A 360-degree Panoramic Video System Design

Kai-Chen Huang, Po-Yu Chien, Cheng-An Chien, Hsiu-Cheng Chang and Jiun-In Guo
National Chiao Tumg University, Taiwan

ABSTRACT

In this paper, a low-complexity video stitching algorithm and its system prototype are proposed. With the novel design, users can obtain a high-resolution, high quality and seamless 360-degree panoramic video immediately by stitching the images with overlapped regions. Most of the present works are focused on image stitching instead of video stitching. In the proposed design, we develop some novel methods to solve the problems encountered in video stitching. First, we provide a new blending method to remove the color difference in video stitching. Moreover, we avoid the moving objects in the overlapped area by using the dynamic seam adjustment scheme. Finally, we remove the drift problem and obtain a better visual quality while displaying the 360 degree panoramic video scenes. The implementation results show that the entire system achieves 4-channel D1 30fps real-time video stitching on an Intel i7 3930K CPU 2.3GHz machine with 8GB DDR3 memory and Linux Ubuntu 12.10 operation system.

1 INTRODUCTION

The image/video stitching technology has become one of the most popular topics in consumer electronics products like digital still cameras. But, most of the approaches in relative literatures are usually focused on stitching a set of static images rather than videos.

The present methods can be divided into two major categories. One is using a special lens such like fish-eye lenses、panoramic lenses or a single camera with a rotatable shaft to enable camera to spin while capturing, which is known as "one-shot". Through these specialized devices can capture panoramas rapidly, its outputs are generally in low quality. Besides, cost of special lenses is expensive, so that users usually tend to use multiple images to create panoramic photos.

The other method is called "multi-shot", which is based on the technology of stitching a set of images captured by multiple same cameras. This type of method obtaining a panorama through several necessary image processing steps, including image alignment, image projection and image blending. The panoramas stitching by multiple images contain better quality than that generated by "one-shot", but it also suffers from high computational complexity.

Fig. 1 An omni-directional camera(Left) and its result(Right)

Compared with stitching images, there are more issues needed to be concerned in stitching videos. For example, there might be some moving objects passing through the seam dynamically. Also, there would be color difference between consecutive frames in the videos captured by the neighboring cameras, which will result in a degraded visualization, as shown in Fig. 2.

Fig. 2 An example of panoramic scene containing moving objects and color difference

To deal with high computation, moving objects and common lens setup, we come out several novel approaches to deal with the problems encountered in video stitching. First, we develop a new blending method to remove the color difference. Next, we avoid the moving objects in the overlapped area by dynamic seam adjustment scheme. In addition, we remove the drift problem to obtain a better visual quality while displaying a 360-degree panoramic video. In this paper, we focus on decreasing the complexity of image stitching and maintain better image quality meanwhile. The implementation results show that the whole system achieves 4-channel D1 real-time video stitching on an Intel i7 3930K CPU 2.3GHz machine with 8GB DDR3 memory and Linux Ubuntu 12.10 operation system.

The rest of this paper is organized as follows. In Section II, an overview on 360 degree cylinder projection method is introduced. In Section III, a low-complexity 360 degree video stitching algorithm is introduced. The performance evaluation and comparison of the proposed design is included in Section IV. Finally, a simple conclusion is given in Section V.

2 OVERVIEW ON 360 DEGREE CYLINDER PROJECTION

2.1 Image alignment, feature matching and perspective transform

Nowadays, various methods are used to determine corresponding points or regions in the overlapped images, such as block matching [1] and feature matching [2, 3]. The feature points contain information about its position and descriptor, so we can judge the degree of similarity between features in different images by comparing their descriptors. The designs [2, 3] used the feature-based mosaic algorithm to extract feature points and transform images into the same coordinate system by homography matrix. As shown in Fig. 3, a homography matrix is a 3x3 matrix includes eight elements to describe the relation of two image coordinate systems, as shown in Eq. 1. The relationship between the original and the corresponding coordinate can be presented by Eq. 2. In order to compute the eight parameters in matrix, we need several pairs of matched features. In [4, 11], Brown and Lowe used SIFT (Scale Invariant Feature Transform) algorithm to extract and match points between images. Then, they use RANSAC to select a set of inliers that are compatible with a homography between images.

