ECE661: Homework 8

Fall 2018

Deadline : November 8, 2018, 1:30 pm

Turn in your solution via Blackboard. Additional instructions can be found at [I]

1 Introduction

The goal in this homework is to implement the popular Zhang's algorithm for camera calibration. A format description of the algorithm can be found in the Zhang's technical report available at [I].

For this assignment, you can assume the camera to be a pin-hole camera. This implies that a complete calibration procedure will involve estimating all the 5 intrinsic parameters and the 6 extrinsic parameters that determine the position and orientation of the camera with respect to a reference world coordinate system. This requires you to establish correspondences between image points and their world coordinates. To this end, you can download and use the checker-board pattern from [I]. The checker-board pattern consists of alternating black and white squares. We will be using the corners of these squares in our calibration procedure.

2 Tasks

You will use two datasets for this homework. One dataset consisting of 40 images of the calibration pattern taken from varying viewpoints and orientations can be downloaded from [I]. You will create the second dataset on your own. Your assignment consists of the following steps:

2.1 Creating the dataset

- Print out the calibration pattern that is provided at [I] and mount it on a wall or a large piece of cardboard. Now choose one of the corners on the pattern as your world origin and measure the world coordinates of all the other corner points on the pattern with a ruler. Number the corners appropriately. **A particular corner should get the same number label in all the images**. (This does not mean that you need to write any numbers on the actual calibration pattern).
- For the very first image of the calibration pattern on the wall, position the camera such that its Principal Axis is approximately perpendicular to the plane of the wall on which you mounted the calibration pattern. Also make sure that the x-axis of the image is very roughly along the horizontal axis of the calibration pattern and the y-axis of the image very roughly along the vertical axis of the pattern. These conditions are meant to be satisfied only very, very approximately. Despite the approximations involved, the distance between the camera and calibration pattern that you can measure manually will serve as a check on your calculations of the calibration parameters. In the following discussion, this image will be referred to as 'Fixed Image'.
- Now move your camera in different directions and capture images of the calibration pattern. Obviously you will need to rotate/tilt the camera in order to capture the calibration pattern from different positions. A minimum of 20 different poses of the camera is required for good camera calibration.

2.2 Zhang's Algorithm

For each of the datasets, implement the following:-

2.2.1 Detecting corners

- Extract edges using the Canny edge detector. You can use any open-source implementation of Canny edge detector like cvCanny function from OpenCV.

- Fit straight lines to the edges using the Hough transform. You can again use any open-source implementation of Hough lines transform like the cvHoughLines or the more efficient HoughLinesP functions from OpenCV.
- The corners will be the intersection points of these lines.
- Depending on the accuracy of your corner detection, you might wish to improve your results. You can refer to the previous year solutions for improving this accuracy. However do note that it is not necessary to detect 100 % of the true corners in every image. Since you will be using the Levenberg Marquadt non-linear optimization to refine your calibration, you should have robust calibration as long as you detect sufficiently high number of corners with good accuracy in each image.
- Assign labels to the corners using the same numbering scheme that you used in the previous section. These labels should be indicated on every output image in your report.

Proceed further only when you are sure that your corner detection algorithm is working correctly.

2.2.2 Calibration

- You need to establish correspondences between the extracted corners in each image and their world coordinates.
- Implement Zhang's calibration algorithm.
- Use the Levenberg Marquadt algorithm for non-linear optimization. The Levmar package [II] is a very good resource for this. It can be used with C++, Matlab and Python.
- To measure the accuracy of your camera-calibration, reproject the corner points from 4 or more views back into the 'Fixed Image'. You can do a visual comparison of the locations of the original corners vis-a-vis the reprojected corner points. In each of the (4 or more) images, measure the error for each point using the Euclidean Distance measure. Calculate an estimate of the mean and variance of this error.
- Show a minimum of 2 images where one can see the improvement of your calibration estimate by using the LM optimization.
- Compare your estimated camera pose for the 'Fixed Image' with the measured ground-truth.
- For extra credit:
  1. Estimate the radial distortion parameters. You can refer to Section 3.3 of Zhang's report.
  2. Quantitatively demonstrate good reduction in error by using the LM optimization, on at least 2 images.

3 Submission

Show results on both datasets. You can download the first dataset from [I].

1. Turn in a typed pdf of your report via Blackboard.
2. Your pdf must include the following : -
   - A good description how you implemented each of the tasks above, with relevant equations.
   - For each dataset, a minimum of 2 output images for the edge-detection, hough-line fitting and corner detection steps. Clearly label the detected corners.
   - For each dataset, a minimum of 3 images (corresponding to 3 different views) clearly showing the reprojected corners and the original corners in the Fixed Image. This is to give a visual measure of the accuracy of your calibration procedure. Label the corners. Use different colours to differentiate the reprojected corners from the original corners.
   - A minimum of 2 images clearly demonstrating the gains obtained using the LM optimization.
   - Your estimate of the intrinsic camera matrix. Also include your estimate of the external camera calibration matrix for atleast 4 images.
   - Your source code.

References

[I] `https://engineering.purdue.edu/RVL/ECE661_2018/`
[II] `http://users.ics.forth.gr/~lourakis/levmar/`