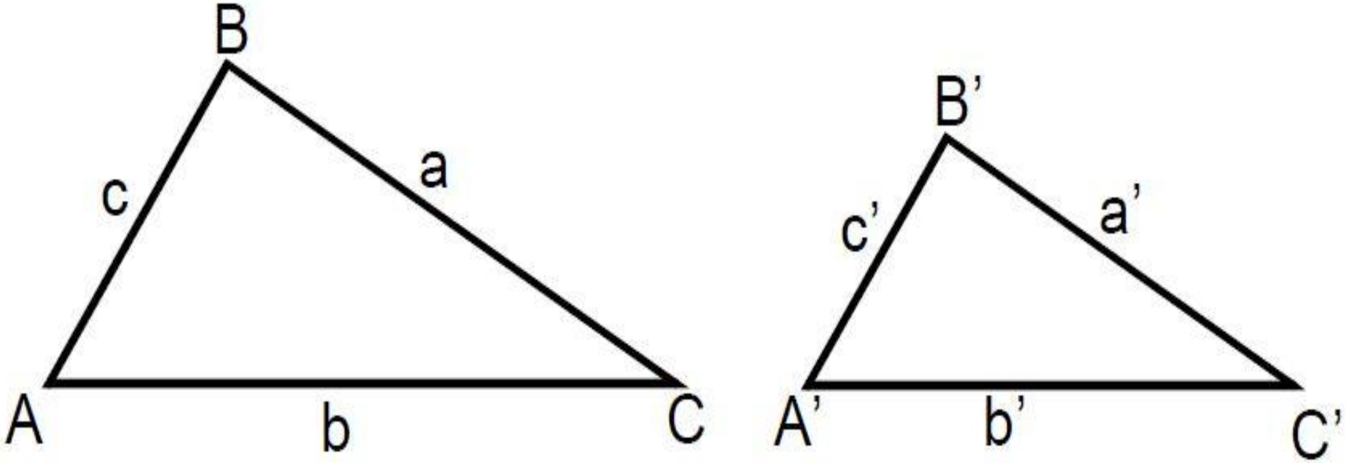
Geometric Model of Camera

Dr. Gerhard Roth

COMP 4102A
Winter 2014
Version 1

Similar Triangles



property (i): corresponding angles are equal

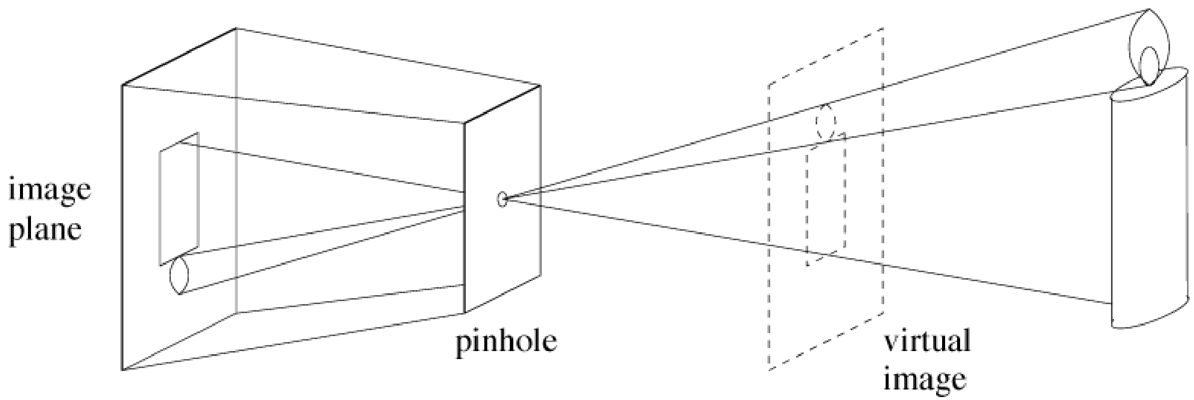
(A = A' and B = B' and C = C')

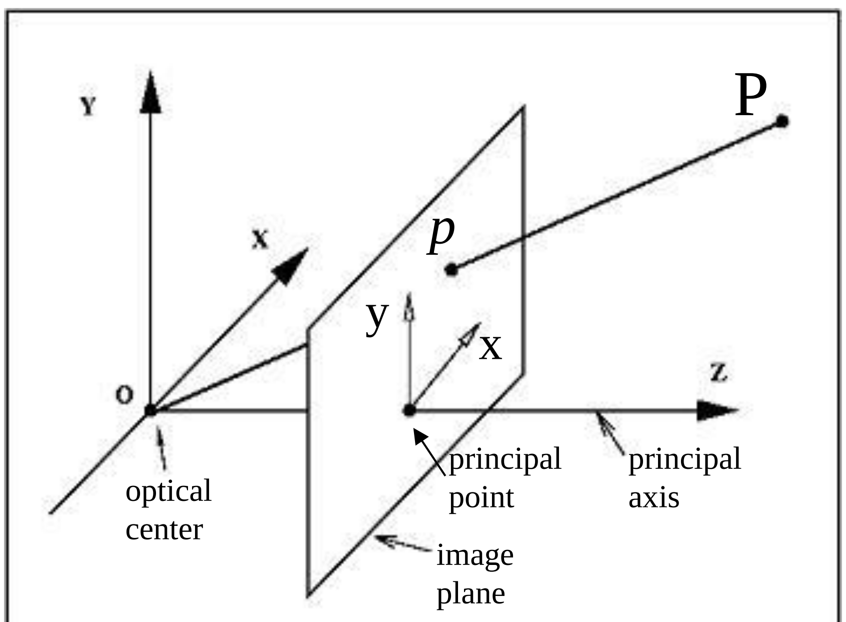
property (ii): corresponding sides have proportional lengths

$$\left(\frac{a}{a'} = \frac{b}{b'} = \frac{c}{c'}\right)$$

Geometric Model of Camera

Perspective projection





P(X,Y,Z) ⟶ *p*(x,y)

