Discovering Panoramas in Web Videos

Feng Liu
Department of Computer Sciences
University of Wisconsin-Madison
fliu@cs.wisc.edu

Yu-hen Hu
Department of Electrical and Computer Engineering
University of Wisconsin-Madison
hu@engr.wisc.edu

Michael L. Gleicher
Department of Computer Sciences
University of Wisconsin-Madison
gleicher@cs.wisc.edu

ABSTRACT

While methods for stitching panoramas have been successful given proper source images, providing these source images still remains a burden. In this paper, we present a method to discover panoramic source images within widely available web videos. The challenge comes from the fact that many of these videos are not recorded intentionally for stitching panoramas. Our method aims to find segments within a video that work as panorama sources. Specifically, we determine a video segment to be a valid panorama source according to the following three criteria. First, its camera motion should cover a wide field-of-view of the scene. Second, its frames should be "mosaicable", which states that the inter-frame motion should observe the underlying conditions for stitching a panorama. Third, its frames should have good image quality. Based on these criteria, we formulate discovering panoramas in a video as an optimization problem that aims to find an optimal set of video segments as panorama sources. After discovering these panorama sources, we synthesize regular scene panoramas using them. When significant dynamics is detected in the sources, we fuse the dynamics into the scene panoramas to make activity synopses to convey the dynamics. Our experiment of querying panoramas from *YouTube* confirms the feasibility of using web videos as panorama sources and demonstrates the effectiveness of our method.

Categories and Subject Descriptors

I.4.9 [**Image Processing and Computer Vision**]: Applications

General Terms

Algorithms

Keywords

Discovering panorama, Scene panorama, Activity synopsis

MM'08, October 26–31, 2008, Vancouver, British Columbia, Canada.

1. INTRODUCTION

A panoramic image presents a wide field-of-view of a scene. Many kinds of artworks, such as painting, drawing, photography and film, use panoramic imagery to reveal a wide, all-encompassing view of a scene [12]. Panoramic imagery has also been used in many applications in multimedia, computer vision and graphics, such as video summarization and abstraction [5, 29, 13], video browsing [28], video stabilization [37, 20],environment visualization [25], virtual reality [15], background modeling [6], compression [13], etc.

Creating panoramic imagery is a well studied topic. It usually involves two steps: image alignment and stitching. The first step builds the correspondence between each image pair by estimating a homography. The second step uses the alignment result to blend the images in a seamless manner, taking care to deal with potential problems such as blurring, ghosting, etc. Szeliski [26] provides a good survey on panorama creation.

While existing algorithms provide successful solutions to stitching panoramas, they require carefully selected and properly ordered source images. The projective model used in panorama stitching requires the source images to be taken from the same viewpoint. A common goal of panoramas is that they cover a wide field-of-view, which requires the source images to provide such coverage. Although the popularity of image capturing devices makes obtaining images easy, creating a good panorama still requires careful planning to get useful source images. Moreover, due to temporal and geographical constraints, obtaining specific images might be difficult and expensive. For example, if we want to have a panorama of *Lake Louise, Canada*, we may have to fly there to take photos.

Emerging applications also cast difficulty for users to provide/find panorama sources. For example, panorama imageries can be effective representations of an image set or even a video set. Under such an application scenario, relying on manual selection of panorama sources will be infeasible.

Nowadays, a huge amount of imageries are available, such as images on *Flickr* [10] and videos on *YouTube* [38]. Often these imagery collections contain a lot of images describing the same scene, thus possibly providing image sources for stitching a panorama. However, not all these images describing the scene can be used as panorama sources. For example, using keywords to search images from *Flickr* can return many images irrelevant to the scene because possibly they are not properly tagged. Also many of these images were taken at extremely different lighting situations. For videos from *YouTube*, many of them are of low quality. Also

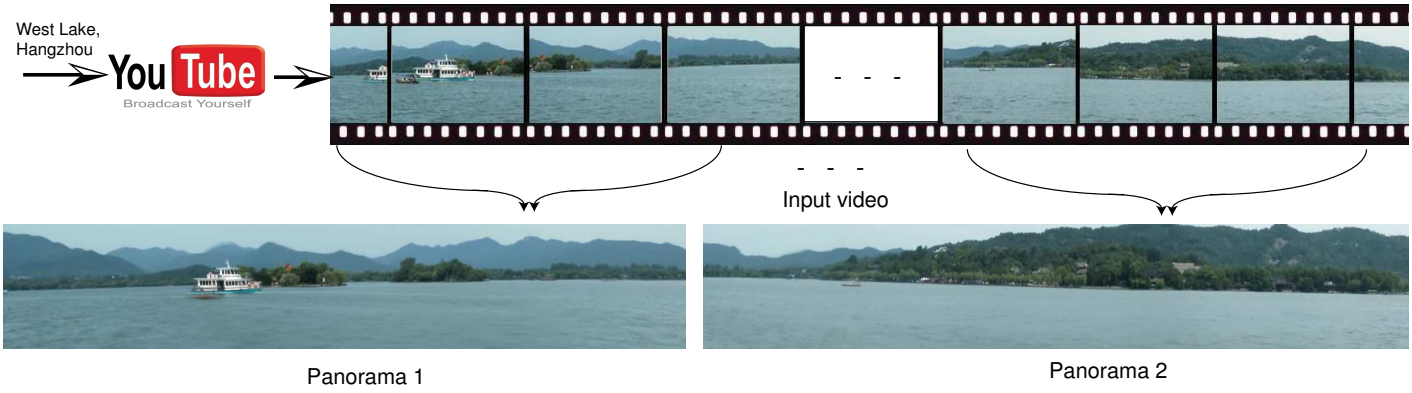


Figure 1: Working example. A user makes a query of "West Lake, Hangzhou" to *YouTube*, and feeds retrieved video clips into our system. Our system selects useful frames from the given videos and synthesizes panoramas using the selected source frames.

many videos do not contain a wide field-of-view of the scene. Methods that can automatically identify valid sources from imagery collections for stitching panoramas are desired.

