SSY098 - Image Analysis

Lecture 9 - Camera Geometry

Torsten Sattler
(slides adapted from Olof Enqvist)

Last Lecture

Jan. 20	Introduction, Linear classifiers and filtering	
Jan. 23	Filtering, gradients, scale	Lab 1
Jan. 27	Local features	
Jan. 30	Learning a classifier	Lab 2
Feb. 3	Convolutional neural networks	
Feb. 6	More convolutional neural networks	
Feb. 10	Robust model fitting and RANSAC	Lab 3
Feb. 13	**Image registration**	
Feb. 17	Camera Geometry	Lab 4
Feb. 20	More camera geometry	
Feb. 24	Generative neural networks	
Feb. 27	Generative neural networks	
Mar. 2	TBA	
Mar. 9	TBA	

Image Stitching

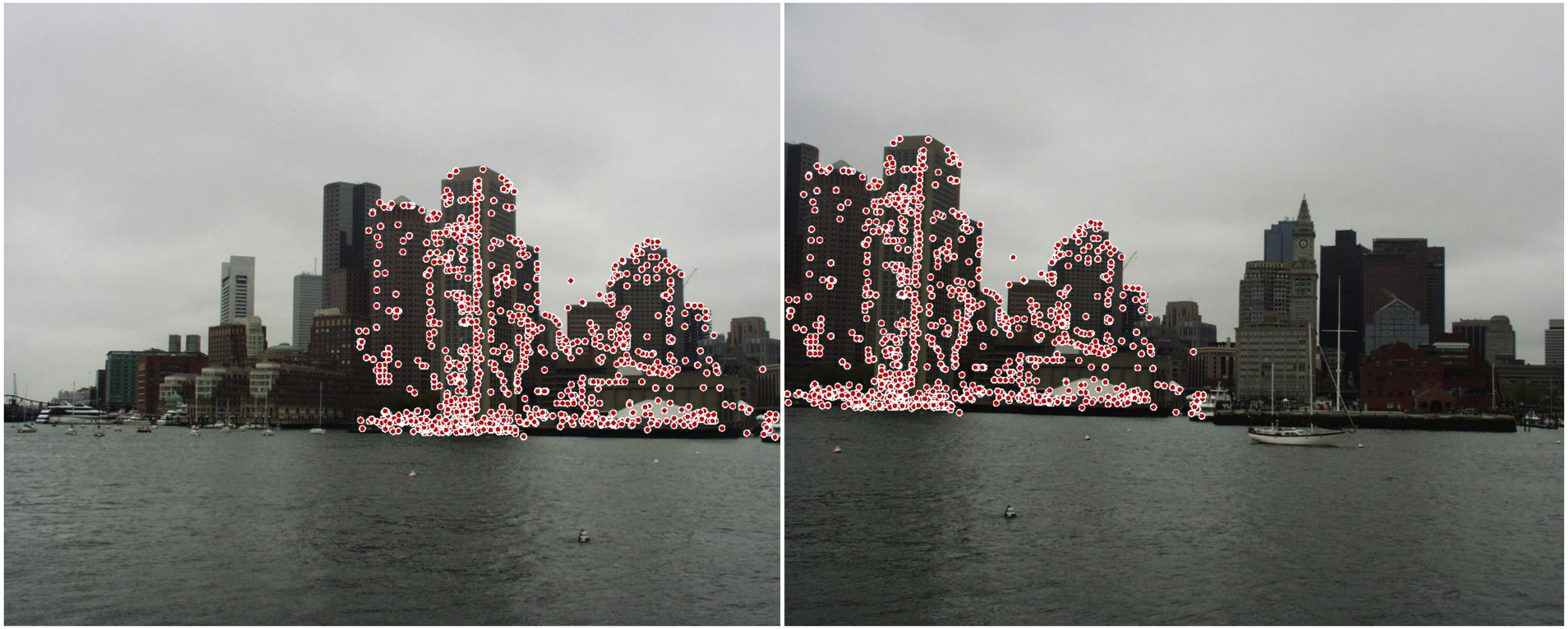
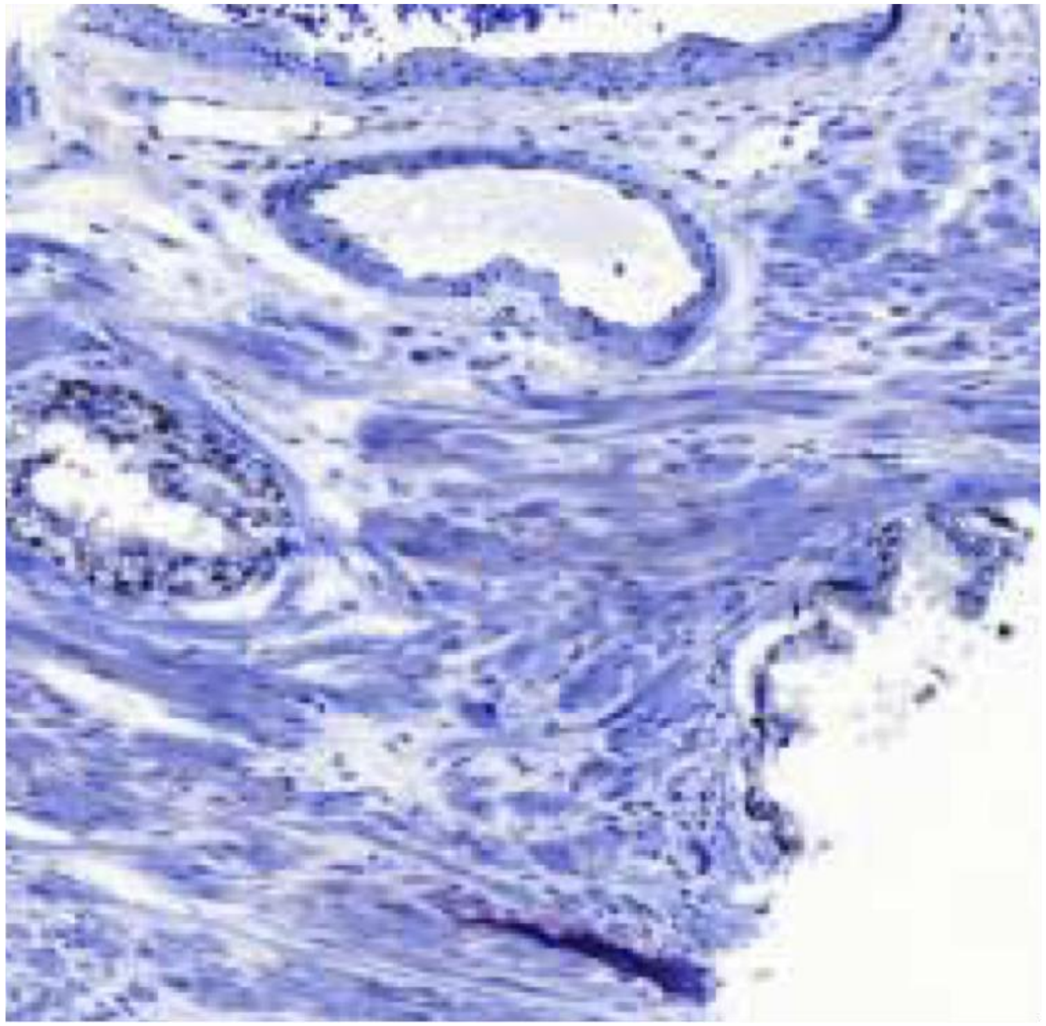


