Computer Vision

8. Camera Models

I-Chen Lin

College of Computer Science

National Yang Ming Chiao Tung University

Objective

- Geometric camera models
  - Intrinsic and extrinsic parameters
  - Projection equations
- Least square methods
- Geometric camera calibration
  - Linear and non-linear methods

Textbook:

• David A. Forsyth and Jean Ponce, Computer Vision: A Modern Approach, Prentice Hall, New Jersey, (Ed1. 2003, Ed2, 2012).

Plenty of slides are modified from the reference lecture notes or project pages:

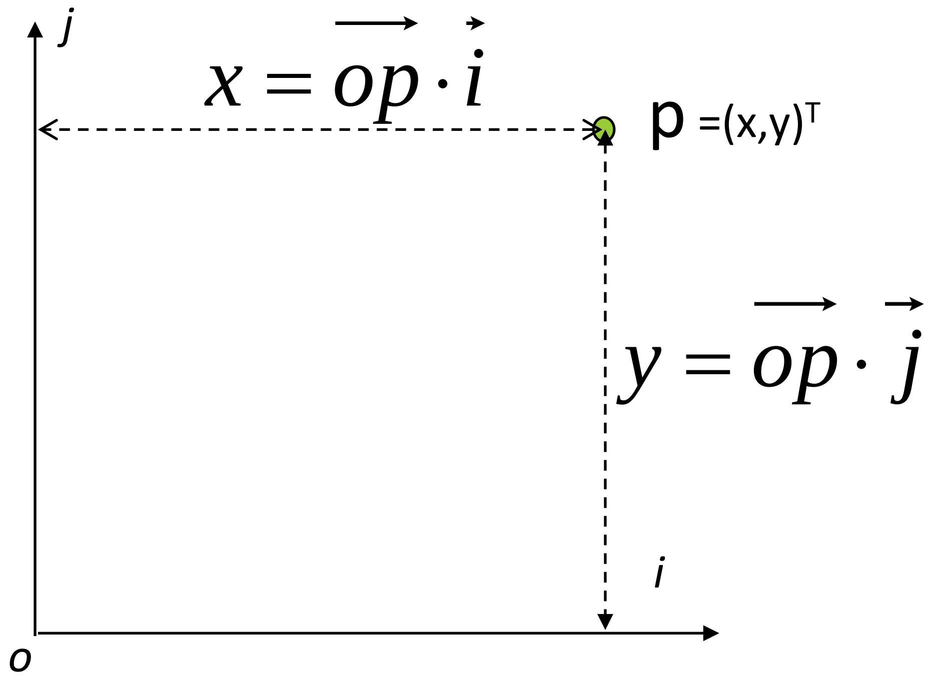
• Prof. J. Rehg, Computer Vision, Georgia Inst. of Tech.

• Prof. T. Darrell, Computer Vision and Applications, MIT.

• Prof. D.A. Forsyth, Computer Vision, UIUC.

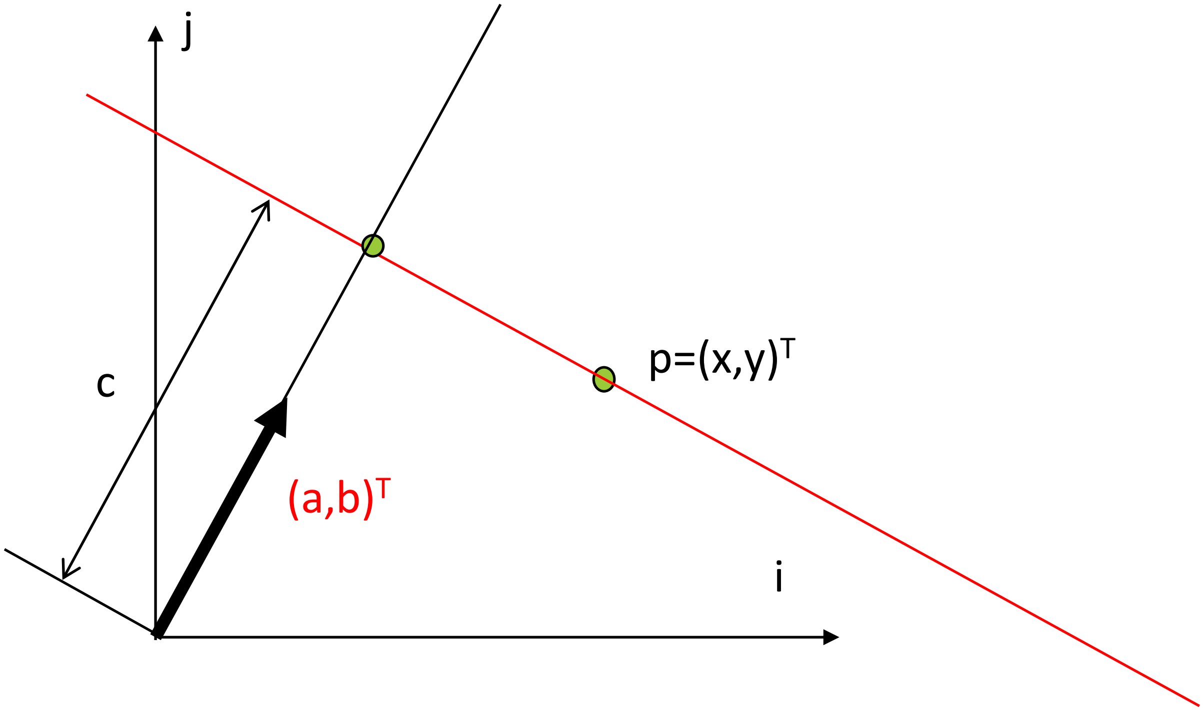
2D coordinate frames & points

- Coordinates x and y
- For a more general coordinate representation, we usually use a vector form.



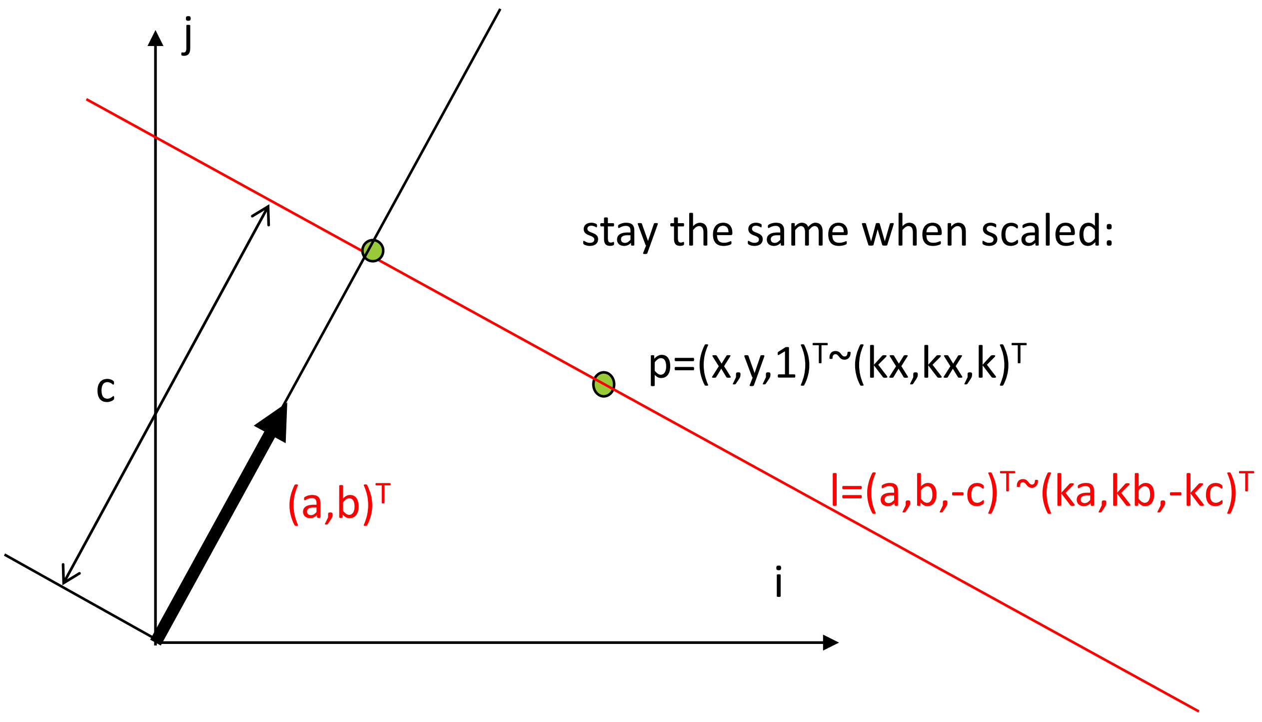
2D lines

- Line l: $ax+by=c$ <-> $(a,b)^T(x,y)=c$



Homogeneous coordinates

- Uniform treatment of points and lines
- Line-point incidence: $\boldsymbol{l}^{T}p=0$

