19CSE435: Computer Vision

Image formation: Geometric Primitives

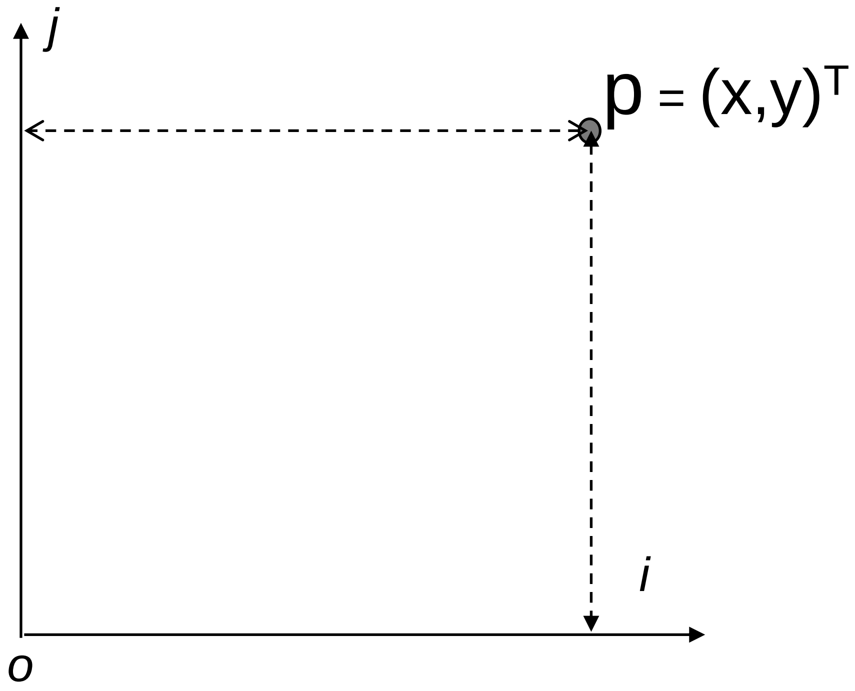
Adopted from Computer Vision Textbook and course materials R_Szeliski

2.1.1 Geometric Primitives

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

2D Coordinate Frames & Points

- coordinates x and y



$$\boldsymbol{x} = \begin{bmatrix} x \\ y \end{bmatrix}.$$

homogeneous coordinates,

$$\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) \in \mathcal{P}^2,$$
$$\mathcal{P}^2 = \mathcal{R}^3 - (0,0,0)$$

2D *projective space.*

inhomogeneous vector $\boldsymbol{x}$

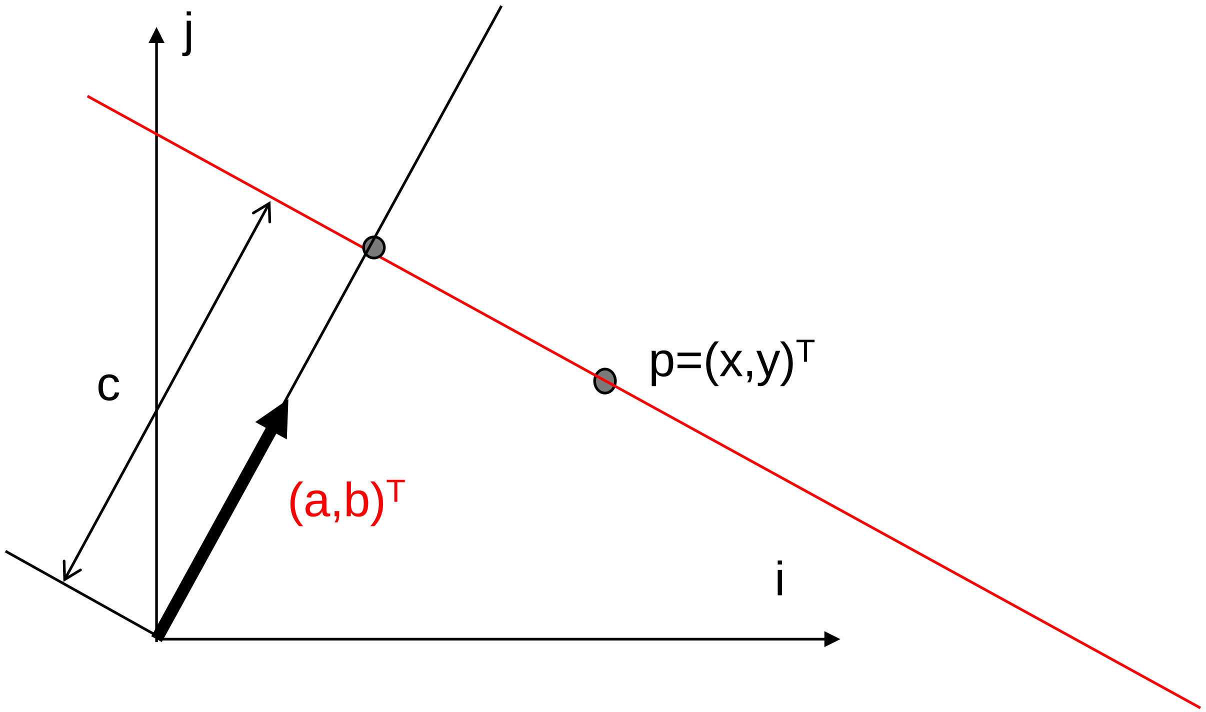
$$\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{\boldsymbol{x}},$$

$\bar{\boldsymbol{x}} = (x, y, 1)$ is the *augmented vector.*

$\tilde{w} = 0$ are called *ideal points* or *points at infinity*

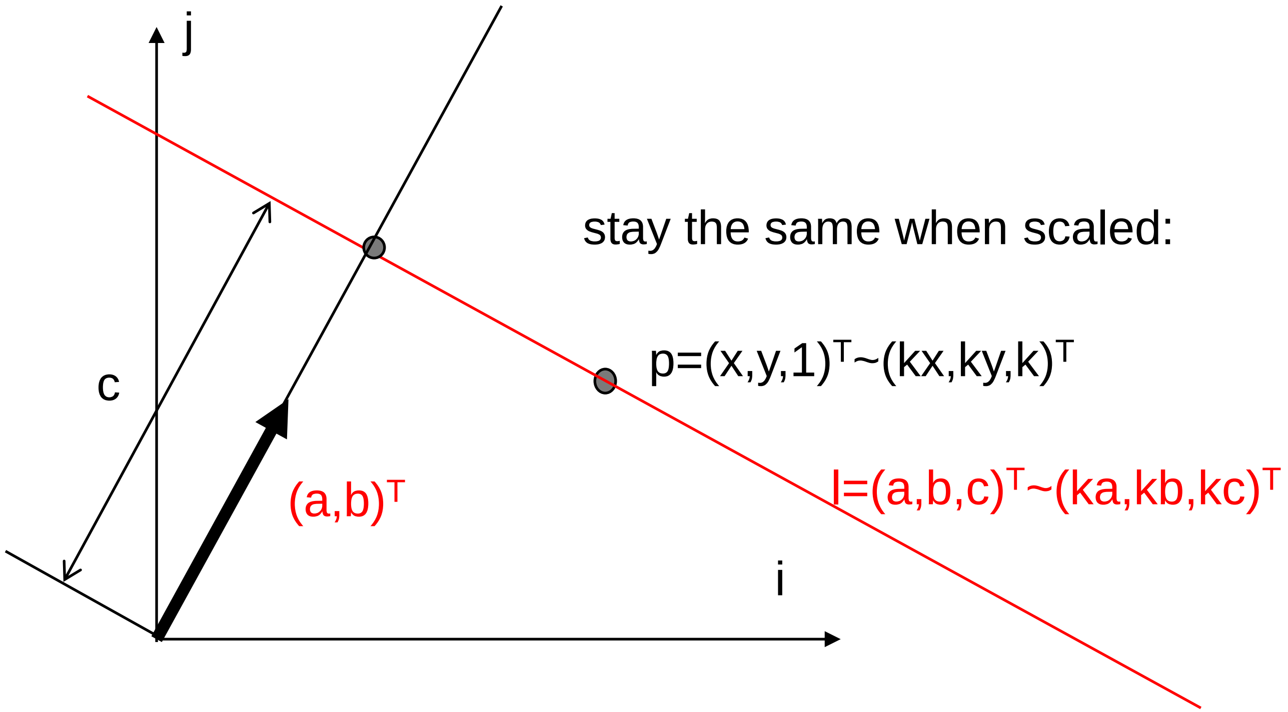
2D Lines

- **Line l = ax+by=c**



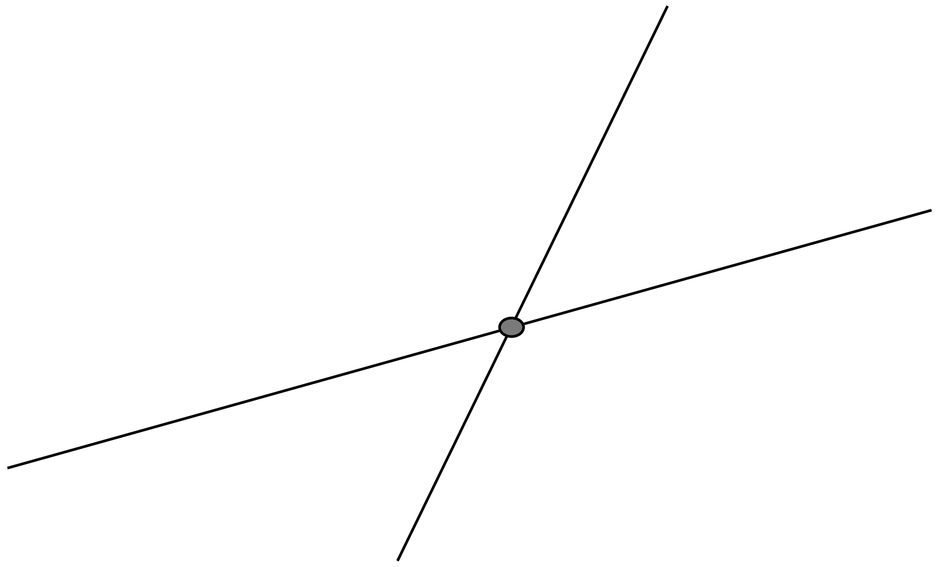
Homogeneous Coordinates

- **Uniform treatment of points and lines**
- **Line-point incidence: $l^T p=0$**

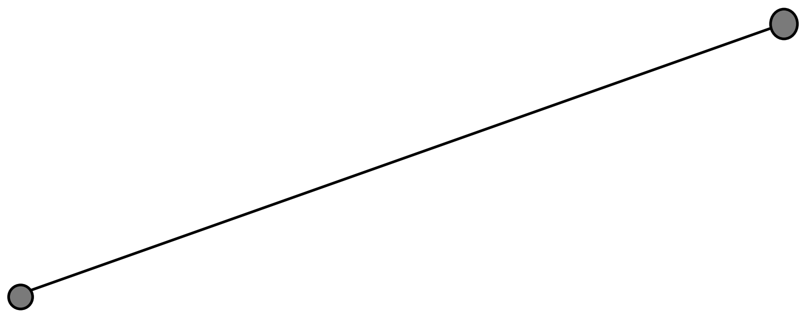


Join = cross product !

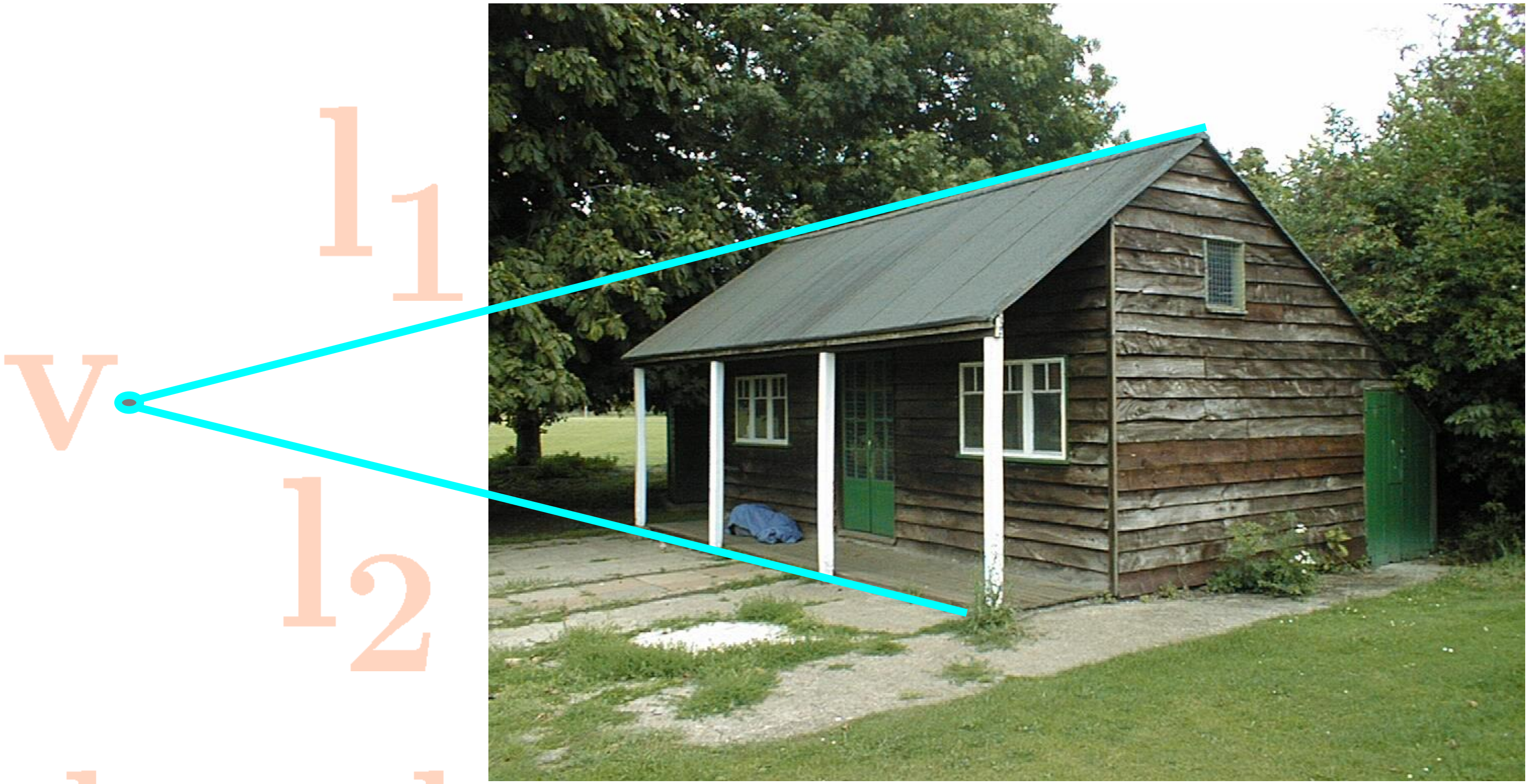
- Join of two lines is a point: $p = l_1 x l_2$



