

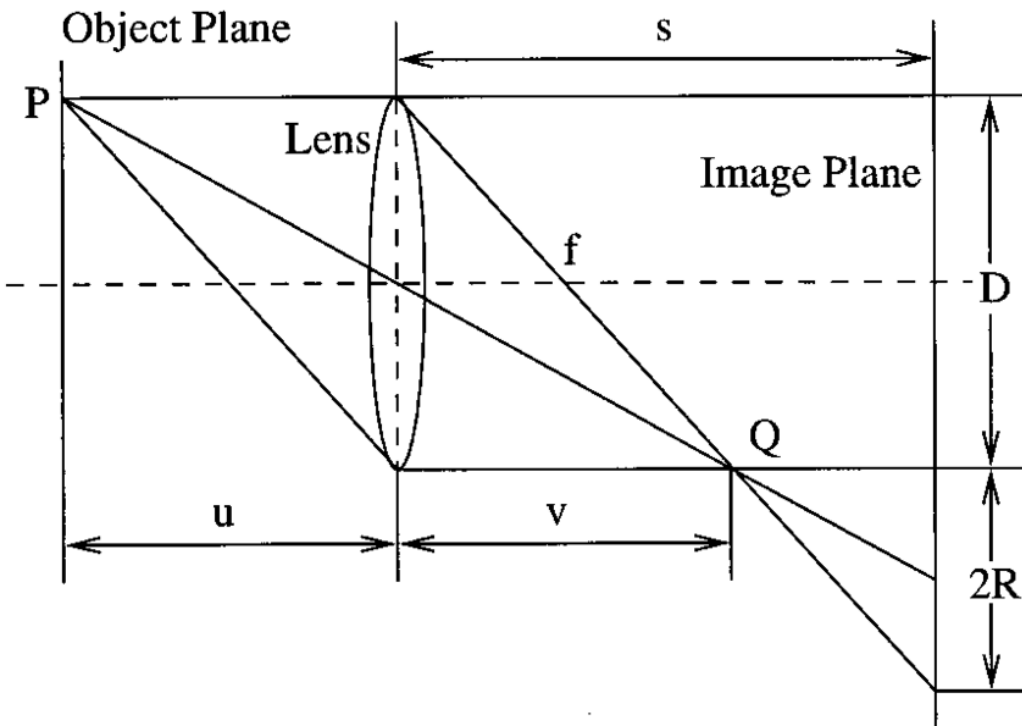


# Lecture 12

# Camera Calibration

ECE 1390/2390

# Camera Calibration



Ideal camera

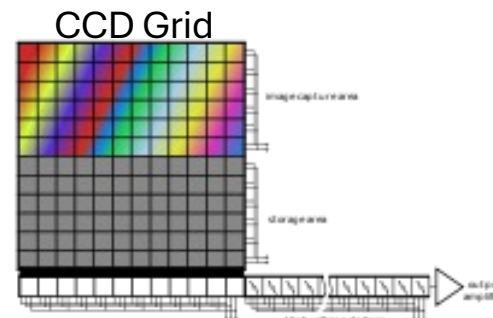
$$K = \begin{bmatrix} f_x & 0 & 0 \\ 0 & f_y & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Real camera

