Space-Time Video Montage

Hong-Wen Kang* Yasuyuki Matsushita† Xiaoou Tang† Xue-Quan Chen

University of Science and Technology of China
Hefei, P.R.China
{hwkang@mail.,chenxq@}ustc.edu.cn

Microsoft Research Asia†
Beijing , P.R.China
{yasumat,xitang}@microsoft.com

Abstract

Conventional video summarization methods focus predominantly on summarizing videos along the time axis, such as building a movie trailer. The resulting video trailer tends to retain much empty space in the background of the video frames while discarding much informative video content due to size limit. In this paper, we propose a novel space-time video summarization method which we call space-time video montage. The method simultaneously analyzes both the spatial and temporal information distribution in a video sequence, and extracts the visually informative space-time *portions of the input videos. The informative video portions are represented in volumetric layers. The layers are then packed together in a small output video volume such that the total amount of visual information in the video volume is maximized. To achieve the packing process, we develop a new algorithm based upon the first-fit and Graph cut optimization techniques. Since our method is able to cut off spatially and temporally less informative portions, it is able to generate much more compact yet highly informative output videos. The effectiveness of our method is validated by extensive experiments over a wide variety of videos.*

1. Introduction

The rapid increase of the amount of on-line and off-line video data necessitates development of efficient tools for fast video browsing. Video summarization [6, 14, 13] is one approach toward tackling this problem, in that it automatically creates a short version of the original input video. Summarized video content is important for many practical applications such as archiving 24-hour security videos and providing easy access to large sets of digital video documentaries.

This paper addresses the problem of automatically synthesizing a new short/small video from a long input video sequence by extracting and fusing the *space-time* informative portions of the input video. Unlike prior video summarization methods, our method is not limited to the frame-basis, but uses an arbitrary space-time volume that is extracted as the informative video portion. The extracted 3D informative video portions are packed together in the output video volume in a way in which the total visual information is maximized. This approach generates a compact yet highly informative video that tries to retain most informative portions of the input videos.

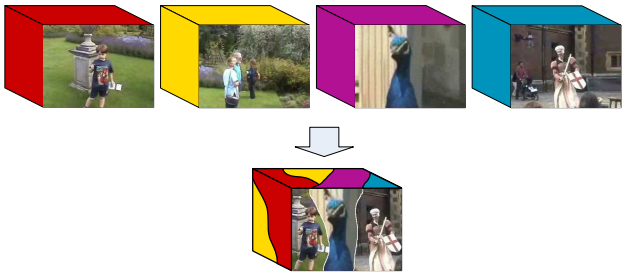

Figure 1. Idea of the space-time video montage.

Prior works on video summarization [6, 14, 13] have typically been based on a two-step approach. First, video streams are divided into a set of meaningful and manageable segments called shots. Then key frames were selected according to criteria from each video shot to generate a summary video. Although these approaches can extract some basic information of the video, they suffer a common disadvantage. They are all frame-based, i.e. they treat a video frame as a non-decomposable unit. Therefore, the resulting video tends to appear to be a fast-forward version of the original video while retaining a large amount of empty space in the video frame background.

Our approach is built upon the idea that some space-time video portions are more informative than others. Considering that visual redundancy exists in videos, the assumption is apparently correct. However the definition of "informa-

*This work was done while the first author was visiting Microsoft Research Asia.

0-7695-2646-2/06 $20.00 (c) 2006 IEEE

tive" is not straightforward since it involves the problem of image understanding. There has been extensive work aimed at extracting salient image/video parts [11, 8, 15]. In general, video summarization methods try to determine important video portions while relying on studies of pattern analysis and recognition.

In our method, we extract and represent space-time informative video portions in volumetric layers. The idea of layered representations was introduced by Wang *et al*. [16] in computer vision and has been widely used in many different contexts [2, 17]. The layered representation has often been used for describing regional information such as foreground and background or different objects with independent motion. Here, we use the layered representation for depicting saliency distribution. A layer is assigned to each high-saliency video portion, each of which represents a different saliency distribution.

1.1. Proposed approach

We try to develop an effective video summarization technique that can generate a highly informative video, in which the space-time informative portions of the input videos are densely packed.

This paper has three major contributions:

Space-time video summarization. Our method treats the informative video portions as a space-time volume without being limited by a per-frame basis. It allows us to develop a new video summarization method which can generate compact yet highly informative summary videos.

Layered representation of informative video portions. We propose an idea of representing informative video portions in the form of volumetric layers such that each layer contains an informative space-time video portion. We call the volumetric layer a saliency layer. The saliency layers are used to compute the optimal arrangement of the input video portions for maximizing the total amount of information in the output video.

