Image Formation

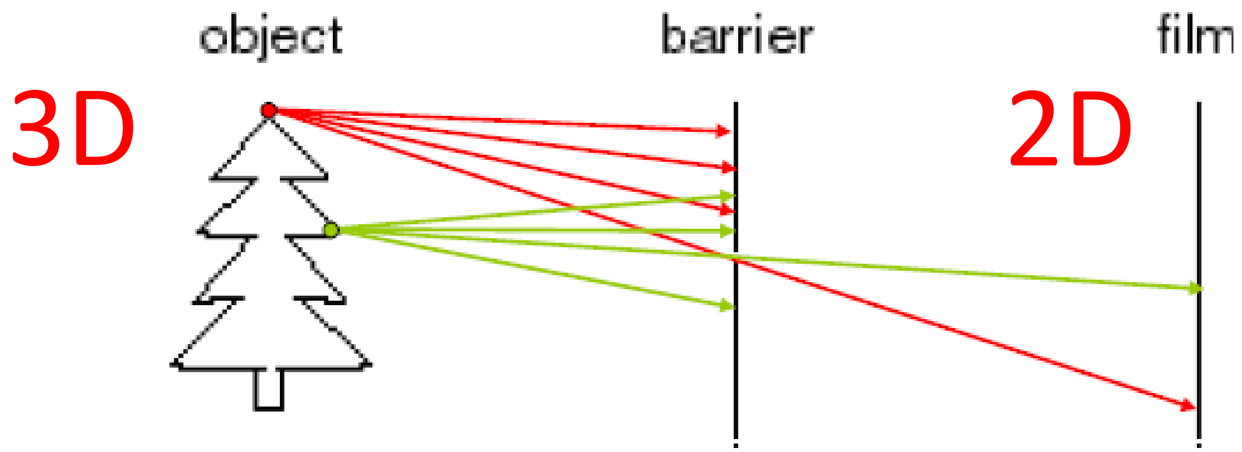
Lecture 02

Saket Anand

Outline

- Pinhole camera model
  - Need for Geometric Transformations
- Geometric Transformations – 2D and 3D
  - Scaling, Rotation and Translation
  - Homogeneous Coordinates
  - Perspective Projection
- Image Formation Process

Pinhole camera



Pinhole model:

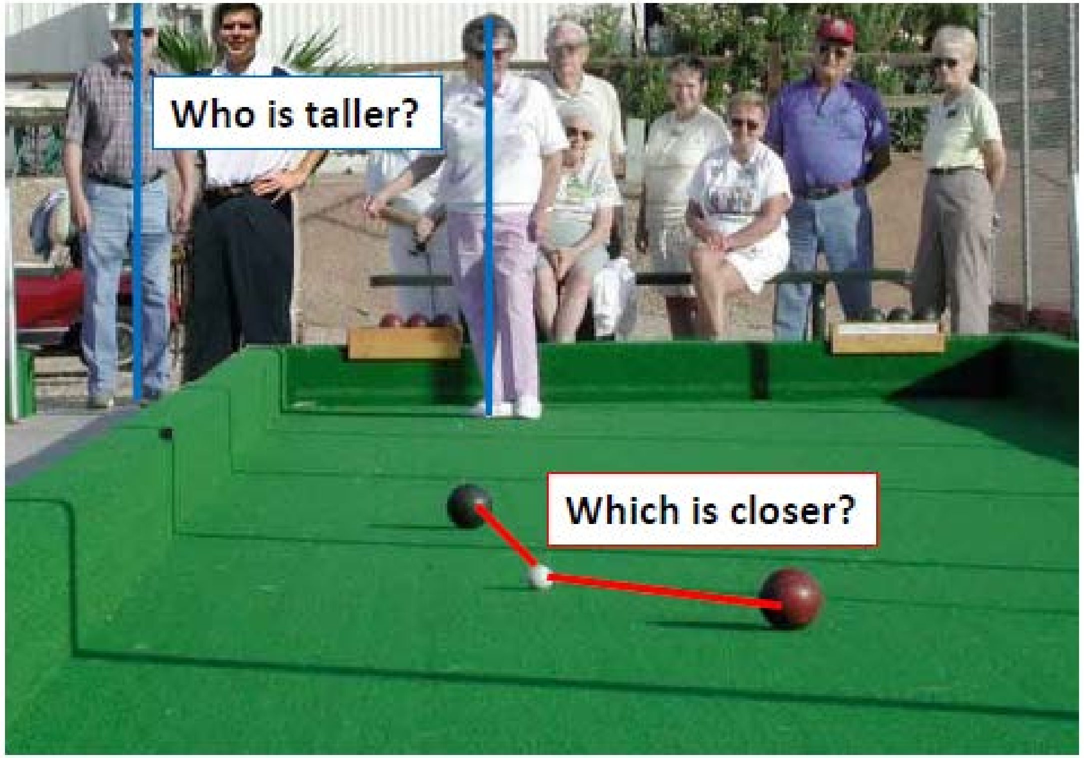
- Captures **pencil of rays** – all rays through a single point
- The point is called **Center of Projection (focal point)**
- The image is formed on the **Image Plane**

- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening known as the **aperture**

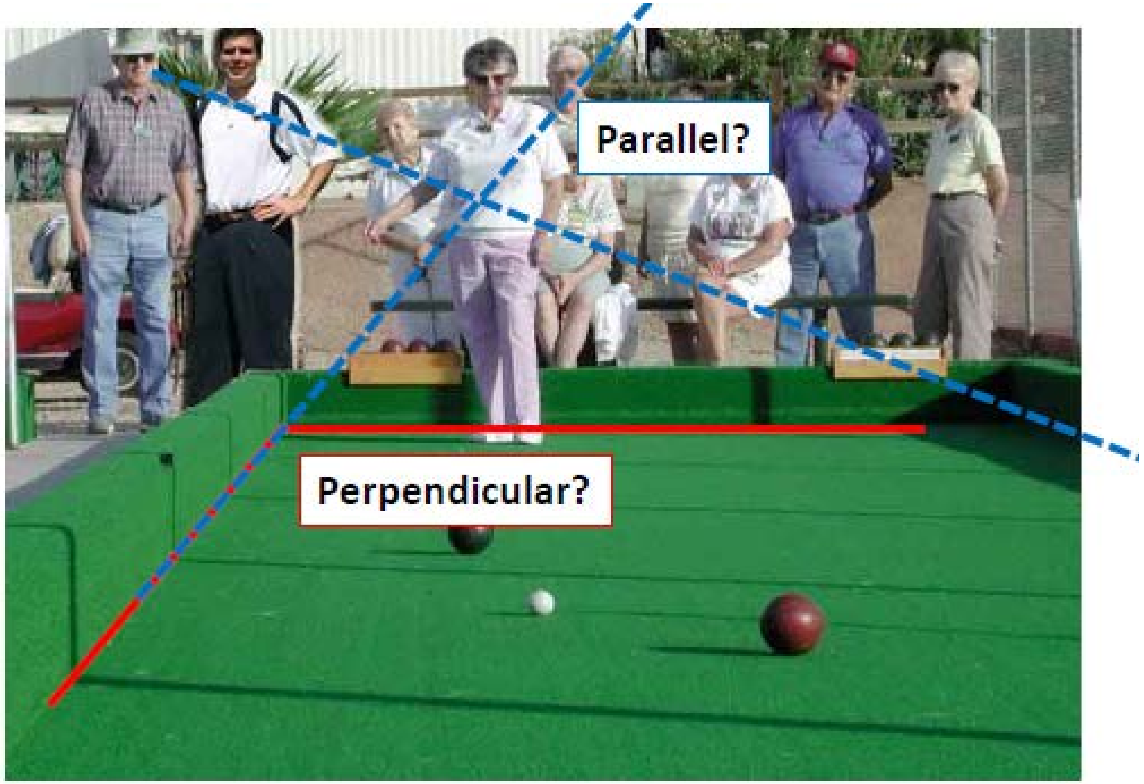
Object to Image (3D to 2D) transformation occurs via a **perspective projection**

Information Loss in Perspective Projection

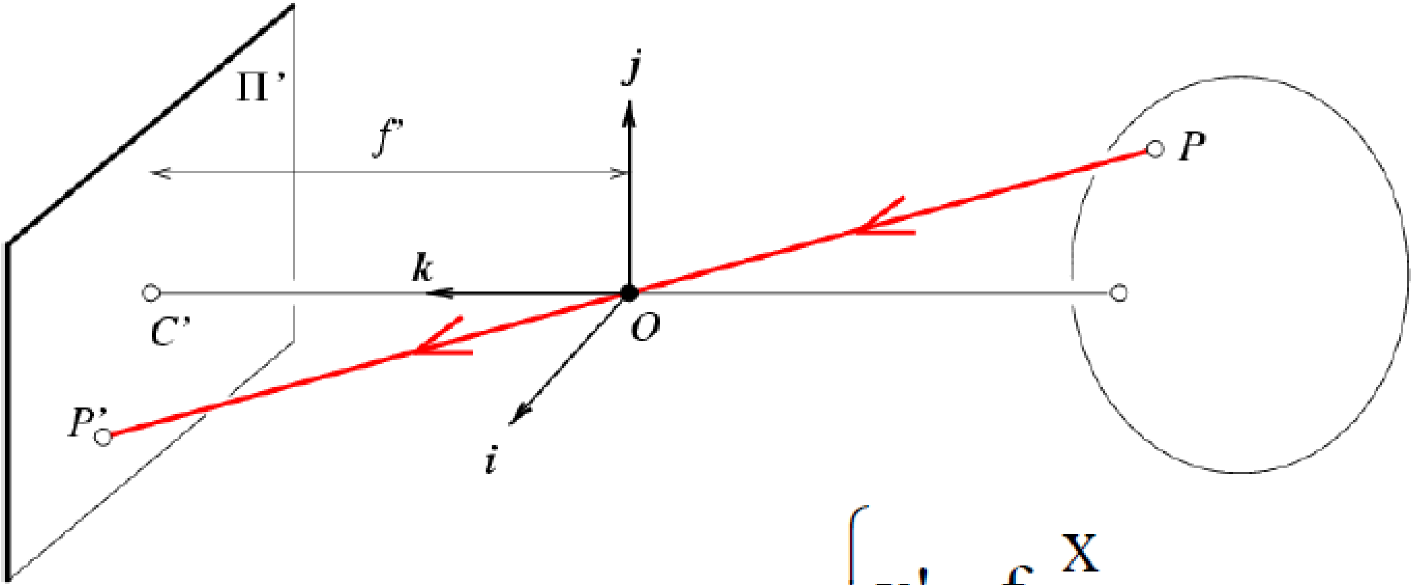
- Length and Distances



- Angles



Pinhole Camera Model



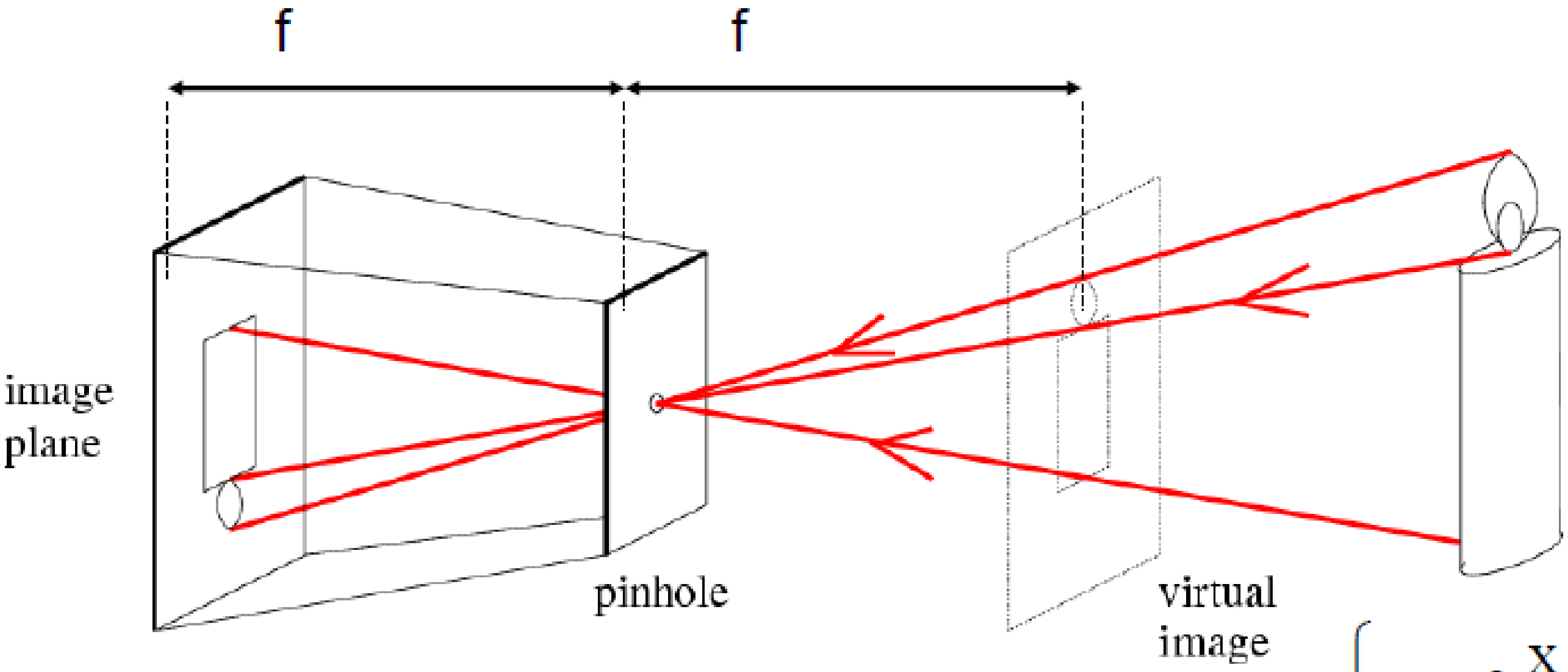
$$P = \begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow P' = \begin{bmatrix} x' \\ y' \end{bmatrix}$$