- Join of two points is a line: $l = p_1 x p_2$



Automatic estimation of vanishing points and lines

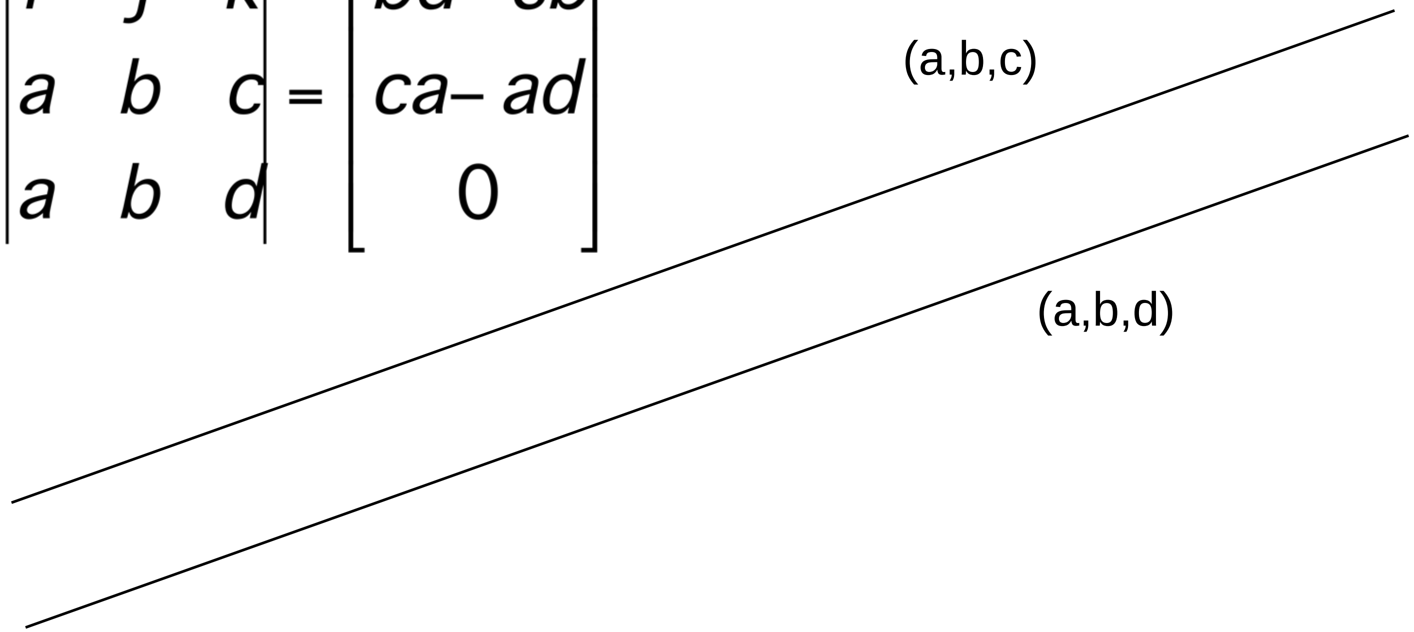


$$\mathbf{v} = \mathbf{l}_1 \times \mathbf{l}_2$$

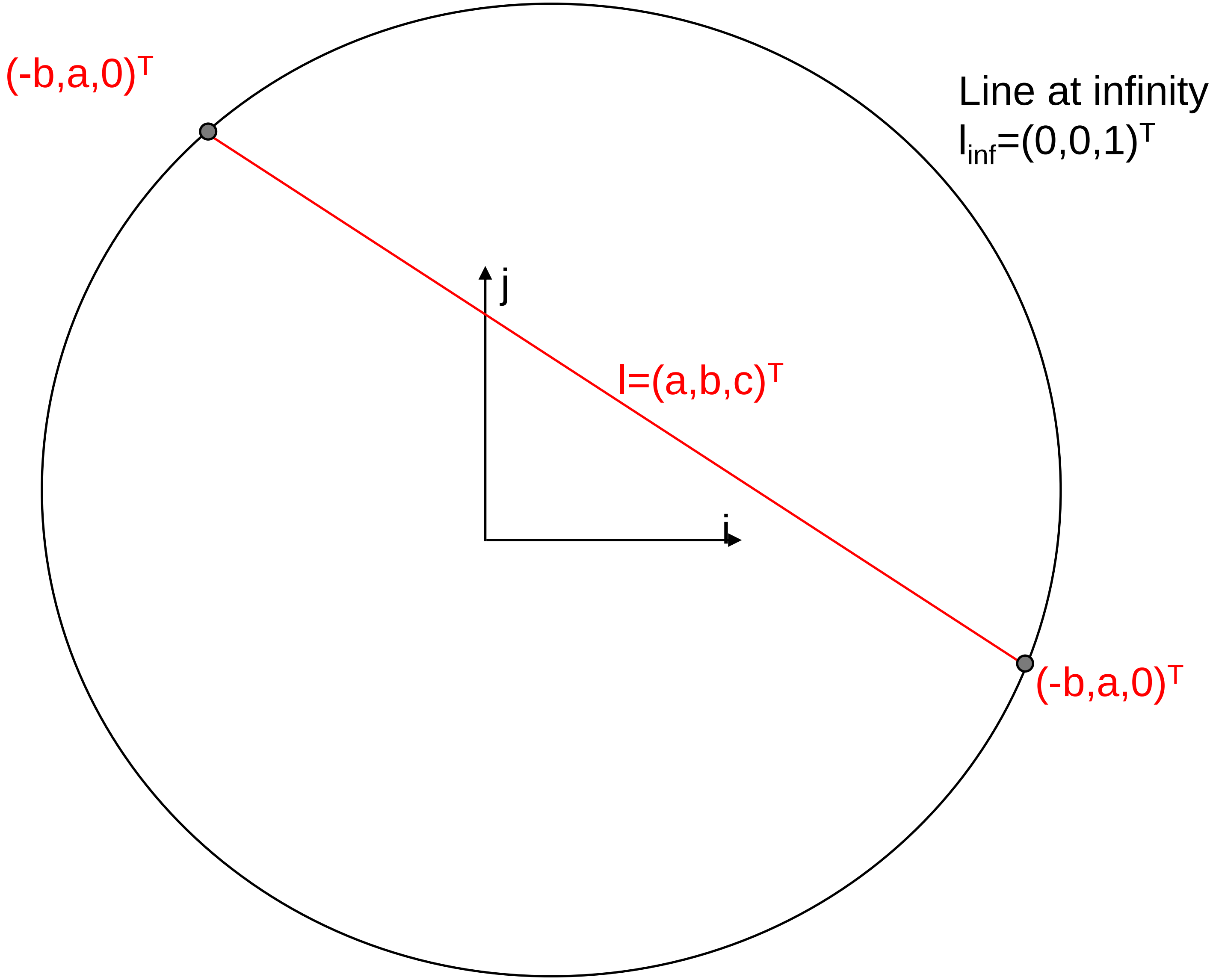
Joining two parallel lines ?

(a,b,c)

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



Points at Infinity !



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

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

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

2.1.1 Geometric Primitives

homogeneous augmented

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

2.1.1 Geometric Primitives

homogeneous augmented

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

2.1.1 Geometric Primitives

homogeneous augmented

- **2D points:** *(x,y)*, $\tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{w}) = \tilde{w}(x, y, 1) = \tilde{w}\bar{\boldsymbol{x}}$
- **2D lines:** $\bar{\boldsymbol{x}} \cdot \tilde{\boldsymbol{l}} = ax + by + c = 0$
- **2D conics:** $\tilde{\boldsymbol{x}}^T \boldsymbol{Q} \tilde{\boldsymbol{x}} = 0$
- **3D points:** $\boldsymbol{x} = (x, y, z) \quad \tilde{\boldsymbol{x}} = (\tilde{x}, \tilde{y}, \tilde{z}, \tilde{w})$
- **3D planes:** $\bar{\boldsymbol{x}} \cdot \tilde{\boldsymbol{m}} = ax + by + cz + d = 0$
- **3D lines:**

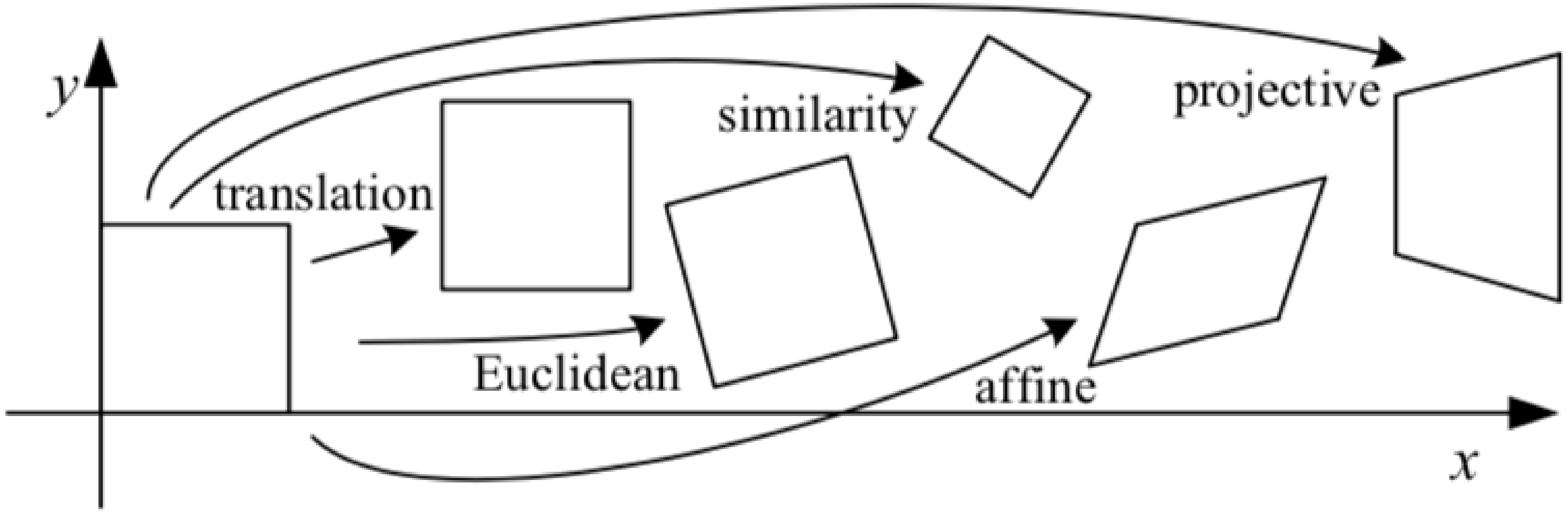
$$\boldsymbol{r} = (1 - \lambda)\boldsymbol{p} + \lambda\boldsymbol{q}$$

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

$$\boldsymbol{r} = \boldsymbol{p} + \lambda\hat{\boldsymbol{d}}.$$

See Chapter 2.1.1 for conics, quadrics, 3D lines

2.1.2: 2D Transformations



2.1.2: 2D Transformations

translation

rotation

aspect

affine

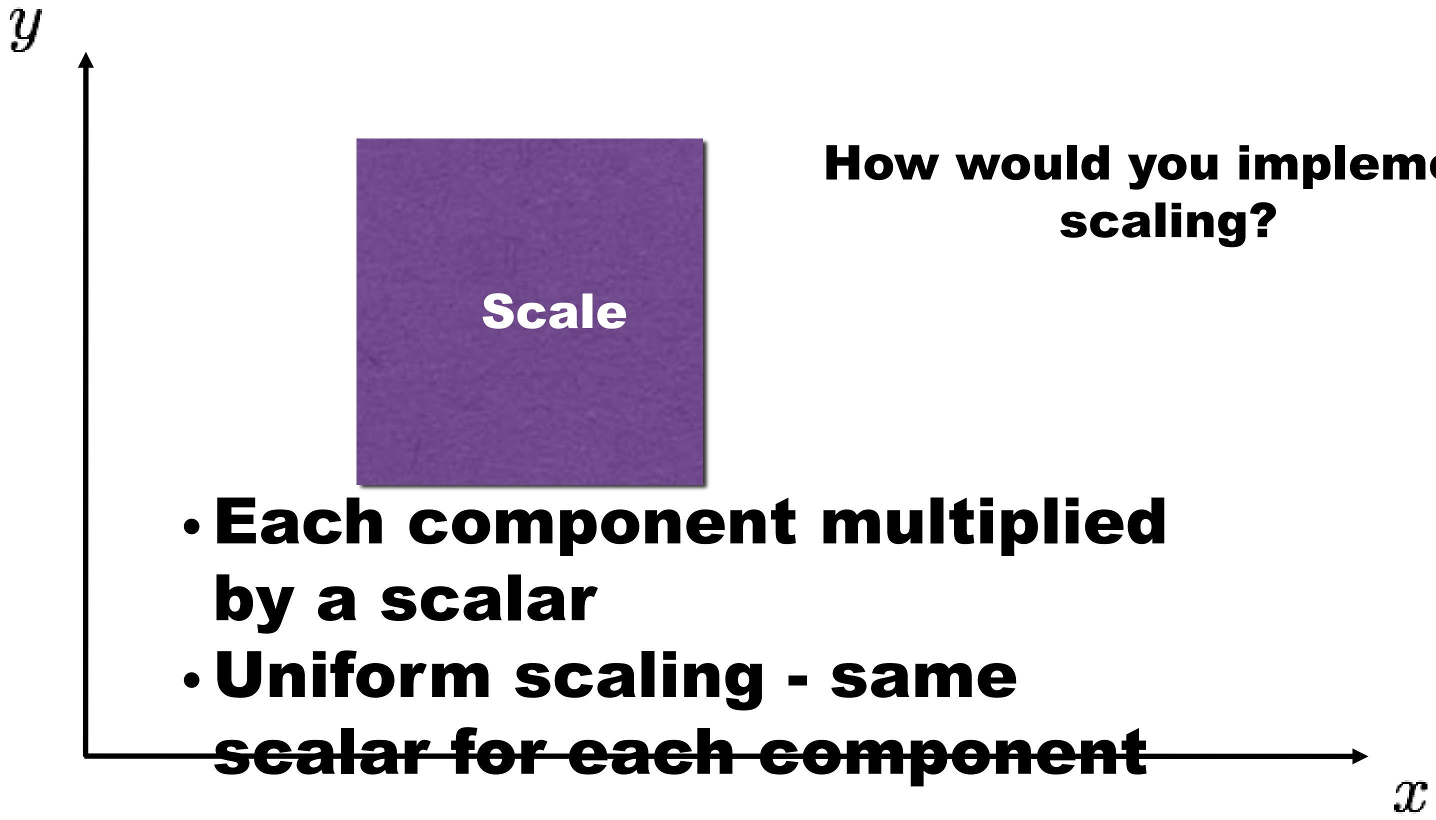
perspective

cylindrical

2D transformation slides adapted from CMU courses by Gkioulekas, Kitani

2D planar transformations

2D planar transformations



How would you implement scaling?

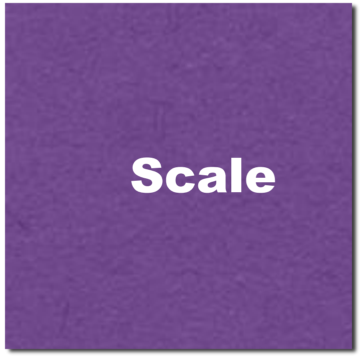
- **Each component multiplied by a scalar**
- **Uniform scaling - same scalar for each component**

2D planar transformations

$$x' = ax$$

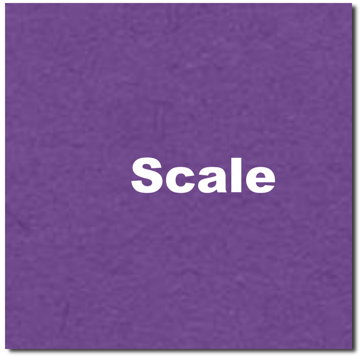
$$y' = by$$

What's the effect of using different scale factors?



- **Each component multiplied by a scalar**
- **Uniform scaling - same scalar for each component**

2D planar transformations



$$x' = ax$$

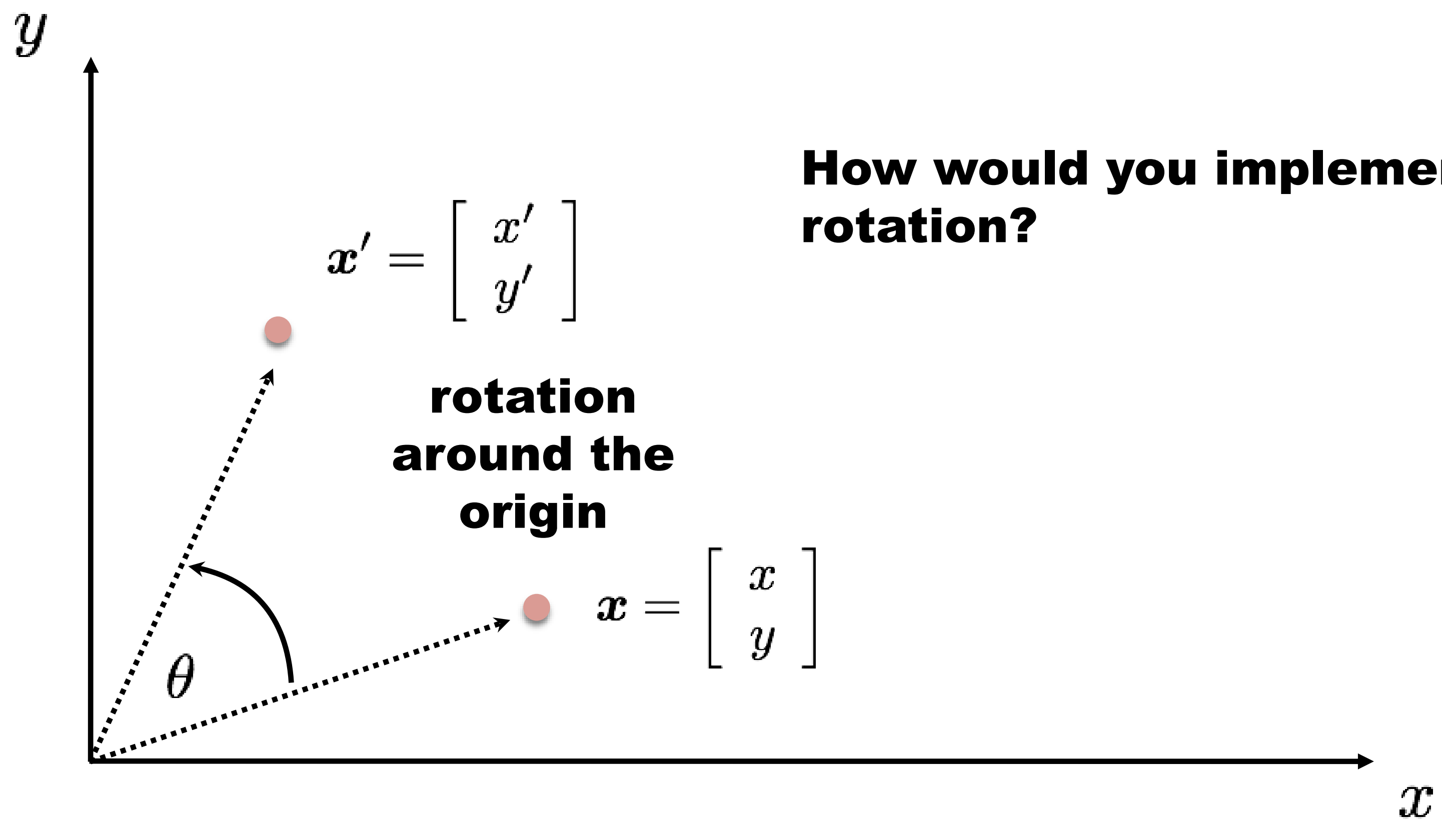
$$y' = by$$

matrix representation of scaling:

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \underbrace{\begin{bmatrix} a & 0 \\ 0 & b \end{bmatrix}}_{\text{scaling matrix S}} \begin{bmatrix} x \\ y \end{bmatrix}$$

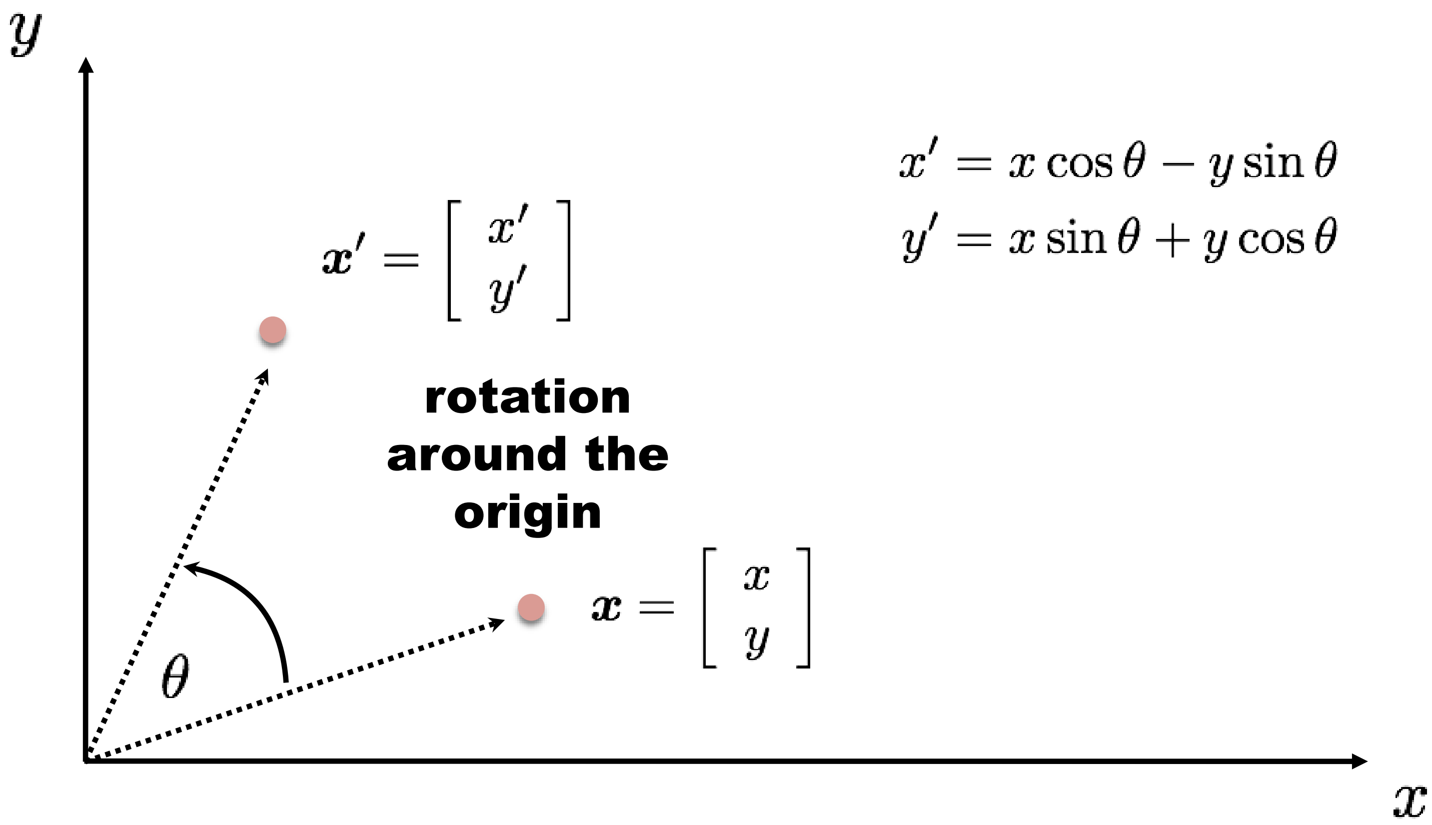
- **Each component multiplied by a scalar**
- **Uniform scaling - same scalar for each component**

2D planar transformations



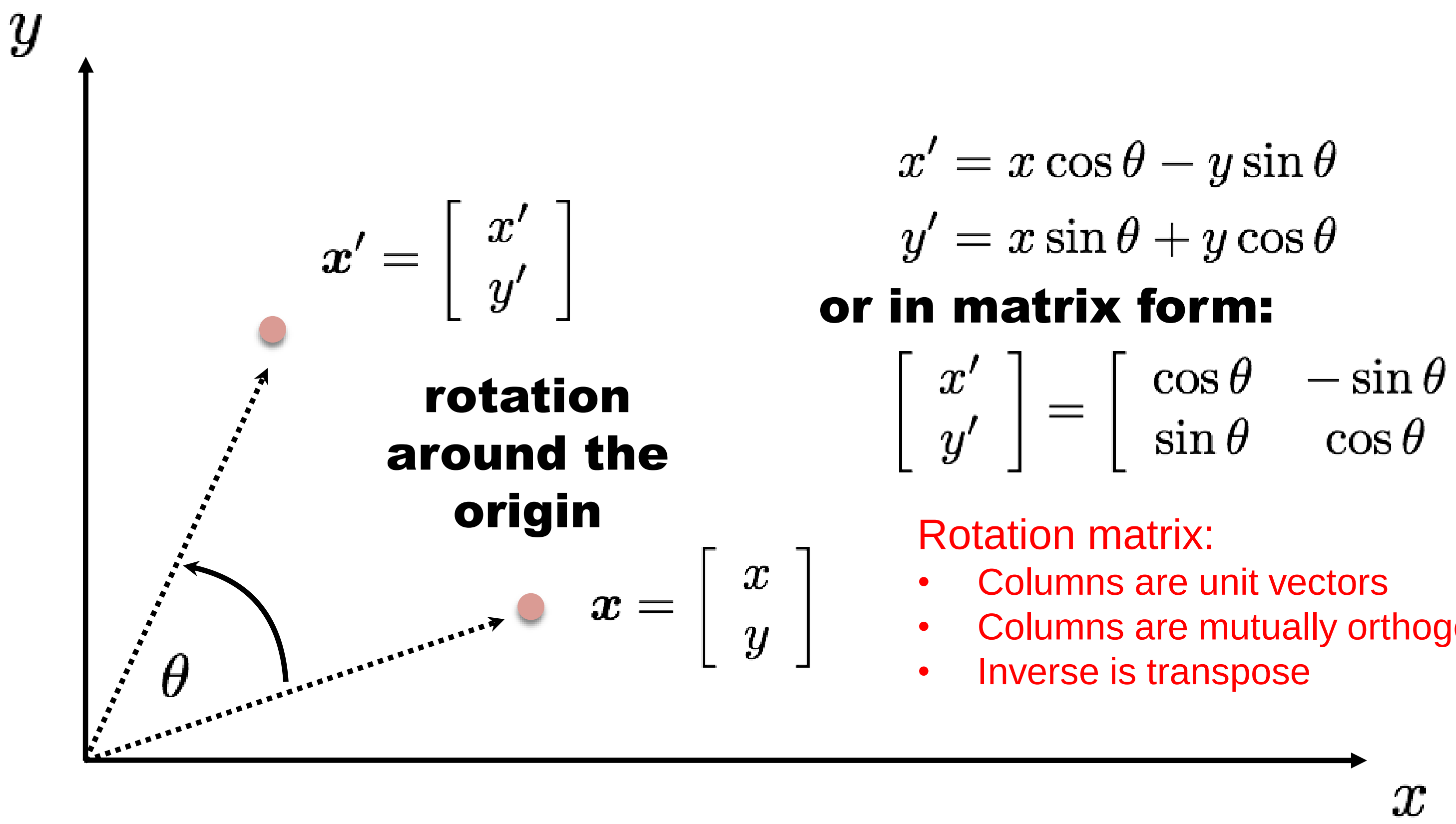
How would you implement rotation?

2D planar transformations



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

2D planar transformations



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

or in matrix form:

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

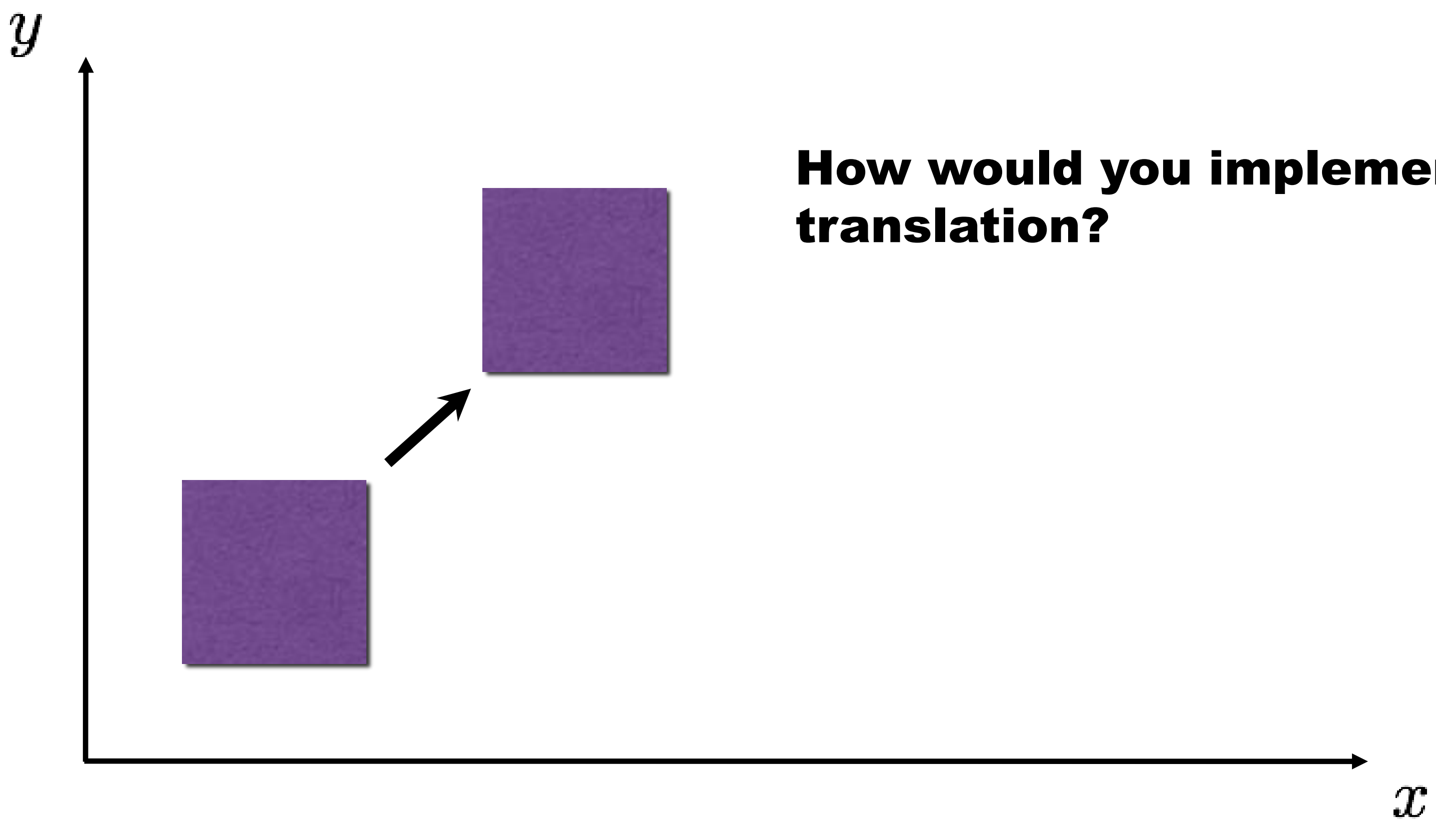
Rotation matrix:

- Columns are unit vectors
- Columns are mutually orthogonal
- Inverse is transpose

2D planar and linear transformations

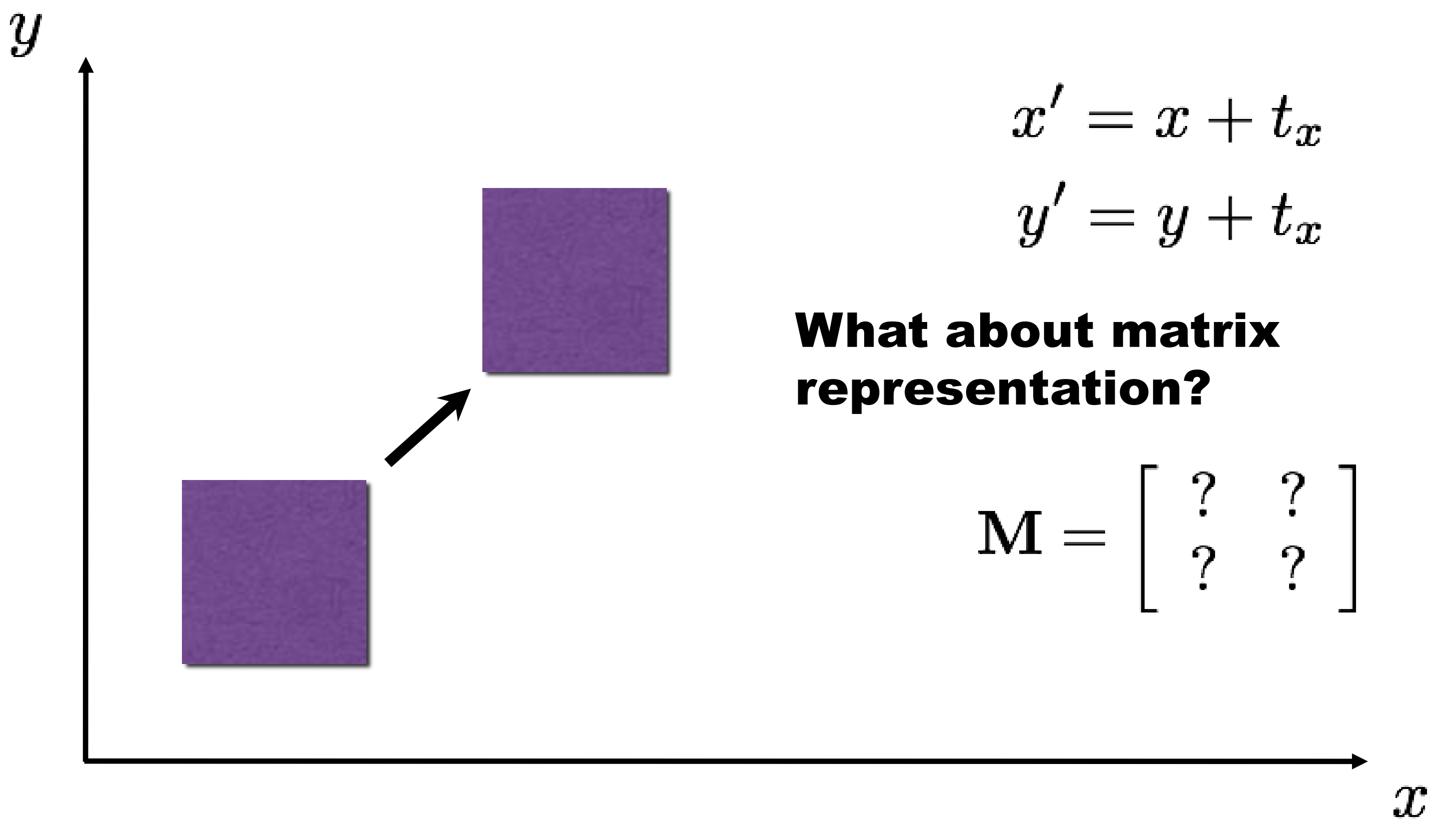
Scale $$\mathbf{M} = \begin{bmatrix} s_x & 0 \\ 0 & s_y \end{bmatrix}$$	**Flip across y** $$\mathbf{M} = \begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix}$$
Rotate $$\mathbf{M} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix}$$	**Flip across origin** $$\mathbf{M} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}$$
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



How would you implement translation?

