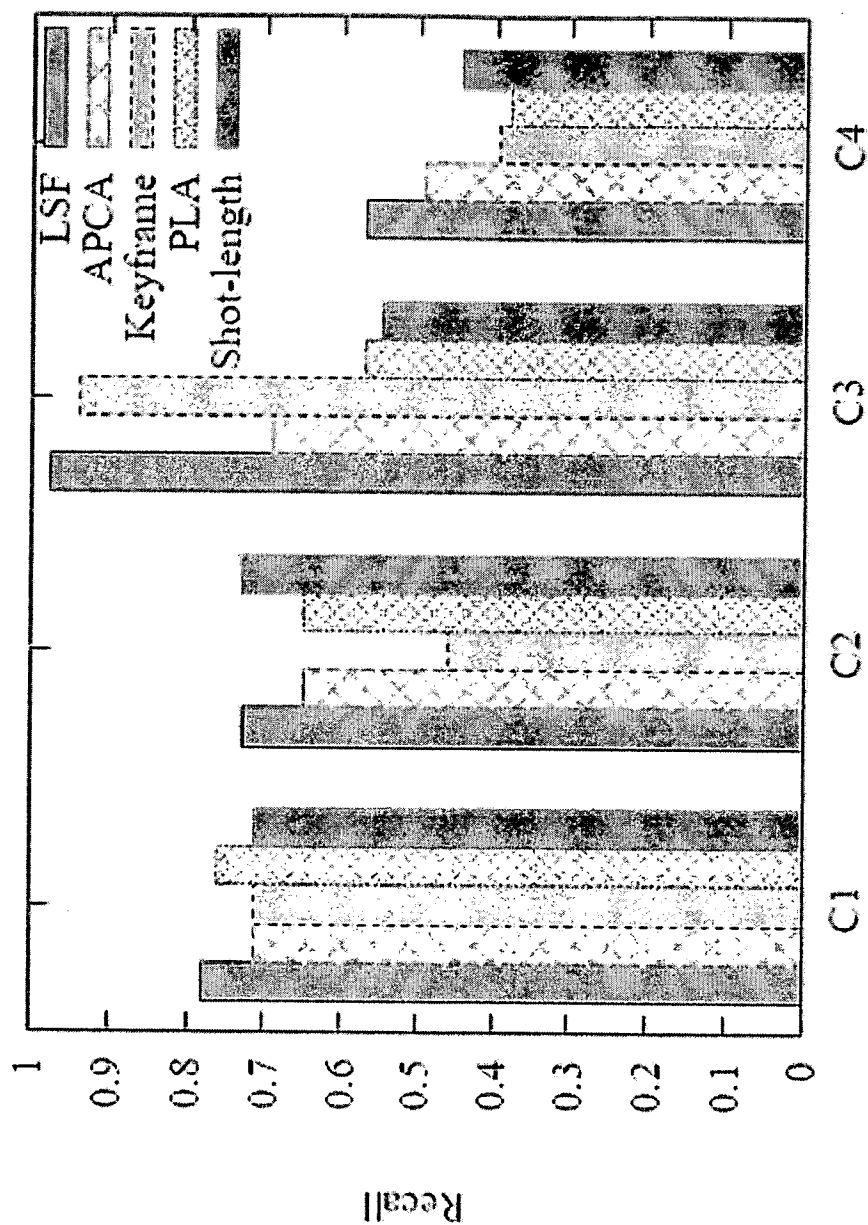
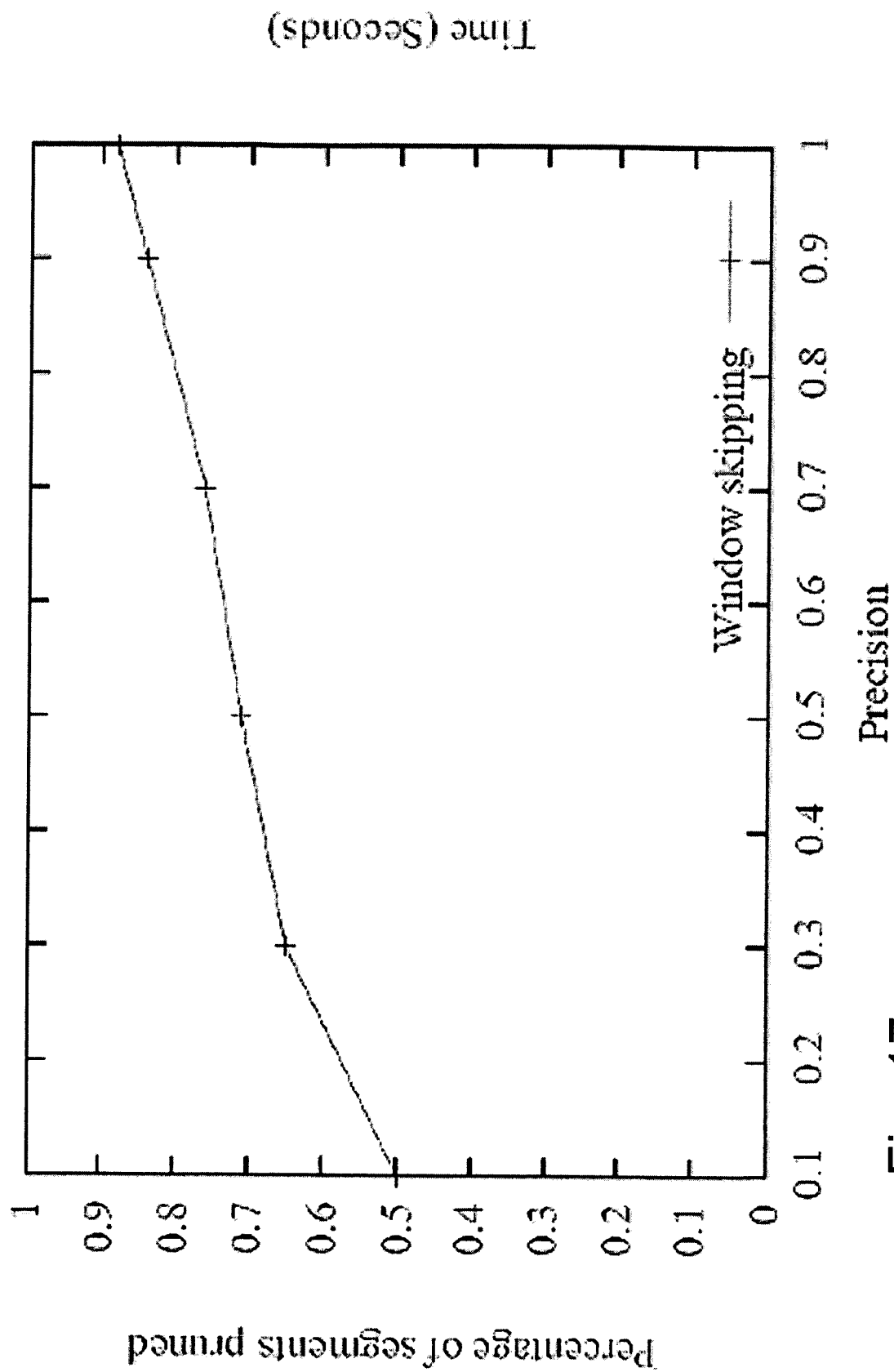
