ETH Zürich
Computer Science Department

Computer Vision

Stereo Matching

Objective:
In this exercise you will implement a winner-takes-all and a graph-cut based stereo algorithm to recover a disparity map from a set of images. Using the results of dense stereo, you will generate a textured 3D model.

5.1 Disparity computation (30%)

Implement winner-takes-all stereo using SAD or SSD. The code framework provides you with functions to automatically rectify a pair of images. The disparity range can be left as constant or it can be estimated manually by clicking points in the images and checking their range in disparity. Try different average filter window sizes (3×3, 5×5, 7×7, etc.) for the disparity computation. Show the disparity map and compare results among sizes.

5.2 Graph-cut (30%)

Computing the disparity can be formulated as a graph labeling problem. Each pixel corresponds to a graph node and each disparity to a label.
The goal is to find a labeling $f : P \rightarrow L$ which minimizes

$$E(f) = E_{data}(f) + \lambda E_{smooth}(f) = \sum_{p \in P} D_p(f_p) + \lambda \sum_{p,q \in \mathcal{N}} S(f_p, f_q). \quad (1)$$

P represents the set of pixels and L represents a discrete set of labels corresponding to different disparities. $D_p(f_p)$ is the cost of assigning label f_p to pixel p and it is given by the SSD value calculated for the disparity corresponding to label f_p. $\mathcal{N}$ is the set of neighboring pixels and $S(f_p, f_q)$ is the cost of assigning labels f_p and f_q to neighboring pixels p and q. The idea is to penalize neighboring pixels having different labels.

- Familiarize yourself with the graphcut code framework. An example demonstrating its usage is given in *gc_example.m*.
- The SSD values needed for E_{data} are provided as a $m \times n \times r$ matrix, where $m \times n$ are the dimensions of the image and r is the number of disparities taken into account.
- Use the same smoothness term as in the sample code.

Include the disparity map obtained from Graph-cut in your report. Compare it with the results obtained from the winner-takes-all algorithm.

5.3 Generating a textured 3D model (40%)

For the image pair included in the framework, the camera parameters used for capturing are provided. Using the result of your dense stereo algorithm and the camera parameters, generate a textured 3D model. For each pixel, find the corresponding 3D point. Remember that the disparity map image coordinates are rectified.

The framework contains code to generate textured .obj-files from 3D coordinates and images. To view your 3D model, you can use Meshlab (`http://meshlab.sourceforge.net/`). Additional images, including camera parameters, can be found at `http://cvlab.epfl.ch/~strecha/multiview/denseMVS.html`.

5.4 Bonus (10%)

Automatically compute the disparity range using the point correspondences for the Graph-Cut version, compare results.

Hand in:

Write a short report explaining the main steps of your implementation and discussing the results of the methods. Make sure to include:

- Images and a comparison of the disparity maps obtained using both implemented methods and the images included in the framework.
- A screenshot of your textured 3D model, again using the provided images.

Upload the report together with your source code to moodle.

For more info contact georgous@vision.ee.ethz.ch.