Universiteit Leiden

Computer Science

Panoramic Image Stitching

Ye Ji (S0938866)

24/08/2011

MASTER'S THESIS

Supervisor: Dr. Michael S. Lew

Leiden Institute of Advanced Computer Science (LIACS)
Leiden University
Niels Bohrweg 1
2333 CA Leiden
The Netherlands

Panoramic Image Stitching

Ye Ji

Abstract—This thesis investigates the panoramic image stitching problem in computer vision area. The panoramic image stitching algorithms are divided into image alignment and image stitching approaches. The former can find the correspondence relationships among images with certain degrees of overlap, while the latter takes the correspondence relationships estimated by the registration algorithm and uses blending method to blend images in a seamless manner. Furthermore, image stitching algorithm is able to solve blurring or ghosting effects resulted from scene movements and exposure differences. This paper adopts the feature-based alignment algorithms and blending algorithms to produce high-quality image mosaics.

Index Terms—image mosaics, feature correspondence, registration, pixel labeling.

1 Introduction

Panoramic Image Stitching commonly refers to combining a set of pictures with overlaps to a larger high-resolution panoramic image. It has been well studied with extensive research literature in the field of computer vision. Currently both commercal and free softwares (eg. ptGUI, AutoStitch, Autopano) exist to servie this purpose with varying degrees of effectiveness and success.

Image stitching can be applied in various areas, including medical image analysis, video stabilization [2], video compression [3], panorama creation, etc. Among these applications, panorama creation is a facinating topic which attracts lots of attention. The panorama's field-of-view (FOV) is 360*180, while the FOV of a common compact camera is only 50*35. Therefore, we need connect a number of images that indicate different parts of a scene in the original spatial sequence in order to get a full view. There are many applications which provide people living convinience and brand new experiences. For instance, Google Street View, providing panoramic views of streets, gives the users the exprience of walking along the street while browsing the map indoor. Gigapixel image, supported by panoramic image stitching algorithms, is one-billion-pixels digital image created by making mosaics of a huge number of high-resolution digital images. Gigapixel image is quite high-resolution panoramic, allowing us to zoom in/out in very large extent and capture the very details of images.

Generally, panoramic image stitching problem can be divided into two main research areas - image alignment and image stitching problems, depending on the algorithems used. The former is able to discover the geometric relationships among images with certain amount of overlap. The latter concatenate all the aligned images and employ blending algorithms to generate the final result, and in the meantime deal with mutiple problems, such as blurring or ghosting caused by scene and parallax movement, different image exposures as well as distortions caused by camara lens so that seamless high-quality panoramas can be achieved[4].

Image alignment (Image registration) algorithms can be classified into two types: pixel-based (direct) [5], [6], [7], [10] and feature based [1], [8]. Pixel-based methods use estimated transforms to map the images relative to each other and compare the intensity distance in the overlap part by minimizing suitable error metric. Since the contribution of every pixel in the images is measured when employing an appropriate metric and image feature, it is possible to obtain very accurate alignment results. However, the computational load becomes heavier if the size of the overlap part increases. Feature-based methods, on the other hand, match the feature points extracted from each image to find point correspondences of a pair of images, and then use these point correspondences to estimate the geometric transformation between the two images. Nowadays, since the features tend to carry the properties of scale-invaiant, orientation-invairant, besides, the feature detection, extraction and matching schemes are more robust, the feature-based approaches have been widly investigated and used in the panoramic image stitching area. The most important advantage of feature-based methods lies in its capability of "recognising panorama" [1][8], the process of generating the final panorama from an unordered set of images, even with the presence of noise images that do not belong to the final panorama. Image stitching algorithm involves employing blending algorithm to blend the final panorama onto some chosen surface (planar, cylindrical, spherical, etc.). Meanwhile, it bears the power to eliminate blurring or ghosting effect, balance exposure difference, and hide the seam between the stitched images. The purpose of a stitching algorithm is to produce high-quality mosaic with two properties: the final mosaic should be seamless [9], and should preserve as many original details as possible of the input images.

In this paper feature-based alignment approach is used to register the images. Scale Invariant Feature Transform (SIFT) [11] is adopeted as the descriptor. After computing the SIFT features, we are able to establish the point corre-

- *Ye Ji is with Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA, Leiden, The Netherlands.*
  E-mail: jiyeanne1986@gmail.com..
- *Michael S. Lew is with Leiden Institute of Advanced Computer Science, Niels Bohrweg 1, 2333 CA, Leiden, The Netherlands.*
  E-mail: mlew@liacs.nl.

spondeces between every two images,correspondences making the estimation of the motions (transformations) possible. However, since mismatches are inavoidable and will certainly result in incorrect or imprecise motion assessemtn, we use RANdom SAmple Consensus (RANSAC) [12] algorithm to filter out "outliers" and calculate the transformation or homography between images. In the image stitching section, we choose to project panorama on cylindrical and planar coordinate. Radial distortions are corrected to preserve the straight lines. Moreover, we exploit two methods to adjust the exposure difference – gain compensation and linear color correction. Sometimes, due to the scene or parallax movement, the images in the overlap part are likely to have ghost effects. In this case, graph cut optimization [13] is used to determine the seam between the pair images. Finally, feathering and multiresolution blending methods are implemented in order to generate the high-quality, seamless panoramic image.

The rest of the paper is organized as follows: Section 2 describes feature-based image algnment approaches. Image calibration including radial distortion correction and exposure differences correction is introduced in Section 3. In Section 4, the author presents algorithms to choose the projection surface, identify seams between each pair images, and blend the final image. Section 5 builds a framework of applying image stitching algorithm to achieve image browsing when zooming in and out of the image to a very large extent. The final section concludes this paper and briefly talks about the prospect of future research.

2 Image Alignment

Feature-based image alignment approach proceeds as follows. First, local features are extracted from each pair of images and the globle correspondences are established by matching their local image descriptors. Then, since there must exist mismatches after preliminary matching, these outliers should be weeded out and use the set of inliers to estimate a geometric transformation between images.

2.1 Local Feature

For image feature extraction, we use Lowe's Difference-of-Gaussian (DoF) [11] to detect the key points in scale-space. The image is resized to half size to generate the octives. For each octave, the initial image is processively convolved with Gaussian operator. By doing this, a set of scale space images is created. Then, calculate the difference between 2 consecutive blurred levels in one octave. Applying this operation for all the octaves generates DoG images of multiple sizes, after which the keypoints are detected by finding local extremas in DoG images. Low contrast keypoints and edges should be removed for stability. By assigning a dominant orientation to each keypoint so to achieve rotation-invariance.

The SIFT descriptor is created over 16*16 windows around the keypoint then broken into 16 4*4 windows. Within each 4*4 unit, the gradient magnitudes and orientations are calculated, and the orientations are put into 8 bins. Therefore, the result SIFT descriptor is of 128 dimensions.