Recently, Brown and Lowe [2, 3] presented an approach for recognizing panorama sources from photo collections. Their methods formulate panorama stitching as a multi-image matching problem, and use invariant local features to find matches between all the images. Based on image matching, multiple panoramas can be created from an unordered image set. Compared to still images, videos provide a dense set of ordered images, and the temporally adjacent frames most likely describe the same scene, which relieves the difficult image matching problem. Moreover, videos contain more information than still images. For example, videos provide dynamic information in the scene. Using the dynamic information can create a more informative panoramic image, such as activity synopsis(c.f. [13, 5, 29]), than the standard scene panorama.

In this paper, we explore existing web videos for panorama sources. As illustrated in Figure 1, our method takes a video as input, identifies segments of the video as panorama sources, determines proper stitching parameters for each segment, and synthesizes panoramas from these segments.

The challenge of discovering panorama sources in a video is that many of these videos are recorded casually, not intended for stitching panoramas. They may not cover a wide field-of-view of a scene. Their camera motion may be random and not follow the common motion models used for stitching panoramas. The image quality may be bad due to amateur videography skills and improperly set-up devices and environments. Moreover, many videos are compressed several times for the sake of web-sharing.

Considering these facts, we regard a video segment to be a valid panorama source if it meets the following three criteria. First, its camera motion should cover a wide field-of-view of a scene. Second, its frames should be "mosaicable", which states that the inter-frame motion should observe the underlying conditions for stitching a panorama. Third, its frames should have high image quality to create a high-quality panorama. Based on these criteria, we formulate discovering panorama sources from a video as an optimization problem that aims to find an optimal set of video segments as panorama sources. After solving this optimization problem for the panorama sources, we build scene panoramas or activity synopses according to the video dynamics. During stitching a panorama, pixel sampling is biased according to the image quality of each frame to create high-quality panoramas.

Our current system consists of three parts: video analysis, panorama source selection, and panorama synthesis. As described in Section 2, through video analysis, our system estimates image alignment models between frames, detects moving objects, and measures visual image quality for each frame. Based on the video analysis information, our system discovers proper video segments which can be used to generate high quality panoramas as detailed in Section 3. Finally, the panorama synthesis engine takes the sources together with the video analysis result, and synthesizes panoramic images as described in Section 4. We evaluate our algorithm in the scenario of querying panoramas from *YouTube*. The experiments confirm the feasibility of using casually recorded videos as panorama sources and demonstrate the effectiveness of our method as detailed in Section 5.

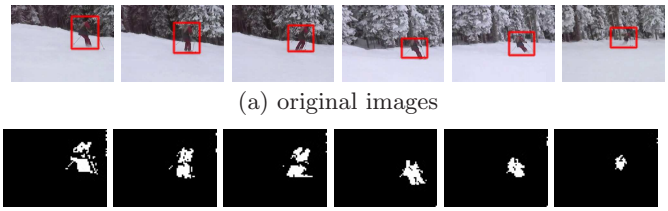
2. VIDEO ANALYSIS

A video clip can usually be divided into a sequence of shots. Each shot is defined as a continuous frame sequence recorded by a camera setting. Different shots most likely contain different scenes. So we break a video into a shot sequence and discover panoramas from each shot independently. Shot boundary detection is a well studied topic [7]. We apply a histogram based method for shot boundary detection [22]. This algorithm is efficient and robust against object and camera motions. First, a color histogram is computed for each frame. Color is quantized to improve performance. A shot boundary is detected whenever the histogram intersection between two neighboring frames is below a threshold.

In the following subsections, we describe video analysis components necessary for discovering and stitching panoramas, namely estimating inter-frame motions, detecting moving objects and measuring visual quality.

2.1 Camera motion estimation

The temporal frame order in a video gives an implicit constraint on image matching. The temporal neighbors in a



(a) original images

(b) moving objects

Figure 2: Moving object detection. Moving objects are identified by red rectangles.

shot are most likely spatial neighbors, describing the same scene. Based on this implicit order, we first estimate motion between consecutive frames, and then calculate motion between every two frames by simple matrix composition.

We use a homography [11] to describe the geometric relationship between every 2 consecutive frames. A homography is a 2D perspective matrix described by 8 parameters. The relationship between corresponding points in 2 images ***I*** and ***I'*** can be described as

$$\begin{bmatrix} sx \\ sy \\ s \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x^{'} \\ x^{'} \\ 1 \end{bmatrix},$$

where$s$ is a scalar, (x, y) is the position of a pixel in image ***I***, (x', y') is its counterpart in image ***I'***, and $h_?$ is a homography matrix element. Many methods have been presented to estimate the homography. A good survey can be found in [26]. Since many web videos have poor quality due to compression, noise and distortion from sensors, blurring from motion, and jittering camera motion, we use a feature-based method. SIFT features [17] are used due to their robustness. We extract SIFT features from each frame and use a RANSAC [9] algorithm to obtain a robust estimation of the homography between consecutive frames. Typically, we extract from each video frame of size 320×240 about 500 SIFT features, about half of which can be well matched to those from its neighboring frame. The homography between any two non-adjacent frames can be calculated by matrix composition. Bundle adjustment is applied to improve the accuracy and consistency of motion estimation between any two frames(c.f. [26, 19, 23]).

