Project #4 report

Part #2

Forward warping is easily to implement but it gives us problems. Since in forward transformation our new coordinates are fractional number, the coordinates of positions are rounded when pixels are copied. Thus some pixels may lose during transformation. So we choose use backward transformation. Using a lager output image so it guarantee that every pixel will be copied into new image.

Forward transformation

Backward transformation

Part #3

To execute part 3 easily we have created a bash script. The script takes 4 arguments and creates the appropriate call to Project 4 executable.

```
./do_p4_part3 100.0 attraction_images/bigben_10.jpg attraction_images bigben
```

The list of arguments is the following:

1) Threshold
2) The file that will be the query
3) The directory where ALL files are (including the query file!)
4) The output filename

The execution won't print anything because all stream output is being sent to a file at output/ folder. The file is created on the following format: out_*OUTPUTFILENAME_THRESHOLD* . In case of duplicate, it adds a number to differentiate it. This script was made to make the parallel execution of multiple instances of part3 possible.
The script will also create the list of 100 arguments that we needed for the execution of the attraction comparison. The script takes the folder of the third argument and creates the argument list with all the files of that folder but the file that is in the second argument(because the file of the second argument will be the query).

The output of the script will call the program with something like this:

```
./p4 part3 100.0 attraction_images/bigben_10.jpg {LIST OF ALL IMAGES BUT QUERY} > output/out_bigben_130.0
```

Inside the output/out_bigben_100.0 the results of the #100 ranked images will be printed. Also the output file will be called sift.png and will show the matches between the query and the best image.

To find the best image first, the program will compute the SIFT descriptors for the query image. For each one of the 99 images left, the program will compute the SIFT descriptors and find the closest matches of the two descriptors and pair up. The condition of pairing is to pair two descriptors that the distance is less than the threshold. If this condition is satisfied, the vector between the two descriptors is computed and then added to the list of descriptor pairs. The images are ranked according two the number of matches.

Results:

For each attraction we randomly choose two query images then we have top 10 ranked images related to each query image. The output includes the sift image of each query image and its best-fit image. Basing on the top 10 ranked images we calculate the precisionof matching for each attraction (precision = correct_image / 10).

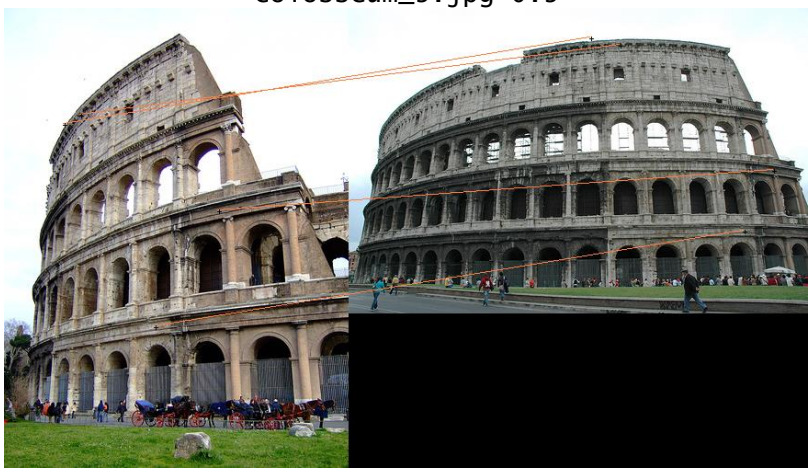
Query image / Rank	bigben_10.jpg	bigben_3.jpg
1	londoneye_13.jpg	bigben_12.jpg
2	tatemodern_9.jpg	sanmarco_19.jpg
3	notredame_1.jpg	bigben_14.jpg
4	empirestate_9.jpg	sanmarco_4.jpg
5	tatemodern_4.jpg	bigben_6.jpg
6	colosseum_12.jpg	tatemodern_13.jpg
7	notredame_4.jpg	notredame_25.jpg
8	notredame_5.jpg	sanmarco_1.jpg
9	empirestate_12.jpg	sanmarco_18.jpg
10	bigben_14.jpg	sanmarco_14.jpg
Precision	0.1	0.3

