ETH Zürich
Computer Science Department

Computer Vision

Camera calibration

Objective:

In this exercise you will learn how to calibrate a camera. The camera calibration computes the intrinsic parameters of a camera and estimates the distortion coefficients of the lens. Two algorithms will be implemented: the simple Direct Linear Transform (DLT) algorithm and the Gold Standard algorithm. If you get stuck check the exercise slides for some additional hints.

As a preparatory step you have to take at least one picture of a 3D calibration object. An example is shown in Figure 1. Make sure that the object fills almost the whole image. The closer the checkerboard patterns are to the image borders, the more precise the calibration and especially the radial distortion can be estimated. For your convenience, we provide the original image of Figure 1 in the attachment. You can use it directly for your implementation and report.

The main entry point for the exercise is the file `exercise1.m`. In there you will find comments about the overall structure as well as the necessary function calls. You don't need to change any code in the main file, the different tasks will be implemented in the respective functions. DO NOT CHANGE THE FUNCTION INTERFACES! We will run an automatic evaluation of your code.

To run the calibration you need the image coordinates and the 3D coordinates of the calibration object corners. To get the 3D coordinates you can simply assign the origin of the coordinate system somewhere on the calibration object (as shown in Figure 1). The code framework provides a function for clicking the point correspondences. For each clicked image point you have to enter the 3D-coordinate.

2.1 Data Normalization (20%)

Assume we are give N 2D-3D point correspondences, we can denote them with $\mathbf{x}_i = [x_i, y_i, 1]^T$ and $\mathbf{X}_i = [X_i, Y_i, Z_i, 1]^T$ in homogeneous coordinate respectively, where $i = 0, 1, \cdots, N-1$.

Data normalization is an essential part of the DLT algorithm to improve the numerical stability. For this exercise, transform the input points $\mathbf{x}_i$, $\mathbf{X}_i$ to normalized points $\hat{\mathbf{x}}_i$, $\hat{\mathbf{X}}_i$ so that they have zero mean and unit mean distance to the origin, e.g., $\frac{1}{N}\sum_{i=0}^{N-1} \hat{\mathbf{x}}_i = 0$ and $\frac{1}{N}\sum_{i=0}^{N-1} ||\hat{\mathbf{x}}_i|| = 1$ (This does not include the last axis of the homogeneous coordinate).

You have to find transformation matrices `T` and `U` to generate the normalized points: $\hat{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and $\hat{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$.

$$T^{-1} = \begin{bmatrix} \sigma_2 & 0 & \bar{x} \\ 0 & \sigma_2 & \bar{y} \\ 0 & 0 & 1 \end{bmatrix}$$

U^{-1} = SAME FOR 3D POINTS

$$\begin{bmatrix} \bar{x} \\ \bar{y} \end{bmatrix} = \frac{1}{K}\sum_{1}^{K} x_i \qquad \sigma_2 = \frac{1}{K}\sum^{K}\sqrt{x^2 - 0^2}$$

MEAN EUCLIDEAN DISTANCE TO ORIGIN

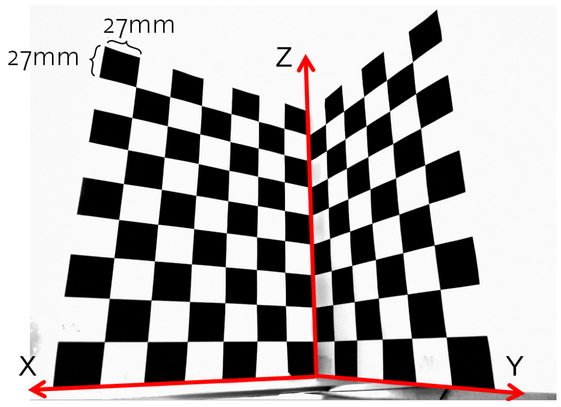

Figure 1: Typical calibration object.

2.2 Direct Linear Transform (40%)

Implement the DLT algorithm to estimate the camera matrix from the given point correspondences. The goal of the DLT is to minimize the algebraic error of the linear system of our camera model.

