

EN 161: Image Understanding

Assignment 11: Stereo Reconstruction

Due Date: Wednesday, December 14, 2005

Notes:

Here are a couple of important things you should keep in mind when submitting your solutions for the next MATLAB assignment.

- You **must** send your source code for each function or MATLAB script you used in your solution and prepare a lab report including your output images and results.
- The reports should include: important technical details to run your applications and explanation of your algorithm including some specific parts of your code.
- You can download the dataset for this assignment from the course website:

http://www.lems.brown.edu/~ieden/en161/lab11_files/en161-assignment11-dataset.zip

Stereo Reconstruction

In this assignment, you will implement a **simplified** stereo reconstruction system by following the steps described below.

- Download the dataset file for this assignment, and unzip it to a folder. Inside the folder, you will see 3 files: “**StereoPair1.mat**”, “**StereoPair2.mat**” and “**StereoPair3.mat**”. These are binary Mat-files that includes several MATLAB variables. You can load these variables into your workspace by using MATLAB's “**load**” command.
- After loading one of the files, you will see several variables in your workspace. These variables are:

im_Left	Left image (Grayscale)
im_Right	Right image (Grayscale)
pExt_Left	External Camera Matrix for left image
pExt_Right	External Camera Matrix for right image
pInt_Left	Internal Camera Matrix for left image
pInt_Right	Internal Camera Matrix for right image
p_Left	Camera Proj. Matrix for left image
p_Right	Camera Proj. Matrix for right image
fMatrix	Fundamental Matrix for this Stereo Pair

Your main task is to write MATLAB program to select a few **matching** points on two images, and reconstruct the corresponding 3-D points by using the triangulation algorithm. You can specify the number of points that you want to reconstruct.

Here is the *pseudo code* of the algorithm that you will implement:

- load one of the Mat-files into the workspace
- for each point $p_i=(x, y, z)^T$ you want to reconstruct
 - display the left image
 - let the user select a point $p_i^L=(u, v, 1)^T$ on the left image
 - find the corresponding epipolar line on the right image $l_i^R=(a, b, c)^T$ by using the following formula:

$$l_i^R = F \times p_i^L$$
 (Where F is the 3x3 fundamental matrix)
 - create a new image from the right image by writing the epipolar line l_i^R on the right image
 - display this new image to the user (assume the user will select the matching point)
 - let the user select a point $p_i^R=(u, v, 1)^T$ on the right image
 - reconstruct p_i by using p_i^L and p_i^R (you will use the triangulation algorithm)
- display $\{p_i\}_1^N$ in a 3-D plot, where N is the number of points you reconstruct

Suggestions and Questions:

- Try to choose points on the images that has a certain geometry (such as points on a straight line, points on the corners of a rectangle, points on the corners of a cube, etc.) Check accuracy of your point reconstructions by seeing how they satisfy the semi-global geometric constraints.
- While choosing matching points on the epipolar line, try 1 or more slightly different locations to see the effect on 3-D estimation error.
- What can you say analytically about the 3-D point estimation error as a function of noise in image point location? Discuss.
- For your final result, try to use number of points in the range of 10-15
- Be careful about the **image coordinate systems!!!**