bigben_10.jpg=0.1

bigben_3.jpg=0.3

Query image / Rank	colosseum_3.jpg	colosseum_13.jpg
1	colosseum_8.jpg	trafalgarsquare_20.jpg
2	trafalgarsquare_22.jpg	tatemodern_14.jpg
3	eiffel_15.jpg	notredame_1.jpg
4	tatemodern_4.jpg	notredame_24.jpg
5	notredame_19.jpg	colosseum_3.jpg
6	tatemodern_8.jpg	sanmarco_1.jpg
7	tatemodern_24.jpg	sanmarco_19.jpg
8	trafalgarsquare_15.jpg	sanmarco_18.jpg
9	colosseum_13.jpg	sanmarco_14.jpg
10	colosseum_12.jpg	sanmarco_13.jpg
precision	0.3	0.1

colosseum_3.jpg=0.3

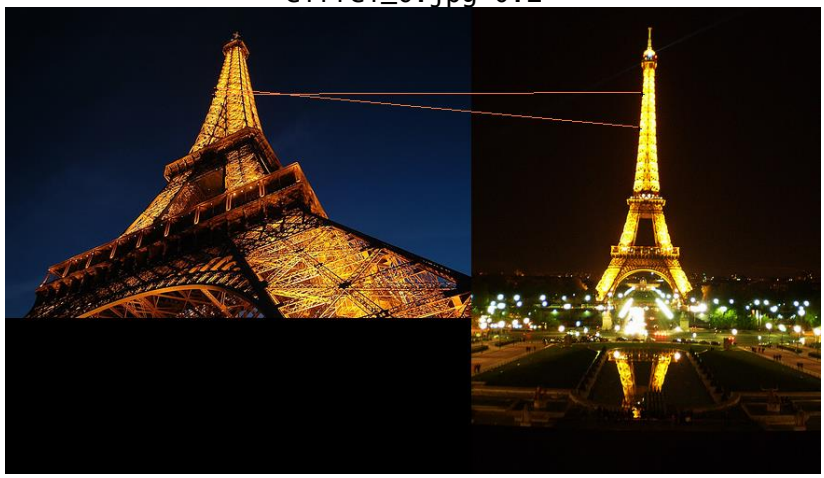


colosseum_13.jpg=0.1

Query image / Rank	eiffel_18.jpg	eiffel_6.jpg
1	eiffel_19.jpg	eiffel_18.jpg
2	empirestate_12.jpg	empirestate_15.jpg
3	louvre_3.jpg	sanmarco_20.jpg
4	eiffel_22.jpg	sanmarco_1.jpg
5	trafalgarsquare_15.jpg	sanmarco_19.jpg
6	bigben_12.jpg	sanmarco_18.jpg
7	eiffel_6.jpg	sanmarco_14.jpg
8	trafalgarsquare_22.jpg	sanmarco_13.jpg
9	empirestate_25.jpg	notredame_8.jpg
10	empirestate_10.jpg	notredame_5.jpg
precision	0.3	0.1

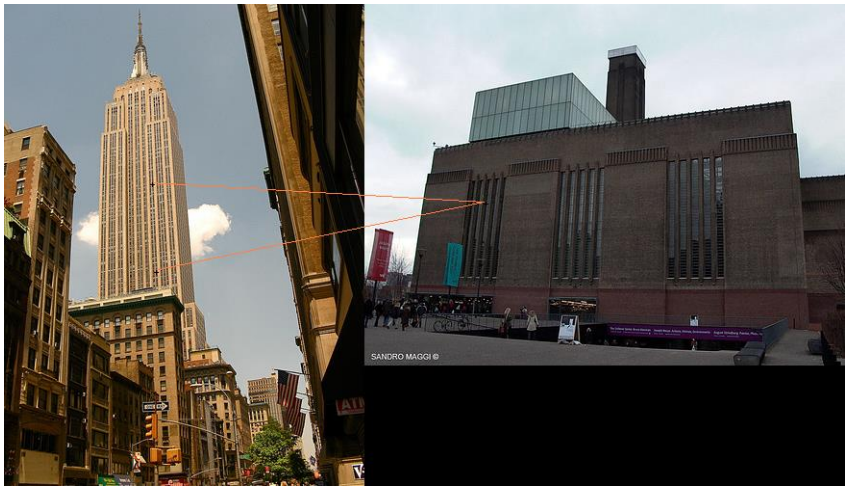
eiffel_18.jpg=0.3

eiffel_6.jpg=0.1



Query image / Rank	empirestate_14.jpg	empirestate_25.jpg
1	tatemodern_13.jpg	notredame_20.jpg
2	tatemodern_8.jpg	eiffel_18.jpg
3	londoneye_9.jpg	sanmarco_22.jpg
4	sanmarco_20.jpg	empirestate_23.jpg
5	louvre_4.jpg	empirestate_16.jpg
6	trafalgarsquare_22.jpg	empirestate_12.jpg
7	louvre_3.jpg	sanmarco_18.jpg
8	bigben_7.jpg	notredame_14.jpg
9	empirestate_15.jpg	eiffel_2.jpg
10	empirestate_12.jpg	trafalgarsquare_5.jp
precision	0.2	0.3

