MAN-522: COMPUTER VISION
SET-2
Projections and Camera Calibration

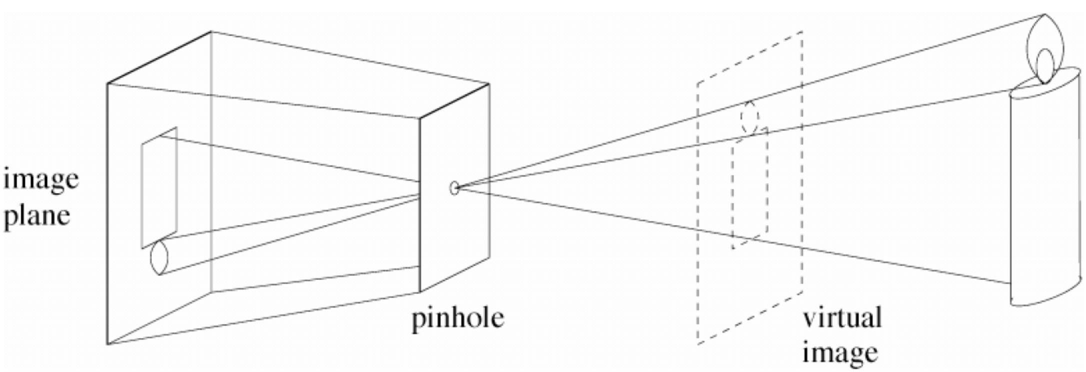


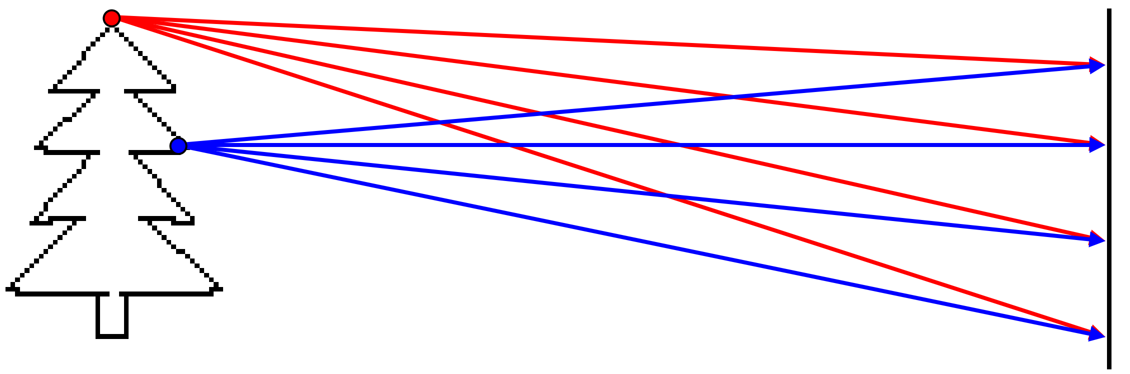
Image formation

- How are objects in the world captured in an image?

Physical parameters of image formation

- Geometric
  - Type of projection
  - Camera pose
- Optical
  - Sensor's lens type
  - focal length, field of view, aperture
- Photometric
  - Type, direction, intensity of light reaching sensor
  - Surfaces' reflectance properties

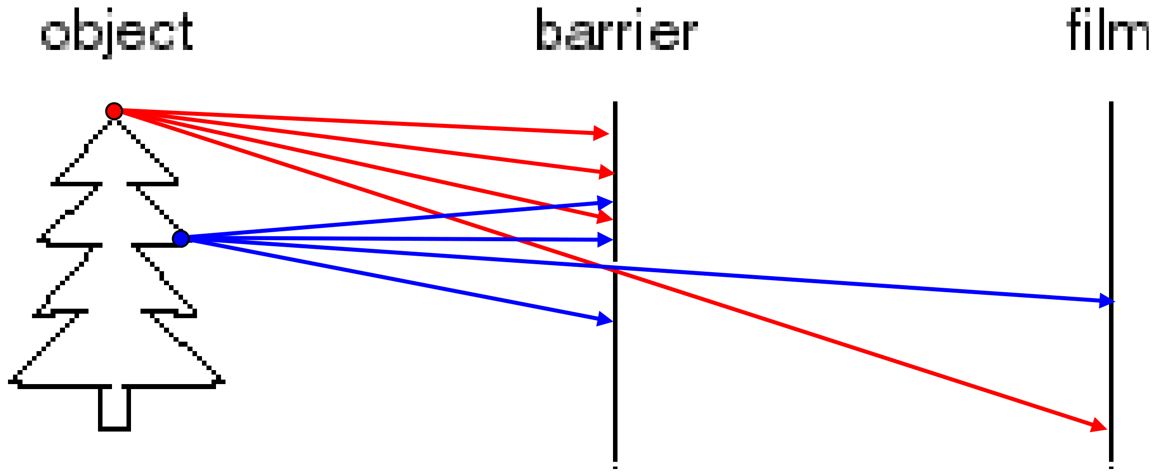
Image formation



- Let’s design a camera
  - Idea 1: put a piece of film in front of an object
  - Do we get a reasonable image?

Slide by Steve Seitz

Pinhole camera

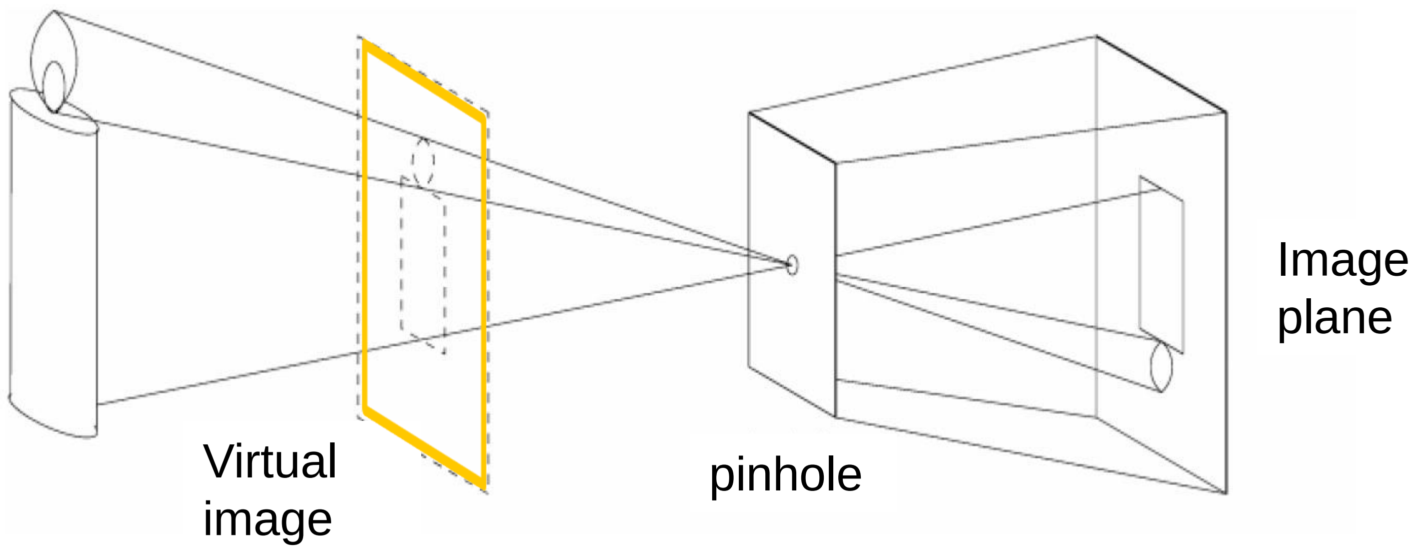


- Add a barrier to block off most of the rays
  - This reduces blurring
  - The opening is known as the **aperture**
  - How does this transform the image?

Slide by Steve Seitz

Pinhole camera

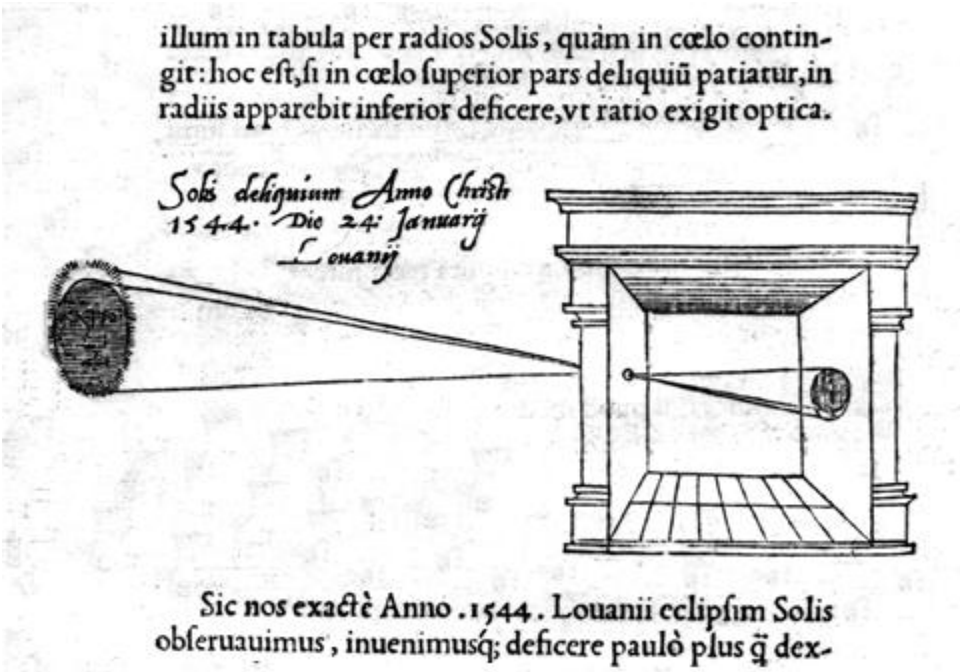
- Pinhole camera is a simple model to approximate imaging process, perspective **projection**.



If we treat pinhole as a point, only one ray from any given point can enter the camera.

Fig from Forsyth and Ponce

Camera obscura

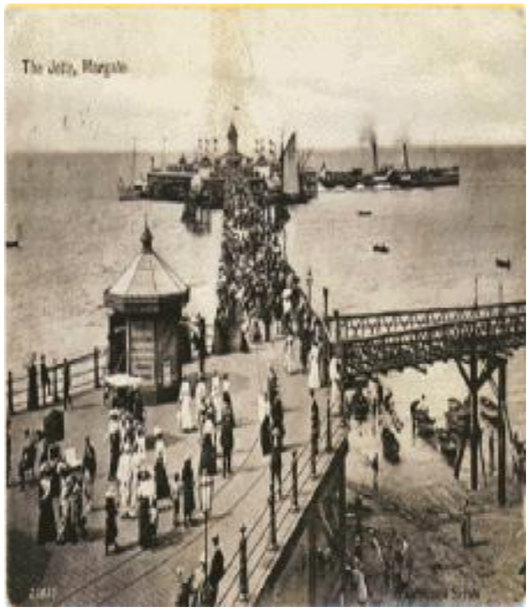


In Latin, means
‘dark room’

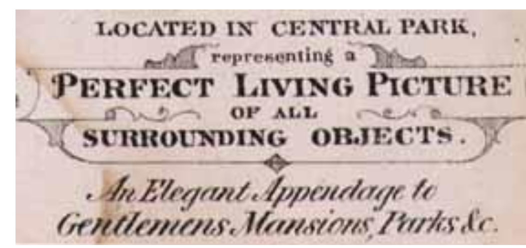
"**Reinerus Gemma-Frisius**, observed an eclipse of the sun at Louvain on January 24, 1544, and later he used this illustration of the event in his book De Radio Astronomica et Geometrica, 1545. It is thought to be the first published illustration of a camera obscura..."
Hammond, John H., The Camera Obscura, A Chronicle

http://www.acmi.net.au/AIC/CAMERA_OBSCURA.html

Camera obscura



Jetty at Margate England, 1898.



Around 1870s

An attraction in the late 19th century

http://brightbytes.com/cosite/collection2.html

Adapted from R. Duraiswami

Camera obscura at home

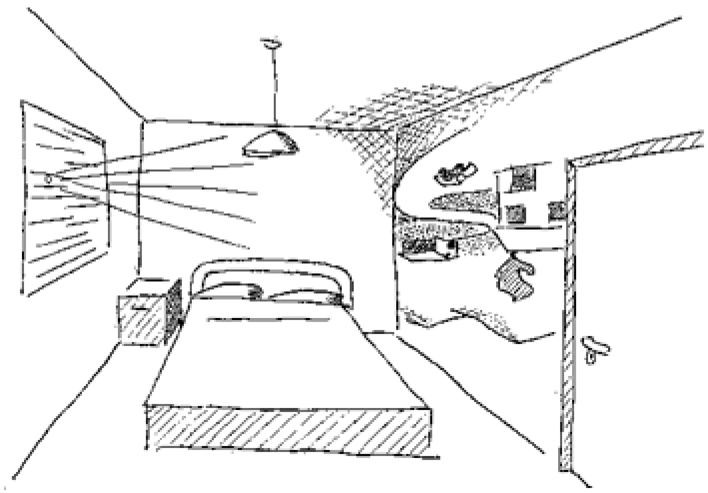


Figure 1 - A lens on the window creates the image of the external world on the opposite wall and you can see it every morning, when you wake up.

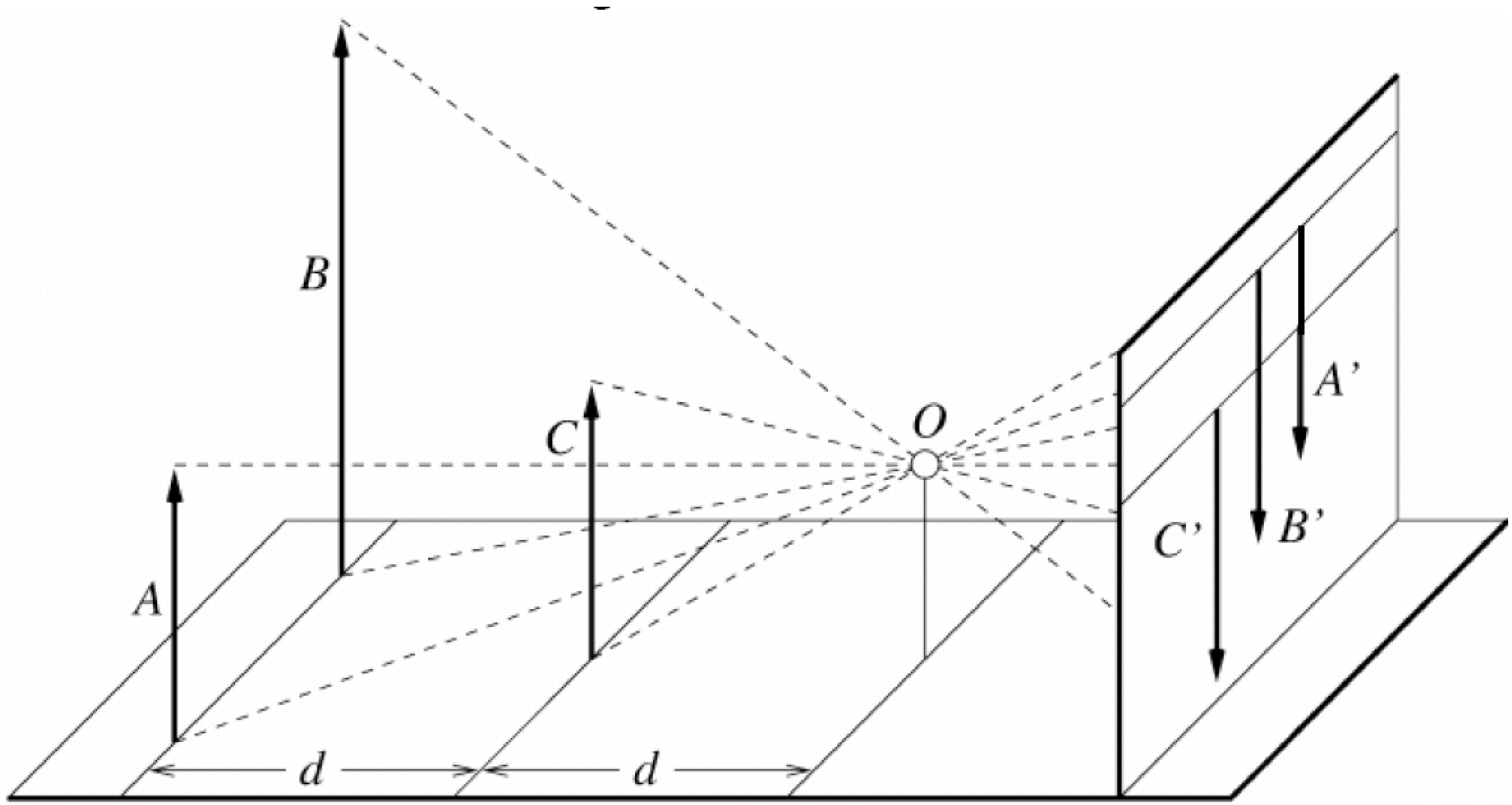
Sketch from http://www.funsci.com/fun3_en/sky/sky.htm

http://blog.makezine.com/archive/2006/02/how_to_room_sized_camera_obscu.html

Perspective effects

Perspective effects

- Far away objects appear smaller

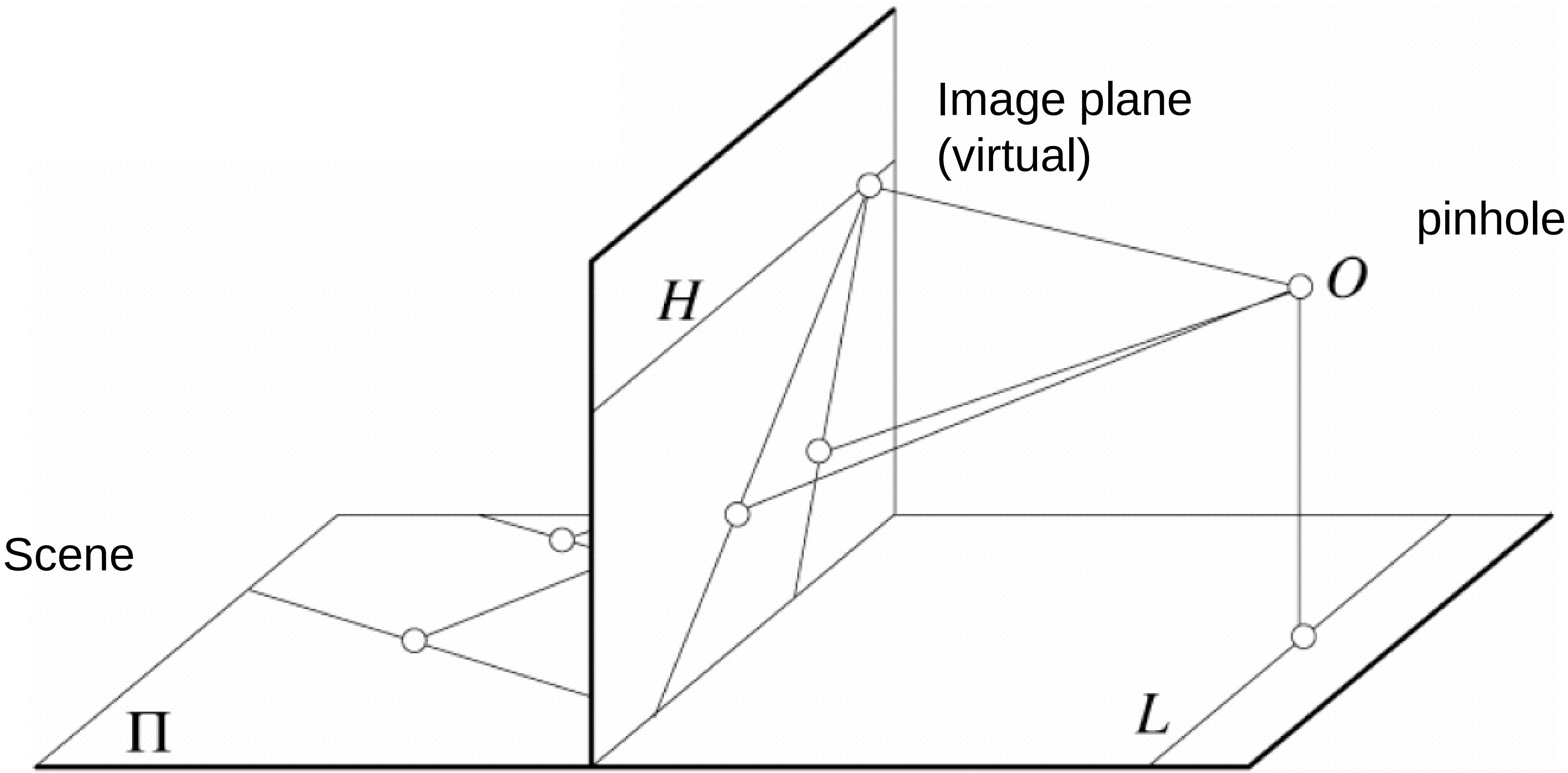


Forsyth and Ponce

Perspective effects

Perspective effects

- Parallel lines in the scene intersect in the image
- Converge in image on horizon line



Projection properties

- Many-to-one: any points along same ray map to same point in image
- Points → points
- Lines → lines (collinearity preserved)
- Distances and angles are **not** preserved
- Degenerate cases:
  - Line through focal point projects to a point.
  - Plane through focal point projects to line
  - Plane perpendicular to image plane projects to part of the image.

