Computer Vision
Final Project

This assignment combines all the previous steps that students have implemented and add more steps to finalize the project. This is the last guideline regarding your project. *Model Castle* and *Teddy-bear* images are used for the project which have been provided in the Brightspace. We provide the feature point descriptors on Brightspace for everyone.

You are asked to report results on both the *'Teddy-bear'* images used throughout the extent of the lab, and the *'Castle'* images. For every of the subsequent topics you are asked to visualize intermediate results in both of these datasets, side-by-side. Also, please upload images from multiple views (not just one view).

1 Components of the Project and Grading

1. **(.5 + 1* pt)** Use relevant tools (e.g. vl_feat: http://www.vlfeat.org) to find interest points and correspondence (matches) between consecutive images.
   Note: You can get 1 bonus point if you submit a working implementation of your own for Harris corner detection and feature matching. (Lab assignment 2).

2. **(2.5 pt)** Apply normalized 8-point RANSAC algorithm to find best matches. (Lab assignment 3+5)

3. **(2.5 pt) Chaining:** Create point-view matrix to represent point correspondences for different camera views (Lab assignment 6).

4. **(3.5 pt) Stiching:**
   (a) **(1 pt)** Create the measurements matrix from the point-view matrix and take blocks of the point-view matrix that are composed of three and four consecutive images.
   (b) **(1.5 pt)** Estimate 3D coordinates of each block using Tomasi-Kanade factorization (Lab assignment 4).
   (c) **(1 pt)** By an iterative manner, stitch each 3D point set to the main view using the point correspondences i.e., finding optimal transformation between shared points in your 3D point clouds. Transformation between different sets can be found using Procrustes analysis. The MATLAB `procrustes` function can be used in the project.
5. **(1 pt)** Eliminate affine ambiguity (consult your lecture slides on affine structure from motion).
6. **(.5* pt) 3D Surface rendering:** This task is a bonus task. When you have the 3D point cloud of the castle, use the built-in `surf` function for the 3D surface plot. Then include RGB (texture) colour of the related points on surf visualization (interpolate colour values for the filled areas on the surface using the known points).

The final report for the project (in pdf format) together with the code must be submitted via Brightspace.

The report needs to be within 7 pages and explain and motivate your choices and results clearly.

- Organize the document into subsections, each focusing on one of the above list of topics.
- Explain as clearly as possible every step you have taken in solving each task, listed above.
- To exemplify your solution for each task, use in the report visualizations/plots of intermediate results, obtained from your own code.

- For the visualizations, show examples from both datasets: ‘Teddy-bear’ and ‘Castle’ images, side-by-side, and use more than one view to exemplify your results.
- For a writing guideline please have a look at: http://jvgemert.github.io/links.html.

The code must have one main script that calls different functions to perform the 3D reconstruction and should use relative paths and be ready to run. The code should be well modularized into functions, where ideally each function implements one of the modules that is described in this guideline.