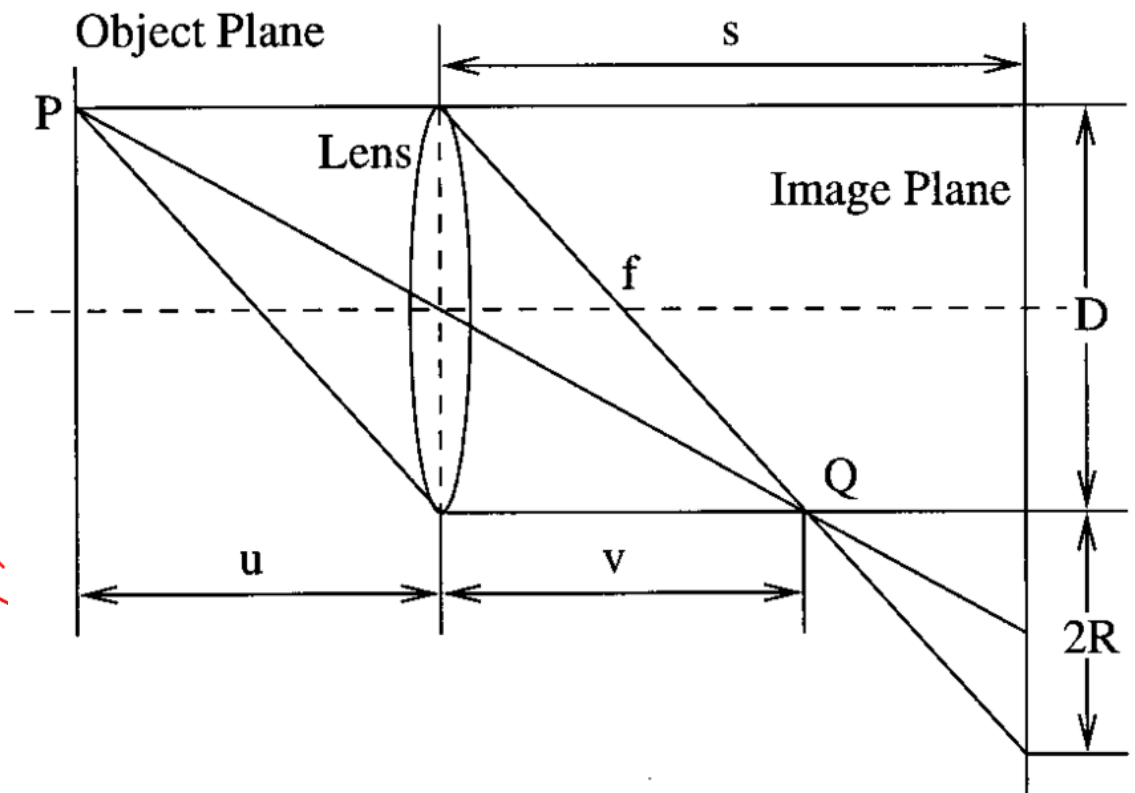
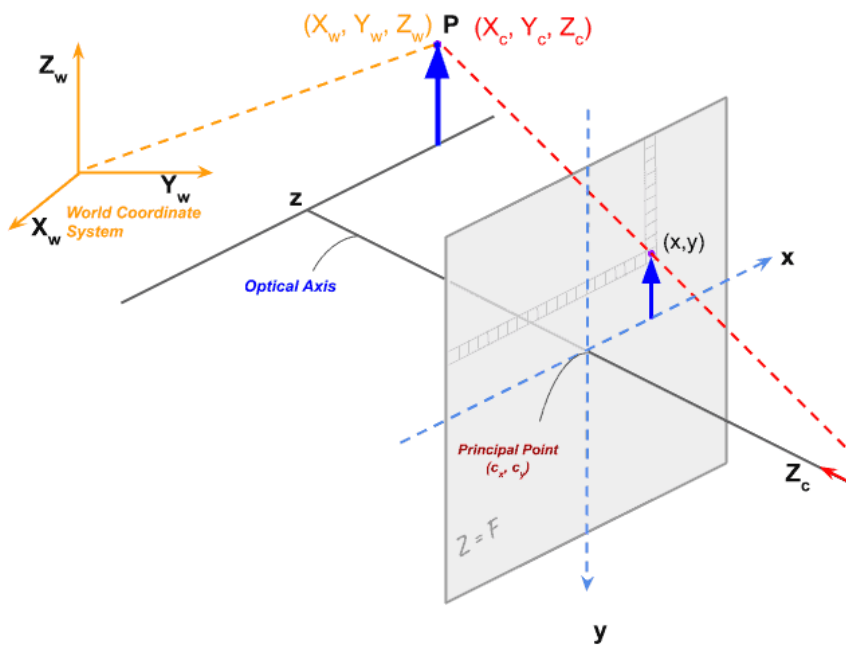
$$K = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

X/Y CCD grid skew

Offset of CCD center from optical axis



# Camera Calibration



# Fourier Optics

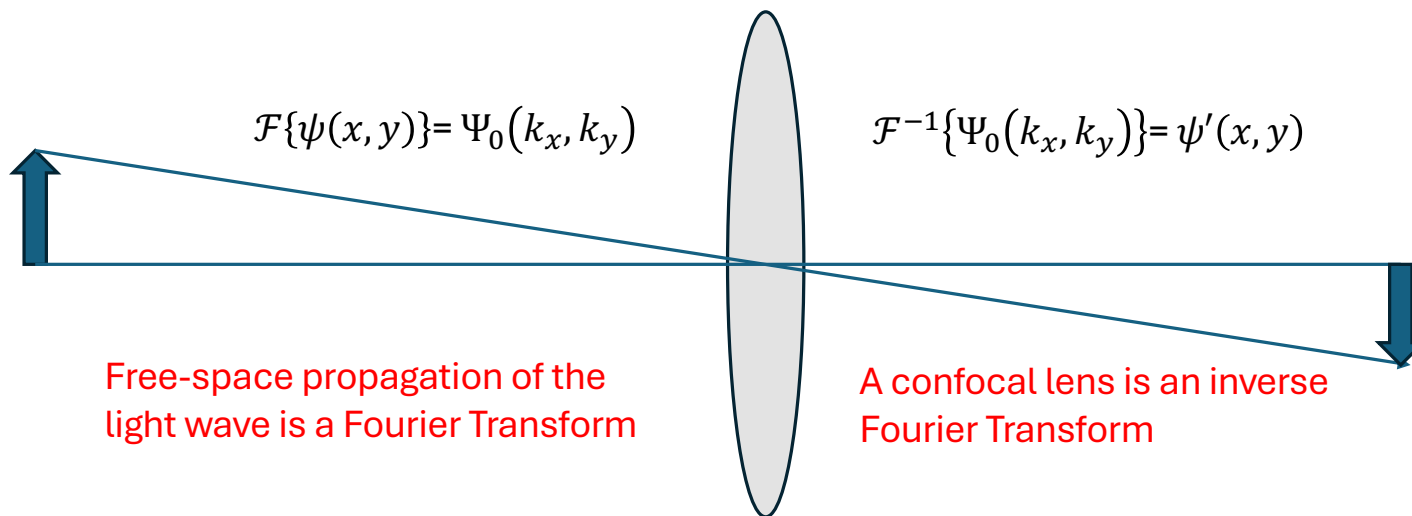
$$\left( \nabla^2 - \frac{1}{c^2} \frac{\partial^2}{\partial t^2} \right) u(r, t) = 0$$

$$\nabla^2 \psi = \frac{\partial^2 \psi}{\partial x^2} + \frac{\partial^2 \psi}{\partial y^2} + \frac{\partial^2 \psi}{\partial z^2} + \frac{\partial^2 \psi}{\partial t^2}$$

