DFX Project 2: Stitching

B00901022祝成豪＆B00902004徐粲翔

1. Project Description

The topic of this project is Image Stitching. It’s a step by step process of turning a set of pictures taken in one scene into a larger images with bigger FOV in horizontal or vertical direction or both.

First, we take a lot of pictures of one scene, then we re-project them onto a cylindrical surface to create a slightly distorted image. Second, we run feature detection and matching algorithm on the images to extract feature points. Finally, we match the images with matching feature points, and blend or stitch them up according to the detected feature points matches.

1. Algorithms
2. )Feature Detection :  
   We use Harris Corner Detection as our feature detection algorithm. It is quite simple to implement it. However, it is a scale variant algorithm. Since we assume the photos of our scene will be taken in pure rotation. Harris Corner is a suitable algorithm.
3. )Feature Descriptor :  
   小邦
4. )Feature Matching :  
   小邦
5. )Image Matching :  
   We use RANSAC algorithm to exclude feature matching outliers from our matches. And then we use the inliers to retrieve a reliable translation model between adjacent images.
6. )Blending :  
   We studied the Poisson Image Editing paper and tried to implement poisson blending. Though the concept is easy to understand, the procedure is complicated and we failed to implement it. So we turned to 2 kinds of simpler methods to blend the images. One is using the distance to the center of images, the other one is using the distance to the boundary of stitching.
7. Code Implementation

We use MATLAB to write our program. Our original photos taken by camera have resolutions of 3456x5184. We first shrink them to 1/8 of their original sizes. After that, we re-project them onto a cylindrical surface, the focal length used in re-projection is generated by AutoStitch program.

小邦 -> 寫 feature部分

After we retrieve the feature points matches. We use our RANSAC function to randomly choose 2 points and generate a translation model. Then we apply that model to all other feature matches and calculate their distance. If the distance is smaller than the threshold we set. We look at that match as an inlier. The problem here is that there is no standard way to set the threshold. We need to adjust it iteratively and check the performance of the result. Besides we are not exactly sure that what portion of feature matches can we really trust, so we set the probability of inlier to a small number (ex. 0.2). After we retrieve the transformation matrix, we can blend the images.

To stitch or blend the images. We first check through all the transformation matrices. And store the shift of each image in two directions relative to the leftmost (first) image. Then we create an empty image with sizes set according to those image shifts. We can then put the copy the pixels of cylindrical images onto the empty image and blend them to generate our final panorama.

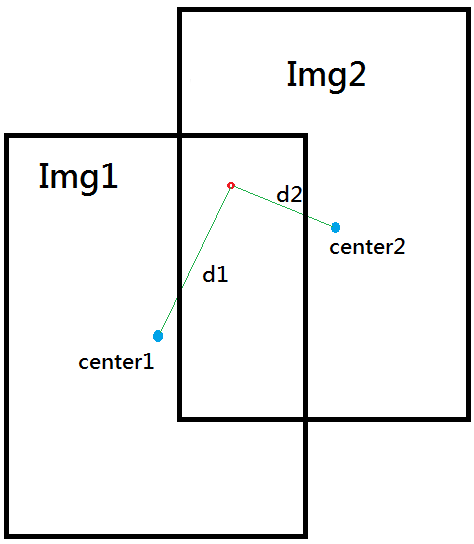
When we were testing our blending function, we found out that some bizarre black areas appeared near the border between two photos. The blended result is shown below.



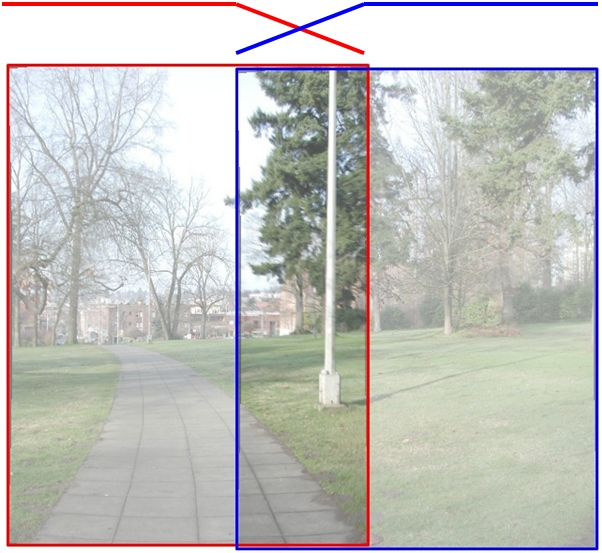
After further examination, we found the source of the problem. The reason is that when we were doing cylindrical reprojection. We saved the result as .jpg files. Since .jpg photos are the results of compressing processes. The cylindrical images we generated are flawed. Some pixels that were meant to be black (0,0,0) got contaminated by nearby pixels as shown below.