2D translation



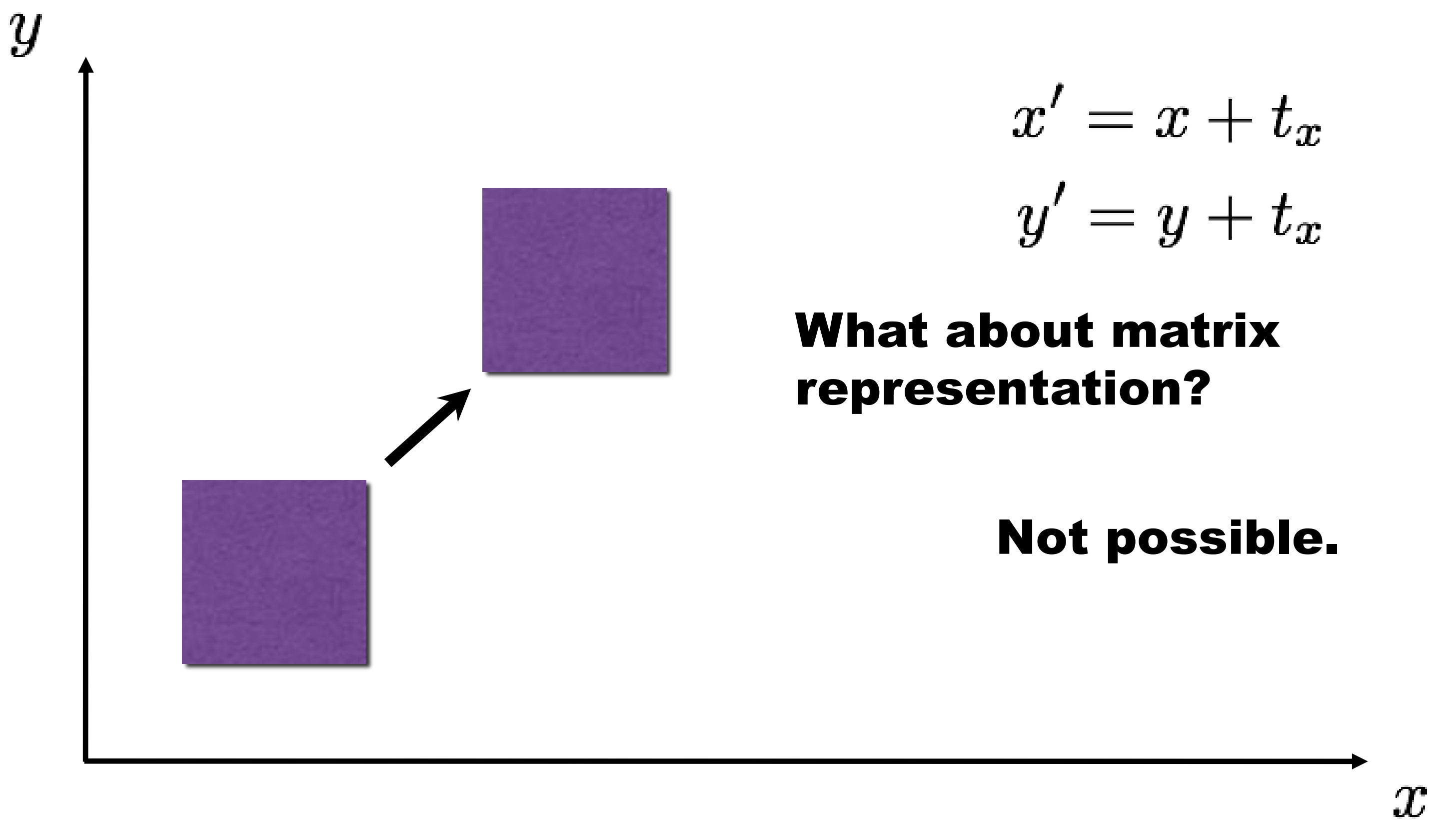
$$x' = x + t_x$$

$$y' = y + t_x$$

What about matrix representation?

$$\mathbf{M} = \begin{bmatrix} ? & ? \\ ? & ? \end{bmatrix}$$

2D translation

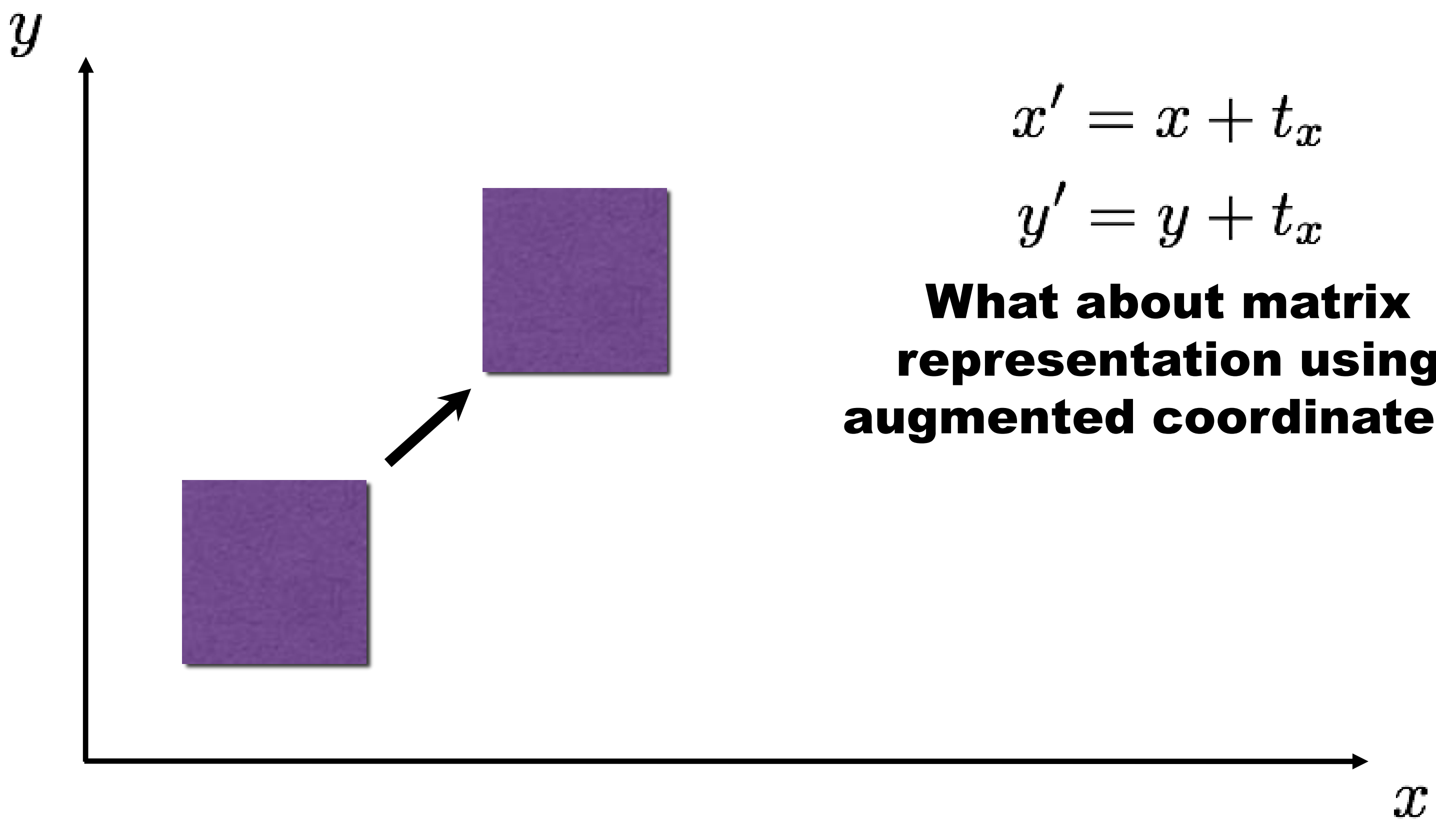


$$x' = x + t_x$$
$$y' = y + t_x$$

What about matrix representation?

Not possible.

2D translation

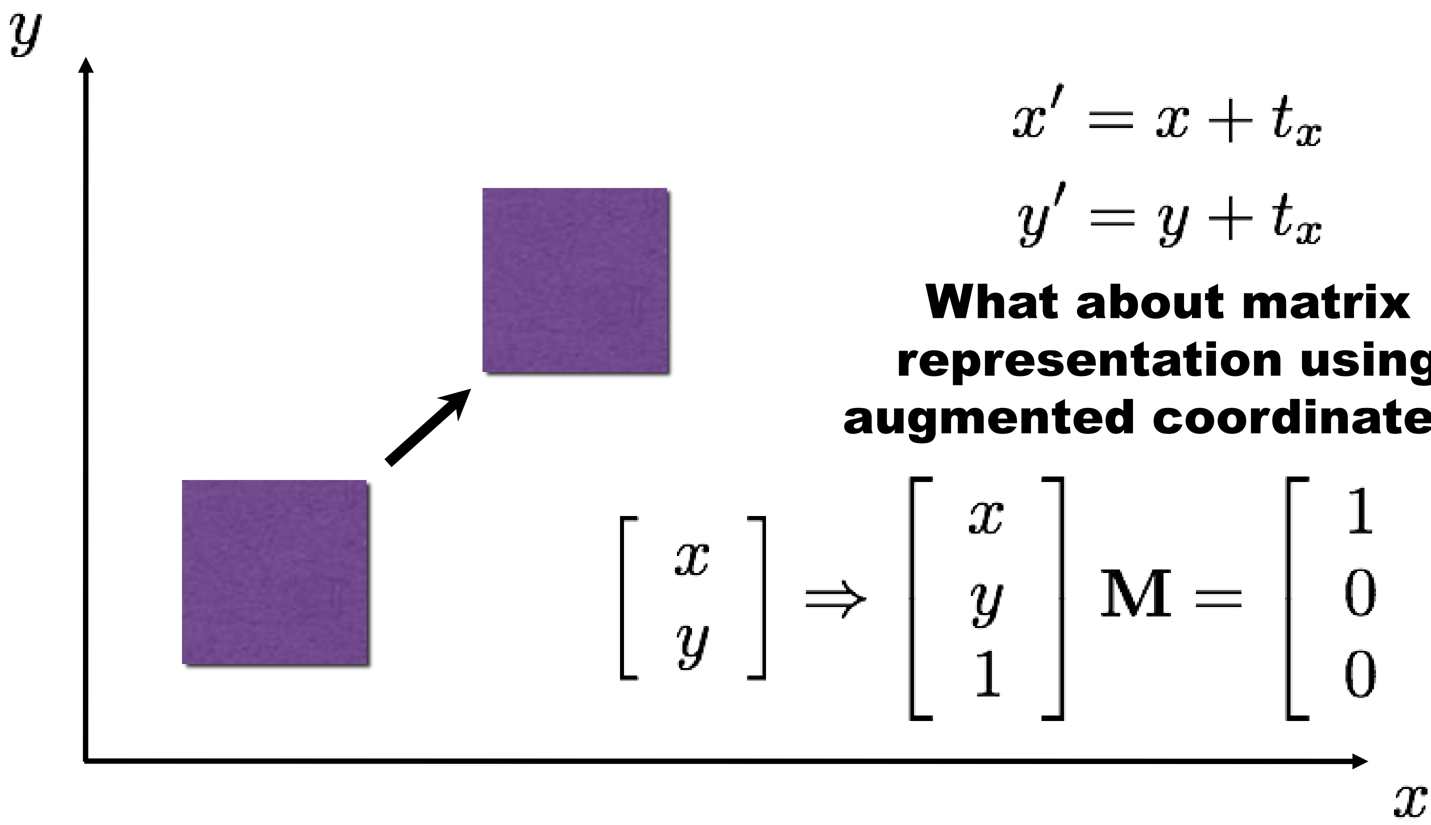


$$x' = x + t_x$$

$$y' = y + t_x$$

What about matrix representation using augmented coordinates?

2D translation



$$x' = x + t_x$$

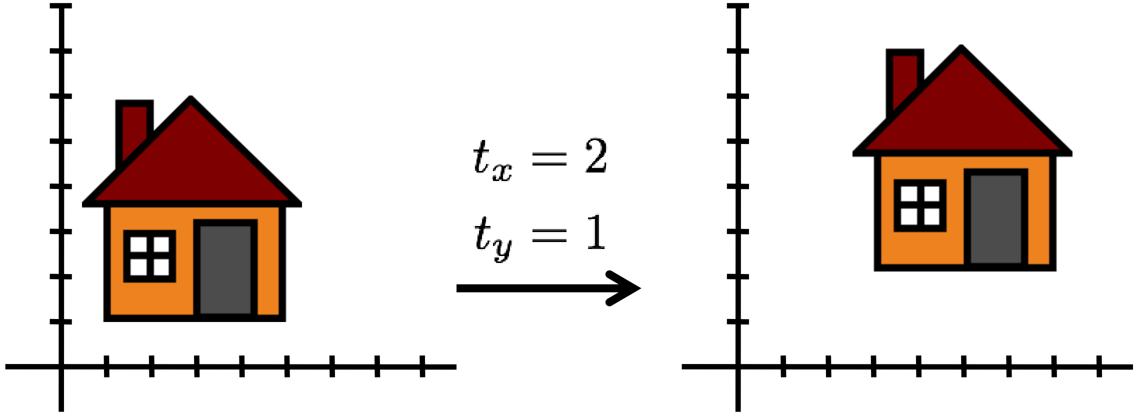
$$y' = y + t_x$$

What about matrix representation using augmented coordinates?

$$\begin{bmatrix} x \\ y \end{bmatrix} \Rightarrow \begin{bmatrix} x \\ y \\ 1 \end{bmatrix} \quad \mathbf{M} = \begin{bmatrix} 1 & 0 & t_x \\ 0 & 1 & t_y \\ 0 & 0 & 1 \end{bmatrix}$$

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**

$$\begin{bmatrix} x & y & w \end{bmatrix}^\top = \lambda \begin{bmatrix} x & y & w \end{bmatrix}^\top$$

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

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

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

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

scaling

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

rotation

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

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

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

scaling

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

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

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

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:

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

p’ = ? ? ?

Matrix composition

Transformations can be combined by matrix multiplication:

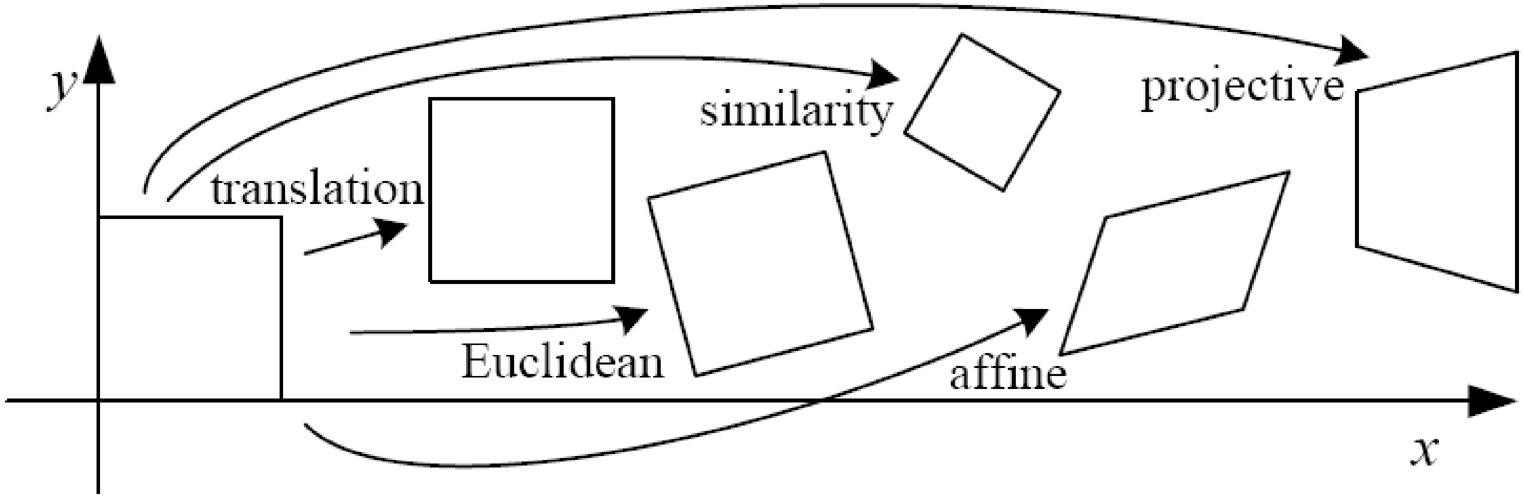
$$\begin{bmatrix} x' \\ y' \\ w' \end{bmatrix} = \left(\begin{bmatrix} 1 & 0 & tx \\ 0 & 1 & ty \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\Theta & -\sin\Theta & 0 \\ \sin\Theta & \cos\Theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} sx & 0 & 0 \\ 0 & sy & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)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

p' = translation(t_x,t_y) rotation(θ) scale(s,s)

Does the multiplication order matter?

Classification of 2D transformations

Classification of 2D transformations



Classification of 2D transformations

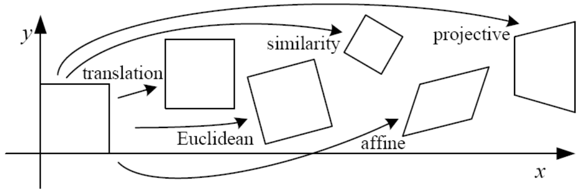
Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} & \vert & \boldsymbol{t} \end{bmatrix}$	?
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} & \vert & \boldsymbol{t} \end{bmatrix}$	?
similarity	$\begin{bmatrix} s\boldsymbol{R} & \vert & \boldsymbol{t} \end{bmatrix}$	?
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}$	?
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}$	?

Translation

Translat ion:

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

How many degrees of freedom?

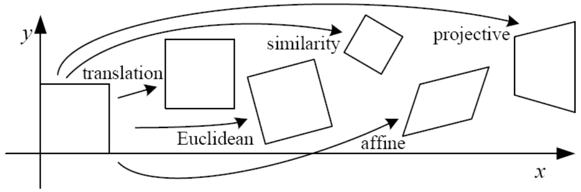


Euclidean/Rigid

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?