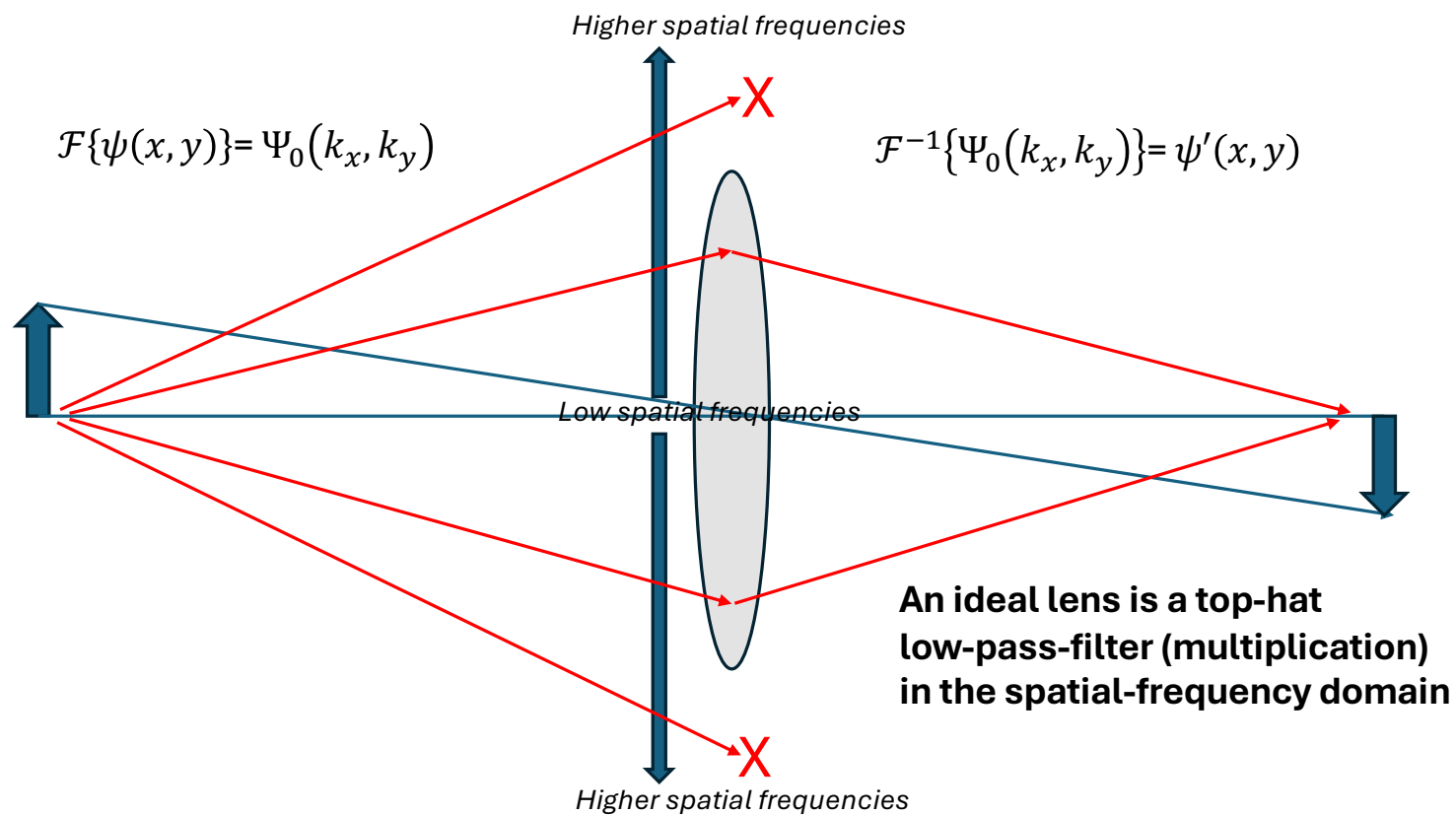
$$k_x = \frac{2\pi f_x}{c}$$

$$\psi(x, y) = \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} dk_x \cdot dk_y \cdot \Psi_0(k_x, k_y) e^{i \cdot (k_x x + k_y y)}$$

Solving for the spherical wave propagation at point (x,y)