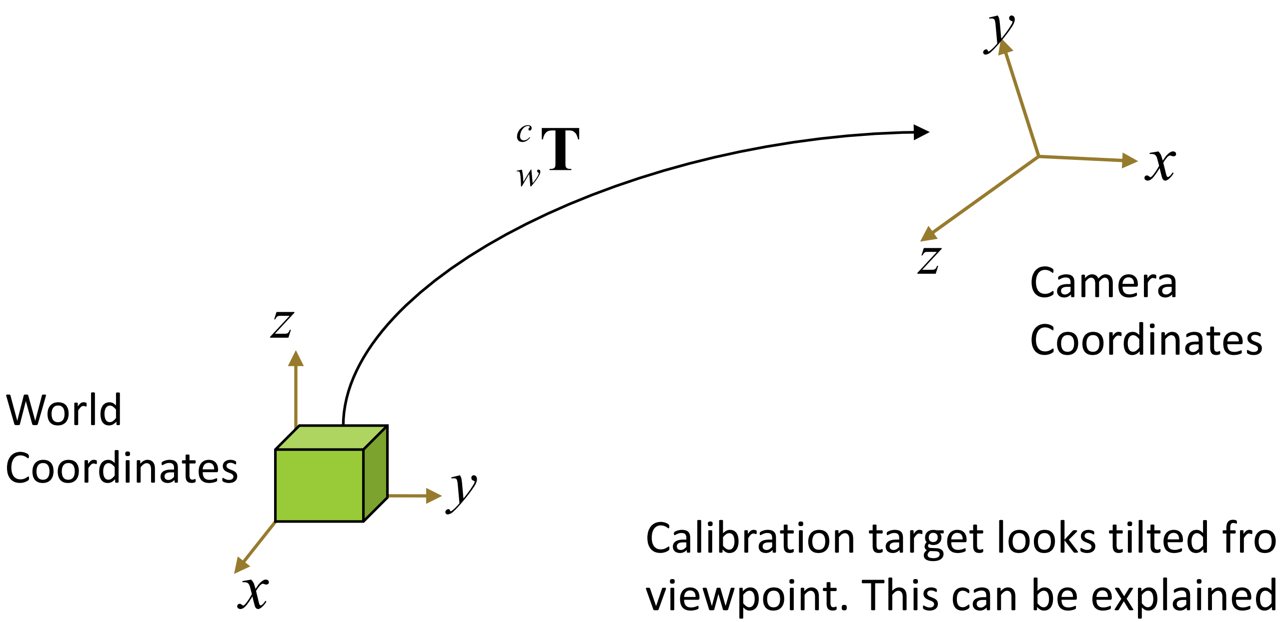


Homogeneous coordinates (cont.)

- Furthermore, …
  - We use homogenous coordinates to combine rotation and translation into same framework: matrix transformation.
  - It allows easy transformation between “frames” – common between computer vision and graphics.

Camera pose

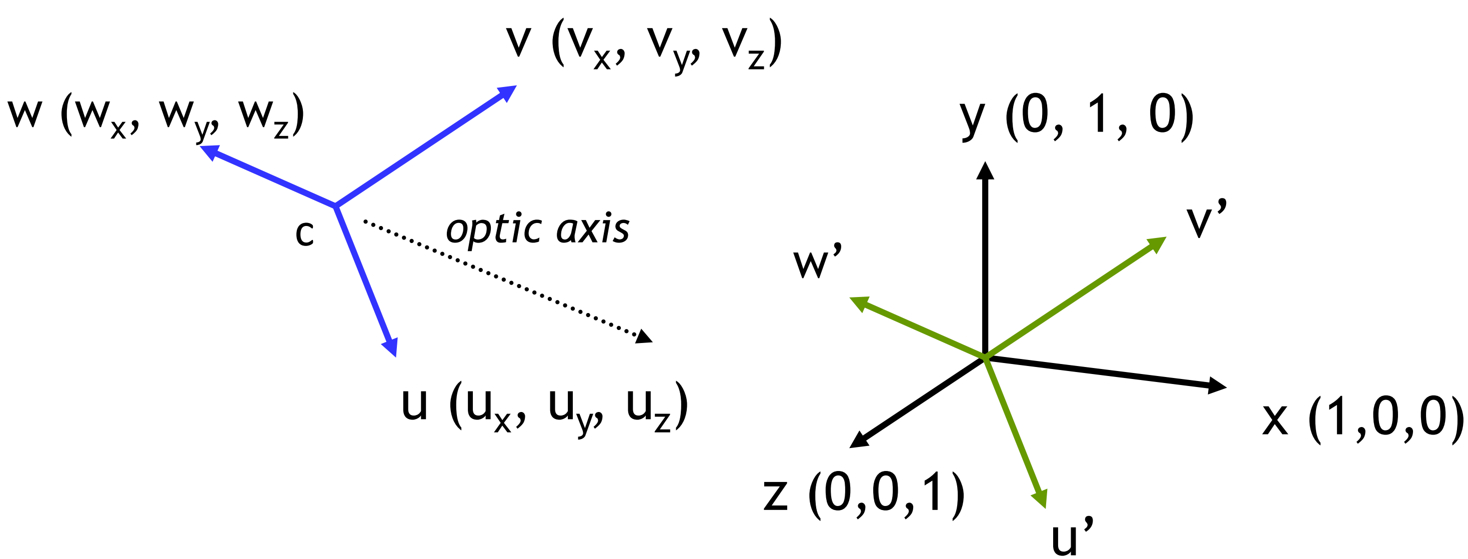
- To apply the camera model, objects in the scene must be expressed in *camera coordinates*.



Calibration target looks tilted from camera viewpoint. This can be explained as a difference in coordinate systems.

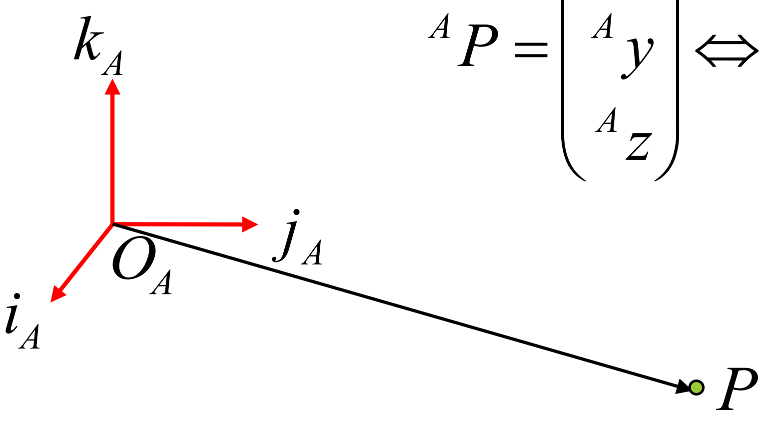
Rigid body transformations

- Need a way to specify the six degrees-of-freedom of a rigid body.
- 3 rotation and 3 translation DOFs.
- *R*, *t*: the extrinsic parameters.