Volume packing and merging algorithm. To achieve the goal of packing the informative video portions, we develop a new 3D volume packing and merging algorithm based upon the first-fit algorithm [5] and Graph cut [4] optimization technique. Using this method, the saliency layers are packed and merged together in the output video volume to generate the final montaged video.

In the rest of the paper, we first formulate the problem of space-time video montage in Sec. 2. In Sec. 3, we describe the detailed algorithm of our space-time video montage method. We show the experimental results in Sec. 4 followed by conclusions in Sec. 5.

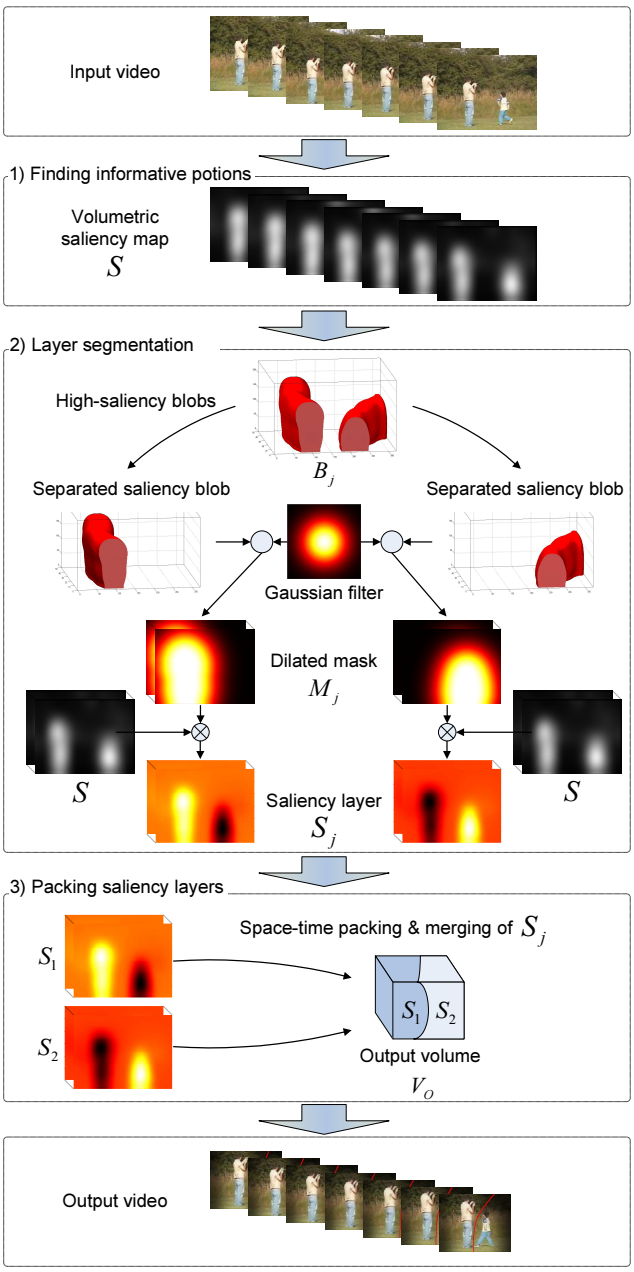

Figure 2. Overview of the space-time video montage.

2. Overview of Space-time Video Montage

The problem of space-time video montage consists of three sub-problems, i.e., finding informative video portions, layer segmentation of informative video portions and packing them in an output video volume. In this section, we present an overview of the problem of space-time video montage and notations which are used in the rest of the paper.

Finding informative video portions. The first problem in space-time video summarization is finding infor-

mative video portions from the long input video sequence V. Defining the amount of information is a difficult problem since it requires image understanding. There exist many methods that try to extract saliency from images/videos [11, 8, 15]. The actual implementation of our saliency measure will be detailed in Sec. 3. Supposing that we are able to assign saliency values to all the video pixels, we are able to obtain a *saliency volume* S that is associated with the input video volume V.

Layer segmentation. The saliency volume S may contain a number of isolated informative portions where high-saliency values are assigned. Here we introduce the idea of *saliency layers* to separately treat those informative portions. The saliency layers $\mathcal{S} = \{S_j : j = 1, \ldots, n\}$ are extracted from the original saliency volume S, where n is the number of layers. We use the notation S_j to represent the j-th layer.

Packing saliency layers. The problem of packing salient video portions into an output video volume such that the total saliency value grows to its maximum and can be viewed as a variant of the *Knapsack problem* [7], which is a classic combinatorial optimization problem. The goal of the Knapsack problem is to pack a set of items into a limited size container such that the total importance of items becomes maximum. Although our problem is similar to the Knapsack problem, the following differences exist: input items are video volumes, each of which can have a larger volume than the output volume; every video pixel in the video volumes is associated with its importance; and input items can overlap each other.