# Fourier Optics



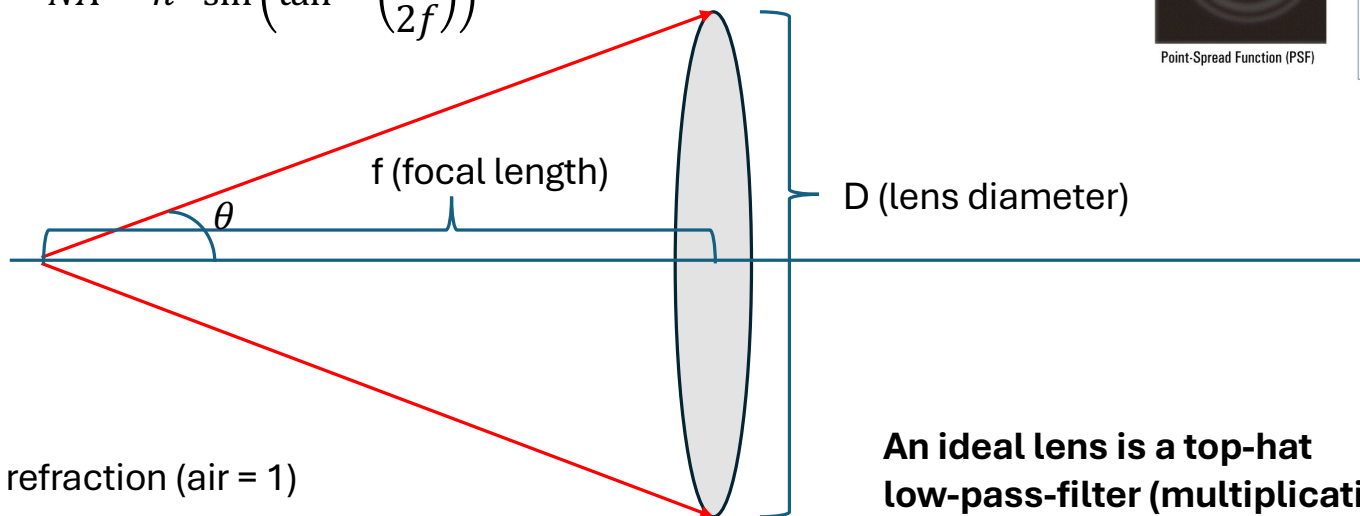
# Fourier Optics

## Numerical Aperture

$$NA = n \cdot \sin(\theta)$$

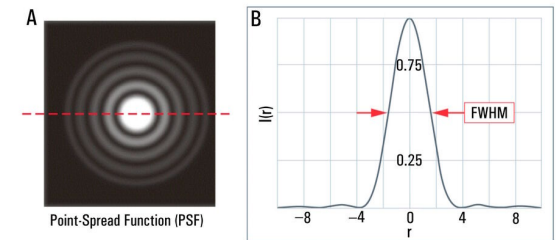
$$NA = n \cdot \sin\left(\tan^{-1}\left(\frac{D}{2f}\right)\right)$$

$n$  = Index of refraction (air = 1)



$$\Delta x_{min} = \frac{\lambda}{2 \cdot NA}$$

Diffraction resolution  
limit of a microscope



**An ideal lens is a top-hat  
low-pass-filter (multiplication)  
in the spatial-frequency domain**

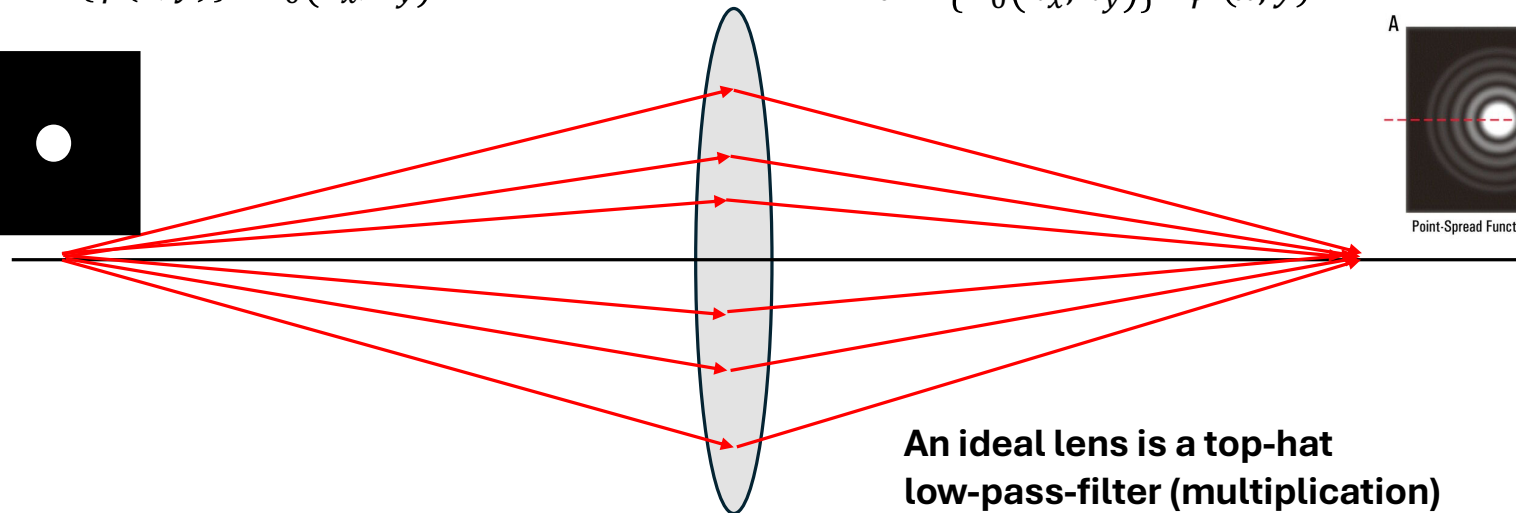
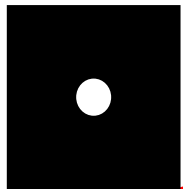
# Fourier Optics

$$I_{\text{photo}}(u, v) = K * \mathcal{F}^{-1}\{\mathcal{F}\{I_{\text{object}}(x, y)\} \cdot l(k_x, k_y, \lambda)\}$$

