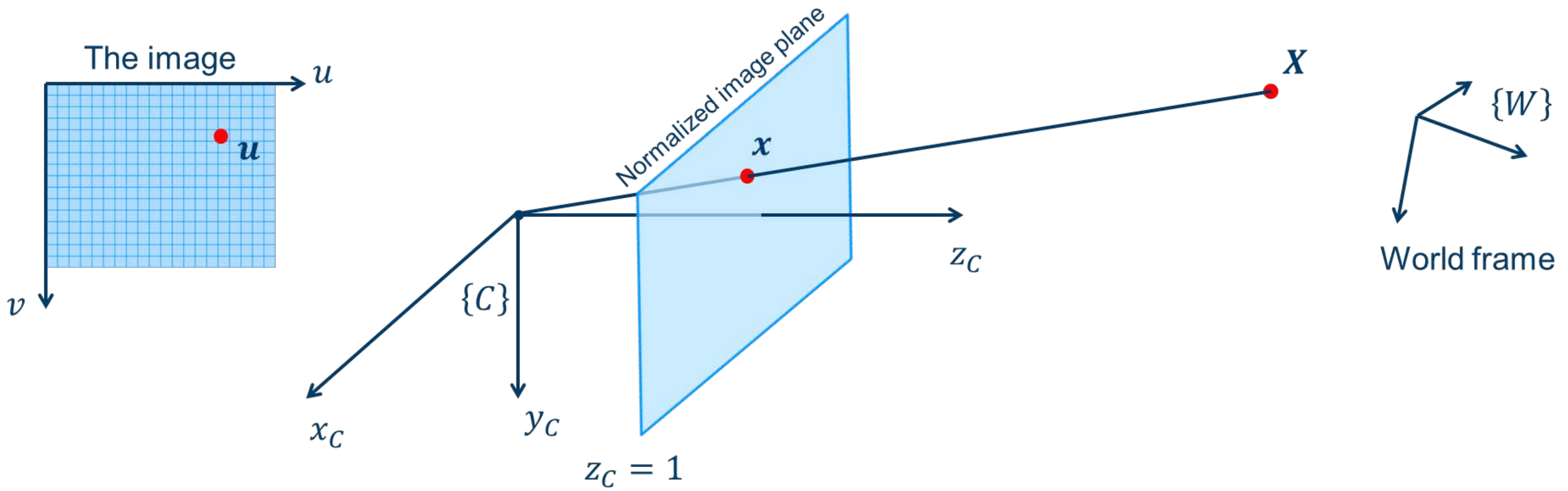
# Lecture 5.3
# Camera calibration

Thomas Opsahl

# Introduction




- For finite projective cameras, the correspondence between points in the world and points in the image can be described by the simple model

$$P = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} R & \boldsymbol{t} \end{bmatrix}$$

- This camera model is typically not good enough for accurate geometrical computations based on images

