CMSC 733 Project 1

Wei-An Lin
The Department of Electrical and Computer Engineering
University of Maryland
Email: walin@terpmail.umd.edu

I. Adaptive Non-maximal Suppression

We first apply Matlab build-in function 'cornermetric' to detect corners in an image. Local maximum of the corner responses can be found using 'imregionalmax'. ANMS algorithm is then applied to those local maximum. Fig. 1 shows the results with $N_{\text{best}} = 400$.

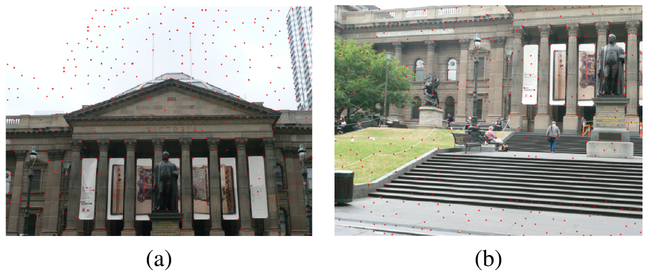

(a) (b)

Fig. 1: Sample ANMS results.

II. Feature Matching

To represent the detected corners in a compact way, we use the following procedures: (1) Smooth the entire image using 5×5 Gaussian kernel with standard deviation 1. (2) Padding the image with a margin 20. (3) For each corner response, we crop a 41×41 region, and reshape it to a one-dimensional vector. (4) Normalize the vector to zero mean and unit standard deviation. If we directly match the features obtained in this way, the result would be the one shown in Fig. 2. We can observe some mismatch from the figure. To overcome this effect, we apply the RANSAC algorithm on those initial matches. We iteratively choose four pairs from those matches, and compute corresponding homography. The sum of squared distances (SSDs) are evaluated at the target image, for each feature point in the target image, if SSD is less than 1, it is chosen as an inlier. Finally, we accept the best homography with the most inliers. The final homography will be computed using those inliers. The result after refining those pairs is shown in Fig. 3

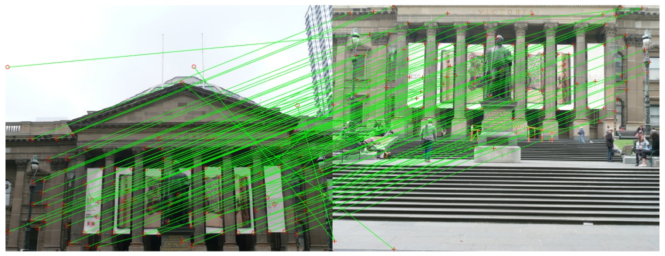


Fig. 2: Direct feature matching.

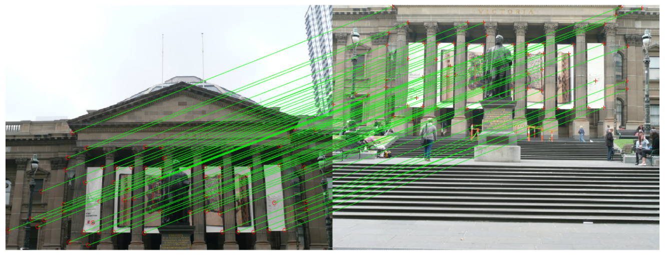


Fig. 3: Feature matching after applying the RANSAC algorithm.

III. Cylindrical Projection

We apply the inverse warping from the cylindrical coordinate to the Cartesion coordinate system. We choose focal length $f = 400$. If f is too small relative to the width of the original image, the warped image will contain vertically flipped duplications. If f is too large, the warped image will be nearly the same as the unwarped image. We choose f experimentally in this version. For interpolation, we simply round the coordinate if it is not integer. The projected images can be seen from Fig. 4.

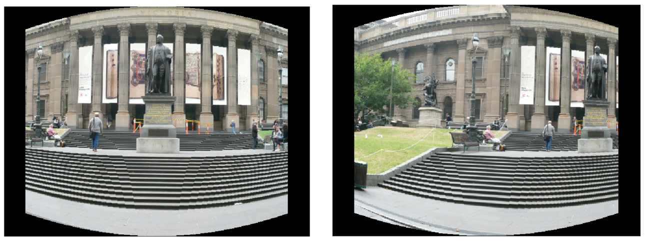

(a) (b)

Fig. 4: Sample cylindrical projection results.

