Video Summarization Using Greedy Method in a Constraint Satisfaction Framework

Lu Shi, Irwin King, and Michael R. Lyu
Department of Computer Science and Engineering
The Chinese University of Hong Kong
Shatin, N.T., Hong Kong SAR
{slu, king, lyu}@cse.cuhk.edu.hk

Abstract

Video resources are pervasive nowadays. Since it is time-consuming for the user to download and browse videos, video summarization techniques, which aim at providing a way for the user to grasp the major content of the video without browsing the whole video, have become more and more important. In this paper, we present a feature-based video summarization framework based on constraint satisfaction programming. We transform the feature distribution and the user requirements to a set of constraints to be satisfied. We will show some experimental results and discuss the effectiveness and efficiency of our solution. We have also developed a system to generate video summaries with the suggested constraint-satisfaction programming framework.

1 Introduction

With the rapid growth of network bandwidth and high-capacity storage devices, videos have become pervasive nowadays. However, the abundance of video data gives rise to a new challenge: since it is time-consuming to download and browse most parts of a video before we know the video contents, it is difficult to find out a video file that we want from the vast video repositories. Moreover, in some cases in which we do not have enough bandwidth to stream all the original video, e.g, some wireless hand-hold devices using MMS (Multimedia message services), a digest version of the video is needed to meet the limited bandwidth. Therefore, in order to help the user to grasp the essence of the video quickly, *video summarization* has received more and more attention.

Video summarization is a short summary of the content of a longer video document. Specifically, it is a condensed sequence of still or moving images representing a video in such a way that the user is provided with concise presentation of the video content. There are two different kinds of video summarizations:

1. **Still-image Storyboard**– The still-image representation is composed of a collection of salient images extracted or synthesized from the underlying video source. For example, some MMS service provides video summarization services to the users by delivering still-image storyboard accompanied with audio track.

2. **Moving-image Skimming**– The moving-image representation, also called video skimming, is made up of a set of video clips. Movie trailer is a good example for moving-image skimming.

To generate a perfect video summary requires good understanding of the video semantic content. However, understanding the semantic content of the video is still far beyond the capability of today's intelligent systems, despite the significant advances in computer vision, image understanding, and pattern recognition algorithms. So, we can only rely on some low-level features to generate video summaries.

In this paper we propose a novel video summarization framework based on constraint-satisfaction programming (CSP). The contributions of this paper are:

1. **Flexible Framework**– The proposed system is based on constraint satisfaction programming which allows the user to add constraints easily.

2. **Multiple Solutions**– As a result of CSP, users are able to view various possible solutions that will be most suitable for their needs.

This paper is organized as follows. In the next section we review some related work done on video summarization field. In Section 3 we model our feature-based moving-image video summarization generation problem as

a constraint-satisfaction problem and suggest an intermediate solution for it. Experiment results are shown and analyzed. In Section 4, we briefly describe the video summarization system we have designed and implemented. In Section 5 we make conclusion and discuss our future work.

2 Related work

In recent years much work has been done on video summarization. For still-image-based story boards, most of the early work selects key frame images by random or uniform sampling, like the MiniVideo systems [1]. Later work tends to extract key frame images by adapting to the video content. According to the applied features, we can categorize the methods into color-based approach like [2, 3], motion-based approach like pixel-based image difference [4], optical flow [5], and mosaic-based approach [6]. In [7] and [8], the authors did video segmentation and analyzed the video structure to get a tree-structured Video-Table-Of-Contents. In [9], the authors proposed an importance measure for each selected frame, and a frame packing algorithm to adjust the shown size of the frames according to their importance. Still image story boards can be constructed faster but their descriptive ability is limited for they cannot convey the temporal properties of the original video, and in many cases the audio information is not preserved.