2.2 Keypoint Matching

After the features are extracted from images, they must be matched to build point correspondences. The geometric transformations can be estimated once the point correspondecs are established. The best candidate match for each keypoint in one image is found by identifying the keypoint, which is with minimal Euclidean distance for the invariant descriptor vector, in another image. However, some keypoints do not have any correct match in other image, in order to discard these features; Lowe's ratio of distances [11] is introduced. A match is accepted when the ratio of distance from the closest neighbor to the distance of the second closest neighbor below a certain velue, while in Lowe's paper [11] this vlue is set to be 0.8. In our implementation, Best-Bin-First (BBF) [14] search algorithm is used to identify the approximate nearest neighbor of each keypoint.

2.2.1 Best-Bin-First

BBF is a new form of k-d tree [15] search algorithm, while still makes use of the standard k-d tree structure, which is built as follows. With a set of N keypoints, choose a dimension i in which the data is of greatest variance, then split the rest of the points into two parts according to the median value m of the chosed dimension so that each side has the equal number of points. The value of i and m are stored. This operation iterates on both side of the points to create a binary tree.

Each leaf of a k-d tree is a keypoint in the dataspace. The standard k-d tree uses backtracking, branch-and-bound scheme to search the NN of a query point p. It works as follows. Starting with the root node and moving down the tree recursively until it reaches a leaf note. It goes either part of the tree depending on wether the value of p in the split dimension is greater than or less than the value of current note. The leaf note that the algorithm reaches is marked as the candidate. Then the algorithm backtracks to check for the note that has smaller distance from the p than the candidate. The searching process terminates when the whole tree structure is traversed.

The BBF algorithm uses priority search [14] for the k-d tree algorithm. This allows searching the bins in feature space in the order of their closest distance from the query location. This search order is achieved by using a heap-based priority queue to store the distance information at the node not taken. Besides, BBF algorithm searches a fix number of nearest neighbor then stops, so to provide an approximate NN with low costs.

2.3 Geometric Transformation Estimation

Geometric transformations between every pair of overlapping images need to be calculated to align images. We need to find a set of feature correspondences that will result in a high-accuracy alignment and to use these correspondeces to estimate the motions between each input image and the final result image. To serve this purpose, RANSAC is employed to select a set of inliers that are consistent with some particular transformations. In this thesis, final image is going to be represented both on cylindrical and planar surface. With regard to the planar compositing surface, the transformation is homography, while the transformation of cylindrical surface is only translation. The details of projecting images onto cylindrical and planar compositing surfaces are described in section 4.

2.3.1 Homography

Considering two images taken from the same optical center, the relation between two overlapping images can be described by a planar perspective transformation, which can also be called homography [10]. Besides, if taking two images of one same planar surface in 3D world, these two images can also be related by a homography. For homogeneous coordinates x and x',

$$x' \sim Hx \,, \tag{1}$$

where ~ denotes up to scale, and homography H is an arbitrary 3*3 invertible matrix. At least 4 feature correspondences are needed to compute the homography. In the case of our implementation, homography is sufficient to describe the relationship between the images taken rotating about the same optical center.

2.3.2 RANSAC Algorithm

RANSAC (RANdom SAmple Consensus) first published by Fischler and Bolles [12] in 1981 is a robust estimation approach to estimate the parameters of a certain model using data with a large proportion of outliers. Inliers are the data consistent with the estimated model while outliers are not. RANSAC algorithm is applied under the two steps hypothesize-and-test framework [16]. In the hypothesize step, minimal samples sets (MSSs) are randowly picked from the dataset, and the parameters of a model is calculated only from this minimal sample set. Then it comes to the test step. In the test step, RANSAC checks which point is consistent with the model instantiated with the parameters calculated in the first step. This set of points is inliers with this model, and it is called consensus set (CS). RANSAC algorithm should iterate these two steps k times to ensure a good chance of finding a true set of inliers. Let p be the probability that any given correspondence is from the CS. Besides, the size of MSS is m, therefore the probability that all these m samples come from CS is p^m. Let P be the probability that in k-time iterations all the sets of the random samples contain at

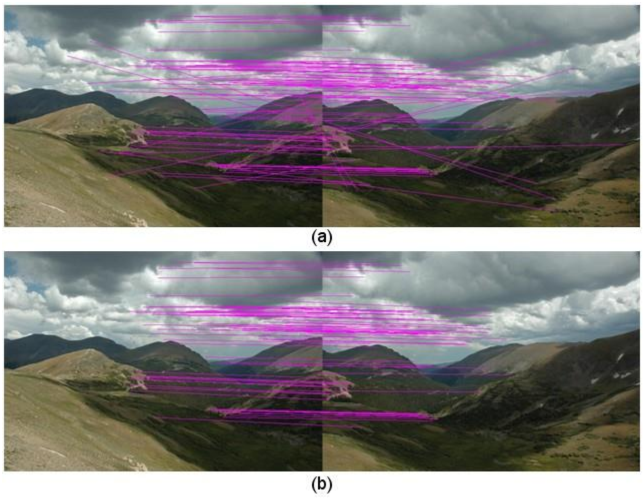


Fig. 1 (a) shows the preliminary keypoint matches, and (b) shows the inliers after applying RANSAC, where the mis-matches are pruned out.

least one outlier.

$$P = \left(1 - p^m\right)^k . \tag{2}$$

The iteration is terminated when P falls under a certain threshold η (usually set to be 0.01).

$$P = \left(1 - p^m\right)^k \leq \eta \,. \tag{3}$$

Since p is unknown beforehand, it can start with a certain value then get updated with the new maximum fraction of the inliers generated in the test step after each iteration. Fig. 1 shows the inliers correspondences after RANSAC operation.

In this paper, RANSAC is used to estimate parameters for homography and translation model, while homography is used to describe the relationships between overlapping images in planar coordinates and translation for cylidrical coordinates. We use four feature correspondences as MSS to calculate parameters for homography and one feature correspondence for translation relationship.

3 Image Calibration

Image calibration aims to minimize the differences of the input images caused by thick camera lens, large exposure differences, etc. In our implementation, we calibrate radial distortion and exposure difference.

3.1 Radial Distortion Correction

Due to the simplified lens construction and lens imperfection, many wide-angle lenses have noticeable radial distortion [18]. As a result of radial distortion, staight line in the real world is not straight in the image; instead, it appears to be curved. A highly accurate panorama would be created only if the radial distortion is removed [17]. For instance, radial distortion could cause feature mis-

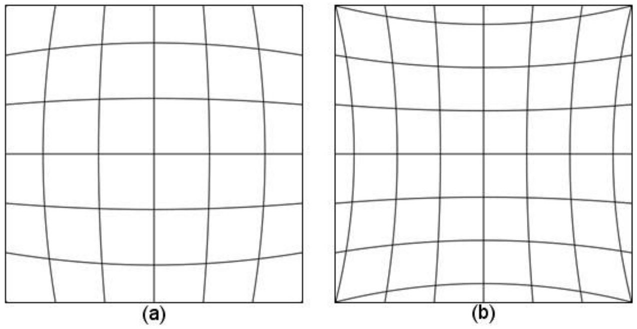


Fig. 2 Examples of radial distiortion: (a) barrel and (b) pincushion.

registration, which would lead to the blurring effect.

