CS 484 Spring 2022 Homework 2

Full Name: Zübeyir Bodur **ID**: 21702382

1. Preparing Data

The images were successfully downloaded and read in MATLAB. The images that will be stitched together are given below:

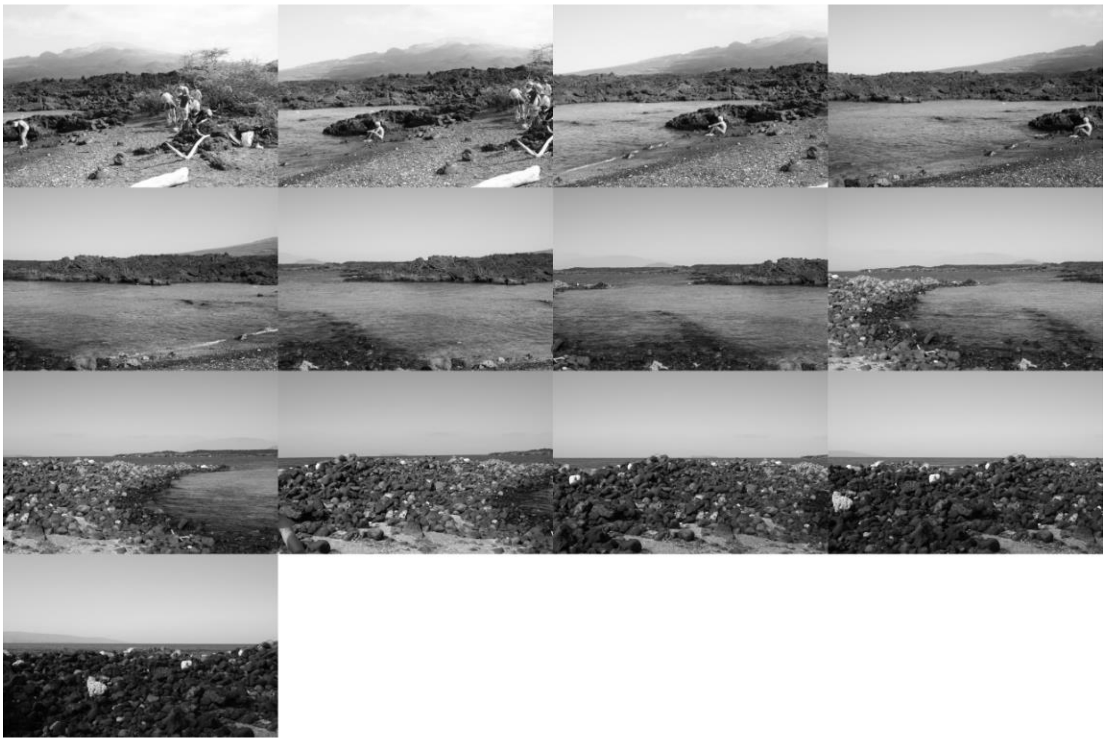

Figure 1: Fishbowl image set

Figure 2: Golden Gate image set

2. Detecting and Describing Local Features

a) Key Points (Interest Points)

The following images show all drawn key points in the first images of both image sets. Some of the key points were too small, so they were not shown by MATLAB.