# Introduction

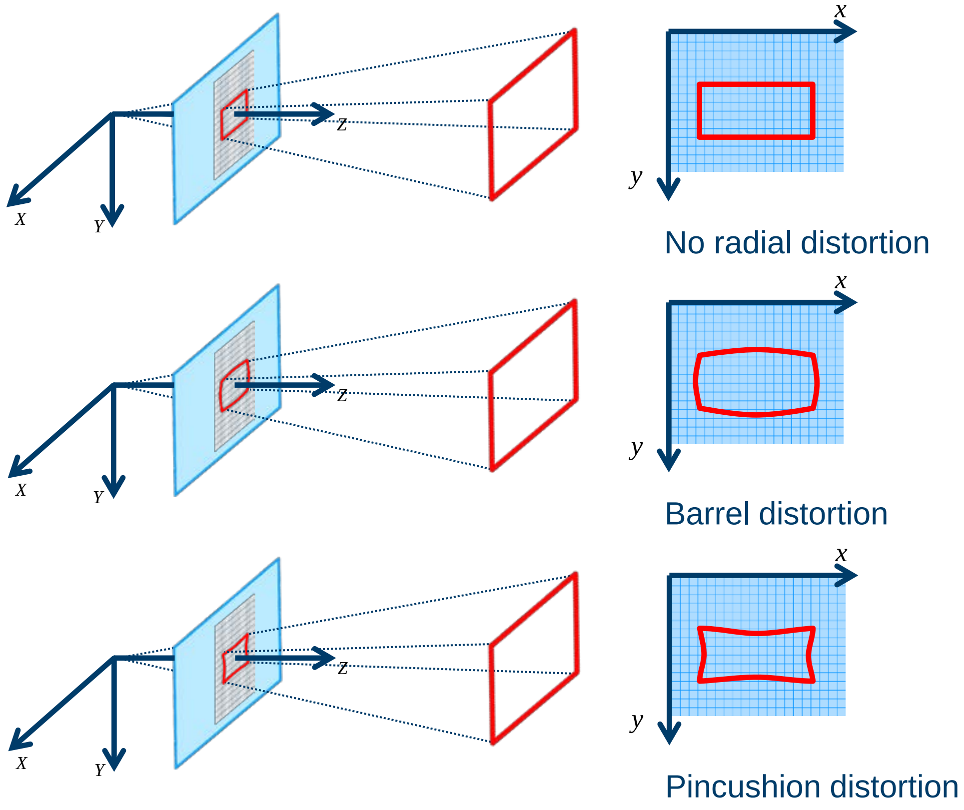
- Since light enters the camera through a lens instead of a pinhole, most cameras suffer from some kind of distortion

- This kind of distortion can be modeled and compensated for

- A radial distortion model can look like this

  $$\hat{x} = x(1 + \kappa_1(x^2 + y^2) + \kappa_2(x^2 + y^2)^2)$$
  $$\hat{y} = y(1 + \kappa_1(x^2 + y^2) + \kappa_2(x^2 + y^2)^2)$$

  where $\kappa_1$ and $\kappa_2$ are the radial distortion parameters



# Introduction

- When we calibrate a camera we typically estimate the camera calibration matrix $K$ together with distortion parameters

- A model $K[R \quad \boldsymbol{t}]$, using such an estimated $K$, does not in general describe the correspondence between points in the world and points in the image

- Instead it describes the correspondence between points in the world and points in a undistorted image – an image where the distortion effects have been removed

Image from a non-projective camera


Equivalent projective-camera image


Images: http://www.robots.ox.ac.uk/~vgg/hzbook/

# Undistortion

- So earlier when we estimated homographies between overlapping images, we should really have been working with undistorted images!

- How to undistort?
  - Matlab
```
[undist_img,newOrigin] = undistortImage(img,cameraParams);
undistortedPoints = undistortPoints(points,cameraParams);
```
  - OpenCV
```
cv::undistort(img, undist_img, P, distCoeffs);
cv::undistortPoints(pts,undist_pts,P,distCoeffs);
```

- The effect of undistortion is that we get an image or a set of points that satisfy the perspective camera model $\tilde{\boldsymbol{u}} = P\tilde{\boldsymbol{X}}$ much better than the original image or points
  - So we can continue working with the simple model

# Camera calibration

- Camera calibration is the process of estimating the matrix $K$ together with any distortion parameter that we use to describe radial/tangential distortion

