**HW #3 Due Feb 23rd 10PM EST**

In this project we will ramp up our understanding of image registration by working on a low contrast, somewhat challenging dataset.

The paper describing the work for photomosaicking is attached Pizarro2003joe.pdf. We want to read this paper and see if we can duplicate some of the work in it while using some of the techniques we have learned in the class so far.

***Datasets*** - the Skerki dataset of images is available in the folder tape018. If you want to know more about this Roman shipwreck here are two references

R. D. Ballard, "High-Tech Search for Roman Shipwrecks," National Geographic (April 1998) pp. 32-41.

R. D. Ballard, A. M. McCann et al., "The discovery of ancient history in the deep sea using advanced deep submergence technology," Deep Sea Research I,47,9, 2000, pp. 1591-1620.

I have identified 6 initial images to work with as well as the set of 29 images that were used for illustrating the paper and put the images in appropriately named folders.

***Part 1*** Modify your registration module from the previous home works to allow you to robustly match features across two images. Please continue to work with Python Notebooks as much as possible.

In order to make your matching more robust, please normalize the images before you match them.

Also we will use the Levenberg Marquadt algorithm for computing the homography on the inlier set that we obtain after feature matching with your favorite feature/descriptor in combination with outlier rejection using RANSAC

Now run your registration code to match

1. the 6 images in a loop
2. images that are non sequential and might have much smaller overlap, for example image pairs 1-6, 2-5, 2-4, 3-5, 1-5, 2-6.

***Part 2***. Go around the entire loop of 6 images and set up the problem by outputting vertices and edges in terms of a factor graph. Images that you manage to register in part (b) above can be thought of as loop closure events.

Justify your choice of the covariance (and thus information) matrix that you use for each edge. Once you have completed the loop run a GTSAM script on the output vertices and edges and see how that

(a) Affects the location of individual image centers - your "odometry" (make a plot before and after optimization)

(b) Plot the covariances for each measurement before and after optimization

***Part 3.*** Now go to the set of 29 images and repeat everything you did in Part 2 above for this larger dataset.

***Part 4 (optional!)*** Do it for the entire dataset.