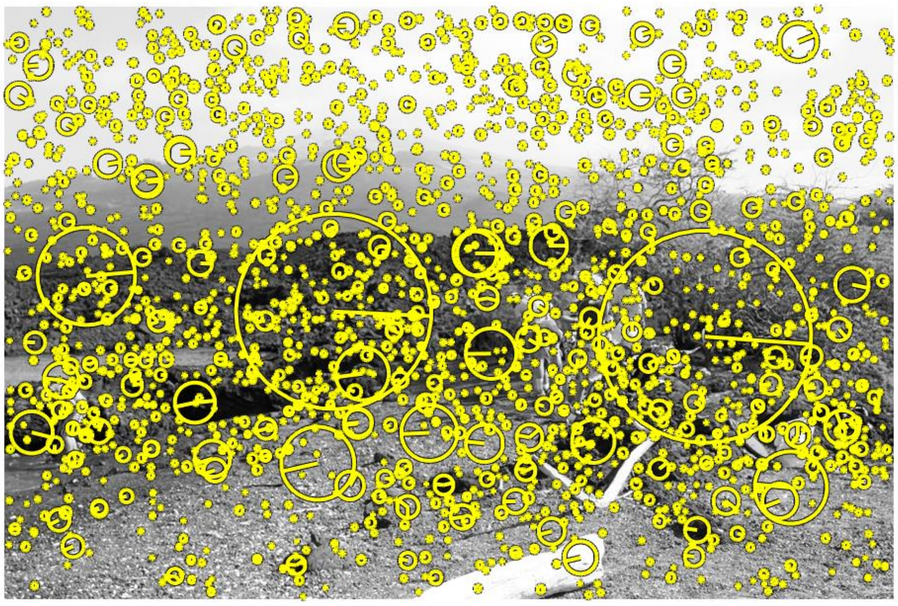

Figure 3: Key Points for the 0^{th} image of fishbowl image set

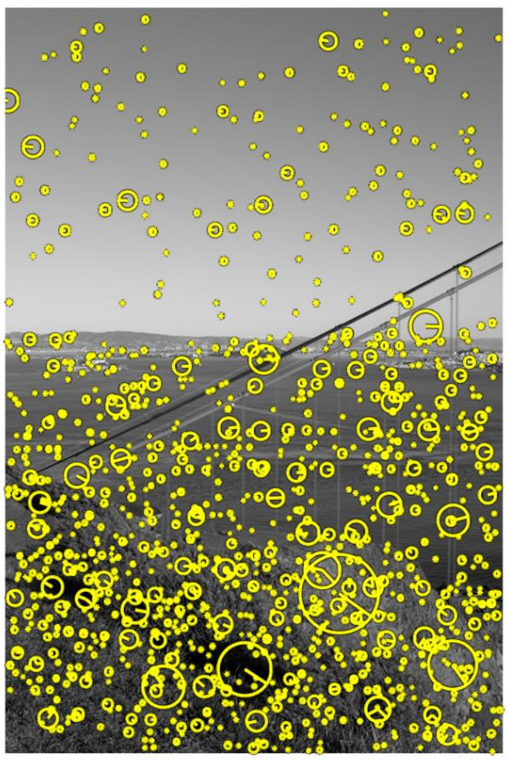

Figure 4: Key Points for the 0^{th} image of golden gate image set

b) Descriptors

i. SIFT Descriptor (Gradient Based)

The following images show 200 randomly selected SIFT descriptors from the first images of Golden Gate and Fishbowl image sets. The reason for displaying 200 random descriptors is that descriptors are large squares, therefore it is difficult to display all of them at once.