$$x = f\,\frac{X}{Z} \quad y = f\,\frac{Y}{Z}$$

Parallel lines aren't…

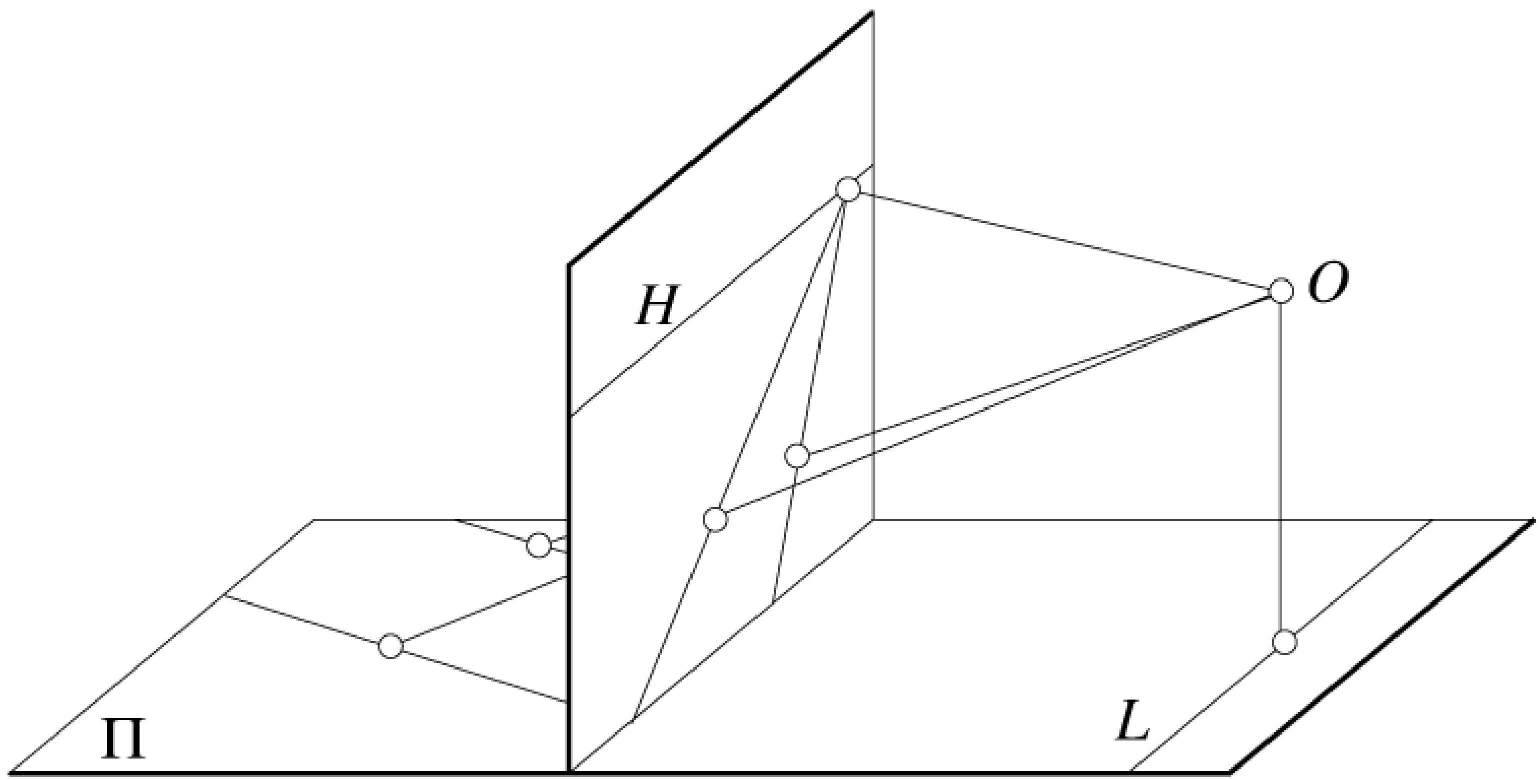
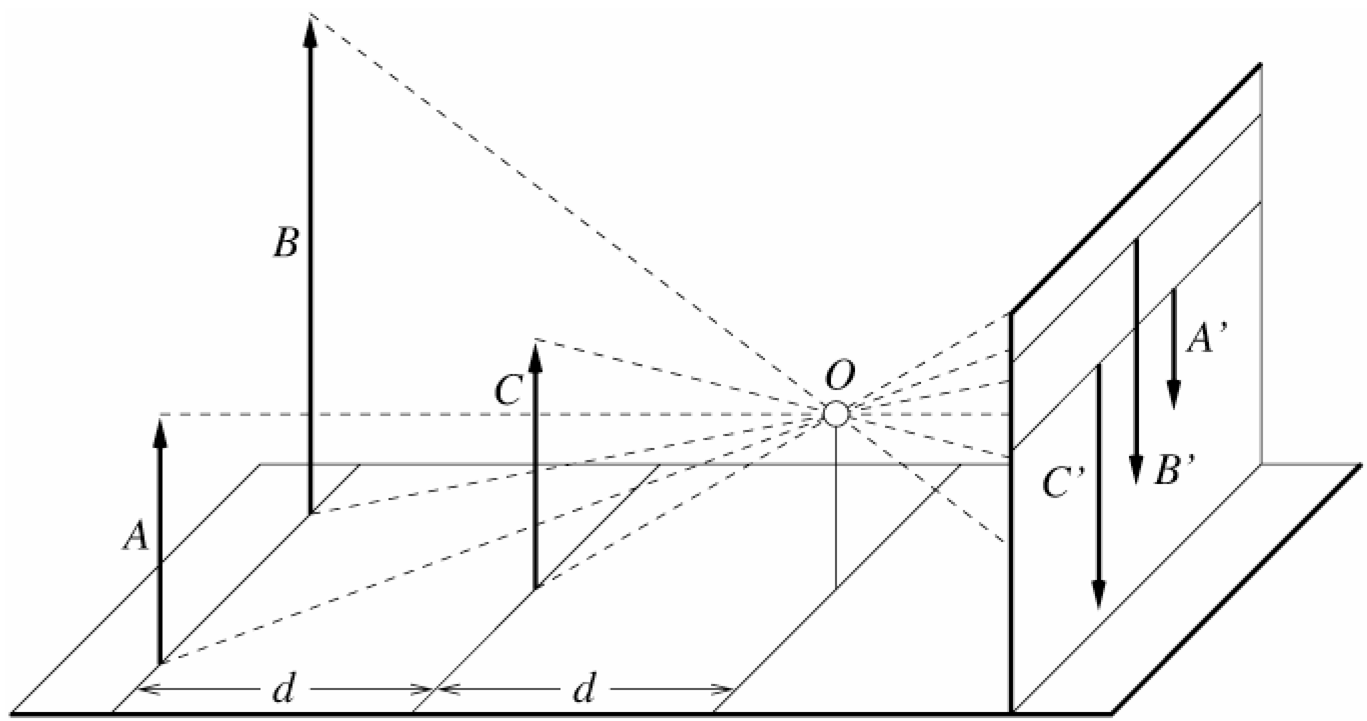
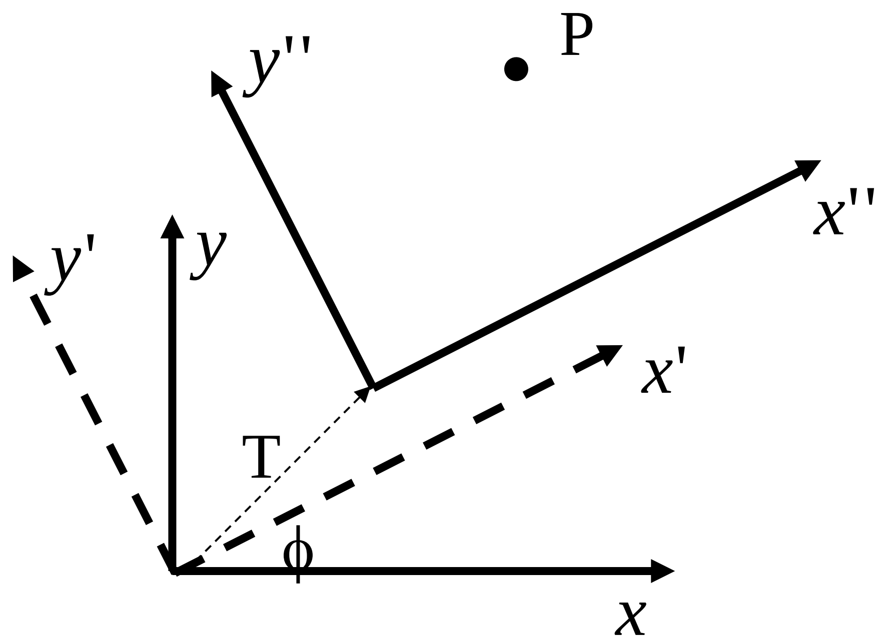


Figure by David Forsyth

Lengths can't be trusted...



