Near-Lossless Semantic Video Summarization and Its Applications to Video Analysis

Tao Mei, Microsoft Research Asia
Lin-Xie Tang, University of Science and Technology of China
Jinhui Tang, Nanjing University of Science and Technology
Xian-Sheng Hua, Microsoft

The ever increasing volume of video content on the Web has created profound challenges for developing efficient indexing and search techniques to manage video data. Conventional techniques such as video compression and summarization strive for the two commonly conflicting goals of low storage and high visual and semantic fidelity. With the goal of balancing both video compression and summarization, this paper presents a novel approach, called "Near-Lossless Semantic Summarization" (NLSS), to summarize a video stream with the least high-level semantic information loss by using an extremely small piece of metadata. The summary consists of compressed image and audio streams, as well as the metadata for temporal structure and motion information. Although at a very low compression rate (around $\frac{1}{40}$ of H.264 baseline, where traditional compression techniques can hardly preserve an acceptable visual fidelity), the proposed NLSS still can be applied to many video-oriented tasks, such as visualization, indexing and browsing, duplicate detection, concept detection, and so on. We evaluate the NLSS on TRECVID and other video collections, and demonstrate that it is a powerful tool for significantly reducing storage consumption, while keeping high-level semantic fidelity.

Categories and Subject Descriptors: H.5.1 [**Information Interfaces and Presentation**]: Multimedia Information Systems—*video*; H.3.5 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*

General Terms: Algorithms, Experimentation, Human Factors.

Additional Key Words and Phrases: Video summarization, video storage, video content analysis, video applications.

ACM Reference Format:
Mei, T., *et al.*, 2012. ACM Trans. Multimedia Comput. Commun. Appl. 2, 3, Article 1 (May 2010), 23 pages.
DOI = 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

1. INTRODUCTION

We have witnessed an increase in the number of enormous videos on the Web, as well as private video collections on personal devices. The daunting yet increasing volume of video data has brought profound challenging problems to video-oriented services: 1) *limited storage capacity*—it would be difficult for a

T. Mei is with Microsoft Research Asia, Beijing 100080, P. R. China (email: tmei@microsoft.com). L.-X. Tang is with the Department of Electronic Engineering and Information Science, University of Science and Technology of China, Hefei 230027, P. R. China (email: lxtang@gmail.com). J. Tang is with the School of Computer Science, Nanjing University of Science and Technology, Nanjing, P. R. China (email: jinhuitang@mail.njust.edu.cn). X.-S. Hua is with Microsoft, USA (email: xshua@microsoft.com).

DOI 10.1145/0000000.0000000 http://doi.acm.org/10.1145/0000000.0000000

single video site to host all the uploaded videos; 2) *considerable streaming latency*—streaming such a large amount of videos over limited bandwidth capabilities would lead to considerable latency; and 3) *unsatisfying video quality*—blocking artifacts and visual distortion are usually expected by traditional compression techniques, especially when extremely low bitrate is applied, which in turn degrades user experience of video browsing and the performance of video analysis.

Generally, there are two solutions to the problems mentioned above, i.e., very-low bitrate video compression and video summarization. Video compression is designed from the perspective of signal processing, aiming to eliminate spatio-temporal signal redundancy. It is widely used in a majority of online video sites. For example, YouTube employs H.263 (at the bitrate of $200 \sim 900$ kbps) as the codec for "standard quality" video [H263 2000], and H.264 (at the bitrate of $2,000$ kbps) for "High-Definition (HD) quality" video [H264 2003]. However, these compression techniques only achieve limited reduction of original signal, compared with previous compression standards (e.g., H.264 usually obtains half or less the bitrate of MPEG-2 [Wiegand et al. 2003] [MPEG-2].) Moreover, when given an extremely low bitrate (e.g., less than $\frac{1}{20}$ of H.264 baseline profile), they usually introduce severe blocking artifacts and visual distortions which in turn degrade visual fidelity and user experience.

Video summarization is an alternative approach to handle large-scale video storage. It is designed from the perspective of computer vision, aiming to select a subset of informative frames or segments (e.g., shots or subshots) from original video and represent the video content in a static (i.e., a collection of synthesized images or keyframes) or dynamic (i.e., a new generated shorter video than the original) form. These frames or segments are then concatenated according to certain spatio-temporal orders to form a compact visual summary. Such summarization techniques can be used by any video search system to index the videos in the backend and present the search results in the front. For example, some video search engines present a short dynamic thumbnail (e.g., a 15 or 30 seconds clip) for each searched video for fast preview [Bing]. However, as content understanding is still in its infancy, the video summary cannot guarantee the preservation of all informative content. In other words, such a summary is a kind of lossy representation without keeping the semantic fidelity, especially when compression rate is very high.

Both conventional techniques of video compression and summarization are incapable of dealing with either of the above problems. An effective approach that fulfills the following objectives is highly desirable: 1) extremely low storage consumption, which can solve the problems of *limited storage capacity* and *streaming latency*, 2) the ability to be decoded so that the original video can be "reconstructed" or visualized without much visual fidelity loss, through which we can achieve *satisfying video quality* for browsing, and 3) capable of performing video content analysis with the least semantic loss based on the summary, through which high-level semantic video analysis can be performed effectively.

With these objectives in mind, we are seeking summarization techniques for keeping high-level semantic information as much as possible, while using an extremely small piece of metadata. We call this kind of summarization "semantic-preserving summarization" since the summarization is designed for high-level semantic analysis tasks. In particular, we extend our previous work [Tang et al. 2009] to a more efficient and comprehensive approach in this paper. The difference comes from the use of mature techniques in video processing and computer vision. The proposed new approach, called "Near-Lossless Semantic Summarization" (NLSS), can achieve an extremely low compression rate with the least semantic information loss. The near-lossless summary is achieved by making a trade-off between video compression and video summarization—using a set of compressed visual and audio streams, as well as the metadata of video structure and motion information, to summarize the semantic content of a video. This summary can in turn be used for a wide variety of video-oriented applications, including visualization, indexing, presentation, and video content analysis. By "near-lossless summarization," we refer to: by making a trade-off between video compression and video summarization, we can keep a small piece of metadata as the video summary for the purpose of semantic analysis, while maintaining acceptable visual fidelity if the summary is used for reconstructing the original video. Towards near-lossless semantic summarization, we identify several design principles as follows: 1) a video is processed in the granularity of the subshot (which is a sub segment within a shot, indicating the dominant coherent camera motion within this segment); 2) each subshot is then classified into four classes according to the dominant camera motion within this subshot; 3) a predefined set of mechanisms is performed to extract metadata (i.e., representative keyframes and synthesized mosaic images, as well as structure and motion information) from these subshots, resulting in an extremely low storage consumption. Moreover, through the summary, a series of applications can be applied with the least semantic information loss and performance degradation. NLSS significantly differs from traditional video compression in that it achieves a much lower compression rate without much visual distortion, and also differs from video summarization in that it contains all the information (can be used to reconstruct the original video with the same length) with the least semantic information loss.

In contrast to our previous "near-lossless video summarization" (NLVS) [Tang et al. 2009], the new proposed NLSS further compresses the mosaics together with the selected keyframes, making the summarization process more unified and the summary much more compact than those from NLVS. The NLSS segments the audio track into units and classifies each unit into different categories (i.e., *silence*, *music*, *pure speech*, *non-pure-speech*, and *background*), and further compresses the non-silence (i.e., the four categories except for *silence*) units with very-low bitrate compression technique, while NLVS simply treats the audio track as a whole. As a result, storage consumption is significantly reduced in NLSS. Moreover, we propose a wide variety of video applications based on the summary, and validate the effectiveness of NLSS summary for semantic video analysis through comprehensive evaluations.

To summarize, the contributions of this paper are as follows:

—We investigate the problem of video summarization from a novel perspective of high-level semantic-preserving, and propose an approach to near-lossless semantic summarization of video content, which is a powerful complement to existing video compression and summarization techniques. The proposed NLSS represents a tradeoff between video compression and summarization.

—We achieve extremely low storage consumption (around $\frac{1}{40}$ of the video compressed by H.264 baseline), where traditional video compression struggles to preserve a satisfying visual fidelity.

—We applied the summary to a wide variety of video-oriented applications, such as visualization, indexing, presentation, duplicate and concept detection, showing that the summary is able to keep both visual and semantic fidelity, making it a promising solution for many online video services.

The rest of the paper is organized as follows: Section 2 reviews related research. Section 3 presents the overview and details of NLSS. Section 4 discusses several applications based on the summary generated from NLSS. Section 5 presents the evaluations, followed by the conclusion in Section 6.

2. RELATED WORK

We review the techniques for video summarization, video compression, and video content analysis (especially content-based video indexing and browsing, video duplicate detection, as well as concept detection) in this section.

2.1 Video Summarization

Video summarization uses a subset of representative frames or segments from the original video to generate a compact representation [Truong and Venkatesh 2007]. Conventional video summarization has been developed along two dimensions according to the metadata used for visualization, i.e., static summarization and dynamic skimming. Static summarization represents a video sequence in a static imagery form (one or more selected representative frames from the original video, or a synthesized image generated from the selected keyframes) [Mei et al. 2009]. According to different sampling mechanisms, a set of keyframes are extracted from shots of the original video. Then, the selected keyframes are arranged or blended in a two dimensional space. Storyboarding is the most straightforward representation of video in the static form. By discovering the inherent relationship between the characters in a movie, Wang *et al.* propose a poster generation system [Wang et al. 2011]. Boreczky *et al.* presents an interactive comic book presentation for video browsing [Boreczky et al. 2000], while Tse *et al.* propose to display multiple keyframes in a "filmstrip" manner for nonlinear video browsing [Tse et al. 1998].

For dynamic summarization, most mechanisms extract and segment video clips from the original video. TRECVID provides a platform to summarize rush videos in a given compression ratio, where the rush videos are usually the raw materials for generating a compact news program [Over et al. 2008]. Ma *et al.* propose a user attention model for dynamic skimming [Ma et al. 2002]. Observing that users prefer content with high visual quality rather than video seen as "attractive" while containing distortions, Mei *et al*. design a video summarization system based on visual quality [Mei et al. 2007]. Shao *et al.* leverage both the video and music tracks to detect meaningful content, and create a summary for music videos [Shao et al. 2006]. In the sports domain, it is regarded that the recurrent highlights could be detected using domain-specific knowledge and used for sports video abstraction [Tjondronegoro et al. 2003] [Nitta et al. 2009]. According to different content adaptation requirements, the system developed in [Bescos et al. 2007] selects a variable number of frames from a compressed video sequence based on the level of motion activity.

