Report for Problem Set 2

1 Introduction to files

`func.py`: Basic functions used in later implementations.

`harris.py`: Implementation of Q1 (Harris corner detector).

`histogram.py`: Implementation of Q2 (histogram of gradients).

`matching.py`: Implementation of Q3 (local feature matching).

`stitching.py`: Implementation of Q4, Step 1 – Step 3 (Blending two images).

`panorama.py`: Implementation of Q4, Step 4 (generating a panorama).

`better_blending.py`: Implement of Gaussian filtering and Poisson blending, for BQ1.

`SIFT.py`: Implementation of SIFT detector and descriptor, for BQ2.

2 Q5. Analyze

2.1 Different camera settings

(a) (b) (c) (d)

Figure 1: **Various settings for shooting.** (a) original image; (b) translate the camera only; (c) rotate the camera only; (d) simultaneously rotate and translate the camera.

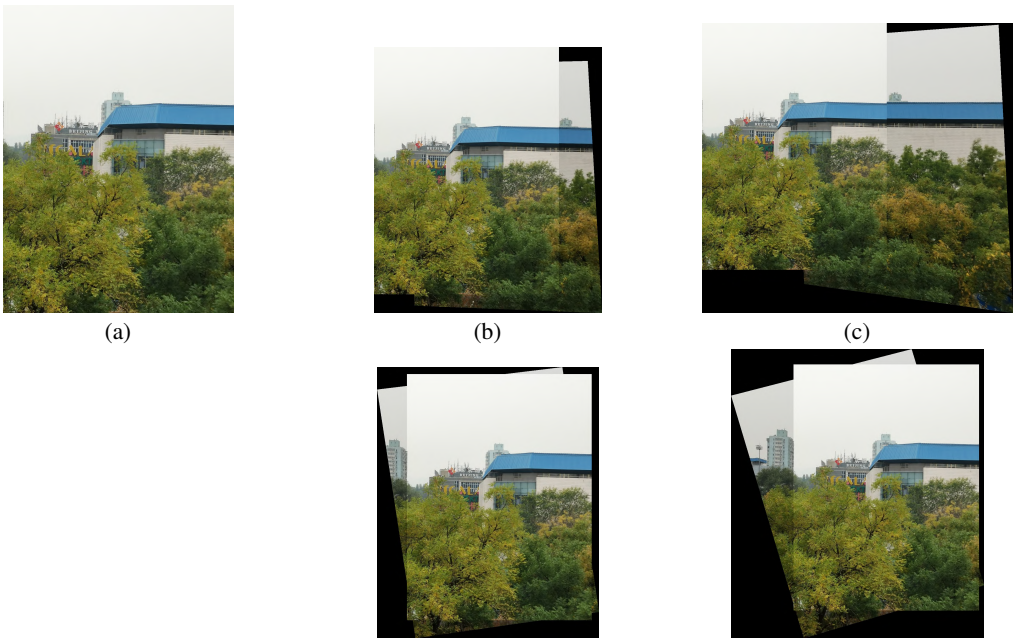
2.2 Small and large translation/rotation

As shown in Fig. 2, larger translation/rotation will cause worse results when stitching images. The main reason is that, when translation/rotation gets larger, the matching features between the two images turn less, and thus results in a higher error when computing homography matrix.