$$\begin{cases} x' = f \dfrac{x}{z} \\ y' = f \dfrac{y}{z} \end{cases}$$

nonhomogeneous coordinates

Derived using similar triangles

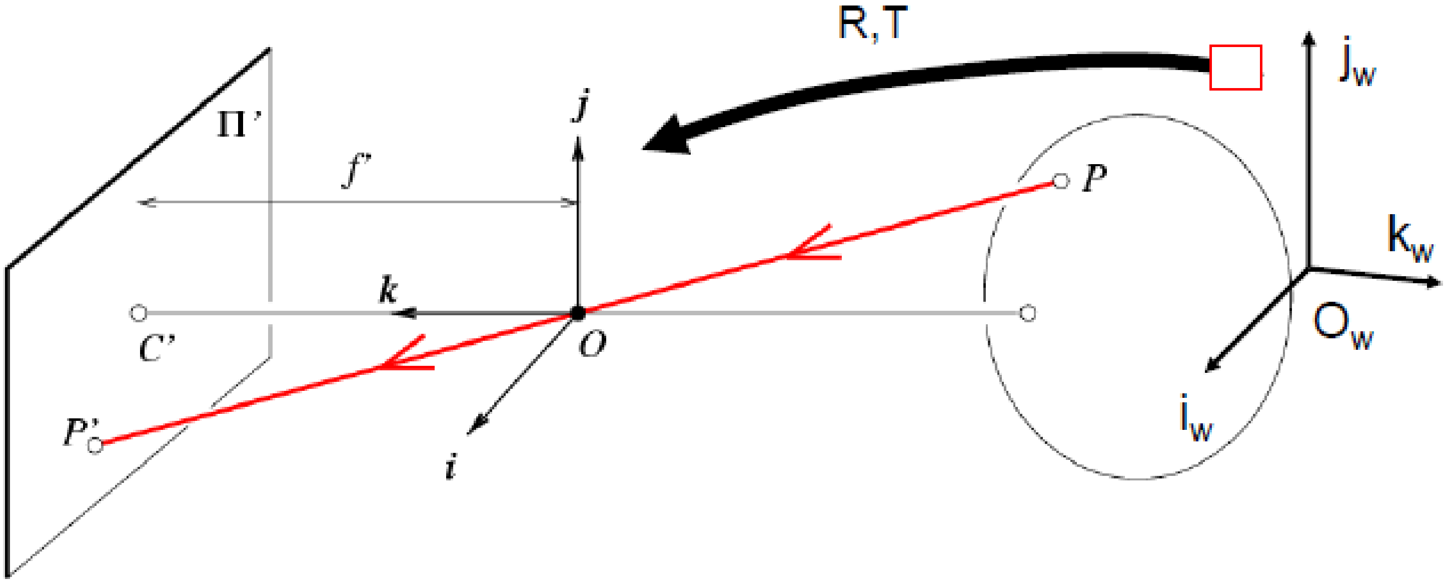
Pinhole Camera Model



Common to draw image plane *in front* of the focal point. Moving the image plane merely scales the image.

$$\begin{cases} x' = f\,\dfrac{x}{z} \\[2ex] y' = f\,\dfrac{y}{z} \end{cases}$$

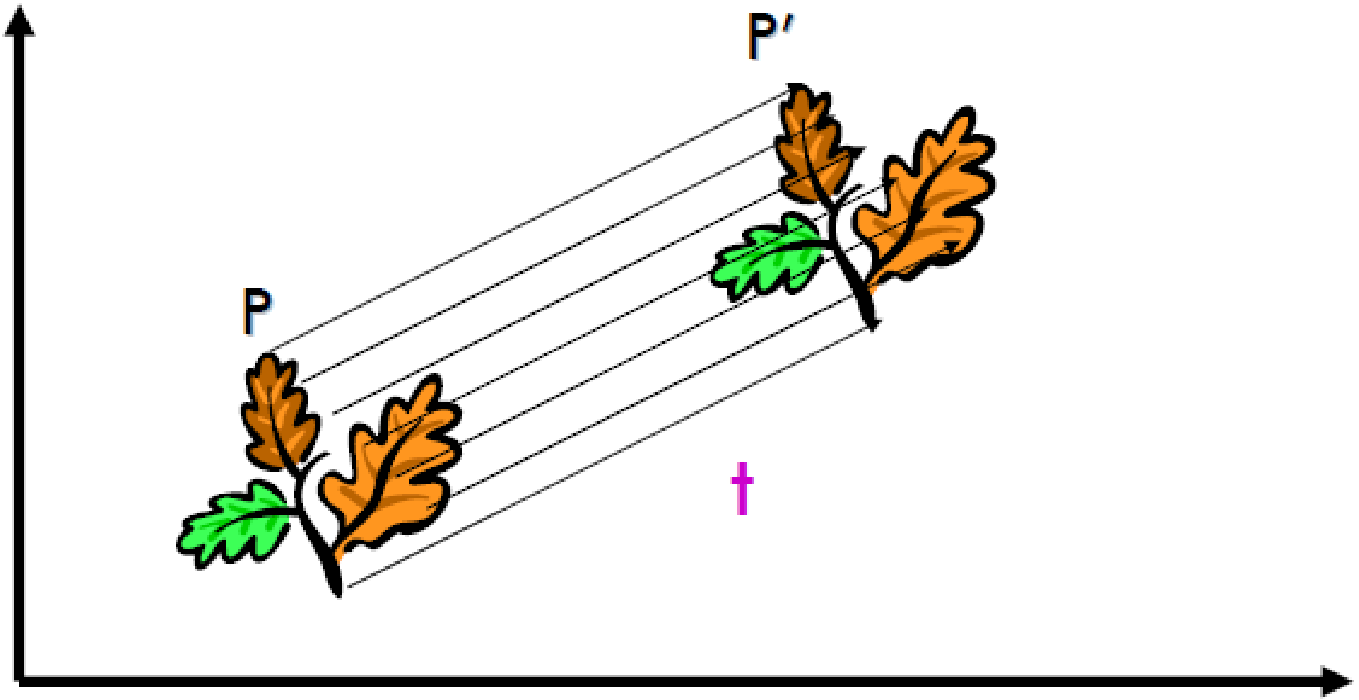
World to Camera – Camera to World

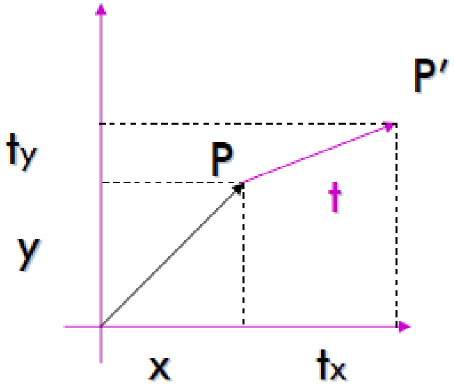


2D point (3x1)	=	Camera to pixel coord. trans. matrix (3x3)	Perspective projection matrix (3x4)	World to camera coord. trans. matrix (4x4)	3D point (4x1)
camera		K matrix 3x3 internal matrix	[I 0] matrix	R, t matrix ~ external matrix	world

Geometric Transformations

2D Translation





$$\mathbf{P} = (x, y)$$

$$\mathbf{t} = (t_x, t_y)$$

$$\mathbf{P}' = \mathbf{P} + \mathbf{t} = (\mathrm{x} + \mathrm{t}_\mathrm{x}, \mathrm{y} + \mathrm{t}_\mathrm{y})$$

$$\mathbf{P}' \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Homogeneous Coordinates

- Multiply the coordinates by a non-zero scalar and add an extra coordinate equal to that scalar. For example,

$$(x, y) \rightarrow (x \cdot z, y \cdot z, z) \quad z \neq 0$$

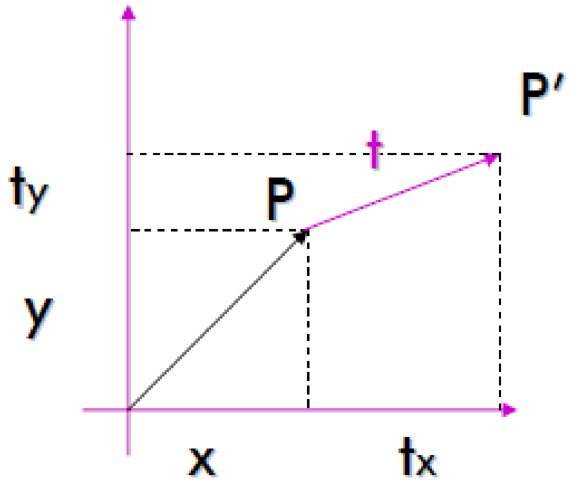
$$(x, y, z) \rightarrow (x \cdot w, y \cdot w, z \cdot w, w) \quad w \neq 0$$

- Back to Cartesian coordinates

$$(x, y, z) \quad z \neq 0 \rightarrow (x / z, y / z)$$

$$(x, y, z, w) \quad w \neq 0 \rightarrow (x / w, y / w, z / w)$$

2D Translation using Homogeneous Coordinates



$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

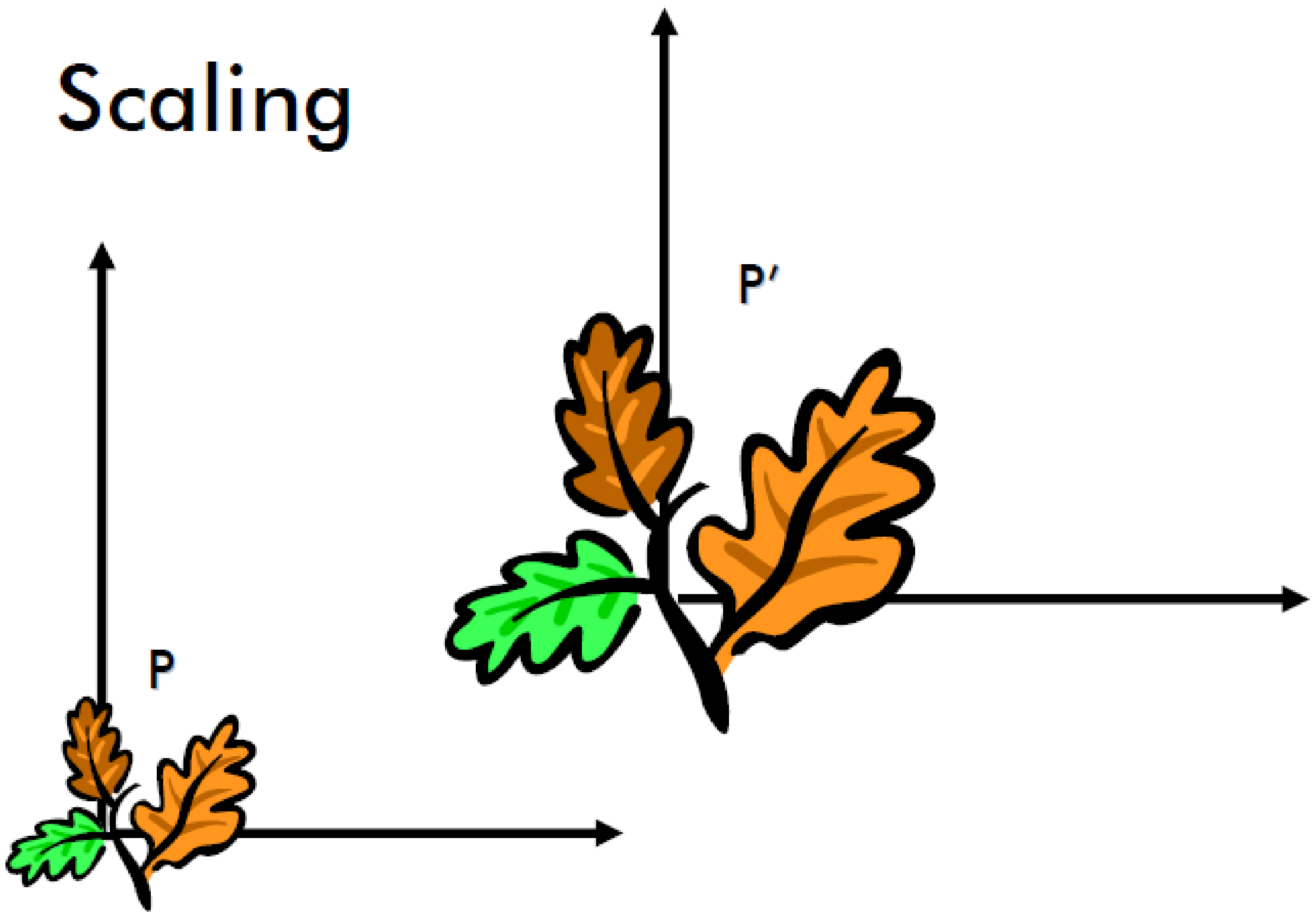
$$\mathbf{t} = (t_x, t_y) \rightarrow (t_x, t_y, 1)$$

t P

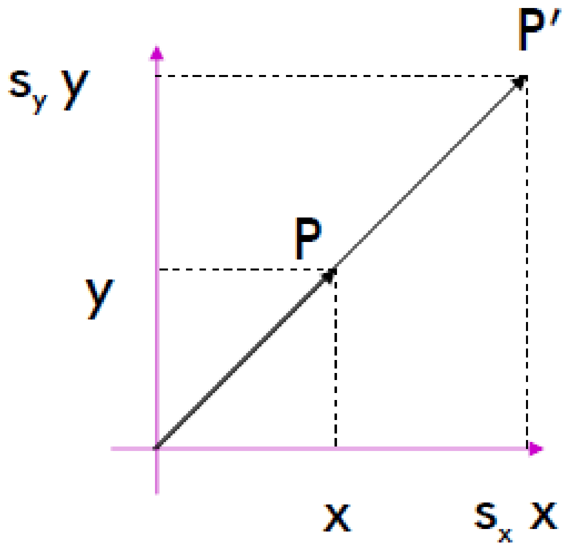
$$\mathbf{P}' \rightarrow \begin{bmatrix} x + t_x \\ y + t_y \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \mathbf{I} & \mathbf{t} \\ 0 & 1 \end{bmatrix} \cdot \mathbf{P} = \mathbf{T} \cdot \mathbf{P}$$

