Dept. of CST, Tsinghua University
Digital Image Processing

Image Stitching
Project Report

Name	Yuxin Wu
Student No.	2011011271
Class	CST-14
Mail	ppwwyyxxc@gmail.com

Contents

1 Usage

The program is written in C++11, and is also available at https://github.com/ppwwyyxx/panorama.

1.1 Compilation

Dependencies:

1. gcc >= 4.7
2. GNU make
3. Magick++
4. boost MTL (Matrix Template Library) installed in /usr/include/boost/numeric/. It can be downloaded from http://www.simunova.com/node/145.

Compilation:

```
$ make
```

1.2 Run

Various parameters are saved in `src/config.cfg`. Without special needs, we only have to modify `PANO, TRANS, CROP`.

The program does some extra work to beautify the output if knowing the input pictures were taken by a camera moving in pure translation or pure rotation.

`PANO = 1` tells that the camera moved in pure rotation. A panorama is expected to be the output;

`TRANS = 1` tells that the camera moved in pure translation. Result will be better than the one with `TRANS` unset;

`CROP` decides whether to crop the final image to a rectangular;

Use `./main <file1> <file2> <file3> ...` in the command line to run the program. Output file is `out.png`.

Usually, input images should not exeeds $20(files) \times 1500(pixels) \times 1000(pixels)$, since your computer may not have enough memory.

The input file names given in the command line need to be well ordered, to make sure the adjacent images can be stitched together.

1.3 Examples

1. TRANS = 1

```
$ ./main ../data/flower/small*
```

2. PANO = 1

```
$ ./main ../data/ground/small*
```

The result is shown in Fig14.

2 Algorithms

2.1 Feature Detection

Lowe's SIFT algorithm[2] is implemented.

The algorithm follows the following procedure:

1. Build a Scale-Space, which consists of $S \times O$ grey images. The original image is resized in O different sizes (octaves), and each is then Gaussian-Blured by S different σs.

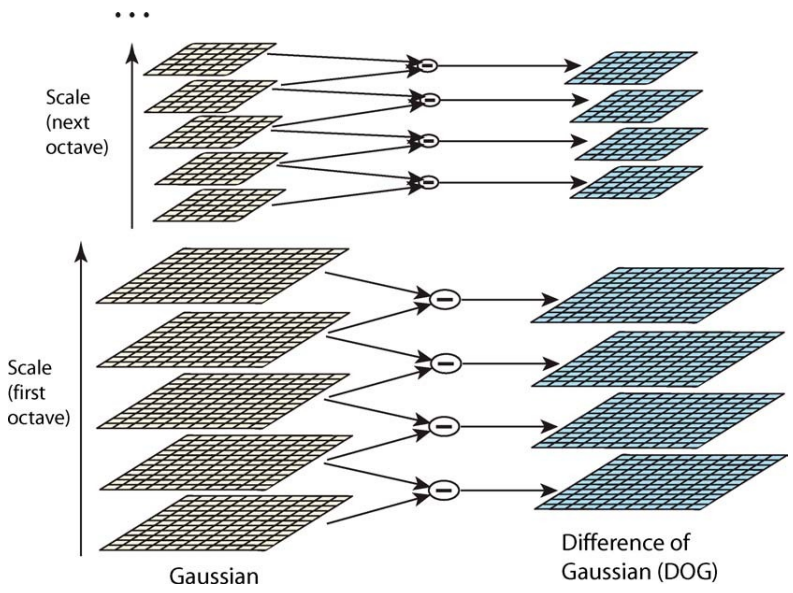

Figure 1: Scale Space and DOG Space

2. Build a Difference-of-Gaussian Space. In each octave, calculate the differences of every two adjacent blured images. Therefore, DOG Space consists of $(S-1) \times O$ grey images. As shown in Fig1.

3. Detect extrema. In DOG Space, detect all the minimum and maximum by comparing a pixel with its 26 neighbors in three directions: x, y, σ. See Fig2 and Fig3

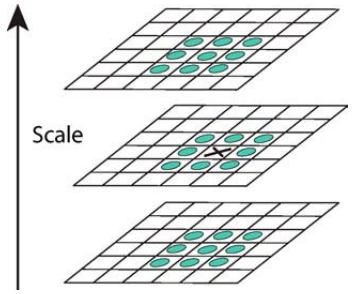

Figure 2: Extrema Detection

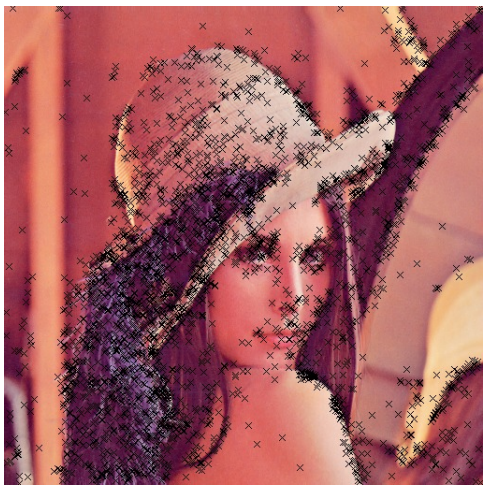

Figure 3: Extrema Example

4. Keypoint Localization. Use linear interpolation to calculate offset, moving the extrema to a more accurate location. Then reject the points with low contrast or on the edge to get distinctive features. See Fig4, Fig5, Fig6.

