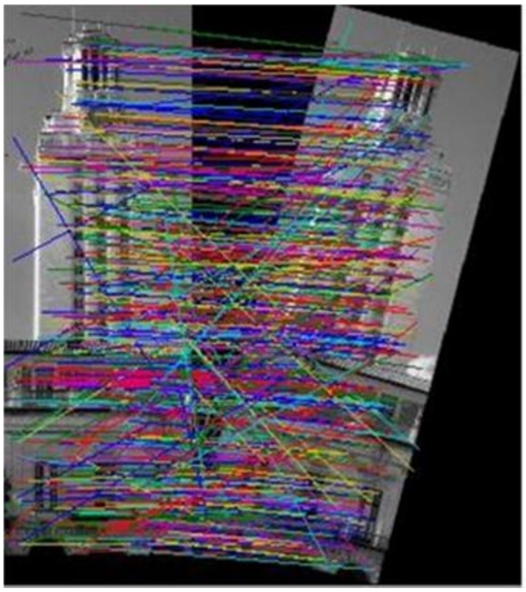
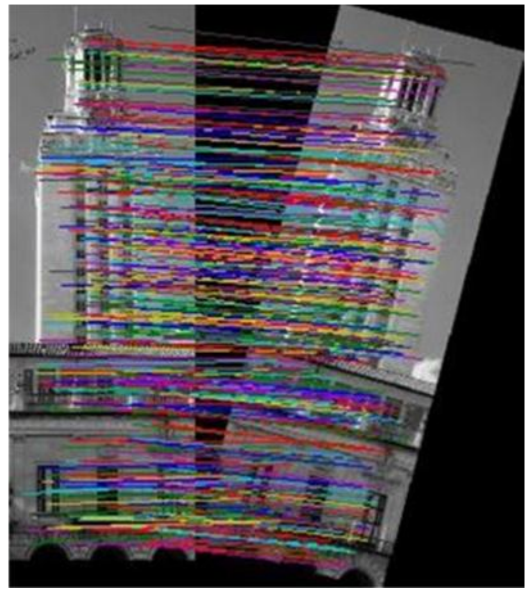
Name: SHAH, MIT KALPESHKUMAR **UT EID**: mks3226

Visual Recognition *Coding Assignment 1- Recognizing specific objects with local feature matching*

Question 1: **Matches with Object-template-rotated**

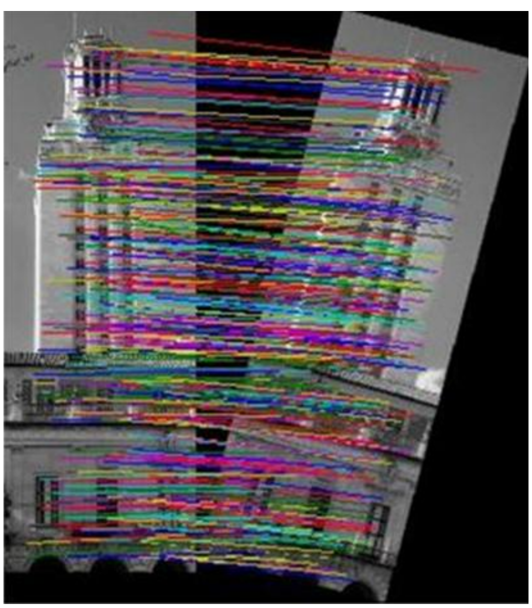


Initial Matches (Without Any test)



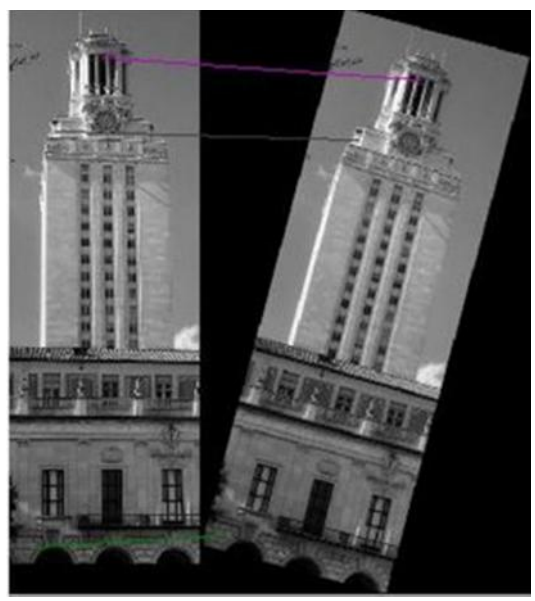
After Applying **Threshold Nearest Neighbor**