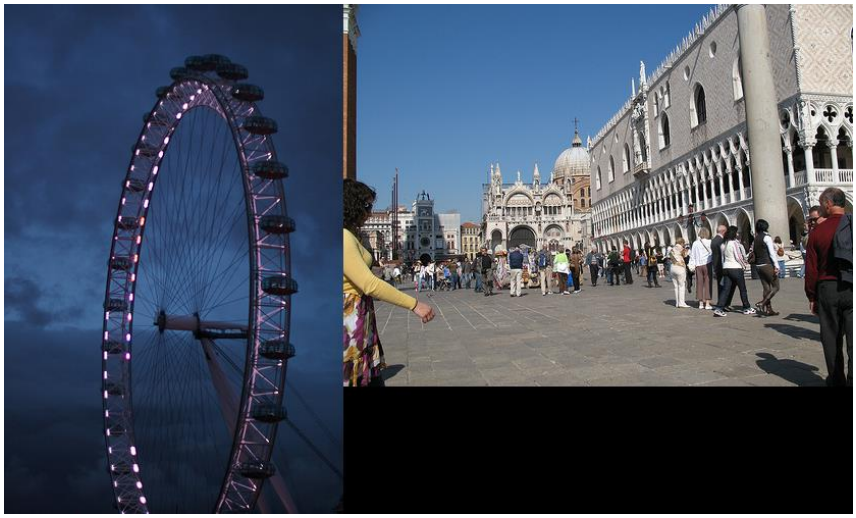
empirestate_14.jpg=0.2



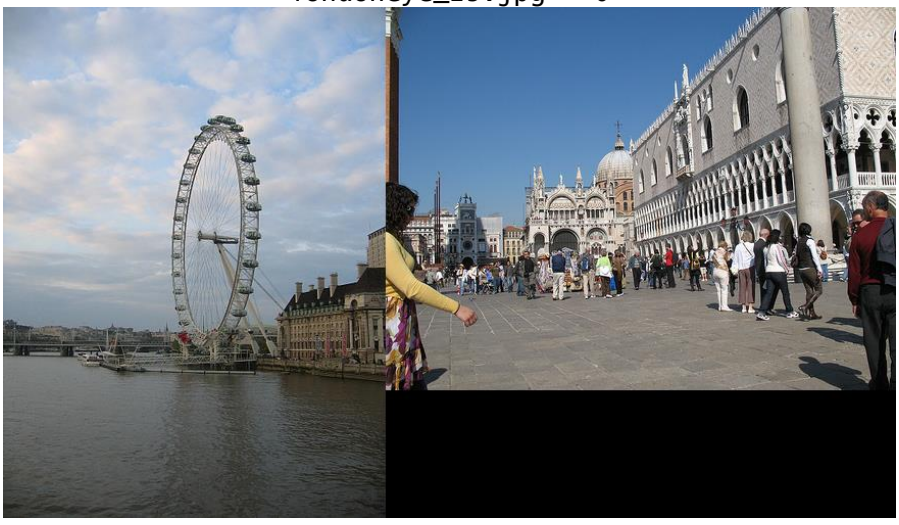
empirestate_25.jpg=0.3

Query image / Rank	londoneye_8.jpg	londoneye_23.jpg
1	sanmarco_20.jpg	sanmarco_20.jpg
2	sanmarco_1.jpg	sanmarco_1.jpg
3	sanmarco_19.jpg	sanmarco_19.jpg
4	sanmarco_18.jpg	sanmarco_18.jpg
5	sanmarco_14.jpg	sanmarco_14.jpg
6	sanmarco_13.jpg	sanmarco_13.jpg
7	notredame_8.jpg	notredame_8.jpg
8	notredame_5.jpg	notredame_5.jpg
9	notredame_4.jpg	notredame_4.jpg
10	notredame_3.jpg	notredame_3.jpg
precision	0	0

