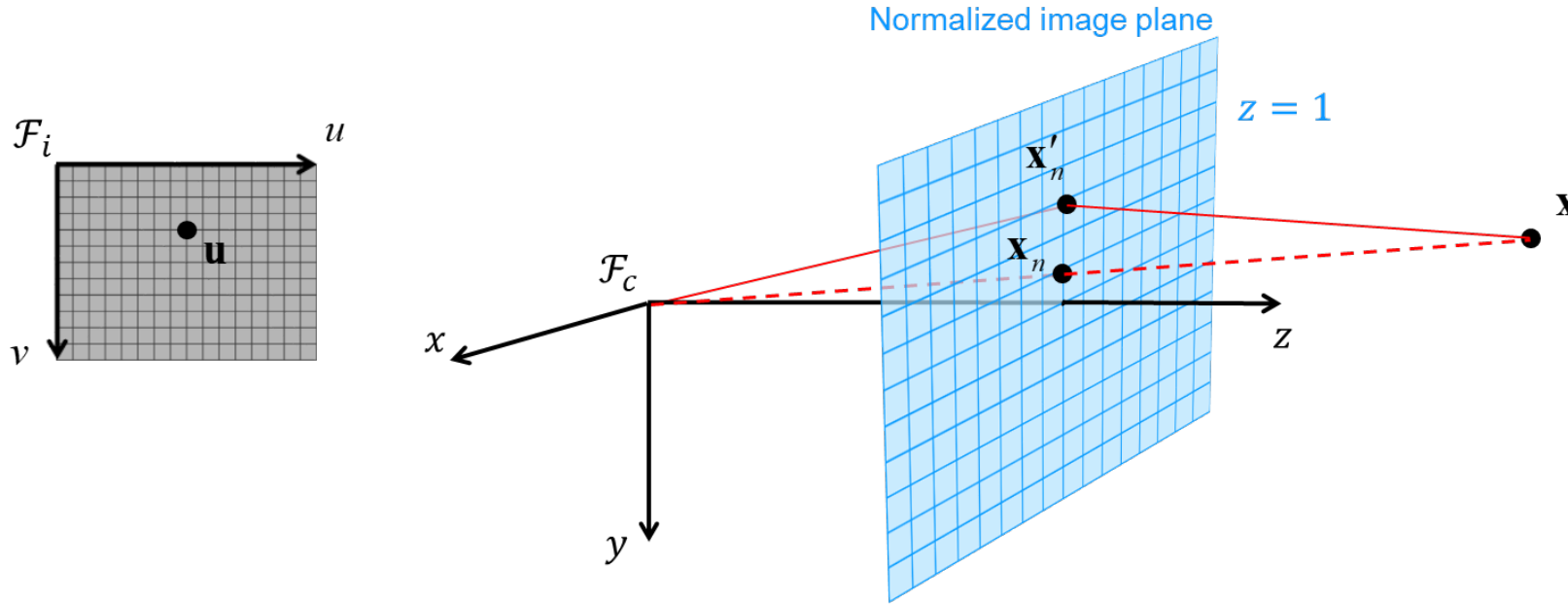


Camera calibration

Thomas Opsahl



Calibration



Perspective camera model

$$\tilde{\mathbf{u}} = \begin{bmatrix} \mathbf{K} & \mathbf{\Pi}_0 \\ f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{x}}^c$$

Distortion model (example)

$$\begin{aligned} x'_n &= x_n(1 + k_1 r_n^2 + k_2 r_n^4) \\ y'_n &= y_n(1 + k_1 r_n^2 + k_2 r_n^4) \end{aligned}$$