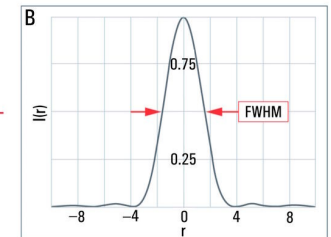
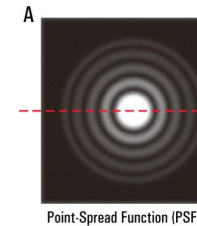
$$I_{\text{photo}}(u, v) = K * (I_{\text{object}}(x, y) \otimes L(x, y, \lambda))$$

$$\mathcal{F}\{\psi(x, y)\} = \Psi_0(k_x, k_y)$$

$$\mathcal{F}^{-1}\{\Psi_0(k_x, k_y)\} = \psi'(x, y)$$



**An ideal lens is a top-hat  
low-pass-filter (multiplication)  
in the spatial-frequency domain**



# Fourier Optics

Spherical aberration

Negative  
aberration

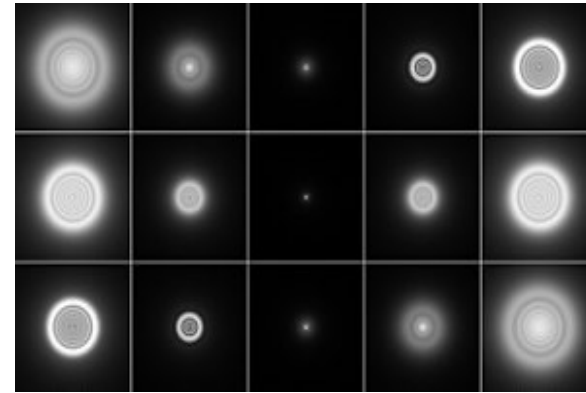
Ideal

Positive  
aberration

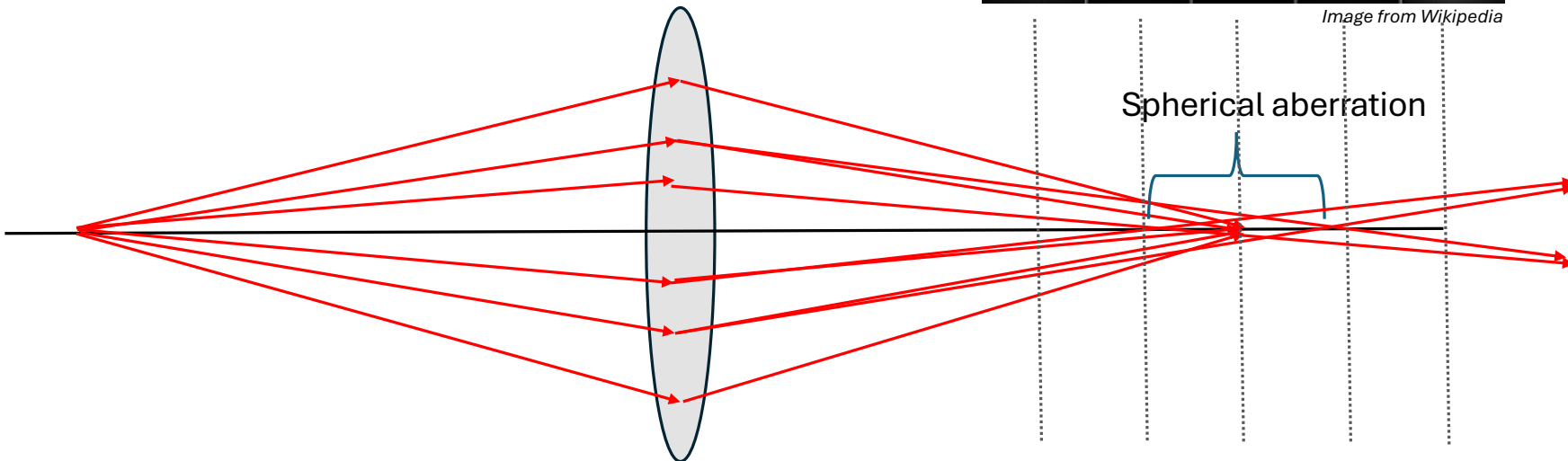
