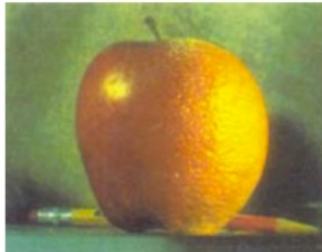


2. Image Formation



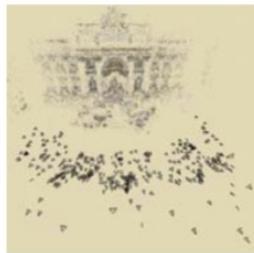
3. Image Processing



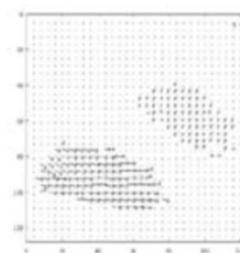
4. Features



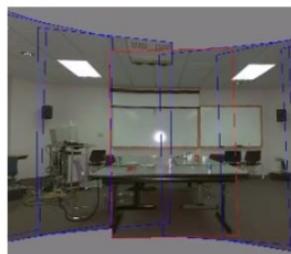
5. Segmentation



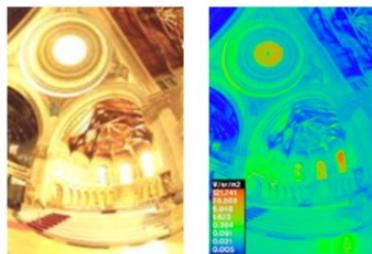
6-7. Structure from Motion



8. Motion



9. Stitching



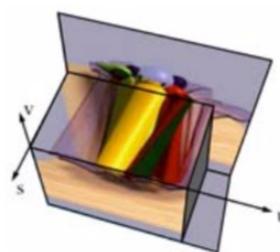
10. Computational Photography



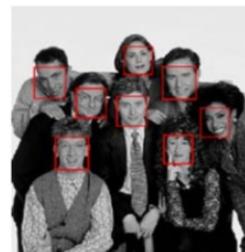
11. Stereo



12. 3D Shape



13. Image-based Rendering



14. Recognition

Image Formation

2.1	Geometric primitives and transformations	31
2.1.1	Geometric primitives	32
2.1.2	2D transformations	35
2.1.3	3D transformations	39
2.1.4	3D rotations	41
2.1.5	3D to 2D projections	46
2.1.6	Lens distortions	58
2.2	Photometric image formation	60
2.2.1	Lighting	60
2.2.2	Reflectance and shading	62
2.2.3	Optics	68
2.3	The digital camera	73
2.3.1	Sampling and aliasing	77
2.3.2	Color	80
2.3.3	Compression	90
2.4	Additional reading	93
2.5	Exercises	93

Image Formation

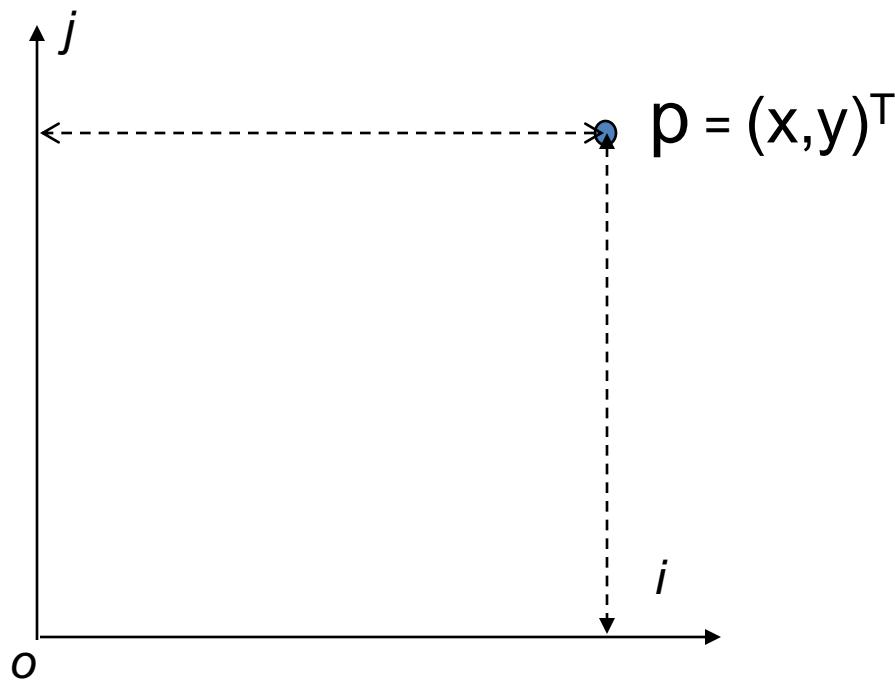
2.1	Geometric primitives and transformations	31
2.1.1	Geometric primitives	32
2.1.2	2D transformations	35
2.1.3	3D transformations	39
2.1.4	3D rotations	41
2.1.5	3D to 2D projections	46
2.1.6	Lens distortions	58
2.2	Photometric image formation	60
2.2.1	Lighting	60
2.2.2	Reflectance and shading	62
2.2.3	Optics	68
2.3	The digital camera	73
2.3.1	Sampling and aliasing	77
2.3.2	Color	80
2.3.3	Compression	90
2.4	Additional reading	93
2.5	Exercises	93

2.1.1 Geometric Primitives

- 2D points:
- 2D lines:
- 2D conics:
- 3D points:
- 3D planes:
- 3D lines:

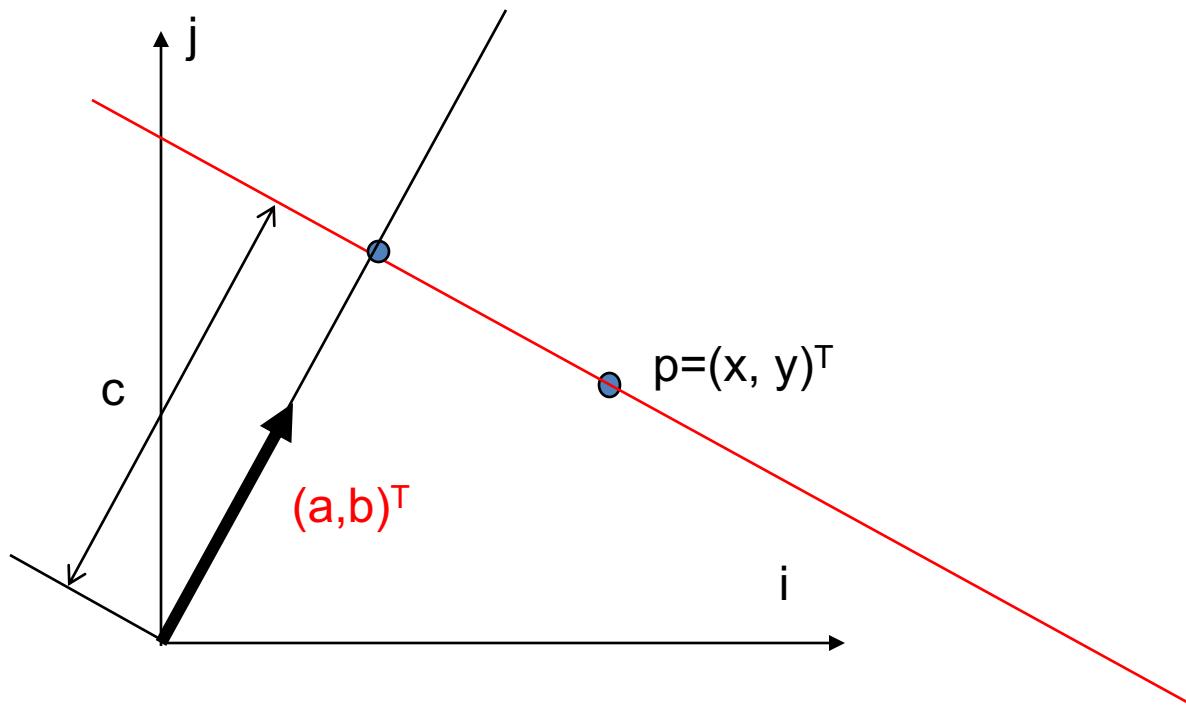
2D Coordinate Frames & Points

- 2D coordinates x and y
- Point $p = (x, y)^T$



2D Lines

- Line $l = (a, b, c)$
- Point p coincides with line iff $ax + by = c$



Homogeneous coordinates

Conversion

Converting to *homogeneous* coordinates

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

homogeneous image
coordinates

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

homogeneous scene
coordinates

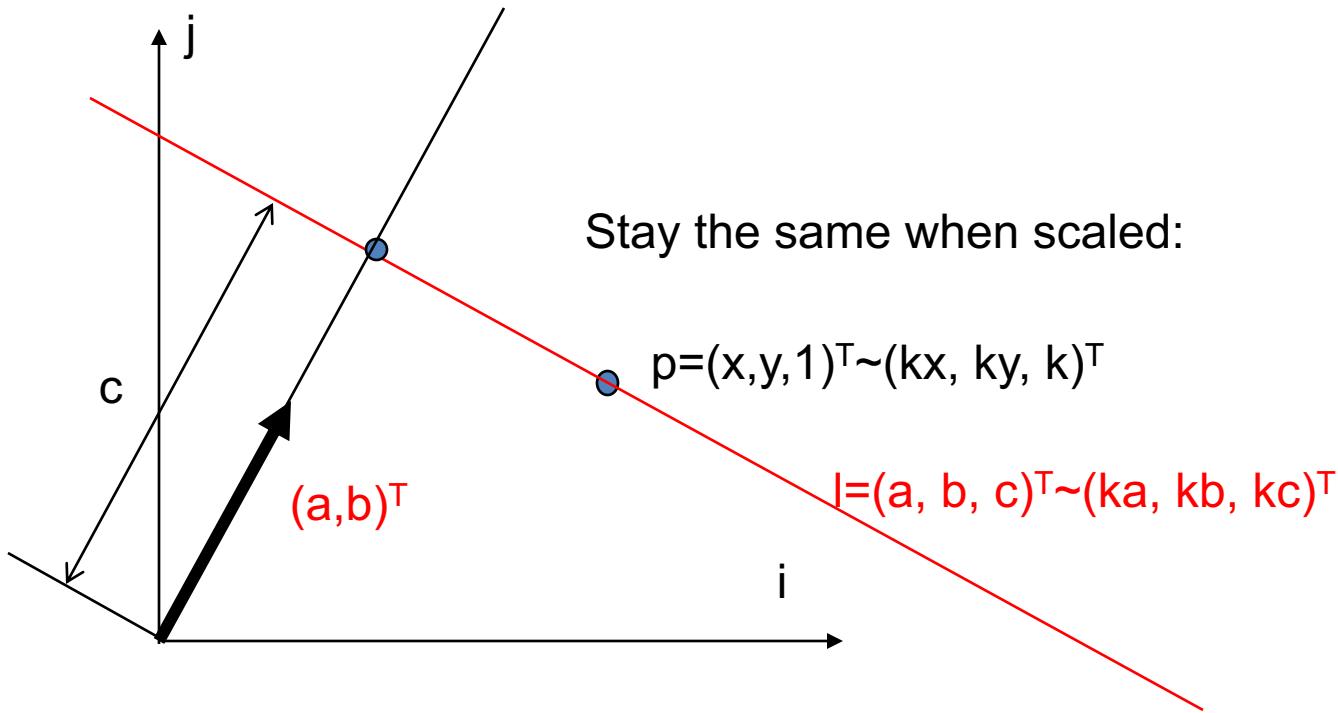
Converting *from* homogeneous coordinates

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

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

Homogeneous Coordinates

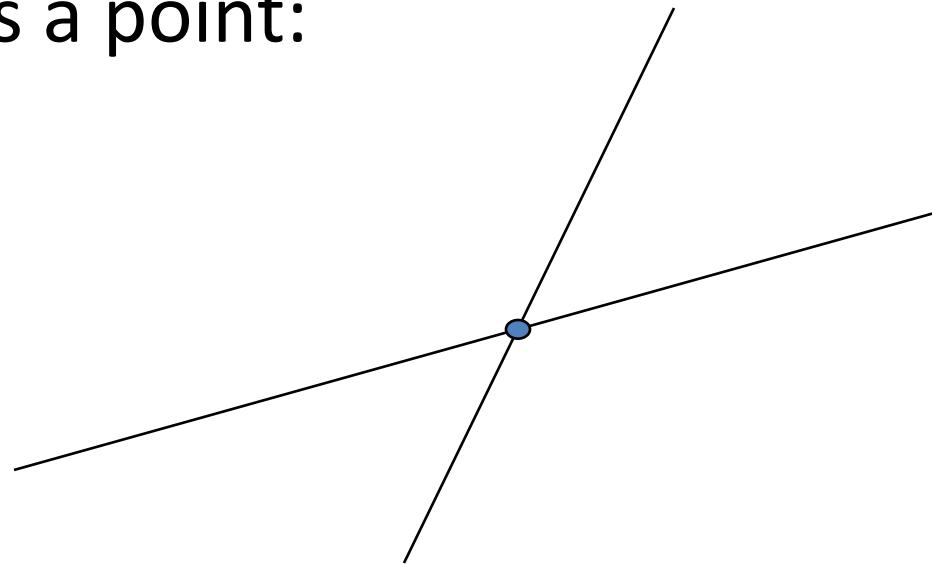
- Uniform treatment of points and lines
- Line-point incidence: $\mathbf{l}^T \mathbf{p} = 0$



Join = cross product !

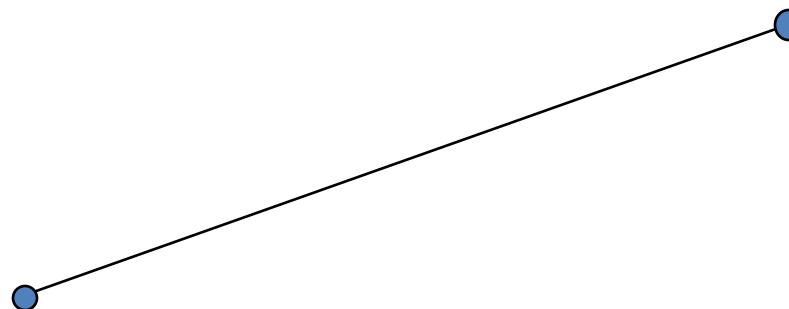
- Join of two lines is a point:

$$p = l_1 \times l_2$$



- Join of two points is a line:

$$l = p_1 \times p_2$$



Automatic estimation of vanishing points and lines

$$v = l_1 \times l_2$$

The diagram illustrates the concept of vanishing points and lines. It features two orange lines, labeled l_1 and l_2 , which intersect at a point labeled V . A cyan line segment connects the endpoints of l_1 and l_2 to the point V .



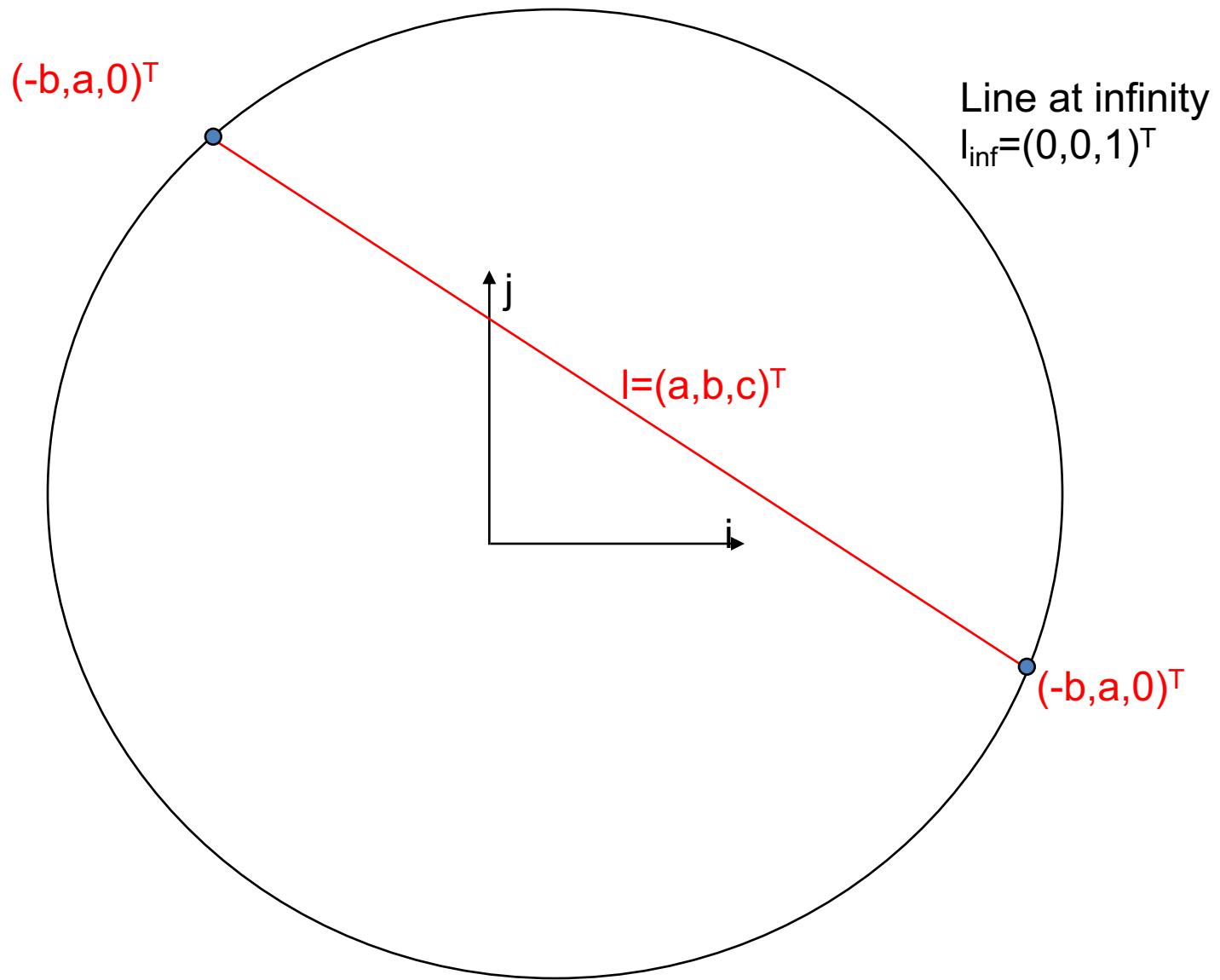
Joining two parallel lines ?