Perspective and art

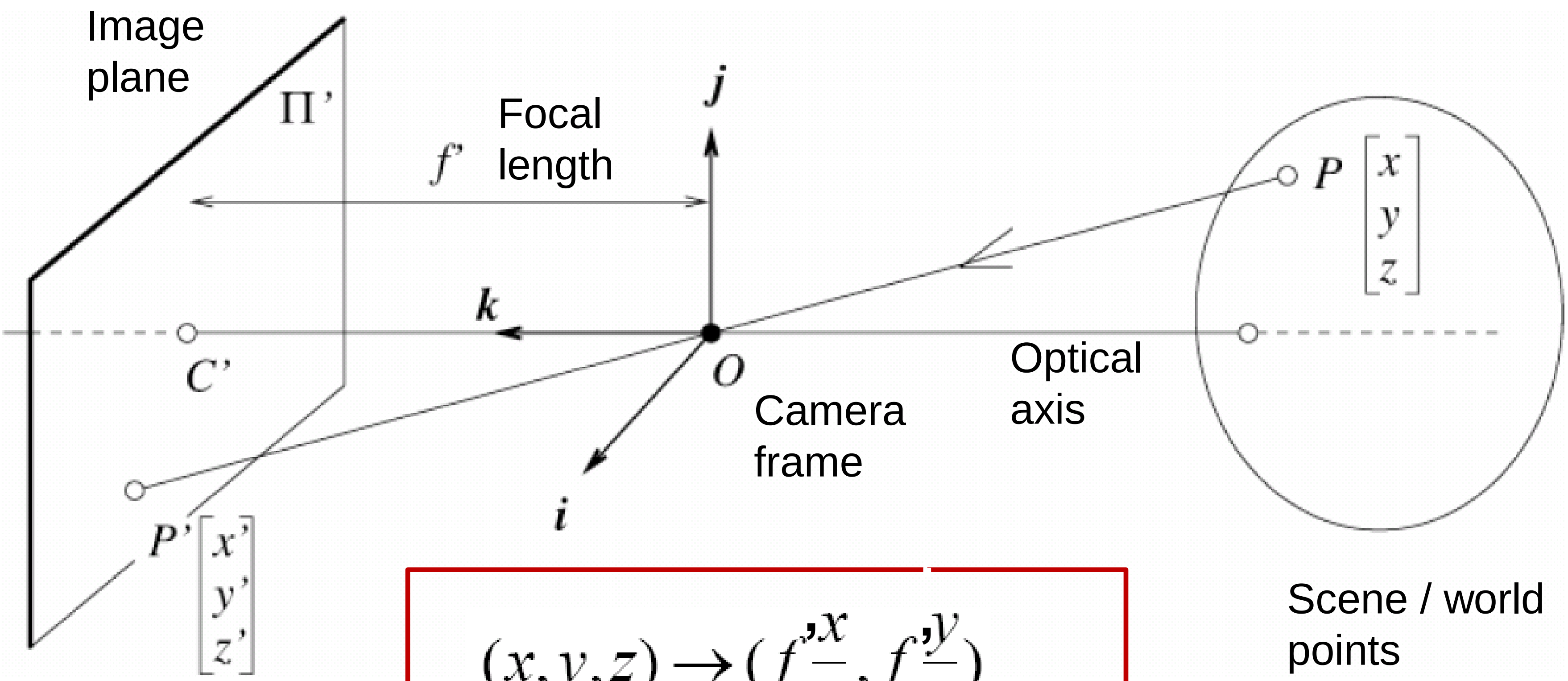
- Use of correct perspective projection indicated in 1st century B.C. frescoes
- Skill resurfaces in Renaissance: artists develop systematic methods to determine perspective projection (around 1480-1515)

Raphael

Durer, 1525

Perspective projection equations

- 3d world mapped to 2d projection in image plane



$$(x, y, z) \rightarrow (f' \frac{x}{z}, f' \frac{y}{z})$$

Scene point $\longrightarrow$ Image coordinates

Forsyth and Ponce

Homogeneous coordinates

Is this a linear transformation?

- no—division by z is nonlinear

Trick: add one more coordinate:

$$(x, y) \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

homogeneous image coordinates

$$(x, y, z) \Rightarrow \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix}$$

homogeneous scene coordinates

Converting *from* homogeneous coordinates

$$\begin{bmatrix} x \\ y \\ w \end{bmatrix} \Rightarrow (x/w, y/w)$$

$$\begin{bmatrix} x \\ y \\ z \\ w \end{bmatrix} \Rightarrow (x/w, y/w, z/w)$$

Slide by Steve Seitz

Perspective Projection Matrix

- Projection is a matrix multiplication using homogeneous coordinates:

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/f' & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ z/f' \end{bmatrix} \Rightarrow (f'\frac{x}{z}, f'\frac{y}{z})$$

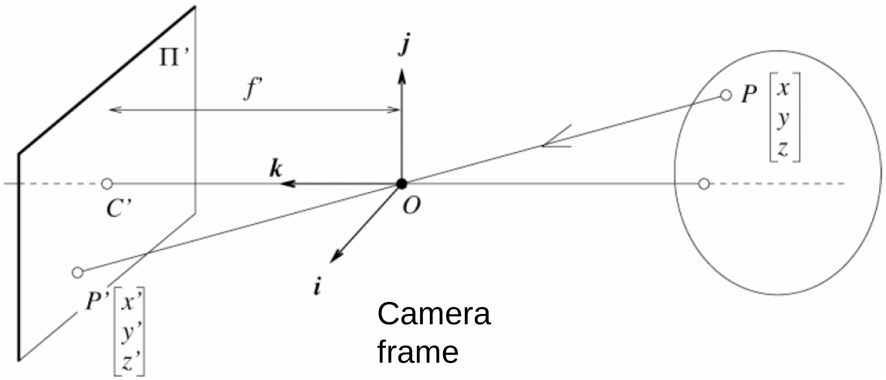
divide by the third coordinate to convert back to non-homogeneous coordinates

Complete mapping from world points to image pixel positions?

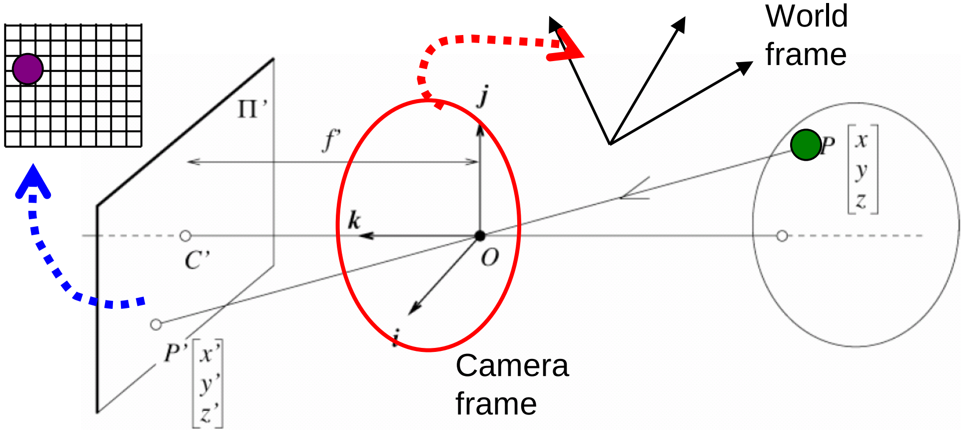
Slide by Steve Seitz

Perspective projection & calibration

- Perspective equations so far in terms of *camera's* reference frame….
- Camera's *intrinsic* and *extrinsic* parameters needed to calibrate geometry.



Perspective projection & calibration



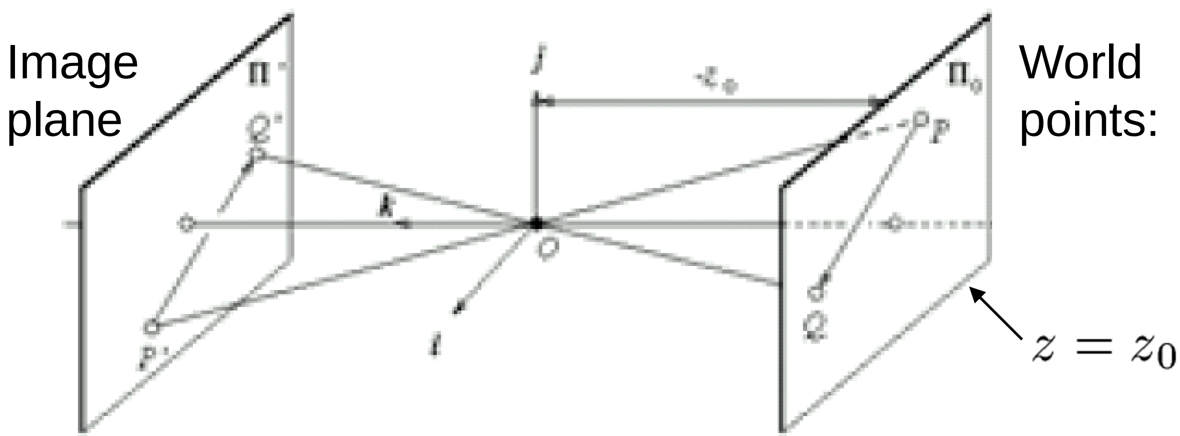
Extrinsic:
Camera frame ←→World frame

Intrinsic:
Image coordinates relative to
camera ←→ Pixel coordinates

3D point (4x1)

Weak perspective

- Approximation: treat magnification as constant
- Assumes scene depth << average distance to camera

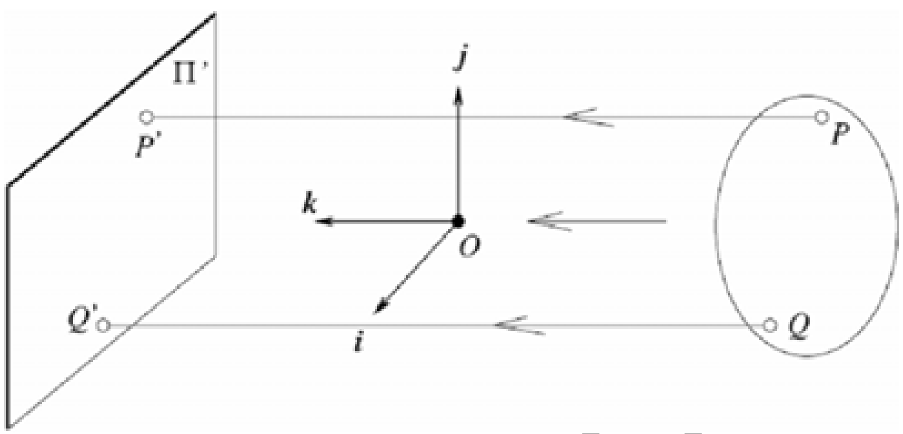


$$x' = f\frac{x}{z} \approx \frac{f}{z_0}x$$

$$y' = f\frac{y}{z} \approx \frac{f}{z_0}y$$

Orthographic projection

- Given camera at **constant** distance from scene
- World points projected along rays parallel to optical access



$$x' = x$$
$$y' = y$$

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \\ 1 \end{bmatrix} = \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \Rightarrow (x, y)$$

Pinhole size / aperture

How does the size of the aperture affect the image we'd get?

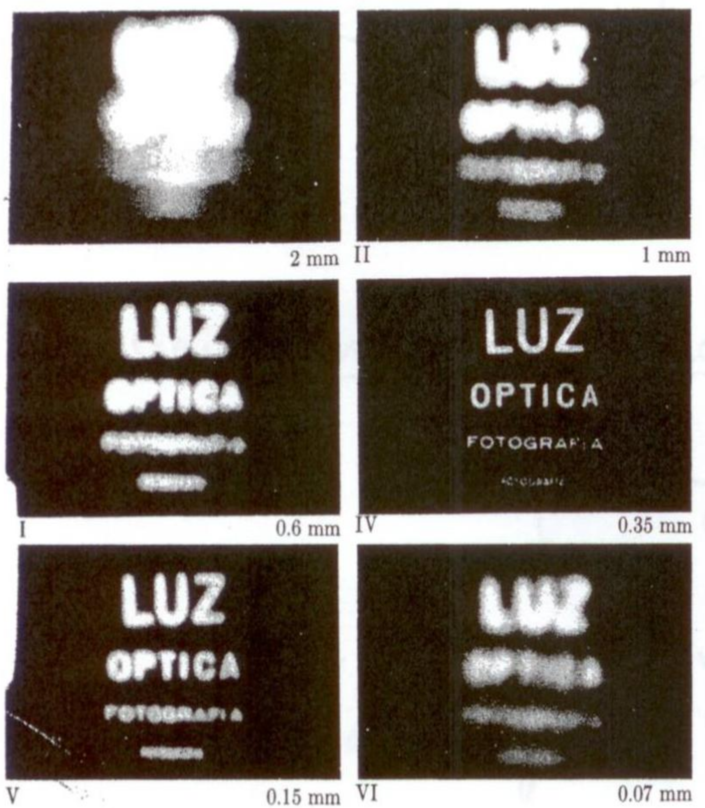
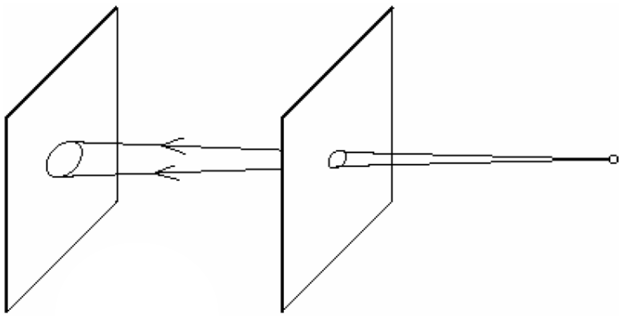


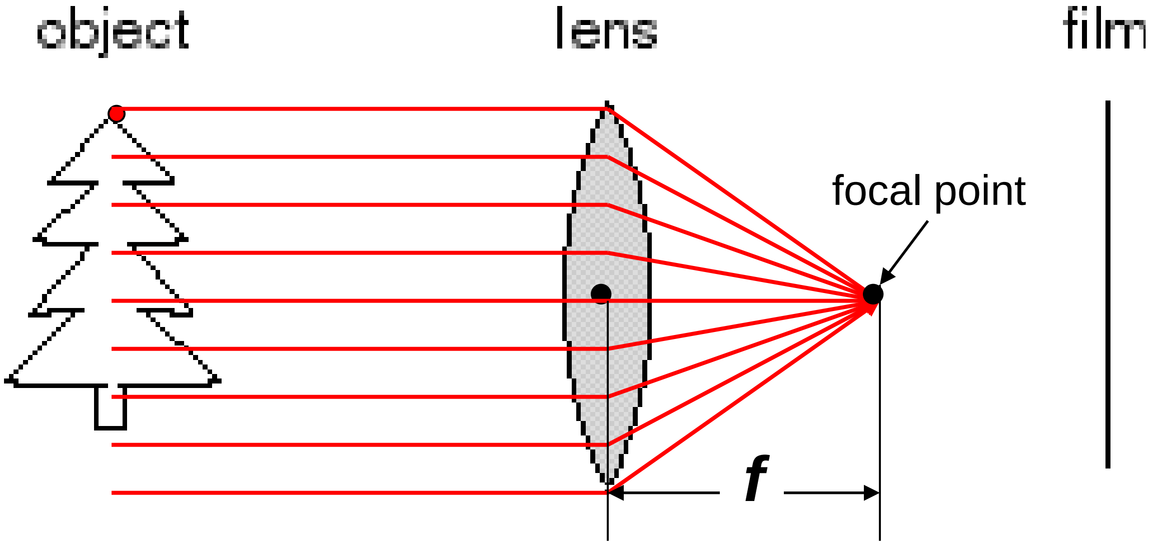
Fig. 5.96 The pinhole camera. Note the variation in image clarity as the hole diameter decreases. [Photos courtesy Dr. N. Joel, UNESCO.]

Larger

Smaller



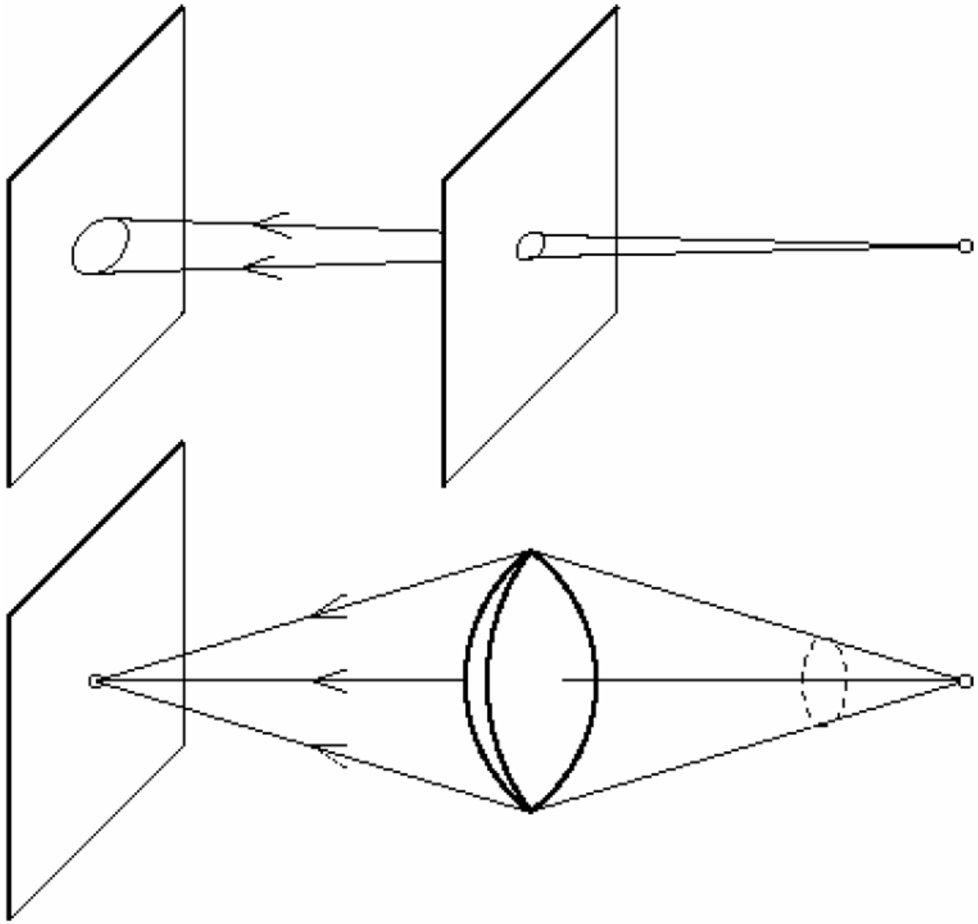
Adding a lens



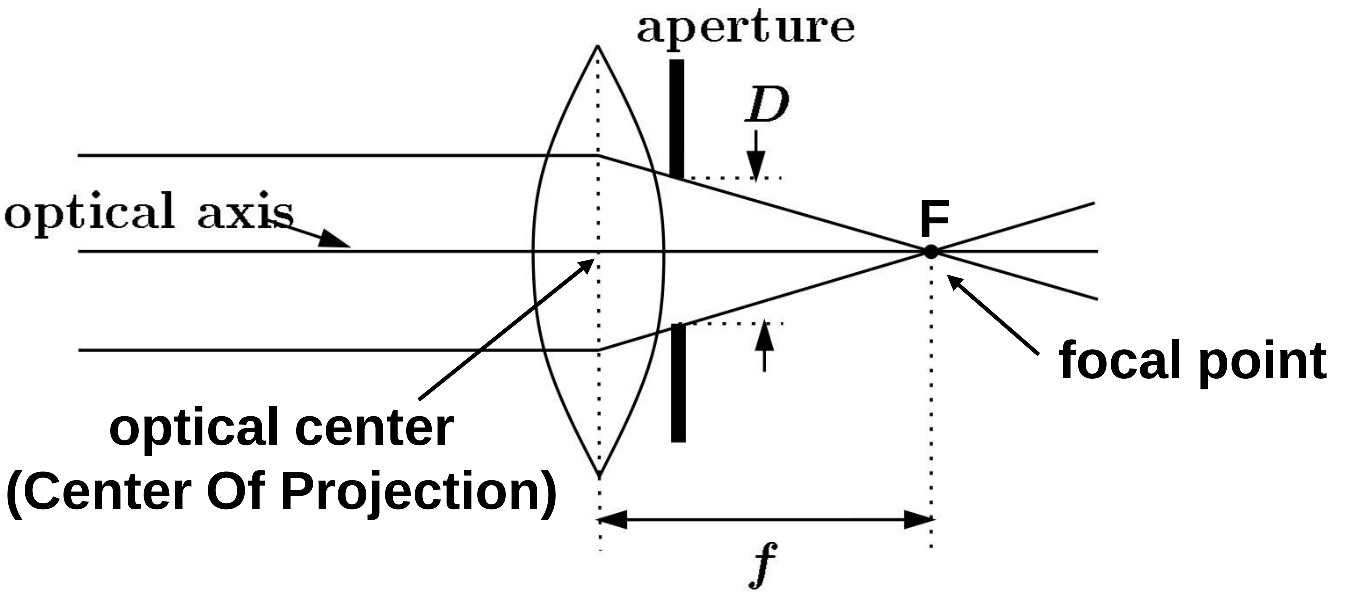
- A lens focuses light onto the film
  - Rays passing through the center are not deviated
  - All parallel rays converge to one point on a plane located at the *focal length f*

Slide by Steve Seitz

Pinhole vs. lens



Cameras with lenses

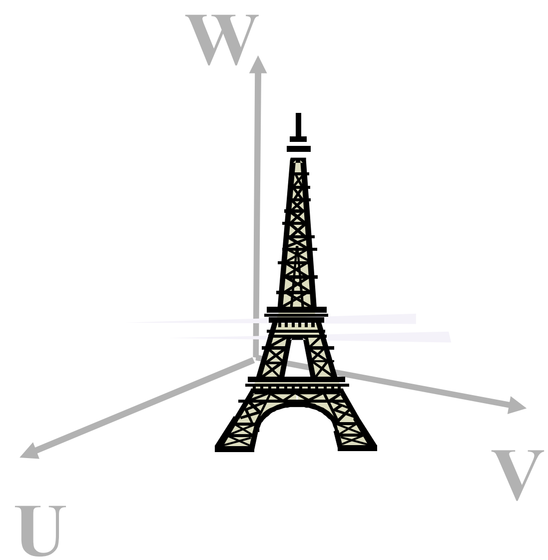


- A lens focuses parallel rays onto a single focal point
- Gather more light, while keeping focus; make pinhole perspective projection practical

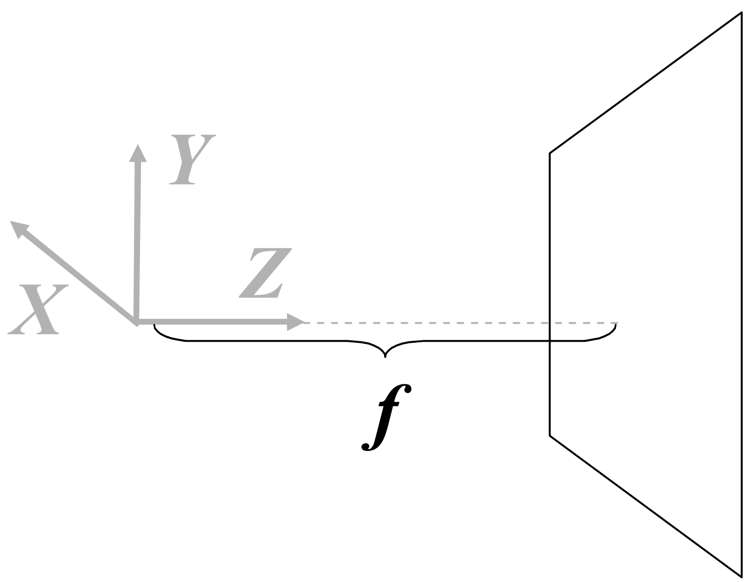
Camera Parameters

Imaging Geometry

Object of Interest in World Coordinate System (U,V,W)



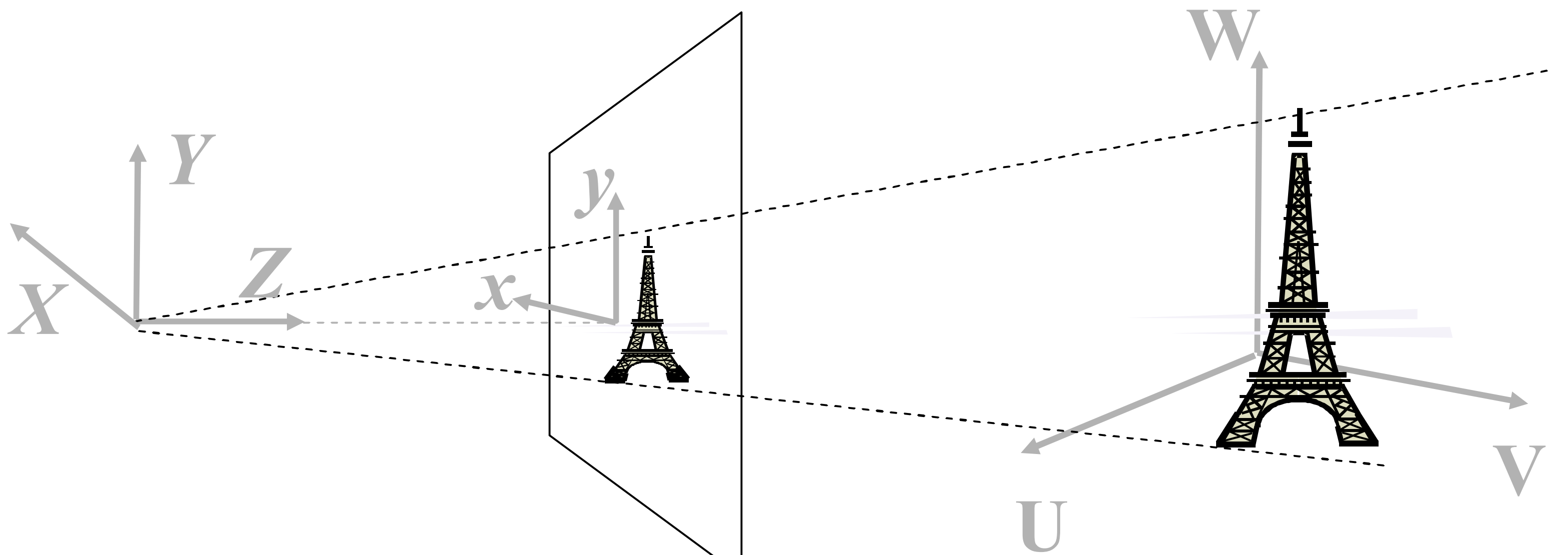
Imaging Geometry



Camera Coordinate System (X,Y,Z).

