Video Summarization based on User Log Enhanced Link Analysis

Bin Yu
Wei-Ying Ma
Klara Nahrstedt
Hong-Jiang Zhang

Aug 14, 2003

Technical Report
MSR-TR-2003-51

Microsoft Research
Microsoft Corporation
One Microsoft Way
Redmond, WA 98052

Video Summarization based on User Log Enhanced Link Analysis

Bin Yu
Department of Computer Science
University of Illinois at Urbana-Champaign,
1304 W. Springfield
Urbana, IL 61801, U.S.A.
binyu@uiuc.edu

Wei-Ying Ma
Microsoft Research Asia
5/F, Beijing Sigma Center,
Zhichun Road,
Haidian District, Beijing
100084, PRC
wyma@microsoft.com

Klara Nahrstedt
Department of Computer Science
University of Illinois at Urbana-Champaign,
1304 W. Springfield
Urbana, IL 61801, U.S.A.
klara@uiuc.edu

Hong-Jiang Zhang
Microsoft Research Asia
5/F, Beijing Sigma Center,
Zhichun Road,
Haidian District, Beijing
100084, PRC
hjzhang@microsoft.com

ABSTRACT

Efficient video data management calls for intelligent video summarization tools that automatically generate concise video summaries for fast skimming and browsing. Traditional video summarization techniques are based on low-level feature analysis, which generally fails to capture the semantics of video content. Our vision is that users *unintentionally* embed their understanding of the video content in their interaction with computers. This valuable knowledge, which is difficult for computers to learn autonomously, can be utilized for video summarization process. In this paper, we present an intelligent video browsing and summarization system that utilizes previous viewers' browsing log to facilitate future viewers. Specifically, a novel ***ShotRank*** notion is proposed as a measure of the subjective interestingness and importance of each video shot. A ShotRank computation framework is constructed to seamlessly unify low-level video analysis and user browsing log mining. The resulting ShotRank is used to organize the presentation of video shots and generate video skims. Experimental results from user studies have strongly confirmed that ShotRank indeed represents the subjective notion of interestingness and importance of each video shot, and it significantly improves future viewers' browsing experience.

Categories & Subject Descriptors

I.2.10 [**Vision and Scene Understanding**]: Video Analysis

General Terms

Algorithms, Management, Design, Human Factors.

Keywords

Video summarization, Link analysis, User behavior, Log mining, Skimming, Video content analysis

MM'03, ***NOVEMBER 2-8, 2003, BERKELEY, CALIFORNIA, USA.***

1. INTRODUCTION

With the fast development in video capture, storage and distribution technologies, the amount of video content accessible in people's daily life is growing exponentially. To handle such overwhelming amount of data, efficient video management technologies are urgently called for. *Video summarization*, which generates a concise summary of the semantics in a video clip, is one of the key technologies to help people browse and search the large amount of video data. Such a summary can be either *static*, consisting of a sequence of key frames, or *dynamic*, consisting of a dynamically-composed collection of audio-video sub-clips, and in both cases the goal is to find the most *interesting* or *important* video segments that capture the essence of the original clips.

A great amount of efforts have been devoted to the problem of video summarization, and various assumptions have been made on how low-level features indicate semantics of the video content. For example, *visual features*, such as color histogram, texture, shape and edges, are utilized to detect significant changes between video frames and to group similar frames together [1, 2, 3, 4, 5, 6, 7]. *Aural features*, such as changes in the audio level and speech analysis, have been used to detect occurrence of meaningful events [3, 7]. Besides visual/aural features, certain *visual events* are also explored, such as close-up of humans or talking heads, objects movements, emotional dialogs, violent-featured actions and scenes with high contrasts [8, 9]. The goal is to capture the knowledge embedded in the video creation process by the directors, cinematographers and the editors. From another perspective, *textual information* and *meta-data* accompanying a video clip have been studied in [3, 7, 10, 11, 12]. Such information is assumed to bare the content creators' knowledge in the video content, and normally includes closed-caption and titles, date/time stamps, game scores, presentation scripts, meeting minutes and so on. A recent effort is the *Attention Model* by Ma et al. [13] that aims at bridging low-level features and human perception by analyzing viewers' *attention*. Specifically, several low-level models such as motion, face, camera, speech and music are analyzed and fused into an *attention value* for each frame, and the most "attention-drawing" segments are selected for the summary. Finally, many successful systems have already been deployed that integrate these video analysis techniques, such as the Informedia Project from [14].

Although these efforts work well for some particular set of video clips under certain settings, generally the results are still far from satisfactory. Besides reasons related to the unavailability of necessary materials (e.g. meaningful dialogues and captions) and the difficulties in natural language understanding, one funda-mental

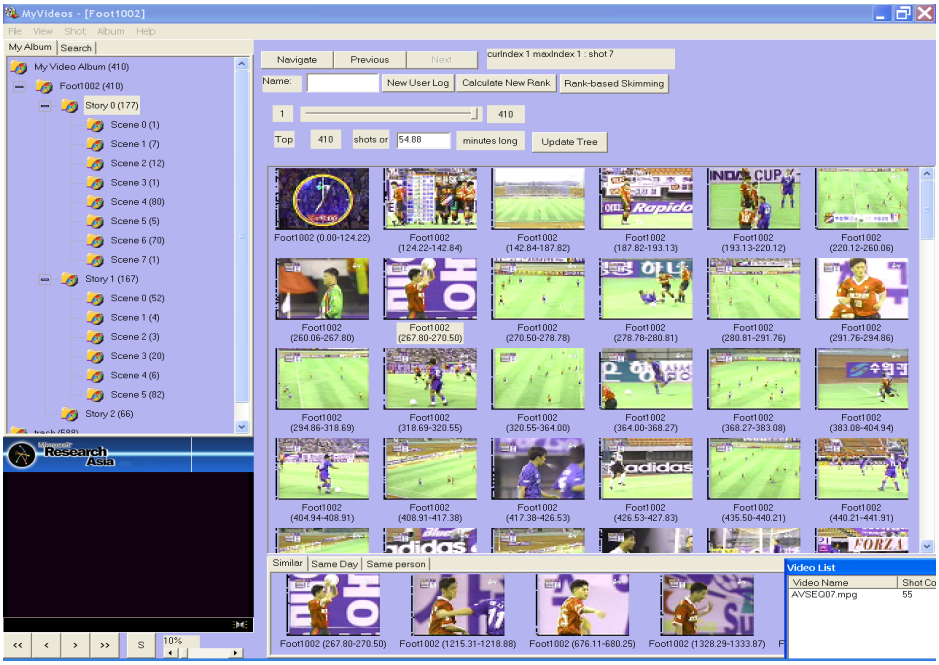

Figure 1. Screenshot of the MyVideo video browser interface

problem is that the mapping from low-level features to high-level human understanding is not valid sometimes.