Regardless of whether they employ static or dynamic forms, few existing video summarization techniques are able to losslessly preserve the semantic information. For example, none of those summaries can be used to reconstruct a video with the same duration as the original video. Moreover, most current summarization techniques highly depend on the semantic understanding of video content, which still remains a challenge in this community.

2.2 Video Compression

In general, the commonly adopted video compression standards are provided by ITU-T and ISO/IEC, although there exists rich research on video compression in the research community. For example, the MPEG-x family standards from ISO/IEC Moving Picture Experts Group (MPEG) are widely adopted for high quality professional programs [MPEG-2] [MPEG-4] , while H.26x standards from ITU-T Video Coding Experts Group (VCEG) are predominantly applied for low bitrate videos such as conference and user-generated videos [H264 2003]. H.264 (also known as MPEG-4 part 10) is the newest video compression standard jointly approved by MPEG and VCEG, which contains a rich family of "profiles," targeting specific classes of applications [H264 2003]. H.264 has achieved a significant improvement in rate-distortion efficiency. Although effective, even the most popular video codec (e.g., H.264) can only achieve the best compression rate of $\frac{1}{25} \sim \frac{1}{15}$ of its baseline profile with visually acceptable distortion. Therefore, a summarization scheme which can "compress" the video content at a much lower rate while preserving semantic information for further content analysis is desired.

2.3 Video Content Analysis

To evaluate the effectiveness of preserving visual fidelity and semantic information through NLSS summary, we performed two typical video content analysis tasks: *duplicate detection* and *concept detection*. The literature review will focus on these two domains. For more comprehensive literature surveys, please refer to [Smeulders et al. 2000] and [Lew et al. 2006].

2.3.1 *Video Duplicate Detection.*

Video duplicate detection aims to find copies of video, which are transformed by photometric or geometric transformations from the original video. Current methods usually extract a small piece of features (called signature) derived from the temporal and spatial statistics of the video content. Hampapur *et al*. propose a scheme to compare videos using global features [Hampapur et al. 2002]. Li *et al*. extract a compact binary signature involving color histograms from video to detect copies for monitoring commercials [Li et al. 2005]. Zhao *et al*. propose a series of methods using local features for near-duplicate detection [Zhao et al. 2007] [Zhao and Ngo 2009]. To balance the aspects of speed and accuracy, Wu *et al*. combine content and context information for real-time near-duplicate elimination [Wu et al. 2009]. Paisitkriangkrai *et al*. apply a scalable framework to clip-based near-duplicate video detection by using ordinal measures [Paisitkriangkrai et al. 2010].

2.3.2 *Video Concept Detection.*

The video annotation problem has received significant attention, since annotation helps bridge the semantic gap that results from querying using one mode (e.g., text) for returns of another mode (e.g., video). Video annotation algorithms can be said to be *supervised* or *unsupervised* based on whether it uses known training data [Mei et al. 2007] [Jiang et al. 2010]. An annotation method can also be described as a *computer vision* approach if it builds word-specific models from low-level visual features [Jiang et al. 2010], or a *data mining* approach if it mines correlations among annotations or propagates existing information [Moxley et al. 2010].

To build ontology for effective concept detection, researchers tried to provide standard video data in this community. For example, Large Scale Ontology for Multimedia (LSCOM) has defined 834 concepts [Naphade et al. 2006] from the TREC Video Retrieval Evaluation (TRECVID) [TRECVID]. The MediaMill team released 101 concept detectors [Snoek et al. 2006].

3. NEAR-LOSSLESS SEMANTIC SUMMARIZATION

In this section, we first give an overview of the proposed NLSS. Then, we describe the details of the summary generation.

3.1 Overview

Video is an information-intensive yet structured medium conveying time-evolving dynamics (i.e., camera and object motion). To design an effective summarization system that achieves the two commonly conflicting goals of near-lossless visual/semantical information and extremely low compression rate, the principles listed should be deliberated.

—To reduce redundancy as much as possible, the temporal structure of a video sequence such as shot and keyframe should be taken into consideration. Because the content in the entire video has large variations, a smaller segment with coherent content should be considered as the basic unit for summarization. Extracting metadata from this unit would significantly reduce the redundancy, while preserving semantic information within each unit. This is different from conventional video compression as the latter usually does not consider the semantic information of a video sequence.

—To achieve the least semantic information loss, all the segments in the video should be included to generate the metadata. We keep the dominant motion information (both camera and object motion) together with a set of representative images in the summary. This is different from conventional video summarization as the latter only selects a subset of segments from the original video.

—The summarization mechanism should be capable of maintaining not only low-level content descriptors (e.g., color, texture and shape in a single frame), but also high-level semantic information (e.g., the motion along successive frames), so that the summary could be used to reconstruct the original video and potential semantic-based applications.

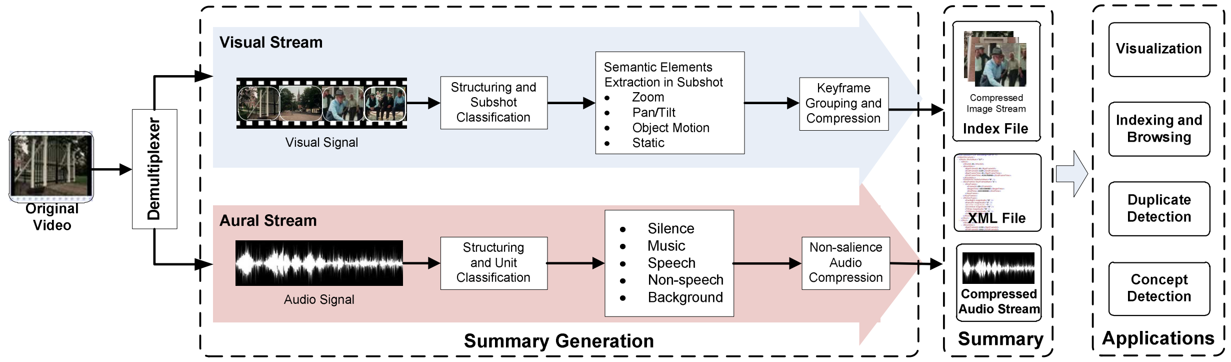


Fig. 1. Framework of NLSS. The system consists of three major components: 1) summary generation, 2) NLSS summary, and 3) applications.

Following the above principles, we propose the framework of NLSS in Figure 1. The original video stream is first de-multiplexed into two streams, i.e., visual and aural streams. Each stream will be handled separately to generate the corresponding component in the summary. The system then consists of three main components: 1) summary generation, 2) NLSS summary, and 3) video applications based on NLSS summary.

(1) **Summary generation**. The basic idea of NLSS is to segment the visual and aural streams into basic units, extract representative information from these units, and then employ efficient compression techniques to make this information compact. Regarding visual streams, as a $shot$ is usually too long and contains diverse contents, a sub segment of a shot, i.e., the $subshot$, is selected as the basic unit for metadata extraction [1]. The subshot is a sub segment within a shot, so that each shot can be divided into one or more consecutive subshots. In NLSS, subshot segmentation is equivalent to dominant camera motion detection, indicating that one subshot corresponds to one unique camera motion. The subshot is widely used as a basic semantic unit in many video applications from TRECVID [TRECVID]. One may argue that a subshot may contain more than one type of camera motion. However, we detect one unique dominant motion for each subshot for simplicity. As a result, the visual track is decomposed into a series of subshots by a motion-based method [Kim et al. 2000], where each subshot is further classified into one of the four categories on the basis of dominant motion (i.e., *zoom*, *translation* (*pan/tilt*), *object motion*, and *static*). An appropriate number of frames or synthesized mosaic images are extracted from each subshot (details will be given in the next subsection) [Mei and Hua 2008] [2]. To further reduce the storage, all images (including keyframes and mosaic images) are grouped according to color similarity and compressed by H.264. Regarding the aural track, we propose an "audio content analysis compression" (ACAC) scheme to first segment the audio track into units at every 0.50 second [Lu et al. 2003], and then classify each unit into one of the five categories—i.e., *silence*, *music*, *background*, *pure-speech* and *non-pure-speech*. The non-silence units (i.e., *music*, *background*, *pure-speech* and *non-pure-speech*) are compressed with a very-low bitrate compression standard (i.e., AMR at 6.7 kbps) to keep the most semantic fidelity [AMR 2002].

[1] Shot is an uninterrupted temporal segment in a video, recorded by a single camera.
[2] Mosaic is a synthesized static image by stitching successive video frames in a large canvas [Irani and Anandan 1998].

(2) **NLSS Summary**. The NLSS summary consists of three components—an XML file describing the temporal structure and motion information, as well as the compressed image stream and the audio stream compressed by ACAC.

(3) **Applications**. The summary generated from the NLSS can be applied to a wide variety of video-based applications. First, we can "reconstruct" a video based on the summary, which is of the same length as the original video. The NLSS summary is informative to index videos and present search results in any video search engine. For example, the extracted keyframes and mosaic images, together with the aural track, can be used to provide a preview of the original video [Bing]. Moreover, the summary can be applied for many video analysis tasks, e.g., video duplicate detection and concept detection, as we can extract meaningful features from the NLSS summary.

3.2 Summary Generation

Compared to our previous work [Tang et al. 2009], we propose a more comprehensive framework to generate the NLSS summary for a given video, in which by unifying the selected keyframes and mosaic images, we achieve a higher compression rate. We first present how to decompose the original video into subshots and classify subshots into four dominant motions. Then, we perform different summarization methods for different types of subshots. Finally, we discuss how to further reduce the metadata storage by frame grouping and compression, as well as audio segmentation, classification, and compression via a very low bitrate audio codec.