Denoting the output video volume as V_o and the associated saliency volume as S_o, our goal is to pack the input video volume V into the output video volume V_o in a way that S_o contains maximal saliency from $\mathcal{S}$. It is equivalent to finding the optimal space-time translations $\mathbf{x}_j$ of the saliency layers S_j which maximizes the following objective function:

$$\sum_{\mathbf{p} \in S_o} f\big(S_j(\mathbf{p} - \mathbf{x}_j)\big), \tag{1}$$

where $f(\cdot)$ is the function which evaluates the saliency value for each pixel $\mathbf{p} = (x, y, t)^T$. For instance, $f(\cdot)$ can be defined as $f(\cdot) = \max_j(\cdot)$ which takes the maximum saliency value at a pixel where the saliency layers are overlapped. Since the saliency layers are bounded by the original input video volume, it follows $S_j(\mathbf{x}) = 0$ if $\mathbf{x} \notin S_j$. Once the positions $\mathbf{x}_j$ are determined, the color values of the output video V_o are assigned by composing the input video according to the arrangement of the saliency layers. In the case of $f(\cdot) = \max_j(\cdot)$, for instance, by denoting $V(\mathbf{p})$ to represent the color value at the pixel $\mathbf{p}$ in the video volume V, a simple form of the composition can be described as

$$V_o(\mathbf{p}) = \big\{V(\mathbf{p} - \mathbf{x}_j) : j = \arg\max_j \big(S_j(\mathbf{p} - \mathbf{x}_j)\big)\big\}. \tag{2}$$

In the following sections, we describe the implementation details to solve this problem.

3. Implementation

In this section, we describe the details of the algorithm. The overview of the proposed method is illustrated in Fig. 2. Our algorithm consists of three major stages: (1) finding informative video portions, (2) layer segmentation of the saliency volumes and (3) packing saliency layers. In the following subsections, we describe the implementation details of each stage.

3.1. Finding informative video portions

In order to determine salient portions in video, we define a spatio-temporal saliency measure using the spatio-temporal contrast. Our spatio-temporal saliency $S(\cdot)$ at a video pixel position $\mathbf{p}$ is defined using the neighboring pixels $\mathbf{q} \in \mathcal{Q}$ as

$$S(\mathbf{p}) = \mathcal{GS}\big\{\sum_{\mathbf{q} \in \mathcal{Q}} d_S(\mathbf{p}, \mathbf{q})\big\}, \tag{3}$$

where the distance function d_S denotes the stimulus measure and $\mathcal{GS}$ is a Gaussian smooth operator with $\sigma = 3.0$. We define d_S as the l^2-norm color distance:

$$d_S(\mathbf{p}, \mathbf{q}) = ||\mathbf{I}(\mathbf{p}) - \mathbf{I}(\mathbf{q})||_2, \tag{4}$$

where $\mathbf{I}(\cdot)$ is the color vector in the LUV space.

Once the saliency values are computed for all of the pixels in the video volume, the saliency values are normalized to the range of $[0, 1]$.

3.2. Layer segmentation of saliency volumes

In the original volumetric saliency map S, there exist a set of high saliency portions in a low-saliency background. In order to treat the high saliency portions separately, we perform layer segmentation of the volumetric saliency map so that each layer only contains one salient portion. The layer segmentation consists of three steps: (a) segmentation of high saliency portions, (b) morphological growing of saliency portions and (c) assignment of negative saliency values.

(a) Segmentation of high saliency portions. The first stage of the layer segmentation is generating *saliency blobs* that represent the high-saliency portions in the input video. To locate the high-saliency portions and separate them from

their background, we first group the original saliency values in S into three different groups, i.e., high, mid and low-saliency groups. These groups are considered the informative parts, the skirts of the informative parts, and the background portions. To achieve the grouping, K-means clustering is applied with $K = 3$. When n isolated saliency portions are found, n saliency blobs $\mathcal{B} = \{B_j : j = 1, \ldots, n\}$ are generated. The blob B_j represents a set of pixels in the corresponding j-th high-saliency video portion.