Our key vision is that the understanding of the semantics of a video clip is a highly subjective process, so the ground truth used in video summarization must also come from the viewers themselves. The insight is that viewers unintentionally leave footprints of their video content understanding and evaluation (e.g. frequency of browsing a video segment) during their browsing process. By mining past viewers' browsing log, we can generate meaningful video summaries that assist future viewers, who in turn will render more meaningful browsing trails.

The most straightforward way of utilizing user log follows the "majority rule" in statistics. For example, in DirectHit [15], it is assumed that the more frequently a web-page is visited by users, the more important that web-page is. However, when it comes to video log mining, the user browsing behavior is more complex. The study done by Acharya et al. [16] has revealed that on the Internet, almost 45% video playback requests stop very early. If the naive counting technique is used, this would be misinterpreted as that the beginning of the video is always more interesting.

The IBM MediaMiner project by Syeda-Mahmood et al. [17] is a representative work that utilizes user logs to generate video summaries. In this work, the viewers are provided with a VCR-like video player with controls such as PLAY, PAUSE, STOP, FAST FORWARD and FAST BACKWARD. A Hidden Markov Model is applied to predict users' internal states from their browsing behavior, such as "CURIOUS", "LOOKING FOR SOMETHING", "FOUND SOMETHING INTERESTING" etc. The video segments corresponding to the viewers' interested states are taken as candidates for generating a video preview. Though this work has demonstrated that previous users' experiences can indirectly help future users in their browsing, some problems are still unsolved. The internal state sometimes may not be predicted correctly: for example, if the viewer goes through a sequence of "FAST FORWARD → PLAY → PAUSE", it is equally likely that either he has found something interesting or he has discovered it is not what he is looking for. Besides, there is not a general model of leveraging low-level feature analysis techniques, and no distinction is made between "engaged" viewers whose browsing log is abundant of information and "random" viewers who generate near-random trails. Another work that aims at integrating semantic and visual similarity is proposed by X. Zhu in [18], which utilizes semi-automatic annotation to acquire semantic information. Our work distinguishes from this work in that we are looking at pure automatic approaches, and, rather than weighted average adopted in [18], we use an integrated framework to combine semantics with low-level similarities.

In this paper, we present a framework that combines video content analysis and user log mining to generate a video summary. Specifically, we adopt the link analysis technique used in web mining, and propose a novel concept of ***ShotRank*** that measures the interestingness or importance of each video shot. Similar to the Random Walk model for PageRank proposed by Brin et al. in [19], the user behavior is simulated with an ***Interest-guided Walk*** model, and the probability of a shot being visited is taken as an indication of the interestingness and importance of that shot. Low-level features and user logs are represented by virtual links among video objects and between users and video objects, and a unified link analysis is applied to derive the ShotRank. Note that there is the concern that different users may have different, or even opposite ideas of interestingness for a particular shot, but we assume that there are always some segments of a video that is commonly interesting to most viewers. It is also possible to group viewers based on their interests.[1]

We have also developed a novel user interface to utilize the ShotRank to generate video skims and hierarchical story trees that

[1] For example, viewers from the same social group or background tend to have similar interests.

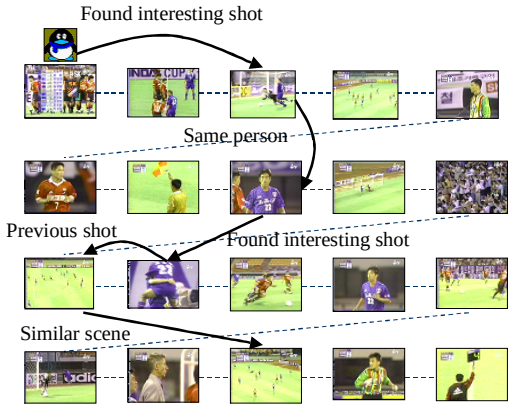

Figure 2. Interest-guided Walk Model for ShotRank interpretation

improve viewers' browsing experiences. To achieve objectivity into the evaluation of such a subjective study, we have adopted a 2-stage questionnaire methodology in our experiments, where satisfactory and convincing results have been achieved.

This paper is organized as follows. In section 2, we present the notion of ShotRank and the Interest-guided Walk Model. Section 3 discusses how they are utilized in video summarization. The implementation details are given in section 4, followed by a theoretic discussion in section 5. Finally, section 6 describes the evaluation results, and section 7 gives the conclusion and some future directions.

2. ShotRank: Bringing Order to Video Shots

The key notion that distinguishes our work from previous ones is the computation of the ***ShotRank***. Similar to PageRank [19], which sorts web pages according to their *authority* weights, ShotRank is used to measure the *interestingness* and *importance* of video shots. We utilize such an objective measure to tell us which shots are more interesting and more significant for the understanding of the original video clip.

2.1 Interest-guided Walk Model

To better explain the notion of ShotRank, let us first study a video viewer's browsing behavior. Figure 1 (on previous page) shows the browsing interface we have developed based on our previous MyVideo project [20]. Each video clip is first segmented into video shots using low-level shot boundary detection algorithms by Zhang et al. [21], and then one key frame is selected to represent each shot. As can be seen in Figure 1, the result of video segmentation is shown on the upper-left panel where the video "Foot1002" is organized in a hierarchy of "video → story → scene", and when a tree node is selected, all the shots (represented by key frame icons) are then displayed in the right panel. If a shot is selected, then all the similar shots (identified with low-level feature analysis) will be listed in the bottom-right panel. Double-clicking on a shot icon will trigger the windows media player to play the shot in the bottom-left panel. The media player provides VCR-like controls such as PLAY, STOP and PAUSE.

With this interface, the viewer can get an overview of the video based on the key frames, select a shot to play, and jump to another shot at any time. A natural assumption is that each viewer's browsing process is guided by the search for interesting and important shots. Figure 2 shows the track of an ordinary viewer's behavior as he browses through 20 shots from a football game. He may jump to a particular shot when he finds an interesting icon, and he might go through the precedent or subsequent shots to learn more after the jump. In addition, since our user interface provides the viewer with related shots of a similar scene or the same person, the viewer can jump to those shots directly. In summary, in our approach, video shots are not only sequentially connected, but also non-linearly "linked" to each other if

1. they look similar, or
2. they depict the same person, or
3. they share the same audio/voice track, or
4. they are commonly interesting, or
5. they are neighboring on the time axis, or
6. they are on a common topic.