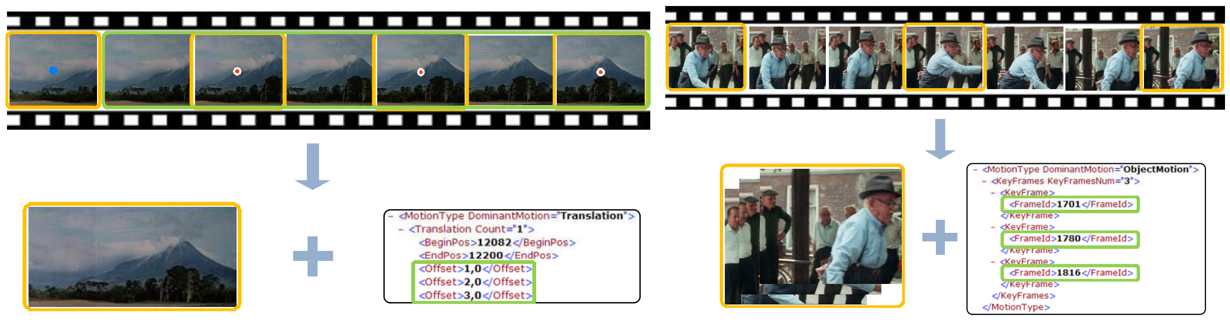
3.2.1 *Subshot Detection and Classification.* In [Bescos et al. 2007], the basic unit for video summarization is subshot. We adopt similar mechanism in our system. The video track is first segmented into a series of shots based on a color-based algorithm [Zhang et al. 1993]. Each shot is then decomposed into one or more subshots by a motion threshold-based approach [Kim et al. 2000], where each subshot is further classified into one of the five categories according to the dominant camera motion, i.e., *static*, *translation* (including *pan* and *tilt*), *rotation*, *zoom*, and *object motion*. The algorithm proposed by Konrad *et al.* [Konrad and Dufaux 1998] is employed for estimating the following affine model parameters between two consecutive frames due to its robustness.

$$\begin{cases} v_x = a_0 + a_1 x + a_2 y \\ v_y = a_3 + a_4 x + a_5 y \end{cases} \tag{1}$$

where a_i $(i = 0, \ldots, 5)$ denote the motion parameters and (v_x, v_y) the flow vector at pixel (x, y). The motion parameters in the equation (1) can be represented by a set of more meaningful parameters in [Bouthemy et al. 1999] to illustrate the dominant motion in each subshot as follows:

$$\begin{cases} b_{pan} = a_0,\ b_{tilt} = a_3 \\ b_{zoom} = \frac{1}{2}(a_1 + a_5),\ b_{rot} = \frac{1}{2}(a_4 - a_2) \\ b_{hyp} = \frac{1}{2}(|a_1 - a_5| + |a_2 + a_4|) \\ b_{err} = \frac{1}{M \times N} \sum_{j=1}^{N} \sum_{i=1}^{M} |p(i,j) - p'(i,j)| \end{cases} \tag{2}$$

where $p(i, j)$ and $p^{'}(i, j)$ denote the value of pixel (i, j) in the original and the wrapped frame reconstructed by equation (1), respectively. M and N denote the width and height. b_{pan}, b_{tilt}, b_{zoom}, b_{rot}, and b_{hyp} denote the actual camera motion parameters for *pan*, *tilt*, *zoom*, *rotation*, and *object motion*, while b_{err} denotes the estimation *error* of this affine model. Based on the parameters in equation (2), a qualitative threshold-based method can be used to sequentially identify each of the camera motion categories in the order as follows: *zoom* $\rightarrow$ *rotation* $\rightarrow$ *pan* $\rightarrow$ *tilt* $\rightarrow$ *object motion* $\rightarrow$ *static* [Kim et al. 2000]. Specifically, we first classify the camera motion between two consecutive frames by adaptive


Summary: Mosaic + XML File (trans. info)
Summary: Keyframes + XML file (frame id)

```
- <MotionType DominantMotion="Translation">
  - <Translation Count="1">
      <BeginPos>12082</BeginPos>
      <EndPos>12200</EndPos>
      <Offset>1,0</Offset>
      <Offset>2,0</Offset>
      <Offset>3,0</Offset>
```

```
- <MotionType DominantMotion="ObjectMotion">
  - <KeyFrames KeyFramesNum="3">
    - <KeyFrame>
        <FrameId>1701</FrameId>
      </KeyFrame>
    - <KeyFrame>
        <FrameId>1780</FrameId>
      </KeyFrame>
    - <KeyFrame>
        <FrameId>1816</FrameId>
      </KeyFrame>
    </KeyFrames>
  </MotionType>
```

(a) *translation* subshot (b) *object motion* subshot

Fig. 2. Summary generation of *translation* and *object motion* subshots.

thresholding method in the above order. For example, the motion is regarded *zoom* if b_{zoom} is above a threshold; otherwise, the motion parameter b_{rot} will be compared with another threshold. Such process is performed sequentially in the above order. At a result, the motion between each frame pair will be classified into one of the six categories. Then, we use a temporal sliding window with the duration of 0.5 sec to smooth the frame-level categories. In the end, we can obtain the subshot boundaries based on the frame-level categories.

In the NLSS, we treat *pan* and *tilt* in a single category of *translation*, as to be explained later, the mechanism for extracting metadata from these kinds of subshots is identical. As *rotation* motion seldom occurs, we take it as a special case and regard it as the *object motion*. As a result, each subshot belongs to one of the four classes, i.e., *zoom, translation* (i.e., *pan*/*tilt*), *object motion*, and *static*.

3.2.2 *Subshot Summarization.* For the sake of simplicity, we define the following terms: a video V consisting of N subshots S_i is denoted by $V = \{S_i\}_{i=1}^{N}$; similarly, a subshot S_i can be represented by a set of successive frames $S_i = \{F_{i,j}\}_{j=1}^{N_i}$ or keyframes $S_i = \{KF_{i,k}\}_{k=1}^{M_i}$. After classifying each subshot into one predefined category, we summarize each type of subshot independently as follows:

—***Zoom subshot***. Each *zoom* subshot is labeled as *zoom-in* or *zoom-out* based on the parameter b_{zoom}, which indicates the magnitude and direction of the *zoom*. As a *zoom-out* presents the reverse process of a *zoom-in*, only *zoom-in* is taken as an example. In the *zoom-in* subshot, the consecutive frames describe the gradual changes from a distant view to a close-up view. Therefore, the first frame is sufficient to represent the overview content in a *zoom-in* subshot [3]. Thus, we design a summarization scheme focusing on keyframe selection and motion metadata generation for *zoom* subshot. In addition, the information of the dominant camera motion—the camera focus (the center in the frame) and the accumulated zoom-in factors of all frames with respect to the keyframe are recorded with the XML format. For each frame in a subshot, we calculate the center of the wrapped image and the accumulated zoom factor in the summary.

—***Translation subshot***. Compared to a *zoom* subshot, a keyframe is far from enough to describe the whole story in the *translation* subshot. To describe the wide view of the subshot in a compact form, we adopt a mosaicing technique for summarizing the panoramic view of a subshot by stitching the frames in a *translation* subshot [Irani and Anandan 1998]. Image mosaicing usually involves two steps: image alignment and wrapping. As proposed in [Tang et al. 2009], before generating a mosaic image for each subshot, we first segment a subshot into units using the leaky bucket algorithm to

[3] Note that for the *zoom-out* subshot, the last frame is selected to represent the overview.

avoid distortion [Truong and Venkatesh 2007] [Kim and Hwang 2002]. When generating the summary, a mosaic is generated from the frames in the unit by using the temporal-centered frame as a reference. Moreover, we perform a $\frac{1}{2}$ sampling in the successive frames to reduce computational costs and eliminate the eclipsing artifacts. Meanwhile, the gradual change of the camera focus is recorded. Figure 2 (a) shows this process.

—***Object motion subshot***. In the *object motion* subshot, there are significant content variance. We propose to summarize the *object motion* subshot by extracting a set of most representative frames. To select the most representative frames (more than one as there is much content diversity within each *object motion* subshot), a *object motion* subshot is further decomposed into units by thresholding the accumulated parameter of b_{err} along the successive frames, which represents the content diversity in terms of the motion prediction error. If the accumulated b_{err} exceeds a predefined threshold T_b, then there is a unit boundary claimed right after the current frame. This is also called "leaky bucket" algorithm in [Truong and Venkatesh 2007] [Kim and Hwang 2002]. Moreover, we also employ another threshold T_f to avoid successive selection in a highly active subshot. That is, for each selected keyframe $KF_{i,k}$ $(k = 0, \ldots, M_i)$, it satisfies

$$I(KF_{i,k}) - I(KF_{i,k-1}) \geqslant \max(T_b, T_f), \tag{3}$$

where $I(KF_{i,k})$ is the frame index of keyframe $KF_{i,k}$. For each keyframe, we record the timestamp and frame index into the XML file. Figure 2 (b) shows this process.

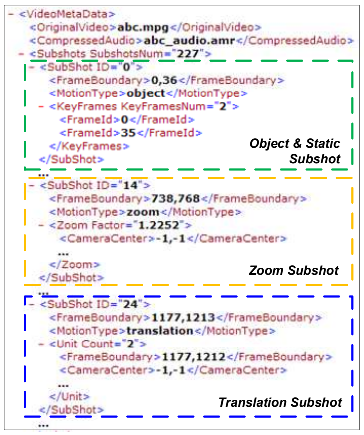
—***Static subshot***. In a *static* subshot, the objects are static and the background merely changes. Keeping in mind the need for simplicity and efficiency, we can directly set any frame as the representative one. Here, we select the frame in the middle of the subshot as the keyframe, and record the timestamp and frame index of the selected frame into the XML file.

3.2.3 *Near-Lossless Semantic Summary*. The near-lossless semantic summary of a given video consists of three components—an XML file describing the temporal and motion information, as well as the compressed image stream and audio stream extracted from the original video. Figure 3 (a) shows the XML file format, while (b) shows the process for generating these three components.

—***XML file***. The XML file consists of successive sections of the time and motion information for each subshot in sequence marked by timestamp. Figure 3 (a) shows a typical description of XML file. Subshots #0, #14, and #24 illustrate the time and motion information for *object motion* (also for *static*), *zoom* and *translation* subshots, respectively.