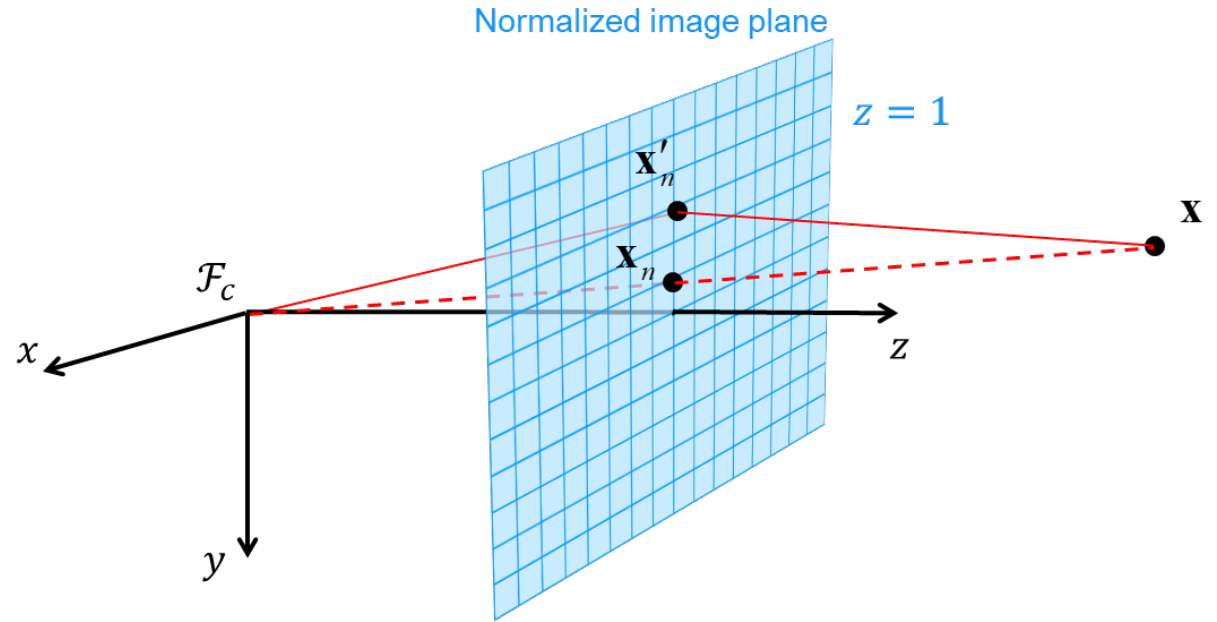
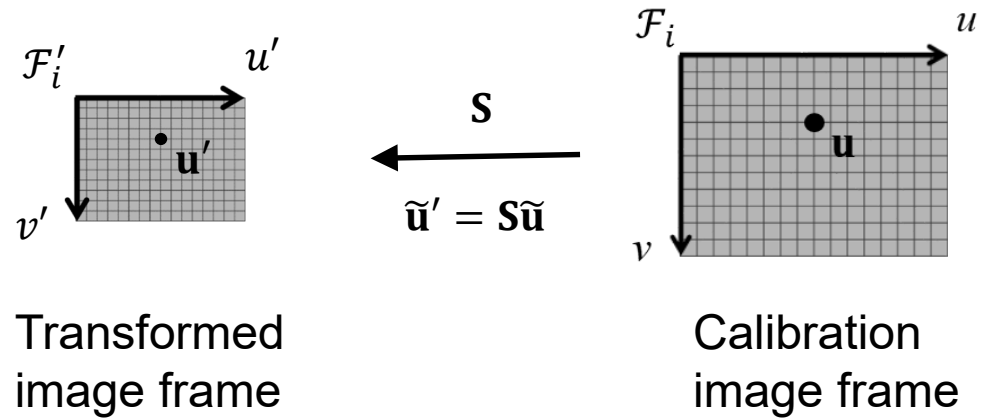
where $r_n^2 = x_n^2 + y_n^2$

- A process for estimating a camera's intrinsic parameters
 - Focal length, principal point, skew
 - Distortion coefficients

Calibration

- The estimated camera model corresponds to the physical condition of the camera during the calibration and the image size used in the calibration
- If the camera changes, the camera model also changes!
- Most changes to the camera optics will require a recalibration
 - Changing the zoom level of a variable zoom lens
 - Swapping one camera lens for another
 - Changing the focus is OK
- Working on rescaled/cropped (or otherwise transformed) images is OK as long as we adjust the camera model accordingly
 - Rescaling/cropping of images are common in image processing