Note that all previous results in video object hyper-linking can be incorporated here, such as work from [22, 23, 24], but we further include the links for non-visual similarities that guide viewers in their navigation (previous and next). As these links indicate some latent relationship between shots, the shots' importance and interestingness are mutually transferred. Intuitively, the higher a shot is ranked, the more likely that some other shots will guide the viewer to this shot via virtual links, and the more likely that this shot is visited by the viewer.

2.2 Definition of ShotRank

As we discussed above, the ShotRank of a video shot can be interpreted as ***the probability that a viewer would visit a shot during his browsing***. Since the user's browsing behavior is controlled by his subjective evaluation on each video shot and guided through the "links" between video shots, this probability corresponds to the subjective notion of interestingness and importance of each shot relative to other shots in the same video clip.

Formally, for a shot A, suppose there are N shots, B_1 through B_N, linked to A, then the ShotRank of A can be calculated as

$$ShotRank(A) = \frac{1}{N} \cdot \sum_{i=1}^{N} [w_i \cdot ShotRank(B_i)] \qquad (1)$$

where the weights w_i control how much influence one shot has on other shots it links to. It could be a normalization factor using the total number of links the shot B_i has, but we will simply set all weights to *1* for simplified discussion below.

2.3 Utilizing User Browsing Log

Now that we have the definition of ShotRank and Equation (1) to compute one shot's ShotRank via shots linked to it, we still need some knowledge about each video shot to bootstrap our ShotRank computation process. First, some shots are "attractive" themselves, and they do not rely on links from other shots to earn their high rank. Second, even though links are important in guiding viewers in their browsing, they *transfer* rather than *indicate* importance of shots. That is, unlike the previous link analysis algorithms, such as PageRank, that rely on the knowledge put into hyperlinks between web pages by pages designers(e.g. anchor text), the links between video shots in our model do not indicate interestingness or importance themselves. Therefore, we need to inject some additional knowledge that comes from the understanding of the semantics of the original video.

As introduced before, the key observation is that, as viewers browse through the video shots on their own judgment, their browsing logs reflect their subjective understanding and evaluation of the video. For example, a nostalgic father who has collected dozens of hours of video of his son may review his favorite segments again and again; viewers will seek for the most wonderful scenes when they watch the video of a recorded soccer game; a user of a New-On-Demand service will stick to the hottest news while ignoring non-significant ones; and when students try to make up the last lecture on the web, they will jump directly to the discussion about the final exam. In summary, viewers' evaluation and understanding of a video are expressed in their browsing behavior and accumulated in their browsing log. If we could effectively extract this valuable knowledge, then it can be used to enhance the computation of ShotRank.

2.4 The Unified Computation Framework

To naturally combine the latent knowledge in user log and the similarity links between video shots, a unified framework similar to the one proposed by Chen et al. in [25] for web mining is used to compute the ShotRank for video shots.

Figure 3 shows the unified link structure. *Visiting links* are created between users and the video shots they have visited, and *similarity links* connect shots based on various similarity measures discussed previously. This way, we get the ShotRank computation algorithm.

ShotRank Computation Algorithm

Let s denote the vector of ShotRank of all shots of a video clip, u denotes the vector of user weighting (user log quality or importance), or *UserRank*, of all users, A and V denote the adjacency matrix between shots and the visiting matrix between users and shots, and β denotes the tuning parameter. Then the ShotRank can be calculated iteratively using the following formula:

$$\begin{cases} s = \beta \cdot A^T \cdot s + (1-\beta) \cdot V^T \cdot u \\ u = (1-\beta) \cdot V \cdot s \end{cases} \quad (2)$$

The visiting links between users and shots have the following effects on computing the ShotRank:

- These links explicitly indicate the voting of users on the importance and interestingness of the video shots.
- Because user weighting is adjusted based on the ShotRank of all shots at each iteration, the visiting links implicitly provide a mutual reinforcement relation between "interesting" shots and "engaged" users, which is very similar to the relation between high quality authorities and hubs in Kleinberg's HITS algorithm [26]. This helps filter out the effect of "random" viewers.

In our definition, ***engaged viewers*** are (a) interested in what they are browsing, (b) experienced with the browsing tool interface, (c) patient to look at each key frame because they might have more time, (d) eager to find something, (e) having high bandwidth connection in the case of Video-On-Demand, and etc. On the contrary, ***random viewers*** are the opposite of engaged viewers. They may not be interested or do not have enough time, so they give up very quickly. They may be novice users of the browsing tool and not familiar with the controls, or they may have some peculiar interests not common to others. Therefore, there is a mutual reinforcement relation that engaged viewers tend to select

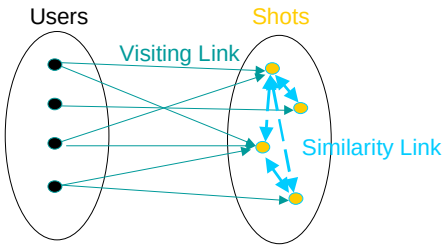

Figure 3. Bipartite graph of users and shots

and visit interesting shots, and interesting shots are normally discovered by most engaged viewers. Our iterative algorithm gradually builds up the reinforcement between interesting shots and engaged viewers and so distinguishes them from random shots/viewers.

The adjacency matrix A represents the similarity links between shots, and it transfers one shot's importance to other shots linked to that shot. This represents another level of reinforcement between interesting shots themselves. If a shot is interesting to a viewer, then it is likely that those shots linked to this shot (e.g. shots depicting the same person or scene) may also be interesting to the viewer.

We also consider *sequential similarity*, which takes into account the reinforcement between neighboring shots. For example, if a viewer finds one shot to have something he is looking for, then it is likely that those shots around this shot will also be interesting to the viewer.

In summary, our framework unifies low-level feature analysis with user browsing log mining in a seamless and flexible way.

3. Video Summarization based on ShotRank

In our system, two types of video summaries are constructed based on the ShotRank. We now discuss each of them below.

3.1 ShotRank-Based Skims

Video skim is a sequential playback of some audio-video sub-clips, which preserves the time-evolving characteristic of the original video. With the help of ShotRank, generating video skim becomes as simple as selecting the top ranking shots and concatenating them together into a running video stream.

This feature is integrated in our browser interface by providing the users a sliding bar to specify how many top ranking shots he would like to see, as shown in Figure 1. For example, if the user chooses to see the 30 top-ranking shots and clicks "rank-based skimming", then these 30 shots are played back one by one as a skim. Alternatively, he may also specify approximately how much time he has to watch a video skim. Based on the length and ShotRank of each shot, the browser can calculate exactly how many top-ranking shots to present to the viewer as a skim.

This feature gives the users a fine granularity in requesting various lengths of video skims, and is especially suitable for video previews presented to Video-On-Demand clients.