The two methods used in our stitching function are a little bit different from each other. First, we tried to use the distance from the target pixel to the centers of two Images as their weights.



Our second method is the method which teacher talked about in class. Simply use the horizontal distance to the stitching boundary as their weights.



We will find out that second method is better at eliminating the boundary between photos since the weight of the right image reaches 0 at the left boundary.

1. Results

We took many set of photos. The MRT station set is taken without a tripod. Others are taken with a tripod. For the Outside Library scene, we took multiple sets, each has different amount of pictures. The results are shown below.

**First blending method (distance to center) :**

|  |
| --- |
| 1. **Testing photos** |
| Number of Pictures : 18 |
| RANSAC threshold : 20 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\test\panorama.bmp |
| 1. **MRT station** |
| Number of Pictures : 17 |
| RANSAC threshold : 15.125 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\mrt\panorama_15.125.bmp |
| RANSAC threshold : 2.5 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\mrt\panorama2.5.bmp |
| RANSAC threshold : 1.5 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\mrt\panorama_1.5.bmp |
| 1. **Outside Library** |
| Number of Pictures : 16 |
| RANSAC threshold : 20 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\lib_out\panorama_RANSAC20.bmp |
| RANSAC threshold : 15.125 |
| D:\DropBox\DigitalVisualEffects\HW2_DVFX_STITCHING\result_photos\panorama\lib_out\panorama_RANSAC15.125.bmp |
| Number of Pictures : 20 |
| RANSAC threshold : 2.21 |
|  |
| RANSAC threshold : 5 |
|  |
| RANSAC threshold : 10 |
|  |
| Number of Pictures : 25 |
| RANSAC threshold : 3 |
|  |
| RANSAC threshold : 5 |
|  |
| Number of Pictures : 40 |
| RANSAC threshold : 2.125 |
|  |
| RANSAC threshold : 2.21 |
|  |
| RANSAC threshold : 2.25 |
|  |
| RANSAC threshold : 3 |
|  |
| RANSAC threshold : 5 |
| C:\Users\Nick\AppData\Local\Microsoft\Windows\INetCache\Content.Word\panorama_3.bmp |

When using the first blending method, we found out that the boundaries are very obvious. This can actually help us tune the RANSAC parameters since the distortion can be easily observed. Then we turned to the second method.

**Second blending method (distance to boundary) :**

|  |
| --- |
| 1. **Testing photos** |
| Number of Pictures : 18 |
| RANSAC threshold : 20 |
|  |
| 1. **MRT station** |
| Number of Pictures : 17 |
| RANSAC threshold : 1.5 |
|  |
| RANSAC threshold : 2.5 |
|  |
| 1. **Outside Library** |
| Number of Pictures : 17 |
| RANSAC threshold : 14 |
|  |
| RANSAC threshold : 15 |
|  |
| Number of Pictures : 20 |
| RANSAC threshold : 5 |
|  |
| RANSAC threshold : 7 |
|  |
| RANSAC threshold : 8 |
|  |
| RANSAC threshold : 9.5 |
|  |
| RANSAC threshold : 10 |
|  |
| Number of Pictures : 25 |
| RANSAC threshold : 3 |
|  |
| RANSAC threshold : 5 |
|  |
| RANSAC threshold : 7 |
|  |
| RANSAC threshold : 9 |
|  |
| RANSAC threshold : 11 |
|  |
| Number of Pictures : 40 |
| RANSAC threshold : 13 |
|  |
| RANSAC threshold :15 |
|  |
| RANSAC threshold : |
|  |
| RANSAC threshold : |
|  |
| RANSAC threshold : |
|  |

It is obvious that the test photos generate best the result of panorama overall. We took many sets of photos because we wanted to know that whether the number of pictures and the use of tripod will affect the performance of our final panorama.

1. Summary