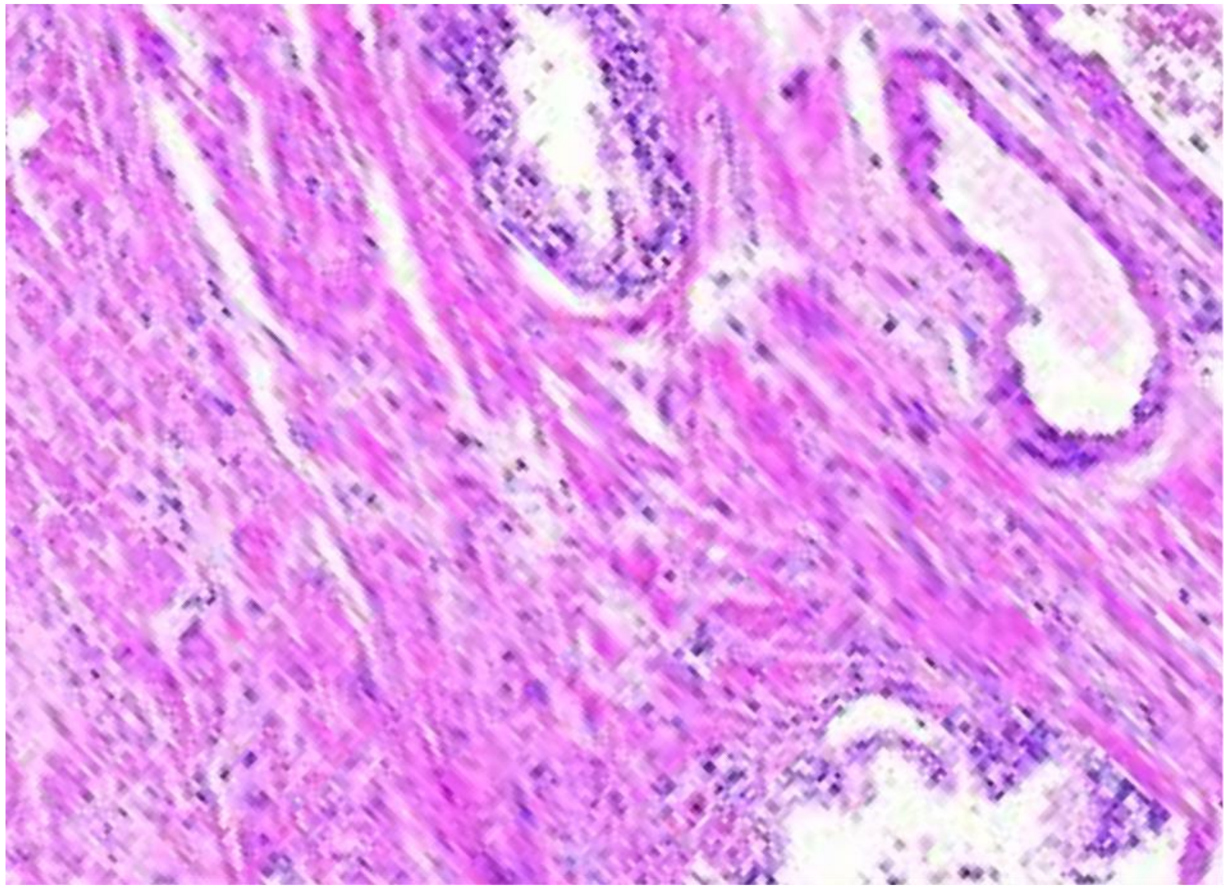
Image Stitching

Last Lecture

transformation



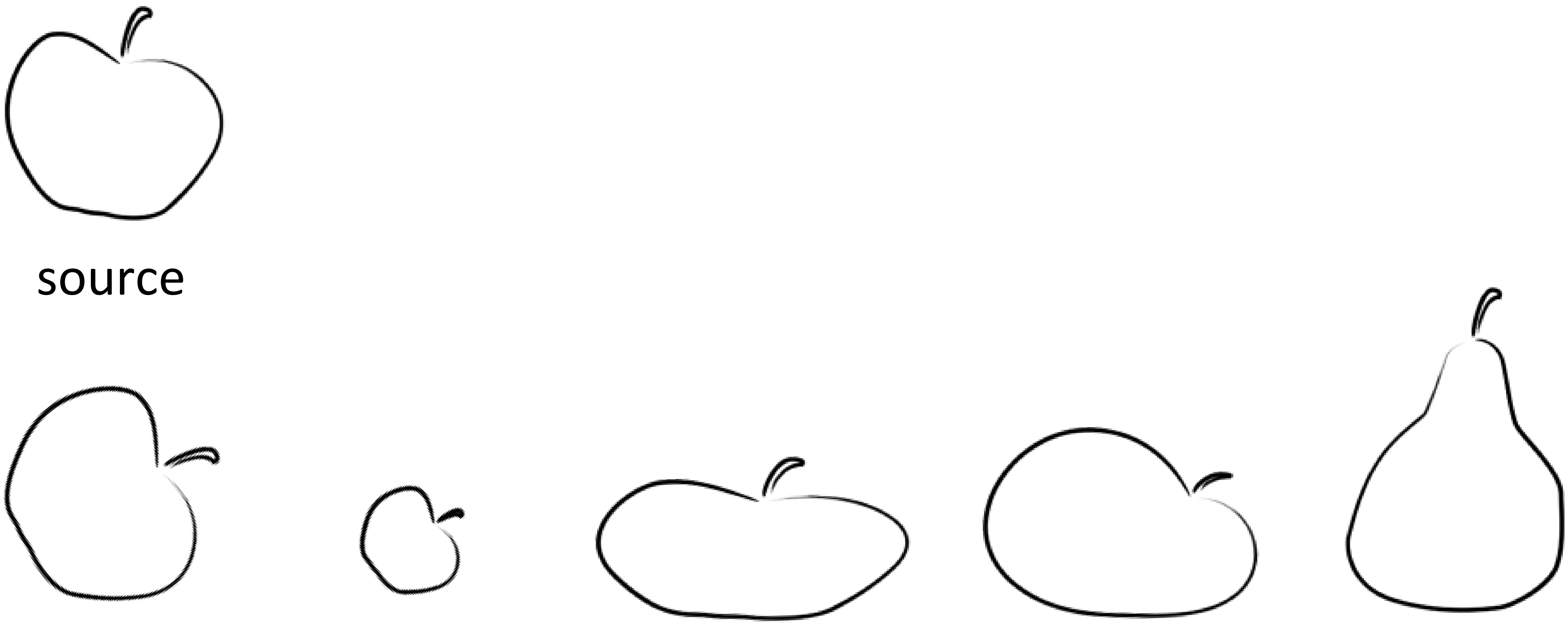
source image



target image

Pixel transfer

Last Lecture

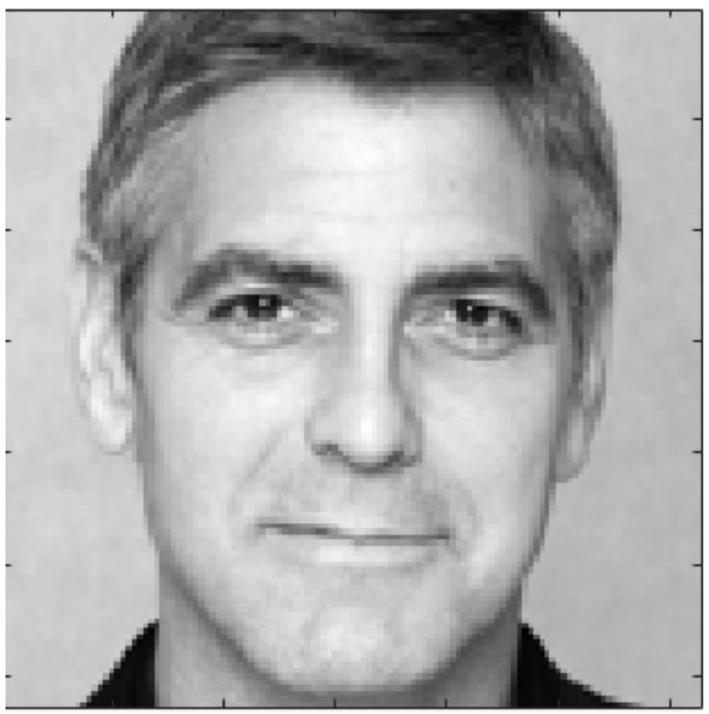


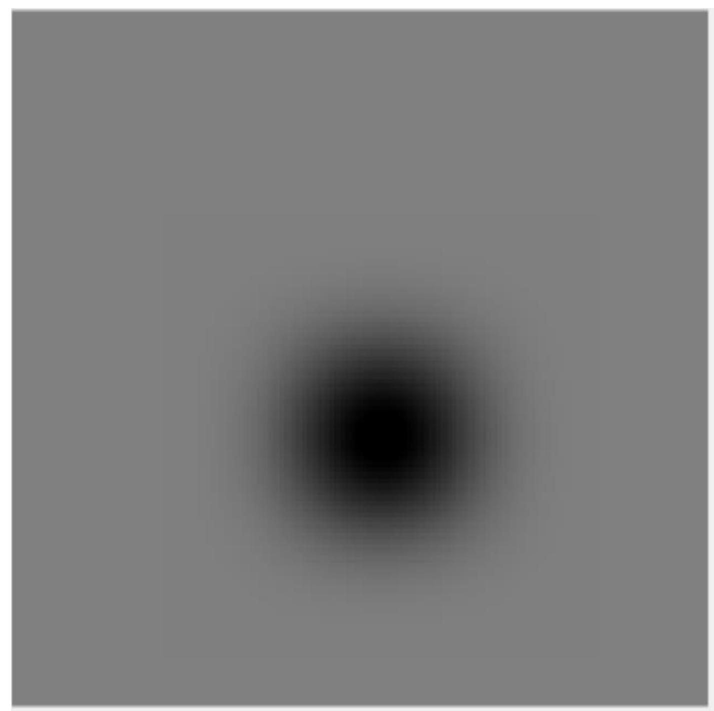
source

rigid similarity affine perspective free-form

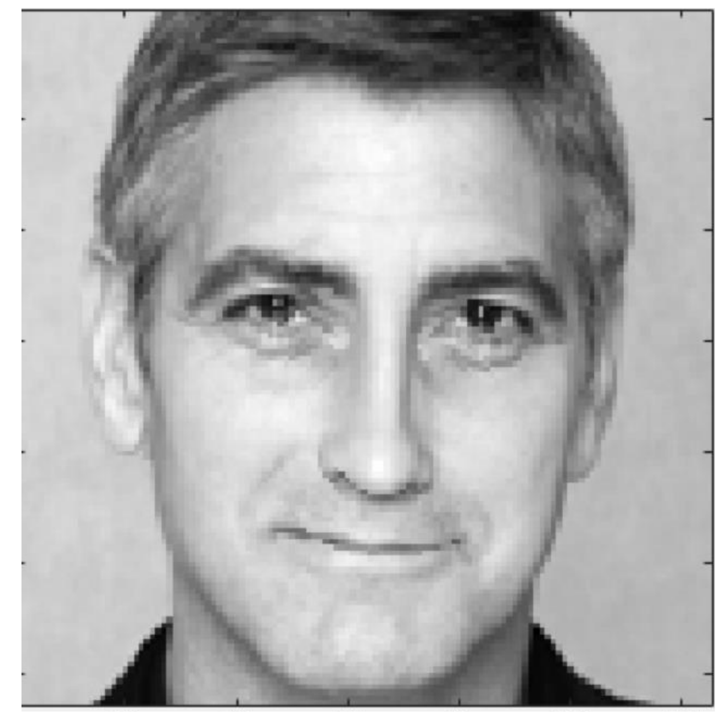
Transformation hierarchy

Last Lecture



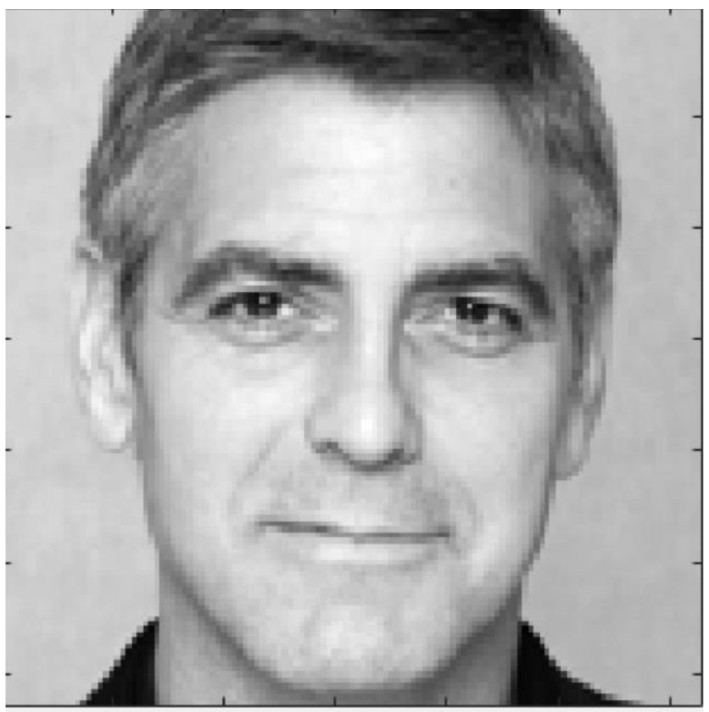


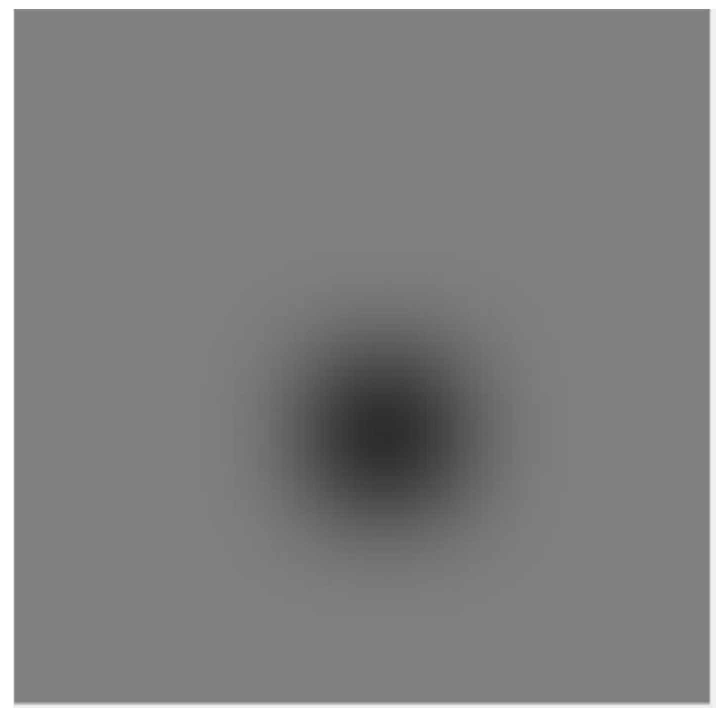
Δ_y



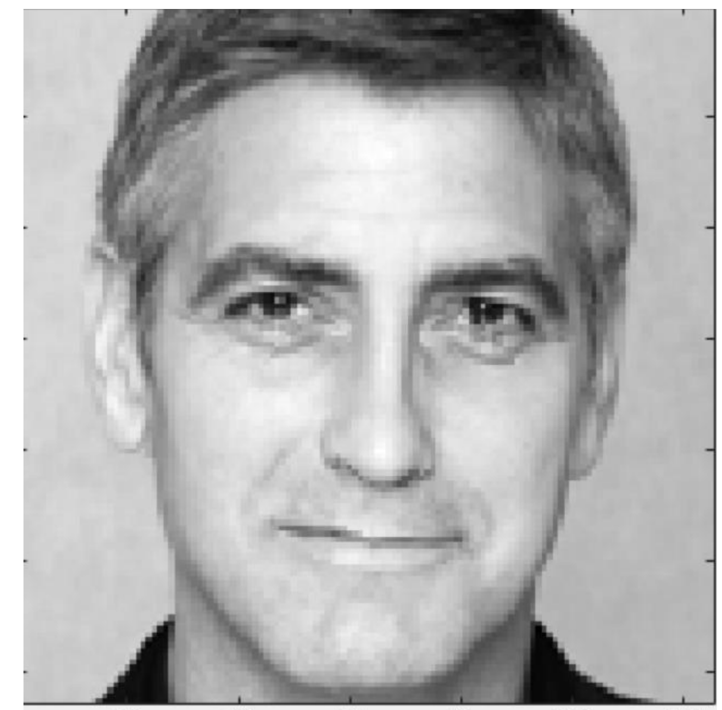
Free-form registration

Last Lecture



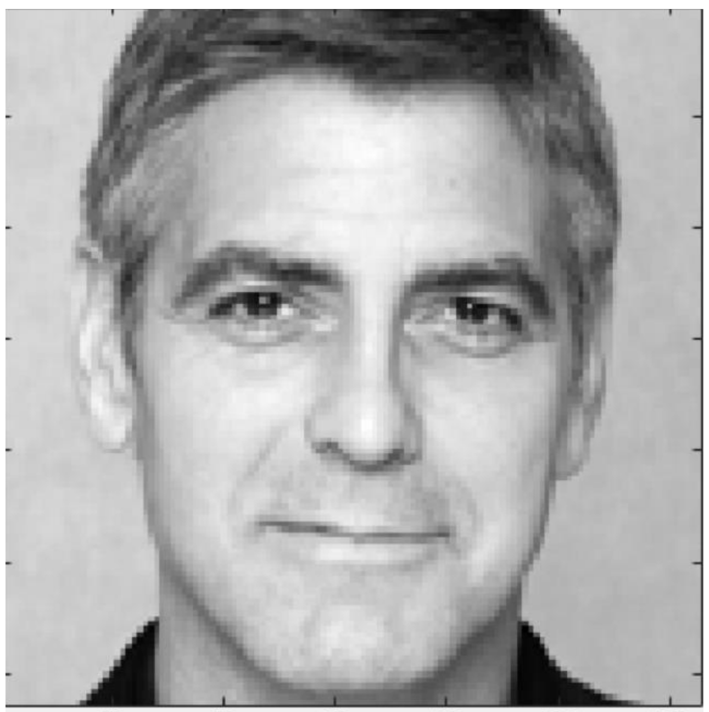


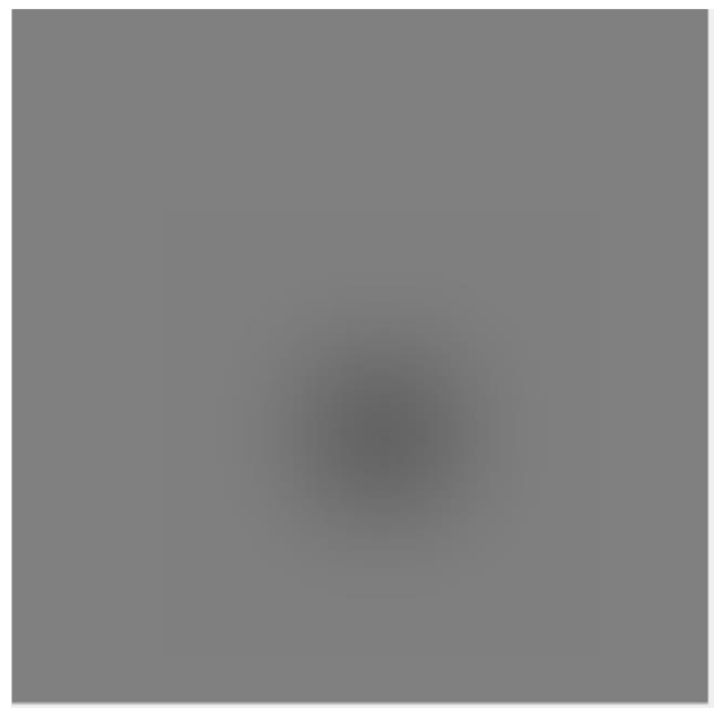
Δ_y



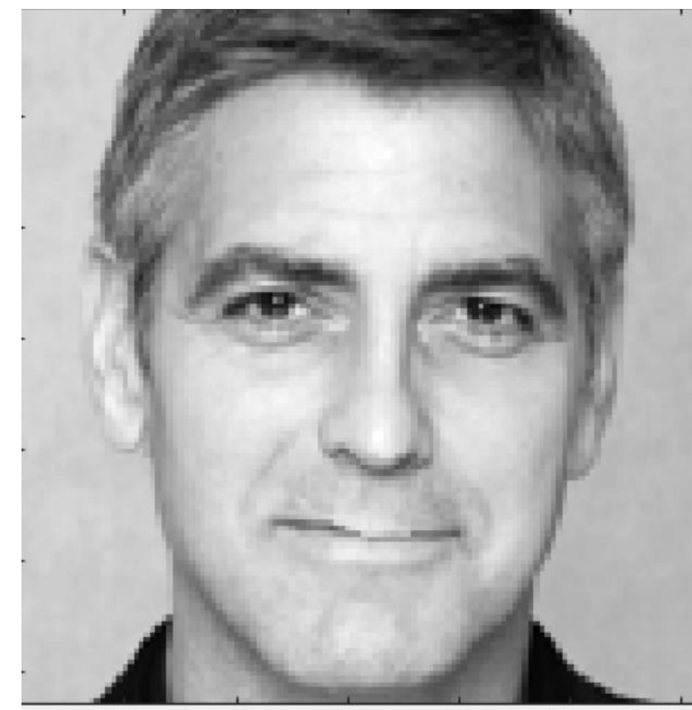
Free-form registration

Last Lecture



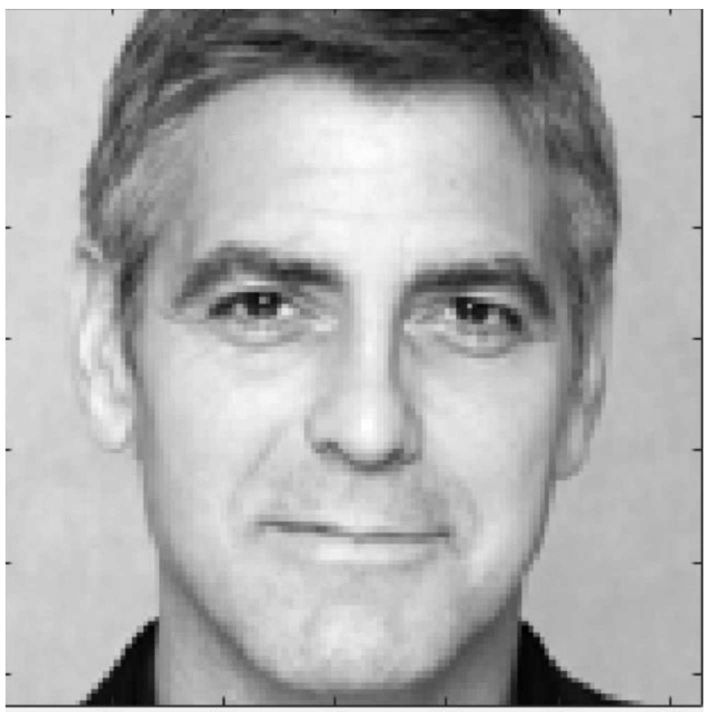


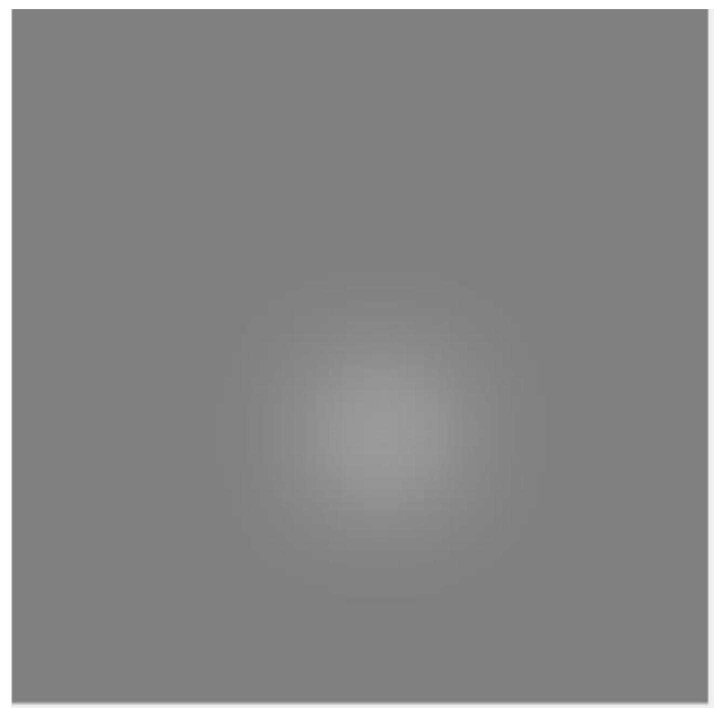
Δ_y



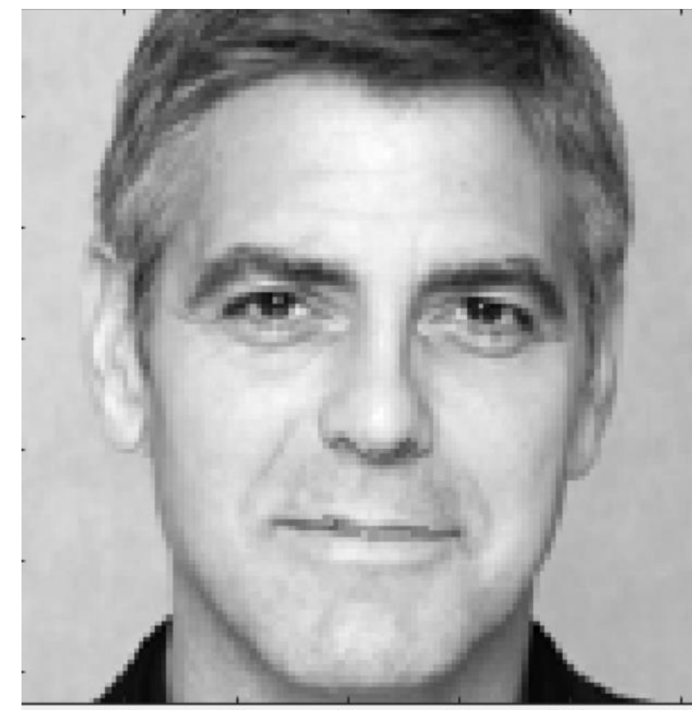
Free-form registration

Last Lecture



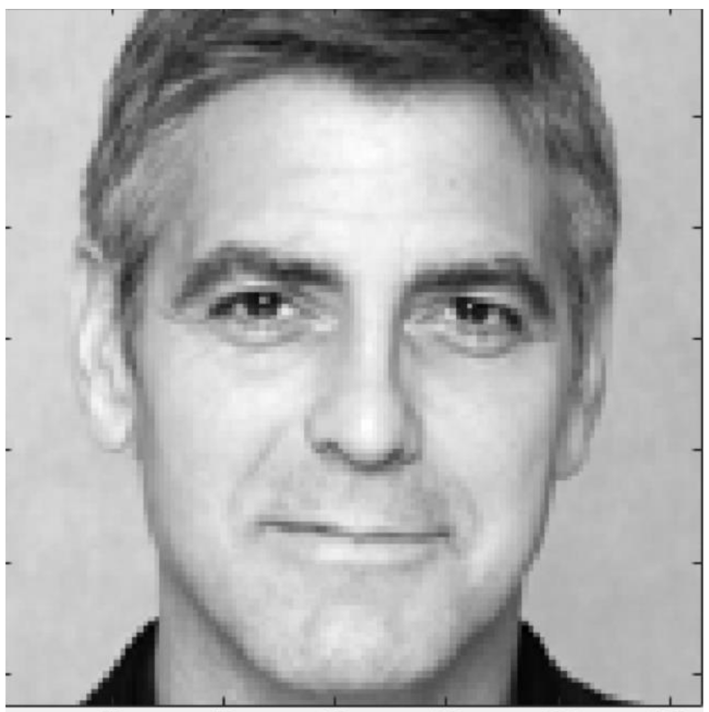


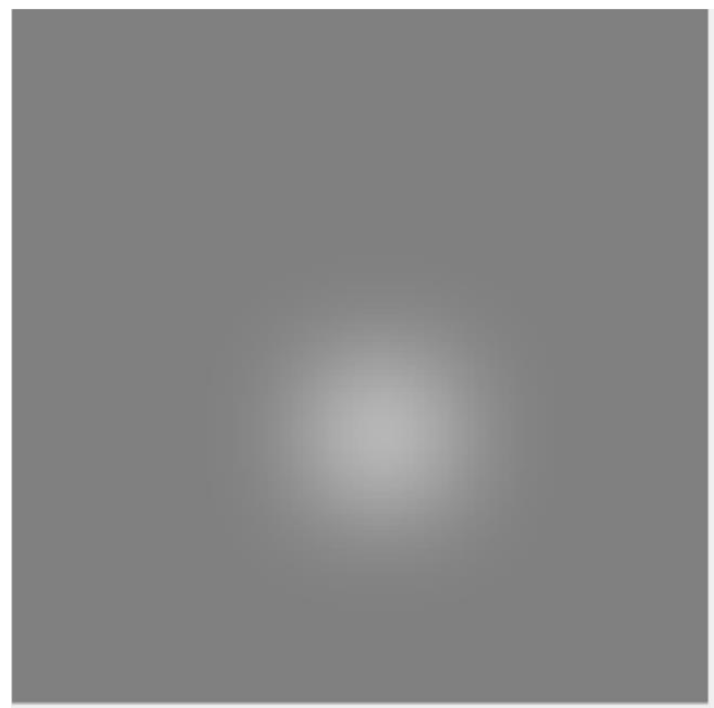
Δ_y



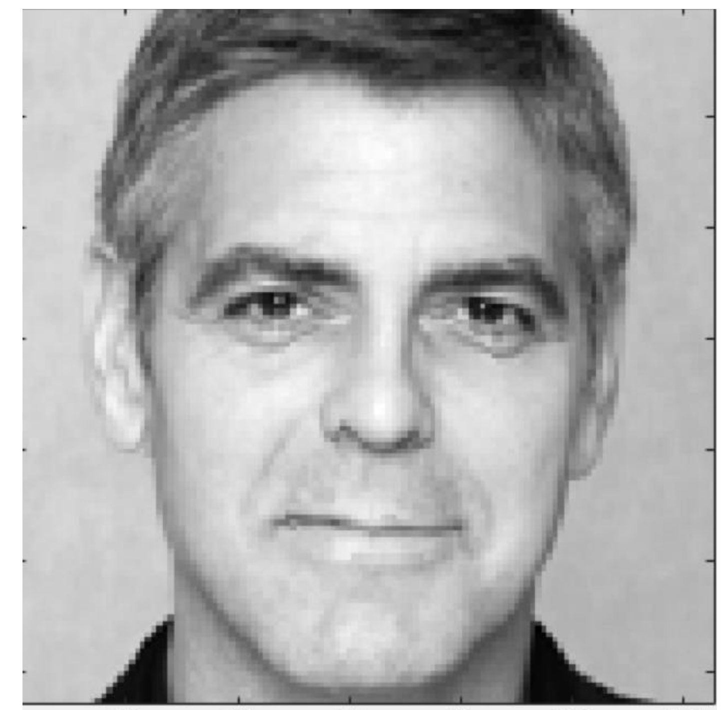
Free-form registration

Last Lecture



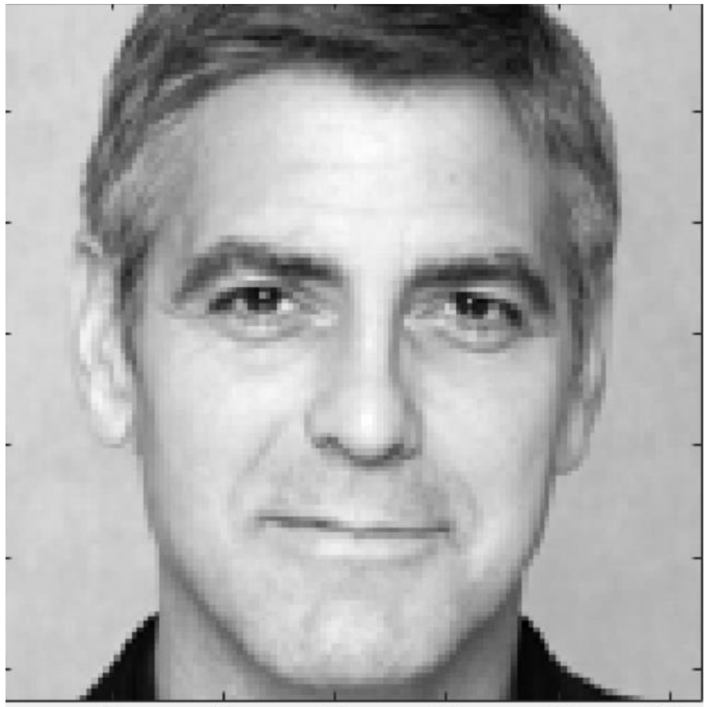


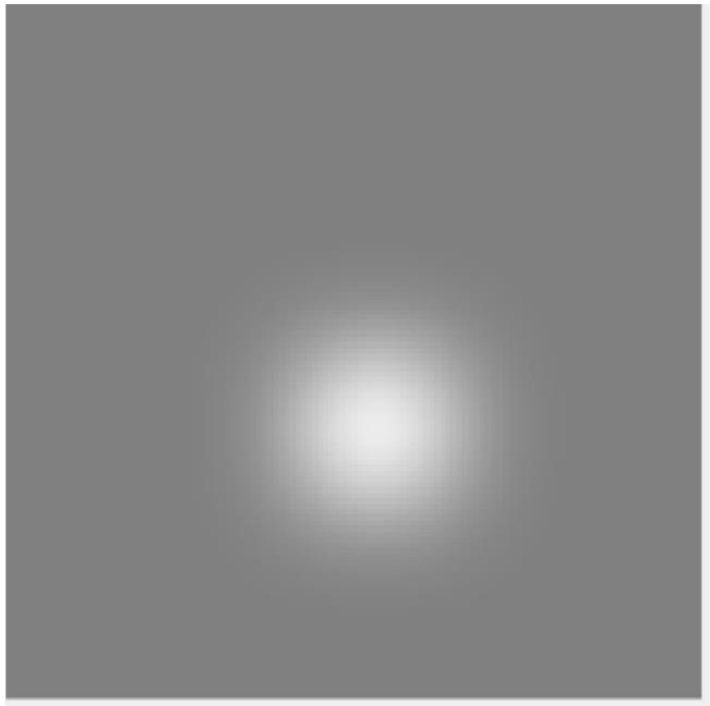
$$\Delta_y$$



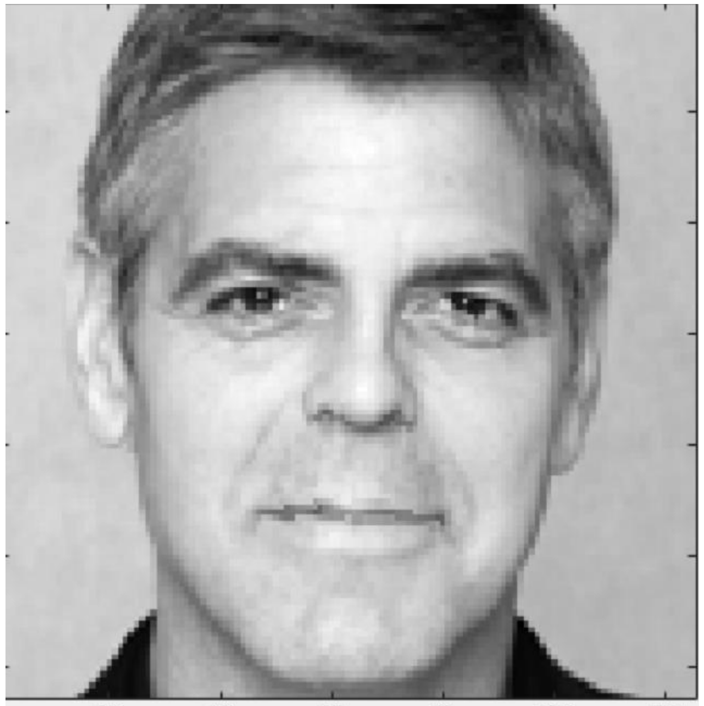
Free-form registration

Last Lecture



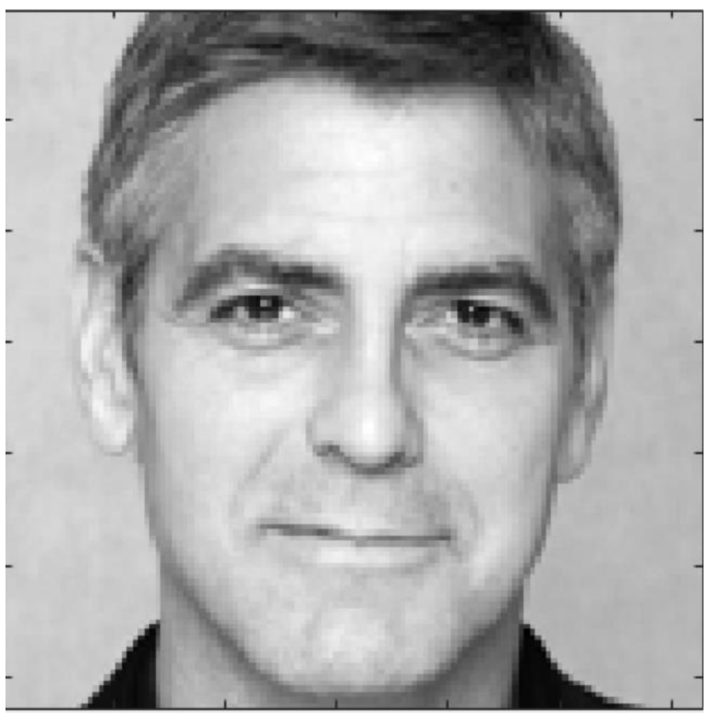


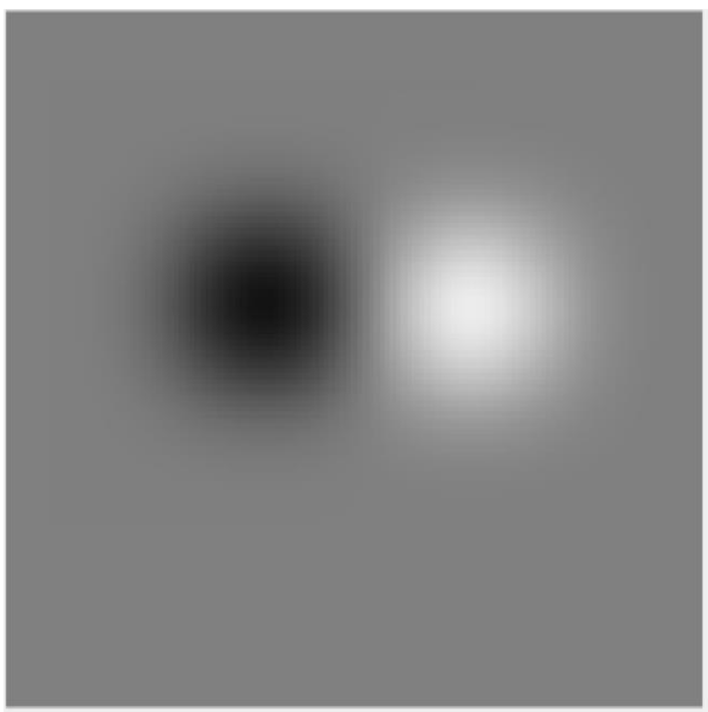
Δ_y



Free-form registration

Last Lecture

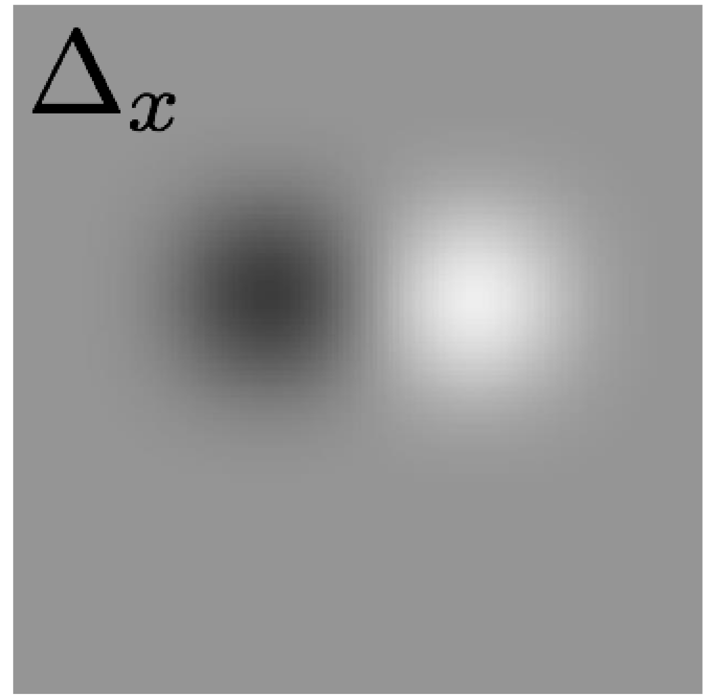


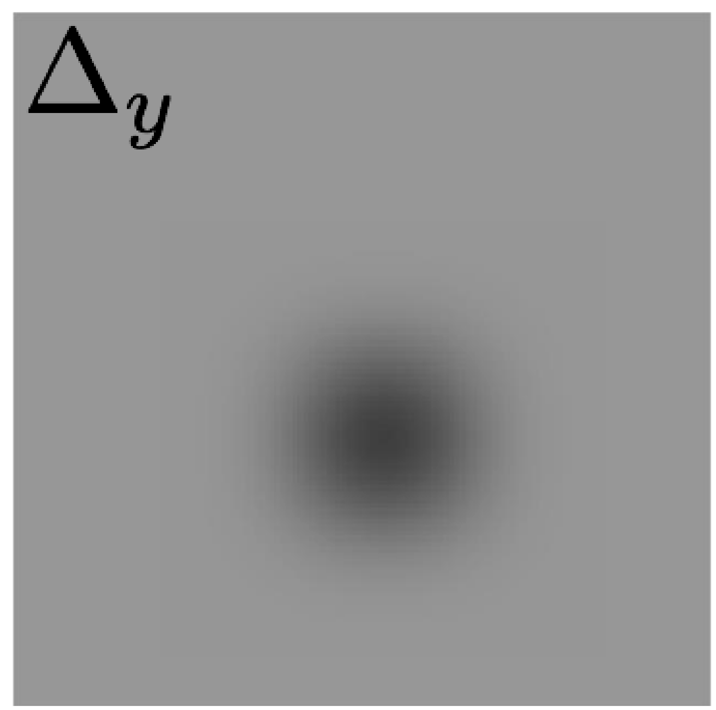


$$\Delta_x$$

Free-form registration

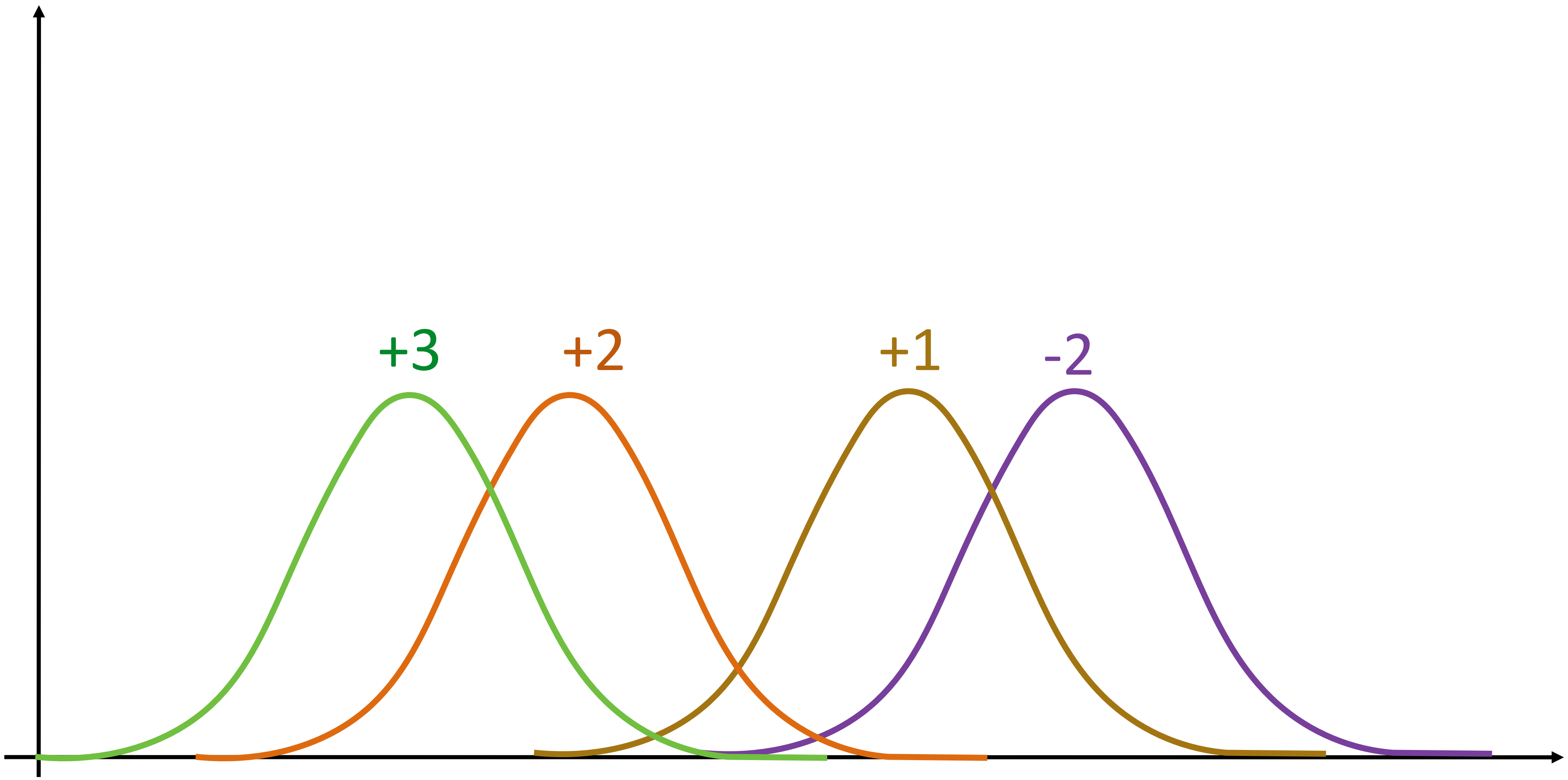
Last Lecture





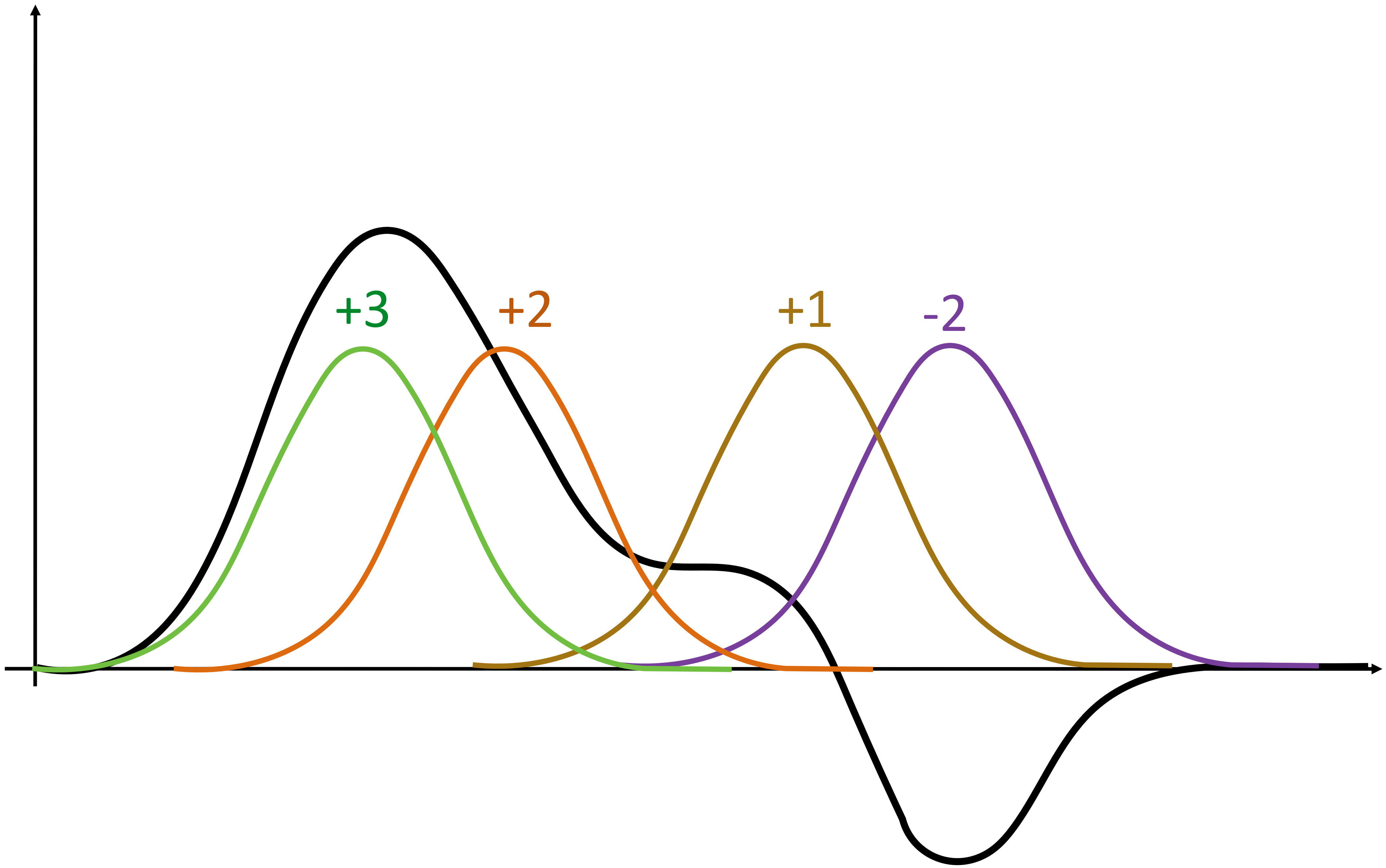
Free-form registration

Last Lecture



$$\Delta_x(x, y) = \sum_{k=1}^{n} \alpha_k B_k(x, y)$$

Last Lecture

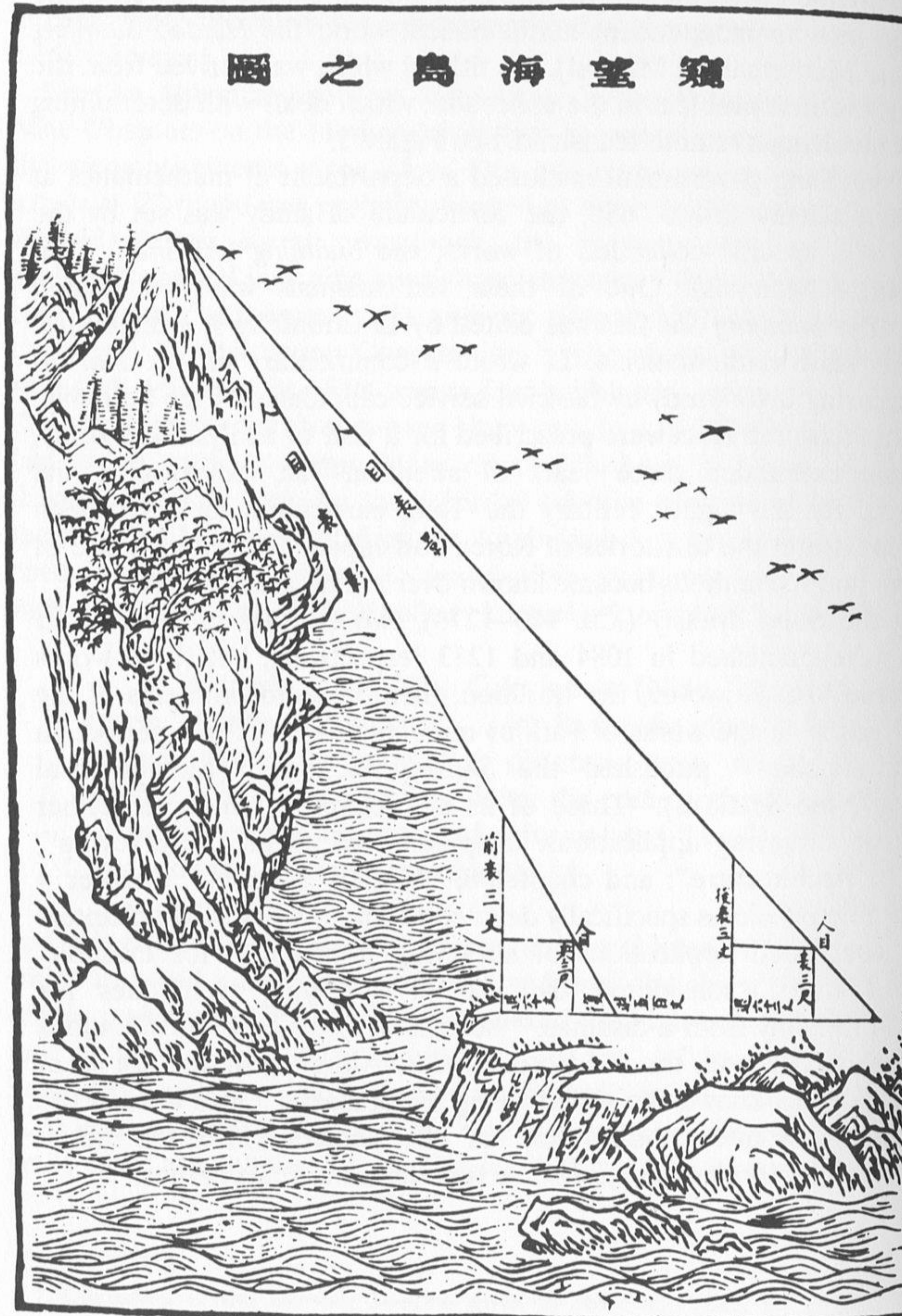


$$\Delta_x(x, y) = \sum_{k=1}^{n} \alpha_k B_k(x, y)$$

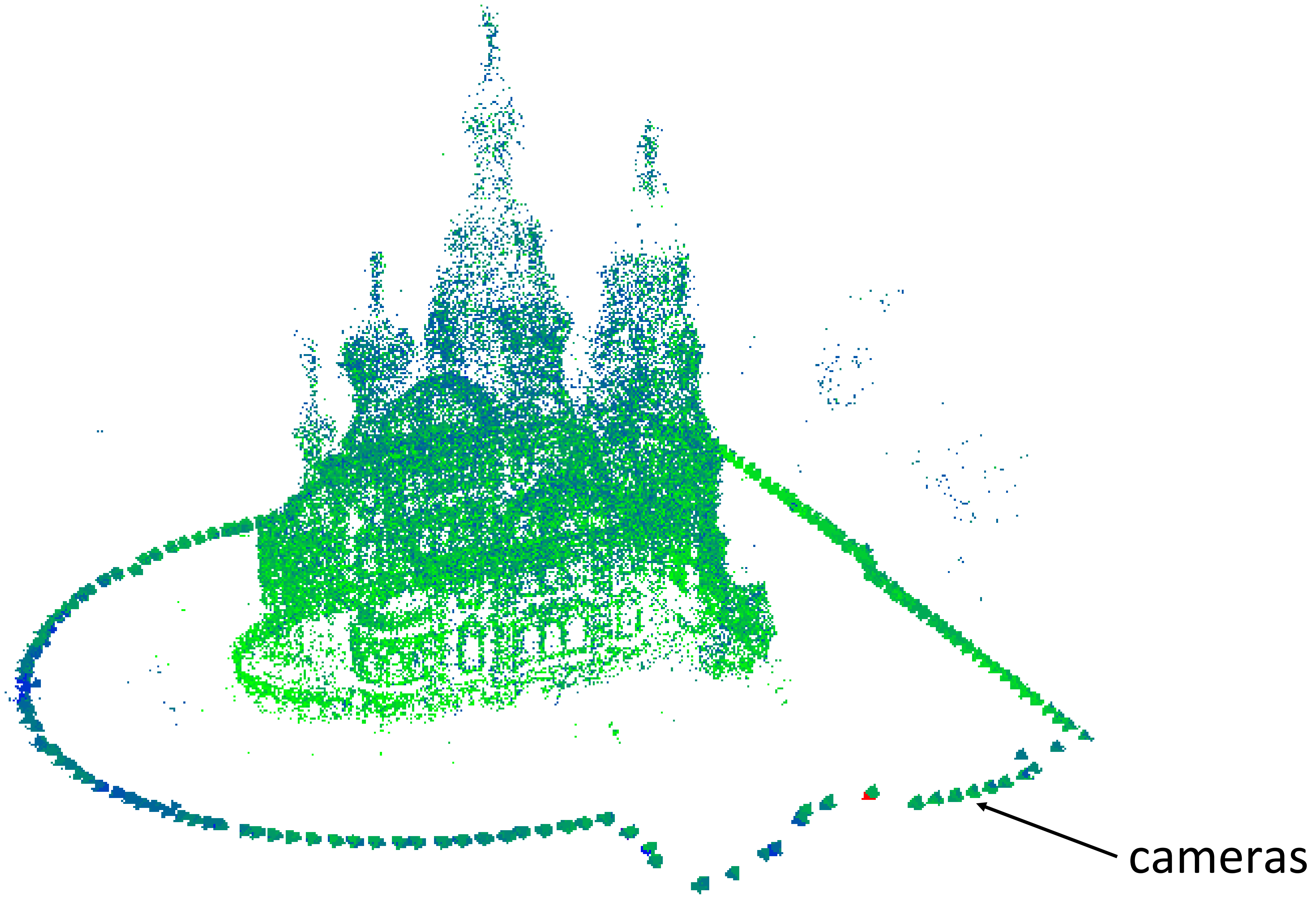
Today

Camera Geometry

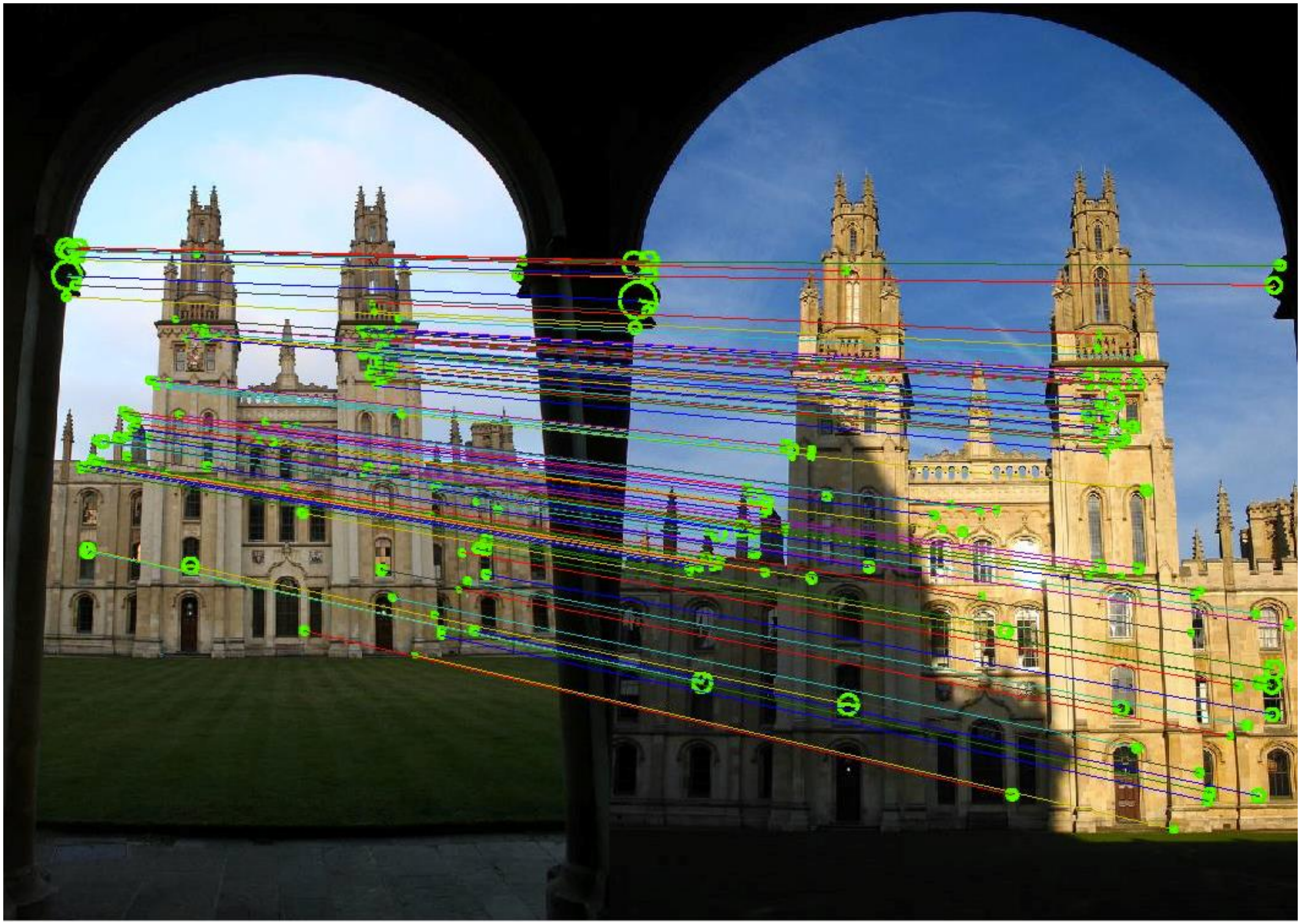
Triangulation



Structure-from-Motion



The Measurements



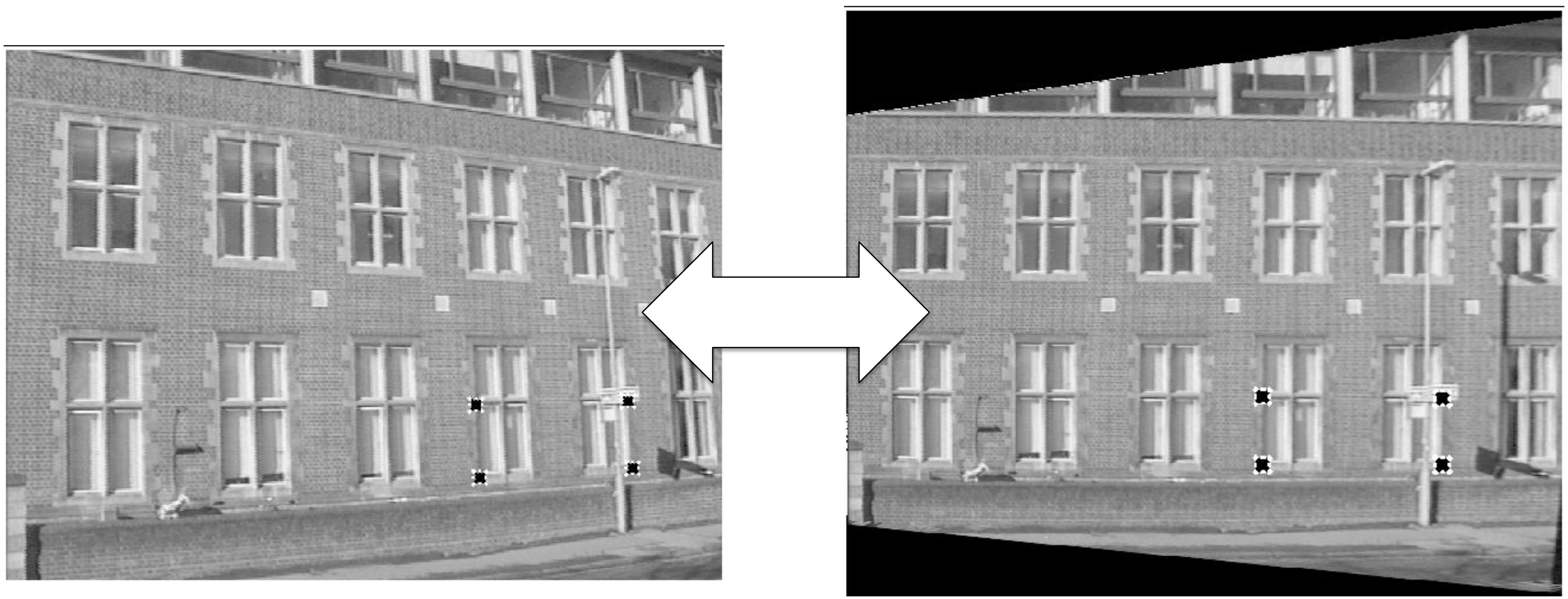
Camera Obscura

Today

- 2D Projective Geometry
- Camera Geometry

2D Projective Geometry?