3.2 ShotRank-Based Story-tree

Though simple, the ShotRank-based skims are not suitable for more advanced video browsing tasks. It is likely that a user would prefer to browse through the video manually to search for interesting scenes or some particular information. For example, if a student wants to find out what the professor's comments are while reviewing a recorded lecture, he might just directly jump to that

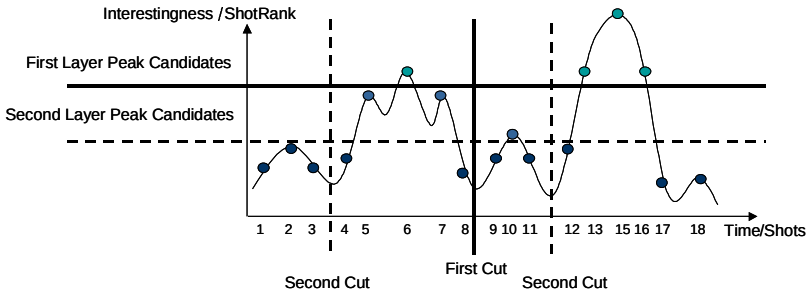

Figure 4. ShotRank-Based Hierarchical Clustering of Video Shots

segment using the browser. Therefore, we need to go beyond video skimming and re-organize the shots to facilitate users in their browsing and searching.

We have adopted the hierarchical structure proposed by Zhang et al. in [27], but applied an original hierarchical clustering algorithm that is tailored for our application. In our case, the resulting tree of shots need to facilitate users in their browsing, and low-level similarity should yield to high-level semantic relations between shots in organizing the shots. Besides, we also need to consider the temporal relation between shots, since shots in each cluster have to be sequentially continuous. In such circumstances, we have developed a *ShotRank-based Story-Tree (SST)* clustering algorithm.

The key observation is that almost all video clips are more or less "structured" in that they can be segmented into several sub-clips, each expressing a standalone topic or depicting a certain event. Besides, within each sub-clip, further segmentation is often possible to delimit different sub-events at a finer level. Depending on the scale of an event, it varies how fine granularity such segmentation may go to. In our current implementation, we have identified two intermediate abstraction levels called "stories" and "scenes". For example, for CNN news broadcast, there might be several pieces of news focusing on certain events, so all these shots form a *story* and each piece of news can be called a *scene*. For a presentation on a research paper, stories might be "introduction", "related work", "our solution" and "evaluation". Within the story of "evaluation", "experimental setup" and "experimental results" are probably typical scenes. For less structured videos, this segmentation may not be obvious, but still reasonable. For a soccer game, certain events such as a goal or a serious fault form certain stories separated by the relatively boring periods in between. Even for movies, the scenario will never evolve steadily, but fluctuate through climaxes and anticlimaxes. In summary, almost all types of videos can be viewed in such a *structured* way, and the benefit is that viewers could focus on certain important stories or events and neglect others in their browsing.

From the plot of the ShotRank vector of all shots for a soccer game, we observe that interesting events such as goals are ranked higher than other non-interesting scenes. Each major peak in the ShotRank curve corresponds exactly to the shots depicting one interesting event, while its preceding and succeeding shots serve as the preludes and epilogues for that event. Consequently, we assume that ShotRank reflects the semantic plot of a video, and the high and low points in the curve of ShotRank for all shots can be used for clustering. For example, Figure 4 shows how the interestingness of a video fluctuates (the solid curve), which is reflected in the changing value of the ShotRank value for each shot (dots in the figure). Specifically, the clustering algorithm follows four steps:

1. **Prepare the candidate set for peak shots**: If we describe each story on the story line as an "event hill", then each such hill can be identified by the shot at its peak. The *N* top ranking shots over all shots, S_1 through S_N, are selected as the candidates for such peaks. The ideal value of *N* should be a little more than the number of stories in a particular video, but in our implementation, we heuristically set *N* to be *1/5* of the total number of shots.
2. **Find the peak shots:** Since the ShotRank of neighboring shots could be very close, one significant event hill may contain several high-ranking shots. However, it is not easy to tell whether two sequentially close high-ranking shots are from one big event or two consecutive events. Therefore, we adopt a simple algorithm to determine the "real" event peaks along the ShotRank curve:
    a. Calculate the average difference $d_{average}$ between timestamps of every 2 consecutive shots.
    b. For each pair of consecutive shots, if their temporal distance is smaller than a heuristic threshold $\gamma * d_{average}$, then the one shot with a smaller ShotRank is dismissed from the set of candidate event peak shots.

    After this step, each of the remaining shots is assumed to be the peak of one event hill.
3. **Cut event boundary:** After identifying the peak shots, we need to decide on the boundary between each pair of neighboring event hills. This is done by taking the shot between each two peak shots with the lowest ShotRank. Intuitively, the appearance of such a "boring" shot indicates the end of the previous event and the start of a new event.
4. **Recursion:** When more than one level of hierarchy is needed to provide finer granularity, the 3 steps above could be applied again within each event hill. In our current implementation, we adopt a 4-level hierarchy: whole clip → story → scene → shot.

4. Implementation

An important functionality of the video browser interface is to observe and log valuable user behavior for computing ShotRank automatically based on users' understanding of the video. In our user interface design, we stick to two general principles: *straightforwardness* and *versatility*. First, the interface should be very simple to understand and use. With the help of the story tree and the key frame list, the user could easily and quickly browse

Action Type	Action Value In Log	Interpretation
STOP/SELECT	0	User selects a key frame which is set to stopped mode
PLAY	1	User selects to play a shot
PAUSE	2	User pauses a playing shot
JUMP	3	User double-clicks on a similar shot listed below and jumps to it
PREVIOUS	4	User wants to see the previous shot visited
NEXT	5	User wants to see the next shot visited

Table 1 User behavior log types

through a whole story at a glance. Each frame is clearly presented and self-explained, so when a user selects it and plays the corresponding shot, it more likely indicates that he feels interested in knowing more about that shot.

The second principle is to provide as much functionalities as possible for the users to use when they need them. Specifically, we have borrowed the idea of "navigation history" from web browsers so that viewers could go back to a previous shot he has visited before or the next shot he has visited. Besides, shots similar to the current selected one are listed in the bottom right panel in the interface, and by double-clicking them the user could jump to those shots immediately. We also list shots of similar person or similar place for the user's convenience when a particular shot is selected. In addition, advanced users could utilize the ShotRank of shots by specifying how many top ranking shots he wants to view. This way, he could quickly locate the most likely interesting shots.

In summary, the kinds of user behavior log we currently collect are listed in Table 1. A segment of example user log is shown in Figure 5, and each line is in the format of

"*Action_Value Shot_In_Concern Time_Stamp // Explanation*".