The radial distortion can usually be classified into two types: barrel distortion and pincushion distortion (Fig. 2 shows the example of these two types of radial distortion). In our implementation, the simplest radial distortion model is used.

$$\begin{aligned} x' &= x\left(1+k_1r^2+k_2r^4\right), \\ y' &= y\left(1+k_1r^2+k_2r^4\right), \\ r^2 &= x^2+y^2, \end{aligned} \tag{4}$$

where (x,y) is the pixel in distorted image while (x',y') is in distortion corrected image, and k_1 and k_2 are called radial distortion parameters [19].

Ratial distortion parameters can be estimated as the result of the extern camera calibration operation. However, Plumb line method [19] is a simplest way to do the work in practice. Using the camera take an image of a scene with a lot of straight lines, then the radial distortion parameter can be adjusted until all the lines appear straight. In our case, because there are two distortion coeffience, for a very simple solution, we can set k_2 to zero, and then slightly adjust k_1 until the lines in the images are straight.

3.2 Color Correction

Panoramic images are usally constructed from a sequence of images taken by rotating around the same optical center. When capturing these images, the automatic gain control in cameras often lead to different exposure levels [20], which makes it possible to have one image in dark and the next one being exposed bright. Images taken from different cameras could also have this problem. Fig. 3 shows an example, where the source images have large color and luminance difference. In the panorama construction process, color, as a very important factor to human perception, should be adjusted consistent to create a better result. In this situation, color correction should be taken to eliminate the visible contrast or color discontinuities. The essence of all the color correction algorithms is transferring color or luminance information from a source image to a target image [21]. In this paper, two approaches aims to correct color are implemented.

3.2.1 Diagonal Model

The color transformation between two overlapping images can be modeled by a linear mapping [22],

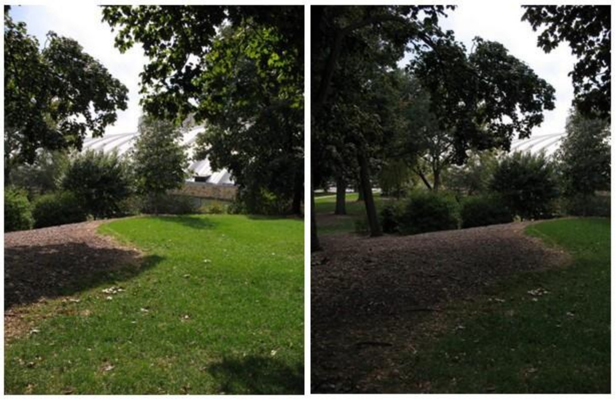


Fig. 3 Two panorama source images with exposure difference by the same camera

$$C_{c,k}(p) = \alpha C_{c,j}(p) \qquad c \in \{R,G,B\}\,, \tag{5}$$

where $C_{c,k}(p)$ and $C_{c,j}(p)$ denote the color value at pixel p in overlapped image j and k . α is the mapping coefficient.

Finlayson et al [23] suggested that a diagonal model can be employed to model a color transformation if a transformation is linear. With the diagonal model, we can represent the linear correction for color as

$$S_k = S_j * M\,, \tag{6}$$

where S_k and S_j are the overlapped area of image j and k , and M is a transform matrix crossing these two images' overlapping parts.

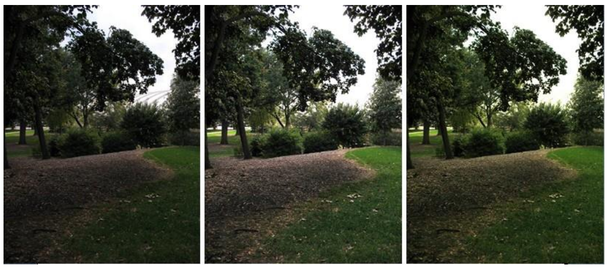
$$M = \begin{bmatrix} \alpha_R & & \\ & \alpha_G & \\ & & \alpha_B \end{bmatrix} \tag{7}$$

$$\alpha_c = \frac{mean\left(C_{c,k}\right)}{mean\left(C_{c,j}\right)} \tag{8}$$

where α_R , α_G and α_B are the coefficients of the channel R, G, B respectively, and can be calculated by the mean value of each channel in the overlapped parts of image j and k .
Then the color correction for the overlapped image can be completed for R, G and B channels respectively by

$$C_{c,k}(p) \leftarrow \alpha_c C_{c,j}(p) \qquad c \in \{R,G,B\} \tag{9}$$

The diagonal model is a simple and fast color correction approach. Accurate image registration is not needed in this case. Fig. 4b shows the result of applying diagonal model color correction to two overlapped images that of color difference.



(a) (b) (c)

Fig. 4 The results of applying color correction: (a) is the original image, (b) shows the result of diagonal model, and (c) is gain compensation. Both (b) and (c) show the results of adjusting color of the right image according to the left image in Fig. 3.

3.2.2 Gain Compensation

Gain compensation is initially used to solve the color balancing problem in panorama stitching using sequences of partial images [21]. The intensity gain level of input images is adjusted to balance the color difference. In our implement, we adopted Brown and Lowe's solution [1].

In their solution, instead of seperately calculating the pairwise color correlation coeffience, they calculate the overall gain parameters for all the images by minimizing a globle error function. The error function calculates the sum of intensity differences of the all overlapping pixels

$$E_{LS} = \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n}\sum_{p\in S_{j,k}}\left(\alpha_j * I\left(p_j\right) - \alpha_k * I\left(p_k\right)\right)^2 \qquad (10)$$

where α_j , α_k are the correction coefficients, and $S_{j,k}$ is the overlap region between image j and k . The pixel p_j and p_k are correspondent points transformed by the estimated homography

$$p_j = H_{jk} p_k \, . \qquad (11)$$

In practice, the intensity used is the mean value of the the intensities of all the pixels in the overlap region $\bar{I}_{j,k}$. By doing this, the computation would be simplified and the small misregistrations are tolerable. As we can see, in equation (10), if the α is set to 0, the error equals zero, which is definitely an optimal solution. Therefore, the error function is modified as

$$E = \frac{1}{2}\sum_{j=1}^{n}\sum_{k=1}^{n} n_{j,k}\left(\left(\alpha_j \bar{I}_j - \alpha_k \bar{I}_k\right)^2 / \sigma_I^2 + \left(1-\alpha_j\right)^2 / \sigma_\alpha^2\right) \quad (12)$$

where $n_{j,k}$ is the number of pixels in the image j overlapping with image k , σ_I is the standard deviation of the intensity while σ_α is that of color coefficient. In Brown and Lowe's paper, they set σ_I to 10.0, while σ_α to 0.1. They choose these values to balance the two parts in equation (12), since $I \in \{0..255\}$ and the values of α are around 1. Fig. 4c shows the performance of gain compensation method.