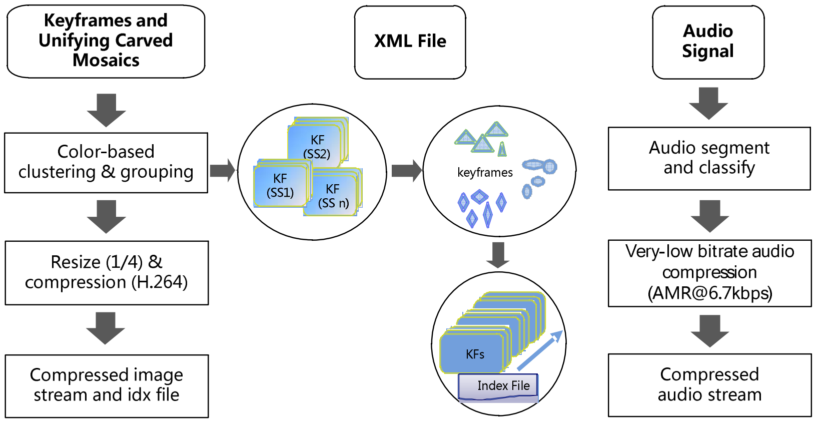
—***Compressed image stream***. There are two types of image data in the metadata—keyframes and mosaic images. To reduce the size of these metadata, we employed JPEG standard with $Quality = 95$ to compress these images [ISO/IEC 1991], and resize them to $\frac{1}{2}$ of the original resolution. As there is considerable redundancy in images of similar scenes, we employ a clustering-based grouping and compression scheme to eliminate the redundancy. In contrast to our previous work [Tang et al. 2009] which does not consider compressing mosaic images, we further make the mosaic image compression unified with the scope of keyframes by carving the mosaic image into grids. The carved images from the mosaic can be treated as keyframes using the same grouping and compression scheme in Figure 3 (b). The first keyframe of each subshot is first selected as the representative keyframes. Then, K-means clustering is performed on these representative keyframes using color motion features with N_c clusters. All the keyframes are aligned with their representative keyframes in the same cluster. We employ H.264 baseline profile to further compress the keyframe sequence into a compressed image stream [H264 2003], together with the index file corresponding to these keyframes.

—***Compressed low bitrate audio track***. In contrast to previous work [Tang et al. 2009] which simply compressed the audio track with a very-low bitrate compression standard AMR at 6.7 kbps [AMR



```
- <VideoMetaData>
    <OriginalVideo>abc.mpg</OriginalVideo>
    <CompressedAudio>abc_audio.amr</CompressedAudio>
  - <Subshots SubshotsNum="227">
    - <SubShot ID="0">
        <FrameBoundary>0,36</FrameBoundary>
        <MotionType>object</MotionType>
      - <KeyFrames KeyFramesNum="2">
          <FrameId>0</FrameId>
          <FrameId>35</FrameId>
        </KeyFrames>                                  Object & Static
      </SubShot>                                              Subshot
      ...
    - <SubShot ID="14">
        <FrameBoundary>738,768</FrameBoundary>
        <MotionType>zoom</MotionType>
      - <Zoom Factor="1.2252">
          <CameraCenter>-1,-1</CameraCenter>
          ...
        </Zoom>                                       Zoom Subshot
      </SubShot>
      ...
    - <SubShot ID="24">
        <FrameBoundary>1177,1213</FrameBoundary>
        <MotionType>translation</MotionType>
      - <Unit Count="2">
          <FrameBoundary>1177,1212</FrameBoundary>
          <CameraCenter>-1,-1</CameraCenter>
          ...
        </Unit>                                Translation Subshot
      </SubShot>
      ...
```

(a) XML file format


Keyframes and Unifying Carved Mosaics
Color-based clustering & grouping
Resize (1/4) & compression (H.264)
Compressed image stream and idx file
KF (SS2)
KF (SS1)
KF (SS n)
XML File
keyframes
KFs
Index File
Audio Signal
Audio segment and classify
Very-low bitrate audio compression (AMR@6.7kbps)
Compressed audio stream

(b) NLSS summary

Fig. 3. Generation of near-lossless semantic summary. Note that we resize both the width and height of the frame to $\frac{1}{2}$, which corresponds to the overall resizing factor of $\frac{1}{4}$.

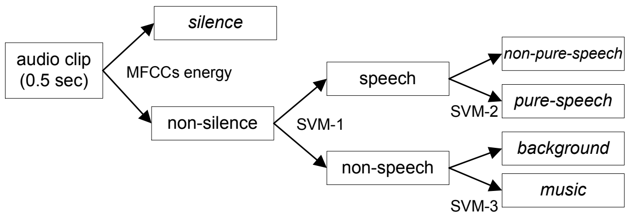


Fig. 4. The binary tree structure for multi-class classification of audio track. A audio clip with 0.5 second is classified into five categories: *silence*, *music*, *background*, *pure-speech* and *non-pure-speech*.

2002], we propose an "audio content analysis compression" (ACAC) scheme to further reduce the size of the aural summary in the NLSS. By leveraging audio content analysis [Lu et al. 2003], we partition the audio track into units (at the interval of 0.5 second) and classify each unit into five categories—i.e., *silence*, *music*, *background*, *pure-speech* and *non-pure-speech*. The *pure-speech* unit includes speech over music and noise. Specifically, the audio track is divided into non-overlapping 0.5 second sub-clips. These sub-clips are first classified into *silence* and non-silence clip, depending on the energy information derived from the mel-frequency cepstral coefficients (MFCCs). The non-silence clips are further classified into two categories based on the MFCCs and the kernel Support Vector Machines (SVMs), i.e., speech and non-speech. Then, non-speech clips are further classified into *background* and *music*, while speech unit are classified into *non-pure-speech* and *pure-speech*. The average accuracy for the classification is about 95% [Lu et al. 2003]. Figure 4 shows the classification scheme for audio tracks, where SVM-1, SVM-2, and SVM-3 are the classifiers trained by SVMs. As there is no informative content within the *silence* unit, we compress the non-silence units (i.e., *music*, *background*, *speech*, and *non-speech* units) with a very-low bitrate compression standard (i.e., AMR [AMR 2002] at 6.7 kbps) to keep the most semantic fidelity. This leads to the significant reducing of the audio summary. Other possible solutions to audio summarization include the March1 technique [Covell et al. 1998], which is designed for nonuniform time compression of speech signal. However, the high-level semantic information, such as the five categories mentioned above, may not be obtained in March1.

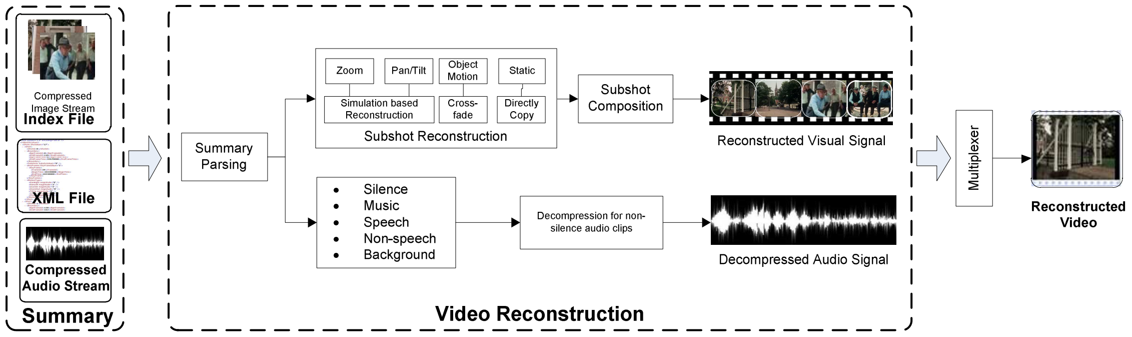


Fig. 5. Video visualization from NLSS summary.

4. APPLICATIONS

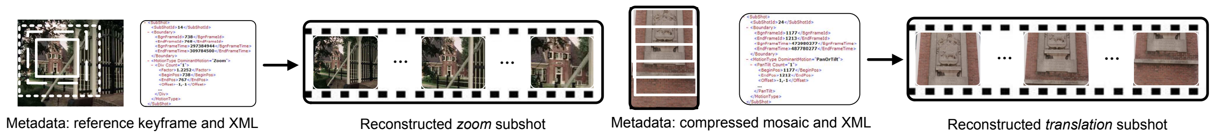
By “near-lossless semantic summarization,” we refer to the concept that the NLSS summary can keep both visual and semantic fidelities for a wide variety of video-oriented applications. In this section, we will show a series of applications of NLSS, including video visualization (or reconstruction), indexing and presentation for a video search engine, as well as duplicate detection and concept detection for video content analysis.

4.1 Video Visualization

Video visualization indicates the reconstruction of a video with the same duration as the original video based on the NLSS summary, while trying to maintain the visual fidelity as much as possible. The reconstruction consists of three steps: metadata parsing, subshot reconstruction, and video generation. Since the subshot is the basic unit in NLSS, subshot reconstruction is an important component for visualization.

Figure 5 shows the process of reconstructing a video based on NLSS summary. First, we parse the XML file with the subshot index, as well as extract all the keyframes and mosaic images from the H.264 compressed stream. Then, we describe different mechanisms to reconstruct each subshot frame by frame. Similar to Section 3.2.2, we define the following notations: a reconstructed video V' consists of N reconstructed subshots S'_i, denoted by $V' = \{S'_i\}_{i=1}^{N}$. A subshot S'_i can be represented as a series of reconstructed frames $S'_i = \{F'_{i,j}\}_{j=1}^{N_i}$.

—**Zoom subshot**. To reconstruct a *zoom* subshot, we simulate the camera motion on the selected keyframes. There are three steps in motion simulation of a *zoom* subshot, i.e., zoom factor simulation, camera focus trajectory smoothing, and frame reconstruction. We take *zoom-in* as an example. In zoom factor simulation, we simulate the subshot as a zoom-in procedure with the constant zooming speed, where the zoom factor between successive frames is a constant $\sqrt[N_i-1]{Z^{acc}(S_i)}$ in one subshot. To reconstruct the j-th frame in the subshot S'_i, we calculate the zoom factor of the j-th frame $Z(F'_{i,j})$ with respect to the first keyframe by $Z(F'_{i,j}) = \left(\sqrt[N_i-1]{Z^{acc}(S_i)}\right)^{j-1}$, where N_i $(i = 2, \ldots, N_i)$ is the number of frames in the subshot. Moreover, the camera focus of each frame with respect to the keyframe is calculated from the wrapping process. A 5-point Gaussian smoothing with the template template $[\frac{1}{16}, \frac{4}{16}, \frac{6}{16}, \frac{4}{16}, \frac{1}{16}]$ is employed to eliminate the jitter for camera focus trajectory smoothing. When reconstructing the j-th frame, as shown in Figure 6 (a), we first shift the center of the keyframe with the smoothed camera focus, and then resize the keyframe with the zoom factor $F_{i,j}$. Finally, we carve the breadth of the original frame from the resized keyframe with respect to the camera focus offset. Note that for the subshot with a large zoom factor (e.g., tight zoom subshots


Metadata: reference keyframe and XML
Reconstructed *zoom* subshot
Metadata: compressed mosaic and XML
Reconstructed *translation* subshot

(a) reconstruction of *zoom* subshot (b) reconstruction of *translation* subshot

Fig. 6. Subshot reconstruction.

in a nature documentary video), we may include both the first and last frames to make use of both overview and close-up view, since the visual quality of frames might be poor.

