Depth by triangulation

Josef Bigun,
josef.bigun(at)hh.se

March 2011, v1.0

1 Introduction

This lab-exercise is intended as a free educational resource as well as a toolbox for those wishing to have a deeper understanding of stereo algorithms, in particular the triangulation method providing depth measurements from stereo images. It assumes that you have studied the subject matter theoretically at level similar to Chapter 13 of my book, [1], and you wish to see the theory in practice. We use bare matlab with no particular toolboxes as prerequisite.

2 The stage and assumptions on data

In this lab exercise we will aim to find metric distances (in 3D) between various points of objects visible in a pair of Stereo images. The idea is to demonstrate that a pair of images taken from two sufficiently different locations can enable us to measure reasonably accurately distances in the real world.

To speed up the learning, the stage is already prepared for you. In particular we have a pair of such stereo images taken by two cameras ready for you. Access to data is a basic requirement.

An additional requirement is that information about cameras providing images, i.e. intrinsic and extrinsic variables must be available. Determination of this information is a whole science in itself, the camera calibration.

An example of camera calibration what and how it does is found in another publicly available lab-exercise,
http://www2.hh.se/staff/josef/public/education/image_analysis_II.html

We will assume that the two stereo cameras are calibrated individually, e.g. as in the reference or via matlab app for camera calibration, and the corresponding information about them is avalable to us.

3 Identify available variables and what they correspond to

This tutorial is centered on completing some code; about 10 lines in total. We will work with separate m-files. These are discussed next in the order you need to fix them.

Start by running the script `depth_by_triangulation`. This will result in errors, which is intentional. The corresponding matlab script file `depth_by_triangulation.m` needs to be complemented by you, as well as another file `pp2p.m`.

You can issue `whos` command (or look at the workspace) to quickly see which variables are available to you. Try to identify matrix and vector variable names corresponding to the stereo system used in eq. (13.85) of [1]. In particular you can put a break point in matlab at the line producing the error, rerun the script and point (without clicking) with mouse on any variable you wish to see explicitly. You should in particular be able to verify that both intrinsic matrices $\mathbf{M}_I^L$, $\mathbf{M}_I^R$, extrinsic matrices and translation matrices are available.

Can you figure out what the distance is between the world and one of the cameras by using `norm` command of matlab? Given your calculation, and assuming that the unit is metric, what is the most reasonable to conclude concerning the metric unit used in the two translation vectors, i.e. is the unit millimeter, centimeter or meter?

Can you verify that the extrinsic matrices really contain rotation matrices?

Please imagine for yourself on what frame is rotation matrix is operating? That is from which frame it is bringing one vector to deliver its vector representation in another? The way it is formulated in the code and in [1], in which coordinate-frame is the two translation vectors $\mathbf{t}_l$, $\mathbf{t}_r$ are given, i.e. world, left-camera or right-camera?

4 Obtain stereo extrinsic matrix

The extrinsic matrices you have at hand provide information about which viewing location and aspect-angle the two cameras have *with respect to* a common world coordinate-frame used in the calibration process. The world-frame is shown as small, red coordinates in Fig. 13.7. You donot have, as of yet, explicit information about how the two cameras are located w.r.t. each other in the room nor the amount of rotations relative each other, called here the *stereo extrinsic matrix*.

Remember please that stereo extrinsic matrix was fixed (unchanged) during the calibration as well as when the two pictures were taken, i.e. cameras were not moved with respect to each other. However, please note that it is perfectly possible to mount the stereo set-up on a vehicle and move it, without that the stereo extrinsic matrix changes. This is because while the vehicle moves the relative distance and rotation of the two cameras remain the same even if the distance and rotation angles relative the world frame changes. This is the reason why we are interested in the stereo extrinsic matrix, which remains the same even if the extrinsic matrices of each camera relative the world frame changes.

To pull-out stereo extrinsic matrix from individual exrinsic matrices (relative the world-frame) is your first task to be acheived via a few matlab commands involving matrix algebra. To do that you need to assume that one of the two cameras frame is the reference frame, e.g. the left camera-frame. The reference frame will then play the role of the world-frame, eliminating the need for the world frame. Consequently, as the vehicle in the example moves around distances of observed points will be computed relative the reference camera-frame. Hint: the relationship is made explicit in Lemma 13.8 of [1].

5 Triangulation

Triangulation is a term borrowed from the science of photogrametry, and the triangle refers to the triangle formed by camera centers and a point in the room, $\mathbf{O}^L$, $\mathbf{P}$, $\mathbf{O}^R$ in Fig. 13.7 of [1]. The triangle is known as soon as the two image points corresponding to $\mathbf{P}$ in the room are known.

Your second mission is to make P explicit, given column, row coordinates of a pair of image points, $\mathbf{P'}^L$ and $\mathbf{P'}^R$ corresponding to any point $\mathbf{P}$ in the room. As you will need to use this piece of code for many point pairs, you will need to write the necessary code as a function, `p2p.m` which will be called as needed in `depth_by_triangulation.m`.

6 Verification

To save you further time, fourteen pairs of image points corresponding to fourteen locations on objects are provided for you in matrices `ppl` (p prime left) and `ppr` (p prime right). Evidently you can append your own pairs image points to these matrices by using `ginput` command of matlab. Issue `whos` on these matrices and verify their size and look at Fig. 1 and Fig. 2 of matlab to see where they are in the image.

Your pp2p function is called in the for loop to provide the 3D locations of room $\mathbf{P}$s relative the reference camera. Now is the time to verify that these vectors make sense. Since their 3D locations are known, we are to compute select distances between room points by simple vector algebra. A distance between two room points is calculated with norm applied differences of their location vectors which your function has made available.

Are sizes of the rubic-cubes, the box on which they stand, and the diameter of the ball correct?

References

[1] J. Bigun. *Vision with Direction*. Heidelberg: Springer, 2006. DOI: 10.1007/b138918.