4 Image Stitching

After registering the input images, we can get the geometric transforms between each pair of overlapped images; therefore, the images can be warpped with respect to transformations. Besids, the color differences between each pair of overlapped images are balanced due to the color correction operation. In the image stitching stage, the final image should be projected onto a certain surface after it is stitched. In addition, to achieve a satisfying result without blurring, visible seams and ghosting, several techniques should be taken, for example, a suitable compositing surface to represent final image without distorting severely, better blending method to cover the differences between images and pixel labeling method to hide the seams. In this section, these techniques will be introduced respectively.

4.1 Image Warping

In our implementation, we choose to project final result to planar and cylindrical coordinates. In planar coordinate, one image is set as the reference and to warp all the other images into the reference coordinate using homographies, while in cylindrical coordinate, each image should be warpped first into cylindrical coordinate, then stitched together by some geometric model. In this situation, image warping is used to do this job.

Image warping is defined as a operation that maps a source image to a destination image according to a transformation between the source image space (u,v) and destination image space (x,y) . Image warping operation consists of image mapping and resampling method.

There are two ways to map one image to another, namely forward mapping and inverse mapping. Forward mapping is mapping each pixel in the source image (u,v) to the target image (x,y) according to some relationship f_x and f_y .

$$\begin{aligned} x &= f_x(u,v), \\ y &= f_y(u,v), \quad , \\ dest(x,y) &= src(u,v) \end{aligned} \qquad (13)$$

Since, in forward mapping, many source pixels may map to same destination pixel and hence some destination pixels might not be coverd, some "holes" may appear in the result image. In this case, inverse mapping could be used to solve this problem, as inverse mapping maps the pixels in the destination image back to source image to make sure every pixel in destination image is coverd.

$$\begin{aligned} u &= f_x^{-1}(x,y), \\ v &= f_y^{-1}(x,y), \\ dest(x,y) &= src(u,v) \end{aligned} \qquad (14)$$

The pixel coordinates calculated from the mapping operation may not always be integers, which means it might fall between the pixels. As a result, resampling method should be employed to interpolate the pixel value. In our implementation, we use nearest neighbor and bilinear

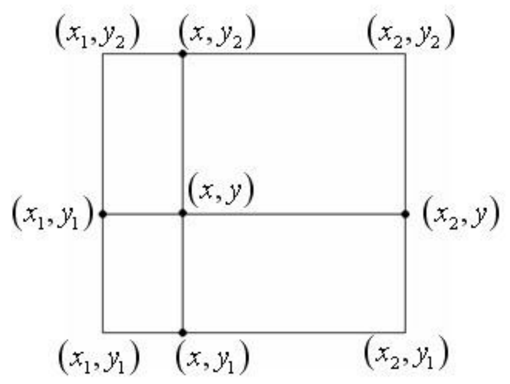


Fig. 5 Illustration of Bilinear Interpolation

interpolation method to do the pixel interpolation.

Nearest neighbor sets the intensity at position (x,y) to the intensity at $(round(x),round(y))$. This method is very fast, however, it will provide aliasing effects along edges.

Bilinear interpolation method calculates the intensity at position (x,y) using four pixels intensity values around the position (x,y). It defines as (Fig. 5)

$$\begin{aligned} I(x,y) &= w_{x_1,y_1} I(x_1,y_1) + w_{x_2,y_1} I(x_2,y_1) \\ &+ w_{x_1,y_2} I(x_1,y_2) + w_{x_2,y_2} I(x_2,y_2), \end{aligned} \quad (15)$$

where

$$\begin{aligned} w_{x_1,y_1} &= (x_2 - x)(y_2 - y), \\ w_{x_2,y_1} &= (x - x_1)(y_2 - y), \\ w_{x_1,y_2} &= (x_2 - x)(y - y_1), \\ w_{x_2,y_2} &= (x - x_1)(y - y_1) \end{aligned} . \quad (16)$$

We can resample images using bilinear interpolation without obvious aliasing; besides, the processing time is relatively slow comparing to other more complicated interpolations.

4.2 Compositing Coordinate

In our implementation, we choose planar and cylindrical coordinates as the compositing surface to represent the final images. For larger field of views, cylindrical representation is a better choice since the final image will get distorted severely when projected as a flat panorama.

4.2.1 Planar Image Mosaics

The very common set of images to stitch is the view of planar scene, for instance, a sequence of images of a whiteboard to mosaic into a larger field of view [25], or one larger-scene image as background image on which several detailed images are stitched to make these parts clearer while the rest parts of the background image get blurred, which as a framework proposed in our implementation, will be descrined in section 5. In the case of planar image mosaics, any two overlapped images are related to each other by a homography. Fig. 6 is a result of warping image using the homography related with its overlapping image.

4.2.2 Cylindrical Image Mosaics

Cylindrical panoramas are usually used due to their ease of construction. Cylindrical image stitching algorithms can only work under the condition of taking all the images by a camera mounted to a tripod to make sure that the camera is level and only rotating around its vertical axis [24]. Under these conditions, the relations between each pair of overlapped images are purely translations [5]. In addition, focal length should be known beforehand, and hence the approach of estimating focal length will be described later in this section.

As using cylindrical image mosiacing algorithms, each image should be projected into cylindrical coordinate first. The points on the cylindrical surface are parameterized by an angle θ and a height h, and the points in the 3D world can be represented by (x,y,f), where f is focal length. We now have the relations between a point in 3D world and it in cylindrical coordinates,

$$(\sin\theta, h, \cos\theta) \propto (x,y,f) \quad (17)$$

as shown in figure 3. With this relation, we can map world coordinates into 2D cylindrical coordinates (θ,h) using

$$\begin{aligned} \theta &= \tan^{-1}\frac{x}{f}, \qquad h = \frac{y}{\sqrt{x^2+y^2}}, \\ x' &= s\theta + x_c, \qquad y' = sh + y_c, \end{aligned} \quad (18)$$

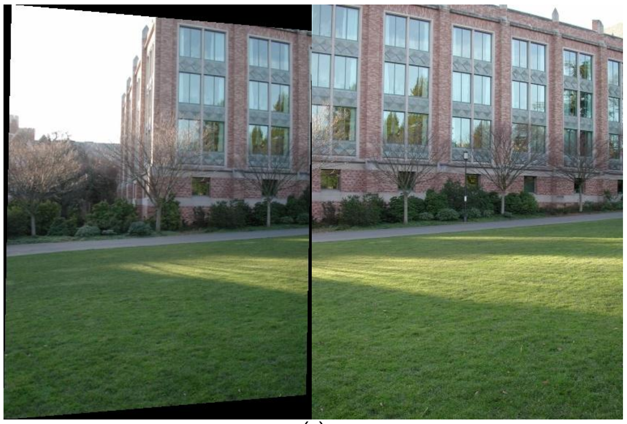
where s is an scaling factor that can be set to f so to minimize the distortion near the center of the image[4], (x_c,y_c) is image center and (x',y') is the result pixel in the cylindrical coordinate. Because we need to use inverse mapping as we discussed above, the formulas are,

$$\begin{aligned} &\theta = \frac{(x'-x_c)}{f},\ h = \frac{(y'-y_c)}{f}, \\ &\hat{x} = \sin\theta,\ \hat{y} = h,\ \hat{z} = \cos\theta, \quad . \\ &x = f\frac{\hat{x}}{\hat{z}} + x_c,\ \ y = f\frac{\hat{y}}{\hat{z}} + y_c \end{aligned} \quad (19)$$

After we get these mapping formula, we can warp each input image to cylindrical coordinate. Fig. 7 shows the original image and the warped image (radial distortion corrected). Both image warping and radial distortion need to use interpolation to calculate color value at each pixel, however, interpolation is not only computational expensive, it also smoothes the features making images aliasing or blurring. Therefore, in our implementation, we only interpolate the image once after both of these two operations done.