(a,b,c)

$$p = \begin{vmatrix} i & j & k \\ a & b & c \\ a & b & d \end{vmatrix} = \begin{bmatrix} bd - cb \\ ca - ad \\ 0 \end{bmatrix}$$

The diagram shows two parallel lines in 3D space. The top line is labeled (a,b,c) and the bottom line is labeled (a,b,d). The lines are represented by two parallel lines in perspective, pointing upwards and to the right.

Points at Infinity !



2.1.1 Geometric Primitives

homogeneous

- 2D points: (x,y) , $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{\mathbf{x}}$
- 2D lines: $\bar{\mathbf{x}} \cdot \tilde{\mathbf{l}} = ax + by + c = 0$
- 2D conics:
- 3D points:
- 3D planes:
- 3D lines:

2.1.1 Geometric Primitives

- | | | |
|--|-------------|-----------|
| | homogeneous | augmented |
|--|-------------|-----------|
- 2D points: (x, y) , $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{\mathbf{x}}$
 - 2D lines: $\bar{\mathbf{x}} \cdot \tilde{\mathbf{l}} = ax + by + c = 0$
 - 2D conics:
 - 3D points: $\mathbf{x} = (x, y, z)$ $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w})$
 - 3D planes: $\bar{\mathbf{x}} \cdot \tilde{\mathbf{m}} = ax + by + cz + d = 0$
 - 3D lines:

2.1.1 Geometric Primitives

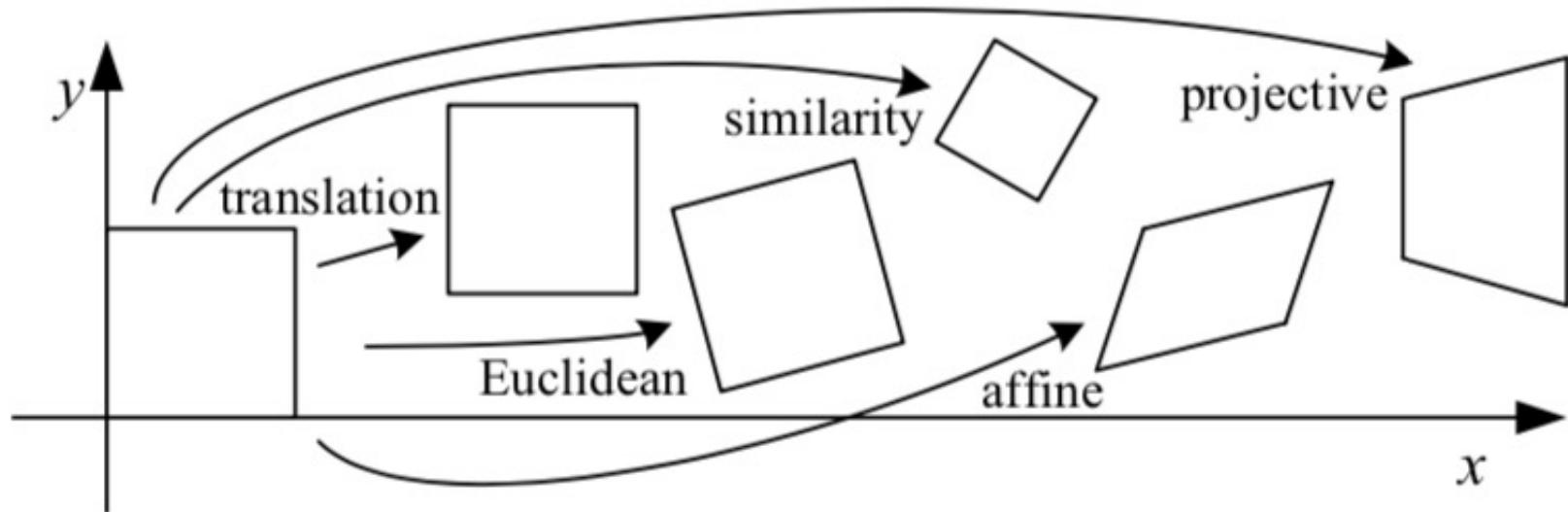
- | | | |
|--|-------------|-----------|
| | homogeneous | augmented |
|--|-------------|-----------|
- 2D points: (x, y) , $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{\mathbf{x}}$
 - 2D lines: $\bar{\mathbf{x}} \cdot \tilde{\mathbf{l}} = ax + by + c = 0$
 - 2D conics: $\tilde{\mathbf{x}}^T Q \tilde{\mathbf{x}} = 0$
 - 3D points: $\mathbf{x} = (x, y, z)$ $\tilde{\mathbf{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w})$
 - 3D planes: $\bar{\mathbf{x}} \cdot \tilde{\mathbf{m}} = ax + by + cz + d = 0$
 - 3D lines: $\mathbf{r} = (1 - \lambda)\mathbf{p} + \lambda\mathbf{q}$

