Robotic 3D Vision
Prof. Dr. Jörg Stückler, Rui Wang
Winter Semester 2017/2018

Computer Vision Group
Department of Informatics
Technical University of Munich

Exercise Sheet 6

Topics: Dense Stereo Reconstruction and Map Representations

Submission deadline: Wednesday, 07.02.2018, 23:59
Hand-in via email to rob3dvis-ws17@vision.in.tum.de

General Notice

All exercises can be done in teams of up to three students. Please hand-in your solution before the submission deadline, indicating names and matriculation numbers of your team members. Teams are encouraged to present their submitted solution during the exercise sessions.

Exercise 6.1: Dense Stereo Reconstruction

In this exercise, you will implement a basic approach for dense stereo reconstruction.

(a) Inspect the code and data provided for this exercise in the materials archive. The data are left and right rectified stereo images from the KITTI dataset, contained in the folders `left` and `right`. The image file names give the number of the images in the sequence. Corresponding left and right images have the same number. The camera intrinsics are specified in the file `K.txt`.

The provided code contains a file `main.m` which implements basic loading of images and camera intrinsics. It also sets up some parameters for the stereo reconstruction. It calls functions `calcDisparity` and `disparity2PointCloud` to determine the disparity between images and visualize a 3D point cloud. Stubs for these functions are also included in the files `calcDisparity.m` and `disparity2PointCloud.m`.

(b) Implement the function `calcDisparity` to find the disparity for each pixel in the left image that minimizes the SAD and SSD patch comparison measures (see lecture). Visualize your disparity estimation result for the first image pair in the sequence as a color image. Hint: use the `pdist2` function. The KITTI images are already stereo rectified such that epipolar lines are horizontal on the same rows between the left and right images.

(c) Filter outliers by rejecting ambiguous matches. To this end, reset all disparities to zero where there are at least two second best disparity candidates with a matching cost which is similar by a factor of 1.5. Why at least two similar disparities? Also reset disparities to zero if the corresponding pixel is found at the border of the right image. Why?

(d) Implement the `disparity2PointCloud` function. For this, you will need the camera intrinsics and the baseline (the latter is defined in `main.m`). Visualize your result for the first image pair in the sequence.

(e) Generate a video of the disparity images for the whole image sequence.

Exercise 5.2: Map Representation with Signed Distance Function and Occupancy Maps

In this exercise, you will get familiar with volumetric mapping using signed distance function (SDF) and occupancy representations. Consider the following 2D mapping example: a robot measures depth with a 1D camera with a pinhole projection model $z = x/y$, where $\mathbf{y} = (x, y)$ is the measured 2D point in the coordinate frame of the camera and z is its projected position in the 1D image. The robot obtains 20 successive image measurements which are contained in the file `data/map_measurements.txt` included in the exercise materials. Each image is stored in a row and contains the x-coordinates of the measured points for image coordinate range $z = [-0.2 : 0.01 : 0.2]$.

(a) Determine a projective truncated SDF from the image measurements according to the update rules from the lecture. Discretize the SDF in a 2D grid centered at the robot with resolution 0.1 m. Use the following weighting function for the updates

$$w(d) = \begin{cases} 1 & \text{, if } d < \eta_1 \\ 1 - \frac{d-\eta_1}{\eta_2} & \text{, if } \eta_1 < d < \eta_2 \\ 0 & \text{otherwise} \end{cases}, \tag{1}$$

where $\eta_1 = 0.2$ and $\eta_2 = 0.1$ are thresholds that defines an update band around the measured surface and d is the Euclidean distance of the grid cell from the measurement. Visualize the resulting 2D map using matlab.

(b) Find the intersection with the zero-level set for the ray through the tenth pixel in the first image measurement $y_{1,10}$ in the SDF grid.

Hints: First find the cells between which the first zero-crossing occurs along the ray. Then interpolate the signed distance function along the ray from the nearest cells in the grid and determine the position of the zero-level set on the ray with sub-cell-discretization accuracy. Only use cells with an accumulated weight larger zero.

(c) Use the following inverse sensor model to incorporate the measurements into an occupancy grid map with the same resolution like the SDF grid,

$$p(m \mid y_t) = \begin{cases} p_{free} & \text{, if } d(m) < y_t - r/2 \\ p_{occ} & \text{, if } y_t - r/2 < d(m) < y_t + r/2 \\ p_{prior} & \text{, if } d(m) > y_t + r/2 \end{cases}, \tag{2}$$

where $r = 0.05$ is a radius around the measured surface and $d(m)$ is the distance of the cell m from the robot. Use the following settings: $p_{free} = 0.3$, $p_{occ} = 0.7$, $p_{prior} = 0.5$. Initialize the map with a prior occupancy probability of 0.5. Use log-odds updates to accumulate the occupancy probabilities in the grid cells. Determine the distance to the first occupied cell in the map for the measurement $y_{1,10}$. Visualize these distances and the resulting 2D map using matlab.

(d) Derive the occupancy probability of a voxel $p(m = occ \mid y_{1:t})$ from its log-odds belief $l(m = occ \mid y_{1:t})$.

(e) Why is the inverse sensor model $p(m \mid y_t)$ used instead of the forward model $p(y_t \mid m)$ in occupancy mapping? Give an interpretation of the two models in your own words.

Submission instructions

A complete submission consists both of a PDF file with the solutions/answers to the questions on the exercise sheet and a ZIP file containing the source code that you used to solve the given problems. Note all names and matriculation numbers of your team members in the PDF file. Make sure that your ZIP file contains all files necessary to compile and run your code, but it should not contain any build files or binaries. Please submit your solution via email to `rob3dvis-ws17@vision.in.tum.de`.