The user log is used to create the *A* and *V* matrices for ShotRank computation. For the matrix *A*, we consider six kinds of similarities detected by low level features or from user log:

- ***Same person***: if 2 shots *i* and *j* are depicting the same person, then *A(i,j)* is added 1;
- ***Same place***: if 2 shots *i* and *j* are depicting events happening at the same place, then *A(i,j)* is added 1;
- ***Similar shots***: if 2 shots *i* and *j* are judged to be similar based on low level similarity tests, then *A(i,j)* is added 1;
- ***Neighboring shots (sequential similarity)***: if 2 shots *i* and *j* are next to each other in their time stamp, then *A(i,j)* is added 1;
- ***Browsing link***: we analyze the user log and search for trails of continuous forward or backward. This is interpreted as the following: the current shot that the user selects reminds him of some shots he has seen before, and so he is resorting to the history log to traverse back and forth looking for that shot. Therefore, the two shots at the starting and ending points of such trails are assumed to be similar and the corresponding slot in the matrix *A* is added 1.
- Jumping link: if the user has double-clicked one of the "similar shots" listed in the right bottom rectangle, then it could surely be interpreted as the confirmation of the user of the similarity between the currently selected shot and the shot he jumps to.

```
…
0  11    21002796 //SELECT/STOPPED at SHOT 11
0  18    21003390 //SELECT/STOPPED at SHOT 18
0  19    21003796 //SELECT/STOPPED at SHOT 19
1  19    21004210 //PLAY SHOT 19
0  7     21007312 //SELECT/STOPPED at SHOT 7
3  7     21007312 //JUMP FOR SIMILARITY FROM SHOT 7
0  9     21008484 //SELECT/STOPPED at SHOT 9
4  9     21010562 //PREVIOUS OF SHOT 9
0  7     21010562 //SELECT/STOPPED at SHOT 7
4  7     21010906 //PREVIOUS OF SHOT 7
0  19    21010953 //SELECT/STOPPED at SHOT 19
4  19    21011375 //PREVIOUS OF SHOT 19
0  18    21011375 //SELECT/STOPPED at SHOT 18
4  18    21011812 //PREVIOUS OF SHOT 18
0  11    21011843 //SELECT/STOPPED at SHOT 11
5  11    21012875 //NEXT OF SHOT 11
0  18    21012921 //SELECT/STOPPED at SHOT 18
… …
```

Figure 5. Example User Log

Our framework is completely open in that other similarity measurement methods of video shots could be easily utilized by adding them to the creation process of the matrix *A*.

As for the matrix *V*, it is created based on the user log. As we have said earlier, since our key frame based interface is very straightforward, the access log could be mapped to user interest directly as follows for user *i* 's log retaining shot *j*:

- SELECT: *V(i,j)* is added 1. Normally the user selects a frame because he wants to see a larger picture of the same frame and similar shots listed, so this indicates a basic interests in this frame.
- PLAY: *V(i,j)* is added 0.5. If the user chooses to play a shot after studying its key frame, then this confirms his further interests. If a frame is played for multiple times, it also indicates the user has found something he is looking for.
- PAUSE: *V(i,j)* is added 0.5. If the user pauses a playing shot, it is probably that he wants to take a closer look at a particular scene.
- JUMP: *V(i,j)* is added 1. If the user chooses to jump to a similar shot listed below, then this means he is not only interested in this shot, but also that related shot as well.

Note that currently the impact of each user action on *V(i,j)* is determined heuristically. We believe *V(i,j)* can be learned from user logs and feedbacks, and we will explore this issue in future work.

5. Discussion

5.1 Convergence Proof

To prove the computation of ShotRank in Equation (2) will always converge, we need to simplify Equation (2) into a reiterative format shown below:

$$\begin{aligned} s &= \beta \cdot A^T \cdot s + (1-\beta)^2 \cdot V^T \cdot V \cdot s \\ &= [\beta \cdot A^T + (1-\beta)^2 \cdot V^T \cdot V] \cdot s \end{aligned} \quad \textbf{(3)}$$

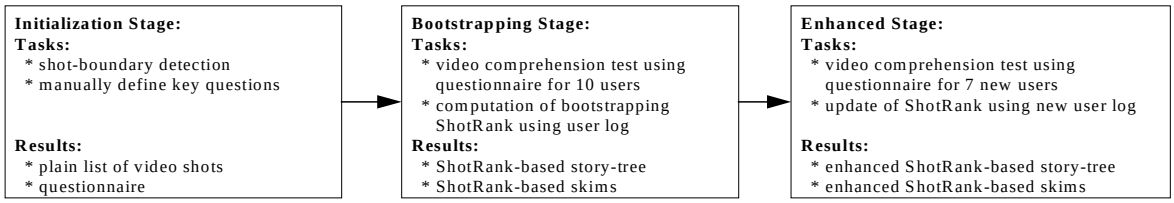

Figure 6. Flow chart of evaluation methodology

Let z denote the vector (1, 1, … 1) $\in R^n$, the initial value of s. Let T denote the matrix $[\beta \cdot A^T + (1-\beta)^2 \cdot V^T \cdot V]$ and s_k denote the ShotRank vector value after the kth round of the iteration according to Equation (4), then we know

$$s_k = T^k \cdot z \quad \textbf{(4)}$$

In the HITS paper [26], Kleinberg proves that the value of s_k will converge to $\omega_1(T)$, the principle eigenvector of T, if T is non-negative and symmetric, and z is not orthogonal to $\omega_1(T)$. In our case, since A^T and V^TV are both non-negative symmetric matrices, we know T is also non-negative and symmetric. Also, since z is set to the unit vector, the probability that z is orthogonal to $\omega_1(T)$ is almost 0 in practice. Therefore, Equation (2) will converge.

5.2 Algorithm Extension

Our algorithm for computing ShotRank can be extended to incorporate the knowledge of some inherent properties of each video shot. For example, if we know from user study that users do not like shots that are too short or too long, we could reflect this preference in the ranking of the shots and so naturally generate a better skim. To do this, a diagonal matrix D can be introduced into Equation (2) to incorporate any analysis results on each individual shot. Now the computation of ShotRank becomes:

$$\begin{cases} s = [\beta \cdot A^T + \alpha \cdot D] \cdot s + (1-\beta) \cdot V^T \cdot u \\ u = 2 \cdot (1-\beta) \cdot V \cdot s \end{cases} \quad \textbf{(5)}$$

where α is used to adjust the effect of D on the final ShotRank. If α is 0, then Equation (5) degenerates to Equation (2); when α is not 0, the value of the components on the diagonal of D will affect the final rank. Intuitively, the higher $D[i,i]$ is for any i, the higher ShotRank s_i. For the example given above, we could simply set the value of $D[i,i]$ of shot i to be lower if it is too long or too short according to certain thresholds.