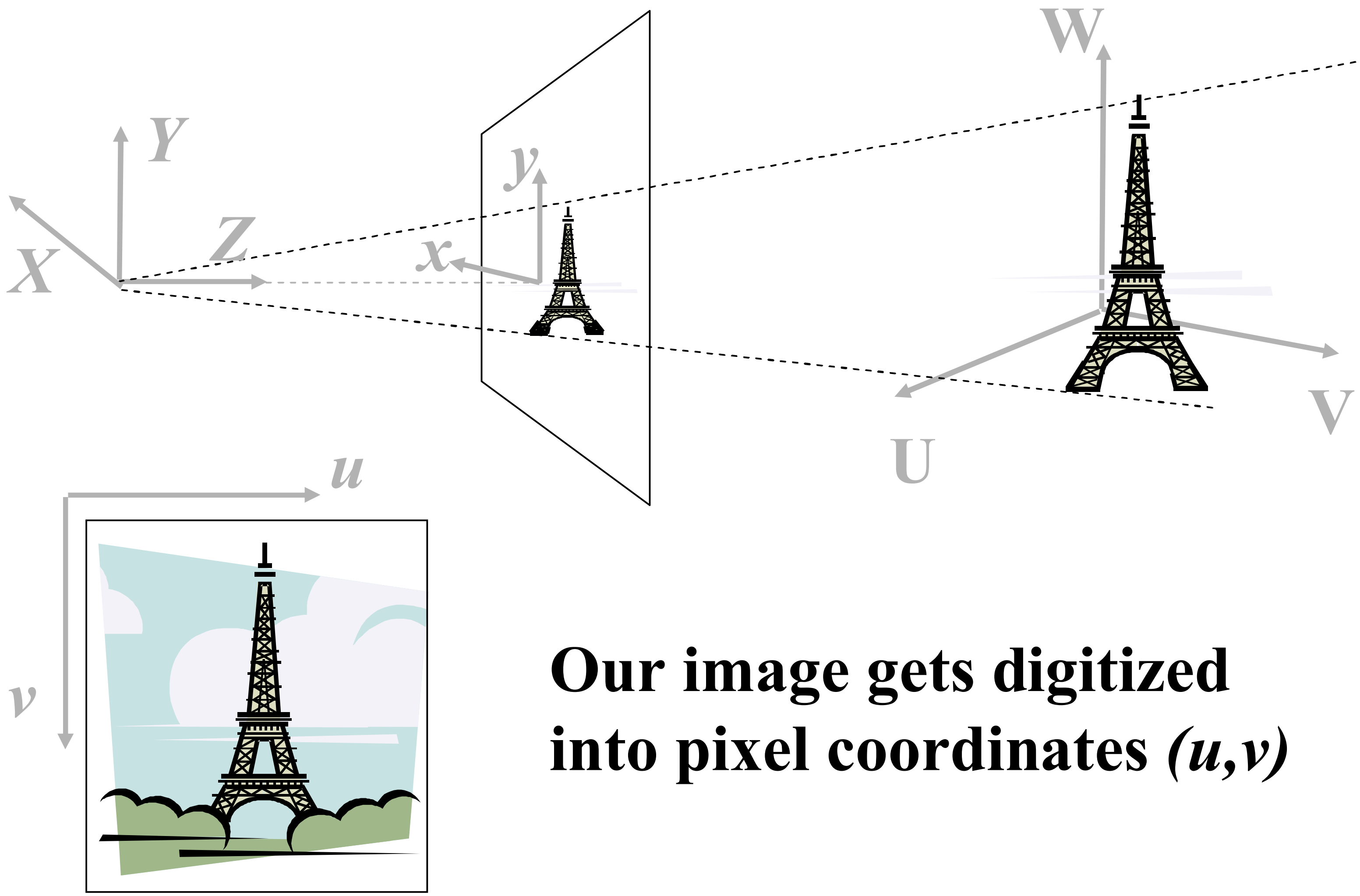
- Z is optic axis
- Image plane located f units out along optic axis
- f is called focal length

Imaging Geometry



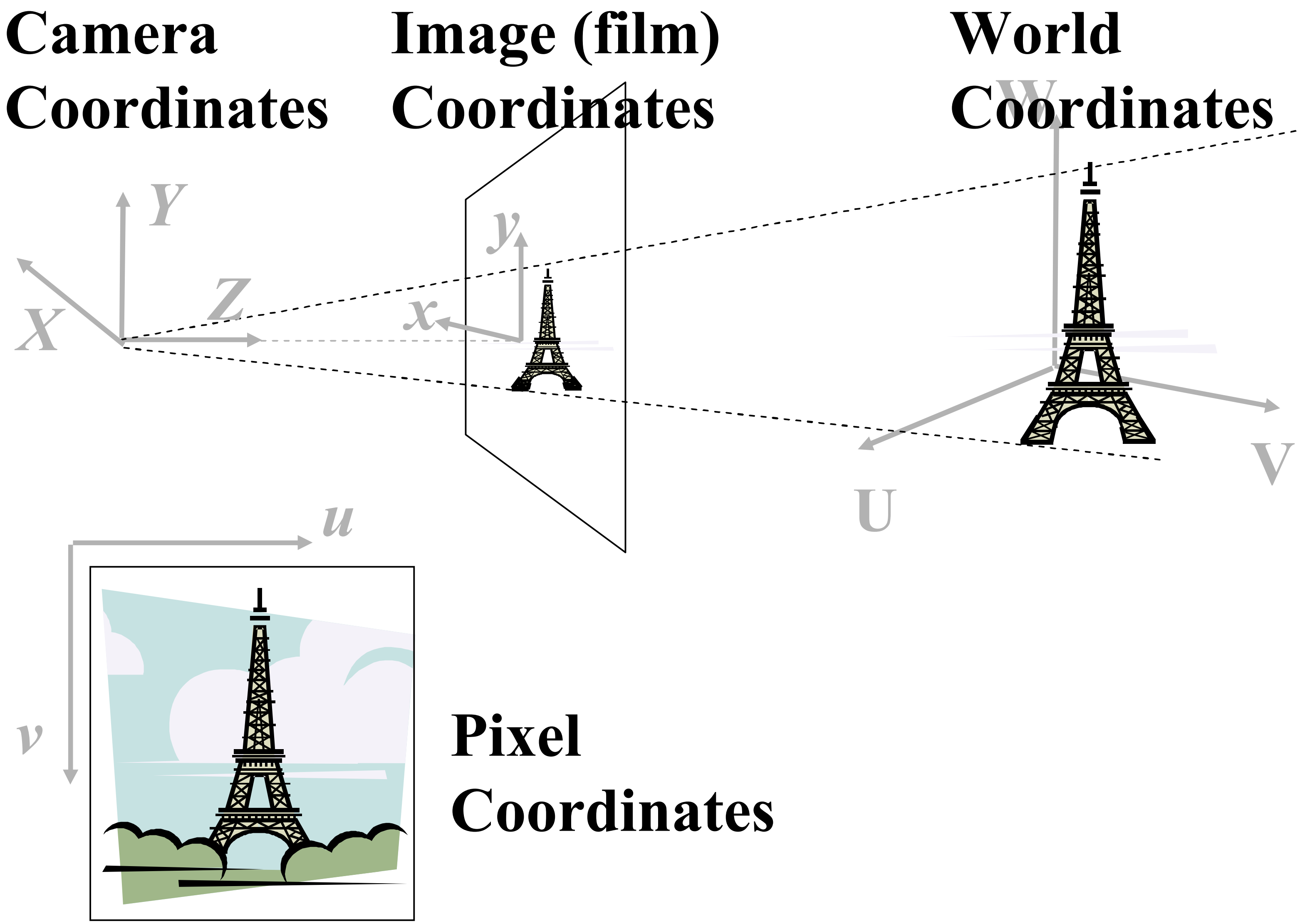
Forward Projection onto image plane.
3D (X,Y,Z) projected to 2D *(x,y)*

Imaging Geometry

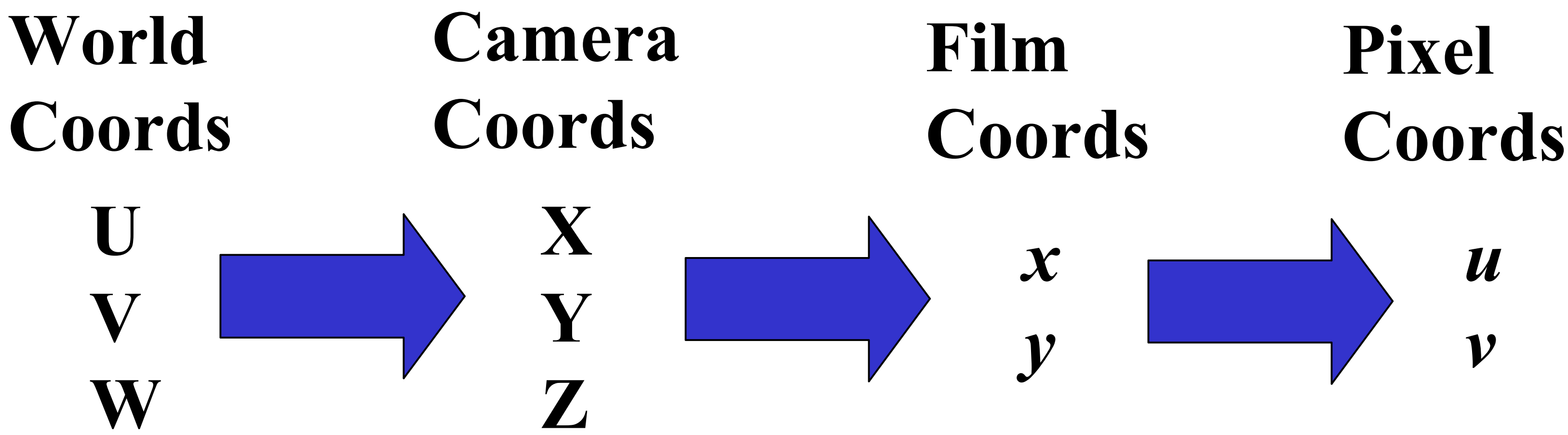


Our image gets digitized into pixel coordinates *(u,v)*

Imaging Geometry



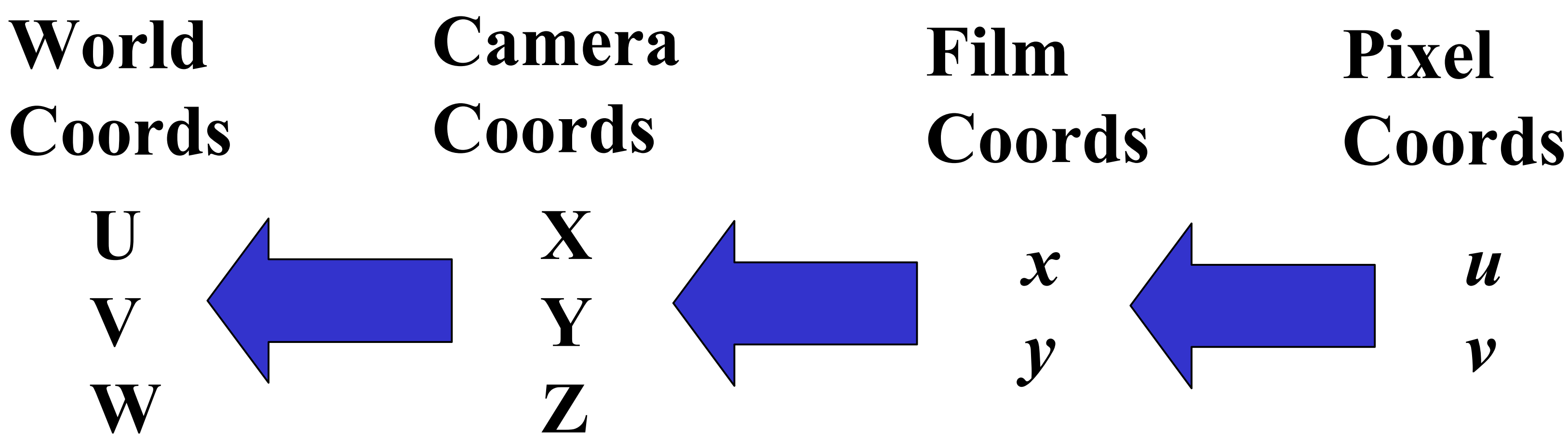
Forward Projection



We want a mathematical model to describe how 3D World points get projected into 2D Pixel coordinates.

Our goal: describe this sequence of transformations by a big matrix equation!

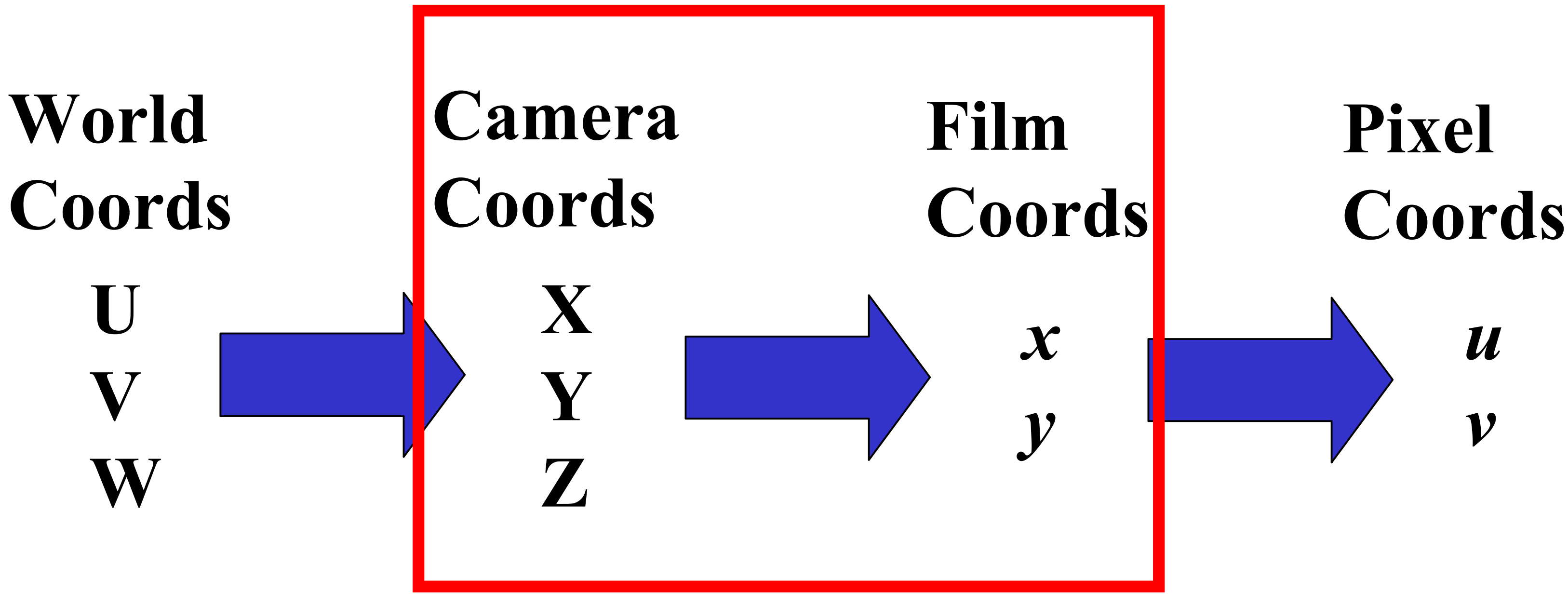
Backward Projection



Note, much of vision concerns trying to derive backward projection equations to recover 3D scene structure from images (via stereo or motion)

But first, we have to understand forward projection…

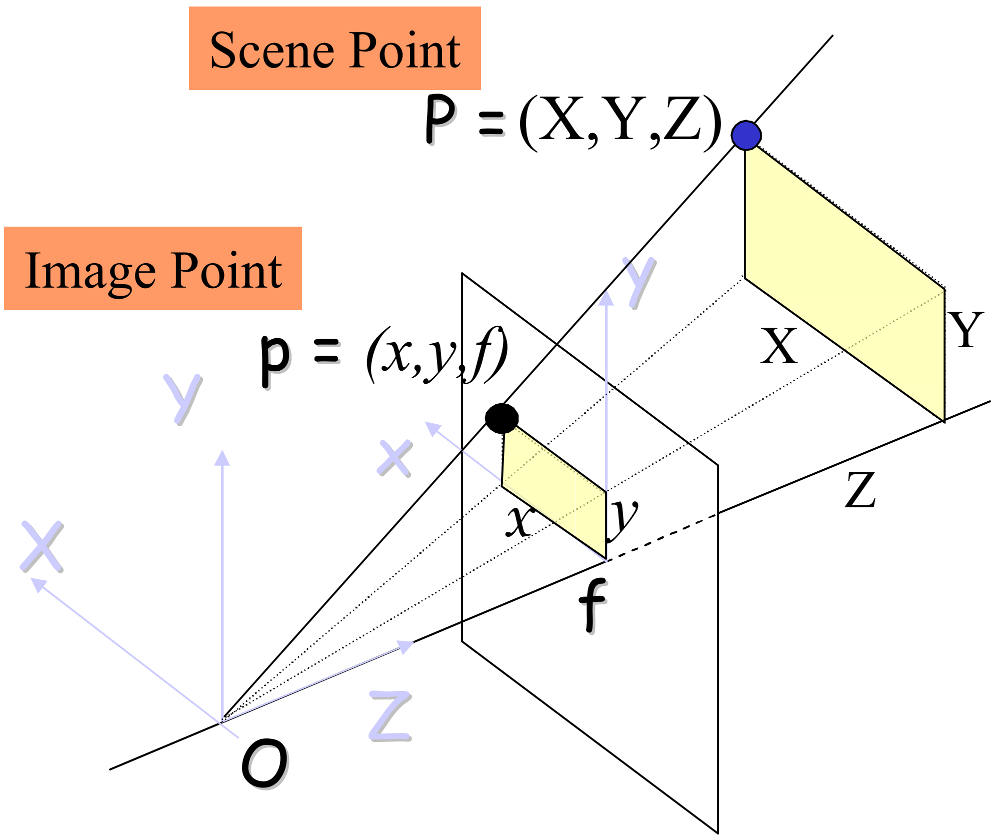
Forward Projection



3D-to-2D Projection
- **perspective projection**

We will start here in the middle, since we've already talked about this when discussing stereo.