IV. Stitching

To stitch a small number of images (e.g. less than five), we choose a base image and project the rest of the images to the same orientation as the base image. To fit all the images in one figure, we use Matlab build-in 'imtransform' to estimate the range for each warped image, and tune the final canvas to be able to contain all the regions. We take 'max' operation to handle overlapped and nonoverlapped regions. Fig. 5 shows the results of stitching images in Cartesian coordinate and cylindrical coordinate.

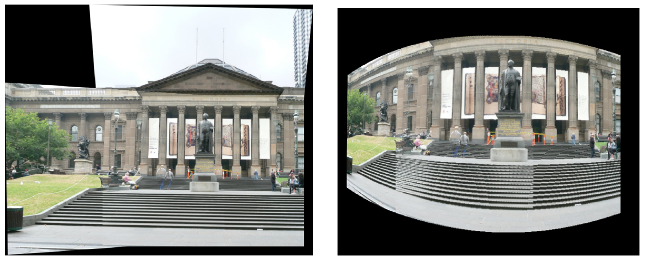

(a) (b)

Fig. 5: Stitch results. The left side shows the stitching for 1.jpg, 2.jpg, and 3.jpg. The right side shows stitching in cylindrical coordinate for 2.jpg and 3.jpg.

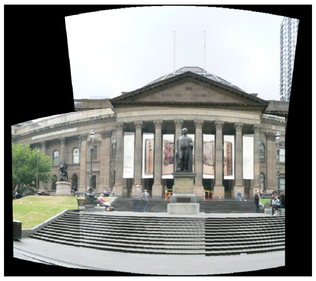
Stitching multiple images is more challenging. We address some issues in the following. First, we noticed that the quality of the feature extraction plays important role in stitching. However, in the preprocessing step, we warp each image into cylindrical coordinate, which distorts regions near image boundaries. The resulting feature would be very different from the one extracted in the Cartesian coordinate. To solve this, we extract features in the original Cartesian coordinate, and map only their positions into cylindrical counterparts.

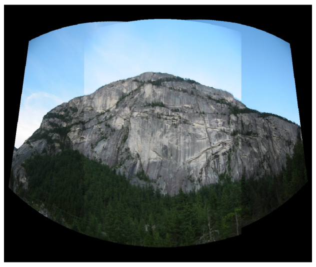
Second, we adaptively choose focal length based on the formula $f = 3W/\pi$, where W is the width of the input image. Third, to determine connected components in a set of images, we compute homography for each pair of images. Two images are viewed 'connected' if the maximum inliers computed from the RANSAC algorithm is greater than five and the conditional number of the homographic transform matrix H is less than 10^8. Then we apply the DFS algorithm to find all the connected components. In addition, it is desirable to choose the image which has large connectivity with others as the base image. We modify the DFS algorithm such that images are searched in the order of decreasing number of total inliers connecting to them.

Fig. 6 shows some basic results. Fig. 7 shows panoramas from the test dataset. Fig. 8 shows panoramas from the custom dataset. From Fig. 7a, only 1.jpg and 2.jpg are stitched while the rest of the images are viewed as single images. We plot the matched features between 2.jpg and 3.jpg in Fig. 9. Features extracted from corners in a checkerboard are nearly indistinguishable from one another. In our matching strategy, indistinguishable pairs will be rejected. In addition, the matching scenario shown in Fig. 7a will yield singular homographic transformation, and crash the panoramic stitching. We already handle this case by rejecting the homographies H that are near singular. In Fig. 7d, the algorithm successfully recognize the panorama and reject those single images. Fig. 8b shows an example that although features match pretty well, the resulting panorama has severe artifacts. This might be because the images are taken in short distance, and the camera movements were a combination of rotation and translation, the relative scene position changed significantly.

V. Bonus Credit

As introduced above, the implemented algorithm can automatically discovers panoramas and stitches them.


(a)


(b)