Notations

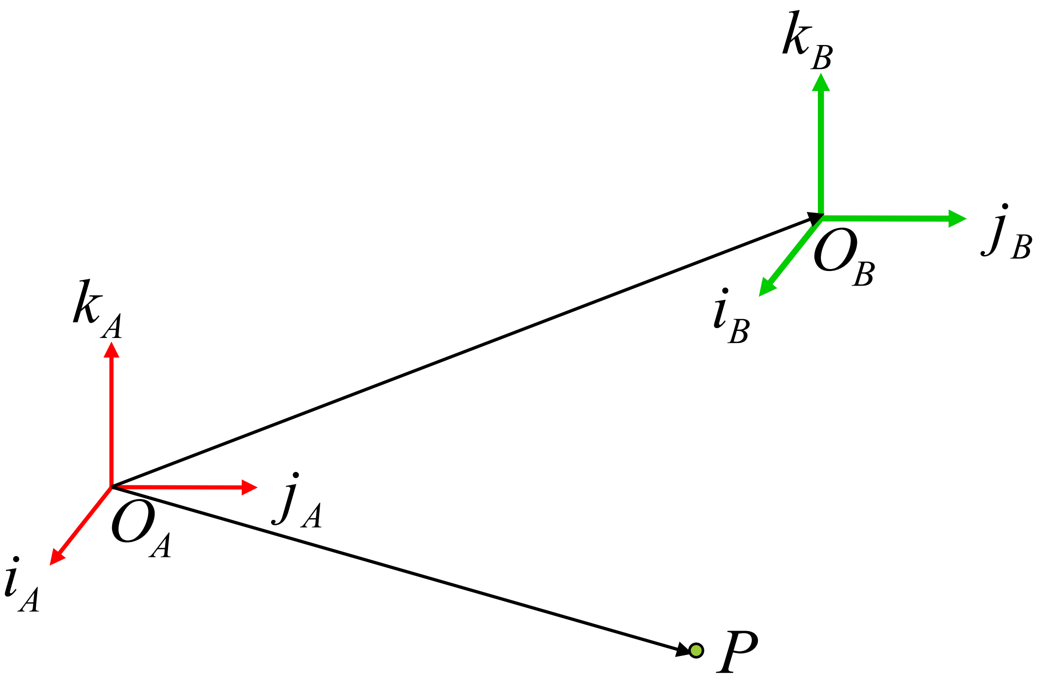
- Superscript references coordinate frame
- ^{A}P is coordinates of P in frame A
- ^{B}P is coordinates of P in frame B



$$^{A}P = \begin{pmatrix} ^{A}x \\ ^{A}y \\ ^{A}z \end{pmatrix} \Leftrightarrow \overline{OP} = \left(^{A}x \bullet \overline{i_A}\right) + \left(^{A}y \bullet \overline{j_A}\right) + \left(^{A}z \bullet \overline{k_A}\right)$$

Translation

$$ {}^{B}P = {}^{A}P + {}^{B}(O_A) $$



Translation

- Using homogeneous coordinates, translation can be expressed as a matrix multiplication.

$$ {}^{B}P = {}^{A}P + {}^{B}O_{A} $$

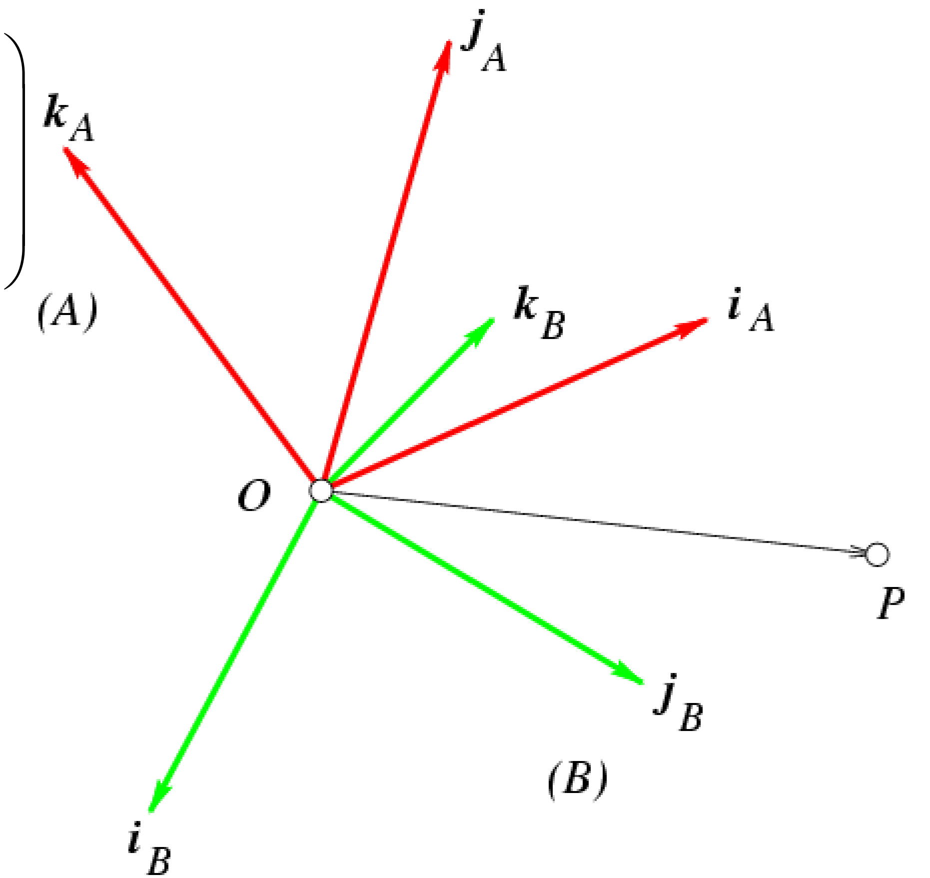
$$ \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} I & {}^{B}O_{A} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} $$

- Translation is commutative

Rotation

From the aspect of frames

$$\overrightarrow{OP} = \begin{pmatrix} i_A & j_A & k_A \end{pmatrix} \begin{pmatrix} {}^{A}x \\ {}^{A}y \\ {}^{A}z \end{pmatrix} = \begin{pmatrix} i_B & j_B & k_B \end{pmatrix} \begin{pmatrix} {}^{B}x \\ {}^{B}y \\ {}^{B}z \end{pmatrix}$$



$$ {}^{B}P = {}^{B}_{A}R \, {}^{A}P $$

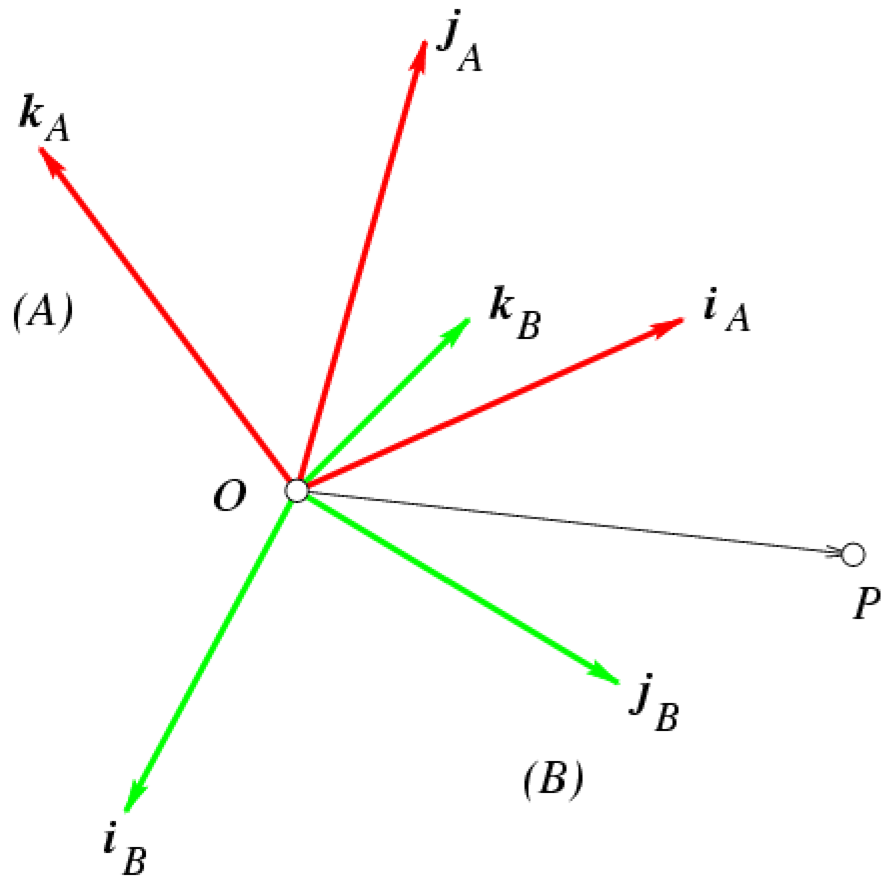
${}^{B}_{A}R$ means describing frame A in The coordinate system of frame B