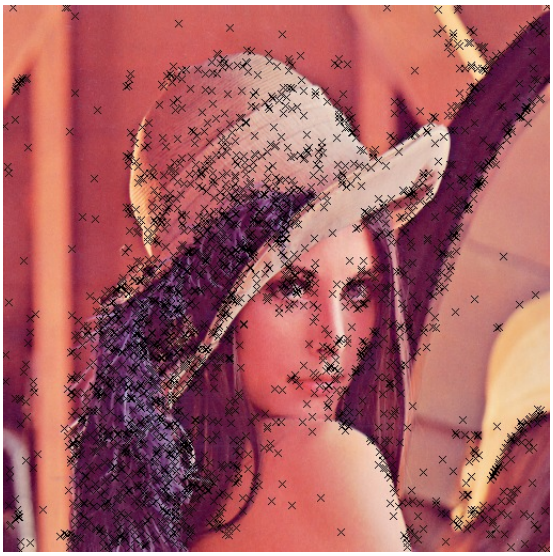

Figure 4: After Localization

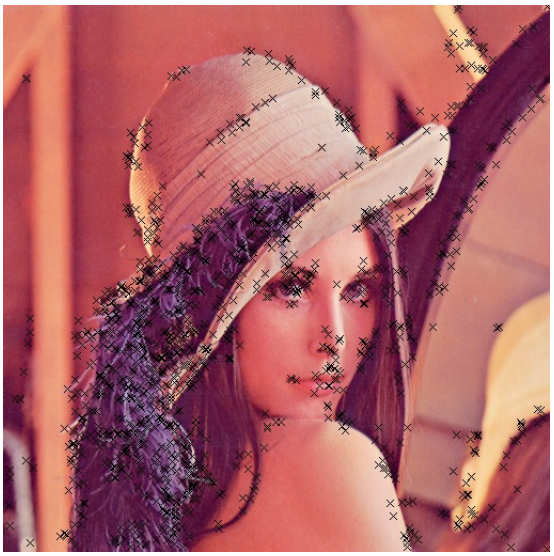

Figure 5: After Rejecting Low Contrast

Figure 6: After Eliminating Edge Point

Figure 7: After Assigning Orientation

5. Orientation Assignment. We have to first calculated the gradient and orientation of every point in the Scale Space. For each keypoint, a histogram of the orientations of its nearby points is built, weighted by a gaussian kernel and the magnitude of gradient. The peak in the histogram is chosen to be the orientation of the keypoint, as shown by the arrow in Fig7.

6. Descriptor Representation. Lowe suggested choosing 16 points around the keypoint and building 16 orientation histograms, each with 8 different possible values. Since all the orientation is relative to the keypoint's orientation, this feature is roatation-invariant. The final SIFT feature is a 128-dimensional array.

7. Feature Matching. The Euclidean distance of the 128-dimensional descriptor is the criteria for feature matching between two images. A match is considered not convincing and therefore rejected if the distances to its closest neighbor and second-closest neighbor are similar. The result is shown in Fig8.