Survived: **369**



After Applying **Threshold Ratio Test**

Survived: **357**

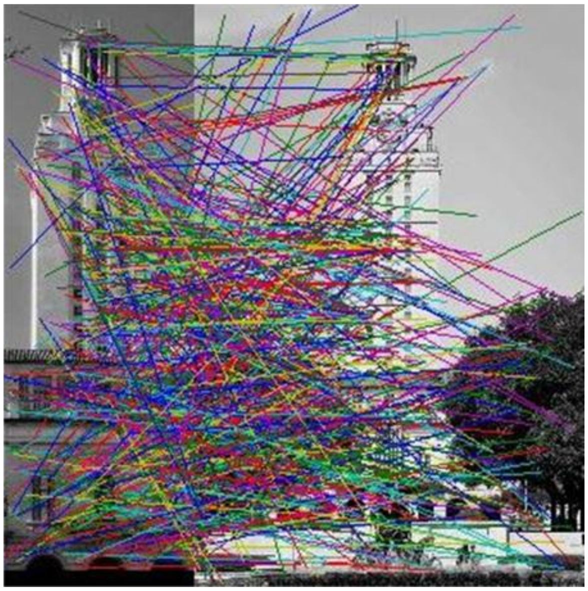


After Applying **RANSAC**

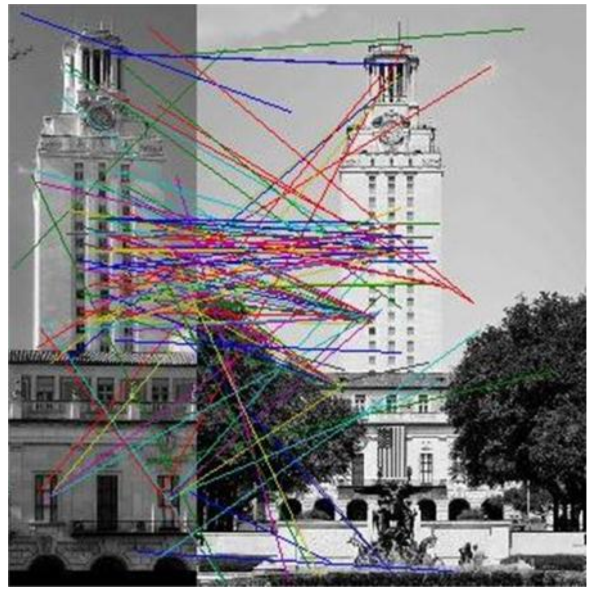
Survived: **7**

Question 1

Matches with Scene1

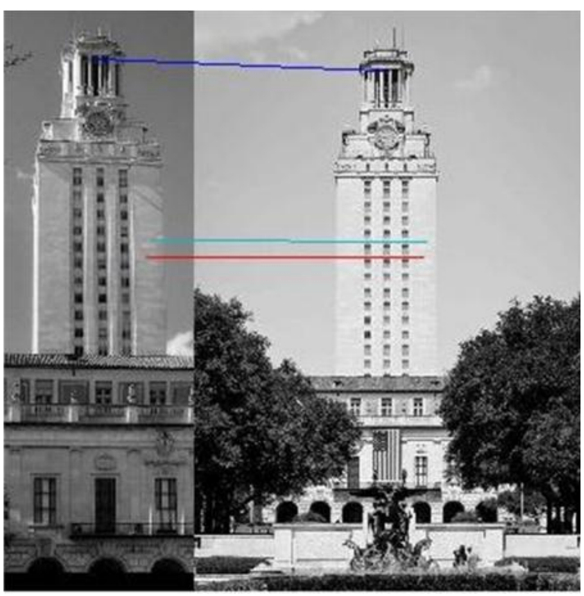


Initial Matches (Without Any test)



