3D Data Processing
Camera Calibration Tutorial

Alberto Pretto

Camera Calibration

Estimate camera matrix + distortion coefficients

$$camera\ matrix = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

$$Distortion\ coefficients = (k_1 \quad k_2 \quad p_1 \quad p_2 \quad k_3)$$

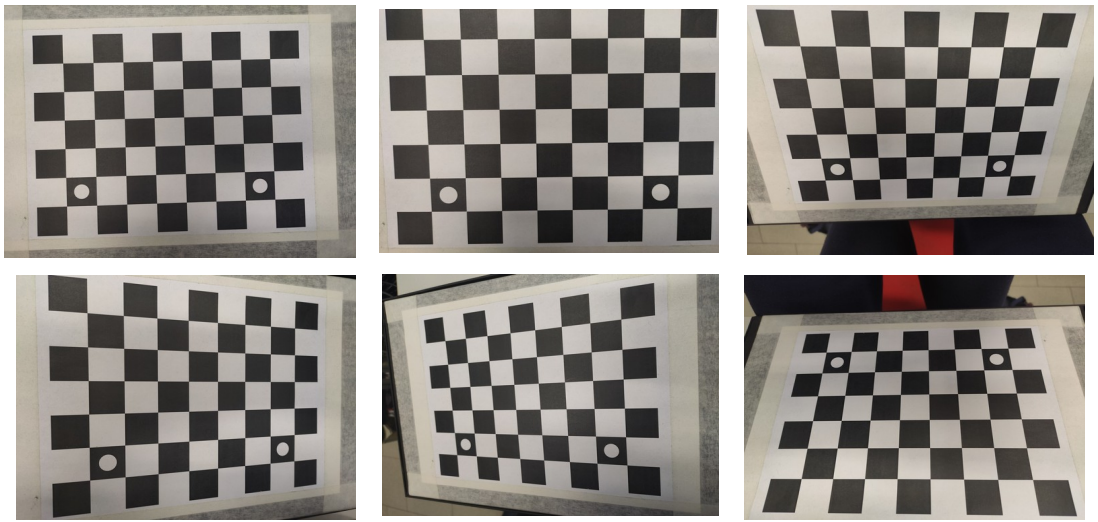
Procedure

1) Print the chessboard (with 2 white circles markers) available for download on the course Moodle page
2) Carefully measure the size of the squares of the printed chessboard (in A4 format, the size of each square should be around 27mm)
3) Place the chessboard in a fixed position
4) Acquire calibration images with your camera (e.g. mobile phone camera)
5) Copy the acquired images on your computer
6) Run the camera calibration app

Calibration Images Acquisition

- Frame the chessboard from different viewpoints
- Try to cover the entire image plane
- Frame the chessboard even from slightly skewed viewpoints
- 15-20 images should be enough

Sample acquisitions



Camera Calibration App

- Build instructions

sudo apt install build-essentimake libboost-filesystem-dev libopencv-dev libomp-dev libceres-dev libyaml-cpp-dev libgtest-dev libeigen3-dev

git clone https://bitbucket.org/alberto_pretto/cv_ext.git
cd cv_ext
git checkout origin/dev --track
git pull
mkdir build
cd build
cmake -DBUILD_EXAMPLES=ON ..
make

- Run instructions

./bin/cam_calib -f <images dir> -c <output calibration yaml file> --bw 8 --bh 6 -q <square size in meters> -k -s 4 -u --opencv_format