Scaling



Scaling Equation



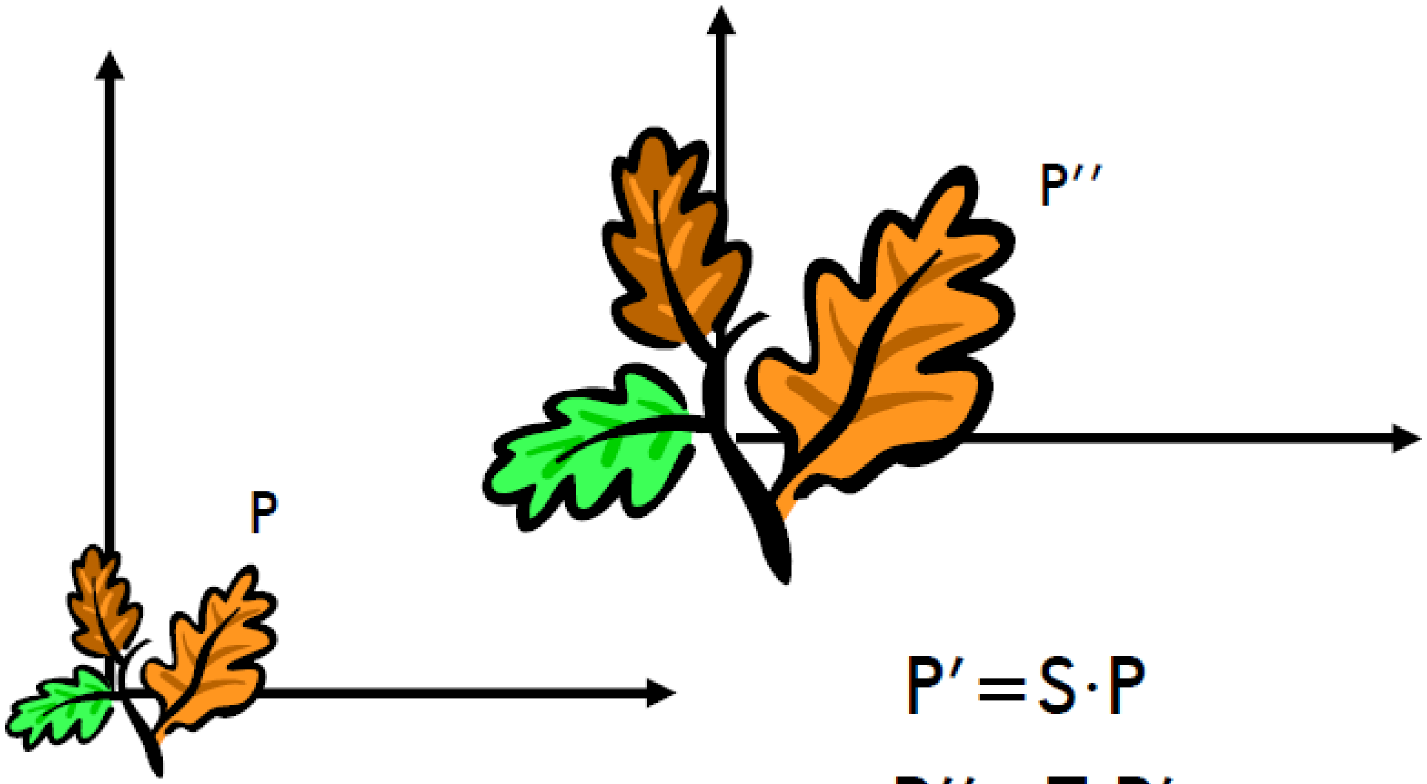
$$\mathbf{P} = (\mathrm{x}, \mathrm{y}) \rightarrow \mathbf{P}' = (\mathrm{s}_\mathrm{x}\mathrm{x}, \mathrm{s}_\mathrm{y}\mathrm{y})$$

$$\mathbf{P} = (x, y) \rightarrow (x, y, 1)$$

$$\mathbf{P}' = (s_x x, s_y y) \rightarrow (s_x x, s_y y, 1)$$

$$\mathbf{P}' \rightarrow \begin{bmatrix} s_x x \\ s_y y \\ 1 \end{bmatrix} = \underbrace{\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}}_{\mathbf{S}} \cdot \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{S}' & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \end{bmatrix} \cdot \mathbf{P} = \mathbf{S} \cdot \mathbf{P}$$

Scaling & Translating



$$P' = S \cdot P$$

$$P'' = T \cdot P'$$

$$P'' = T \cdot P' = T \cdot (S \cdot P) = (T \cdot S) \cdot P = A \cdot P$$

Scaling & Translating

$$\mathbf{P}''=\mathbf{T}\cdot\mathbf{S}\cdot\mathbf{P}=\begin{bmatrix}1 & 0 & t_x\\ 0 & 1 & t_y\\ 0 & 0 & 1\end{bmatrix}\cdot\begin{bmatrix}s_x & 0 & 0\\ 0 & s_y & 0\\ 0 & 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}=$$

$$=\underbrace{\begin{bmatrix}s_x & 0 & t_x\\ 0 & s_y & t_y\\ 0 & 0 & 1\end{bmatrix}}_{A}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}=\begin{bmatrix}S & t\\ 0 & 1\end{bmatrix}\begin{bmatrix}x\\ y\\ 1\end{bmatrix}=\begin{bmatrix}s_x x+t_x\\ s_y y+t_y\\ 1\end{bmatrix}$$

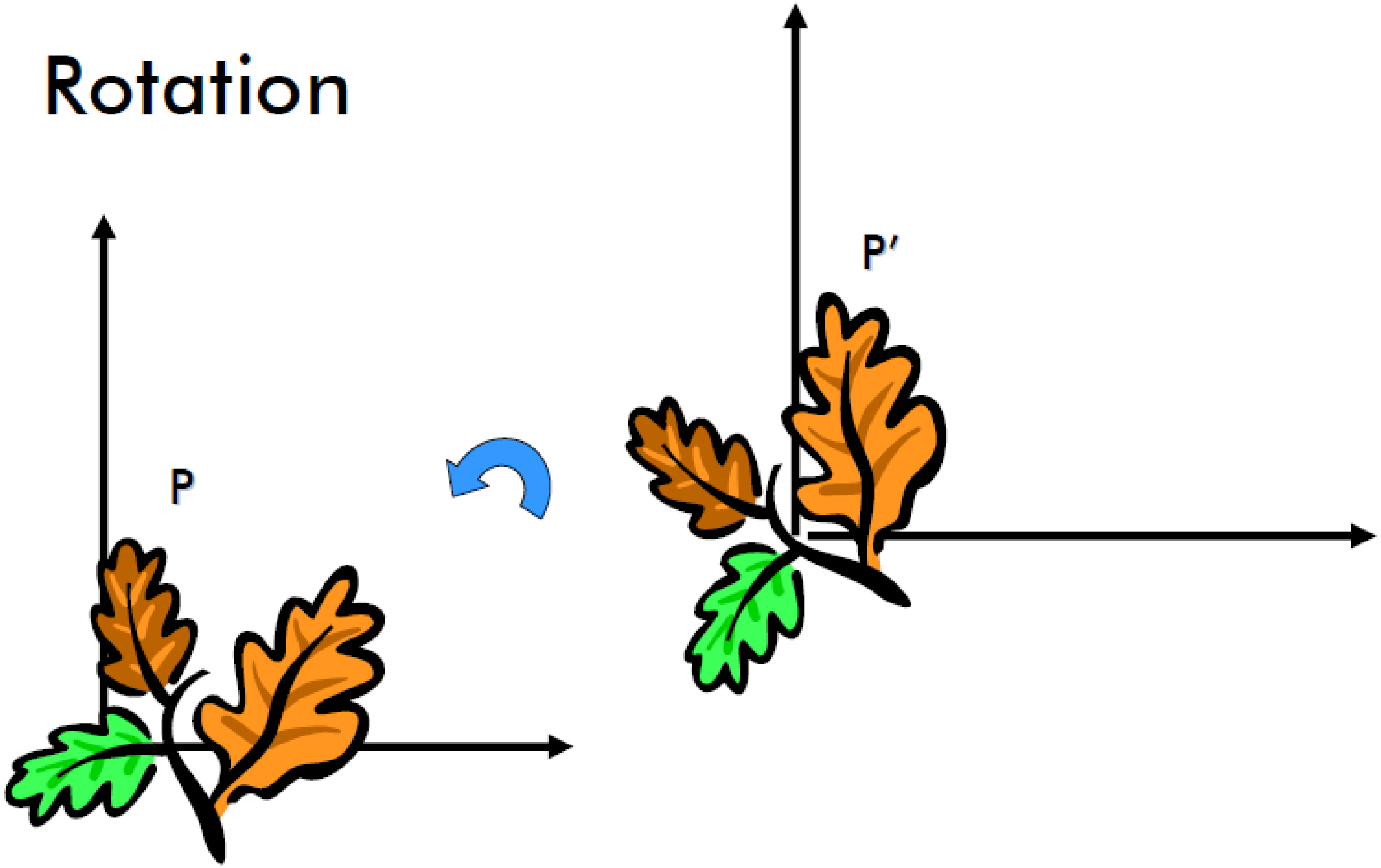
Translating & Scaling = Scaling & Translating ?

$$\mathbf{P}''' = \mathbf{T}\cdot\mathbf{S}\cdot\mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x & 0 & t_x \\ 0 & s_y & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + t_x \\ s_y y + t_y \\ 1 \end{bmatrix}$$

$$\mathbf{P}''' = \mathbf{S}\cdot\mathbf{T}\cdot\mathbf{P} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

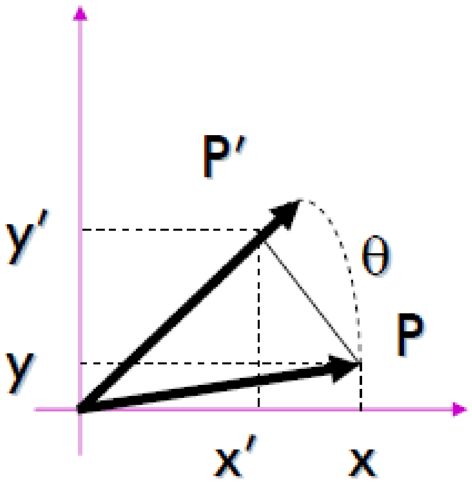
$$= \begin{bmatrix} s_x & 0 & s_x t_x \\ 0 & s_y & s_y t_y \\ 0 & 0 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} s_x x + s_x t_x \\ s_y y + s_y t_y \\ 1 \end{bmatrix}$$

Rotation



Rotation Equations

Counter-clockwise rotation by an angle θ



$$x' = \cos\theta\ x - \sin\theta\ y$$

$$y' = \cos\theta\ y + \sin\theta\ x$$

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

$$\mathbf{P' = R\ P}$$

Degrees of Freedom

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix}$$

R is 2x2 $\Rightarrow$ 4 elements

Note: R belongs to the category of *normal* matrices and satisfies many interesting properties:

$$\mathbf{R} \cdot \mathbf{R}^{\mathrm{T}} = \mathbf{R}^{\mathrm{T}} \cdot \mathbf{R} = \mathbf{I}$$

$$\det(\mathbf{R}) = 1$$

Rotation+ Scaling +Translation

P′= (T R S) P

$$\mathbf{P}' = \mathbf{T}\cdot \mathrm{R} \cdot \mathbf{S} \cdot \mathbf{P} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} =$$

$$= \begin{bmatrix} R' & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} S & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \boxed{\begin{bmatrix} R'S & t \\ 0 & 1 \end{bmatrix}} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

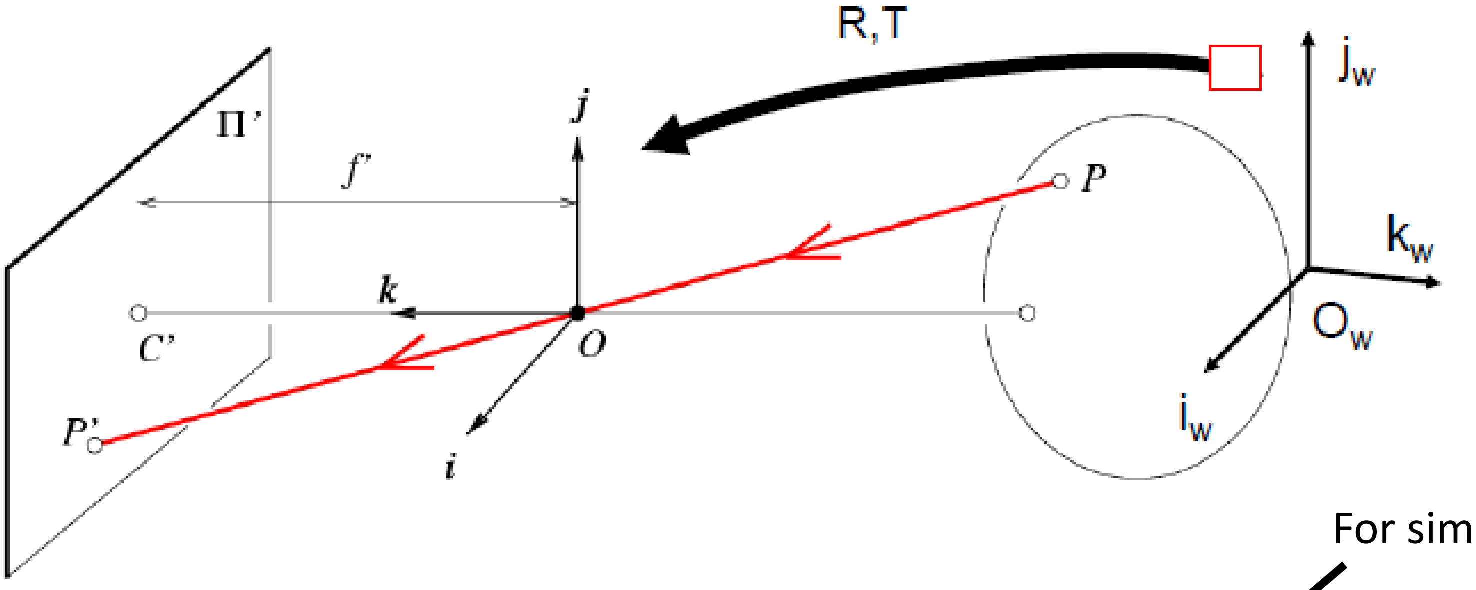
If $s_x = s_y$, this is a similarity transformation!