Calibration



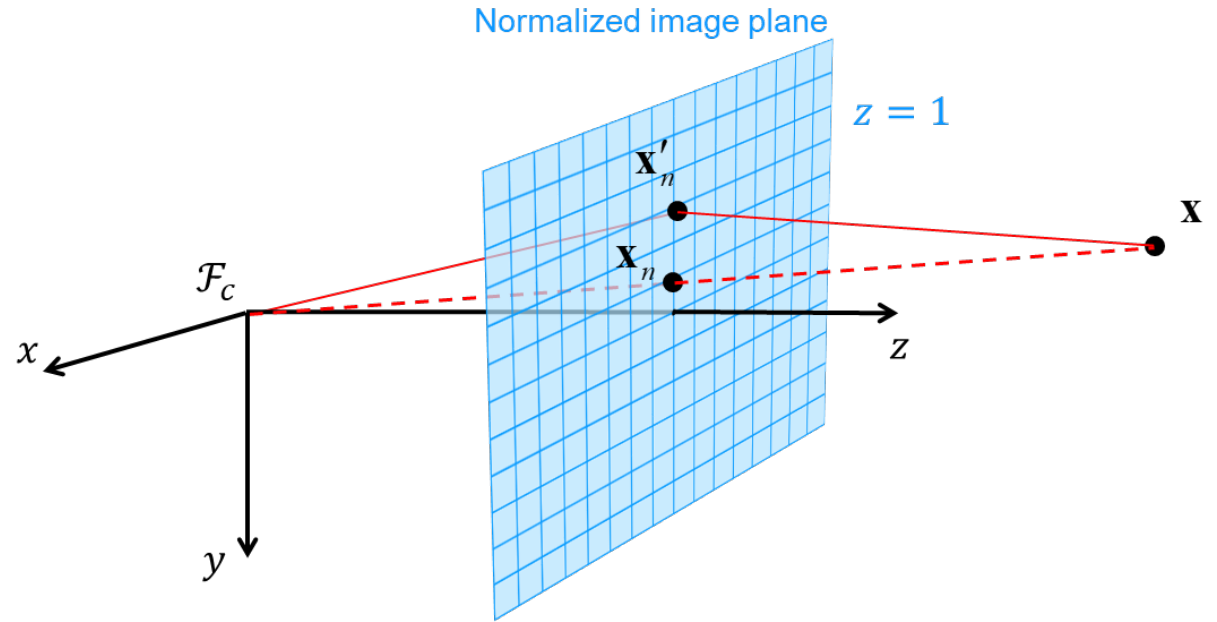
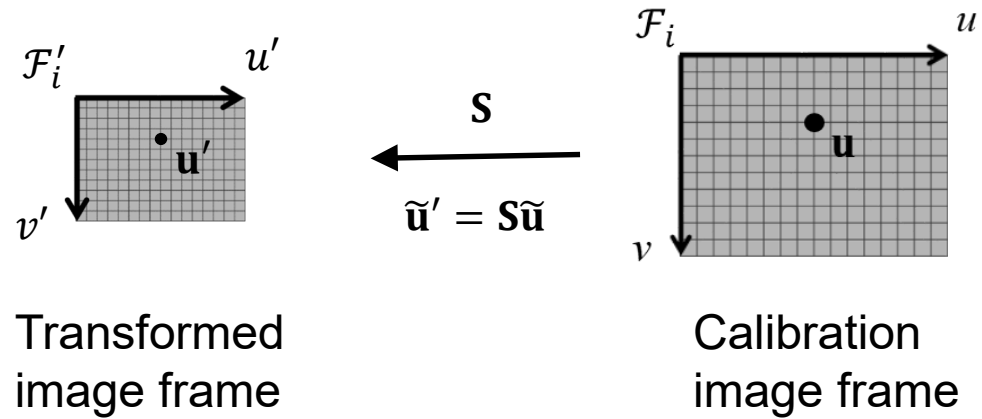
Original camera model

$$\tilde{u} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$

Adjusted camera model

$$\tilde{u}' = \mathbf{S}\mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$

Calibration



Original camera model

$$\tilde{u} = \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$

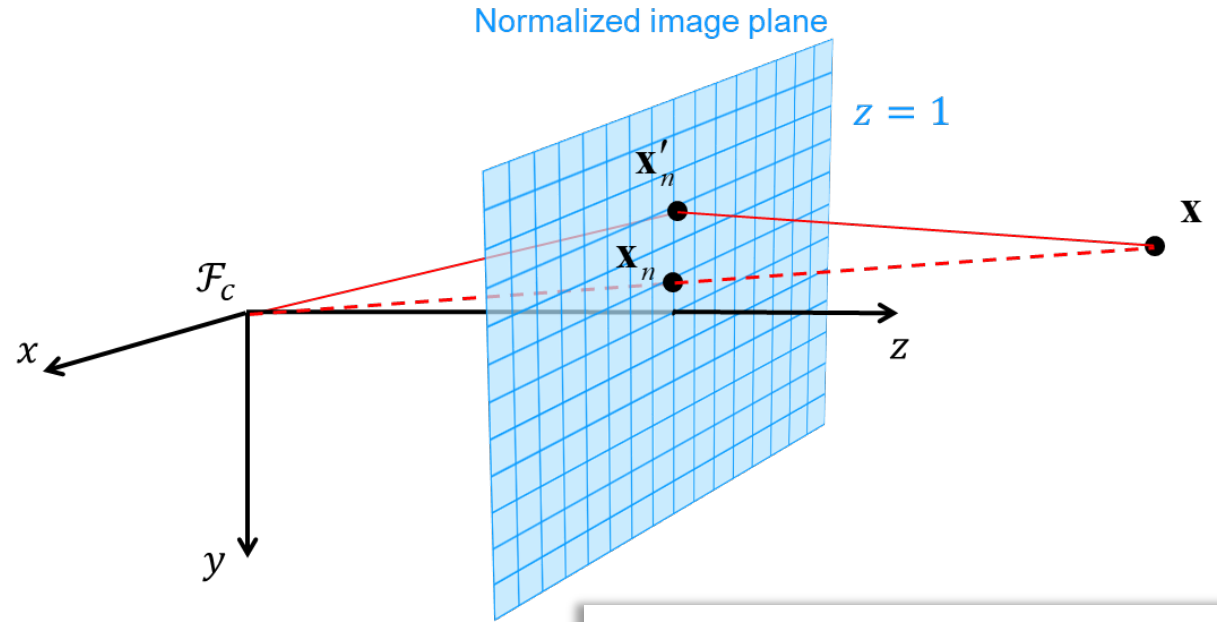
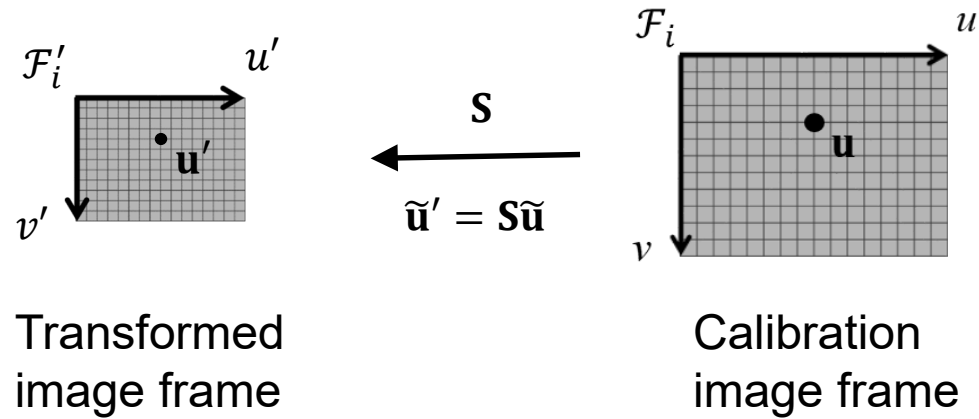
Distortion model