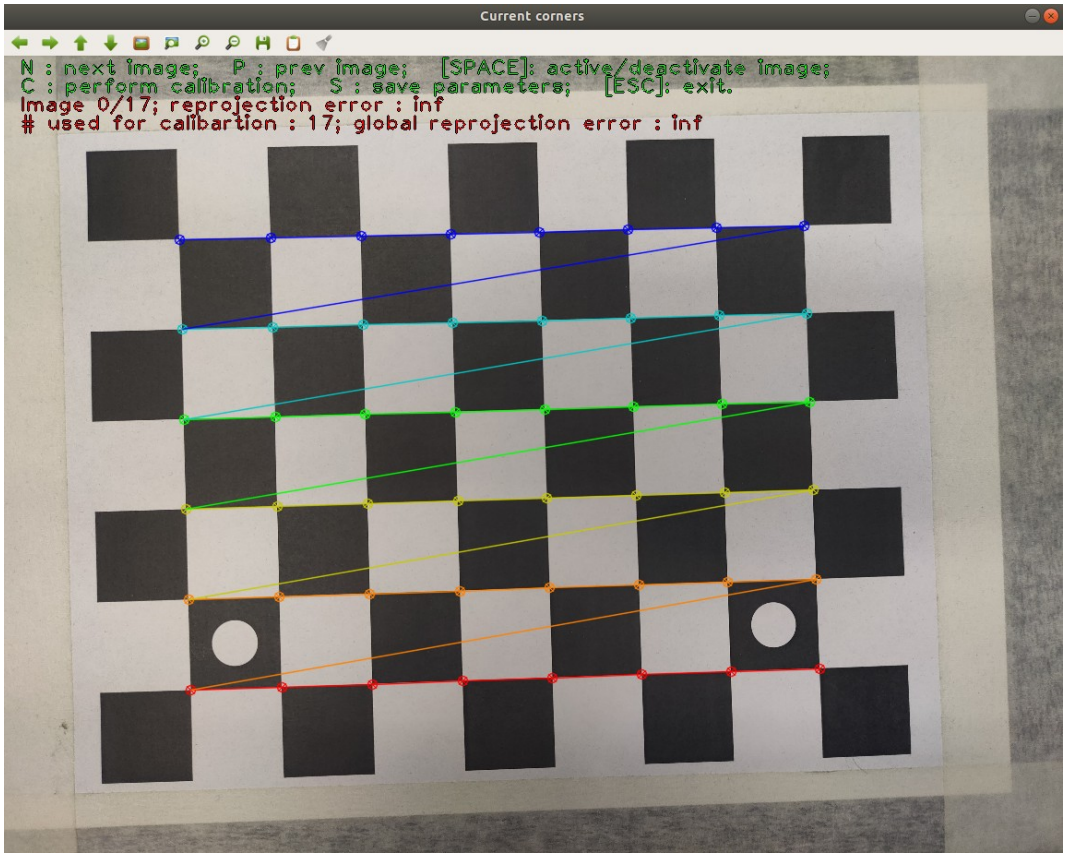
Run ./bin/cam_calib --help for more instructions

Camera Calibration App - GUI

Commands:

- (N/P): cycle through images
- (SPACE): activate/deactivate image in the calibration process (used for discarding bad chessboard detections)
- (C): run calibration
- (S): save calibration yaml file in OpenCV format
- (ESC): show undistorted images (using estimated calibration) and exit

For a good calibration, the global reprojection error should be <1.0



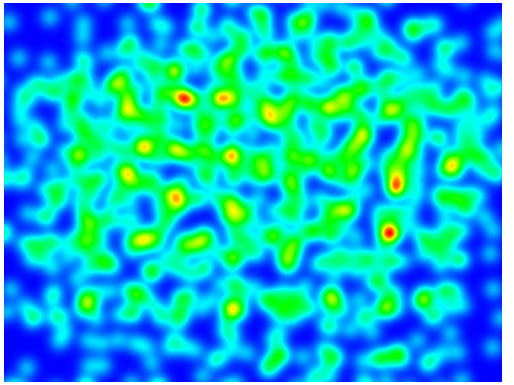
Camera Calibration App - GUI

The detected chessboard corners distribution is shown.

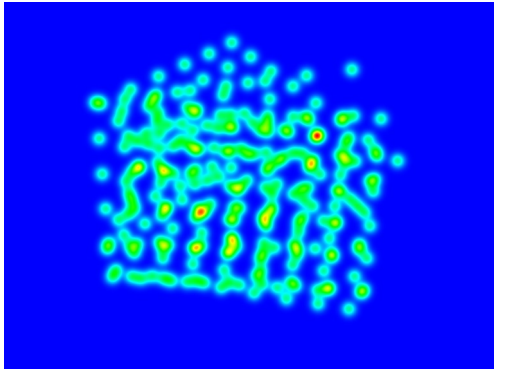
Detected corners should be uniformly distributed over the entire image plane.

If part of image plane remains uncovered, further images should be acquired for calibration.

Good distribution



Bad distribution



Camera Calibration - Output

The camera calibration app generates an OpenCV compliant yaml file storing:

- **width/height**: image size
- **K**: 3x3 camera matrix
- **D**: 8-vector distortion coefficients (only the first five elements are used)

```
%YAML:1.0
---
width: 4608
height: 3456
K: !!opencv-matrix
   rows: 3
   cols: 3
   dt: d
   data: [ 3.6691836702103928e+03, 0., 2.3257537713180132e+03, 0.,
       3.6632030236174128e+03, 1.7236960483738796e+03, 0., 0., 1. ]
D: !!opencv-matrix
   rows: 8
   cols: 1
   dt: d
   data: [ -1.0661430154175276e-02, 1.0496154350180879e-01,
       -1.6570585758935688e-03, -1.1478660716590949e-03,
       -2.5123421922977196e-01, 0., 0., 0. ]
```