Geometric Transformations

- Isometries or Euclidean Transformation
- Similarity Transformation
- Affine Transformation
- Projective Transformations

Camera Coordinates and Image Formation

World to Camera – Camera to World



For simplicity assume

$$\underset{\text{camera}}{\begin{pmatrix}\text{2D}\\ \text{point}\\ \text{(3x1)}\end{pmatrix}} = \underset{\substack{\text{K matrix}\\ \text{3x3 internal matrix}}}{\begin{pmatrix}\text{Camera to}\\ \text{pixel coord.}\\ \text{trans. matrix}\\ \text{(3x3)}\end{pmatrix}} \underset{\text{[I 0] matrix}}{\boxed{\begin{pmatrix}\text{Perspective}\\ \text{projection matrix}\\ \text{(3x4)}\end{pmatrix}}} \underset{\substack{\text{R, t matrix}\\ \sim\text{ external matrix}}}{\begin{pmatrix}1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\\ 0 & 0 & 1 & 0\\ 0 & 0 & 0 & 1\end{pmatrix}} \underset{\text{world}}{\begin{pmatrix}\text{3D}\\ \text{point}\\ \text{(4x1)}\end{pmatrix}}$$

Perspective Projection Transformation

$$X' = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\text{M}} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} \qquad X' = M\,X$$

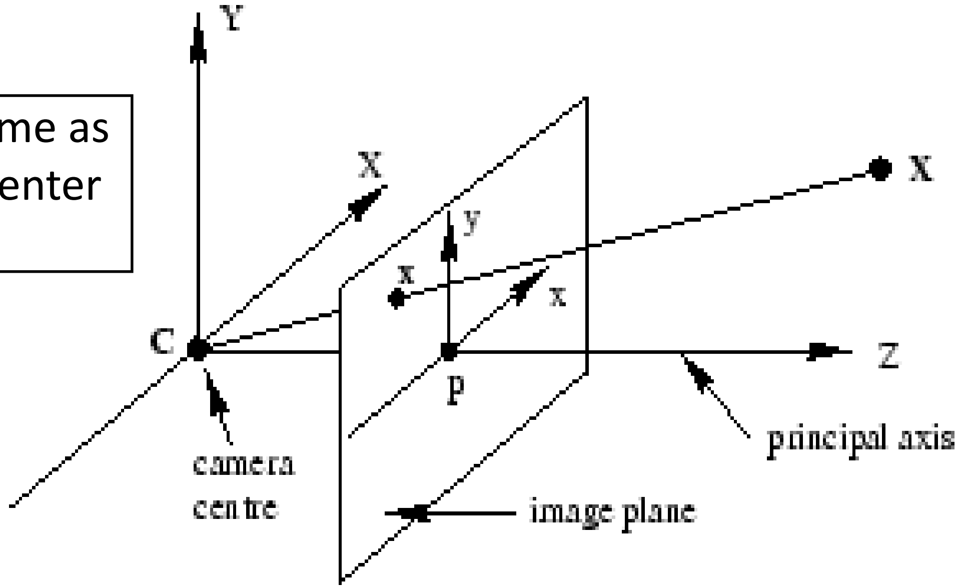
$$\text{pixel measured} = \begin{bmatrix} \frac{x}{z} \\ \frac{y}{z} \end{bmatrix}$$

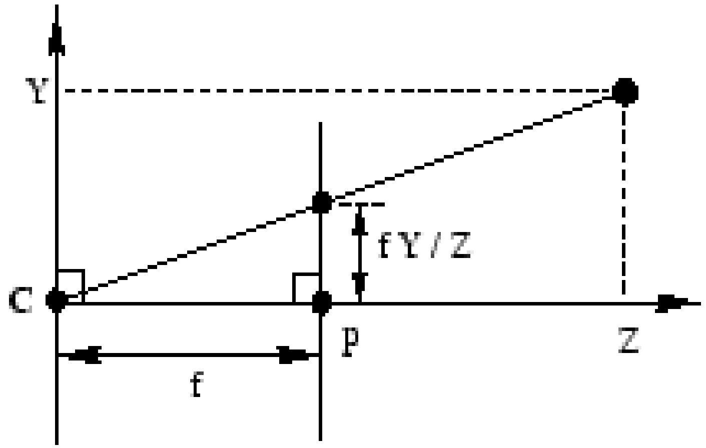
X point in 3D homogeneous c.

X' point in 2D homogeneous c.

Pinhole Camera Model

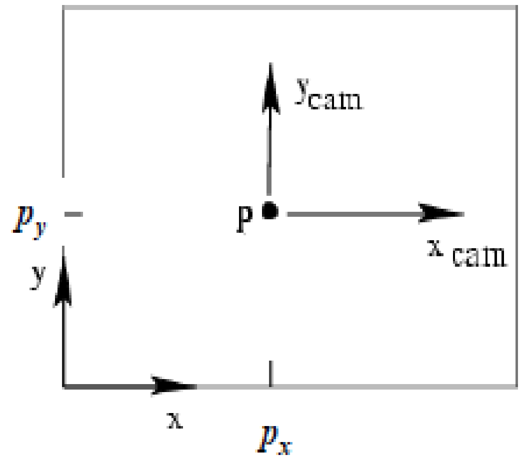
Camera Center is the same as the focal point, or the center of projection (‘**C**’)





- **Principal Axis** – Line from the camera centre perpendicular to the image plane.
- **Normalized Camera Coordinate** – Camera centre at the origin (**C**), x and y axes are aligned with the image axes and image plane ($P_z = f$); units in m/cm/ft etc.
- **Principal point** – point where the principal axis intersects the image plane ($z=f$)
- **Pixel coordinate frame** – Origin (0,0) is in the corner of an image; units are in pixels.

Principal point offset



principal point: (p_x, p_y)

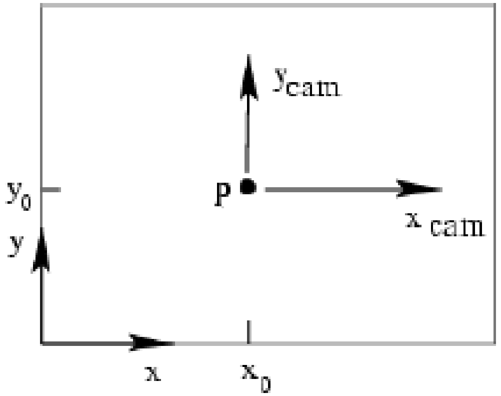
$$(X,Y,Z) \mapsto (fX/Z + p_x, fY/Z + p_y)$$

Pixel coordinate frame

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

Homogeneous coordinates

Principal point offset



principal point: (p_x, p_y)

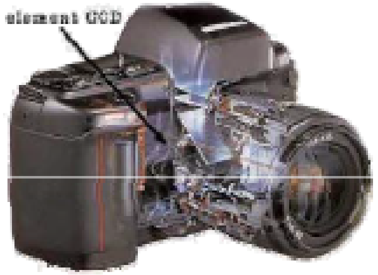
$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

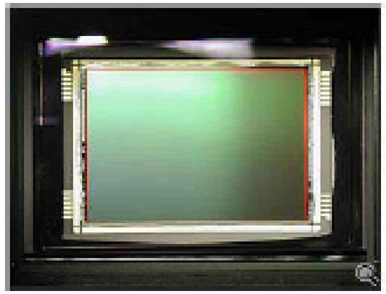
$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

a simple **intrinsic** matrix

$$\mathrm{P} = \mathrm{K}[\mathrm{I} \mid 0]$$

Pixel coordinates





Pixel size: $\frac{1}{m_x} \times \frac{1}{m_y}$

m_x pixels per meter in horizontal direction
m_y pixels per meter in vertical direction

$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}$$

pixels/m m pixels

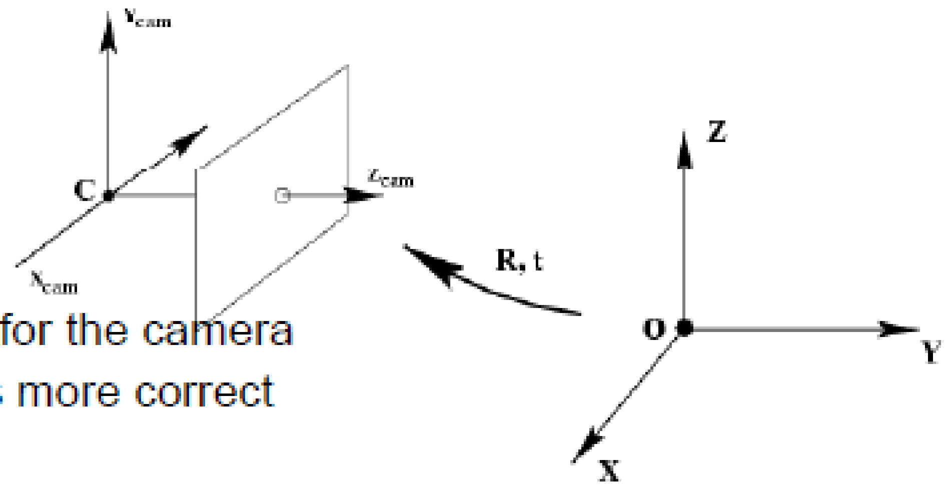
Intrinsic Camera Matrix

- Skew parameter *s* is non-zero, only if x and y axes are non-orthogonal, i.e., pixels are not rectangular
- Recent cameras rarely have ***non-square*** pixels

skew parameter

$$\underbrace{\begin{bmatrix} x \\ y \\ z \end{bmatrix}}_{\text{homogeneous coordinates}} = \underbrace{\begin{bmatrix} \alpha_x & \boxed{s} & \beta_x \\ 0 & \alpha_y & \beta_y \\ 0 & 0 & 1 \end{bmatrix}}_{K_{3x3}} \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{[\mathbf{I} \mid \mathbf{0}]} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & s & \beta_x & 0 \\ 0 & \alpha_y & \beta_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Camera rotation and translation



z-axis for the camera
here is more correct

- In general, the camera coordinate frame will be related to the world coordinate frame by a rotation and a translation

3D nonhomogeneous

$$\tilde{X}_{cam} = R\left(\tilde{X} - \tilde{C}\right)$$

rotation to camera frame * vector from camera center to the point

coords. of point in camera frame

coords. of camera center in world frame

coords. of a point in world frame (nonhomogeneous)

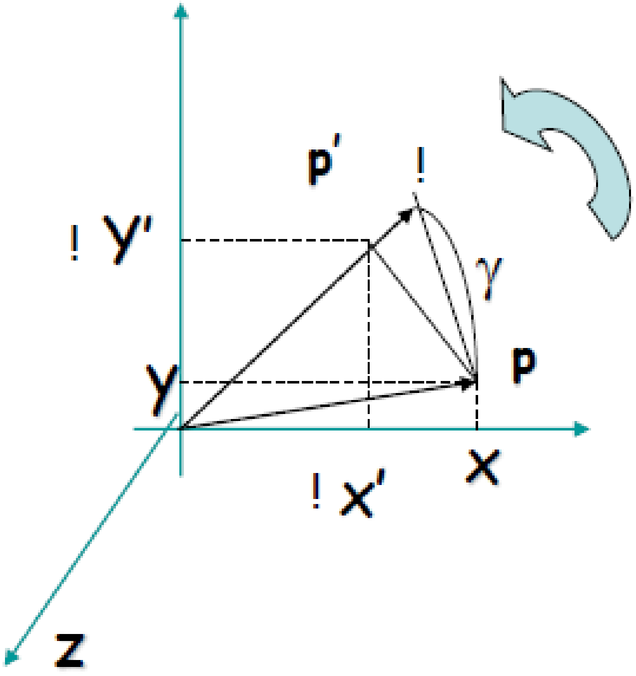
3D Rotation Matrices

- Representations
  - Euler angles
  - Axis-angle
  - Rodrigues’ formula
  - ~~Unit quaternion~~

- Important to note:
  - Points along the axis of rotation are invariant to rotation

Euler Angles

3D Rotation around the coordinate axes counter-clockwise.



$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix}$$

$$R_y(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix}$$

$$R_z(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Axis Angle

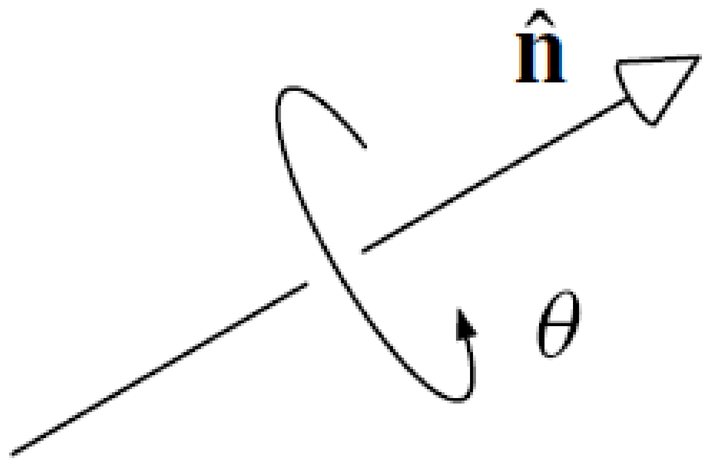
Let $\mathrm{rot}(\hat{\mathbf{n}}, \theta)$ be the corresponding rotation.