Basic Perspective Projection

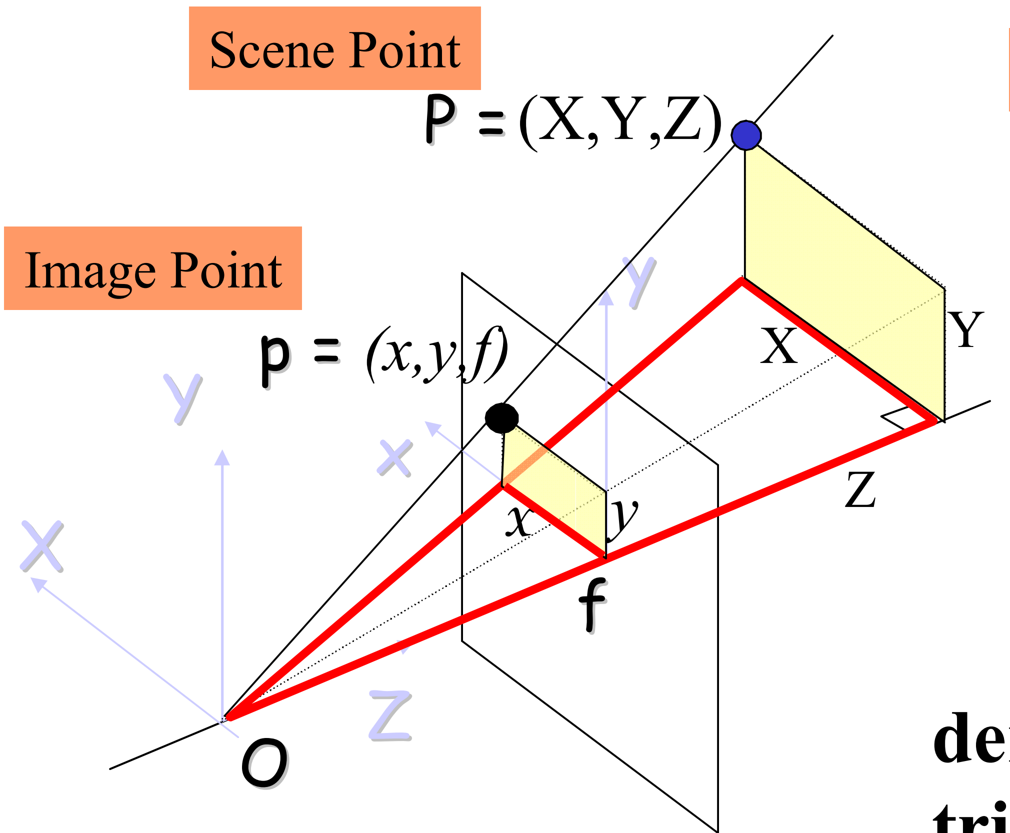


Perspective Projection Eqns

$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

Basic Perspective Projection

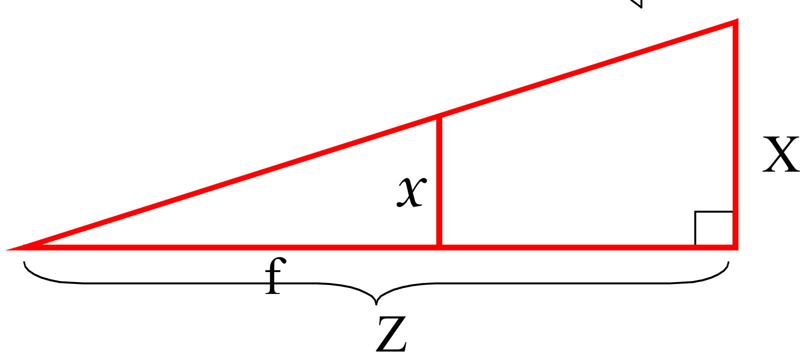


Perspective Projection Eqns

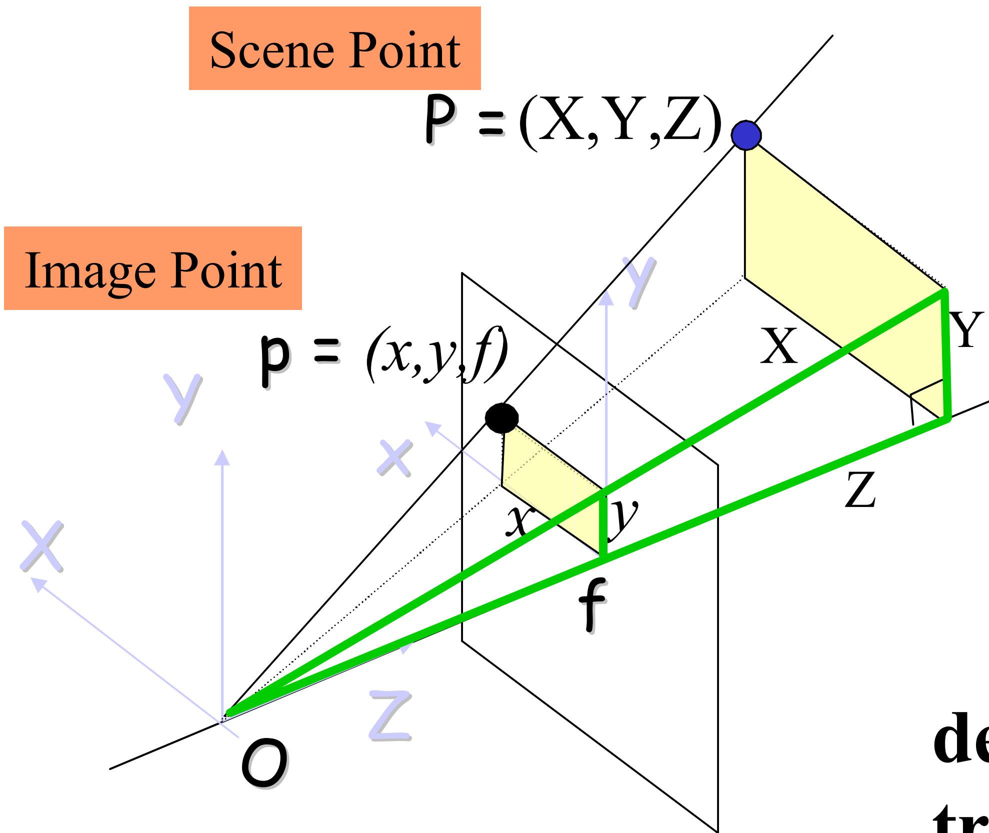
$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

derived via similar triangles rule



Basic Perspective Projection

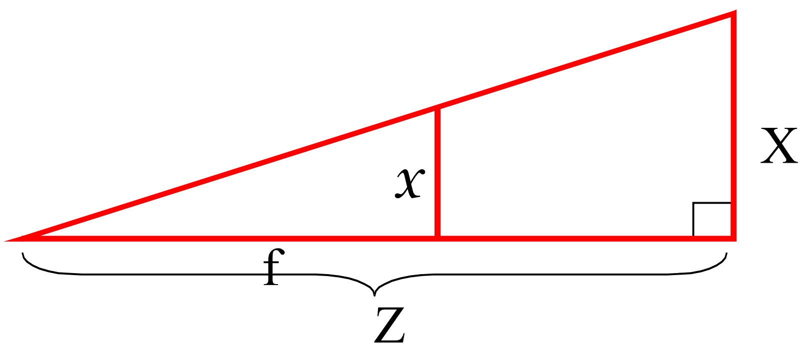


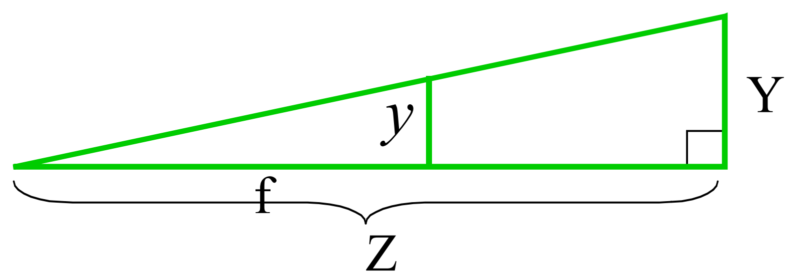
Perspective Projection Eqns

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

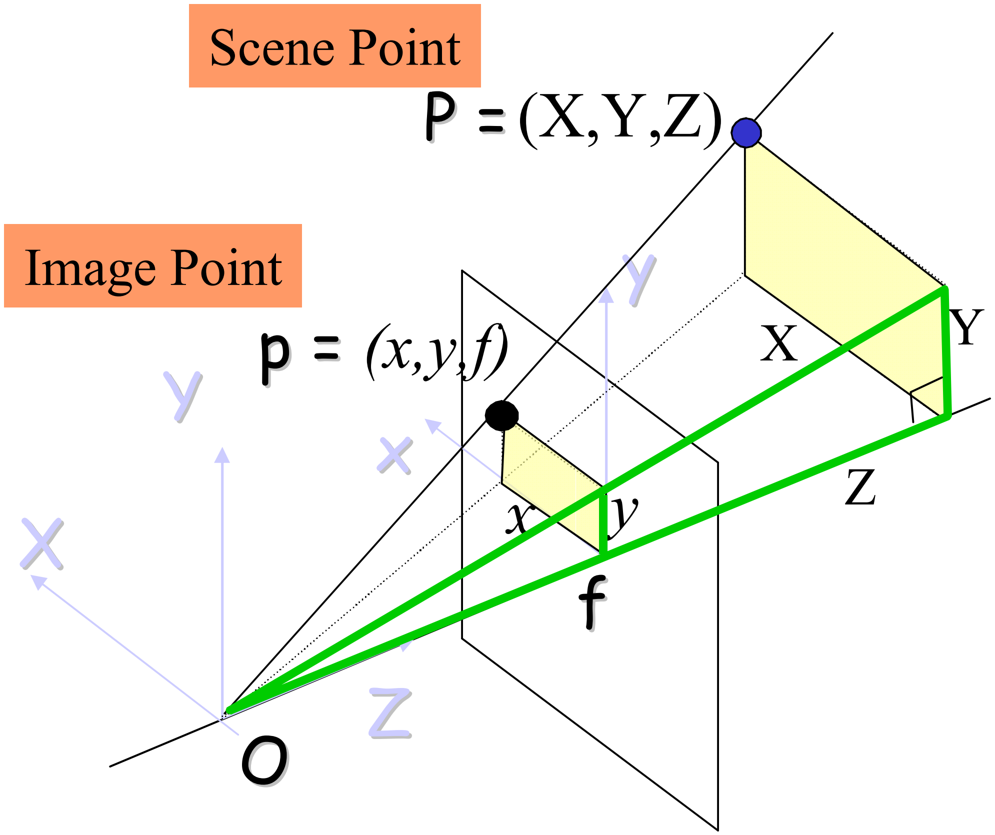
$$y = f\,\frac{Y}{Z}$$

derived via similar triangles rule





Basic Perspective Projection



Perspective Projection Eqns

$$x = f\frac{X}{Z}$$

$$y = f\frac{Y}{Z}$$

So how do we represent this as a matrix equation?
We need to introduce homogeneous coordinates.

Homogeneous Coordinates

Represent a 2D point (x,y) by a 3D point (x',y',z') by adding a "fictitious" third coordinate.

By convention, we specify that given (x',y',z') we can recover the 2D point (x,y) as

$$x = \frac{x'}{z'} \quad y = \frac{y'}{z'}$$

Note: (x,y) = (x,y,1) = (2x, 2y, 2) = (k x, ky, k)
for any nonzero k (can be negative as well as positive)

Perspective Matrix Equation

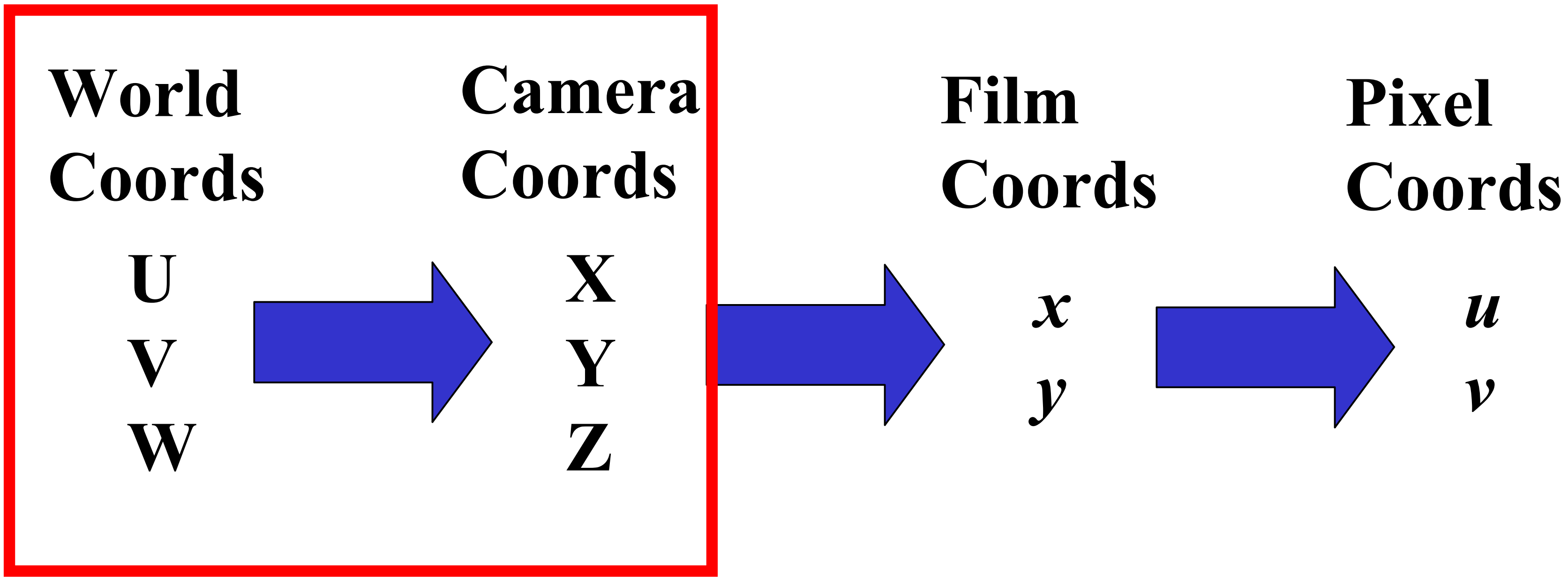
(in Camera Coordinates)

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

$$\Longleftrightarrow$$

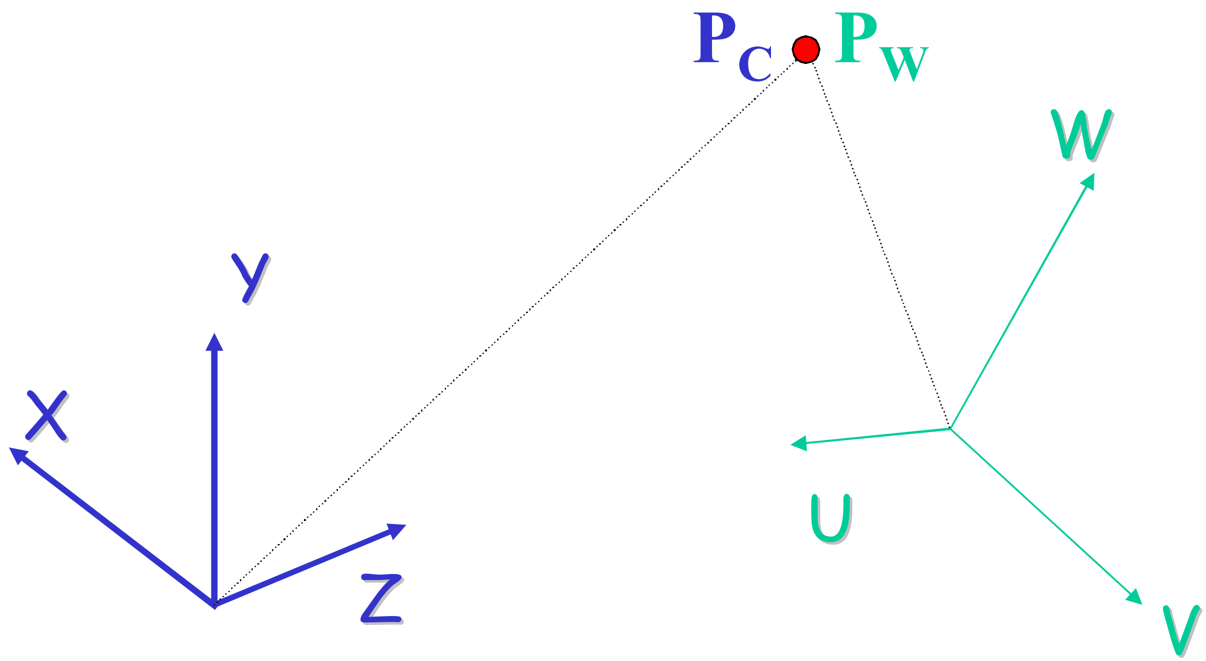
$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Forward Projection



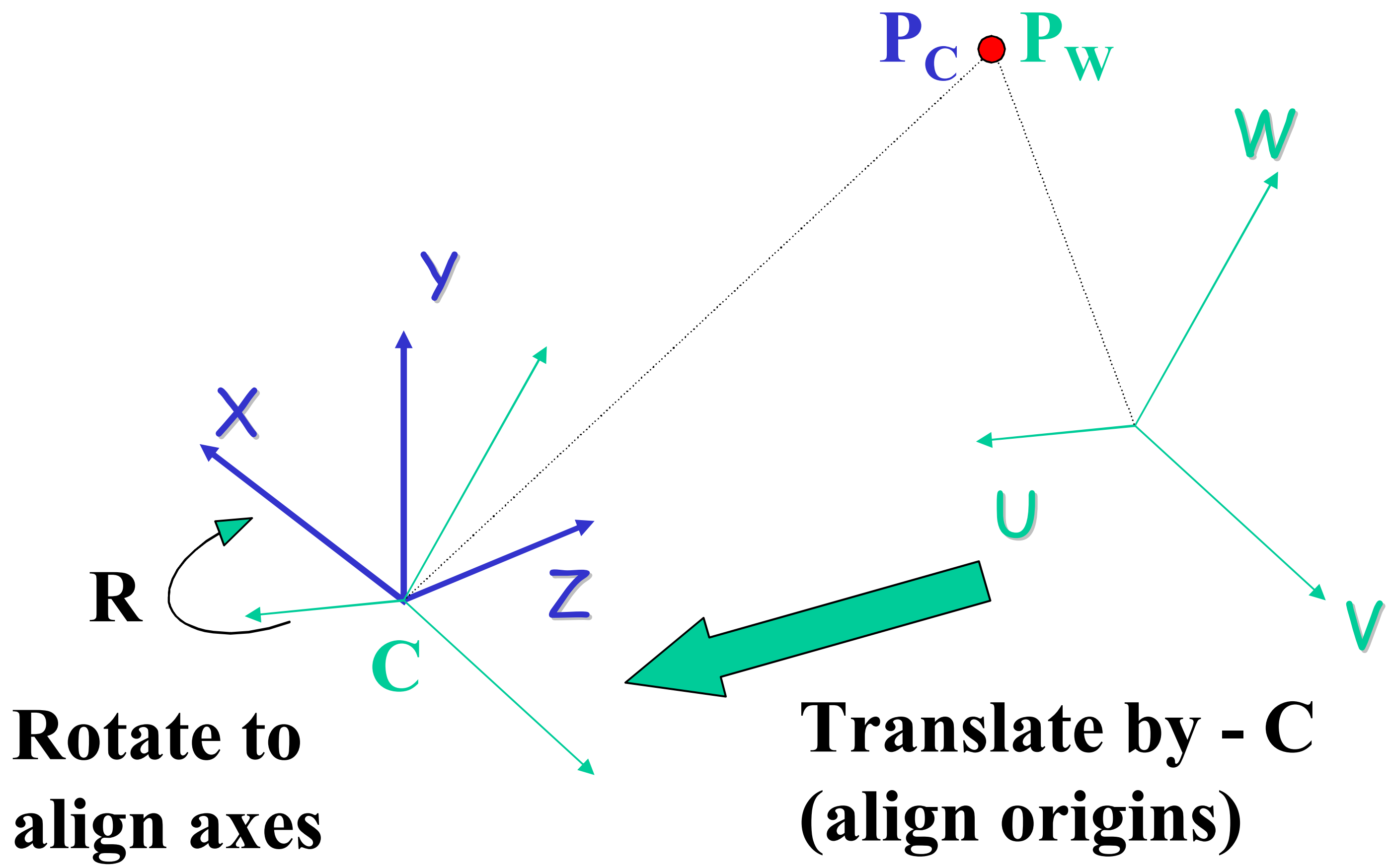
Rigid Transformation (rotation+translation) between world and camera coordinate systems

World to Camera Transformation



Avoid confusion: Pw and Pc are not two different points. They are the same physical point, described in two different coordinate systems.

World to Camera Transformation



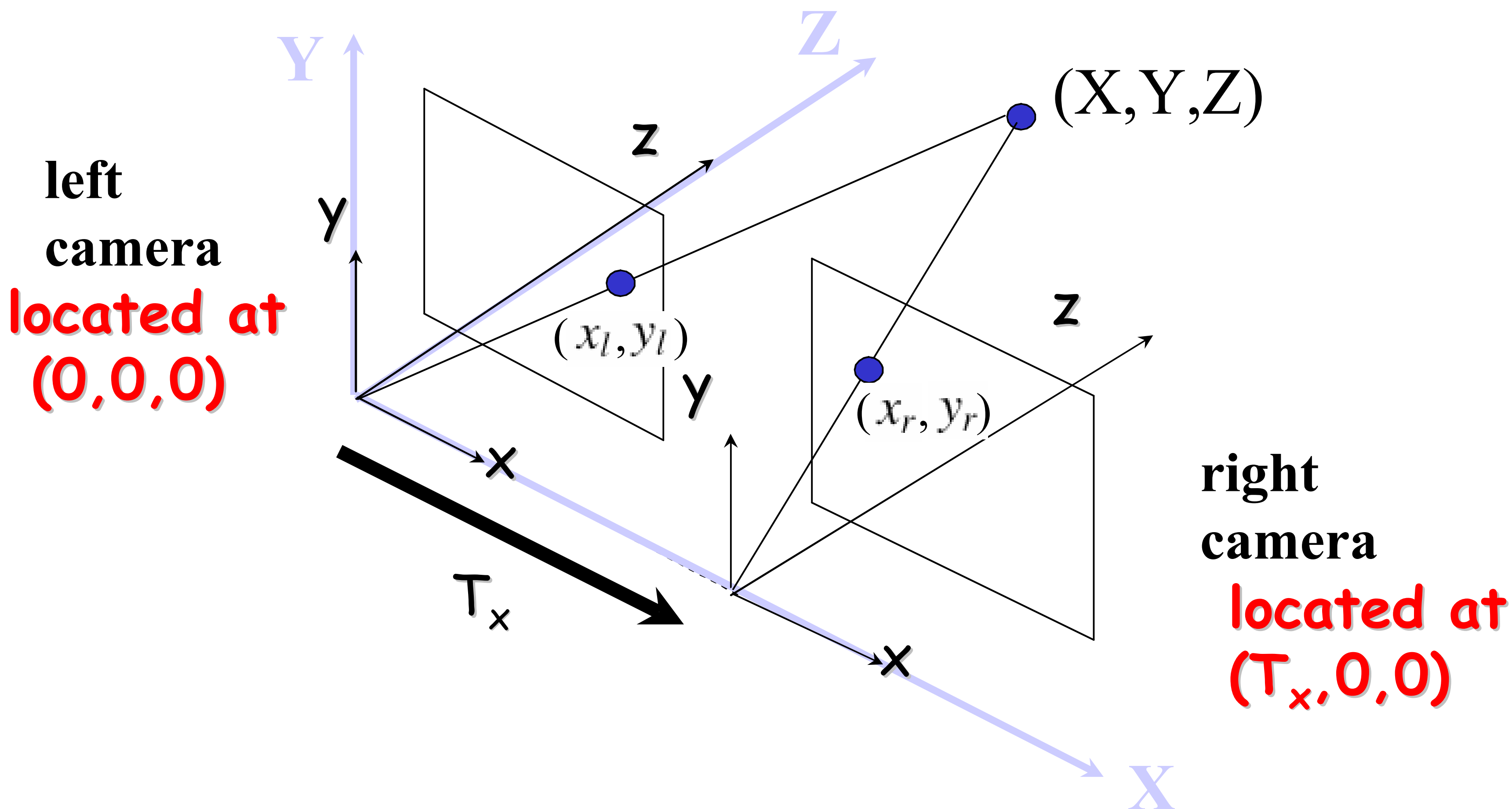
$$P_C = R\,(\,P_W - C\,)$$

Matrix Form, Homogeneous Coords

$$\mathbf{P_C = R\,(\,P_W - C\,)}$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

Example: Simple Stereo System



Left camera located at world origin (0,0,0)
and camera axes aligned with world coord axes.