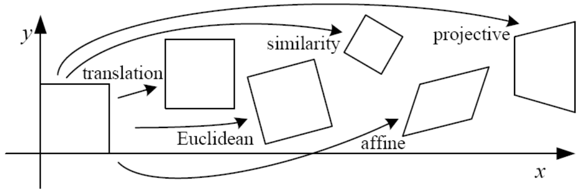


Affine

**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

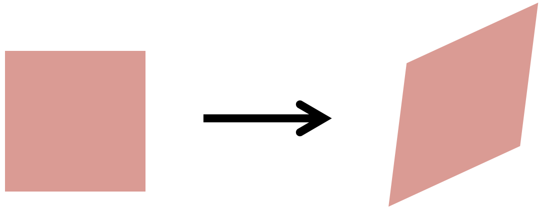
Affine transformations are combinations of

- **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

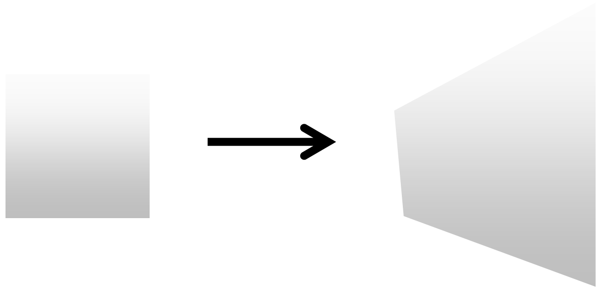
- **affine transformations;**
- **+ projective wraps**

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

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

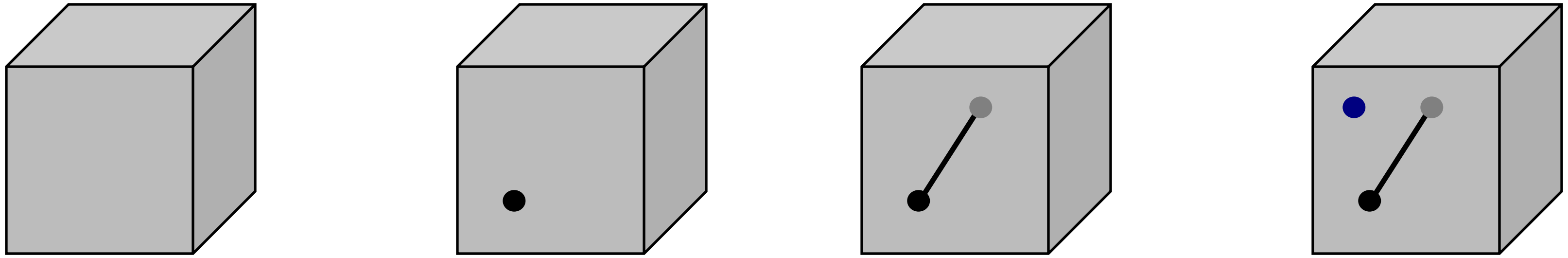
Name	Matrix	# D.O.F.
translation	$\begin{bmatrix} \boldsymbol{I} & \vert & \boldsymbol{t} \end{bmatrix}$	?
rigid (Euclidean)	$\begin{bmatrix} \boldsymbol{R} & \vert & \boldsymbol{t} \end{bmatrix}$	?
similarity	$\begin{bmatrix} s\boldsymbol{R} & \vert & \boldsymbol{t} \end{bmatrix}$	?
affine	$\begin{bmatrix} \boldsymbol{A} \end{bmatrix}$	?
projective	$\begin{bmatrix} \tilde{\boldsymbol{H}} \end{bmatrix}$	?

Classification of 2D transformations

Name	Matrix	# D.O.F.
translation	$\left[\begin{array}{c\|c} \boldsymbol{I} & \boldsymbol{t} \end{array}\right]$	2
rigid (Euclidean)	$\left[\begin{array}{c\|c} \boldsymbol{R} & \boldsymbol{t} \end{array}\right]$	3
similarity	$\left[\begin{array}{c\|c} s\boldsymbol{R} & \boldsymbol{t} \end{array}\right]$	4
affine	$\left[\begin{array}{c} \boldsymbol{A} \end{array}\right]$	6
projective	$\left[\begin{array}{c} \tilde{\boldsymbol{H}} \end{array}\right]$	8

2.1.3: 3D Transformations

- **Need a way to specify the six degrees-of-freedom of a rigid body.**
- **Why are their 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)

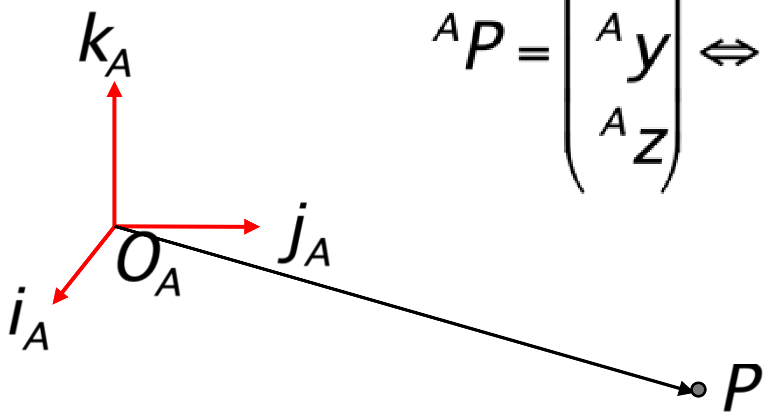
Third point adds one more DOF, for rotation around line

Notations

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

- Example :

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



Translation

- **Using augmented/homogeneous coordinates, translation is expressed as a matrix multiplication.**

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

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

Rotation in homogeneous coordinates

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

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

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

- **R is a rotation matrix:**
  - Columns are unit vectors
  - Columns are mutually orthogonal
  - Inverse is transpose

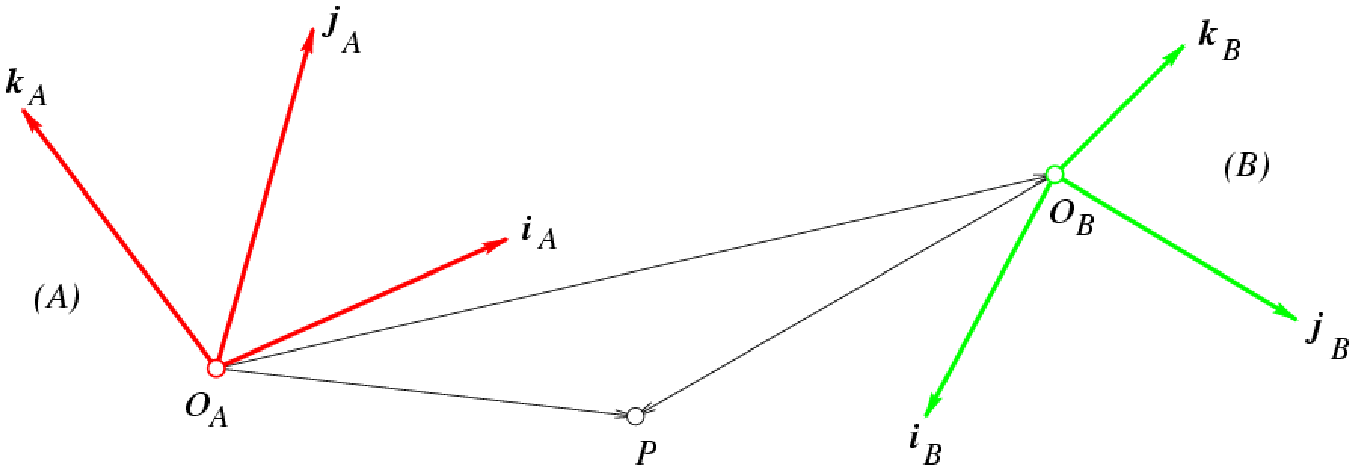
3 D Rotations

- **Using Euler Angles**

rotation along cardinal axes eg x,y,z. Need to keep the order in mind. Usually where hard because the euler angles change a lot for the small rotations. Used in Robotic Transformation. Sometimes useful when you are referring to situations of PAN TILT and HEAD.

Using

3D Rigid transformations



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

3D Rigid transformations

- **Unified treatment using homogeneous coordinates.**

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

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

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

Hierarchy of 3D Transforms

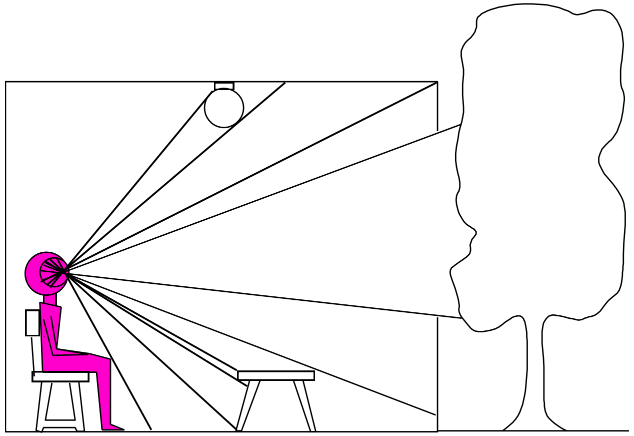
- **Subgroup Structure:**
  - Translation (? DOF)
  - Rigid 3D (? DOF)
  - Affine (? DOF)
  - Projective (? DOF)