Coordinate Transformation – 2D



Rotation and Translation

$$p' = \begin{bmatrix} p_x' \\ p_y' \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix}$$

$$p'' = \begin{bmatrix} p_x' \\ p_y' \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix}$$

Homogeneous Coordinates

Go one dimensional higher:

$$\begin{bmatrix} x \\ y \end{bmatrix} \rightarrow \begin{bmatrix} wx \\ wy \\ w \end{bmatrix} \qquad \begin{bmatrix} x \\ y \\ z \end{bmatrix} \rightarrow \begin{bmatrix} wx \\ wy \\ wz \\ w \end{bmatrix}$$

w is an arbitrary non-zero scalar, usually we choose 1.

From homogeneous coordinates to Cartesian coordinates:

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \rightarrow \begin{bmatrix} x_1 / x_3 \\ x_2 / x_3 \end{bmatrix} \qquad \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \rightarrow \begin{bmatrix} x_1 / x_4 \\ x_2 / x_4 \\ x_3 / x_4 \end{bmatrix}$$

2D Transformation with Homogeneous Coordinates

2D coordinate transformation:

$$p'' = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix} \begin{bmatrix} p_x \\ p_y \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \end{bmatrix}$$

2D coordinate transformation using homogeneous coordinates:

$$\begin{bmatrix} p_x'' \\ p_y'' \\ 1 \end{bmatrix} = \begin{bmatrix} \cos\phi & \sin\phi & T_x \\ -\sin\phi & \cos\phi & T_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ 1 \end{bmatrix}$$

Homogeneous coordinates (In 2d)

Two points are equal if and only if:

$x'/w' = x/w$ and $y'/w' = y/w$

$w=0$: points at infinity

- useful for projections and curve drawing

Homogenize = divide by w.

Homogenized points:

$$\begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translations with homogeneous

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = x + wt_x \\ y' = y + wt_y \\ w' = \quad w \end{cases}$$

$$\begin{cases} \dfrac{x'}{w'} = \dfrac{x}{w} + t_x \\ \dfrac{y'}{w'} = \dfrac{y}{w} + t_y \end{cases}$$

Scaling with homogeneous

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} s_x & 0 & 0 \\ 0 & s_y & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = s_x x \\ y' = s_y y \\ w' = w \end{cases}$$

$$\begin{cases} \dfrac{x'}{w'} = s_x \dfrac{x}{w} \\ \dfrac{y'}{w'} = s_y \dfrac{y}{w} \end{cases}$$

Rotation with homogeneous

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\begin{cases} x' = \cos\theta x - \sin\theta y \\ y' = \sin\theta x + \cos\theta y \\ w' = w \end{cases}$$

$$\begin{cases} \frac{x'}{w'} = \cos\theta \frac{x}{w} - \sin\theta \frac{y}{w} \\ \frac{y'}{w'} = \sin\theta \frac{x}{w} + \cos\theta \frac{y}{w} \end{cases}$$

3D Rotation Matrix

Rotate around each coordinate axis:

$$R_1(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\alpha & -\sin\alpha \\ 0 & \sin\alpha & \cos\alpha \end{bmatrix} \quad R_2(\beta) = \begin{bmatrix} \cos\beta & 0 & \sin\beta \\ 0 & 1 & 0 \\ -\sin\beta & 0 & \cos\beta \end{bmatrix} \quad R_3(\gamma) = \begin{bmatrix} \cos\gamma & -\sin\gamma & 0 \\ \sin\gamma & \cos\gamma & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Combine the three rotations:

$$R = R_1 R_2 R_3$$

3D rotation matrix has three parameters,
no matter how it is specified.

Rotation Matrices

- Both 2d and 3d rotation matrices have two characteristics
- They are orthogonal (also called orthonormal)

$$R^T R = I \qquad\qquad R^T = R^{-1}$$

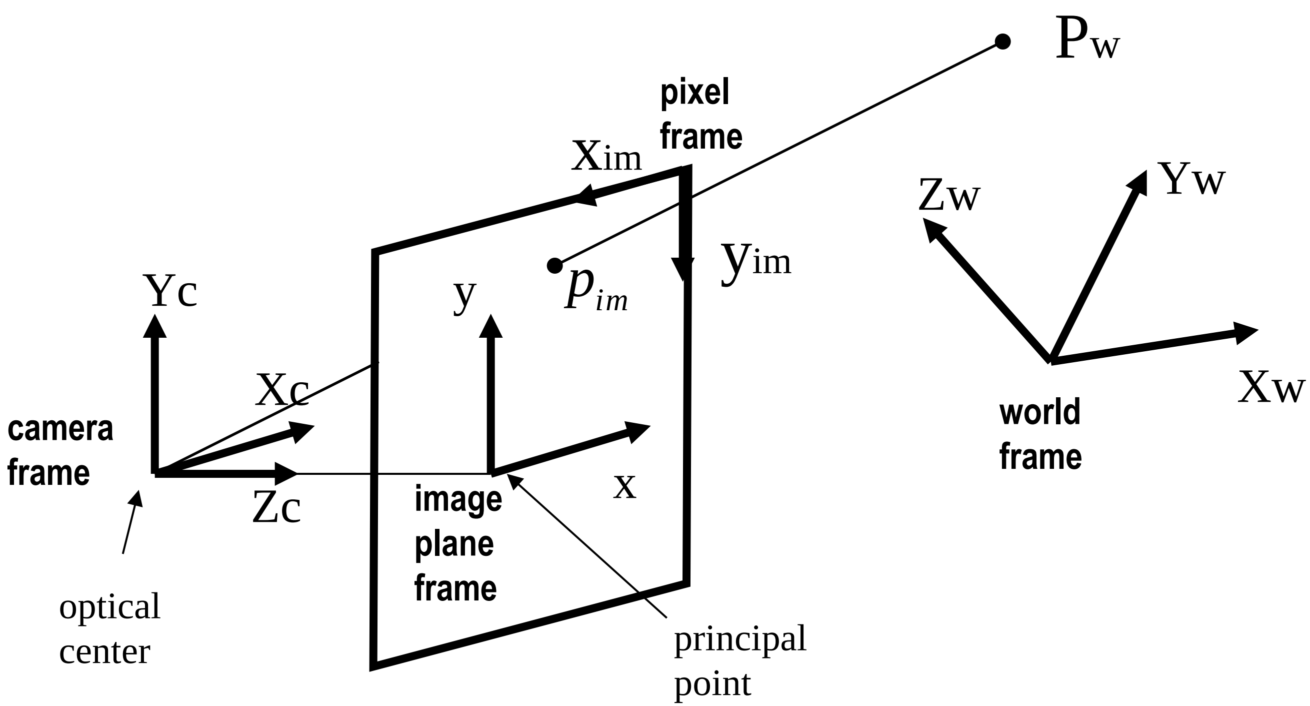
- Their determinant is 1
- Matrix below is orthogonal but not a rotation matrix because the determinate is not 1