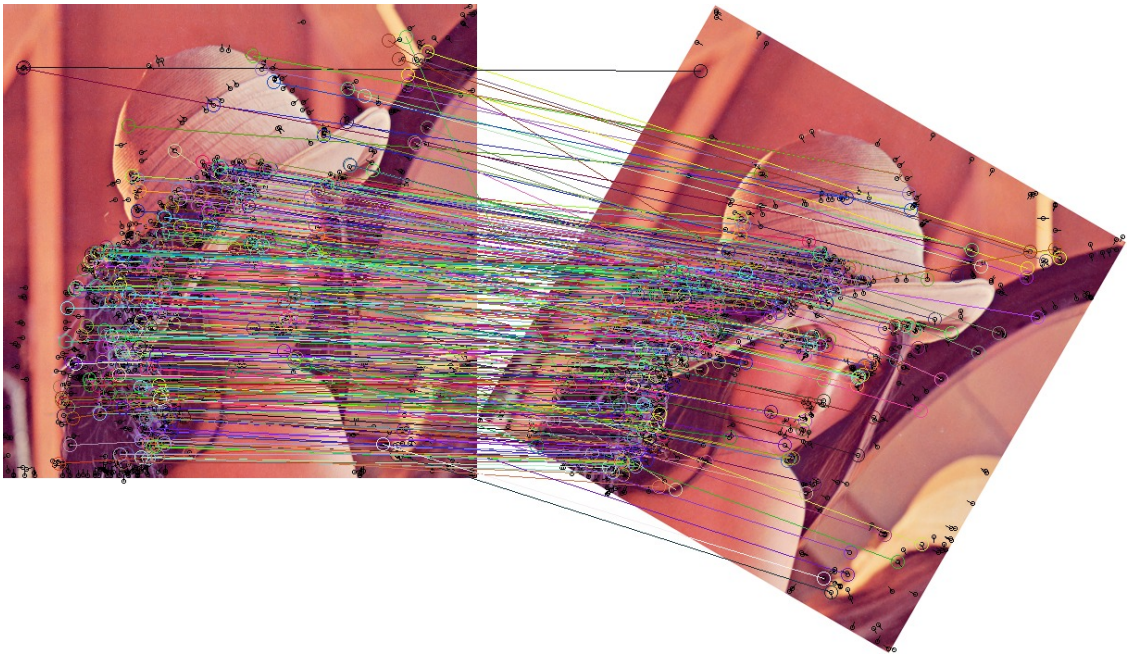

Figure 8: Matching Result

2.2 Warping

A homography transform is needed to match the keypoints. But a pure homography transform on a planar works only if the camera moved in pure translation or toward a fixed center.

If a rotational input is given, using homography on a planar leads to vertical distortion, like Fig9.

Figure 9: Vertical Distortion with Homography on a Planar

Therefore, if the option `TRANS = 0`, every input image will be projected to a cylinder by the following formula:

$$\begin{cases} x' = k \arccos \dfrac{x}{\sqrt{x^2 + z^2}} \\ y' = k \dfrac{y}{\sqrt{x^2 + z^2}} \end{cases}$$

where z is chosen to be $\max\{width, height\}$. The result is in shown in Fig10.

Figure 10: Stitching Result After Projection

2.3 Transformation

RANSAC (Random Sample Consensus) algorithm[1] is used to calculate a transformation matrix. In every iteration, several matched pairs is randomly chosen to calculate a best-fit transformation matrix, and the number of pairs it fits within all the data is calculated.

I tried three kinds of transformation:

1. Homography One: $H = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & 1 \end{bmatrix}$
2. Homography Two: $H = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}$ and $\|H\| = 1$
3. Affine: $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}$

Given a set of matches, affine and the first kind of homography transformation can be solved by least-squares fitting an over-determined linear system. The second kind of homography transformation can be solved by singular value decomposition. The two homography lead to similar results, which are used when `TRANS = 1`. For warped images, all the transformation work well. In this project, affine transformation is used for generating panoramas.

2.4 Straightening

Figure 11: Bended Panorama

Since the tilt of camera is unknown, the output panorama might be bended as shown in Fig11. Instead of using the first image as the pivot, and calculating all the other transformation relative to it, using the image in the middle as the pivot can help reduce the tilt effect. Moreover, when the option `PANO = 1` is set, an automatic straightening process will be executed.

First, a binary search on the tilt factor is performed, aiming to make the result "longer". Second, the program will try to catenate the first image after the last one, and then do a global rotation to ensure that the head and tail are aligned horizontally. After this two processes, the result is like Fig12.

Figure 12: Straightened Panorama

2.5 Blending

The size of the final result is determined immediately after calculating all the transformations. And the pixel value in the result image is calculated by an inverse transformation and interpolation with nearby pixels, in order to reduce alias effect.

As for the overlap effect, the distance of the pixel to each image center is used to calculate a weighted sum of the pixel value. The result is almost seamless, as shown in Fig13.

Figure 13: Blended Result

2.6 Cropping

When the option `CROP = 1` is set, the program simply manage to find a rectangular with largest area within the original irregular result.

A classic $O(n \times m)$ algorithm is used, where n, m is the height and width of the original result.

For each row i, update

$$h[j] = \begin{cases} 0, & if\ i = 0\ or\ A[i][j]\ is\ outside \\ h[j] + 1, & otherwise \end{cases}$$
$$r[j] = \max\{k \in [0, m) \cap \mathbf{N} : h[t] \geq h[j], \forall j \leq t \leq k\}$$
$$l[j] = \min\{k \in [0, m) \cap \mathbf{N} : h[t] \geq h[j], \forall k \leq t \leq j\}$$

for every $j \in [0, n)$ in amortized $O(1)$ time, and the corresponding area is $(r[j] - l[j] + 1) \times h[j]$.

The cropped final result is shown in Fig14

Figure 14: Cropped Result

3 More Results

4 References

[1] Martin A Fischler and Robert C Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[2] David G Lowe. "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2 (2004), pp. 91–110.