Rotation (from frame A to B)

$$ {}^{B}_{A}R = \begin{bmatrix} \mathbf{i}_A.\mathbf{i}_B & \mathbf{j}_A.\mathbf{i}_B & \mathbf{k}_A.\mathbf{i}_B \\ \mathbf{i}_A.\mathbf{j}_B & \mathbf{j}_A.\mathbf{j}_B & \mathbf{k}_A.\mathbf{j}_B \\ \mathbf{i}_A.\mathbf{k}_B & \mathbf{j}_A.\mathbf{k}_B & \mathbf{k}_A.\mathbf{k}_B \end{bmatrix} $$

$$ = \begin{bmatrix} {}^{B}\mathbf{i}_A & {}^{B}\mathbf{j}_A & {}^{B}\mathbf{k}_A \end{bmatrix} $$

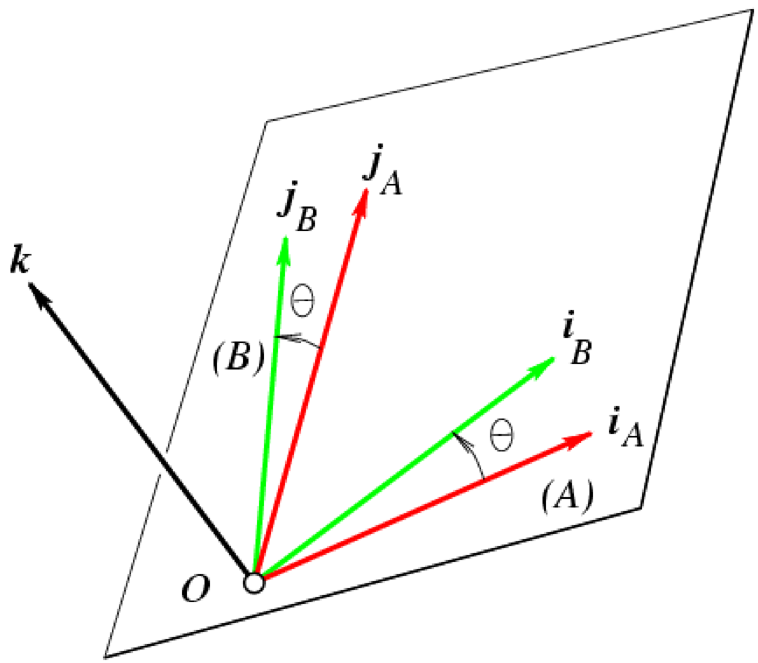
$$ = \begin{bmatrix} {}^{A}\mathbf{i}_B^T \\ {}^{A}\mathbf{j}_B^T \\ {}^{A}\mathbf{k}_B^T \end{bmatrix} $$



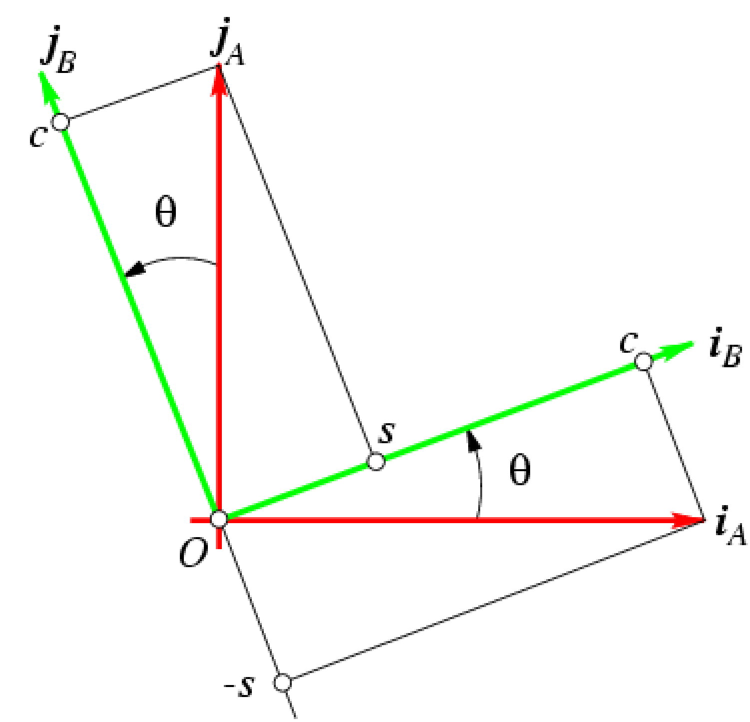
Orthogonal matrix: $R^{-1} = R^T$

Example: Rotation about z axis

From the aspect of Euler angles



What is the rotation matrix?



Combine 3 to get arbitrary rotation

- Euler angles: Z, X', Y''
- Heading, pitch roll: world Z, new X, new Y
- Three basic matrices: order matters, but we'll probably not focus on that

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

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

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

Remind: applying coordinate rotation φ is equal to applying rotation – φ to objects.

Rotation in homogeneous coordinates

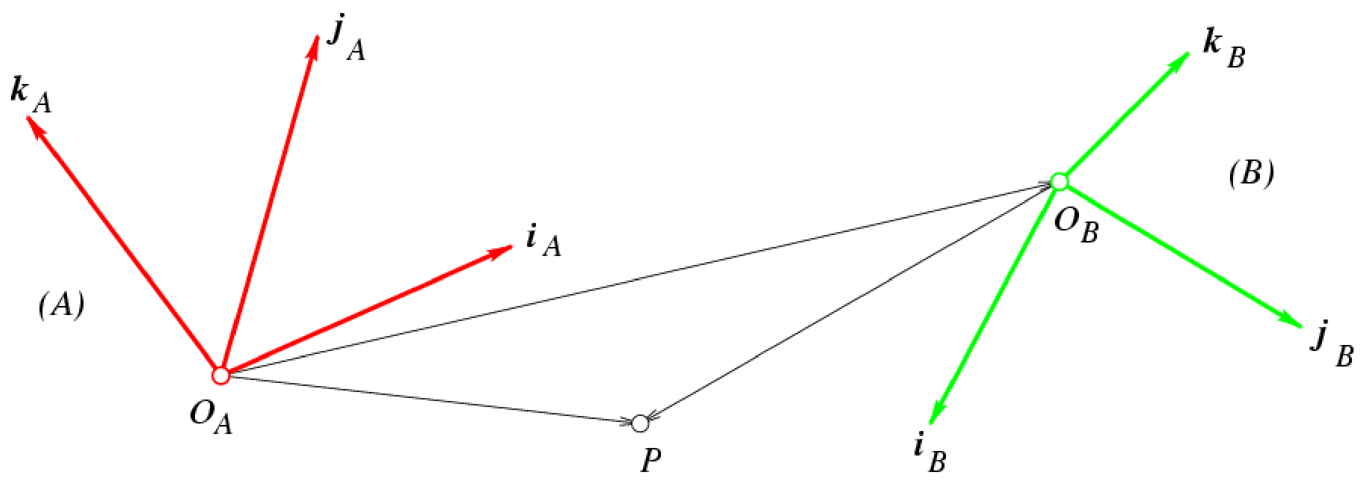
- Using homogeneous coordinates, rotation can be expressed as a matrix multiplication.

$$ {}^{B}P = {}^{B}_{A}R \, {}^{A}P $$

$$ \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} {}^{B}_{A}R & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix} $$

- Rotation is not commutative

Rigid transformations



$$^{B}P = {}^{B}_{A}R\,{}^{A}P + {}^{B}O_{A}$$

Rigid transformations (cont.)