4.2.3 Estimating the Focal Length

In order to build cylindrical image mosaics, we need to know the focal length. We can get focal length from prior


(a)

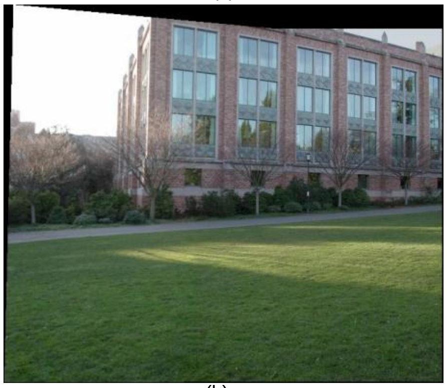

(b)

Fig. 6 (a) are the warped version of the two images, the image on the left is warped using geometric transformation according to the right image. (b) shows the two warped images stitched together.

camera calibration, but the extern camera calibration will be needed. The simplest way is estimating focal length information from EXIF tags. However, CCD width is still needed because it differs from camera to camera. There is a convient way to obtain the estimation from the homography [10].

The relationship between a 3D point $w=(X,Y,Z)$ and its image coordinate $i=(x,y,1)$ can be stated as,

$$i \sim TKRw \tag{20}$$

where T is the translation of image plane matrix, K is the simplified form of camera calibration matrix and R is the rotation matrix. They are in the form of

$$T=\begin{bmatrix}1 & 0 & c_x\\ 0 & 1 & c_y\\ 0 & 0 & 1\end{bmatrix}, K=\begin{bmatrix}f & 0 & 0\\ 0 & f & 0\\ 0 & 0 & 1\end{bmatrix}, R=\begin{bmatrix}r_0 & r_1 & r_2\\ r_3 & r_4 & r_5\\ r_6 & r_7 & r_8\end{bmatrix}. \tag{21}$$

We set the c_x and c_y to zero by assuming that the origin is at the image center.

If a camera rotates around its projection center, the homography H between two images j and k is given by,

$$H \sim K_j R_j R_k^{-1} K_k^{-1} = K_j R_{jk} K_k^{-1} \tag{22}$$

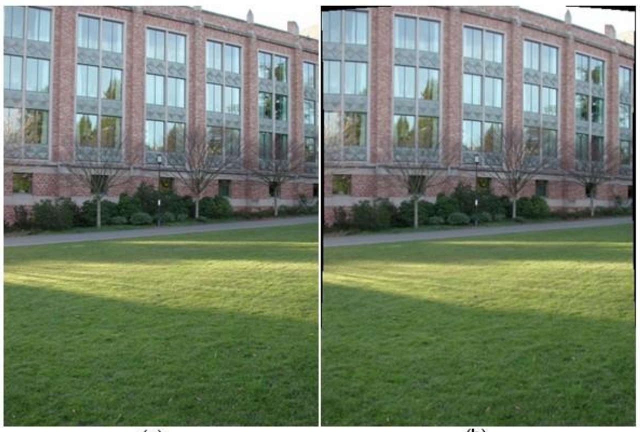


Fig. 7 Warp image (a) into cylindrical coordinate, and (b) shows the warped image.

where each image is represented by its focal length and rotation, i.e., $K_j R_j$. Now, we can re-write this formula as

$$R_{ij} \sim K_j^{-1} H K_k . \tag{23}$$

Because we use RANSAC to estimate the homographies between each pair of overlapped images, homographies are known by now. Hereby, we have

$$H=\begin{bmatrix}m_0 & m_1 & m_2\\ m_3 & m_4 & m_5\\ m_6 & m_7 & m_8\end{bmatrix} \sim \begin{bmatrix}r_0 & r_1 & r_2 f_j\\ r_3 & r_4 & r_5 f_j\\ r_6/f_k & r_7/f_k & r_8 f_j/f_k\end{bmatrix}. \tag{24}$$

According to equation (23), equation (24) can be re-writed as,

$$R_{jk} \sim \begin{bmatrix}m_0 & m_1 & m_2/f_k\\ m_3 & m_4 & m_5/f_k\\ f_j m_6 & f_j m_7 & f_j m_8/f_k\end{bmatrix}. \tag{25}$$

Since R has property of being orthogonal and the first two rows and columns of the same norm, the focal length f_j and f_k can be calculated as,

$$\begin{gathered} m_0^2+m_1^2+m_2^2/f_k^2 = m_3^2+m_4^2+m_5^2/f_k^2, \\ m_0 m_3 + m_1 m_4 + m_2 m_5/f_k^2 = 0, \end{gathered} , \tag{26}$$

and

$$\begin{gathered} m_0^2+m_3^2+m_6^2 f_j^2 = m_1^2+m_4^2+m_7^2 f_j^2, \\ m_0 m_1 + m_3 m_4 + m_6^2 m_7^2 f_j^2 = 0 \end{gathered} . \tag{27}$$

From the formulas above, we can get the focal lengths f_j and f_k, and we let the estimated focal length $f=\sqrt{f_k f_j}$ - the geometric mean of f_j and f_k. If there are multiple input images, we can have the estimate of the median value of the focal lengthes.

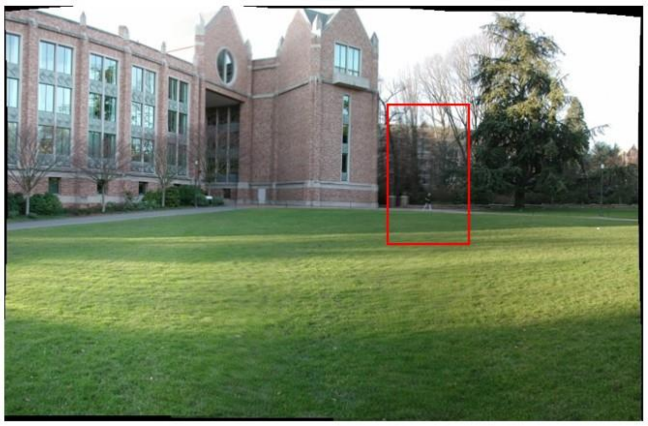


Fig. 8 This is a cylindrical image stitching combining 3 images with feathering blending method. The ghost or blurring in this image is caused by moving objects, as shown in the red rectangle the man, the trees as well as the bluring on grass. These effects exist in overlap region.