Adjusted camera model

$$\tilde{u}' = \mathbf{S} \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$

Distortion model (unchanged)

Calibration



Adjusted camera matrix
 $K' = SK$

Original camera model

$$\tilde{u} = K \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$

Distortion model

Adjusted camera model

$$\tilde{u}' = SK \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{x}^c$$


Distortion model (unchanged)

Example

- Say that we want to work on images that are half the size as those we used for calibration
- Then we can easily determine the corresponding camera model

Adjusted camera model

Distortion model from calibration

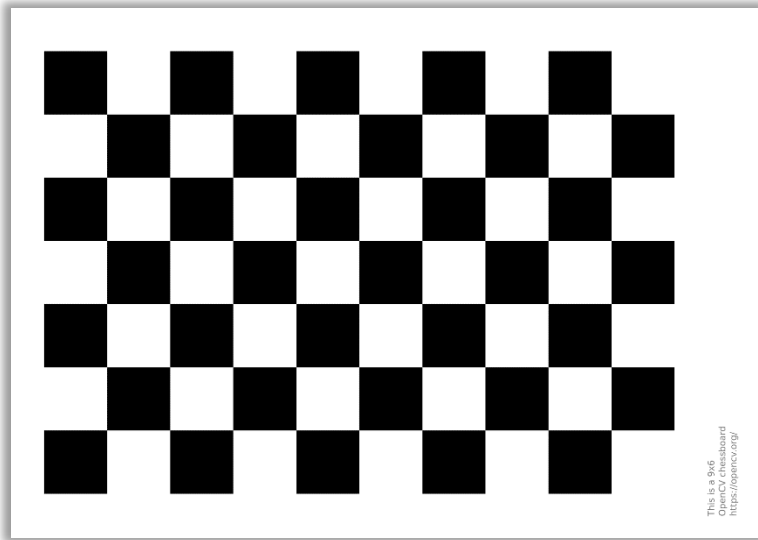
$$\tilde{\mathbf{u}}' = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tilde{\mathbf{x}}^c$$


Adjusted camera matrix

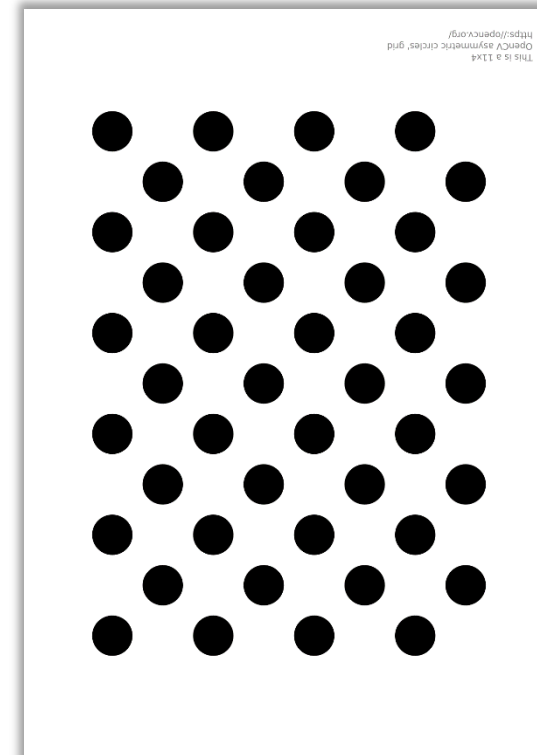
$$\mathbf{K}' = \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.5 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 0.5f_u & 0.5s & 0.5c_u \\ 0 & 0.5f_v & 0.5c_v \\ 0 & 0 & 1 \end{bmatrix}$$