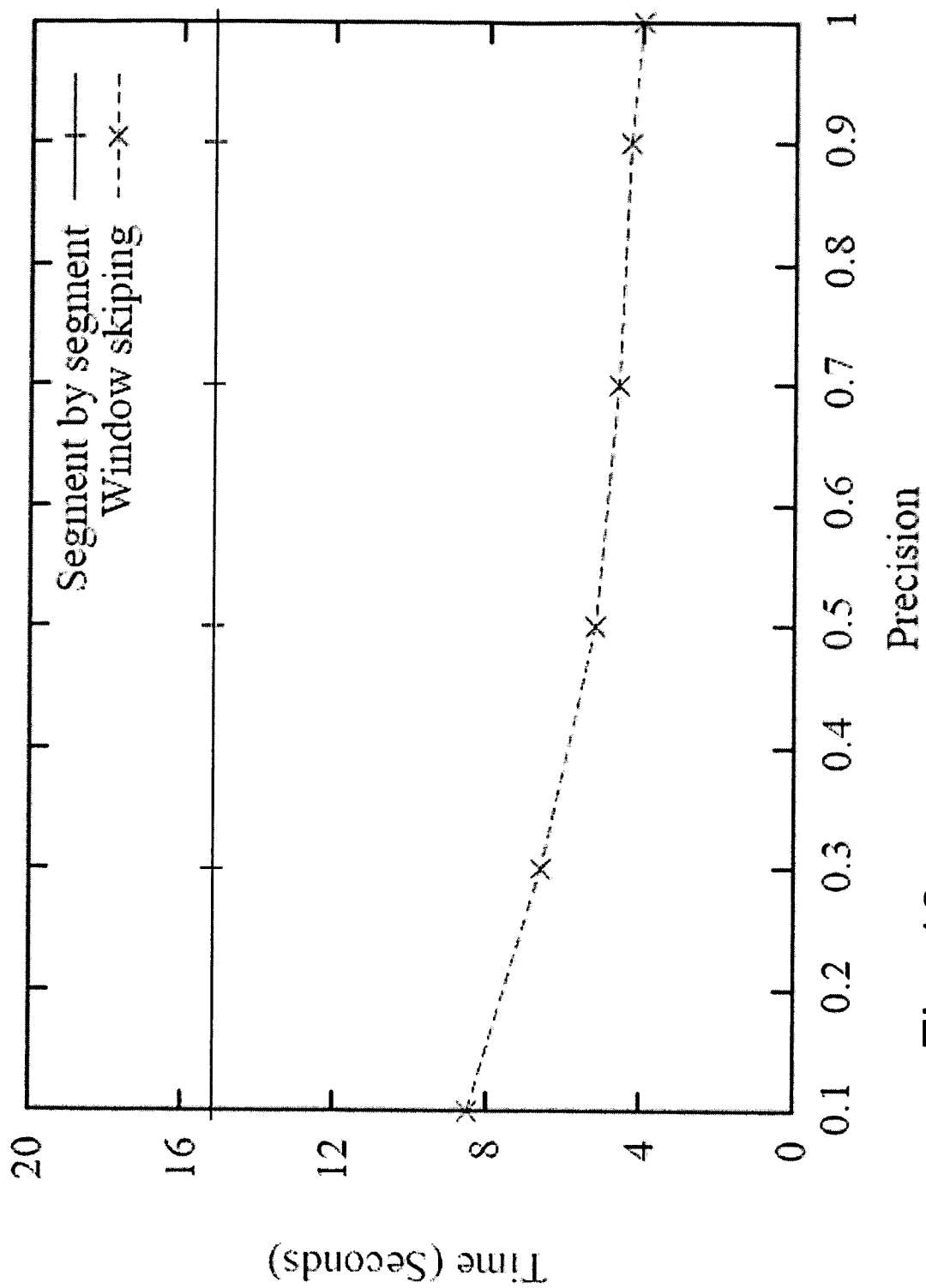