londoneye_8.jpg = 0

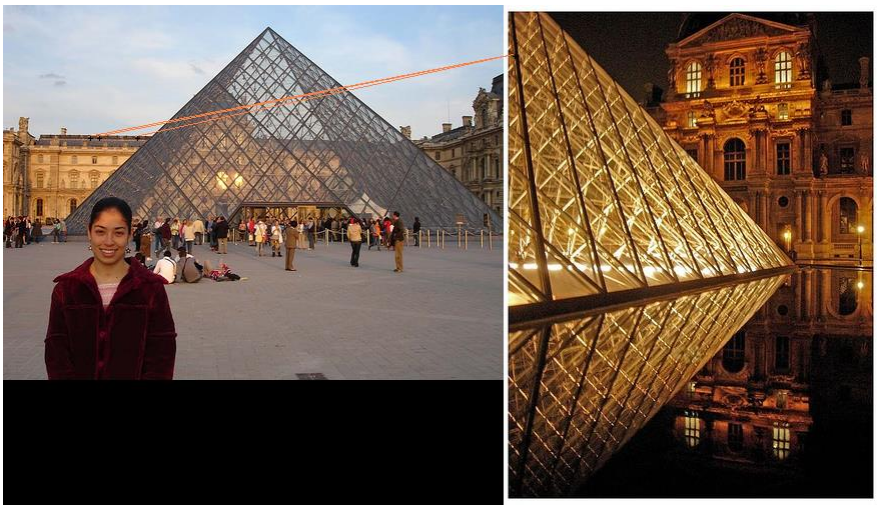


londoneye_23.jpg = 0

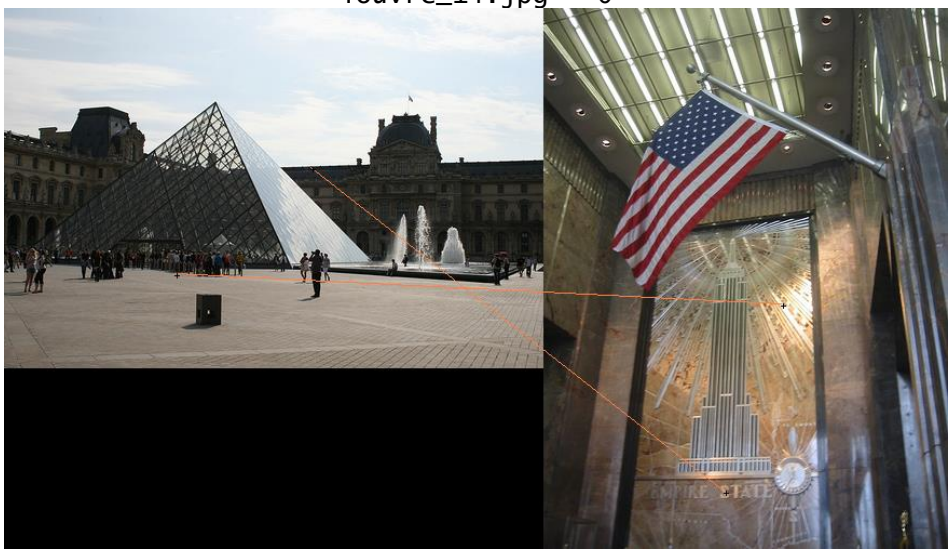


Query image / Rank	louvre_4.jpg	louvre_14.jpg
1	louvre_15.jpg	empirestate_15.jpg
2	bigben_12.jpg	bigben_12.jpg
3	empirestate_23.jpg	tatemodern_24.jpg
4	empirestate_16.jpg	tatemodern_6.jpg
5	empirestate_15.jpg	bigben_7.jpg
6	empirestate_14.jpg	sanmarco_18.jpg
7	empirestate_12.jpg	empirestate_9.jpg
8	sanmarco_20.jpg	tatemodern_13.jpg
9	eiffel_22.jpg	londoneye_13.jpg
10	notredame_4.jpg	colosseum_5.jpg
precision	0.1	0