1. For the soccer game:
   a) In which shot does the first kick take place?
   b) For how long is the first half extended?
   c) What's the score after the 45 minutes match? At what time do these goals occur?
   d) Which 3 shots do you think are worth viewing by others (That is, you would like to recommend them to others)?
   e) Do you prefer the current ShotRank-based browsing tool to the traditional VCR-like player for browsing soccer game?
2. For the home video:
   a) How many major scenes are there?
   b) How many people are there in this family? Give an example shot.
   c) Enumerate 2 kinds of snow skiing tools they used. Give example shots.
   d) Suppose this is your family, which 3 shots would you include if you want to generate a summary for it. (That is, you would like to recommend them to other family members and friend for watching)?
   e) Do you prefer the current ShotRank-based browsing tool to the traditional VCR-like player for browsing home video?

Figure 7. Questionnaire used for the bootstrapping stage

Note that although a naive approach that filters out shots directly based on length would achieve a similar goal, it will fail to consider other factors that determine the importance of a shot in a combined way. Besides, determining a hard threshold is almost always difficult.

As to convergence, since the new matrix D is diagonal and so symmetric, the proof in Section 5.1 also works for Equation (5).

6. Evaluation

In this section we describe the experiments and user studies we have done to evaluate our work. We will first discuss the methodology used for the subjective evaluation, and then describe in detail how the user studies are performed and conclude the results.

6.1 Methodology

Because our work deals with human understanding and interest in video clips, such strong subjectivity makes an objective evaluation very difficult. Although almost all users confirmed that our tool is more favorable than traditional VCR-like players for video browsing, we want to have some quantitative measurement. Therefore, we need to have a more concrete definition of ***interesting shots***. For each video clip, we have selected some "significant events" and ask one question aiming at each of these events. The users are requested to do a "video comprehension test" – to browse through the video in search for answers to a questionnaire. Our basic assumption is that there exist segments of a video clip that are commonly interesting to most users, and users might as well browse a video clip in searching for answers to some interesting questions. Therefore, in our experiment, we use some questions to help mimic user interests and focus user behavior. We believe when enough user data is available, user behavior will exhibit similar patterns even if they are not explicitly asked to answer questions.

Our overall evaluation process is shown in Figure 6. Note that our algorithm is based on user interest study and not any special feature of a particular type of video, it will work with any video data as long as it does contain some interesting content commonly attractive to many users. We selected a length of soccer game recording between 2 Japanese teams named *Soccer.mpg* and a piece of home-made video named *Home.mpg*. The soccer game is the first half of a soccer game (about 45 minutes long) and the comments in the audio track are in Japanese, so our Chinese users in this experiment have to rely on the pictures alone. The home video is a 26-minute long clip edited from recording of the celebration activities of an American family during Christmas (including snow skiing, Christmas dinner, gift opening, etc), which

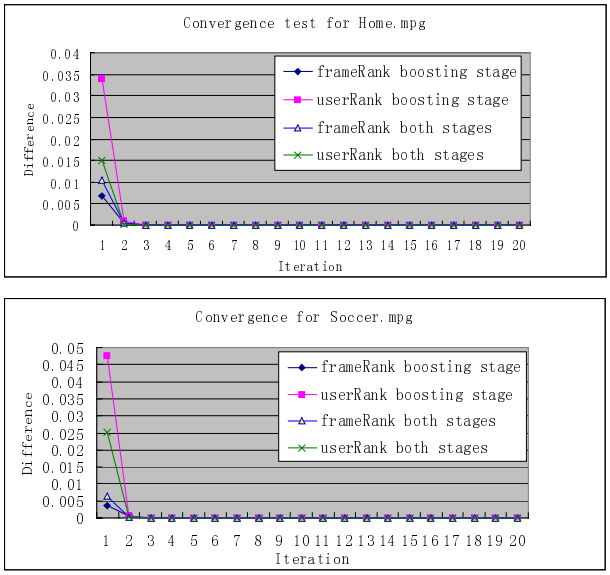

Figure 8. Convergence test

represents normal narrative video clips such as movies and TV programs. These two clips are segmented into video shots by a shot-boundary detection algorithm using low-level features such as color histogram and texture.

At the beginning of evaluation, we do not have any user browsing log yet, so no ShotRank is available. Therefore, we just present the shots to the users as a uniform list. To acquire user knowledge to bootstrap the ShotRank (*Bootstrapping Stage*), we invited 10 users to answer a questionnaire (given in Figure 7 on last page) by browsing through the two videos with our interface. We explained how to use the interface in advance to all users. As they searched for the answers in the video, we noted down all of their interactions with the browser interface. All the users are also requested to answer the questions as fast as possible, and the time they spent to finish the questionnaires was also recorded.

After the 10 users have answered these questions, totally 2440 observations of their browsing behavior are logged, which is used to generate the initial ShotRank for every shot. With this bootstrapping ShotRank, we are able to re-organize the shots into a story-tree as described in section *3.2*, and the ShotRank-based skimming function is also available to new users. Then, we enter the *Enhanced Stage* of the user study, where another 7 subject users are invited to answer the questionnaire. A new question is added to the questionnaire for both video clips: *"Try to use the "ShotRank-based skimming" function. How many top ranking shots do you think are enough to render an informative video skim?"* After the users have answered these questions, totally 1010 observations of their browsing behavior is logged, and this data is used for ShotRank update and shot re-organization.

6.2 Evaluation Result Summary

6.2.1 Speed of Convergence

In both the bootstrapping and the enhanced stages, we executed the iterative ShotRank computation algorithm for each of the two video clips. The difference between consecutive iterations for both ShotRank and UserRank are computed using the following definition of distance: for 2 vectors V_1 and V_2 of dimension N, their distance d is defined in Equation (6):

	Soccer.mpg		Home.mpg	
	Yes	No	Yes	No
Bootstrapping Stage	80%	20%	70%	30%
Enhanced Stage	100%	0%	85%	15%

Table 2. Comparison of user satisfaction

	Soccer.mpg	Home.mpg
Average time spent in bootstrapping stage	21.2 min	15.6 min
Average time spent in enhanced stage	11.8 min	9.5 min
Shrinkage	44.3%	39.1%

Table 3. Comparison of time spent in answering the questionnaire

$$d = \sum_{i=1}^{N} (V_1[i] - V_2[i])^2 \qquad \textbf{(6)}$$

The results are plotted in Figure 8. We can see that the difference between consecutive iterations drops quickly and shows a strong tendency towards zero. Although the user data is limited, this result confirms with our theoretical proof for the convergence of our algorithm, and also indicates fast convergence for large data set.

