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 in 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 set of input videos by extracting and fusing the space-time informative portions of the input videos. 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 retains informative portions of the input videos as great an extent possible.

Prior works on video summarization [6, 14, 13] has typically been based on a two-step approach: First, video streams have been divided into a set of meaningful and manageable segments called shots, and “key frames” were selected according to criteria from each video shot to generate a summary video. These approaches are successful in extracting video essences, but they are all limited to extracting important video portions using a frame-basis. They treat a video frame as an 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. Image summarization work, such as [15], on the other side reduces the spatial redundancy.

Our approach is built upon the idea that some space-time video portions are more informative than others. Consider-
ing that visual redundancy exists in videos, the assumption is apparently correct. However the definition of “informative” 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, 16]. 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. [17] in computer vision and has been widely used in many different contexts [2, 18]. The layered representation has often been used for describing regional information such as foreground and background or of different objects with independent motion. Instead, 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

Practical demands motivated us 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.

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 in-
formative video portions from $m$ input video volumes $V = \{V^i : i = 1, \ldots, m\}$. 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, 16]. 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 set of saliency volumes $S^{IN} = \{S^i : i = 1, \ldots, m\}$ that are associated with the input volume $V$.

**Layer segmentation.** Each of $S^{IN}$ 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 $\mathcal{S} = \{S_j : i = 1, \ldots, m; j = 1, \ldots, n_i\}$ are extracted from the original saliency volumes $S^{IN}$, where $n_i$ is the number of layers extracted from $S^i$. We use the notation $S_j^i$ to represent $j$-th layer in the saliency volume $S^i$.

**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 volumes $V$ 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 $x_j^i$ of the saliency layers $S_j^i$ which maximizes the following objective function:

$$\sum_{p \in S_o} f(S_j^i(p - x_j^i)).$$

(1)

where $f(\cdot)$ is the function which evaluates the saliency value for each pixel $p = (x, y, t)^T$. For instance, $f(\cdot)$ can be defined as $f(\cdot) = \max_{j}\cdot(\cdot)$ 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 $S_j^i(x) = 0$ if $x \notin S_j^i$. Once the positions $x_j^i$ are determined, the color values of the output video $V_o$ are assigned by composing the input videos according to the arrangement of the saliency layers.

In the case of $f(\cdot) = \max(\cdot)$, for instance, by denoting $V(p)$ to represent the color value at the pixel $p$ in the video volume $V$, a simple form of the composition can be described as

$$V_o(p) = \{V_i(p - x_j^i) : (i, j) = \arg\max_{(i, j)}(S_j^i(p - x_j^i))\}. \quad (2)$$

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. To simplify the explanation, we first consider the problem of generating an output video from a single input video, i.e., the case of $m = 1$. Under this condition, we drop the superscript.\(^1\) to make the notation clearer, e.g., $S^i$ becomes $S$, and $x_j^i$ becomes $x_j$. Later in Sec. 3.4, we describe that the algorithm explained here is still valid for the case of multiple video inputs.

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 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 $p$ is defined using the neighboring pixels $q \in Q$ as

$$S(p) = GS\{\sum_{q \in Q} d_S(p, q)\}, \quad (3)$$

where the distance function $d_S$ denotes the stimulus measure and $GS$ is a Gaussian smooth operator with $\sigma = 2.0$. We define $d_S$ as the Gaussian distance of the intensities as

$$d_S(p, q) = \frac{1}{2\pi\sigma^2_S}\exp\left(-\frac{|I(p) - I(q)|^2}{2\sigma^2_S}\right), \quad (4)$$

where the intensity $I$ is computed by $I = 0.3R + 0.6G + 0.1B$. 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
detect 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 informa-
tive parts, the skirts of the informative parts, and the back-
ground portions. To achieve the grouping, K-means clus-
tering is applied with K = 3. When n isolated saliency por-
tions are found, n saliency blobs \( B = \{ B_j : j = 1, \ldots, n \} \)
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 \( B \) are extracted, we generate mask volumes
\( M = \{ M_j : j = 1, \ldots, n \} \) from \( B \) 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:

\[
M_j(p) = \exp \left( -\frac{d(p, B_j)^2}{2\sigma_M^2} \right) \quad \forall p \in S_j,
\]

where the distance function \( d \) is defined as

\[
d(p, B_j) = \min_{q \in B_j} \| p - q \|.
\]

In Eq. (5), the Gaussian kernel sigma \( \sigma_M \) 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
\( S = \{ S_j \} \) by taking the product with the original saliency
volume \( S \) for each pixel \( p \) as

\[
S_j(p) = M_j(p)S(p) \quad \forall p \in S_j.
\]

(c) Assigning negative saliency values. At the last step in
the layer segmentation, negative saliency values are as-
signed to each layer \( S_j \). In the saliency layers \( S \), positive
saliency values are assigned in each layer to represent the
corresponding salient video portion. Besides that, the nega-
tive saliency values are used to indicate the other salient
portions that appear in other layers. This is equivalent to reduc-
ing 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:

\[
S_j(p) \leftarrow S_j(p) - \sum_{k \neq j} S_k(p) \quad \forall p \in S_j.
\]

After the above three steps, the saliency layers \( S \) are ex-
tracted from the original volumetric saliency map \( S \).

3.3. Packing saliency maps

Given the saliency layers \( S \), this step tries to find the
optimal arrangement of \( S \) 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 implementa-
tion 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 \( S \) are ordered by the size of
the corresponding saliency blobs \( B \). The major reason for
choosing this approach is that the largest-block-serve-first
algorithm is known to result in denser packing [5]. Here, we
assume that the saliency layers \( S \) are ordered in descend-
ing 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 pro-
cess 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 computa-
tion, which is initialized by filling with zeros.