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \longleftarrow \text{this is a reflection matrix}$$

Homogeneous co-ordinates

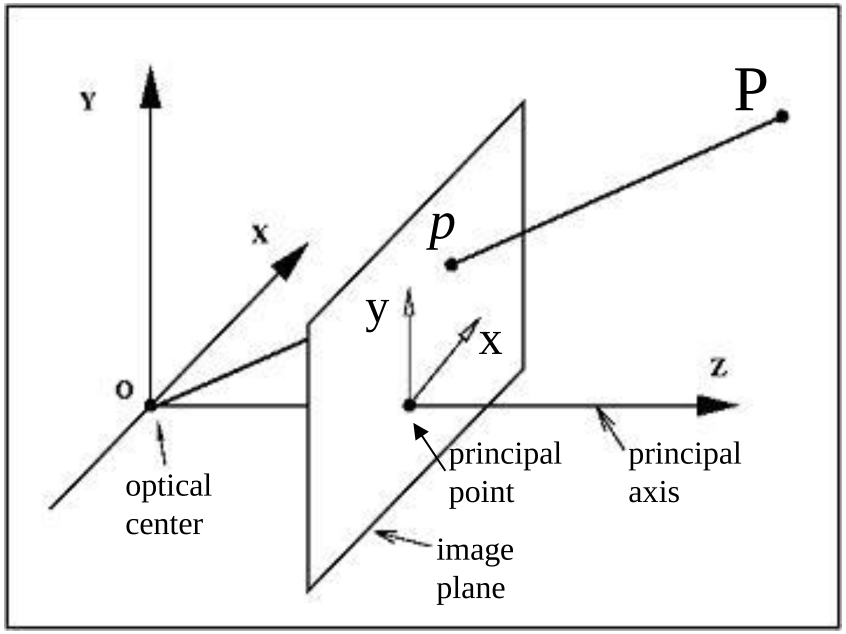
- Transformation – transform a point in an n dimensional space to another n dim point
  - Transformations are scale, rotations, translations, etc.
  - You can represent all these by multiplication by one appropriate matrix using homogeneous co-ordinates
- Projection – transform a point in an n dimensional space to an m dim point
  - For projection m is normally less than n
  - Perspective projection is a projection (3d to 2d)
  - From the 3d world to a 2d point in the image
  - You can also represent a projection as matrix multiplication with one appropriate matrix and homogeneouse co-ordinates

Four Coordinate Frames



Camera model: $$p_{im} = \begin{bmatrix} transformation \\ matrix \end{bmatrix} P_w$$

Perspective Projection



$$x = f\,\frac{X}{Z} \quad y = f\,\frac{Y}{Z}$$

These are *nonlinear*.

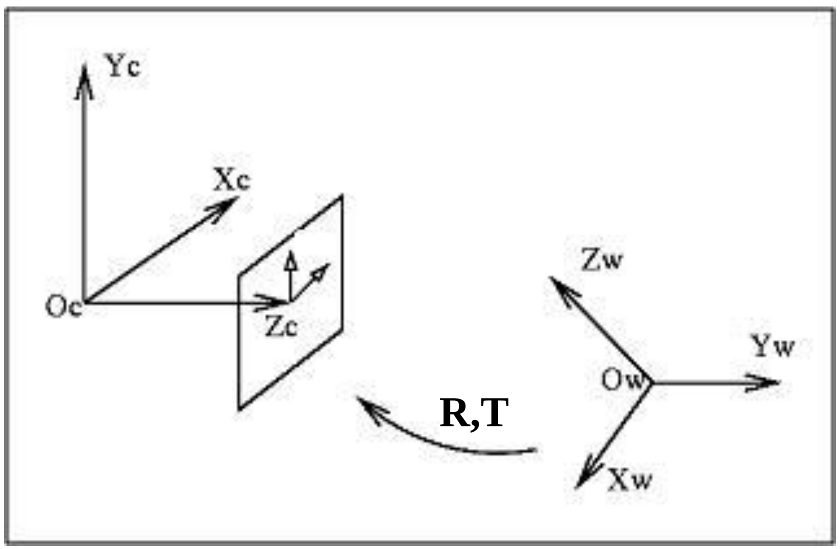
Using homogenous coordinate, we have a *linear* relation:

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

$$x = u / w \qquad y = v / w$$

World to Camera Coordinate

Transformation between the camera and world coordinates. Here we rotate, then translate, to go from world to camera co-ordinates which is opposite of book, but is simpler and is the way in which OpenCV routines do it:



$$\mathbf{X}_c = \mathbf{R}\mathbf{X}_w + \mathbf{T}$$

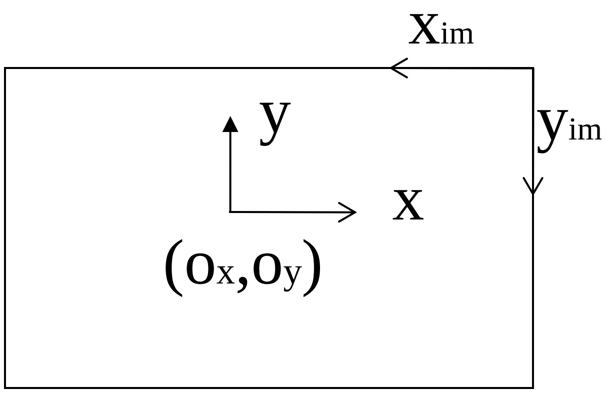
$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ \mathbf{0}^{\mathrm{T}} & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

After R, and T we have converted from world to camera frame. In the camera frame the z axis is along the optical center.