$$\tilde{\mathbf{r}} = \mu\tilde{\mathbf{p}} + \lambda\tilde{\mathbf{q}}$$

$$\mathbf{r} = \mathbf{p} + \lambda\hat{\mathbf{d}}$$

See Chapter 2.1.1 for
conics, quadrics, 3D lines

2.1.2: 2D Transformations



2.1.2: 2D Transformations



translation



rotation



aspect



affine

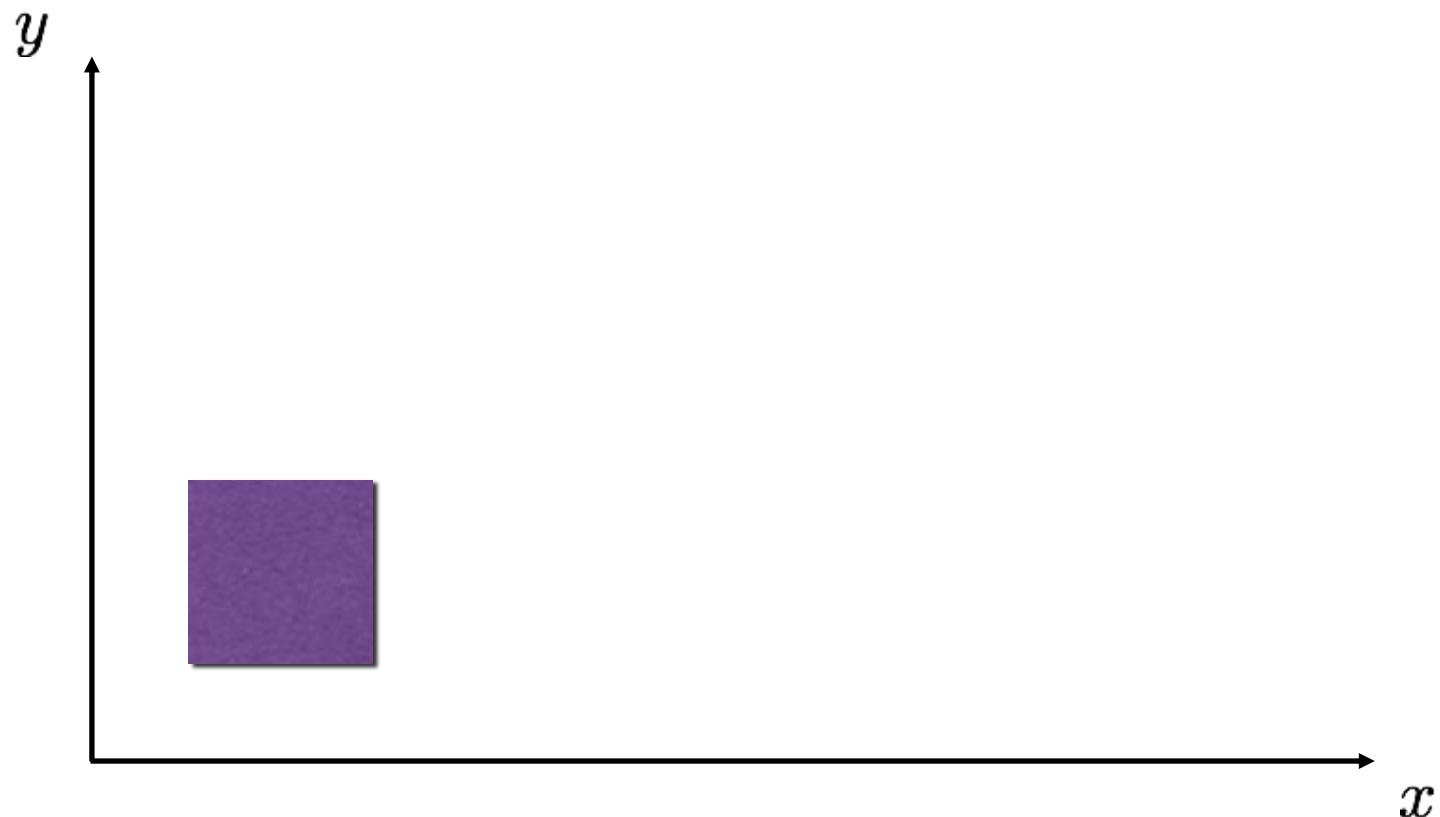


perspective

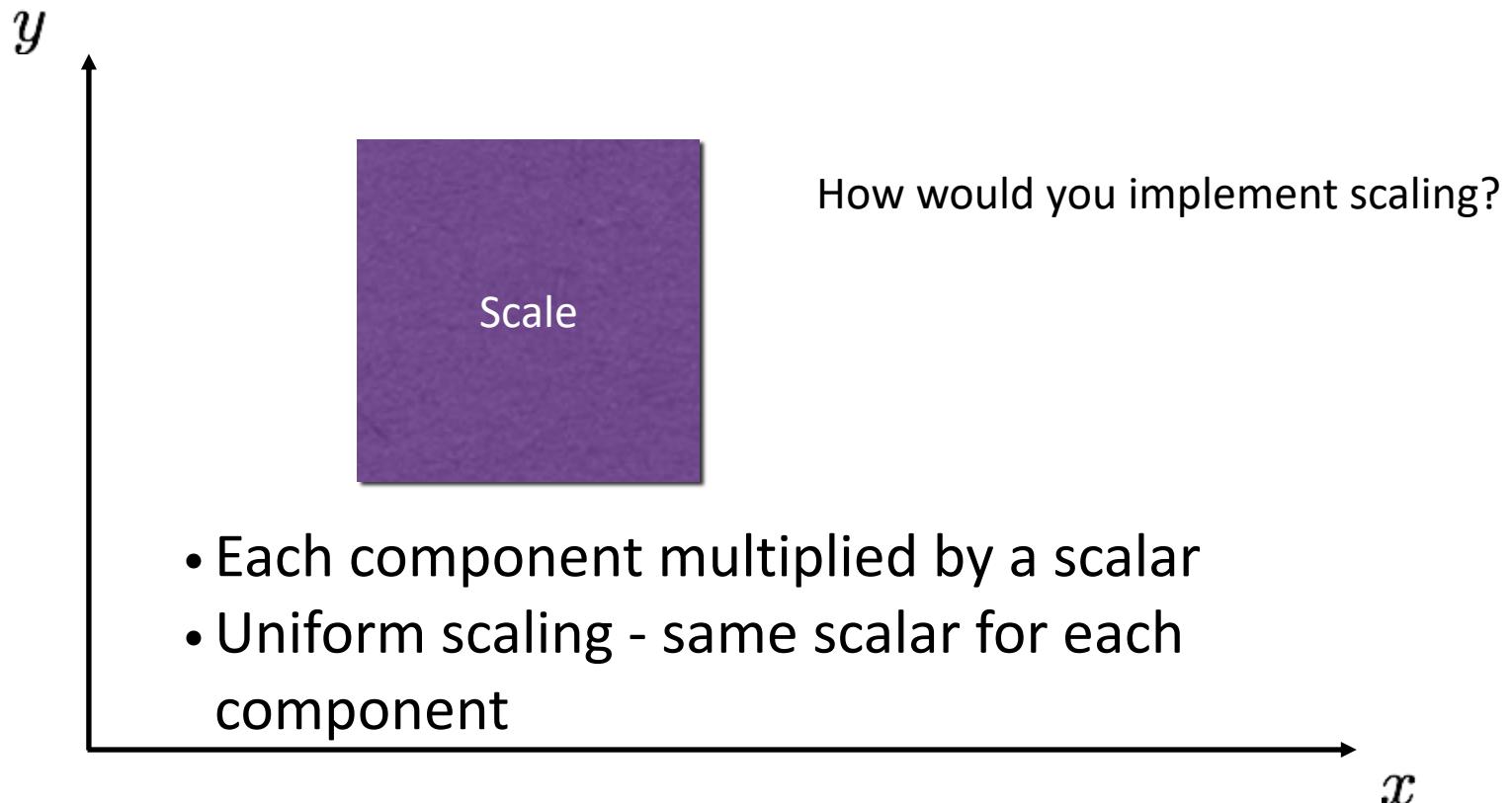


cylindrical

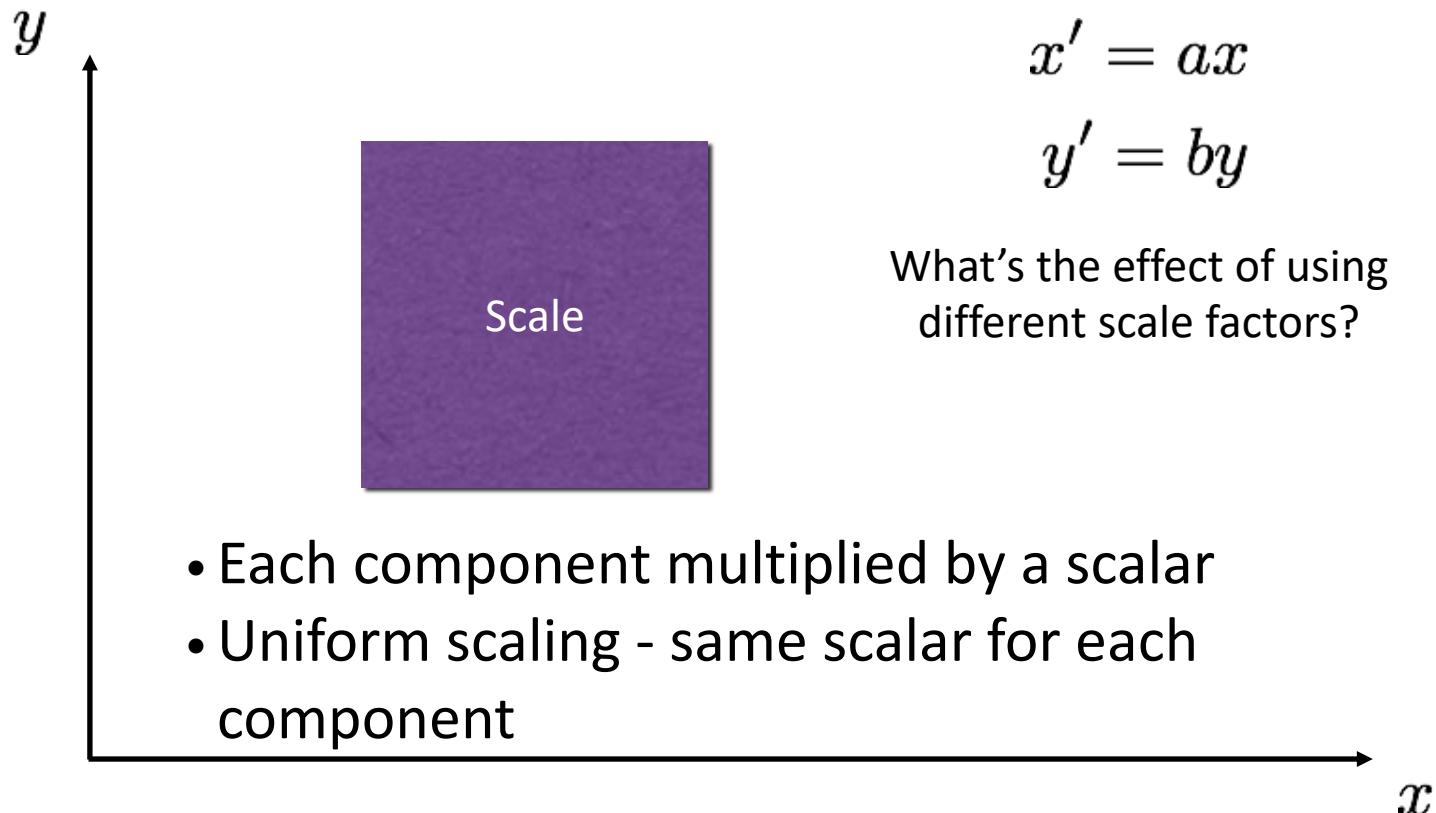
2D planar transformations



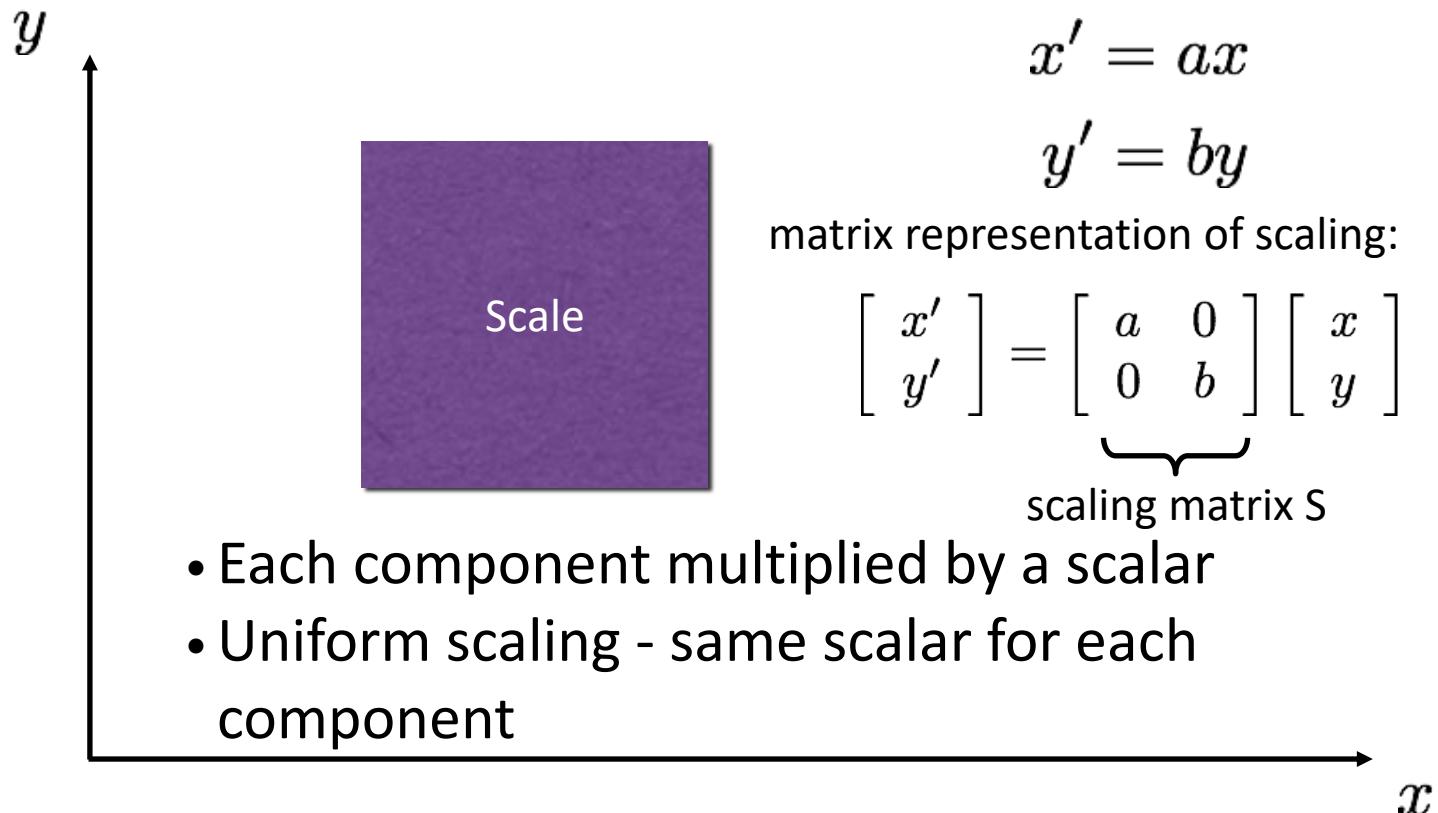
2D planar transformations



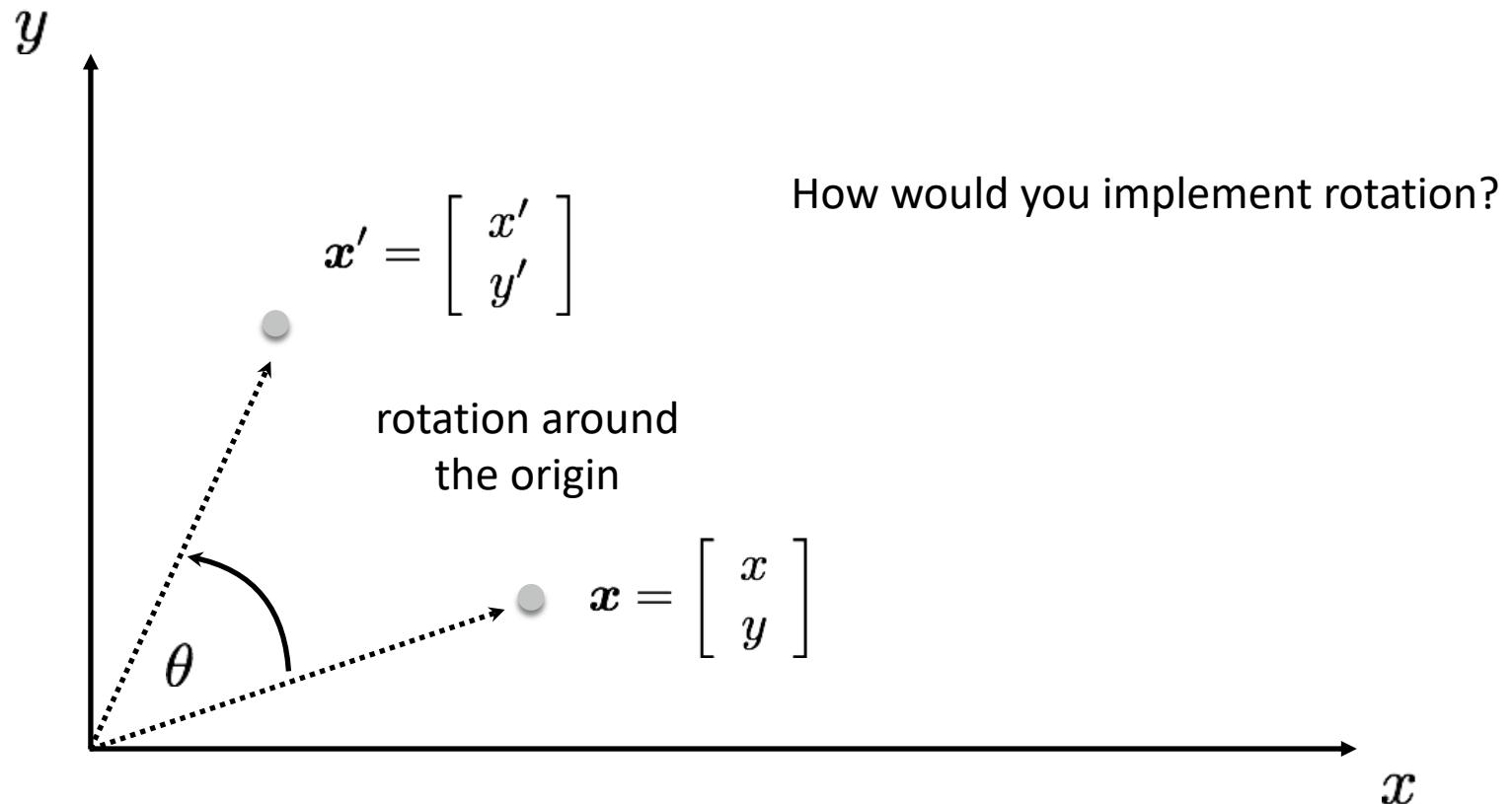
2D planar transformations



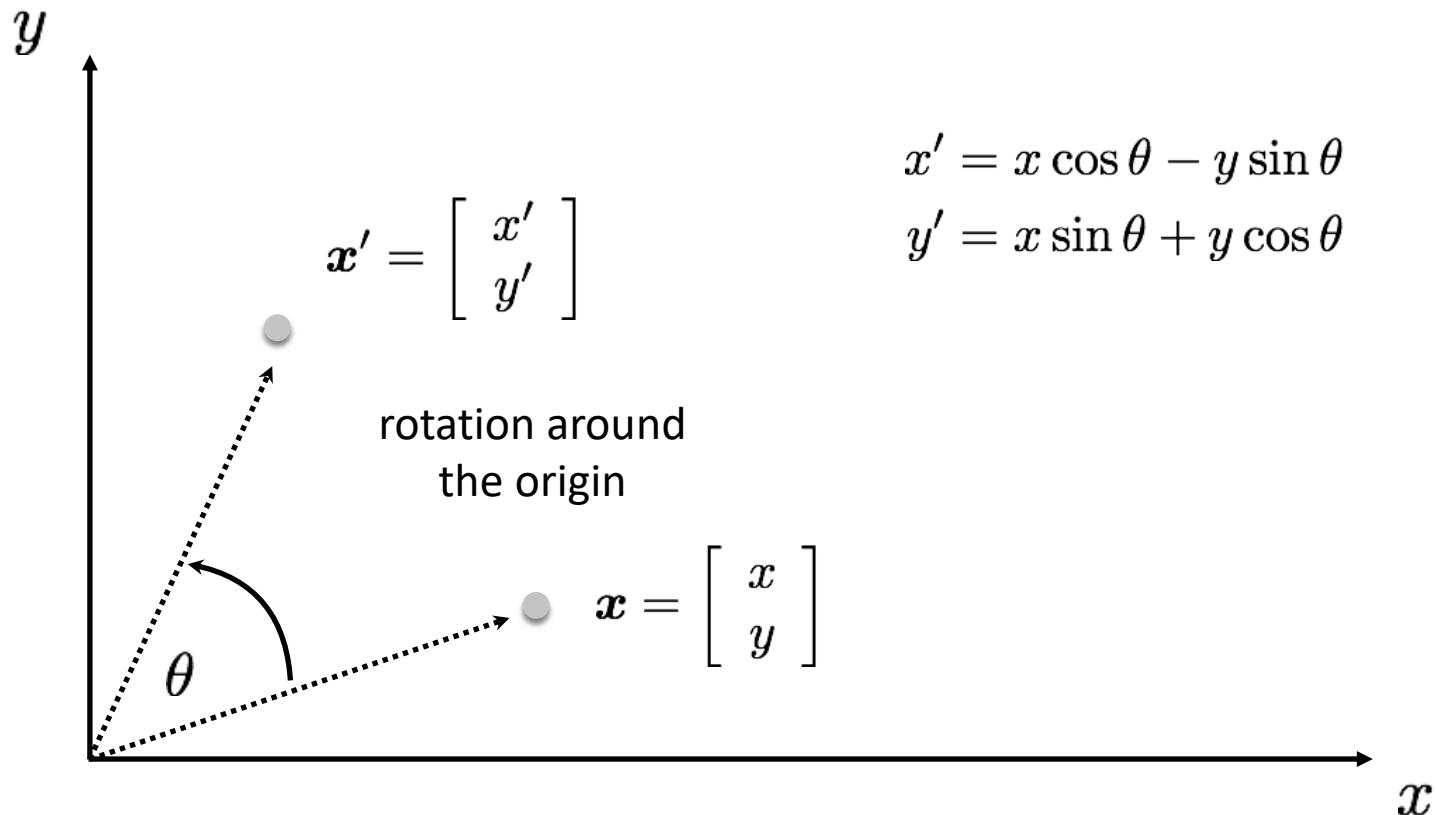
2D planar transformations



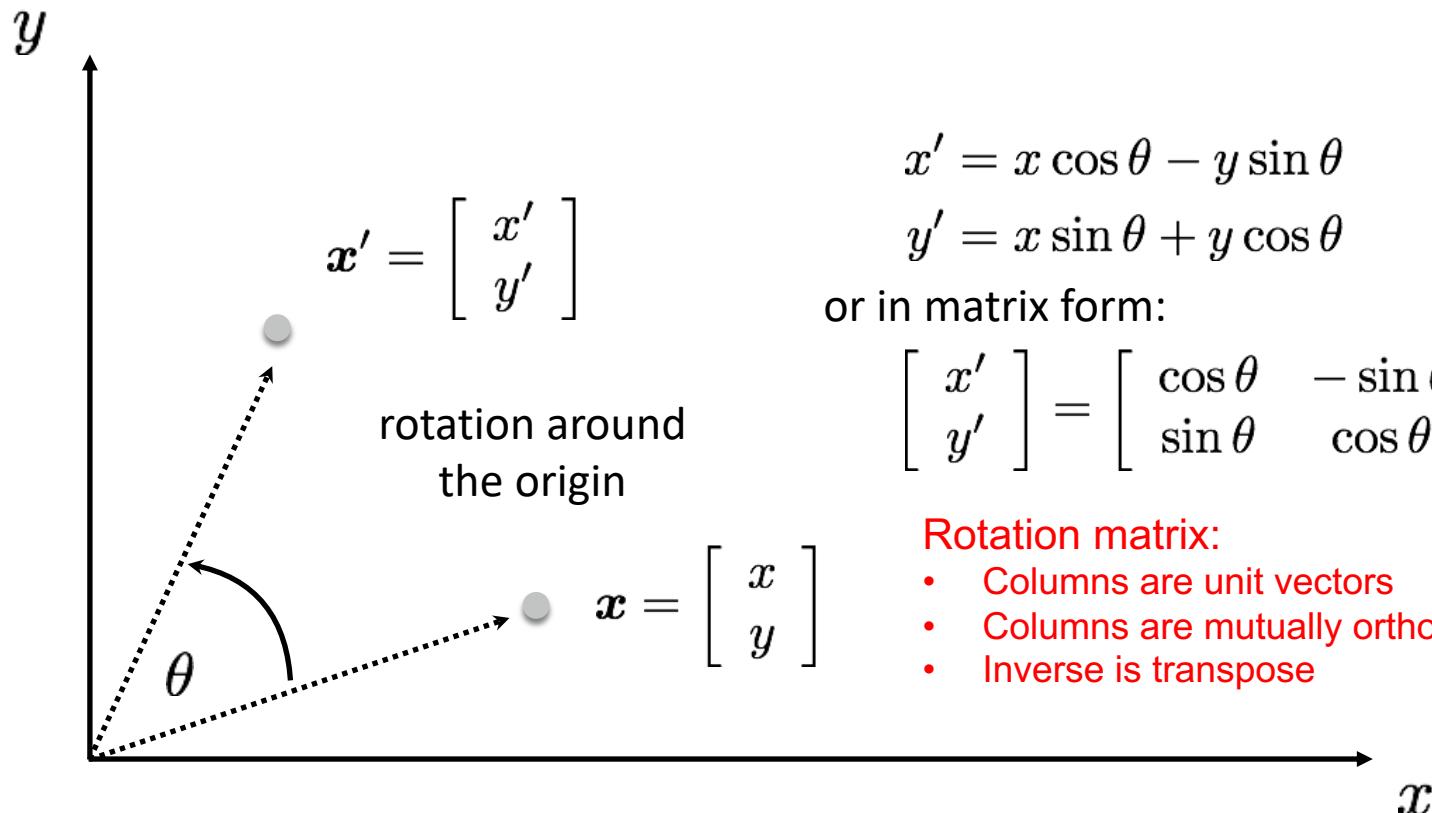
2D planar transformations



2D planar transformations



2D planar transformations



2D planar and linear transformations

Scale

$$\mathbf{M} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$

Flip across y

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

Rotate

$$\mathbf{M} = \begin{bmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{bmatrix}$$