978-1-4799-2776-0/14/$31.00 ©2014 IEEE

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = z' \begin{bmatrix} (\frac{x'}{z'}) \\ (\frac{y'}{z'}) \\ 1 \end{bmatrix} \qquad \text{Eq. 1}$$

$$x' = \frac{h_{11}x + h_{12}y + h_{13}}{h_{31}x + h_{32}y + 1}, \quad y' = \frac{h_{21}x + h_{22}y + h_{23}}{h_{31}x + h_{32}y + 1} \qquad \text{Eq. 2}$$

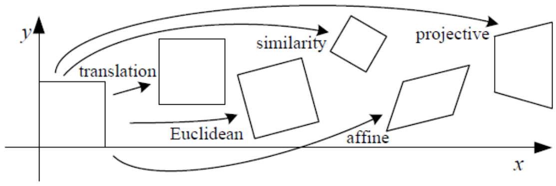

Fig. 3 Coordinate transformation models

Since homography-based method projects every image to the same reference plane, it would result in a distorted panorama which is not attractive and implausible. To solve the distortion problem, a method called "Cylindrical Projection" is proposed to generate 360-degree images or videos with acceptable distortion, which will be introduced in the following.

2.2 Cylindrical projection

To construct a plausible 360-degree view, Szeliski and Shum provided a method to stitch cylindrical panoramas. This type of panoramas maps pixels to a virtual cylindrical coordinate. There are many non-planar transformation models have been developed in the literature [5, 6], such as cylindrical projection, spherical projection, and cubic projection. As shown in Fig. 4, from this correspondence, we can compute the cylindrical coordinate from the planar coordinate. Considering a point P(X,Y,Z) in space, when this point gets projected to a unit radius cylinder, let's say the project point is specified by (0,theta) on the surface of the cylinder. By similar triangles we have: $(\sin\theta, h, \cos\theta) \propto (x, y, f)$

From this relationship, formula that maps a point to its projection on the cylindrical surface can be derived as

$$x' = s\tan^{-1}\frac{x}{f} = s\theta \quad , y' = s\frac{y}{\sqrt{x^2 + f^2}} = sh \qquad \text{Eq. 3}$$

Where s is an arbitrary scaling factor, which is also called the radius of the cylinder. The scaling factor s can be set as $s = f$ *(focal length of camera)* to minimize the distortion of the image near its center. If the focal length of the camera or field of view is known, each image can be warped into cylindrical coordinates by using Eq. 3.

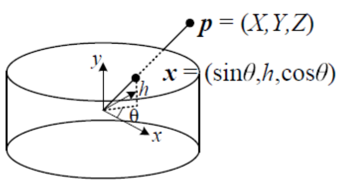

Fig. 4 Projection from 3D space to cylindrical coordinates.

3 PROPOSED ALGORITHM

In this section, we propose a low-complexity algorithm of 360-degree video stitching which can obtain high resolution and high quality panoramic videos by stitching a set of videos which contain overlapped regions between each other. In addition, users can determine the number of cameras according to their preference when using the proposed algorithm.

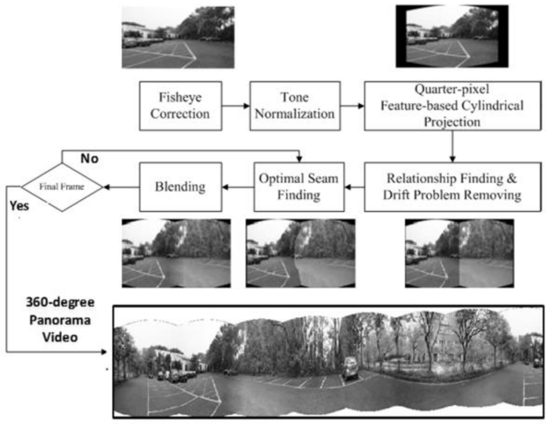

Fig. 5 Flow chart of the proposed video stitching algorithm

3.1 Pre-processing

In our approach, we first perform some pre-processing on the input source images, like fish-eye correction for image distortion caused by the wide view angle lens. Next, we implement cylindrical image alignment using SIFT to find the corresponding feature points between the two neighboring videos for preparation of video stitching. For the image alignment, we use a pure translational model to align the cylindrical images, where the cylindrical image alignment flow is described in Fig. 6. The coordinate of feature points in source images will map a corresponding position in the global table after transforming to cylindrical coordinate.