2.2 Moving object detection

One of the important aspects in which a video differs from a still image is that it contains more than visual information. Dynamics is one of the major characteristics of a video. As far as panoramas are concerned, understanding the dynamics can help convey more information about the video using panoramic images. Particularly, knowing moving foreground objects enables creating a special kind of panorama, activity synopsis.

We assume that the background is dominant in a video. Based on this assumption, the background motion can be characterized by the homography, and each pixel in a video frame can be classified into the moving foreground or background by examining the discrepancy between its local motion vector and the global motion. If the discrepancy is bigger than a threshold, the pixel is determined belonging to the moving foreground object. The local motion vector

(a) blurriness=0.049 (b) blurriness=0.299

(c) blockness=0.204 (d) blockness=0.497

Figure 3: Image quality assessment.

can be estimated as optical flow [18], and the global motion is characterized by the homography. Directly estimating the optical flow suffers from noise, aperture problem, etc. Since we get a robust estimation of the homography, we use the homography as an initial guess to guide a block matching search for the optical flow. The moving object region is determined as the smallest rectangle containing a certain amount of moving pixels. An example of the moving foreground object is illustrated in Figure 2. More advanced methods can be used to estimate moving object with accurate shapes at the expense of more computational cost. For this application, a rough estimation from the current method is good enough.

2.3 Visual quality measures

Many web videos are of low quality for a variety of reasons. For example, videos are often highly compressed for web-sharing, causing blurring and blocking artifacts. The original videos could also be of low quality due to bad capture settings or environments. Low quality image sources usually lead to low-quality panoramas. So measuring visual quality of video frames is important. Instead of measuring high-level semantic qualities [14] that are most likely subjective, we measure visual quality by estimating low-level distortions that degrade the images, because these low-level distortions can directly affect the image quality of the resulting panoramas.

Existing approaches to measuring low-level image distortions can be divided into two categories, reference-based and non-reference based methods. Reference-based distortion measures require the original non-distorted image to assess the quality of the distorted one [8, 36]. Although these methods can provide reasonable assessments, they are not suitable for our purpose. While blind image quality assessment is difficult, there have been several useful methods that measure image quality without reference images(c.f. [34, 35, 16, 31, 32]).

In this paper, we measure the image quality in terms of blurring and blocking artifacts. We adopt Tong *et al.*'s method to detect blurring artifacts based on wavelet transform [31]. This method measures the blurriness based on the image edge type and sharpness analysis using the Haar Wavelet Transform [24]. An example of blur detection is illustrated in Figure 3(a) and (b). We use Wang *et al.*'s method to measure blocking artifacts [35]. This method estimates the blockiness as the average difference across block boundaries modulated by image activities. An example of blocking measurement is illustrated in Figure 3(c) and (d).

3. DISCOVERING PANORAMAS

Not all videos can be used as panorama sources. For example, a video recorded by a static camera does not cover a wide field-of-view of a scene, so it cannot provide panoramic source images. Moreover, even if a video contains panoramic source images, not all its frames are useful. We consider a video or its segment with the following properties to be a valid panorama source.

- C1: A video (segment) should cover a wide field-of-view based on the definition of panorama imagery. Since field-of-view is hard to measure without knowing accurate camera settings, we approximate it with the extent of the scene covered the video segment as will be described in Section 3.2.

- C2: A video should be "mosaicable" using the state-of-the-art technologies. Specifically, the underlying camera motion between frames shall observe a certain camera motion model. We use a homography to model the motion between frames. A homography requires either a planar scene or that the camera only rotates around its optical center. Few casual videos meet these requirements. In practice, if the inter-frame motion is close to the homography, the quality of the panorama is high, and vice versa. The "mosaicability" of a video directly related to the quality of the resulting panoramic imagery. We measure the "mosaicability" by the image distortion caused by panorama stitching as detailed later in this section.

- C3: Video frames should have high image quality. Although there exist methods to improve the image visual quality, such as de-blurring and de-blocking, we require high visual quality sources to create high quality panoramas. We adopt this conservative strategy not only because automatic methods could fail but also a video is such a dense sampling of a scene that missing some frames will hurt the final result little.

Ideally, we want to discover from a video panoramas that have a very wide field-of-view of a scene and are of very high visual quality. However, in practice, the goals of having a wide field-of-view of the scene and having high visual quality often collide with each other. For example, to create a panorama with a wider field-of-view requires including more source frames. However, including more source frames can often degrade the visual quality. For example, the accumulated motion estimation error can be bigger when including more frames, leading to misalignment during stitching. Also including more frames likely includes more bad-quality source frames. We formulate discovering panoramas from a video as an optimization problem that aims to find a series of video segments that achieve an optimal balance between maximizing the visual quality of the resulting panoramas and maximizing the scenes they cover.

$$\begin{array}{l}\hat{S} = \arg\min_S\{\sum_{S_i\in S} E_v(S_i) + \sum_{S_i,S_j\in S} E_a(S_i,S_j)\} \\ \text{where} \quad S = \{S_i | S_i \subseteq V\}, \forall S_i, S_j \in S, S_i \bigcap S_j = \emptyset \\ s.t. \quad \begin{cases} E_v(S_i) < \delta, \forall S_i \in V \\ \mathcal{E}(S_i) > \beta A, \forall S_i \in V \end{cases}\end{array} \tag{1}$$

where S denotes a set that contains non-overlapping segments S_i of the video clip V. $E_v(S_i)$ is the visual quality cost of stitching a panorama from S_i based on criterion **C2** and **C3**, and $E_a(S_i, S_j)$ is the cost of splitting a panorama from $S_i \bigcup S_j$ into two smaller ones from S_i and S_j respectively according to criterion **C1**. $\mathcal{E}(S_i)$ denotes the extent of the scene in S_i, which is required to be bigger than β times the original video frame size A. To guarantee the visual quality of the panorama, the visual quality distortion $E_v(S_i)$ is required to be less than a threshold δ. We explain the above terms in detail in the following subsections.