—**Translation subshot**. As we mentioned in Section 3.2.2, a translation subshot consists of one or more units. For a subshot, we sequentially reconstruct the units by simulating the camera focus trajectory along the mosaic. There are two steps in the motion simulation of the translation subshot—camera focus trajectory smoothing and frame reconstruction. The generation of camera focus is identical in the *zoom* and *translation* subshots. When reconstructing the j-th frame in the subshot, we simulate the smoothed trajectory of camera focus along the mosaic. Then, we carve the mosaic with the same width as the original frame as the reconstructed frame. Figure 6 (b) shows the reconstruction of a *translation* subshot.

—**Object motion subshot**. To reconstruct the *object motion* subshot, we simulate the object motion with the gradual transition between selected keyframes. Concerning reconstruction efficiency and visual pleasure, we employ a fixed-length cross-fade transition between each keyframe to simulate the object motion. By modifying the fade-in and fade-out expression in [Fernando et al. 1999], we define the cross-fade transition to reconstruct the j-th frame $F'_{i,j}$ in the subshot S'_i by

$$F_{i,j} = \begin{cases} KF_{i,k}, & 0 \leqslant j \leqslant l_i \\ \left(1 - \frac{j-l_i}{L}\right) \times KF_{i,k} + \left(\frac{j-l_i}{L}\right) \times KF_{i,k+1}, & l_i \leqslant j \leqslant l_i + L \\ KF_{i,k+1}, & l_i + L \leqslant j \leqslant 2l_i + L \end{cases} \tag{4}$$

where $2l_i + L = N_i$, and the length L of the cross-fade procedure is set as $0.5 \times fps$ frames here.

—**Static subshot** For the *static* subshot, we select one of the frames in the image sequence to represent the whole subshot. As a result, we can reconstruct all the frames in the subshot by copying from the keyframe selected.

By visualizing each subshot, all the frames in each subshot are reconstructed using the metadata. Finally, all the reconstructed frames are resized to the original scale with the factor $\frac{1}{2}$. Together with the compressed audio track, we then integrate the reconstructed frames sequentially into a video with the same duration as the original video.

4.2 Video Indexing and Browsing

The backend of a video search engine crawls and indexes video data on the basis of the visual features and the associated text. Most existing video search engines do not keep the original video on their servers, or only store a very short summary for each crawled video [Bing] [Google]. The proposed NLSS can be easily integrated into the backend in that: 1) the extremely small piece of NLSS video summary can be stored in the server with a limited consumption of storage; 2) the existing indexing methods can be performed on the metadata in the video summary, so that the relevance can be improved via the rich metadata; 3) based on the video summary, the backend can perform many tasks which cannot be accomplished currently, e.g., video search re-ranking and so on. For example, there are many works on video search and search reranking based on the metadata associated with the subshots in the research community [TRECVID] [Liu et al. 2009] [Hsu et al. 2007].

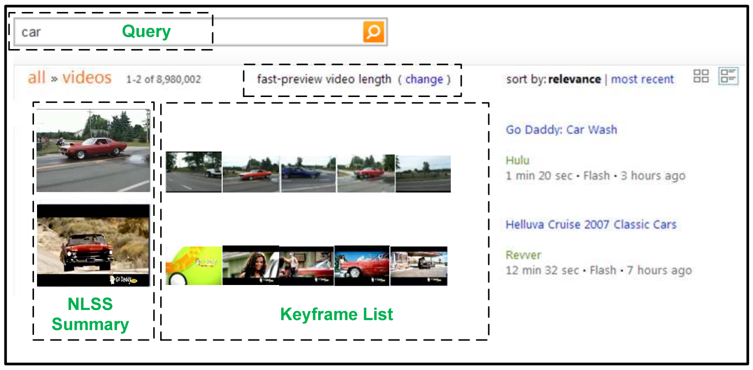

car
Query
all » videos 1-2 of 8,980,002
fast-preview video length (change)
sort by: relevance | most recent
Go Daddy: Car Wash
Hulu
1 min 20 sec • Flash • 3 hours ago
Helluva Cruise 2007 Classic Cars
Revver
12 min 32 sec • Flash • 7 hours ago
NLSS Summary
Keyframe List

Fig. 7. An example of video presentation for a commercial video search engine. Users can browse the entire video content and also have a quick preview of the representative keyframes or any highlighted subshots based on NLSS summary.

The frontend of a video search engine returns and presents the search results according to the queries. On the one hand, as there are very little visual information about the original videos in the backend of a search engine, typically each searched video is represented by a single frame (usually the first frame in a video) [Google], or a very short clip (called dynamic thumbnail) [Bing]. On the other hand, when users click one of the search results, they will be redirected to the host sites as the search engines do not keep the original videos partially due to the limited storage ability. Moreover, in some cases, the link to the original video may be not valid. In a word, there is a gap between the user need of efficient browsing and the limited presentation capabilities of current video sites. The NLSS can be applied in the frontend to enrich video presentation in that: 1) users do not need to be redirected for browsing the entire video, as the original video can be fully reconstructed (without semantic loss) from the summary stored in the backend; 2) users can have a quick preview of the highlight of each video, as the summary can be easily and efficiently used to generate a scalable video highlight; and 3) users can have a glance of the main content by just browsing a list of the most representative keyframes which can be easily extracted from the summary. Figure 7 shows an example of the user interface of the frontend with NLSS based video presentation.

4.3 Video Duplicate Detection

To evaluate the preserving of semantic fidelity in NLSS, we apply the same video duplicate detection scheme on the collection of original videos and their corresponding NLSS summary. In typical video duplicate detection, a seed video and a set of candidate videos are simultaneously processed by a feature extraction and matching scheme. Here, we adopt an effective and simple scheme as an example. First, the keyframes of the video are detected. The Scale Invariant Feature Transform (SIFT) features are extracted from each keyframe [Lowe 2004]. Then, pairwise matching is performed between the seed and each candidate keyframe based on the local features. Finally, the number of the matched local feature pairs is taken as the criterion for duplication detection. Note that the difference between NLSS and the original video is that we reconstruct the keyframes from the NLSS summary, while we use the source keyframes in the original video. Thus, we can perform duplicate detection on NLSS without the original video content. In the evaluation, we will demonstrate the performance of duplicate detection on both NLSS summary and original video content. Note that although there exists rich research on (near-) duplicate detection, we adopt the most straightforward approach in order to validate that NLSS summary can preserve the visual fidelity of the original video.

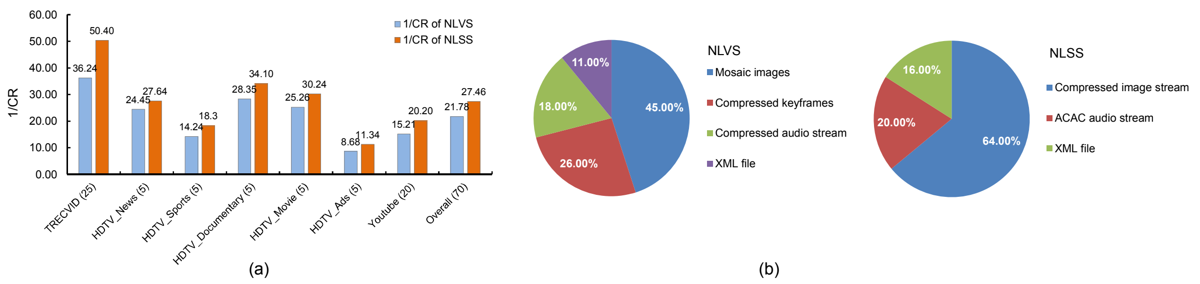


Fig. 8. Evaluation of storage consumption: (a) storage consumption (the number along x-axis indicates the number of videos for each category); (b) proportion of components in NLVS [Tang et al. 2009] and NLSS.

4.4 Video Concept Detection

Concept detection is a major task in TRECVID benchmarks [TRECVID]. To validate that NLSS summary is able to preserve the semantic fidelity of the original video, we evaluate the performance of video concept detection based on the summary and compare it with the original video content. We adopt a fundamental framework to perform video concept detection on the original videos in the task of TRECVID 2007 high level feature extraction [TRECVID] and their NLSS summaries, respectively. First, keyframes are extracted from the original video and NLSS summaries, using the same strategies mentioned in Section 4.1. Then, we extract the widely adopted 5×5 color moment feature from each keyframe [Mei et al. 2007]. Finally, we split the keyframes as training, testing, and evaluation sets, and evaluate the 36 concepts based on 36 trained SVM-based concept detectors based on TRECVID 2007 training data. The performance of these 36 concept detectors could be found in [Mei et al. 2007]. Next, we will demonstrate the performance of concept detection on both the NLSS summaries and original videos.

5. EVALUATION

The proposed NLSS is evaluated from the following aspects: storage consumption, computational costs, user impression of video reconstruction, and the applications of video duplicate detection and concept detection.

5.1 Experimental Settings

To evaluate the NLSS performance on storage consumption, we use the same video collection mentioned in our previous work to compare the performance between NLSS and our previous NLVS [Tang et al. 2009]. There are 70 representative videos (2,162 minutes with 14,322 subshots in total), including 25 videos from TRECVID 2007 [TRECVID] (TRECVID videos), 5 HDTV news programs (HDTV_News), 5 HDTV sports programs (HDTV_Sports), 5 HDTV documentary videos (HDTV_Documentary), 5 HDTV movies (HDTV_Movie), 5 HDTV advertisement videos (HDTV_Ads), and 20 online videos collected from Youtube (Youtube). The 25 videos are randomly selected from the development set of TRECVID 2007, which includes 399 news video in total [Mei et al. 2007], while the 20 online videos are also randomly selected from the videos used for online advertising [Mei et al. 2009]. To generate the NLSS summary, we set the thresholds as $T_b = 800$ and $T_f = 10$.

To evaluate the performance of duplicate detection based on NLSS summary, we use the subset of web video dataset CC_WEB_VIDEO [CC_ WEB_ VIDEO]. There are 2,493 videos in 24 categories, where 748 videos are duplicates. To evaluate the performance of concept detection based on NLSS summary, videos for developing TRECVID 2007 high level feature extraction task are adopted, includ-

Table I. Computational costs of NLSS (sec).

	$1\ T$	$\frac{1}{2}\ T$
$1\ S$	16,456	7,561
$\frac{1}{2}\ S$	4,638	2,129

ing 110 videos [TRECVID]. Moreover, 21,532 keyframes are extracted from the videos to detect 36 concepts defined in TRECVID 2007.