Much effort is also devoted to generating moving-image based video summaries. In the VAbstract system [10], key movie segments are selected to form a movie trailer. The Informedia system at CMU first generates caption text from the audio stream of the video by speech recognition, then selects the video segments according to the occurrence of important keywords in the video sequence [11]. However, selecting video shots solely by the occurrence of keywords cannot ensure that the video summary will cover the main idea of the original video. The CueVideo system from IBM provides faster playback speed when playing long, static video scenes and slower speed for short, dynamic video scenes [12]. Although the playback time has been reduced but the temporal property of the original video is distorted, which may mislead the users. In broadcasted sports video, highlight detection and extraction have been achieved in basketball videos [13] and baseball videos [14]. Highlight detection is highly dependent on domain-specific knowledge. The methods listed above may not always yield perceptional important video summaries, and they leave the users with only limited choices.

To ensure the quality of the video summary, we transform the users' preference into constraints then find results that satisfy all of them. Since the method enables the users to specify parameters according to their preference, it is much more flexible than the previous methods. Another merit of this framework is that it is easy to expand by adding new constraints into the existing framework.

3 Feature based video summarization

The video summarization procedure has two objectives: the first is to find the most important video segments of the original video, the second is to shorten the original video to the target length. To ensure the quality of the generated summary, we specify some video features to measure the importance of the video segments then generate the video summarization according to the feature distribution.

3.1 Problem definition

Here we propose a framework for video summarization based on constraint satisfaction programming (CSP). We will model the vide summarization as a constraint satisfaction problem, give a formal definition to it and describe an algorithm to solve it. Experimental results will be provided to demonstrate that the algorithm works.

Definition 3.1 *A video is defined as a sequence of images. It is represented by* $V = (I_1, I_2 ... I_n)$*, where* I_i *is the* i*-th indexed frame image of the video and* n *is the number of total frames in the video* V*. The video can also be represented in the short form as* $V = [I_{begin}, I_{end}]$ *where* $1 \leqslant begin \leqslant end \leqslant n$*. Moreover, the length of a video,* $length(V)$ *is* $end - begin + 1$*.*

Definition 3.2 *The feature score function* f_i *of a frame is a function over the whole frame image set* I*. Without loss of generality, we assume these functions to be non-negative. It can be discrete, which indicates the occurrence of a feature of interest like human face; or it can be continuous to describe the magnitude of the feature like noise volume, percentage of the interesting color, etc.*

Definition 3.3 *An extracted video is defined as a mapping of a video with a set of feature functions to a set of video clips through a video feature extraction algorithm. This can be written as* $h : V \times F \rightarrow \{V_i\}$*, where* F *is a nonempty set of feature functions and* V *is a set of zero or more of valid video clips. If some items in* $\{V_i\}$ *are overlapping, i.e.,* $V_i \bigcap V_j \neq \emptyset, i \neq j$*, we call this the overlapping extracted video. Otherwise, we call it the non-overlapping extracted video. We define the length of an extracted video to be the summation of all the elements' length in the extracted video.*

We can formulate the process to find the final extracted video as a constraint satisfaction problem. A constraint satisfaction problem is defined as:

- A set of variables $X = \{x_1, x_2 ... x_n\}$;
- For each variable, a set D_i of possible values (domain);

- A set of constraints restricting the values that the variables can simultaneously take;

We can define the set of constraints as follows:

$$\begin{aligned} f_1(x_1, x_2, \cdots, x_n) &> g_1(x_1, x_2, \cdots, x_n) \\ f_2(x_1, x_2, \cdots, x_n) &> g_2(x_1, x_2, \cdots, x_n) \\ &\vdots \\ f_k(x_1, x_2, \cdots, x_n) &> g_k(x_1, x_2, \cdots, x_n) \end{aligned}$$

Here, $f_i()$ is the feature extracted from the video and $g_i()$ is the parameter for the constraint. Normally, A constraint is in an inequality form, and $g_i()$ sets the threshold, range, or valid elements for the variables. For example, consider the audio volume. Here $f_i()$ is the actual audio volume, $g_i()$ can simply be a threshold so that the user could cut out those frames with a volume greater than a constant db, or $g_i()$ can be a range so that only those frames with a volume within the desired range is selected. Many other features can be added as constraints, e.g caption text occurrence, human face detection, some high-level semantic features like commercial detection and dialogue detection. Finally, the length of the video summary must not exceed the pre-specified time threshold, T. The set of constraints defines a final video summary which satisfies all the constraints, or satisfies as many constraints as possible. The domain of the frame index variable x is from 0 to the whole length of the video. In real applications, the users may adjust the parameters if they are not satisfied by the summary, and such iteration continues until the users are satisfied.