(a) Positioning the saliency layer We seek the optimal po-
sition \( x_j \) for the saliency layer \( S_j \) which maximizes the to-
tal 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 re-
fines the position \( x_j \) by searching the local region in the
finer scale. The amount of saliency gain \( \Delta S_o \) in the output
saliency volume \( S_o \) is computed by

\[
\Delta S_o (x_j) = \sum_{p \in V_o} \{ S_j(p - x_j) - S_o(p) \},
\]

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

\[
\hat{x}_j = \arg \max_x \{ \Delta S_o (x) \}.
\]

(b) Merging saliency layers

Once the optimal position \( \hat{x}_j \) 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 \( p \). In this case, the
saliency value \( S_o \) and the color value \( V_o \) at the pixel \( p \) are
determined by

\[
S_o(p) \leftarrow \max \{ S_o(p), S_j(p - \hat{x}_j) \},
\]

\[
V_o(p) = V(p - \hat{x}_j) \quad \text{if} \quad S_j(p - \hat{x}_j) > S_o(p).
\]
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 \( G = (\mathcal{N}, \mathcal{A}) \) to represent the output saliency volume \( S_o \). \( \mathcal{N} \) is the set of nodes which correspond to pixels \( p \) 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 \( p \) while it is used for representing pixels as well. Each node \( p \) 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 \( p \) a label \{0, 1\} representing \( S_j \) and \( S_o \), respectively. We also use the notation \( p^o \) to represent the label value of the node \( p \). To efficiently optimize the labeling problem under the soft constraints, we define the energy function \( E \) as

\[
E = \sum_{p \in \mathcal{N}} E_1(p) + \alpha \sum_{p \in \mathcal{N}} E_2(p) + \beta \sum_{\{p, q\} \in \mathcal{A}} E_3(p, q),
\]

where \( \alpha_{pq} \) represents the arc which connects nodes \( p \) and \( q \). 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 \( S_o \). \( E_1(p) \) is defined as follows.

\[
E_1(p) = s_m - (1 - p^L) S_o(p) - p^L S_j(p)
\]

and

\[
s_m = \max_{p} \{S_o(p), S_j(p)\}.
\]

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 \( \{C^B_o\} \) for representing the major colors associated to \( B_o \) and \( \{C^B_j\} \) 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 \( p \), the minimum color distance between \( V_o(p) \) and the mean colors \( \{C^B_j\} \) is computed by

\[
d^B_o = \min_k \|V_o(p) - C^B_o\|.
\]

The minimum color distance between \( V(p - x_j) \) and the mean colors \( \{C^B_j\} \) is also obtained by

\[
d^B_j = \min_k \|V(p - x_j) - C^B_j\|.
\]

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

\[
E_2(p^l=0) = 0, \quad E_2(p^l=1) = \infty, \quad \forall (p \in B_o, p \notin B_j)
\]

\[
E_2(p^l=0) = \infty, \quad E_2(p^l=1) = 0, \quad \forall (p \notin B_o, p \in B_j)
\]

\[
E_2(p^l=0) = a^k_{pq} + \frac{1}{d^o_{pq}}, \quad E_2(p^l=1) = a^k_{pq} + \frac{1}{d^j_{pq}}, \quad \forall p \in \text{elsewhere}
\]

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 \( (p, q) \) connected by the arc \( \alpha_{pq} \) are labeled differently. We define the coherence energy \( E_3 \) as follows.

\[
E_3(p, q) = |p^L - q^L| \cdot \|H(p) - H(q)\|^2,
\]

where \( H(x) \) is defined as

\[
H(x) = (1 - x^L) V_o(x) + x^L V(x - x_j).
\]

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 \( p \) and \( q \). 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 \( G \) 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

\[
S_o(p) \leftarrow \begin{cases} S_o(p) & \text{if } p^L = 0 \\ S_j(p - x_j) & \text{else} \end{cases},
\]

\[
V_o(p) \leftarrow \begin{cases} V_o(p) & \text{if } p^L = 0 \\ V(p - x_j) & \text{else} \end{cases}.
\]

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

\[
B_o(p) \leftarrow \begin{cases} B_o(p) & \text{if } p^L = 0 \\ B_j(p - x_j) & \text{else} \end{cases}.
\]
3.4. Packing multiple videos into a single video

So far, we have described the proposed algorithm in the case of \( m = 1 \), i.e., the case of a single input video. When multiple input videos are provided, we treat them in the same manner as if they are supplied from a single video. The size of the input video can vary, and this is reflected by the size of saliency layers, e.g., the sizes of \( S^i \) and \( S^{i'} \) are treated differently. Nonetheless, the algorithm described above is still valid.

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 \( 320 \times 240 \). 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 X 30 spatially. \( |Q| \) in Eq. (3) is a set of pixels in \( 5 \times 5 \times 5 \) window, and \( \sigma_S \) in Eq. (4) is set to 1.0. 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 \( 320 \times 240 \) with 167 frames, and that of the output video is \( 220 \times 176 \) 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 \( 320 \times 240 \). 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 \( 320 \times 240 \). The output video volume is 44 frames with resolution \( 220 \times 176 \). This input video has much visual redundancy. By applying our method, the compact and informative small video is successfully generated.

Fusing multiple videos. 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 \( 320 \times 240 \). The bottom row shows the result of our video montage. Here, we show several frames from 60 frames of output with a resolution of \( 320 \times 240 \).

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 set of input videos and fuses them together in an output video volume. This results in a space-time montage of input videos where multiple portions from different video clips 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, and fusing multiple different input videos.

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 \( 320 \times 240 \) 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

\(^1\)Supplementary material includes the movie files for all the experimental results shown in this section.
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.

generate more meaningful summary videos.

References