2.3 Moving objects and "ghosted" visions

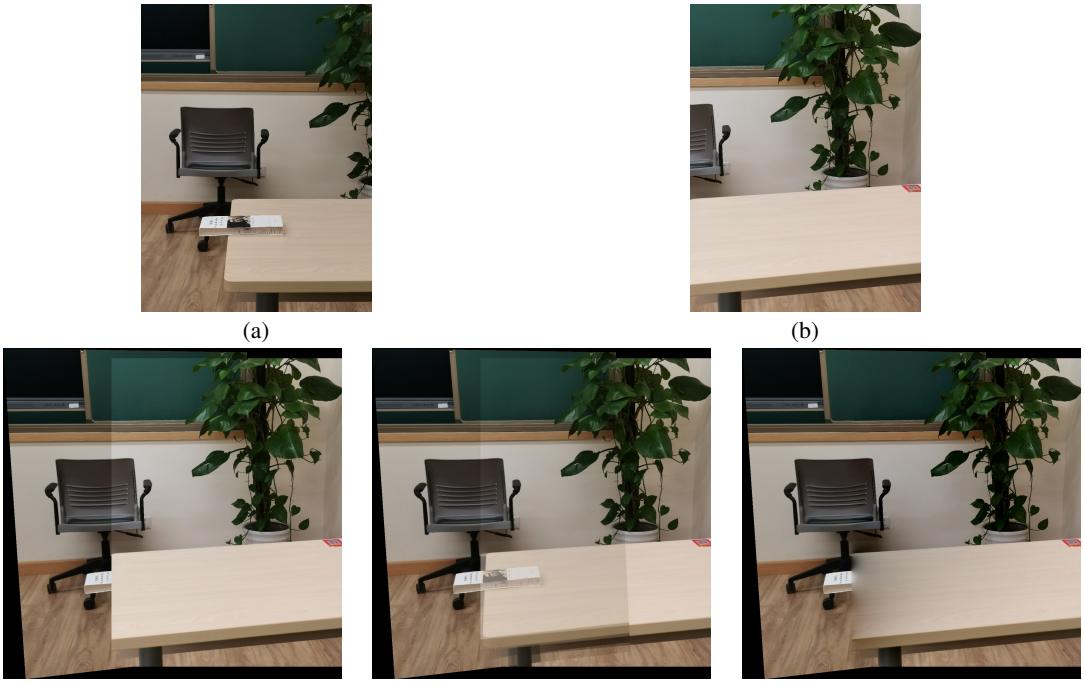
As shown in Fig. 3, when an object appears in one image (in this case, a book put at the edge of the desk) but moves away in another, just do simple blending (no matter what type of blending) can cause "ghosted" visions.

The basic idea to de-ghosting is to first determine which region of the input images are not static and then which images to cover others in a certain area. One method can be seen in [1].



(a) (b) (c) (d) (e)

Figure 2: **Various settings for shooting.** (a) original image; (b) small translation; (c) large translation; (d) small rotation; (e) large rotation.



(a) (b) (c) (d) (e)

Figure 3: **"Ghosted" vision.** (a) original image 1; (b) original image 2; (c) overlap blending; (d) average blending; (e) Poisson blending.

2.4 The same person appearing multiple times

As shown in Fig. 4a and Fig. 4b, the same person appears in two images and is standing at different places.

Experiments show that, if hyperparameters are approximately tuned, the program can still produce reasonably good result (Fig. 4c, Fig. 4d). But it is almost inevitable to observe "ghosted" visions.

(a) (b) (c) (d)

Figure 4: (a) original image 1; (b) original image 2; (c) average blending; (d) Poisson blending.

3 BQ1. Filtering and advanced blending techniques

3.1 Gaussian filtering

Obvious difference can be seen on the boarder when we simply put the images together.

Figure 5: Great difference of the road color can be easily observed.

When convolved with Gaussian kernel, though the stitching image get blurred, it appears more continuous.

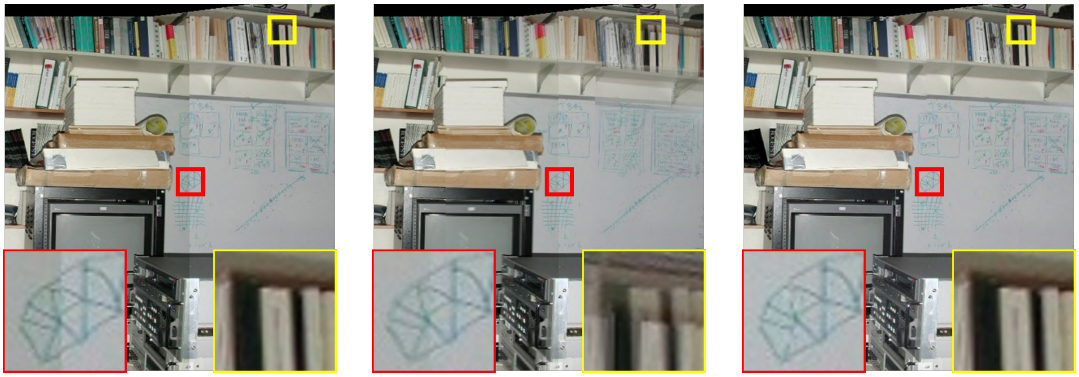
3.2 Poisson blending

Though Gaussian kernel has the ability to smooth the result, at the same time, it blurs the output and abandons some details.