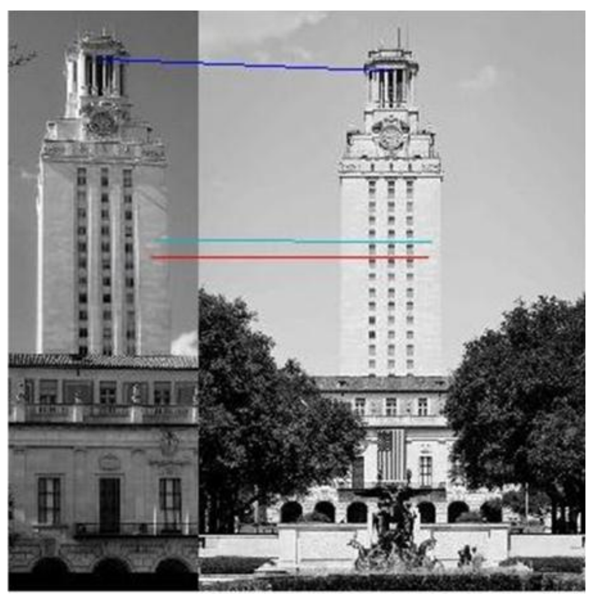
After Applying **Threshold Nearest Neighbor**

Survived: 124



After Applying **Threshold Ratio Test**

Survived: 8

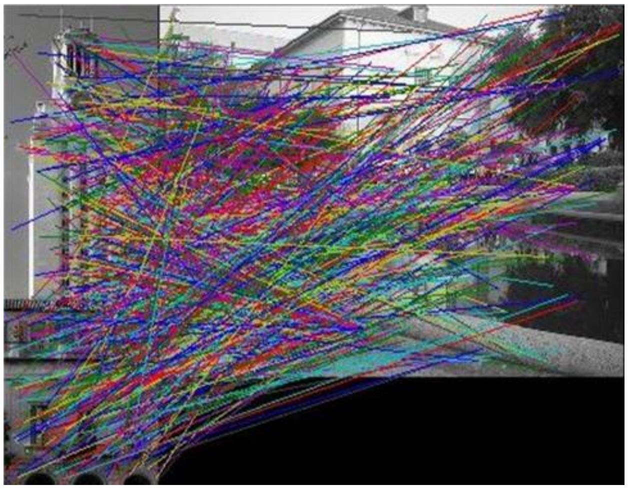


After Applying **RANSAC**

Survived: 4

Question 1

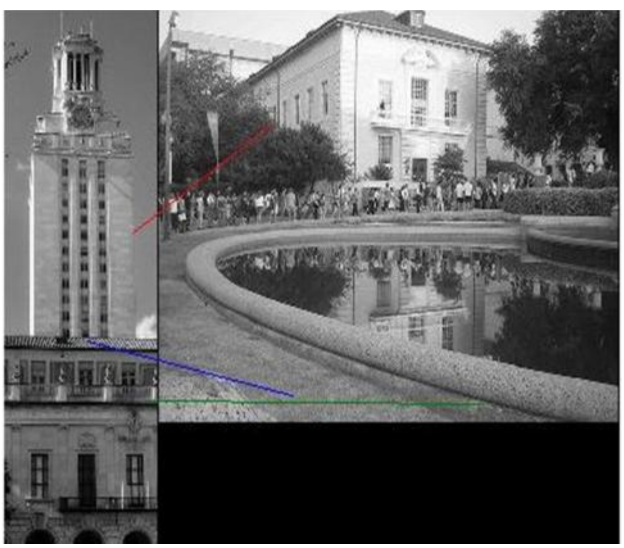
Matches with Scene2



Initial Matches (Without Any test)