Many to one:

$\mathrm{rot}(-\hat{\mathbf{n}}, -\theta) = \mathrm{rot}(\hat{\mathbf{n}}, \theta)$

$\mathrm{rot}(\hat{\mathbf{n}}, \theta + 2k\pi) = \mathrm{rot}(\hat{\mathbf{n}}, \theta)$, for any integer k.

When $\theta = 0$, the rotation axis is indeterminate, giving an infinity-to-one mapping.

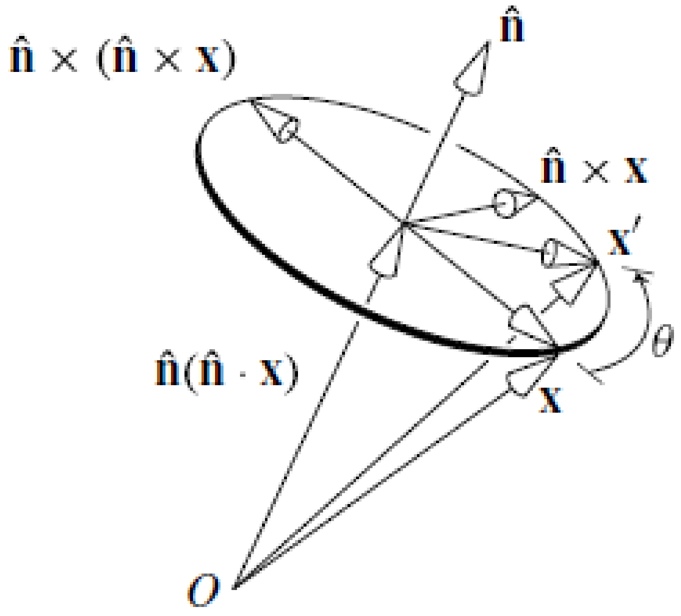


Rodrigues's Formula

Given point $\mathbf{x}$, decompose into components parallel and perpendicular to the rotation axis

$$\mathbf{x} = \hat{\mathbf{n}}(\hat{\mathbf{n}} \cdot \mathbf{x}) - \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \mathbf{x})$$

Only $\mathbf{x}_\perp$ is affected by the rotation, yielding *Rodrigues's formula*:



Rodrigues's Formula

Axis-angle to $\boldsymbol{R}$

$$\mathbf{x}' = \mathbf{x} + (\sin\theta)\ \hat{\mathbf{n}} \times \mathbf{x} + (1-\cos\theta)\ \hat{\mathbf{n}} \times (\hat{\mathbf{n}} \times \mathbf{x})$$

$$N = \begin{pmatrix} 0 & -n_3 & n_2 \\ n_3 & 0 & -n_1 \\ -n_2 & n_1 & 0 \end{pmatrix}$$

$$N\mathbf{x} = \hat{\mathbf{n}} \times \mathbf{x}$$

$$\mathbf{x}' = \mathbf{x} + (\sin\theta) N\mathbf{x} + (1-\cos\theta) N^2\mathbf{x}$$

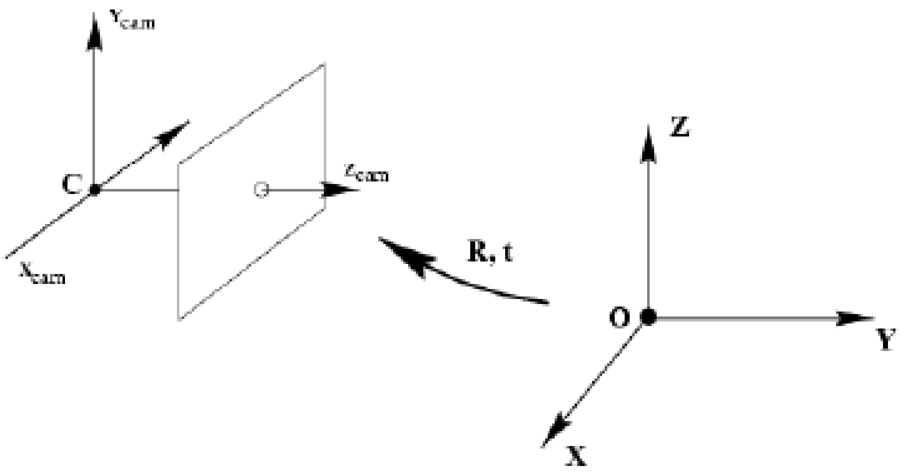
$$R = I + (\sin\theta) N + (1-\cos\theta) N^2$$

$\boldsymbol{R}$ to Axis-angle

$$\theta = \cos^{-1}\left(\frac{trace(\boldsymbol{R})-1}{2}\right)$$

$$\boldsymbol{n} = \frac{1}{2\sin(\theta)}\begin{bmatrix} \boldsymbol{R}_{32} - \boldsymbol{R}_{23} \\ \boldsymbol{R}_{13} - \boldsymbol{R}_{31} \\ \boldsymbol{R}_{21} - \boldsymbol{R}_{12} \end{bmatrix}$$

Camera rotation and translation



In nonhomogeneous coordinates:

$$\widetilde{X}_{cam} = R\left(\widetilde{X} - \widetilde{C}\right)$$

homogeneous coor.
in 3D

$$X_{cam} = \begin{bmatrix} R & -R\widetilde{C} \\ 0 & 1 \end{bmatrix} \begin{pmatrix} \widetilde{X} \\ 1 \end{pmatrix} = \begin{bmatrix} R & -R\widetilde{C} \\ 0 & 1 \end{bmatrix} X$$

in 2D plane

$$x = K[I \mid 0]X_{cam} = K\left[R \mid -R\widetilde{C}\right]X$$

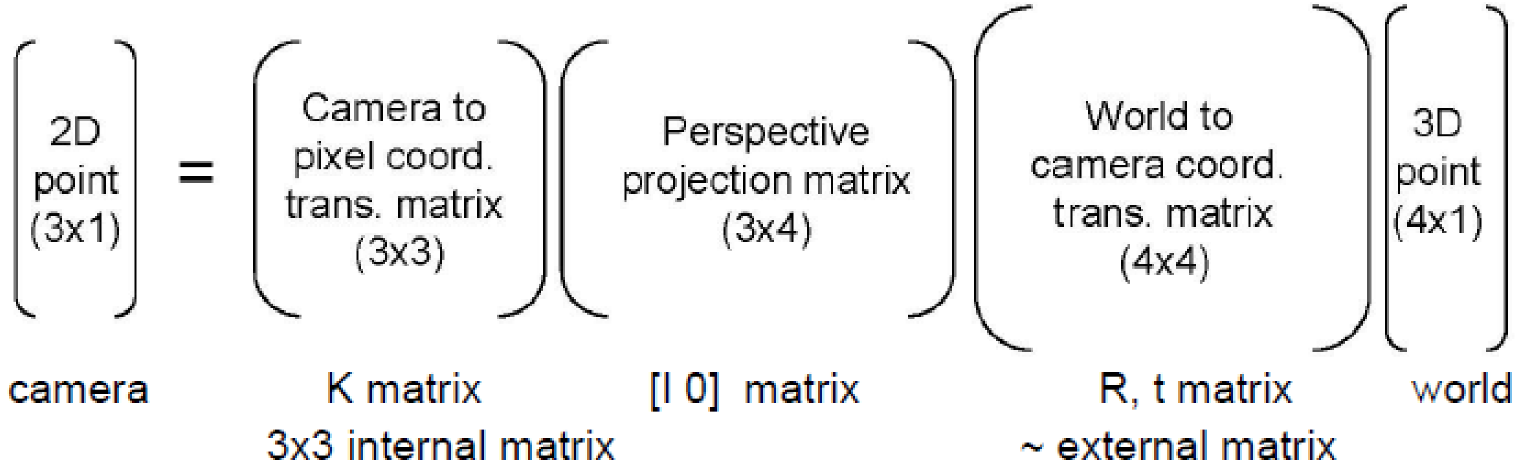
3D to 2D

$$P = K[R \mid t], \quad t = -R\widetilde{C}$$

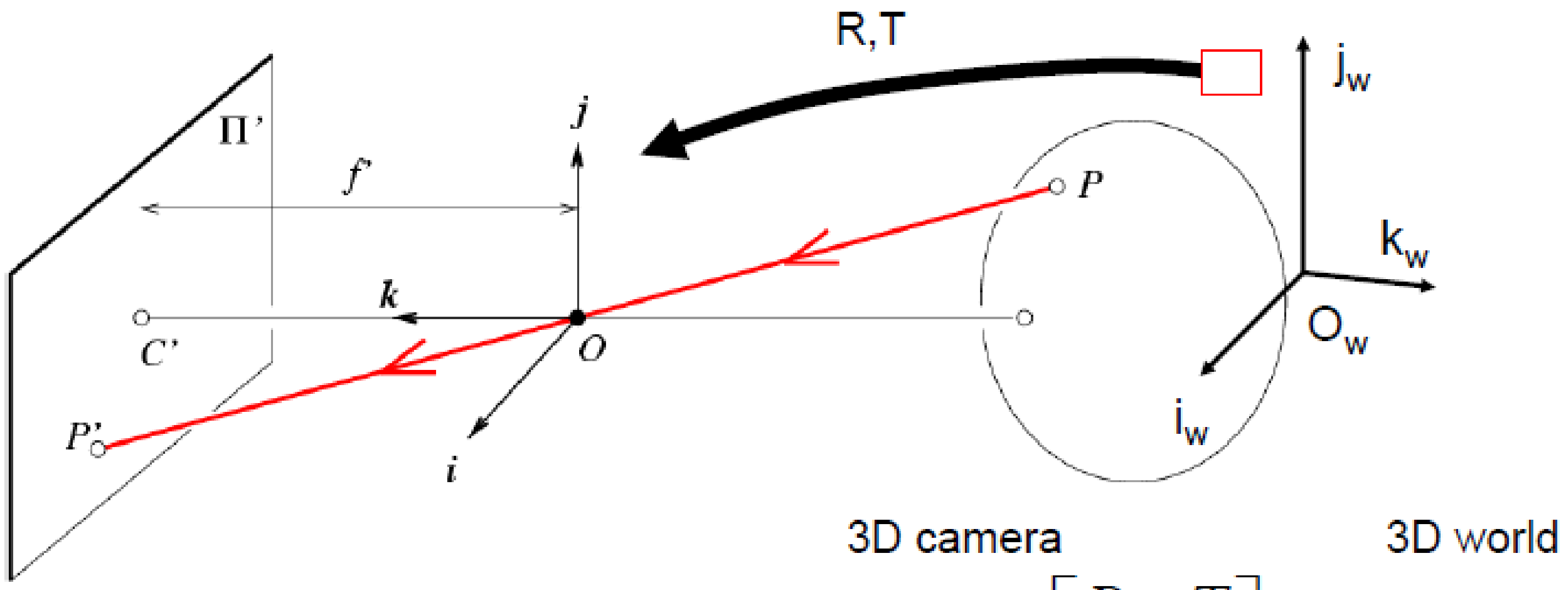
Note: C is the null space of the camera projection matrix (PC=0)

$K[R \mid -R\tilde{C}]\,[\tilde{C}\ 1]^T = 0$

Image Formation Summary



- Rotation and Translation (world to camera)
- Perspective Projection (camera 3D to image plane 2D)
- Scaling and Shifting (image plane 2D to pixel 2D)
- Use of homogeneous coordinates



In homogeneous coordinates
$$X = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}_{4\times 4} X_w$$

We need 5 + 3 + 3 =11 degrees of freedom (DOF) maximum.