From the feature score function set we can construct an importance measure for each video frame. The importance measure function is defined as follows:

Definition 3.4 *The importance measure function of a video frame is a function over the whole image set I, which indicates the importance of the image. Here we employ the feature score summation as the importance function:*

$$IP(I_i) = \sum_j f_j(I_i)$$

After we have obtained the features, we aim to get the final video summary that mostly represents the content of the original video. To achieve this, we can select to maximize the feature score summation of the video summary. Then the problem can be formalized as described in Problem 3.1:

Problem 3.1 *Given a set of video features, a time threshold T, and $\{V_i\}$ obtained from a set of video feature extraction functions, find the final video summary, $\{V_{final}\}$, an non-overlapping extracted video, such that it maximizes the total feature score summation and $length(\{V_{final}\}) = T$.*

Obviously, a smoother video summary may seem better than a jumpy one to the users. To do so we need to decrease the number of video segments in the video summary to make it more coherent. To describe the smoothness of the extracted video summary, we define the transition of an extracted video as follows:

Definition 3.5 *The transition in the non-overlapping extracted video is defined as the cardinality of the video as $\#(\{V_i\})$. It is the number of segments in $\{V_i\}$.*

With the above definition we can extend Problem 3.1 as:

Problem 3.2 *Given a set of video features, $\{V_i\}$ obtained from the a set of video feature extraction functions, find the final video summary, $\{V_{final}\}$, an non-overlapping extracted video such that $length(\{V_{final}\}) = T$. Moreover, it maximizes the total feature score and minimizes the transition number.*

3.2 Solution and algorithm

To solve Problem 3.1 and Problem 3.2, we need to solve the constraints. We may solve the constraints with various methods like greedy method, lagrange multiplier, dynamic programming, etc. Here we use a straight-forward greedy method to solve Problems 3.1. We first calculate the feature score summation value for each video frame, then sort them according to their feature score values, and select the T frames with highest scores then we get an extracted video set $\{V_i\}$ as the video summary. The time complexity for the greedy method is $O(n \log(n))$, which is the complexity of the sorting process.

We can solve Problem 3.2 by refining the video summary $\{V_i\}$ that we get by the greedy method. To minimize the transition number in video summary $\{V_i\}$, we need to decrease the granularity of the selected segments to make it smoother. We perform a segment merging process to minimize the number of the transitions in the extracted video summary. Each time we find the two nearest selected video segments and join them to form a longer and more coherent segment until the extracted video is smooth enough. Let $\{E_i\}$ be the set of unselected segments, we introduce a penalty function to measure the granularity of the $\{V_i\}$ as

$$p(\{V_i\}, \{E_i\}) = \sum_j \frac{\sum_t length(E_t)}{length(E_j)}.$$

Consequently, we derive the new importance function as:

$$N(\{V_i\}, \{E_i\}) = f(\{v_i\}) - w \cdot p(\{V_i\}, \{E_i\}),$$

where $f()$ is the feature score summation function for the selected segments $\{V_i\}$, $p()$ is the penalty function, and w is the weight of the penalty function. Our goal is to maximize the function N.

To maximize function N, we propose a search algorithm based on the segments generated by the greedy method as described below: Note that during the 4th step in the repeat loop, we merge the two adjacent selected segments in the

Algorithm 1 Video summary refinement algorithm

Find the result segments $\{E_i\}, \{V_i\}$ with the greedy method;
repeat
1. $TEMPTVALUE = N(\{E_i\}, \{V_i\})$;
2. Find the shortest unselected segment e_i;
3. Find the adjacent selected segment s_1, s_2 of e_i;
4. Merge s_1 and s_2, so the shortest unselected segment e_i is eliminated. Now we get updated $\{E_i\}$,$\{V_i\}$.
5. Calculate $N(\{E_i\}, \{V_i\})$ for updated $\{E_i\}$,$\{V_i\}$.

until $N(\{E_i\}, \{V_i\}) < TEMPTVALUE$
Undo the last merge;
The final smoother video summary is found.

way that we move the segment with smaller feature score values, so that the loss of feature score is minimized.