Flip across origin

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

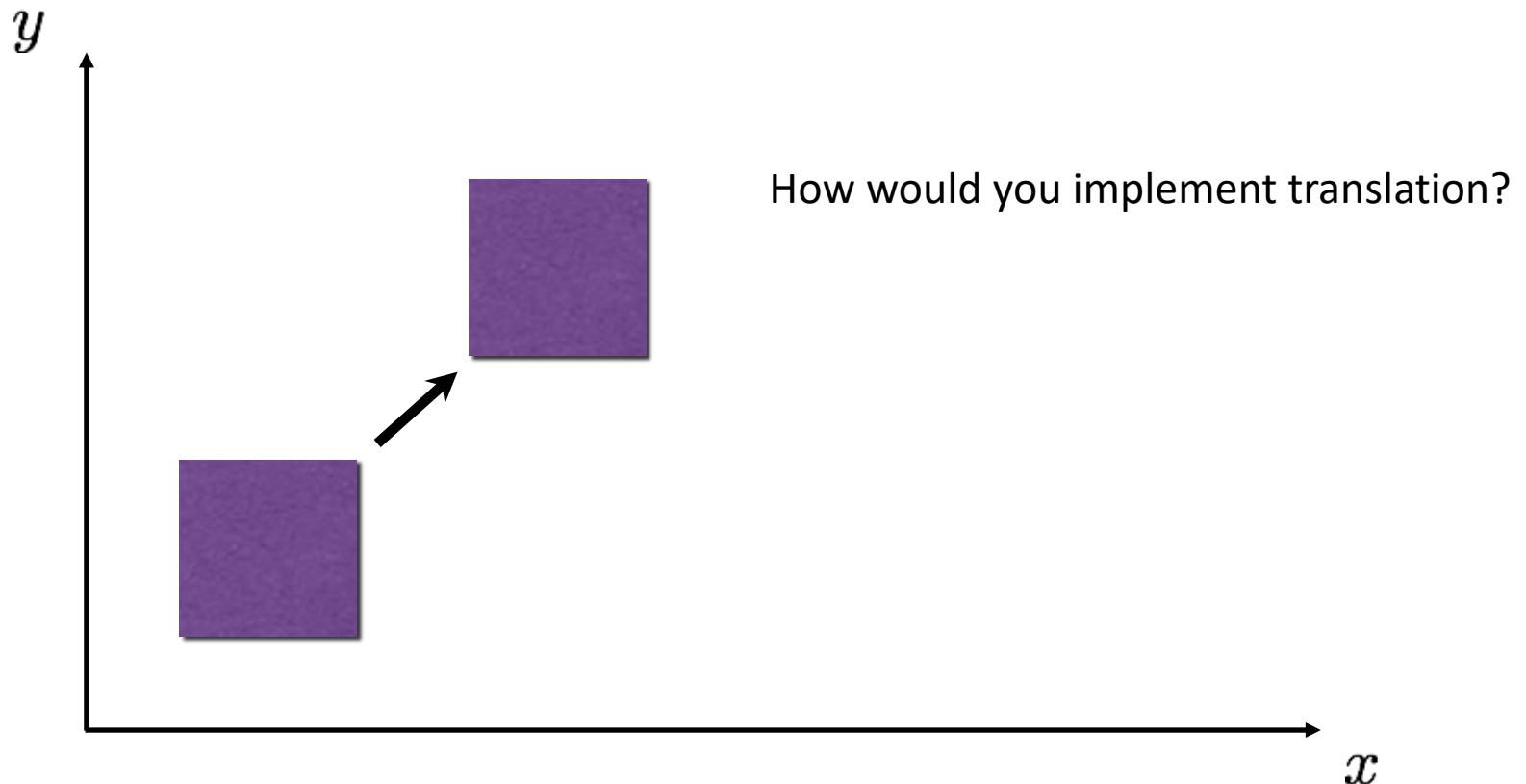
Shear

$$\mathbf{M} = \begin{bmatrix} 1 & s_x \\ s_y & 1 \end{bmatrix}$$

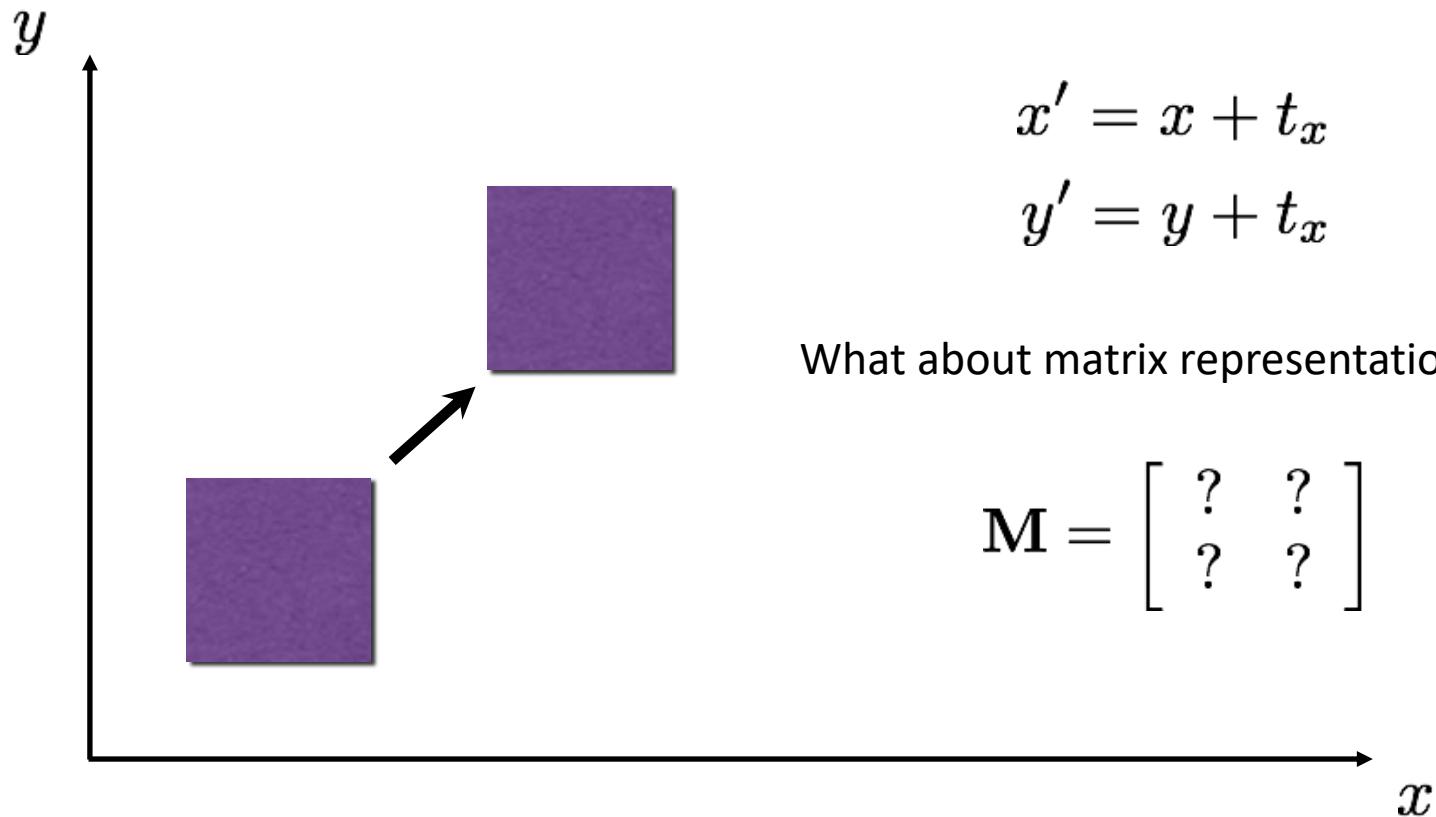
Identity

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

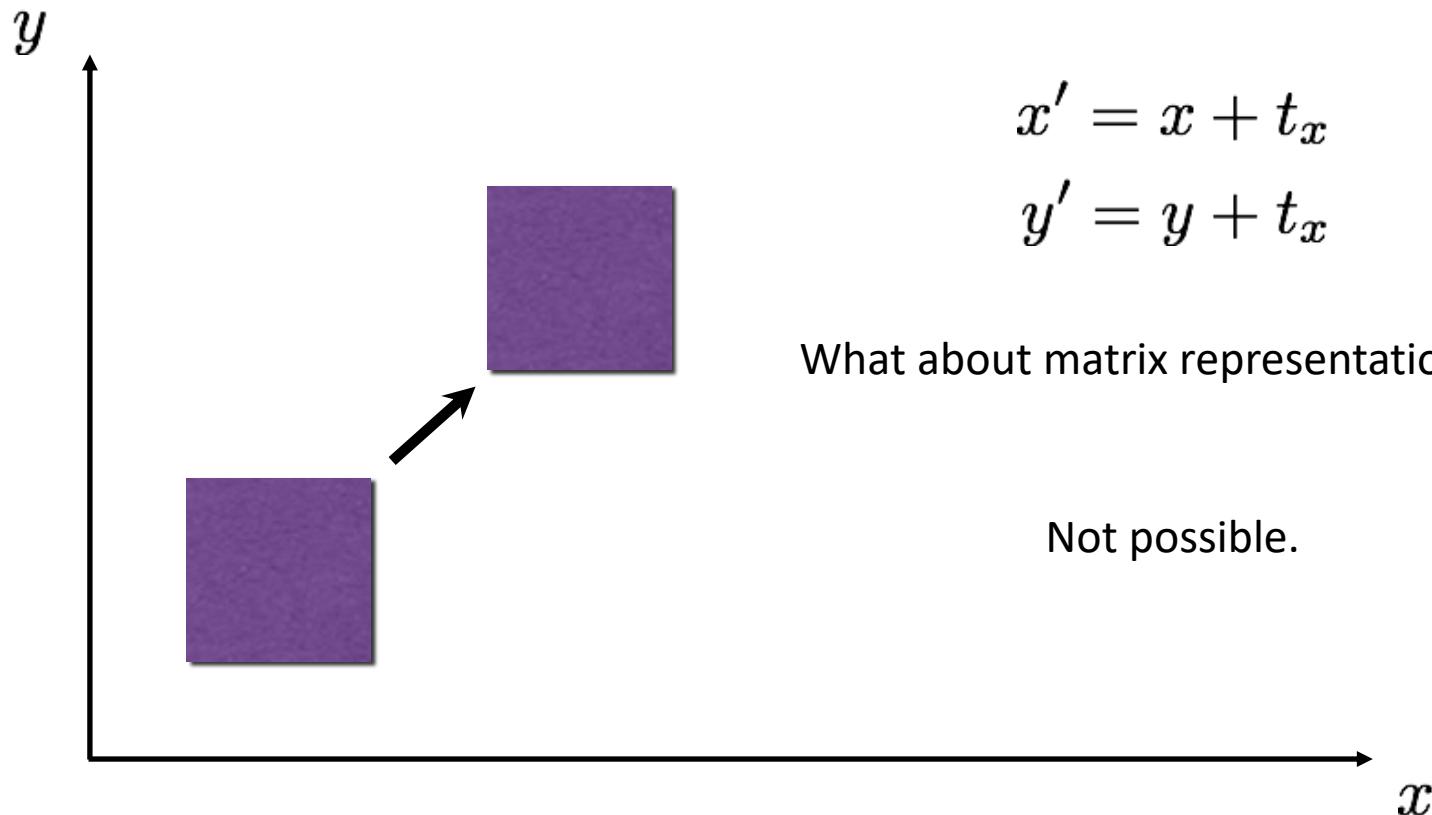
2D translation



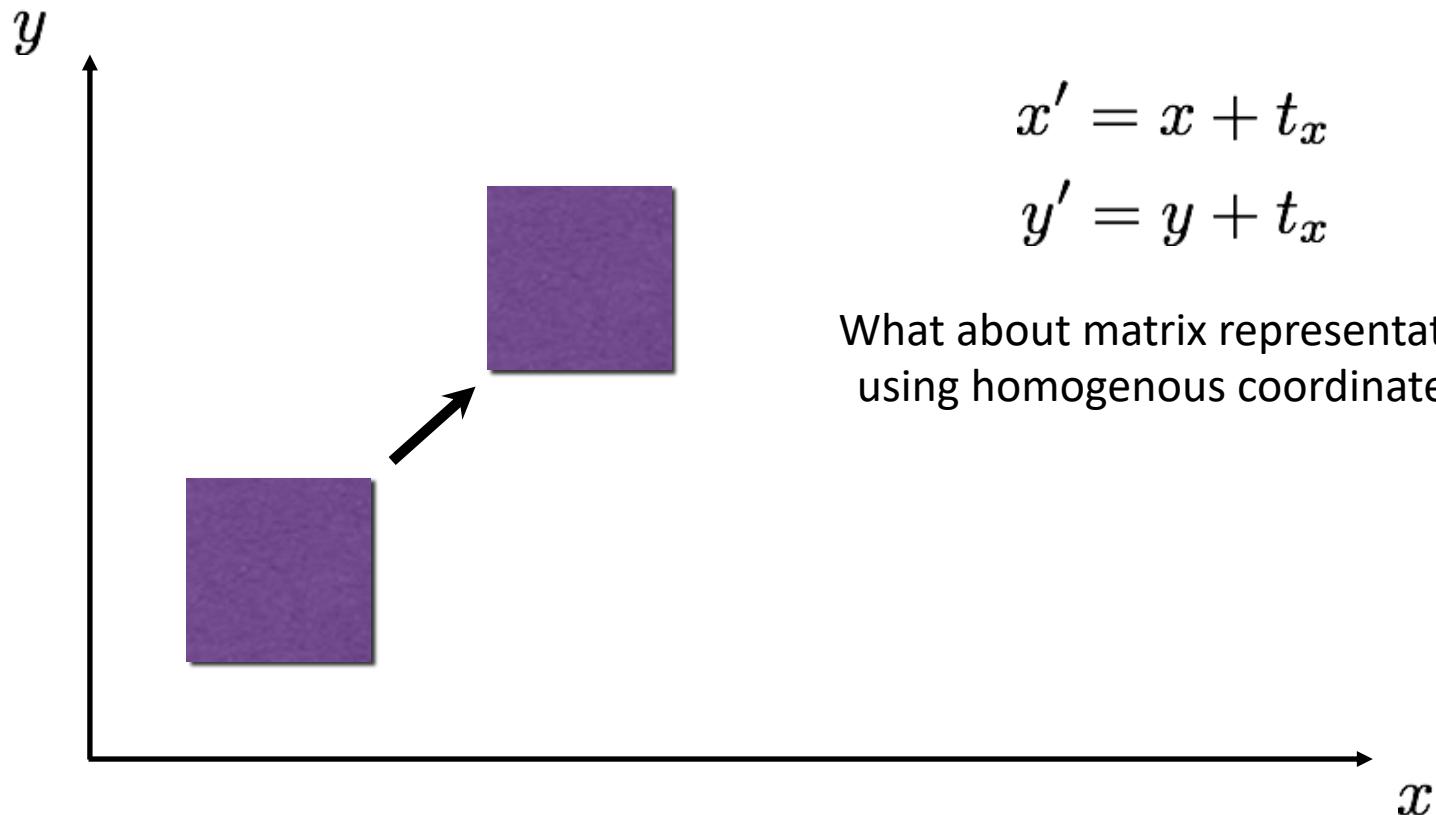
2D translation



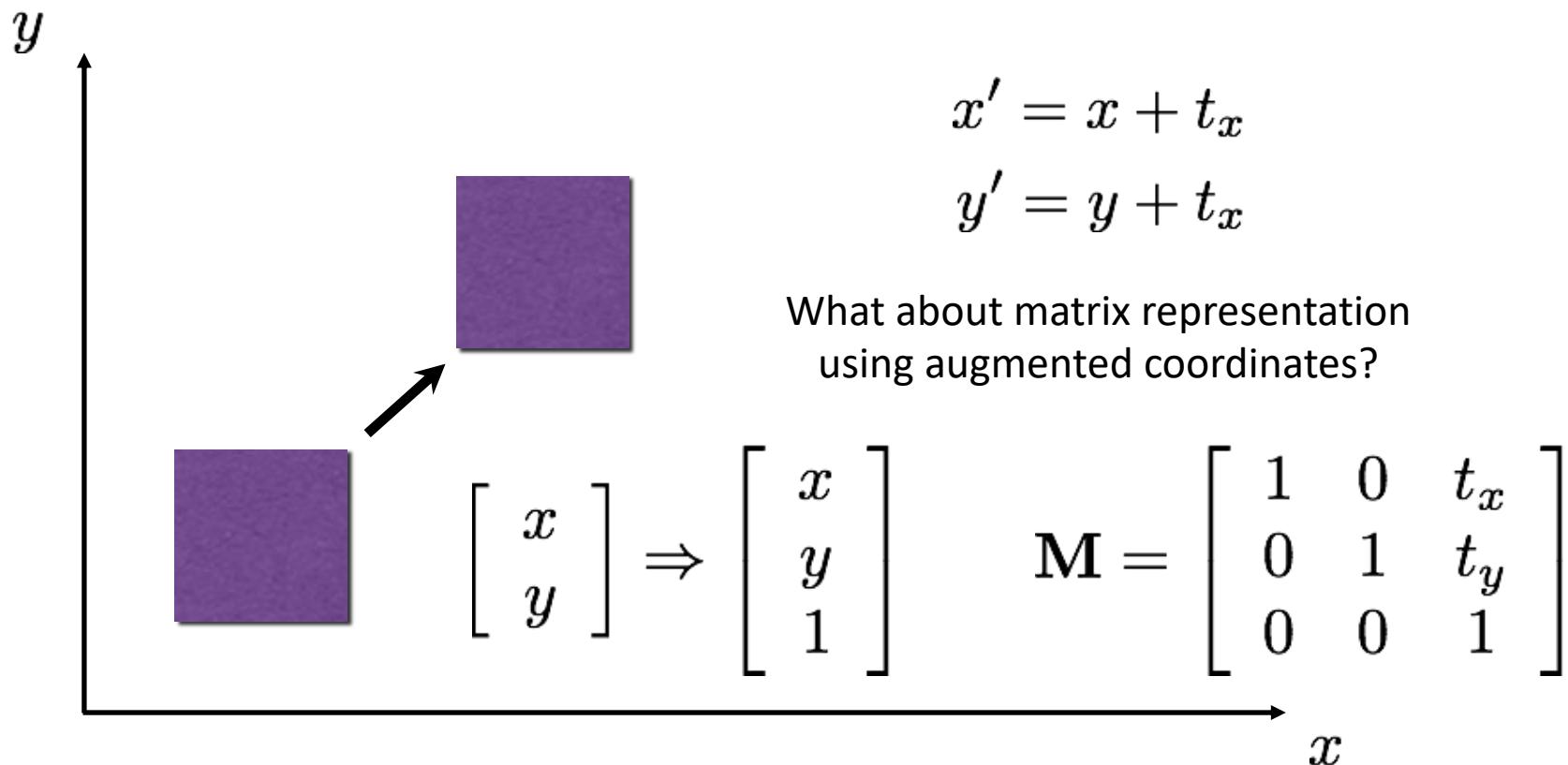
2D translation



2D translation

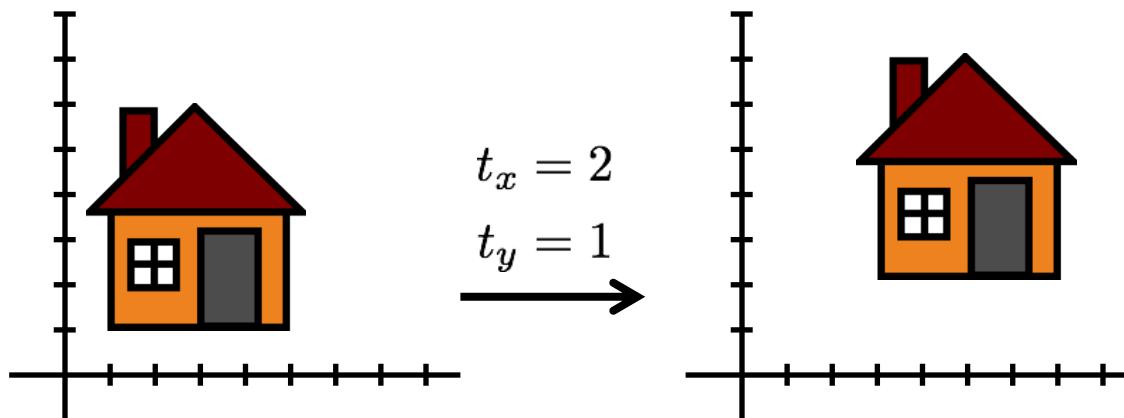


2D translation



2D translation using homogeneous coordinates

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



2D Transformations in homogeneous coordinates

Reminder: Homogeneous coordinates

Conversion:

- inhomogeneous → augmented/homogeneous

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

- homogeneous → inhomogeneous

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

- scale invariance

$$[x \ y \ w]^\top = \lambda [x \ y \ w]^\top$$

Special points:

- point at infinity

$$\begin{bmatrix} x & y & 0 \end{bmatrix}$$

- undefined

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

2D transformations

Re-write these transformations as 3x3 matrices:

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

translation

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

scaling

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

rotation

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

shearing

2D transformations

Re-write these transformations as 3x3 matrices:

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

translation

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

scaling

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

rotation

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

shearing

2D transformations

Re-write these transformations as 3x3 matrices:

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

translation

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

scaling

$$\begin{bmatrix} x' \\ y' \\ 1 \end{bmatrix} = \begin{bmatrix} \quad & \quad & \quad \\ \quad & \quad & ? \\ \quad & \quad & \quad \end{bmatrix} \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

rotation

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

shearing

2D transformations

Re-write these transformations as 3x3 matrices:

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

translation

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

scaling

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

rotation

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

shearing

Matrix composition

Transformations can be combined by matrix multiplication:

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$
$$\mathbf{p}' = ? ? ? \mathbf{p}$$

Matrix composition

Transformations can be combined by matrix multiplication:

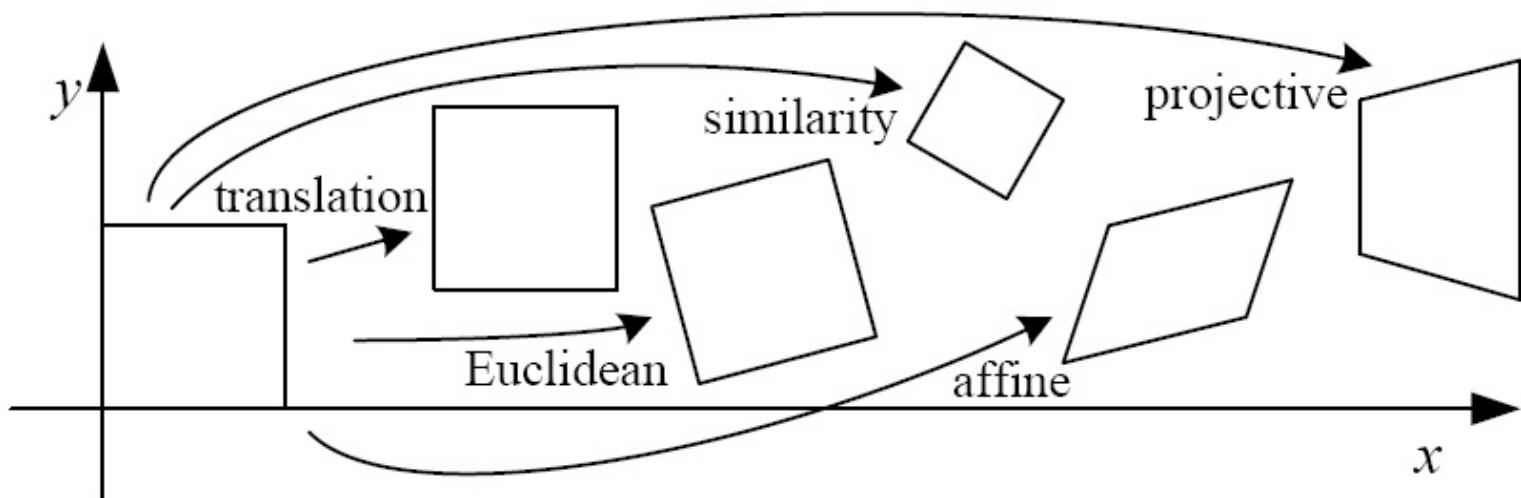
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \Theta & -\sin \Theta & 0 \\ \sin \Theta & \cos \Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

$$\mathbf{p}' = \text{translation}(t_x, t_y) \quad \text{rotation}(\theta) \quad \text{scale}(s, s) \quad \mathbf{p}$$

Does the multiplication order matter?

