Assignment 5 of ELEC 547: Camera calibration

Due Date: March 9th 2020 (Monday)

Total points: 15

Reading assignment (3 pts): Before we go to the actual assignment of camera calibration, please familiarize yourselves by reading about the following pre-requisite topics. To get credit for the reading portion of the assignment, add a statement that you pledge to have read the reading material in your assignment submission.

i) *2D Projective space, homogeneous coordinates and 2D projective transformation (Homography)*: pp 25–44 of [1], scanned copy available in the "Readings" folder of the assignment.

Synopsis: The fundamental geometric concept in image formation (by a pin-hole camera) is that all points which lies on a line passing through the pin-hole (camera center) projects to a point. This geometric concept is best captured by "Projective Geometry", a field of mathematics. And images being 2D, a projective space of dimension 2 is the appropriate space to model them. The appropriate coordinate representation for a projective space is the homogeneous coordinate system. The transformations in projective space are known as homographies, which in the 2D case describes the relation between two images that are looking at a planar (2D) scene.

ii) *Computation of homography by DLT method:* pp 87-93 of [1], scanned copy available

Synopsis: Though homography is a non-linear transformation, in "homogenous" coordinates homography can be computed as a linear relation. Direct Linear Transformation (DLT) algorithm tells us how this can be achieved.

iii) *Camera models:* pp 153-165 of [1], scanned copy available

Camera models describe the relation between the 3D object and its imaged 2D points.

iv) *Computation of camera projection matrix P by DLT method:* pp 178-181, scanned copy available

Synopsis: Similar to homography, camera projection transformation is a non-linear transformation, which can be made linear in the homogenous coordinates. DLT method specifies how this can be achieved.

Practical Assignment: The goal of camera calibration is to find the intrinsic and extrinsic parameters of a camera. There are two major approaches for doing this: In the first approach, we image a 3D object such as a patterned cube and use the 3D-2D point correspondences between the 3D object and its 2d image to find the camera parameters. In the second approach, we take many images of a 2D (planar) object such as a checkerboard pattern and then use the 2D-2D point correspondences between the images of the object and the 2D object itself to find the camera parameters. In the first part of the assignment, we will use a Rubik's cube as a calibrating object to find the camera parameters. In the

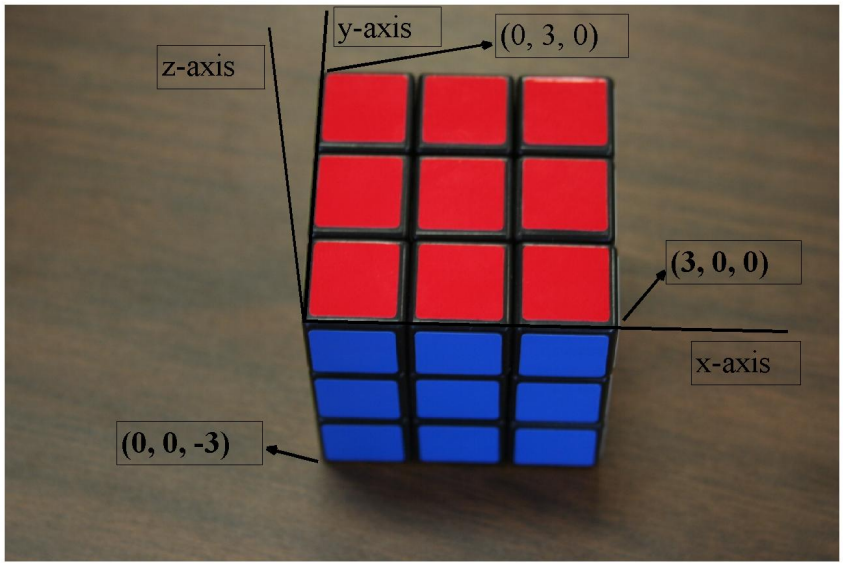
second part, we will use a state of the art calibration toolbox to calibrate our camera using a checkerboard pattern.

1. **Calibration using a 3D object:**

There are two set of parameters that we need to find: intrinsic parameters and extrinsic parameters. Intrinsic parameters are those parameters which are internal to the camera such as focal length, principal points, etc., whereas extrinsic parameters are those parameters that prescribe the location t (translation vector) and orientation R (rotation matrix) of the camera with respect to an external coordinate system (usually called the world coordinate system). In the first part, we will compute only the intrinsic parameters (assuming that the extrinsic parameters are known) and in the second part we will jointly compute the intrinsic and extrinsic parameters.

Part A. Intrinsic parameter computation (3 pts):

The calibrating object that we use is a Rubik's cube. We image the cube as shown in figure 1. We then obtain many 3D-2D point correspondences. In this part, we have already computed the point correspondences and all you have to do is to compute the intrinsic parameters from them.



The 3D-2D correspondences are given in the Matlab data file "pt_corres.mat" located in the folder "Matlab_funs_data". Use the "load" command of Matlab to load this data file in the Matlab workspace. Once you load this, you can find two variables: "pts_2D", which contains all the 2D points and "pts_3D", which contains all the corresponding 3D points. Obtain the intrinsic parameter matrix K, which is given by:

$$K = \begin{bmatrix} \alpha_x & s & p_x \\ & \alpha_y & p_y \\ & & 1 \end{bmatrix}$$, where α_x and α_y represents the focal length in terms of the x and y pixel dimensions, (p_x, p_y) are the principle points and s is the skew parameter.