- Unified treatment using homogeneous coordinates.

$$\begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = \begin{bmatrix} 1 & {}^{B}O_{A} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{B}_{A}R & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} {}^{B}_{A}R & {}^{B}O_{A} \\ \mathbf{0}^{T} & 1 \end{bmatrix} \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

Invertible!

$$\longrightarrow \begin{bmatrix} {}^{B}P \\ 1 \end{bmatrix} = {}^{B}_{A}T \begin{bmatrix} {}^{A}P \\ 1 \end{bmatrix}$$

Affine camera model

Pretend depth is constant (often OK !) , for some simple applications

$$\hat{u} = \frac{X}{Z_r}$$

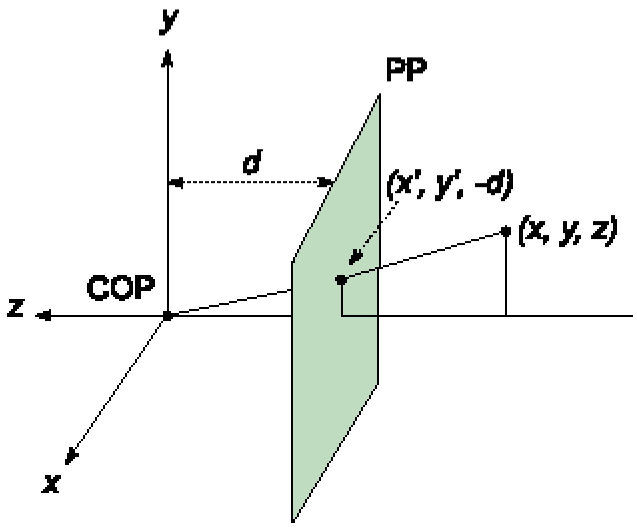
$$\hat{v} = \frac{Y}{Z_r}$$

Can also be written as a linear tra nsformatio n :

$$\begin{bmatrix} u \\ v \\ w \end{bmatrix} = \frac{1}{Z_r} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & Z_r \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

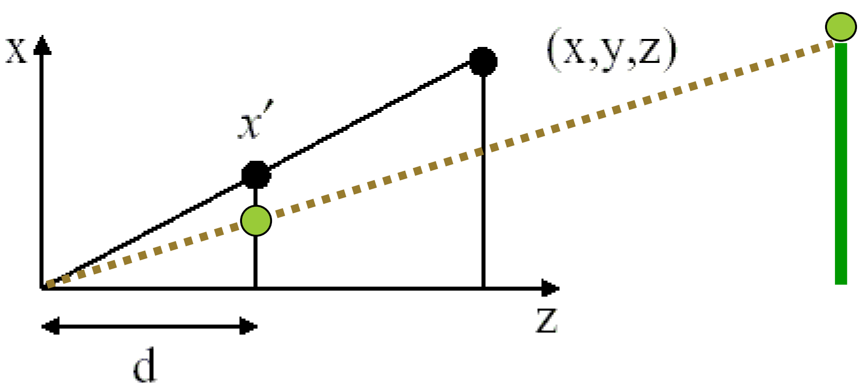
Perspective projection

- Recall perspective projection



Using similar triangles gives:

http://commons.wikimedia.org/wiki/File:Taiwan_HighSpeedRail_Train_Business_Class_Car.JPG



Perspective camera model

- Linear transformation of perspective projection coordinate.

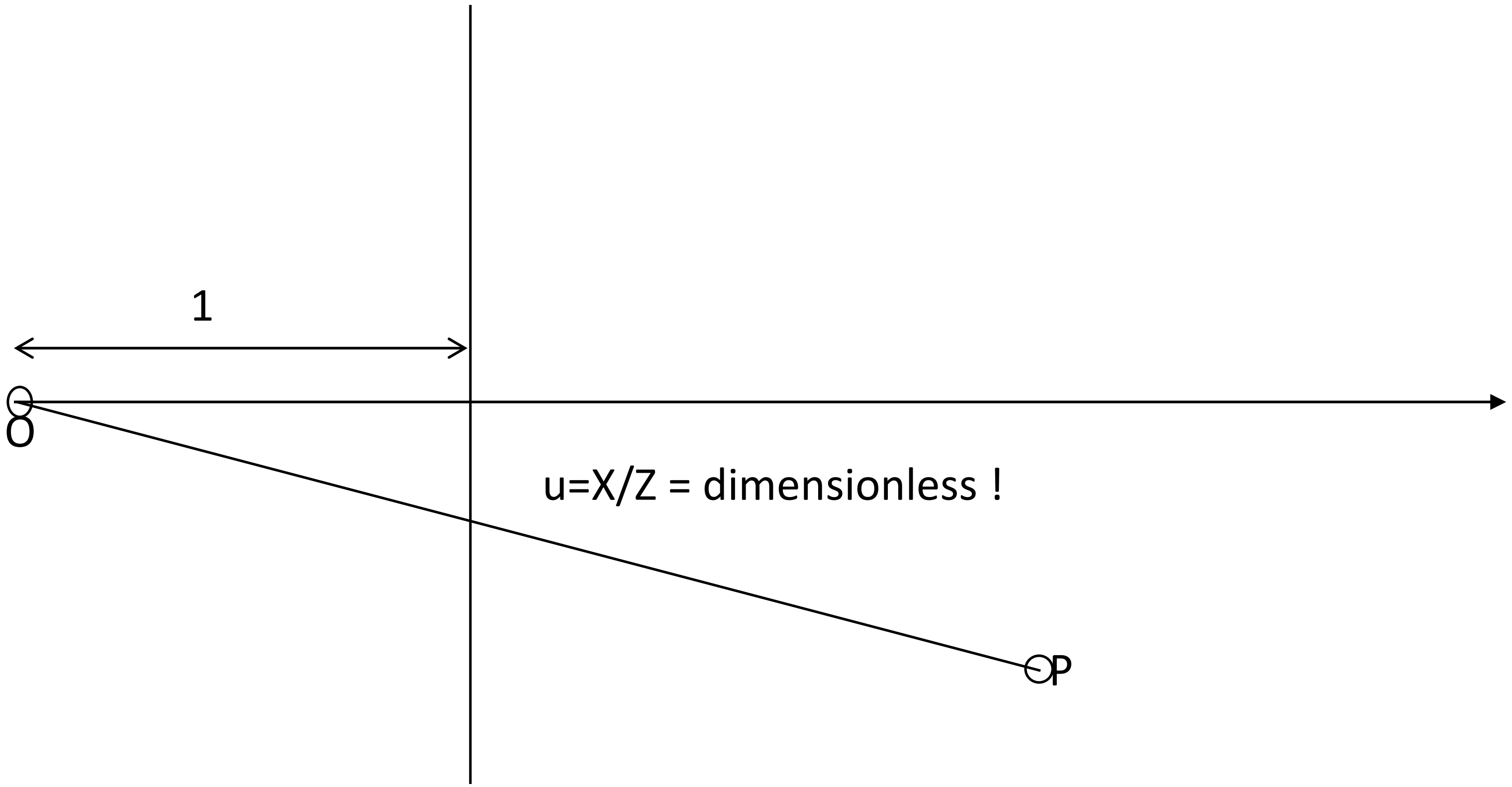
$$p = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = [I \quad 0]P = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Recover image (normalized) coordinate by projection.