- Compute the normalized camera matrix $\hat{\mathtt{P}}$ with the DLT algorithm:

  Each correspondence $\hat{\mathbf{X}}_i \leftrightarrow \hat{\mathbf{x}}_i$ is inserted to the equation

$$\boxed{\begin{bmatrix} \hat{\mathbf{X}}_i^\top & \mathbf{0}_{1\times4} & -\hat{x}_i\hat{\mathbf{X}}_i^\top \\ \mathbf{0}_{1\times4} & -\hat{\mathbf{X}}_i^\top & \hat{y}_i\hat{\mathbf{X}}_i^\top \end{bmatrix} \begin{pmatrix} \hat{\mathbf{P}}^1 \\ \hat{\mathbf{P}}^2 \\ \hat{\mathbf{P}}^3 \end{pmatrix} = \mathbf{0}} \tag{1}$$

  where $\hat{\mathbf{P}}^i$ are the row vectors of $\hat{\mathbf{P}}$.

$A = U \cdot \Sigma \cdot V^T$

$P = V_N$

P = last column of V

$$P = \begin{bmatrix} v_1 & v_2 & v_3 & v_4 \\ v_5 & v_6 & v_7 & v_8 \\ v_9 & v_{10} & v_{11} & v_{12} \end{bmatrix}$$

  Stacking the equations for all point correspondences (i.e., 6 correspondences for DLT algorithm) results in a 12×12 matrix $\mathtt{A}$ where $\mathtt{A}\mathbf{P} = \mathbf{0}$. The solution for $\mathbf{P}$ is the right null-vector of $\mathtt{A}$ that can be computed with the singular value decomposition of $\mathtt{A}$.

- Compute the denormalized camera matrix $\mathtt{P} = \mathtt{T}^{-1}\hat{\mathtt{P}}\mathtt{U}$ using the matrices from last section.

- Factorize the camera matrix into the intrinsic camera matrix $\mathtt{K}$, a rotation matrix $\mathtt{R}$ and the camera center $\mathbf{C}$.

$$\mathtt{P} = [\mathtt{M}|-\mathtt{M}\mathbf{C}] = \mathtt{K}[\mathtt{R}|-\mathtt{R}\mathbf{C}] \tag{2}$$

  where $\mathtt{M} = \mathtt{KR}$ is the product of the intrinsic matrix and the camera orientation rotation matrix and $\mathbf{C}$ is the camera center. The matrix $\mathtt{M}$ can be decomposed with a RQ-decomposition. The camera center $\mathbf{C}$ is the point for which $\mathtt{P}\mathbf{C} = \mathbf{0}$. Which means that it corresponds to the the right null-vector that can be obtained from the SVD of $\mathtt{P}$.

$$P = k \cdot [R|t]$$
$$= k[R|-RC]$$

- Visualize the reprojected points of all checkerboard corners on the calibration object with the computed camera matrix.
- What happens if you use the unnormalized points?
  We loose Numerical Stability

2.3 Gold Standard algorithm (40%)

The Gold Standard algorithm minimizes the geometric error $\sum_i d(\hat{x}_i, \tilde{x}_i)^2$ where $\hat{x}_i$ is the normalized measured (clicked) point and $\tilde{x}_i$ is the projected object point $\hat{P}\hat{\mathbf{X}}_i$. Here $d(\hat{x}_i, \tilde{x}_i)$ is the Euclidean distance between $\hat{x}_i$ and $\tilde{x}_i$, and is defined as $\sqrt{(\hat{x}_i - \tilde{x}_i)^2 + (\hat{y}_i - \tilde{y}_i)^2}$.

- Normalize the input points and run the DLT algorithm to get an initial camera matrix $\hat{P}$ for the optimization.
- Visualize the hand-clicked points and the reprojection of the points on the calibration object obtained by using the computed camera matrix.
- Refine the estimated camera matrix $\hat{P}$ by minimizing $\sum_i d(\hat{x}_i, \tilde{x}_i)^2$. The cost function can be minimized with *fminsearch* from MATLAB. For Gold Standard algorithm, you can use more than 6 points to get more accurate results.
- Denormalize and factorize the camera matrix as in task 2.2
- Compute the mean reprojection-error (unit should be in pixels) of the used point correspondences.
- Visualize the reprojected points of all checkerboard corners on the calibration object with the computed camera matrix.

2.4 MATLAB Calibration Toolbox

This task will not be graded. If you are interested, you can try to calibrate your own camera with a standard toolbox provided by MATLAB. Details on how to use it can be found from [1]. You need to print out a calibration pattern and take several pictures of it. Make sure that the camera parameters stay constant for all images, e.g., do not change the zoom level and disable autofocus.

Hand in:

Write a short report explaining the main steps of your implementation and discussing the results of the methods. The report should contain images showing the clicked 2D points before calibration and the reprojected 3D points using the calibration parameters. Upload the report together with your source code to moodle.

References:

[1] `https://nl.mathworks.com/help/vision/ug/single-camera-calibrator-app.html`