(b) Dilation of saliency blobs. Once the saliency blobs $\mathcal{B}$ are extracted, we generate *mask volumes* $\mathcal{M} = \{M_j : j = 1, \ldots, n\}$ from $\mathcal{B}$ in order to compute the dilation of the saliency blobs. This dilation operation tries to simulate the spread of the high-saliency portions. Using the Gaussian filter, the mask volume M_j for the saliency blob B_j is generated by the following equation:

$$M_j(\mathbf{p}) = \exp\left(-\frac{d(\mathbf{p}, B_j)^2}{2\sigma_M^2}\right) \quad \forall \mathbf{p} \in S_j, \tag{5}$$

where the distance function d is defined as

$$d(\mathbf{p}, B_j) = \min_{\mathbf{q} \in B_j} \|\mathbf{p} - \mathbf{q}\|. \tag{6}$$

In Eq. (5), the Gaussian kernel sigma σ_M controls the size of dilation and is set to 50 in our experiments. The mask volumes M_j are then used to generate the saliency layers $\mathcal{S} = \{S_j\}$ by taking the product with the original saliency volume S for each pixel $\mathbf{p}$ as

$$S_j(\mathbf{p}) = M_j(\mathbf{p}) S(\mathbf{p}) \quad \forall \mathbf{p} \in S_j. \tag{7}$$

(c) Assigning negative saliency values. At the last step in the layer segmentation, *negative saliency values* are assigned to each layer S_j. In the saliency layers $\mathcal{S}$, positive saliency values are assigned in each layer to represent the corresponding salient video portion. Besides that, the negative saliency values are used to indicate the other salient portions that appear in other layers. This is equivalent to reducing the importance of the pixels in a specific layer, when the pixels are important in the other layers. This helps reduce the possibility of multiple appearances of the same salient portions in the output video. To assign negative saliency values, S_j is updated by:

$$S_j(\mathbf{p}) \leftarrow S_j(\mathbf{p}) - \sum_{k \neq j} S_k(\mathbf{p}) \quad \forall \mathbf{p} \in S_j. \tag{8}$$

After the above three steps, the saliency layers $\mathcal{S}$ are extracted from the original volumetric saliency map S.

3.3. Packing saliency maps

Given the saliency layers $\mathcal{S}$, this step tries to find the optimal arrangement of $\mathcal{S}$ in the output video volume V_o such that V_o will contain the maximal informative portions of the input video.

In our approach, we adopt a multi-resolution implementation of the first-fit algorithm [5] to efficiently compute the solution from the large-scale data. The first-fit algorithm is a sequential optimization algorithm that first orders the items and places the items one-by-one in the container. In our case, the saliency layers $\mathcal{S}$ are ordered by the size of the corresponding saliency blobs $\mathcal{B}$. The major reason for choosing this approach is that the largest-block-serve-first approach is known to result in denser packing [5]. Here, we assume that the saliency layers $\mathcal{S}$ are ordered in descending order, i.e., the smaller index represents the larger size. We use the output saliency volume S_o which has the same volume as V_o to achieve the optimization. The packing process consists of two steps: (a) positioning the saliency layer and (b) merging saliency layers. The algorithm proceeds sequentially starting from $i = 1$ to n with an initialization that fills the output saliency volume S_o with $-\infty$. We also prepare for the output saliency blob B_o for the computation, which is initialized by filling with zeros.

(a) Positioning the saliency layer We seek the optimal position $\mathbf{x}_j$ for the saliency layer S_j which maximizes the total saliency value in S_o. To achieve this, multi-resolution search is performed in a coarse-to-fine manner. It first searches all possible positions in the coarse scale and refines the position $\mathbf{x}_j$ by searching the local region in the finer scale. The amount of saliency gain ΔS_o in the output saliency volume S_o is computed by

$$\Delta S_o(\mathbf{x}_j) = \sum_{\mathbf{p} \in V_o} \left\{ S_j(\mathbf{p} - \mathbf{x}_j) - S_o(\mathbf{p}) \right\}, \tag{9}$$

and the optimal position $\hat{\mathbf{x}}_j$ for the saliency layer S_j is obtained by finding the position $\mathbf{x}_j$ which maximizes the saliency gain by

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} \{\Delta S_o(\mathbf{x})\}. \tag{10}$$

(b) Merging saliency layers

Once the optimal position $\hat{\mathbf{x}_j}$ is determined, the saliency layer S_j is merged to the output saliency volume S_o in this step. At the same time, color values are simultaneously assigned to the output video. The straightforward approach of merging two saliency volumes S_o and S_j is finding the maximum saliency value for each pixel $\mathbf{p}$. In this case, the saliency value S_o and the color value V_o at the pixel $\mathbf{p}$ are determined by

$$\begin{aligned} S_o(\mathbf{p}) &\leftarrow \max\{S_o(\mathbf{p}), S_j(\mathbf{p} - \hat{\mathbf{x}_j})\}, \\ V_o(\mathbf{p}) &= V(\mathbf{p} - \hat{\mathbf{x}_j}) \text{ if } S_j(\mathbf{p} - \hat{\mathbf{x}_j}) > S_o(\mathbf{p}). \end{aligned} \tag{11}$$

However, this approach may produce a choppy result since it is a local operation that does not consider the connectivity of the video portions. In our approach, we try to generate a visually plausible output video by merging saliency

layers using three different soft constraints: a) maximizing saliency, b) maximizing the continuity of high-saliency portions and c) maximizing the color smoothness at the seam boundaries.

