Discovering Panoramas in Web Videos

Apoorva Bansal, Jay Patel, Vivek Gokuldas

December 14, 2012

Abstract

In this project, we present a method to discover panoramic source frames within web videos. The project is based on existing paper [1]. Videos are a good source for panoramas as camera panning naturally creates a wide field-of-view of a scene. But the problem is challenging because many of the videos are recorded casually, without the intent of capturing a panorama. Our method finds segments within a video that are good candidates for panorama sources. Finding these segments requires extracting the different scenes in the video. After finding these segments, we keep the ones that achieve an optimal balance between maximizing the visual quality and the wide field-of-view. Once these segments are discovered, good quality panoramas are synthesized using them. Our method was able to detect good quality panoramas in videos with high accuracy. We ran experiments on various web videos to confirm its feasibility and to demonstrate the utility of this method.

1 Introduction

A panoramic image is a wide angle view of a particular scene. Panoramic imagery can be found in many places and has been used throughout history as a method to showcase large scenes in their entirety. Often, it is difficult to take photographs of a wide field-of-view without specialized equipement. As a result, research in computer vision has identified methods in panorama stitching that combine multiple images with some overlap into a single panoramic image. Videos provide a source in which the likelihood of a panorama is high; when a video pans, each frame overlaps with the previous to form a series of images with a wide field-of-view.

Panorama stitching algorithms consist of two main steps; aligning images together by calculating homography values between image pairs followed by blending them together to provide a clean transition between images. Most algorithms also require that the input images have been identified as high quality and are supplied in the correct order ready for alignment and blending. Additionally, panoramas are restricted by the supply of photographs of the physical destination.

In these times, there exist vast caches of pictures and videos made available online through services such as YouTube [REFERENCE], Vimeo [REFERENCE], Picasa [REFERENCE], and many more. These resources can provide access to multimedia content pertaining to many places around the world. Video sources have especially become common for many of the popular places around the world, with many of these having the potential for becoming future panoramas. Some of these videos are not of high quality and thus cannot be used to create a panorama. In an effort to identify these videos, we use three conditions; a video must cover a wide field-of-view, the video must satisfy the thresholds for each image quality critereon, and that the video provides frames that relate to each other through homographies and are not an individual collection of separate images.

In this paper, we apply the strategy of panorama discovery in web videos identified in Liu et al [REFERENCE] with some improvements to shot detection and panorama discovery. A large part of our implementation focuses on detecting panoramas compared with the majority of research being done on panorama stitching. This focus is important as it allows users to automate the process of readying media for panorama stitching and provides a framework for discovering panoramas automatically from videos. We have split our design into five specific parts, shot detection, homography calculation, quality calculation, panorama discovery, and image stitching. The overall flow and how these parts are connected is shown in [FIGURE XX].