louvre_4.jpg = 0.1

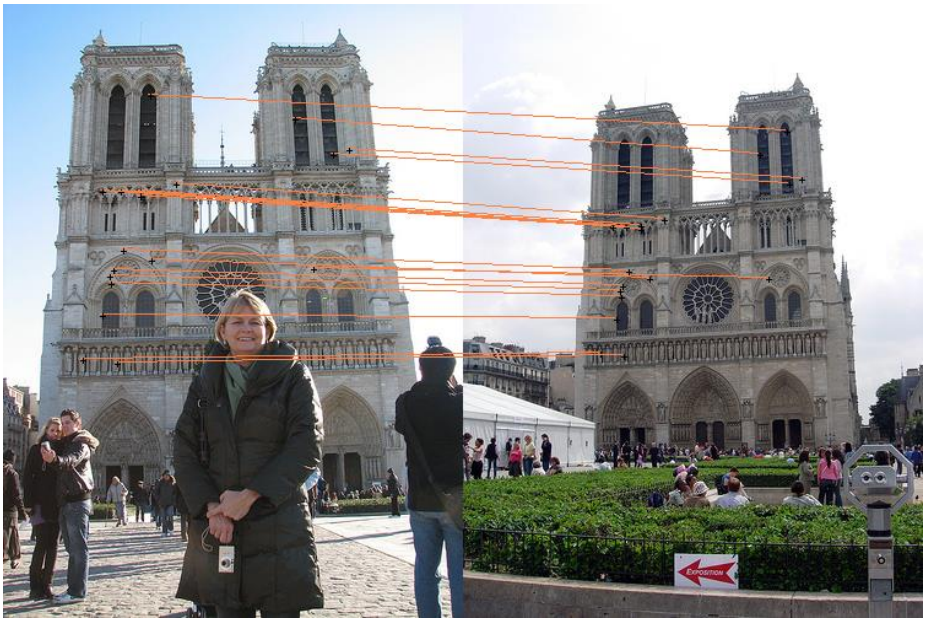


louvre_14.jpg = 0

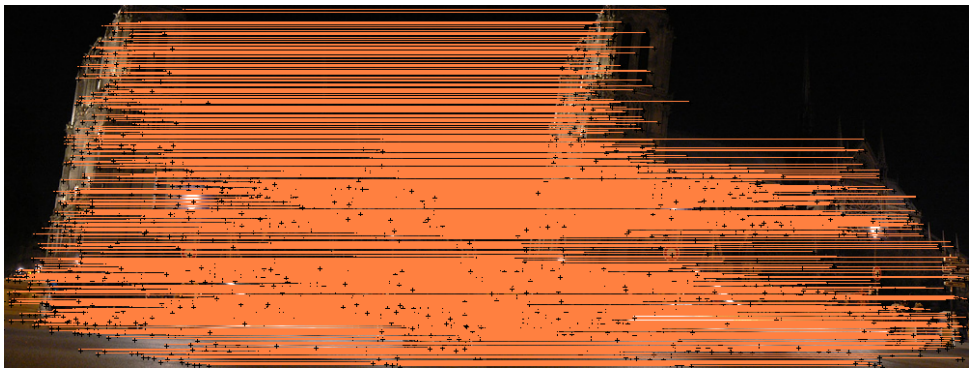


Query image / Rank	notredame_19.jpg	notredame_20.jpg
1	notredame_24.jpg	notredame_20.jpg
2	trafalgarsquare_5.jpg	trafalgarsquare_22.jpg
3	empirestate_15.jpg	louvre_16.jpg
4	tatemodern_24.jpg	eiffel_3.jpg
5	trafalgarsquare_20.jpg	londoneye_9.jpg
6	notredame_25.jpg	empirestate_25.jpg
7	bigben_7.jpg	notredame_25.jpg
8	tatemodern_13.jpg	londoneye_17.jpg
9	empirestate_12.jpg	notredame_24.jpg
10	louvre_3.jpg	empirestate_15.jpg
precision	0.2	0.2

