VFX Project 2 Report

Steven R04922170 / Han B01902078

2016/5/5

Contents

1 Project Description

In this project, we are asked to implement part of the *Recognising Panoramas* paper.

There are basically 4 steps that we have to do:

- feature detection
- feature matching
- image matching
- blending

In order to complete the tasks, we have implemented several vital functions in MATLAB as follows.

- `my_harris()`
- `descriptor()`
- `cylindrical_projection()`
- `matches()`
- `plotMatches()`
- `combine()`

The main program entry would be `hw2()` function in `hw2.m`. We will be discussing what we have done in those functions as well as their functionality in the sections below.

2 Vital Functions

2.1 my_harris()

In this function, we implemented the Harris Corner Detection with `k=0.04` and `threshold=50000`. We also applied a 5x5 Gaussian window. This function takes 1 parameter `rgbimg`, where we'd like to extract feature points, and a matrix of `0`'s and `1`'s is returned. Those positions with value 1 in the binary matrix returned are the keypoints.

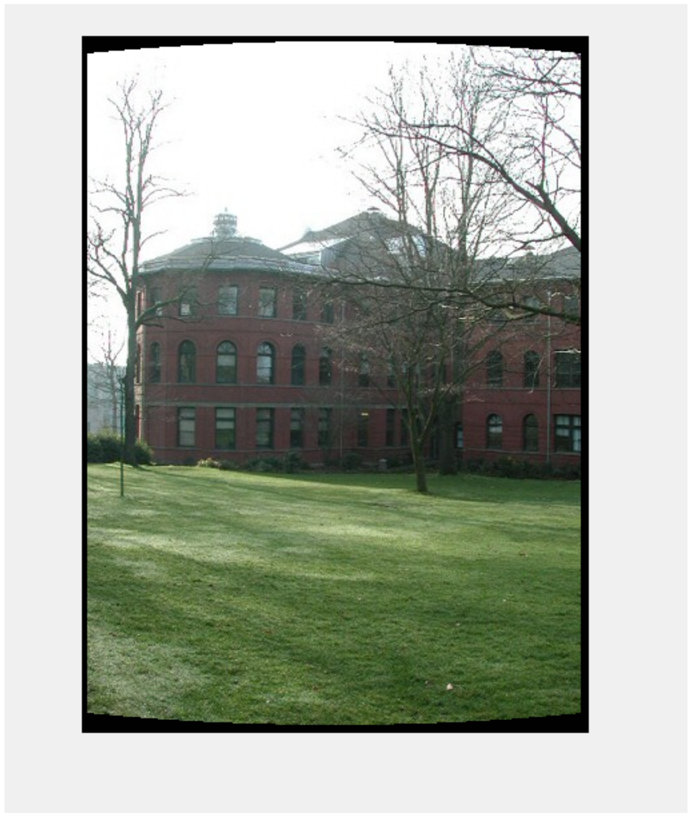
2.2 descriptors()

This function helps "describe" the keypoints we retrieved with Harris Corner Detection algorithm. Each keypoint is described with a 128-dimension descriptor vector. The first parameter for this function is the binary matrix we'd get from `my_harris()` function, and the second parameter is the grayscale image. A set of 128-dimension descriptor vectors will be returned.

2.3 cylindrical_projection()

Given focal length and cylindrical radius, this function warp the original image onto cylindrical surface. In addition to the warped image, which will be returned as the first of the 2 returned elements, the coordinate look-up table will also be returned.

Here's the warped result after calling this function:

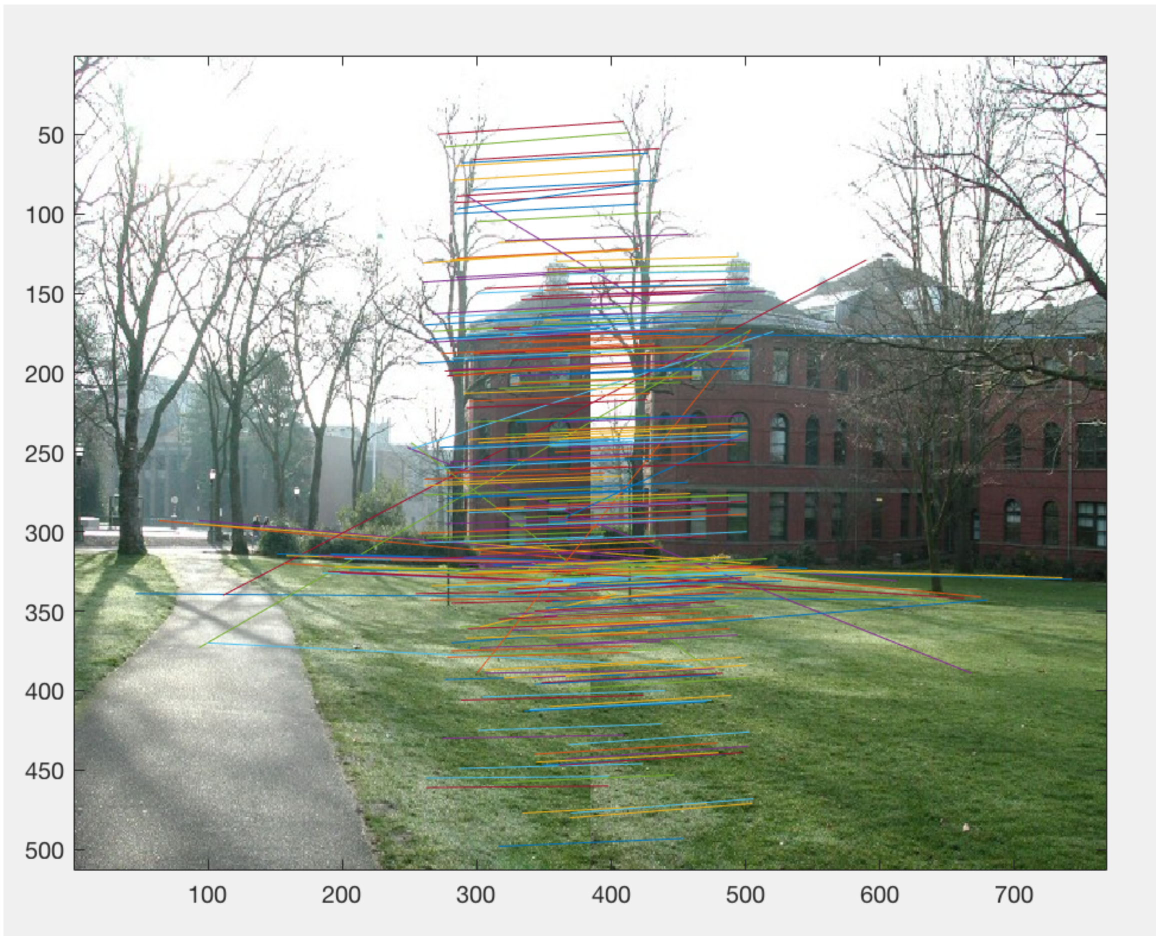


2.4 matches()

Taking the descriptor vectors of an image, say `des1`, and the descriptor vectors of another image, say `des2`, this function calculates the cosine similarity to match the keypoints into pairs. There are two parameters, `des1` and `des2`, and the function returns a structure containing all matched pairs as well as their cosine similarity.

2.5 plotMatches()

This is a function that we don't actually have to use for this homework. However, it helps illustrate the result of the matching of keypoints in 2 images. It would output something as below:



2.6 combine()

The `combine()` function is very essential in our implementation. Basically it takes 2 cylindrical images, and returns the combined image. After calculating the shift of the matched keypoints in the two given images, voting mechanism is applied to choose the highest agreement. As for blending, we use a weighted function for the overlapped pixels with regard to the distance to each of the 2 images.

The parameters for this function are listed as follows:

`rgbimgR` the color image on the right side

`CylImg1` the cylindrical image on the left

`CylImg2` the cylindrical image on the right

`c_coord1` the look-up table for `CylImg1`

`c_coord2` the look-up table for `CylImg2` `matched_ans` the thing returned by `matches()`

3 How to Run Our Program

For running our program, besides all the images that you'd like to stitch together, there's also one plain-text file required, and we call it *The Description File*.

3.1 The Description File

This file should contain the information about the images, both their paths and their focal lengths. The format of this plain-text file is as follows.

```
<image1 path> <image1 focal length>
<image2 path> <image2 focal length>
<image3 path> <image3 focal length>
...
```

There should be `N` lines, where `N` is the number of images that we'd like to stitch. For each line, a string and a floating-point number are presented with a space in between them, and they are the path and the focal length for the image, respectively.

3.2 Execution

After the description file and all the images are ready, we can now execute the program. First, let's go into the `matlab/` directory by:

```
cd matlab/
```

, and then run this command:

```
matlab -r "hw2('/path/to/the/description/file.txt');exit"
```

. Don't forget to replace the path of the description file to yours.

3.3 The Result

When the `matlab` command is executed completely, the stitched panorama image will be stored as `result.jpg` in the exactly same directory. The following panorama image is our favorite artifact.