Assignment 4: *Camera Calibration and Epipolar Geometry*

Deadline: **23:59, Apr 17, 2024.**

Task

In this assignment, you are going to implement functions for (1) estimating a planar projective transformation for each of the planes of a calibration grid, (2) generating 2D-3D correspondences for corners on a calibration grid, (3) estimating a projection matrix for a camera, (4) decomposing a projection matrix into a camera calibration matrix and a matrix composed of the rigid body motion, and (5) estimating an essential matrix for a pair of cameras. To guide your coding, two partially completed Python programs are provided to you, which provide implementations for parsing the program arguments, loading inputs, GUI, and loading/saving output from/to a file. In completing this assignment, you only need to modify the following functions: `calibrate2D()`, `gen_correspondences()`, `calibrate3D()`, `decompose_P()` and `compose_E()`. Please refer to the tutorial notes as well as the comments in the source code for details of these functions.

Requirements

Your implementation should:

- ☐ [Calibration] Estimate a plane-to-plane projectivity for each of the two planes of the calibration grid using the 2D-3D point pairs picked by the user.
- ☐ [Calibration] Use the estimated plane-to-plane projectivities to assign 3D coordinates to all the detected corners on the calibration grid.
- ☐ [Calibration] Estimate a 3×4 camera projection matrix **P** from all the detected corners on the calibration grid using linear least squares.
- ☐ [Decomposition] Use QR decomposition to decompose the camera projection matrix **P** into the product of a 3×3 camera calibration matrix **K** and a 3×4 matrix [**R T**] composed of the rigid body motion of the camera.
- ☐ [Decomposition] Normalize the resulting camera calibration matrix **K** into the standard form (refer to slide 19 of lecture notes on camera calibration).
- ☐ [Epiplar geometry] Estimate an essential matrix from the camera projection matrices of a pair of cameras.

You can compare your results against the sample outputs for checking the correctness of your programs.

Submission

Points to note when submitting your assignment:

- You should include a `readme.txt` file describing the features you have implemented, especially when you have turned in a partially finished implementation.
- Pack your source code `calib.py`, `epipolar.py` and the `readme.txt` into a zip file and submit it via the Moodle page.
- No late submission will be accepted.