Calibration target

- The calibration process is typically based on the detection of a calibration target
- The calibration target is typically chosen so that it is relatively easy to detect in images
 - Planar black/white pattern of corners or circles



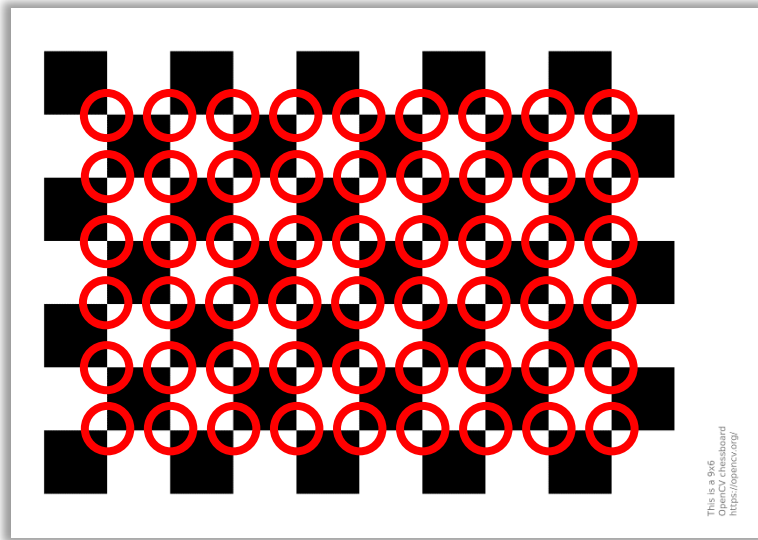
9 x 6 grid of corners
(in a 10 x 7 grid chessboard)



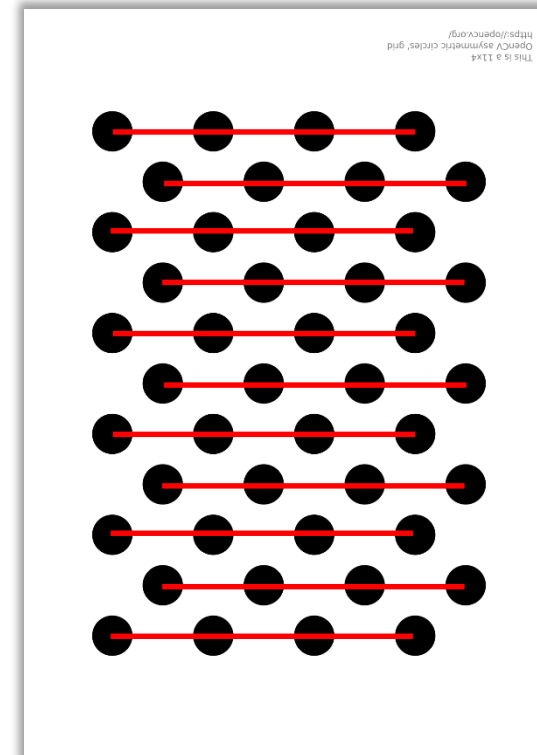
4 x 11 grid of circles

Calibration target

- The calibration process is typically based on the detection of a calibration target
- The calibration target is typically chosen so that it is relatively easy to detect in images
 - Planar black/white pattern of corners or circles



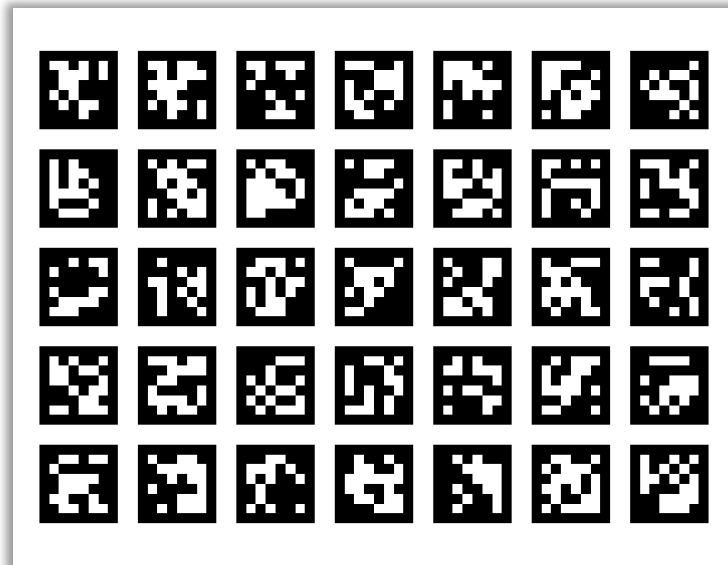
9 x 6 grid of corners
(in a 10 x 7 grid chessboard)