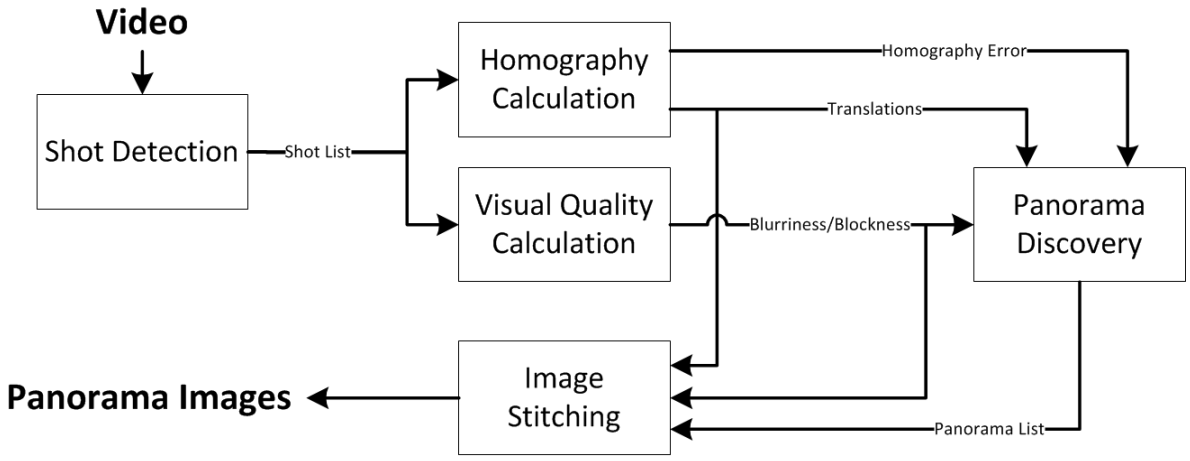


Figure 1: High Level Diagram.

2 Contributions

2.1 Previous Work

2.2 Improvements

3 Technical Solution

Our technical solution summary is discussion in Section (3.1) and details are discussed in Section (3.2).

3.1 Summary

The technical solution used to extract panoramas from web videos is split into different steps shown in Figure (1). A video clip is first divided into a sequence of shots using a shot boundary detection algorithm. A shot is defined as a continuous frame sequence recorded by a camera setting. We implemented an edge detection based method to identify shots. The next step is homography and visual quality calculation. Homographies between frames were determined using SIFT and RANSAC. We assumed no skew or rotation in the homographies to simplify our stitching implementation. Visual quality of each frame was determined using a Blurriness and Blockness algorithm. Our implementation of the Blurriness algorithm was based on a paper by Tong [2]. Our implementation of the Blockness algorithm was based on a paper by Wang [3]. The next step is panorama discovery. This is viewed as a computationally difficult optimization problem. The optimization problem is meant to find a balance between maximizing visual quality and scene extent. We implemented an approximation of this optimization problem. The last step is image stitching. We implemented a three stage blending algorithm. The stages were feathering, median bilateral filter, and visual quality weighting. The end result is the discovered panoramas from the video clip.

3.2 Details

Each major component of our technical solution shown in Figure (1) is discussed in the following subsections.

3.3 Shot Detection

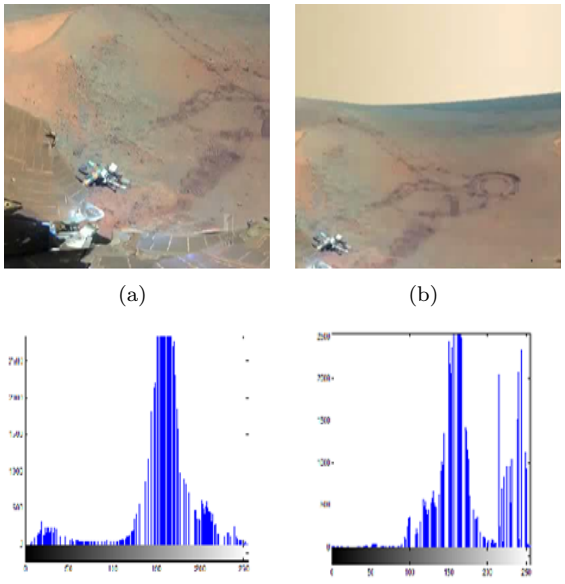
The video can be split into a sequence of shots. Each shot is defined as a continuous frame sequence of shots. Each shot is defined as a continuous frame sequence recorded by a camera setting. Different shots most likely contain different scenes. So we break a video into a shot sequence and discover panoramas from each shot independently. We evaluated couple of methods to find the shots in the video. The first method we implemented was based on color histogram-based shot boundary detection. Its basic idea is that color content does not change rapidly within but across shots. Thus, hard cuts can be detected as single peaks in the time series of the differences between color histograms of contiguous frames.