Classification of 2D transformations

Classification of 2D transformations



Classification of 2D transformations

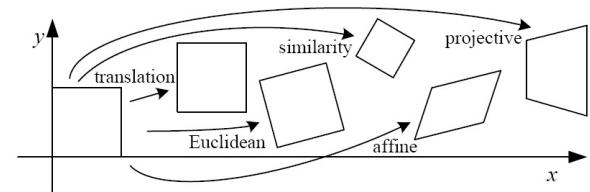
Name	Matrix	# D.O.F.
translation	$[\mathbf{I} \mid \mathbf{t}]$?
rigid (Euclidean)	$[\mathbf{R} \mid \mathbf{t}]$?
similarity	$[s\mathbf{R} \mid \mathbf{t}]$?
affine	$[\mathbf{A}]$?
projective	$[\tilde{\mathbf{H}}]$?

Translation

Translation:

$$\begin{bmatrix} 1 & 0 & t_1 \\ 0 & 1 & t_2 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

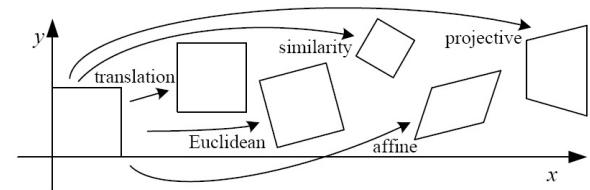


Euclidean/Rigid

Euclidean (rigid):
rotation + translation

$$\begin{bmatrix} \cos \theta & -\sin \theta & r_3 \\ \sin \theta & \cos \theta & r_6 \\ 0 & 0 & 1 \end{bmatrix}$$

How many degrees of freedom?

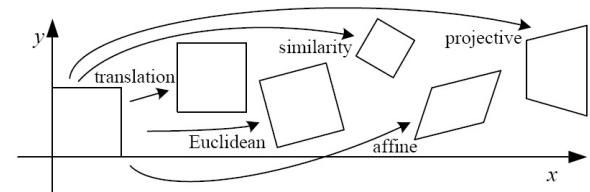


Affine

Affine transform:
uniform scaling + shearing
+ rotation + translation

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \\ 0 & 0 & 1 \end{bmatrix}$$

Are there any values that are related?



Affine transformations

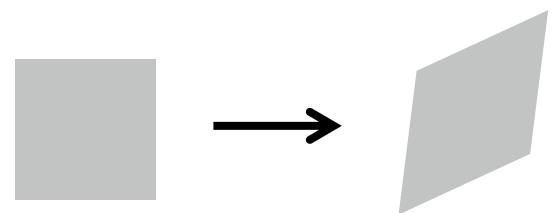
Affine transformations are combinations of

- arbitrary (4-DOF) linear transformations
- + translations

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

Properties of affine transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines map to parallel lines
- ratios are preserved
- compositions of affine transforms are also affine transforms



Does the last coordinate w ever change?

Projective transformations

Projective transformations are combinations of

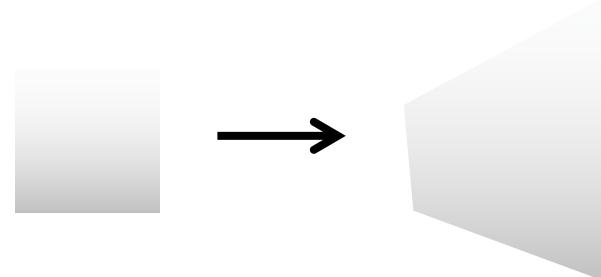
- affine transformations;
- + projective wraps

$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

8 DOF: vectors (and therefore matrices) are defined up to scale)

Properties of projective transformations:

- origin does not necessarily map to origin
- lines map to lines
- parallel lines do not necessarily map to parallel lines
- ratios are not necessarily preserved
- compositions of projective transforms are also projective transforms



Classification of 2D transformations

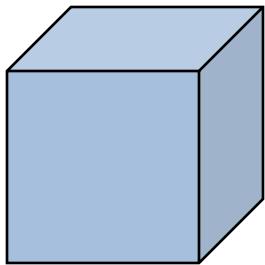
Name	Matrix	# D.O.F.
translation	$[\mathbf{I} \mid \mathbf{t}]$?
rigid (Euclidean)	$[\mathbf{R} \mid \mathbf{t}]$?
similarity	$[s\mathbf{R} \mid \mathbf{t}]$?
affine	$[\mathbf{A}]$?
projective	$[\tilde{\mathbf{H}}]$?

Classification of 2D transformations

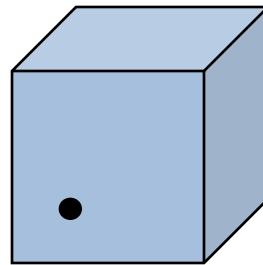
Name	Matrix	# D.O.F.
translation	$[I \mid t]$	2
rigid (Euclidean)	$[R \mid t]$	3
similarity	$[sR \mid t]$	4
affine	$[A]$	6
projective	$[\tilde{H}]$	8

2.1.3: 3D Transformations

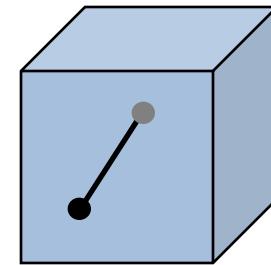
- Need a way to specify the six degrees-of-freedom of a rigid body.
- Why are there 6 DOF?



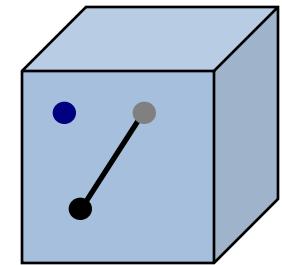
A rigid body is a collection of points whose positions relative to each other can't change



Fix one point,
three DOF



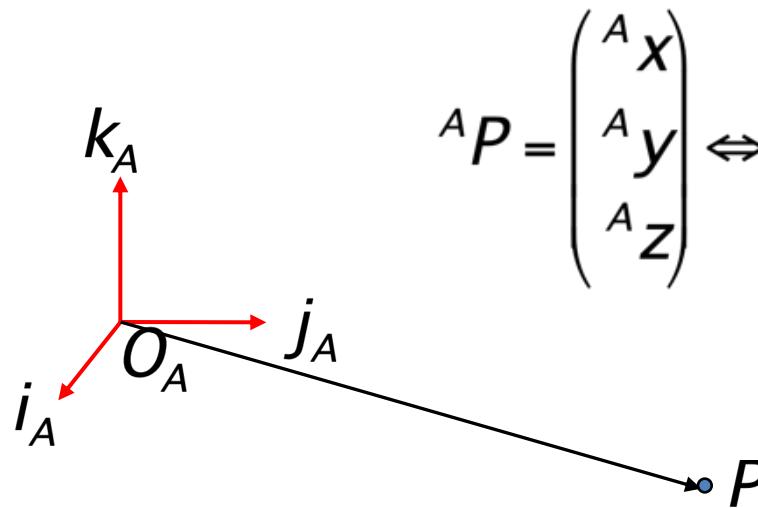
Fix second point,
two more DOF
(must maintain
distance constraint)



Third point adds
one more DOF,
for rotation
around line

Notations

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

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


The diagram shows a 3D coordinate system A centered at point O_A . The system has three red unit vectors originating from O_A : i_A pointing along the negative x-axis, j_A pointing along the positive y-axis, and k_A pointing along the positive z-axis. A point P is shown in space, with a line segment connecting it to the origin O_A .

Translation

- Using augmented/homogeneous coordinates, translation is 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 in homogeneous coordinates

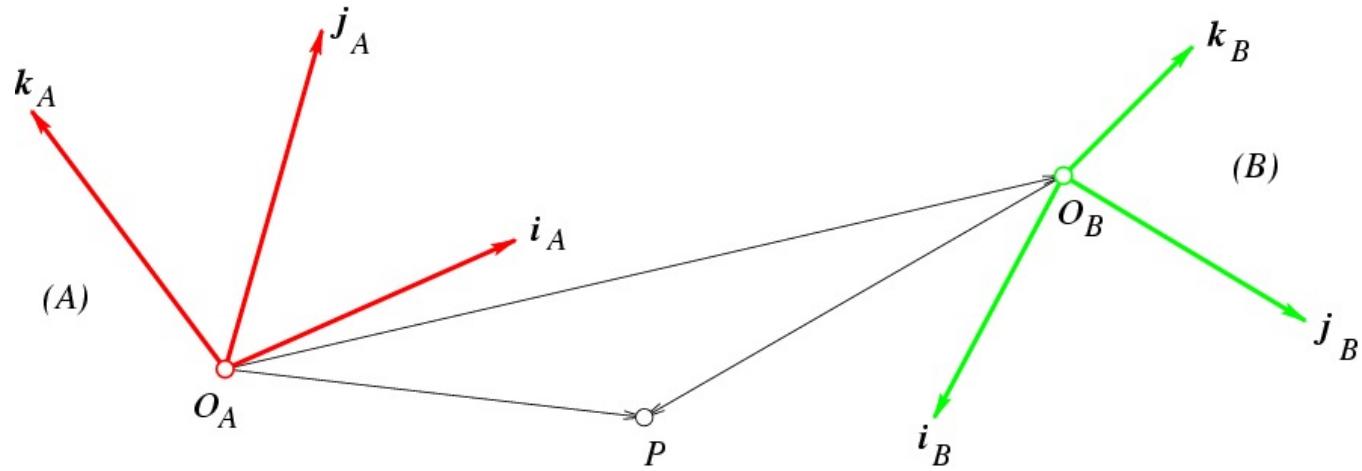
- Using homogeneous coordinates, rotation can be expressed as a matrix multiplication.

$${}^B P = {}_A^B R {}^A P$$

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

- R is a rotation matrix:
 - Columns are unit vectors
 - Columns are mutually orthogonal
 - Inverse is transpose
- Rotation is not commutative

3D Rigid transformations



$${}^B P = {}_A^B R {}^A P + {}^B O_A$$

3D Rigid transformations

- Unified treatment using homogeneous coordinates.

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

$$= \begin{bmatrix} {}^B R & {}^B O_A \\ {}^A O^T & 1 \end{bmatrix} \begin{bmatrix} {}^A P \\ 1 \end{bmatrix}$$



$$\begin{bmatrix} {}^B P \\ 1 \end{bmatrix} = {}^B T_A \begin{bmatrix} {}^A P \\ 1 \end{bmatrix}$$

Hierarchy of 3D Transforms



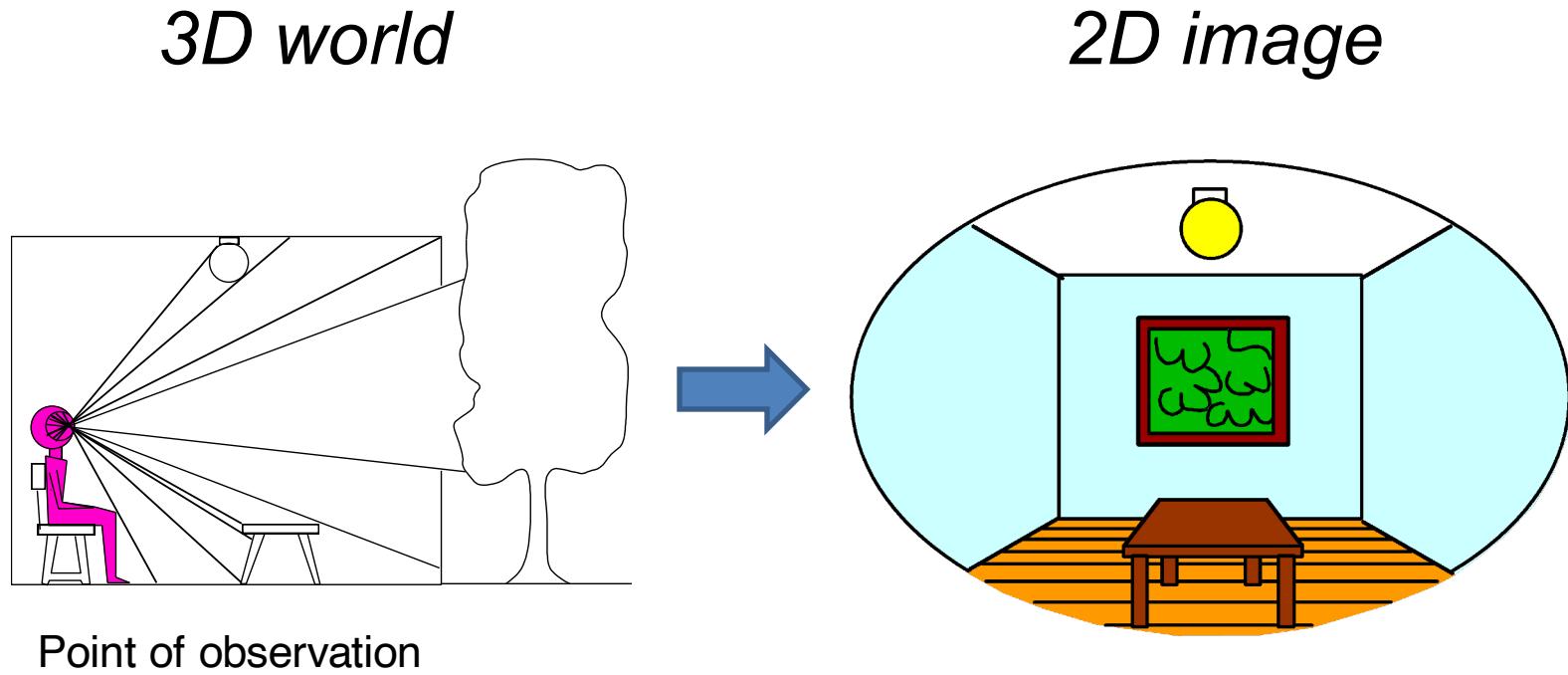
- Subgroup Structure:
 - Translation (? DOF)
 - Rigid 3D (? DOF)
 - Affine (? DOF)
 - Projective (? DOF)

Hierarchy of 3D Transforms

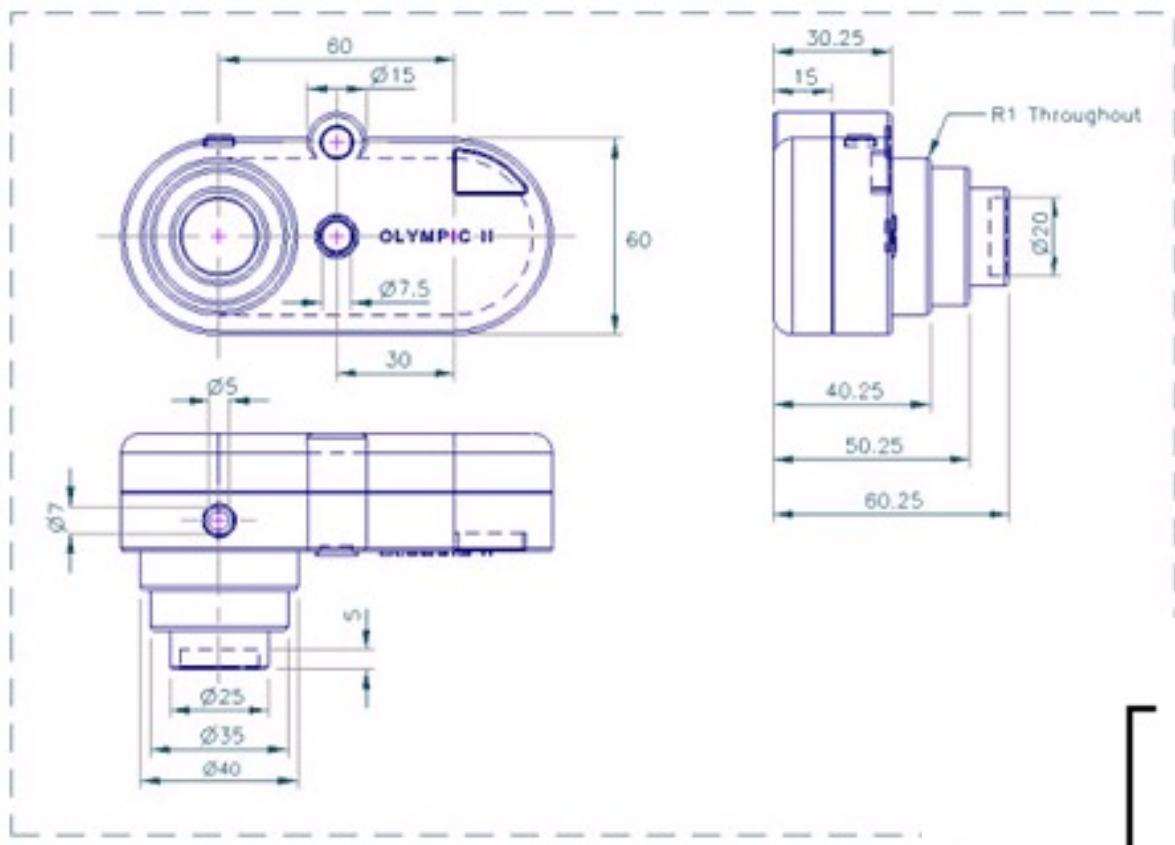


- Subgroup Structure:
 - Translation (3 DOF)
 - Rigid 3D (6 DOF)
 - Affine (12 DOF)
 - Projective (15 DOF)