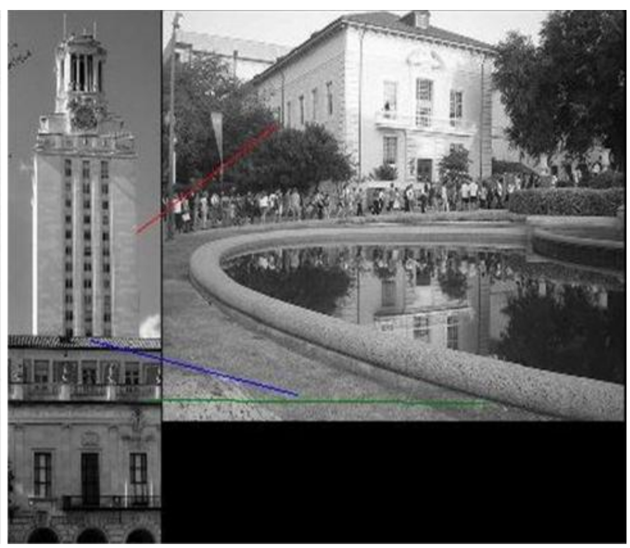
After Applying **Threshold Nearest Neighbor**

Survived: 123



After Applying **Threshold Ratio Test**

Survived: 3



After Applying **RANSAC**

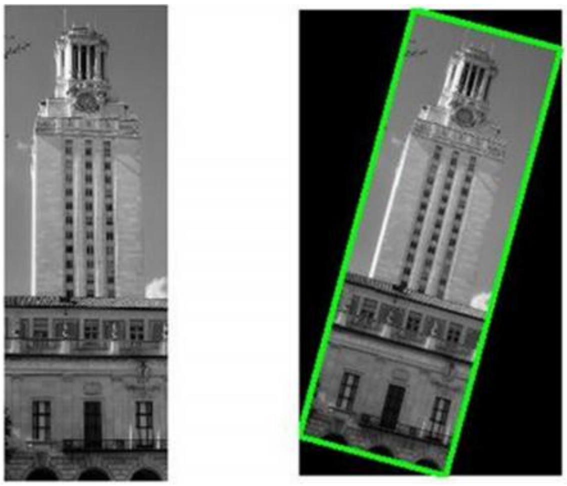
Survived: 3

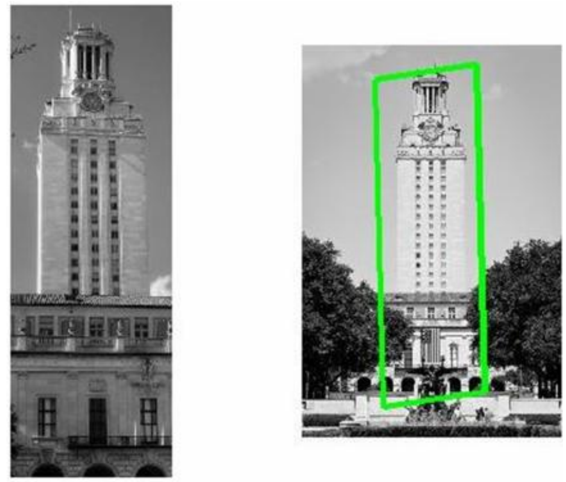
Question 1

Explanation

In all the cases, from the figures previous to any tests, we can see that a whole lot of matches are found by SIFT method, with many of them having no connections apparently. They only start making sense once we apply Nearest Neighbor test. After applying that, most of the matches survived somewhat similar in overall look. Once Ratio Test is applied, only those features are surviving, who are strongly similar; most of them are removed. Though effect of RANSAC is not that visible form above 3 examples, we can note that in first image – Object-template-rotated, it reduces matches from 357 to just 7.

Question 2





Detection in Object-template-rotated **Detection in Scene1**

Approach to make the detection decision:

Mainly Grid Search (file: gridSearch.m) was performed to find the optimum configuration of thresholds for all the three tests, number of iterations in RANSAC and threshold on inliers to finally classify image as positive / negative.

Additional 18 images (manually labeled) were added for the task.

Following ranges were selected for doing grid search.

1) Nearest Neighbor threshold -> [0.7, 0.75, 0.8, 0.85]

2) Threshold Ratio Test -> [0.5, 0.55, 0.6, 0.65, 0.7]

3) Threshold for RANSAC -> [1e-5, 1e-7, 1e-10]

4) Iterations for RANSAC -> [4000, 6000]

5) Threshold for inliers -> [2, 3, 4]

- For **3)**, initially grid was taken as [0.1, 0.05, 0.01, 0.005], but later was fine-tuned again and again to get the maximum f-measure.