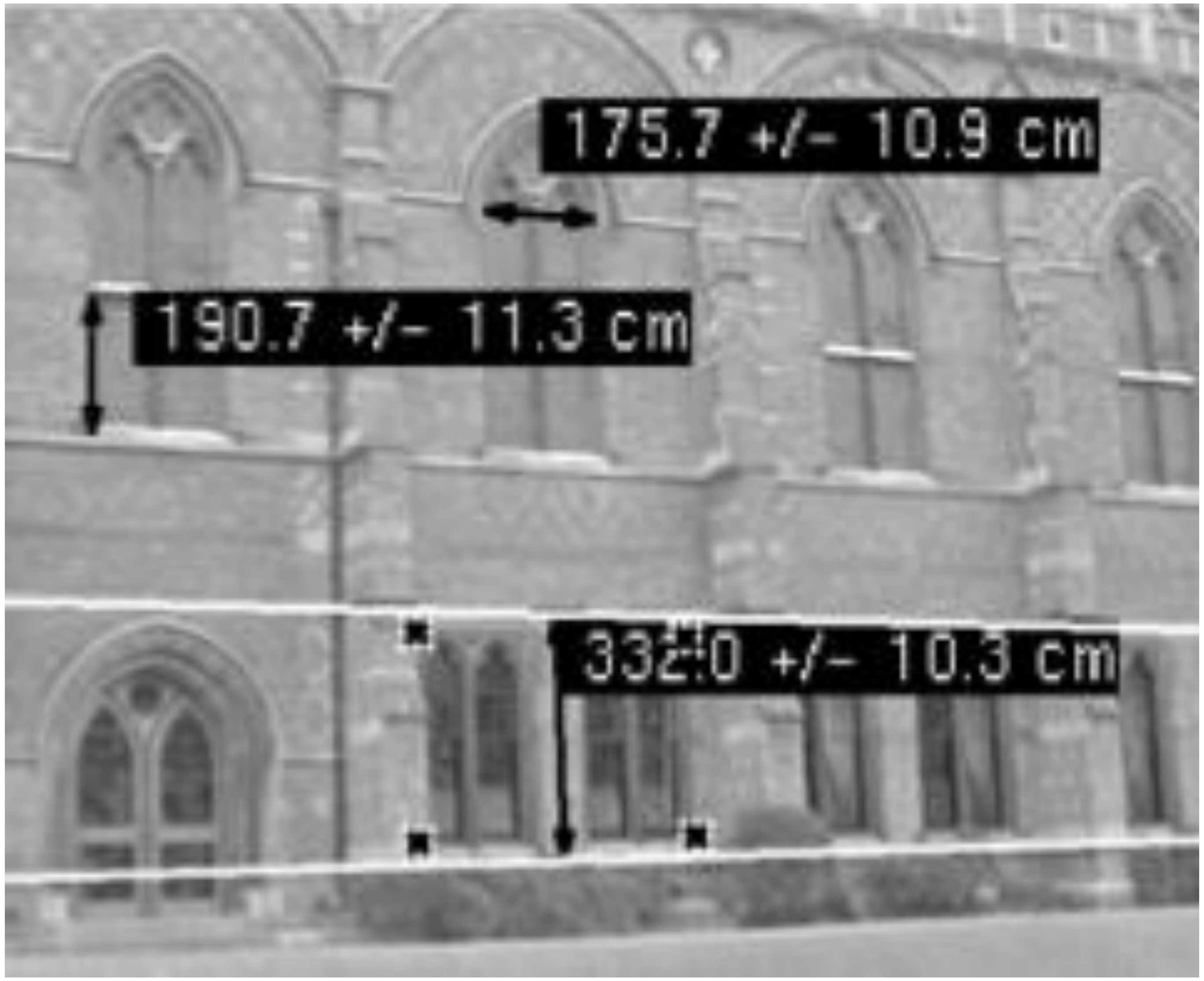
Projections of planar surfaces



A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD Thesis 1999.

2D Projective Geometry?

Measure distances



A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD Thesis 1999.

2D Projective Geometry?

Discover details

Piero della Francesca, La Flagellazione di Cristo (1460)

A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD Thesis 1999.

2D Projective Geometry?

Discover details

Piero della Francesca, La Flagellazione di Cristo (1460)

A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD Thesis 1999.

2D Projective Geometry?

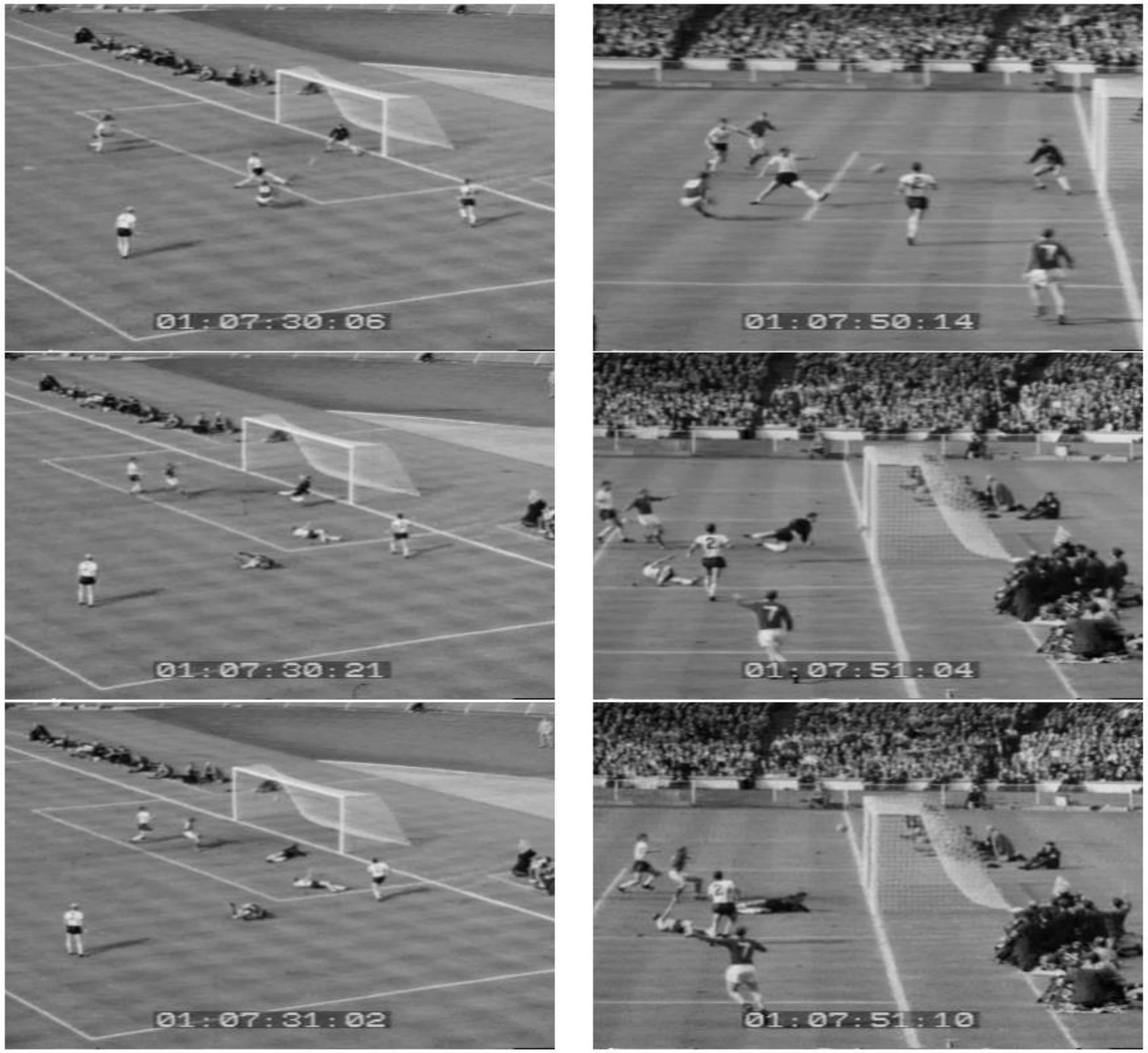
Discover details

Piero della Francesca, La Flagellazione di Cristo (1460)

A. Criminisi. *Accurate Visual Metrology from Single and Multiple Uncalibrated Images*. PhD Thesis 1999.

2D Projective Geometry?

Settling important questions



I. Reid and A. Zisserman. *Goal-directed video metrology,* ECCV 1996.

2D Projective Geometry?

Settling important questions

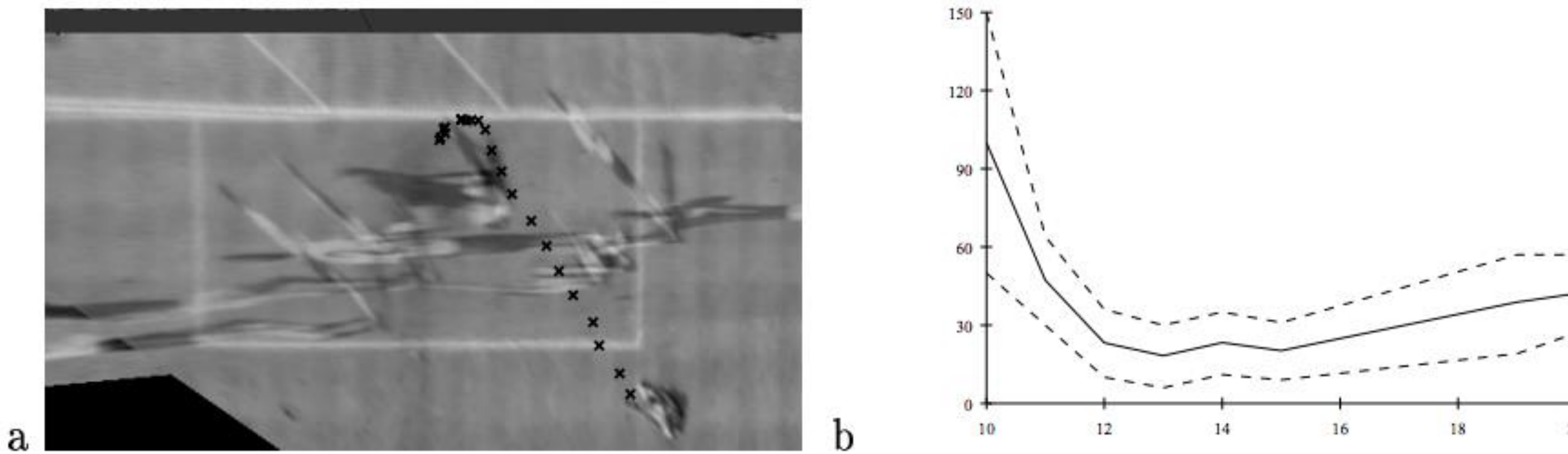


Fig. 8. (a) The computed position of the ball throughout the sequence; (b) distance away from being a goal (in cm) versus frame number for the crucial frames of the sequence.

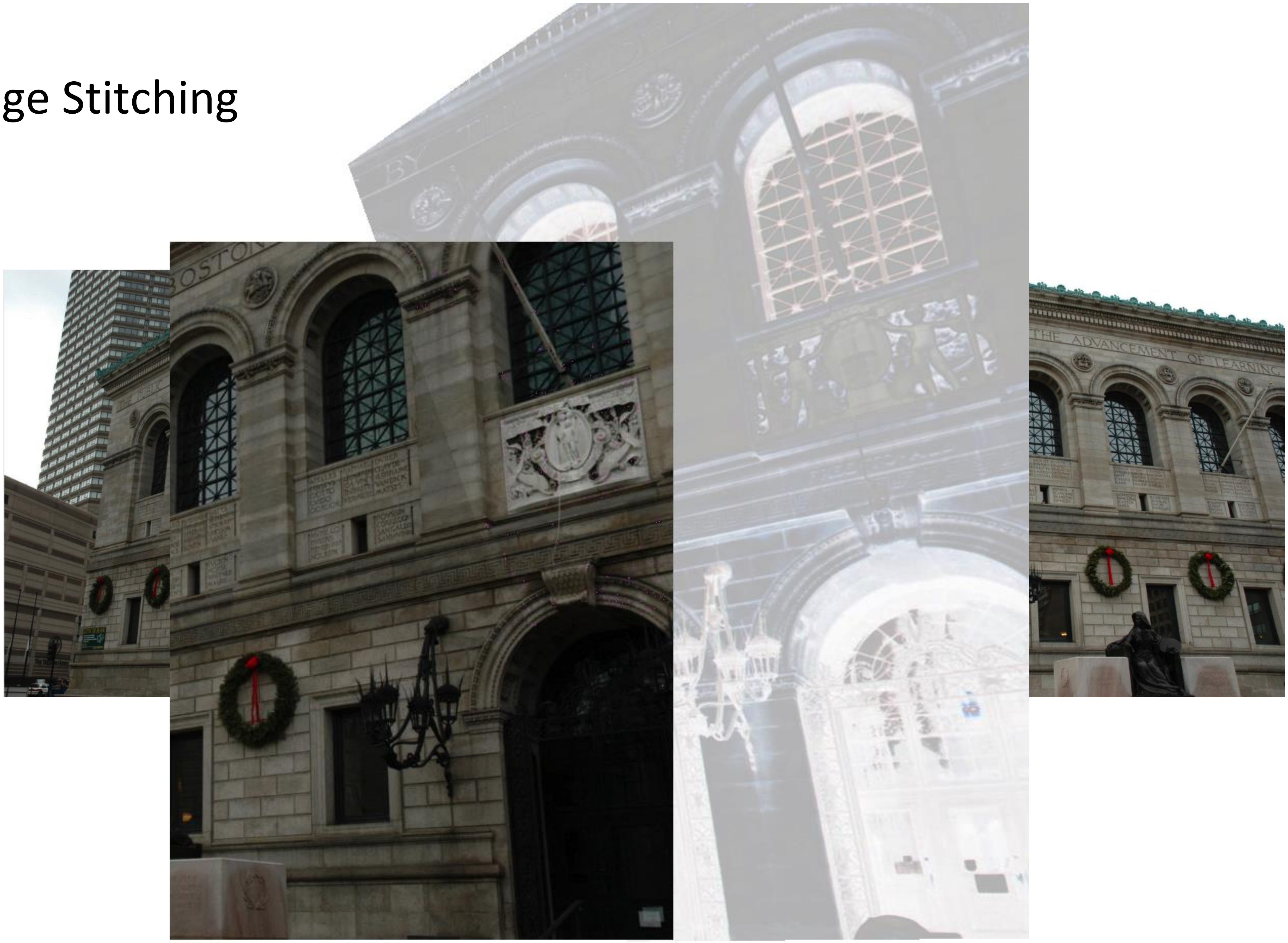
I. Reid and A. Zisserman. *Goal-directed video metrology,* ECCV 1996.

2D Projective Geometry?

Image Stitching

2D Projective Geometry?

Image Stitching



Hierarchy of 2D Transformations

rigid
$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Hierarchy of 2D Transformations

rigid $$\begin{pmatrix}\hat{x}\\ \hat{y}\end{pmatrix} = \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix}$$

similarity $$\begin{pmatrix}\hat{x}\\ \hat{y}\end{pmatrix} = s\begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix}$$

Hierarchy of 2D Transformations

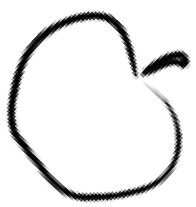
rigid $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

similarity $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = s \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

affine $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Hierarchy of 2D Transformations

rigid $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$



similarity $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = s \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

affine $$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

perspective $$\hat{w} \begin{pmatrix} \hat{x} \\ \hat{y} \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Extended Coordinates

``Extend" 2D coordinate to 3D coordinate

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Extended Coordinates

``Extend" 2D coordinate to 3D coordinate

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Write affine transformations as matrix multiplication

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

Extended Coordinates

``Extend” 2D coordinate to 3D coordinate

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Write affine transformations as matrix multiplication

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} \hat{x} \\ \hat{y} \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Extended Coordinates

``Extend" 2D coordinate to 3D coordinate

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Write affine transformations as matrix multiplication

$$\begin{pmatrix} \hat{x} \\ \hat{y} \end{pmatrix} = \begin{pmatrix} a & b \\ c & d \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} t_x \\ t_y \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} \hat{x} \\ \hat{y} \\ 1 \end{pmatrix} = \begin{pmatrix} a & b & t_x \\ c & d & t_y \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Easy to concatenate transformations via matrix multiplication

Hierarchy of 2D Transformations

rigid $$\begin{pmatrix}\hat{x}\\ \hat{y}\end{pmatrix} = \begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix}$$

similarity $$\begin{pmatrix}\hat{x}\\ \hat{y}\end{pmatrix} = s\begin{pmatrix}\cos\alpha & -\sin\alpha\\ \sin\alpha & \cos\alpha\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix}$$

affine $$\begin{pmatrix}\hat{x}\\ \hat{y}\end{pmatrix} = \begin{pmatrix}a & b\\ c & d\end{pmatrix}\begin{pmatrix}x\\ y\end{pmatrix} + \begin{pmatrix}t_x\\ t_y\end{pmatrix}$$

perspective $$\hat{w}\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}a & b & c\\ d & e & f\\ g & h & i\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

Hierarchy of 2D Transformations

rigid

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}\cos\alpha & -\sin\alpha & t_x\\ \sin\alpha & \cos\alpha & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

similarity

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}s\cos\alpha & -s\sin\alpha & t_x\\ s\sin\alpha & s\cos\alpha & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

affine

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}a & b & t_x\\ c & d & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

perspective

$$\hat{w}\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}a & b & c\\ d & e & f\\ g & h & i\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

Hierarchy of 2D Transformations

rigid

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}\cos\alpha & -\sin\alpha & t_x\\ \sin\alpha & \cos\alpha & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

similarity

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}s\cos\alpha & -s\sin\alpha & t_x\\ s\sin\alpha & s\cos\alpha & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

affine

$$\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix} = \begin{pmatrix}a & b & t_x\\ c & d & t_y\\ 0 & 0 & 1\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

perspective

$$\boxed{\hat{w}\begin{pmatrix}\hat{x}\\ \hat{y}\\ 1\end{pmatrix}} = \begin{pmatrix}a & b & c\\ d & e & f\\ g & h & i\end{pmatrix}\begin{pmatrix}x\\ y\\ 1\end{pmatrix}$$

Homogenous Coordinates

Homogenous coordinates

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \quad \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

Homogenous Coordinates

Homogenous coordinates

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow w \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad , \quad w \neq 0$$

Homogenous Coordinates

Homogenous coordinates

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow w \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad , \quad w \neq 0$$

De-homogenization:

$$w \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x/w \\ y/w \end{pmatrix}$$

Homogenous Coordinates

Homogenous coordinates

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow w \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \quad , \quad w \neq 0$$

De-homogenization:

$$w \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \rightarrow \begin{pmatrix} x/w \\ y/w \end{pmatrix}$$

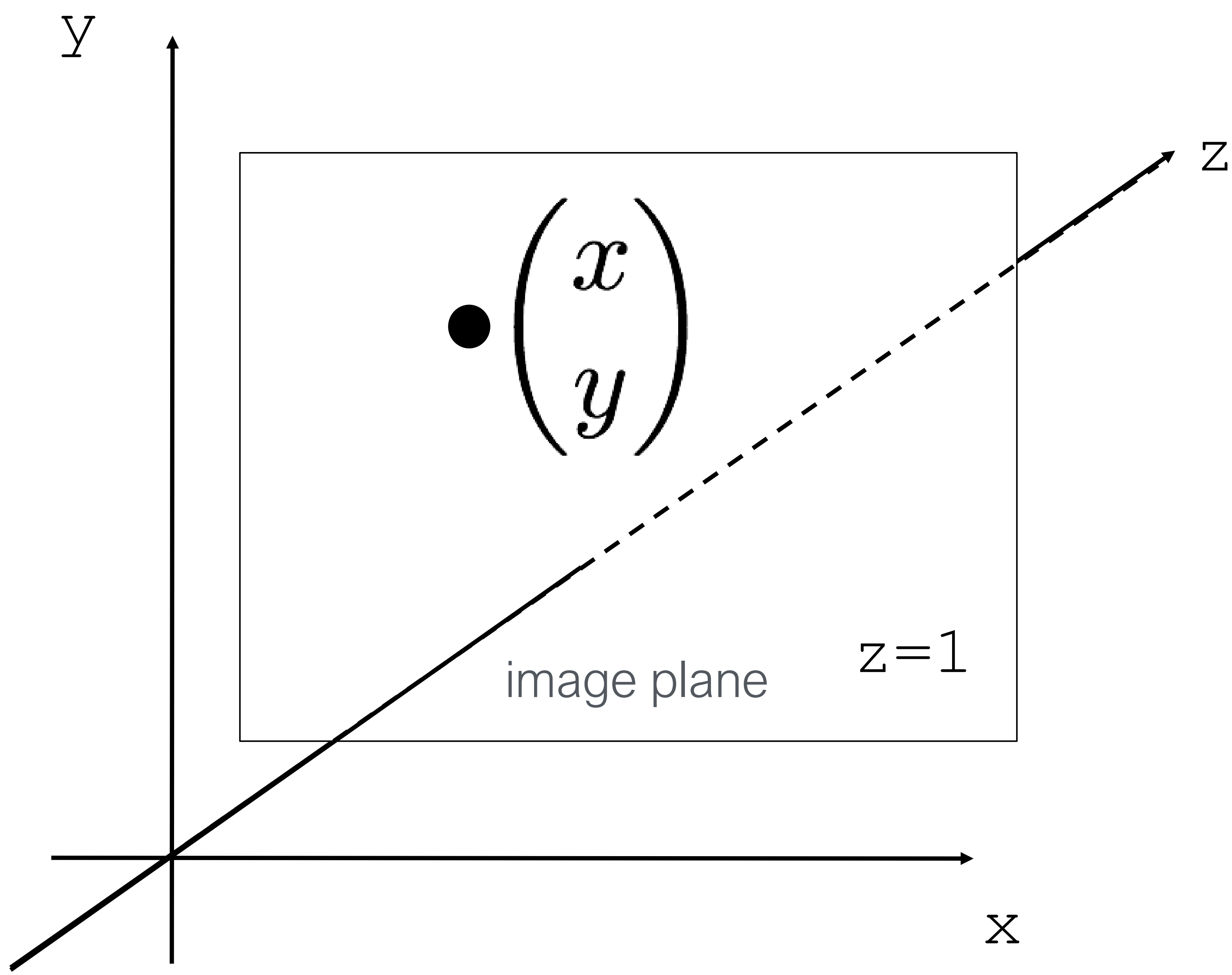
2D projective space: $\mathcal{P}^2 = \mathcal{R}^3 \setminus \{(0,0,0)^T\}$

Geometric Interpretation

Mapping 2D points $\longleftrightarrow$ 3D lines

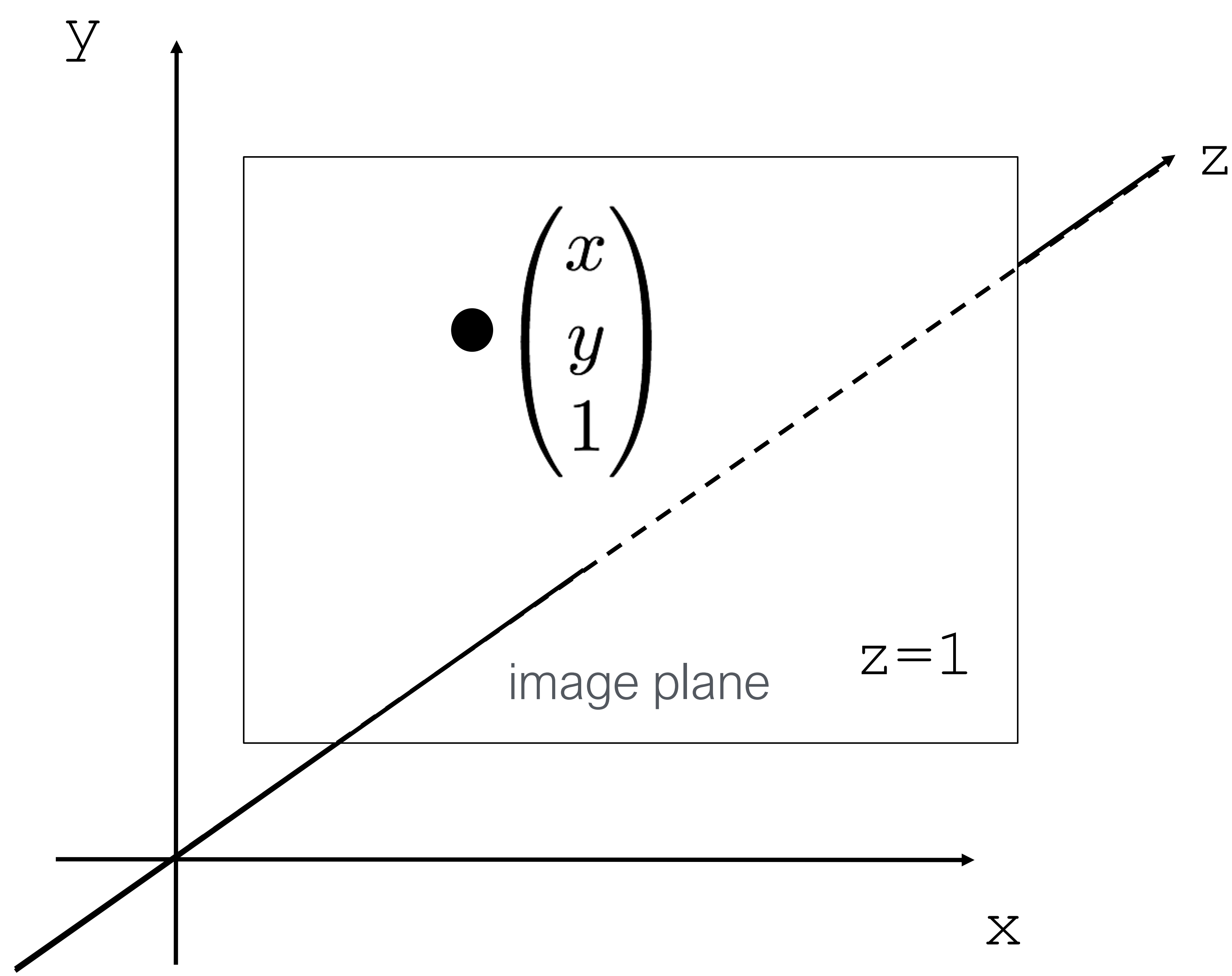
Geometric Interpretation

Mapping 2D points $\leftrightarrow$ 3D lines



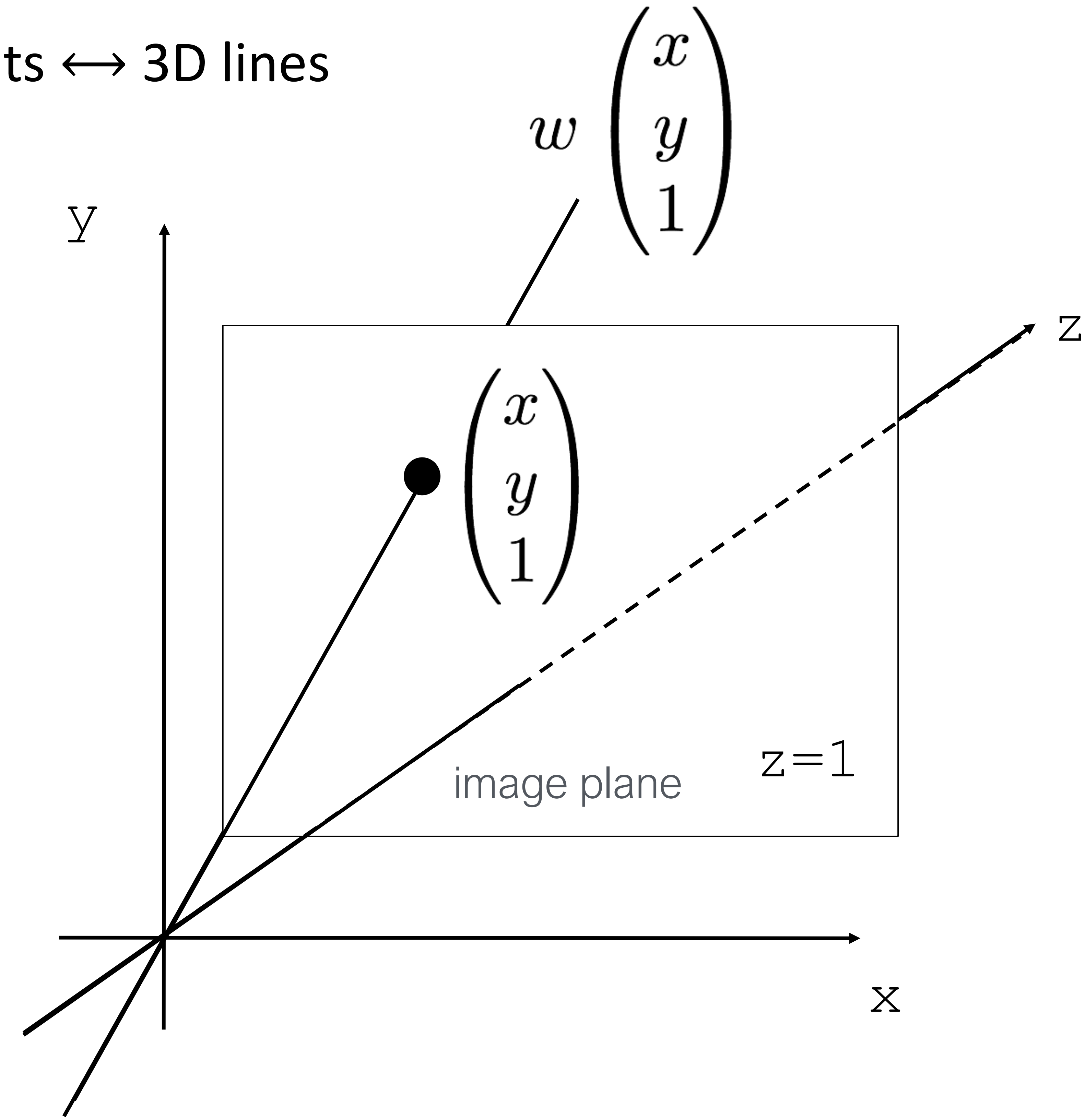
Geometric Interpretation

Mapping 2D points $\leftrightarrow$ 3D lines



Geometric Interpretation

Mapping 2D points ⟷ 3D lines



Point $\Leftrightarrow$ Line Duality

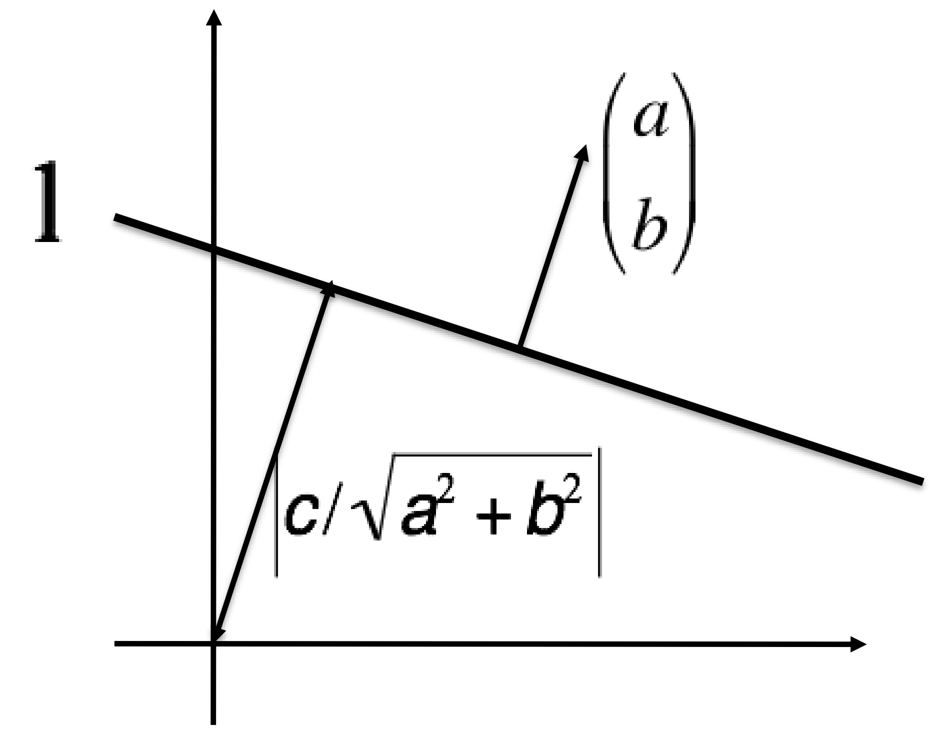
(Homogenous) 2D line representation:

$$ax + by + c = 0$$

Point ⇔ Line Duality

(Homogenous) 2D line representation:

$$ax + by + c = 0$$

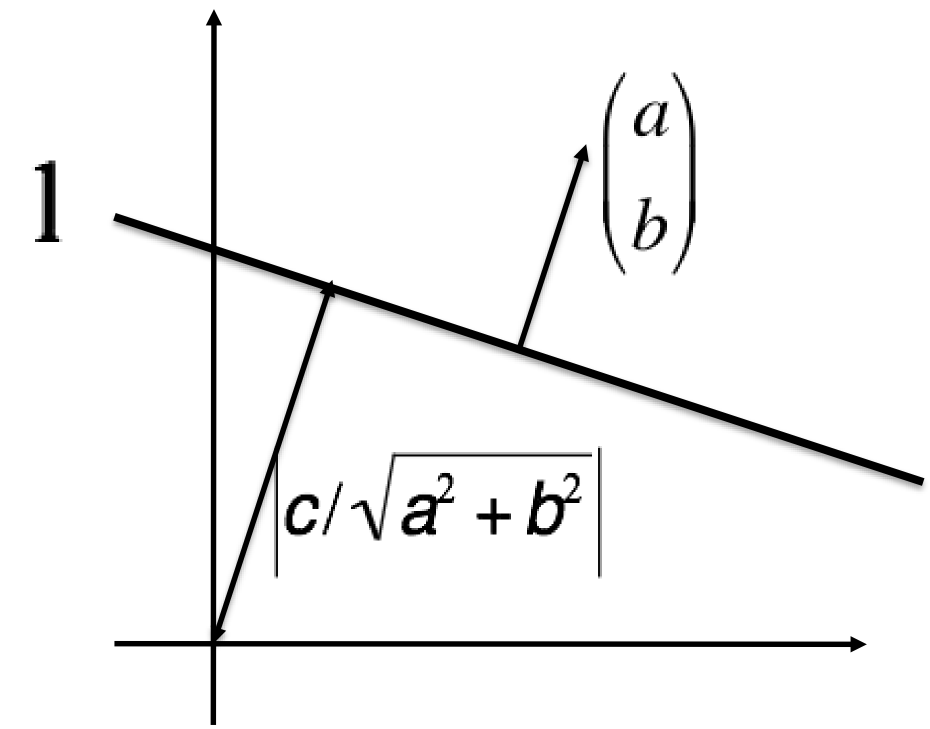


Point ⇔ Line Duality

(Homogenous) 2D line representation:

$$ax + by + c = 0 \Leftrightarrow (a, b, c) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

$$\Leftrightarrow l^T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

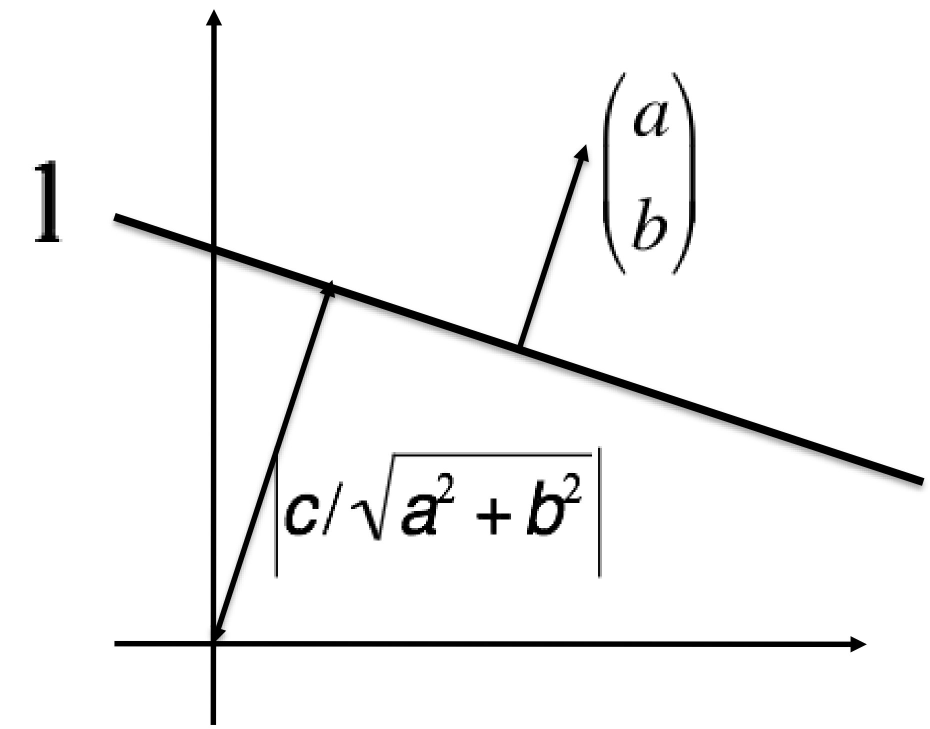


Point $\Leftrightarrow$ Line Duality

(Homogenous) 2D line representation:

$$ax + by + c = 0 \Leftrightarrow (a, b, c) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