5.2 Evaluation of Storage Consumption

For NLSS, storage consumption is the size of video summary, including the XML file, the compressed image and mosaic streams, and the compressed audio stream. We compare the storage consumption between NLSS and our previous NLVS [Tang et al. 2009], by measuring their improvements in storage to a state-of-art video compression standard in terms of the compression rate (*CR*). We define *CR* as the ratio of the bitrate of NLSS or NLVS against that of video compression standard (VCS). Since H.264 has become the most popular compression technique on the Internet, its baseline profile is adopted to compress the TVS and HDTV videos, while MP3 is adopted to compress the audio track at 128 kbps. Then, we compare NLSS and NLVS with the H.264 baseline profile ("H.264.baseline") in terms of storage. For online videos, we use FLV for video compression.

The results of all 70 selected videos are shown in Figure 8 (a). We can see that in NLSS, the improvement in storage consumption is significant when compared with NLVS. We also look into the components of NLSS and NLVS to see why the above improvement can be achieved. We can see that different video genres depict different CR scores. For HDTV_Ads and HDTV_Sports, the CR scores are not high as those for HDTV_Documentary and TRECVID. This is due to the composition style and nature of advertisement and sport videos—there are always strong motion within these videos. Therefore, the *object motion* subshots will dominate, which makes the compression rate not that high. Figure 8 (b) shows the change of the proportion of each component from NLVS to NLSS. We can see that the uncompressed mosaic takes over the major proportion of the summary of NLVS. However in NLSS, we unify the mosaic images with the scope of the keyframes by carving the mosaic into image grids, and then compressed them. This technique significantly reduces the proportion of the mosaic in the summary, thus improving the storage consumption of NLSS.

Figure 9 shows some frames reconstructed by NLSS, as well as the corresponding frames of original video and H.264.rc. Although there exists displacement in the *zoom* subshot and fold-over in the *object motion* subshot, frames reconstructed by NLSS achieve a comparable visual quality to the original frames. Through the same bit-rate as NLSS, frames of H.264.rc contain a large amount of blocking artifacts, mainly due to the extremely low bit-rate in the H.264 rate control setting.

5.3 Evaluation of Computational Costs

To demonstrate the computation costs of generating a NLSS summary, we investigate two main factors—video resolution and temporal sample rate. By video resolution, we compare the time consumed in NLSS generation of the same video in two different spatial sampling rates (i.e., 1 and $\frac{1}{2}$, where $\frac{1}{2}$ means the frame of video is downsampled by the ratio of $\frac{1}{2}$). By temporal sample rate, we compare the time consumed in two temporal sampling rates, (i.e., 1 and $\frac{1}{2}$ of the original rate). We take a video with a duration of $2,988$ seconds at a resolution of 352×288 and FPS of 25 as an example. We adopt all these settings in a desktop with 1.87 GHz CPU and 2 GB RAM.

Table I lists the variation of the two mentioned factors in different settings—"$1\ T$" and "$1\ S$" indicates keeping the same video resolution and sampling rate of the original video, while "$\frac{1}{2}T$" and "$\frac{1}{2}S$" indicates the video is downsampled in temporal or spatial by the factor $\frac{1}{2}$. We can see that NLSS achieved

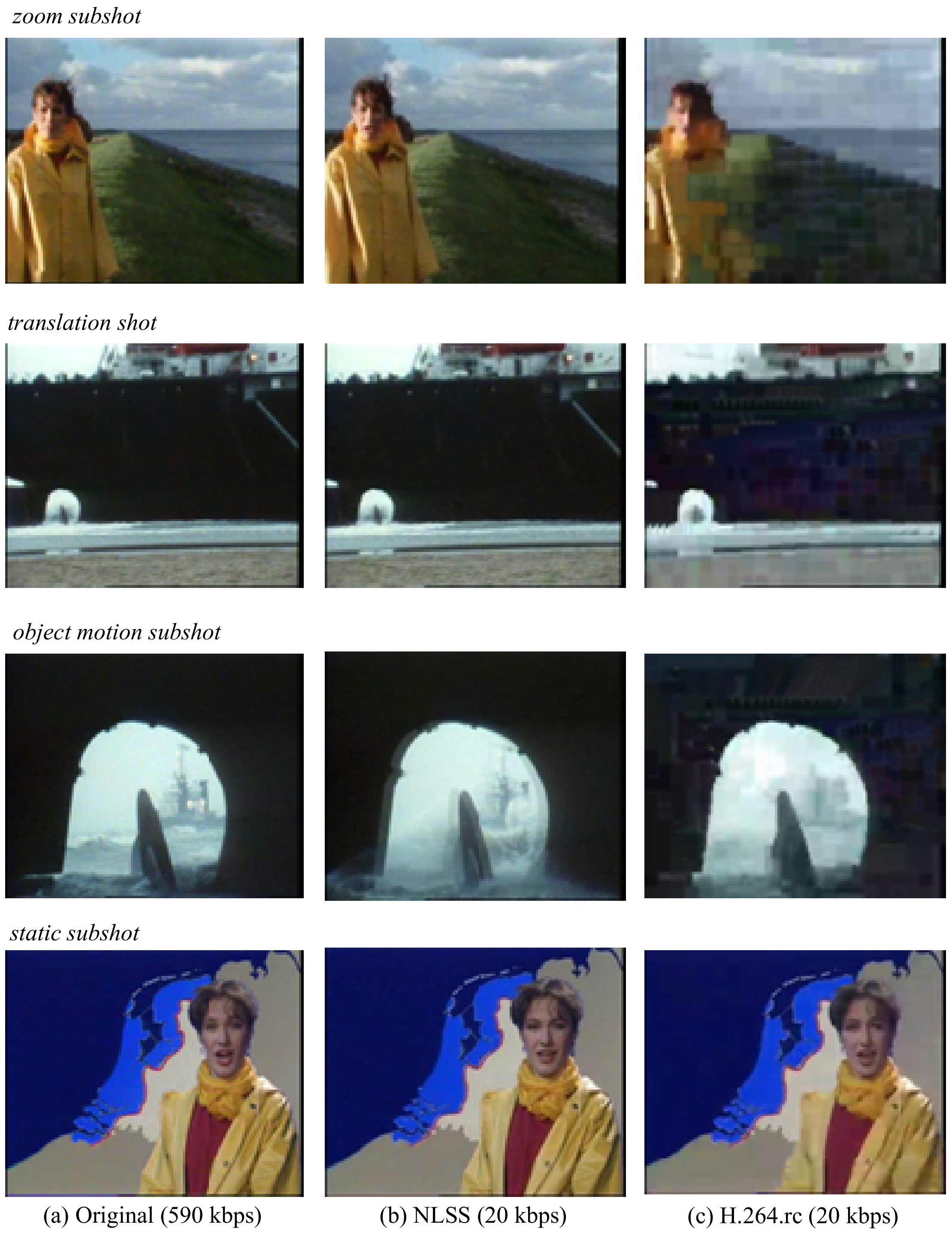


Fig. 9. Example of the frames from the original video (Original), NLSS, and H.264.rc for different types of subshots.

the least computation costs at "$\frac{1}{2}T$" and "$\frac{1}{2}S$", which cost only $\frac{1}{8}$ of the time of the original resolution and duration. Further, we can also discover that spatial downsampling decreased more computation costs than temporal downsampling. The reason for the two factors is that, there is a strong relationship between these two factors and motion estimation, which is the main computation consumer in our proposed NLSS framework. Therefore, to improve the efficiency of motion estimation is a main target of future NLSS work. Fortunately, for many compressed video streams, motion vectors between frames are pre-computed and embedded in the video stream, which decreases the consumption when generating a NLSS summary. Note that the computation costs of NLSS are close to those of NLVS, as the process of carving mosaics and compressing them is rapid and can be ignored when compared to other computation costs in the summary generation.

5.4 Evaluation of Video Visualization

As evaluating video summarization is highly subjective, we have conducted a user study to evaluate user satisfaction on the visual quality of the NLSS-reconstructed videos. We invite 30 volunteers including 15 males and 15 females with diverse backgrounds to participate the user studies over the 70 videos. These subjects like to browse online videos. Similar with the settings in [Tang et al. 2009], we generate seven different forms of video summaries for each video according to the following compression and summarization techniques:

(1) Original video (Original). We directly used the original videos, without any codec conversion.
(2) H.264 compressed video with rate control (H264). We compressed the videos by H.264 to make the compressed signals be with the same storage consumption with NLSS (i.e., the same bit rate setting and audio track setting).
(3) Static Summarization (Static). For each subshot, we selected one keyframe and arranged them on a one-dimensional storyboard. This is similar with many existing static video summarization techniques.
(4) Dynamic Summarization 1 (Attention). We used the attention-based dynamic skimming technique proposed in [Ma et al. 2002] to summarize the original videos, so that the summaries are with the same filesize as those of NLSS [4].
(5) Dynamic Summarization 2 (CMU). We adopted the video skimming scheme proposed by the Carnegie Mellon University (CMU) in TRECVID 2007 [Hauptmann et al. 2007] to summarize the videos. The sample rate is set as the same as (4).
(6) NLVS reconstructed video (NLVS). We summarized the videos using NLVS and then generated reconstruction video with the same duration [Tang et al. 2009].
(7) NLSS reconstructed video (NLSS). We summarized the videos using NLVS and then generated reconstruction video with the same duration.

In total, we have 490 (70×7) forms of video summaries (including the original, reconstructed, and generated summary) for evaluation. Each subject was asked to randomly select 16–17 summaries and answer a questionnaire for each summary. The subjects are not aware of the compression technique of a given summary. The questions include:

—Content comprehension. This is designed to see if the summary can maintain the major content of the original video. We design five tasks for each summary, e.g., "who is the baby's father," "how

[4] Sample rate is the skimming ratio in video skimming scheme. For example, if we perform video skimming on a video with duration of 10 minutes, and get a video summary with duration of 1 minute, then the sample rate is 10%.