Let $p_i(r, g, b)$ be the number of pixels of color (r,g,b) in frame I_i of N pixels. Each color component is discretized to 2^B different values, resulting in r,g,b ϵ $[0, 2^{B-1}]$. Usually B is set to 2 or 3 in order to reduce sensitivity to noise and slight light, object as well as view changes. Then, the color histogram

difference between two color frames I_{i-1} and I_i is given by

$$CHD_i = \frac{1}{N}. \sum_{r=0}^{2^B-1} \sum_{g=0}^{2^B-1} \sum_{b=0}^{2^B-1} \mid p_i(r,g,b) - p_{i-1}(r,g,b) \mid \quad (1)$$

A hard cut or a scene transition is detected when CHD_i exceeds a certain threshold. This method didnt perform well on all our data sets. This is because; the color histogram approach ignores the spatial distribution information of various colors. If the shot boundary has similar RGB color composition to its previous frame, it fails to detect the scene boundary. So we implemented a second method for shot boundary detection using edge detection. Here it finds the edges using a canny edge detector on frame I_{i-1} and I_i. Next, it divided these edge detected frames into blocks. Now the difference of the mean of every block of the edge image I_i with the mean of every block of previous image I_{i-1} is taken. When this difference exceeds a threshold, a scene boundary is detected. A scene boundary detection depends on two threshold parameters, one is the difference threshold and the other threshold is the number of blocks where the difference of the means exceeds the difference threshold. After various trial and error experiments, these thresholds have been adjusted to find the hard and soft transitions. In the following sub sections, we describe the video analysis components necessary for discovering and stitching panoramas.



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

Figure 2: Example of a scene transition which is a soft cut that is undetected by color histogram method. a) and b) are two contiguous frames and b) is a scene transition boundary. c) and d) are the red channel histograms of a) and b) respectively.

3.4 Homography Calculation

The homography is determined between consecutive frames in order to align images. SIFT is used to extract feature points of every frame. We integrated the SIFT demo program provided by David Lowe [4] in our technical solution to expedite the process. The RANSAC algorithm is used to obtain a robust homography from matching points between two sequential frames. H shown in equation (2) is the constrained homography model we used in our RANSAC implementation. We assumed no skew or rotation in the homographies to simplify our image stitching implementation. As a result, there were only translations between frames. The translations are used in discovering panoramas and image stitching.

$$H = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \quad (2)$$

The homography error is used as a measure for the motion model accuracy. It is calculated as the average of second norm errors of each point. The error of each point is between the matching SIFT feature point and calculated point from the homography.

3.5 Visual Quality Calculation

Visual quality calculation is used to measure the distortion of the source image. Many web videos are of low quality for a variety of reasons. They are typically highly compressed for web-sharing, causing blurring and blocking artifacts. Image distortion is measured as a combination of blurriness extend and blockiness extent.

3.5.1 Blurriness

Our implementation of the Blurriness algorithm was based on a paper by Tong [2]. The blurriness algorithm is based on the edge type and sharpness analysis using Harr wavelet transform. Generally edges are classified into three types; Dirac Structure, Step-Structure, and Roof-Structure. In the paper by Tong [2], Step-Structure is further classified into Astep-Structure and Gstep-Structure. The algorithm measures the extent of blurriness as a ratio. The ratio is the number of likely blurred Gstep-Structure and Roof-Structure edges to the total number of Gstep-Structure and Roof-Structure edges. The classification of edges is obtained from a set of rules applied to the edge maps. The steps to acquire the edge maps are shown below. The blurriness extent is demonstrated in Figure (3(a)) and Figure (3(b)).

Step 1: Perform Harr wavelet transform to the original image and the decomposition level is 3. The result is a hierarchical pyramid-like structure shown in Figure (2).

Step 2: Construct the edge map in each scale (i=1,2,3).

$$Emap_i(k,l) = \sqrt{LH_i^2 + HL_i^2 + HH_i^2} \qquad (3)$$

Step 3: Partition the edge map and find the local maxima in each window. The window size in the highest scale is 2 x 2, the next coarser scale is 4 x 4, and the coarsest one is 8 x 8. The result will be the final edge map.