Camera Coordinates to Pixel Coordinates

$$x = (o_x - x_{im})s_x \quad y = (o_y - y_{im})s_y$$

s_x, s_y : pixel sizes in millimeters per pixel

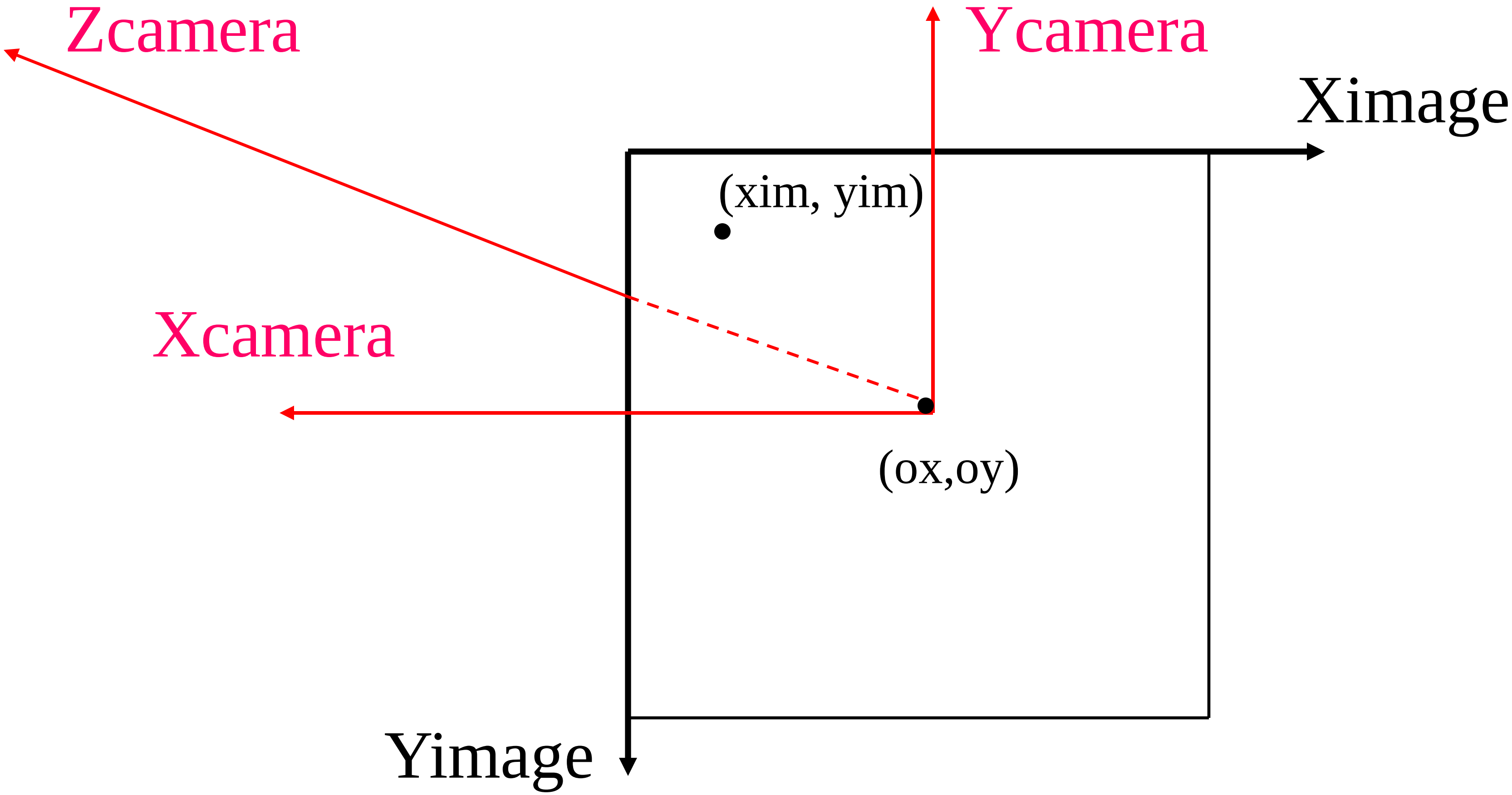


$$\begin{bmatrix} x_{im} \\ y_{im} \\ 1 \end{bmatrix} = \begin{bmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Camera co-ordinates *x*, and *y* are in millimetres
Image co-ordinates x_{im}, y_{im}, are in pixels
Center of projection o_x, o_y is in pixels
Sign change because horizontal and vertical axis of the image and camera frame have opposite directions.

Image and Camera frames

Now we look from the camera outward and image origin is the top left pixel (0,0)



Put All Together – World to Pixel

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ w \end{bmatrix}$$

From camera to pixel

$$= \begin{bmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

Add projection

$$= \begin{bmatrix} -1/s_x & 0 & o_x \\ 0 & -1/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

Add Rotation
And Translation

$$= \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = K \begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

$$x_{im} = x_1 / x_3 \quad y_{im} = x_2 / x_3$$

Camera Parameters

- Extrinsic parameters define the location and orientation of the camera reference frame with respect to a world reference frame
  - Depend on the external world, so they are extrinsic
- Intrinsic parameters link pixel co-ordinates in the image with the corresponding co-ordinates in the camera reference frame
  - An intrinsic characteristic of the camera
- Image co-ordinates are in pixels
- Camera co-ordinates are in millimetres
  - In formulas that do conversions the units must match!

Intrinsic Camera Parameters

$$K = \begin{bmatrix} -f/s_x & 0 & o_x \\ 0 & -f/s_y & o_y \\ 0 & 0 & 1 \end{bmatrix}$$

K is a 3x3 upper triangular matrix, called the **Camera Calibration Matrix**.

There are five intrinsic parameters:

(a) The pixel sizes in x and y directions s_x, s_y in millimeters/pixel

(b) The focal length f in millimeters

(c) The principal point (o_x,o_y) in pixels, which is the point where the optic axis intersects the image plane.

(d) The units of f/Sx and f/Sy are in pixels, why is this so?

Camera intrinsic parameters

- Can write three of these parameters differently by letting f/sx = fx and f/sy = fy
  - Then intrinsic parameters are ox,oy,fx,fy
  - The units of these parameters are pixels!
- In practice pixels are square (sx = sy) so that means fx should equal fy for most cameras
  - However, every explicit camera calibration process (using calibration objects) introduces some small errors
  - These calibration errors make fx not exactly equal to fy
- So in OpenCV the intrinsic camera parameters are the four following ox,oy,fx,fy
  - However fx is usually very close to fy and if this is not the case then there is a problem

Extrinsic Parameters and Proj. Matrix

$$p_{im} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = K\begin{bmatrix} R & T \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = M \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}$$

[R|T] defines the **extrinsic parameters**.
The 3x4 matrix M = K[R|T] is called the **projection matrix**.
It takes 3d points in the world co-ordinate system and maps them to the appropriate image co-ordinates in pixels

Using the projection matrix - example

$$\begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} = \begin{bmatrix} 1000 & 0 & 320 & -19600 \\ 0 & 1000 & 240 & -27200 \\ 0 & 0 & 1 & -30 \end{bmatrix}$$

$$u = \frac{u'}{w'} \quad v = \frac{v'}{w'}$$

- Where would the point (20,50,200) project to in the image?