When we continue merging the unselected segments, the penalty function p changes with the reciprocal of the unselected segments' length. As the merge process continues, the length of the shortest unselected segments becomes longer and its reciprocal becomes smaller. Thus function p decreases quickly and the refinement process soon converges. The complexity of the merge process would not be greater than $O(n\log(n))$. So we can still regard the complexity of the whole process as $O(n\log(n))$.

3.3 Experiments and Discussions

To test the performance of our video summarization procedure, we implemented the proposed algorithm and tested it on some video clips. Our experiment consisted of a subjective test and a quantitative test. We employed a PC platform with 2.0G hz P4 CPU on the Win2000 OS. We developed a system to perform the summarization procedure and played the video summary. We will briefly describe the system in the next section.

We selected two types of video clips for our summarization experiment. The first set was news video clips, the other set was movie clips. We select these two types of video because they are quite pervasive and representative. Our test includes ten news video clips that were about 1 to 2 minutes long, and three movie clips that were about 7 to 10 minutes long. We show the experimental results for one sample video from each type of the videos.

For the news video clips, we selected the following features for video summarization: human face detection, male/female voices occurrence, camera zooming and caption text occurrence. Table 1 shows the selected features and the corresponding parameters specified when we engaged summarization to one of the video clips. Note that the weights here are normalized so that $\sum_i Weight_i = 1$.

The feature distribution and the finally generated segments for a sample news video are shown in Fig. 1. The horizontal axes denote the time.

Table 1. Parameters for news clips

Features	Weight
f_1:Face occurrence	0.30
f_2:Text occurrence	0.20
f_3:Voice	0.20
f_4:Camera zooming	0.30
Parameters	**Value**
Original length	95 sec
Summary length	22 sec
w	20.0

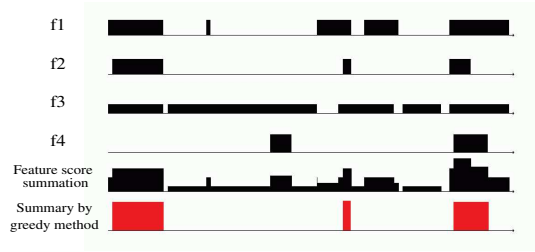

Figure 1. Result for news video

From Fig. 1 we can see that the feature distribution and the selected segments are all quite long and coherent, so that no refinement process is needed. This is because the news video itself is well structured and composed with longer video shots. We played the summary and observed that it did contain the major content of the news story. Tests on most other news video clips yielded similar results.

For the movie clips, we selected the following features for video summarization generation: human face occurrence, loud human voices/cries, loud noises like gun shots and explosion, and the color of fire. Table 2 shows the selected features and the corresponding parameters specified in the experiment.

Table 2. Parameters for movie clip

Features	Weight
f_1:Gunshot/explode noise	0.45
f_2:loud voice	0.25
f_3:Face occurrence	0.10
f_4:Fire Color	0.20
Parameters	**Value**
Original length	477 sec
Summary length	50 sec
w	20.0

The feature distribution and the finally generated segments are shown in Fig. 2.

From Fig. 2 we can see that in movies, shot cut and feature change more frequently than in the news video, es-

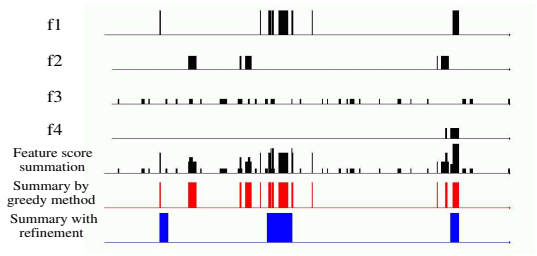

Figure 2. Result for movie clip

pecially for the action movies. The result of the greedy method contains many broken short segments, which causes jumpy scenes. Consequently, the refinement process is needed for better results. The refinement process generated several longer and more coherent video segments from the short fragments yielded by the greedy method. From Fig. 2 we can see that those refined longer segments still cover most of the parts with the highest feature scores in the feature summation distribution graph. Moreover, they matched the segment clusters generated by the greedy method quite well. We played the summary and observed that it still covered the major key events of the original movie.