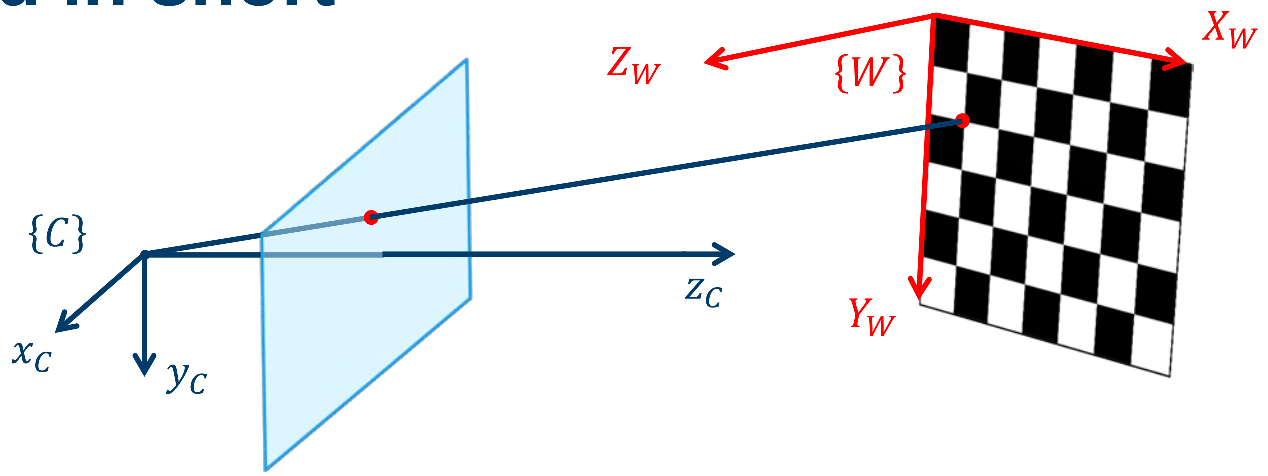
- We have seen how $K$ can be found directly from the camera matrix $P$

- The estimation of distortion parameters can be baked into this

- One of the most common calibration algorithms was proposed by Zhegyou Zhang in the paper ''*A Flexible New Technique for Camera Calibration*'' in 2000
  - OpenCV: *calibrateCamera*
  - Matlab: Camera calibration app

- This calibration algorithm makes use of multiple images of a asymmetric chessboard

# Zhang's method in short




- Zhang's method requires that the calibration object is planar

- Then the 3D-2D relationship is described by a homography

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K\begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{r}_3 & \boldsymbol{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} = K\begin{bmatrix} \boldsymbol{r}_1 & \boldsymbol{r}_2 & \boldsymbol{t} \end{bmatrix}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} = H\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix}$$

# Zhang's method in short

- This observation puts 2 constraints on the intrinsic parameters due to the fact that $R$ is orthonormal

$$\left.\begin{array}{r} \boldsymbol{r}_1^T \boldsymbol{r}_2 = 0 \\ \boldsymbol{r}_1^T \boldsymbol{r}_1 = \boldsymbol{r}_2^T \boldsymbol{r}_2 \end{array}\right\} \Rightarrow \left\{\begin{array}{c} \boldsymbol{h}_1^T K^{-T} K^{-1} \boldsymbol{h}_2 = 0 \\ \boldsymbol{h}_1^T K^{-T} K^{-1} \boldsymbol{h}_1 = \boldsymbol{h}_2^T K^{-T} K^{-1} \boldsymbol{h}_2 \end{array}\right.$$

- Where $H = \begin{bmatrix} \boldsymbol{h}_1{}^T & \boldsymbol{h}_2{}^T & \boldsymbol{h}_3{}^T \end{bmatrix}$ and $K^{-T} = (K^T)^{-1} = (K^{-1})^T$

- So for each 3D-2D correspondence we get 2 constraints on the matrix $B = K^{-T} K^{-1}$

- Next step is to isolate the unknown parameters in order to compute $B$

# Zhang's method in short

- $B$ can be computed directly from $K$

$$B = K^{-T}K^{-1} = \begin{bmatrix} b_{11} & b_{12} & b_{13} \\ b_{21} & b_{22} & b_{23} \\ b_{31} & b_{32} & b_{33} \end{bmatrix} = \begin{bmatrix} \frac{1}{f_u^2} & -\frac{s}{f_u^2 f_v} & \frac{v_0 s - u_0 f_y}{f_u^2 f_v} \\ -\frac{s}{f_u^2 f_v} & \frac{s^2}{f_u^2 f_v^2} + \frac{1}{f_v^2} & \frac{s\left(v_0 s - u_0 f_y\right)}{f_u^2 f_v^2} - \frac{v_0}{f_v^2} \\ \frac{v_0 s - u_0 f_y}{f_u^2 f_v} & \frac{s\left(v_0 s - u_0 f_y\right)}{f_u^2 f_v^2} - \frac{v_0}{f_v^2} & \frac{\left(v_0 s - u_0 f_y\right)^2}{f_u^2 f_v^2} + \frac{v_0^2}{f_v^2} + 1 \end{bmatrix}$$