- F-measure was calculated from the output given by detectObject.m script (a vector containing 0/1 for each image) and a vector containing manually given labels (0s/1s) for each image.
- Best configuration found was **0.75, 0.55, 1e-10, 4000, 3** respectively for all 5 parameters.
- So, after matchComparison, images had to go through threshold on number of inliers, where images containing **more than 3 (4, 5, ...) inliers** were only classified as positive.
- Result from this configuration contained **1 False positive** and **2 False Negatives**.

Extra – Testing on Additional Images

The code was tested on Additional 18 images, out of which 11 contained the template image and 7 did not. As mentioned in approach to make the detection; hyper-parameters were chosen by doing grid search on all of these images. When applied final selection of hyper-parameters on all the 21 images (3 given + 18 additional); number of false positives were 1 and false negatives were 2. Additional Images are shown below (Images are scaled to fit the page):


sc3


sc4


sc5


sc6


sc7


sc8


sc9 **(FN)**


sc10**(FN)**


sc11


sc12


sc13


sc14

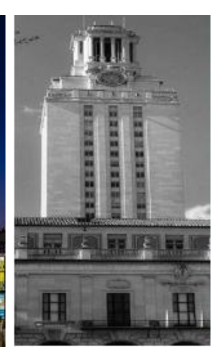

Sc15


sc16**(FP)**


sc17


sc18


sc19


sc20

Positives: sc3, sc5, sc7, sc9, sc10, sc12, sc13, sc15, sc17, sc19, sc20

Negatives: sc4,sc 6, sc8, sc11, sc14, sc16, sc18

False Positives: sc16

False Negatives: sc9, sc10

Notes:

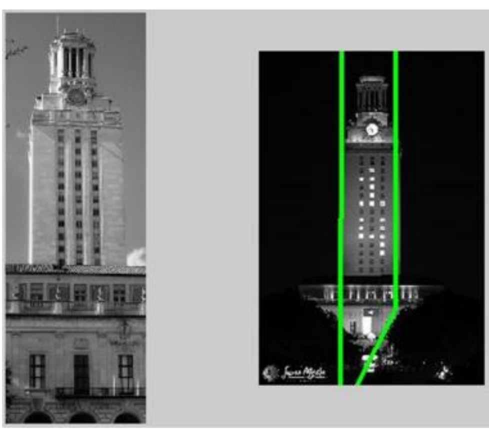
- It seems that, algorithm was good at detecting Negatives, as only 1 of them – sc16 is misclassified as positive. If we look at the image, it does look like some tower, though not of UT.
- Though it was able to detect object in all the images except 2 of them, in most of them it failed to draw a good rectangle around them. Maybe, this is a limitation of affine transformations.

Extra – Question 1

Examples with stronger illumination variation that pass / fail:

Template **sc9** **sc13** **sc19**

From additional images; sc9, sc13, sc19 have great illumination variance from template image. Algorithm is **correctly** able to identify sc13 and sc19; but it **fails** to identify sc9.

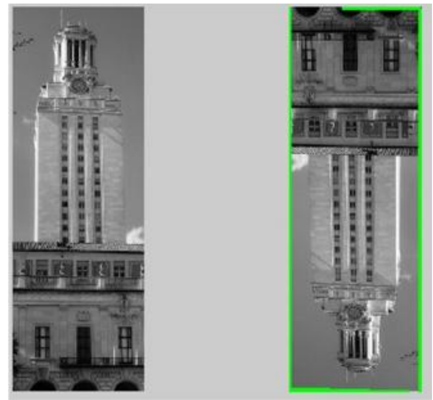


Match for sc13

Match for sc19

As noted above, though detected sc19, it is not able to draw a good rectangle in it. All 4 corners of template are mapped to very nearby points and so rectangle is not visible.