X' in the plane $$X' = M\,X_w = \boxed{K}\boxed{[R \quad T]}\,X_w$$

Internal parameters

External parameters

Camera Parameters

- Intrinsic
  - Focal length (2-dof) f_x and f_y
  - Principal point (2-dof) c_x, c_y
  - Skew factor (1-dof) s
- Extrinsic
  - Rotation (3-dof) - $\mathbf{R}$
  - Translation (3-dof) – $\mathbf{t}$
- Total degrees of freedom: 3 + 3 + 2 + 2 + 1 = 11
  - Need 11 equations to estimate these parameters
  - Direct Linear Transform
  - Calibration toolboxes

Camera Calibration

Projective Camera

$$P' = M\,P_w \quad = \boxed{K}\boxed{[R \quad T]}\,P_w$$

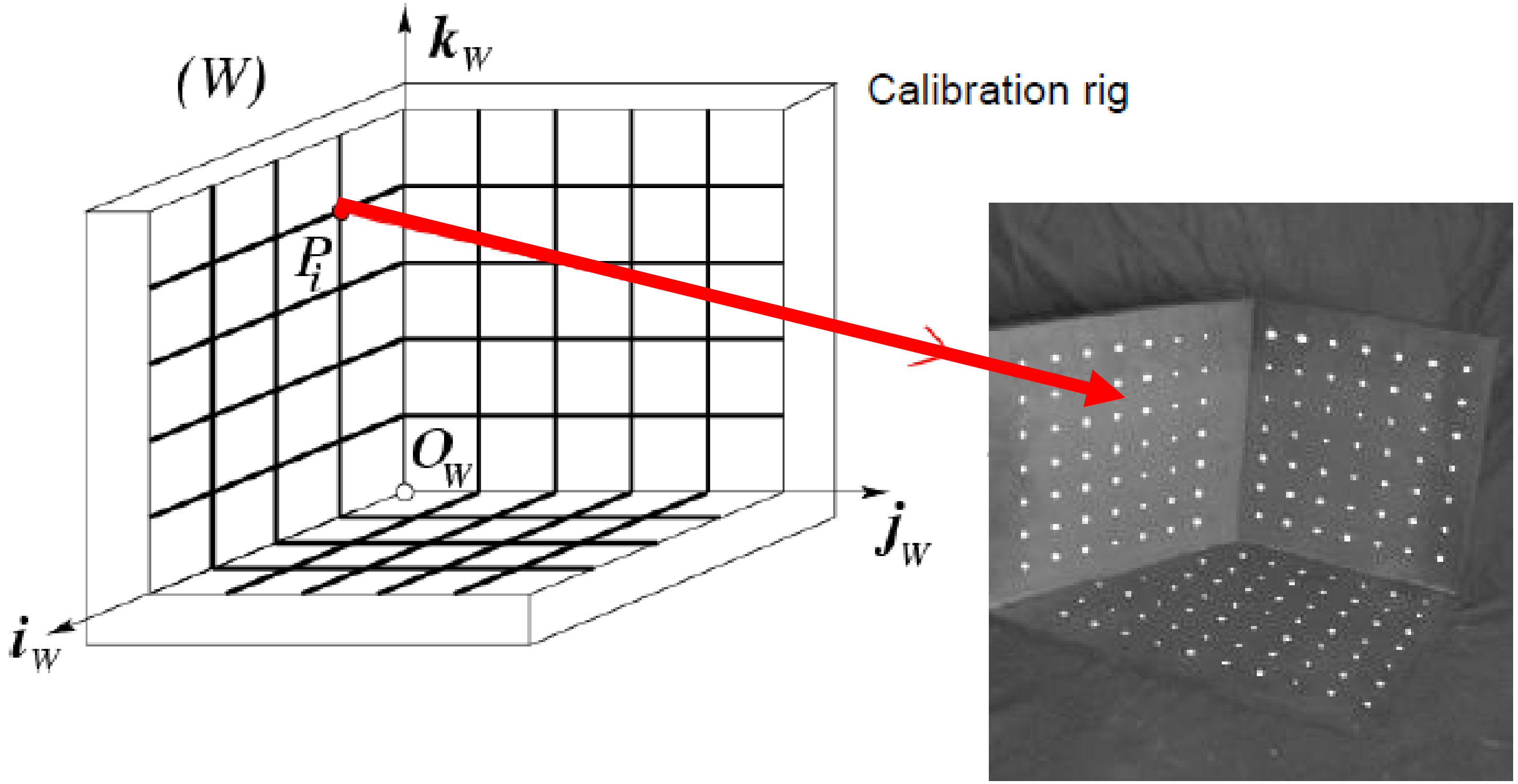
Internal parameters

External parameters

$$K = \begin{bmatrix} \alpha & -\alpha\cot\theta & u_o \\ 0 & \dfrac{\beta}{\sin\theta} & v_o \\ 0 & 0 & 1 \end{bmatrix} \qquad R = \begin{bmatrix} \mathbf{r}_1^T \\ \mathbf{r}_2^T \\ \mathbf{r}_3^T \end{bmatrix} \qquad T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}$$

Goal: To estimate internal and external parameters from one or multiple images

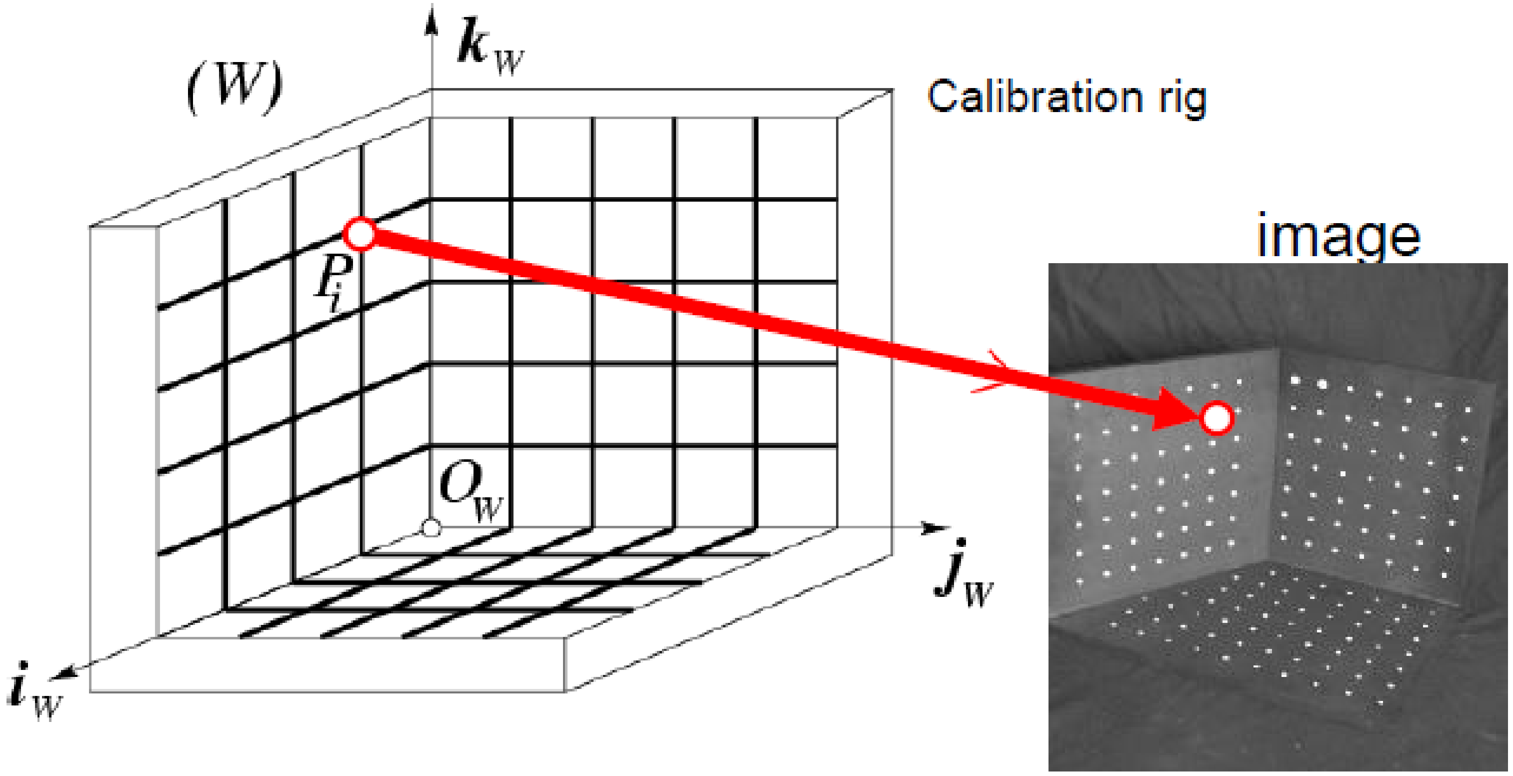
Calibration Problem



- $P_1 \ldots P_n$ with known positions in $[O_W, i_W, j_W, k_W]$
- $p_1, \ldots p_n$ known positions in the image

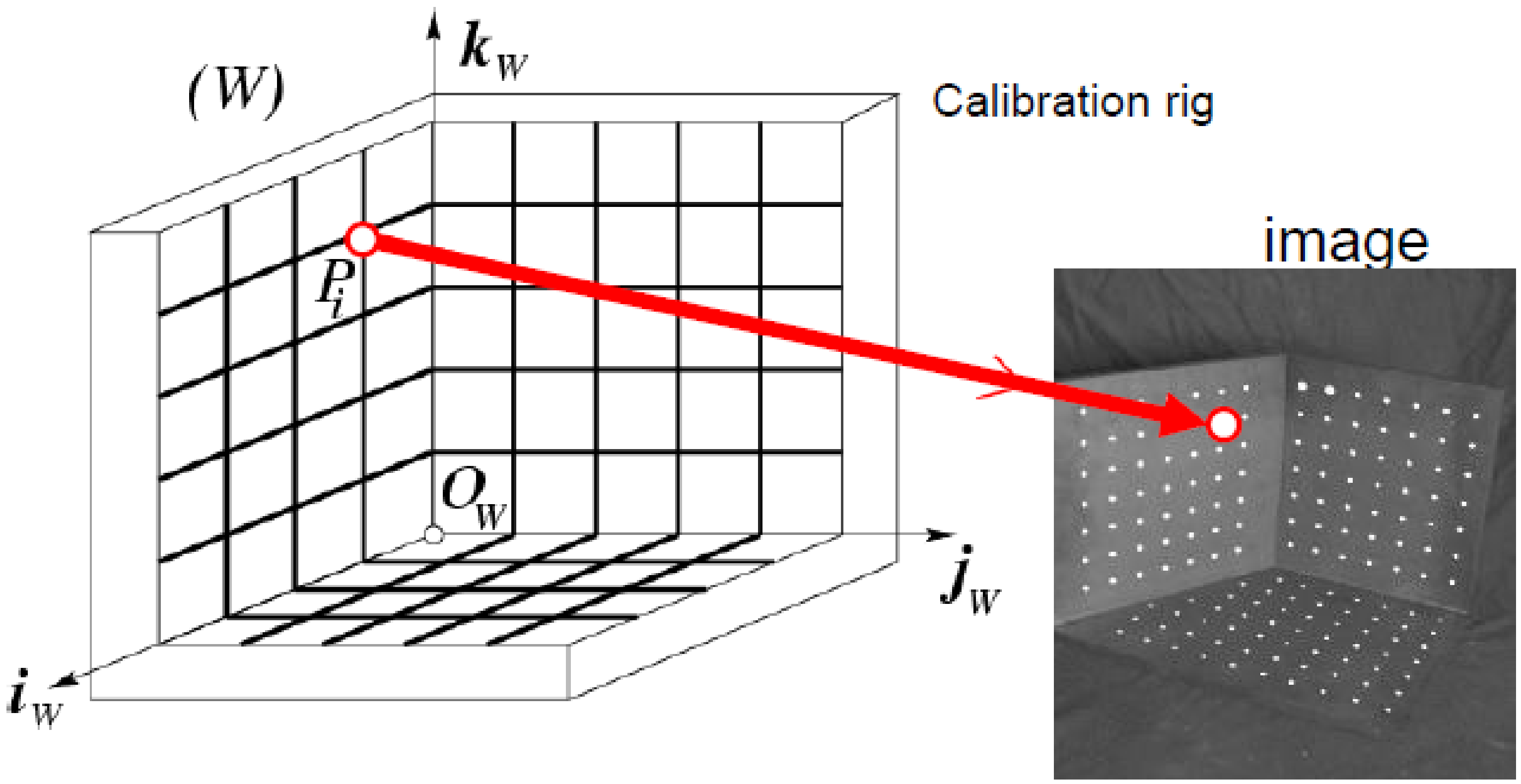
Goal: compute intrinsic and extrinsic parameters

Calibration Problem



In practice, using more than 6 correspondences enables more robust results

Calibration Problem



$$P_i \rightarrow M\,P_i \rightarrow p_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} \dfrac{\mathbf{m}_1\,P_i}{\mathbf{m}_3\,P_i} \\ \dfrac{\mathbf{m}_2\,P_i}{\mathbf{m}_3\,P_i} \end{bmatrix} \qquad M = \begin{bmatrix} \mathbf{m}_1 \\ \mathbf{m}_2 \\ \mathbf{m}_3 \end{bmatrix}$$

in pixels

Calibration Problem

$$\begin{cases} -u_1(\mathbf{m}_3\, P_1) + \mathbf{m}_1\, P_1 = 0 \\ -v_1(\mathbf{m}_3\, P_1) + \mathbf{m}_2\, P_1 = 0 \\ \vdots \\ -u_n(\mathbf{m}_3\, P_n) + \mathbf{m}_1\, P_n = 0 \\ -v_n(\mathbf{m}_3\, P_n) + \mathbf{m}_2\, P_n = 0 \end{cases} \longrightarrow \boxed{\mathbf{P}\,\mathbf{m} = 0}$$

known

unknown

Homogenous linear system

1x4

$$\mathbf{P} \stackrel{\text{def}}{=} \begin{pmatrix} \mathrm{P}_1^T & \mathbf{0}^T & -u_1\mathrm{P}_1^T \\ \mathbf{0}^T & \mathrm{P}_1^T & -v_1\mathrm{P}_1^T \\ & \vdots & \\ \mathrm{P}_n^T & \mathbf{0}^T & -u_n\mathrm{P}_n^T \\ \mathbf{0}^T & \mathrm{P}_n^T & -u_n\mathrm{P}_n^T \end{pmatrix}_{2n \times 12}$$

4x1

$$\boldsymbol{m} \stackrel{\text{def}}{=} \begin{pmatrix} \mathbf{m}_1^T \\ \mathbf{m}_2^T \\ \mathbf{m}_3^T \end{pmatrix}_{12\text{x}1}$$

Decomposition of the Camera Matrix

- Camera Centre: $\mathbf{MC}=\mathbf{0}$
  - One dimensional null space of camera matrix $\mathbf{M}$

- Extrinsic Parameters $(\mathbf{R},\mathbf{t})$ and Intrinsic Parameters $(\mathbf{K})$
  - Use RQ decomposition where R is upper triangular and Q is orthogonal

RQ Decomposition

A 3-dimensional Givens rotation is a rotation about one of the three coordinate axes. The three Givens rotations are

$$\mathtt{Q}_x = \begin{bmatrix} 1 & & \\ & c & -s \\ & s & c \end{bmatrix} \quad \mathtt{Q}_y = \begin{bmatrix} c & & s \\ & 1 & \\ -s & & c \end{bmatrix} \quad \mathtt{Q}_z = \begin{bmatrix} c & -s & \\ s & c & \\ & & 1 \end{bmatrix} \tag{A4.1}$$

<u>Objective</u>

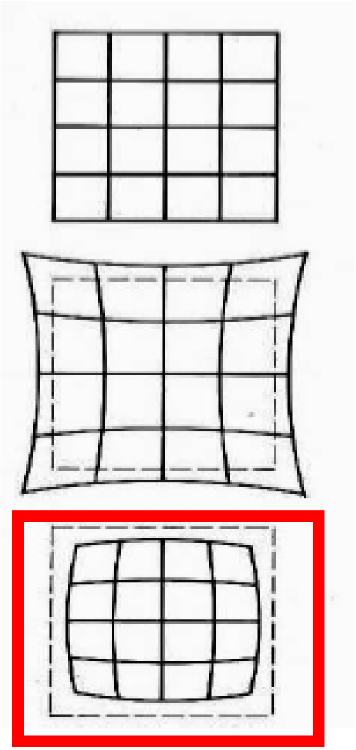
Carry out the RQ decomposition of a 3×3 matrix $\mathtt{A}$ using Givens rotations.

