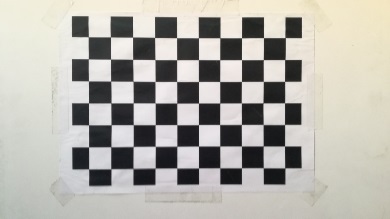
# COMPUTER VISION

**EXERCISE 9a: Camera calibration**

Concepts: Camera calibration, intrinsic parameters, distortion.

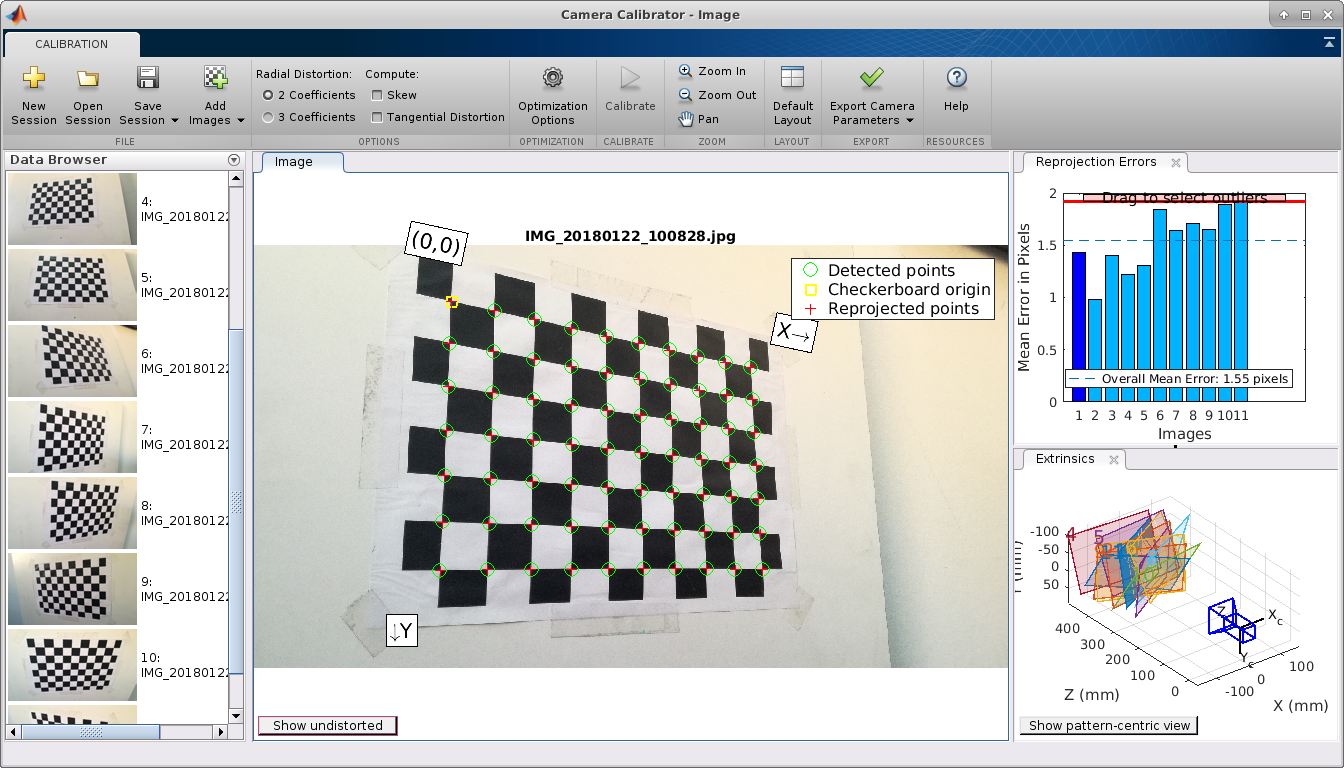
**Camera calibration.** Since we are almost experts in computer vision, we are going to calibrate the camera of our smartphone or laptop. For that we are going to use the amazing tutorial and calibration application from MATLAB. The tutorial is full of interesting information and directions about how to properly calibrate the camera, as well as how to interpret the obtained results. To calibrate the camera, you only need the provided application, a pattern with a chessboard, and the camera itself! Take different pictures of the chessboard with different perspectives, run the application, and enjoy! **For that,** **follow the next steps:**

* 1. Take a look at the tutorial of the *Camera Calibrator**application*. With such an application you can calibrate standard or fisheye cameras. We will focus on standard cameras. Tutorial url: <https://es.mathworks.com/help/vision/ug/single-camera-calibrator-app.html>
  2. Following the hints given in the tutorial, take around 15 images of the calibration pattern with your camera, and add them to the application (*Add Images* button). If the camera resolution is too high, resize the images. Include thumbs of the used images in the report.