Hierarchy of 3D Transforms

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

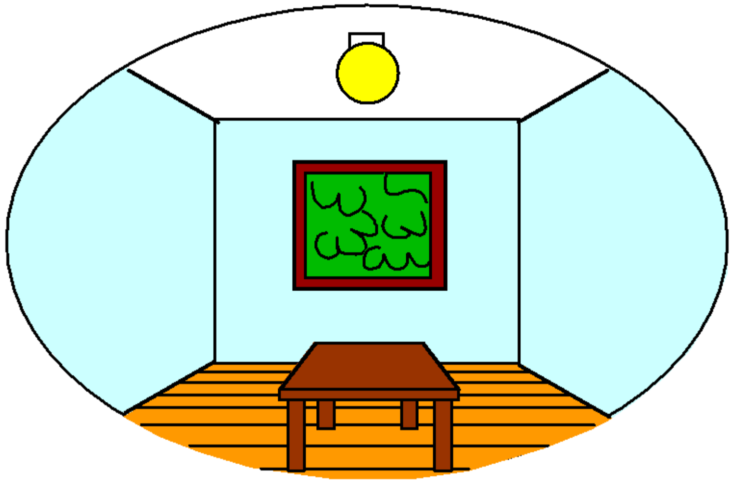
2.1.5: 3D to 2D: Projection

3D world

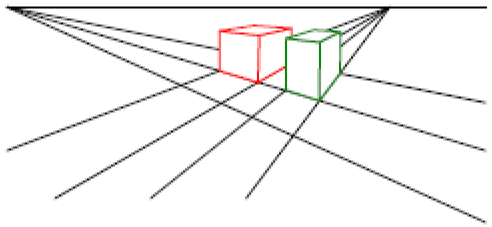


Point of observation

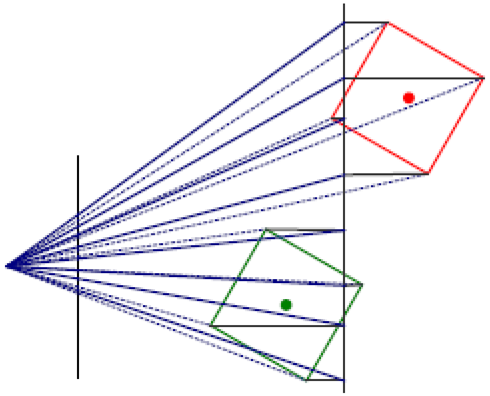
2D image



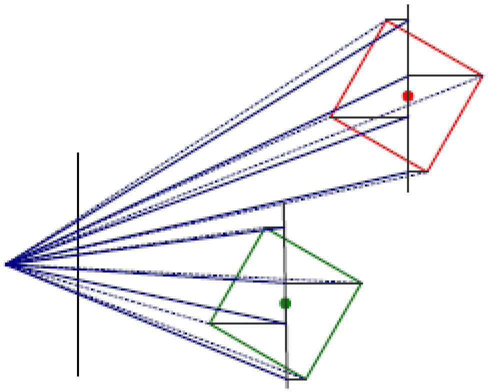
Figures © Stephen E. Palmer, 2002



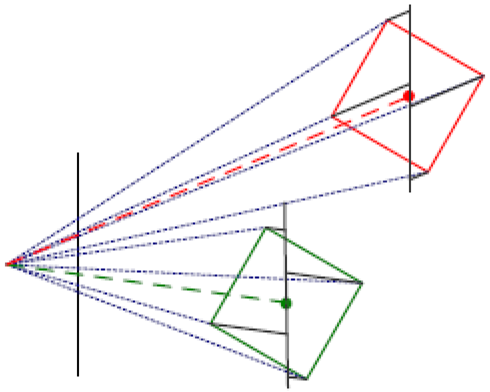
(a) 3D view



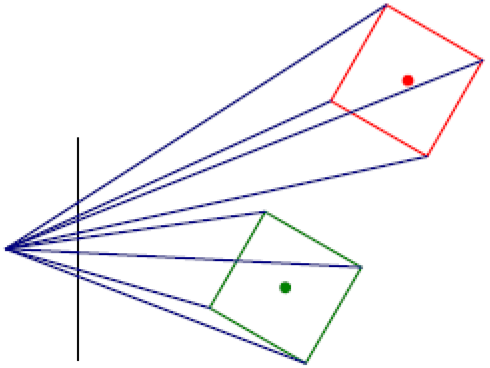
(b) orthography



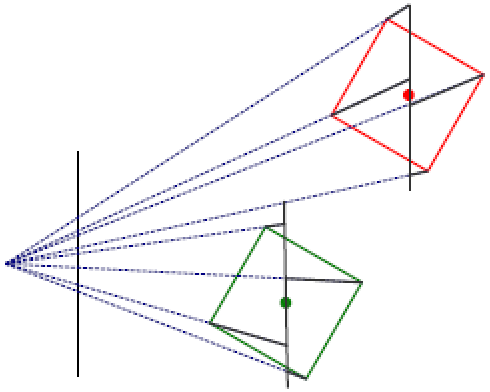
(c) scaled orthography



(d) para-perspective

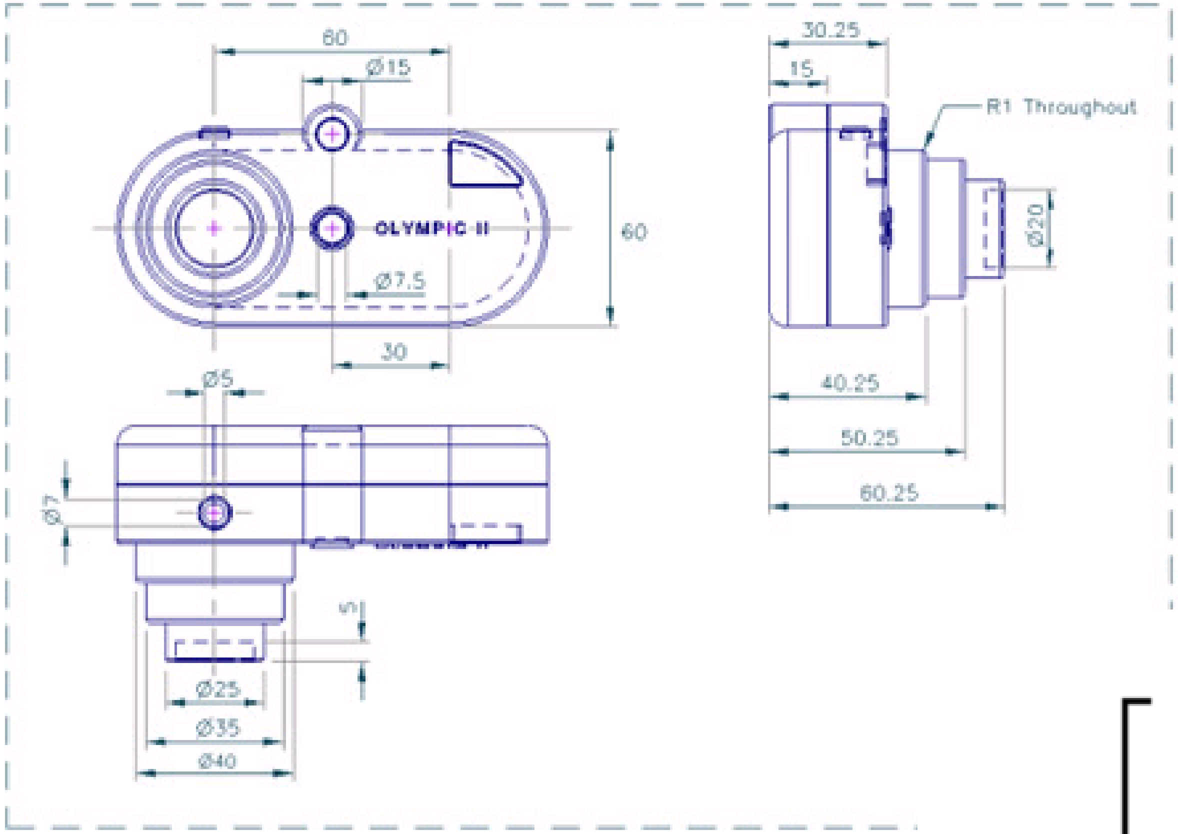


(e) perspective



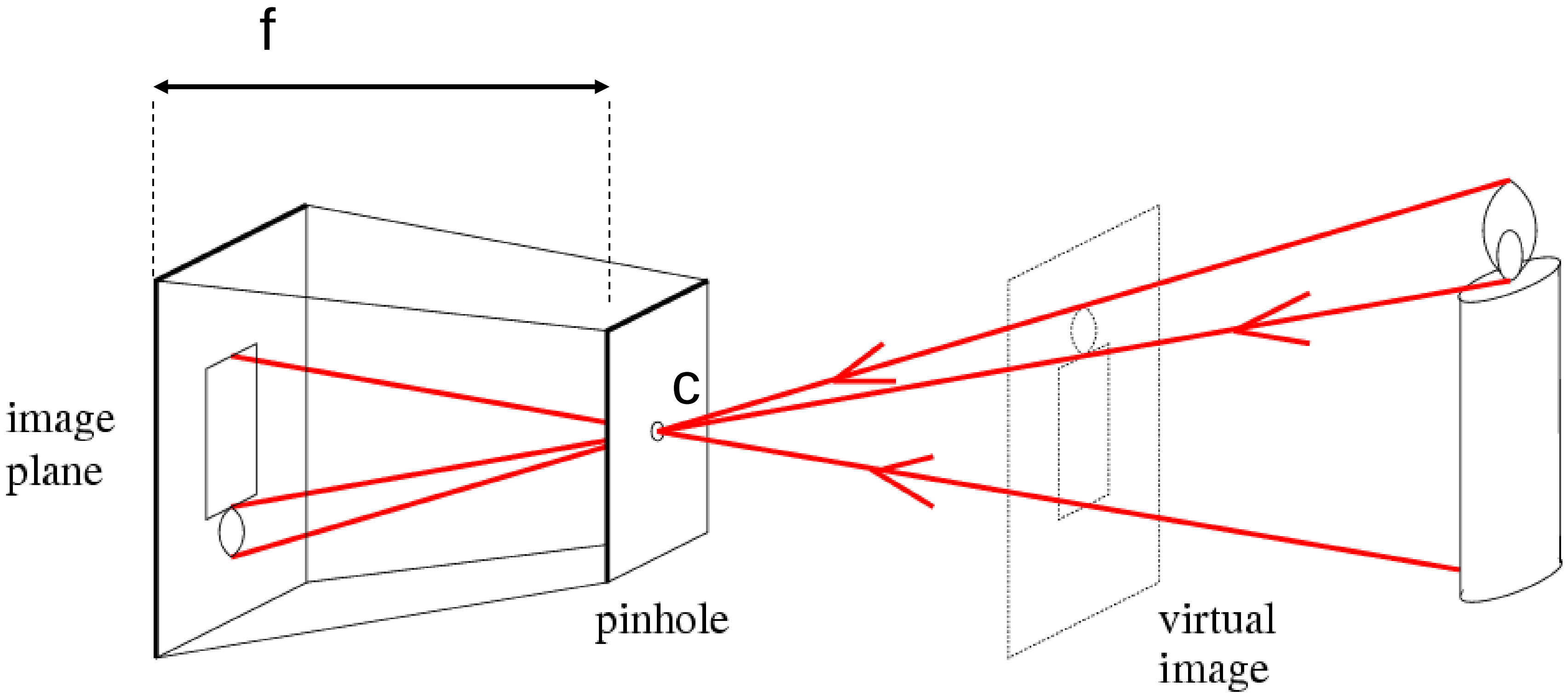
(f) object-centered

Orthographic Projection



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

Pinhole camera



f = focal length
c = center of the camera

Figure from Forsyth

Camera obscura: the pre-camera

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

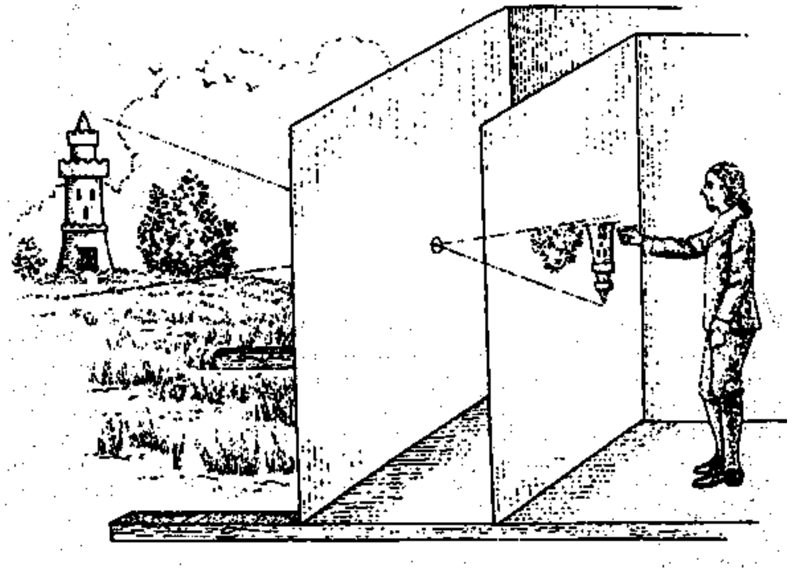
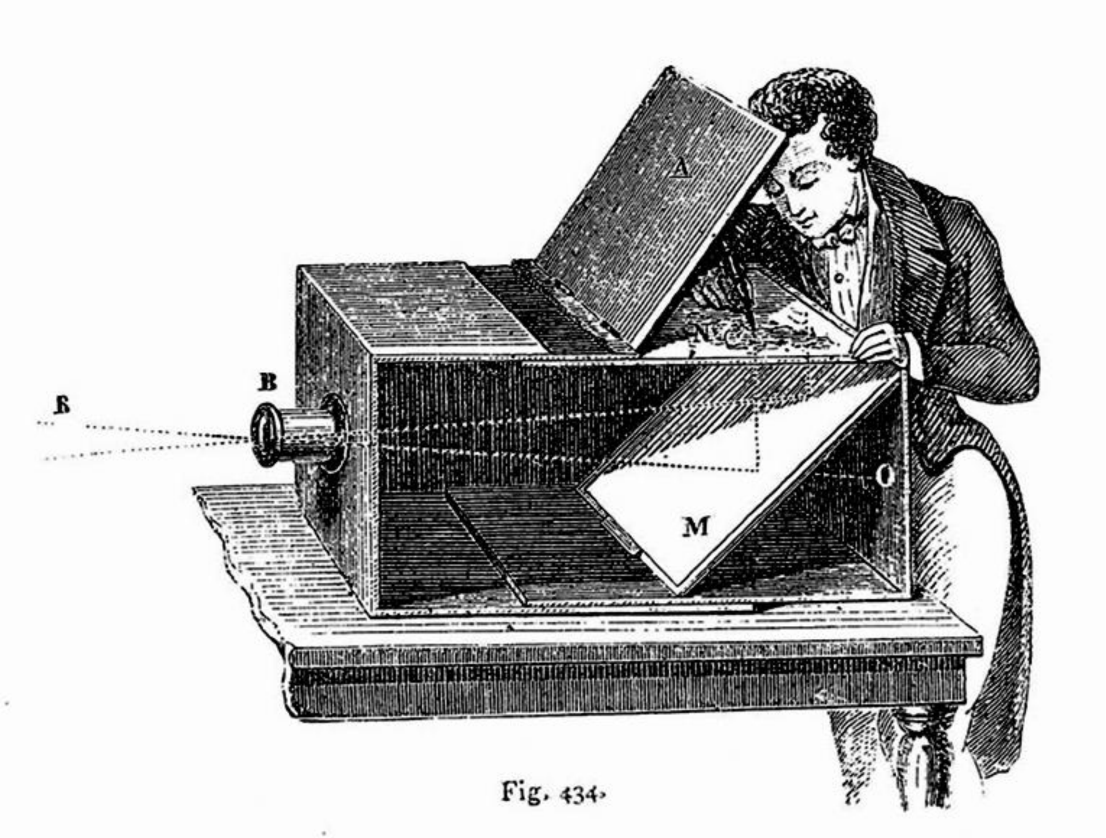


Illustration of Camera Obscura

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

Slide source: Seitz

Projection can be tricky…

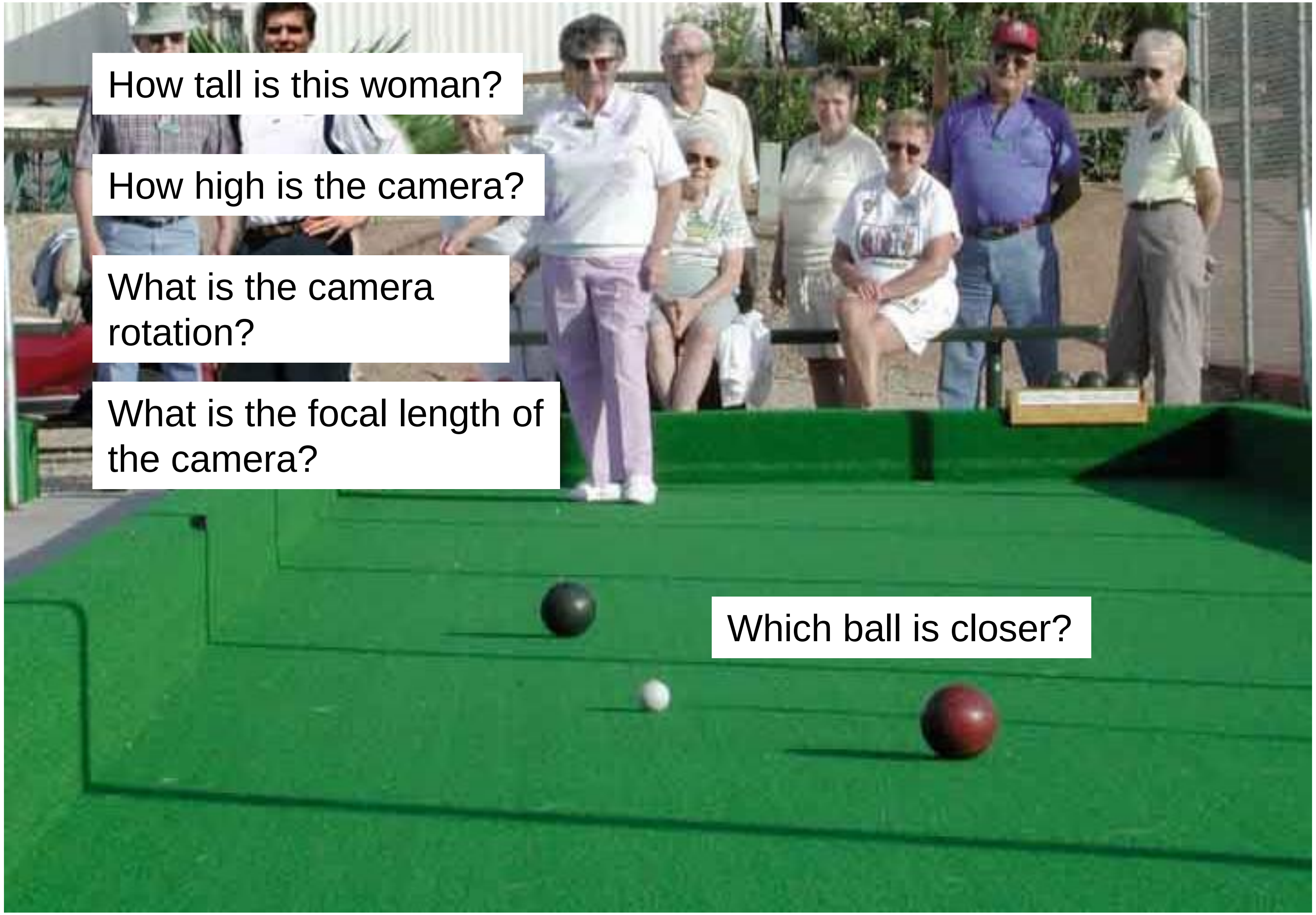
Slide source: Seitz

Projection can be tricky…

AMRITA
VISHWA VIDYAPEETHAM


WÓDKA
Cuisinart
SNOW CRYSTALS
STOWASSER
MATHEMATICS
MAN AND SPACE
DRUGS
STACK-ON

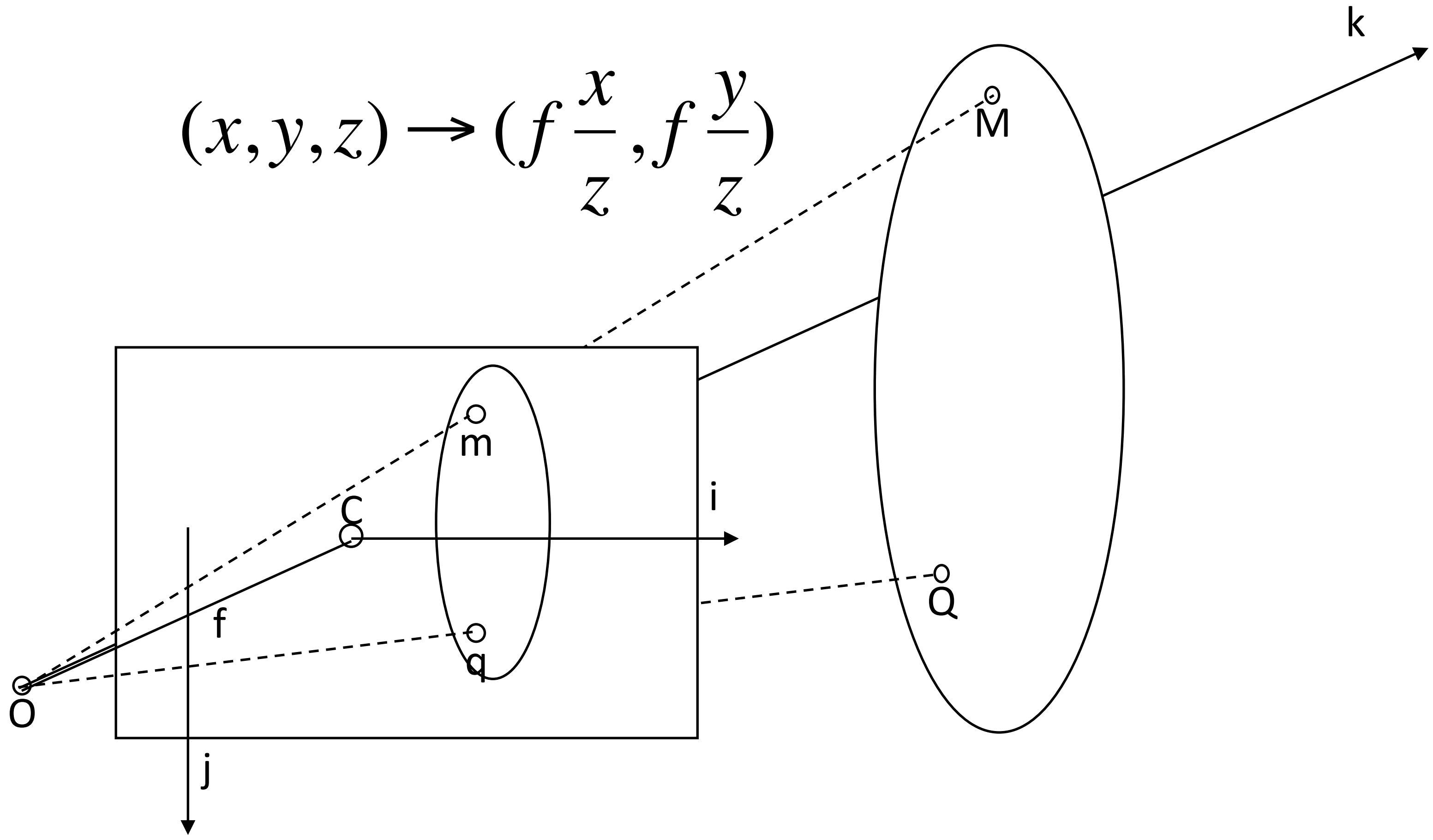
Camera and World Geometry



Pinhole Camera

- **Fundamental equation:**

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



Homogeneous Coordinates

Linear transformation of homogeneous (projective) coordinates

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

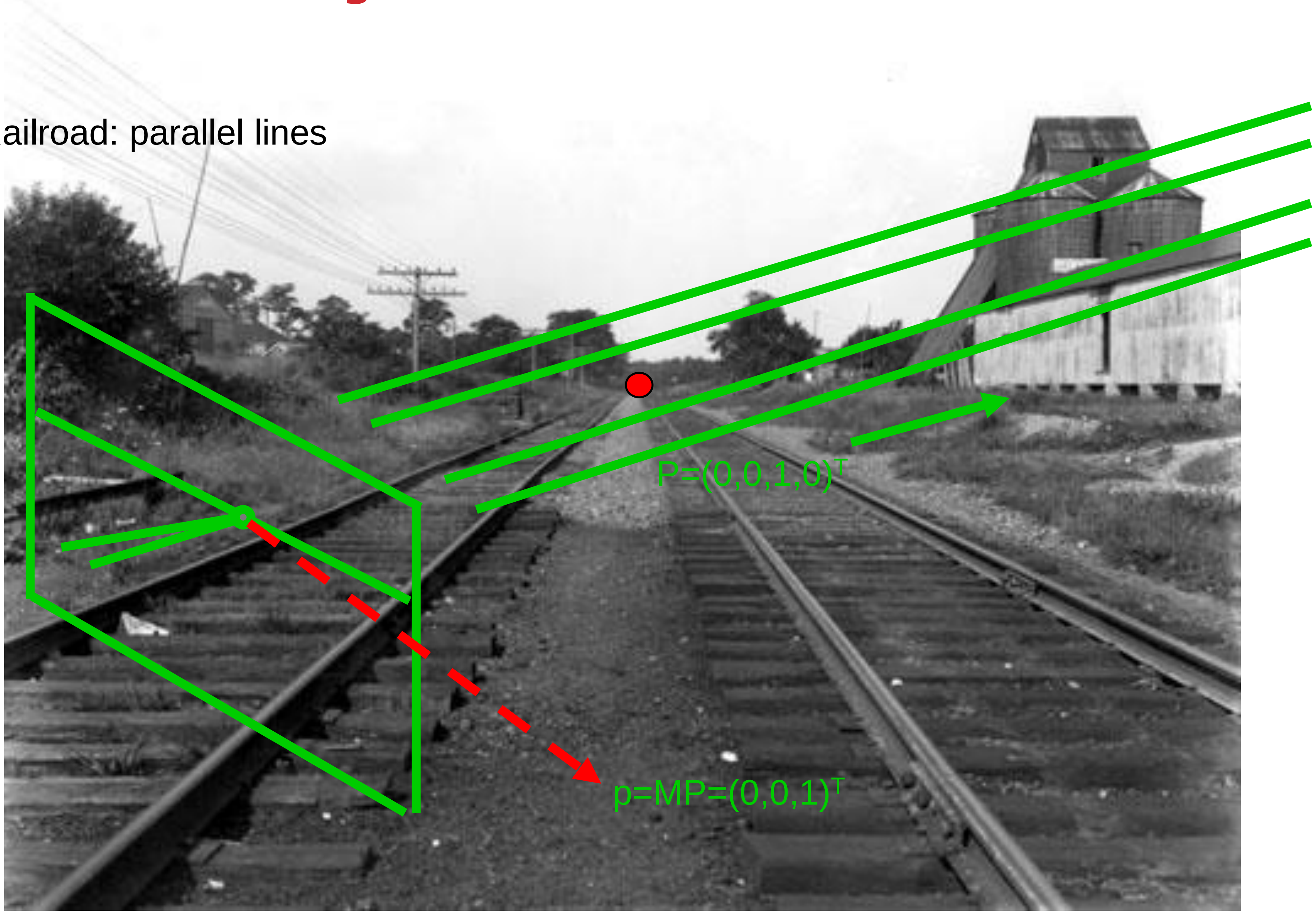
Recover image (Euclidean) coordinates by normalizing:

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

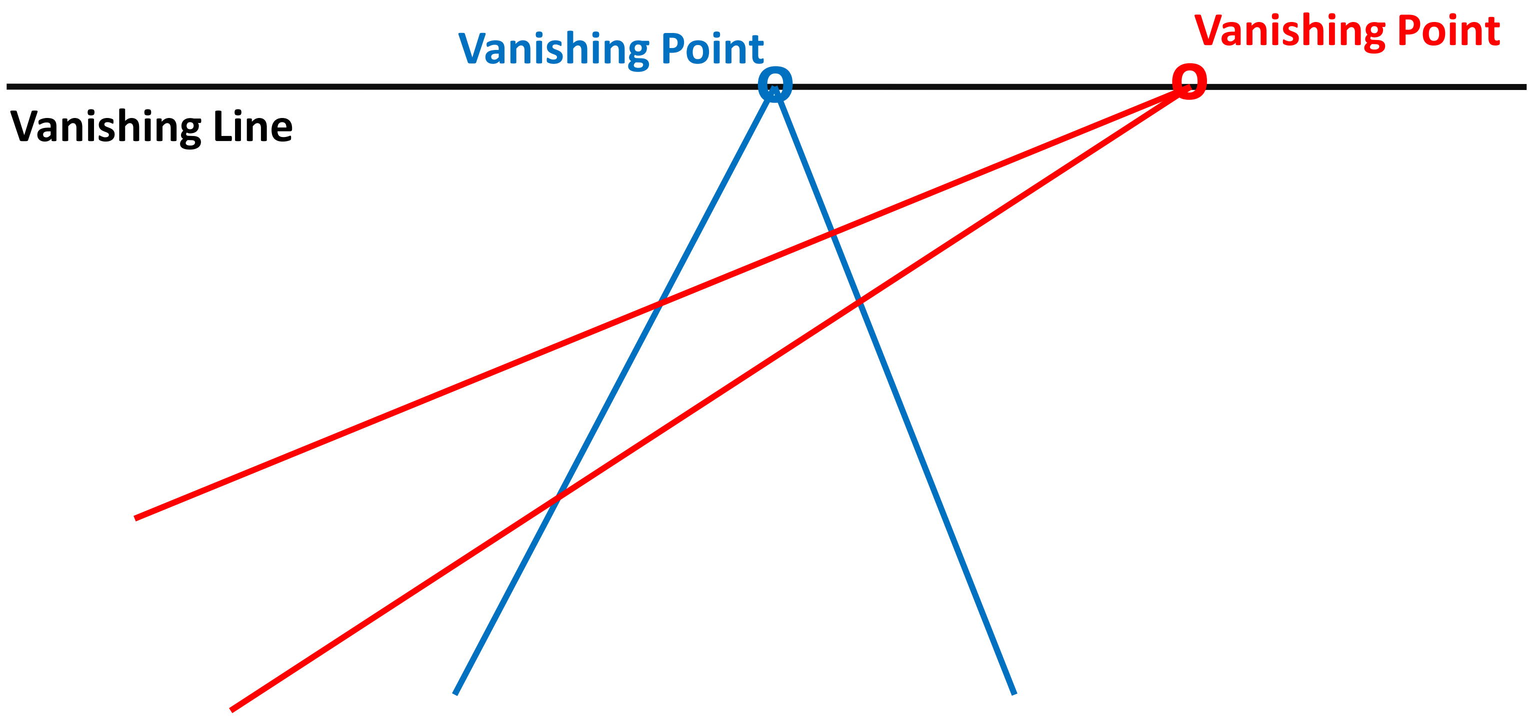
$$\hat{v} = \frac{v}{w} = \frac{Y}{Z}$$

We can see infinity !

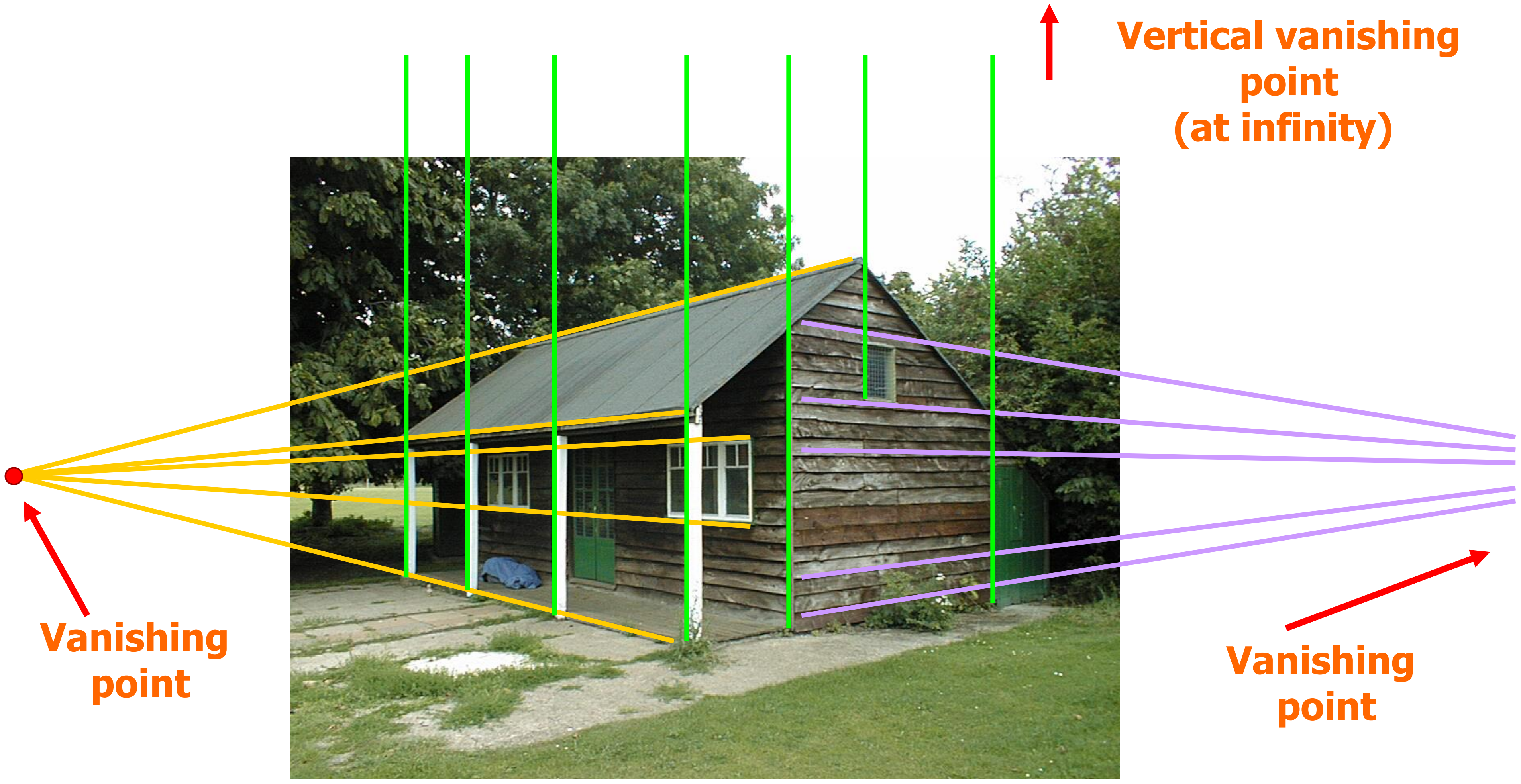
Railroad: parallel lines



Vanishing points and lines

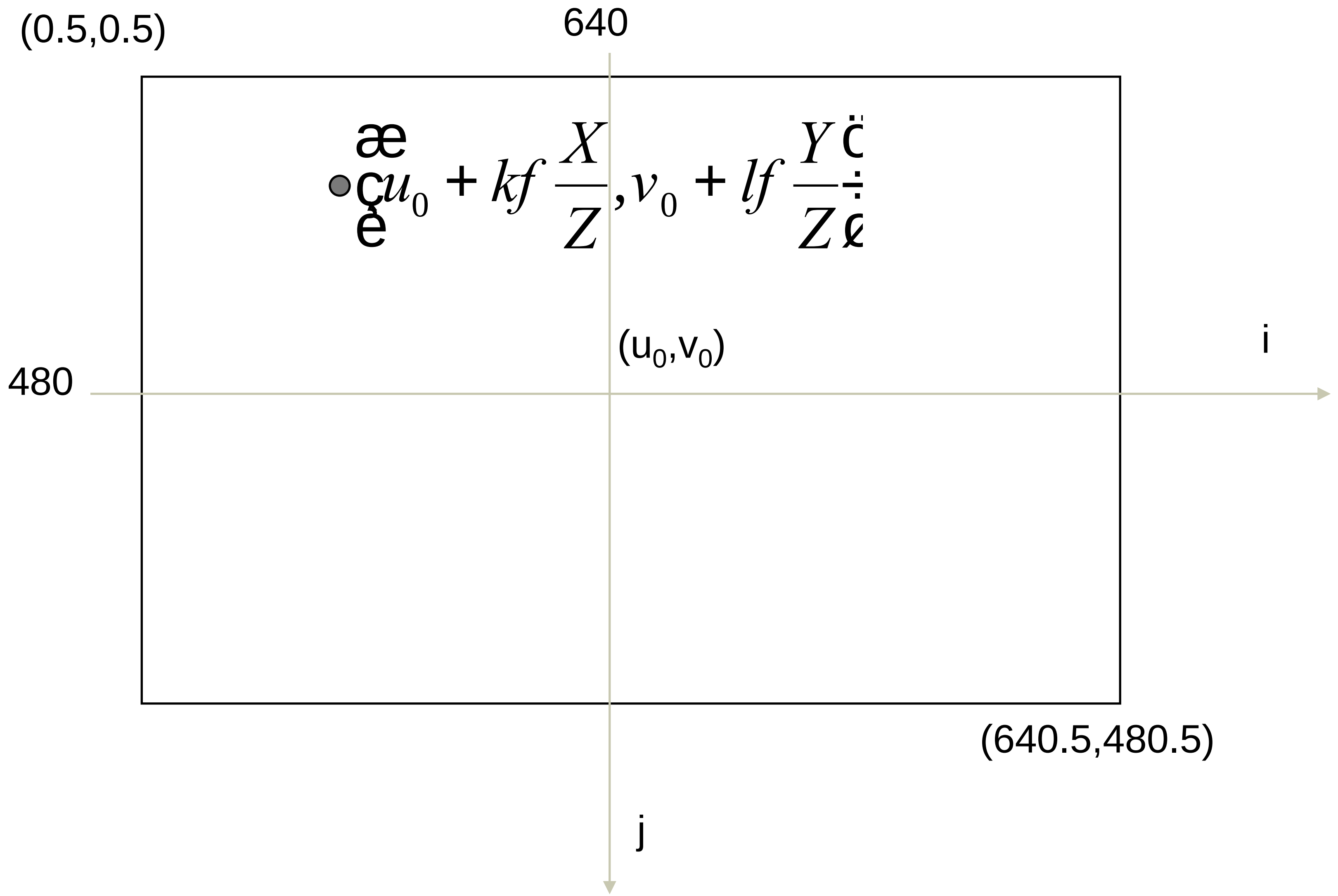


Vanishing points and lines



Slide from Efros, Photo from Criminisi

Pixel coordinates in 2D



Intrinsic Calibration

3×3 Calibration Matrix K

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

skew

Recover image (Euclidean) coordinates by normalizing :

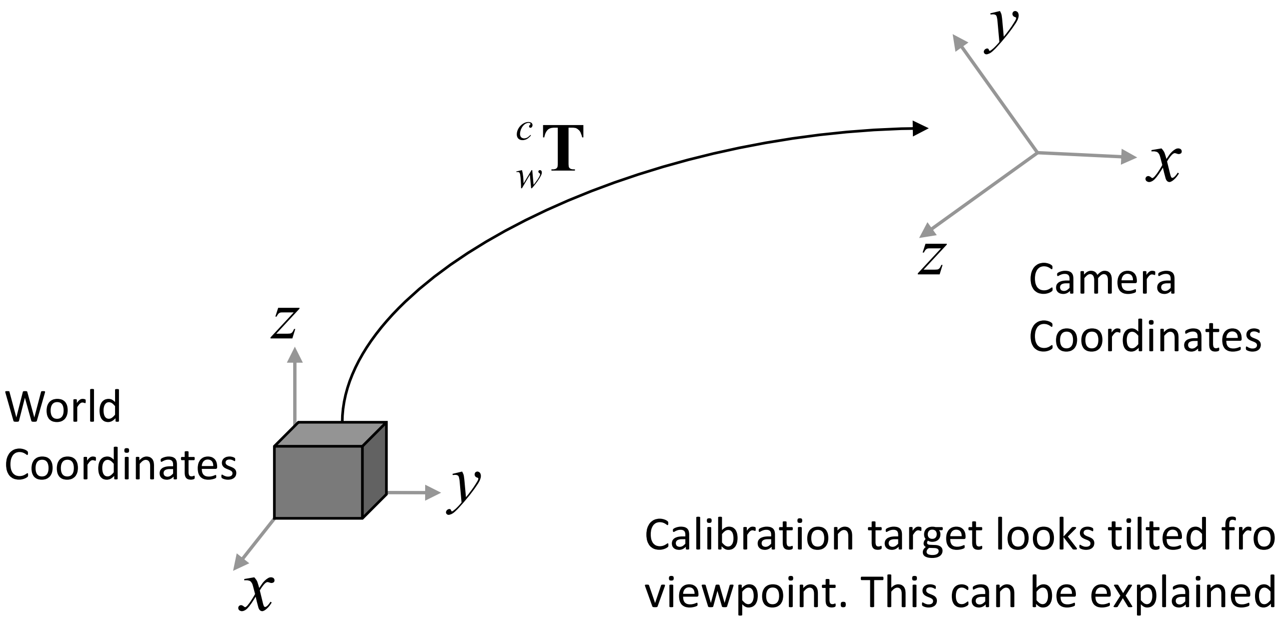
$$\hat{u} = \frac{u}{w} = \frac{aX + sY + u_0}{Z}$$

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

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

$$Camera = Calibration \times \text{Projection} \times Extrinsics$$

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

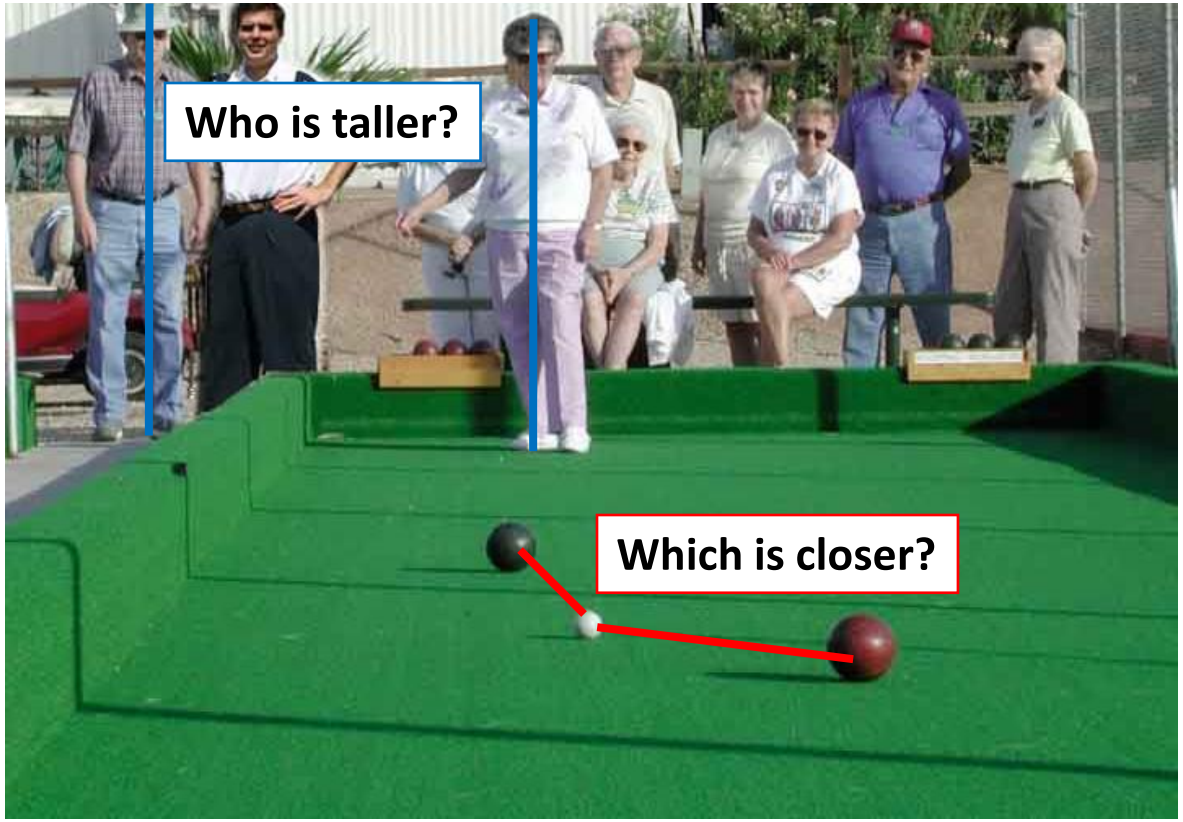
$$= K\left[R \quad t\right]M = PM$$

5+6 DOF = 11 !