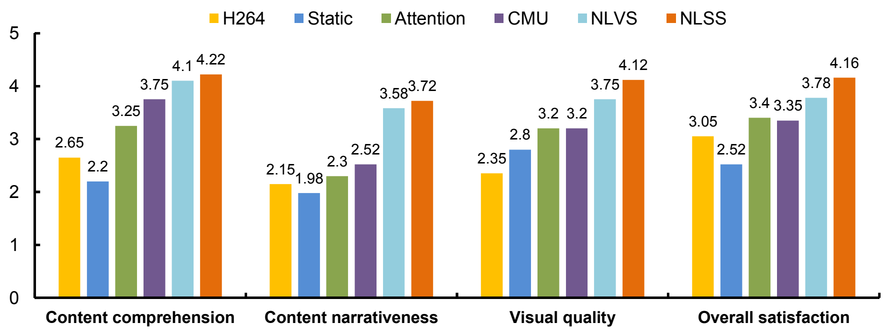


Fig. 10. Results of user studies.

many people appearing in this video," etc. The content comprehension score is added "1" point if current user comes out with the correct answer. We then use the average scores as the final content comprehension score.

—Content narrativeness. The subjects are asked to write down what they think about the storyboard of the browsed video summary. Then, a score in the scale of 1–5 is assigned to each summary by assessing the user's description.

—Visual quality. This is to ask subjects about the smoothness of camera motion and object motion, as well as the sharpness of the frames, after they browse the summary. A subject is asked to give a score in the scale of 1–5. The higher the score, the better the visual quality.

—Overall satisfaction. This is to ask subjects about their overall satisfaction of the summaries. A subject is asked to give a score in the scale of 1–5, where 5 indicates the best.

Figure 10 shows the results of user studies. Note that the scores for the original videos are fixed to 5 for calibration. We can see that the videos reconstructed from NLSS achieves the best user satisfaction among all the six summarization and compression techniques. We can have the following observations from the results: 1) The static summary is not good at helping subjects understand video content thoroughly, since it is only a synthesized image containing static objects. Instead, the static summary is good to provide a very quick glance of the whole content. 2) Since the compression rate is set extremely low, the H264 cannot achieve acceptable visual quality, resulting in poor performance in terms of content comprehension and narrativeness. 3) As the clips used for generating dynamic summaries come from the original video, the two dynamic summarization techniques achieve fairly good performance in terms of visual quality, while average performance for content understanding. This also validates that content understanding is still a challenging problem for any high level video applications.

5.5 Evaluation of Duplicate Detection

We use a subset of the near-duplicate web video dataset CC_WEB_VIDEO [CC_ WEB_ VIDEO] to evaluate the performance of NLSS for duplicate detection task, which is an important yet hot topic in the field of multimedia [Wu et al. 2009] [TRECVID]. We apply the duplicate detection method described in Section 4.3 to both the videos reconstructed from NLSS summaries and the original videos. The evaluation can provide a view of the capability of NLSS in keeping the semantic fidelity of the original videos. In the experiment, for each category of the video, we select a video as the seed, with which we implement a pairwise comparison on the candidate video for duplicate validation. First, Hessian-Laplace and PSIFT [Zhao and Ngo 2009] are selected as the detector and descriptor, respectively, to extract the local-feature from the keyframes. Then, keyframe-based pairwise matching is performed on these local descriptors. The videos of CC_WEB_VIDEO are all web videos with a short duration,

Table II. Results of video duplicate detection.

		Recall		Precision	
Category	Duplicate num.	Original	NLSS	Original	NLSS
1	58	1	1	1	1
2	8	1	1	1	1
3	68	1	1	1	1
4	2	1	1	1	1
5	7	1	1	1	1
6	6	1	1	1	1
7	43	1	1	1	1
8	1	1	1	1	1
9	1	1	1	1	1
10	7	1	1	1	1
11	2	1	1	1	1
12	26	1	1	1	1
13	149	1	1	1	1
14	9	1	1	1	0.75
15	150	1	1	0.99	0.97
16	2	1	1	1	1
17	77	1	1	0.99	0.99
18	3	1	1	1	1
19	66	1	1	0.99	0.97
20	25	1	1	1	0.86
21	14	1	1	1	1
22	2	1	1	1	1
23	6	1	1	1	1
24	16	1	1	1	1
all	748 (Total video num.: 2,493)				

thus the keyframes extracted from a single video are limited. Therefore, we set 2 as the threshold for matching local feature pairs.

We summarize the results of the experiment in Table II. Note that "original" indicates the duplicate detection on the original videos, while "NLSS" indicates the videos reconstructed from our proposed NLSS summary. We can see that we achieve the same recall of 100% on both video collections across all 24 categories. However, the precision of duplicate detection on the videos reconstructed from the NLSS summary is a little bit lower than that of the original video in four categories (i.e., 14, 15, 19, and 20). We investigate the results (i.e., the keyframes, local features, and pairwise matching results) and have the following observations:

—Since the keyframes in the NLSS summary are $\frac{1}{4}$ of the original resolution, some local feature points from the reconstructed keyframes in NLSS are different from those detected in the original videos. The other reason is that the visual fidelity (e.g., the color entropy and detailed texture) of NLSS-reconstructed keyframes are slightly degraded by the summary generation process, which further affected the extracted local features.

—If we apply timeline validation to our duplicate detection [Wu et al. 2009], then all the false alarms will be eliminated in the NLSS.

Over all, the results show that NLSS can achieve an satisfying performance compared with the original videos in the duplicate detection task, which indicates that NLSS summary is able to preserve satisfying visual fidelity without significantly degrading system performance.

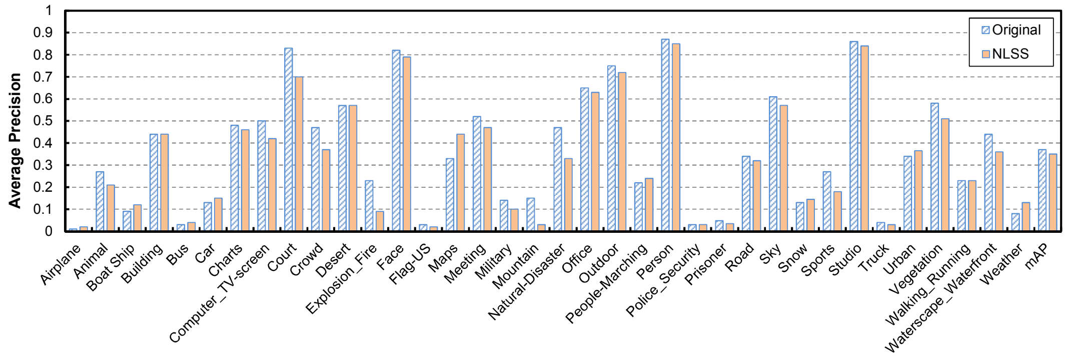


Fig. 11. Performance of video concept detection (data split by keyframes).

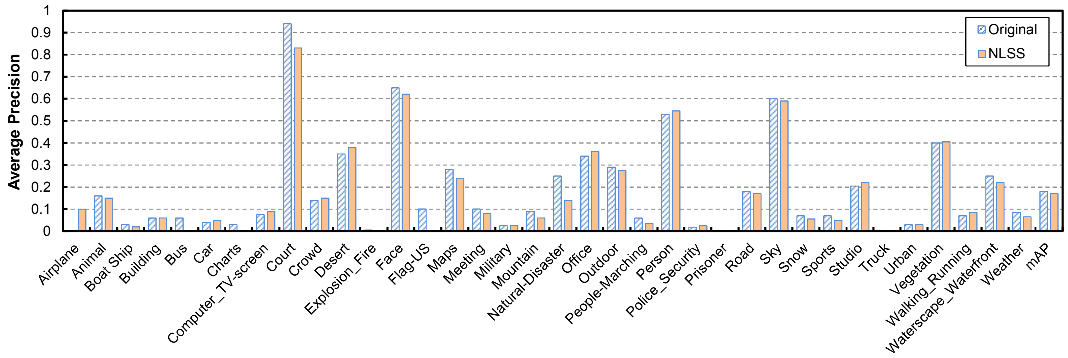


Fig. 12. Performance of video concept detection (data split by videos).

5.6 Evaluation of Concept Detection

To evaluate video concept detection based on the reconstructed videos by NLSS summary, we implement the approach in Section 4.4 to validate the capability of preserving semantic information via the proposed NLSS. We extract the widely adopted 5×5 color moment feature from each keyframe of the videos in TRECVID 2007. Then, we split the keyframes into $training$, $validation$, and $testing$ sets according to two strategies: one is splitting all the videos according to the ratio ($training = 70\%$, $validation = 20\%$, $testing = 10\%$); and the other is splitting all the keyframes of videos with the same ratio as the previous strategy. These two strategies may prevent the unbalance of the keyframes distribution among videos. We trained 36 SVM-based concept detectors with RBF kernel (all 36 concepts are defined by TRECVID for high level feature extraction [TRECVID]). During the validation, we set the parameters in SVMs as $c = [0.5, 1, 10, 100]$, $g = [0.65, 1.0, 1.5, 2.0]$ for a grid search. We adopt the Average-Precision (AP) and Mean Average Precision (MAP) as the performance metric. Note that the above processes are the same for both the original video and the videos reconstructed from NLSS.

Figure 11 and 12 show the overall performance. Note that “original” indicates the concept detection on the original videos, while “NLSS” indicates the videos reconstructed from our proposed NLSS summary. We can see that among most concepts, NLSS is capable of obtaining a similar performance to that of the original video. Even in some concepts, the performance of NLSS exceeds that of the original video. However, from our viewpoint, this kind of superiority cannot be taken as the strength of NLSS in concept discrimination. First, the superiority is limited. Second, the state-of-the-art performance of concept detection in this community is also very limited, due to the noise and variance in the video data. Nevertheless, the results still indicate that the performance of NLSS in concept detection is comparable to that of original video collection.

6. CONCLUSION

By leveraging the current technology of video compression and video summarization, we have presented a novel approach to tackle the large-scale video storage problem. The proposed "near-lossless semantic summarization" (NLSS) is able to achieve extremely low storage consumption—compared with existing popular video coding standards such as H.264, NLSS achieves much higher compression rate, while NLSS can also keep visual and semantic fidelity in a wide variety of video-based applications, such as visualization, indexing, presentation, duplicate detection and concept detection.

NLSS represents an effective tradeoff between conventional video compression and summarization, aiming to preserve visual and semantic fidelity for video applications. The proposed NLSS represents one implementation of the idea of application-dependent video summarization, which is characterized by: 1) application-oriented video analysis for metadata extraction, 2) extremely low bit rate compression of metadata (i.e., summary is with very small size), and 3) semantic-driven performance evaluation (e.g., video indexing and browsing, high-level semantic analysis, and so on). We hope to open a new door or paradigm shift for traditional video summarization and compression. Our future work may include speeding up the motion estimation to generate NLSS summary and applying NLSS to more video applications.