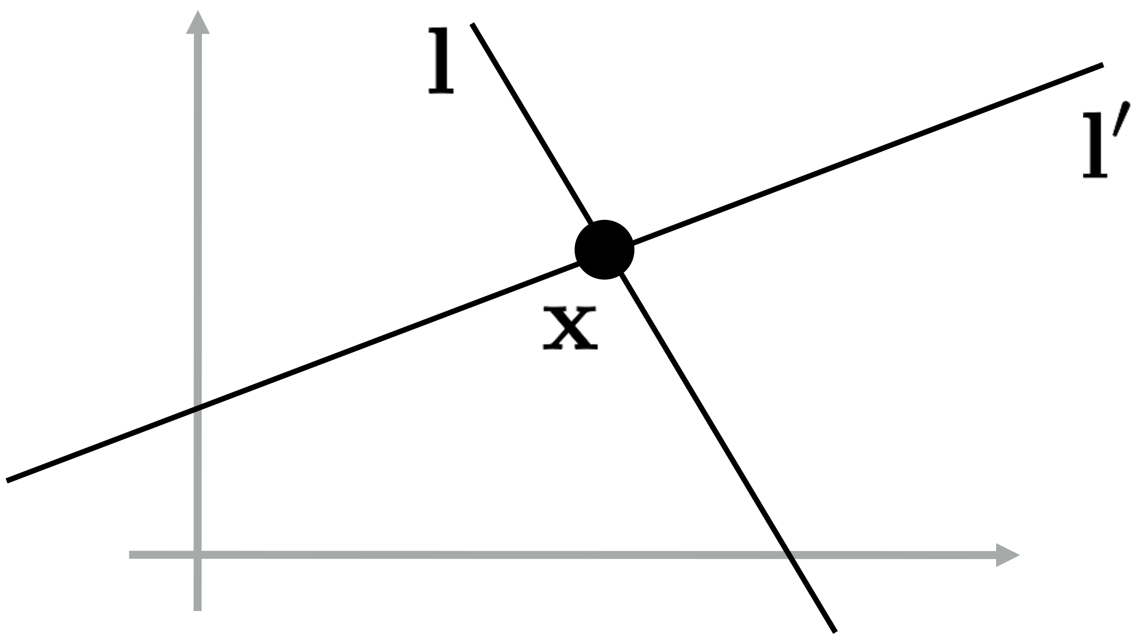
$$\Leftrightarrow l^T \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$



2D lines are points in $\mathcal{R}^3$

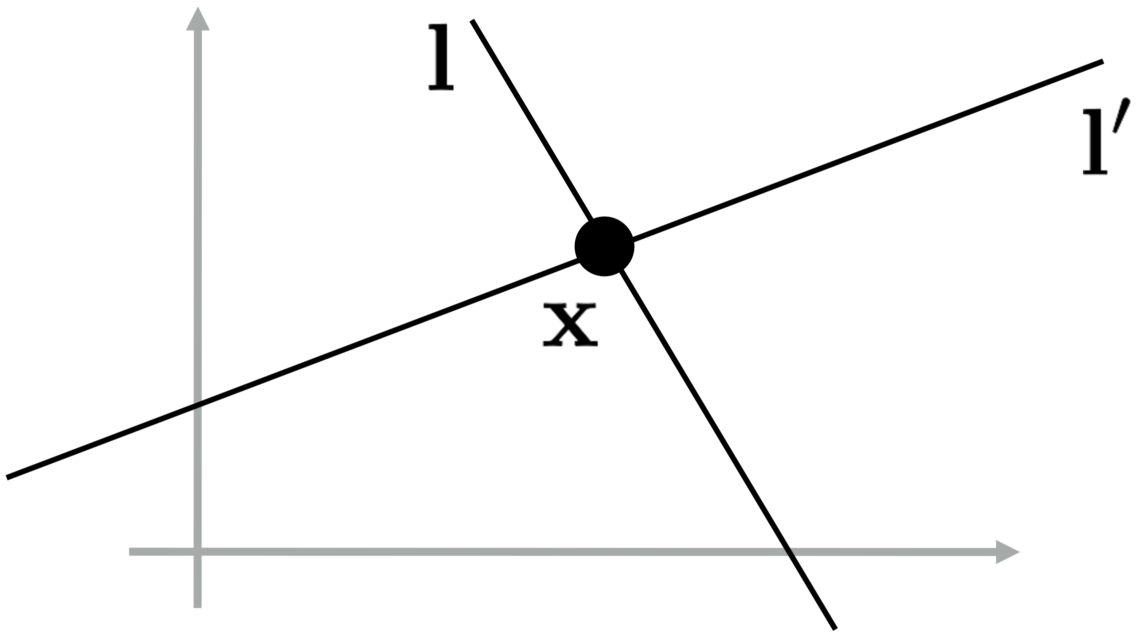
Point ⇔ Line Duality

Intersection of two lines



Point $\Leftrightarrow$ Line Duality

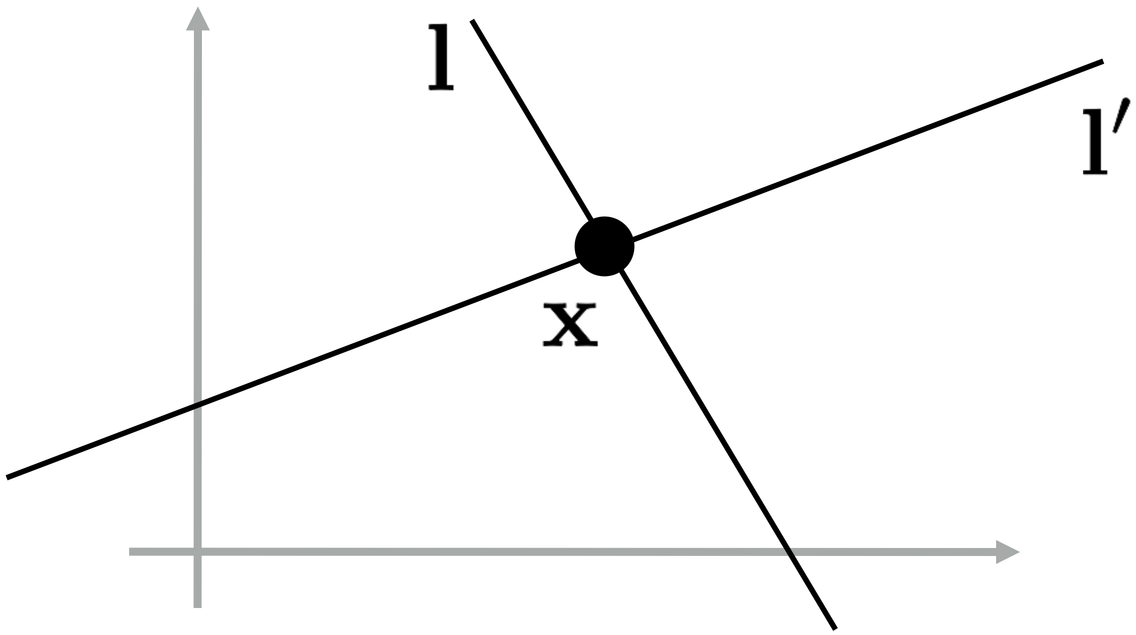
Intersection of two lines



$$\mathbf{l}^T\mathbf{x} = 0$$
$$\mathbf{l}'^T\mathbf{x} = 0$$

Point ⇔ Line Duality

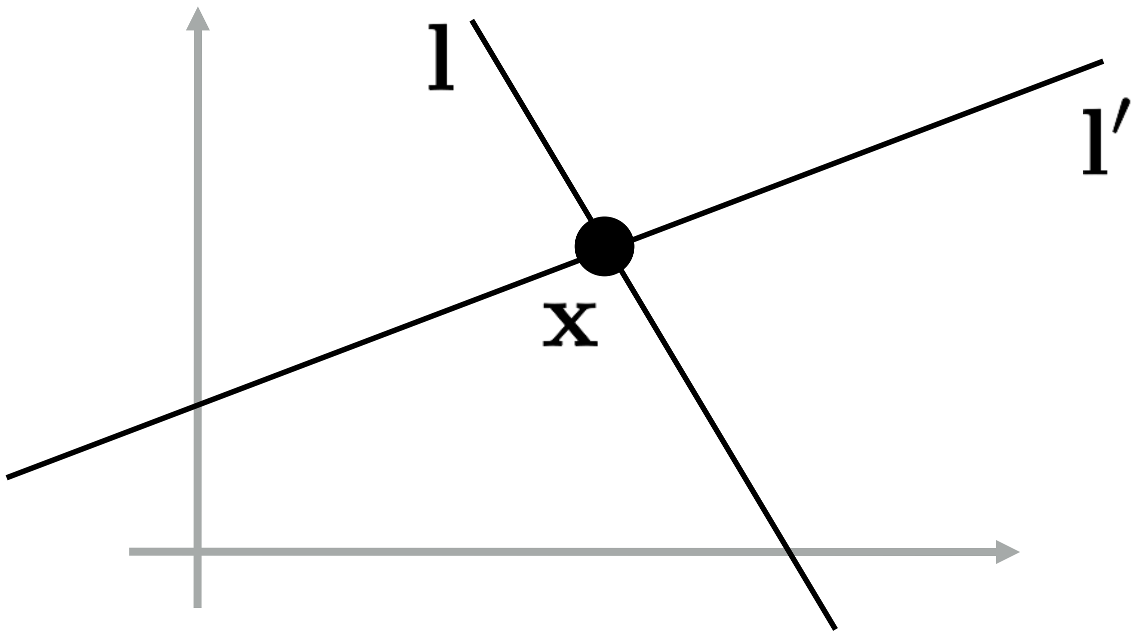
Intersection of two lines



$$\left.\begin{array}{r}\mathbf{l}^T\mathbf{x} = 0\\ \mathbf{l}'^T\mathbf{x} = 0\end{array}\right\}\mathbf{x} = \mathbf{l} \times \mathbf{l}'$$

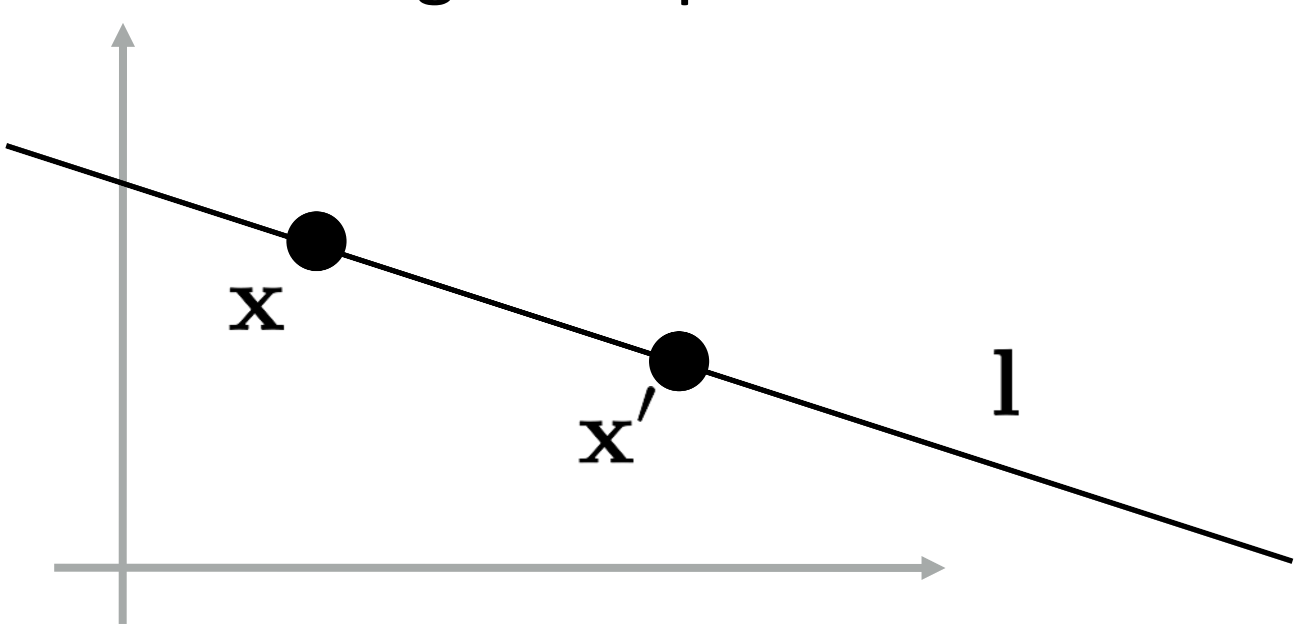
Point ⇔ Line Duality

Intersection of two lines



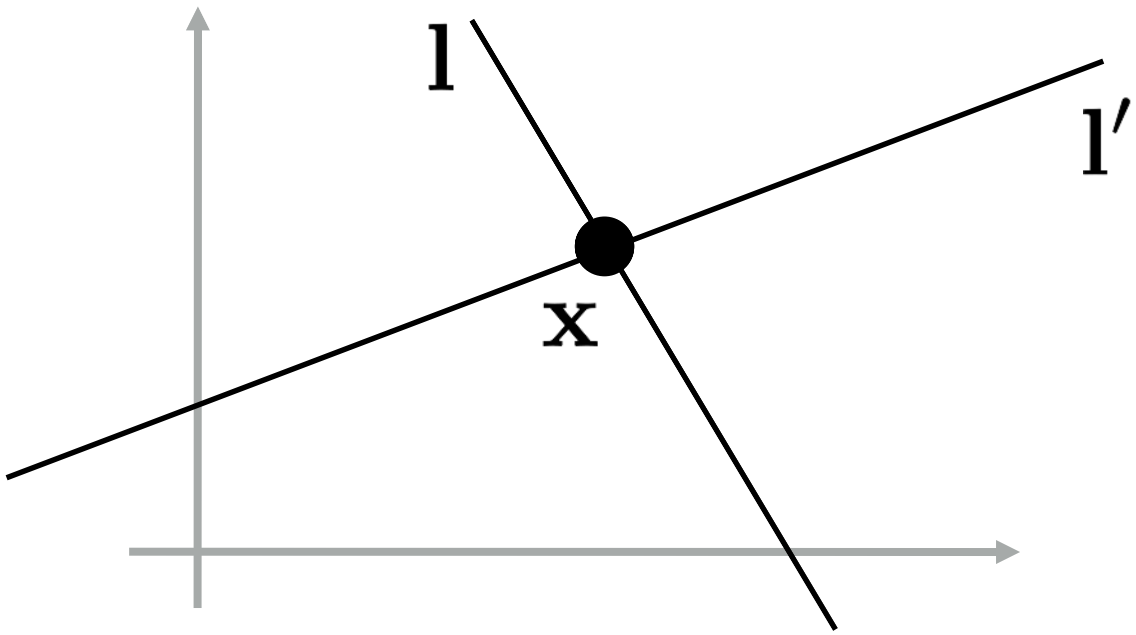
$$\left.\begin{array}{r}\mathbf{l}^T\mathbf{x}=0\\ \mathbf{l}'^T\mathbf{x}=0\end{array}\right\}\mathbf{x}=\mathbf{l}\times\mathbf{l}'$$

Lines through two points



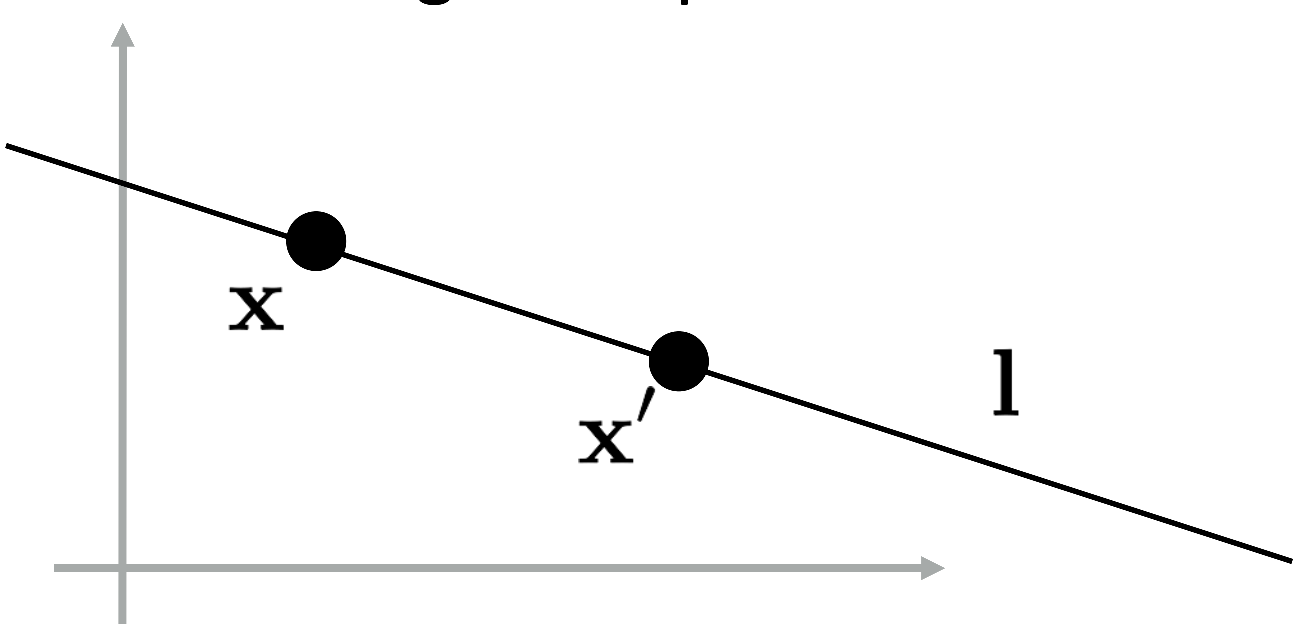
Point ⇔ Line Duality

Intersection of two lines



$$\left.\begin{array}{r}\mathbf{l}^T\mathbf{x}=0\\ \mathbf{l}'^T\mathbf{x}=0\end{array}\right\}\mathbf{x}=\mathbf{l}\times\mathbf{l}'$$

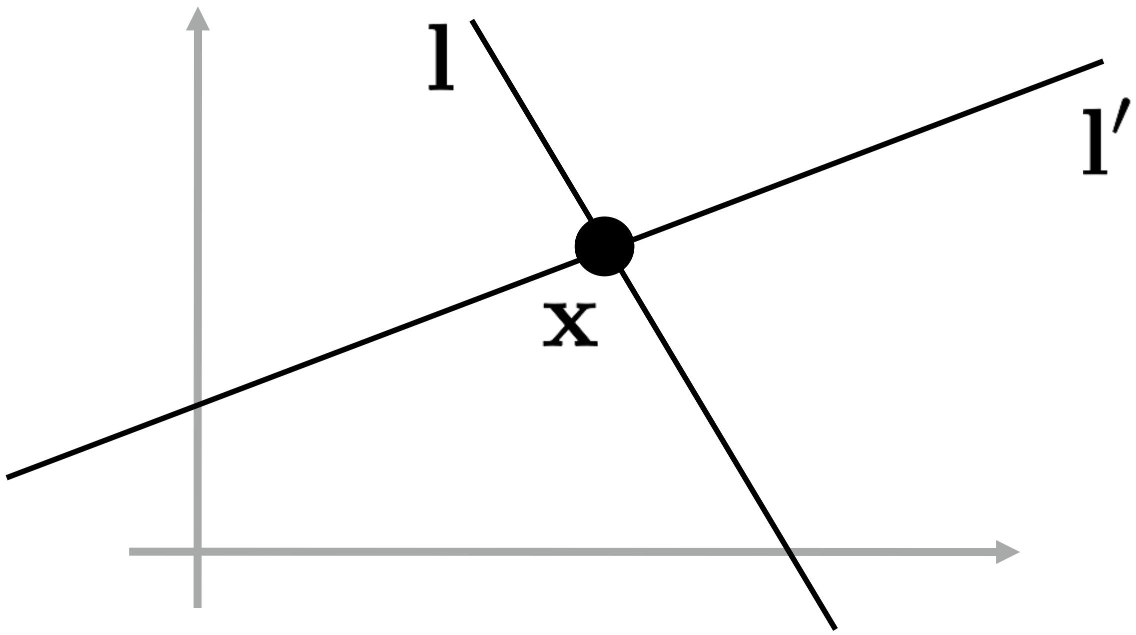
Lines through two points



$$\begin{array}{r}\mathbf{l}^T\mathbf{x}=0\\ \mathbf{l}^T\mathbf{x}'=0\end{array}$$

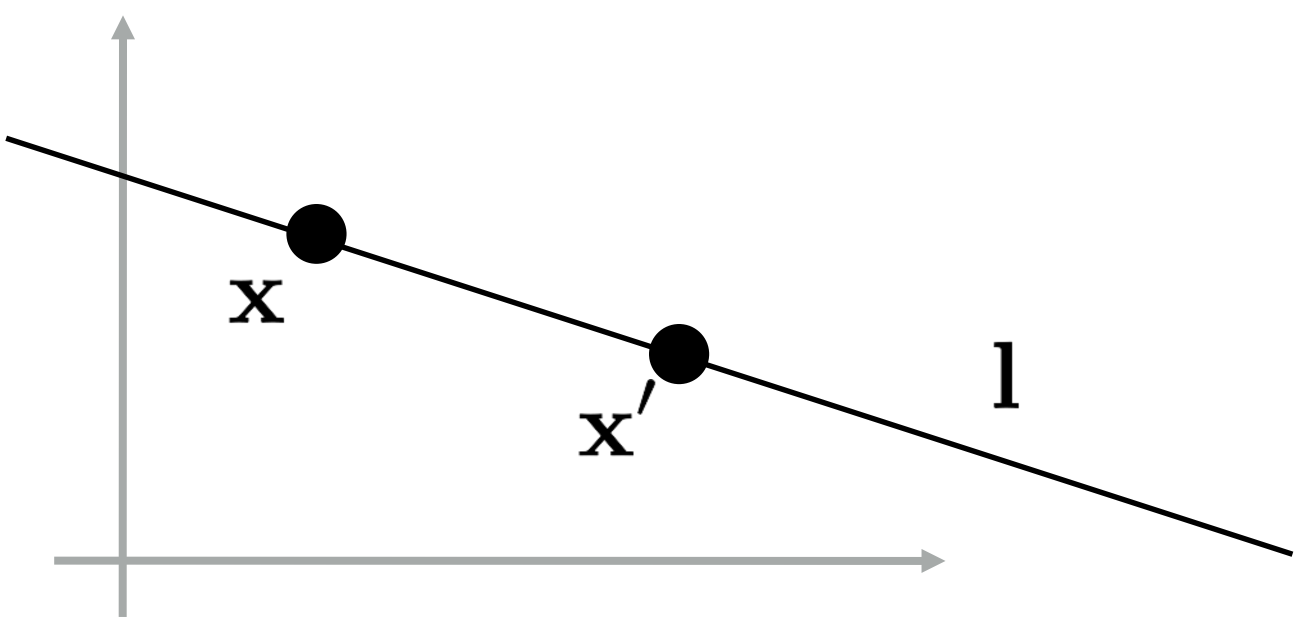
Point $\Leftrightarrow$ Line Duality

Intersection of two lines



$$\left.\begin{aligned} \mathbf{l}^T\mathbf{x} = 0 \\ \mathbf{l}'^T\mathbf{x} = 0 \end{aligned}\right\} \mathbf{x} = \mathbf{l} \times \mathbf{l}'$$

Lines through two points



$$\left.\begin{aligned} \mathbf{l}^T\mathbf{x} = 0 \\ \mathbf{l}^T\mathbf{x}' = 0 \end{aligned}\right\} \mathbf{l} = \mathbf{x} \times \mathbf{x}'$$

2D Projective Transformations

Perspective mapping (projectivity, **homography**)

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

or $\hat{\mathbf{x}} = \mathbf{H}\mathbf{x}$

2D Projective Transformations

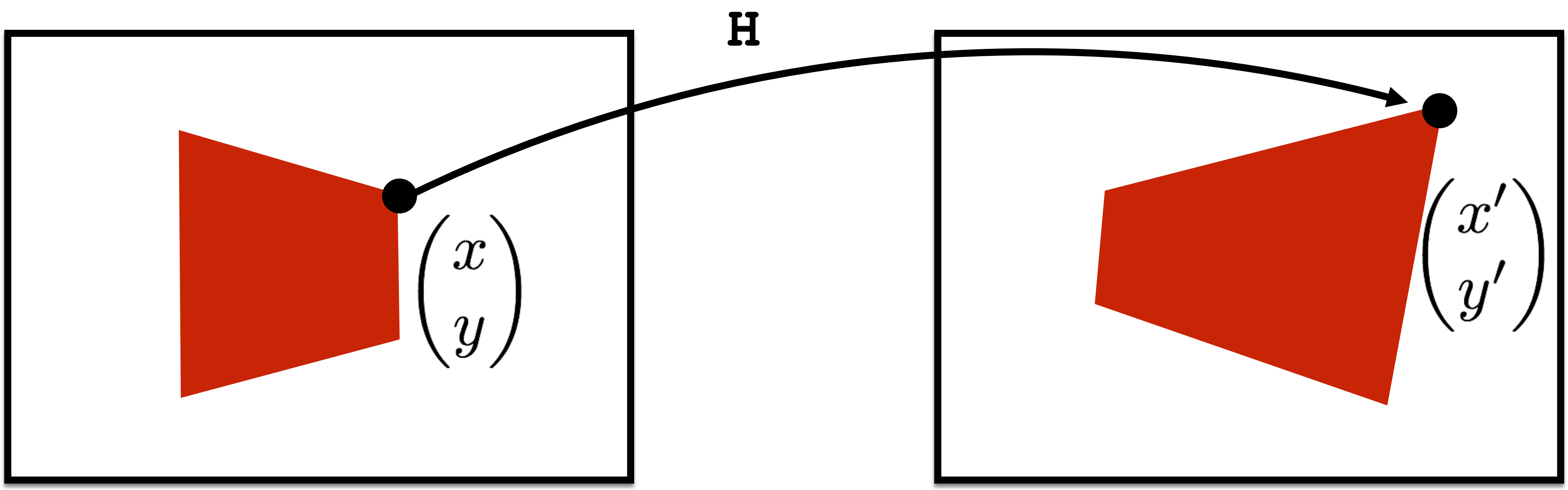
Perspective mapping (projectivity, **homography**)

$$\begin{pmatrix} \hat{x} \\ \hat{y} \\ \hat{z} \end{pmatrix} = \begin{pmatrix} a & b & c \\ d & e & f \\ g & h & i \end{pmatrix} \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

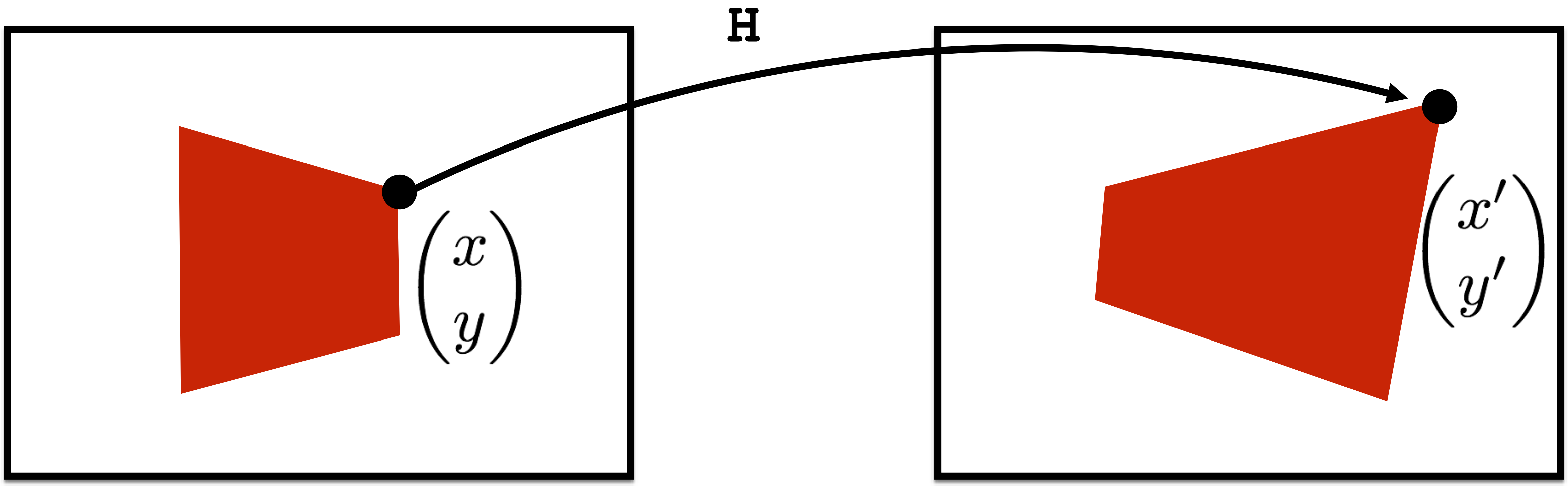
or $\hat{\mathbf{x}} = \mathbf{H}\mathbf{x}$

- **H** needs to be invertible
- **H** has 8 Degrees-of-Freedom (DoF)
  - **H** and s**H** $(s \neq 0)$ define the same transformation

Using Homogenous Coordinates

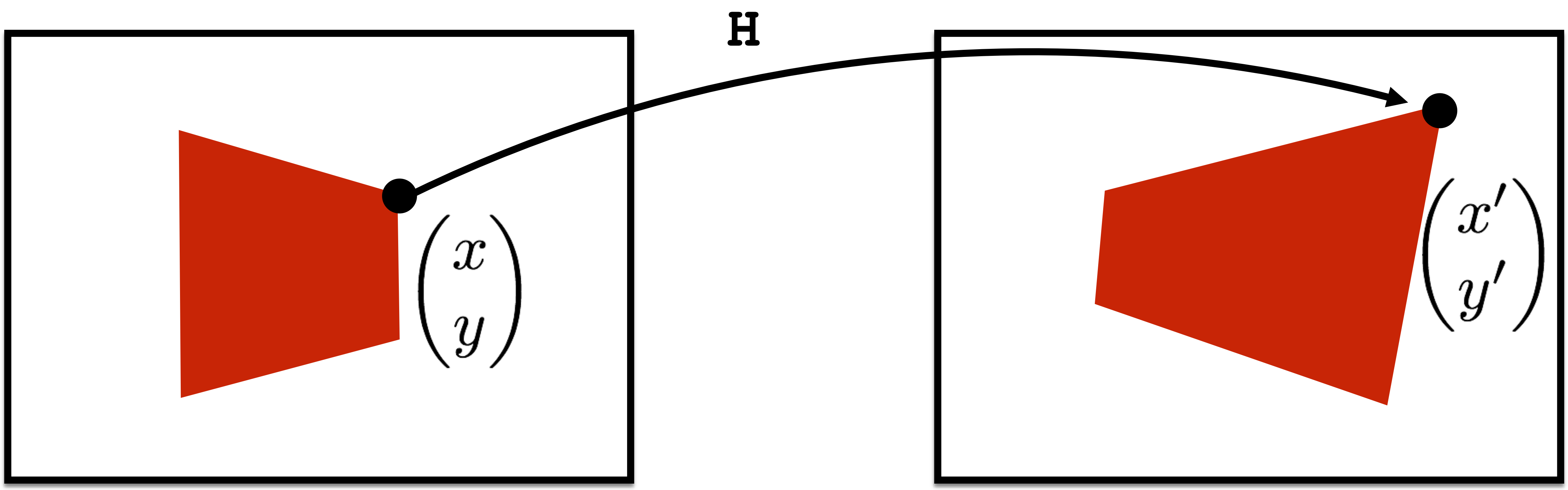


Using Homogenous Coordinates



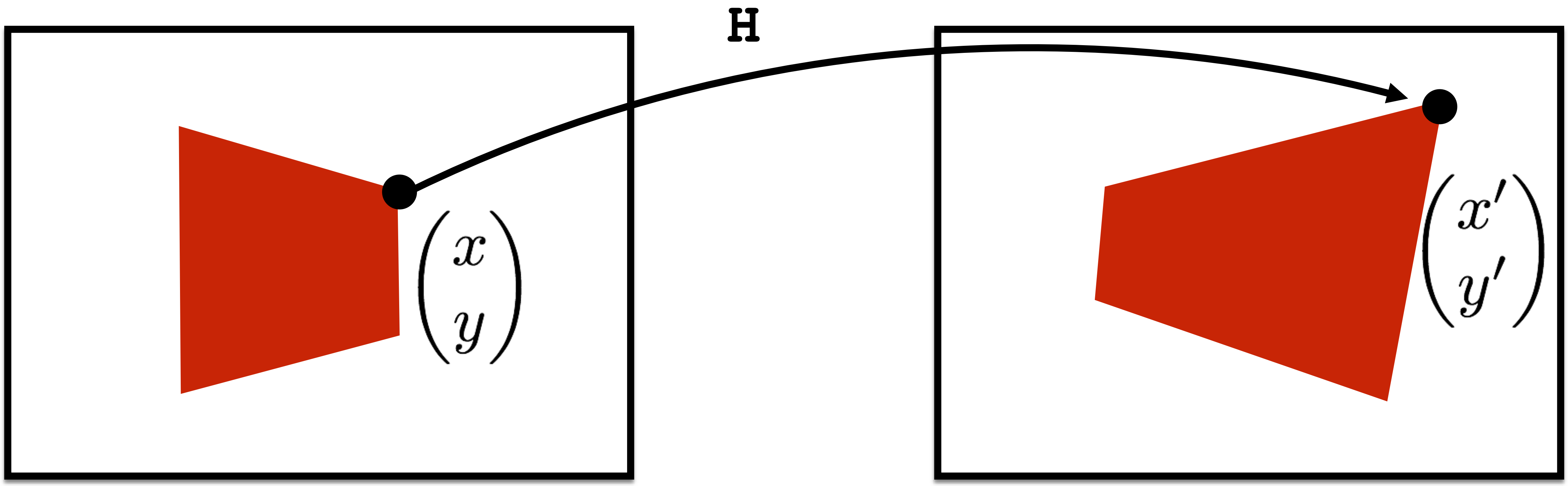
- "Homogenize": $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

Using Homogenous Coordinates



- "Homogenize": $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$
- Apply $\mathtt{H}$: $\begin{pmatrix} x'' \\ y'' \\ z'' \end{pmatrix} = \mathtt{H} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