4.3 Feathering

In this paper, we implemented a feathering blending method to blend images. The pixel values in the overlapping region between two images are weighted average from each image. Considering the fact that images are less likely to be distorted near the center [26], every pixel in each image contributes differently to the blending region according to the distance from center to edge. The pixels near the center of image weight higher than that near the edge. The value of the result pixel $C_r(x,y)$ in the overlapping part is the weighted sum of pixel values of image i (on the left) and j (on the right) at correspondent position related by the transformation.

$$C_r(x,y) = (1-w)C_i(x,y) + wC_j(\hat{x},\hat{y}), \tag{28}$$

where

$$w = (x - x_{start})/(x_{end} - x_{start}). \tag{29}$$

In the above equation, x denotes the column in the overlapping region, and x_{start} denotes the column where the image on the right begins while x_{end} as the column where the image on the left ends.

Feathering processes very fast and is capable of hiding the seams and blending over exposure differences to certain extend. However, when the exposure differences are too large, color correction need to be applied via using the operation that we dicussed in the previous section. In addition, ghost and blurring effects can still be problematic, when there are moving objects in the image, especially in the overlapping part with another image. In Fig. 8 the blurring and ghost effect can be noticed in the result blended by feathering method, because that people in the input images is moving and the trees moves as well.

4.4 Graph Cut Finding Seams

The option to solve the aforementioned problem is to locate a seam in the overlapping region, which can be stated as the problem of pixel labeling. It is the process of selecting the image to be used at each pixel. As a result, we can get a series of masks of the input images, and then use these masks for Laplacian pyramid blending, which will be discussed later in this section.

This pixel labeling problems can be formulated in terms of energy minimization problems. First, we create a cost function, and then we use max-flow/min-cut to minimize this cost function to determine the seams. The output of this process will tell us, at every pixel in the compositing image, from which image the pixel should be selected. In our implementation, we use Boykov and kolmogorov's approach and implementation of max-flow/min-cut algorithm [30].

Represent the overlapping part of each pair of images by graph, which could be done by constructing a planar graph, with a node corresponding to each pixel and weighted edge connecting a pair of pixels. The edge weights shows how important to assign two neighboring pixels the same label, which means they will appear on the same side of the seam. For example, if the edge weight between two neighboring pixel is small, we can say the min-cut can break this edge and pay little penalty. If the weight is large, the min-cut is more likely to choose other path to cut.

In out implementation, we define the cost function according to the color difference between the pairs of pixels [27], [28], [29]. Let s and t be two neighboring pixels in the overlap region, and I and J be the two images that overlapped. Therefore, $I(s)$ and $J(s)$ are the colors of pixel s in image I and J respectively. This color-based cost function E can be defined as,

$$E = \sum_{s,t \in N_4} M_{I,J}(s,t), \tag{30}$$

$$M_{I,J}(s,t) = \sqrt{\left(I(s) - J(s)\right)^2 + \left(I(t) - J(t)\right)^2} \tag{31}$$

where $M_{I,J}(s,t)$ is the weight in terms of color differentces between pairs of adjacent pixels, and N_4 denotes the set of 4 neighboring pixels. By minimizing this cost function using min-cut algorithm to find where to cut, we can get seams in the overlap region where the color differecce between each pair of images are minimal, and a series of mask of each input image. The masks calculated using this graph cut algorithm are shown in Fig. 9.

4.5 Laplacian Pyramid Blending

Once we determine the seams, we still need to blend images together and compensating for color difference that has not been corrected completely in color correction phase, and other mis-alignments. The multiresolution blending algorithm introduced in [31] can be a good solution to the aforementioned problem.

The multiresolution blending algorithm consists of 3 steps:

Step1. Build Laplacian pyramids for images A and B respectively, namely LA and LB , and build Guassian pyramids for masks of image A and B,

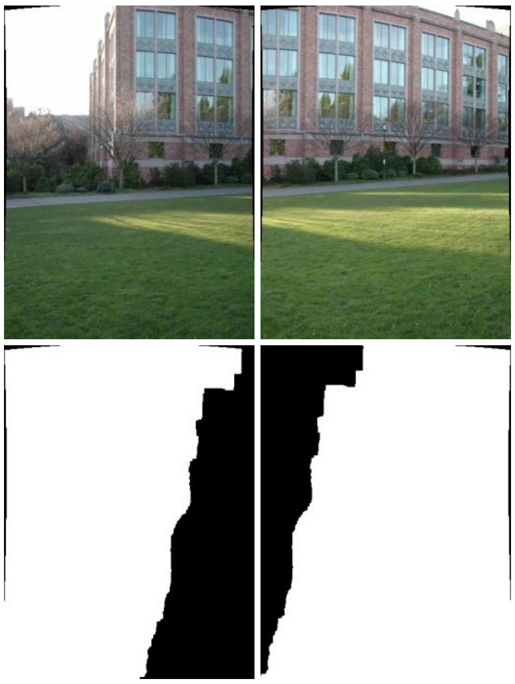


Fig. 9 The 2 images in the upper row are the 2 cylindrical warped images and the other 2 images are the masks, in which white part denotes the valid pixels and invalid pixels in black part.

namely MA and MB.

Step2. Merge LA and LB to create one pyramid LC, using the value of MA and MB as weights. That is, for each pyramid level l, pixel i:

$$LC_l(i) = MA_l(i)LA_l(i) + MB_l(i)LB_l(i) \tag{32}$$

Step3. The final image C is constructed by expanding and summing all the pyramid levels of LC.

The Laplacian pyramid in step 1 is builded as follows: First, construct a Gaussian pyramid of the original image by smoothing the image with a $\frac{1}{16}(1,4,6,4,1)$ binomial kernel and then subsampling the smoothed image by rejecting even rows and columns. By now, the image is reduced to half size. Once the Gaussian has been built, substracting the image in one level from the image in the previous level, which can be formulated as

$$LC_l = G_l - G_{l+1}, \tag{33}$$

where G_l represents the l level of Gaussian pyramid. Given the fact that, the images of different Gaussian pyramid are not of the same size, interpolation should be used to double the size of the image in $l+1$ level to make the substraction operation work. Also, in step 3, the final image is construced by summing all the levels of LC, which is the operation as

$$C = \sum_{l=0}^{N} LC_l \,. \tag{34}$$

In this operation, interpolation is also employed to make the summing work. The result of applying muiltiresolution blending is shown in Fig. 10 where compares with simply compose images with their masks. Fig. 11 shows more panorama examples using Laplacian pyramid blending.

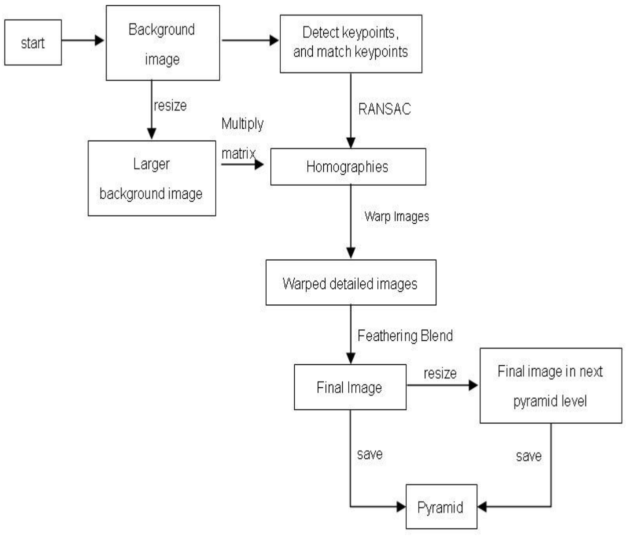


Fig. 12 The overall structure of the framework

5 DEEP ZOOM

In this section, we apply the image stitching algorithm to an application, which enables zoom in or out of certain parts of image to a very large extend. This means, when zooming in the image, some parts get increasingly clear while the rest get blurrd. To realize this function, an image pyramid is built to store the stitched images of different sizes, which aims to provide the ability of smoothly zooming in and out.