Figure 5: Random 200 SIFT Descriptors for the 0^{th} image in fishbowl image set

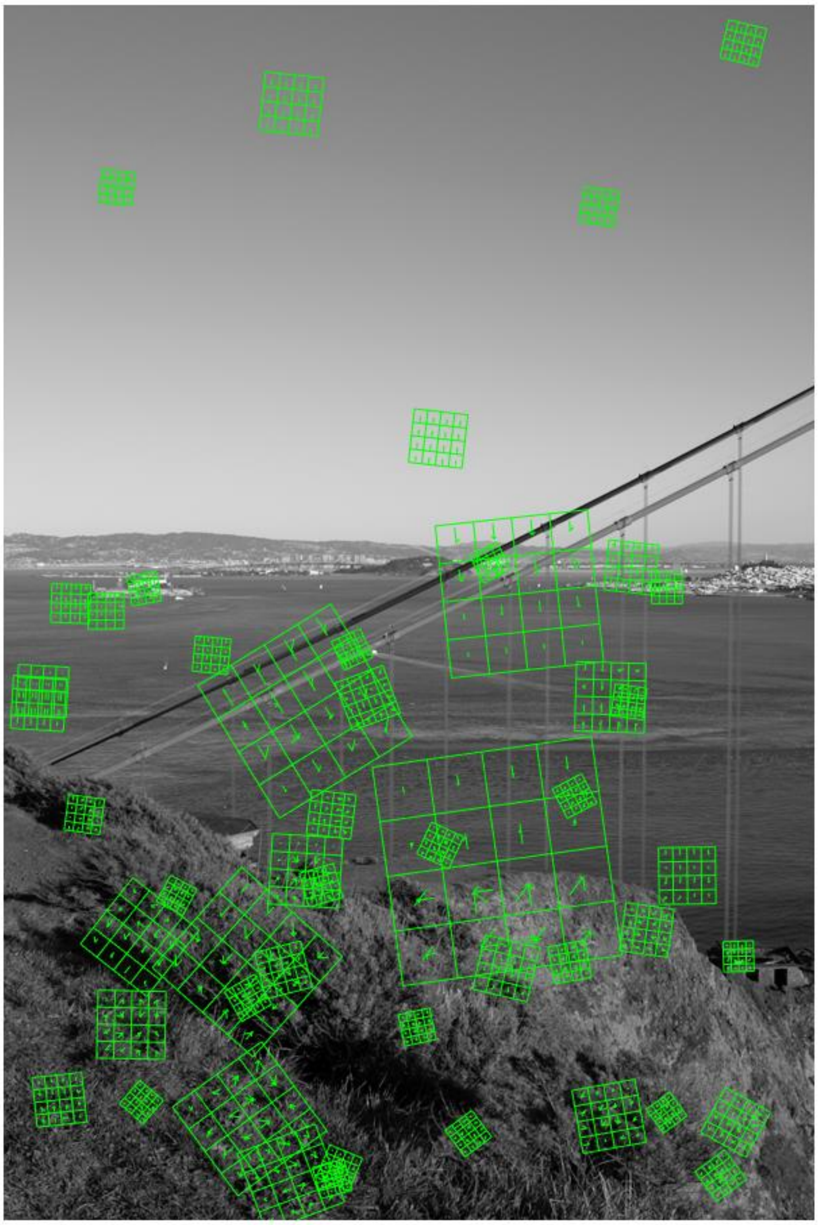

Figure 6: Random 200 SIFT Descriptors for the 0^{th} image in fishbowl image set

ii. Raw-Pixel Based Descriptor

Raw-pixel based descriptor was implemented as follows.

Consider a keypoint k_1 in I_1

Form a square binary mask, located at k_1,
where the distance between sides and k_1 is the scale of the keypoint.

Rotate this square counter − clockwise, amount of orientation of k_1

Compute the histogram of I with respect to the mask

Below images show example rotated square window for some key point:

Figure 7 : Example key point selected from one of the images. Scale is maginified, the angle was calculated to be apx. -168 degrees.

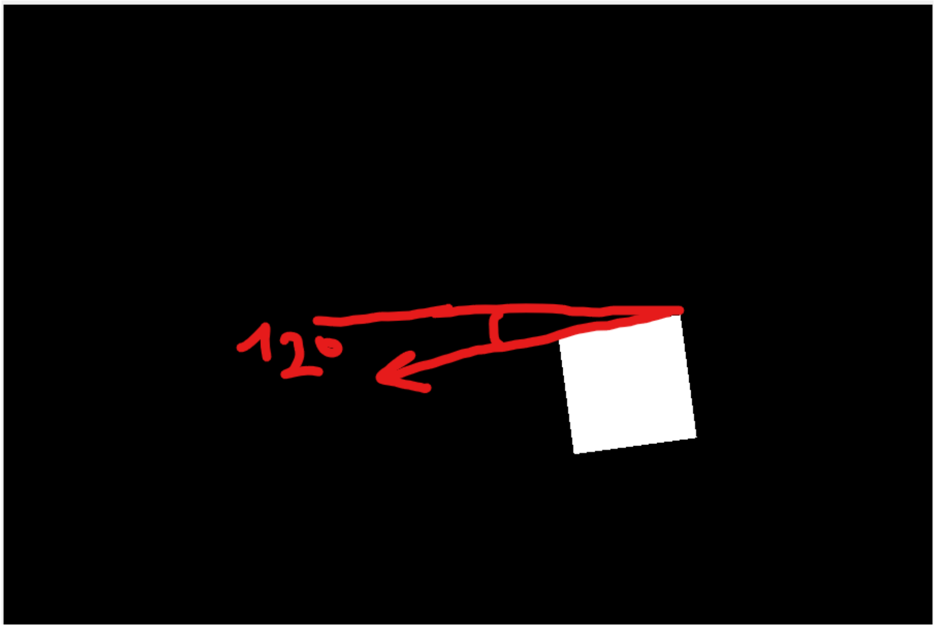

Figure 8 : The rotated square window of the key point in Figure 7. The windows was rotated successfully, as the small angle it is making with horizontal axis is apx. 12 degrees.

Below are randomly selected 5 histograms from the selected key points shown below.

Figure 9: Randomly selected 5 key points, whose descriptors were plotted in Figure 10.