Using Homogenous Coordinates



- "Homogenize": $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$
- Apply $\mathbf{H}$: $\begin{pmatrix} x'' \\ y'' \\ z'' \end{pmatrix} = \mathrm{H} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$
- De-homogenize: $\begin{pmatrix} x'' \\ y'' \\ z'' \end{pmatrix} \mapsto \begin{pmatrix} x''/z'' \\ y''/z'' \end{pmatrix} = \begin{pmatrix} x' \\ y' \end{pmatrix}$

Homography Estimation

Compute homography $\mathbf{H}$ from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i$$

$$\lambda_i \neq 0$$

Homography Estimation

Compute homography **H** from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \qquad \mathbf{x}'_i = (x'_i, y'_i, w'_i)^\top \qquad \mathbf{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\top}\mathbf{x}_i \\ \mathbf{h}^{2\top}\mathbf{x}_i \\ \mathbf{h}^{3\top}\mathbf{x}_i \end{pmatrix}$$

$$\lambda_i \neq 0$$

$$H = \begin{pmatrix} - & h^{1T} & - \\ - & h^{2T} & - \\ - & h^{3T} & - \end{pmatrix}$$

$$h^i \in \mathbb{R}^3$$

Homography Estimation

Compute homography $\mathbf{H}$ from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \qquad \mathbf{x}'_i = \left(x'_i, y'_i, w'_i\right)^\top \qquad \mathbf{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1^\top}\mathbf{x}_i \\ \mathbf{h}^{2^\top}\mathbf{x}_i \\ \mathbf{h}^{3^\top}\mathbf{x}_i \end{pmatrix}$$

$$\lambda_i \neq 0$$

Only defined up to scale, use cross-product instead

Homography Estimation

Compute homography $\mathbf{H}$ from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \qquad \mathbf{x}'_i = \left(x'_i, y'_i, w'_i\right)^\top \qquad \mathbf{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\top}\mathbf{x}_i \\ \mathbf{h}^{2\top}\mathbf{x}_i \\ \mathbf{h}^{3\top}\mathbf{x}_i \end{pmatrix}$$

$$\lambda_i \neq 0$$

Only defined up to scale, use cross-product instead

$$\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = 0$$

Homography Estimation

Compute homography $\mathbf{H}$ from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \qquad \mathbf{x}'_i = \left(x'_i, y'_i, w'_i\right)^\top \qquad \mathbf{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\top}\mathbf{x}_i \\ \mathbf{h}^{2\top}\mathbf{x}_i \\ \mathbf{h}^{3\top}\mathbf{x}_i \end{pmatrix}$$

$$\lambda_i \neq 0$$

Only defined up to scale, use cross-product instead

$$\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = 0 \iff \mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \begin{pmatrix} y'_i\mathbf{h}^{3\top}\mathbf{x}_i - w'_i\mathbf{h}^{2\top}\mathbf{x}_i \\ w'_i\mathbf{h}^{1\top}\mathbf{x}_i - x'_i\mathbf{h}^{3\top}\mathbf{x}_i \\ x'_i\mathbf{h}^{2\top}\mathbf{x}_i - y'_i\mathbf{h}^{1\top}\mathbf{x}_i \end{pmatrix}$$

Homography Estimation

Compute homography **H** from 2D-2D matches:

$$\lambda_i \mathbf{x}'_i = \mathbf{H}\mathbf{x}_i \qquad \mathbf{x}'_i = (x'_i, y'_i, w'_i)^\top \qquad \mathbf{H}\mathbf{x}_i = \begin{pmatrix} \mathbf{h}^{1\top}\mathbf{x}_i \\ \mathbf{h}^{2\top}\mathbf{x}_i \\ \mathbf{h}^{3\top}\mathbf{x}_i \end{pmatrix}$$

$\lambda_i \neq 0$

Only defined up to scale, use cross-product instead

$$\mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = 0 \Leftrightarrow \mathbf{x}'_i \times \mathbf{H}\mathbf{x}_i = \begin{pmatrix} y'_i\mathbf{h}^{3\top}\mathbf{x}_i - w'_i\mathbf{h}^{2\top}\mathbf{x}_i \\ w'_i\mathbf{h}^{1\top}\mathbf{x}_i - x'_i\mathbf{h}^{3\top}\mathbf{x}_i \\ x'_i\mathbf{h}^{2\top}\mathbf{x}_i - y'_i\mathbf{h}^{1\top}\mathbf{x}_i \end{pmatrix}$$

$$\Rightarrow \begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

Homography Estimation

Single match gives 3 equations (2 linearly independent)

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i^\top \\ -y_i'\mathbf{x}_i^\top & x_i'\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

Homography Estimation

Single match gives 3 equations (2 linearly independent)

$$\begin{bmatrix} \mathbf{0}^\top & -w_i'\mathbf{x}_i^\top & y_i'\mathbf{x}_i^\top \\ w_i'\mathbf{x}_i^\top & \mathbf{0}^\top & -x_i'\mathbf{x}_i^\top \\ -y_i'\mathbf{x}_i^\top & x_i'\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

Stack equations from 4 matches (8x9 or 12x9 matrix of rank 8)

$$\mathtt{A}\mathbf{h} = \mathbf{0}$$

Homography Estimation

Single match gives 3 equations (2 linearly independent)

$$\begin{bmatrix} \mathbf{0}^\top & -w'_i\mathbf{x}_i^\top & y'_i\mathbf{x}_i^\top \\ w'_i\mathbf{x}_i^\top & \mathbf{0}^\top & -x'_i\mathbf{x}_i^\top \\ -y'_i\mathbf{x}_i^\top & x'_i\mathbf{x}_i^\top & \mathbf{0}^\top \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

Stack equations from 4 matches (8x9 or 12x9 matrix of rank 8)

$$\mathtt{A}\mathbf{h} = \mathbf{0}$$

No exact solution due to noise. Use least squares solution instead:

- Constraint needed to avoid $\mathbf{h} = \mathbf{0}$, use $||\mathbf{h}||=1$
- Minimize $||\mathtt{A}\mathbf{h}||=1$

Direct Linear Transform (DLT)

<u>Objective</u>

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Direct Linear Transform (DLT)

Objective

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Algorithm

- For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ compute A_i. Usually only two first rows needed.

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Direct Linear Transform (DLT)

Objective

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Algorithm

- For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ compute A_i. Usually only two first rows needed.
- Assemble *n* 2x9 matrices A_i into a single $2n\mathrm{x}9$ matrix A

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Direct Linear Transform (DLT)

Objective

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = H\mathbf{x}_i$

Algorithm

- For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ compute A_i. Usually only two first rows needed.
- Assemble *n* 2x9 matrices A_i into a single $2n\text{x}9$ matrix A
- Obtain SVD of A: $A = U\Sigma V^T$. Solution for h is last column of V

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Direct Linear Transform (DLT)

Objective

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Algorithm

- For each correspondence $\mathbf{x}_i \leftrightarrow \mathbf{x}_i'$ compute A_i. Usually only two first rows needed.
- Assemble *n* 2x9 matrices A_i into a single $2n \times 9$ matrix A
- Obtain SVD of A: $\mathrm{A} = \mathrm{U}\Sigma\mathrm{V}^\mathrm{T}$. Solution for h is last column of V
- Determine H from h

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$\sim 10^2$

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$\sim 10^2$ $\sim 10^2$

Coordinate Normalization

$$\begin{array}{ccccccccc}
\multicolumn{9}{c}{\begin{bmatrix} 0 & 0 & 0 & -x'_i & -y'_i & -1 & y'_i x_i & y'_i y_i & y'_i \\ x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}} \\
\sim 10^2 & \sim 10^2 & 1 & & & & & &
\end{array}$$

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x'_i & -y'_i & -1 & y'_i x_i & y'_i y_i & y'_i \\ x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$\sim 10^2$ $\sim 10^2$ 1 $\sim 10^2$ $\sim 10^2$ 1

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$\sim 10^2$ $\sim 10^2$ 1 $\sim 10^2$ $\sim 10^2$ 1 $\sim 10^4$

Coordinate Normalization

$$\begin{bmatrix}0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i'\end{bmatrix}\begin{pmatrix}\mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3\end{pmatrix} = \mathbf{0}$$

$\sim 10^2$ $\sim 10^2$ 1 $\sim 10^2$ $\sim 10^2$ 1 $\sim 10^4$ $\sim 10^4$

Coordinate Normalization

$$\begin{array}{l}
\begin{bmatrix} 0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0} \\
\begin{array}{ccccccccc} \sim 10^2 & \sim 10^2 & 1 & \sim 10^2 & \sim 10^2 & 1 & \sim 10^4 & \sim 10^4 & \sim 10^2 \end{array}
\end{array}$$

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x'_i & -y'_i & -1 & y'_i x_i & y'_i y_i & y'_i \\ x_i & y_i & 1 & 0 & 0 & 0 & -x'_i x_i & -x'_i y_i & -x'_i \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$\sim 10^2$ $\sim 10^2$ 1 $\sim 10^2$ $\sim 10^2$ 1 $\sim 10^4$ $\sim 10^4$ $\sim 10^2$

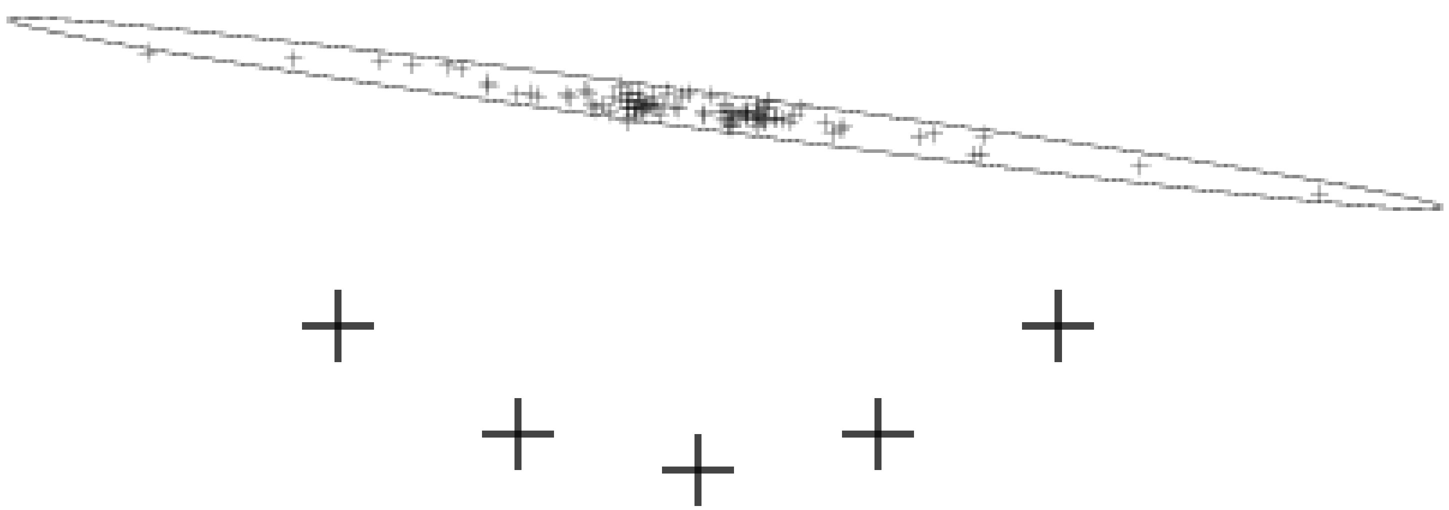
orders of magnitude difference!

Coordinate Normalization

$$\begin{bmatrix} 0 & 0 & 0 & -x_i' & -y_i' & -1 & y_i'x_i & y_i'y_i & y_i' \\ x_i & y_i & 1 & 0 & 0 & 0 & -x_i'x_i & -x_i'y_i & -x_i' \end{bmatrix} \begin{pmatrix} \mathbf{h}^1 \\ \mathbf{h}^2 \\ \mathbf{h}^3 \end{pmatrix} = \mathbf{0}$$

$$\begin{array}{ccccccccc} \sim 10^2 & \sim 10^2 & 1 & \sim 10^2 & \sim 10^2 & 1 & \sim 10^4 & \sim 10^4 & \sim 10^2 \end{array}$$

orders of magnitude difference!



Monte Carlo simulation for identity computation based on 5 points

Normalized Direct Linear Transform (DLT)

<u>Objective</u>

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Normalized Direct Linear Transform (DLT)

<u>Objective</u>

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

<u>Algorithm</u>

- Normalize points: $\tilde{\mathbf{x}}_i = \mathbf{T}_{norm}\mathbf{x}_i, \tilde{\mathbf{x}}_i' = \mathbf{T}_{norm}'\mathbf{x}_i'$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Normalized Direct Linear Transform (DLT)

Objective

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

Algorithm

- Normalize points: $\tilde{\mathbf{x}}_i = \mathbf{T}_{norm}\mathbf{x}_i, \tilde{\mathbf{x}}_i' = \mathbf{T}_{norm}'\mathbf{x}_i'$

Normalization (independently per image):

- Translate points such that centroid is at origin
- Isotropic scaling such that mean distance to origin is $\sqrt{2}$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Normalized Direct Linear Transform (DLT)

<u>Objective</u>

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

<u>Algorithm</u>

- Normalize points: $\tilde{\mathbf{x}}_i = \mathbf{T}_{norm}\mathbf{x}_i, \tilde{\mathbf{x}}_i' = \mathbf{T}_{norm}'\mathbf{x}_i'$
- Apply DLT algorithm to $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}_i'$

Normalization (independently per image):

- Translate points such that centroid is at origin
- Isotropic scaling such that mean distance to origin is $\sqrt{2}$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Normalized Direct Linear Transform (DLT)

<u>Objective</u>

Given n≥4 2D to 2D point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}_i'\}$, determine the 2D homography matrix H such that $\mathbf{x}_i' = \mathrm{H}\mathbf{x}_i$

<u>Algorithm</u>

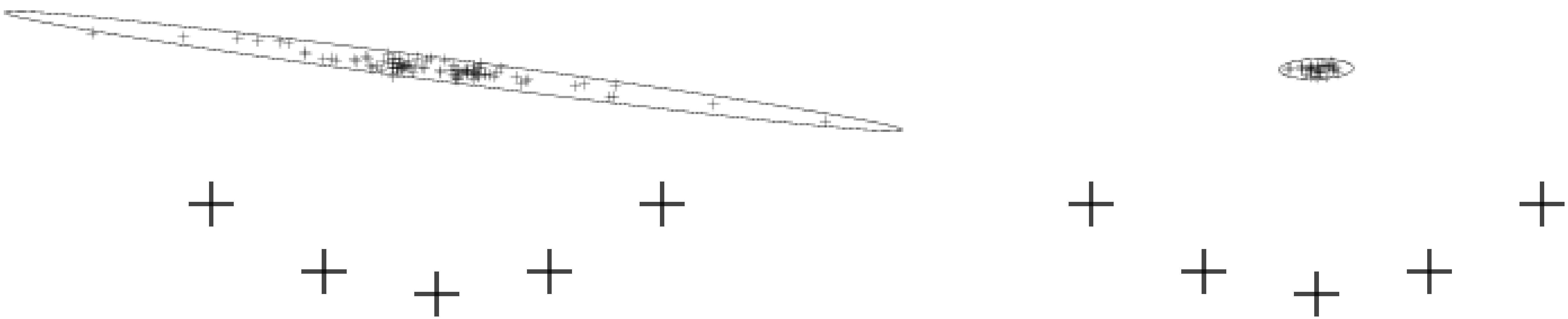
- Normalize points: $\tilde{\mathbf{x}}_i = \mathbf{T}_{norm}\mathbf{x}_i, \tilde{\mathbf{x}}_i' = \mathbf{T}_{norm}'\mathbf{x}_i'$
- Apply DLT algorithm to $\tilde{\mathbf{x}}_i \leftrightarrow \tilde{\mathbf{x}}_i'$,
- Denormalize solution: $\mathbf{H} = \mathbf{T}'^{-1}_{norm}\tilde{\mathbf{H}}\mathbf{T}_{norm}$

Normalization (independently per image):

- Translate points such that centroid is at origin
- Isotropic scaling such that mean distance to origin is $\sqrt{2}$

Hartley and Zisserman. *Multiple View Geometry in Computer Vision*, 2nd edition, Cambridge University Press, 2004.

Impact of Coordinate Normalization



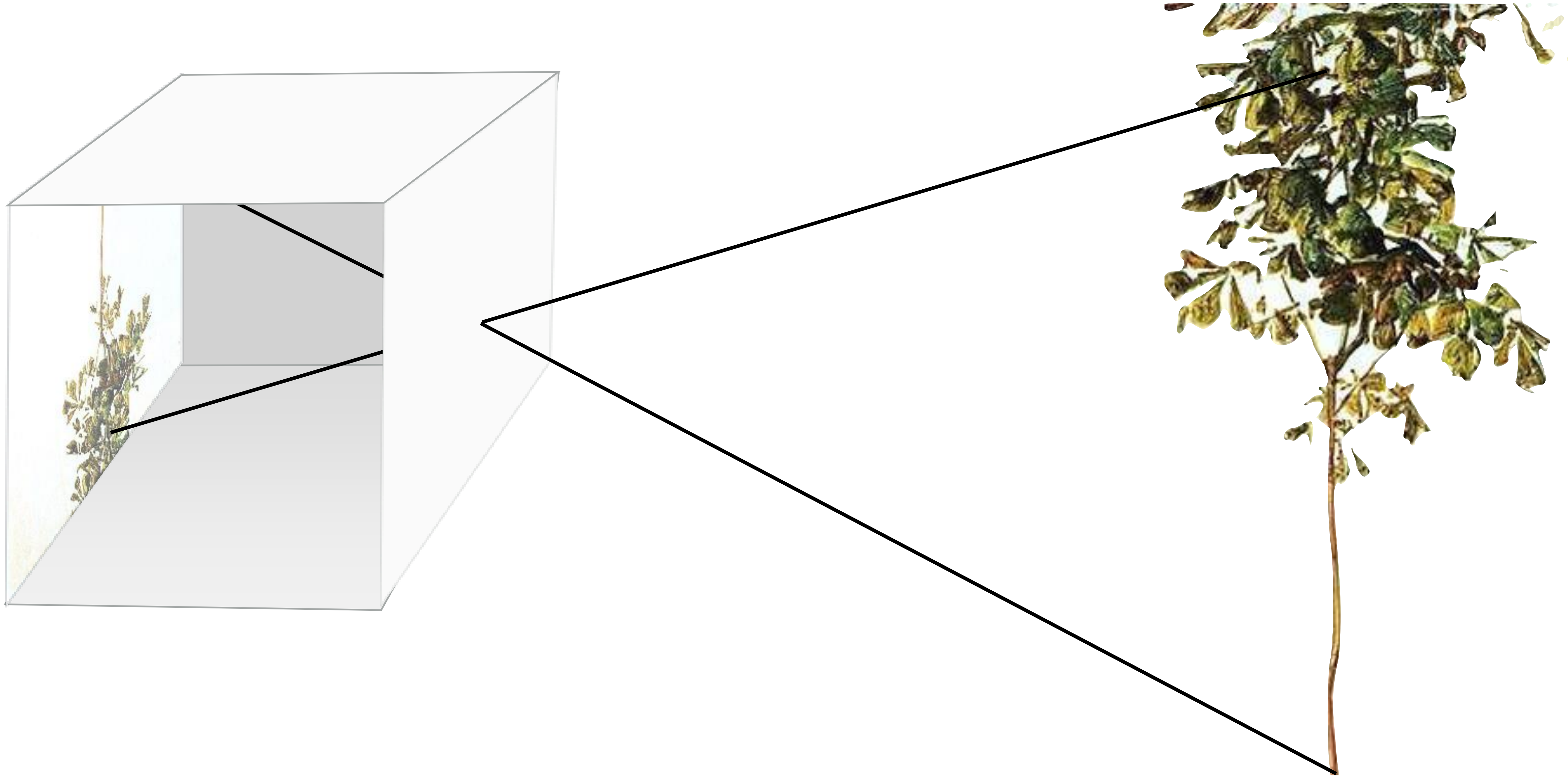
Monte Carlo simulation
for identity computation based on 5 points
(not normalized ↔ normalized)

Today

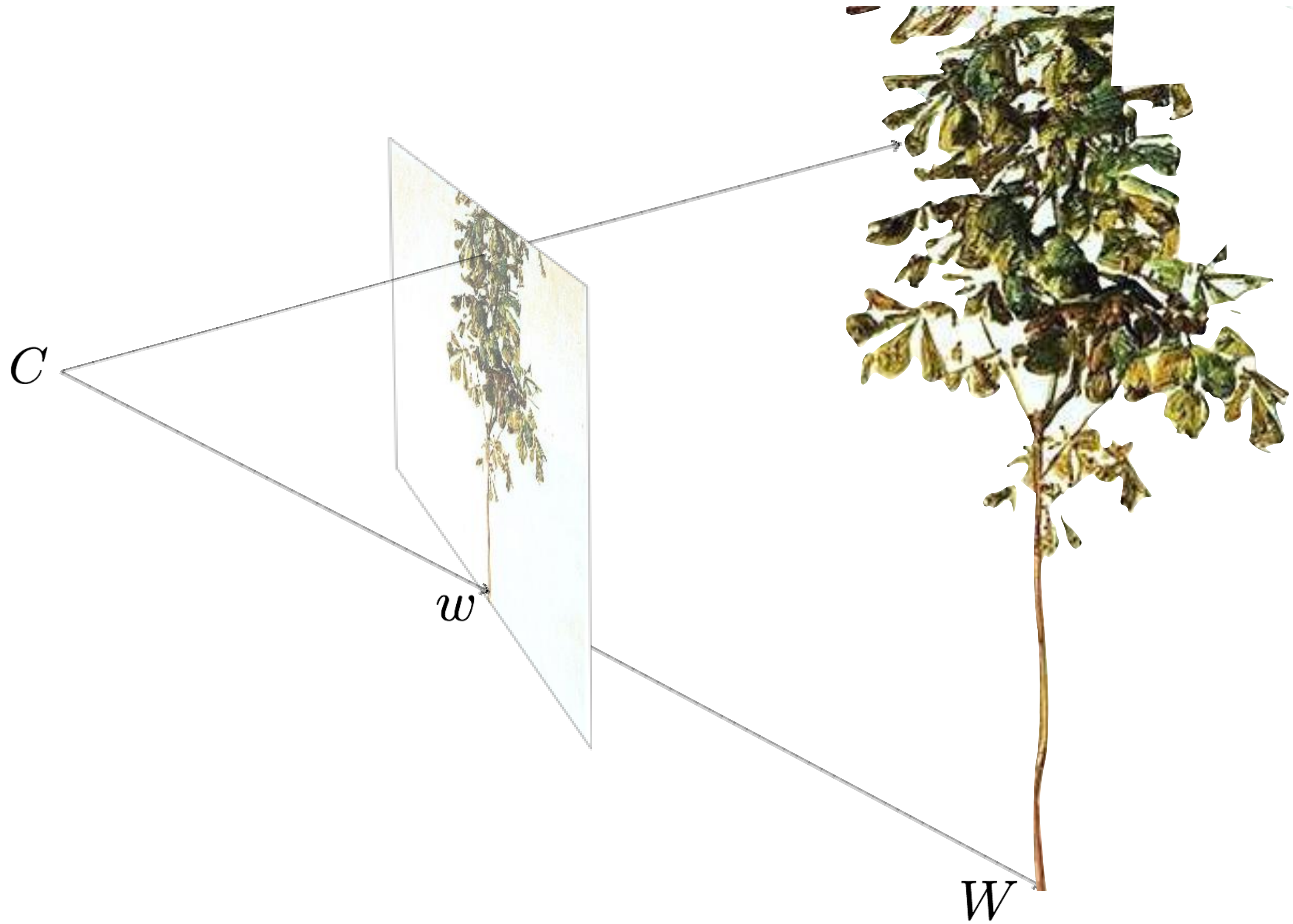
- 2D Geometry
- **Camera Geometry**

Camera Obscura

Pinhole Camera Principle



Pinhole Camera



Pinhole Camera Model

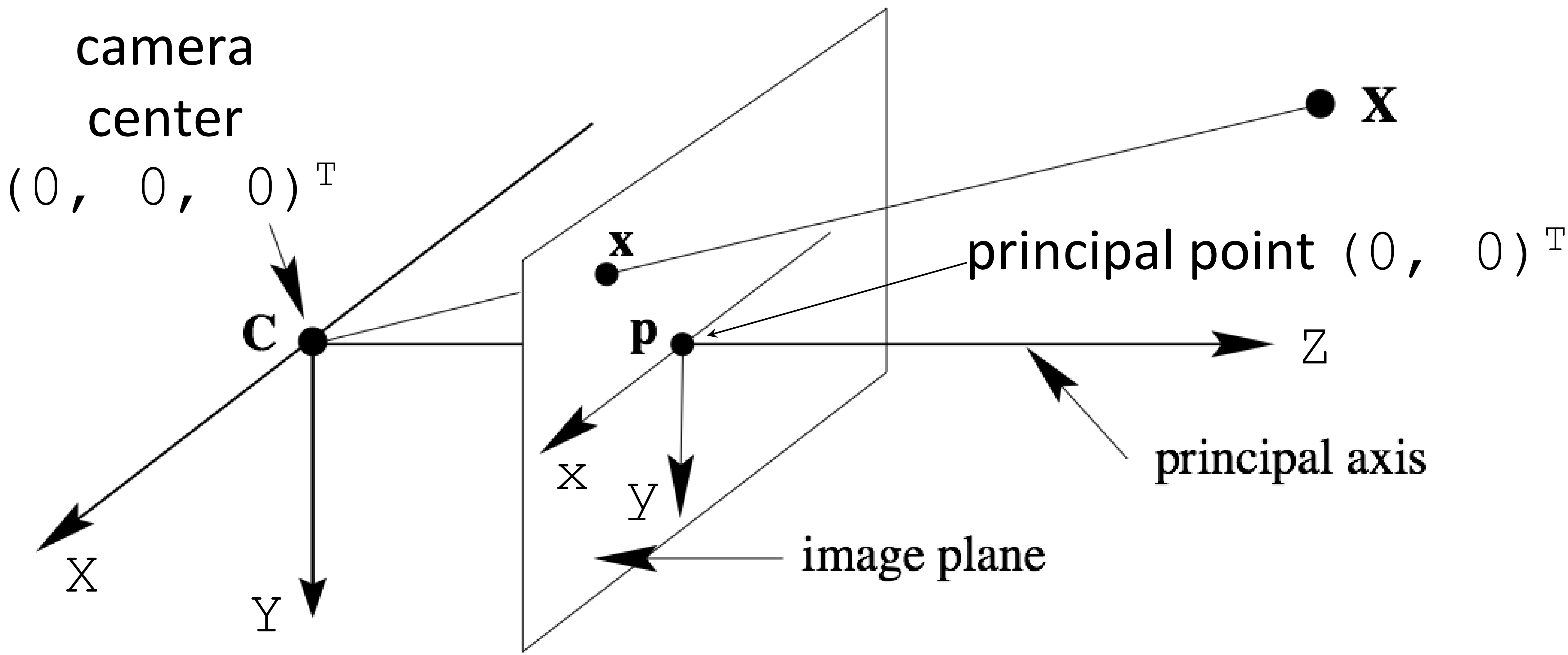
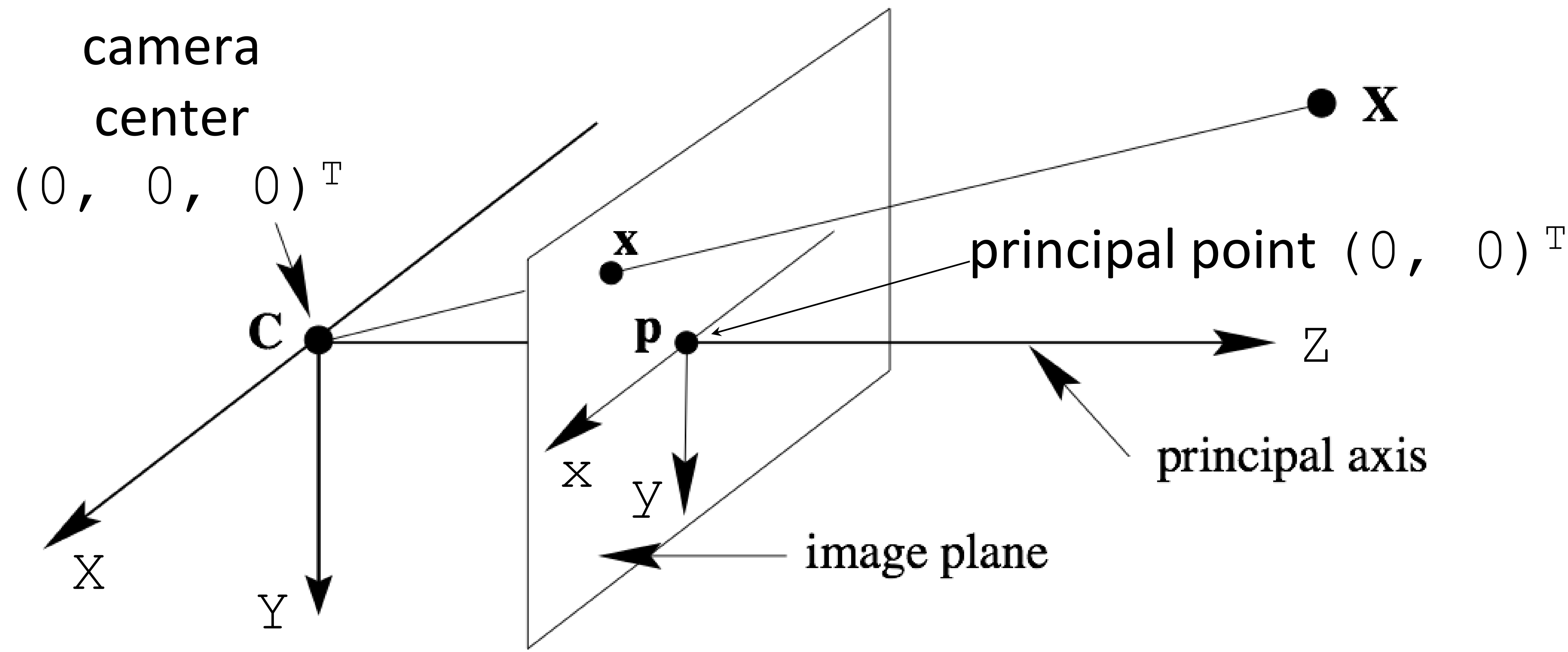


