ETH Zürich
Computer Science Department

Computer Vision

Exercise 2

Hand-out: 06-10-2016
Hand-in: 20-10-2016 15:00

Objective:

In this exercise you will calibrate your own 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 algorithm and the Gold Standard algorithm.

As a preparatory step you have to make at least one image 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.

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). When taking multiple images, make sure to keep the coordinate systems consistent. The code framework provides a function for clicking the point correspondences. For each clicked image point you have to enter the 3D-coordinate. For the first part of the assignment 1 image is enough, but you will need to select at least 6 points. This 6 points must **not** lie on the same plane.

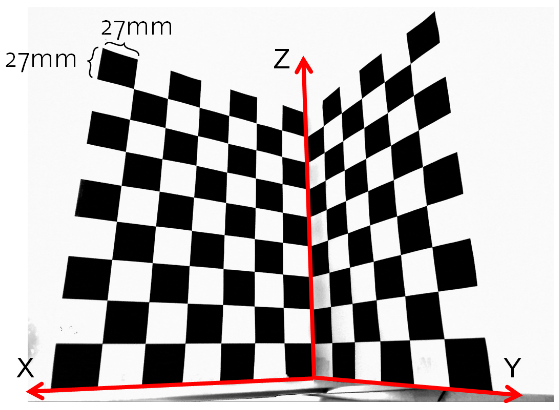

Figure 1: Typical calibration object.

2.1 Data Normalization (20%)

Data normalization is an essential part of the DLT algorithm. First, for all points, scale the homogeneous vectors such that the last entry becomes 1, *i.e.*, $x = (x, y, 1)^T$ and $X = (x, y, z, 1)^T$.

Then, transform the input points so that the centroid of the points is at the origin and that the average Euclidean distance from the origin is $\sqrt{2}$ for the image points, and $\sqrt{3}$ for the object points. The distance to the origin should not contain the last value of the homogeneous coordinate, so for instance, the distance to the origin for image points is $\sqrt{x^2 + y^2}$. Find transformations matrices $\mathtt{T}$ and $\mathtt{U}$ such that the normalized image points are $\hat{\mathbf{x}}_i = \mathtt{T}\mathbf{x}_i$ and the normalized object points are $\hat{\mathbf{X}}_i = \mathtt{U}\mathbf{X}_i$. Make sure the last entries remain as 1.

2.2 Direct Linear Transform (30%)

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

  $$\begin{bmatrix} w_i\hat{\mathbf{X}}_i^\top & \mathbf{0}^\top & -x_i\hat{\mathbf{X}}_i \\ \mathbf{0}^\top & -w_i\hat{\mathbf{X}}_i^\top & y_i\hat{\mathbf{X}}_i \end{bmatrix} \begin{pmatrix} \hat{\mathbf{P}}^1 \\ \hat{\mathbf{P}}^2 \\ \hat{\mathbf{P}}^3 \end{pmatrix} = \mathbf{0} \tag{1}$$

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

  Stacking the equations for all point correspondences results in a $2n \times 12$ 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 $\mathtt{P}$.

- 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?

2.3 Gold Standard algorithm (30%)

The Gold Standard algorithm minimizes the geometric error $\sum_i d(x_i, \hat{x}_i)^2$ where x_i is the measured (clicked) point and $\hat{x}_i$ is the projected object point $\mathtt{P}\mathbf{X}_i$. Here $d(x_i, \hat{x}_i)$ is the Euclidean distance between x_i and $\hat{x}_i$.

- Normalize the input points and run the DLT algorithm to get an initial camera matrix $\tilde{P}$ 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.
- Denormalize and factorize the camera matrix as in task 1.
- Visualize the reprojected points of all checkerboard corners on the calibration object with the computed camera matrix.

2.4 Bouget's Calibration Toolbox (20%)

There is a standard toolbox for MATLAB that is widely used for camera calibration. The toolbox can be downloaded from [1]. Read the instructions and the tutorial on that website and run the calibration with your camera. You need to print out a calibration pattern and make several images of it. Compare the results of this calibration with your own functions.

Helpful Matlab commands:

- ginput
- cross
- maketform
- imtransform

Be careful when using 'maketform'. Here, transformation matrices are applied to image coordinates from the right side. This means that you may need to transpose your transformation matrix.

2.5 Bonus:

- **Gold Standard algorithm with radial distortion estimation (+10%)**

  During the optimization of $\tilde{P}$ also the coefficients of the radial distortion introduced by the lens can be estimated. The standard model for radial distortion is

$$\begin{pmatrix} x_d \\ y_d \end{pmatrix} = L(\tilde{r}) \begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} \quad (3)$$

where $(\tilde{x}, \tilde{y})$ is the ideal linear projection of a 3D point $\mathbf{X}$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = [\mathtt{R}|-\mathtt{R}\mathbf{C}]\mathbf{X}$$

$$\tilde{x} = x/z$$

$$\tilde{y} = y/z$$

and (x_d, y_d) is the position after radial distortion. To compute the actual position in the image, the position $(x_d, y_d, 1)$ has to be multiplied with the intrinsic matrix $\mathtt{K}$. $\tilde{r}$ is the radial distance $\sqrt{\tilde{x}^2 + \tilde{y}^2}$ from the center for radial distortion and $L(\tilde{r})$ is the distortion factor, which is a function of the radius $\tilde{r}$ only.

The standard approximation of $L(r)$ may be given by its Taylor expansion $L(r) = 1 + \kappa_1 r^2 + \kappa_2 r^4 + ...$ Where $r > 0$ and $L(0) = 1$. The center of the radial distortion can be assumed to be the same as the principal point (p_x, p_y).

Add the radial distortion model to the computation of the projection of $\tilde{\mathbf{X}}_i$ (when computing the reprojection-error $d(\hat{\mathbf{x}}_i, \tilde{\mathtt{P}}\hat{\mathbf{X}}_i)^2$) in the Gold Standard minimization and estimate the first two radial distortion coefficients κ_1 and κ_2.

- **Image undistortion (+10%)**

  If the radial distortion coefficients are known, the introduced distortion can be removed. To correct points the following equations can be used

$$\tilde{x} = p_x + L(r)(x_d - p_x)$$

$$\tilde{y} = p_y + L(r)(y_d - p_y)$$

  where $r^2 = (x - p_x)^2 + (y - p_y)^2$. It is also possible to warp the whole image so that all the radial distortion vanishes.

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 the reprojected 3D points using the calibration parameters. Send the report together with your source code to moodle.

References:

[1] `http://www.vision.caltech.edu/bouguetj/calib_doc/index.html`