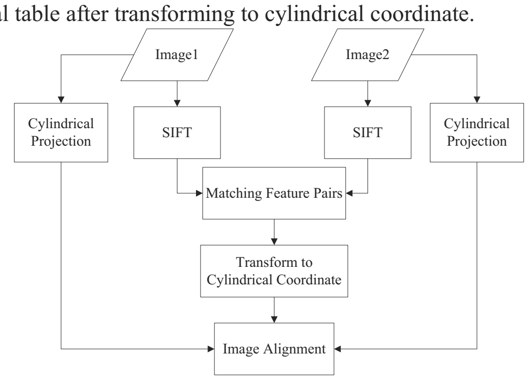

Fig. 6 The proposed cylindrical image alignment flow

3.2 Dynamic seam and blending

After all pre-processing steps, there are still some problems exist in the preliminary result. Therefore, in this subsection we develop two methods to resolve these problems. We first use a dynamic seam adjustment to avoid the moving objects crossing the overlapped region. And then, we implement a feature-based image tone normalization for blending to smooth the region near to the seam.

3.2.1 Dynamic seam adjustment

In video stitching, once an object moves from one side of the seam to the other through a fixed seam, it might cause some fragments on the moving objects. Therefore, the position of the seam should be changed dynamically according to the current circumstances. The proposed dynamic seam adjustment scheme is composed of three phases.

First, we build disparity tables about the overlapped areas. Each cost element *A(x,y)* is the sum of a block of luminance differences in two images. As shown in Eq. 4, Y_L and Y_R represents the luminance values of the different views, respectively. Calculating the sum of a block of pixels, instead of using pixels alone, can reduce the error situation that different objects have the similar color.

$$A(x,y)=\sum_i\sum_j |Y_L(x+i,y+j)-Y_R(x+i,y+j)|\ ,\ -1\le i,j\le 1 \qquad \text{Eq. 4}$$

Second, referring to the disparity table we find out the path which minimizes total cost in the current frame. This path indicates that elements besides this path have higher degree of similarity. Eq. 5 shows the method which utilizes dynamic programming to acquire the shortest path in each frame, where $D(x, y)$ represents the shortest path from top of the frame to the position (x, y), R is the width of the overlapped region, and W is width of input image. After finding out the minimum total cost, we could trace back the path to decide whether the path is the best seam in the current frame or not.

$$D(x,y)=\begin{cases} A(x,y)+\min\{D(x-1,y-1),D(x,y-1),D(x+1,y-1)\}, R<\frac{1}{3}W \\ A(x,y)+\min\{D(x-2,y-1),D(x-1,y-1),D(x,y-1),D(x+1,y-1),D(x+2,y-1)\}, R>\frac{1}{3}W \end{cases} \qquad \text{Eq. 5}$$

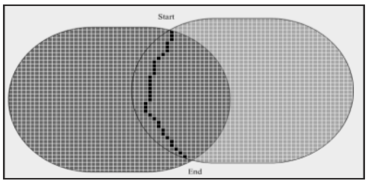

Fig. 7 Find the best seam in the overlapped area

In particular, finding the best seam in the overlapping area should follow one constraint, i.e. the path should begin from the top point of intersection between two cylindrical images, and the destination is at the bottom point, as shown in Fig. 7.

In the final step, we compare the total cost of the best seam of the current frame with that of the previous frame. If the former is larger than a predefined threshold, we replace the seam in the previous frame with the new one in the current frame. Otherwise, we maintain the previous seam. This mechanism is to prevent frame from beating caused by the rapid change of seams. The experimental results are shown in Fig. 8.

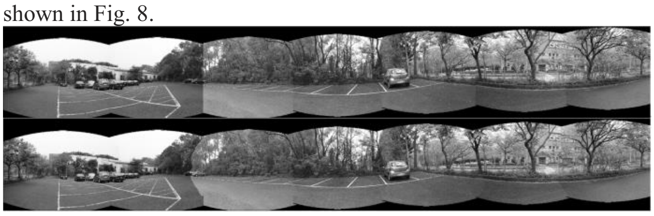

Fig. 8 The results without adjusting the seam (above) and with adjusting the seam (below)

3.2.2 Blending

According to different camera settings, or different exposure time caused by environmental factors, the color tone of the same objects in different images will be not alike. Overall, the serious chromatic aberration will lead to some obvious seams in the stitched frames. To solve this problem, we develop a feature-based image tone normalization for color blending to correct the difference of color before stitching, as shown in Fig. 9.