figure adapted from Hartley and Zisserman, 2004

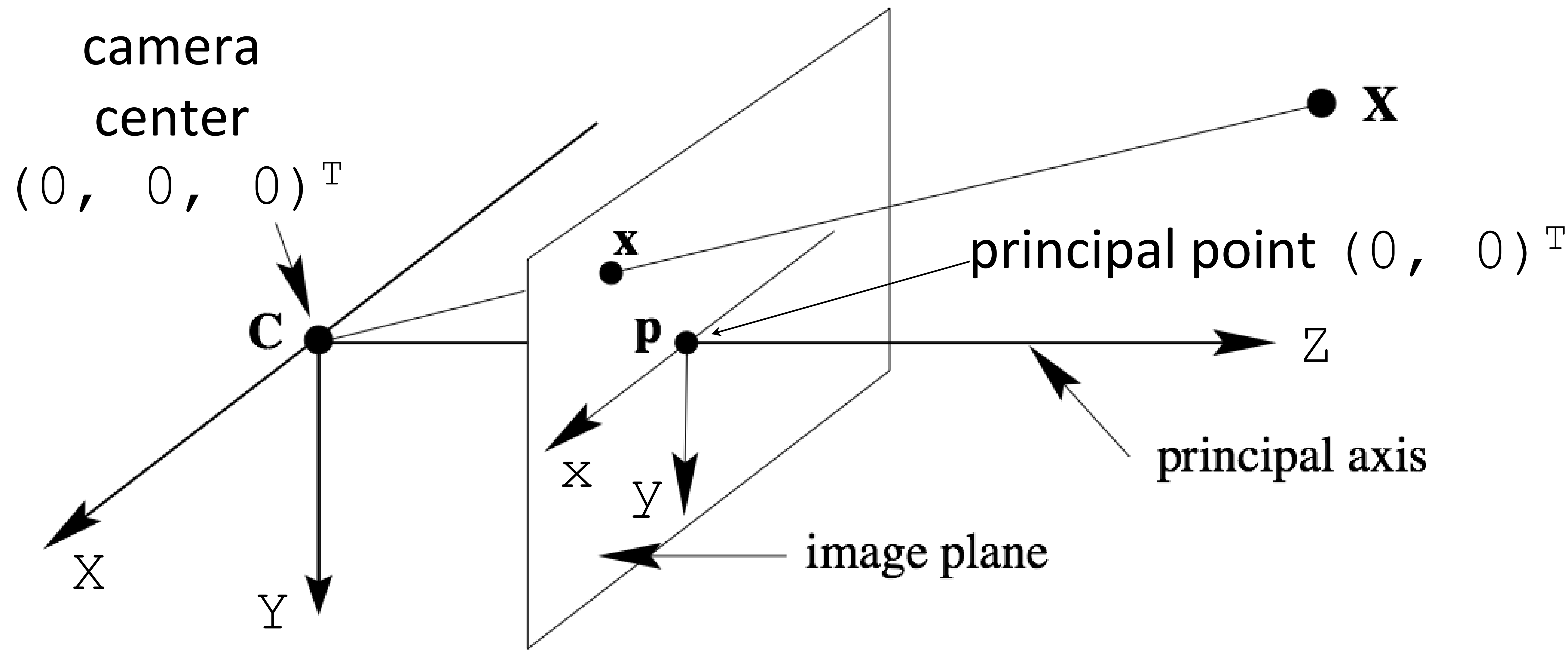
Pinhole Camera Model



Projection in homogenous coordinates: $\lambda\mathbf{x} = \mathbf{X}$

figure adapted from Hartley and Zisserman, 2004

Pinhole Camera Model



$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

Projection in homogenous coordinates: $\lambda\mathbf{x} = \mathbf{X}$

figure adapted from Hartley and Zisserman, 2004

Pinhole Camera Model

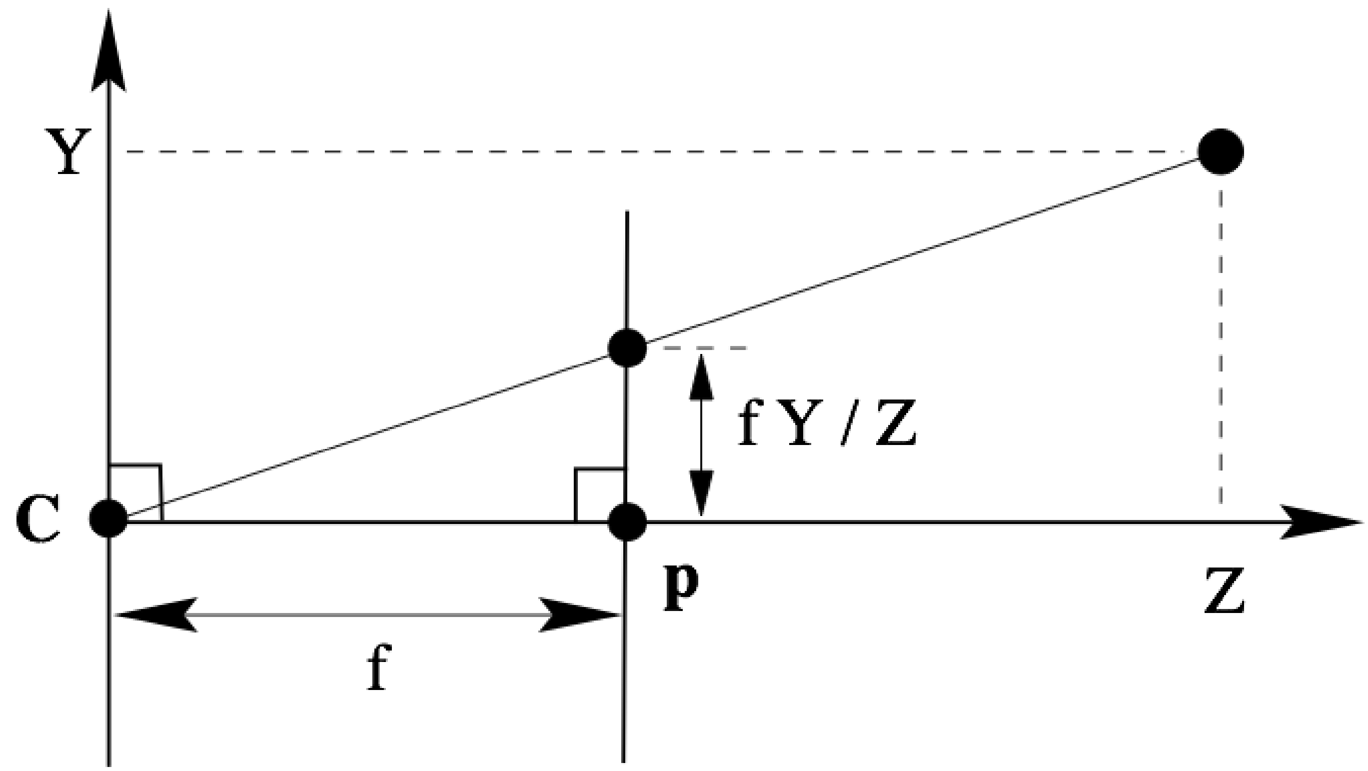


figure from Hartley and Zisserman, 2004

Pinhole Camera Model

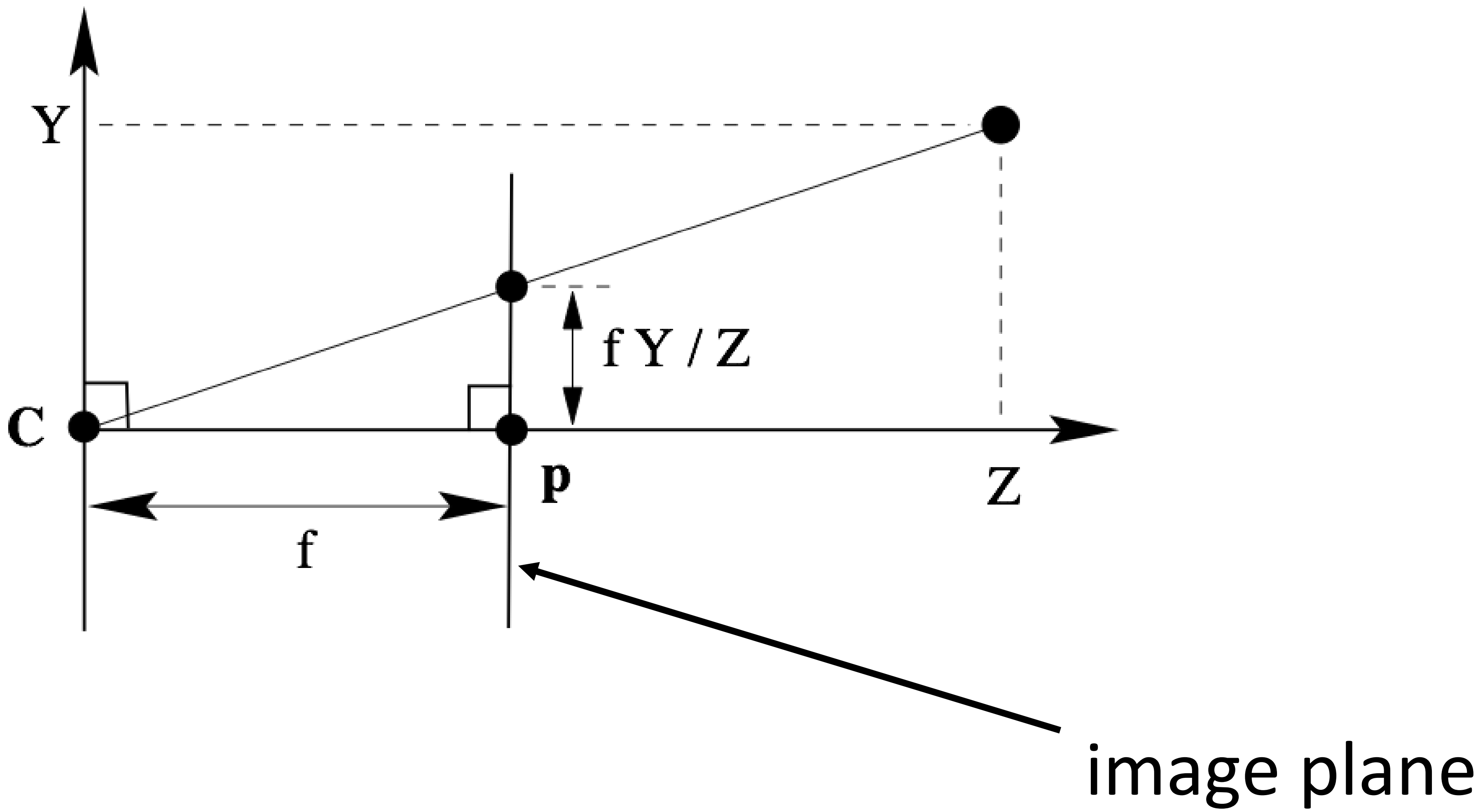


figure from Hartley and Zisserman, 2004

Pinhole Camera Model

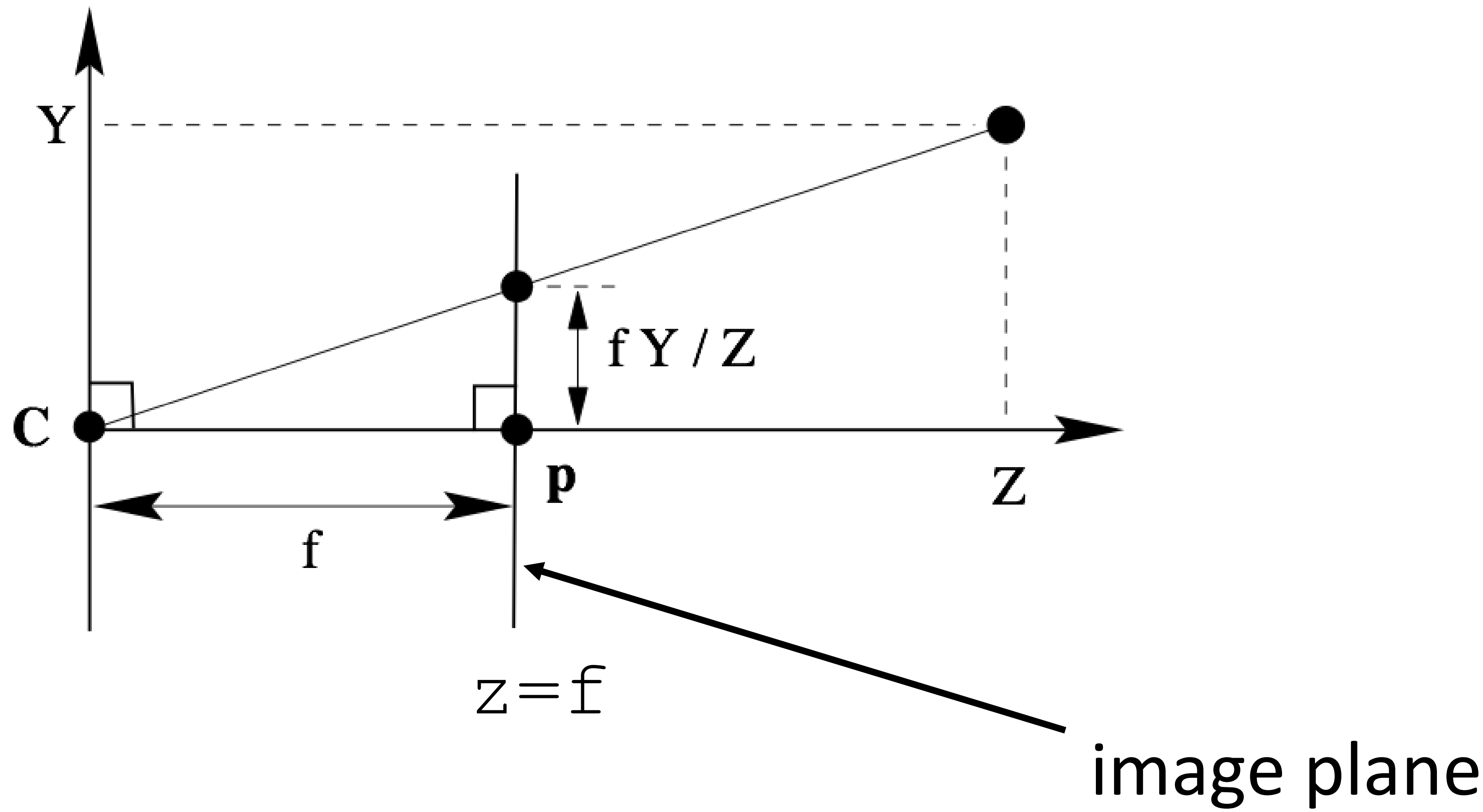


figure from Hartley and Zisserman, 2004

Pinhole Camera Model

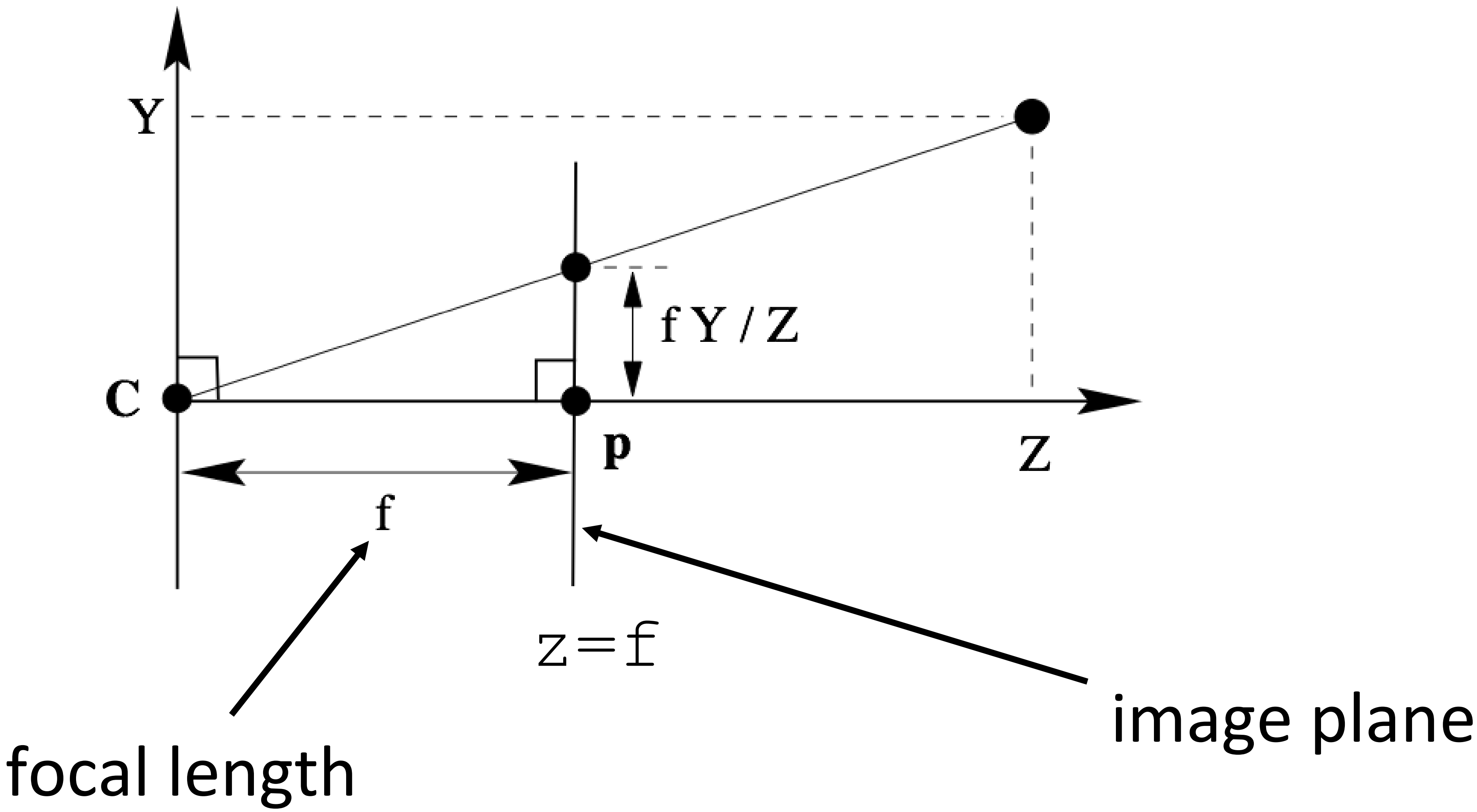
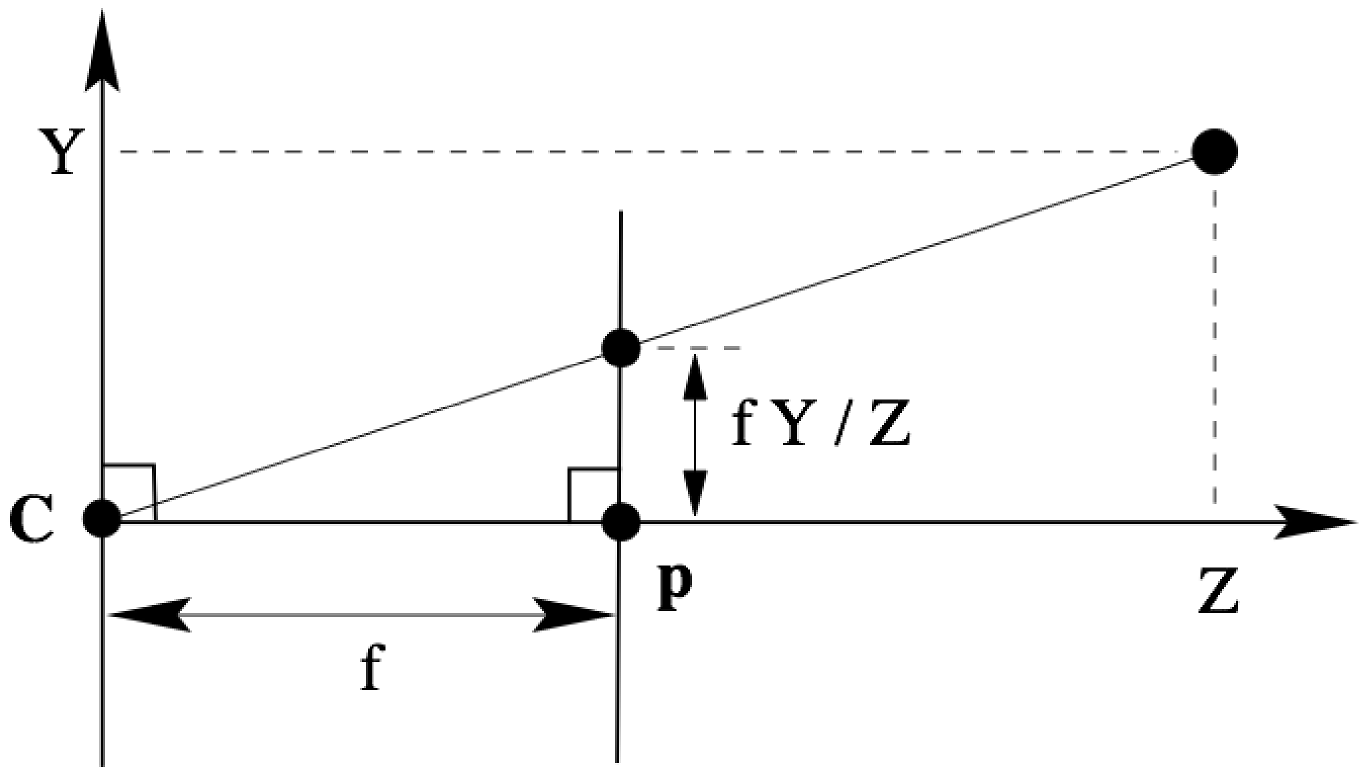


figure from Hartley and Zisserman, 2004

Pinhole Camera Model

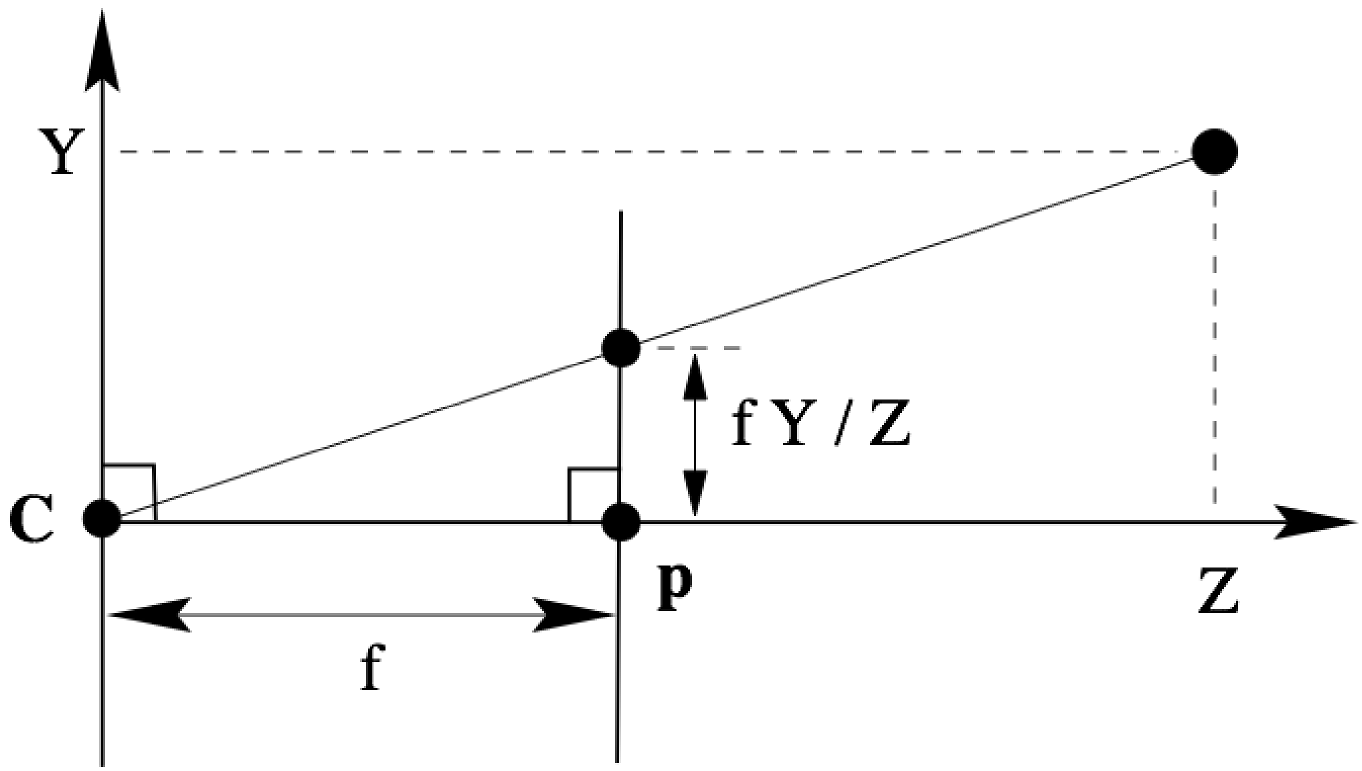


Projection as matrix multiplication:

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

figure from Hartley and Zisserman, 2004

Pinhole Camera Model

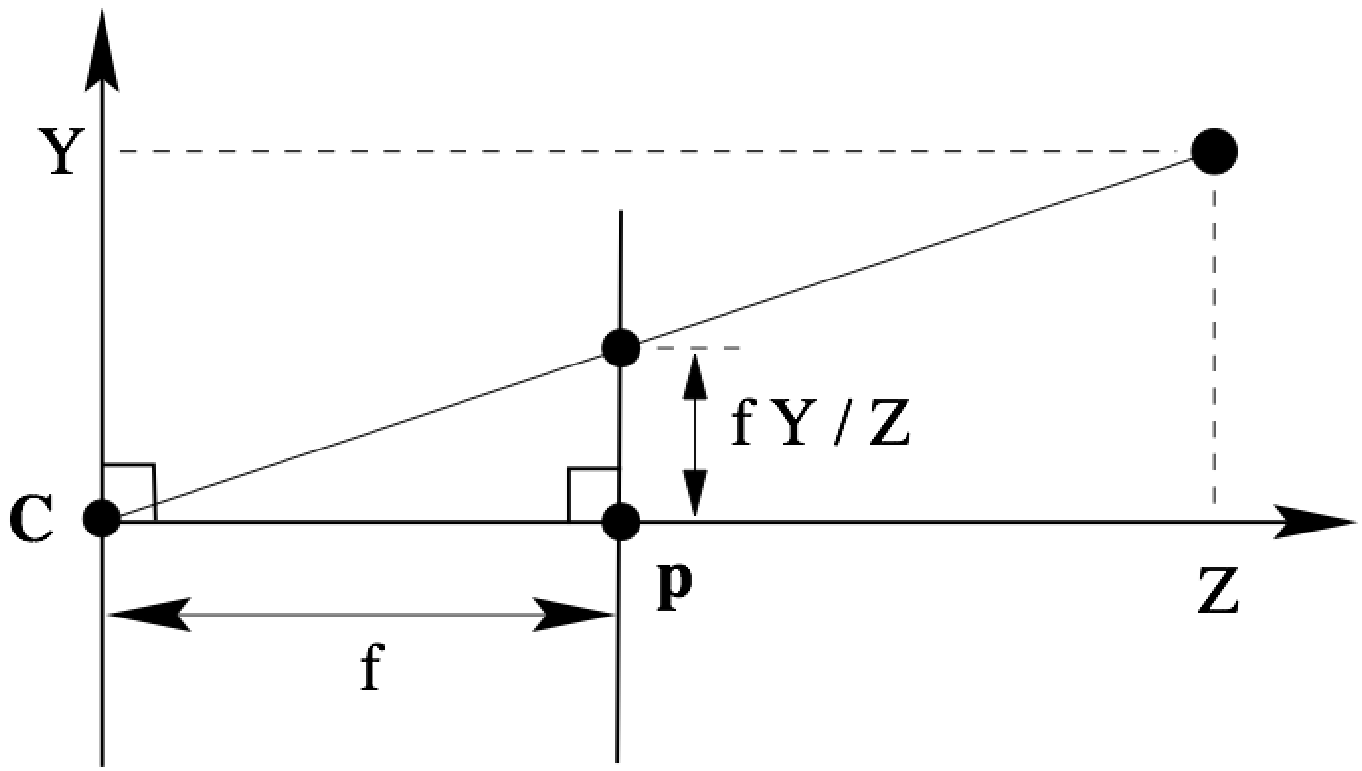


Projection as matrix multiplication:

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

figure from Hartley and Zisserman, 2004

Pinhole Camera Model



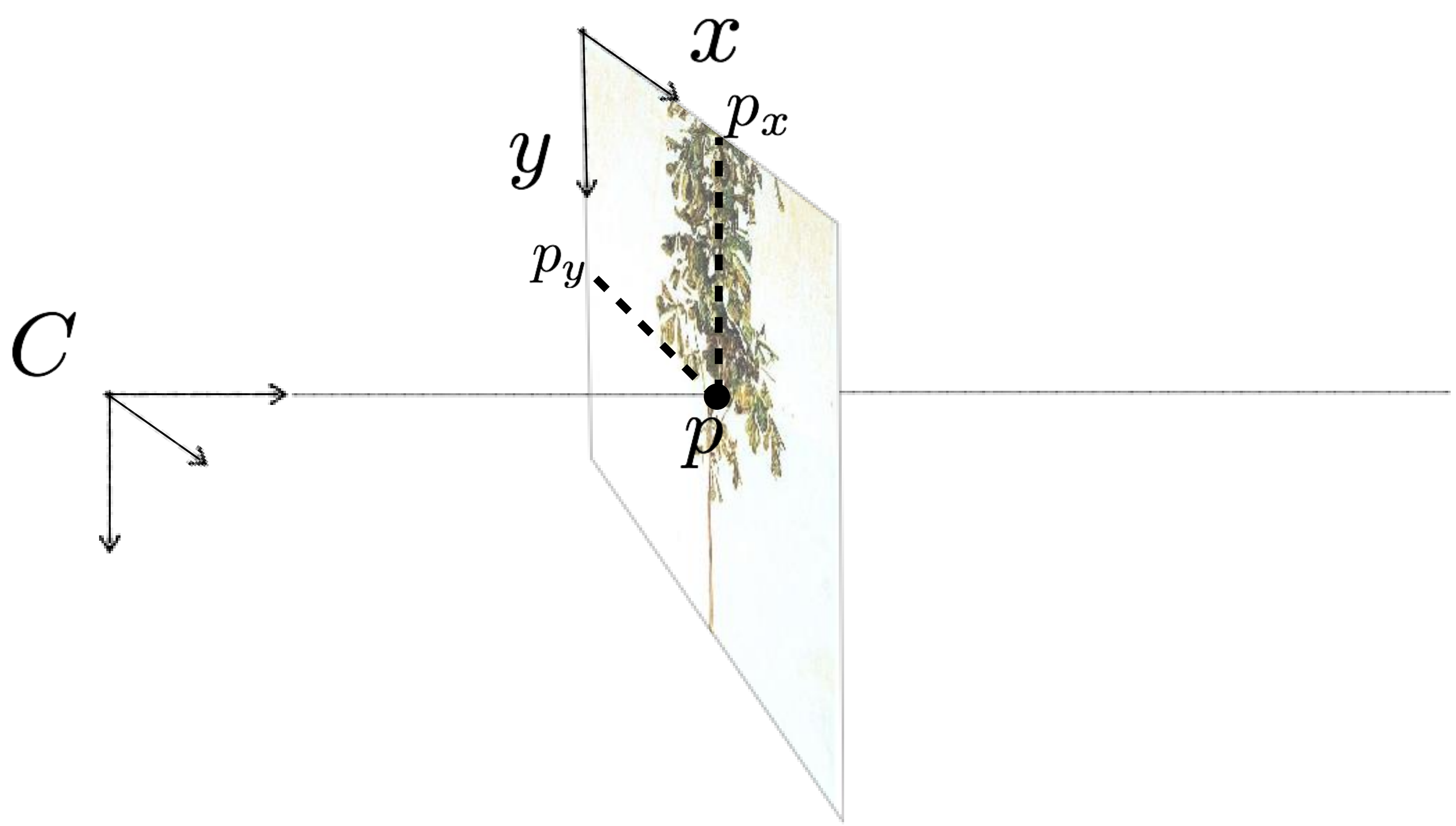
Projection as matrix multiplication:

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

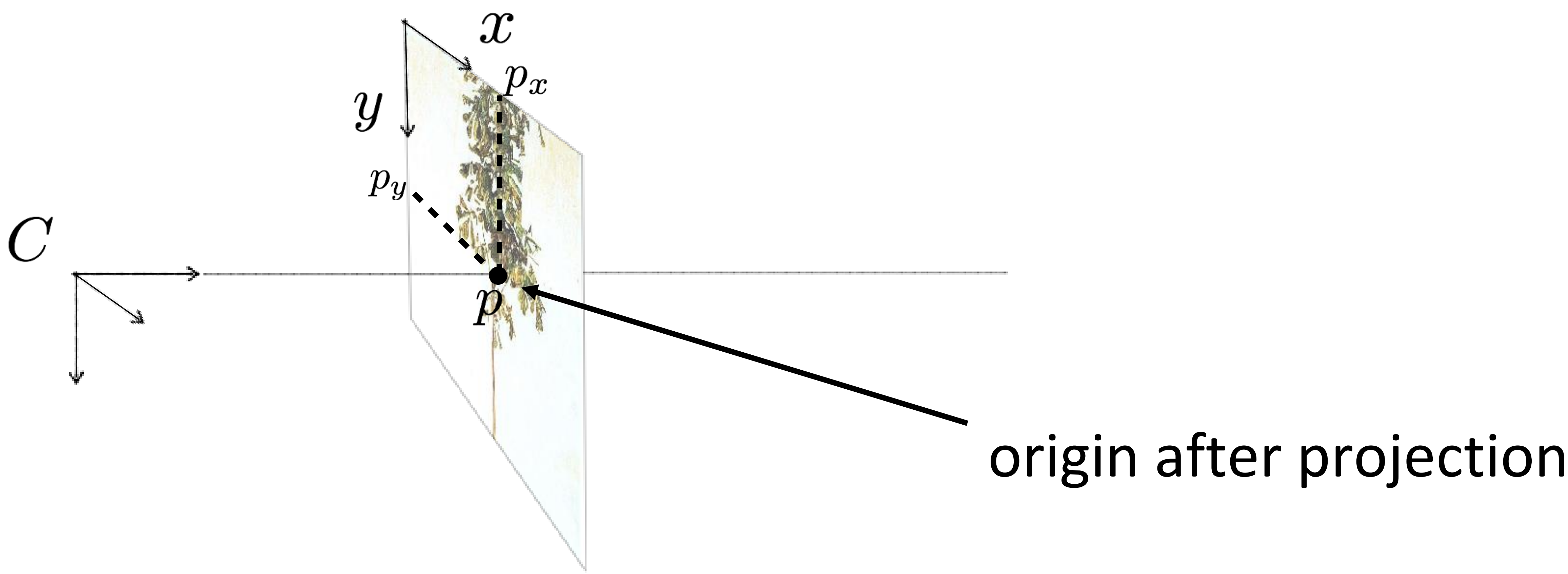
De-homogenization: $$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x'/z' \\ y'/z' \end{pmatrix} = \begin{pmatrix} fX/Z \\ fY/Z \end{pmatrix}$$

figure from Hartley and Zisserman, 2004

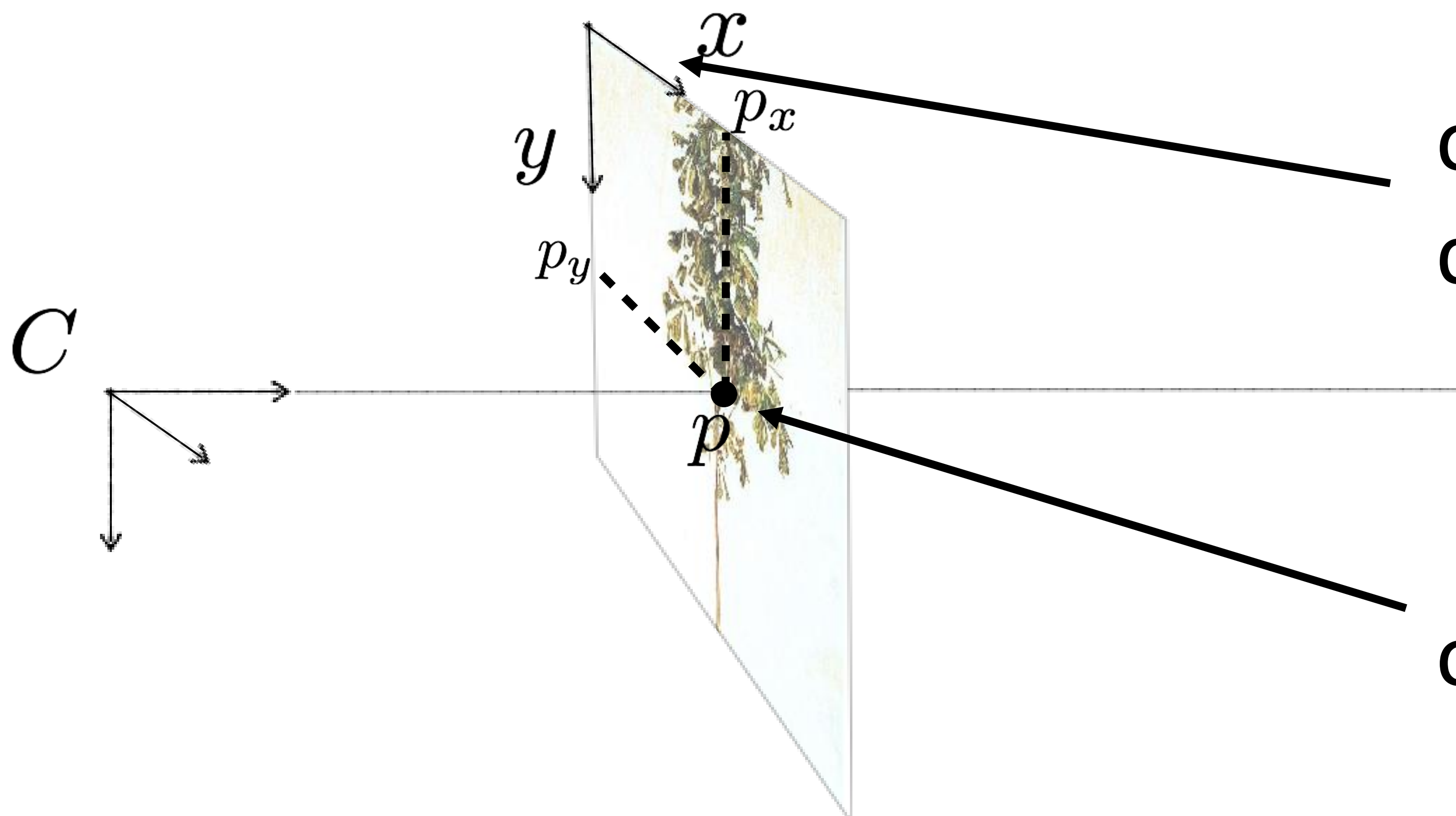
To Pixel Coordinates



To Pixel Coordinates



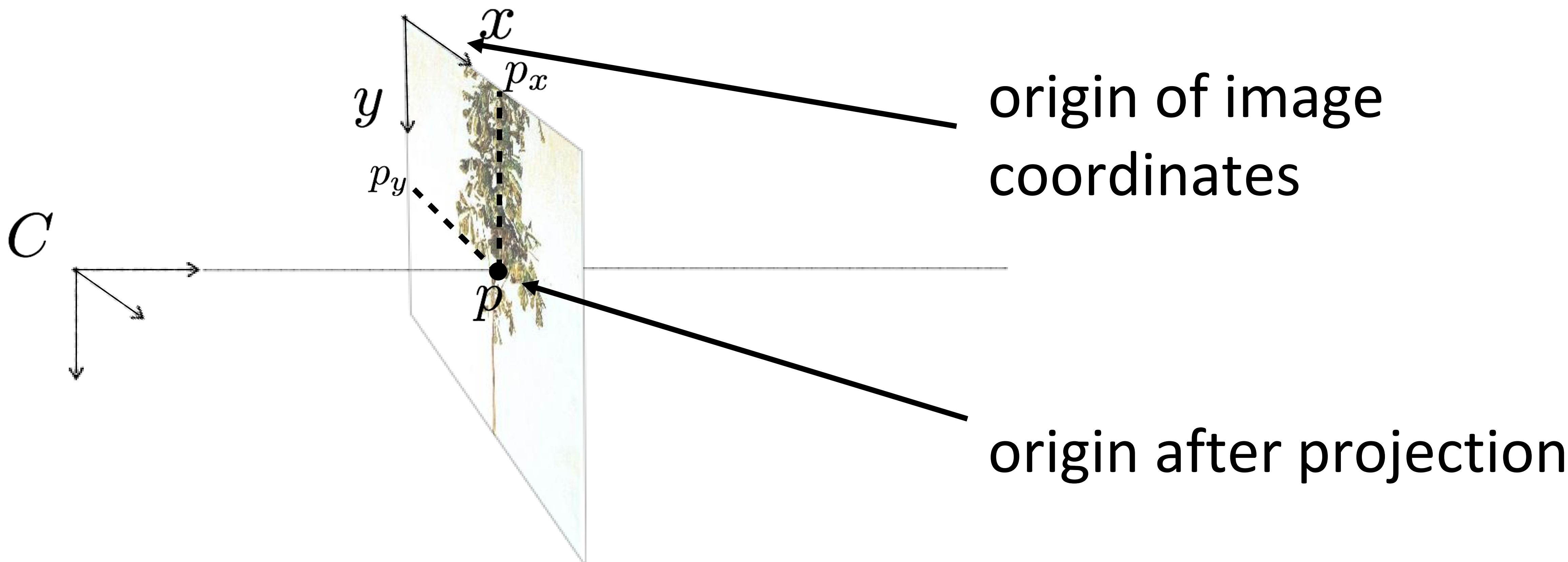
To Pixel Coordinates



origin of image coordinates

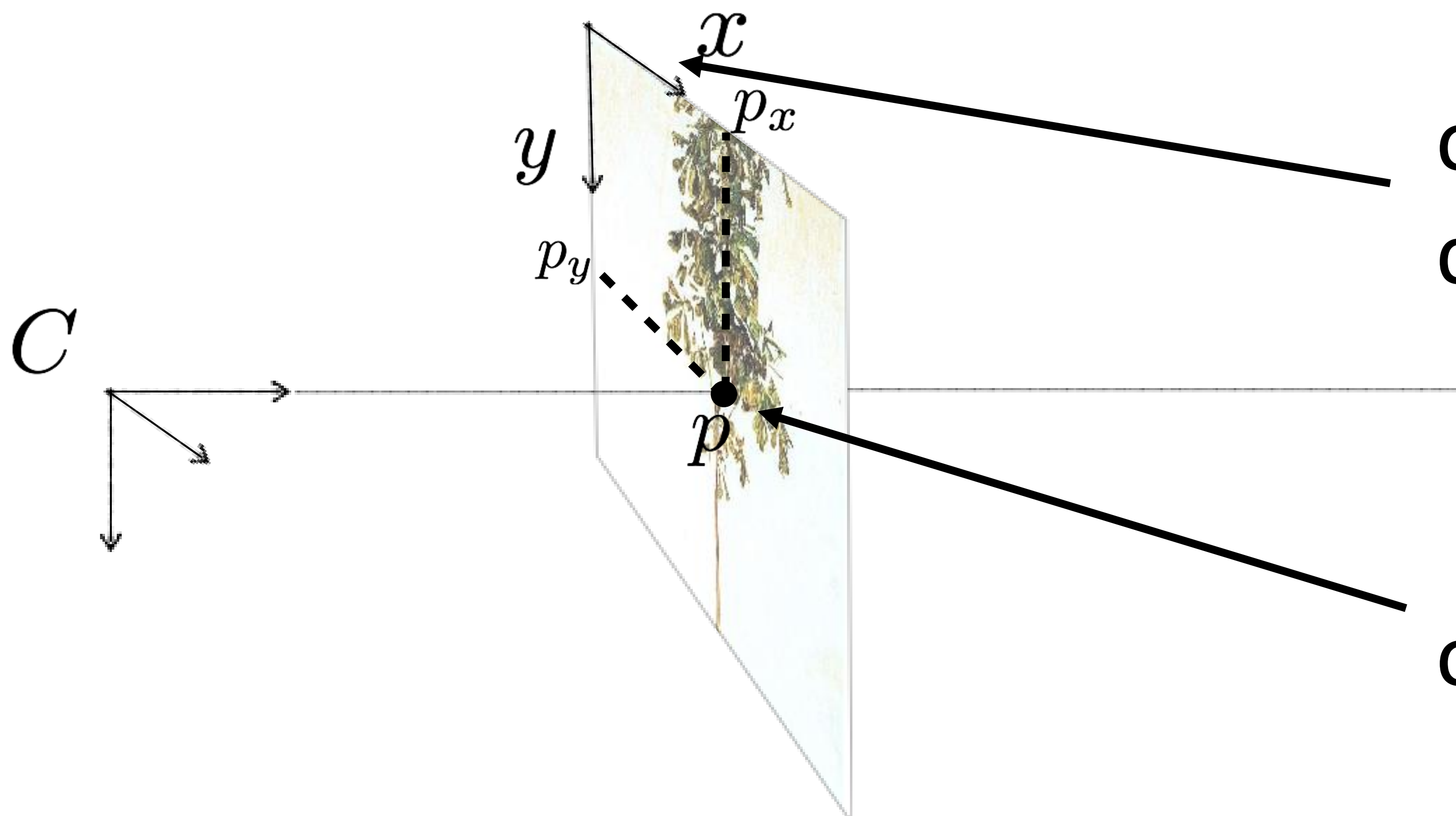
origin after projection

To Pixel Coordinates



Mapping to pixel coordinates: $$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x + p_y \\ y + p_y \end{pmatrix}$$