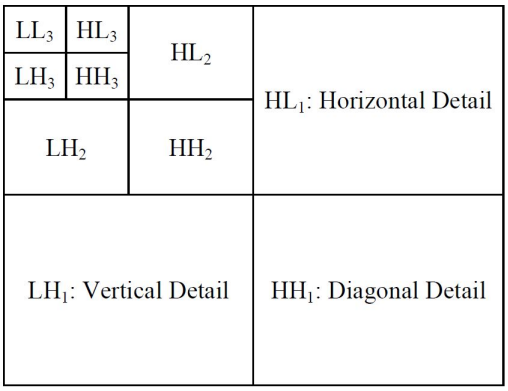


Figure 3: Pyramid of images with related subbands. [2]

(a) Blurriness = 0.2058

(b) Blurriness = 0.7557

(c) Blockiness = 0.0231

(d) Blockiness = 0.6744

Figure 4: Blurriness and Blockiness Results.

3.5.2 Blockiness

Our implementation of the Blockiness algorithm was based on a paper by Wang [3]. The blockiness algorithm is designed as a No-Reference quality measurement algorithm for JPEG compressed images. Blockiness is estimated as the average difference across block boundaries. B_h is the horizontal blockiness and B_v is the vertical blockiness.

$$B_h = \frac{1}{M([N/8]-1)} \sum_{i=1}^{M} \sum_{j=1}^{[N/8]-1} |d_h(i,8j)| \qquad (4)$$

$$d_h(m,n) = x(m,n+1) - x(m,n) \tag{5}$$

$$B_v = \frac{1}{N([M/8]-1)} \sum_{i=1}^{[M/8]-1} \sum_{j=1}^{N} |d_v(8i,j)| \tag{6}$$

$$d_v(m,n) = x(m+1,n) - x(m,n) \tag{7}$$

The final blockiness model is given in the equation below. We used a value of 0.01 for β and 1.0 for γ. We also took into account the activity of the image signal as discussed in the paper [3]. The blockiness extent is demonstrated in Figure (3(c)) and Figure (3(d)).

$$S = \beta \left(\frac{B_h + Bv}{2} \right)^{\gamma} \tag{8}$$

3.6 Panorama Discovery

Panorama discovery is defined as finding frames in a video that constitutes a panorama. Videos may contain frames that are not panoramic source images. We consider a video segment with the following properties to be a valid panorama source. A video segment is defined as a series of sequential frames. There may be multiple video segments within a shot.

Property 1: A video segment should cover a wide field-of-view based on the definition of panorama imagery.

Property 2: A video segment should be mosaicable. Specifically, the underlying camera motion between frames should observe a certain camera motion model. We use a homography to model the motion between frames. This property should be met after the shot detection algorithm.

Property 3: The frames in a video segment should have high image quality. This conservative strategy is adopted so we can ignore methods to improve image visual quality, such as de-blurring and de-blocking.

Ideally, we want to discover panoramas that have a very wide field-of-view and are of very high visual quality. Unfortunately, there is a tradeoff in practice. More frames need to be included in order to have a wider field-of-view. However, including more source frames can often degrade the visual quality due to motion estimation error. We formulate discovering panoramas from a video shot as an optimization problem. The optimization problem 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. We implemented a variation of the original optimization problem presented in the paper by Feng [1]. An overview of the orginal optimization problem is described below followed by a detailed explaination of our modified optimization problem. Lastly, our implementation is discussed in greater detail.