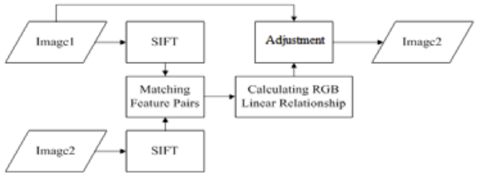

Fig. 9 Proposed feature-based image tone normalization flow

First, we use SIFT to extract features from the each source images respectively, and then find the corresponding feature points. By using the RGB data of the matched points between images, we derive a linear equation to find the relationship to make the RGB data of the feature points become similar. The linear equation is shown in Eq. 6.

$$\begin{pmatrix} r_2 \\ g_2 \\ b_2 \end{pmatrix}=\begin{bmatrix} \alpha_r & 0 & 0 \\ 0 & \alpha_g & 0 \\ 0 & 0 & \alpha_b \end{bmatrix}\begin{pmatrix} r_1 \\ g_1 \\ b_1 \end{pmatrix}+\begin{pmatrix} \beta_r \\ \beta_g \\ \beta_b \end{pmatrix} \qquad \text{Eq. 6}$$

The parameter alpha is a scale factor and beta is a constant to do fine-tuning. With this feature-based image tone normalization algorithm, we reduce the complexity by no calculating the histogram of the entire image, and get a set of images with the same color base to increase accuracy of image alignment. Fig. 10~Fig. 12 shows the simulation of the proposed tone normalization algorithm.

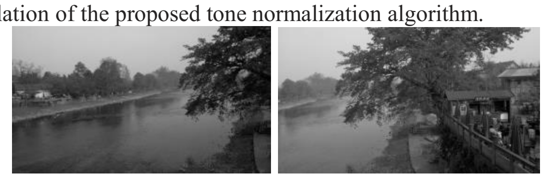

Fig. 10 The source images without tone normalization

Fig. 11 The source images after the tone normalization

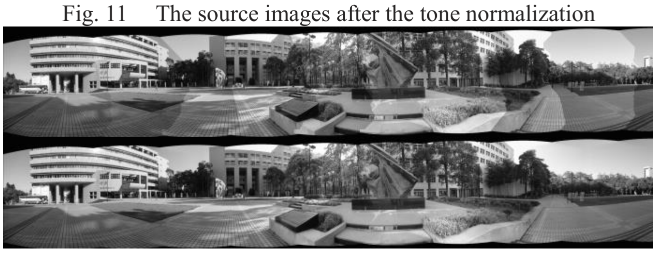

Fig. 12 (a)Without blending, (b)With blending

3.3 Removing drift problem

During the processing of cylindrical image stitching, the vertical shift produced by image alignment might be accumulated due to the impact of the terrain and the angle between cameras. The phenomenon that the front edge and the tail edge are misaligned to the other is called the drift problem.

To correct the drift problem, we have to know the sum of accumulated error. As mentioned above, we can use the matched features to find the amount of vertical shift. Therefore, we copy the leftmost image containing the front edge of the panorama and combine it after the tail edge, as shown in Fig. 13. By using Eq. 7, we will obtain the total vertical shift y_{total} between the same image with different heights and the widths of the panorama x_{total}. After obtaining x_{total} and y_{total}, we distribute the sum of error evenly to the entire panoramic image. Fig. 14 shows the results with and without the drift problems.

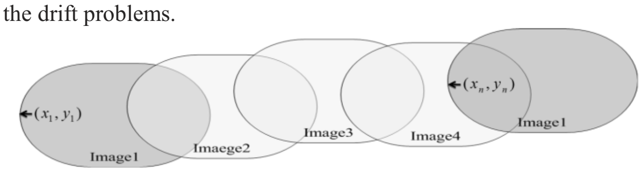

Fig. 13 An example of drifting problem

$$\begin{cases} x_{total} = x_n - x_1 \\ y_{total} = y_n - y_1 \end{cases} \quad \text{Eq. 7}$$

Fig. 14 The results with drift problem (left) and without drift problem (right)

Table 1 Comparison of proposed system with other products

	The proposed work	ImmerVison [7]	Point Grey [8]	MakingView [9]	CASEVision [10]
Product		IMV1-1/3	Ladybug 5	Viewcam 360	PVS9112
Viewing angle (in degree)	360	182	360	360	360
Projection Model	cylinder	PTZ	sphere	sphere	cylinder
Number of Cameras	4	1	5+1	5	12
Fisheye lens	○	○	○	○	×
Blending	○	--	○	○	×
Dynamic Seam	○	×	×	×	×
Performance	Real time	--	Offline	Offline	Real time
Compression standard	H.264	None.	Motion-JPEG	Unknow	None.

