Manual of Software Package for Precise Camera Calibration

1. Overview

This is a manual of Precise Camera Calibration for Matlab®. This package is based on Camera Calibration Toolbox for Matlab® [1]. vgg_condition_2d.m, vgg_get_nonhomg.m, and vgg_Haffine_from_x_MLE.m are download from the below website [2].

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

[2] http://www.robots.ox.ac.uk/~vgg/hzbook/code/

Fig.1 shows the flowchart of our software package. Step1, 2, and 3 are same as the traditional camera calibration. CircleCalibrateScript.m performs these steps. CalibrateMain.m and CalibrateMainReal.m call CircleCalibrateScript.m. CalibrateMain.m and CalibrateMainReal.m load input images and sets parameters. CalibrateMain.m is for synthetic image dataset. CalibrateMainReal.m is for real image dataset. Modify and execute these files.

2. Parameters

- data_flag: 1 for real datasets; 0 for synthetic datasets
- N_ima: number of images
- numIter: number of iteration
- dX: distance between adjacent control points on planer grid pattern(x-axis) [mm]
- dY: distance between adjacent control points on planer grid pattern(y-axis) [mm]
- dx: number of control points (x-axis)
- dy: number of control points (y-axis)
- concentric_flag: 0 for circle pattern; 1 for ring pattern; -1 for square pattern
- optimization_flag: 0 for Bundle Adjustment without Step 7; 1 for Weighted Bundle Adjustment
- outlier_flag: 0 for without Step 6; 1 for with Step 6

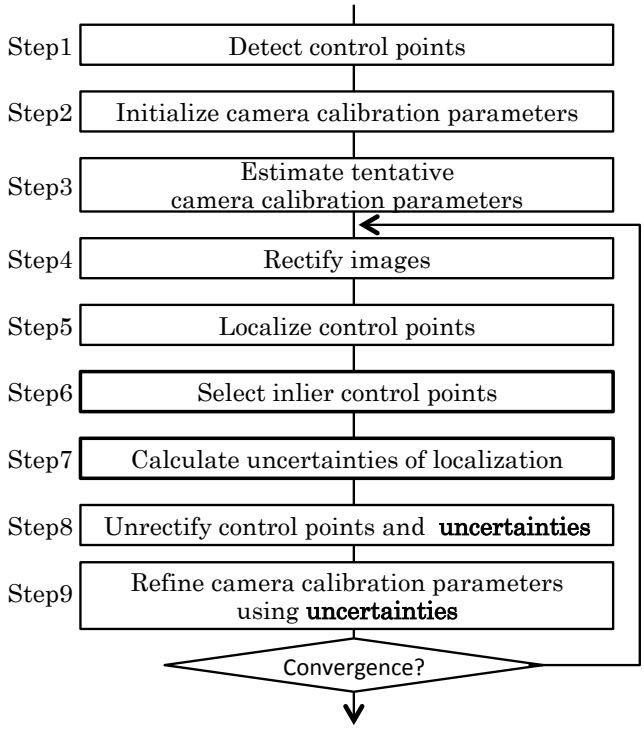


Fig. 1. Flowchart

3. Description of the calibration parameters

The calibration parameters of this software package are same as those of Camera Calibration Toolbox for Matlab® [1].

4. Description of the Fuctions in the software package

The functions in Fig.1;

- Step 1: for circle and ring; corner_grid_detect.m for square
- Step 2: init_intrinsic_param.m called in go_calib_optim_iter.m
- Step 3: go_calib_optim_iter.m
- Step 4: Proj_Undistort3.m
- Step 5: computeCorrelationSSD.m for circle and ring; computeSquareCorrelation.m for square
- Step 6: removeOutlier.m
- Step 7: calcUncertainty.m
- Step 8: calcWeight.m and Proj_Points.m
- Step 9: go_calib_optim_iter.m

Other functions;

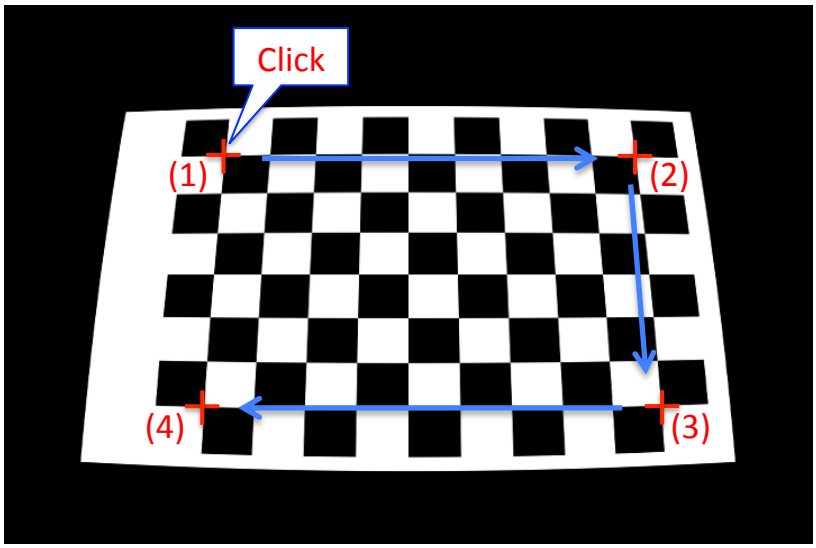
- Calc_maxgrid.m: Calculates the maximum distance between any adjacent control points on input images
- createFilter2.m: Creates a template image for Sum of Squared Differences (SSD).
- curve2d.m: Fits quadric surface.
- quad_subpixelssd.m: Performs sub-pixel localization of control points using the result of SSD.
- Unshading.m: Corrects unevenness of image brightness such as optical vignetting.
- computeCorrelationNCC.m: We don't use this function. When you want to try NCC, change the code in CircleCalibrateScript.m from computeCorrelationSSD.m to computeCorrelationNCC.m.
- quad_subpixel.m: Performs sub-pixel localization of control points using the result of NCC.