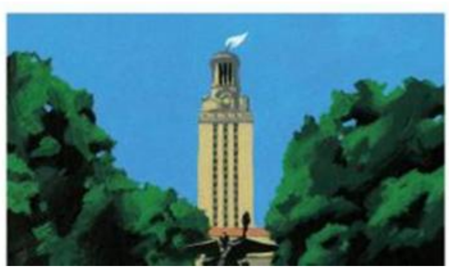
<u>Examples with stronger rotation variation that pass:</u>



Template **sc15**

As we can see, sc15 is 180 degree rotation of the template image. It is classified **correctly**.

<u>Examples with stronger scale variation that pass:</u>



Template **sc17**

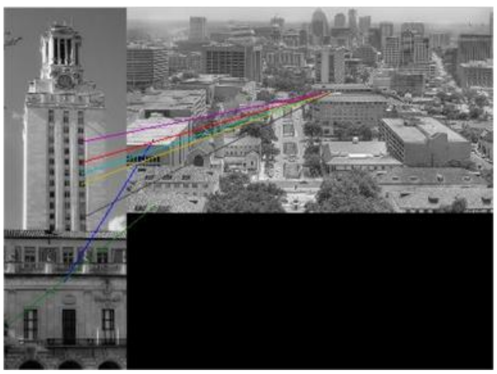
Actual dimensions of template image are **213x620**, where of sc17 image are **300x169**. So, approximately tower is 320 units high in template and 165 units in sc17, which is almost 2:1. Still algorithm classifies it **correctly**.

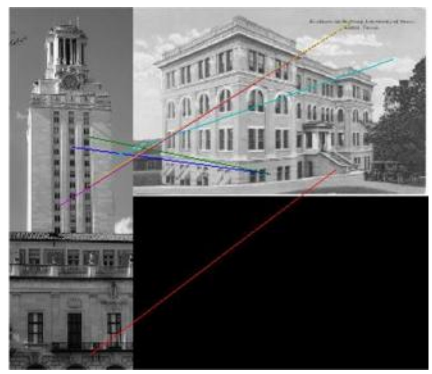
Here also, it is not able to draw good visible rectangle around object.

Extra – Question 2

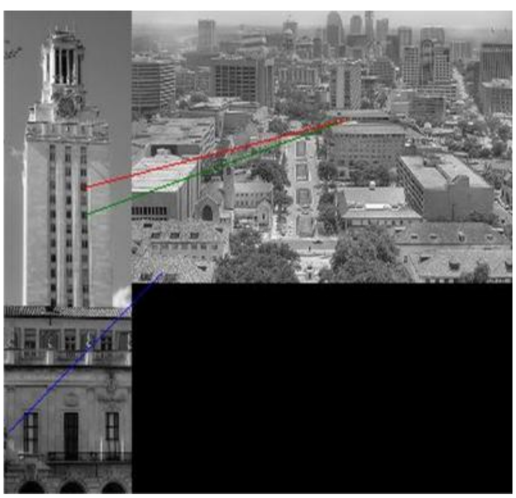
Sc11

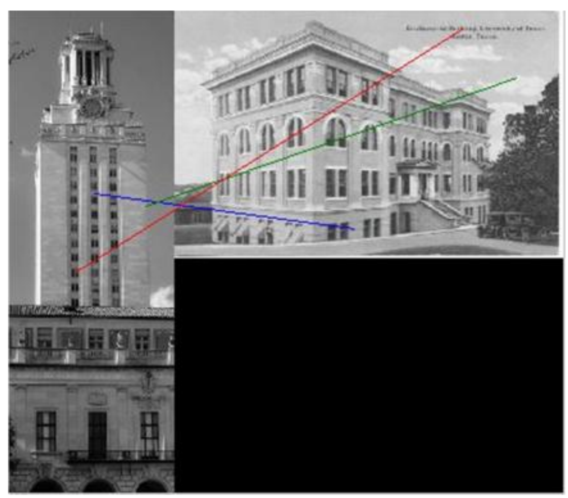
sc8





After Ratio Test





After RANSAC

In images Sc11 and Sc8, **after threshold ratio test**, 7 and 6 matches survived respectively. But **after going through RANSAC**, only 3 matches survived in both of them. As in optimum configuration found through grid search, minimum no of inliers required to classify image as positive were 4 and so, both of them were classified negative.