- We see that $B$ is symmetric, so that we can represent it by the parameter vector

$$\boldsymbol{b} = [b_{11} \quad b_{12} \quad b_{22} \quad b_{13} \quad b_{23} \quad b_{33}]^T$$

# Zhang's method in short

- If we denote

$$\boldsymbol{v}_{ij} = \begin{bmatrix} h_{i1}h_{j1} \\ h_{i1}h_{j2} + h_{i2}h_{j1} \\ h_{i2}h_{j2} \\ h_{i3}h_{j1} + h_{i1}h_{j3} \\ h_{i3}h_{j2} + h_{i2}h_{j3} \\ h_{i3}h_{j3} \end{bmatrix} \quad \text{then} \quad \boldsymbol{h}_i^T B \boldsymbol{h}_j = \boldsymbol{v}_{ij}^T \boldsymbol{b}$$

- Thus we have

$$\left.\begin{matrix} \boldsymbol{r}_1^T \boldsymbol{r}_2 = 0 \\ \boldsymbol{r}_1^T \boldsymbol{r}_1 = \boldsymbol{r}_2^T \boldsymbol{r}_2 \end{matrix}\right\} \Rightarrow \left\{\begin{matrix} \boldsymbol{h}_1^T B \boldsymbol{h}_2 = 0 \\ \boldsymbol{h}_1^T B \boldsymbol{h}_1 = \boldsymbol{h}_2^T B \boldsymbol{h}_2 \end{matrix}\right\} \Leftrightarrow \begin{bmatrix} \boldsymbol{v}_{12}^T \\ \boldsymbol{v}_{11}^T - \boldsymbol{v}_{22}^T \end{bmatrix} \boldsymbol{b} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

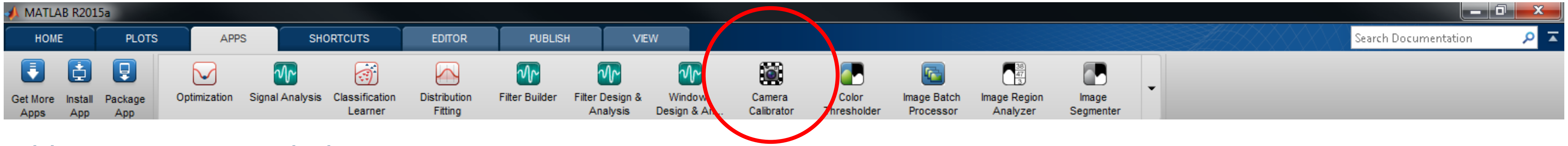
# Zhang's method in short

- Given $N$ images of the planar calibration object we stack the equations to get a homogeneous system of linear equations which can be solved by SVD when $N \geq 3$
- From the estimated $\boldsymbol{b}$ we can recover all the intrinsic parameters
- The distortion coefficients are then estimated solving a linear least-squares problem
- Finally all parameters are refined iteratively
- More details in Zhang's paper
  http://research.microsoft.com/en-us/um/people/zhang/Papers/TR98-71.pdf