$\Delta f < 0$

$\Delta f = 0$

$\Delta f > 0$



*Image from Wikipedia*



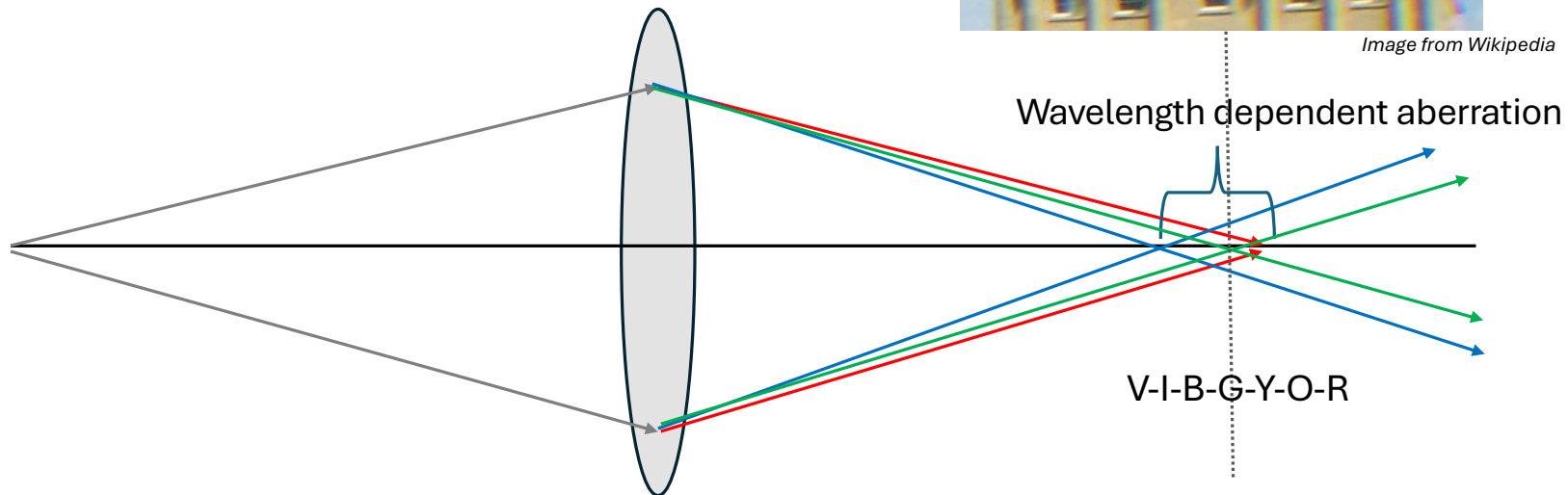


# Fourier Optics

## Chromatic aberration



*Image from Wikipedia*

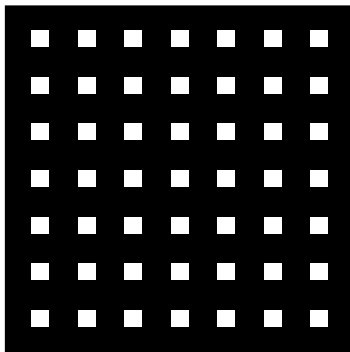


Deconvolution (Wiener Filter)

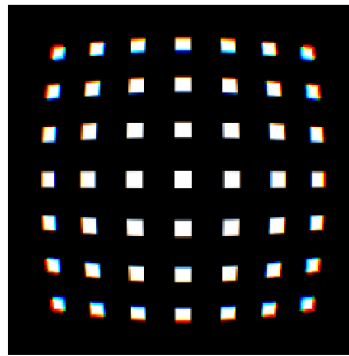
$$l(kx, ky, c) = \mathcal{F}\{I_{\text{Raw}}(u, v, c)\} / \mathcal{F}\{I_{\text{Target}}(u, v, c)\}$$

$$I_{\text{Corrected}}(x, y, c) = \mathcal{F}^{-1}\{\mathcal{F}\{I_{\text{Raw}}(u, v, c)\} / l(k_x, k_y, c)\}$$