Due to the fact that the image just has a certain resolution, it will get blurred when we zooming in or out of the image. Assuming we have one background image and several images that present the detail of parts of the background image of a high resolution. Fig. 12 shows the work flow of our framework. It works as follows. We operate on the original images first. In the phase of image alignment, we detect keypoints in all these images and match. Once we get the tentative matches, run RANSAC to filter the outliers and estimate a homograpies between each detail image and the background image. Since all the images are the scene of a plane surface, homograpy is proper to describe the relations, as what we discussed in section 2. Second, the colors of the detail images are slightly adjusted to minimize the differences with the background image. After doing this, all the detail images are stitched on the background image using the simple feathering blending, which can do very good job in this case. Save the stitched image into our image pyramid as the first layer. Then we resize the background image to larger size with certain stepwise factor (in our implementation, we choose 0.2, which means the size of the image is 1.2 times larger than the original image). In order to save time, we don't have to do ketpoints detecting, matching and running RANSAC to estimate the homog-

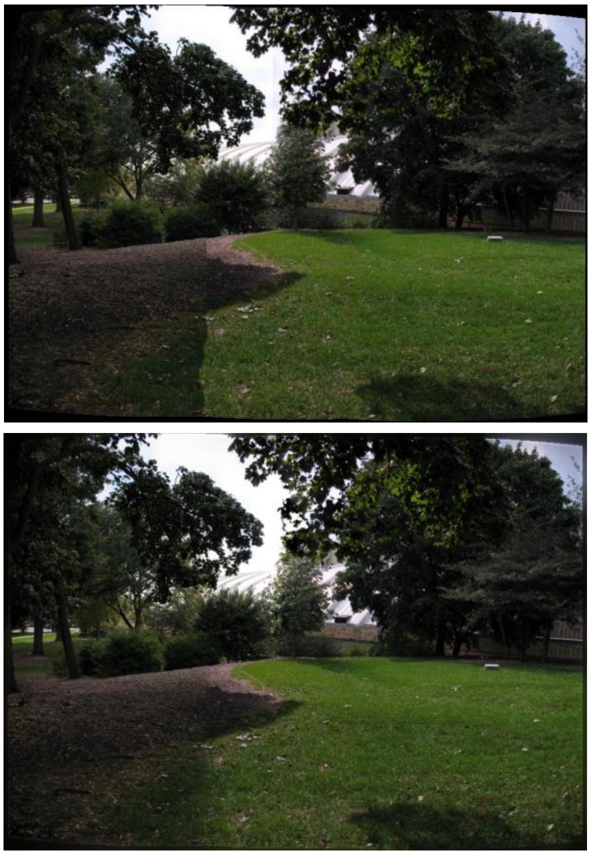


Fig. 10 (b) is the image with using Laplacian pyramid blending, (a) is just the result of compositing images according to the masks produced by graph cut.

raphy. As there are homographies of each image with the original background image, we can get new homographies to the current background just by multiplying matrixs. Stitch them together and then store the final image into the pyramid. In our implementation, the process is accelerated by just resizing the image of the highest level of pyramid. By doing this, the process of zooming in and out appears to be more smoothly.

This framework provides a new experience browsing images. Figure 11 shows examples of the application of Laplacian pyramid blending.

4 Conclusion

The current project explores the algorithms of panoramic image stitching problem. This problem is categorized into three parts in this paper: image alignment, image calibration and image stitching. The image alignment section estimates the geometric transforms between pairs of images using feature-based method. The keypoints are matched employing BBF method with taking SIFT as the keypoint detector and descriptor. This process is followed by applying RANSAC to filter out the outliers and predict geometric transformations. Geometric transformations, in our case, are translations describing the relations in cylindrical panorama and homographies in planar panorama. In the image calibration phase, radial distortion and exposure differences are corrected in order to provide precise registration and less color discontinuity in the final results. The final image is about to be produced once we have registerd all of the input image and when the differences between images are minimized with respect to each other. This paper creates planar and cylindrical panoramas. Since the planar panoramas start to distort severly when representing large fields of view, cylindrical compositing surface is more preferable in this case. Focal length as a prerequisite of the cylindrical panorama can be estimated from the transformations produced in the image alignment phase. From the experiments, this method performs well in giving precise estimations of focal length. When blending the final result, Laplacian pyramid blending is applied by using the masks produced by graph cut algorithm. Laplacian pyramid blending is found to be an attractive solution to remove the seams and balance the exposure differences.

Future research should focus on improving the blending methods to generate high-quality panoramas, and applying image stitching method in the creation of 3D world.

Acknowledgment

I would like to express my respectful acknowledgement to Dr. Michael S. Lew, for his invaluable and unselfish guidance of this thesis.

And the warmest gratitude goes to my family and friends for their persistent love and encouragement.

References

[1] M. Brown and D.G. Lowe, "Automatic Panoramic Image Stitching Using Invariant Features," *International Journal of Computer Vision,* vol. 74 no. 1 pp. 59-73, 2007.

[2] M. Hansen, P. Anandan, K. Dana, G. vander Wal, and P. Burt, "Real-time Scene Stabilization and Mosaic Construction," *IEEE Workshop on Applications of Computer Vision (WACV'94)*, pp. 54-62, 1994.

[3] M. Irani and S. Peleg, "Mosaic Based Presentations of Video Sequences and Their Applications," *fifth International Conference on Computer Vision (ICCV'95)*, pp. 605-611, 1995.

[4] R. Szeliski, "Image Alignment and Stitching: A Tutorial," technical report MSR-TR-2004-92, Microsoft Research, 2004.

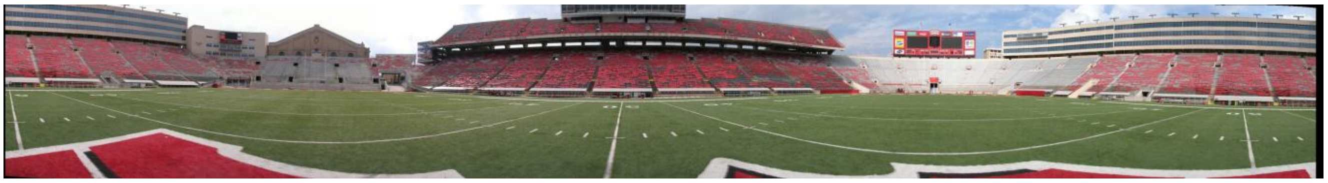
[5] R. Szeliski and H.Y Shum, "Creating Full View Panoramic Image Mosaics and Environment Maps," *Annual Conference on Computer Graphics (SIGGRAPH*), pp. 251-258, 1997.