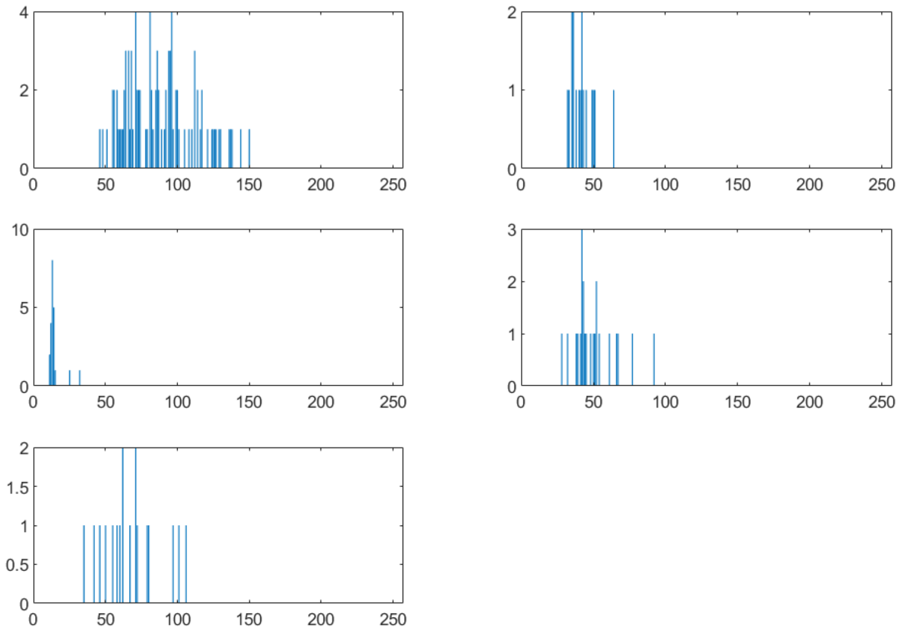

Figure 10 : Raw pixel based descriptors of the key points shown in Figure 9.

3. Feature Matching

For the feature matching, Best-Bin-First Search algorithm, developed by David Lowe, was used, which is explained below [1]. For computing the distance, Euclidean distance was used.

The images below show an example of the features matched from the first two images of both image sets. For this step, finding a matching keypoint k_2 for a key point k_1 was done by finding the key point k_2 that is:

- closest to k_1 (nearest neighbour (nn) of k_1 in the second image)
- its distance between k_1 is smaller than a threshold T, where T is defined by:
  - T = The distance between k_1 and its second nearest neighbour X Threshold ratio R

Instead of finding a **constant** optimal distance T, the T distance was tuned **dynamically**, with respect to the second nearest neighbour, as the distance between key points may vary in different images. The similarity ratio was tuned as R = 2 / 3. Hence, feature matching can be defined as:

$\text{Consider a keypoint } \mathrm{k}_1 \text{ in } \mathrm{I}_1$

$\mathrm{nn\_dist} = \min_{\forall \mathrm{k}_2 \in \mathrm{I}_2}\{\mathrm{eucl\_dist}(\mathrm{k}_1, \mathrm{k}_2)\}, \text{where}$

$\mathrm{nn} = \{\mathrm{k}_2 \mid \mathrm{eucl\_dist}(\mathrm{k}_1, \mathrm{k}_2) = \mathrm{nn\_dist}, \exists \mathrm{k}_2 \in \mathrm{I}_2\}$

$\mathrm{second\_nn\_dist} = \min_{\forall \mathrm{k}_2 \in \mathrm{I}_2 - \{\mathrm{nn}\}}\{\mathrm{eucl\_dist}(\mathrm{k}_1, \mathrm{k}_2)\}$

$\mathrm{second\_nn} = \{\mathrm{k}_2 \mid \mathrm{eucl\_dist}(\mathrm{k}_1, \mathrm{k}_2) = \mathrm{second\_nn\_dist}, \exists \mathrm{k}_2 \in \mathrm{I}_2 - \{\mathrm{nn}\}\}$