notredame_19.jpg = 0.2



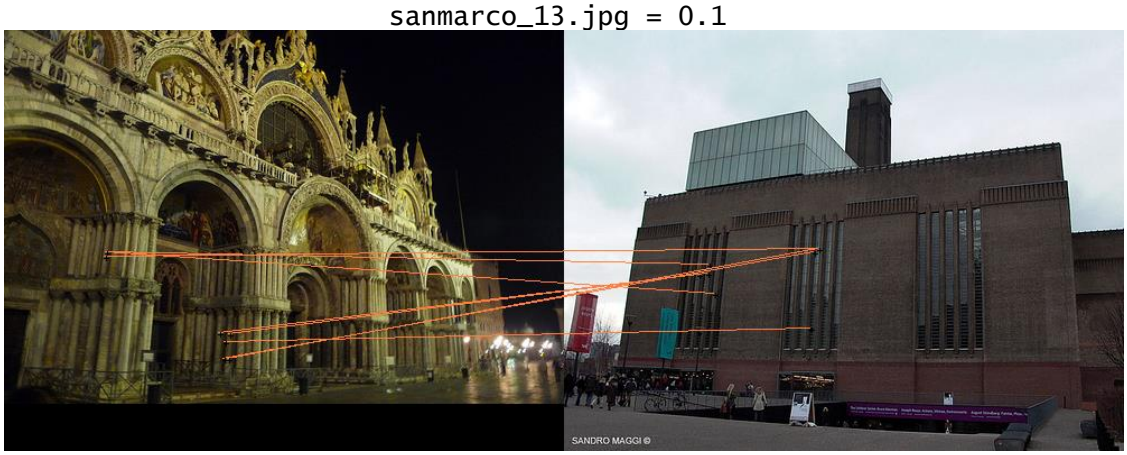
notredame_19.jpg = 0.2



Query image / Rank	sanmarco_13.jpg	sanmarco_19.jpg
1	tatemodern_13.jpg	tatemodern_13.jpg
2	sanmarco_19.jpg	empirestate_27.jpg
3	tatemodern_9.jpg	empirestate_15.jpg
4	sanmarco_4.jpg	sanmarco_13.jpg
5	empirestate_10.jpg	notredame_19.jpg
6	notredame_5.jpg	notredame_4.jpg
7	trafalgarsquare_5.jpg	empirestate_10.jpg
8	notredame_3.jpg	louvre_9.jpg
9	empirestate_23.jpg	londoneye_13.jpg
10	notredame_4.jpg	bigben_3.jpg
precision	0.2	0.1

sanmarco_13.jpg = 0.1

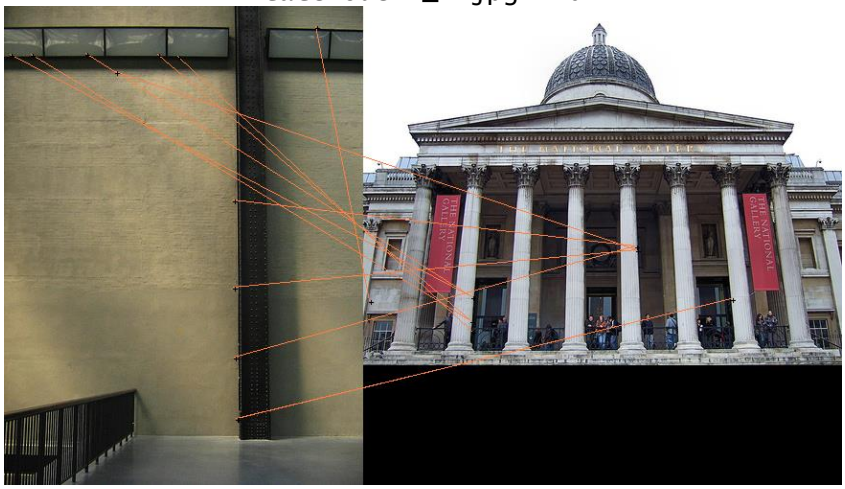
sanmarco_13.jpg = 0.1



Query image / Rank	tatemodern_11.jpg	tatemodern_4.jpg
1	empirestate_12.jpg	trafalgarsquare_5.jpg
2	londoneye_9.jpg	notredame_4.jpg
3	empirestate_27.jpg	empirestate_12.jpg
4	tatemodern_13.jpg	empirestate_15.jpg
5	londoneye_12.jpg	trafalgarsquare_20.jpg
6	empirestate_10.jpg	sanmarco_14.jpg
7	empirestate_16.jpg	empirestate_23.jpg
8	louvre_8.jpg	empirestate_9.jpg
9	empirestate_9.jpg	notredame_25.jpg
10	empirestate_15.jpg	louvre_16.jpg
precision	0.1	0