To solve this problem, we build a graphical model $\mathcal{G} = \langle \mathcal{N}, \mathcal{A} \rangle$ to represent the output saliency volume S_o. $\mathcal{N}$ is the set of nodes which correspond to pixels $\mathbf{p}$ in S_o, and $\mathcal{A}$ is the set of arcs which connect nodes. To simplify the explanation, we denote the nodes in $\mathcal{N}$ as $\mathbf{p}$ while it is used for representing pixels as well. Each node $\mathbf{p}$ has six-neighboring nodes connected by arcs in the spatial and temporal directions.

The problem of merging the saliency layer S_j and the output saliency volume S_o can be viewed as a binary labeling problem, i.e., assigning each node $\mathbf{p}$ a label {0,1} representing S_o and S_j, respectively. We also use the notation $\mathbf{p}^L$ to represent the label value of the node $\mathbf{p}$. To efficiently optimize the labeling problem under the soft constraints, we define the energy function E as

$$E = \sum_{\mathbf{p}\in\mathcal{N}} E_1(\mathbf{p}) + \alpha \sum_{\mathbf{p}\in\mathcal{N}} E_2(\mathbf{p}) + \beta \sum_{\substack{\mathbf{p}\in\mathcal{N} \\ a_{\mathbf{pq}}\in\mathcal{A}}} E_3(\mathbf{p},\mathbf{q}), \quad (12)$$

where $a_{\mathbf{pq}}$ represents the arc which connects nodes $\mathbf{p}$ and $\mathbf{q}$. We solve the optimization problem using the *Graph cut* [4, 1] algorithm. The terms E_1, E_2, and E_3 correspond to the saliency energy, likelihood energy and coherence energy, respectively, each of which corresponds to the soft constraints a), b), and c) described above.

Saliency energy. In Eq. (12), E_1 represents the energy term which contributes to maximizing the total saliency value in the output saliency volume S_o. $E_1(\mathbf{p})$ is defined as follows.

$$\begin{cases} E_1(\mathbf{p}) = s_m - (1-\mathbf{p}^L)S_o(\mathbf{p}) - \mathbf{p}^L S_j(\mathbf{p}) \\ s_m = \max_{\mathbf{p}}\{S_o(\mathbf{p}), S_j(\mathbf{p})\} \end{cases}. \quad (13)$$

The term E_1 is minimized when the total saliency value of the merged saliency volume is maximized.

Likelihood energy. The term E_2 regulates the continuity of the high-saliency portions in both S_o and S_j. By measuring the color similarity of the video pixels with the colors in high-saliency portions, it evaluates the continuity of the high-saliency portions. Similar to Li *et al.*'s method [12], we take an approach of clustering the dominant colors and measuring the similarity of the colors. To compute the dominant colors, we use the saliency blobs B_j and B_o in order to determine the high-saliency pixels. The color values obtained via B_j and B_o are clustered independently by the K-means method. We denote the computed mean colors by $\{C_k^{B_o}\}$ for representing the major colors associated to B_o and $\{C_k^{B_j}\}$ for the major colors associated to the saliency layer B_j. We use 15 clusters ($k = 1, \ldots, 15$) for both of them throughout the experiments. For each node $\mathbf{p}$, the minimum color distance between $V_o(\mathbf{p})$ and the major colors $\{C_k^{B_o}\}$is computed by

$$d_{\mathbf{p}}^{B_o} = \min_k \|V_o(\mathbf{p}) - C_k^{B_o}\|. \quad (14)$$

The minimum color distance between $V(\mathbf{p} - \hat{\mathbf{x}_j})$ and the major colors $\{C_k^{B_j}\}$ is also obtained by

$$d_{\mathbf{p}}^{B_j} = \min_k \|V(\mathbf{p} - \hat{\mathbf{x}_j}) - C_k^{B_j}\|. \quad (15)$$

Using these two color distances, the energy term $E_2(\cdot)$ is defined as follows.