Appendix I: The calibration grid

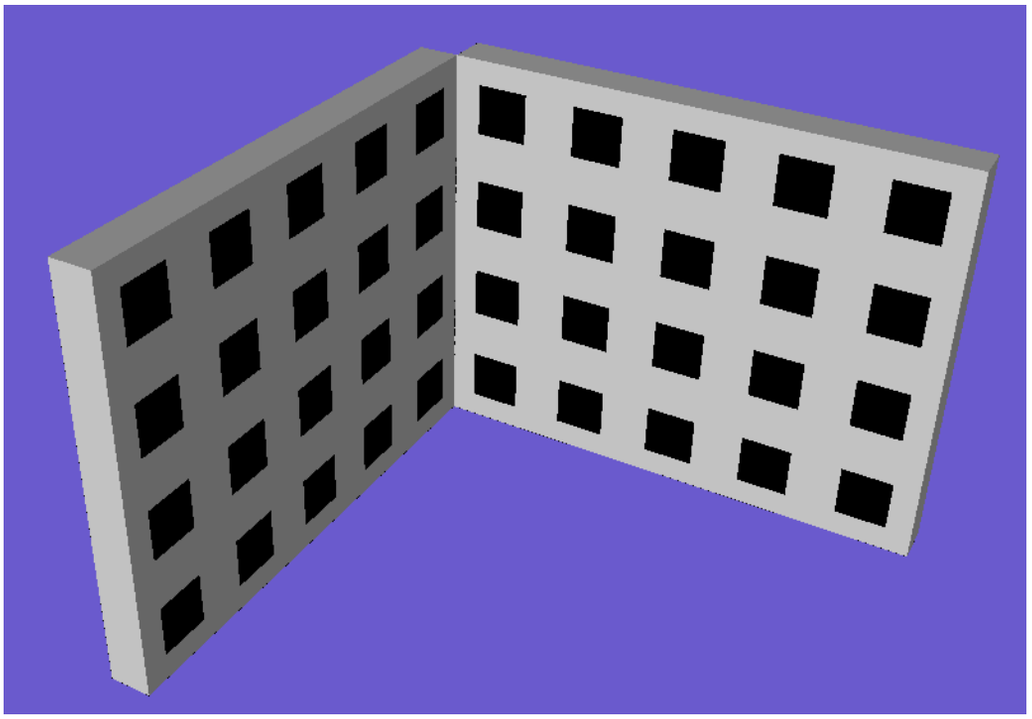

Figure 1: A 3D view of the calibration grid

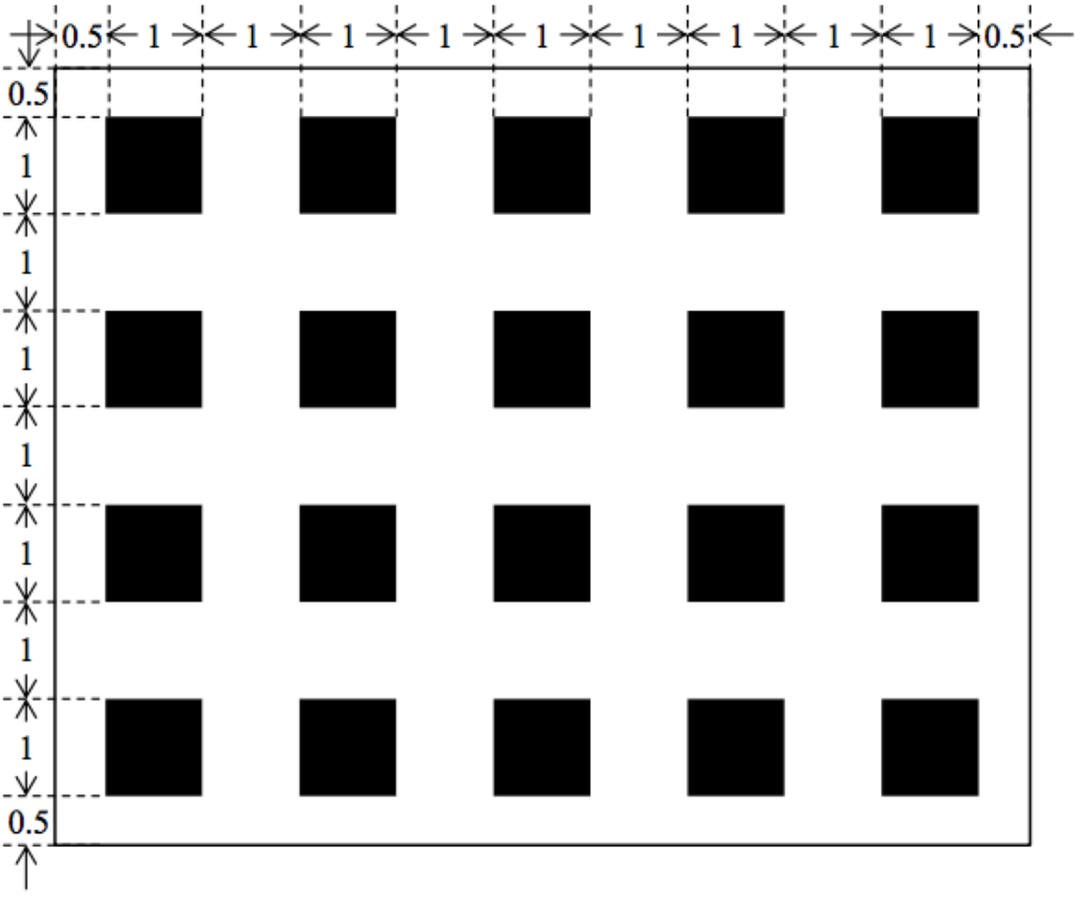

Figure 2: A 2D view of one plane of the calibration grid (unit = inch)

