Due on 12 February 2023

Computer vision - Assignment 3

In this assignment we will learn camera calibration.

For this assignment, you may work in groups of 2 or 3. However, each student should submit their copy on Moodle. On each copy, please state the percent effort each individual member of your team has contributed to this assignment (numbers should add up to 100%).

For calibration, OpenCV implements Zhang's algorithm. This algorithm inputs correspondences between 2D image points and 3D scene points over a number of images and outputs the intrinsic camera matrix as well as the radial distortion coefficients $k1$ and $k2$. It also computes the rotation and translation of the scene towards the camera for every image, but this information is not needed for calibration. The scene points of the correspondence have to be in the same plane and to obtain good results, this plane should be different for every calibration image. In practice, an easy way to achieve this is to print a chessboard pattern onto a piece of paper, attach this paper to a rigid surface and take the corners between the chessboard squares as the correspondence points.

- Print and use this checkerboard calibration pattern.
- Acquire and save at least 10 images for the calibration.
- You may use the following tutorial to guide you through this exercise: `https://docs.opencv.org/4.x/dc/dbb/tutorial_py_calibration.html`
- If you use any other online resource, please mention it in your references.

1. Create the set of correspondence points for each of your images. For the 3D scene coordinates, measure in centimeters and use $(0, 0, 0)$ as coordinates for the top left corner. In the image, measure in pixels using $(0, 0)$ as the top left corner. Add these correspondence points to the solution document.
2. Manually selecting the corner pixels in the image is tedious and we want to automate it. Use cv.findChessBoardCorner and cv.cornerSubPix to find and refine these positions automatically.
3. Use the function cv.calibrateCamera to calibrate your camera using your correspondences. Add the intrinsic camera matrix and the lens distortion parameters to your solution document. Format floats using printf(”%.4f”).
4. Calculate the reprojection error.
5. Repeat the calibration process using the dot grid pattern. For this pattern, you will have to calculate and use the centroids of the dots as the ‘object points’ and ‘image points’.