3.1 Visual quality measure

We measure the visual quality distortion $E_v(S_i)$ of stitching a panorama from the video segment S_i from two aspects: the incorrectness of the motion model denoted as $E_{vm}(S_i)$, and the source image visual quality distortion denoted as $E_{vv}(S_i)$.

$$E_v(S_i) = \alpha_m E_{vm}(S_i) + \alpha_v E_{vv}(S_i) \tag{2}$$

where α_m and α_v are weights, with default values 1.0 and 1.0.

The 2D geometrical motion model usually cannot be perfectly accurate when describing the correspondence between two frames although it is often a close approximation. A homography can be perfectly accurate only in the case of a planar scene or a camera only rotating around its optical center. Moreover, the error can accumulate when the frame span becomes big. So using the homography to stitch frames can lead to misalignment. We measure the error using the homography to stitch the frames as the discrepancy between the real motion vector of the SIFT feature points and its counterpart predicted using the homography.

$$E_{vm}(S_i) = \sum_{I_k\in S_i} \frac{1}{n_k} \sum_{p_{j,k}\in I_k} \|mv(p_{j,k}), mv_h(p_{j,k})\| \tag{3}$$

where $p_{j,k}$ is the j^{th} SIFT feature in frame I_k, $mv(p_{j,k})$ and $mv_h(p_{j,k})$ are the motion vectors of $p_{j,k}$ calculated by matching SIFT feature points and predicted by the homography respectively. n_k is the number of matching SIFT feature points between frame I_k and I_{k+1}.

Using low-quality video frames to stitch a panorama will most likely create a low-quality panorama. Since source frames often suffer from compression distortion, we measure the input visual quality distortion using the blockiness and blurriness discussed in Section 2.3.

$$E_{vv}(S_i) = \sum_{I_k\in S_i} \gamma q_{bk}(I_k) + (1-\gamma) q_{br}(I_k) \tag{4}$$

where $q_{bk}(I_k)$ and $q_{br}(I_k)$ measure the blockiness and blurriness of frame I_k, and γ is a parameter with the default value 0.45.

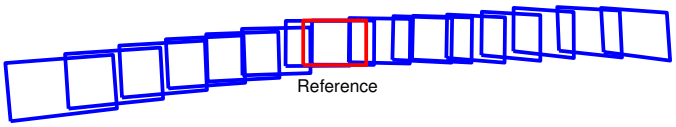


Figure 4: All the frames in the video segment are aligned to the reference frame (in red). The union of all these aligned quadrilaterals is the area covered by the video segment. The extent of the scene in the segment is defined as the minimum of the area with a certain reference.

3.2 Scene extent measure

We measure the extent of the scene covered by a video segment using the area covered by the segment when aligning all its frames onto a reference frame. While choosing a certain reference frame can maximize the area, it will introduce serious distortion. To minimize the distortion, we define the extent of the scene, $\mathcal{E}(S_i)$, as the minimal area covered by S_i when choosing one of its frames as the reference as follows:

$$r = \arg\min_{f_r \in S_i} \bigcup_{f \in S_i} I(f, f_r) \tag{5}$$

$$\mathcal{E}(S_i) = \bigcup_{f \in S_i} I(f, r) \tag{6}$$

where r is the final reference frame, f_r is a frame in S_i, $I(f, r)$ is frame f aligned to frame r, and the $\bigcup$ operator is the union of the quadrilaterals of $I(f, r)$ as illustrated in Figure 4, which can be efficiently calculated using a generic polygon clipping method [33].

A panorama with a large field-of-view is preferred over dividing it into two panoramas with a small field-of-view. So we penalize splitting a long video segment into two short ones with the cost $E_a(S_i, S_j)$.

$$E_a(S_i, S_j) = \frac{\mathcal{E}(S_i \bigcup S_j) - \kappa(\mathcal{E}(S_i) + \mathcal{E}(S_j))}{\mathcal{E}(S_i \bigcup S_j)} \tag{7}$$

where $\mathcal{E}$ is the scene extent defined in Equation 6 and κ is a parameter with the default value 0.7. The idea behind the above energy function is to penalize creating small size panoramas.

Directly solving the constrained nonlinear integer programming problem of Equation 1 is difficult. We approximate the optimal solution $\hat{S}$ with the following steps.

1. Initialize the video segment pool with the whole video, $S^p = \{V\}$, and the panorama segment set as $\hat{S} = \emptyset$.
2. Fetch a segment S_k from S^p.
3. Find the scene extent of S_k and the corresponding reference frame r_k according to Equation 6.
4. Starting from the reference frame r_k as the initial video segment $\hat{S}_i$, append it with one of its immediately adjacent neighboring frames which adds the smaller distortion calculated according to Equation 2. Continue appending $\hat{S}_i$ until $E_v(\hat{S}_i) > \delta$.
5. If the scene extent of $\hat{S}_i$ meets $\mathcal{E}(\hat{S}_i) > \beta A$, append $\hat{S}$: $\hat{S} = \hat{S} \bigcup \{\hat{S}_i\}$. Add the remainder left part and right part of S_k into the pool S^p.
6. If the pool $S^p \neq \emptyset$, go to step 2; Otherwise stop.

