CS/ECE 281B Spring 2016

Homework Assignment #6

DUE: 5:00pm, Sunday May 8th (Electronic turnin required)

You are to implement a multi-view, sparse 3D reconstruction algorithm. Your program uses multiple photos taken by the same camera, with the camera executing a general motion orbiting an object of interest. Again, a rough estimate of the camera's intrinsic parameters will be provided with the test data sets. So you can attempt a similarity 3D reconstruction.

Sample test images can be found in http://www.cs.ucsb.edu/~cs281b/testimages/prog6/ (or follow the local image archive link from the class web page). Make sure that your programs work on images in that directory. You can also try your program on your own photos.

Your Matlab code should accept a directory name that contains the test images, and output a single file containing the reconstructed 3D point cloud, again, in the format of x y z r g b. Adding the proper ply header will allow you to view the 3D point cloud using most 3D software.

As discussed in the lecture, general multi-view 3D reconstruction is challenging and beyond the scope of a programming assignment (but can be extended into a good project). Instead, you should attempt to perform registration of point clouds in 3D. Basically, the previous assignment gives you the ability to recover visible 3D structures from 2 views. If you have multiple such 2-view structures available, and some features are visible in multiple structures (i.e., the 2D correspondences and 3D coordinates of these features are known), they can then be used to fuse the 2-view 3D structures. As these 2-view 3D structures differ by T, R, and S (if the reconstruction is a similarity reconstruction), enough feature correspondences in 3D will allow you to solve for the similarity transform as follow to register them:

$$P' = s\begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} P$$

It is in fact not necessary to recover individual transforms T, R, and s. It is only necessary to note that the combined transform is represented by a 4x4 matrix with 13 parameters. Each 3D point shared by 2 2-view structures provides 3 equations. Theoretically, you can solve for the combined transform with more than 4 correspondences (certainly, a RANSAC-type procedure is preferred for improved robustness). However, do note that the matrix you recover this way may not be decomposable as above into a rotation, translation and scale, so to obtain good results you might have to perform nonlinear optimization on the linear estimates.