Simple Stereo, Left Camera

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \mathbf{0} \\ 0 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 1 & \mathbf{0} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

camera axes aligned with world axes

located at world position (0,0,0)

$$= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

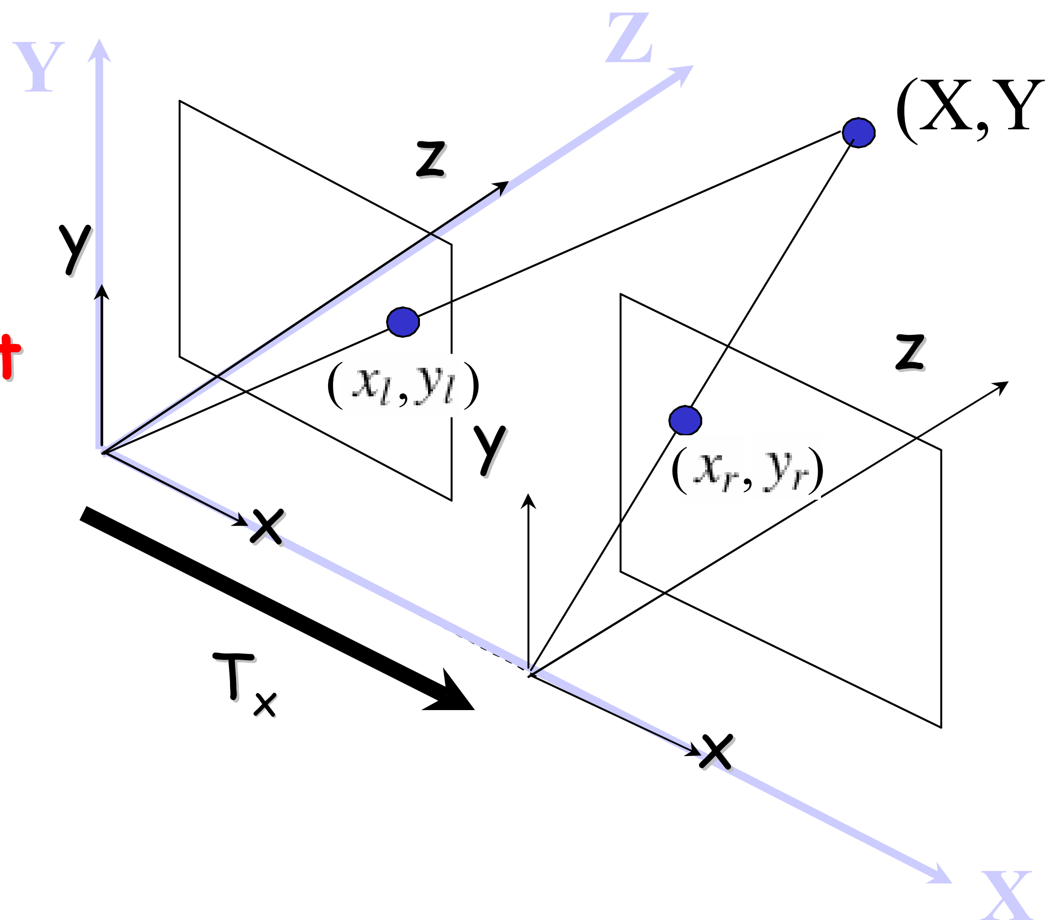
Simple Stereo Projection Equations

Left camera

$$\begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$x_l = f\,\frac{X}{Z} \qquad y_l = f\,\frac{Y}{Z}$$

Example: Simple Stereo System



left camera
located at (0,0,0)

right camera
located at $(T_x,0,0)$

Right camera located at world location (Tx,0,0) and camera axes aligned with world coord axes.

Simple Stereo, Right Camera

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & 0 \\ \mathbf{0} & \mathbf{0} & \mathbf{1} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & \mathbf{-T_x} \\ 0 & 1 & 0 & \mathbf{0} \\ 0 & 0 & 1 & \mathbf{0} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

camera axes aligned with world axes

located at world position (T_x,0,0)

$$= \begin{bmatrix} 1 & 0 & 0 & -T_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Simple Stereo Projection Equations

Left camera

$$\begin{bmatrix} x_l \\ y_l \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$x_l = f\,\frac{X}{Z} \qquad y_l = f\,\frac{Y}{Z}$$

Right camera

$$\begin{bmatrix} x_r \\ y_r \\ 1 \end{bmatrix} \sim \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & -T_x \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$x_r = f\,\frac{X - T_x}{Z} \qquad y_r = f\,\frac{Y}{Z}$$

Bob's sure-fire way(s) to figure out the rotation

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \cancel{\begin{pmatrix} 1 & 0 & 0 & -c_x \\ & & & \\ & & & \\ 0 & 0 & 0 & 1 \end{pmatrix}} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

forget about this while thinking about rotations

$$\mathbf{P_C = R\ P_W}$$

This equation says how vectors in the world coordinate system (including the coordinate axes) get transformed into the camera coordinate system.

Figuring out Rotations

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

$$\mathbf{P_C = R\, P_W}$$

what if world x axis (1,0,0) corresponds to camera axis (a,b,c)?

$$\begin{pmatrix} a \\ b \\ c \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} a \\ b \\ c \\ 1 \end{pmatrix} = \begin{pmatrix} a & r_{12} & r_{13} & 0 \\ b & r_{22} & r_{23} & 0 \\ c & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

we can immediately write down the first column of R!

Figuring out Rotations

and likewise with world Y axis and world Z axis...

$$\underset{\text{same axis in camera coords}}{\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \underset{\text{axis is world coords}}{\begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}}$$

world X axis (1,0,0) in camera coords

world Y axis (0,1,0) in camera coords

world Z axis (0,0,1) in camera coords

Figuring out Rotations

Alternative approach: sometimes it is easier to specify what camera X,Y,or Z axis is in world coordinates. Then do rearrange the equation as follows.

$$\mathbf{P_C = R\ P_W} \Rightarrow \mathbf{R^{-1}P_C = P_W} \Rightarrow \mathbf{R^T P_C = P_W}$$

$$\begin{pmatrix} r_{11} & r_{21} & r_{31} & 0 \\ r_{12} & r_{22} & r_{32} & 0 \\ r_{13} & r_{23} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

Figuring out Rotations

$$\begin{pmatrix} r_{11} & r_{21} & r_{31} & 0 \\ r_{12} & r_{22} & r_{32} & 0 \\ r_{13} & r_{23} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix} \qquad \mathbf{R^T P_C = P_W}$$

what if camera X axis (1,0,0) corresponds to world axis (a,b,c)?

$$\begin{pmatrix} r_{11} & r_{21} & r_{31} & 0 \\ r_{12} & r_{22} & r_{32} & 0 \\ r_{13} & r_{23} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ 1 \end{pmatrix} \Rightarrow \begin{pmatrix} \mathbf{a} & r_{21} & r_{31} & 0 \\ \mathbf{b} & r_{22} & r_{32} & 0 \\ \mathbf{c} & r_{23} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{a} \\ \mathbf{b} \\ \mathbf{c} \\ 1 \end{pmatrix}$$

we can immediately write down the first column of R^T,
(which is the first row of R).

Figuring out Rotations

and likewise with camera Y axis and camera Z axis...

same axis in camera coords

axis is world coords

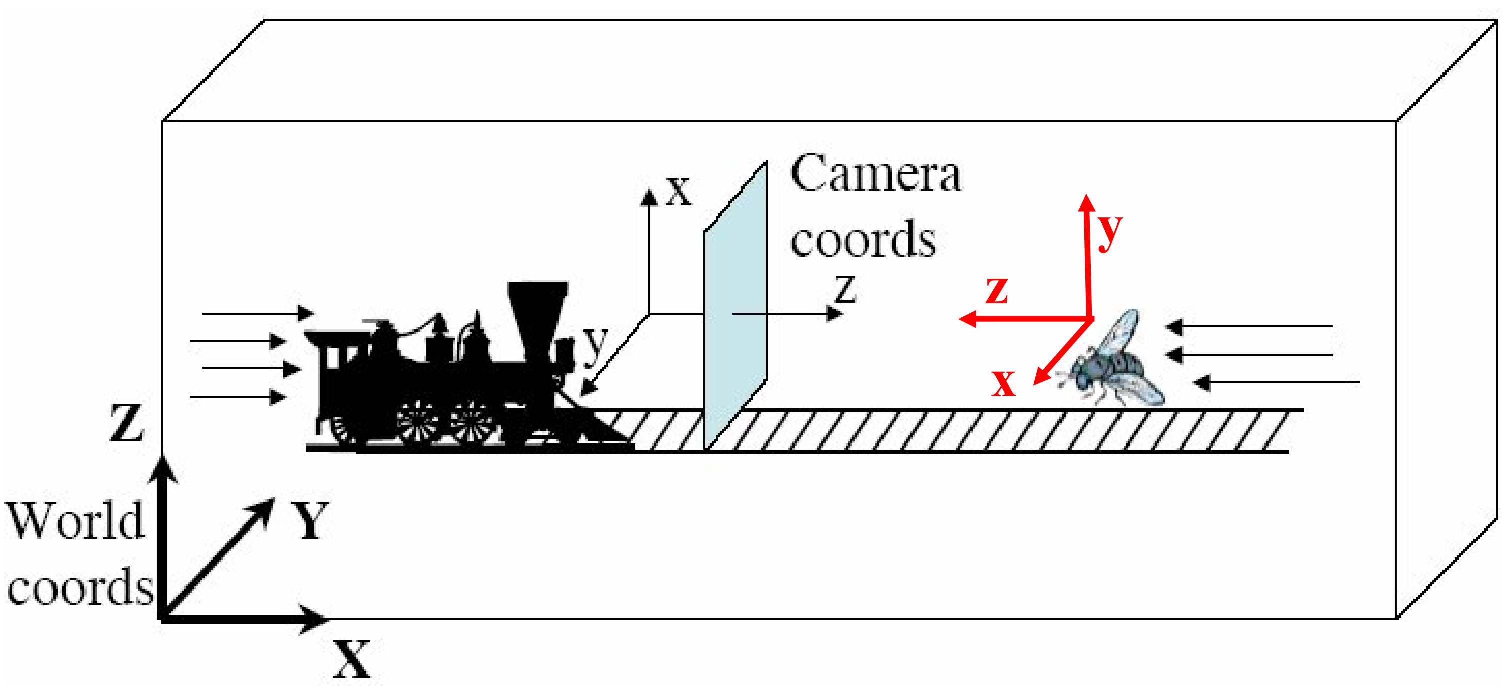
$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

camera X axis (1,0,0)
in world coords

camera Y axis (0,1,0)
in world coords

camera Z axis (0,0,1)
in world coords

Example



$$R_{train} \quad \begin{matrix} 0 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 0 \end{matrix} \qquad\qquad R_{fly} \quad \begin{matrix} 0 & -1 & 0 \\ 0 & 0 & 1 \\ -1 & 0 & 0 \end{matrix}$$

Note: External Parameters also often written as R,T

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} = \underbrace{\begin{pmatrix} r_{11} & r_{12} & r_{13} & 0 \\ r_{21} & r_{22} & r_{23} & 0 \\ r_{31} & r_{32} & r_{33} & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} 1 & 0 & 0 & -c_x \\ 0 & 1 & 0 & -c_y \\ 0 & 0 & 1 & -c_z \\ 0 & 0 & 0 & 1 \end{pmatrix}} \begin{pmatrix} U \\ V \\ W \\ 1 \end{pmatrix}$$

$$\begin{aligned} &\mathbf{R}\,(\mathbf{P_W} - \mathbf{C}) \\ &= \mathbf{R}\,\mathbf{P_W} - \mathbf{R}\,\mathbf{C} \\ &= \mathbf{R}\,\mathbf{P_W} + \mathbf{T} \end{aligned} \qquad \begin{pmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

Summary

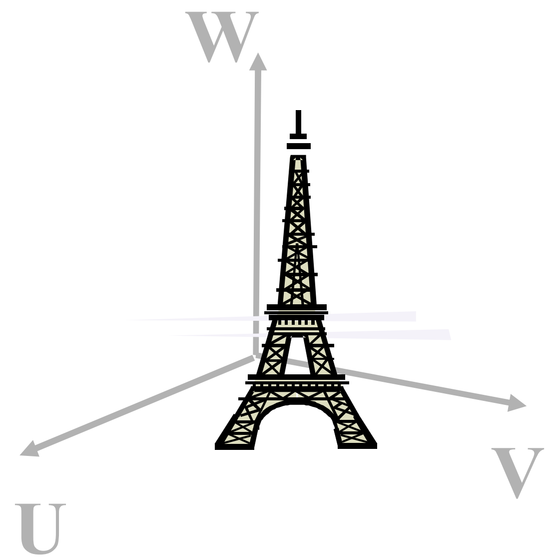
World Coords		Camera Coords		Film Coords		Pixel Coords
U V W	$\rightarrow$	X Y Z	$\rightarrow$	x y	$\rightarrow$	u v

We now know how to transform 3D world coordinate points into camera coords, and then do perspective project to get 2D points in the film plane.

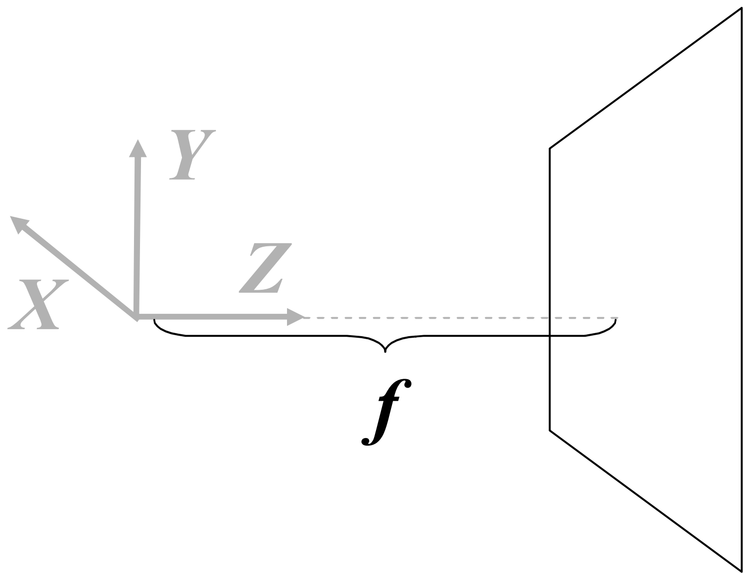
Next time: pixel coordinates

Recall: Imaging Geometry

Object of Interest in World Coordinate System (U,V,W)



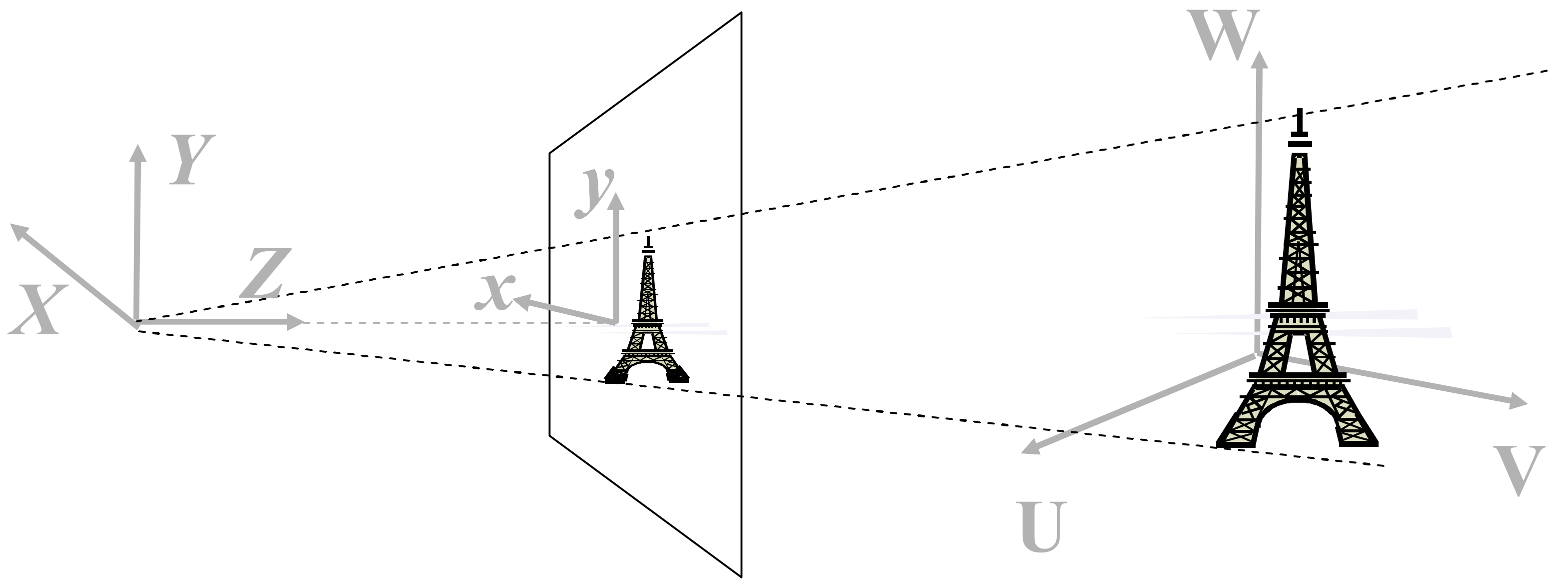
Imaging Geometry



Camera Coordinate System (X,Y,Z).

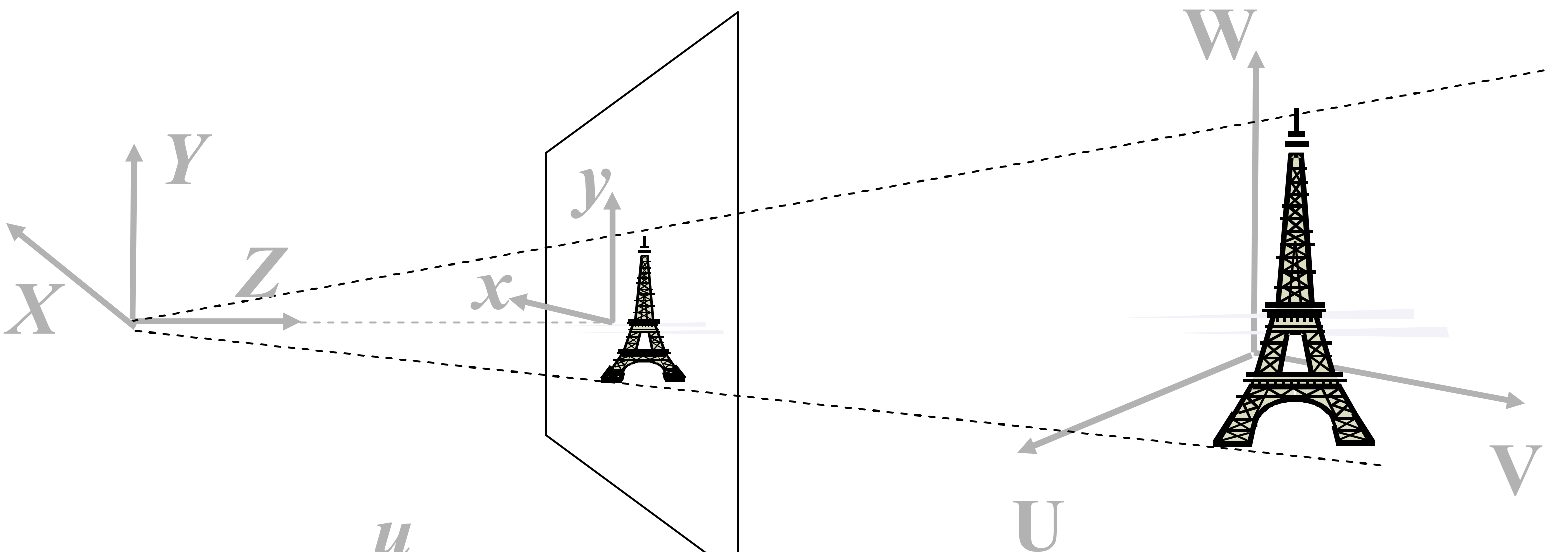
- Z is optic axis
- Image plane located f units out along optic axis
- f is called focal length