$$\begin{cases} E_2(\mathbf{p}^L{=}0) = 0, \quad E_2(\mathbf{p}^L{=}1) = \infty, \quad \forall(\mathbf{p} \in B_o, \mathbf{p} \notin B_j) \\ E_2(\mathbf{p}^L{=}0) = \infty, \quad E_2(\mathbf{p}^L{=}1) = 0, \quad \forall(\mathbf{p} \notin B_o, \mathbf{p} \in B_j) \\ E_2(\mathbf{p}^L{=}0) = \frac{d_{\mathbf{p}}^{B_o}}{d_{\mathbf{p}}^{B_o}+d_{\mathbf{p}}^{B_j}}, \; E_2(\mathbf{p}^L{=}1) = \frac{d_{\mathbf{p}}^{B_j}}{d_{\mathbf{p}}^{B_o}+d_{\mathbf{p}}^{B_j}}, \; \forall \mathbf{p} \in \text{else} \end{cases} \quad (16)$$

With the energy term E_2, pixels that are similar in color with the salient blobs tend to be retained in the output montage.

Coherence Energy. The third term E_3 in Eq. (12) is designed to retain the color coherence at the seam between S_o and S_j. It is penalized when a pair of neighboring nodes $(\mathbf{p}, \mathbf{q})$ connected by the arc $a_{\mathbf{pq}}$ are labeled differently. We define the coherence energy E_3 as follows.

$$E_3(\mathbf{p},\mathbf{q}) = |\mathbf{p}^L - \mathbf{q}^L| \cdot \|H(\mathbf{p}) - H(\mathbf{q})\|^2, \quad (17)$$

where $H(\mathbf{x})$ is defined as

$$H(\mathbf{x}) = (1-\mathbf{x}^L)V_o(\mathbf{x}) + \mathbf{x}^L V(\mathbf{x} - \hat{\mathbf{x}_j}). \quad (18)$$

In Eq. (17), $\|\cdot\|^2$ is the square of the l^2 norm. As it can be seen in Eq. (17), E_3 becomes zero when the same labels are assigned to $\mathbf{p}$ and $\mathbf{q}$. Only when different labels are assigned, e.g., $(0,1)$ and $(1,0)$, E_3 is penalized by the color discontinuity. E_3 satisfies the regularity condition that is necessary for graph representation.

Optimization and update. To achieve the merging step, we apply the Graph cut algorithm only to the volume where S_o and S_j are overlapped. Therefore, the size of the graph $\mathcal{G}$ can be reduced to the overlapped volume. Once the labels are assigned to all the nodes, the output saliency volume S_o and the output video V_o are updated by

$$S_o(\mathbf{p}) \leftarrow \begin{cases} S_o(\mathbf{p}) & \text{if} \quad \mathbf{p}^L = 0 \\ S_j(\mathbf{p} - \hat{\mathbf{x}_j}) & \text{else} \end{cases}, \quad (19)$$

$$V_o(\mathbf{p}) \leftarrow \begin{cases} V_o(\mathbf{p}) & \text{if} \quad \mathbf{p}^L = 0 \\ V(\mathbf{p} - \hat{\mathbf{x}_j}) & \text{else} \end{cases}. \quad (20)$$

In addition, the output saliency blob B_o is also updated as

$$B_o(\mathbf{p}) \leftarrow \begin{cases} B_o(\mathbf{p}) & \text{if} \quad \mathbf{p}^L = 0 \\ B_j(\mathbf{p} - \hat{\mathbf{x}_j}) & \text{else} \end{cases}. \quad (21)$$

4. Experiments

We have tested our method on 30 different videos, all of which are captured in the ordinary situations. The average number of frames is around 100, and the resolution is 320×240. We set $\alpha = 1.0$ and $\beta = 50.0$ in Eq. (12) throughout the experiments. To calculate the saliency measure, the input video frames are first scaled down to 40×30 spatially. $|\mathcal{Q}|$ in Eq. (3) is a set of pixels in $5 \times 5 \times 5$ window. The final saliency value for each pixel is interpolated from the scaled images. In this section, we show the result of four different scenarios, i.e, spatial scale-down (the output volume is spatially smaller than the input volume), temporal scale-down, space-time scale-down, and fusing multiple different input videos.[1]

Spatial scale-down. We first show the result of spatial scale-down, i.e., the output video volume is spatially smaller than that of the input video while the temporal length remains the same. The resolution of the original video is 320×240 with 167 frames, and that of the output video is 220×176 with the same number of frames. The top row of Fig. 3 shows four frames from the original input image sequence. By our space-time video montage method, the output video that has the smaller spatial size is generated as shown in the bottom row of the figure. In the output images, the boundaries of the different video portions are drawn to clearly show the composition result.