4. PANORAMA SYNTHESIS

For each source video segment found in the previous section, we examine its dynamics: If one significantly moving object is detected in the video segment as described in Section 2.2, we create an activity synopsis, which conveys the dynamics of the segment as well as the wide field-of-view scene; otherwise we create a scene panorama using the composition methods described in the following.

4.1 Scene panorama synthesis

Panorama synthesis can be achieved by aligning all the images to a common surface, here the reference frame defined in Equation 5, and blending them in a seamless manner. The feathering algorithm is a commonly used blending method to synthesize a scene panorama [27]. The idea is to take a distance-weighted average value for each pixel as follows:

$$I^p(x) = \sum_k w_k(x)\tilde{I}_k(x) / \sum_k w_k(x) \tag{8}$$

$$w_k(x) = \| \arg\min_y \{\|y\| \mid \tilde{I}_k(x+y) \text{is invalid}\} \| \tag{9}$$

where $I^p(x)$ is the pixel value at x in the panorama I^p, and $\tilde{I}_k(x)$ is the pixel value at x in the source frame k aligned to the reference frame. w_k is the distance weighting map, and its value at position x is proportional to its Euclidean distance to the nearest invalid pixel.

The feathering algorithm does a reasonably good job, however, when there is a moving object in the scene, ghosting artifacts occur as illustrated in Figure 5(b). To solve this problem, we improve the feathering algorithm with a median-bilateral filtering [30] as follows:

$$w'_k(x) = w_k(x) \exp(-(\tilde{I}_k(x) - med(x))^2 / \sigma^2) \tag{10}$$

where $med(x)$ is the median value of pixel x in all the aligned frames. Since a moving object only occurs at a certain position for a short period, the median value shall be a close estimation of the background color. Furthermore, to improve the panorama's visual quality, the weight is modulated according to the visual quality measurement of the source frame as follows

$$w''_k(x) = w'_k(x) \exp(-(\gamma q_{bk}(k) + (1-\gamma) q_{br}(k))) \tag{11}$$

where $q_{bk}(k)$ is the blockiness of frame k, $q_{br}(k)$ is the blurriness and γ is the same constant as in Equation 4. The de-ghosting result is illustrated in Figure 5(c).

4.2 Activity synopsis synthesis

For a video with significantly moving objects, an activity synopsis that conveys the dynamics of the video will be a more informative representation of the video. We first create a scene panorama using the method described in the above subsection. Then we select and composite the moving objects into the scene panorama to convey the dynamics.

To convey the dynamics and avoid occlusion among objects, we roughly evenly distribute the objects along the trajectory of the scene. The object appearing first in the video segment is selected at first. Then we iterate through all the frames and select the next showing object which is a certain distance apart from the previously selected one to avoid occlusion. This procedure is repeated until no more objects can be shown.

(a) original images

(b) Scene panorama: feathering

(c) Scene panorama: feathering + bilateral filtering

(d) Activity synopsis: "over" composition

(e) Activity synopsis: multi-resolution blending

Figure 5: Panorama synthesis.

Inserting the moving objects into the scene panorama using "over" composition will result in a non-smooth transition between the background and the object regions as shown in Figure 5(d). We use a multi-band blending algorithm [4] to address this issue while avoiding blurring. The idea is to blend the low frequency over a larger spatial range and the high frequency over a small spatial range. An example of activity synopsis is illustrated in Figure 5(e).

5. EXPERIMENTS

We experimented with our algorithm on video clips downloaded from *YouTube* [38]. Some of them are highly compressed and do not have good image quality. The frame size of all the videos is 320×240. We report some representative results here.

The two panoramas of the West Lake shown in Figure 6(a) and (b) are created from a 26-second video clip. Its frame rate is 10 fps. This video was taken with a handheld camera on a fast moving boat. The presented method automatically extracts two segments, from frame 0 to frame 86 and from frame 89 to 251 respectively, and creates the two panoramas. Because camera motions between frames from 86 to 89 jump quickly and the overlap between the first and the second segment is very small, our algorithm breaks the original video into two. The panorama of the Lake Louise shown in Figure 6(c) is created from a 21-second video clip. Its frame rate is 10fps. This video was taken with a handheld camera. The camera was "panned" casually to scan the scene and then it underwent quick random movements. The presented method extracts the first 98 frames where the camera was "panned" to create the panorama and discarded the other 110 frames that are not useful to stich a panorama.

Figure 7(a) shows an activity synopses that shows horseback riding in a farmland. Its source images are frame 297 to 495 from a 58-second, 15fps video. Figure 7 (b) shows two activity synopses that convey highlights of mountain biking from a 194-second 30fps video.

5.1 Query panorama from *YouTube*

We further evaluate our algorithm in a promising application scenario[1]: query panorama from *YouTube*. For example, if a user wants to get a panorama of *Notre Dame, Paris*, he makes a query into *YouTube* with corresponding key words. Hypothetically, if *YouTube* had the functionality of our algorithm, it could discover panoramas from the retrieved videos. We simulated this application by downloading and running our algorithm on the top 10 retrieved videos from *YouTube*.

In this experiment, we made 6 queries as listed in Table 1, and ran our algorithm on the top 10 retrieved videos from *YouTube Travel and Events* category. These queries were randomly selected from a pool. The pool is formed by collecting responses to the question "what scene you want to make a panorama of" from our colleagues and friends. Our algorithm determines that a video contains no panorama if no panorama with the size double of the original video frame size can be discovered. Otherwise our algorithm generates panoramas from the video.