Fig. 16

15/17

Fig. 17

16/17

Fig. 18

17/17

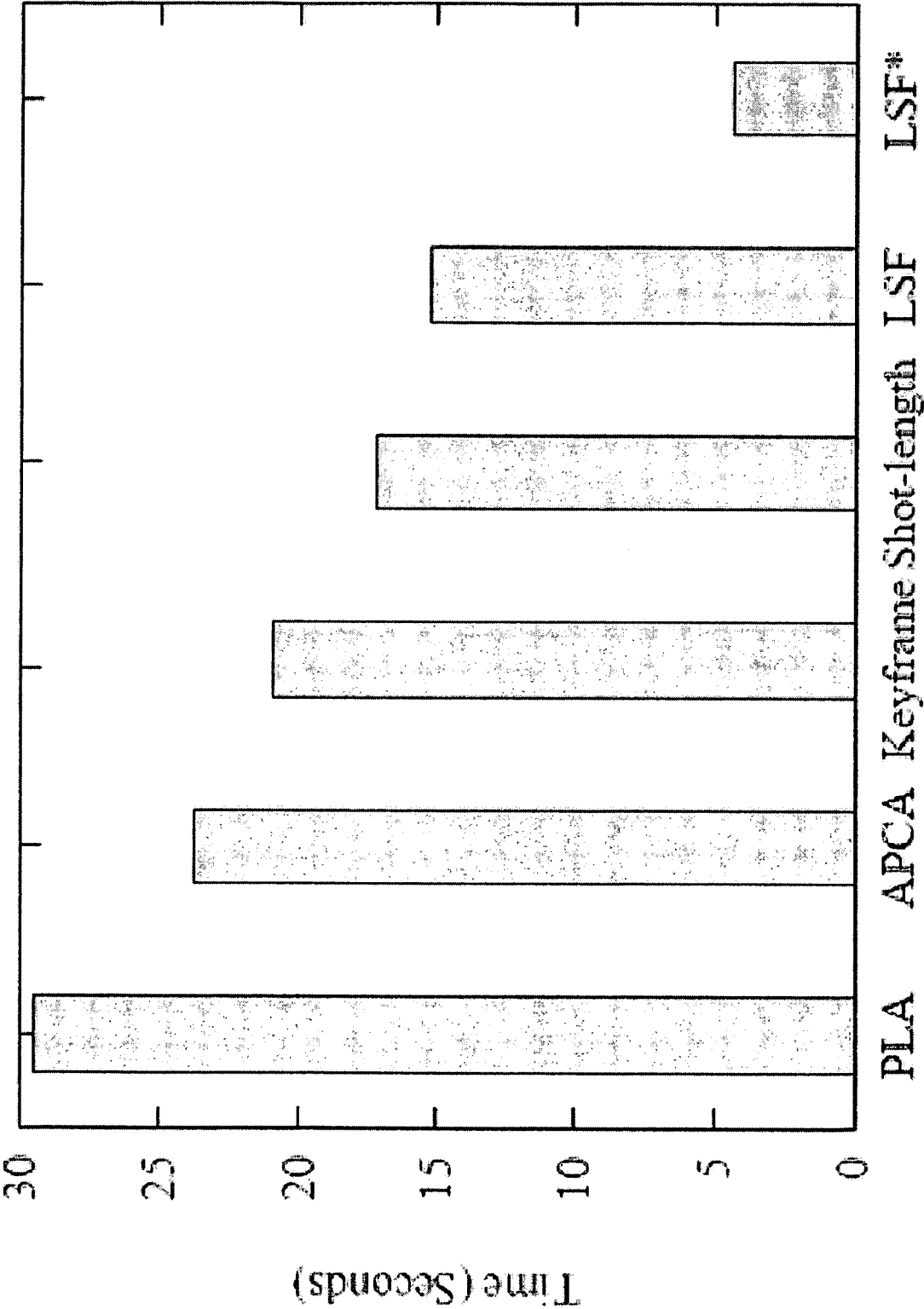


Fig. 19

INTERNATIONAL SEARCH REPORT

International application No.
PCT/AU2010/000020

A. CLASSIFICATION OF SUBJECT MATTER Int. Cl. H04N 5/93 (2006.01) According to International Patent Classification (IPC) or to both national classification and IPC		
B. FIELDS SEARCHED Minimum documentation searched (classification system followed by classification symbols) Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) EPODOC, WPI, INSPEC: IPC H04N 5/93, with keywords duplicate, video, NDVC, detect, query and similar terms. Google Scholar, Google Patents, IEEE Xplore, Google and PatentScope searched with similar keywords as above.		
C. DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	Shen et al., UQLIPS: A Real-time Near-duplicate Video Clip Detection System, VLDB 07, 23-28 September, 2007, Vienna, Austria, pages 1374-1377. See the whole document in particular figure 1, sections 1, 2.1-2.2, 3, and 5.	1, 3-5, 11-12, 15-17, 18-19, 25-26
Y	LU et al., Hierarchical Indexing Structure for Efficient Similarity Search in Video Retrieval, IEEE Transactions on Knowledge and Data Engineering, Vol. 18, No. 11, November 2006, pages 1544-1559.	2, 13
X	See the whole document in particular figure 1, sections 1 and 2.1.	1, 3, 11, 15, 18-19, 25-26
☒ Further documents are listed in the continuation of Box C ☒ See patent family annex		
* Special categories of cited documents: "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier application or patent but published on or after the international filing date "L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "&" document member of the same patent family		