5. Bounding box

User need to set bounding box for Step1.

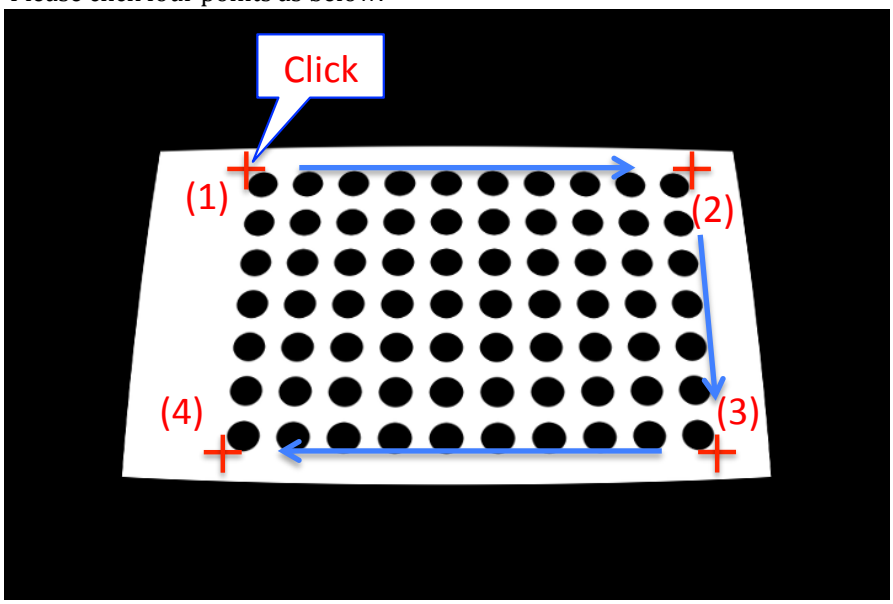
5.1 Square grid pattern

After you execute CalibrateMainReal.m, A figure object will appear. Please click four points as below.



5.2 Circle grid pattern and Ring grid pattern

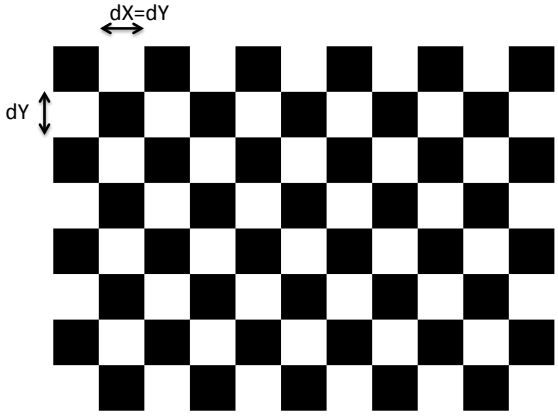
Please click four points as below.



6. Description of grid pattern

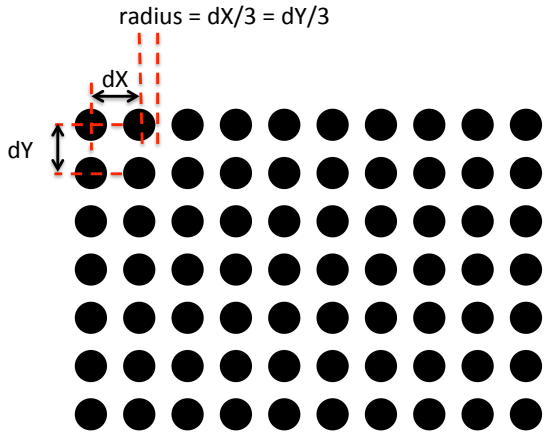
Print a grid pattern and paste on a flat panel. Generate the grid pattern yourself or use our grid pattern (GridPattern.pdf). After printing, measure the dX and dY sizes of the printed grid pattern. And then set those values to CalibrateMainReal.m

6.1 Our square grid pattern



6.2 Our circle grid pattern

If you change the radius of circle grid pattern, you need to modify gcircleFilter.bmp.



6.3 Our ring grid pattern

If you change the radius of ring grid pattern, you need to modify gringFilter.bmp.