$\mathrm{T} = \mathrm{second\_nn\_dist} * \mathrm{R}$

$\text{if } \mathrm{nn\_dist} < \mathrm{T} \text{ and second\_nn exists, then}$

$\quad \mathrm{k}_1 \text{ and } \mathrm{k}_2 \text{ is a match}$

a) Using SIFT Descriptors

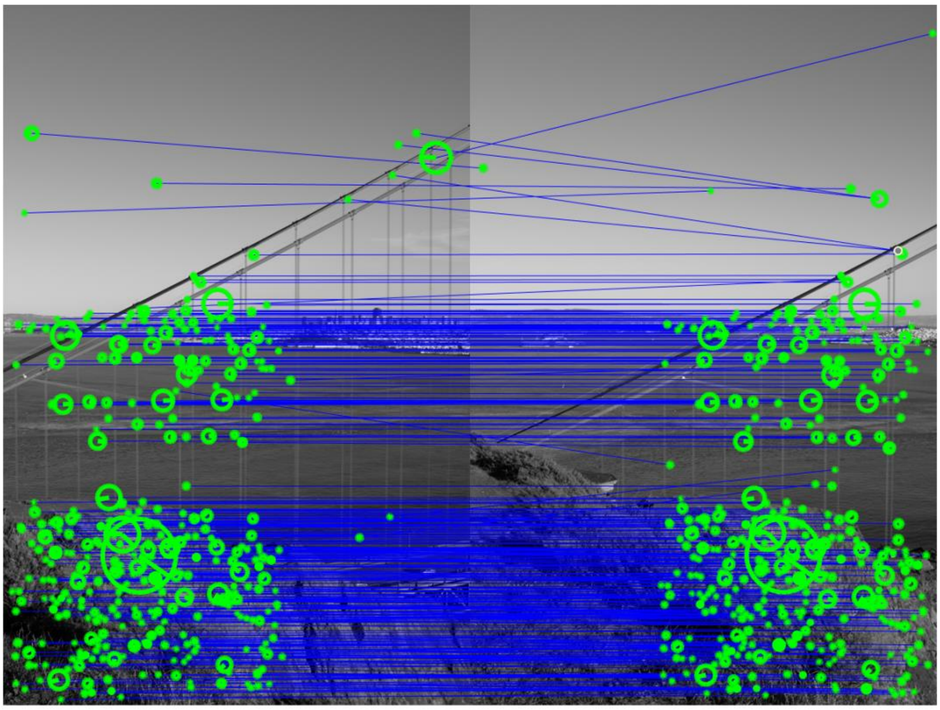

Figure 11: Golden Gate - matched features between 0^{th} (right) image and 1^{st} (left) using SIFT descriptors

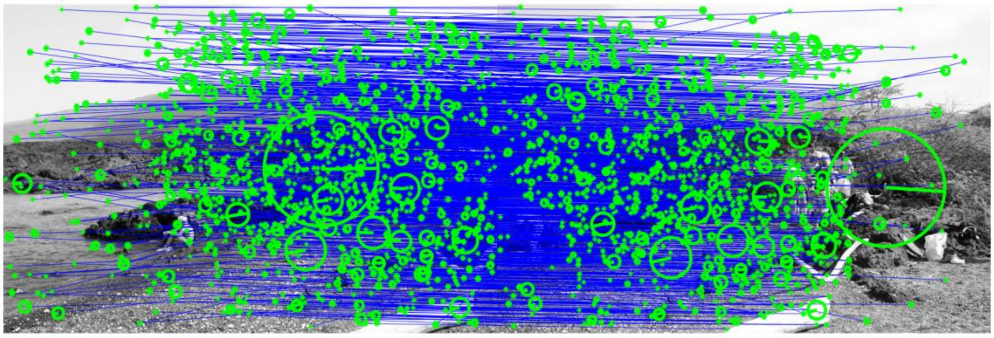

Figure 12: Fishbowl - matched features between 0^{th} (right) image and 1^{st} (left) using SIFT descriptors

b) Using Raw-Pixel Based Descriptors

Below are the matched features for 0^{th} and 1^{st} images of two image sets. It should be noted that, with the raw-pixel based descriptors, we have even less number of matched points.

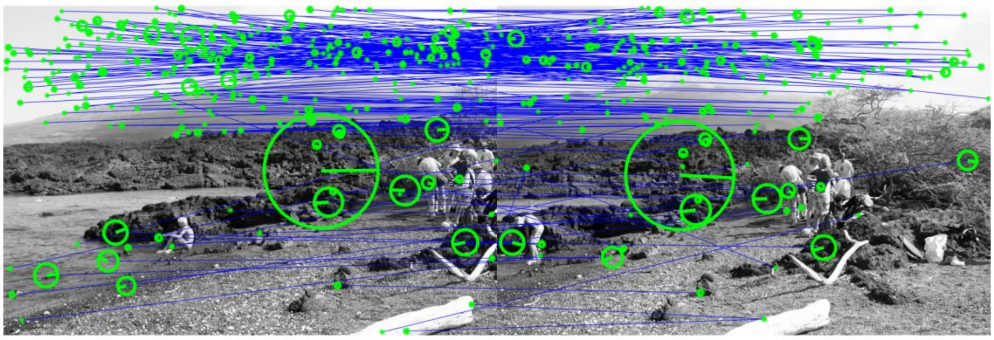

Figure 13: Fishbowl - matched features between 0th (right) image and 1st (left) using Raw-pixel based descriptors