To Pixel Coordinates



origin of image coordinates

origin after projection

Mapping to pixel coordinates: $\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} x + p_y \\ y + p_y \end{pmatrix}$

Projection as matrix multiplication:

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

Intrinsic Camera Calibration

General intrinsic camera calibration matrix:

$$\mathtt{K} = \begin{pmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix}$$

Intrinsic Camera Calibration

General intrinsic camera calibration matrix:

$$\mathtt{K} = \begin{pmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix}$$

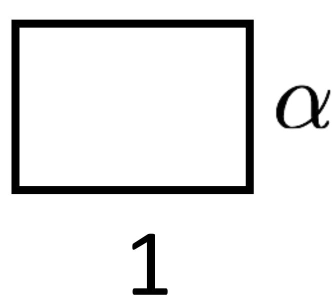
non-rectangular pixels for $s \neq 0$

Intrinsic Camera Calibration

General intrinsic camera calibration matrix:

$$\mathrm{K} = \begin{pmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix}$$

aspect ratio of pixels



non-rectangular pixels for $s \neq 0$

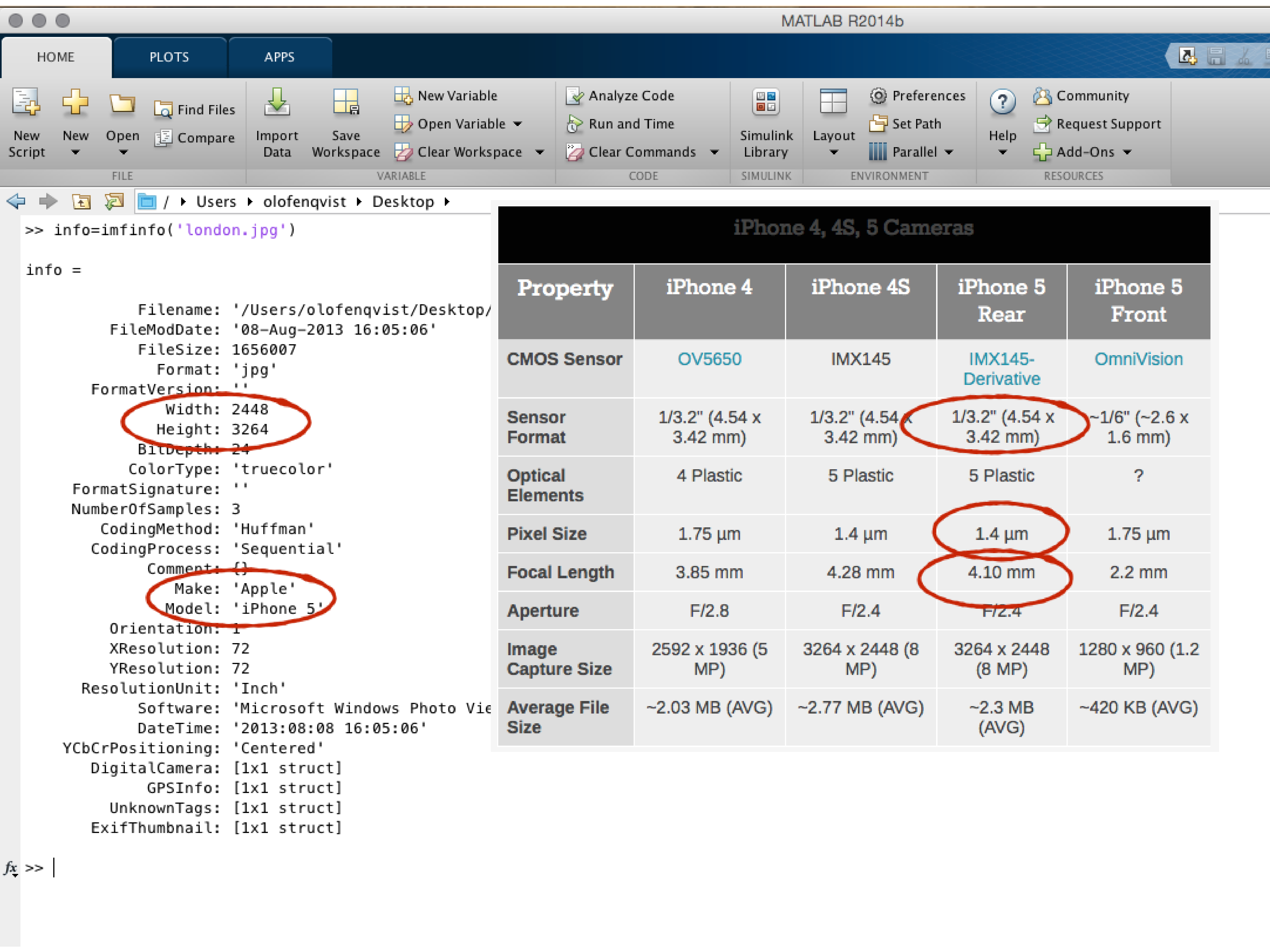
Intrinsic Camera Calibration

General intrinsic camera calibration matrix:

$$\mathtt{K} = \begin{pmatrix} f & s & p_x \\ 0 & \alpha f & p_y \\ 0 & 0 & 1 \end{pmatrix}$$

In practice:

$$\mathtt{K} = \begin{pmatrix} f & 0 & w/2 \\ 0 & f & h/2 \\ 0 & 0 & 1 \end{pmatrix}$$




```
>> info=imfinfo('london.jpg')

info = 

           Filename: '/Users/olofenqvist/Desktop/
        FileModDate: '08-Aug-2013 16:05:06'
           FileSize: 1656007
             Format: 'jpg'
      FormatVersion: ''
              Width: 2448
             Height: 3264
           BitDepth: 24
          ColorType: 'truecolor'
    FormatSignature: ''
    NumberOfSamples: 3
       CodingMethod: 'Huffman'
      CodingProcess: 'Sequential'
            Comment: {}
               Make: 'Apple'
              Model: 'iPhone 5'
        Orientation: 1
        XResolution: 72
        YResolution: 72
     ResolutionUnit: 'Inch'
           Software: 'Microsoft Windows Photo Vie
           DateTime: '2013:08:08 16:05:06'
   YCbCrPositioning: 'Centered'
      DigitalCamera: [1x1 struct]
            GPSInfo: [1x1 struct]
        UnknownTags: [1x1 struct]
      ExifThumbnail: [1x1 struct]

>> 
```

iPhone 4, 4S, 5 Cameras

Property	iPhone 4	iPhone 4S	iPhone 5 Rear	iPhone 5 Front
CMOS Sensor	OV5650	IMX145	IMX145-Derivative	OmniVision
Sensor Format	1/3.2" (4.54 x 3.42 mm)	1/3.2" (4.54 x 3.42 mm)	1/3.2" (4.54 x 3.42 mm)	~1/6" (~2.6 x 1.6 mm)
Optical Elements	4 Plastic	5 Plastic	5 Plastic	?
Pixel Size	1.75 µm	1.4 µm	1.4 µm	1.75 µm
Focal Length	3.85 mm	4.28 mm	4.10 mm	2.2 mm
Aperture	F/2.8	F/2.4	F/2.4	F/2.4
Image Capture Size	2592 x 1936 (5 MP)	3264 x 2448 (8 MP)	3264 x 2448 (8 MP)	1280 x 960 (1.2 MP)
Average File Size	~2.03 MB (AVG)	~2.77 MB (AVG)	~2.3 MB (AVG)	~420 KB (AVG)

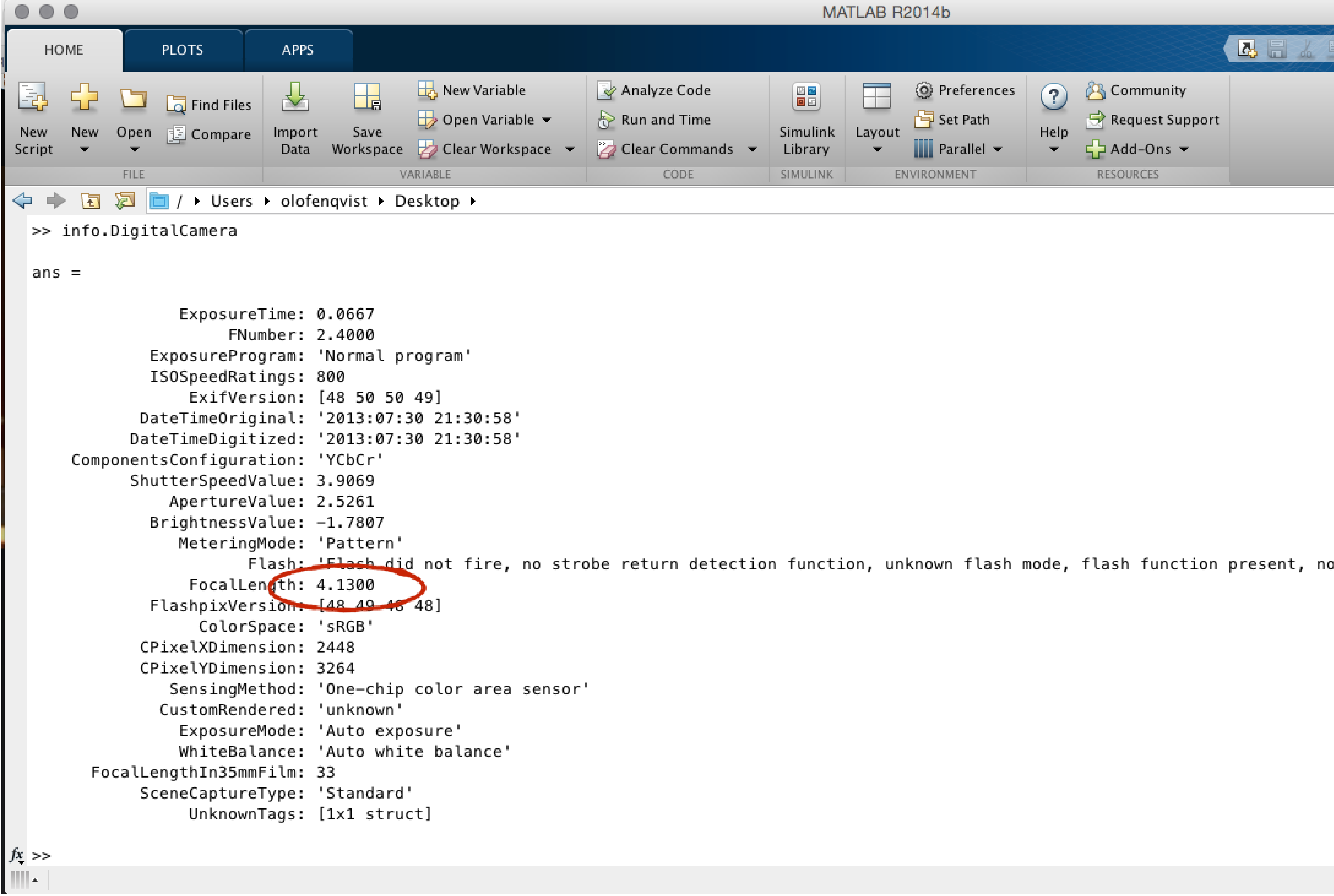
MATLAB R2014b

```
>> info=imfinfo('london.jpg')

info =

         Filename: '/Users/olofenqvist/Desktop/
      FileModDate: '08-Aug-2013 16:05:06'
         FileSize: 1656007
           Format: 'jpg'
    FormatVersion: ''
            Width: 2448
           Height: 3264
         BitDepth: 24
        ColorType: 'truecolor'
  FormatSignature: ''
  NumberOfSamples: 3
     CodingMethod: 'Huffman'
    CodingProcess: 'Sequential'
          Comment: {}
             Make: 'Apple'
            Model: 'iPhone 5'
      Orientation: 1
      XResolution: 72
      YResolution: 72
   ResolutionUnit: 'Inch'
         Software: 'Microsoft Windows Photo Vie
         DateTime: '2013:08:08 16:05:06'
 YCbCrPositi
    DigitalC
          GP
     Unknow
    ExifThum

>> |
```

iPhone 4, 4S, 5 Cameras

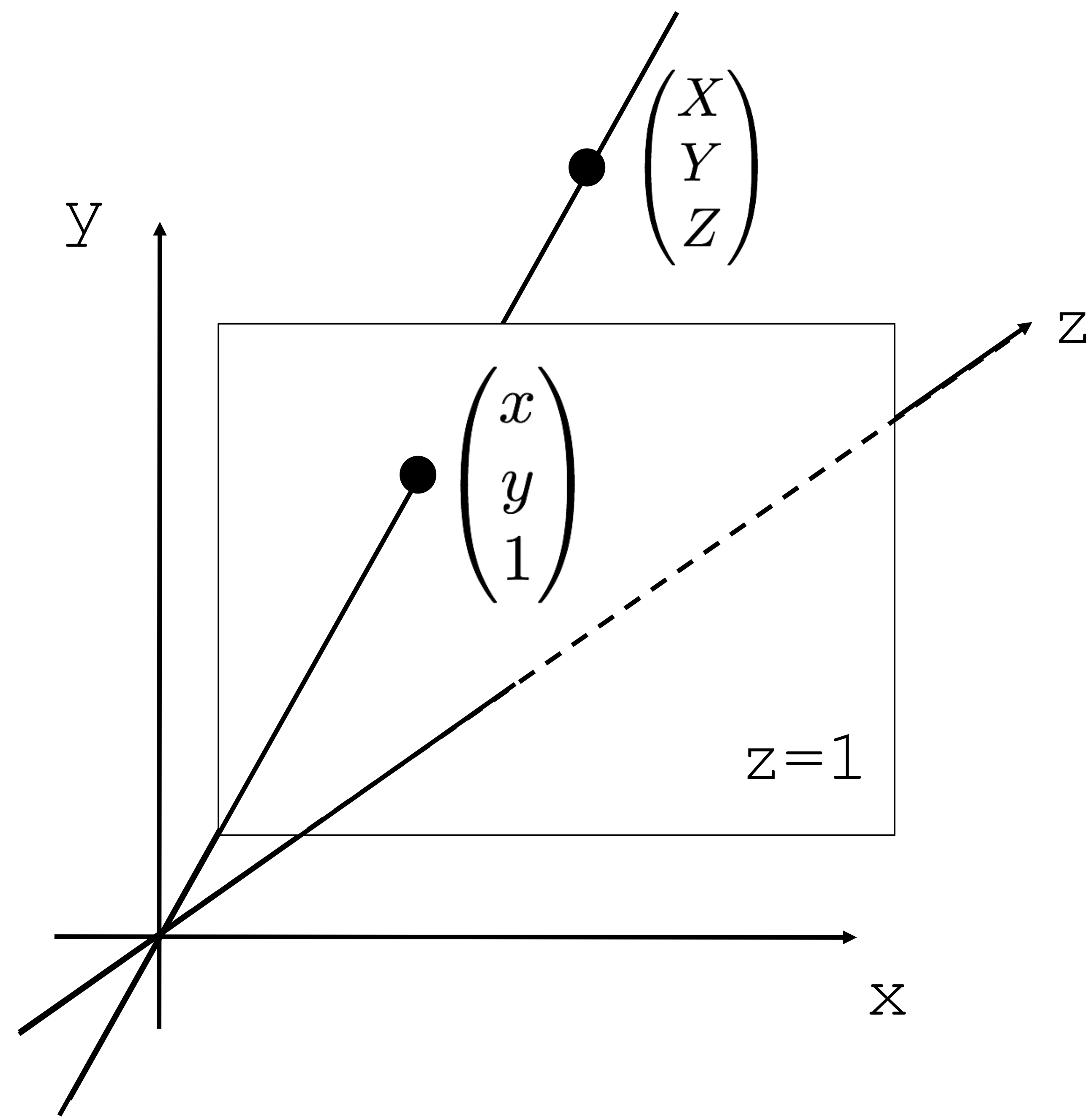
Property	iPhone 4	iPhone 4S	iPhone 5 Rear	iPhone 5 Front
CMOS Sensor	OV5650	IMX145	IMX145-Derivative	OmniVision
Sensor Format	1/3.2" (4.54 x 3.42 mm)	1/3.2" (4.54 x 3.42 mm)	1/3.2" (4.54 x 3.42 mm)	~1/6" (~2.6 x 1.6 mm)
Optical Elements	4 Plastic	5 Plastic	5 Plastic	?
Pixel Size	1.75 µm	1.4 µm	1.4 µm	1.75 µm
Focal Length	3.85 mm	4.28 mm	4.10 mm	2.2 mm
Aperture	F/2.8	F/2.4	F/2.4	F/2.4
Image Capture Size	2592 x 1936 (5 MP)	3264 x 2448 (8 MP)	3264 x 2448 (8 MP)	1280 x 960 (1.2 MP)
Average File Size	~2.03 MB (AVG)	~2.77 MB (AVG)	~2.3 MB (AVG)	~420 KB (AVG)

focal length in pixels = (image width in pixels) * (focal length in mm) / (CCD width in mm)



For details, see: http://www.cs.cornell.edu/~snavely/bundler/focal.html

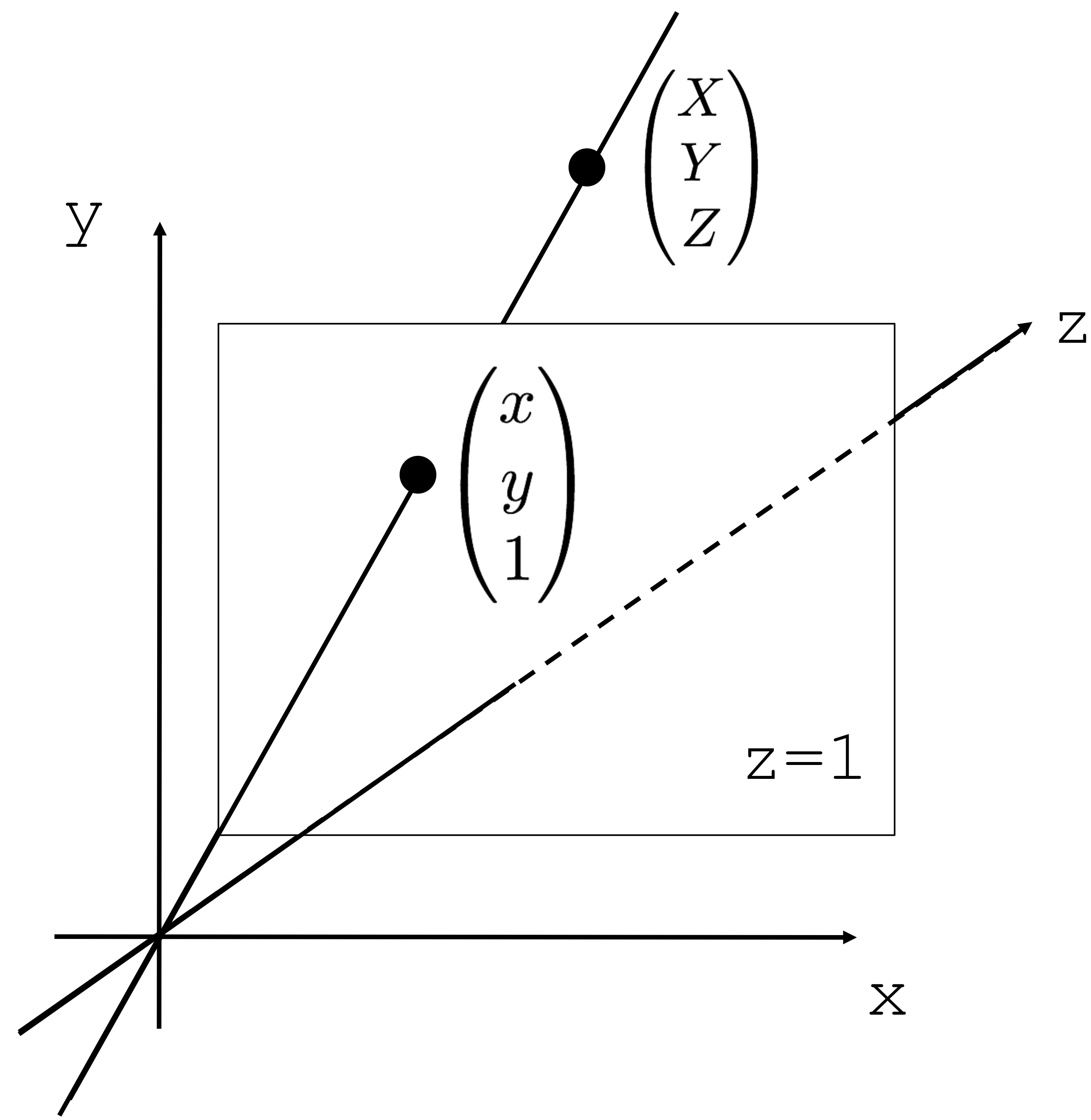
Forward and Backward Projections



Forward and Backward Projections

A 3D point maps to a 2D point:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathtt{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$



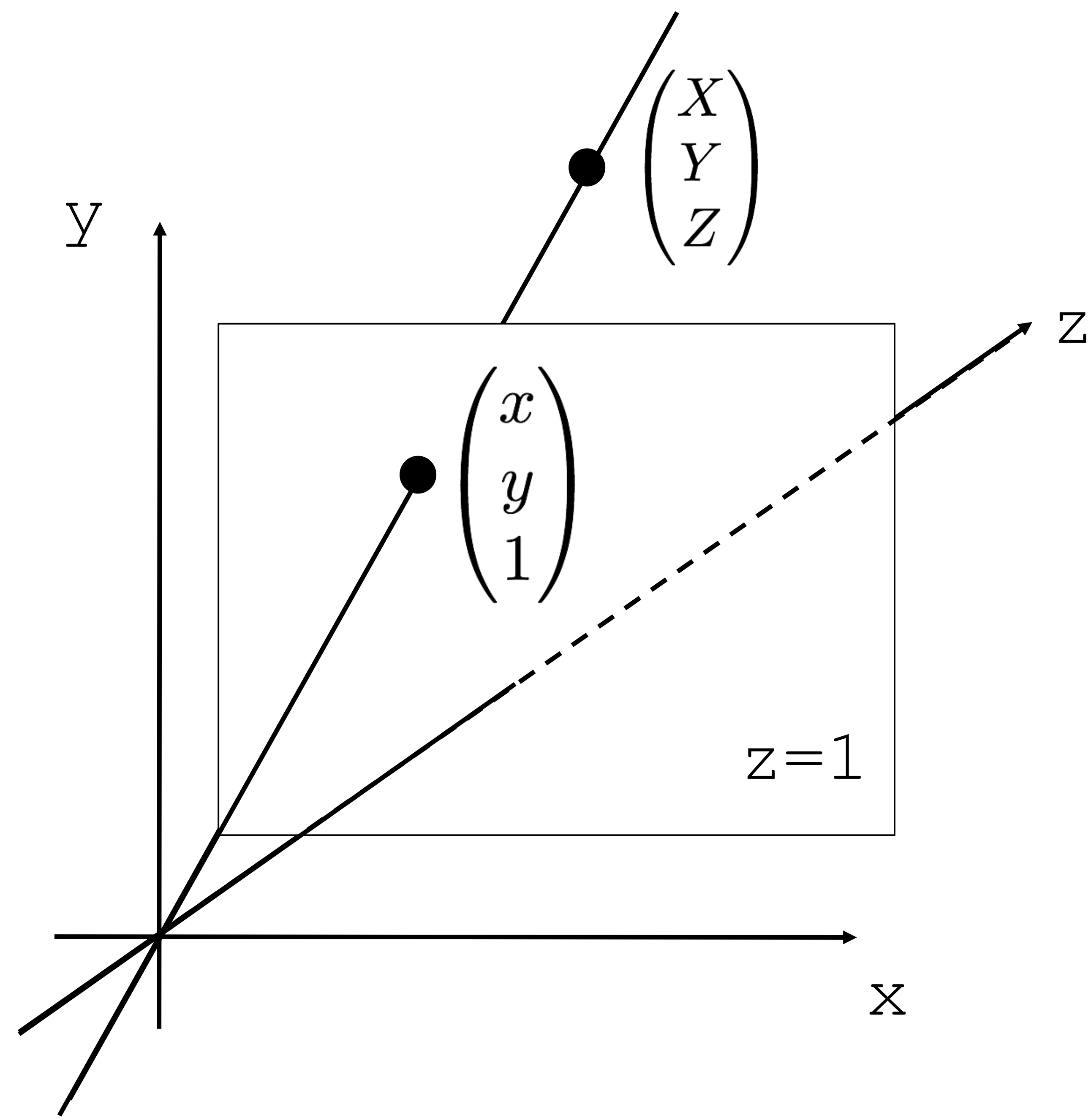
Forward and Backward Projections

A 3D point maps to a 2D point:

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \mathtt{K} \begin{pmatrix} X \\ Y \\ Z \end{pmatrix}$$

A 2D point maps to a ray:

$$\delta \begin{pmatrix} X \\ Y \\ Z \end{pmatrix} = \mathtt{K}^{-1} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$



Forward and Backward Projections

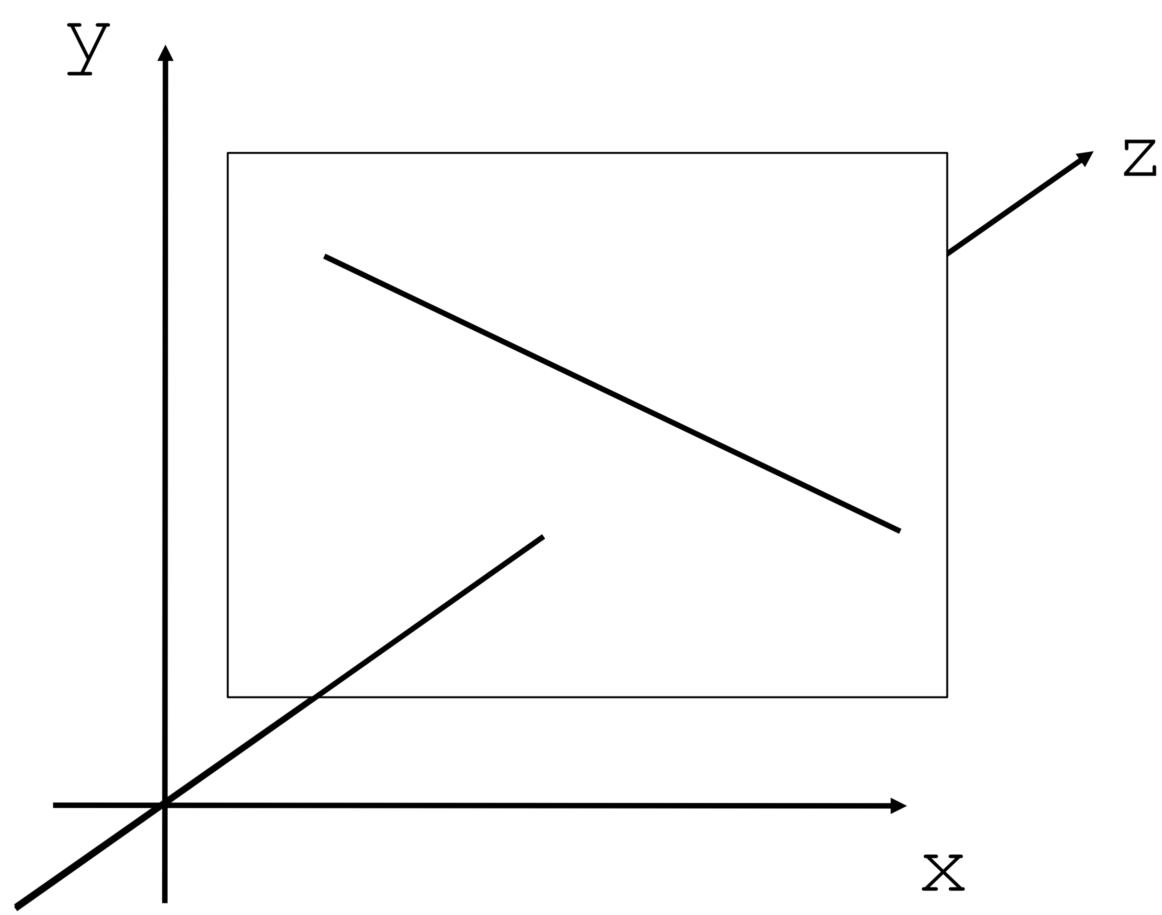
A 3D line maps to a 2D line:

$$\mathtt{K}\left(\mathbf{A} + \lambda\mathbf{B}\right) = \mathtt{K}\mathbf{A} + \lambda\mathtt{K}\mathbf{B}$$

Forward and Backward Projections

A 3D line maps to a 2D line:

$$\mathtt{K}\left(\mathbf{A} + \lambda \mathbf{B}\right) = \mathtt{K}\mathbf{A} + \lambda \mathtt{K}\mathbf{B}$$

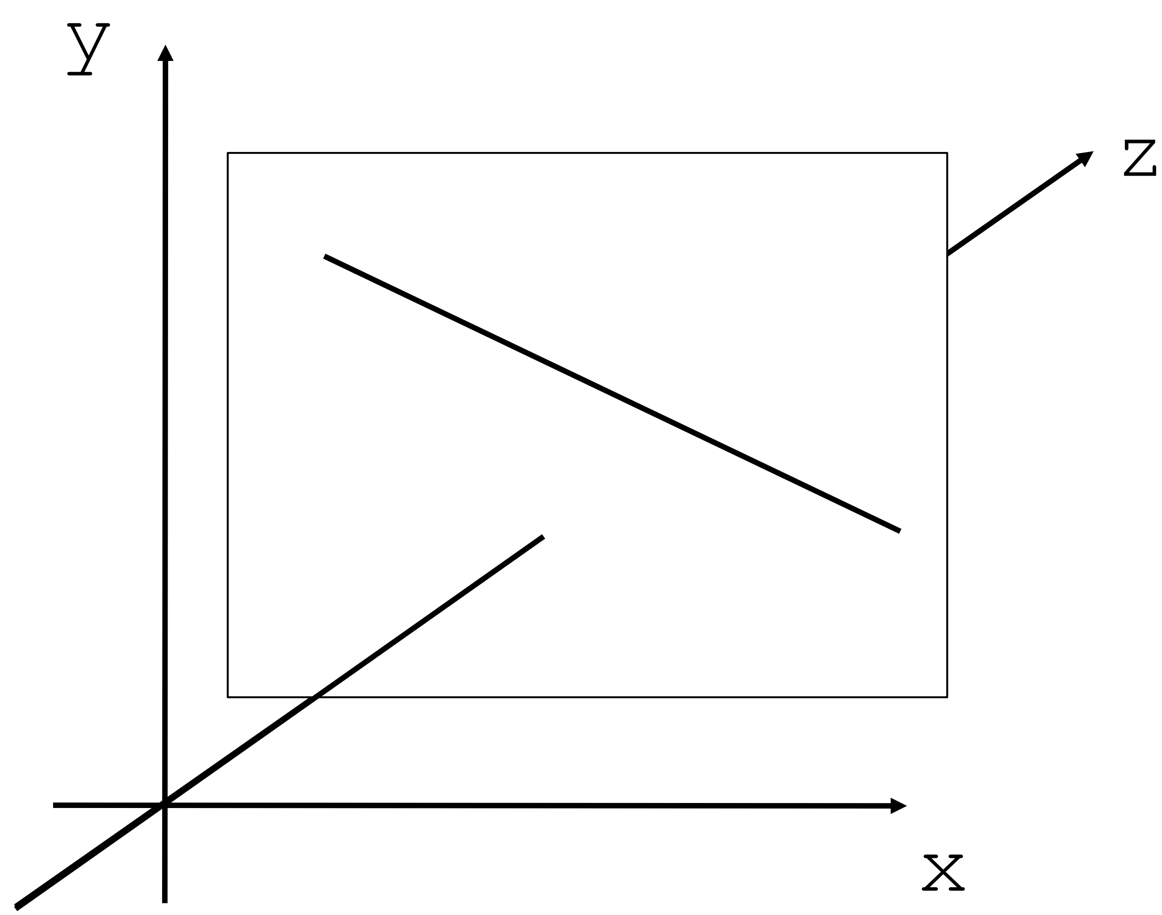


Forward and Backward Projections

A 3D line maps to a 2D line:

$$\mathtt{K}\left(\mathbf{A} + \lambda \mathbf{B}\right) = \mathtt{K}\mathbf{A} + \lambda \mathtt{K}\mathbf{B}$$