# Camera calibration in practice

- OpenCV
  - Camera calibration tutorial
  - We'll test it out in the lab

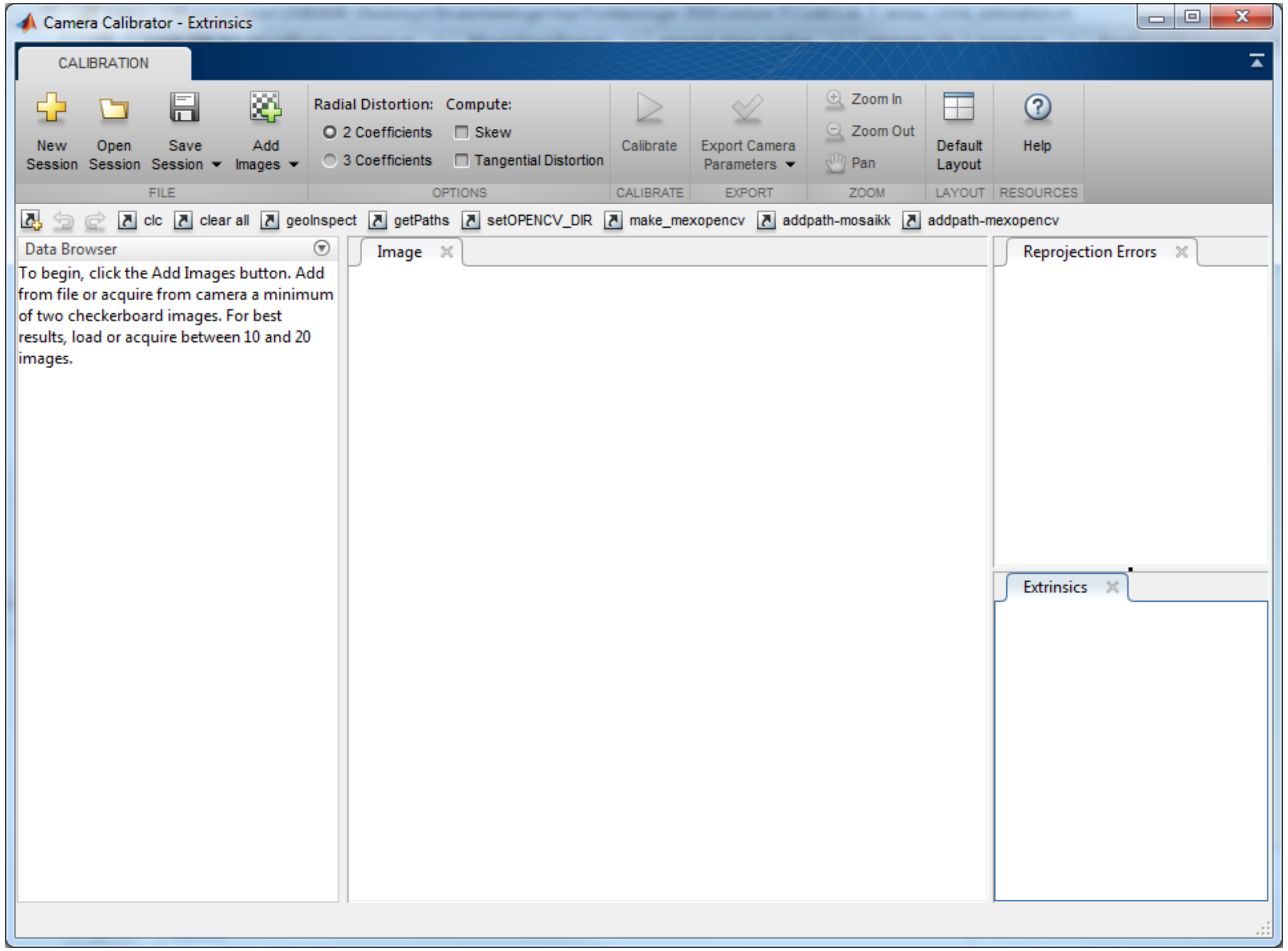
- Matlab
  - App: Camera Calibrator




  - This opens a new window

# Camera calibration in practice

- Add Images and specify the size of the chessboard squares
  - Chessboard detection starts