Projective Geometry

What is lost?

- Length



Length and area are not preserved

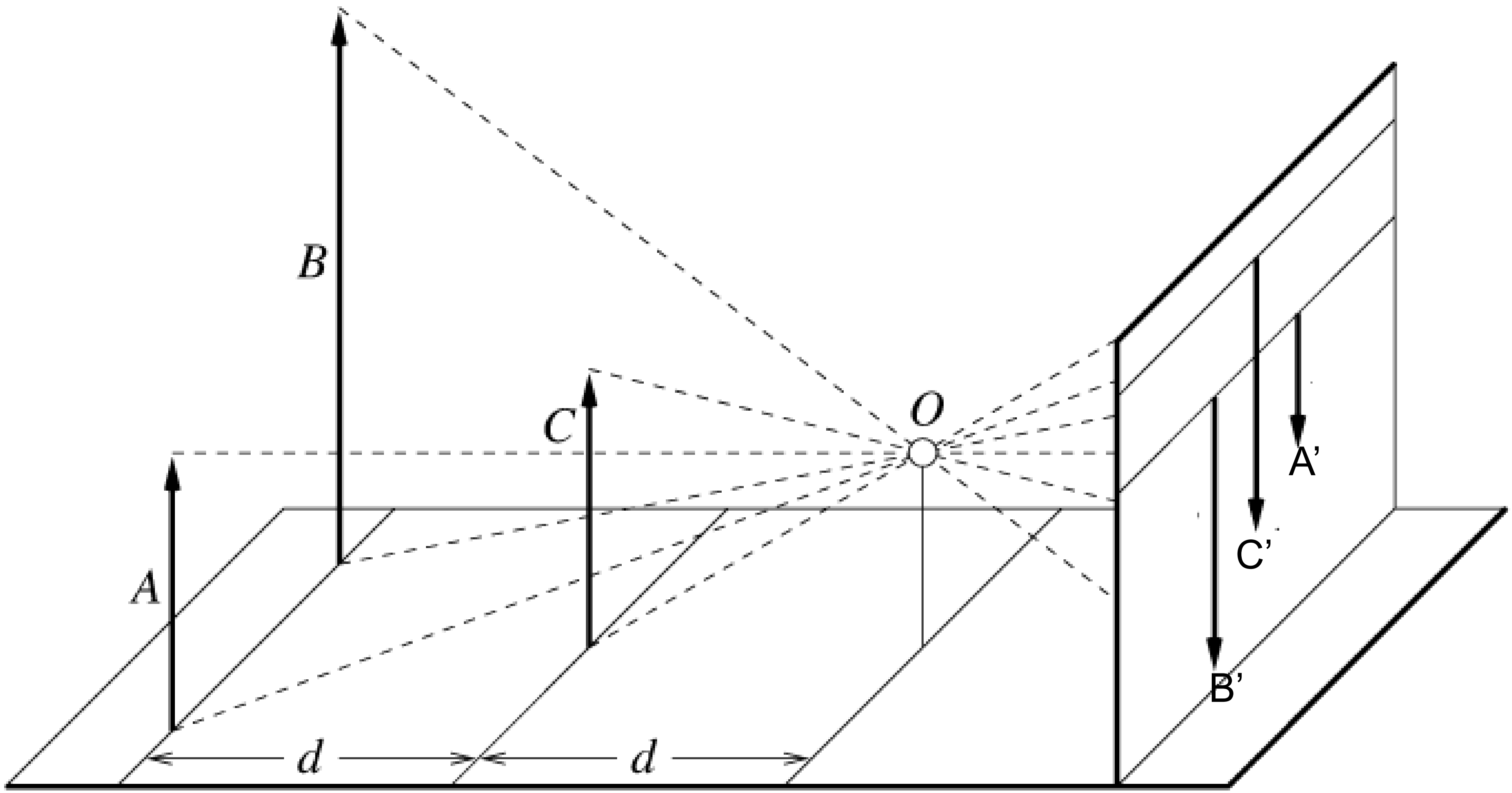
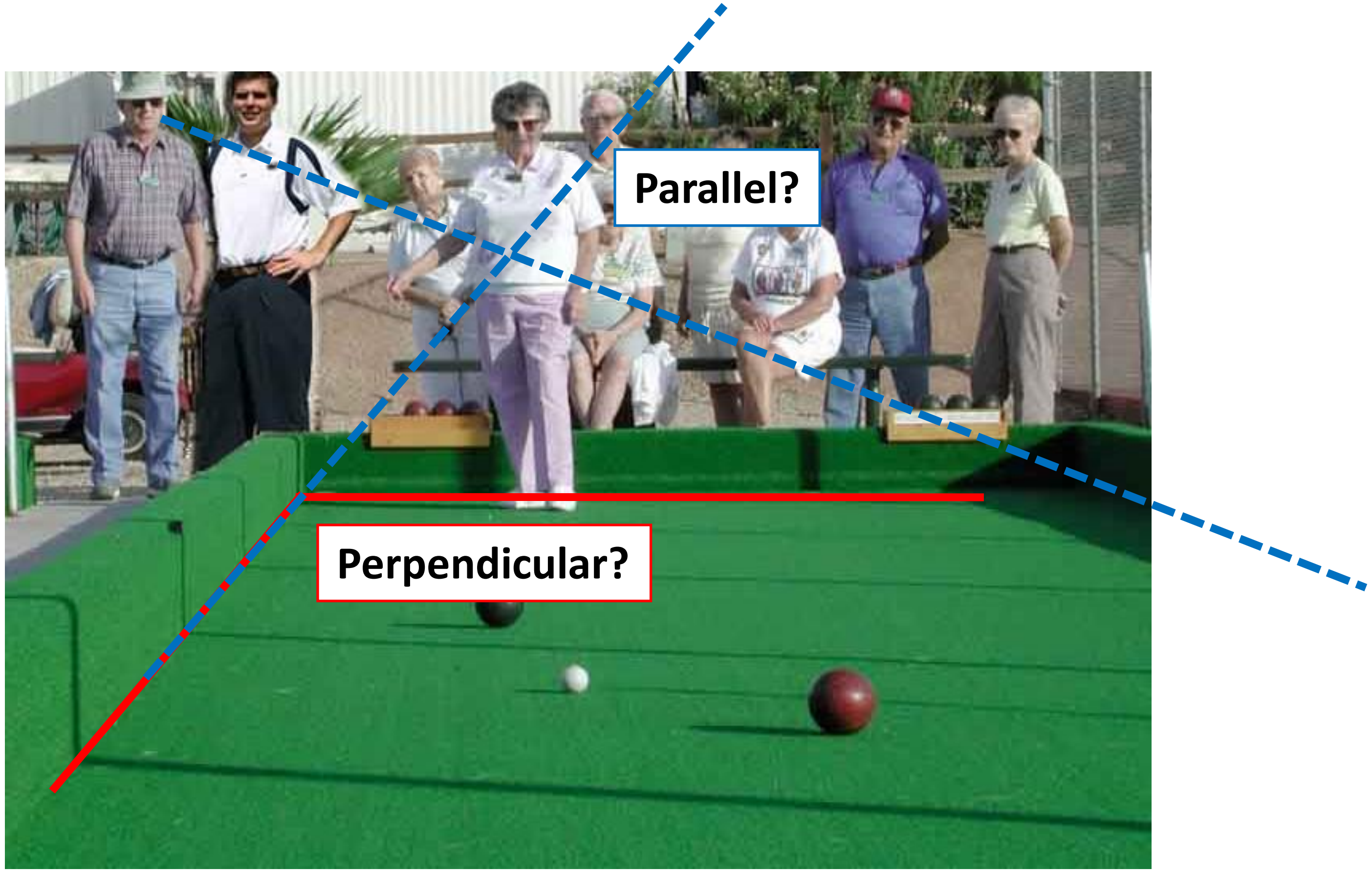


Figure by David Forsyth

Projective Geometry

What is lost?

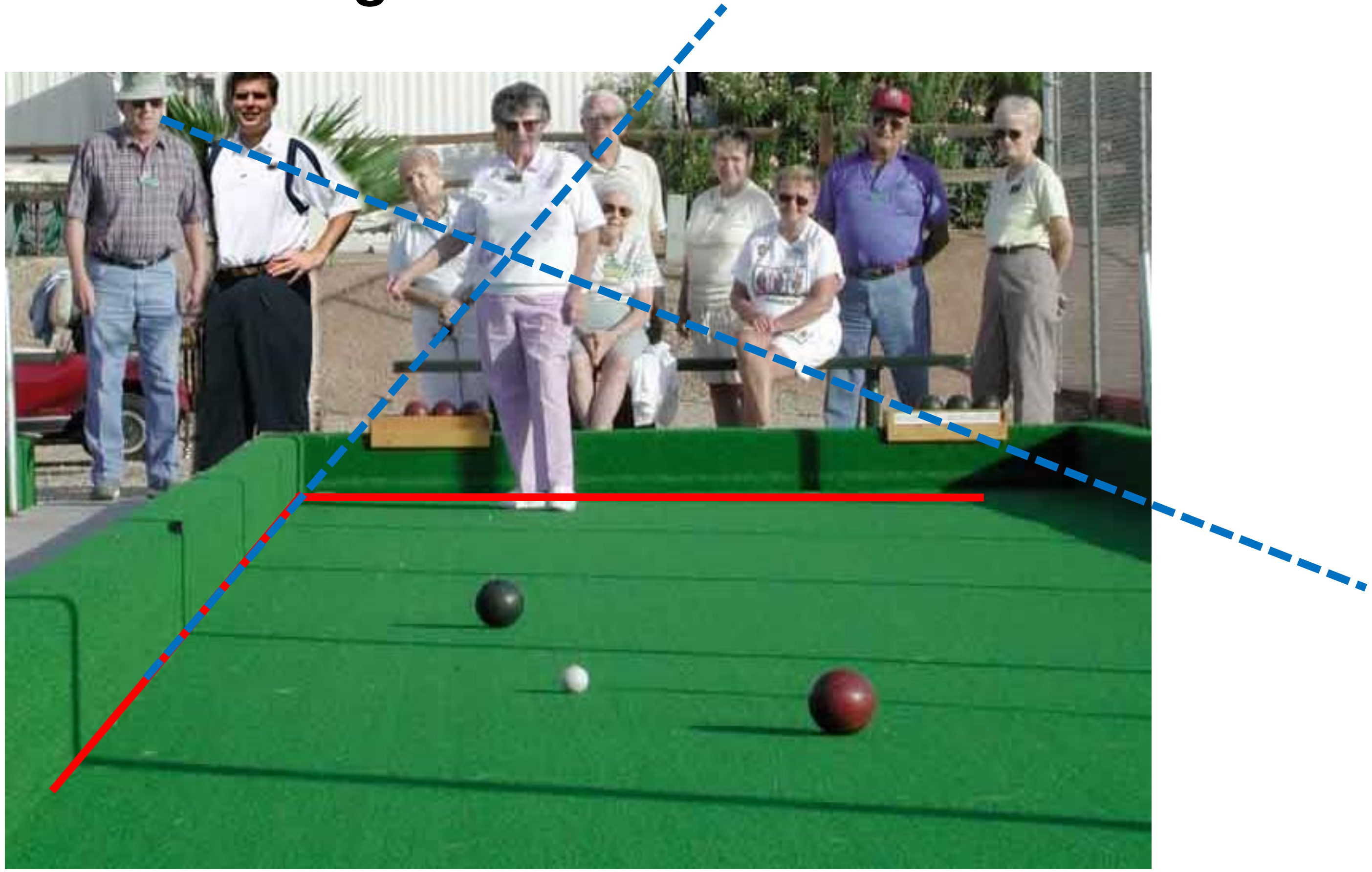
- Length
- Angles



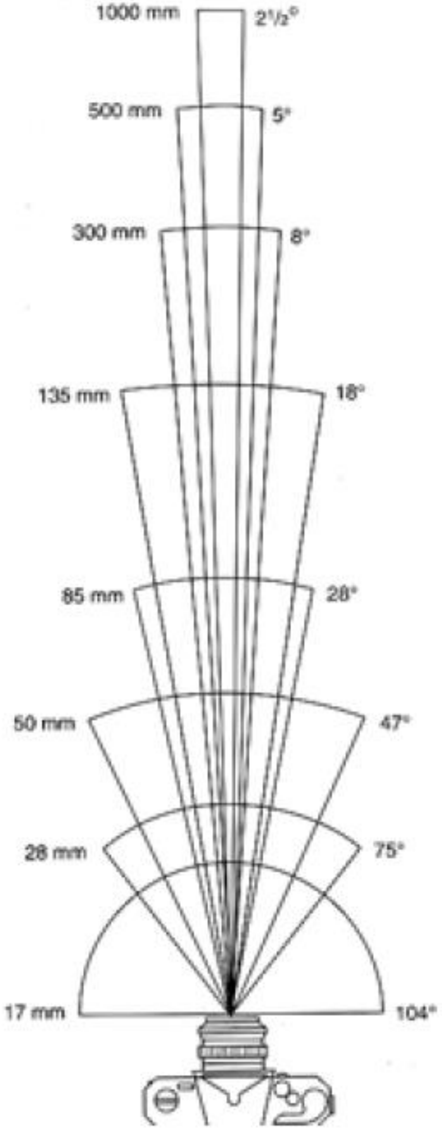
Projective Geometry

What is preserved?

- Straight lines are still straight

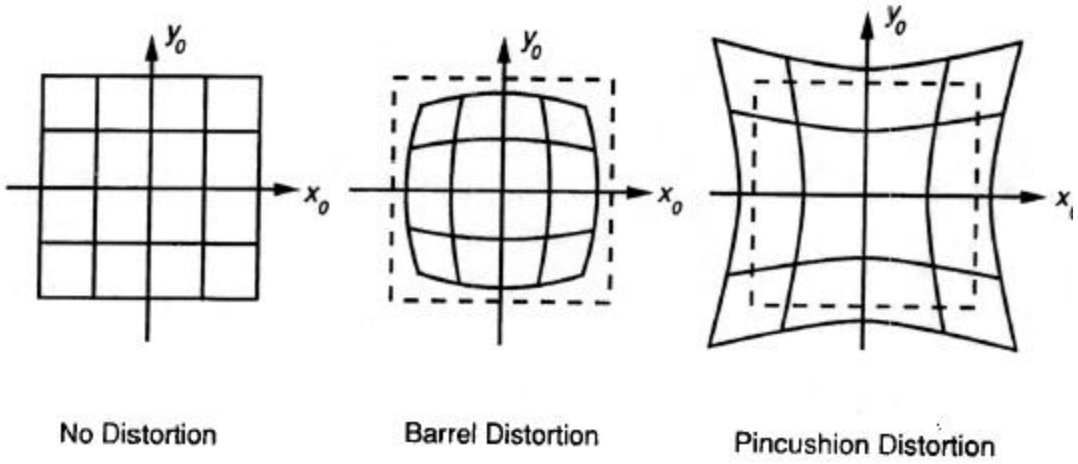


Field of View (Zoom, focal length)



From London and Upton

2.1.6 Radial Distortion



Corrected Barrel Distortion

Image from Martin Habbecke