6.2.2 Skim Usability

6.2.2.1 User Satisfaction

Table 2 shows the users' answer to the question *"Do you prefer the current ShotRank-based browsing tool to the traditional VCR-like player for browsing soccer game/home video?"*

We can see that after the analysis of the user log in the first stage, the ShotRank-based skimming function becomes available, and more users become satisfied with the browsing experience. This confirms that we have successfully mined and utilized the knowledge inside the use log of the bootstrapping stage in generating the new ShotRank for facilitating users in the enhanced stage. In addition, this improvement in user satisfaction is more obvious for the soccer game video, which is longer and has more shots than the home video. This indicates that our scheme is particularly useful for fast browsing when the original video clip is long and the interesting and important information is sparse.

6.2.2.2 Faster Browsing

Table 3 shows the average time spent in browsing the two videos as users searched for answers to the questions. We can see that on average this time has shrank for 44.3% for the soccer game and 39.1% for the home video because of the ShotRank-based browsing function. This is a strong proof that our algorithm has captured important knowledge latent in user log to help future users in their video browsing.

6.2.2.3 Interest-capturing Ability

From the users' answers to the second questionnaire, we have found that the users believe a ShotRank-based skim consisting of only the top 20 or 30 shots are good enough to represent the original video clips. This provides a subjective confirmation of the ability of our solution to automatically capture important and

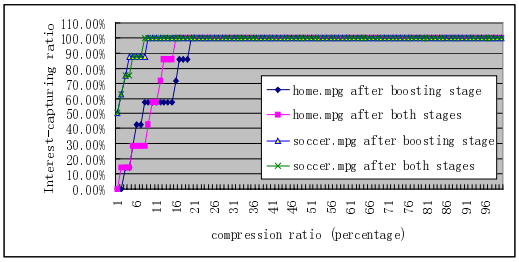

Figure 9. Interest-capturing ratio against compression ratio

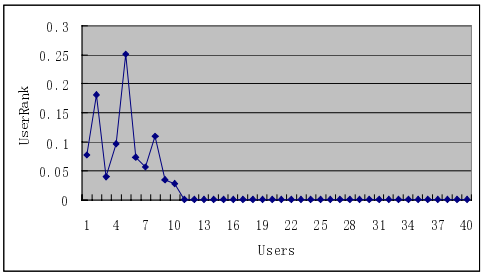

Figure 10. UserRank of 10 good users (1 to 10) and 30 random users (11 to 40)

interesting information that previous users have discovered.

However, we still want to give a more objective measurement on how well the video skim can capture the semantics of video clips. In text summarization, two commonly used metrics are *retention ratio*, which measures how much information has been retained, and *compression ratio*, which measures the ratio in length between the summary and the original video clip. We have modified retention ratio and proposed a new metric called *interests-capturing ratio*, as given in Equation (7):

$$\text{interests-capturing ratio} = \frac{\text{the number of key shots retained in the summary}}{\text{the total number of key shots in the original}} \quad \textbf{(7)}$$

Since key shots are defined based on ground truth, the more key shots retained in the summary, the higher the final interests-capturing ratio. We have generated summaries of various lengths and measured the interests-capturing ratio at each compression ratio, as plotted in Figure 9. We can see that even at a quite aggressive compression ratio below 20%, the interest-capturing ratio is already above 85%. Also, it is interesting to see that if we use the log of both the bootstrapping stage and the enhanced stage together, even higher interest-capturing ratio could be reached than using the bootstrapping stage log alone at the same compression ratio. This demonstrates the "focusing effect" that future users' attention is more focused on those shots already with high ranking. As to the places where the bootstrapping stage works better, we believe it because the compression ratio is too low, and the few key shots' ShotRank are too close to overcome noises.

6.2.3 Comparison with DirectHit

Since DirectHit [15] is a very popular and straightforward algorithm for log mining, it is not obvious why one would want to use more complicated algorithms, such as the one proposed in this paper. As we mentioned above, a major strength of our iterative algorithm is that it could utilize the reinforcement relation between engaged users and interesting shots when ranking the shots, and we have also done an experiment to demonstrate this hypothesis.

To be fair to DirectHit, we only used the log of the 10 users in the bootstrapping stage, which has more noises and the users have not benefited from the ShotRank-based skimming yet. In addition, we assume that another 30 "random" users have also browsed the same soccer game video. They were hasty and not interested in the soccer game, so they only watched the first 3 shots and then left. We believe this should be a common phenomenon for Video-On-Demand applications, and we want to see how much impact they have on both our algorithm and the DirectHit algorithm.

We implemented the DirectHit algorithm by increasing the rank of a shot each time it is accessed. Table 4 shows the indices of the top 10 shots according to the rank generated by our algorithm and by DirectHit before and after the introduction of the 30 random users. All the video shots that contain the answers to our questions are taken as the ground truth. Since there is no explicit ranking between them, we just list them according to the shot index sequence. From the table we could see that without the noises

Ranking	Ground Truth	Before		After	
		Our Algorithm	DirectHit	Our Algorithm	DirectHit
1	**Shot 3**	**Shot 100**	**Shot 205**	**Shot 100**	Shot 2
2	**Shot 4**	**Shot 205**	**Shot 100**	**Shot 255**	Shot 1
3	**Shot 100**	**Shot 255**	**Shot 4**	**Shot 205**	Shot 0
4	**Shot 101**	**Shot 4**	**Shot 255**	**Shot 4**	**Shot 205**
5	**Shot 204**	**Shot 3**	Shot 303	**Shot 3**	**Shot 100**
6	**Shot 205**	**Shot 101**	**Shot 3**	**Shot 101**	**Shot 4**
7	**Shot 206**	Shot 339	Shot 206	Shot 339	**Shot 255**
8	**Shot 255**	**Shot 204**	**Shot 339**	**Shot 204**	Shot 303
9	**Shot 356**	**Shot 341**	**Shot 204**	**Shot 341**	**Shot 3**
10	**Shot 341**	Shot 210	Shot 340	Shot 210	Shot 206
Precision	N/A	80%	70%	80%	50%

Table 4 Comparison of top 10 shots ranked by our algorithm and DirectHit before and after introduction of 30 "random" users

from the random users, both our algorithm and DirectHit perform similarly well (80% vs. 70%). When the log of the 30 random users are considered, our algorithm is not affected by the random users' behavior at all, even though they 3 times out numbered the engaged users (30 vs. 10). However, the results given by the DirectHit algorithm become poorer, with all 3 interfering shots 0, 1 and 2 being listed as top 3 and a precision of only 50%. This could be clearly explained if we take a look at the UserRank for the 40 users given by our algorithm in Figure 10.

We can see that the random users have a very low UserRank value, and this determines that their opinions are not so important in the final judgment of our algorithm. On the other hand, those engaged users gain their rank by picking out many interesting shots, and they collaboratively determines which shots to rank high at last.

