ASSIGNMENT3: MULTI-VIEW COMPUTER VISION

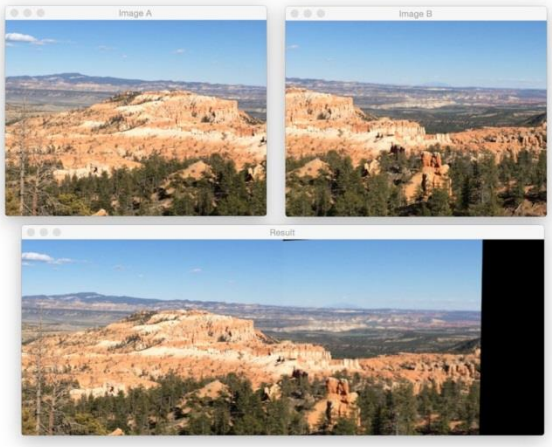
TUTOR: STELIOS ASTERIADIS

TEACHING ASSISTANT: ESAM GHALEB

Due: Sunday, June 10th at 23:59

PART A: IMAGE STITCHING

In this assignment you will develop your own panorama application by stitching together pairs of images, like in the example below. Use your own pairs of pictures (show analytically reported results on one image pair but also visualize how the algorithm works on 2 or 3 more). Your images should be 'parallel' to each other so to avoid perspective distortions.



1. Use feature points that you can detect using the ***harris.m*** code provided along with this assignment. Describe every key-point by extracting fixed-size patches around it (try different sizes and report a short sensitivity analysis of the final result). You will be helped if you turn your descriptor from 2D to 1D vectors.
2. **Optionally**, compute SIFT descriptors on the resulted key-points from Harris corner detector. You can utilize Matlab VLFeat library by using the custom frames (key-points). For this purpose, check vl_sift.
3. Compute the distances between every descriptor in image 1 with every descriptor in image 2. For this, use: a) Normalized correlation and b) Euclidean distance after normalizing each descriptor (zero mean and unit std)

4. Select the best matches based on a threshold or by considering the top few hundred pairs of descriptors. Also make a sensitivity analysis based on these parameters.
5. Make a simple implementation of RANSAC to estimate an Affine transformation mapping one image onto the other. For this, you need 3-5 matches points to initialize the estimation of the transformation. Use number of inliers and – reasonable – number of iterations and inlier thresholds. Report the number of inliers and outliers and the average residual for the inliers (the squared distance between the point coordinates in one image, and the transformed coordinates of the matching points in the other image). Display the location of the inliers matches on both images.
6. Warp image 2 onto image 1 using the best estimated transformation with the highest number of inliers. You will need to learn about maketform and imtransform functions). Check this naïve implementation for image wrapping using maketform and imtransform.
7. Define as a score of accuracy the Euclidean distance between chosen key-points coordinates in image 1 and the corresponding transformed ones in image 2. Base every sensitivity analysis (see steps above) based on this score and plot the results for every parameter you use.

BONUS POINTS:

- Experiment with really difficult cases like *look into the past* image stitching like here (click on the images to see the missing ones), where part of the photo is a modern aspect of a city/landscape and part of it comes from the past. You can do the same with images showing a location at different times of the year, etc.

IMPLEMENTATION TIPS:

- For RANSAC, a very simple implementation is sufficient. Use three to five matches to initialize the Affine transformation in each iteration. You should output a single transformation that gets the most inliers in the course of all the iterations. For the various RANSAC parameters (number of iterations, inlier threshold), play around with a few "reasonable" values and pick the ones that work best. For randomly sampling matches, you can use the ***randperm*** or **randsample** functions.
- In MATLAB, the solution to a nonhomogeneous linear least squares system $\boldsymbol{AX=B}$ is given by $\boldsymbol{X = A \backslash B}$;
- For more details, about feature extraction, matching, image fitting and aligning, and different transformations, please be referred to the following lectures in Eleum: **feature detection matching master** and **Hough-Ransac-ICP-Transformations_master.**

DELIVERABLES (0.1):

1. *A report of 500-1000 words with plots and figures. Do not forget to display everything (key-points used in every step, etc.). Provide the aforementioned information in the implementation steps. For example, describe your implementation, and the interesting parameters for RANSAC, Affine Transformation, feature description and matching.*
2. *Well documented code, with a function output_image=ImStitch(image1, image2, [parameters]) and the related functions . Your code will be tested with **five** image pairs of ours.*
3. *Upload everything on Eleum*

PART B: EPIPOLAR GEOMETRY AND 3D RECONSTRUCTION




The scope of part b of this assignment is to estimate fundamental matrix of two views of the same location and, moreover, develop a 3D reconstruction routine. For this, you will apply your code on two different pairs of images, namely: house1.jpg-house2.jpg and library1.jpg and library2.jpg. Each image comes with the corresponding camera matrices while, for each pair of images, a set of matching points has been extracted (all files can be found on Eleum).

1. The first task for this assignment is to fit the fundamental matrix for each pair of images as well as to report the residual (or mean squared distance) in pixels between points in both images and the corresponding epipolar lines. Part of the deliverable is the visualization of the epipolar lines.

2. Use the camera matrices and the point matches for the triangulation, use the algebraic method to triangulate the position of every pair of matching points and given two cameras. Display the matched points in 3D, along with the camera centers, from different points of view. Use equal scaling for all axis in your plots (you can use plot3 in MATLAB)

BONUS POINTS (0.1):

- Propose and implement a method that uses the match generation technique of **part a** to find key-point correspondences, taking advantage of the epipolar constraint.

- **Hint:** with more matching points, you can do a better estimation for **Fundamental Matrix** using RANSAC algorithm. In addition, you can have denser and better 3D reconstruction.

IMPLEMENTATION TIPS:

- The camera centers are given by the null spaces of the matrices. They can be found by taking the SVD of the camera matrix and taking the last column of V.
- Use the given sample_code.m, to visualize the matched points, and to complete the implementation.
- **Fundamental Matrix Song ☺**
  https://www.youtube.com/watch?v=DgGV3l82NTk

DELIVERABLES:

1. *A report of 500-1000 words with plots and figures. Do not forget to display everything in images (e.g. epipolar lines).*
2. *Well documented code, with the following functions:*
   a. *[fund_matrix, residual]=FMatrix(matchpoints1,matchpoints2, [possible parameters]) .*
   b. *3d_recon=3D_Recon(matchpoints1, matchpoints2, Camera1, Camera2, Image1, Image2)*
3. *Upload everything on Eleum*