<u>Algorithm</u>

(i) Multiply by $\mathtt{Q}_x$ so as to set $\mathtt{A}_{32}$ to zero.
(ii) Multiply by $\mathtt{Q}_y$ so as to set $\mathtt{A}_{31}$ to zero. This multiplication does not change the second column of $\mathtt{A}$, hence $\mathtt{A}_{32}$ remains zero.
(iii) Multiply by $\mathtt{Q}_z$ so as to set $\mathtt{A}_{21}$ to zero. The first two columns are replaced by linear combinations of themselves. Thus, $\mathtt{A}_{31}$ and $\mathtt{A}_{32}$ remain zero.

Radial Distortion

- Caused by imperfect lenses
- Deviations are most noticeable for rays that pass through the edge of the lens



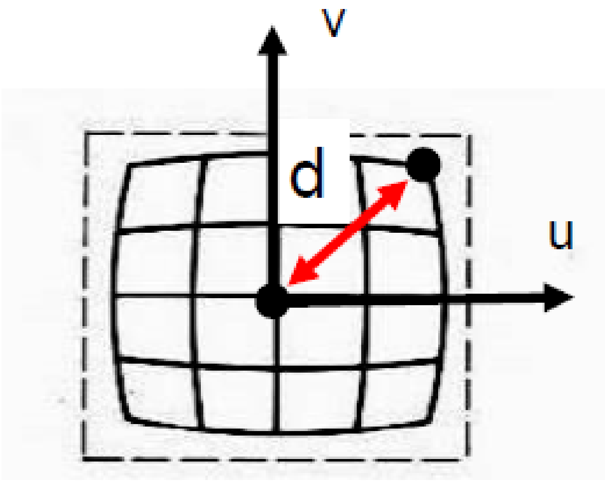
No distortion

Pin cushion

Barrel

Radial Distortion

Image magnification in(de)creases with distance from the optical center



$$\begin{bmatrix} \frac{1}{\lambda} & 0 & 0 \\ 0 & \frac{1}{\lambda} & 0 \\ 0 & 0 & 1 \end{bmatrix} M\, P_i \rightarrow \begin{bmatrix} u_i \\ v_i \end{bmatrix} = p_i$$

$$d^2 = a\, u^2 + b\, v^2 + c\, u\, v$$

To model radial behavior

Distortion coefficient

$$\lambda = \underbrace{1 \pm \sum_{p=1}^{3} \boxed{\kappa_p} d^{2p}}_{\text{Polynomial function}}$$

Radial Distortion

$$\underbrace{\boxed{\begin{bmatrix} \frac{1}{\lambda} & 0 & 0 \\ 0 & \frac{1}{\lambda} & 0 \\ 0 & 0 & 1 \end{bmatrix} M}}_{Q} P_i \rightarrow \begin{bmatrix} u_i \\ v_i \end{bmatrix} = p_i \qquad Q = \begin{bmatrix} \mathbf{q}_1 \\ \mathbf{q}_2 \\ \mathbf{q}_3 \end{bmatrix}$$

$$p_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} \dfrac{\mathbf{q}_1 P_i}{\mathbf{q}_3 P_i} \\ \dfrac{\mathbf{q}_2 P_i}{\mathbf{q}_3 P_i} \end{bmatrix} \rightarrow \overbrace{\begin{cases} u_i \mathbf{q}_3 P_i = \mathbf{q}_1 P_i \\ v_i \mathbf{q}_3 P_i = \mathbf{q}_2 P \end{cases}}^{\text{Is this a linear system of equations?}}$$

No! why?

General Calibration Problem

$$\begin{bmatrix} u_i \\ v_i \end{bmatrix} = \begin{bmatrix} \dfrac{\mathbf{q}_1 \, \mathrm{P_i}}{\mathbf{q}_3 \, \mathrm{P_i}} \\ \dfrac{\mathbf{q}_2 \, \mathrm{P_i}}{\mathbf{q}_3 \, \mathrm{P_i}} \end{bmatrix} \longrightarrow X = f(P)$$

measurement

parameter

f( ) is nonlinear

Typical assumptions:

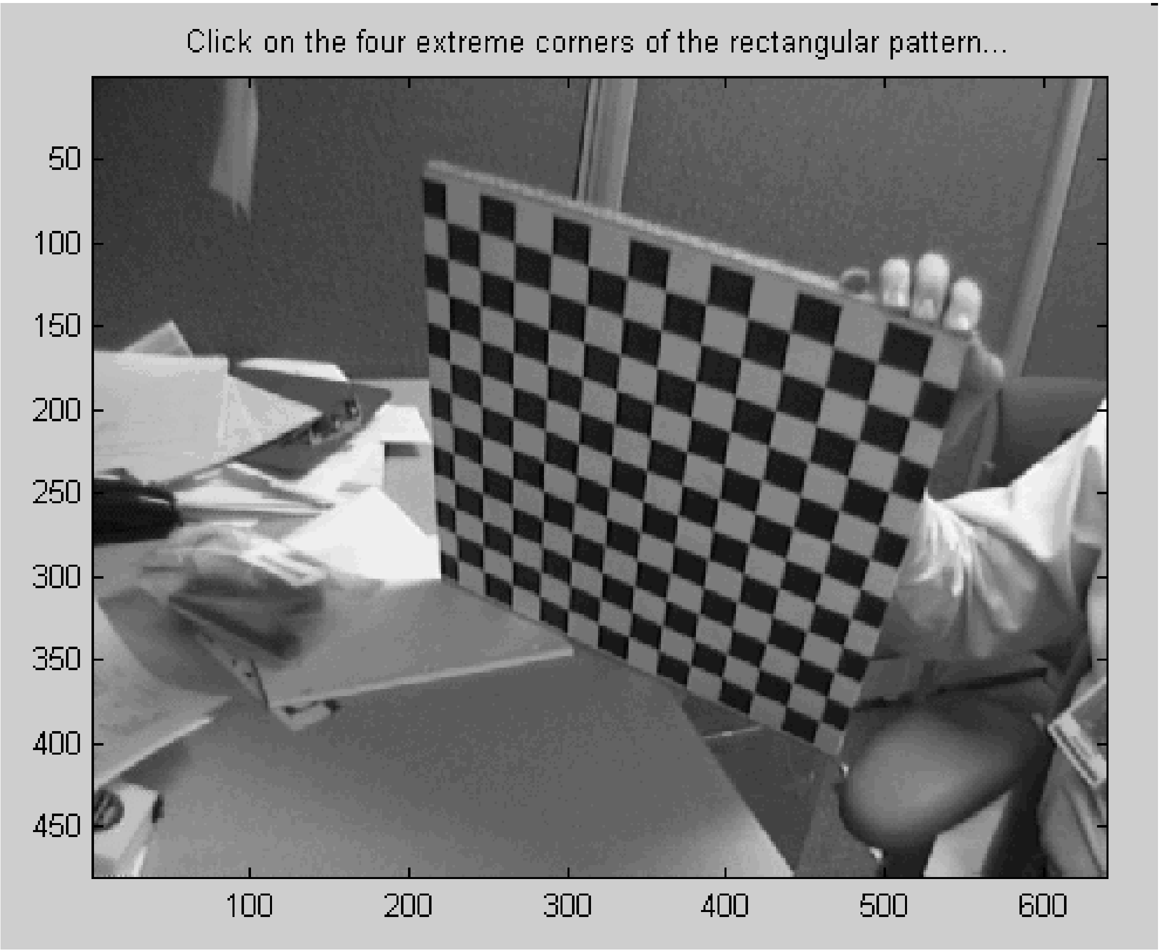
- zero-skew, square pixel
- u_o, v_o = known center of the image
- no distortion

$\longrightarrow$ Just estimate f and R, T

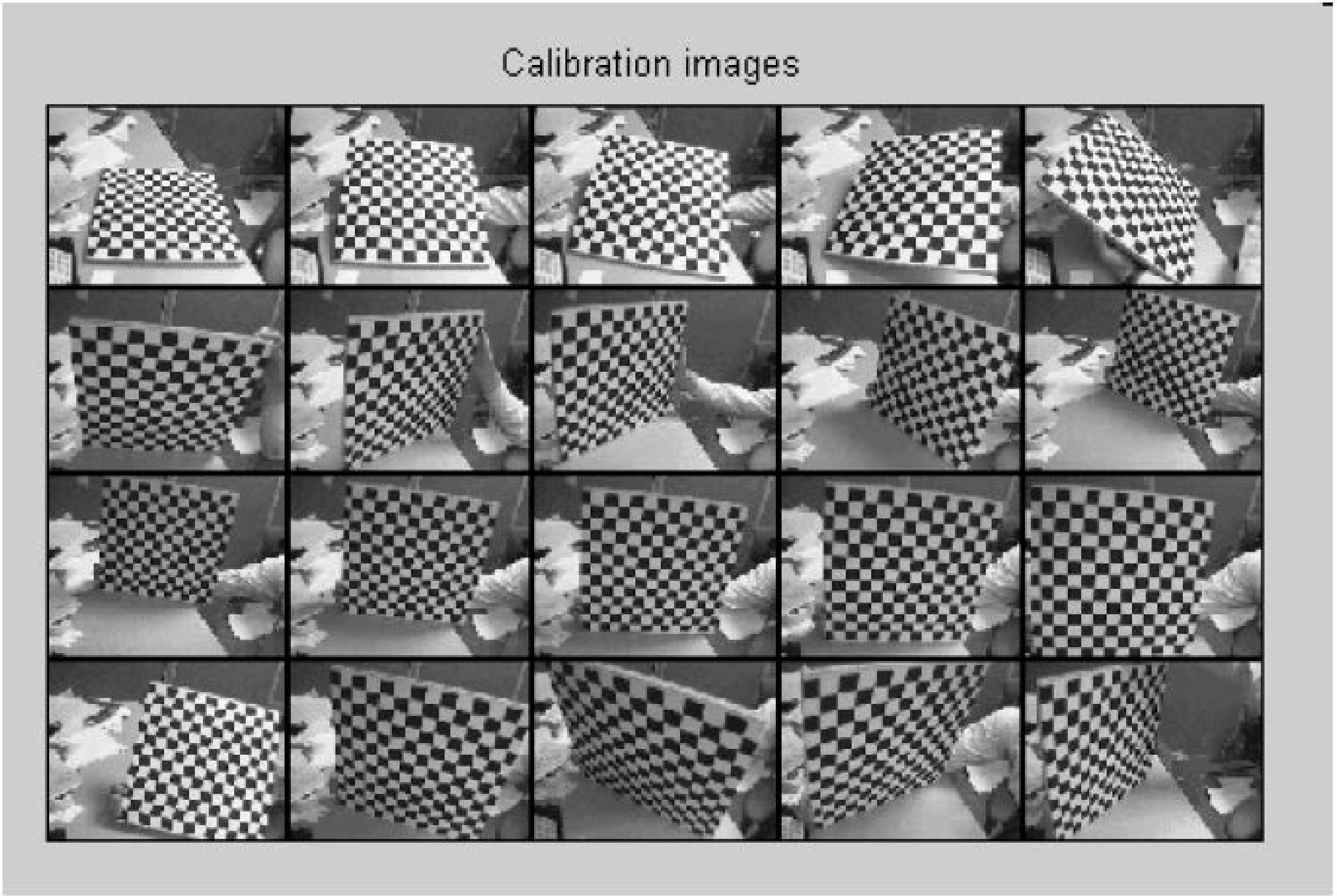
Calibration Procedure

Camera Calibration Toolbox for Matlab
J. Bouguet – [1998-2000]