Date of the actual completion of the international search 03 March 2010		Date of mailing of the international search report 8 MAR 2010
Name and mailing address of the ISA/AU AUSTRALIAN PATENT OFFICE PO BOX 200, WODEN ACT 2606, AUSTRALIA E-mail address: pct@ipaaustralia.gov.au Facsimile No. +61 2 6283 7999		Authorized officer Dr. RASIKA PERERA AUSTRALIAN PATENT OFFICE (ISO 9001 Quality Certified Service) Telephone No : +61 2 6283 3116

INTERNATIONAL SEARCH REPORT

International application No.

PCT/AU2010/000020

C (Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT		
Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
Y	WU et al., Practical Elimination of Near-Duplicates from Web Video Search, MM' 07, 23-28 September 2007, pages 218-227, Augsburg, Bavaria, Germany. See the whole document in particular figure 4, sections 4.1 and 4.2.	2, 13
P, X	US 2009/0028517 A1 (SHEN et al.) 29 January 2009. See the whole document in particular the abstract, figures, paragraphs 0030-0034, 0066 and 0087.	1-5, 11, 15-19, 25-26

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No.

PCT/AU2010/000020

This Annex lists the known "A" publication level patent family members relating to the patent documents cited in the above-mentioned international search report. The Australian Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

Patent Document Cited in Search Report		Patent Family Member
US	2009028517	NONE
Due to data integration issues this family listing may not include 10 digit Australian applications filed since May 2001.		
END OF ANNEX		