By taking average value if multiple images takes up same pixels, we can sometimes get better result (as shown in the red rectangle of Fig. 7b). But if the error of homography matrix turns a bit larger, and the corresponding pixels do not strictly coincide, "ghosted" visions may appear (as shown in the yellow rectangle of Fig. 7b).

The result when using Poisson blending is shown as Fig. 7c. It does not cause messy patterns or blurs, while at the same time smoothing the result as we want.

Figure 6: After Gaussian filtering, the image seems smoother.



(a) (b) (c)

Figure 7: **Different blending tricks.** (a) overlap; (b) average; (c) Poisson.

4 BQ2. SIFT: a more powerful and robust feature detector

Naive HoG with a fixed window size can well handle the case when the size of an object are nearly the same, and rotation cannot be to large.

So in many cases, HoG is not enough and we need a more powerful and robust detector, which can find and match features even if their sizes are changed. SIFT is one of those descriptors. Fig. 8 shows one example where SIFT beats HoG.

References

[1] Matthew Uyttendaele, Ashley Eden, and Richard Skeliski. Eliminating ghosting and exposure artifacts in image mosaics. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 2, pages II–II. IEEE, 2001. 1

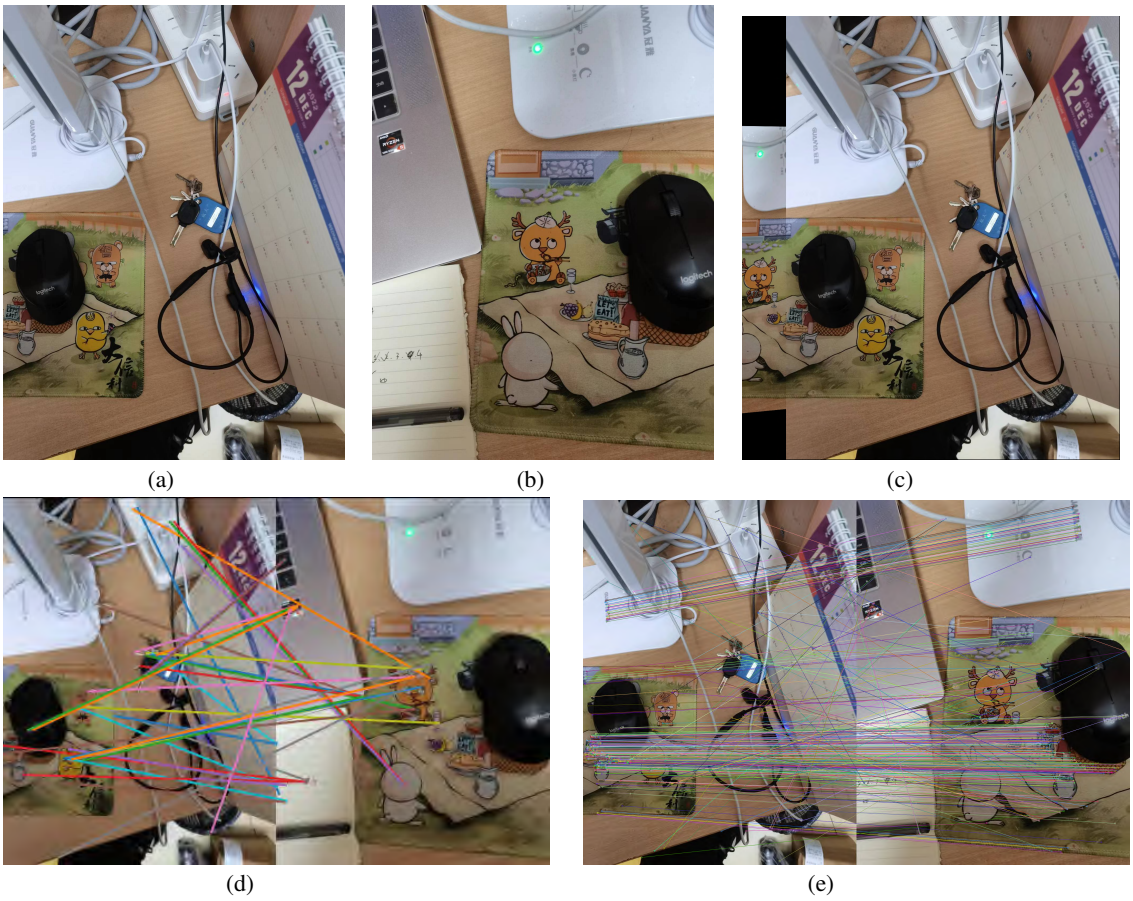


Figure 8: **HoG and SIFT.** (a) original image 1; (b) original image 2; (c) stitched image (using SIFT); (d) HoG feature map; (e) SIFT feature map. HoG is not able to blend the images as it cannot find enough robust feature matches.

A Results

In the program, there are 3 main hyperparameters that can be tuned for better performances, *i.e.*, `threshold` in `harris.corner_selection`, `threshold` in `matching.feature_matching`, and `threshold_2` in `matching.feature_matching`.

Below are some results, captioned the 3 values of hyperparameters.

Figure 9: (0.05, 0.8, 0.55). Blend 2 images.

Figure 10: (0.05, 0.8, 0.55). Blend 2 images.

Figure 11: (0.05, 0.8, 0.55). Blend 2 images.

Figure 12: (0.05, 0.8, 0.55). Blend 5 images.

Figure 13: (0.005, 0.8, 0.55). Blend 7 images.

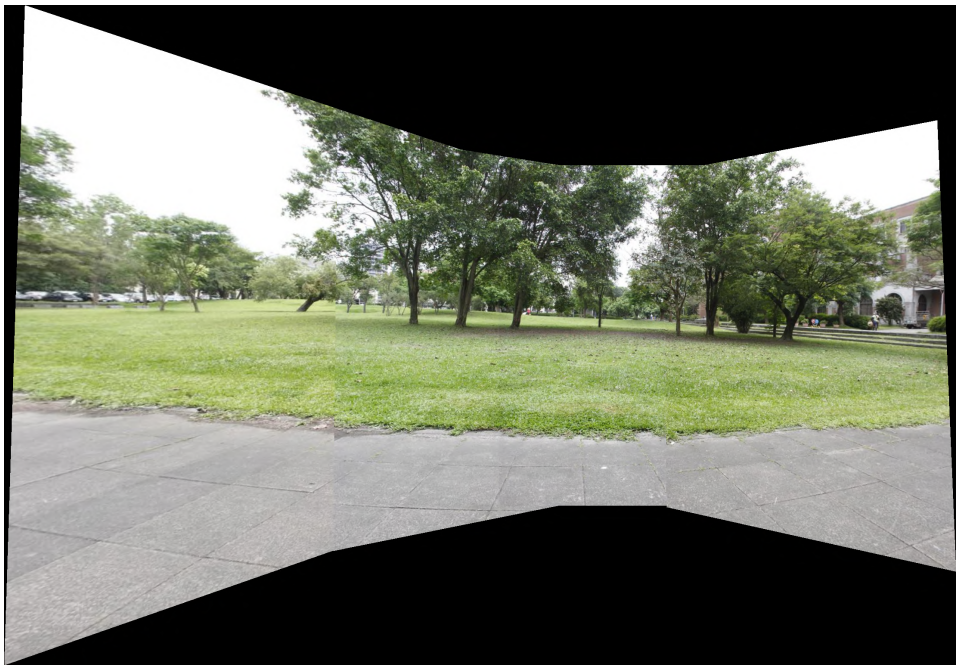


Figure 14: (0.05, 0.8, 0.55). Blend 4 images.

Figure 15: (0.1, 0.5, 0.8). Blend 5 images.

Figure 16: (0.05, 0.8, 0.55). Blend 5 images.

Figure 17: (0.05, 0.8, 0.55). Blend 7 images.

Figure 18: (0.03, 0.5, 0.6). Blend 3 images.

Figure 19: (0.05, 0.8, 0.55). Blend 3 images.