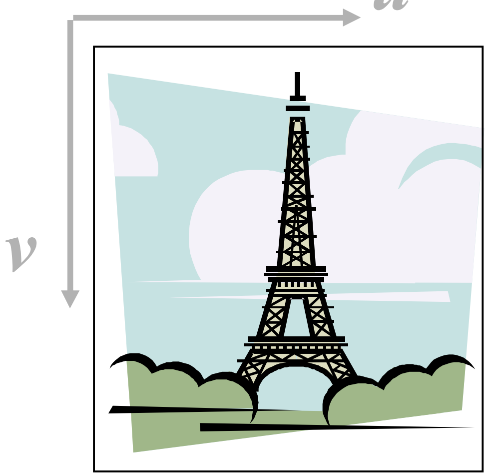
Imaging Geometry



Forward Projection onto image plane.
3D (X,Y,Z) projected to 2D *(x,y)*

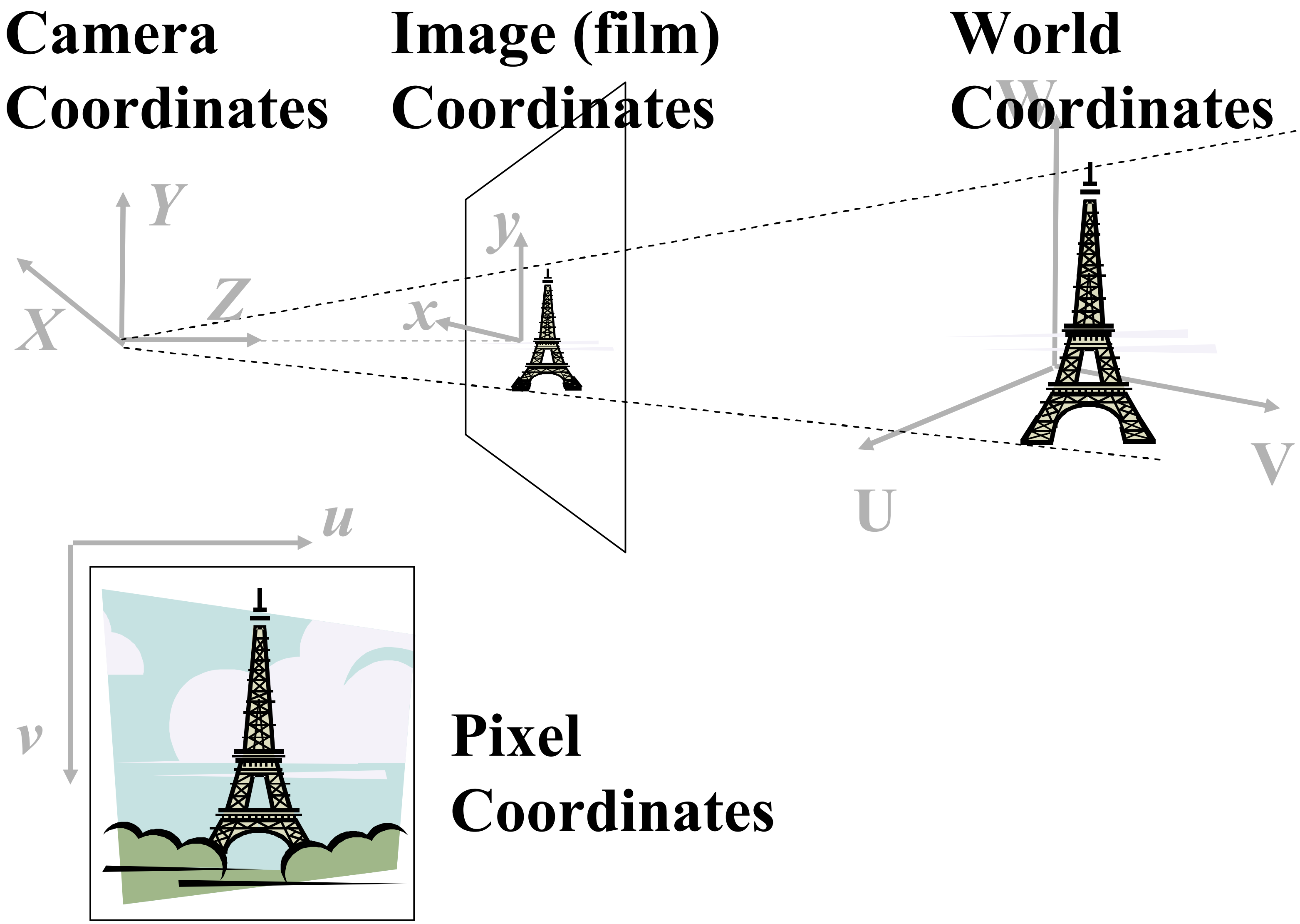
Imaging Geometry





Our image gets digitized into pixel coordinates *(u,v)*

Imaging Geometry



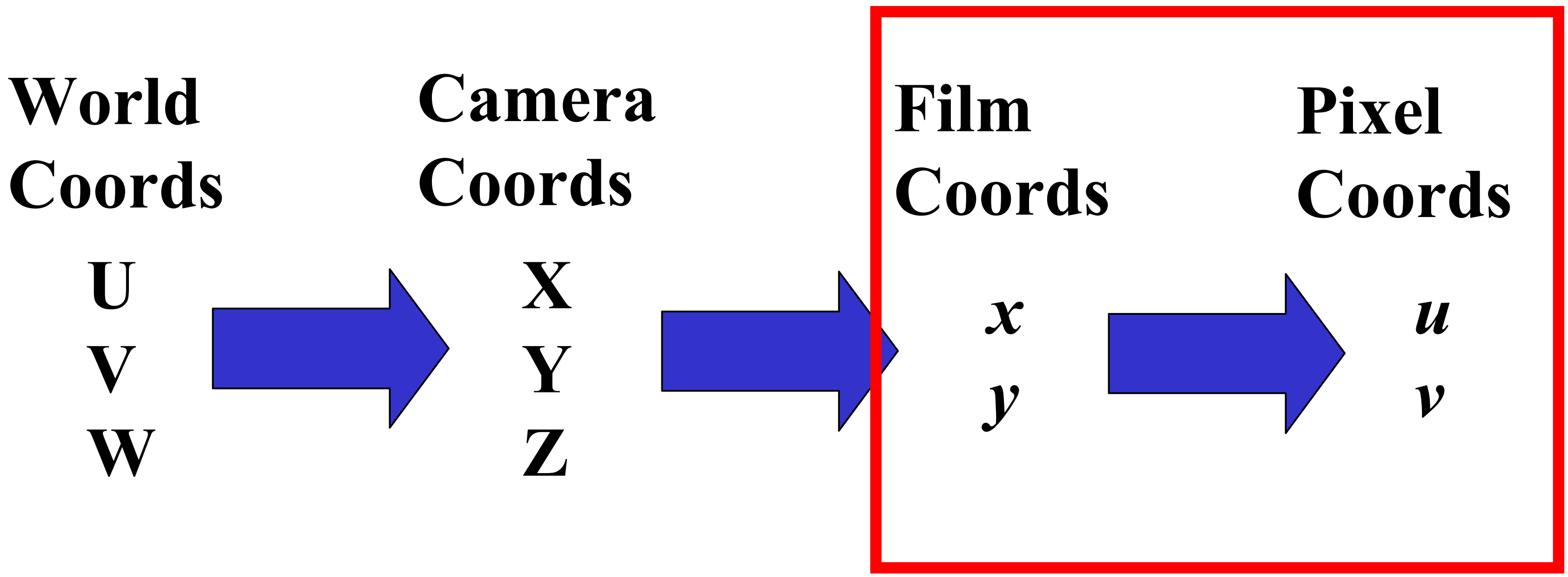
Forward Projection

World Coords		Camera Coords		Film Coords		Pixel Coords
U V W	$\Rightarrow$	X Y Z	$\Rightarrow$	x y	$\Rightarrow$	u v

We want a mathematical model to describe how 3D World points get projected into 2D Pixel coordinates.

Our goal: describe this sequence of transformations by a big matrix equation!

Intrinsic Camera Parameters



Affine Transformation

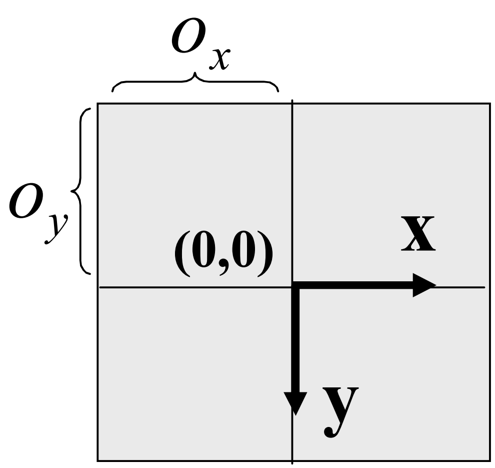
Intrinsic parameters

- Describes coordinate transformation between film coordinates (projected image) and pixel array
- Film cameras: scanning/digitization
- CCD cameras: grid of photosensors

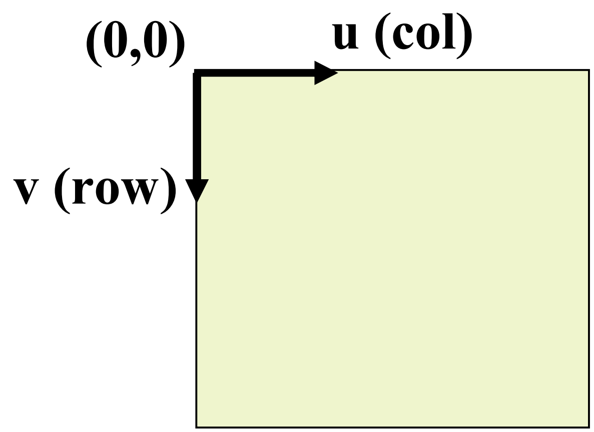
still in T&V section 2.4

Intrinsic parameters (offsets)

film plane
(projected image)



pixel array

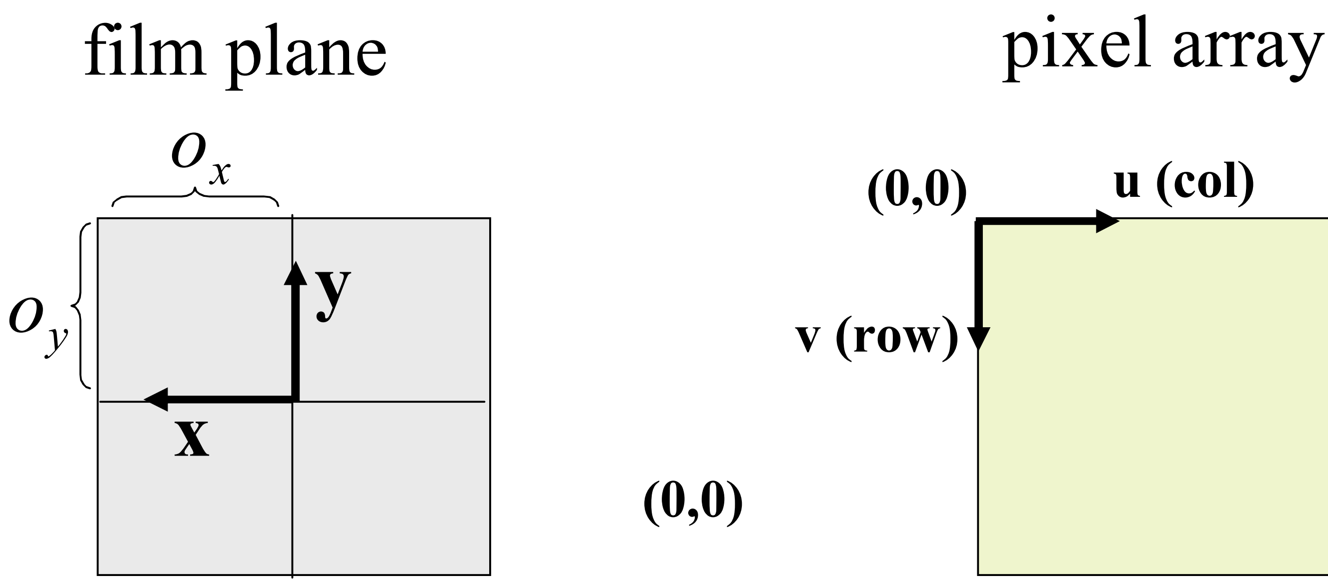


$$u = f\frac{X}{Z} + o_x \qquad v = f\frac{Y}{Z} + o_y$$

o_x and o_y called image center or principle point

Intrinsic parameters

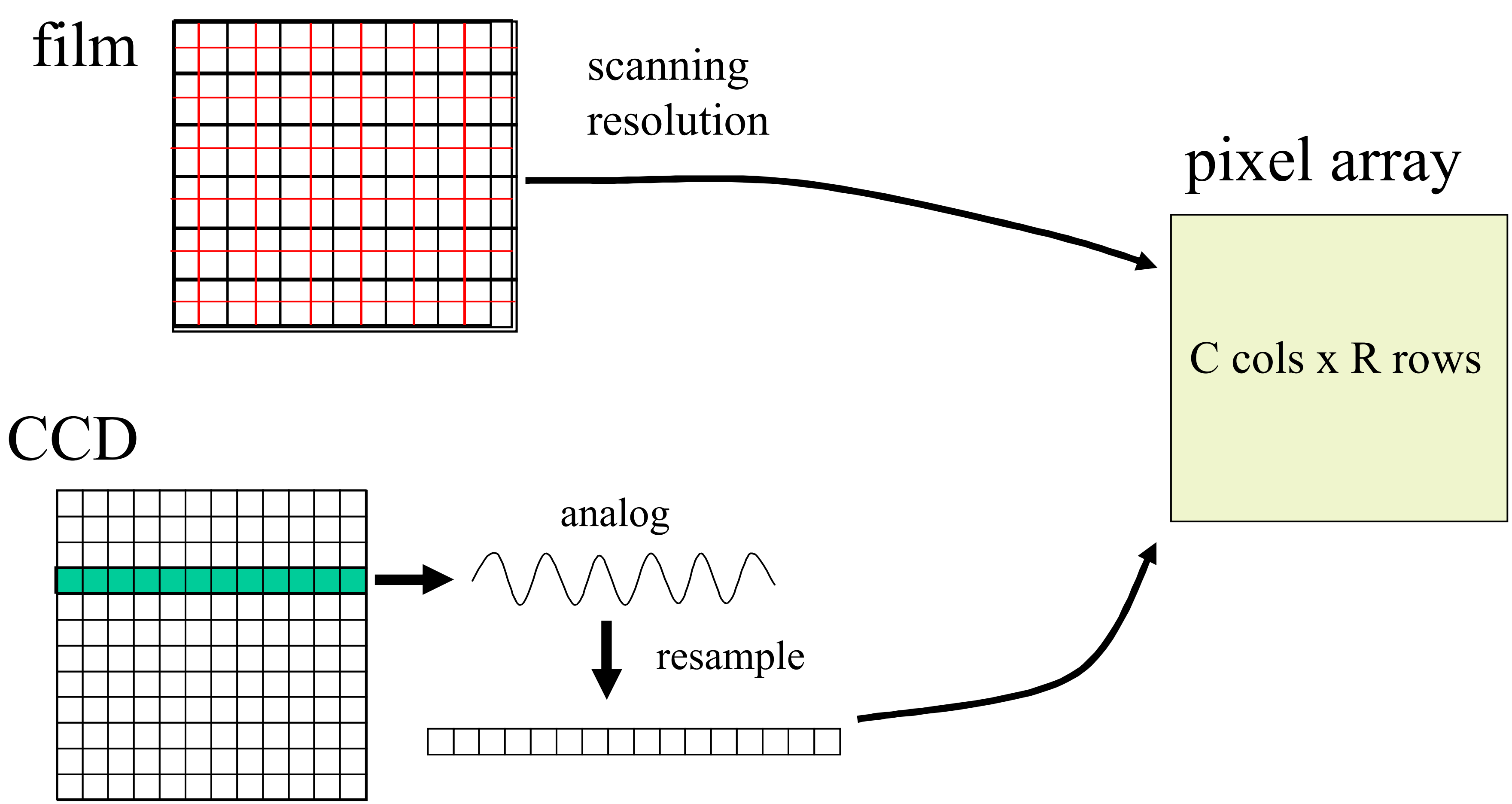
sometimes one or more coordinate axes are flipped (e.g. T&V section 2.4)



$$u = -f\frac{X}{Z} + o_x \qquad v = -f\frac{Y}{Z} + o_y$$

Intrinsic parameters (scales)

sampling determines how many rows/cols in the image



Effective Scales: s_x and s_y

$$u = \frac{1}{s_x} f \frac{X}{Z} + o_x \qquad v = \frac{1}{s_y} f \frac{Y}{Z} + o_y$$

Note, since we have different scale factors in x and y, we don't necessarily have square pixels!

Aspect ratio is s_y / s_x

Perspective projection matrix

Adding the intrinsic parameters into the perspective projection matrix:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} f/s_x & 0 & o_x & 0 \\ 0 & f/s_y & o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

To verify:

$$u = \frac{x'}{z'} \qquad v = \frac{y'}{z'}$$

$$\Rightarrow \quad u = \frac{1}{\mathrm{s_x}} f \frac{X}{Z} + o_x \qquad v = \frac{1}{\mathrm{s_y}} f \frac{Y}{Z} + o_y$$

Note:

Sometimes, the image and the camera coordinate systems have opposite orientations: [the book does it this way]

$$f\frac{X}{Z} = -(u - o_x)s_x$$

$$f\frac{Y}{Z} = -(v - o_y)s_y$$

$$\Rightarrow$$

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} -f/s_x & 0 & +o_x & 0 \\ 0 & -f/s_y & +o_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

Note 2

In general, I like to think of the conversion as a separate 2D affine transformation from film coords (x,y) to pixel coordinates (u,v):

$$\begin{pmatrix} u' \\ v' \\ w' \end{pmatrix} = \underbrace{\begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{pmatrix}}_{\mathbf{M_{aff}}} \underbrace{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}_{\mathbf{M_{proj}}} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{u} = \mathbf{M_{int}}\,\mathbf{P_C} = \mathbf{M_{aff}}\,\mathbf{M_{proj}}\,\mathbf{P_C}$$

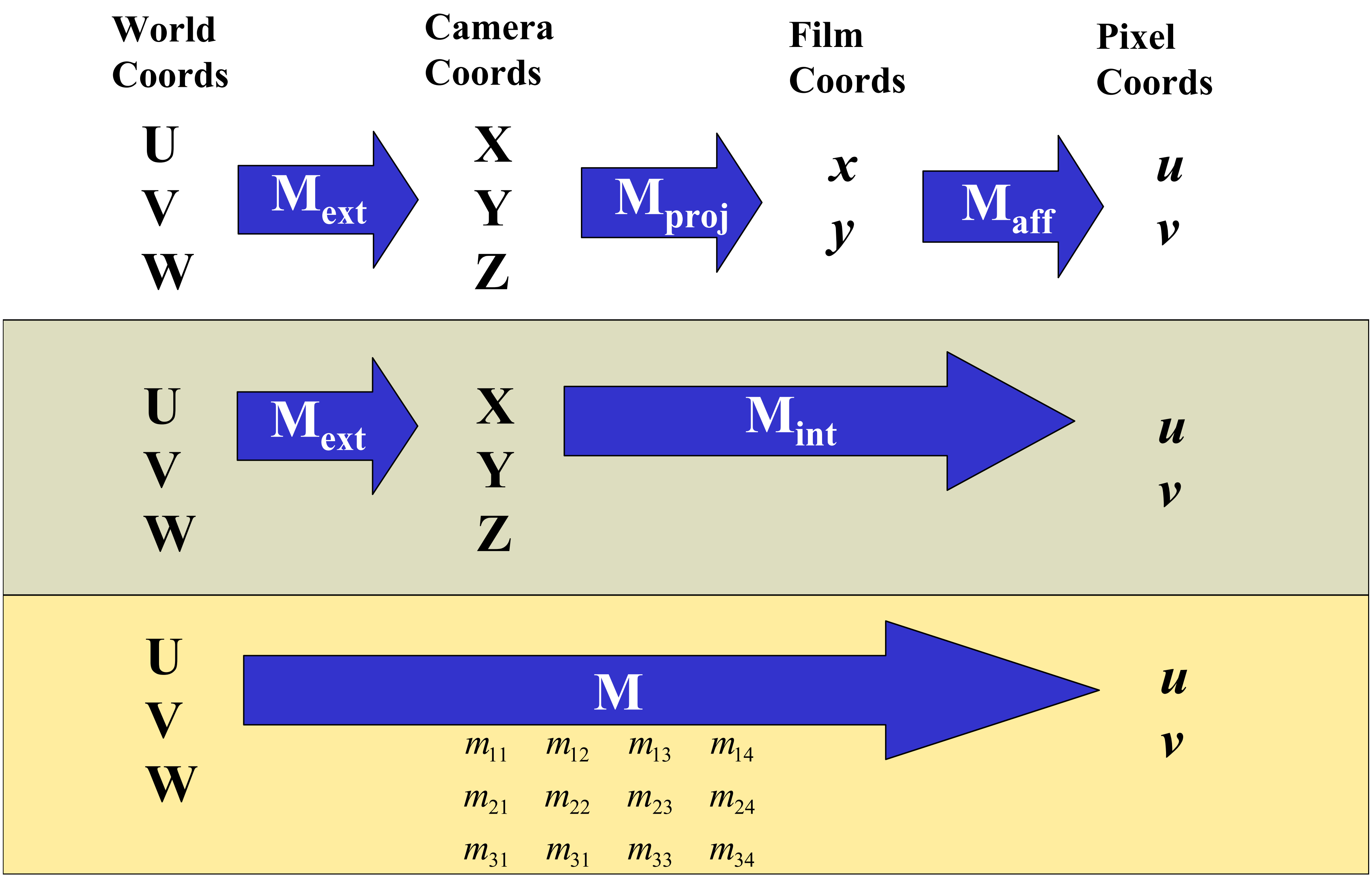
Huh?

Did he just say it was “a fine” transformation?

No, it was “affine” transformation, a type of 2D to 2D mapping defined by 6 parameters.

More on this in a moment...

Summary : Forward Projection



Summary: Projection Equation

$$
\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} \sim
\underbrace{\underbrace{\underbrace{\overset{\text{Film plane to pixels}}{\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ 0 & 0 & 1 \end{bmatrix}}}_{\mathbf{M_{aff}}}
\underbrace{\overset{\text{Perspective projection}}{\begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}}}_{\mathbf{M_{proj}}}}_{\mathbf{M_{int}}}
\underbrace{\overset{\text{World to camera}}{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}}}_{\mathbf{M_{ext}}}}_{\mathbf{M}}
\begin{bmatrix} U \\ V \\ W \\ 1 \end{bmatrix}
$$

Intro to Image Mappings

Image Mappings
Overview

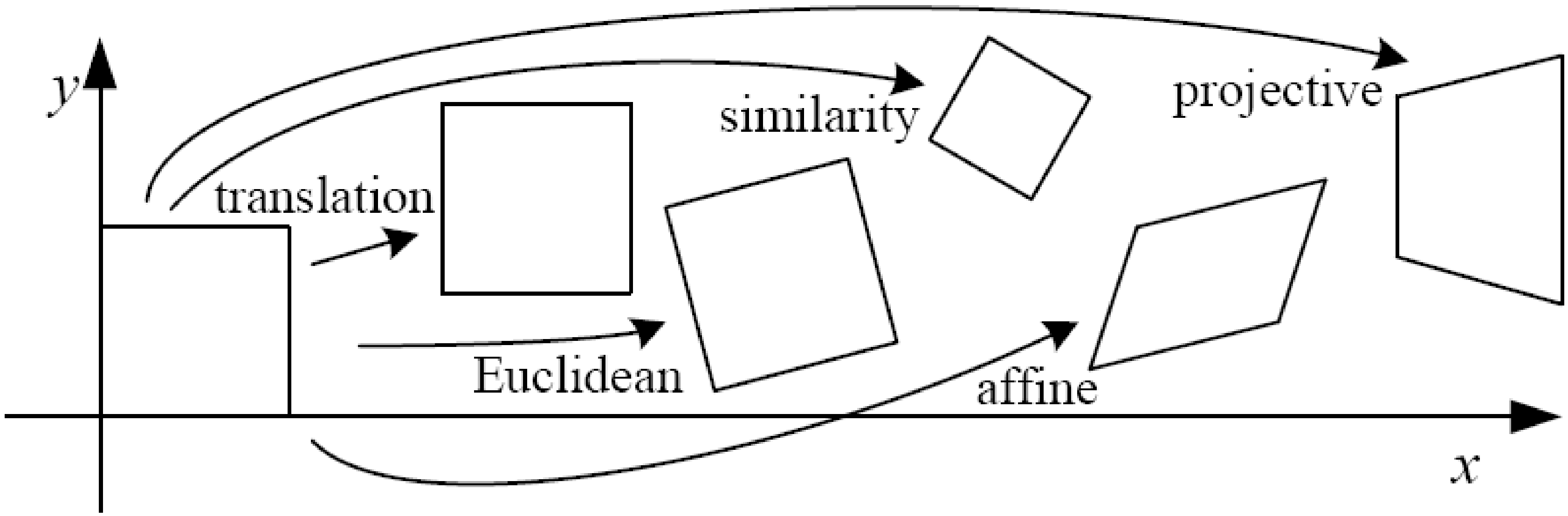
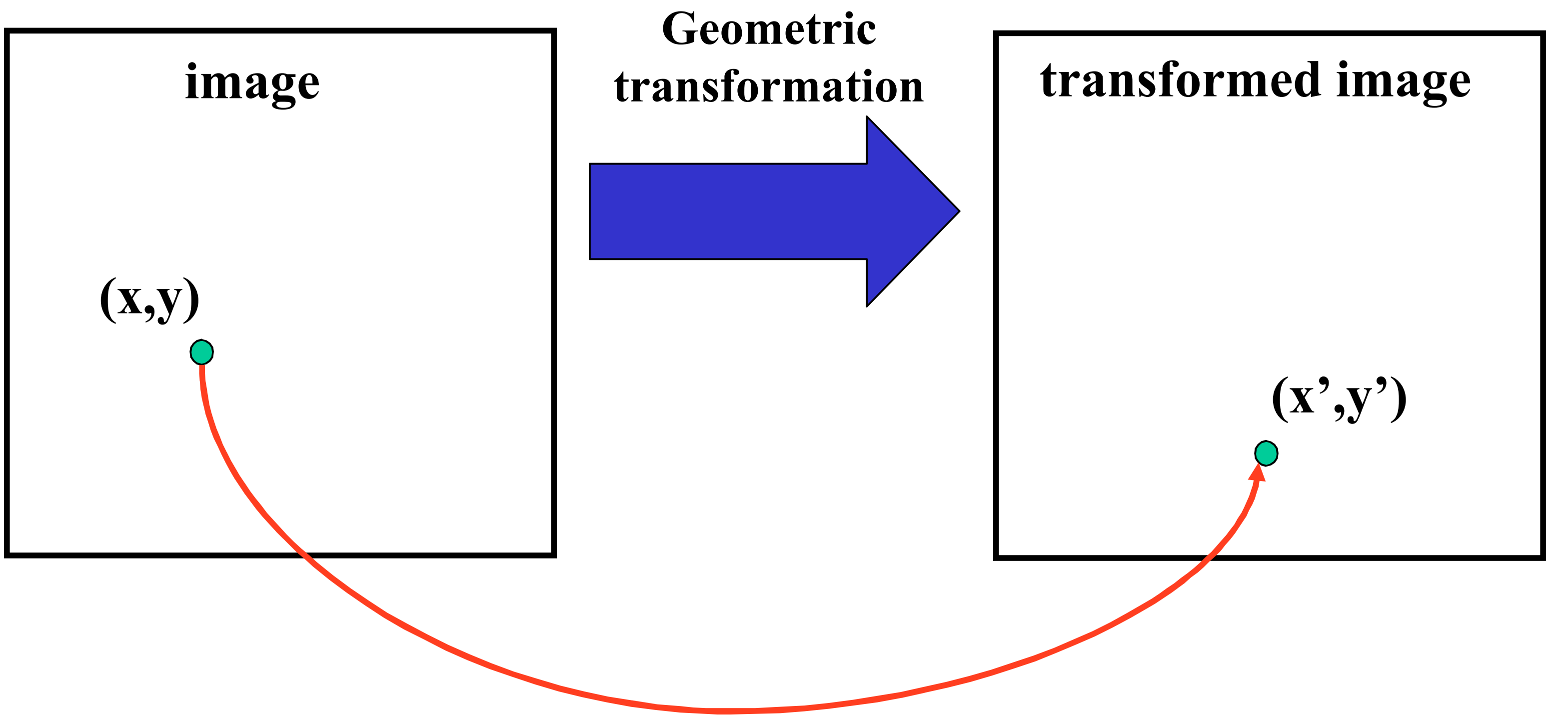