Figure 8 shows several representative panoramas of *Notre Dame, Paris* discovered by our algorithm. Interestingly, our algorithm provides a comprehensive pictorial description of *Notre Dame, Paris*, such as a frontal view of the building (a), an inside view (b), a close-up of the decoration (c), its environment (d), etc. Figure 9 shows two representative panoramas on query *Hong Kong Skyline*. These panoramas provide different views of the Hong Kong Skyline at different time in a day. Figure 10 shows some representative panoramas on query *Vancouver Beach*. These panoramas again give a variety of views of *Vancouver Beach*. Representative panoramas for the other queries are shown in Figure 6.

To quantitatively examine our algorithm, we looked into each retrieved video and manually checked to see if it appeared to contain panoramas or not. If it contained some panoramas, we recorded the frames that can be used to synthesize each panorama. We evaluate the performance of our algorithm as follows:

- For a video containing no panorama, we consider our algorithm succeed if it reports that the video contains no panorama. If our algorithm succeeds, we give it score 1; otherwise 0.
- For a video containing n panoramas, we examine each panorama independently. For each panorama, we consider our algorithm succeed if it outputs a panorama

[1]Please visit the project web site: `http://www.cs.wisc.edu/~fliu/project/discover-pano.htm` for detailed results, including videos and panoramas.

(a) West lake 1. Its source images are frame 0 to 86 from a 26-second, 10fps web video.

(b) West lake 2. Its source images are frame 89 to 251 from the same video as (a).

(c) Lake Louise. Its source images are frame 0 to 98 from a 21-second, 10fps web video.

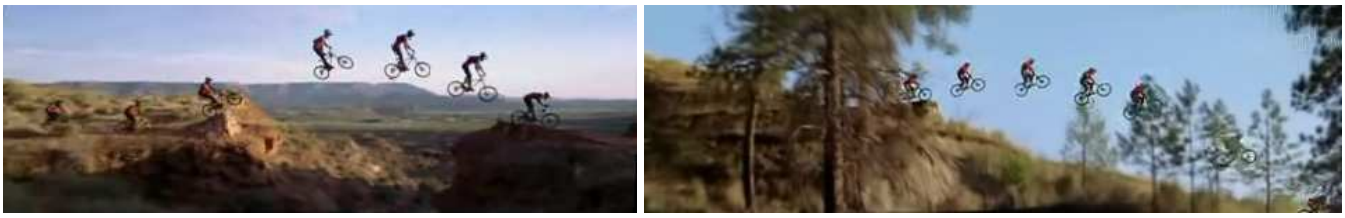
(d) Arches National Park. Its source images are frame 55 to 173 from a 15-second, 15fps web video.

(e) Arches National Park. Its source images are frame 142 to 265 from a 31-second, 15fps web video.

(f) Arches National Park. Its source images are frame 1168 to 1296 from a 83-second, 15fps web video.

Figure 6: Examples. Note, the boundaries of panoramas in this and the following figures are cropped. Please visit our project web site for details.

(a) Horse riding. Its source images are frame 297 to 495 from a 58-second, 15fps web video



(b) Biking. The source images of the two panoramas are frame 2729 to 2856, and frame 4298 to 4625 from a 194-second 30fps web video.

Figure 7: Activity synopsis.

that contains the majority of the panorama in the video. If our algorithm succeeds, we give it score $1/n$; otherwise 0. We sum up the score for each panorama as the score for the video.

The results are reported in Table 1. The average score on these queries is 8.68/10. This result shows that our algorithm works robustly. The majority of failure cases are those that our algorithm did not discover panoramas that exist in videos. Those cases are caused by the inaccurate motion estimation. For example, in some videos, the majority of the content is water surface or clear sky. These kinds of videos are difficult for motion estimation algorithms, including ours, to work correctly. These bad motion estimation can cause high homography errors defined in Equation 3 and mislead our algorithm to discard those video frames. Another type of failure case is that the videos are of very bad image quality. These videos have high image visual quality errors defined in Equation 4. This causes our algorithm to use only a small number of frames to stitch a panorama, leading to small field-of-view panoramas. Since our algorithm only outputs panoramas with area at least double of the original frame size, these panoramas are discarded.

Table 1 and Figure 6, 8, 9 and 10 show that our algorithm discovered proper panoramas for all the queries. Particularly, 86.7% of the top 10 retrieved videos on these queries contain panoramas. This discovery is encouraging, demonstrating the feasibility of obtaining panoramas from the large amount of existing web videos although we admit that these queries are far from a comprehensive sampling of queries for panoramas. Actually, the behavior of people taking a video of a scene can at least partially support this discovery. That is, people often use their camera/video recorder to scan a scene even if creating a panorama was not their intent.

Query	No Pano.	Pano.	Score
West Lake, Hangzhou	0/0	7.08/10	7.08
Lake Louise, Canada	1/1	8.5/9	9.5
Vancouver Beach	4/4	5.67/6	9.67
Delicate Arches National Park	1/1	7.75/9	8.75
Hongkong Skyline	1/1	7/9	8
Notre Dame, Paris	1/1	8.1/9	9.1