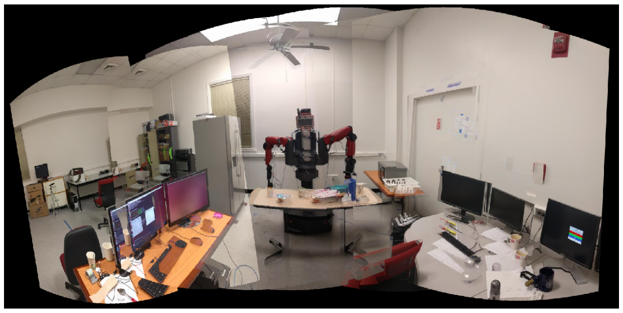
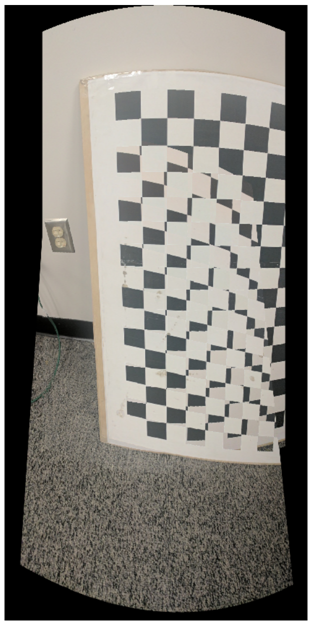
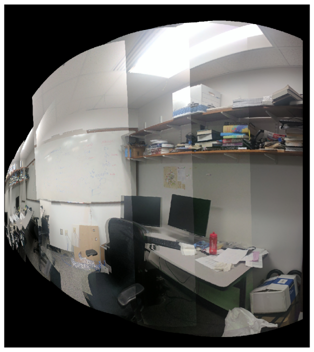
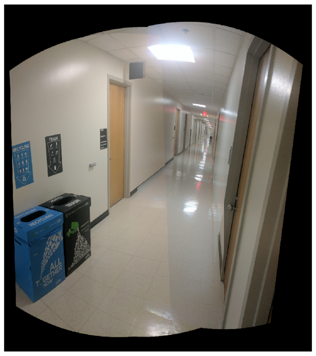

(c)

Fig. 6: Basic stitching results.


(a)


(b)


(c)

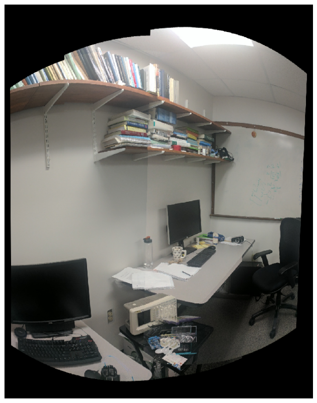
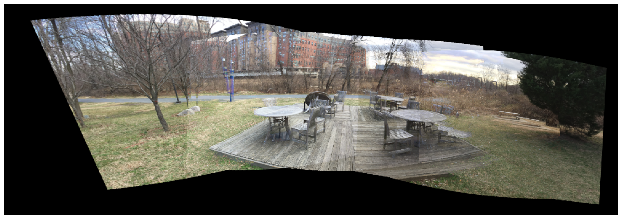
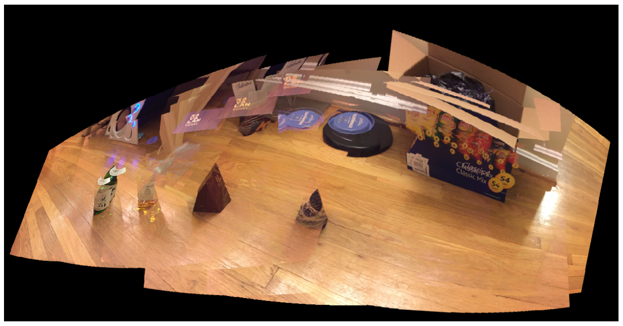

(d)

Fig. 7: Panoramas from the test dataset.



(a)



(b)

Fig. 8: Panoramas from the custom dataset.

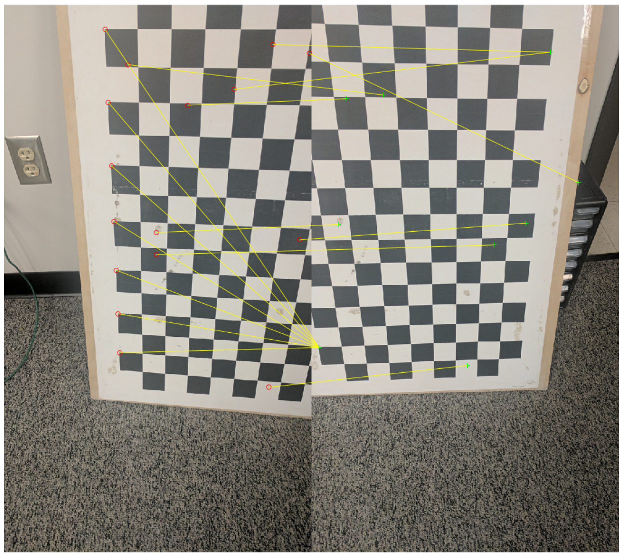


Fig. 9: Feature matching between 2.jpg and 3.jpg in testSet1.