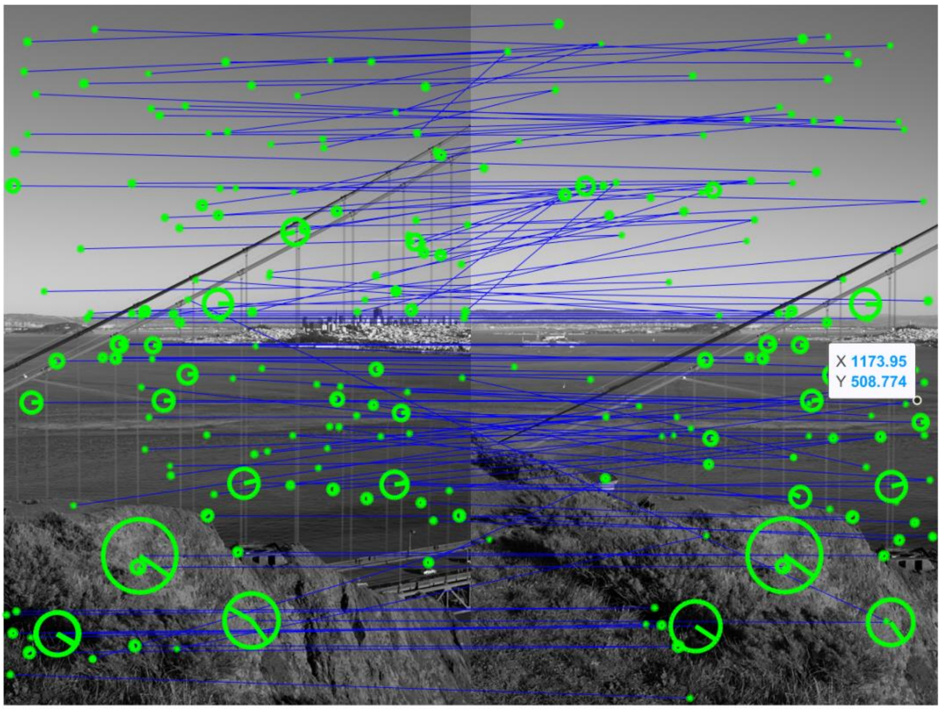

Figure 14: Golden Gate - matched features between 0th (right) image and 1st (left) using Raw-pixel based descriptors

There are several reasons for this but the fact that the **histogram has no direct information about the orientation of the interest point, and the fact that histogram of a window is not rotation invariant are the main reasons**.

4. Image Registration

The source [2], which explains image stitching using projective transformations and SURF descriptors, were helpful in the implementation of this step.

a) RANSAC Parameter Tuning

For RANSAC algorithm, `estimateGeometricTransformation2D()` method was used to estimate an affine transformation between two pairs, whose features were matched [3]. The parameters were tuned as following:

`MaxNumTrials = 50 000` : This parameter was set as high as possible to give freedom for tuning other parameters. However, it should not be too large, like 1 million, so that computation of each panorama does not take more than 1 minute.

`Confidence = 99.99`: Increasing this parameter along with `MaxDistance` was helpful overall. In fact, increasing the confidence interval will lead in more tight bounds for the expected transformations matrices.

`MaxDistance = 0.1`: This parameter had default value of `1.5`. However, with the default values, the estimated transformations had mismatches, even for objects that were very far away, in which affine transformations perform well. Therefore, more features needed to be marked as outliers, so that RANSAC finds a transformation that will result in a rotation, shear, and scaling, instead of straight translations.

While the raw-pixel based descriptor was used, those parameters were tuned such that if estimation fails, the parameters were flexed exponentially:

```
elseif (descriptor_choice == "raw")
    conf=80;
    max_dist=2.5;
    max_num_trials=1000000;
    status = -1;
    while status ~= 0
        [transform(i), ~, status] = estimateGeometricTransform2D(...
        transpose(match_locations), transpose(match_locations_prev), ...
        'affine', MaxNumTrials=max_num_trials, ...
        Confidence=conf, MaxDistance=max_dist);

        if (status ~= 0)
            warning("Could not find enough inliers, now re-adjusting " ...
                + " parameters to find enough inliers.");
            max_dist = max_dist * 2;
            conf = (conf / 3) * 2;
            max_num_trials = max_num_trials * 3;
        end
    end
end
```

Figure 15: Source code showing how the parameters were tuned for raw-pixel based descriptors

b) Estimating the Image in the Center

However, raw image registration was not enough to get good results. Since the transformations were starting from a point and continued with a straight rotation, estimated transformations resulted in spiral images, especially in the Fishbowl image set. In the source given, this problem was solved by estimating the center image, and then multiplying inverse of the transformation matrix of the center image with all transformation matrices.