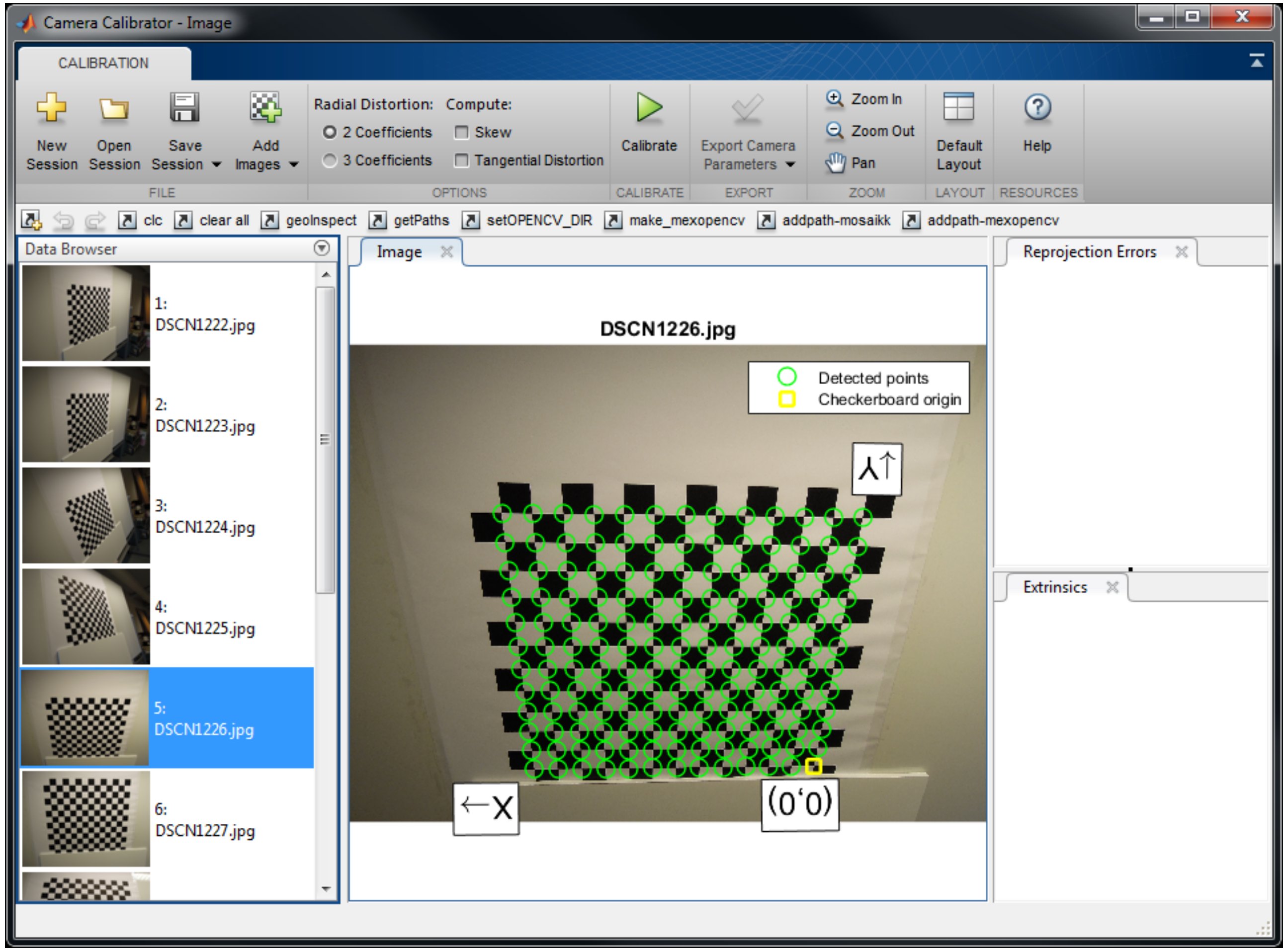


# Camera calibration in practice

- Add Images and specify the size of the chessboard squares
  - Chessboard detection starts