FIGURE 1. Basic set of 2D planar transformations

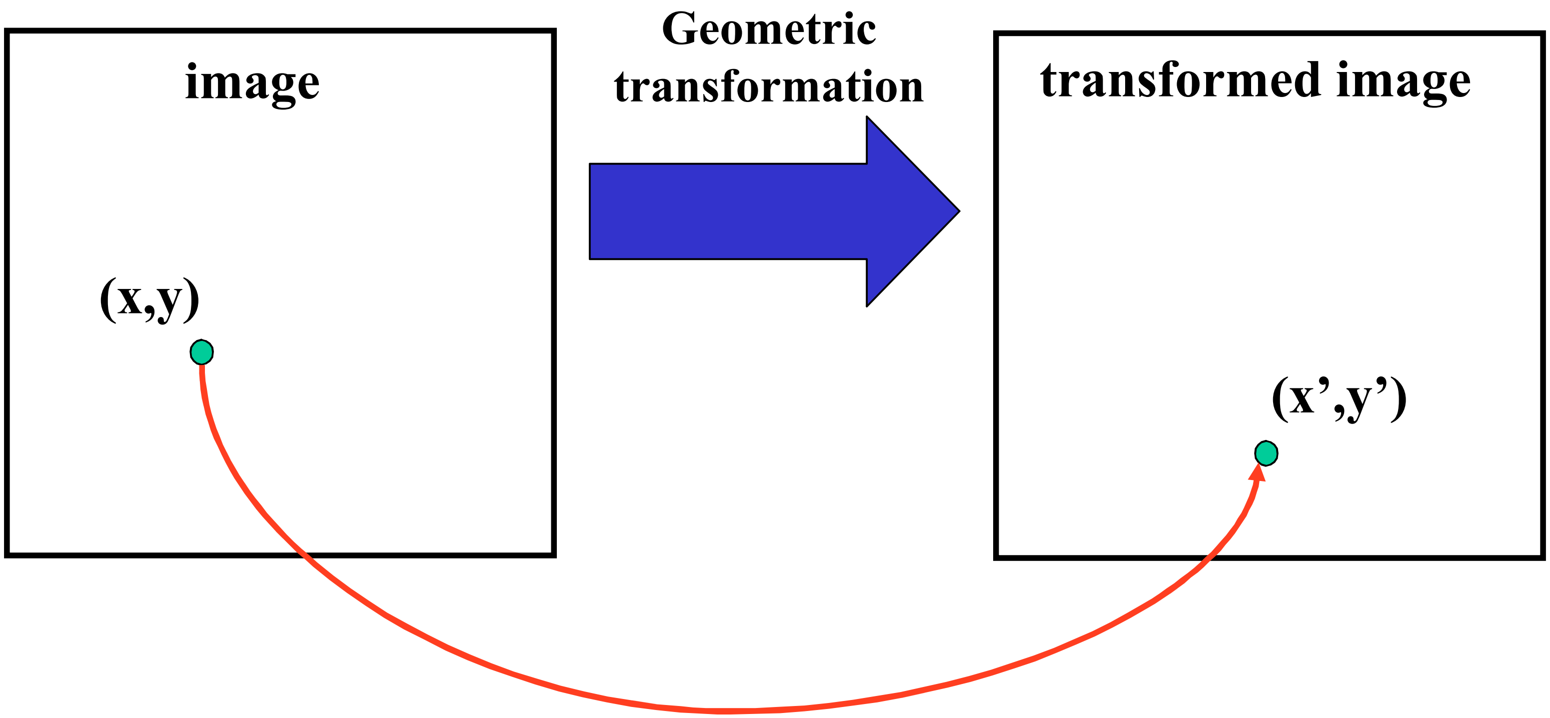
from R.Szeliski

Geometric Image Mappings



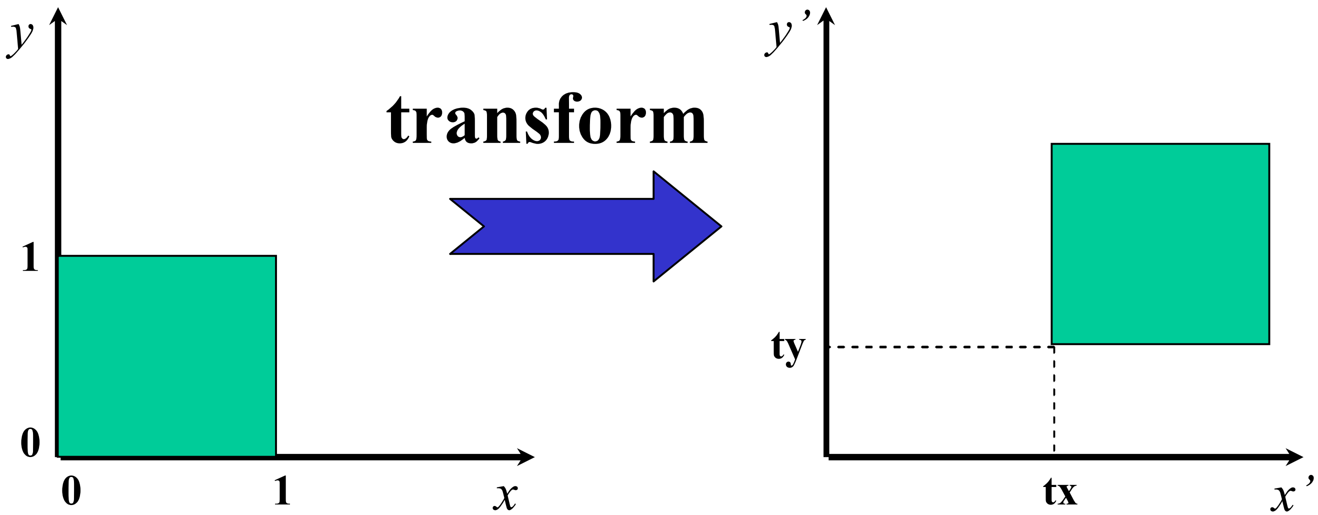
$$x' = f(x, y, \{parameters\})$$
$$y' = g(x, y, \{parameters\})$$

Linear Transformations
(Can be written as matrices)



$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{pmatrix} \\ M(params) \\ \\ \end{pmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

Translation

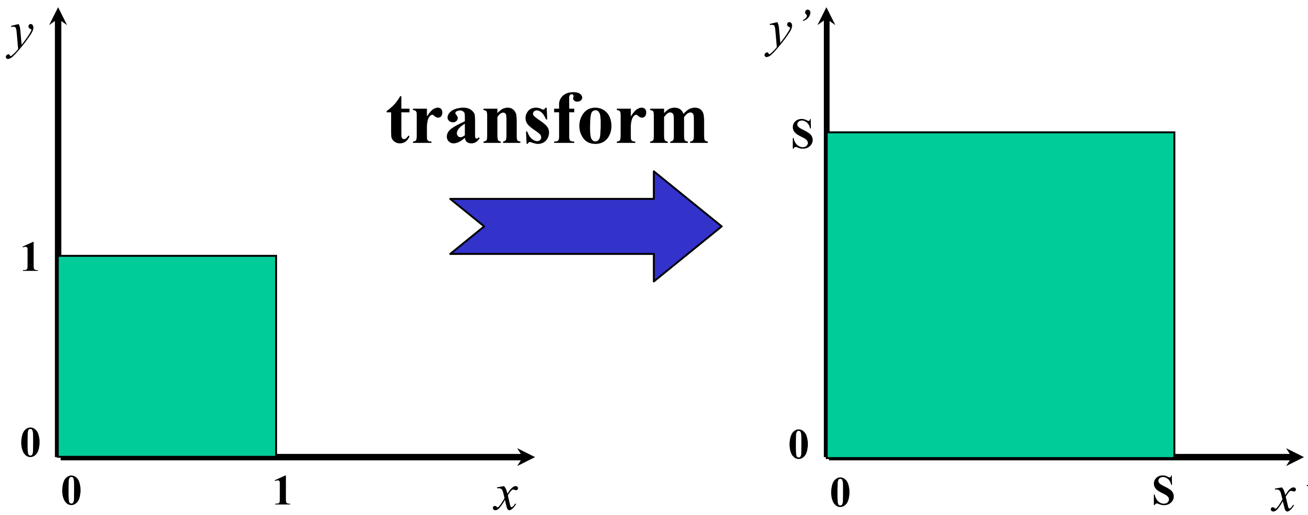


$$\begin{aligned} x' &= x + t_x \\ y' &= y + t_y \end{aligned}$$

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

equations **matrix form**

Scale



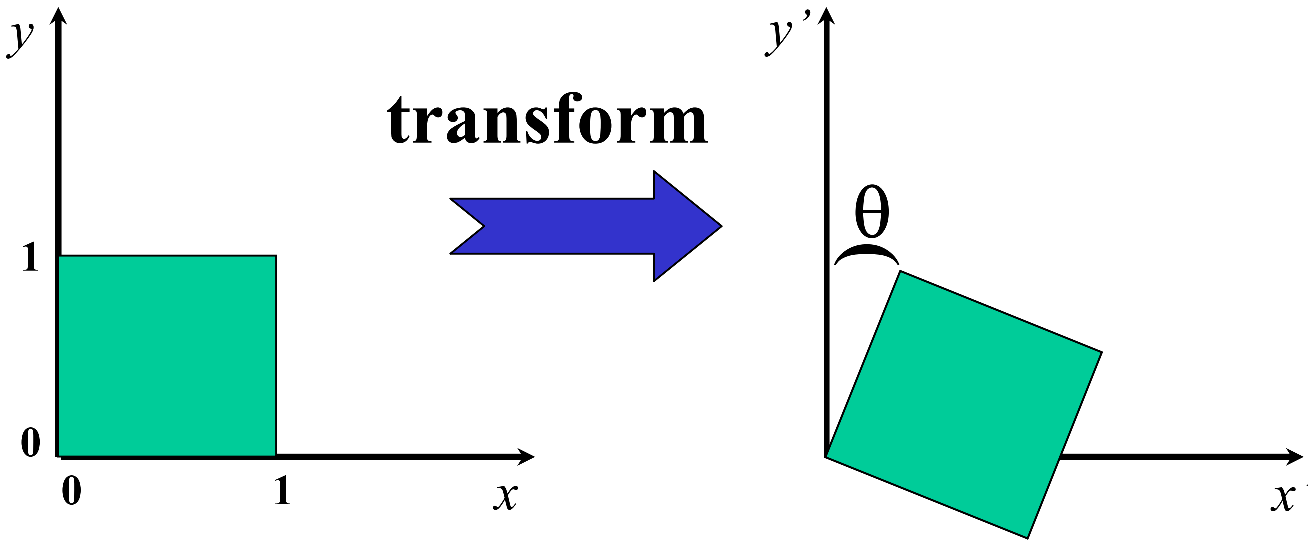
$$x' = s\,x_i$$
$$y' = s\,y_i$$

equations

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} s & 0 & 0 \\ 0 & s & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

matrix form

Rotation

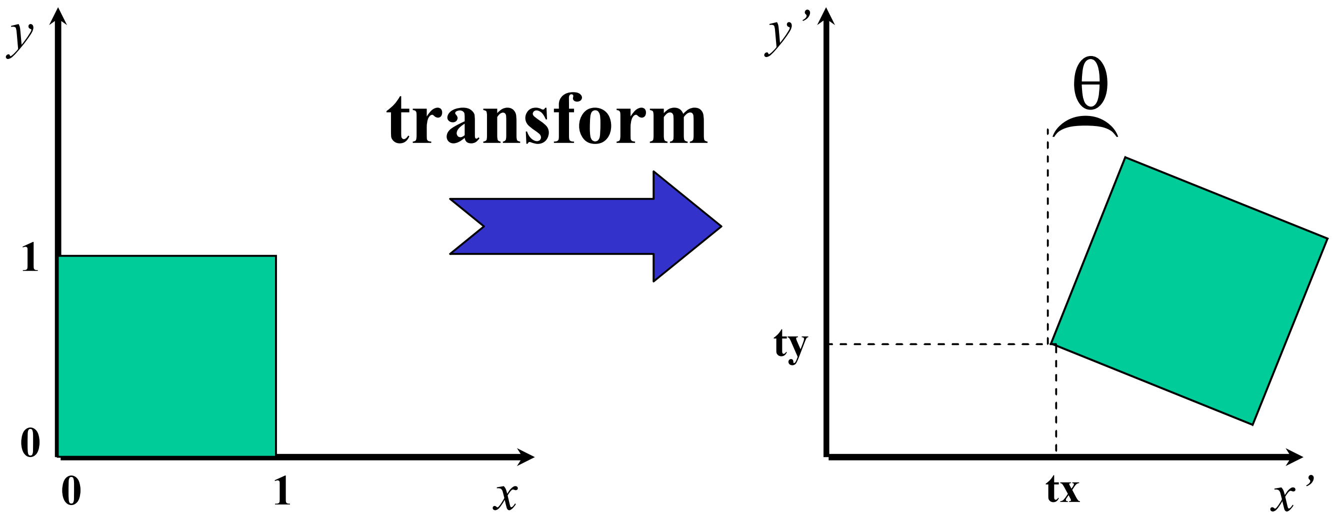


$$
\begin{aligned}
x' &= x_i \cos\theta - y_i \sin\theta \\
y' &= x_i \sin\theta + y_i \cos\theta
\end{aligned}
$$

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

equations **matrix form**

Euclidean (Rigid)



$$x' = x_i \cos\theta - y_i \sin\theta + t_x$$
$$y' = x_i \sin\theta + y_i \cos\theta + t_y$$

equations

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

matrix form

Partitioned Matrices

A *partitioned matrix*, or a *block matrix*, is a matrix M that has been constructed from other smaller matrices. These smaller matrices are called *blocks* or *sub-matrices* of M.

For instance, if we partition the below 5×5 matrix as follows

$$L = \left(\begin{array}{cc|ccc} 1 & 0 & 1 & 2 & 3 \\ 0 & 1 & 1 & 2 & 3 \\ \hline 2 & 3 & 9 & 9 & 9 \\ 2 & 3 & 9 & 9 & 9 \\ 2 & 3 & 9 & 9 & 9 \end{array}\right),$$

then we can define the matrices

$$A = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, B = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}, C = \begin{pmatrix} 2 & 3 \\ 2 & 3 \\ 2 & 3 \end{pmatrix}, D = \begin{pmatrix} 9 & 9 & 9 \\ 9 & 9 & 9 \\ 9 & 9 & 9 \end{pmatrix}$$

and write L as

$$L = \begin{pmatrix} A & B \\ C & D \end{pmatrix}, \quad \text{or} \quad L = \left(\begin{array}{c|c} A & B \\ \hline C & D \end{array}\right).$$

http://planetmath.org/encyclopedia/PartitionedMatrix.html

Partitioned Matrices

$$\begin{bmatrix} x' \\ y' \\ \hline 1 \end{bmatrix} = \left[\begin{array}{cc|c} \cos\theta & -\sin\theta & t_x \\ \sin\theta & \cos\theta & t_y \\ \hline 0 & 0 & 1 \end{array}\right] \begin{bmatrix} x \\ y \\ \hline 1 \end{bmatrix}$$

$$\begin{bmatrix} \overset{2\mathrm{x}1}{p'} \\ \overset{1\mathrm{x}1}{1} \end{bmatrix} = \begin{bmatrix} \overset{2\mathrm{x}2}{R} & \overset{2\mathrm{x}1}{t} \\ \overset{1\mathrm{x}2}{0} & \overset{1\mathrm{x}1}{1} \end{bmatrix} \begin{bmatrix} \overset{2\mathrm{x}1}{p} \\ \overset{1\mathrm{x}1}{1} \end{bmatrix}$$

matrix form

$$p' = Rp + t$$

equation form

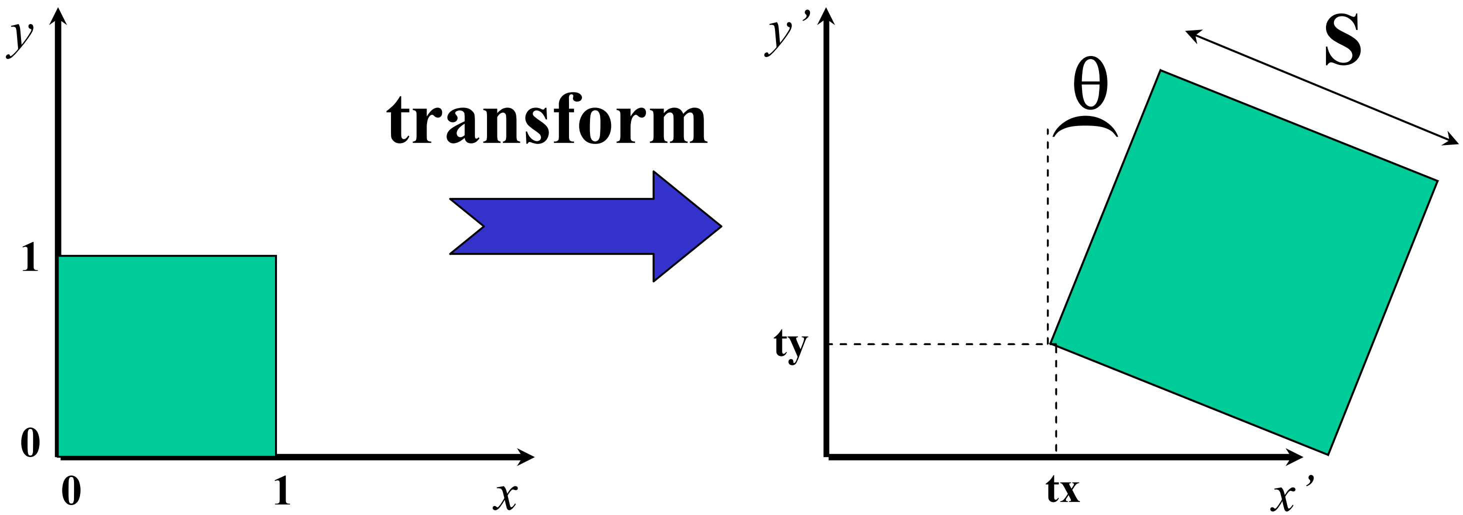
Another Example (from last time)

$$\begin{pmatrix} X \\ Y \\ Z \\ \hline 1 \end{pmatrix} = \left(\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \\ \hline 0 & 0 & 0 & 1 \end{array}\right) \begin{pmatrix} U \\ V \\ W \\ \hline 1 \end{pmatrix}$$

$$\begin{pmatrix} \overset{3x1}{\mathbf{P_C}} \\ \overset{1x1}{1} \end{pmatrix} = \begin{pmatrix} \overset{3x3}{\mathbf{R}} & \overset{3x1}{\mathbf{T}} \\ \overset{1x3}{0} & \overset{1x1}{1} \end{pmatrix} \begin{pmatrix} \overset{3x1}{\mathbf{P_W}} \\ \overset{1x1}{1} \end{pmatrix}$$

$$\mathbf{P_C = R\, P_W + T}$$

Similarity (scaled Euclidean)



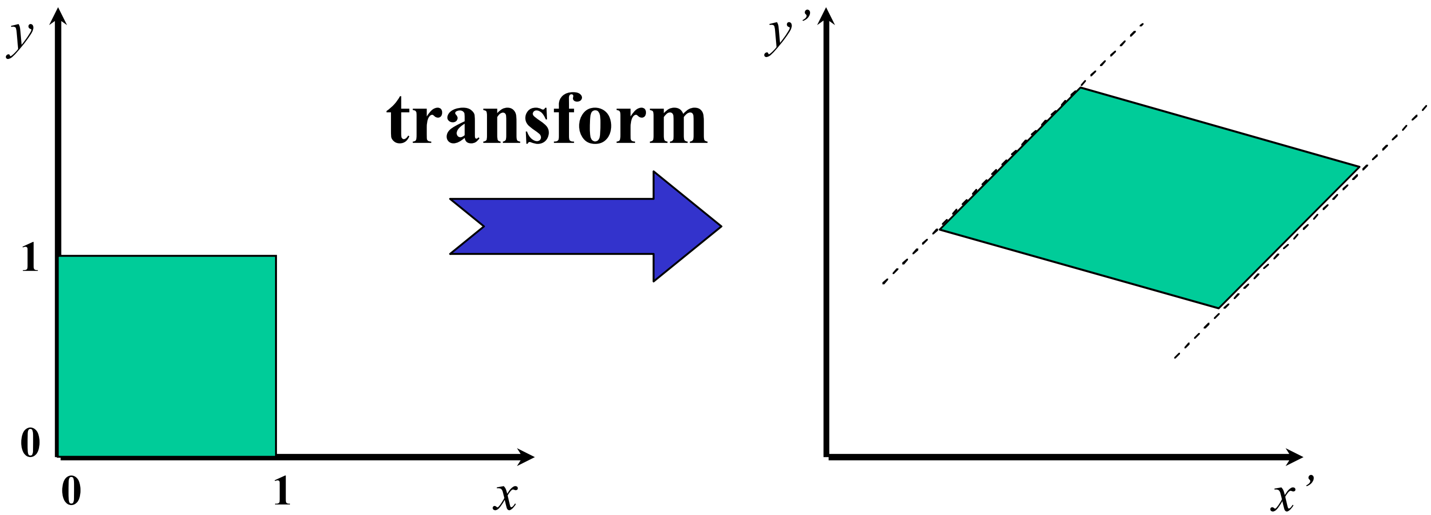
$$p' = sRp + t$$

equations

$$\begin{bmatrix} p' \\ 1 \end{bmatrix} = \begin{bmatrix} sR & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ 1 \end{bmatrix}$$

matrix form

Affine

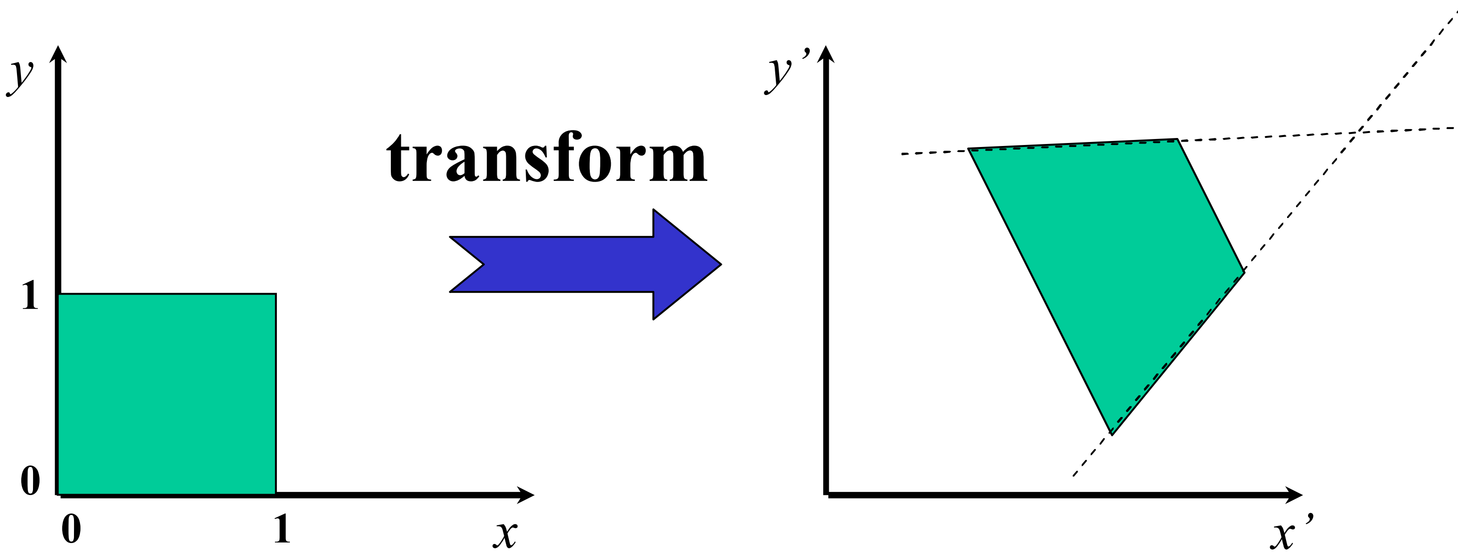


$$p' = Ap + b$$

$$\begin{bmatrix} p' \\ 1 \end{bmatrix} = \begin{bmatrix} A & b \\ 0 & 1 \end{bmatrix} \begin{bmatrix} p \\ 1 \end{bmatrix}$$

equations **matrix form**

Projective



Note!