3.6.1 Original Optimization Problem

The original optimization problem presented in the paper by Feng [1] is shown in (9). Directly solving the constrained nonlinear programming problem is difficult. In fact the authors of the paper approximated the optimal solution. Details of their approximation is discussed in paper.

$$\hat{S} = arg\min_{S} \{ \sum_{S_i \epsilon S} E_v(S_i) + \sum_{S_i, S_j \epsilon S} E_a(S_i, S_j) \} \tag{9}$$

$$where\ S = \{S_i \mid S_i \subseteq V\},\ \forall S_i, S_j \epsilon S,\ S_i \cap S_j = \emptyset$$

$$s.t. \begin{cases} E_v(S_i) < \delta, \forall S_i \epsilon V \\ \varepsilon(S_i) > \beta A, \forall S_i \epsilon V \end{cases}$$

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, and $E_a(S_i, S_j)$ is the cost of splitting a panorama from $S_i \cup S_j$ into smaller ones from S_i and S_j respectively. $\varepsilon(S_i)$ 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 gaurantee the visual quality of the panorama, the visual quality distortion $E_v(S_i)$ is required to be less than a threshold δ.

3.6.2 Modified Optimization Problem

We relaxed the optimization problem as shown in equation (9) to the two step problem shown in equations (10) and (11). This difference provides our final algorithm with a major benefit. The benefit is that $E_a(S_i, S_j)$, the cost of splitting a panorama, is no longer considered in Step 1, which reduces the computational difficulty. Instead Step 2 aims to merge segments that greatly overlap. In our

modified optimization problem we consider overlapping segments S_i of the video clip V as part of the set S. The parameter α is used to tune the extent of overlap required for segments to be merged. This is useful because to some extent identifying panoramas is subjective to the user. This will be discussed in greater detail in Section (4.1).

Step 1:

$$\hat{S} = arg\min_{S}\{\sum_{S_i \epsilon S} E_v(S_i)\} \tag{10}$$

$$where\ S = \{S_i \mid S_i \subseteq V\}$$

$$s.t. \begin{cases} E_v(S_i) < \delta,\ \forall\ S_i \epsilon V \\ \varepsilon(S_i) > \beta A,\ \forall\ S_i \epsilon V \end{cases}$$

Step 2:

$$P = 2^S \tag{11}$$

$$s.t. \{\varepsilon(P_i \cap P_j) < min(\alpha\varepsilon(P_i), \alpha\varepsilon(P_j)),\ \forall\ P_i, P_j \epsilon V$$

S denotes a set that may contain 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. $\varepsilon(S_i)$ denotes the extent of the scene or total area in S_i, which is required to be bigger than β times the original video frame size A. To gaurantee the visual quality of the panorama, the visual quality distortion $E_v(S_i)$ is required to be less than a threshold δ. P is the power set of S that meets the constraint of equation (11). The constraint is for any two subsets P_i, P_j of P, the area of their intersection must be less than $\alpha\varepsilon(P_i)$ and $\alpha\varepsilon(P_j)$. P is the final set of panoramas, which may contain overlapping segments.

As mentioned, $E_v(S_i)$ is the visual quality cost of stitching a panorama from S_i. This is measured from two aspects: the incorrectness of the motion model denoted as $E_{vm}(S_i)$, and the source image visual quality distortion denoted as $E_{vv}(S_i)$.

$$E_v(S_i) = \alpha_m E_{vm}(S_i) + \alpha_v E_{vv}(S_i) \tag{12}$$

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

The motion model usually cannot be perfectly accurate when describing the correspondence between two frames. Since we use a homography for our motion model, homography error is used to measure the incorrectness of the motion model. $E_{vm}(S_i)$ is the sum of the homography errors between adjacent frames in S_i as shown in equation (13).

$$E_{vm}(S_i) = \sum_{I_k, I_{k+1} \epsilon S_i} H_{error}(I_k, I_{k+1}) \tag{13}$$

where I_k is a frame and H_{error} is the homography error between two adjacent frames. Homography error calculation is discussed in Section (3.4).