2.1.5: 3D to 2D: Projection

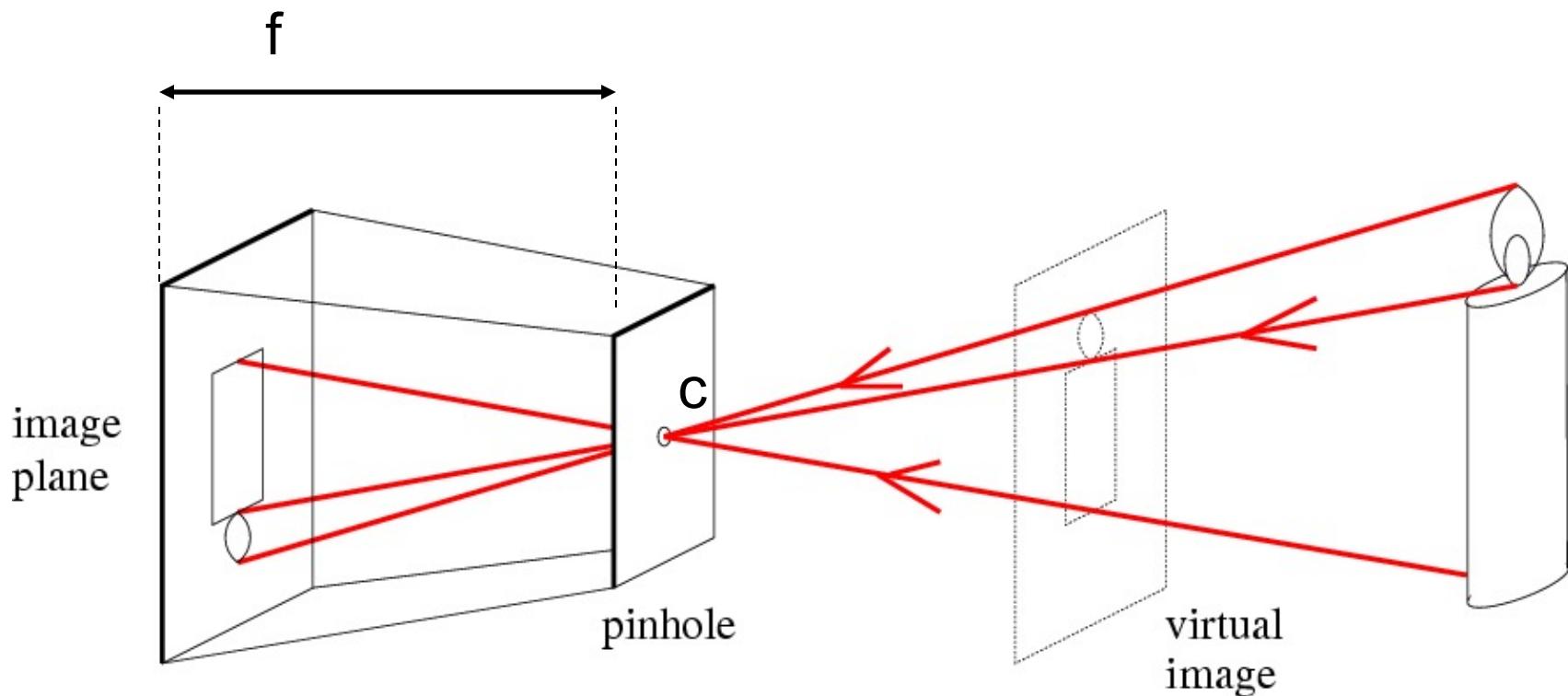


Orthographic Projection



$$\tilde{x} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tilde{p}$$

Pinhole camera



f = focal length

c = center of the camera

Pre-history: the Camera obscura

- Known during classical period in China and Greece
(e.g. Mo-Ti, China, 470BC to 390BC)

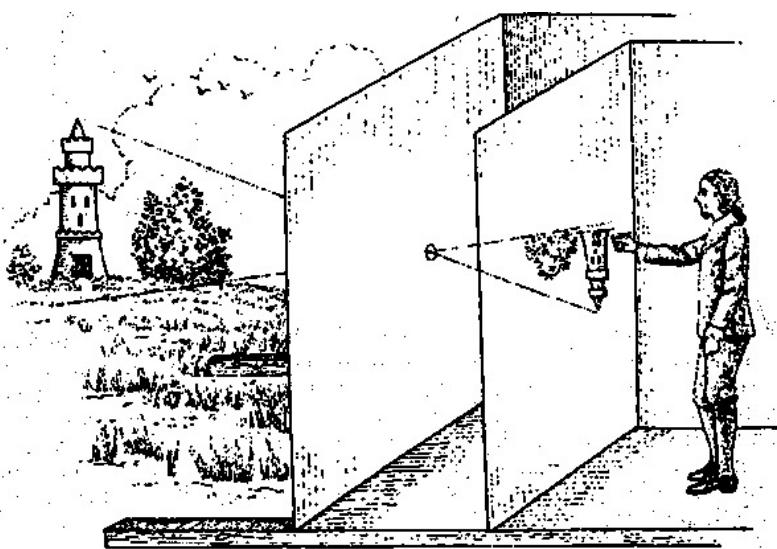


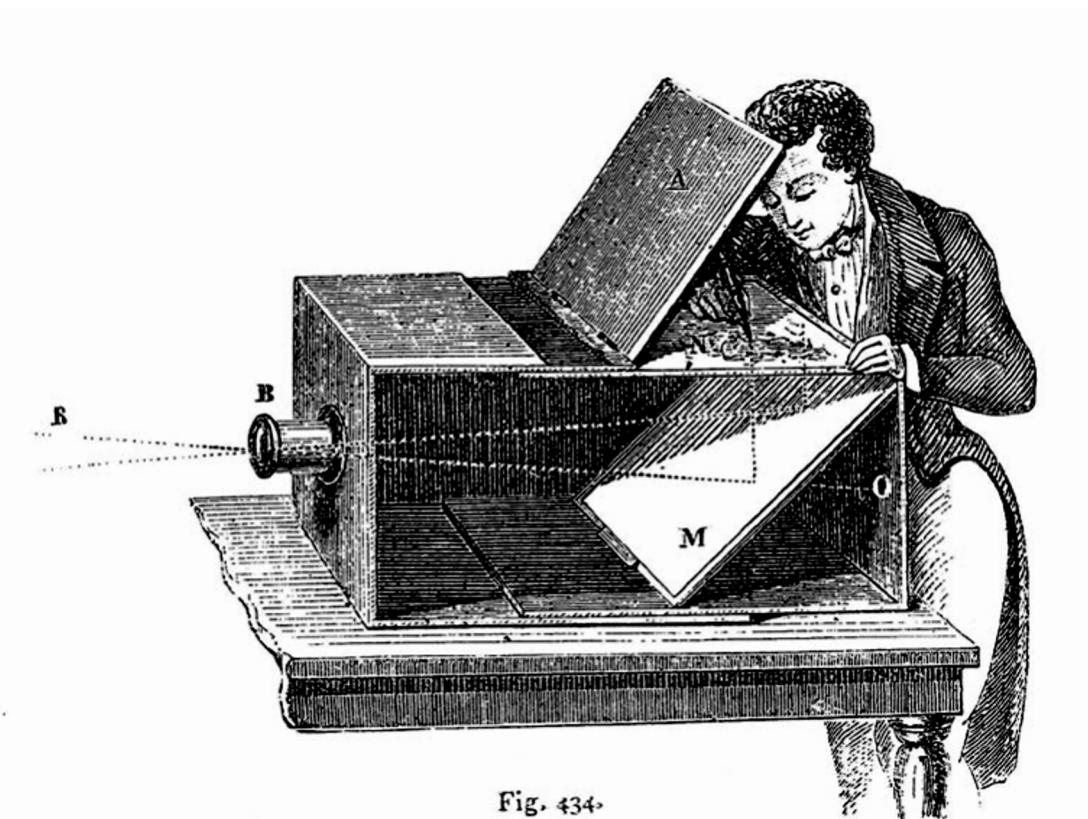
Illustration of Camera Obscura



Freestanding camera obscura at UNC Chapel Hill

Photo by Seth Ilys

Camera Obscura used for Tracing



Lens Based Camera Obscura, 1568

First Photograph

Oldest surviving photograph

- Took 8 hours on pewter plate



Joseph Niepce, 1826

Photograph of the first photograph



Stored at UT Austin

Niepce later teamed up with Daguerre, who eventually created Daguerrotypes

Projection can be tricky...



CoolOpticalIllusions.com

Projection can be tricky...



CoolOpticalIllusions.com

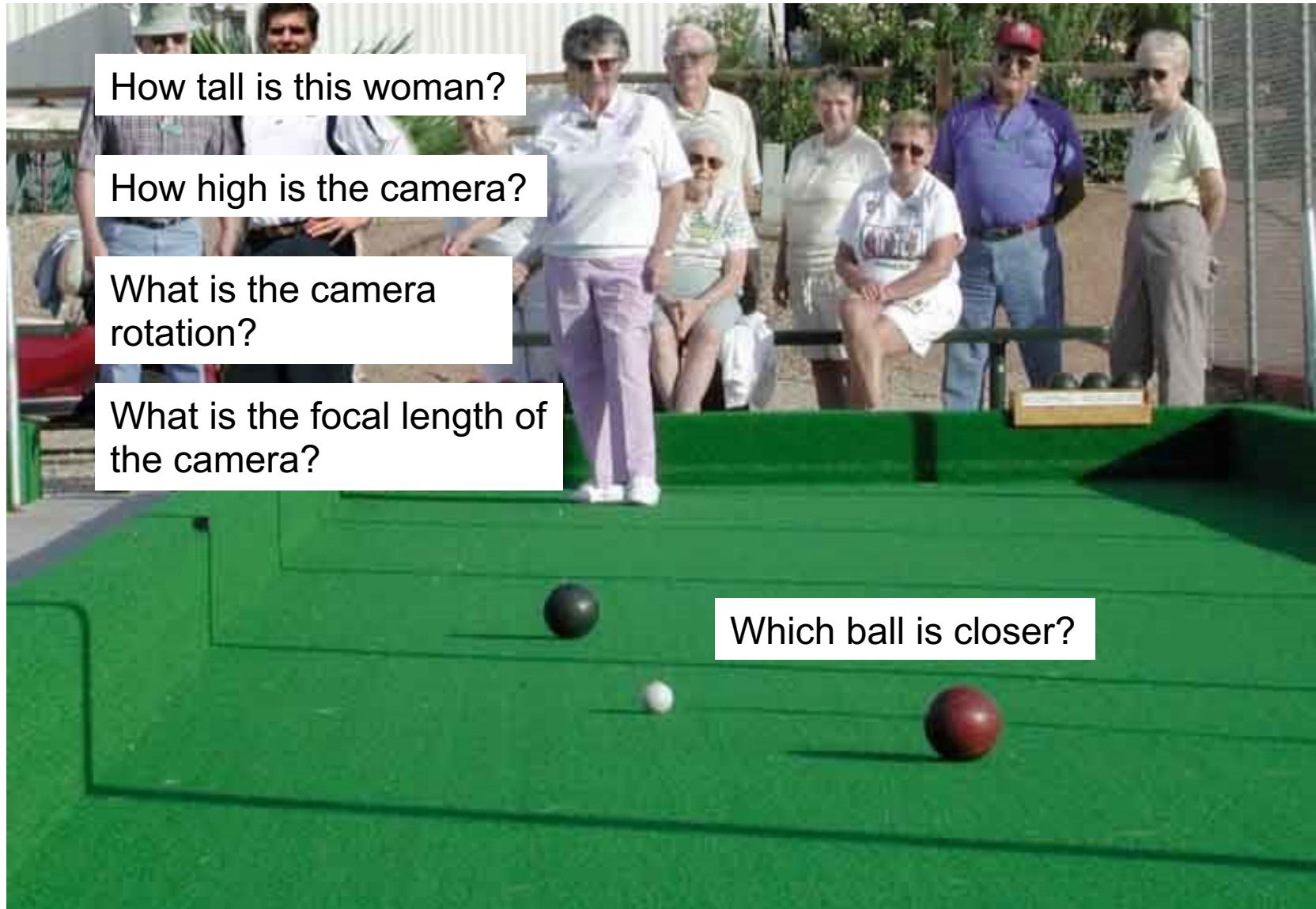






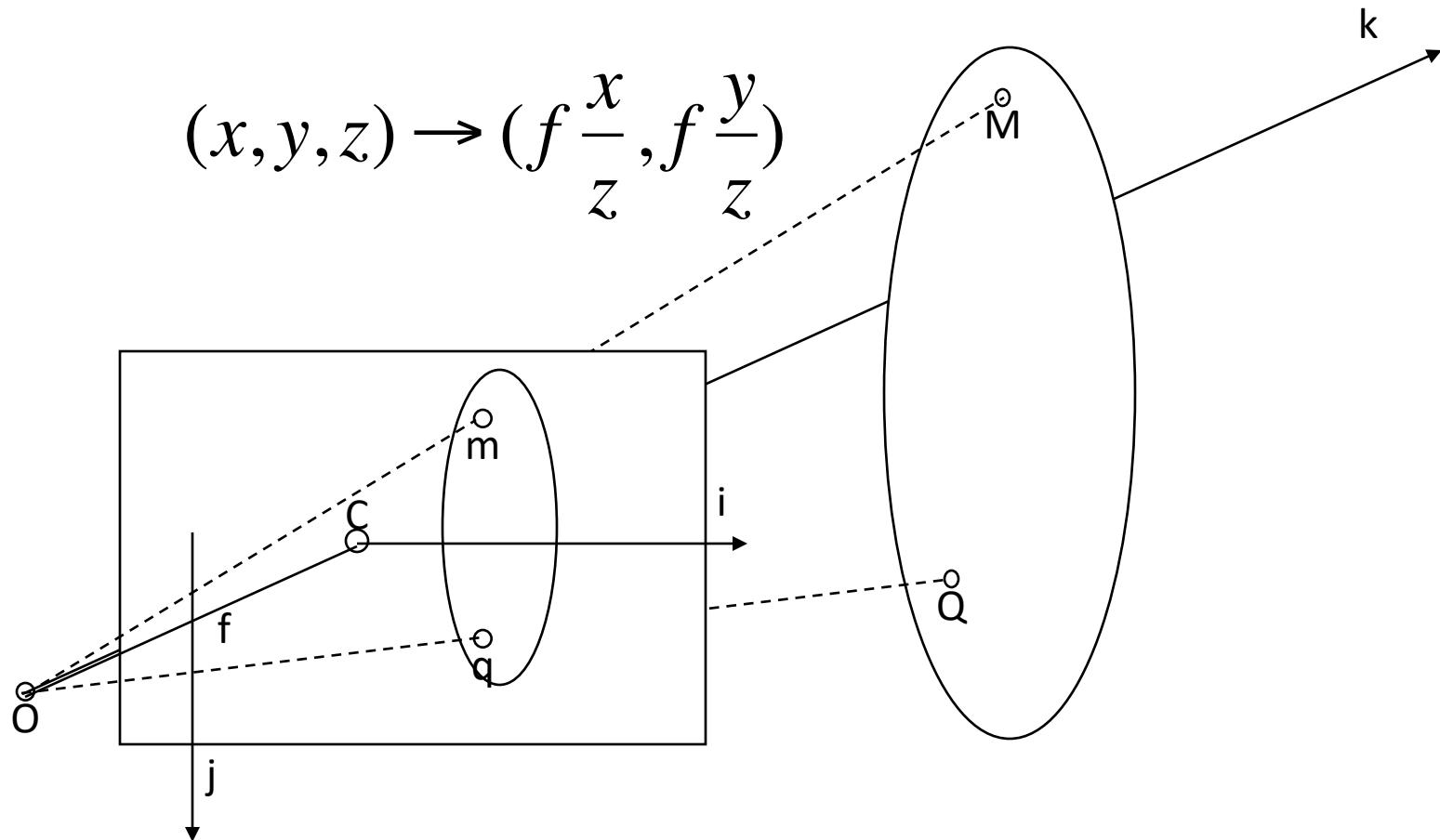


Camera and World Geometry



Pinhole Camera

- Fundamental equation:



Homogeneous Coordinates

Linear transformation of homogeneous (projective) coordinates

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

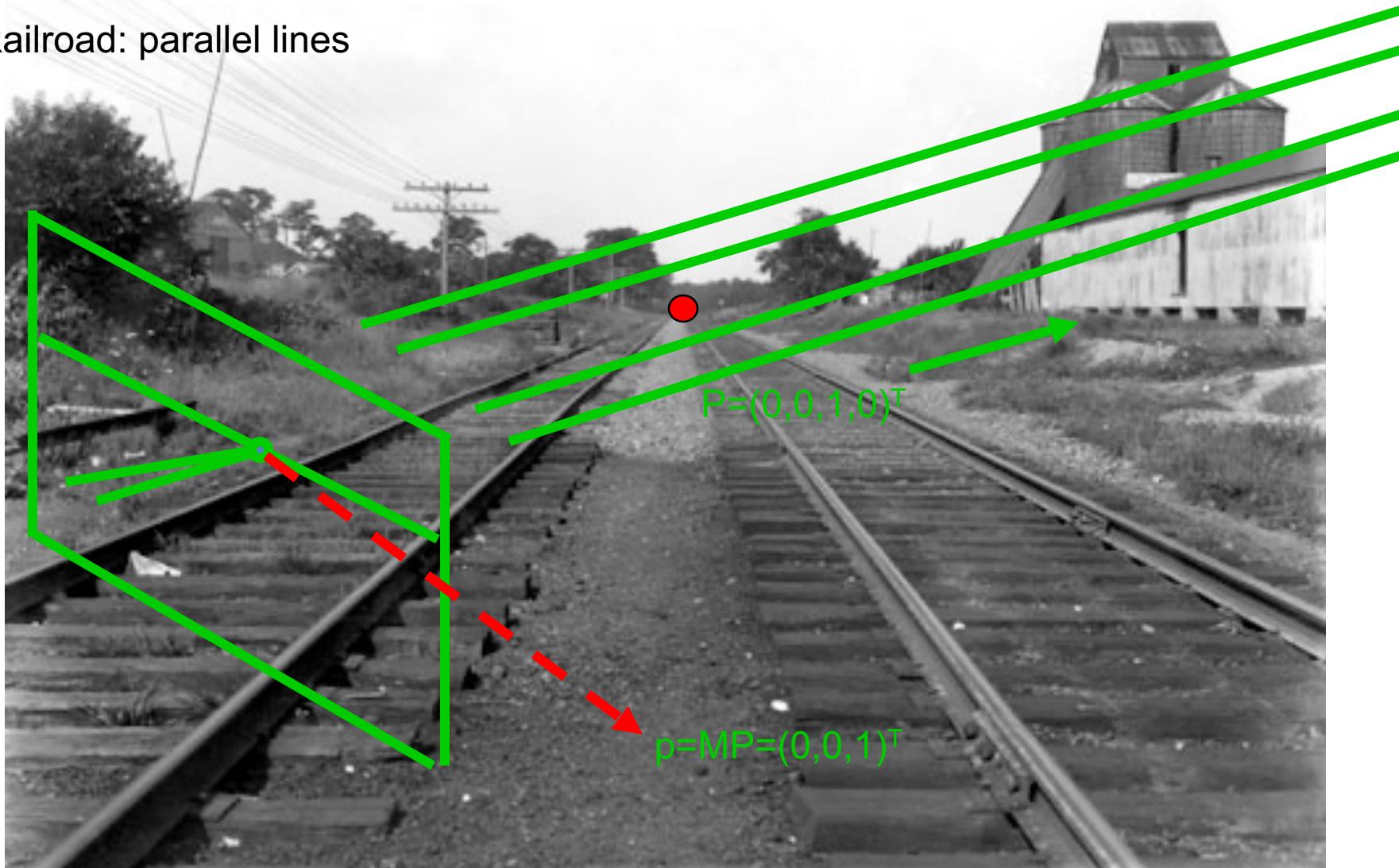
Recover image (Euclidean) coordinates by normalizing:

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

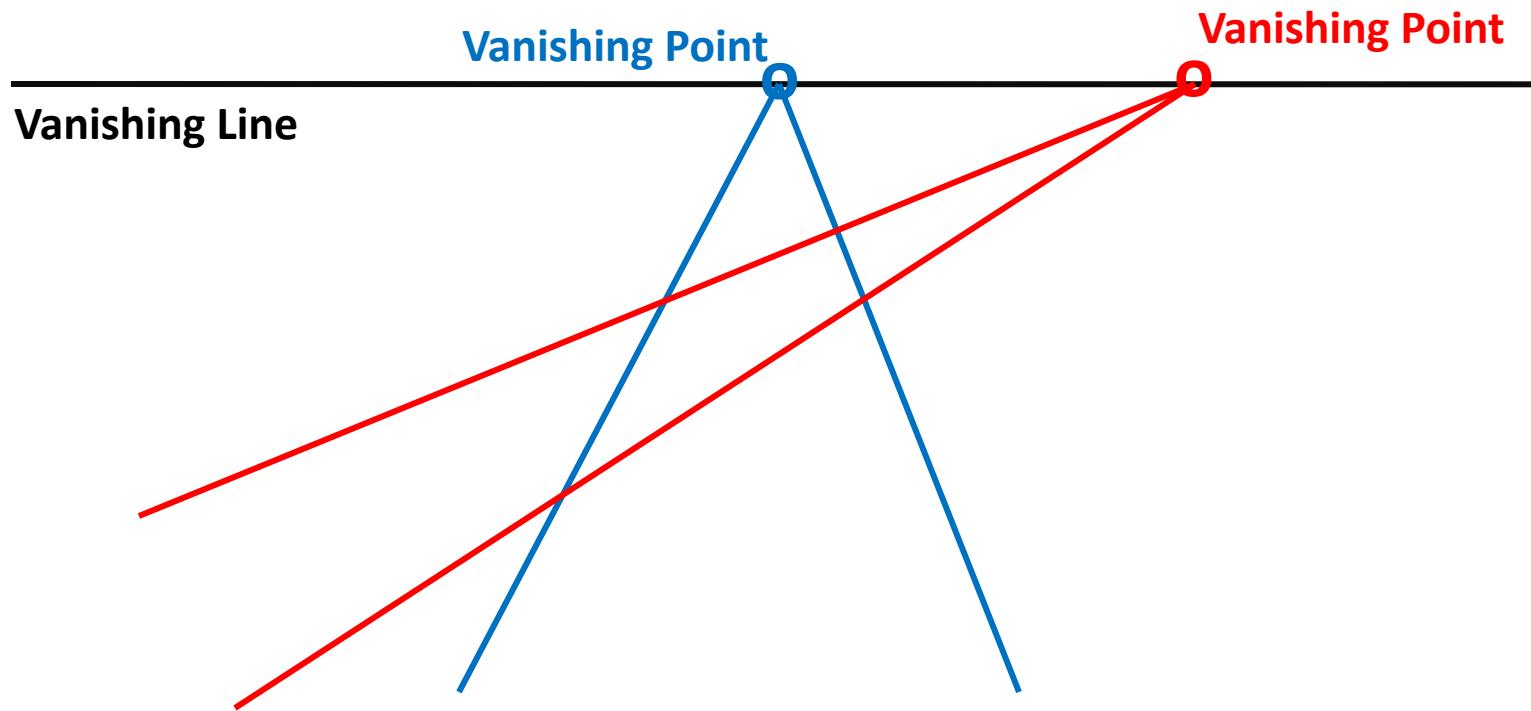
$$\hat{v} = \frac{v}{w} = \frac{Y}{Z}$$

We can see infinity !

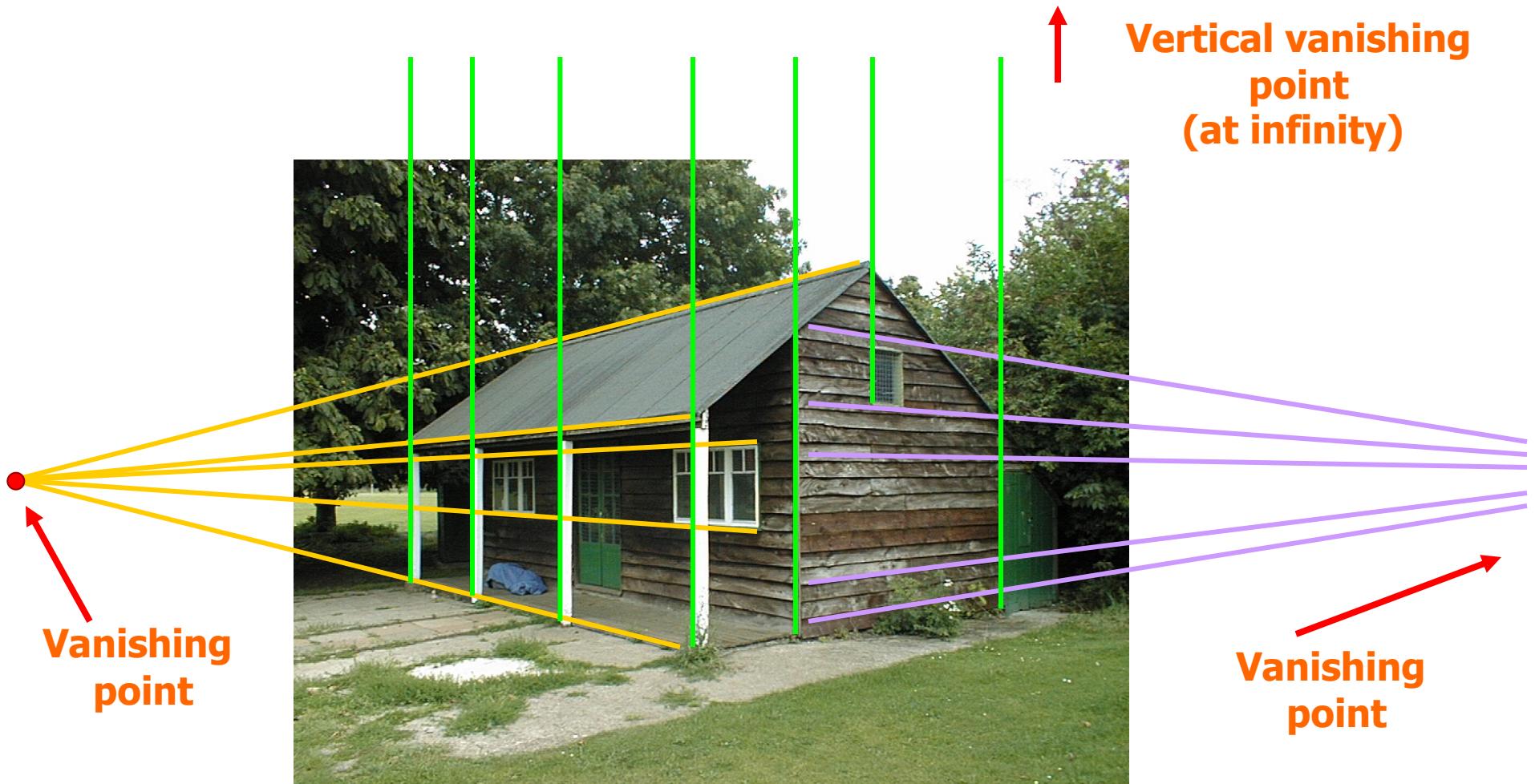
Railroad: parallel lines



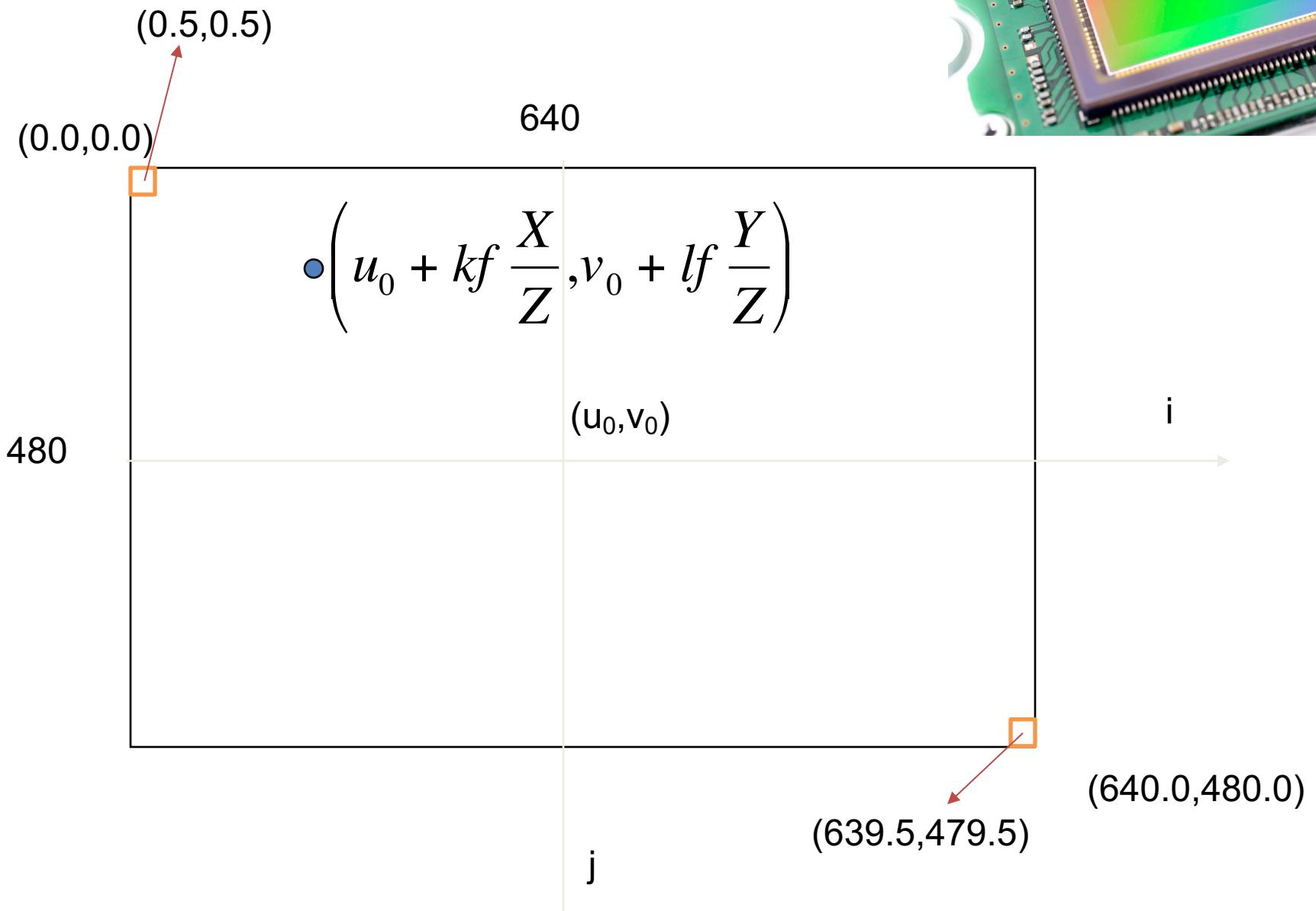
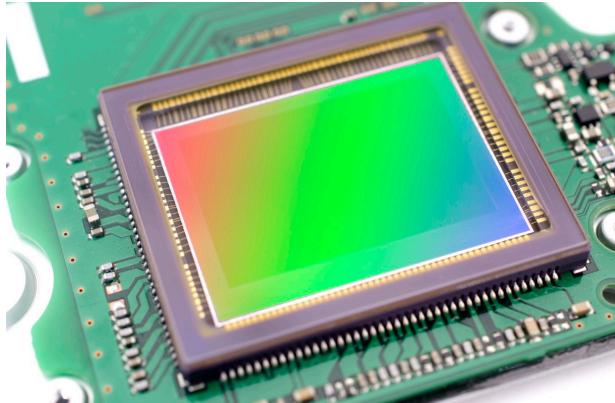
Vanishing points and lines



Vanishing points and lines



Pixel coordinates in 2D



Intrinsic Calibration

3×3 Calibration Matrix K

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

Recover image (Euclidean) coordinates by normalizing :

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

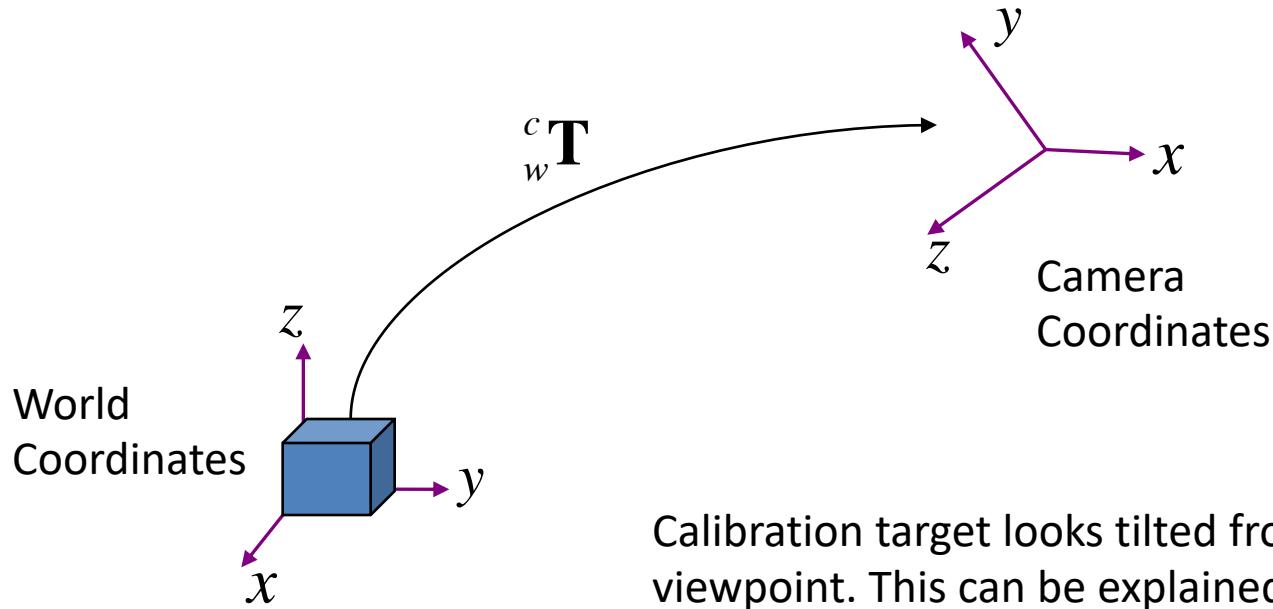
$$\hat{v} = \frac{v}{w} = \frac{\beta Y + v_0}{Z}$$

skew

5 Degrees of Freedom !

Camera Pose

In order 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.