tatemodern_11.jpg = 0.1

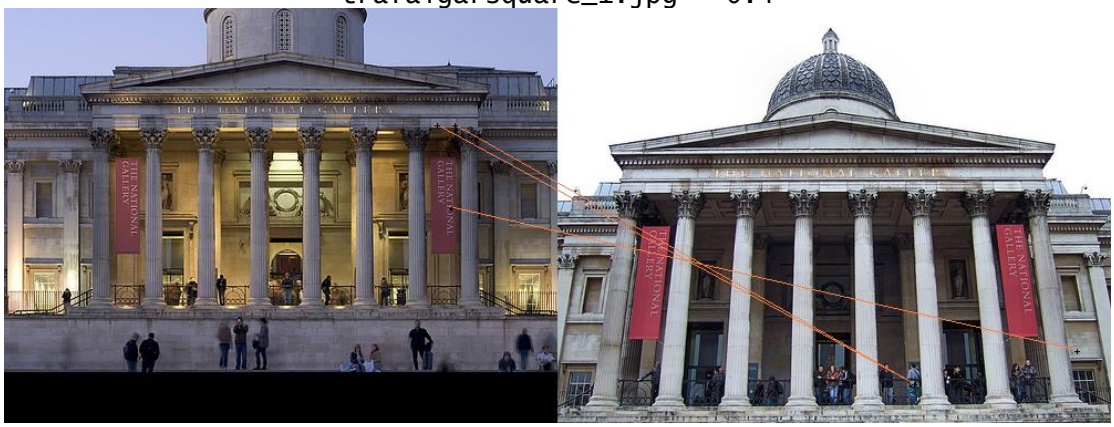
tatemodern_4.jpg = 0



Query image / Rank	trafalgarsquare_15.jpg	trafalgarsquare_1.jpg
1	louvre_3.jpg	trafalgarsquare_5.jpg
2	trafalgarsquare_5.jpg	trafalgarsquare_20.jpg
3	empirestate_10.jpg	trafalgarsquare_22.jpg
4	empirestate_12.jpg	trafalgarsquare_16.jpg
5	eiffel_19.jpg	tatemodern_14.jpg
6	eiffel_18.jpg	sanmarco_14.jpg
7	tatemodern_11.jpg	sanmarco_13.jpg
8	notredame_4.jpg	notredame_8.jpg
9	colosseum_6.jpg	notredame_5.jpg
10	empirestate_9.jpg	notredame_4.jpg
precision	0.1	0.4

trafalgarsquare_15.jpg = 0.1

trafalgarsquare_1.jpg = 0.4



Some analysis: The averaging percision is around 20%. One reason is that we may choose too small threshold so filtering is very strict. However, it also explains that, in general case, our best-fit macthing works well.
The macthing really depends on how the images were taken. Camera angle, shooting distane and irrelevent content can easily mess up the matching. If one image has few and distinct feature (such as londoneye), while other images contains many and trivial features, it results in bad matching. Also our picture base is very small so we can't normalize the error of matching and the weight of features.

Part #4

To execute part 4 easily we have created a bash script. The script takes 3 arguments and creates the appropriate call to Project 4 executable.

```
./do_p4_part4 sage.png sage 0.1
```

The list of arguments is the following:

5) filename of output
6) name of the file inside of stitching_images folder. (It needs 2 images inside the folder following the format name_1.png and name_2.png)
7) ratio of closest distance and second closest distance

The example above executes this line of code:

```
./p4 part4 sage.png stitching_images/sage_1.png stiting_images/sage_2.png 0.1
```

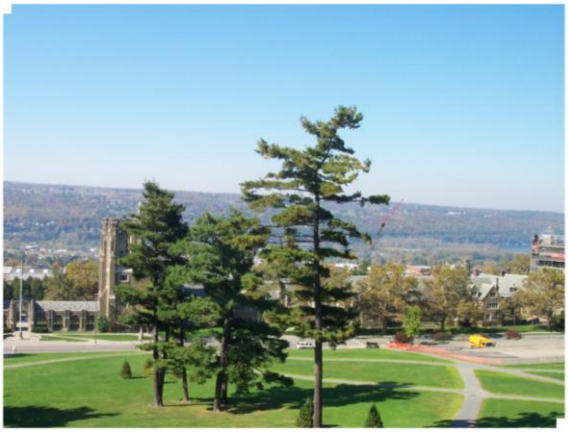
The program will get the two images and compute the SIFT descriptors. The closest matches of the two images will pair up the descriptors. The condition of pairing is to pair two descriptors that are closest to each other only if the ratio between their distance and the distance of their second closest match is less then the ratio. If this condition is satisfied, the vector between the two descriptors is computed and then added to the list of descriptor pairs.

After computing the list of pairs of descriptors, we need to compute the displacement vectors. To do that, we select some elements at random and we compare which of the vectors has the most vectors that agree with its displacement. The number of selected elements is 30% of the total of matches. To be considered an agreement, the absolute difference between two vectors needs to be less then 2 pixels.