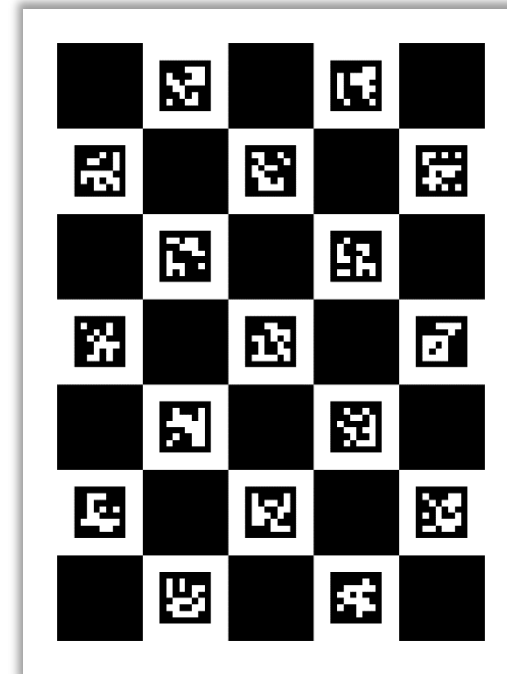
4 x 11 grid of circles

Calibration target

- Some targets, like chessboards, have to be fully detected in order to be used in calibration
 - Partial detections cause ambiguity
- Other targets have been designed for partial detection
 - AprilTags, ChArUco boards



AprilTags



ChArUco board

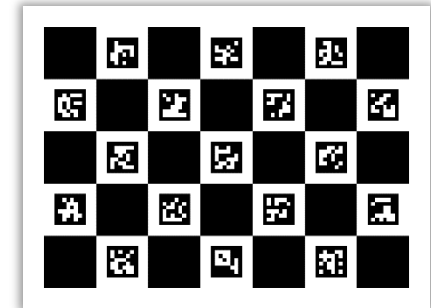
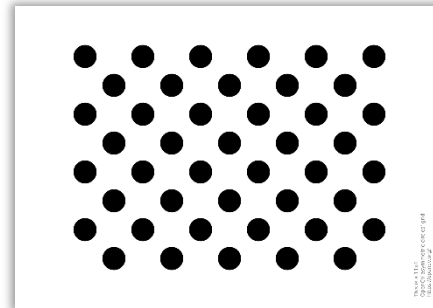
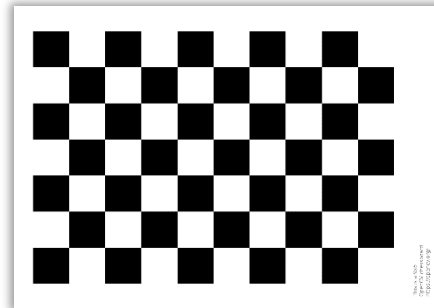
Camera calibration with OpenCV

- OpenCV comes with an application called **opencv_interactive-calibration** ([link](#))
 - We can run it from the terminal
 - Parameter list available through “-help”

```
opencv_interactive-calibration -help
```

- Easy to use
 - Incremental calibration in real-time

- Flexible
 - Chessboard
 - Circles
 - ChArUco



Camera calibration with OpenCV

- The general OpenCV distortion model has 12 parameters

$$\begin{aligned}x'_n &= \frac{1 + k_1 r_n^2 + k_2 r_n^4 + k_3 r_n^6}{1 + k_4 r_n^2 + k_5 r_n^4 + k_6 r_n^6} x_n + 2p_1 x_n y_n + p_2 (r_n^2 + 2x_n^2) + s_1 r_n^2 + s_2 r_n^4 \\y'_n &= \frac{1 + k_1 r_n^2 + k_2 r_n^4 + k_3 r_n^6}{1 + k_4 r_n^2 + k_5 r_n^4 + k_6 r_n^6} y_n + p_1 (r_n^2 + 2y_n^2) + 2p_2 x_n y_n + s_3 r_n^2 + s_4 r_n^4\end{aligned}$$

- The distortion model used in `opencv_interactive-calibration` only has 5
 - k_1, k_2, p_1, p_2 and k_3

$$\begin{aligned}x'_n &= (1 + k_1 r_n^2 + k_2 r_n^4 + k_3 r_n^6) x_n + 2p_1 x_n y_n + p_2 (r_n^2 + 2x_n^2) \\y'_n &= (1 + k_1 r_n^2 + k_2 r_n^4 + k_3 r_n^6) y_n + p_1 (r_n^2 + 2y_n^2) + 2p_2 x_n y_n\end{aligned}$$