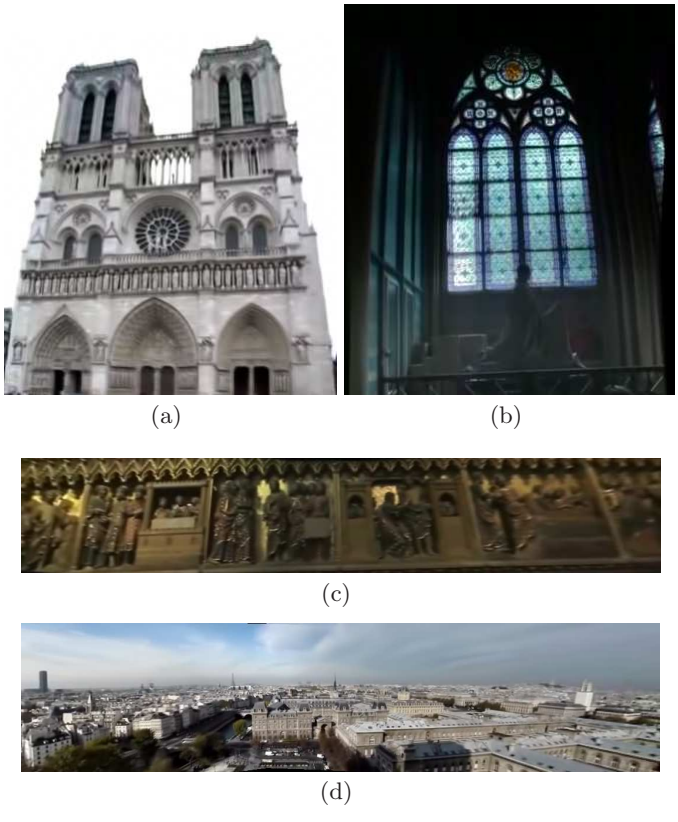
Table 1: Query panorama from *YouTube*. In column *No Pano.*, the denominator is the number of videos that contain no panorama and the nominator is the score of our algorithm on these videos. Similarly, in column *Pano.*, the denominator is the number of videos that contain panoramas, and the nominator is the score of algorithm.

6. CONCLUSION

While stitching panoramas has become an easy task for users thanks to available tools, obtaining panorama sources remains a burden. Also while panoramic imagery can be an effective representation for image database or videos, finding the valid sources from those imageries to create effective panoramas still remains a challenge. In this paper, we presented an automatic method to discover panorama sources from casual videos. We presented three criteria for valid panorama sources, namely the field-of-view covered by the sources, the mosaicability of the sources and the visual quality of the sources. Based on these criteria, we formulate discovering panoramas from a video as an optimization problem that aims to find a series of video segments that achieve optimal balances between maximizing the visual quality of the resulting panoramas and maximizing the scenes the panoramas cover. After determining these sources, we stitch scene panorama or activity synopsis according to the video dynamics analysis. We also explore the source image quality measures to guide panorama synthesis to obtain high quality panoramic images. Our experiment of "Query panoramas from *YouTube*" supports our proposal of using web videos as panorama sources and demonstrates the initial success of the presented method.

The presented method enables users to explore massive amounts of existing imageries to find useful sources for stitching panoramic imageries. More importantly, this method can significantly contribute to presenting or summarizing imagery databases using panoramic imageries by mining the possible sources to synthesize the representations.

The presented method discovers panoramas from videos and the method by Brown and Lowe [2, 3] recognizes panora-



(a) (b) (c) (d)

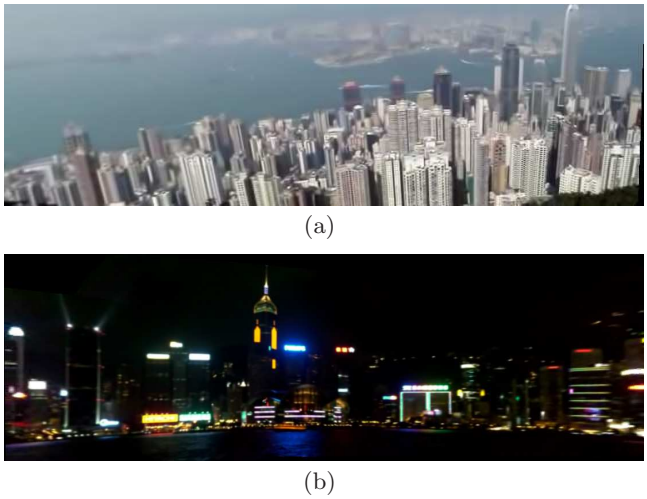
Figure 8: Panoramas on *Notre Dame, Paris*

mas from photo collections. Although these methods work well, combining different imagery sources, including still images and videos, can lead to better panoramas. Usually still images have a better image quality than videos, while videos are more likely for the vision algorithms to provide accurate image alignment. Most recently, Bhat *et al.* [1] enhance videos of a static scene using photographs. Similarly, in future extending current methods to explore the respectively advantages of different imageries can lead to obtaining better panoramic images.

Our current method projects all the frames into a plane, which is reasonable since a panorama image is a 2D static image. However, when there exist large camera rotations, such as large panning, the planar projection surface will introduce serious distortion for frames far away from the reference surface. Adaptive manifold based on methods can relieve this problem [21].

The low quality of web videos currently is a problem to make a high-quality panorama. It makes accurate motion estimation difficult. Also, it limits the image quality of the final panoramas. Our algorithm tries to relieve this problem by selectively using high-quality video frames to stitch the final panoramas. But we have to admit that our scheme cannot be a final cure. Fortunately, web video sharing services, like *YouTube*, are beginning to improve the source video quality[2]. We believe with the advance of hardware facilities, such as network bandwidth, the web video quality

[2]http://googlesystem.blogspot.com/2008/03/youtube-tests-higher-resolution-videos.html



(a) (b)

Figure 9: Panoramas on *Hong Kong Skyline*

will be improved further more in the near future. Thus our algorithm will be even more useful.

Nevertheless, as suggested by our experiments, the large amount of web imagery sources provide a rich resources for creating visual representations, such as panoramas. Our method of discovering panoramas from these often casually recorded videos provides a brave and initially successful exploration.

Acknowledgements. We thank reviewers for their constructive suggestions. The copyright of videos in this project belongs to their respective owners on *Youtube* as detailed in the project web site. This research was sponsored in part by NSF grant IIS-0416284.