4 *PERFORMANCE EVALUATION AND COMPARISON*

Table 1 shows the comparison of the proposed and the existing designs found in the literature. As shown in Table 1, most of the present products can only implement the panoramic stitching on images because of its high computing complexity. Our work provides a real-time panoramic video with its quality as good as the output of Autostitch.

Fig. 15 Camera prototype of the proposed design

During developing a prototype of the proposed design, we adopt 4 cameras with 90 degree view angle difference with each other, where the camera prototype is included in Fig. 15. Under the limitation of USB interface bandwidth, we choose four analog cameras to build up the camera set. Moreover, for the better utilization of CPU, we use the strategy of multithreading to achieve parallel processing to speed up our video stitching algorithms. First, some complex procedure are separated and just completed once in the beginning. The tone normalization and blending methods, unlike other existing methods only can adjust one image at a time, can process the whole image by just focusing on the overlapped regions. Since the overlapped regions are independent, we directly set CPU affinity to assign each work in overlapped area to idle processors. Finally, in color model transformation, we use integer arithmetic instead of floating point arithmetic to optimize performance. After optimization and parallelism, the overall system is able to generate a 360-degree panoramic video based on 4-channel D1 30fps video resolutions through an Intel i7 3930K CPU 2.3GHz machine with 8GB DDR3 memory and Linux Ubuntu 12.10 operation system. Fig. 16 shows the prototype of the proposed system and the 360 degree panoramic video stitching results.

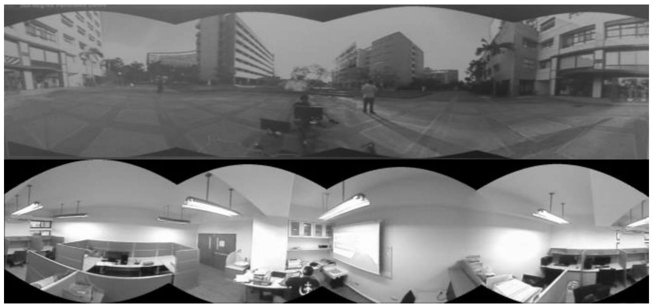

Fig. 16 Results of panoramic video stitching by the proposed system

5 CONCLUSIONS

In this paper, we develop a video stitching algorithm and a prototype system with low complexity for panoramic video applications. Users can obtain a high-resolution, high quality, and seamless 360-degree panoramic video immediately. The proposed dynamic seam adjustment scheme can prevent stitched seams from crossing the moving objects for better visual quality. In particular, our tone normalization and blending methods aim at the independent overlapped regions so that it is easy to be parallelized. Furthermore, we also reduce the computational complexity of the entire algorithm so that it can be realized in a multi-core processing platform with parallelization. By adding the dynamic information into the static panorama, our approach is capable of being applied to more applications such as intelligent vehicles, street view services, surveillance systems, landscape navigation, and so on.

REFERENCES

[1] Li Zhong-xin, Xu Wan-he, Zhang Yue, MaoYao-bin,Wang zhi-quan, “Video Mosaic with Block Matching and M-estimation,” *Proc. 2009 WRI World Congress on Computer Science and Information Engineering,* vol.6, pp.632-635, March 31 2009-April 2, 2009.

[2] Matthew Brown and David G. Lowe, “Automatic Panoramic Image Stitching using Invariant Features,” *International Journal of Computer Vision*, vol.74, no.1,pp.59-73, 2007.

[3] Chiou-Ting Hsu; Tzu-Hung Cheng; Beuker, R.A.; Jyh-Kuen Horng, “Feature-based video mosaic,” *Proc. 2000 International Conference on Image* , vol.2, pp.887-890, vol.2, Sept. 10-13, 2000.

[4] M. Brown and D. G. Lowe, “Recognising Panoramas,” *Proc. 9th ICCV, 2003*, pp.1218-1225.

[5] Richard Szeliski, Heung-Yeung Shum,“Creating Full View Panoramic Image Mosaics and Environment Maps,” *Microsoft Research.*

[6] Wu-Chih Hu, Chin-Hao Chang, Ya-Fang Liang, Chen-Tsung Wu,

[7] ImmerVision, http://www.immervision.com/en/home/index.php

[8] Point Grey Research Inc, http://ww2.ptgrey.com/

[9] MakingView, http://makingview.no/makingview.com/

[10] CASEVisiom, http://www.casevision.net/productinfor/p19_26.html