Temporal scale-down. Fig. 4 shows the result of temporal scale-down, i.e., the output video volume is temporally smaller than that of the input video while the spatial size remains the same. The top two rows show eight images from the original 270 frame video of resolution 320×240. The bottom row shows five frames from the result which is 110 frame video of the same resolution. Since a drastic scene change due to the camera panning and zoom-in exists, the top row and the middle row seem to be of different scenes. Our method is able to generate a short summary from these types of videos by fusing the important portions in short duration.

Space-time scale-down. The third result is the space-time scale-down case, i.e., the output video volume is spatially and temporally smaller than the input video volume. The top two rows show eight images from the original baseball scene video, which is 88 frames with resolution 320×240. The output video volume is 44 frames with resolution 220×176. This input video has much visual redundancy. By applying our method, the compact and informative small video is successfully generated.

Fusing multiple videos. Our method aims at summarizing a single long video sequence. For multiple videos, the same method can be still applied. Here we show this application. Fig. 6 shows the result of fusing multiple videos together into a single output video volume. In this experiment, three input videos are packed and unified in the output video. The top three rows show the images from the three different video clips, each containing 69, 71, and 84 frames, with resolution 320×240. The bottom row shows the result of our video montage. Here, we show several frames from 60 frames of output with a resolution of 320×240. As shown in the figure, our method works for fusing and packing multiple video inputs, however, we would like to point out that the result usually does not make sense due to the simultaneous visualization of different subjects, e.g., basket ball and soccer.

[1] Supplementary material includes the movie files for all the experimental results shown in this section.

5. Discussion and Conclusions

In this paper, we proposed the idea of *space-time video montage* as a new approach to video summarization. The proposed method extracts space-time informative portions from a long input video sequence and fuses them together in an output video volume. This results in a space-time montage of the input video where multiple portions from the input video appear in the same frame at different positions. Our method has been tested on a variety of videos and has been shown to be able to produce compact and informative output videos. We have shown different summarization scenarios including temporal scale-down, spatial scale-down, spatio-temporal scale-down.

This approach is considered as a generalization of the prior video summarization methods in that our approach is not constrained by a frame-basis.

Limitations: Our method has a couple of limitations. First, our algorithm works well for most of the videos; however, it sometimes produces unsatisfactory results due to the lack of an image understanding scheme. Fig. 7 shows the failure case in which the visually less informative input Fig. 7(a) dominates and pushes out the other input video (b) in the output video (c). This type of results is observed when the high-saliency values are assigned to the perceptually less important video portions due to their high contrast values. This problem will be better handled by integrating more sophisticated method such like Ke *et al.*'s event detection [10, 9, 3]. Second, the proposed approach is computationally expensive. Although we use a sequential optimization in the packing stage, it still spends the most of computational time. For instance, fusing two 100 frame videos of resolution 320×240 takes 180 seconds using a 2.2 GHz CPU without the harddisk I/O. We are investigating a faster packing method by approximating the distribution of the saliency values to a simple distribution model.

As mentioned, the quality of salient portion detection is crucial. We would like to investigate on high-level features such as face, text and attention models that would be able to generate more meaningful summary videos.

Figure 3. Result of spatial scale-down. The top row shows some frames from the original input image sequence, and the bottom row shows the video montage result.