7. REFERENCES

[1] P. Bhat, C. L. Zitnick, N. Snavely, A. Agarwala, M. Agrawala, B. Curless, M. Cohen, and S. B. Kang. Using photographs to enhance videos of a static scene. In *Eurographics Symposium on Rendering*, 2007.

[2] M. Brown and D. Lowe. Recognising panoramas. In *IEEE ICCV*, pages 1218 – 1225, 2003.

[3] M. Brown and D. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision*, 2007.

[4] P. J. Burt and E. H. Adelson. A multiresolution spline with application to image mosaics. *ACM Transactions on Graphics*, 2(4):217–236, 1983.

[5] Y. Caspi, A. Axelrod, Y. Matsushita, and A. Gamliel. Dynamic stills and clip trailers. *The Visual Computer*, 22:642–652, 2006.

[6] L.-H. Chen, Y.-C. Lai, and H.-Y. Liao. Video scene extraction using mosaic technique. In *IEEE ICPR*, pages 723–726, 2006.

[7] C. Cotsaces, N. Nikolaidis, and I. Pitas. Video shot detection and condensed representation. a review. *Signal Processing Magazine, IEEE*, 23(2):28–37, March 2006.

[8] N. Damera-Venkata, T. Kite, W. Geisler, B. Evans, and A. Bovik. Image quality assessment based on a

(a)

(b)

(c)

(d)

Figure 10: Panoramas on *Vancouver Beach*

degradation model. *IEEE Transactions on Image Processing*, 9(4):636 – 650, 2000.

[9] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[10] Flickr. http://www.flickr.com/.

[11] P. Heckbert. Fundamentals of texture mapping and image warping. Master's thesis, UC-Berkeley, Computer Science Division, EECS Department, 1989.

[12] http://en.wikipedia.org/wiki/Panorama.

[13] M. Irani, P. Anandan, and S. Hsu. Mosaic based representations of video sequences and their applications. In *IEEE ICCV*, pages 605–611, 1995.

[14] Y. Ke, X. Tang, and F. Jing. The design of high-level features for photo quality assessment. In *IEEE CVPR*, pages 419–426, 2006.

[15] D. Kimber, J. Foote, and S. Lertsithichai. Flyabout: spatially indexed panoramic video. In *ACM Multimedia*, pages 339–347, 2001.

[16] X. Li. Blind image quality assessment. In *IEEE ICIP*, pages 449 – 453, 2002.

[17] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.

[18] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of IJCAI*, pages 674–679, 1981.

[19] R. Marzotto, A. Fusiello, and V. Murino. High resolution video mosaicing with global alignment. In *IEEE CVPR*, pages 692–698, 2004.

[20] Y. Matsushita, E. Ofek, X. Tang, and H.-Y. Shum. Full-frame video stabilization. In *IEEE CVPR*, pages 50–57, 2005.

[21] S. Peleg, B. Rousso, A. Rav-Acha, and A. Zomet. Mosaicing on adaptive manifolds. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(10):1144–1154, 2000.

[22] M. Rasheed, Z.and Shah. Scene detection in hollywood movies and tv shows. In *IEEE CVPR*, pages 343–348, 2003.

[23] D. Steedly, C. Pal, and R. Szeliski. Efficiently registering video into panoramic mosaics. In *IEEE ICCV*, pages 1300–1307, 2005.

[24] G. Strang. Wavelet transforms versus fourier transforms. *Bull. Amer. Math. Soc.*, 28:288–305, 1993.

[25] R. Szeliski. Video mosaics for virtual environments. *IEEE Computer Graphics and Applications*, 16(2):22–30, 1996.

[26] R. Szeliski. Image alignment and stitching: a tutorial. *Found. Trends. Comput. Graph. Vis.*, 2(1):1–104, 2006.

[27] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. In *ACM SIGGRAPH*, pages 251–258, 1997.

[28] Y. Taniguchi, A. Akutsu, and Y. Tonomura. Panoramaexcerpts: extracting and packing panoramas for video browsing. In *ACM Multimedia*, pages 427–436, 1997.

[29] L. Teodosio and W. Bender. Salient stills. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 1(1):16–36, 2005.

[30] C. Tomasi and R.Manduchi. Bilateral filtering for gray and color images. In *IEEE ICCV*, pages 839 – 846, 1998.

[31] H. Tong, M. Li, H. Zhang, and C. Zhang. Blur detection for digital images using wavelet transform. In *IEEE ICME*, 2004.

[32] H. Tong, M. Li, H.-J. Zhang, and C. Zhang. No-reference quality assessment for jpeg2000 compressed images. In *IEEE ICIP*, pages 24–27, 2004.

[33] B. Vatti. A generic solution to polygon clipping. *Communications of the ACM*, 35(7):56–63, 1992.

[34] Z. Wang, A. Bovik, and B. Evans. Blind measurement of blocking artifacts in images. In *IEEE ICIP*, pages 981–984, 2000.

[35] Z. Wang, H. Sheikh, and A. Bovik. No-reference perceptual quality assessment of jpeg compressed images. In *IEEE ICIP*, pages Vol I: 477–480, 2002.

[36] Z. Wang, G. Wu, H. Sheikh, E. Simoncelli, E.-H. Yang, and A. Bovik. Quality-aware images. *IEEE Transactions on Image Processing*, 15(6):1680 – 1689, 2006.

[37] W.-Q. Yan and M. S. Kankanhalli. Detection and removal of lighting & shaking artifacts in home videos. In *ACM Multimedia*, pages 107–116, 2002.

[38] YouTube. http://www.youtube.com/.