7. Conclusion and Future Work

As computer technologies evolve towards the new era of *user-centric model*, collecting digital footprints from user actions for better autonomous and intelligent video analysis becomes increasingly more important. This motivates us to develop a video browsing/skimming system that utilizes users' browsing behavior for semantic understanding of video and improved video summarization. Based on a "shots and links" view of the continuous video content, we propose a novel *ShotRank* framework to measure significance and interestingness of each video shot. ShotRank is computed through a link analysis algorithm that utilizes the voting of users on the subjective significance and interestingness of each shot. The reinforcement relation between "engaged" users and "interesting" shots also helps to filter out the effect of random users. ShotRank can serve as a prioritizing mechanism in organizing video shots and generating abstract presentations such as video skims. User study results have confirmed that ShotRank does capture the subjective notion of importance of each video shot, and it significantly improves the future users' browsing experience.

There are many directions we will explore in the future. First, we will conduct more extensive experiments with larger user base to evaluate our framework without the help of the questionnaire and compare the hierarchical story tree with skim-based re-presentation. Second, there are several tuning parameters used in our algorithms, and we will look into how to train them by (un)supervised learning. Third, we want to incorporate our results into distributed video applications, and extend our model into a general framework for multimedia information summarization. Finally, we are aware of the privacy concerns on mining user action log, but the discussion is out of the scope of this paper.

8. ACKNOWLEDGMENTS

The authors would like to thank Zheng Chen for his suggestions to our work; all friends from MSR Asia who helped us to do the evaluation during the first author's intern at MSR Asia; the reviewers and professor James Wang for their insightful comments; and NSF for the ITR grant that the first author is funded for completing this work.

9. Reference

[1] A. Hanjialic and H.J. Zhang, An Integrated Scheme for Automated Video Summarization Based on Unsupervised Cluster-Validity Analysis, in IEEE Transactions on Circuits and Systems for Video Technology, VOL. 9, NO. 8, PP. 1280-1289, December 1999.

[2] D. DeMenthon, V. Kobla and D. Doermann, in Proc. of Video Summarization by Curve Simplification, ACM MM 1998, United Kingdom, September 1998.

[3] M. Smith and T. Kanade, Video Skimming for Quick Browsing Based on Audio and Image Characterization, Carnegie Mellon University, Technical Report CMU-CS-95-186, July 1995.

[4] H.J. Zhang, C.Y. Low, S.W. Smoliar and J.H. Wu, Video Parsing, Retrieval and Browsing: An Integrated and Content-Based Solution, in Proc. of ACM MM 1995, California, November 1995.

[5] V. Kobla and D. Doermann, VideoTrails: Representing and Visualizing Structure in Video Sequences, in Proc. of ACM MM 1997, Seattle, November 1997.

[6] H. Aoki, S. Shimotsuji and O. Hori, A Shot Classification Method of Selecting Effective Key-Frames for Video Browsing, in Proc. of ACM MM 1996, Massachusetts, November 1996.

[7] Y. Rui, A. Gupta and A. Acero, Automatically Extracting Highlights for TV Baseball Programs, ACM MM 2000, California, November 2000.

[8] S. Uchihashi, J. Foote, A. Girgensohn and J. Boreczky, Video Manga: Generating Semantically Meaningful Video Summaries, in Proc. of ACM MM 1999, Florida, October 1999.

[9] J. Nam and A. H. Tewfik, Dynamic Video Summarization and Visualization, ACM MM 1999, Florida, October 1999.

[10] M. M. Yeung and B. L. Yeo, Time-constrained Clustering for Segmentation of Video into Story Units, in Proc. of International Conference on Pattern Recognition, August 1996.

[11] L.W. He, E. Sanocki, A. Gupta and J. Grudin, Auto-Summarization of Audio-Video Presentations, in Proc. of ACM MM 1999, Florida, October 1999.

[12] R. Lienhart, Abstracting Home Video Automatically, in Proc. of ACM MM 1999, Florida, October 1999.

[13] Y.F. Ma, L. Lu, H.J. Zhang and M. Li, An Attention Model for Video Summarization, in Proc. of ACM MM 2002, France, December 2002.

[14] Informedia Project, http://www.informedia.cs.cmu.edu/

[15] DirecHit: http://www.directhit.com

[16] S. Acharya, B. Smith, and P. Parnes, Characterizing User Access To Videos On the World Wide Web, in Proc. of Multimedia Computing and Networking 2000, California, January 2000.

[17] T. F. Syeda-Mahmood and D. Ponceleon, Learning Video Browsing Behavior and its Application in the Generation of Video Previews, in Proc. of ACM MM 2001, Canada, October 2001.

[18] X. Zhu, J. Fan and A.K. Elmagarmid, Hierarchical Video Summarization and Content Description Joint Semantic and Visual Similarity, in ACM Multimedia Systems, VOL 8, NO 5, 2003.

[19] S. Brin and L. Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, in Proc. of 7th International WWW Conference, Australia, April 1998.

[20] Y. Wang, P. Zhao, D. Zhang, M. Li and H.J. Zhang, MyVideos - A System for Home Video Management, in Proc. of ACM MM 2002, France, December 2002.

[21] H. J. Zhang, A. Kankanhalli, S. W. Smoliar, Automatic Partitioning of Full-motion Video, in ACM Multimedia Systems, VOL. 1, NO.1, PP. 10-28, 1993.

[22] P. Bouthemy, Y. Dufournaud, R. Fablet, R. Mohr, S. Peleg, and A. Zomet, Video Hyper-links Creation for Content-Based Browsing and Navigation, in Proc. of Workshop on Content-Based Multimedia Indexing, France, October 1999.

[23] W.-Y. Ma and H.J. Zhang, An Indexing and Browsing System for Home Video, In Proc. of European Conference on Signal Processing, Greece, September 2000.

[24] K. Ntalianis, A. Doulamis, N. Doulamis, and S. Kollias, Non-Sequential Video Structuring Based on Video Object Linking: An Efficient Tool for Video Browsing and Indexing, in Proc. of IEEE Int. Conf. Image Processing, Greece, October 2001.

[25] Z. Chen, L. Tao, J. Wang, W. Liu and W.Y. Ma, A Unified Framework for Web Link Analysis, in Proc. of WISE 2002, Singapore, December 2002

[26] J. Kleinberg, Authoritative Sources in a Hyperlinked Environment, in Journal of ACM, VOL. 46, NO. 5, PP. 604--632, 1999.

[27] H. Zhang, S. Smoliar and J. Wu. Content-based Video Browsing Tools, in A. Rodriguez and J. Maitan, editors, Symposium on Electronic Imaging Science and Technology: Multimedia Computing and Networking, SPIE VOL 2417, PP. 389--398, 1995.