Source frames often suffer from compression distortion. We measure the input visual quality distortion using the blockiness and blurriness discussed in Section (3.5). $E_{vv}(S_i)$ is the sum of weighted blurriness extent and blockiness extent of each frame in S_i as shown in equation (14).

$$E_{vv}(S_i) = \sum_{I_k \epsilon S_i} \gamma q_{bk}(I_k) + (1-\gamma) q_{bk}(I_k) \tag{14}$$

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.

3.6.3 Implementation

Our implementation is discussed in greater detail due to the complexity of the optimization problem. The first step in our implementation is to find the set S. S_i candidates are determined by selecting a reference frame as shown in Figure (4) and then appending adjacent frames until the threshold δ is met. All possible reference frames are considered to ensure a complete solution. The set S is determined from the set of S_i candidates by verifying the scene extent is sufficient. The second step is to merge video segements S_i that greatly overlap. The parameter α is used to tune the extent of overlap. The set of segments P are the discovered panoramas.

I. Determine the set S

1. Select a Reference Frame I_k and place it in set S_i.
2. Determine the left most frame I_l and right most frame I_r in set S_i.
3. Find the minimum of $E_v(I_l)$ and $E_v(I_r)$.
4. Add I_l or I_r to the set S_i depending on the results of the previous step.
5. Repeat steps 2 - 4 until the error threshold δ is met.
6. If $\varepsilon(S_i) > \beta A$ then add S_i to S.
7. Repeat steps 1 - 6 for all possible Reference Frames.

II. Determine the set P

1. Find the power set of S so that Equation (11) is satisfied.

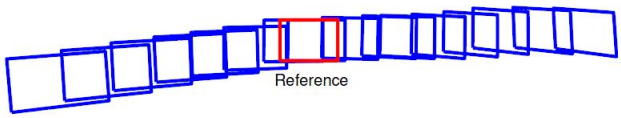


Figure 5: All the frames in the video segment are aligned to the reference frame (in red). The union of all these aligned quadrilaterals is the area covered by the video segment. [1]

3.7 Image Stitching

Image stitching consisted of two main parts; aligning individual images together based off of the homographies and then blending the aligned images together into the final panorama.

3.7.1 Alignment

Image alignment depends solely on the homography between two individual frames. In our implementation we calculate the homography between two consecutive frames and extract the associated translations; this allows us to align the images in preparation for blending later on. From the homography calculation, we disregard any knowledge of skew and rotation and thus maintain the rectangular proportion of each frame. This allows alignment to be simplified in addition to easier blending calculations explained in the next section.

3.7.2 Blending

Image blending is an important part of the stitching process. The method used in blending directly relates to the quality of the final panorama produced. The blending process we utilized consisted of three steps, taking a distance-weighted average for each pixel value, median-bilateral filtering, and incorporating the visual quality measurement. The overlapping area between frames consist of multiple pixels that need to blended together. The overall equation that is used is as follows:

$$I^P(X) = \sum_k w_k(x)\tilde{I}_k(x) / \sum_k w_k(x) \qquad (15)$$

Where $I_P(X)$ is the pixel value at index x for panorama number P, $w_k(x)$ is the weighting for the pixel at index x in frame number k, and $\tilde{I}_k(x)$ is the intensity of the pixel at index x in it's corresponding frame k. The weighting for each pixel is the product of three calculations:

$$w_k(x) = w_{k1}(x)w_{k2}(x)w_{k3}(x) \qquad (16)$$

The final weighting for each pixel is the product of three calculations, $w_{k1}(x)$ is the shortest horizontal distance of the pixel x from the nearest vertical edge; this is done to give more weight to those pixels to the center of the image than those at the edge. This calculation is represented by the following equation:

$$w_{k1}(x) = \|\arg\min_y \{\|y\| \ \tilde{I}_k(x+y) \text{ is invalid}\}\| \qquad (17)$$