REFERENCES

AMR. 2002. AMR speech codec; general description. *TS 26.071 version 5.0.0*.

BESCOS, J., MARTINEZ, J. M., HERRANZ, L., AND TIBURZI, F. 2007. Content-driven adaptation of on-line video. *Signal Processing: Image Communication 22,* 7-8, 651–668.

BING. http://www.bing.com/?scope=video/.

BORECZKY, J., GIRGENSOHN, A., GOLOVCHINSKY, G., AND UCHIHASHI, S. 2000. An interactive comic book presentation for exploring video. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. 185–192.

BOUTHEMY, P., GELGON, M., AND GANANSIA, F. 1999. A unified approach to shot change detection and camera motion characterization. *IEEE Trans. on Circuits and Systems for Video Technology 9,* 7, 1030–1044.

CC_WEB_VIDEO. Near-Duplicate Web Video Dataset. http://vireo.cs.cityu.edu.hk/webvideo/.

COVELL, M., WITHGOTT, M., AND SLANEY, M. 1998. Mach1: Nonuniform time-scale modification of speech. In *Proceedings of IEEE International Conference on Acoustics, Speech, and Signal Processing*.

FERNANDO, W. A. C., CANAGARAJAH, C. N., AND BULL, D. R. 1999. Automatic detection of fade-in and fade-out in video sequences. In *Proceedings of International Symposium on Circuits and Systems*. Vol. 4. 255–258.

GOOGLE. http://video.google.com/.

H263. 2000. ITU-T Rec. H.263, Video coding for low bit rate communication.

H264. 2003. ITU-T Rec. H.264 — ISO/IEC 14496-10 AVC, Draft ITU-T Recommendation and Final Draft International Standard of Joint Video Specification.

HAMPAPUR, A., HYUN, K., AND BOLLE, R. M. 2002. Comparison of sequence matching techniques for video copy detection. In *Proceedings of Storage and Retrieval for Media Databases*. 194–201.

HAUPTMANN, A. G., CHRISTEL, M. G., LIN, W.-H., AND ETC. 2007. Clever clustering vs. simple speed-up for summarizing rushes. In *Proceedings of International Workshop on TRECVID Video Summarization*. 20–24.

HSU, W. H., KENNEDY, L. S., AND CHANG, S.-F. 2007. Reranking methods for visual search. *IEEE MultiMedia 14,* 3, 14–22.

IRANI, M. AND ANANDAN, P. 1998. Video indexing based on mosaic representations. *Proceedings of the IEEE 86,* 5, 905–921.

ISO/IEC. 1991. Digital compression and coding of continuous still images, part 1: Requirements and guidelines. *ISO/IEC JTC1 Draft International Standard 10918-1*.

JIANG, W., COTTON, C. V., CHANG, S.-F., ELLIS, D., AND LOUI, A. C. 2010. Audio-visual atoms for generic video concept classification. *ACM Trans. on Multimedia Computing, Communications and Applications 6,* 3.

JIANG, Y.-G., YANG, J., NGO, C.-W., AND HAUPTMANN, A. G. 2010. Representations of keypoint-based semantic concept detection: A comprehensive study. *IEEE Trans. on Multimedia 12,* 1, 42–53.

KIM, C. AND HWANG, J.-N. 2002. Object-based video abstraction for video surveillance systems. *IEEE Trans. on Circuits and Systems for Video Technology 12,* 12, 1128–1138.

KIM, J. G., CHANG, H. S., KIM, J., AND KIM, H. M. 2000. Efficient camera motion characterization for mpeg video indexing. In *Proceedings of IEEE Intl. Conf. Multimedia & Expo*. 1171–1174.

KONRAD, J. AND DUFAUX, F. 1998. Improved global motion estimation for n3. *ISO/IEC JTC1/SC29/WG11 M3096*.

LEW, M. S., SEBE, N., DJERABA, C., AND JAIN, R. 2006. Content-based multimedia information retrieval: State of the art and challenges. *ACM Transactions on Multimedia Computing, Communications and Applications 2,* 1, 1–19.

LI, Y., JIN, J., AND ZHOU, X. 2005. Video matching using binary signature. In *International Symposium on Intelligent Signal Processing and Communication Systems*. 317–320.

LIU, Y., MEI, T., AND HUA, X.-S. 2009. CrowdReranking: exploring multiple search engines for visual search reranking. In *Proceedings of ACM SIGIR conference on Research and Development in Information Retrieval*. 500–507.

LOWE, D. 2004. Distinctive image features from scale-invariant key points. *International Journal of Computer Vision 60,* 2, 91–110.

LU, L., ZHANG, H.-J., AND LI, S. Z. 2003. Content-based audio classification and segmentation by using support vector machines. *Multimedia Systems 8*, 482–492.

MA, Y.-F., LU, L., ZHANG, H.-J., AND LI, M. 2002. A user attention model for video summarization. In *Proceedings of ACM International Conference on Multimedia*. 533–542.

MEI, T. AND HUA, X.-S. 2008. Structure and event mining in sports video with efficient mosaic. *Multimedia Tools and Applications 40,* 1, 89–110.

MEI, T., HUA, X.-S., LAI, W., YANG, L., AND ET AL. 2007. MSRA-USTC-SJTU at TRECVID 2007: High-level feature extraction and search. In *TREC Video Retrieval Evaluation Online Proceedings*.

MEI, T., HUA, X.-S., AND LI, S. 2009. VideoSense: A contextual in-video advertising system. *IEEE Trans. on Circuits and Systems for Video Technology 19,* 12, 1866–1879.

MEI, T., HUA, X.-S., ZHU, C.-Z., ZHOU, H.-Q., AND LI, S. 2007. Home video visual quality assessment with spatiotemporal factors. *IEEE Trans. on Circuits and Systems for Video Techn. 17,* 6, 699–706.

MEI, T., YANG, B., YANG, S.-Q., AND HUA, X.-S. 2009. Video collage: Presenting a video sequence using a single image. *The Visual Computer 25,* 1, 39–51.

MOXLEY, E., MEI, T., AND MANJUNATH, B. S. 2010. Video annotation through search and graph reinforcement mining. *IEEE Trans. on Multimedia 12,* 3, 184–193.

MPEG-2. MPEG-2 Video Group, Information technology - generic coding of moving pictures and associated audio: Part 2—Video. *ISO/IEC 13818-2*.

MPEG-4. MPEG-4 Video Group, generic coding of audio-visual objects: Part 2—Visual. *ISO/IEC JTC1/SC29/WG11 N1902, FDIS of ISO/IEC 14 496-2*.

NAPHADE, M., SMITH, J. R., TESIC, J., CHANG, S.-F., HSU, W., KENNEDY, L., HAUPTMANN, A., AND CURTIS, J. 2006. Large-scale concept ontology for multimedia. *IEEE MultiMedia 13,* 3, 86–91.

NITTA, N., TAKAHASHI, Y., AND BABAGUCHI, N. 2009. Automatic personalized video abstraction for sports videos using metadata. *Multimedia Tools and Applications 41,* 1, 1–25.

OVER, P., SMEATON, A. F., AND AWAD, G. 2008. The TRECVid 2008 BBC Rushes Summarization Evaluation. In *Proceedings of ACM TRECVid Video Summarization Workshop*. 1–20.

PAISITKRIANGKRAI, S., MEI, T., ZHANG, J., AND HUA, X.-S. 2010. Scalable clip-based near-duplicate video detection with ordinal measure. In *Proceedings of ACM International Conference on Image and Video Retrieval*.

SHAO, X., XU, C., MADDAGE, N. C., TIAN, Q., KANKANHALLI, M. S., AND JIN, J. S. 2006. Automatic summarization of music videos. *ACM Trans. on Multimedia Computing, Communications and Applications 2,* 2, 127–148.

SMEULDERS, A. W. M., WORRING, M., SANTINI, S., GUPTA, A., AND JAIN, R. 2000. Content-based image retrieval at the end of the early years. *IEEE Trans. on Pattern Analysis and Machine Intelligence 22,* 12, 1349–1380.

SNOEK, C. G. M., WORRING, M., VAN GEMERT, J. C., GEUSEBROEK, J.-M., AND SMEULDERS, A. W. M. 2006. The challenge problem for automated detection of 101 semantic concepts in multimedia. In *Proceedings of ACM International Conference on Multimedia*. 421–430.

TANG, L.-X., MEI, T., AND HUA, X.-S. 2009. Near-lossless video summarization. In *Proceedings of ACM International Conference on Multimedia*. Beijing, China, 351–360.

TJONDRONEGORO, D., CHEN, Y.-P. P., AND PHAM, B. 2003. Sports video summarization using highlights and play-breaks. In *Proceedings of ACM SIGMM International Workshop on Multimedia Information Retrieval*. 201–208.

TRECVID. http://www-nlpir.nist.gov/projects/trecvid/.

TRUONG, B. T. AND VENKATESH, S. 2007. Video abstraction: A systematic review and classification. *ACM Trans. on Multimedia Computing, Communications and Applications 3,* 1.

TSE, T., MARCHIONINI, G., DING, W., SLAUGHTER, L., AND KOMLODI, A. 1998. Dynamic key frame presentation techniques for augmenting video browsing. In *Proceedings of the working conference on Advanced visual interfaces*. 185–194.

WANG, Y., MEI, T., AND HUA, X.-S. 2011. Community discovery from movie and its application to poster generation. In *Proceedings of International MultiMedia Modeling Conference*.

WIEGAND, T., SULLIVAN, G. J., BJONTEGAARD, G., AND LUTHRA, A. 2003. Overview of the H.264/AVC video coding standard. *IEEE Trans. on Circuits and Systems for Video Techn. 13,* 7, 560–576.

WU, X., NGO, C.-W., HAUPTMANN, A. G., AND TAN, H.-K. 2009. Real-time near-duplicate elimination for web video search with content and context. *IEEE Trans. on Multimedia 11,* 2, 196–207.

ZHANG, H.-J., KANKANHALLI, A., AND SMOLIAR, S. W. 1993. Automatic partitioning of full-motion video. *Multimedia Systems 1,* 1, 10–28.

ZHAO, W.-L. AND NGO, C.-W. 2009. Scale-rotation invariant pattern entropy for keypoint-based near-duplicate detection. *IEEE Trans. on Image Processing 18,* 2, 412–423.

ZHAO, W.-L., NGO, C.-W., TAN, H.-K., AND WU, X. 2007. Near-duplicate keyframe identification with interest point matching and pattern learning. *IEEE Trans. on Multimedia 9,* 5, 1037–1048.