Projection matrix

$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} P_{11} & P_{12} & P_{13} & P_{14} \\ P_{21} & P_{22} & P_{23} & P_{24} \\ P_{31} & P_{32} & P_{33} & P_{34} \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} 1000 & 0 & 320 & -19600 \\ 0 & 1000 & 240 & -27200 \\ 0 & 0 & 1 & -30 \end{bmatrix} \begin{bmatrix} 20 \\ 50 \\ 200 \\ 1 \end{bmatrix}$$

$$u = \frac{u'}{w'} \quad v = \frac{v'}{w'}$$

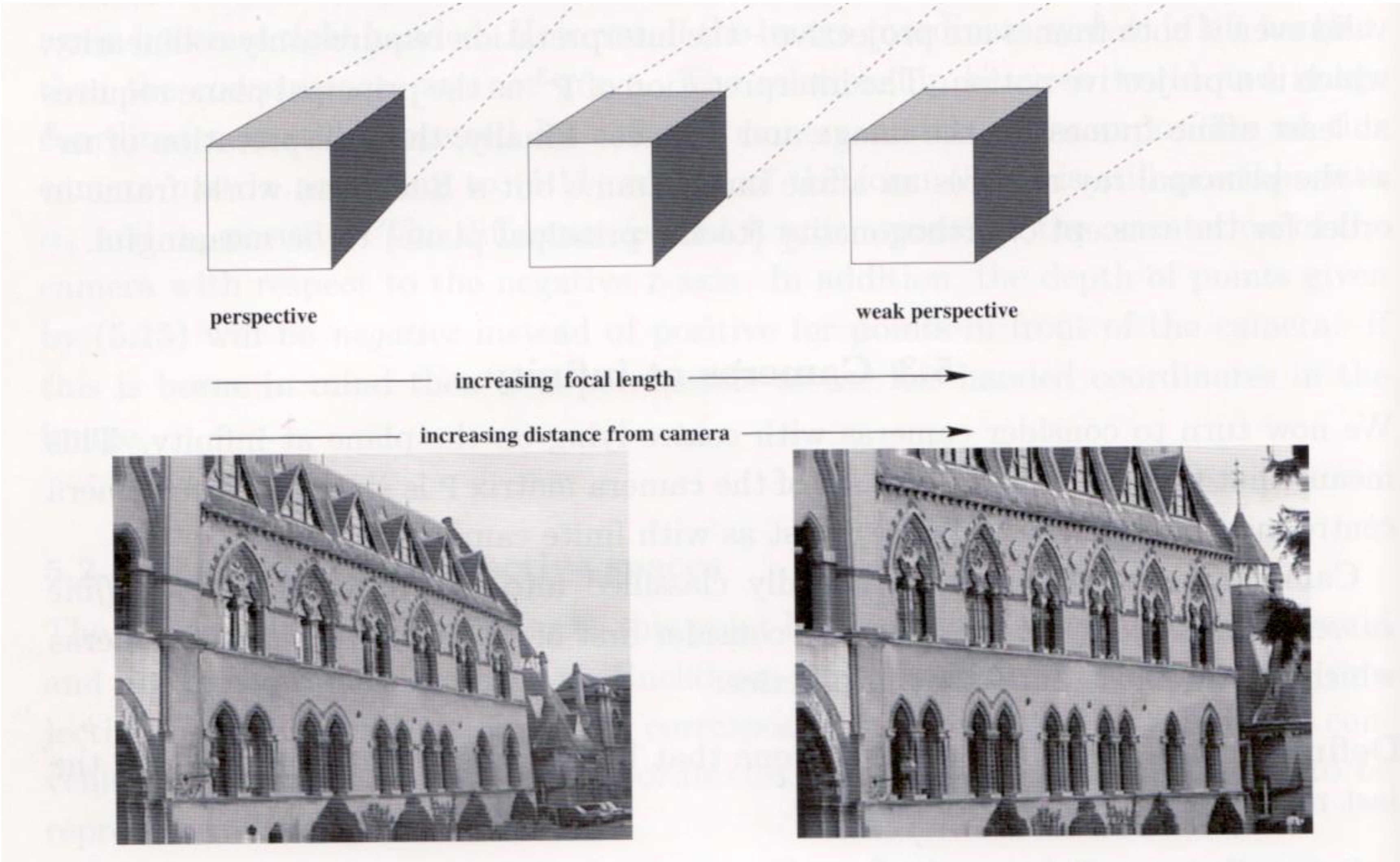
$$\begin{bmatrix} u' \\ v' \\ w' \end{bmatrix} = \begin{bmatrix} 64400 \\ 70800 \\ 170 \end{bmatrix}$$

u=u'/w' = 64400/170 = 378.8
v=v'/w' = 70800/170 = 416.5

- World point (20,50,200) project to pixel With co-ordinates of (379,414)