Following which $w_{k2}(x)$ is the weighting as a result of a median-bilateral filter which takes into account the average (median) value of all the pixels that overlap, from different frames, in the final panorama; this is calculated using the equation shown below:

$$w_{k2}(x) = exp(-(\tilde{I}_k(x) - med(x))^2/\sigma^2) \qquad (18)$$

Lastly, the third term of the weight calculation, $w_{k3}(x)$, takes into account the individual quality of each frame that the pixel belongs to. The pixels that come frame a frame that is of a higher quality contribute a higher weight to the final result; this is shown in the equation below:

$$w_{k3}(x) = exp(-(\gamma q_{bk}(k) + (1-\gamma)q_{br}(k))) \qquad (19)$$

This blending method is applied to each color (RGB) in turn and the resulting image is the blended panorama.

4 Experiments

The experimental results of our shot detection algorithm is shown in Section (**??**), panorama discovery algorithm is shown in Section (4.1), and blending algorithm is shown in Section (4.2).

4.1 Panorama Discovery

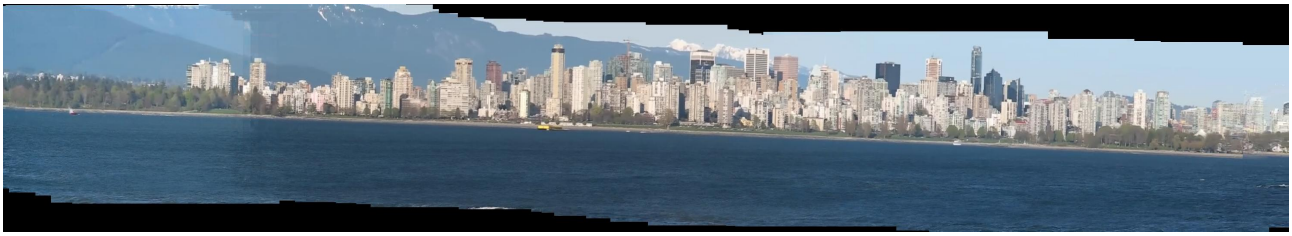
Panorama discovery is defined as finding frames in a video that constitutes a panorama. The solution may be subjective. An example is shown in Figure (5). An argument can be made that there should be only one panorama. However, the quality of the panorama may suffer due to the motion model error. The parameters can be tuned, specifically δ and β, to get different results.

The implemented panorama discovery algorithm is robust enough to account for the video zoom. An example is shown in Figure (6).

(a) Panorama 1.

(b) Panorama 2.

Figure 6: Discovered Panoramas from a Web Video [5].



(a) Panorama 1.

(b) Panorama 2.

Figure 7: Discovered Panoramas from a Web Video [6].

(a) Without Median-Bilateral Filter.

(b) With Median-Bilateral Filter.

Figure 8: Effects of the median-bilateral filter on the final stitched image.

4.2 Blending

We implemented a three stage blending algorithm as discussed in Section (3.7). The first stage is a feathering algorithm, common for stitching panoramas. Feathering algorithms tend to leave ghosting artifacts shown in Figure (7(a)). Ghosting artifacts are a result of moving objects between frames. This is a major problem since we use videos as our panorama sources. Videos tend to have moving objects. Our solution was to implement a median-bilateral filter in the second stage. The median-bilateral filter mitigates ghosting artifacts as shown in Figure (7(b)).

5 Conclusion

References

[1] Feng Liu, Yu-hen Hu and Michael Gleicher. Discovering panoramas in web videos. ACM Multimedia 2008, Vancouver, Canada, October 2008. pp. 329-338.

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

[3] 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.

[4] David Lowe. SIFT Keypoint Detector. http://www.cs.ubc.ca/ lowe/keypoints/.

[5] Delicate Arch. Retrieved December 15, 2012, http://www.youtube.com/watch?v=ZMbr8uRk3nM.

[6] Jericho Beach in Vancouver British Columbia. Retrieved December 15, 2012, http://www.youtube.com/watch?v=BrUgzmgDVw8.