Projective Camera Matrix

Camera = Calibration × Projection × Extrinsics

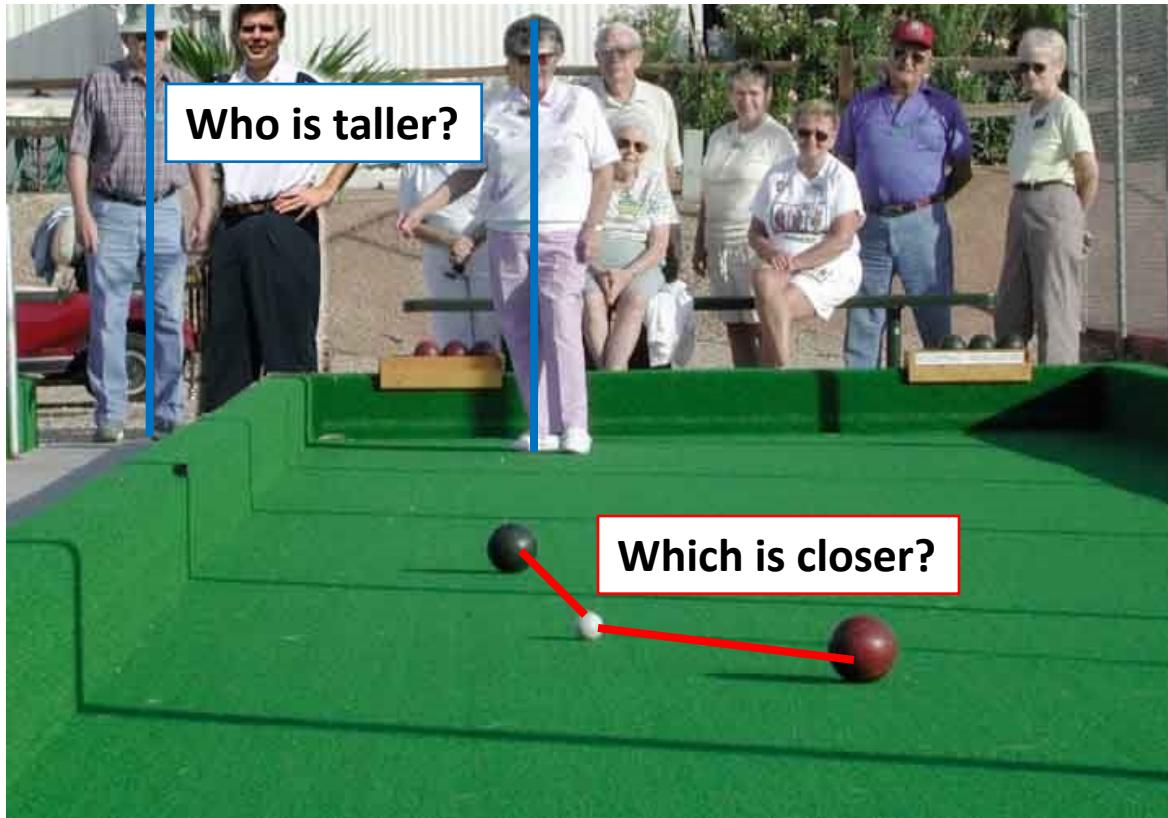
$$m = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = \begin{bmatrix} \alpha & s & u_0 \\ & \beta & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} R & t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ T \end{bmatrix}$$
$$= K[R \quad t]M = PM$$

5+6 DOF = 11 !

Projective Geometry

What is lost?

- Length



Length and area are not preserved

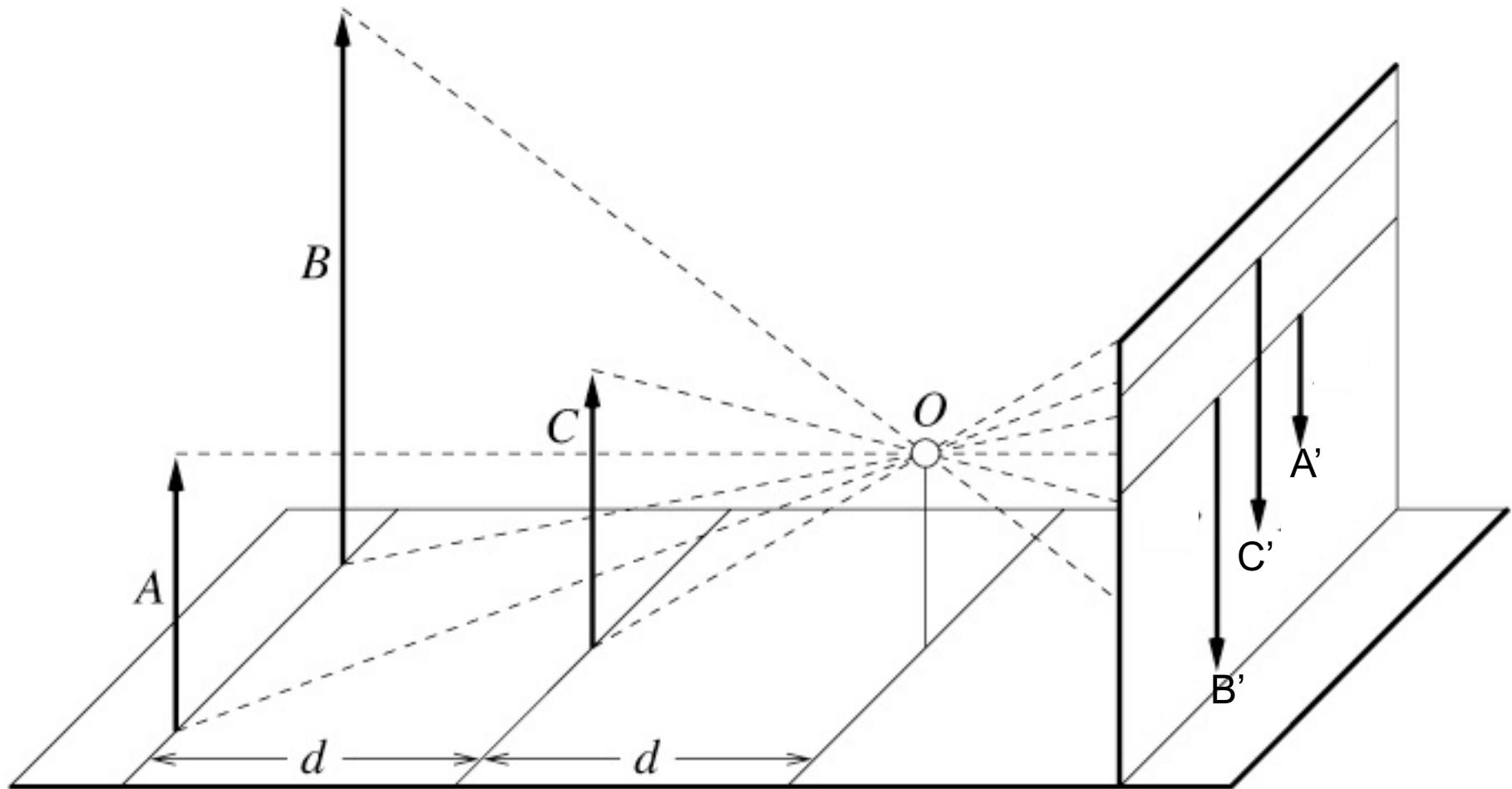
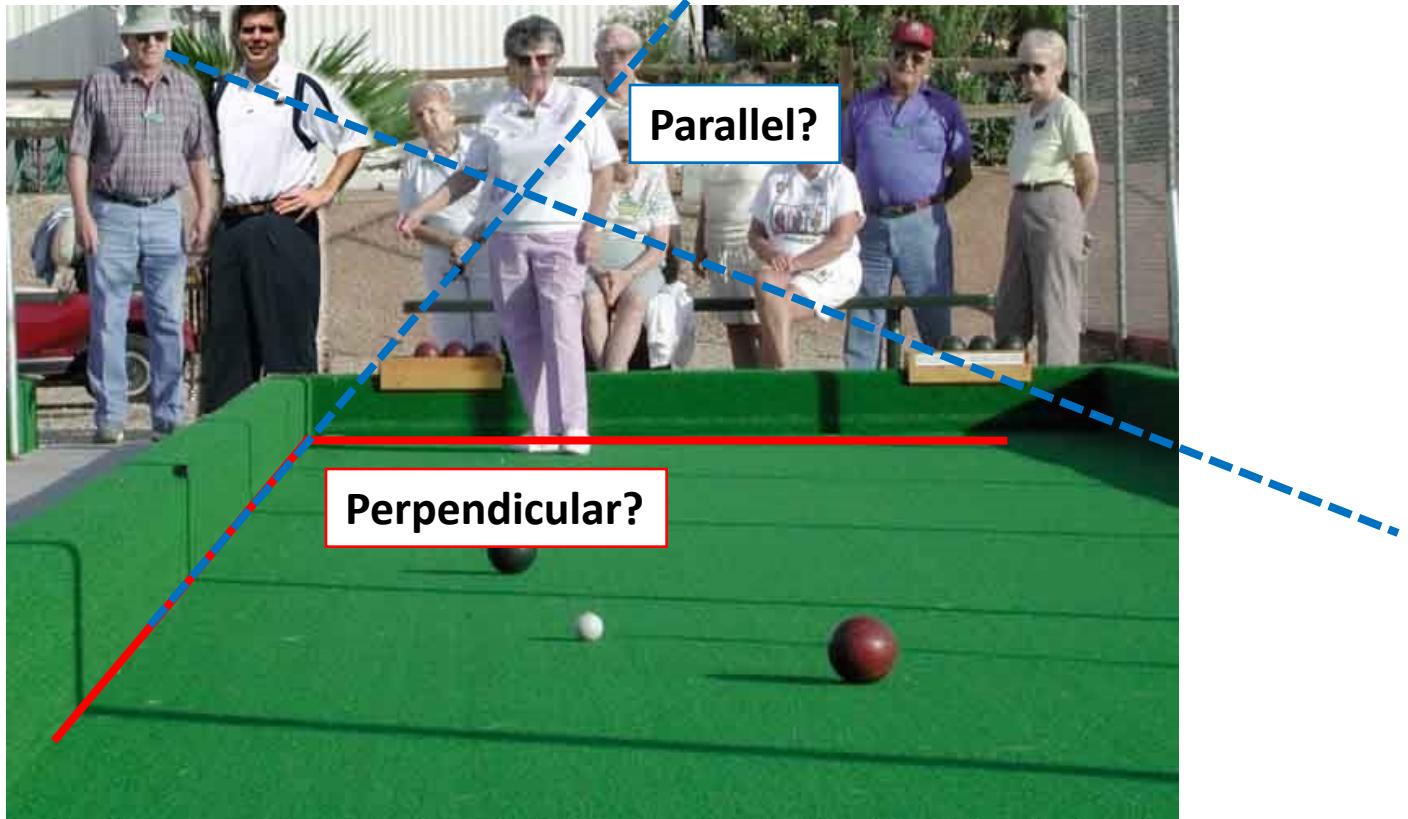


Figure by David Forsyth

Projective Geometry

What is lost?

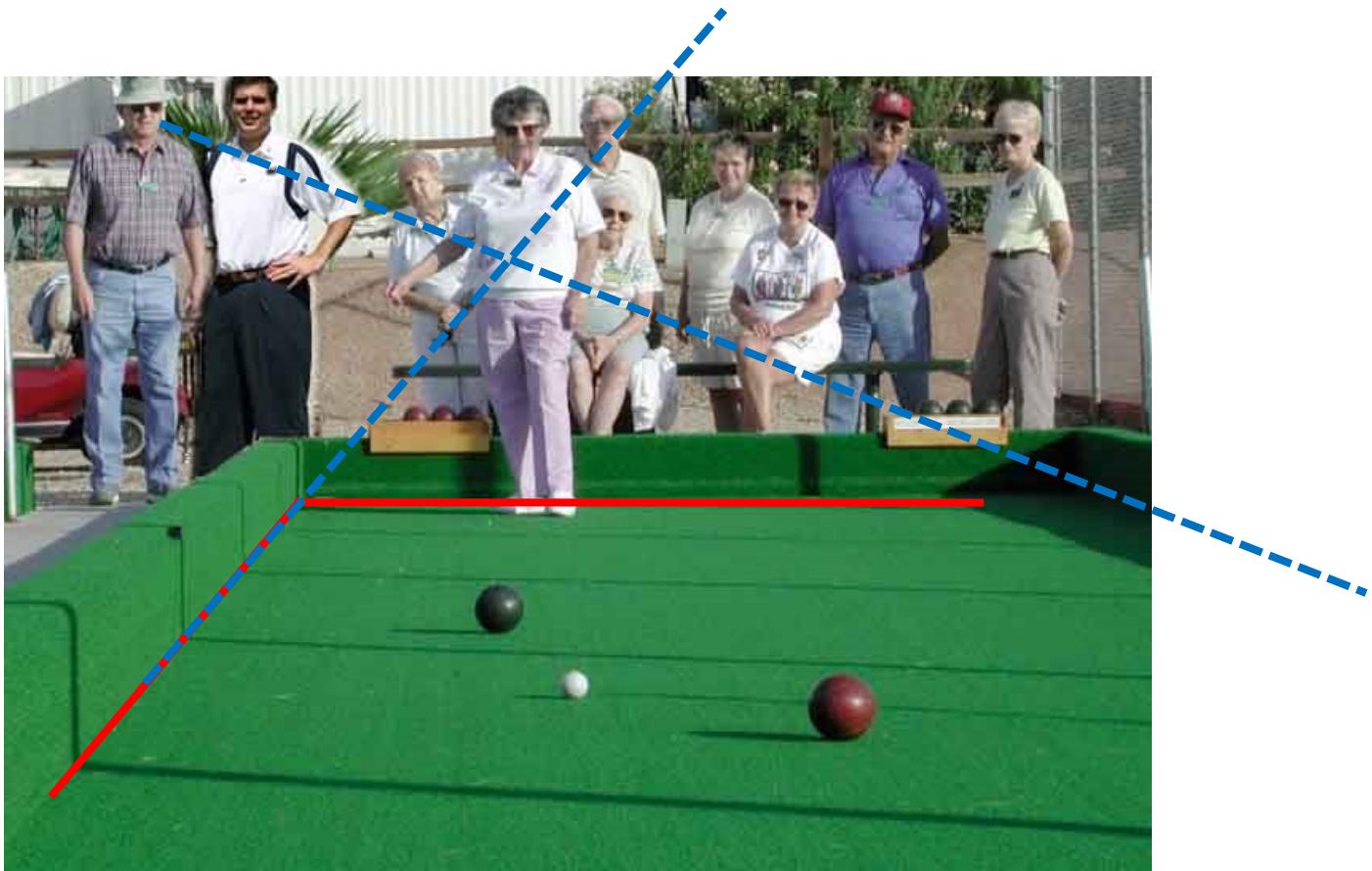
- Length
- Angles



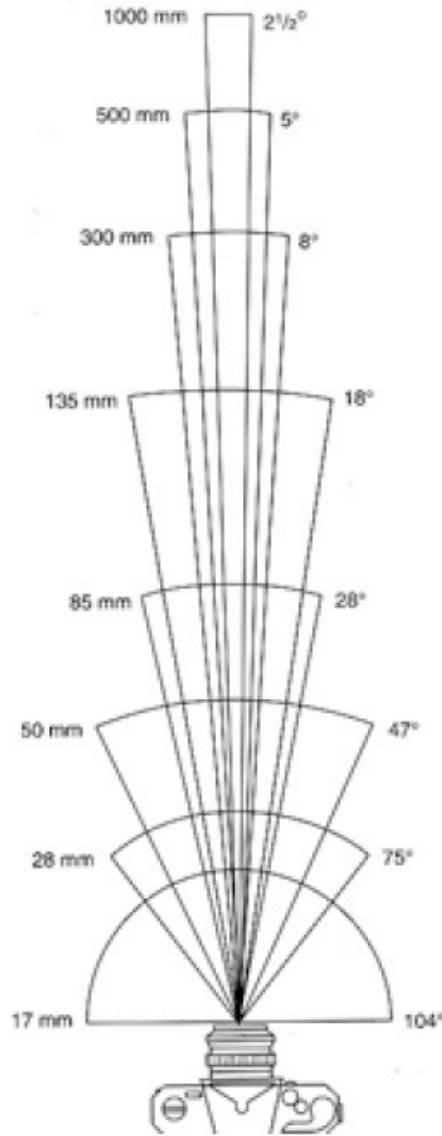
Projective Geometry

What is preserved?

- Straight lines are still straight

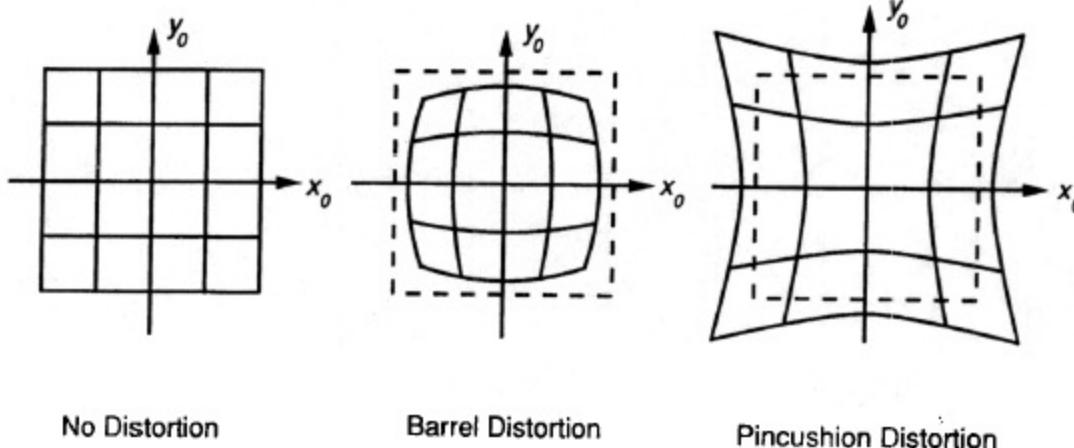


Field of View (Zoom, focal length)



From London and Upton

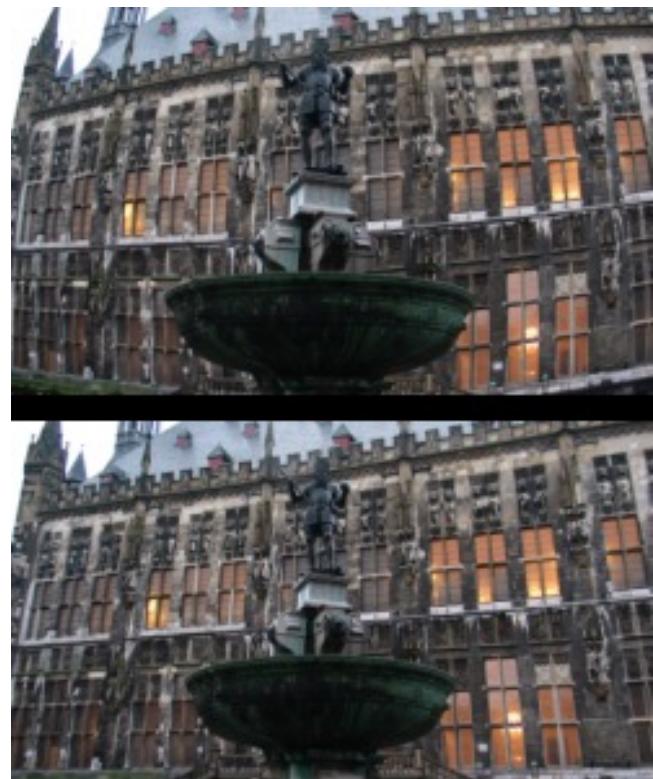
2.1.6 Radial Distortion



No Distortion

Barrel Distortion

Pincushion Distortion



Corrected Barrel Distortion