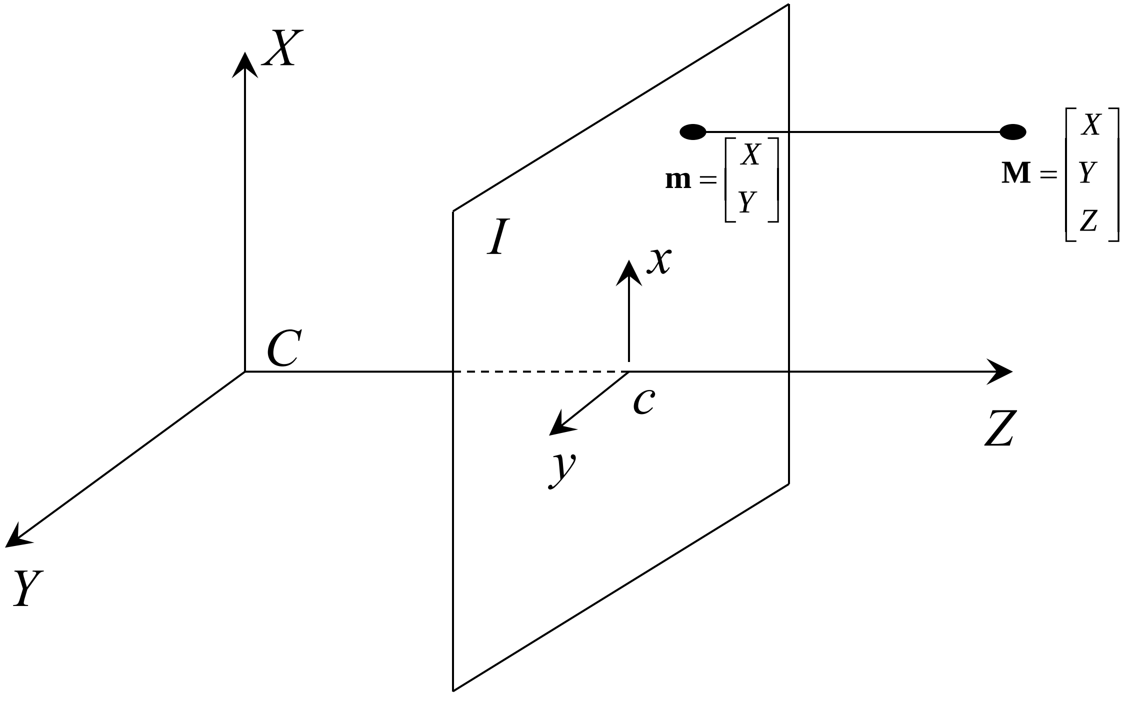
Effect of change in focal length

Small f is wide angle, large f is telescopic



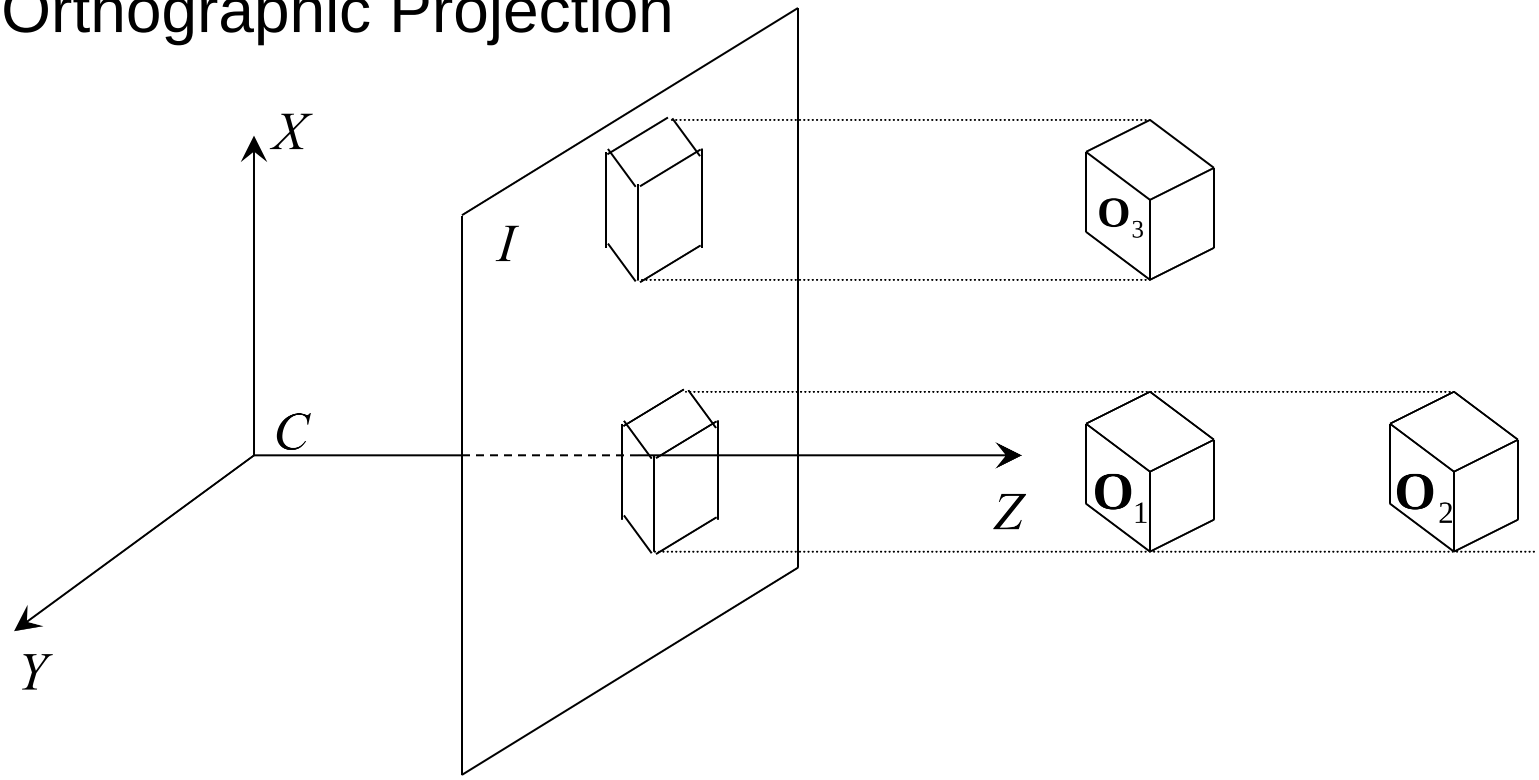
Orthographic Projection

Orthographic Projection



Orthographic Projection

Orthographic Projection



Weak Perspective Model

Assume the relative distance between any two points in an object along the principal axis is much smaller ($1/20^{th}$ at most) than the $\bar{Z}$ average distance of the object. Then the camera projection can be approximated as:

$$x = f\,\frac{X}{Z} \approx \frac{f}{\bar{Z}}\,X$$

$$y = f\,\frac{Y}{Z} \approx \frac{f}{\bar{Z}}\,Y$$

This is the **weak-perspective** camera model.
Sometimes called scaled orthography.

Weak Perspective Projection

Two step projection:

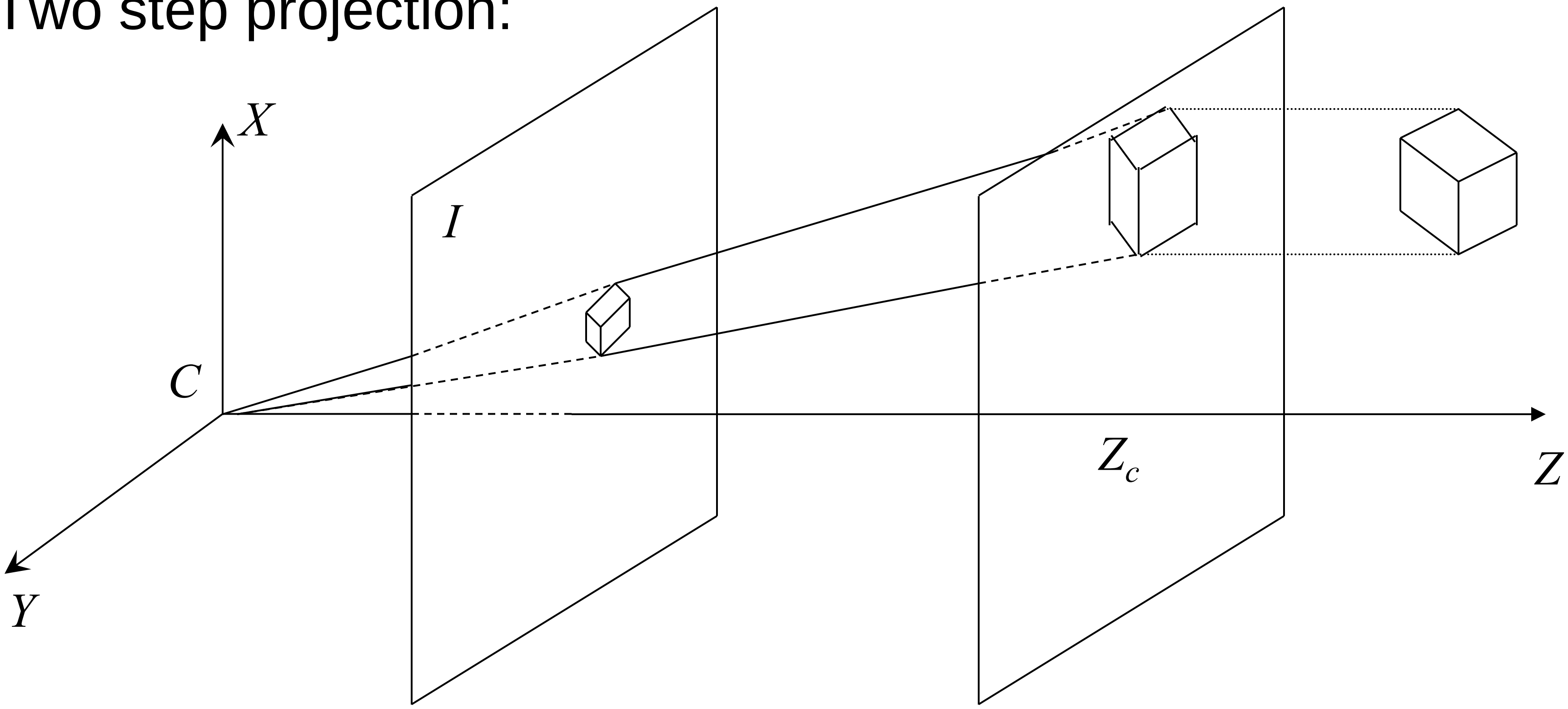
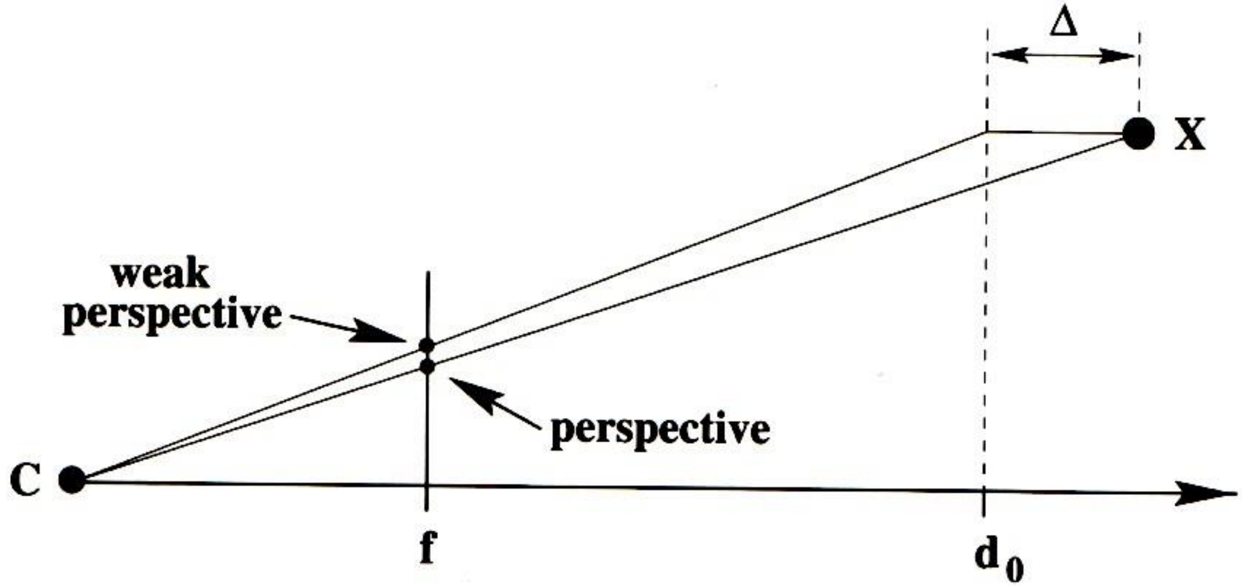


image plane average depth plane

Weak Perspective

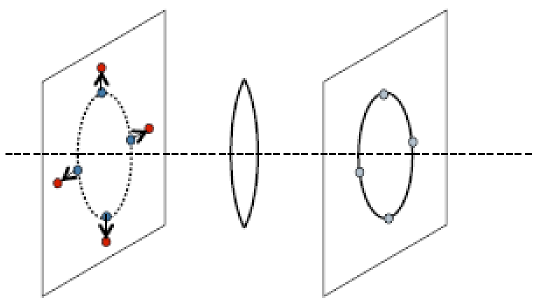


Impact of different projections

- Perspective projections
  - Parallel lines in world are not parallel in the image
  - Object projection gets smaller with distance from camera
- Weak perspective projection
  - Parallel lines in the world are parallel in the image
  - Object projection gets smaller with distance from camera
- Orthographic projection
  - Parallel lines in the world are parallel in the image
  - Object projection is unchanged with distance from camera

Image distortion due to optics

- Radial distortion which depends on radius r, distance of each point from center of image
- $r^2 = (x - o_x)^2 + (y - o_y)^2$



Radial distortion

$$x_{corrected} = x(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$
$$y_{corrected} = y(1 + k_1 r^2 + k_2 r^4 + k_3 r^6)$$

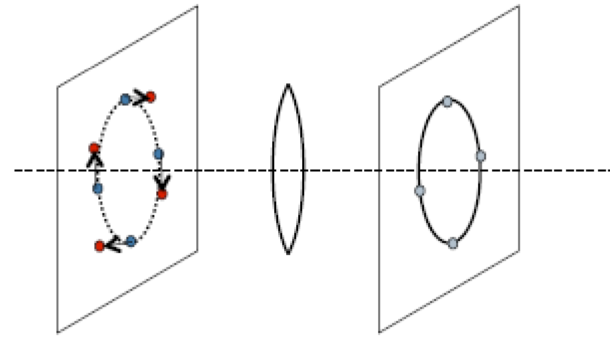
Correction uses three parameters, k_1, k_2, k_3

Correcting Radial Distortions

Tangential Distortion

- Lens not exactly parallel to the image plane

$$x_{\text{corrected}} = x + [2p_1 y + p_2(r^2 + 2x^2)]$$
$$y_{\text{corrected}} = y + [p_1(r^2 + 2y^2) + 2p_2 x]$$



Tangential distortion

- Correction uses two parameter p_1, p_2
- Both types of distortion are removed (image is un-distorted) and only then does standard calibration matrix K apply to the image
- Camera calibration computes both K and these five distortion parameters

How to find the camera parameters

- Can use the EXIF tag for any digital image
  - Has focal length f in millimeters but not the pixel size
  - But you can get the pixel size from the camera manual
  - There are only a finite number of different pixels sizes because the number of sensing element sizes is limited
  - If there is not a lot of image distortion due to optics then this approach is sufficient (this is only a linear calibration)
- Can perform explicit camera calibration
  - Put a calibration pattern in front of the camera
  - Take a number of different pictures of this pattern
  - Now run the calibration algorithm (different types)
  - Result is intrinsic camera parameters (linear and non-linear) and the extrinsic camera parameters of all the images