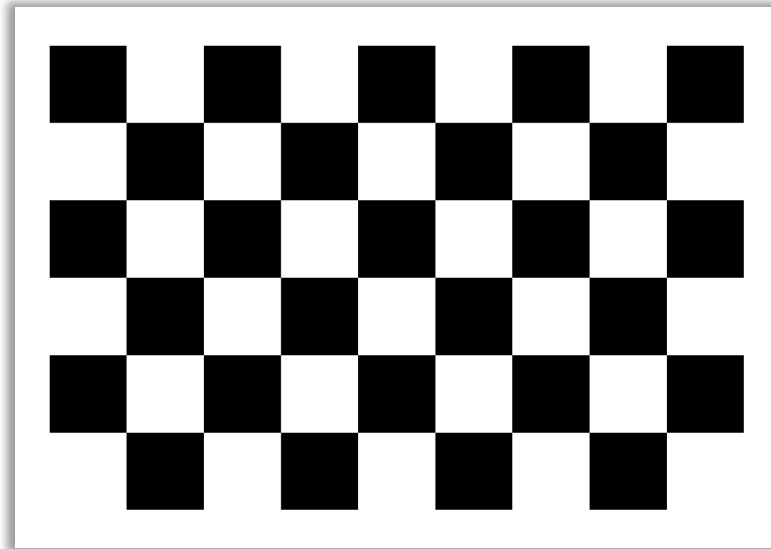
radial distortion
parameters k_1, k_2 and k_3

tangential distortion
parameters p_1 and p_2

$$r_n^2 = x_n^2 + y_n^2$$

A practical calibration example

The calibration target



- A chessboard with 9 columns and 6 rows of 30mm squares
 - Fits nicely on a sheet of A4 paper
- OpenCV regards this as a calibration target with 8 columns and 5 rows of corners

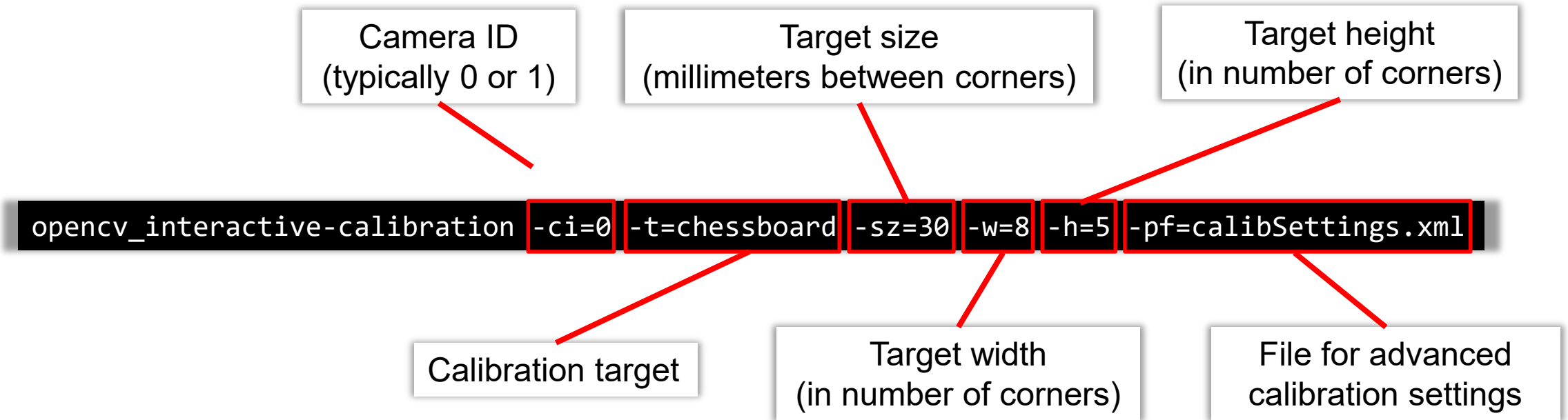
Camera calibration with OpenCV

- The application has several input parameters

```
opencv_interactive-calibration -ci=0 -t=chessboard -sz=30 -w=8 -h=5 -pf=calibSettings.xml
```

Camera calibration with OpenCV

- The application has several input parameters



Advanced calibration settings

- The application is rather well documented and explained online https://docs.opencv.org/master/d7/d21/tutorial_interactive_calibration.html
- This also provides an overview of the advanced parameters that we can adjust through the settings file calibSettings.xml
- Image resolution is one such parameter
 - We can for instance choose a 640x480 resolution by having the following calibSettings.xml file

```
<?xml version="1.0"?>  
<opencv_storage>  
<camera_resolution>640 480</camera_resolution>  
</opencv_storage>
```

- This way we can control the resolution for which our calibration is valid

Running the application

- Before running the application we need
 1. To have our calibration target available (on screen or on paper)
 2. To have a suitable calibSettings.xml file available (if we need it)
- Then we can run the application from the folder containing calibSettings.xml

```
opencv_interactive-calibration -ci=0 -t=chessboard -sz=30 -w=8 -h=5 -pf=calibSettings.xml
```