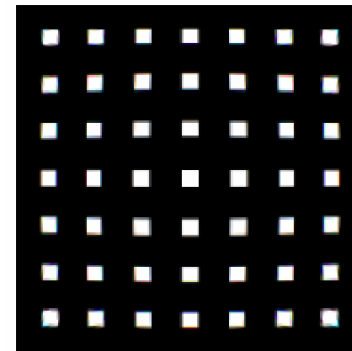
Calibration Target



Photograph Raw



Corrected



*Imperfect*

# OpenCV Camera Calibration

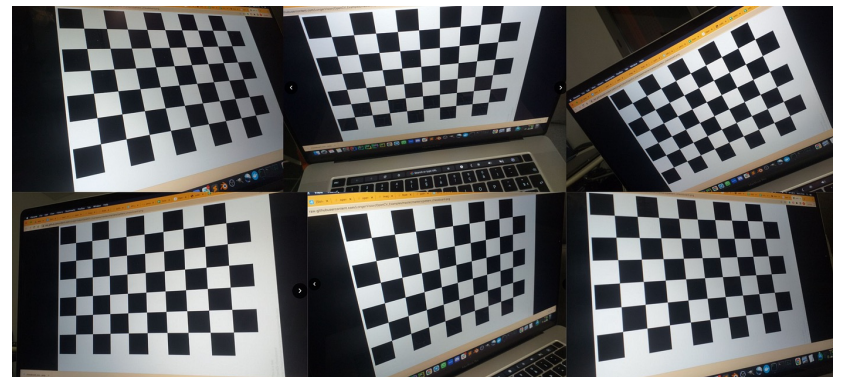
- Internal

- Focal
- Optical center
- Radial Distortion

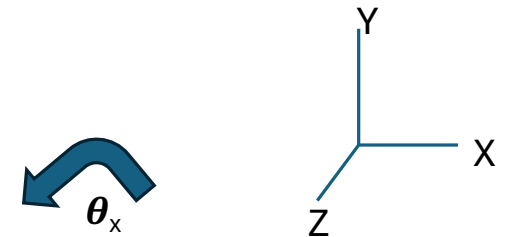
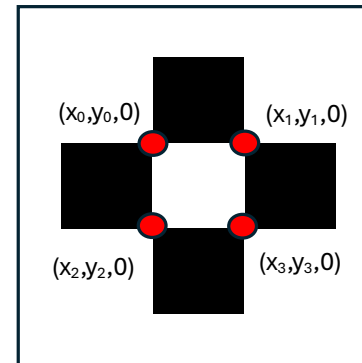
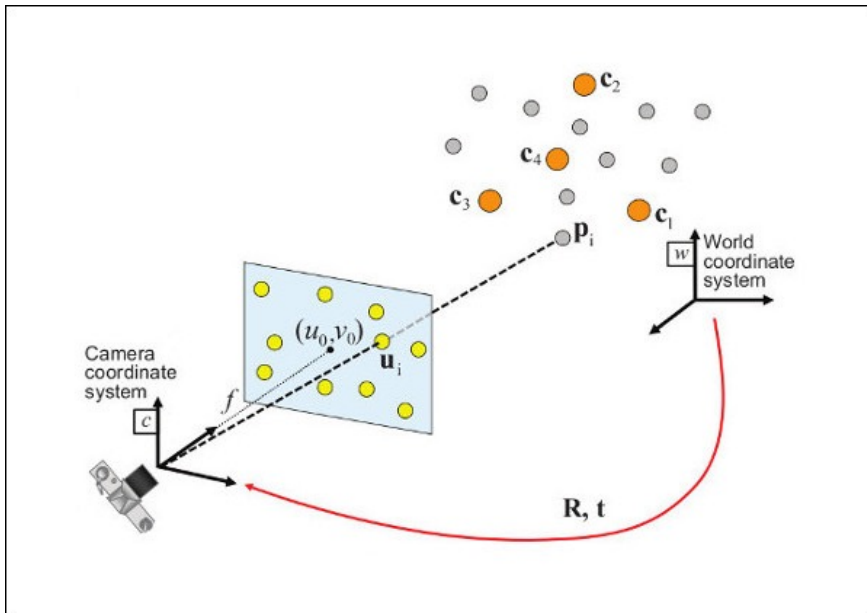
$$K = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}$$

- External

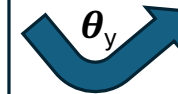
- Orientation of target relative to camera



# Prospective-n-Point (PnP)



$$B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x \\ 0 & \sin \theta_x & \cos \theta_x \end{bmatrix} A$$



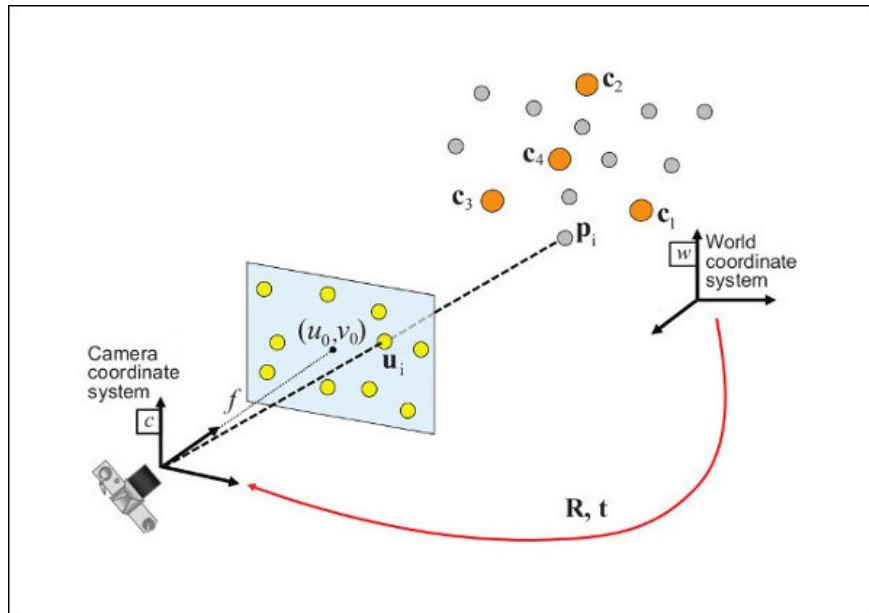
$$B = \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y \\ 0 & 1 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y \end{bmatrix} A$$



$$B = \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 \\ \sin \theta_z & \cos \theta_z & 0 \\ 0 & 0 & 1 \end{bmatrix} A$$

$$\begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix} = \begin{bmatrix} \cos \theta_z & -\sin \theta_z & 0 & 0 \\ \sin \theta_z & \cos \theta_z & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \theta_y & 0 & \sin \theta_y & 0 \\ 0 & 1 & 0 & 0 \\ -\sin \theta_y & 0 & \cos \theta_y & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos \theta_x & -\sin \theta_x & 0 \\ 0 & \sin \theta_x & \cos \theta_x & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \Delta X \\ 0 & 1 & 0 & \Delta Y \\ 0 & 0 & 1 & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

# Prospective-n-Point (PnP)



$$\begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \Delta X \\ r_{1,0} & r_{1,1} & r_{1,2} & \Delta Y \\ r_{2,0} & r_{2,1} & r_{2,2} & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