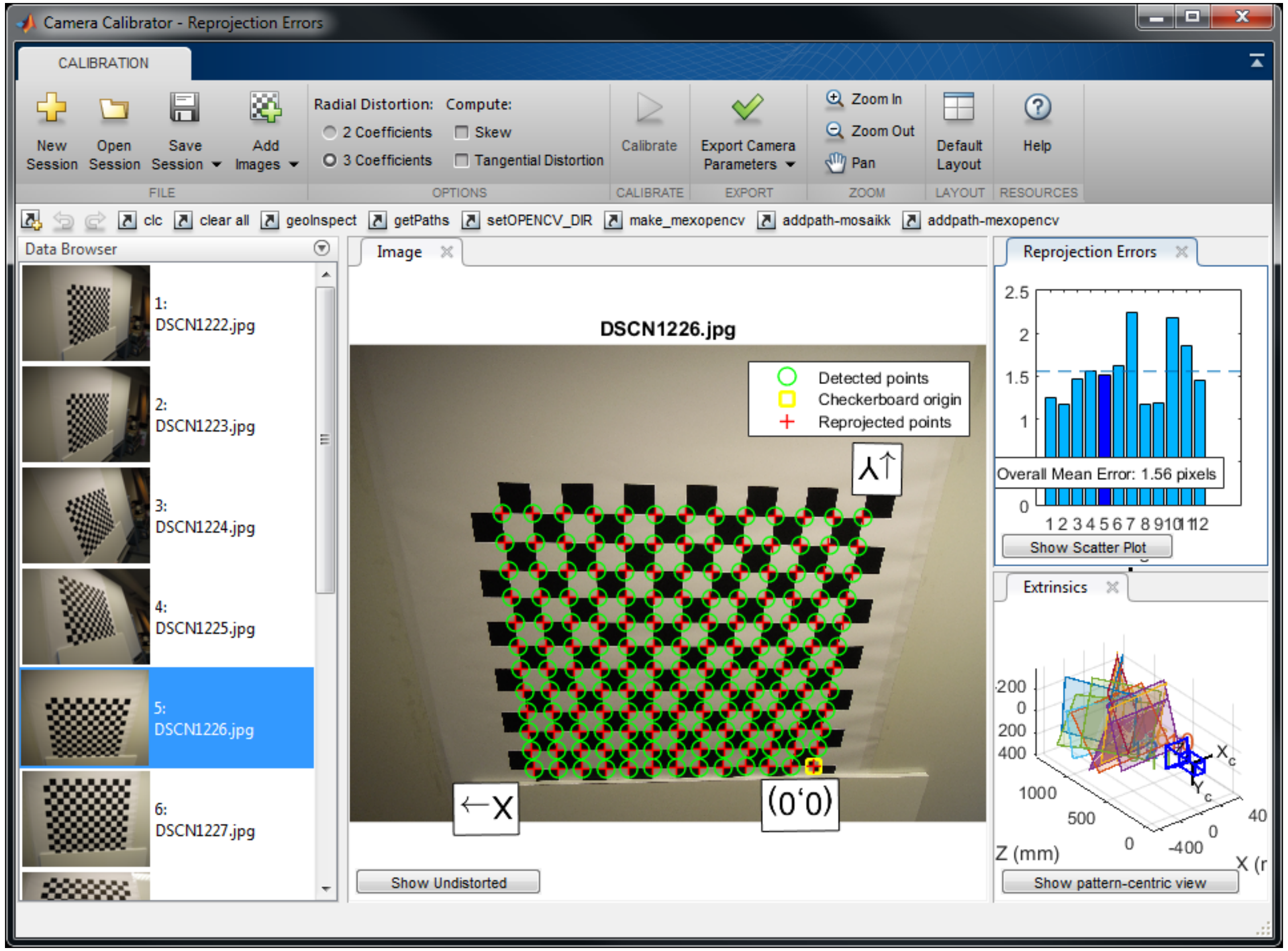
- Inspect the correctness of detection and choose what to estimate
  - 2/3 radial distortion coeffs?
  - Tangential distortion?
  - Skew?