To demonstrate the capability of our framework to generate multiple video summaries, we change the weight for some specific features, for example, if we prefer more on human faces we can increase its weight then more contents with human face should be selected into the video summary. Table 3 shows the effects resulted from changing the weight for the "face occurrence" feature.

Table 3. Changing the weight for face occurrence

Weight	0.2	0.4	0.6	0.8
Rate	0.24	0.32	0.54	0.88

To quantitatively evaluate our video summarization algorithm, we invited some people to manually select a video summary, and we made comparisons between our machine-generated video summary and the video summary selected by them. Both video summaries were produced with the same target length. Suppose the human-selected video summary be $\{U_i\}$, and the machine-generated video summary is $\{V_i\}$, we define the *Common Selection Ratio* (CSR) to measure the quality for our generated video summary.

Definition 3.6 *The common selection ratio*(CSR) *is defined as*

$$CSR(\{U_i\},\{V_i\}) = \frac{length(\{U_i\} \bigcap \{V_i\})}{length(\{V_i\})}.$$

After comparing between the video summaries, we recorded the results shown in Table 4 and Table 5.

Table 4. CSR experiment results with greedy method

Video type	**Clips No.**	**Ave. CSR**	**Max CSR**	**Min CSR**
News	10	0.89	0.94	0.82
Movie	3	0.74	0.84	0.68

Table 5. CSR experiment results with refinement

Video type	**Clips No.**	**Ave. CSR**	**Max CSR**	**Min CSR**
News	10	0.89	0.94	0.77
Movie	3	0.78	0.84	0.71

From Table 4 and Table 5 we can see that the video summaries generated by our algorithm are quite similar to those selected by human. Furthermore, we can see that the algorithm works better for news videos than for the movie clips. This is partially because the structure of the news videos is simpler. The result for movie clips seems to be relatively worse, as the human can semantically interpret the movie before selecting those video shots, while our algorithm selects the video shots with higher feature scores and neglect those meaningful video shots without high video feature scores. To get better results, we will enhance our importance measure for the video frames by integrating more video features, especially some high level semantic features into this framework in the future.

From the data we can also see that the refinement process really makes a smoother video summary without much loss of the important contents of the original video. In some cases, it may generate a video summary which is even more similar to a human selected video summary than the greedy method does. We also have observed that the human-selected video segments are at least 2 seconds long, which also suggests the necessity of the refinement process.

As we have mentioned, the time-complexity of the video summarization procedure is $O(n\log(n))$, which enable the video summarization process to run very quickly. This allows the users to quickly generate various sets of video summaries with different parameters for comprehensive investigation and experimentation.

4 Overview of the video summarization system DImSSum

In this section we briefly introduce DImSSum (Digital IMage Sequence SUMmarization) system, which is developed based on the constraint-satisfaction programming framework discussed in the previous sections.

The system architecture of DImSSum is shown in Fig. 3. It consists of a video preprocessing module, a constraint solver module, a video streaming server, and a web-based interface for the users to see the video summary.

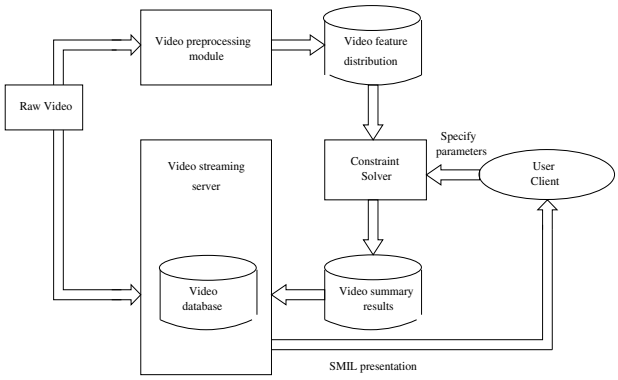

Figure 3. System architecture of DImSSum

The video preprocessing module is employed for video feature extraction. It is an off-line module operated by the administrator. Currently we have integrated face detection, voice detection, color histogram analysis and VOCR into this module.