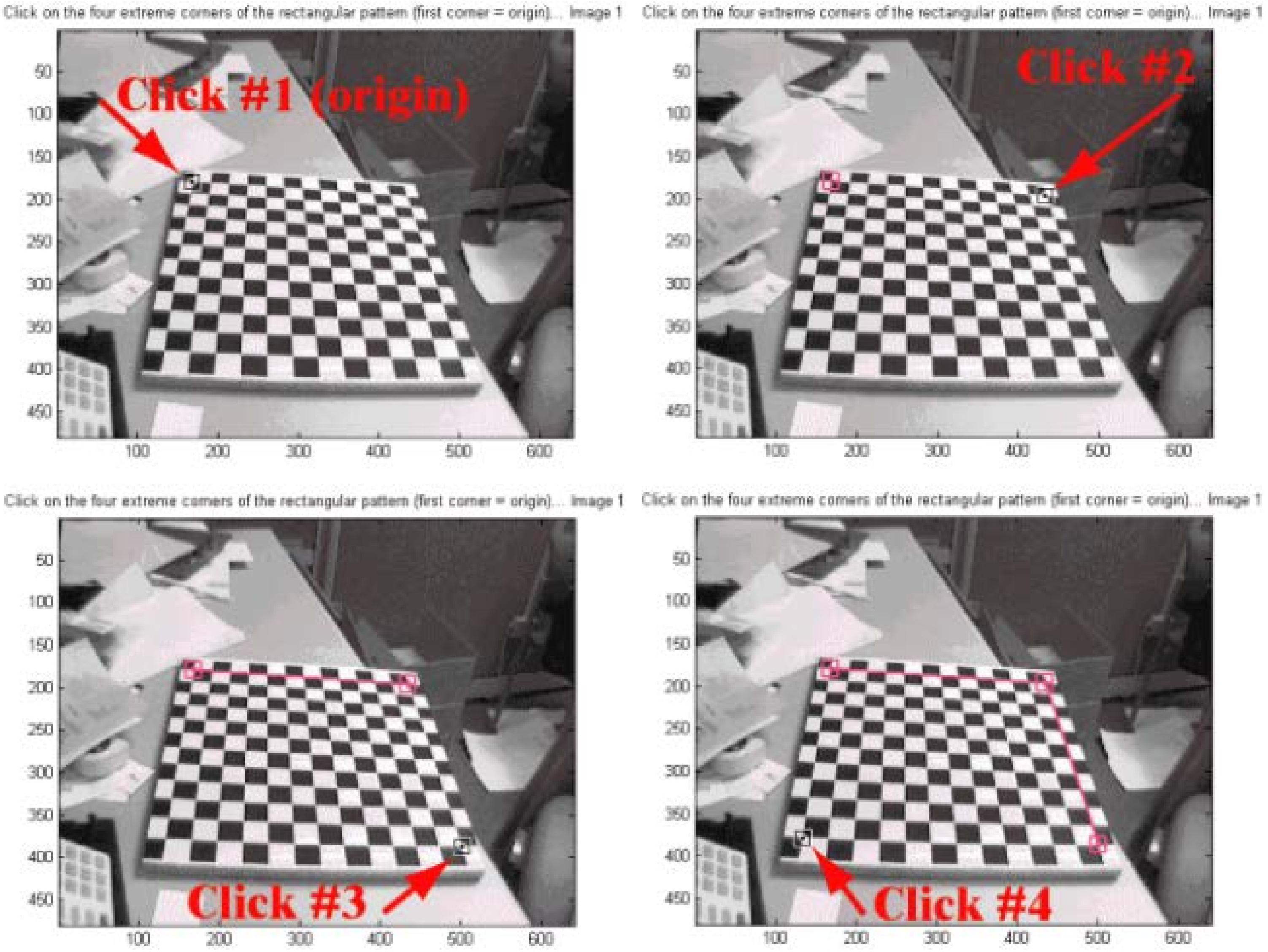
http://www.vision.caltech.edu/bouguetj/calib_doc/index.html#examples



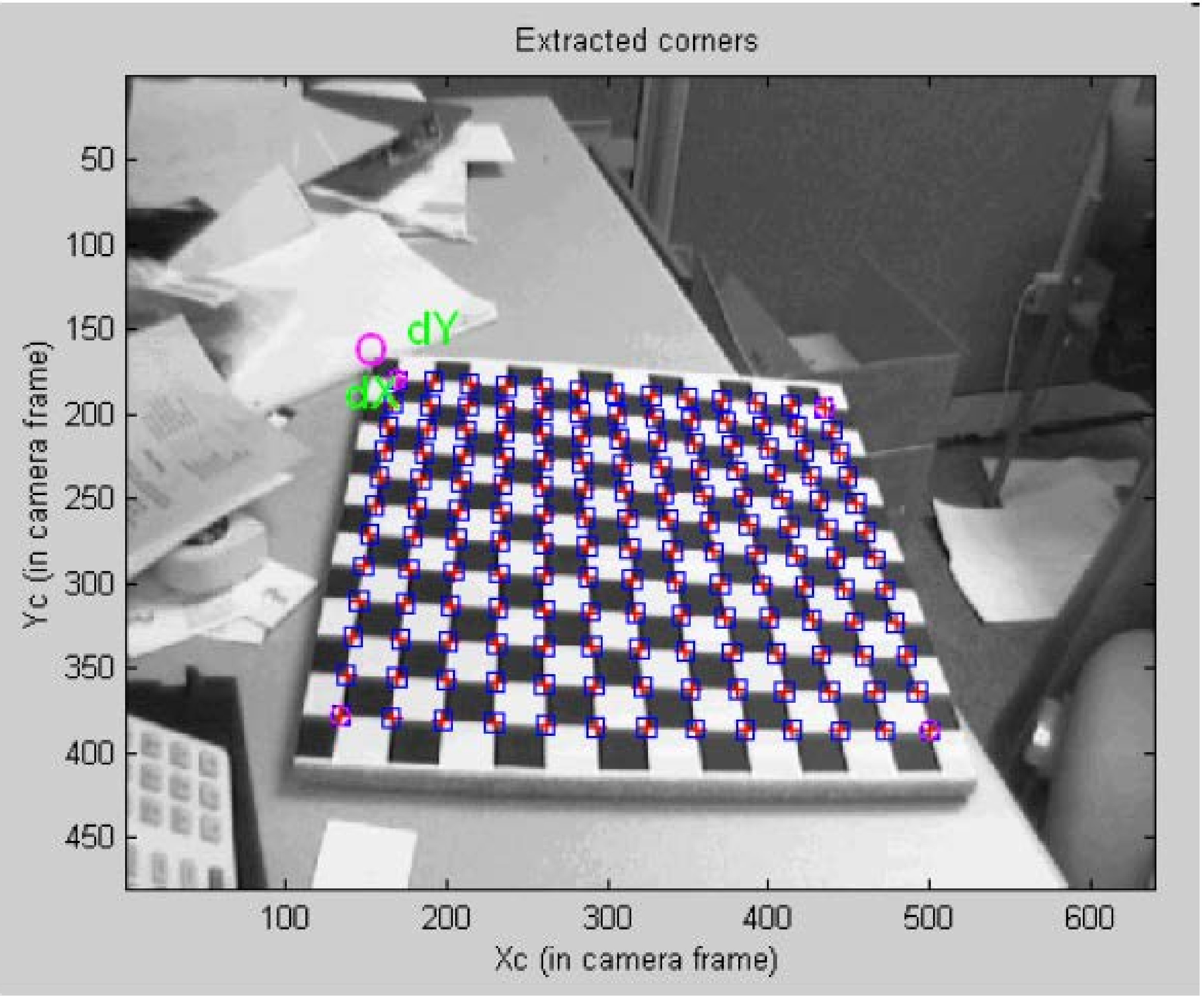
Calibration Procedure



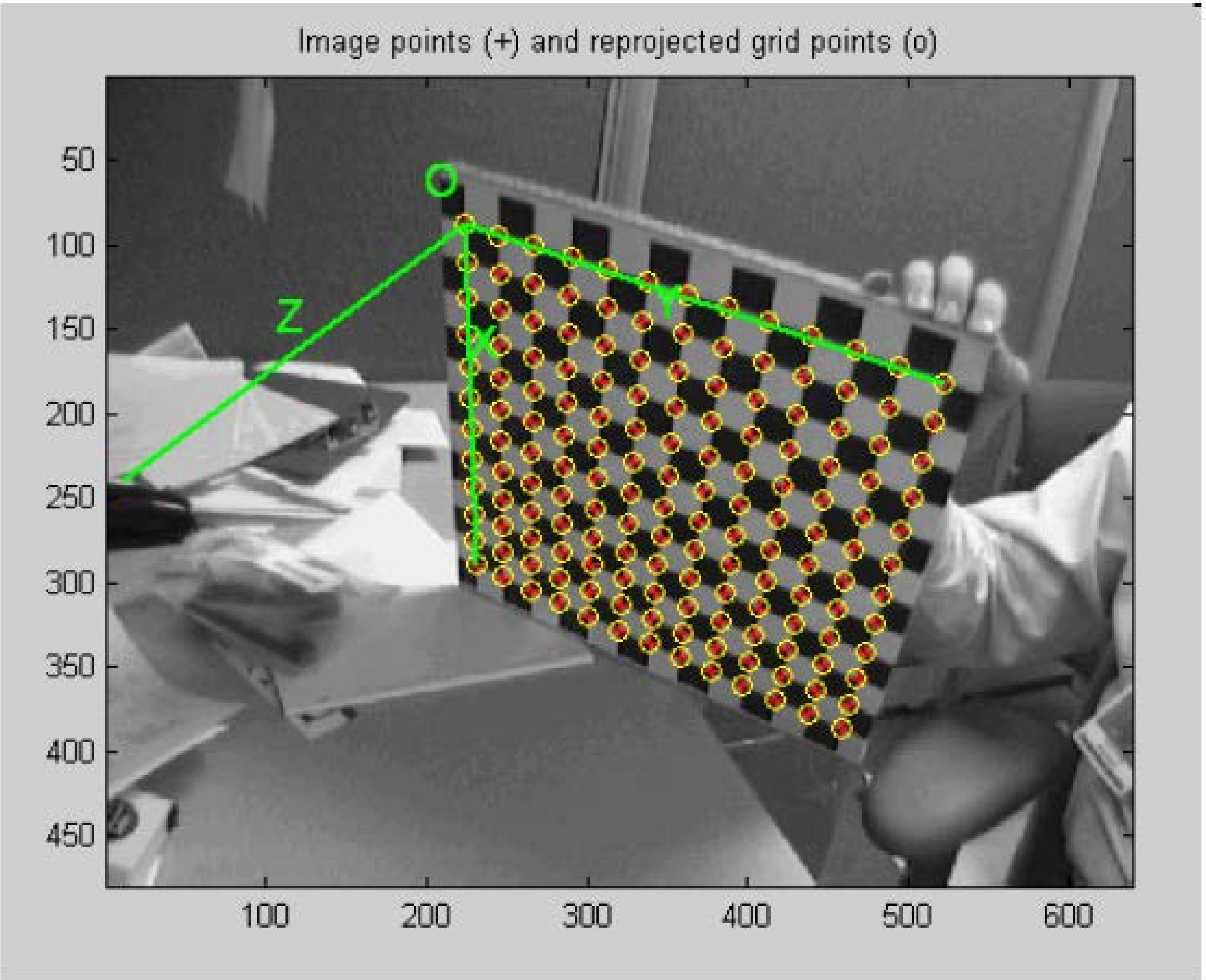
Calibration Procedure



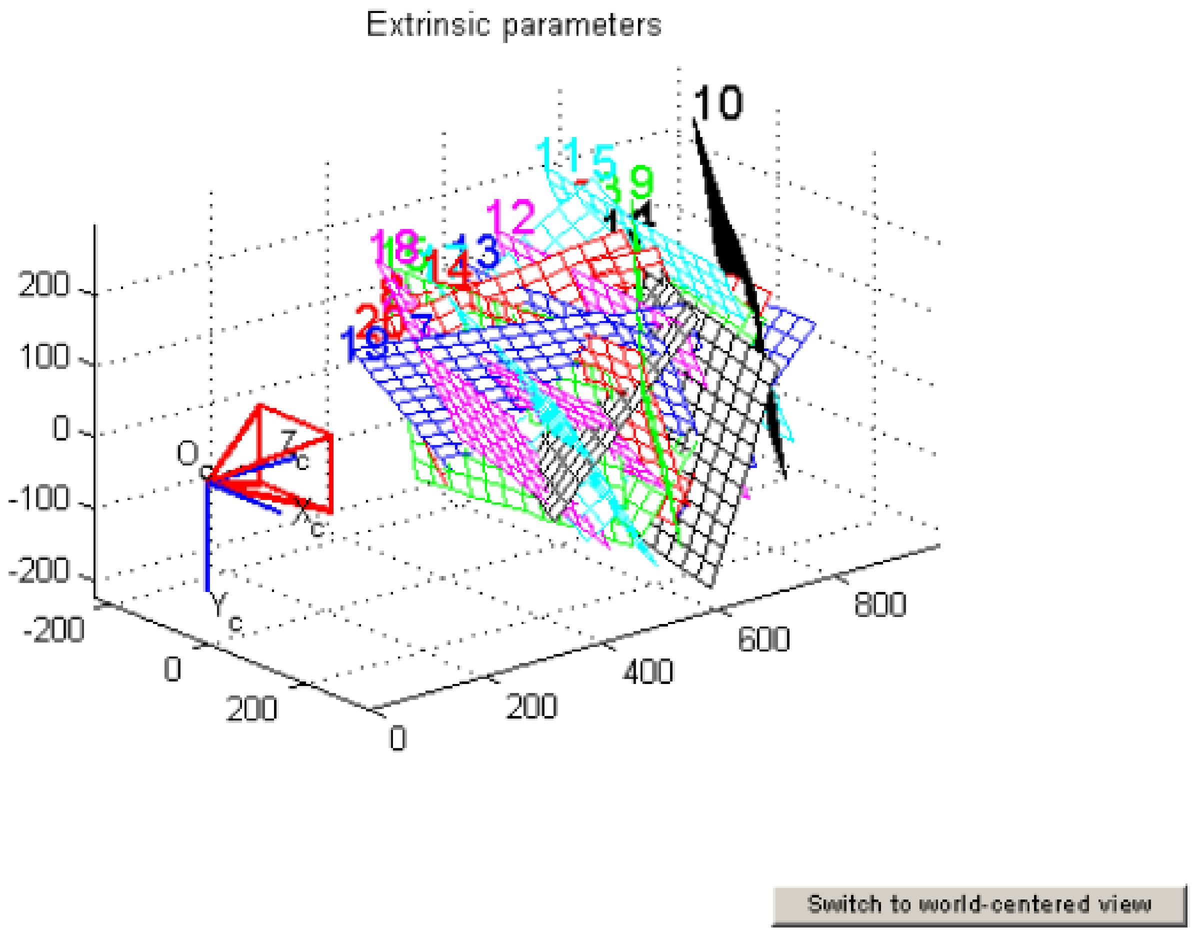
Calibration Procedure



Calibration Procedure



Calibration Procedure



Calibration Procedure