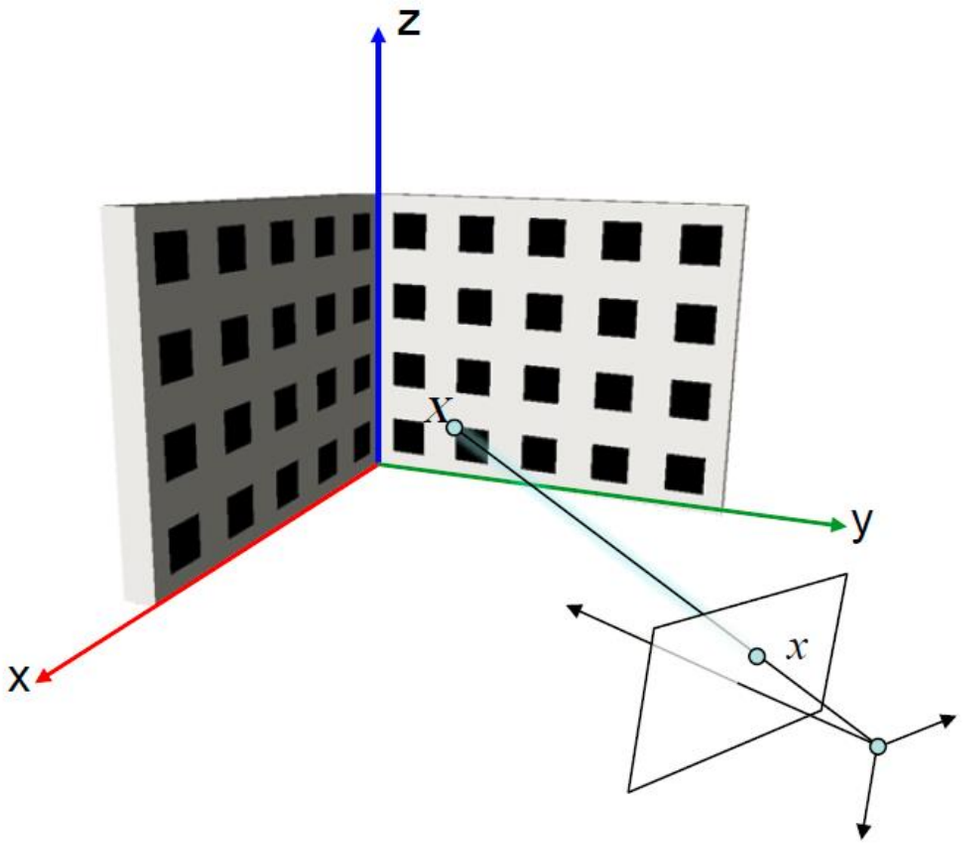

Figure 3: Definition of the world coordinate system

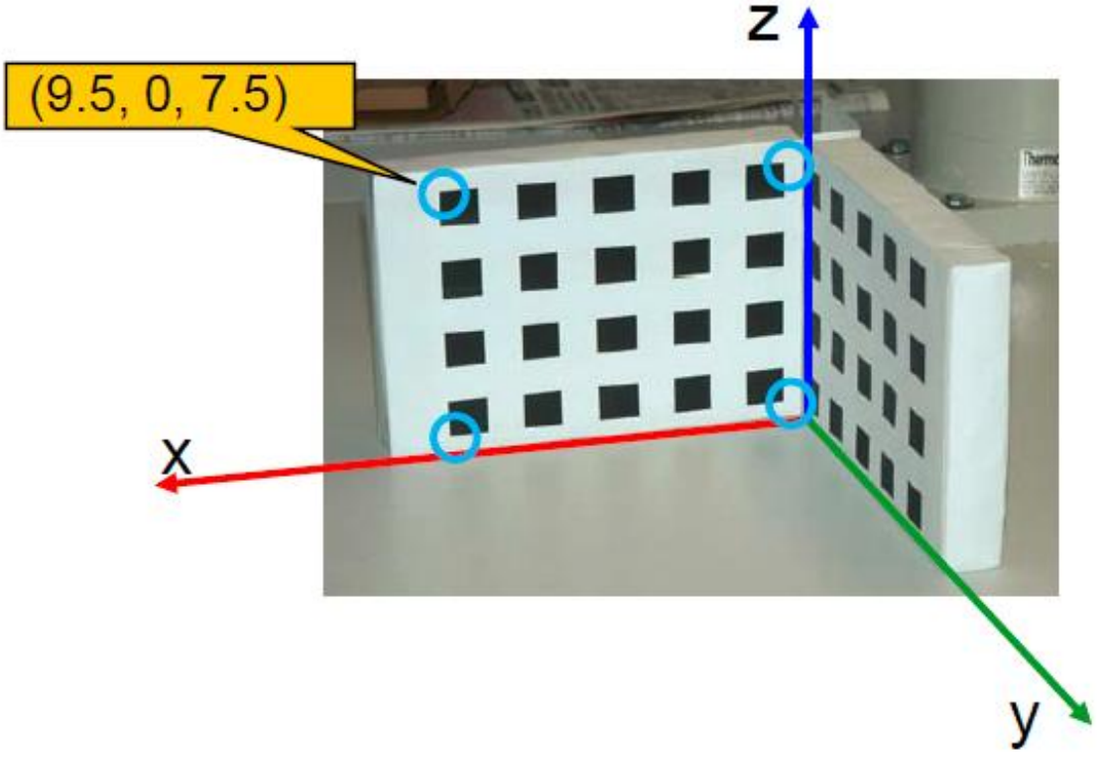

Figure 4: Example of 3D coordinates of a corner point

~ End ~