$$\hat{u} = \frac{u}{w} = \frac{X}{Z}$$

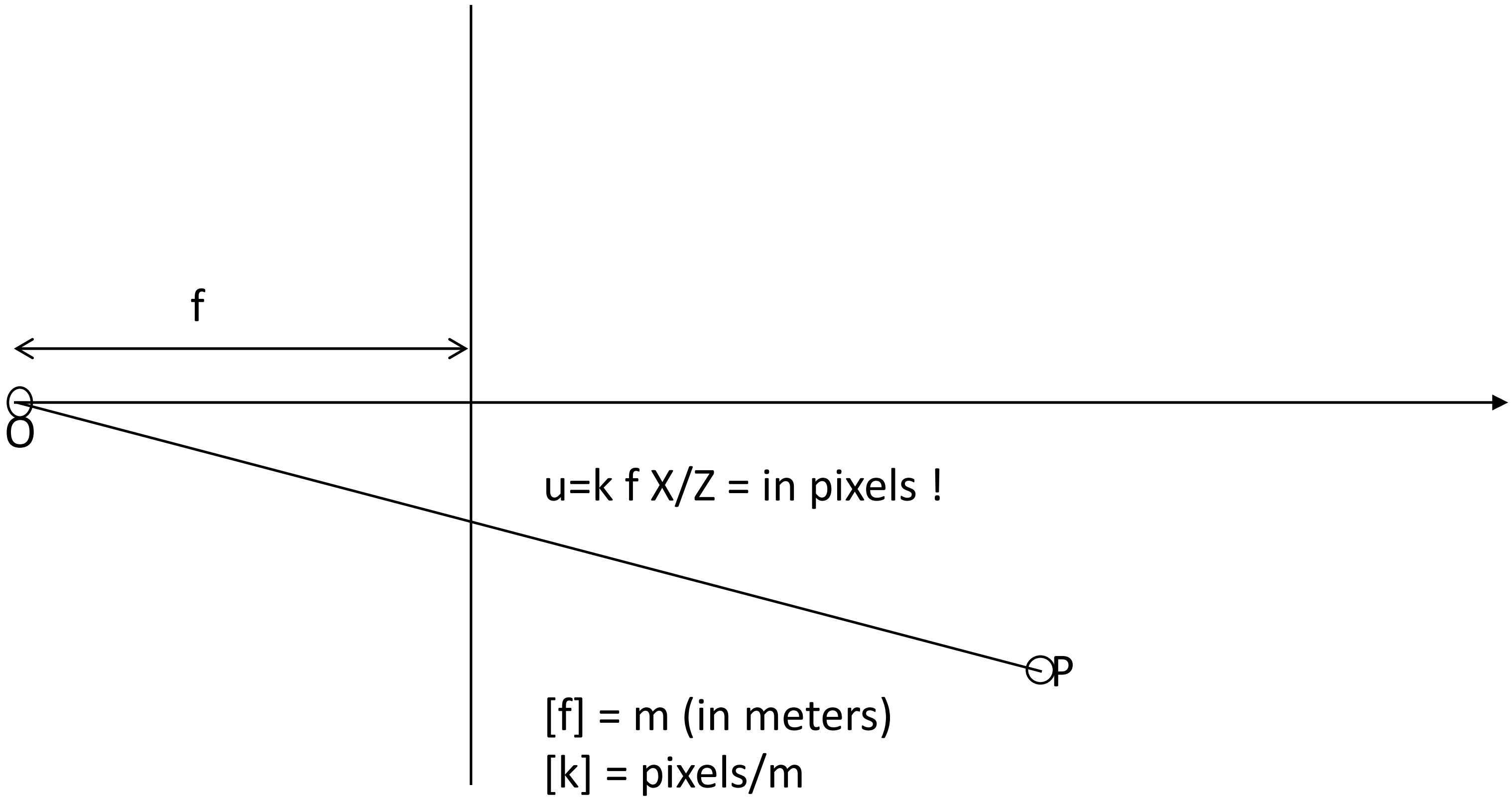
$$\hat{v} = \frac{v}{w} = \frac{Y}{Z}$$

Normalized Image coordinates



Pixel units

Pixels are on a grid of a certain dimension

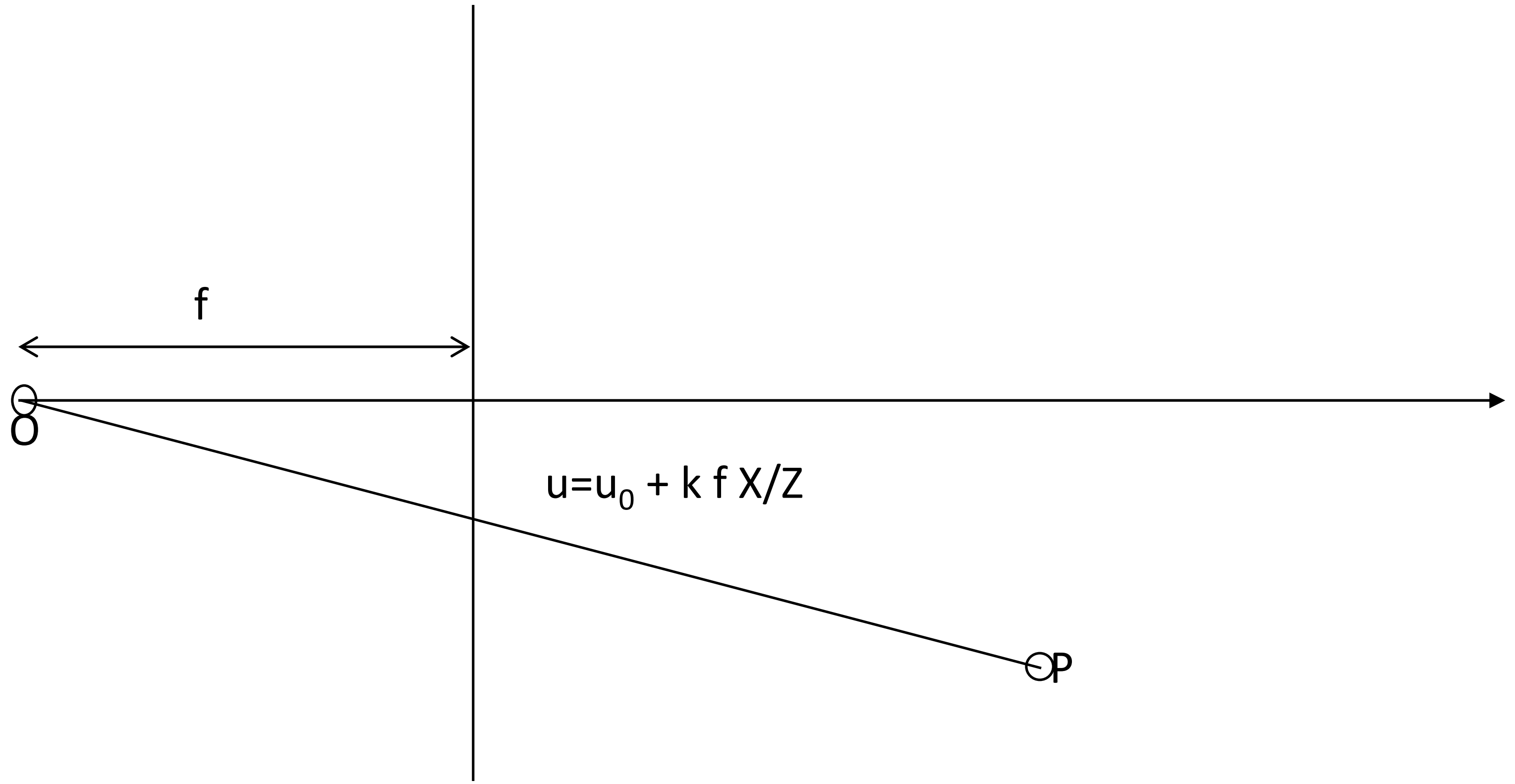


$u = k\, f\, X/Z$ = in pixels !