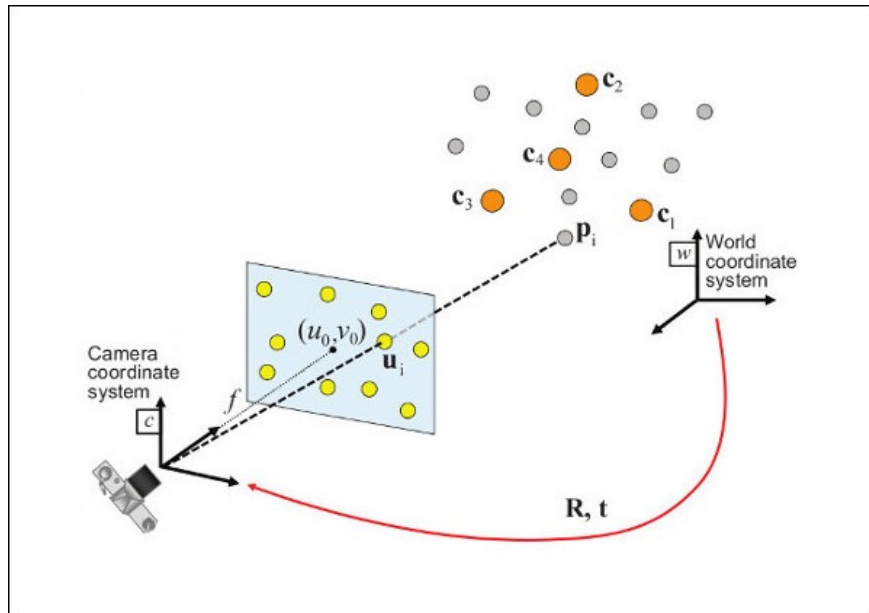
**3D to 2D projection**

$$\begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix}$$

**Intrinsic Camera distortion**

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix}$$

# Prospective-n-Point (PnP)



$$\begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \Delta X \\ r_{1,0} & r_{1,1} & r_{1,2} & \Delta Y \\ r_{2,0} & r_{2,1} & r_{2,2} & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

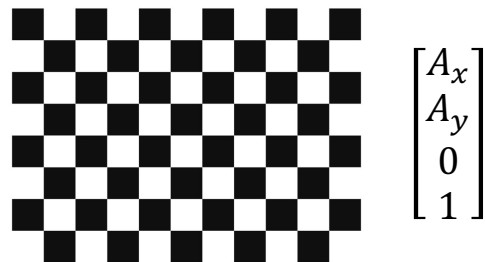
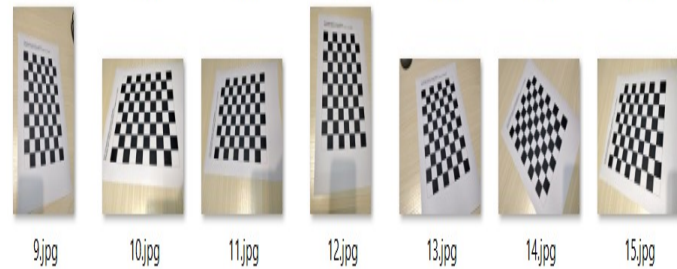
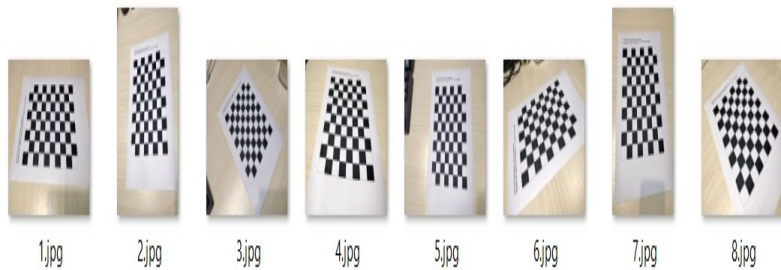
**3D to 2D projection**

$$\begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix}$$

**Intrinsic Camera distortion**

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix}$$

# Prospective-n-Point (PnP)



$$\begin{bmatrix} A_x \\ A_y \\ 0 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix} = \begin{bmatrix} r_{0,0} & r_{0,1} & r_{0,2} & \Delta X \\ r_{1,0} & r_{1,1} & r_{1,2} & \Delta Y \\ r_{2,0} & r_{2,1} & r_{2,2} & \Delta Z \\ 0 & 0 & 0 & 1 \end{bmatrix} * \begin{bmatrix} A_x \\ A_y \\ A_z \\ 1 \end{bmatrix}$$

*Differs per image*

**3D to 2D projection**

$$\begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} B_x \\ B_y \\ B_z \\ 1 \end{bmatrix}$$

*Same across images*

**Intrinsic Camera distortion**

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & \gamma & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u_{ideal} \\ v_{ideal} \\ 1 \end{bmatrix}$$

# OpenCV Camera Calibration

```
img_points = []  
obj_points = []
```

for gray in images:

Step 1.

Finds corners of checkerboard

```
retval, corners = cv2.findChessBoardCorners(image=gray, patternSize=(10,8))
```

(optional)

Estimates sub-pixel position of corners

```
corners = cv2.cornerSubPix(gray, corners, (11,11), (-1,-1), criteria)
```

(optional)

```
cv2.drawChessboardCorners
```

Step 2.

```
img_points.append(corners)
```

```
obj_points.append(real_coors)
```

← These two arrays need to have corresponding points

Step 3.

```
ret, cameramtx, distcorr, rot, trans = cv2.calibrateCamera(obj_points, img_points, gray.shape[:-1], None, None)
```

Camera (K) matrix

Distortion parameters

Estimated rotation/translation of each image



# OpenCV Camera Calibration

*# shape of image needed to design undistortion filter*

```
h,w = img.shape[:2]
```

*# make camera matrix and valid ROI*

```
newcameramtx, roi = cv2.getOptimalNewCameraMatrix(cameramtx, distcorr, (w,h), 1, (w,h))
```

From previous steps



*# undistort image*

```
dst = cv.undistort(img, mtx, dist, None, newcameramtx)
```

*# crop the image*

```
x,y,w,h = roi
```

```
dst = dst[y:y+h, x:x+w]
```