In the **example (not final)** outputs below (Figure 9 and 10), we can see the significance of this step in the fishbowl image:

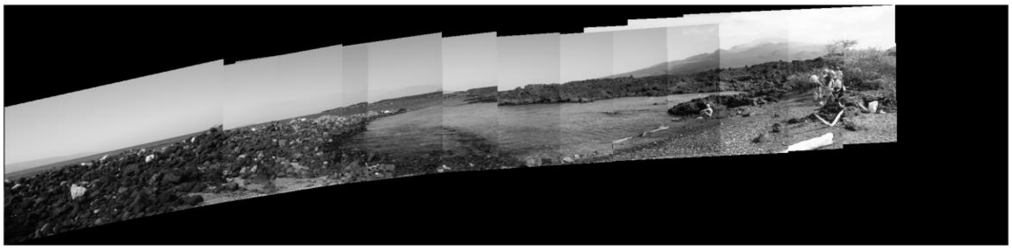

Figure 16: Without estimating the center and applying inverse transform, fishbowl

Figure 17: Without estimating the center, and applying inverse transform, golden gate

c) Outputs - Gradient Based Descriptor

SIFT descriptors work well, and we can observe that matches generated by its descriptor work very well. The only remaining problem with these is the transformation being used, and motion of objects (to be more specific, the human sitting on the lake on 1^{st} image vs beginning to sit in 0^{th} image) along the image set. The second problem is not observed quite well, as the motion is not captured in the overlapping regions.

Figure 18: Output for the image registration step, using gradient based descriptor, fishbowl

Figure 19: Output for the image registration step, using gradient based descriptor, golden gate

d) Outputs - Raw Pixel Based Descriptor

The raw-pixel based descriptor works **surprisingly well** on the golden gate image set. However, it must be noted that when matches coming from raw pixel based descriptors are used, **the parameters must be tuned very carefully**, or the estimated transformations will result in a panorama that has nothing to do with the image.

In fact, with fishbowl image set, the RANSAC algorithm was never observed to estimate a set of transformations where **inverse matrix exists** for the center image, meaning that at the center image's affine transformation was **singular**. This practically means that, if we were not to invert the matrix and **apply the transformations** instead, the **images would disappear**.

Below are the sample outputs using raw-pixel based descriptors.

Figure 20: Output for the image registration step, using raw-pixel based descriptor, golden gate

It should be also noted that Figure 20 is very similar if we only translated the images together in the Euclidean coordinate system, without using affine rotations and shear, contributing the fact that the raw-pixel descriptor is rotation variant.

```
Command Window
Enter file name or full path: "test1.txt"
Descriptor choice, gradient or raw: "raw"
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Could not find enough inliers, now re-adjusting
parameters to find enough inliers.
> In main2 (line 203)
Warning: Matrix is close to singular or badly scaled.
Results may be inaccurate. RCOND =  1.101945e-30.
> In affine2d.get.Tinv (line 385)
In affine2d/invert (line 286)
In main2 (line 235)

Error using affine2d/set.T
The specified transformation matrix is not valid because it
is singular.

Error in main2 (line 237)
fx     transform(i).T = transform(i).T * T_inverse.T;
```

Figure 21: Output for the image registration step, using raw-pixel based descriptor, golden gate - fails completely. Warning messages were generated in the source code given in Figure 15.

From Figure 21, we can infer that the transformation estimated was singular, meaning that RANSAC's parameter was tuned to be too flexible to find a transform, **instead of giving an error about the insufficient number of matching points**. This is caused by **insufficient number of matching points** and/or **wrong**

matches of the raw-pixel based descriptors. This problem may be caused by **the high dimensionality of the descriptor**.

5. Blending

This step is not implemented, therefore there are no outputs showing the images whose overlapping regions are smoothed.

6. Discussion

a) Gradient based descriptors vs. Raw-pixel based descriptors

Raw-pixel based descriptor fails to perform as good as the gradient based descriptor. In addition to the reasons explained in 3.b., this could also stem from the fact that **rotation** of the binary mask **is not unique**, **we can generate the same mask by using different angles**. Therefore, we lose the information about the orientation of the key point, also **making our descriptor orientation variant**.