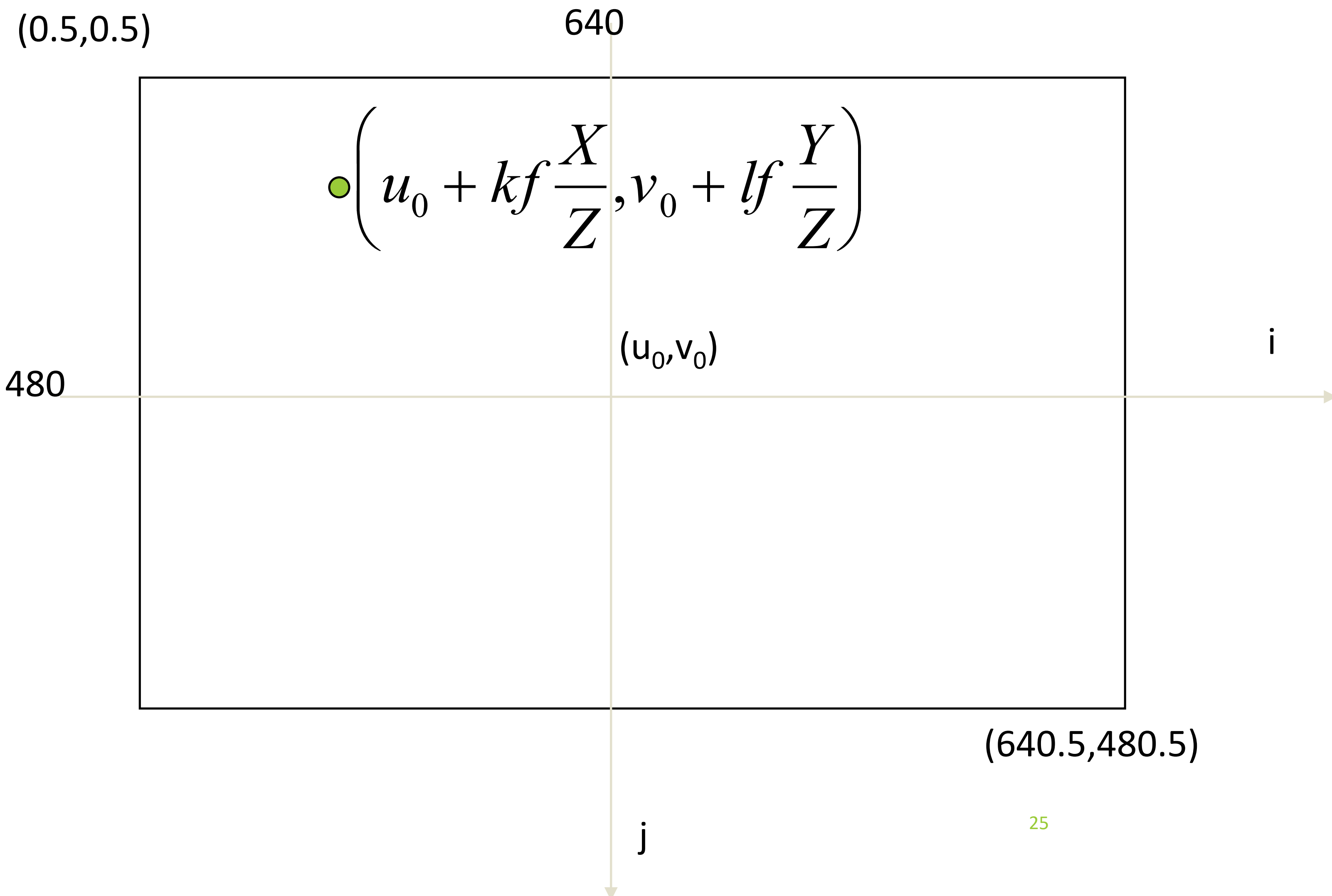
$[f]$ = m (in meters)
$[k]$ = pixels/m

Pixel coordinates

We put the pixel coordinate origin on topleft



Pixel coordinates in 2D



Intrinsic parameters (in references)

3×3 Calibration Matrix K

$$p = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K[I \quad 0]P = \begin{bmatrix} \alpha & s & u_0 \\ & \beta & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

skew

Recover image (Euclidean) coordinates by normalizing:

$$\hat{u} = \frac{u}{w} = \frac{\alpha X + sY}{Z} + u_0$$

$$\hat{v} = \frac{v}{w} = \frac{\beta Y}{Z} + v_0$$

5 Degrees of Freedom !

Intrinsic parameters (in the textbook)

3×3 Calibration Matrix K

$$p=\begin{bmatrix}u\\ v\\ 1\end{bmatrix}=\frac{1}{z}K[I \quad 0]P=\begin{bmatrix}\alpha & -\alpha\cot\theta & u_0\\ & \dfrac{\beta}{\sin\theta} & v_0\\ & & 1\end{bmatrix}\begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&1&0\end{bmatrix}\begin{bmatrix}x\\ y\\ z\\ 1\end{bmatrix}$$

Recover image (Euclidean) coordinates by normalizing :

$$u=\frac{\alpha x-\alpha\cot\theta y}{z}+u_0$$

$$v=\frac{\beta y}{z\sin\theta}+v_0$$

Combining intrinsic and extrinsic param.

- Perspective projection mapping (including intrinsic and extrinsic parameters).

$$p = \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \frac{1}{z'} K \begin{bmatrix} I & 0 \end{bmatrix} T P = \begin{bmatrix} \alpha & -\alpha \cot\theta & u_0 \\ & \dfrac{\beta}{\sin\theta} & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} {}^c_w R & {}^c_w O \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

$$= \frac{1}{z'} K \begin{bmatrix} {}^c_w R & {}^c_w O \end{bmatrix} P = \frac{1}{z'} M P$$

5+6 DOF = 11 !

How to estimate these parameters?

Linear least-squares methods

- *P* linear equations in *q* unknowns:
- *Ux = y*

$$U = \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1q} \\ u_{21} & u_{22} & \cdots & u_{2q} \\ \cdots & \cdots & \cdots & \cdots \\ u_{p1} & u_{p2} & \cdots & u_{pq} \end{bmatrix}, \quad x = \begin{bmatrix} x_1 \\ x_2 \\ \cdots \\ x_q \end{bmatrix}, \quad y = \begin{bmatrix} y_1 \\ y_2 \\ \cdots \\ y_p \end{bmatrix}$$

- When p < q: a (q-p) dimensional vector space
- When p = q: a unique solution
- When p > q: overconstrained system

Normal equations and pseudo-inverse

- min $E = |Ux\text{-}y|^2 = e^Te$, where e = $Ux\text{-}y$.
- The minimum E occurs when the derivatives are zeros.
- We define vector c_j = the j^th column of U

$$\frac{\partial e}{\partial x_i} = \partial\left[\begin{pmatrix} c_1 & \cdots & c_q \end{pmatrix}\begin{pmatrix} x_1 \\ \cdots \\ x_q \end{pmatrix} - y\right] \Big/ \partial x_i = c_i \qquad \frac{\partial E}{\partial x_i} = 2\frac{\partial e}{\partial x_i} \cdot e = 2c_i^T(Ux - y) = 0$$

$$0 = \begin{pmatrix} c_1^T \\ \cdots \\ c_q^T \end{pmatrix}(Ux - y) = U^T(Ux - y) \Leftrightarrow U^TUx = U^Ty$$

$$\boxed{x = \left(U^TU\right)^{-1}U^Ty}$$

Numerical issue: QR or SVD-based methods are more reliable

min Ux, subject to $|x|=1$

- Assume $y=0$, $E=|Ux|^2 = x^T U^T U x$.
- $U^T U$ is symmetric positive semidefinite :
  - $U^T U$'s eigenvalues $0 \le \lambda_1 \le \ldots \le \lambda_q$.
  - $U^T U$ can be decomposed as QDQ^{-1}, where Q and D consist of eigenvectors and eigenvalue respectively.
- Unit vector x can be represented in terms of eigen vector e_i:
  - $x = \mu_1 e_1 + \mu_2 e_2 + \ldots + \mu_q e_q$ and $\mu_1^2 + \ldots. + \mu_q^2 = 1$
- $E(x) - E(e_1) = x^T U^T U x - e_1^T U^T U e_1 = \lambda_1 \mu_1^2 + \ldots. + \lambda_q \mu_q^2 - \lambda_1$

$$\ge \lambda_1 (\mu_1^2 + \ldots. + \mu_q^2 - 1) = 0$$

$$\boxed{\text{The } x \text{ that minimize } E \text{ is eigenvector } e_1 \text{ of } U^T U}$$

Nonlinear least-squares methods

- *P* general equations in *q* unknowns:

$$f_1(x_1, x_2, \cdots, x_q) = 0$$
$$f_2(x_1, x_2, \cdots, x_q) = 0$$
$$\cdots$$
$$f_p(x_1, x_2, \cdots, x_q) = 0$$

- The error function E(x) = $|f(x)|^2 = \sum (f_i(x))^2$
- Taylor expansion of f_i is

$$f_i(x + \delta x) = f_i(x) + \delta x_1 \frac{\partial f_i}{\partial x_1}(x) + \cdots + \delta x_q \frac{\partial f_i}{\partial x_q}(x) + O\left((\delta x)^2\right)$$

$$\approx f_i(x) + \nabla f_i(x) \cdot \delta x$$

Nonlinear least-squares methods (cont.)

$$f(x+\delta x) \approx f(x) + \Im_{f(x)}\delta x$$

$$\Im_{f(x)} = \begin{pmatrix} \nabla f_1^T(x) \\ \cdots \\ \nabla f_p^T(x) \end{pmatrix} = \begin{pmatrix} \frac{\partial f_1}{\partial x_1}(x) & \cdots & \frac{\partial f_1}{\partial x_q}(x) \\ \cdots & \cdots & \cdots \\ \frac{\partial f_p}{\partial x_1}(x) & \cdots & \frac{\partial f_p}{\partial x_q}(x) \end{pmatrix}$$

Jacobian of f

Newton's method

- For p=q (square system), an iterative algorithm.