- Make sure that the size of the squares are 30 millimeters or change this input argument to match the chessboard you are using
- If we do not want to control any of the advanced parameters, we are free to remove -pf=calibSettings.xml from the argument list



Trash



Navigation: < > < Home repos calibration >

Recent

Home

Desktop

Documents

Name	Size	Modified
calibSettings.xml	104 bytes	00:37

```
thomasoo@thomasoo-VirtualBox: ~/repos/calibration
File Edit View Search Terminal Help
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

thomasoo@thomasoo-VirtualBox:~$ cd repos/calibration/
thomasoo@thomasoo-VirtualBox:~/repos/calibration$ opencv_interactive-calibration -ci=0 -t=chessboard -sz=30 -w=8 -h=5 -pf=calibSettings.xml
```

Calibration result

```
<?xml version="1.0"?>
<opencv_storage>
<calibrationDate>"on. 10. mars 2021 kl. 01.25 +0100"</
calibrationDate>
<framesCount>20</framesCount>
<cameraResolution>
  640 480</cameraResolution>
<cameraMatrix type_id="opencv-matrix">
  <rows>3</rows>
  <cols>3</cols>
  <dt>d</dt>
  <data>
    6.1422391521083796e+02 0. 3.1772008668253079e+02 0.
    6.1422391521083796e+02 2.4038977170644114e+02 0. 0. 1.</data></
cameraMatrix>
<cameraMatrix_std_dev type_id="opencv-matrix">
  <rows>4</rows>
  <cols>1</cols>
  <dt>d</dt>
  <data>
    0. 5.6150690725733521e-01 2.2719122638346670e-01
    2.3043130330783056e-01</data></cameraMatrix_std_dev>
<dist_coeffs type_id="opencv-matrix">
  <rows>1</rows>
  <cols>5</cols>
  <dt>d</dt>
  <data>
    1.1653720117454909e-01 -2.2464264350230950e-01 0. 0.
    6.9237300357090442e-02</data></dist_coeffs>
<dist_coeffs_std_dev type_id="opencv-matrix">
  <rows>5</rows>
  <cols>1</cols>
  <dt>d</dt>
  <data>
    4.4374533694545581e-03 2.7959811606570309e-02 0. 0.
    5.1540722400692247e-02</data></dist coeffs std dev>
<avg_reprojection_error>1.6552343368646225e-01</
avg reprojection error>
</opencv_storage>
```

The camera matrix $\mathbf{K} = \begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}$

Standard deviation of f_u , f_v , c_u and c_v

Distortion coefficients k_1 , k_2 , p_1 , p_2 and k_3

Standard deviation of distortion coefficients

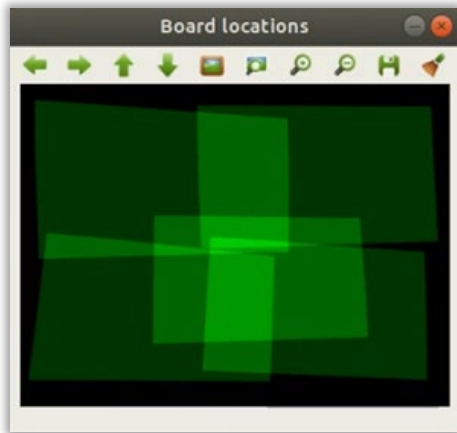
The average reprojection error over all corner points in all images (0.1-0.2 is OK)

Tips for a successful calibration

- Try holding the camera as steady as possible when capturing images of the calibration target.

Minimal motion blurr \Rightarrow High accuracy in corner detection \Rightarrow High accuracy in camera model

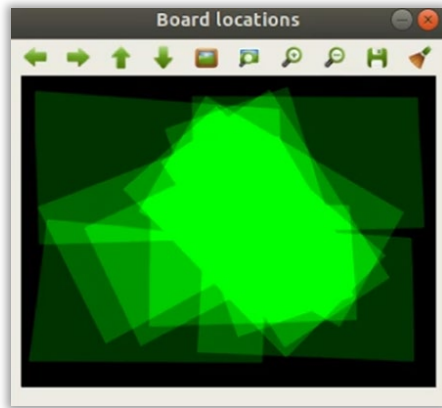
- Try to make sure that the calibration images cover all of the cameras field of view



Uncovered regions does not influence the estimation process, so the camera model might not fit very well there...

Tips for a successful calibration

- Try to sample the camera's field of view uniformly



Oversampling a region of the
camera's field of view



The camera model works very well there
but probably not equally well elsewhere...

- Make sure to vary ALL degrees of freedom between the camera and the calibration target
 - Vary the camera's/target's position and orientation
 - Don't forget to twist about the camera's optical axis

Good luck with your calibration!