- Calibrate



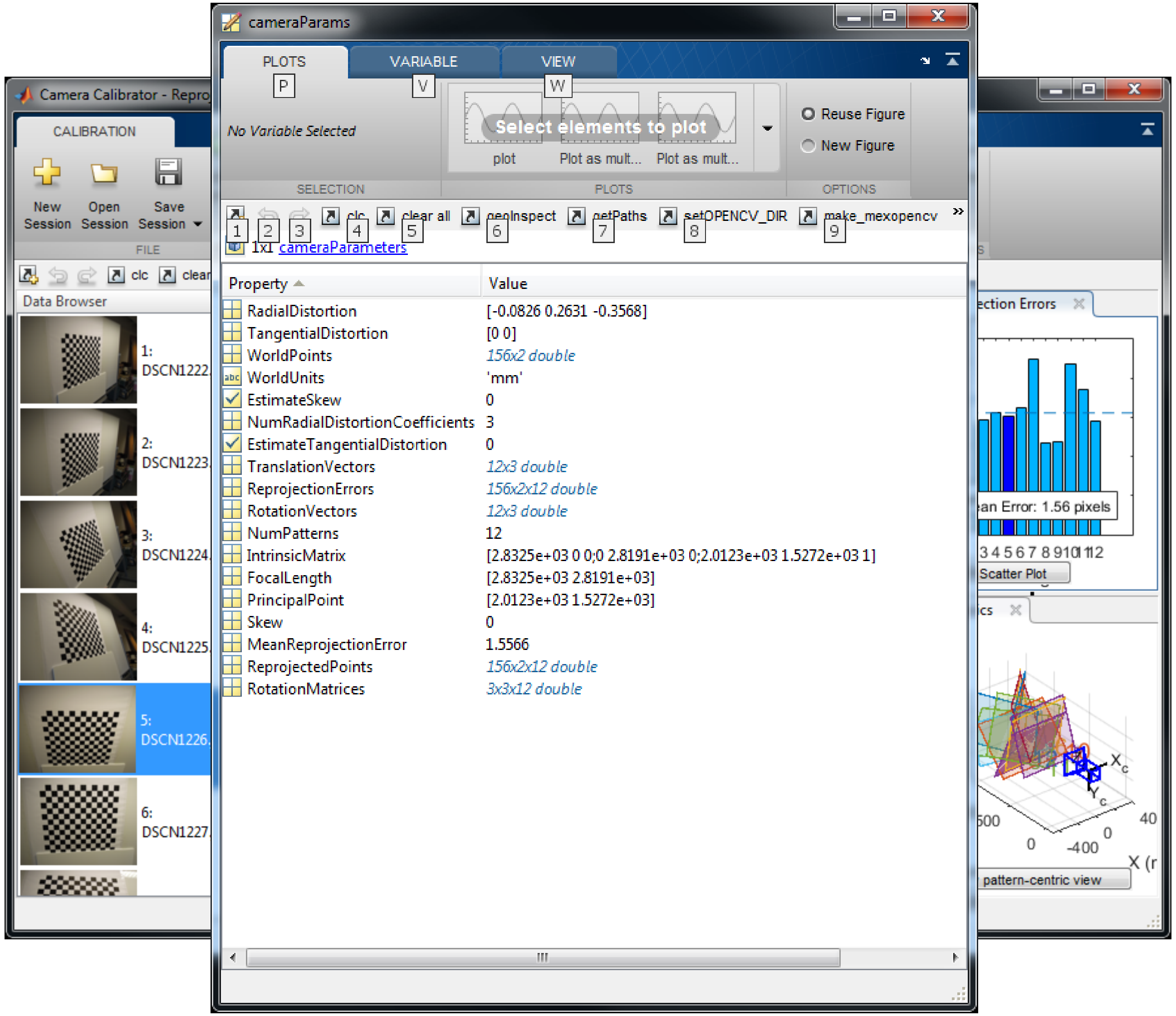
# Camera calibration in practice

- Add Images and specify the size of the chessboard squares
  - Chessboard detection starts
- Inspect the correctness of detection and choose what to estimate
  - 2/3 radial distortion coeffs?
  - Tangential distortion?
  - Skew?
- Calibrate
- Export to a Matlab variable



# Camera calibration in practice

- Add Images and specify the size of the chessboard squares
  - Chessboard detection starts
- Inspect the correctness of detection and choose what to estimate
  - 2/3 radial distortion coeffs?
  - Tangential distortion?
  - Skew?
- Calibrate
- Export to a Matlab variable



# Summary

- Calibration
  - Zhangs method using planar 3D-points
  - Matlab app
  - DLT method: Est $P$, decompose into $K[R \quad \boldsymbol{t}]$
- Undistortion
  - For geometrical computations we work on undistorted images/feature points

- Additional reading
  - Szeliski: 6.3
- Optional reading
  - *A flexible new technique for camera calibration*, by Z. Zhang