The constraint solver module receives the parameters from the users then generates a video summary according to the procedure laid out in solving a constraint satisfaction problem.

The video summarization service is provided to the users via Web service. The system receives the users' requests and parameters, generates the result video summary, then transforms it into SMIL (Synchronized Multimedia Integration Language) presentations, which can be played by some browsers that support SMIL like the RealOne Player. The users may adjust the parameters to generate new video summaries until they are satisfied.

5 Conclusion and future work

In this paper, we modelled the video summarization problem as a constraint-satisfaction programming problem and proposed a feature-based greedy method solution to generate moving video summaries. A refinement process is proposed to make a smoother summary. We also conducted some experiments to evaluate our proposed algorithms, and we described our video summarization system DImSSum. The initial experimental results were encouraging.

The future work includes the following aspects:

1. **More features and more constraints**– The suggested problem solution framework can be extended by adding informative features, especially some high-level semantic features, video structure and style, or by extending the object function with some new importance measures on the selected video segments. The refinement process can also be further improved. Besides the greedy method, we will try other constraint-solving methods to solve the constraints then get the final video summary.

2. **Minimal summarization limit**– As the video summarization length T decreases, the quality of the video summary becomes worse. Therefore, for a specified video file, there may exist a lower bound time limit for a meaningful video summary, which is determined by the content and structure of the video. Finding the lower bound of the video summary length T_{min} within which we can still ensure the quality of the video summary is another problem left for us for further study.

References

[1] Yukinobu Taniguchi, Akihito Akutsu, Yoshinobu Tonomura, and Hiroshi Hamada. An intuitive and efficient access interface to real-time incoming video based on automatic indexing. In *Proceedings of the third ACM international conference on Multimedia*, pages 25–33, 1995.

[2] H. J. Zhang, C. Y. Low, and S. W. Smoliar. Video parsing and browsing using compressed data. *Multimedia Tools and Applications*, 1:89–111, 1995.

[3] H. J. Zhang, D. Zhong, and S. W. Smoliar. An integrated system for content-based video retrieval and browsing. *Pattern Recognition*, 30(4):643–658, 1997.

[4] W. Wolf. Key frame selection by motion analysis. In *Proceeding IEEE ICASP96*, 1996.

[5] R. L. Lagendijk, A. Janjalic, M. Ceccarelli, M. Soletic, and E. Persoon. Visual search in a smash system. In *Proceeding of IEEE ICIP*, 1996.

[6] M. Lee, W. Chen, C. Lin, C. Gu, and T. Markoc. A layered video object coding system using sprite and affine motion model. *lEEE Transactions on Circuits and Systems for Video Technology*, 1:130–145, 1997.

[7] Y. Rui, T.S. Huang, and S. Mehrotra. Constructing table-of-content for videos. *ACM Multimedia Systems Journal, Special Issue Multimedia Systems on Video Libraries*, 7(5):359–368, Sept 1999.

[8] Ng Chung Wing, Michael R. Lyu, and Irwin King. Advise: Advaced digital video information segmentation engine and its applications. In *Proceeding of World Wide Web*, 2002.

[9] Shingo Uchihashi and Jonathan Foote. Summarizing video using a shot importance measure and a frame-packing algorithm. In *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing*, volume 6, pages 3041–3044, 1999.

[10] R. Leinhart, S. Pfeiffer, and W. Effelsberg. Video abstracting. *Communication of the ACM*, pages 55–62, December 1997.

[11] M. A. Smith and T. Kanade. Video skimming and charaterization through the combination of image and language understanding techniques. In *Proceeding of the IEEE Computer Vision and Pattern Recognition*, pages 775–781, 1997.

[12] D. Ponceleon and A. Amir. Cuevideo: Automated multimedia indexing and retrieval. In *Proceeding of ACM Multimedia*, 1999.

[13] Y. Rui, A. Gupta, and A. Acero. Automatically extracting highlights for tv basketball programs. In *Proceeding of ACM Multimedia*, pages 105–115, 2000.

[14] N. Bagaguchi. Generation of personalized abstract of sports video. In *Proceeding of IEEE ICME*, pages 800–803, 2001.