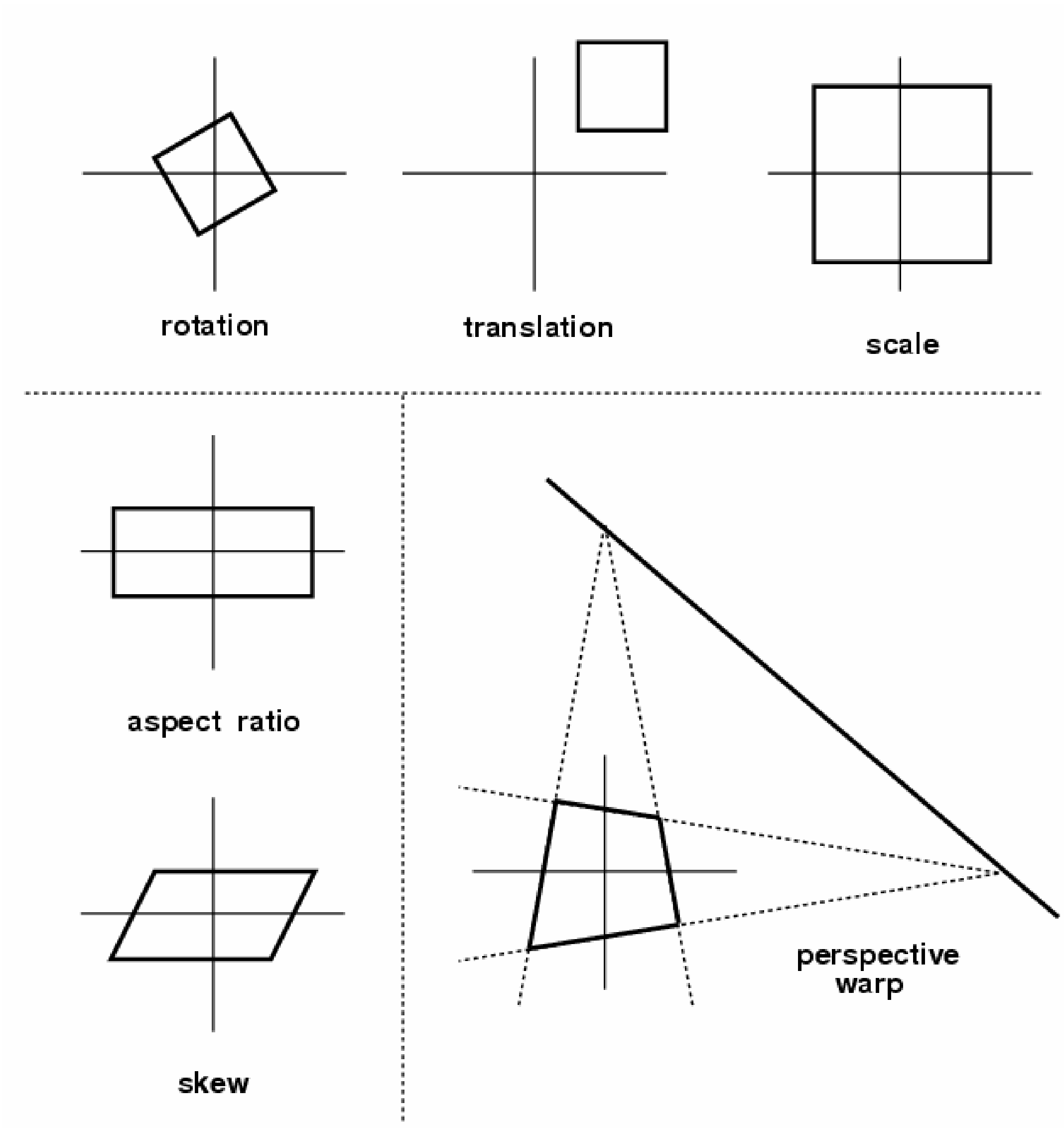
$$p' = \frac{Ap + b}{c^T p + 1}$$

equations

$$\begin{bmatrix} p' \\ 1 \end{bmatrix} \sim \begin{bmatrix} A & b \\ c^T & 1 \end{bmatrix} \begin{bmatrix} p \\ 1 \end{bmatrix}$$

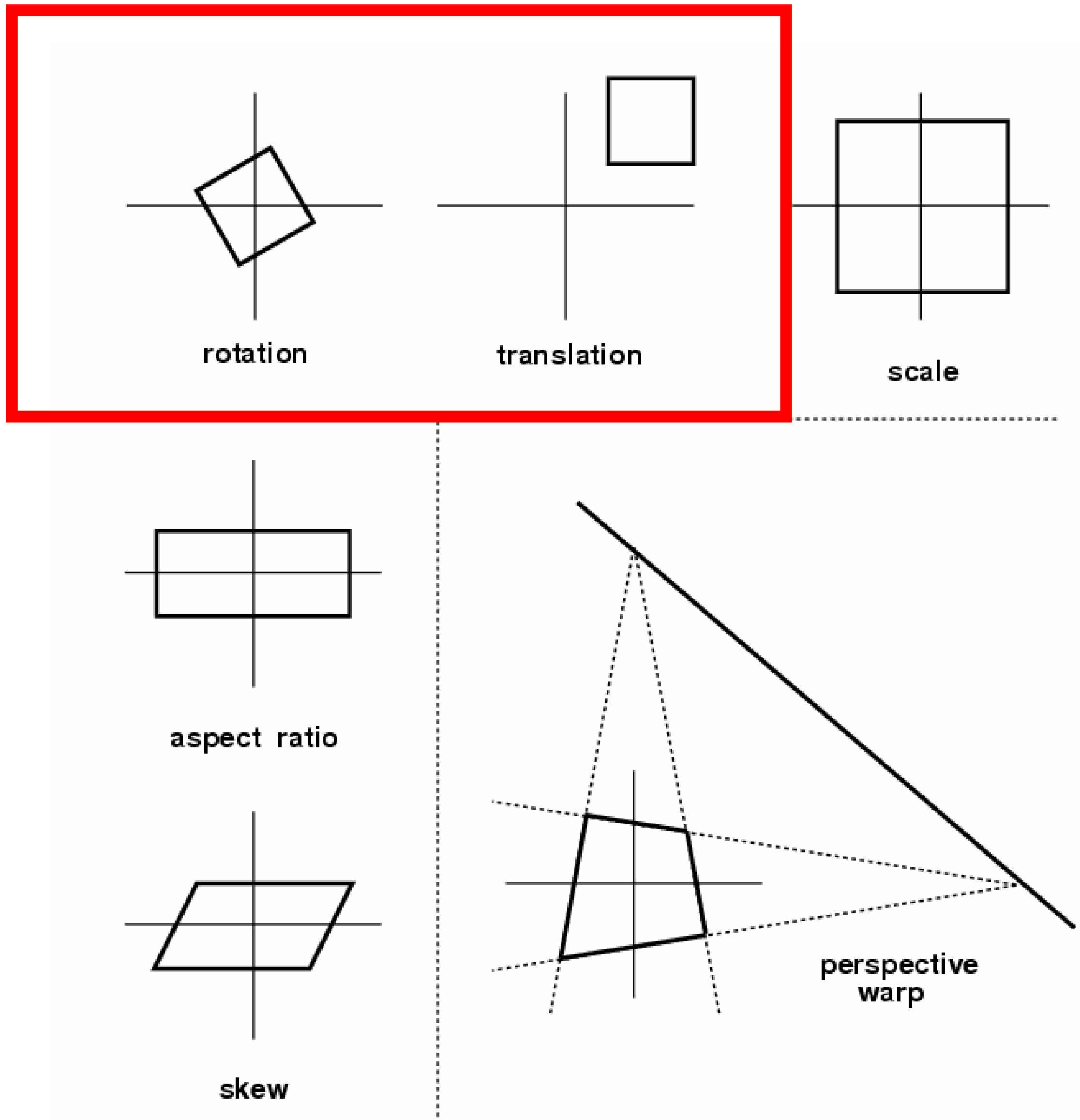
matrix form

Summary of 2D Transformations



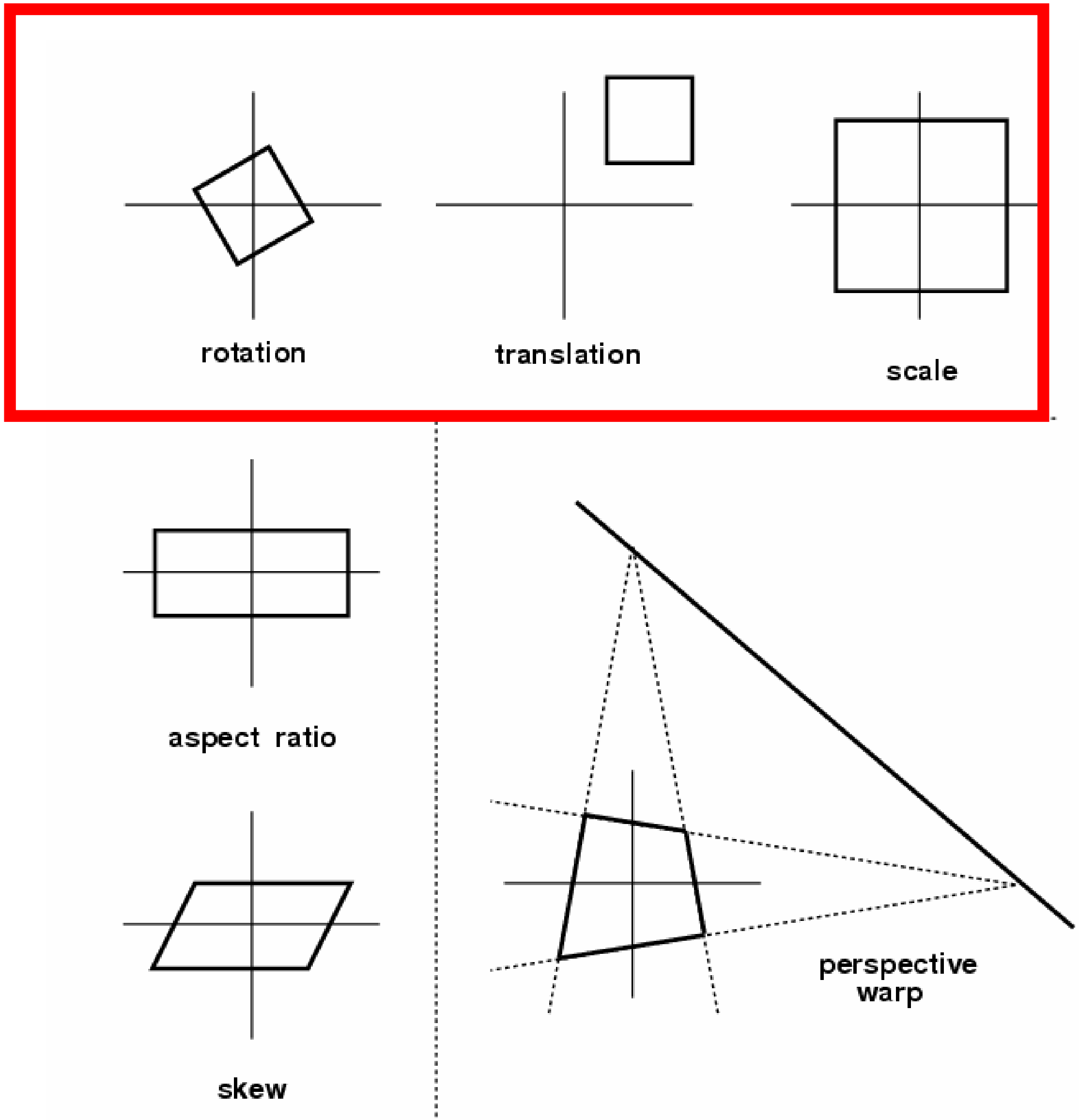
Summary of 2D Transformations

Euclidean



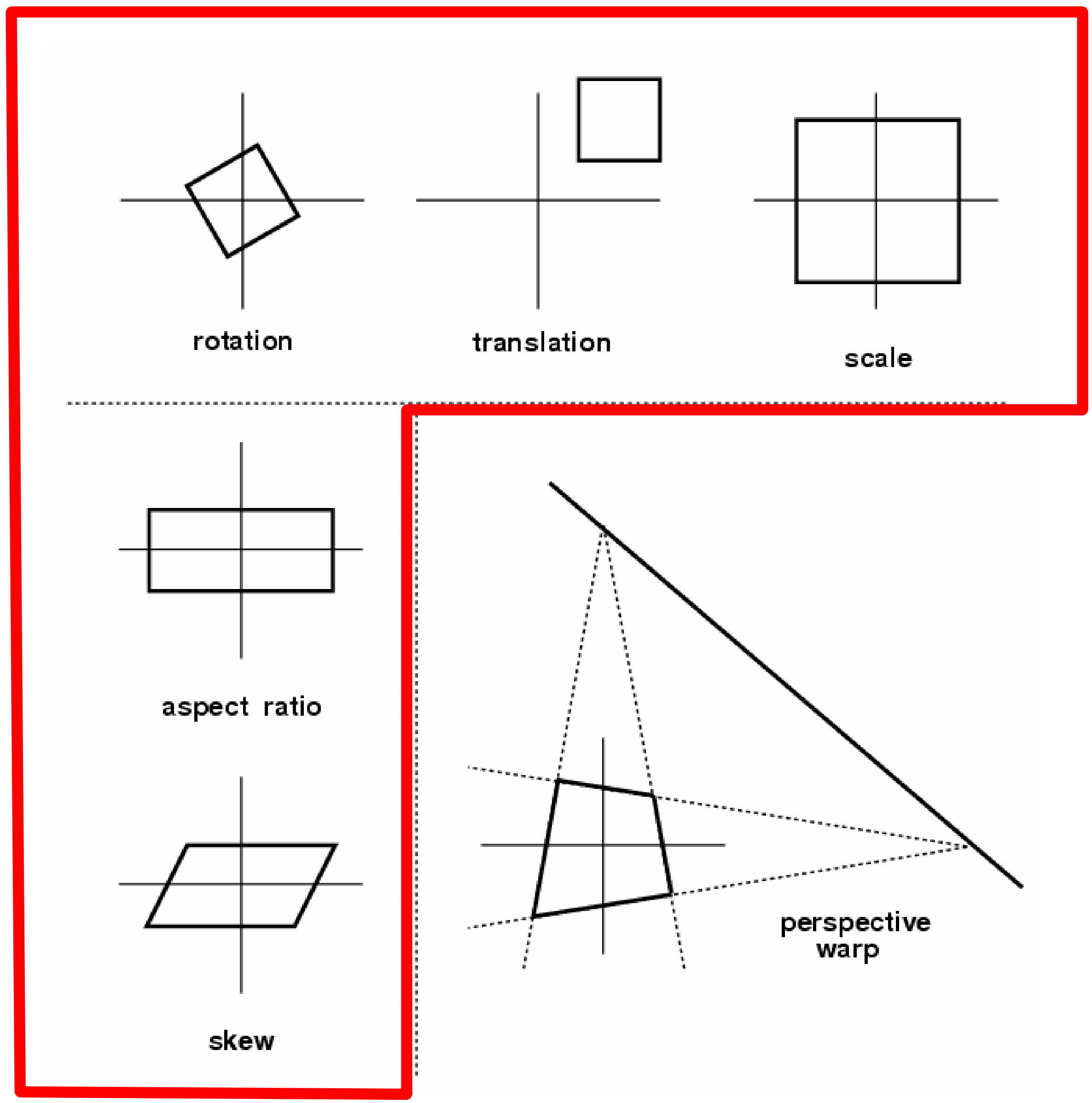
Summary of 2D Transformations

Similarity



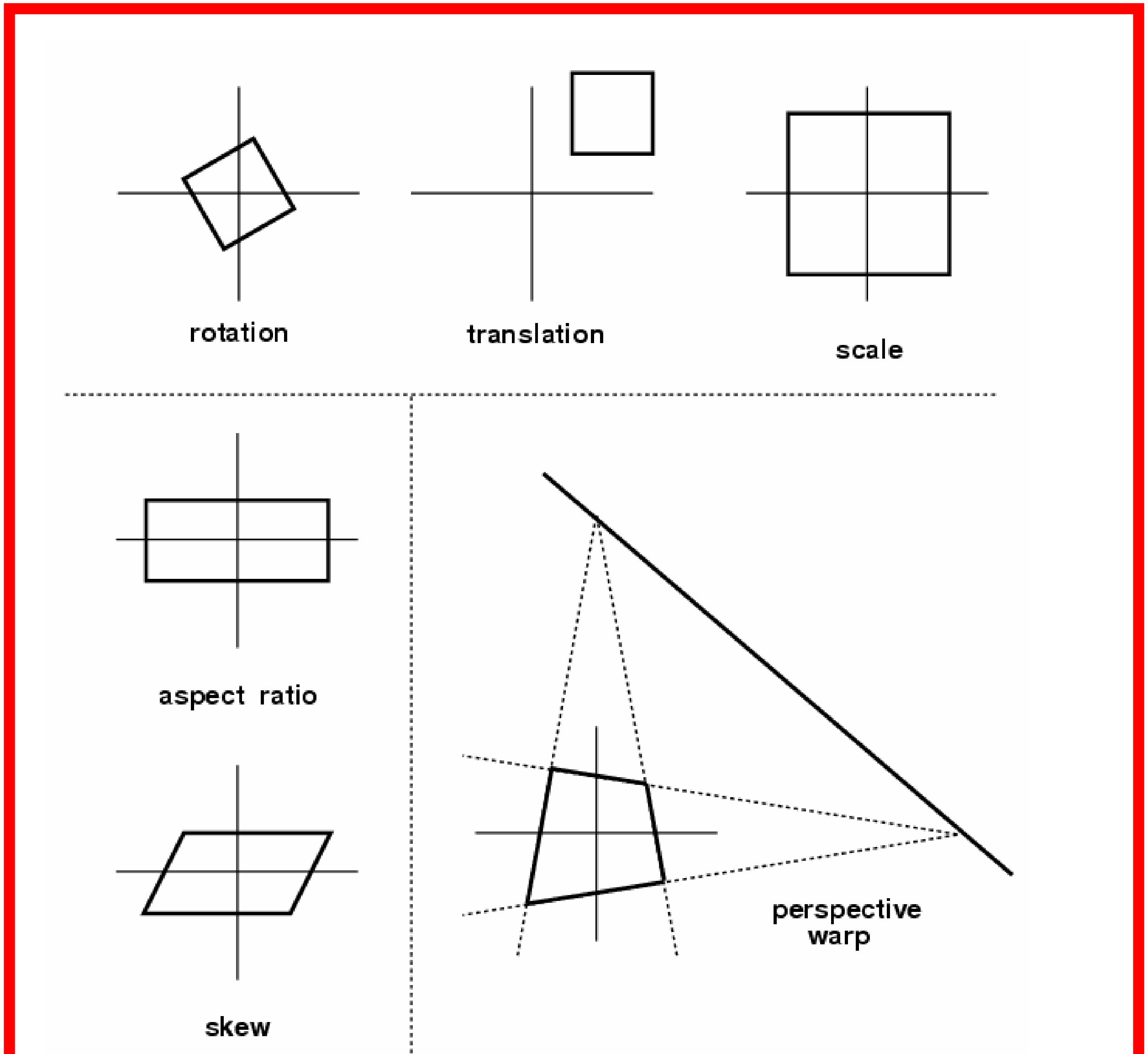
Summary of 2D Transformations

Affine

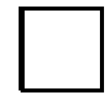
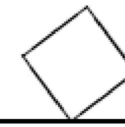
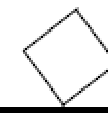
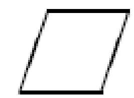
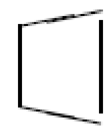


Summary of 2D Transformations

Projective



Summary of 2D Transformations

Name	Matrix	# D.O.F.	Preserves:	Icon
translation	$\begin{bmatrix} \boldsymbol{I} & \big| & \boldsymbol{t} \end{bmatrix}_{2\times 3}$	2	orientation $+ \cdots$	
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} & \big| & \boldsymbol{t} \end{bmatrix}_{2\times 3}$	3	lengths $+ \cdots$	
similarity	$\begin{bmatrix} s\boldsymbol{R} & \big| & \boldsymbol{t} \end{bmatrix}_{2\times 3}$	4	angles $+ \cdots$	
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}_{2\times 3}$	6	parallelism $+ \cdots$	
projective	$\begin{bmatrix} \boldsymbol{H} \end{bmatrix}_{3\times 3}$	8	straight lines	

from R.Szeliski