**  

*Example of images observing the calibration pattern from a smartphone.*

* 1. Calibrate the camera (*Calibrate* button). The output should be similar to the one in the image below. Include a similar screenshot with your output, and explain the meaning of the two subfigures on the right side (*Projection errors* and *Extrinsics*).



*Matlab Camera Calibrator application after calibrating a camera.*

* 1. There are two ways to export the resultant camera parameters:
     1. Export parameters to Workspace: this option will create a structure in your workspace called *cameraParams* with a number of fields, for example: *intrinsicMatrix*, *FocalLength*, *PrincipalPoint*, *RadialDistortion*, etc.
     2. Generate Matlab script: this way automatically generate a script that performs all the previous steps: loads the images, calibrates the camera, and displays the results, also creating a *cameraParams* struct.

Use either option to check the camera parameters. Comment in your report as many parameters as you can identify from the calibration lecture (cx, cy, f, etc.) along with their units (meters, pixels, etc.). Also include the content of the computed *intrinscMatrix*.

**EXERCISE 9b: Fundamental matrix**

Concepts: Harris operator, fundamental matrix, epipolar lines, stereo.

Once the camera is calibrated, take a pair of pictures from surfaces with visual features, and measure the distance that the camera moved between both shots. It will be our baseline **b**. *Note: 10cm or so is ok.*

*Example of three images with a planar surface (the cereal box), where the camera moved 5cm to the right from the image on the left to the one in the middle, and 10 cm to the one on the right.*

1. **Fundamental matrix.** Now run the Harris keypoint detector that we designed in a previous exercise, and match the obtained keypoints. Then compute the fundamental matrix using the provided function **ransacfitfundmatrix**. *Note: if you were unable to calibrate your camera use the pepsi\_left.tif and pepsi\_right.tif images.*
2. **Epipolar lines**. Next, using **ginput**, set any pixel on the left image and draw on the right one the corresponding epipolar line. ¿Where are the epipoles in each image?
3. **Tridimensional reconstruction.** Finally, project in 3D the matched points by cross correlation using the following equations, and represent them in a figure using **plot3**:

|  |
| --- |
| Xi = b\*(xli-cx)/di  Yi = b\*(yli-cy)/di  Zi = f\*b/di |

Use the parameters cx, cy, and f from your camera, and the baseline between the two pictures. If you are working with the provided ones (pepsi images), use the following ones:

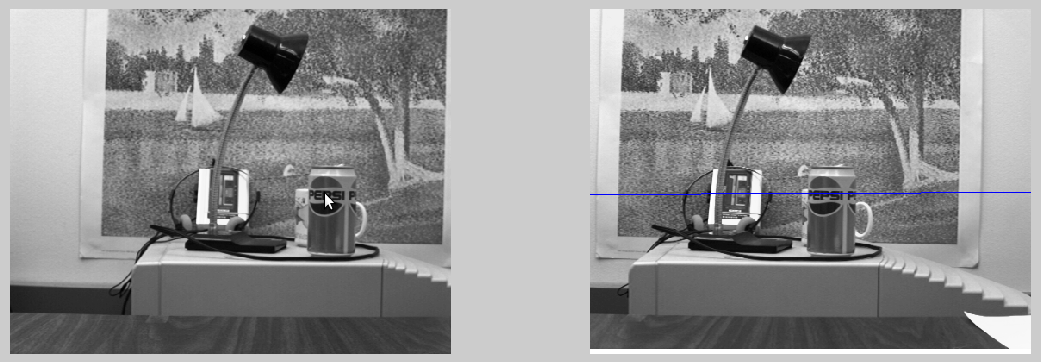
b = 0.119 m, cx = 255.64 px, cy = 201.12 px, f = 351.32 px, di= disparity of i-th point

**Commands:**

|  |  |
| --- | --- |
| **[F, inliers] = ransacfitfund matrix(x1,x2,1e-5);** | Computes the fundamental matrix from two sets of points matched using RANSAC. |
| **plot3(x,y,z)** | The same as plot, but in 3D. |
| **patch(x,y,z,c)** | Draws a 3D surface with color c=[r,g,b]. |

**Results**

*Epipolar lines*



*Projected points in 3D. Green Surface representing the poster.*