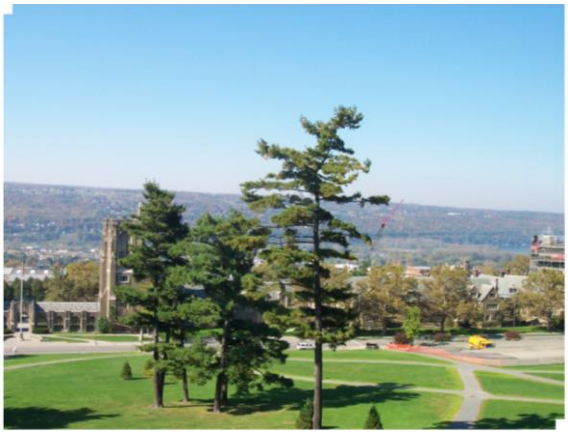
By computing this vector we can place both images correctly on top of each other and save the result using the output filename.

Results:

Part #4 using the first method (only ratio of selection, no inlier count) :



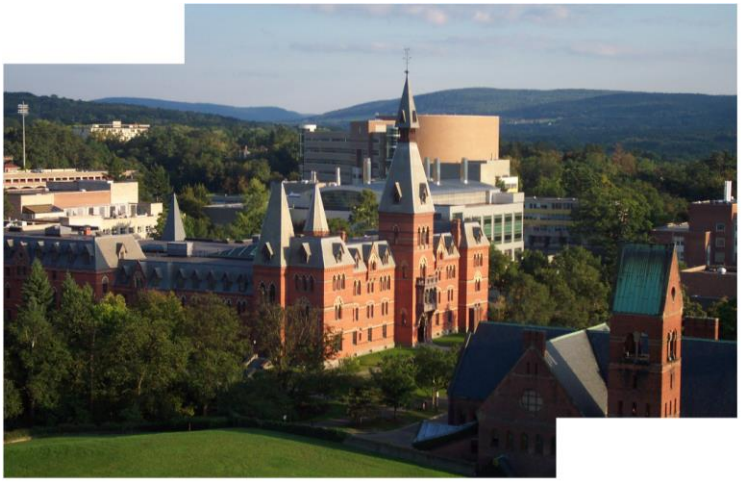
ratio = 0.1



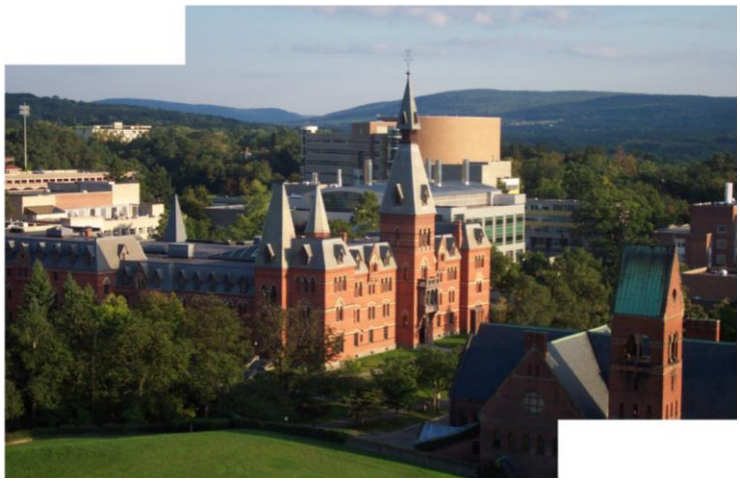
ratio = 0.5

Sage:

ratio = 0.1



ratio = 0.5

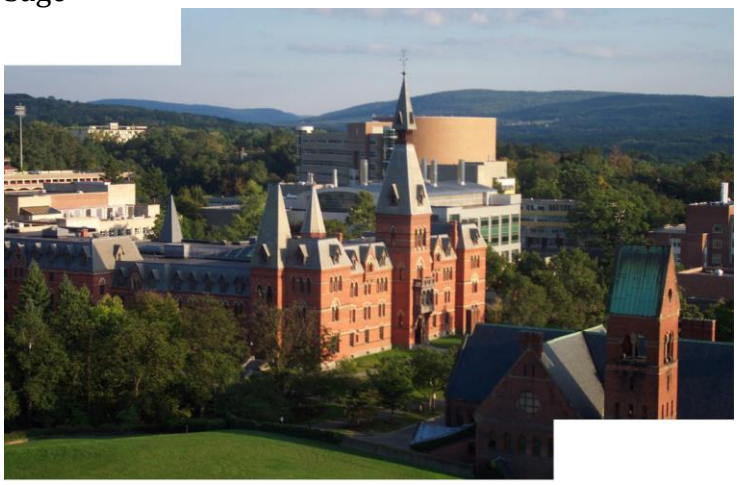


Part #4 with inlier counter

ratio = 0.5

Cayuga

Sage



Mcfaddin

Falls

Start