- Compute perturbation δx such that $f(x + \delta x) \approx 0$:

$$\Im_{f(x)} \delta x = -f(x)$$

Overconstrained system (p>q)

- Gaussian-Newton method, similar to the pseudo-inverse:

$$\Im_{f(x)}{}^{T}\Im_{f(x)}\delta x = -\Im_{f(x)}{}^{T} f(x)$$

- Levenberg-Marquardt method, to avoid degenerate pseudo-inverse of the Jacobian matrix J_f.

$$(\Im_{f(x)}{}^{T}\Im_{f(x)} + \mu I)\delta x = -\Im_{f(x)}{}^{T} f(x)$$

Camera calibration (linear approach)

- Evaluating the projection matrix *M* and camera parameters with known 3D positions P_i and estimated 2D feature points p_i (u_i, v_i).
  - Using corner detection or other filtering to extract features.

$$p = \frac{1}{z'} MP, \text{ where } M = K\begin{pmatrix} R & t \end{pmatrix}$$

m_i^T is the i^{th} row of M $\quad M = \begin{bmatrix} m_1^T \\ m_2^T \\ m_3^T \end{bmatrix}$

$$u = \frac{m_1 \cdot P}{m_3 \cdot P}, v = \frac{m_2 \cdot P}{m_3 \cdot P}$$

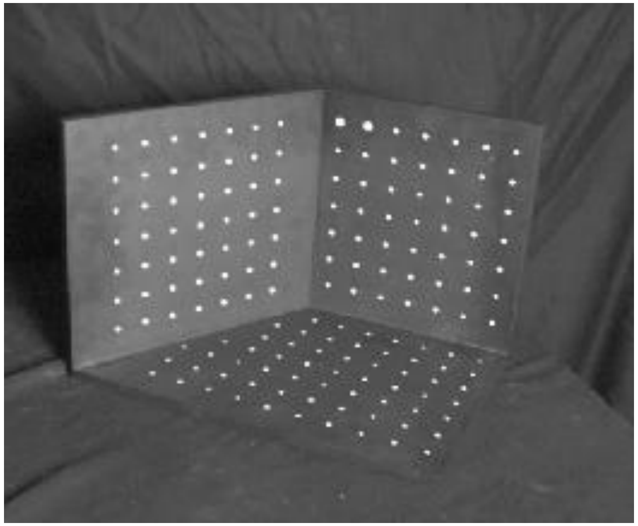


Figure from lecture note of Prof. L.Zhang, Computer Vision, U. Wisconsin-Madison.

Camera calibration (linear approach)

- With n pairs of P_i and (u_i, v_i), we have constraints:

$$(m_1 - u_i m_3) \cdot P_i = 0$$
$$(m_2 - v_i m_3) \cdot P_i = 0$$

- Reform the matrix V and unknown m
- When n > 6, we can estimate m by minimizing $|Vm|^2$

$$V = \begin{bmatrix} P_1^T & 0^T & -u_1 P_1^T \\ 0^T & P_1^T & -v_1 P_1^T \\ \cdots & \cdots & \cdots \\ P_n^T & 0^T & -u_n P_n^T \\ 0^T & P_n^T & -v_n P_n^T \end{bmatrix}_{2n\times 12} \quad \text{and } m = \begin{bmatrix} m_1 \\ m_2 \\ m_3 \end{bmatrix}_{12\times 1}$$

Estimating intrinsic and extrinsic param.

- The matrix we got M=(A b), |m|=1. There is an unknown scale factor ρ.

$$\rho\begin{pmatrix}A & b\end{pmatrix} = K\begin{pmatrix}R & t\end{pmatrix} \Leftrightarrow \rho\begin{pmatrix}a_1^T \\ a_2^T \\ a_3^T\end{pmatrix} = \begin{pmatrix}\alpha r_1^T - \alpha \cot\theta r_2^T + u_0 r_3^T \\ \dfrac{\beta}{\sin\theta} r_2^T + v_0 r_3^T \\ r_3^T\end{pmatrix}$$

$\rho = \varepsilon / \|a_3\|$ where ε =1 or -1.

$$\begin{bmatrix}\alpha & -\alpha\cot\theta & u_0 \\ & \dfrac{\beta}{\sin\theta} & v_0 \\ & & 1\end{bmatrix}\begin{bmatrix}r_1^T \\ r_2^T \\ r_3^T\end{bmatrix}$$

$$r_3 = \rho a_3$$

$$u_0 = \rho^2(a_1 \cdot a_3)$$

$$v_0 = \rho^2(a_2 \cdot a_3)$$

$$\rho^2(a_1 \times a_3) = -\alpha r_2 - \alpha\cot\theta r_1 \Rightarrow \rho^2\|a_1 \times a_3\| = \frac{\|\alpha\|}{\sin\theta}$$

$$1 + cot^2 = csc^2$$

$$\rho^2(a_2 \times a_3) = \frac{\beta}{\sin\theta} r_1 \Rightarrow \rho^2\|a_2 \times a_3\| = \frac{\|\beta\|}{\sin\theta}$$

Estimating intrinsic and extrinsic param.

$$\cos\theta = \frac{-(a_1 \times a_3)\cdot(a_2 \times a_3)}{|a_1 \times a_3||a_2 \times a_3|}$$

$$cos\theta = \frac{cot\theta}{sin\theta}$$

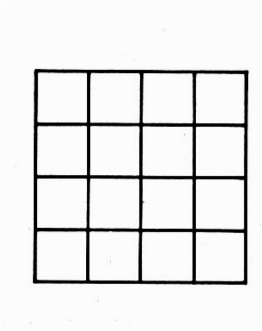
$$\alpha = \rho^2 |a_1 \times a_3| \sin\theta$$

$$\beta = \rho^2 |a_2 \times a_3| \sin\theta$$

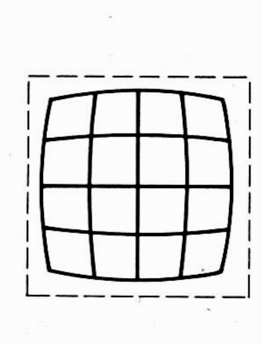
$$r_1 = \frac{(a_2 \times a_3)}{|a_2 \times a_3|}, \text{and } r_2 = r_3 \times r_1$$

Radial distortion

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



No distortion

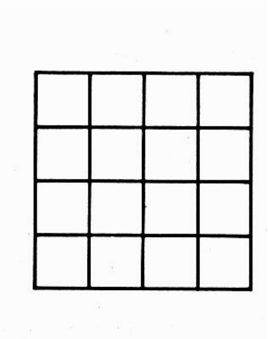


Barrel

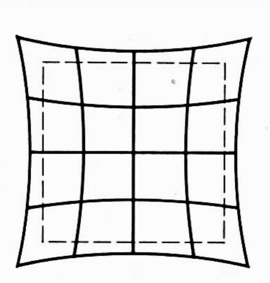
Wide Angle Lens

Figure from lecture note of Prof. L.Zhang, Computer Vision, U. Wisconsin-Madison.

Radial distortion (cont.)



No distortion



Pin cushion

Telephoto lens

Figure from lecture note of Prof. L.Zhang, Computer Vision, U. Wisconsin-Madison.

Radial distortion model

- If $u_0 = v_0 = 0$, we can model the distortion as function of *d*.

$$d^2 = \hat{u}^2 + \hat{v}^2$$

$$p = \frac{1}{z}\begin{pmatrix} r(d) & 0 & 0 \\ 0 & r(d) & 0 \\ 0 & 0 & 1 \end{pmatrix} MP$$

$$r(d) = 1 + \kappa_1 d^2 + \kappa_2 d^4 + \kappa_4 d^6$$

- Need non-linear least squares for general cases.

Calibration with non-linear methods

- The multi-stage linear method can be contaminated by noises or more calibration points are required.
- Using the solution by a linear approach as initial guesses, non-linear optimization can further improve our calibration.

$$E(\xi)=\sum_{i=1}^{n}\left[\left(\tilde{u}_i(\xi)-u_i\right)^2+\left(\tilde{v}_i(\xi)-v_i\right)^2\right],$$

$$\text{where } \tilde{u}_i(\xi)=\frac{m_1(\xi)\cdot P_i}{m_3(\xi)\cdot P_i} \text{ and } \tilde{v}_i(\xi)=\frac{m_2(\xi)\cdot P_i}{m_3(\xi)\cdot P_i}$$

- We can reformulate the objective function for non-linear least square evaluation.

Calibration with non-linear methods

- We can solve the optimization by Gauss-Newton or LM methods.

- In addition to intrinsic and extrinsic parameters, other parameter (e.g. distortions) can also be included.