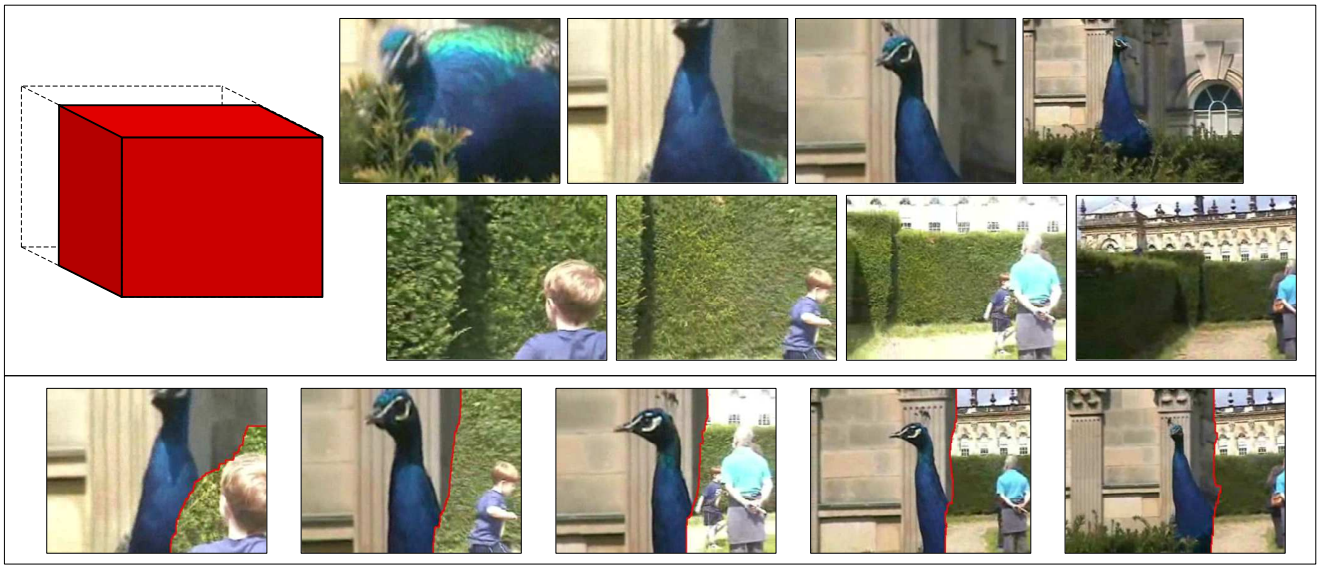

Figure 4. Result of temporal scale-down. The top two rows are some frames from the original input image sequence, and the bottom row shows the video montage result.

Figure 5. Result of space-time scale-down. The top two rows are some frames from the original input image sequence, and the bottom row shows the video montage result.

Figure 6. Result of fusing three different video clips. The top three rows show several frames from the input videos. The bottom row shows the video montage result.

(a) Input video #1 (b) Input video #2 (c) Output video

Figure 7. Failure case of fusing multiple input videos.

References

[1] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. F. Cohen. Interactive digital photomontage. *ACM Trans. Graph.*, 23(3):294–302, 2004. 5

[2] S. Ayer and H. S. Sawhney. Layered representation of motion video using robust maximum-likelihood estimation of mixture models and MDL encoding. In *Proc. of Int'l Conf. on Computer Vision*, pages 777–784, 1995. 2

[3] O. Boiman and M. Irani. Detecting irregularities in images and in video. In *Proc. of Int'l Conference on Computer Vision*, 2005. 6

[4] Y. Boykov, O. Veksler, and R. Zabih. Fast approximate energy minimization via graph cuts. In *Proc. of Int'l Conf. on Computer Vision*, pages 377–384, 1999. 2, 5

[5] E. G. Coffman, M. R. Garey, and D. S. Johnson. Approximation algorithms for bin-packing : an updated survey. *Algorithm Design for Computer Systems Design*, pages 49–106, 1984. 2, 4

[6] N. Doulamis, A. Doulamis, Y. Avrithis, and S. Kollias. Video content representation using optimal extraction of frames and scenes. In *Proc. of Int'l Conf. on Image Processing*, volume 1, pages 875–879, 1998. 1

[7] M. R. Garey and D. S. Johnson. *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., 1990. 3

[8] G. Heidemann. Focus-of-attention from local color symmetries. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 26(7):817–830, July 2004. 2, 3

[9] L. Itti and P. Baldi. A principled approach to detecting surprising events in video. In *Proc. of Computer Vision and Pattern Recognition*, volume 1, pages 631–637, 2005. 6

[10] Y. Ke, R. Sukthankar, and M. Hebert. Efficient visual event detection using volumetric features. In *Int'l Conf. on Computer Vision*, October 2005. 6

[11] I. Laptev and T. Lindeberg. Space-time interest points. In *Proc. of Int'l Conf. on Computer Vision*, pages 432–439, 2003. 2, 3

[12] Y. Li, J. Sun, C.-K. Tang, and H.-Y. Shum. Lazy snapping. *ACM Trans. Graph.*, 23(3):303–308, 2004. 5

[13] Y.-F. Ma, X.-S. Hua, L. Lu, and H.-J. Zhang. A generic framework of user attention model and its application in video summarization. *IEEE Trans. on Multimedia*, 7(5):907 – 919, Oct. 2005. 1

[14] C.-W. Ngo, Y.-F. Ma, and H. Zhang. Automatic video summarization by graph modeling. In *Proc. of Int'l Conf. on Computer Vision*, pages 104–109, 2003. 1

[15] C. Schmid, R. Mohr, and C. Bauckhage. Evaluation of interest point detectors. *Int'l J. Comput. Vision*, 37(2):151–172, 2000. 2, 3

[16] J. Y. A. Wang and E. H. Adelson. Layered representation for motion analysis. In *Proc. Computer Vision and Pattern Recognition*, pages 361–366, June 1993. 2

[17] X. Zeng, L. H. Staib, R. T. Schulz, and J. S. Duncan. Volumetric layer segmentation using coupled surfaces propagation. In *Proc. of Computer Vision and Pattern Recognition*, pages 708–715, 1998. 2