Hints: The relation between the 2D point (x, y) and the corresponding 3D point (X, Y, Z) are given by

$x = \alpha_x(X/Z) + s(Y/Z) + p_x$ and $y = \alpha_y(Y/Z) + p_y$.

Part B. Intrinsic and extrinsic parameter computation (5 pts):

In the previous part, we have computed the intrinsic parameter assuming that the extrinsic parameters are known, i.e., we assumed that we know the 3D point correspondences in the camera coordinate system. But this is rarely the case; almost always we know the 3D point correspondences only in the world coordinate system and hence we need to estimate both the intrinsic and extrinsic parameters. But before that we need to obtain the 3D-2D point correspondences. The 3D points are described with respect to a world coordinate system as shown in the figure 1. The figure shows the x,y and z axes of the world coordinate system along with some sample 3D points, which are the corners of the squares. There are 28 such points.

i) Obtain the 3D-2D point correspondences. In Matlab, the 2D points can be obtained by the command "ginput". You can also use your preferred image viewer for this purpose.

ii) Next we want to compute the camera projection matrix P = K[R t], where K is the internal/intrinsic calibration matrix, R is the rotation matrix which specifies the orientation of the camera coordinate system w.r.t the world coordinate system and t is the translation vector which species the location of the camera center in the world coordinate system.

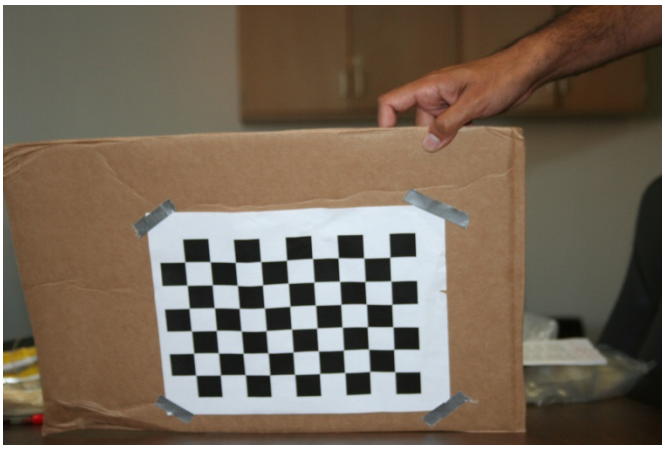
For computing P, use the function "CalibDLT.m"[3] provided in the "Matlab_funs_data" folder. This code is based on the "Direct Linear Transformation (DLT)" algorithm described in section 7.1 of [1]. Once you compute P, we next need to decompose it into the components K, R and t. For this use the function "P_to_KRt.m" located in "Matlab_funs_data" folder. This completes our computations of the extrinsic and intrinsic parameters.

iii) Now answer the following questions:

a. Describe the DLT algorithm in your own words.
b. For computing P, what is the minimum number of point correspondences needed?
c. Describe how the parameters K, R and t were obtained from the projection matrix P? [Hints: Let us write P = [M p_4], where M is the first 3*3 sub-matrix of P. Now, the expression P = K[R t] implies that M = KR, where K is an upper triangular matrix and R is a rotation matrix. This is an instance of a very famous matrix decomposition.]

iv) Next we want to verify the accuracy of the computed parameters. For this we will compute the re-projection error, which is a measure of the distance between the 2D points and the 2D points obtained by projecting the 3D points using the computed camera parameters. Its precise definition is given by:

$$err = \frac{\sqrt{\sum_{i=1}^{N}(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2}}{N}$$, where $(\hat{x}_i, \hat{y}_i)$ are the re-projected 2D points and N is the number of points. Compute this error. Also display all the re-projected 2D points in the original figure of the Rubik's cube.



2. Calibration using a planar object (4 pts):

For this part we will use the state of the art calibration toolbox: http://www.vision.caltech.edu/bouguetj/calib_doc/. The calibrating object here is a planar object (checkerboard pattern) as shown in figure 2. Download this toolbox. Go through the first example http://www.vision.caltech.edu/bouguetj/calib_doc/htmls/example.html , which gives a nice demo on finding the camera parameters.

i) Now use the checkerboard images in assignment folder "images", to compute the camera parameters of our camera. You will need to supply the size of each square (dX and dY variables), which is 25mm in our case. Report the obtained intrinsic parameters and their accuracies. Also include in your report the re-projection error figure (generated by clicking on the "Analyze error" button of the "calib_gui") and the extrinsic parameters figure (generated by clicking on the "Show extrinsic" button of "calib_gui"). Try to improve the accuracy of the calibration result by clicking on "Recomp. corners" and by changing the "window size for corner finder, wintx and winty". Report the new intrinsic parameters, their accuracies and the re-projection error and extrinsic parameters figure.

ii) Describe very briefly the principle behind this calibration technique. This calibration technique essentially follows the Z. Zhang's paper [2]. So all you need to do is summarize this paper very briefly in 2-3 lines (do not include mathematical formulas, just state the principle/high level idea).

References:

[1] R. Hartley and A. Zissermann, Multiview geometry, 2nd edition, Cambridge University Press.

[2] Z. Zhang. A flexible new technique for camera calibration. IEEE Transactions on Pattern Analysis and Machine Intelligence, 22(11):1330-1334, 2000.

[3] Matlab code "CalibDLT.m" obtained courtesy: http://www.daesik80.com/matlabcode/matlabcode.htm