[6] R. Szeliski, "Image Mosaicing for Tele-Reality Applications," *IEEE Workshop on Applications of Computer Vision (WACV)*, pp. 44-53, 1994.

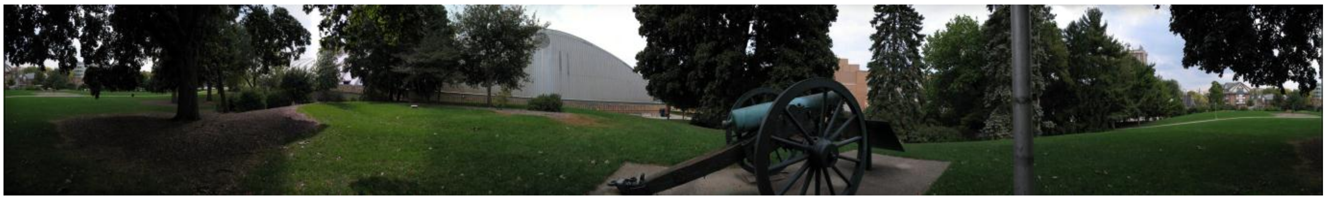
[7] H.S Sawhney and R. Kumar, "True Multi-Image Alignment and its Application to Mosaicing and Lens Distortion Correction," *Computer Vision and Pattern Recognition (CVPR),* pp. 450-456, 1997.

[8] M. Brown and D.G Lowe, "Recognising Panorama," *International Conference on Computer Vision (ICCV),* pp. 1218-1227, 2003.

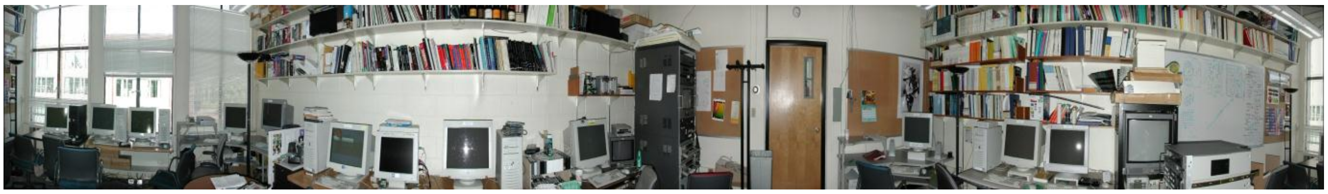
[9] A. Levin, A. Zomet, S. Peleg and Y. Weiss, "Seamless Image Stitching in the Gradient Domain," *European Conference on Com-*



(a)



(b)



(c)

Fig. 11 (a) and (b) are all composed by 20 images, while (c) is by 17 images. All of the images are blended using Laplacian pyramid blending using masks producing by graph cut algorithm, while using gain compensation to correct color differences.

puter Vision (ECCV), pp. 377-389, 2004.

[10] H.Y Shum and R. Szeliski, "Construction of Panoramic Image Mosaics with Global and Local Alignment," *International Journal of Computer Vision (IJCV)*, vol. 36, no. 2, pp. 101-130, 2000.

[11] D.G. Lowe, "Distinctive Image Features from Scale-Invariant Keypoints," *International Journal of Computer Vision (IJCV)*, vol. 60, no. 2, pp. 91-110, 2004.

[12] 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 (CACM)*, vol. 24, no. 6, pp. 381-395, 1981.

[13] Y. Boykov, O. Veksler and R. Zabih, "Fast Approximate Energy Minimization via Graph Cuts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222-1239, 2001.

[14] J.S. Beis and D.G. Lowe, "Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces," *Computer Vision and Pattern Recognition (CVPR)*, pp. 1000-1006, 1997.

[15] J.H. Friedman, J.L. Bentley and R.A. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time," *ACM Transactions on Mathematical Software (TOMS)*, vol. 3, no. 3, pp. 209-226, 1977.

[16] M. Zuliani, "RANSAC for Dummies," unpublished, http://vision.ece.ucsb.edu/~zuliani/.

[17] R. Szeliski, "Computer Vision: Algorithms and Applications," Springer; 1st Edition, 2010.

[18] S.B Kang, "Radial Distortion Snakes," *Machine Vision Applications (MVA)*, pp. 603-606, 2000.

[19] D.C. Brown, "Close-Range Camera Calibration," *Photogrammetric Engineering*, vol. 37, no. 8, pp. 855-866, 1971.

[20] G.Y. Tian, D. Gledhill, D. Taylor and D. Clarke, "Color Correction for Panoramic Imaging," *International Conference on Information Visualisation (IV)*, pp. 483-488, 2002.

[21] W. Xu and J. Mulligan, "Performance Evaluation of Color Correction Approaches for Automatic Multi-View Image and Video Stitching," *Computer Vision and Pattern Recognition (CVPR)*, pp. 263-270, 2010.

[22] Y. Xiong and K. Pulli, "Color Correction for Mobile Panorama Imaging," *First International Conference on Internet Multimedia Computing and Service (ICIMCS '09)*, 2009.

[23] G. Finlayson, M. Drew and B. Funt, "Color Constancy: Generalized Diagonal Transforms Suffice," Journal of the Optical Society of America A-optics Image Science and Vision (J OPT SOC AM A-OPT IMAGE SCI), vol. 11, no. 11, 1994.

[24] S.E Chen, "QuickTime VR – an Image-based Approach to Virtual Environment Navigation," *Computer Graphics (SIGGRAPH'95)*, pp. 29-38, 1995.

[25] R. Szelishi, "Video Mosaics for Virtual Environments," *IEEE Computer Graphics and Applications*, vol. 16, no. 2, pp. 22-30, 1996.

[26] C.Y. Chen, R. Klette, "Image Stitching - Comparisons and New Techniques," *Computer Analysis of Images and Patterns (CAIP'99)*, pp. 615-622, 1999.

[27] V. Kwatra, A. Schodl, I. Essa, G. Turk and A. Bobick, "Graphcut Texture: Image and Video Synthesis Using Graph Cuts," *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 277-286, 2003.

[28] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin and M. Cohen, "Interactive Digital Photomontage," *ACM Transactions on Graphics*, vol. 23, no. 3, pp. 292-300, 2004.

[29] Y. Xiong and K. Pulli, "Fast Image Labeling for Creating High-Resolution Panoramic Images on Mobile Devices," *International Symposium on Multimedia (ISM)*, pp. 369-376, 2009.

[30] Y. Boykov and V. Kolmogorov, "An Experimental Comparison of Min-Cut / Max-Flow Algorithms for Energy Minimization in Vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, vol. 26, no. 9, pp. 1124-1137, 2004.

[31] P.J. Burt and E.H. Adelson, "A Multiresolution Spline with Application to Image Mosaics," *ACM Transactions on Graphics (TOG)*, vol. 2, no. 4, pp. 217-236, 1983.