The fact that it performs well with golden gate image set is **parameter and scene dependent**, and it is **more likely to fail with a different scene**, as the same parameters may not work with those sets, **even if we tune it on runtime**, as shown in Figure 15.

b) Which images are harder to stitch together and why?

Overall, the image stitching process using affine transformations may not perform well **if the scene being captured is close to the camera**, resulting in **large transformations in small movements of the camera**. This can be observed from the outputs in 4.c.

In Figure 17 and 19, for instance, the golden gate output has some errors on the left, which are closer to the camera. In Figure 16 and 18, for instance, the output has more errors than golden gate overall, because **the fishbowl scene is closer** to the camera **compared to golden gate**. Therefore, the set of images **capturing a closer scene will be harder to stitch together** due to the limitations of the **affine** transformations.

In order to improve this, **projective** transformations can be used to improve the performance of the image stitching process. It should also be noted that projective transformations are very expensive and may require GPU optimizations and parallel hardware.

7. Software

The file image_stitcher.m will be used to generate the final output that have been implemented so far. The documentation is as follows:

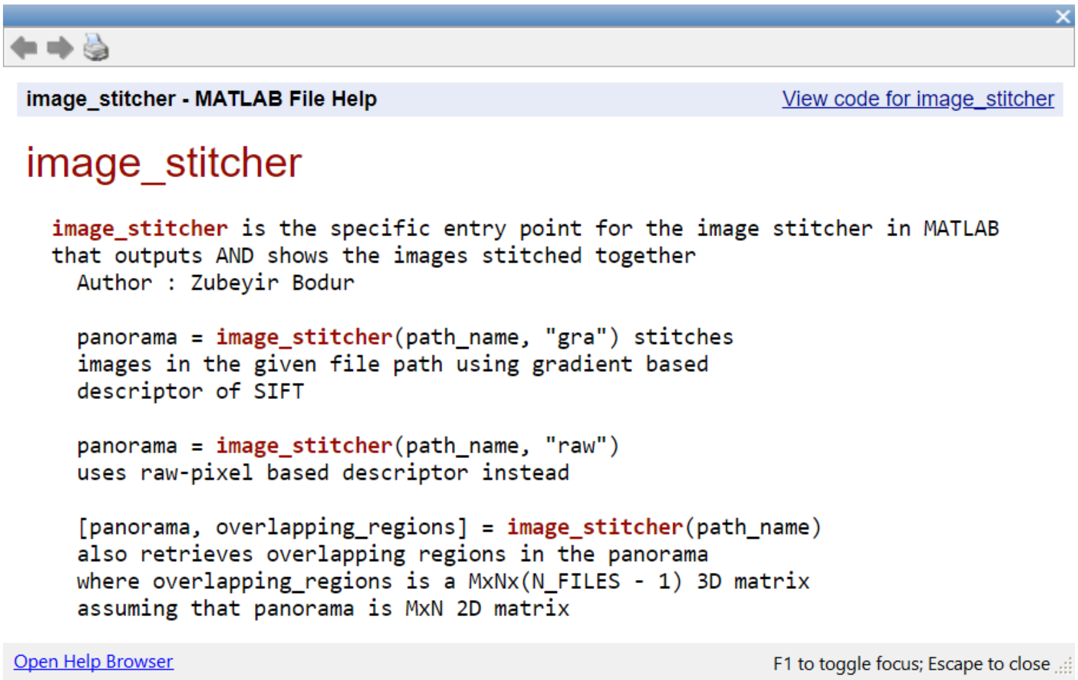

Figure 22: Documentation for image_stitcher.m

8. References

[1] J. Beis, D. G Lowe, "Shape indexing using approximate nearest-neighbour search in high-dimensional spaces," *Conference on Computer Vision and Pattern Recognition*, pp. 1000–1006. doi:10.1109/CVPR.1997.609451. [Online].

[2] "Feature Based Panoramic Image Stitching," *MathWorks*. [Online]. Available: https://www.mathworks.com/help/vision/ug/feature-based-panoramic-image-stitching.html. [Accessed: Apr. 4, 2022].

[3] "estimateGeometricTransform2D," *MathWorks.* [Online]. Available: https://www.mathworks.com/help/vision/ref/estimategeometrictransform2d.html. [Accessed: Apr. 4, 2022].