A 2D line maps to
a 3D plane:

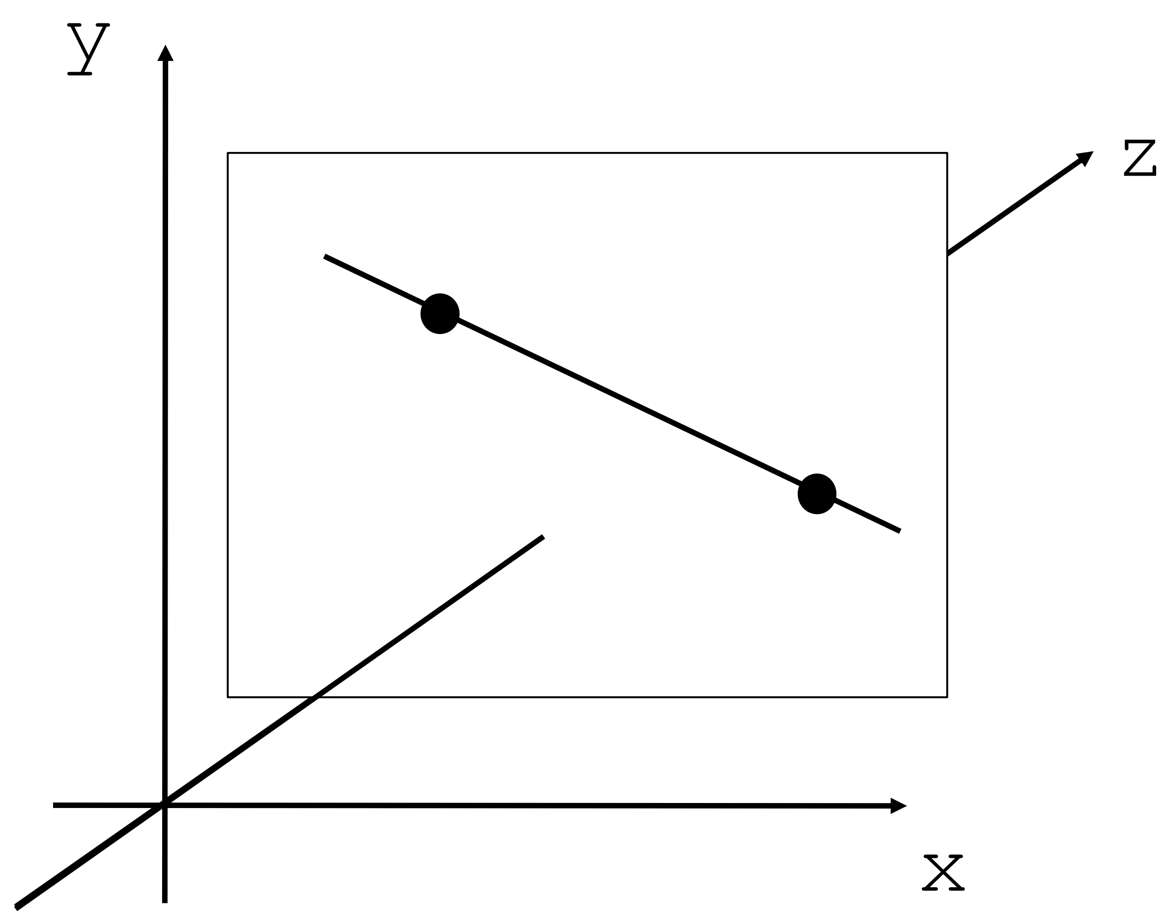


Forward and Backward Projections

A 3D line maps to a 2D line:

$$\mathtt{K}\left(\mathbf{A} + \lambda\mathbf{B}\right) = \mathtt{K}\mathbf{A} + \lambda\mathtt{K}\mathbf{B}$$

A 2D line maps to a 3D plane:

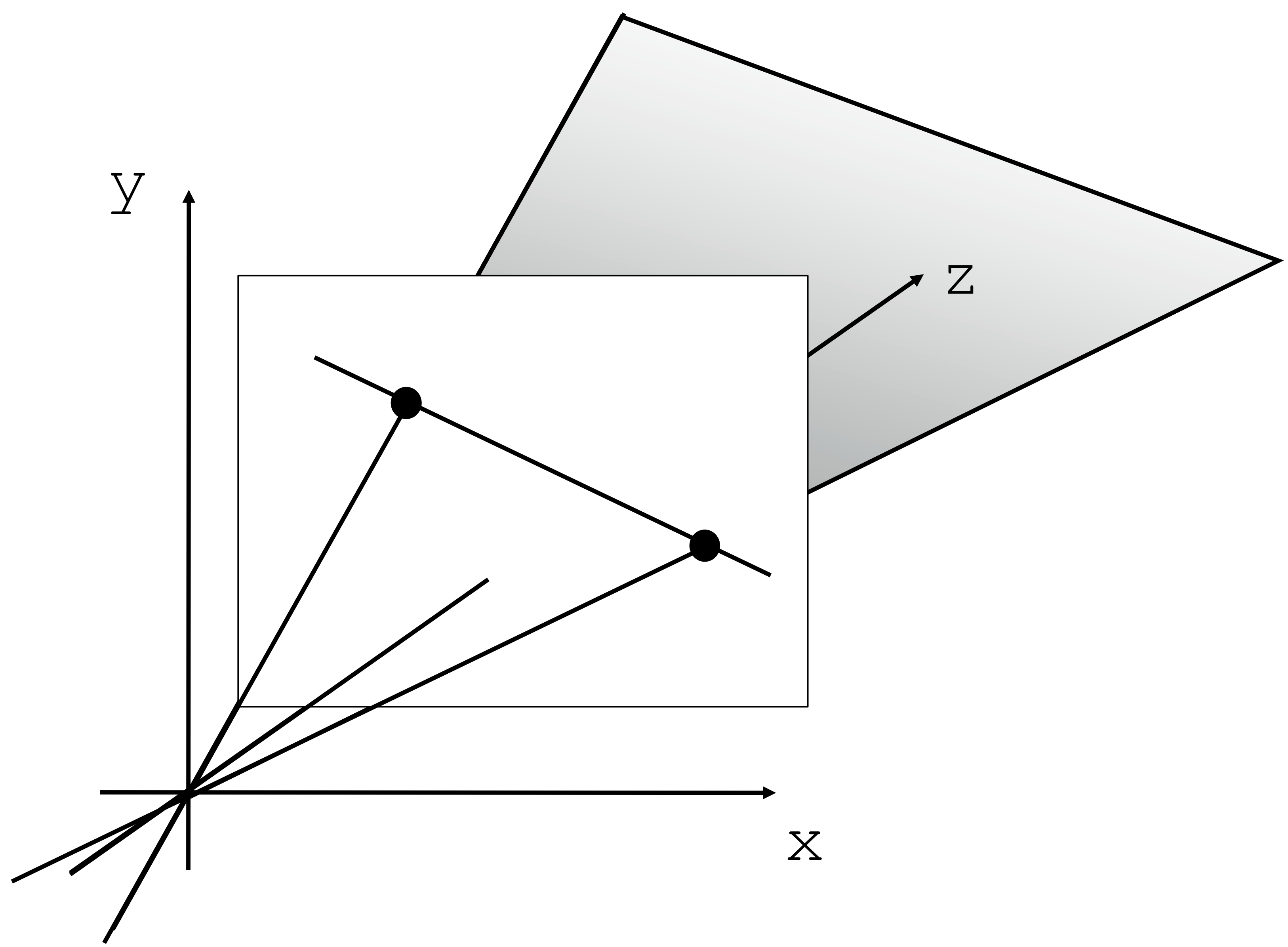


Forward and Backward Projections

A 3D line maps to a 2D line:

$$\mathtt{K}\left(\mathbf{A}+\lambda\mathbf{B}\right)=\mathtt{K}\mathbf{A}+\lambda\mathtt{K}\mathbf{B}$$

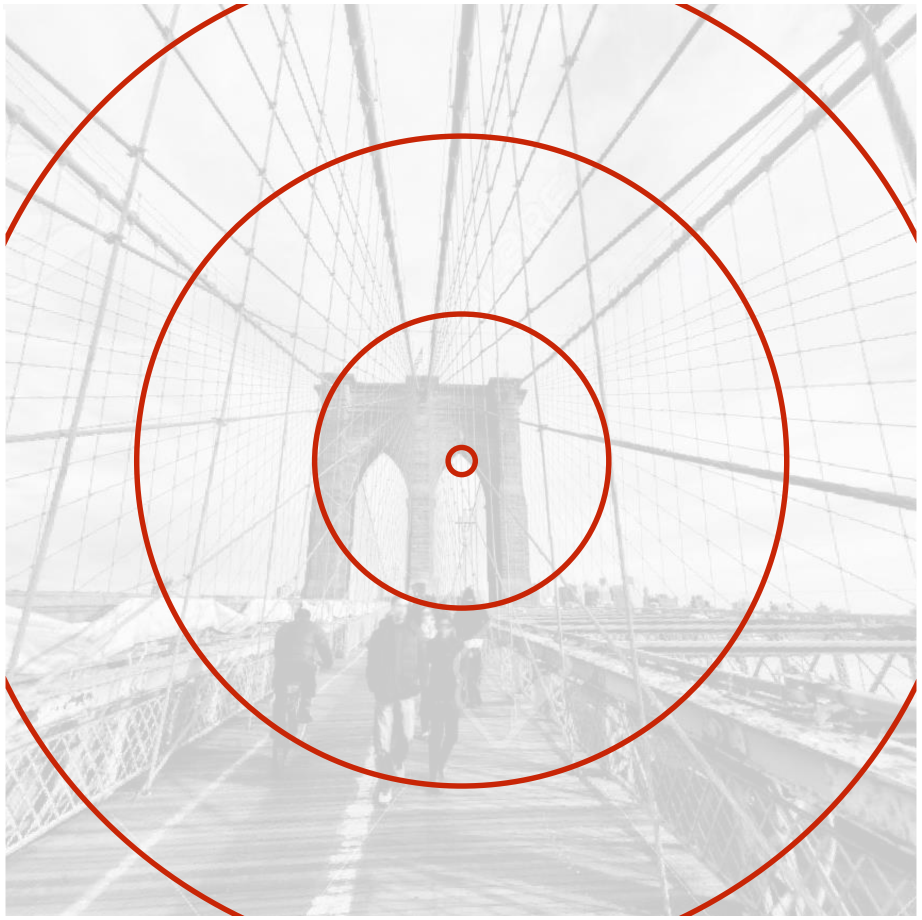
A 2D line maps to a 3D plane:



Lens Distortion

Accurate Physical Model?

Radial Distortion



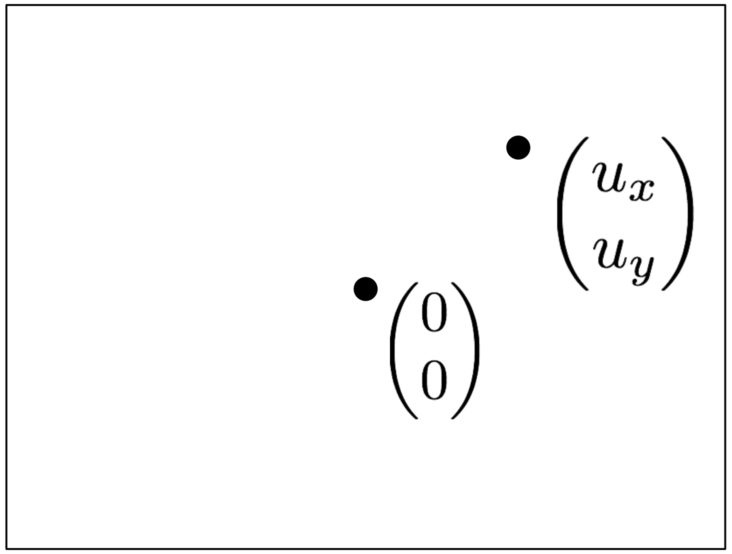
Pinhole camera

Fisheye lens camera

Polynomial Radial Distortion Model

Project 3D point into camera coordinates

$$\mathbf{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix}$$



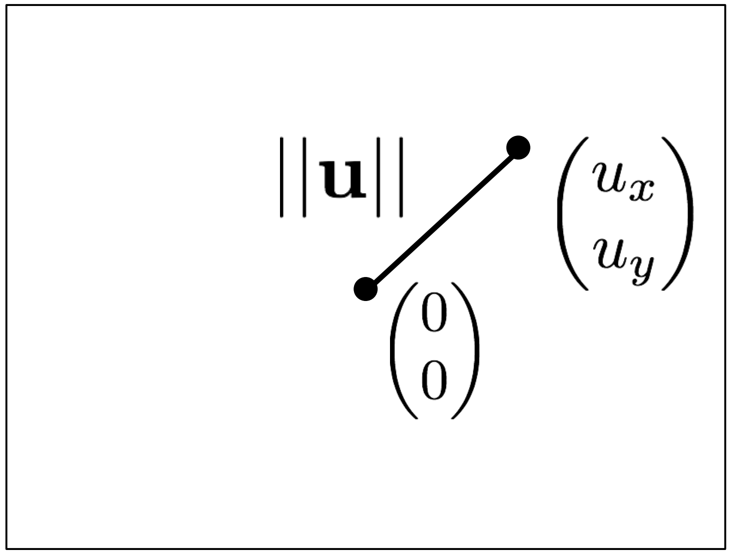
Polynomial Radial Distortion Model

Project 3D point into camera coordinates

$$\mathbf{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix}$$

Compute radial distortion factor

$$r(\mathbf{u}) = 1 + \kappa_1 ||\mathbf{u}||^2 + \kappa_2 ||\mathbf{u}||^4$$



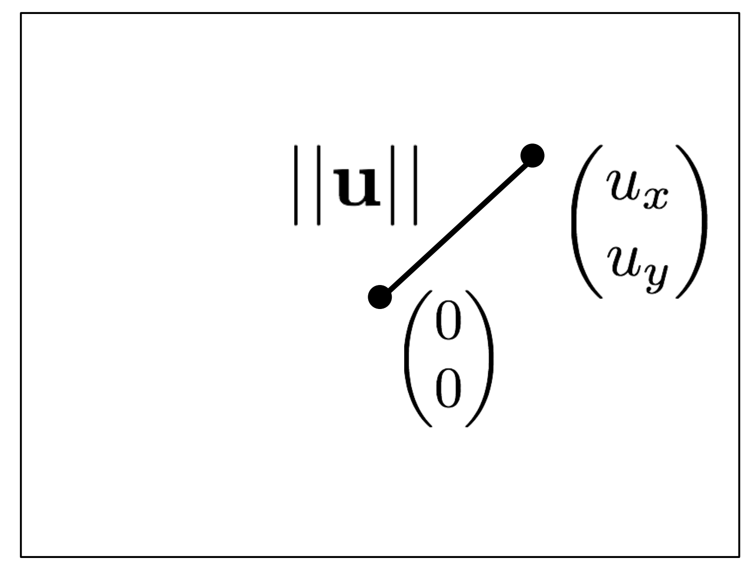
Polynomial Radial Distortion Model

Project 3D point into camera coordinates

$$\mathbf{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix}$$

Compute radial distortion factor

$$r(\mathbf{u}) = 1 + \kappa_1 ||\mathbf{u}||^2 + \kappa_2 ||\mathbf{u}||^4$$



Compute pixel position

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \cdot r(\mathbf{u}) \cdot u_x + p_x \\ f \cdot r(\mathbf{u}) \cdot u_y + p_y \end{pmatrix}$$

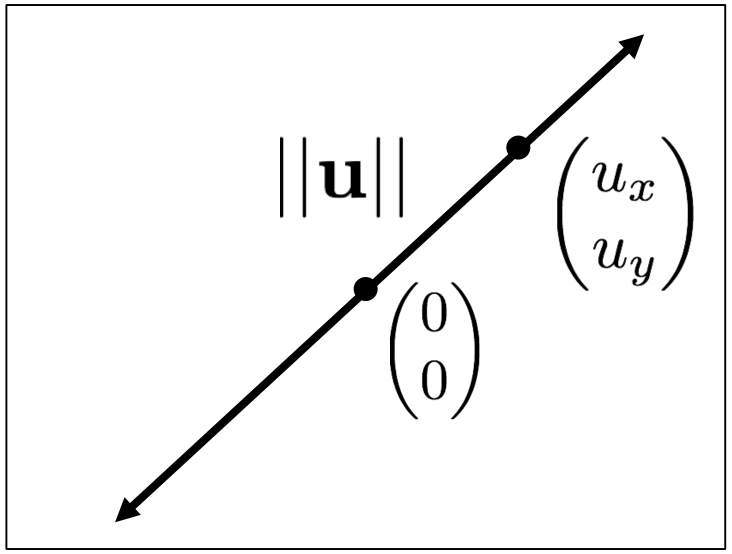
Polynomial Radial Distortion Model

Project 3D point into camera coordinates

$$\mathbf{u} = \begin{pmatrix} u_x \\ u_y \end{pmatrix} = \begin{pmatrix} X/Z \\ Y/Z \end{pmatrix}$$

Compute radial distortion factor

$$r(\mathbf{u}) = 1 + \kappa_1 ||\mathbf{u}||^2 + \kappa_2 ||\mathbf{u}||^4$$



Compute pixel position

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} f \cdot r(\mathbf{u}) \cdot u_x + p_x \\ f \cdot r(\mathbf{u}) \cdot u_y + p_y \end{pmatrix}$$

Removing Lens Distortion

Camera Rotation and Translation

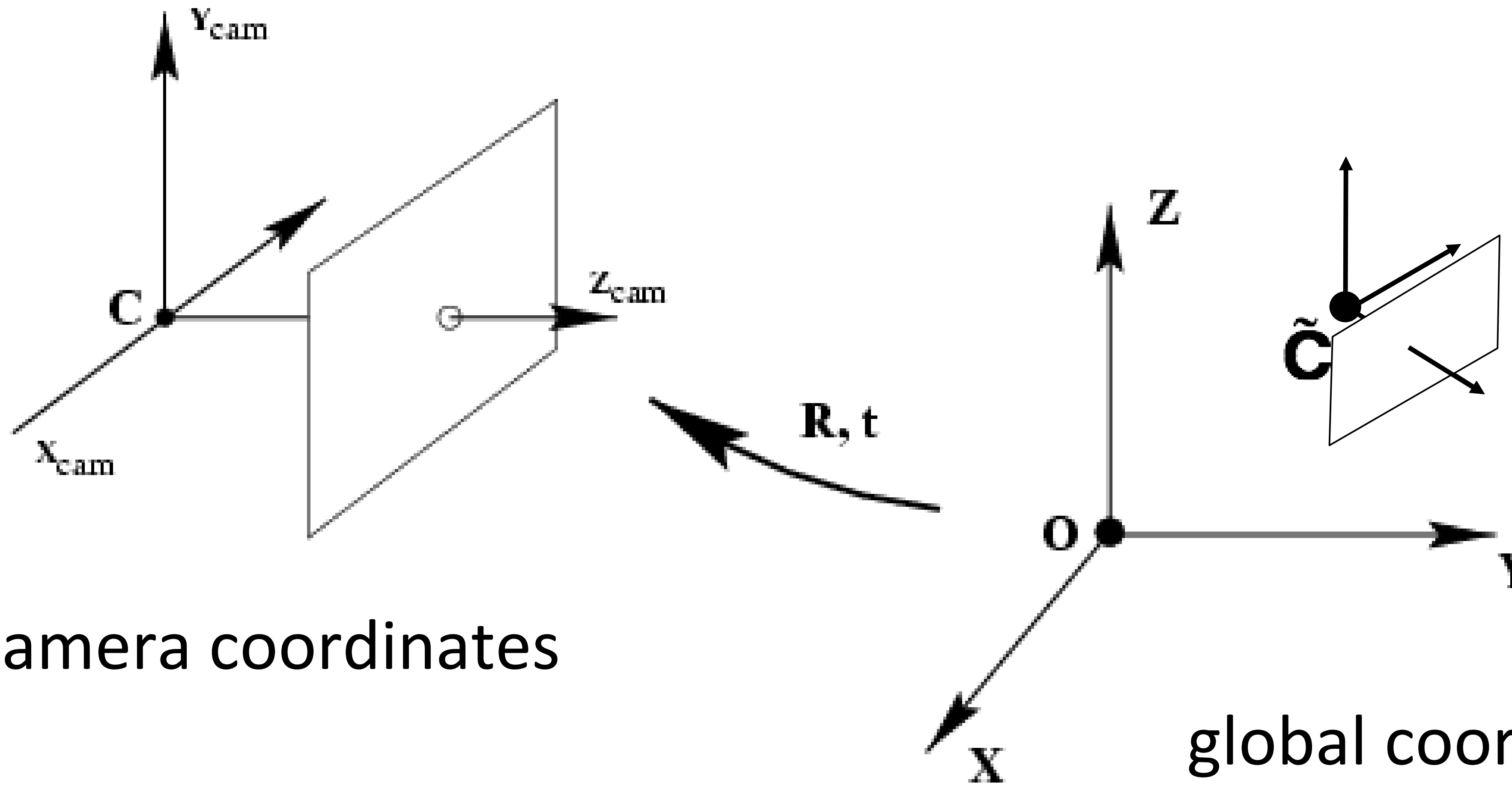


figure adapted from Hartley and Zisserman, 2004

Camera Rotation and Translation

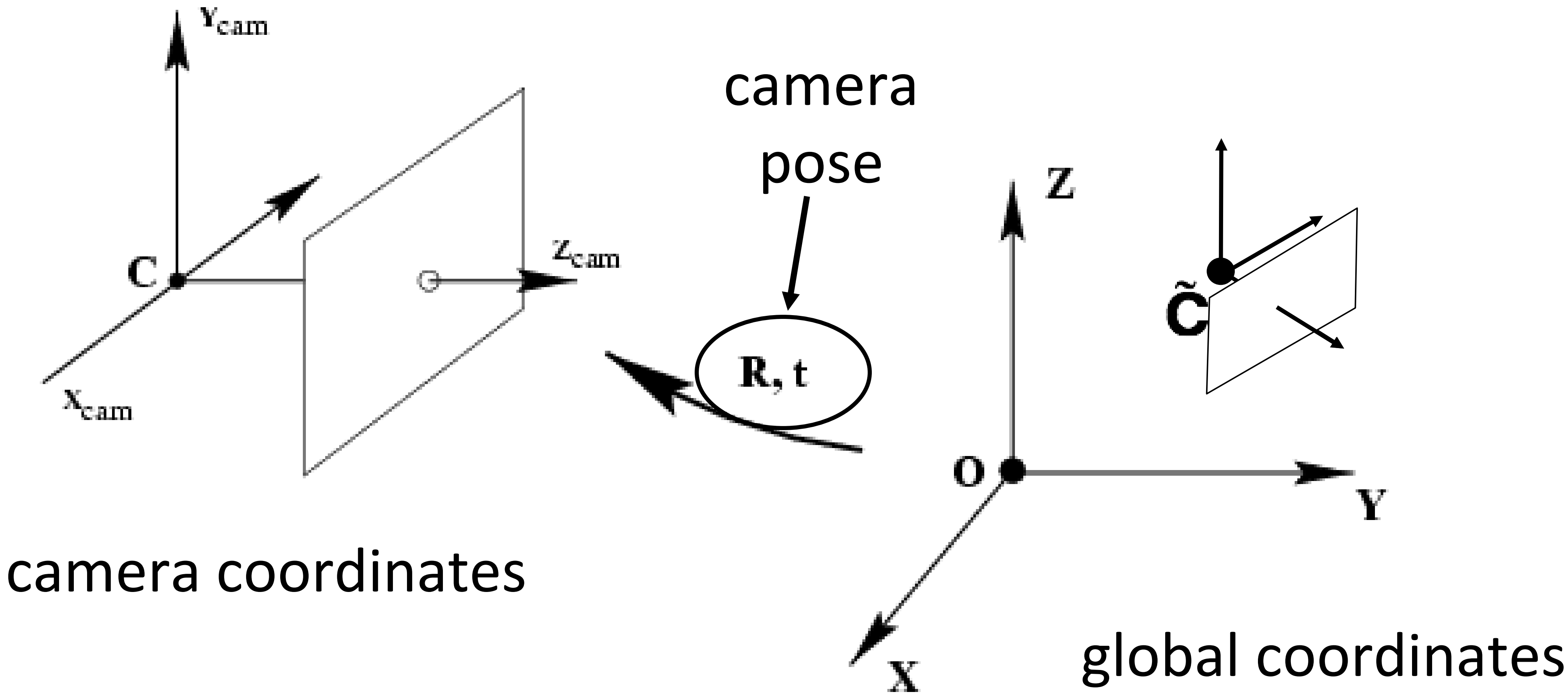
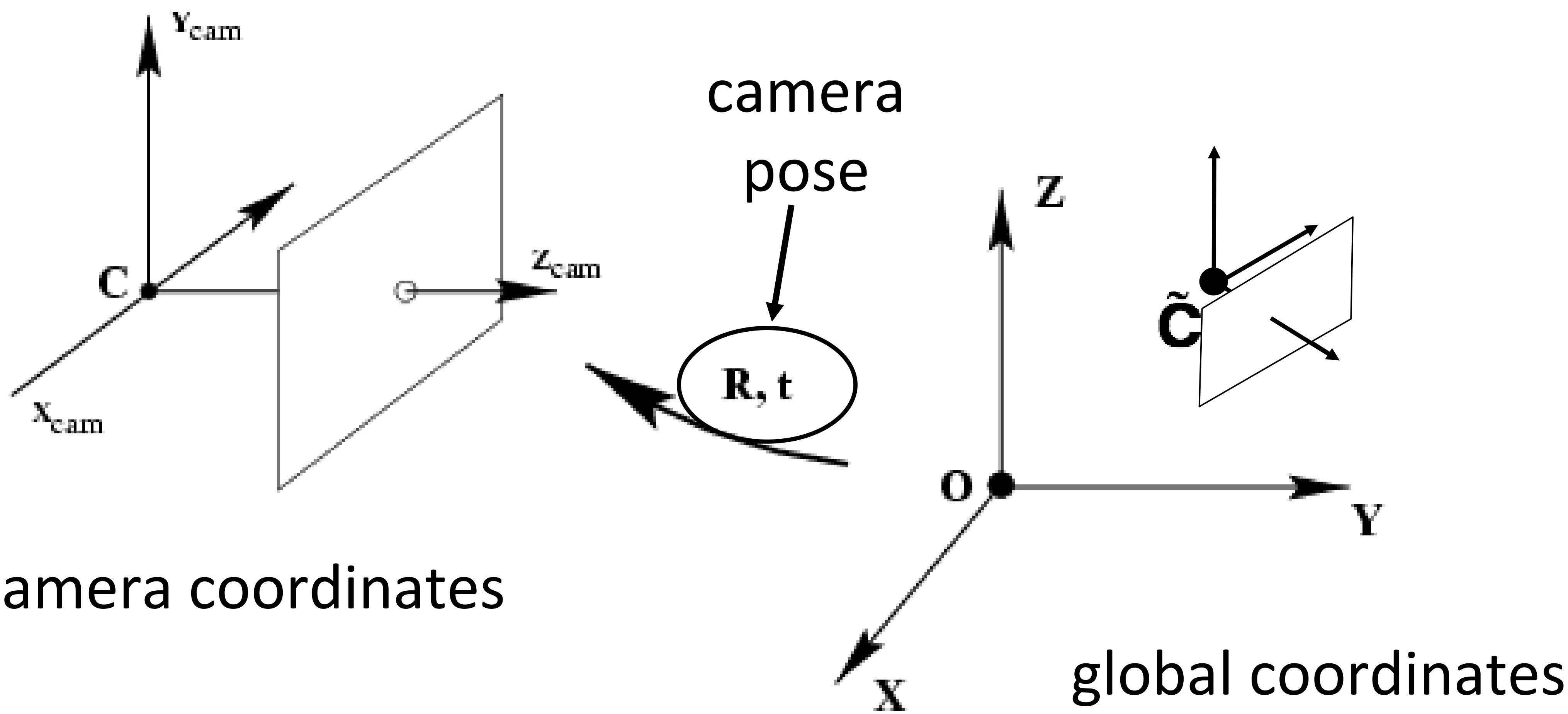


figure adapted from Hartley and Zisserman, 2004

Camera Rotation and Translation



camera coordinates

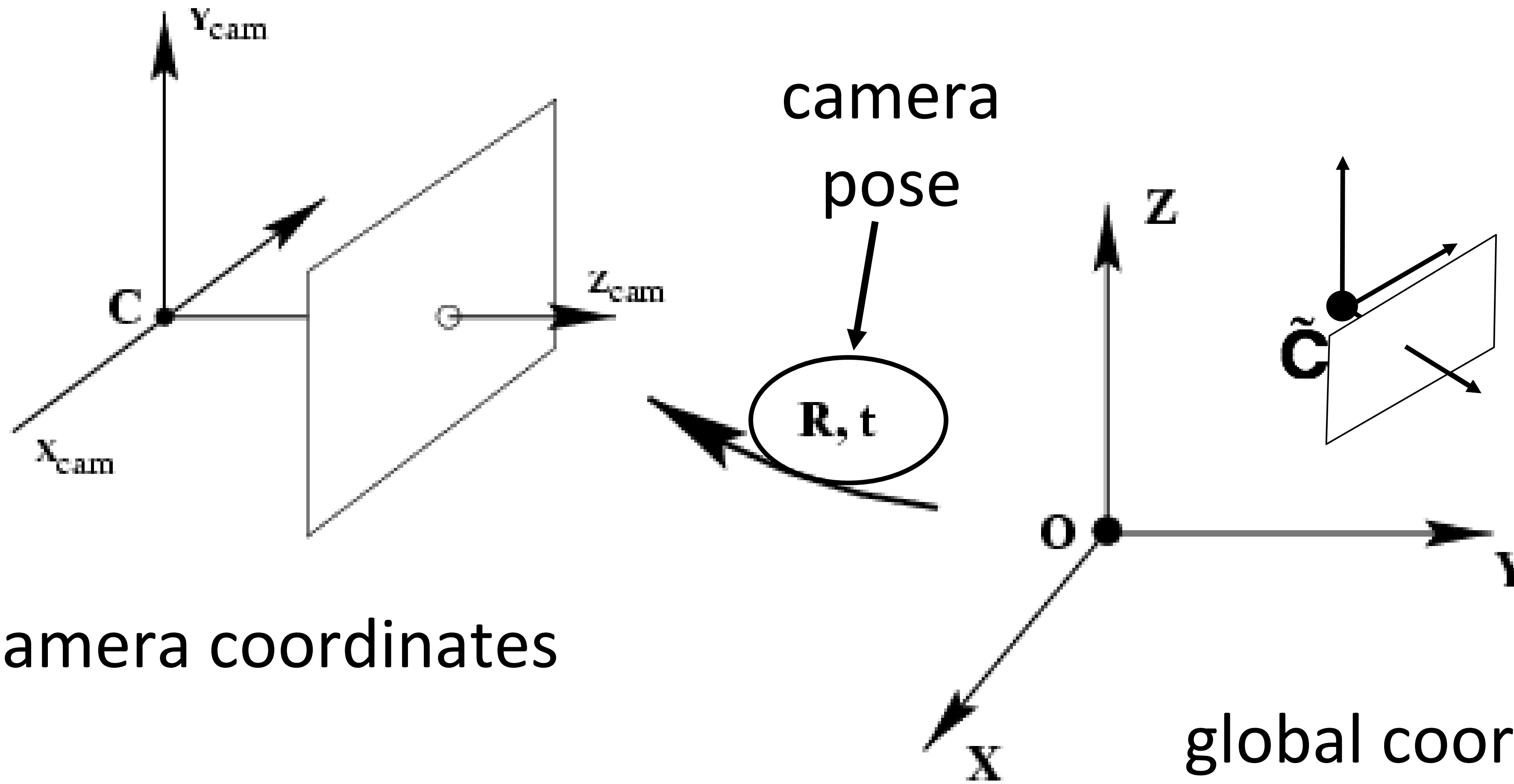
global coordinates

Transformation from global to camera coordinates:

$$\mathbf{X}_{\mathrm{cam}} = \mathtt{R}\left(\mathbf{X}_{\mathrm{global}} - \tilde{\mathbf{C}}\right)$$

figure adapted from Hartley and Zisserman, 2004

Camera Rotation and Translation



camera coordinates

global coordinates

Transformation from global to camera coordinates:

R, t = extrinsic camera parameters

$$
\begin{aligned}
\mathbf{X}_{\mathrm{cam}} &= \mathrm{R}\left(\mathbf{X}_{\mathrm{global}} - \tilde{\mathrm{C}}\right) \\
&= \mathrm{R}\mathbf{X} + \mathbf{t}
\end{aligned}
$$

figure adapted from Hartley and Zisserman, 2004

The Projection Matrix

Projection from 3D global coordinates to pixels:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathtt{K}\left(\mathtt{R}\mathbf{X}_{\text{global}} + \mathbf{t}\right)$$

figure adapted from Hartley and Zisserman, 2004

The Projection Matrix

Projection from 3D global coordinates to pixels:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathtt{K}\left(\mathtt{R}\mathbf{X}_{\text{global}} + \mathbf{t}\right)$$

$$= \underset{\in \mathbb{R}^{3\times 3}}{\mathtt{K}} \underbrace{[\mathtt{R}|\mathbf{t}]}_{\in \mathbb{R}^{3\times 4}} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

$X \in \mathbb{R}^3, \begin{pmatrix} X \\ 1 \end{pmatrix} \in \mathbb{R}^4$

figure adapted from Hartley and Zisserman, 2004

The Projection Matrix

Projection from 3D global coordinates to pixels:

$$\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \mathtt{K}\left(\mathrm{R}\mathbf{X}_{\text{global}} + \mathbf{t}\right)$$

$$= \mathtt{K}\left[\mathrm{R}|\mathbf{t}\right]\begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

$$= \mathtt{K}\left[\mathrm{R}| - \mathrm{R}\tilde{\mathbf{C}}\right]\begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}$$

figure adapted from Hartley and Zisserman, 2004

The Projection Matrix

Projection from 3D global coordinates to pixels:

$$
\begin{aligned}
\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} &= \mathtt{K}\left(\mathrm{R}\mathbf{X}_{\text{global}} + \mathbf{t}\right) \\
&= \mathtt{K}\left[\mathrm{R}|\mathbf{t}\right] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \\
&= \mathtt{K}\left[\mathrm{R}| -\mathrm{R}\tilde{\mathbf{C}}\right] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \\
&= \mathtt{P} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}
\end{aligned}
$$

figure adapted from Hartley and Zisserman, 2004

The Projection Matrix

Projection from 3D global coordinates to pixels:

$$
\begin{aligned}
\lambda \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} &= \mathrm{K}\left(\mathrm{R}\mathbf{X}_{\text{global}} + \mathbf{t}\right) \\
&= \mathrm{K}\left[\mathrm{R}|\mathbf{t}\right] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \\
&= \mathrm{K}\left[\mathrm{R}| - \mathrm{R}\tilde{\mathbf{C}}\right] \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix} \\
&= \mathrm{P} \begin{pmatrix} \mathbf{X} \\ 1 \end{pmatrix}
\end{aligned}
$$

projection matrix $\in \mathbb{R}^{3\times 4}$

figure adapted from Hartley and Zisserman, 2004

Rolling Shutter Effect

[video]

Rolling Shutter Effect

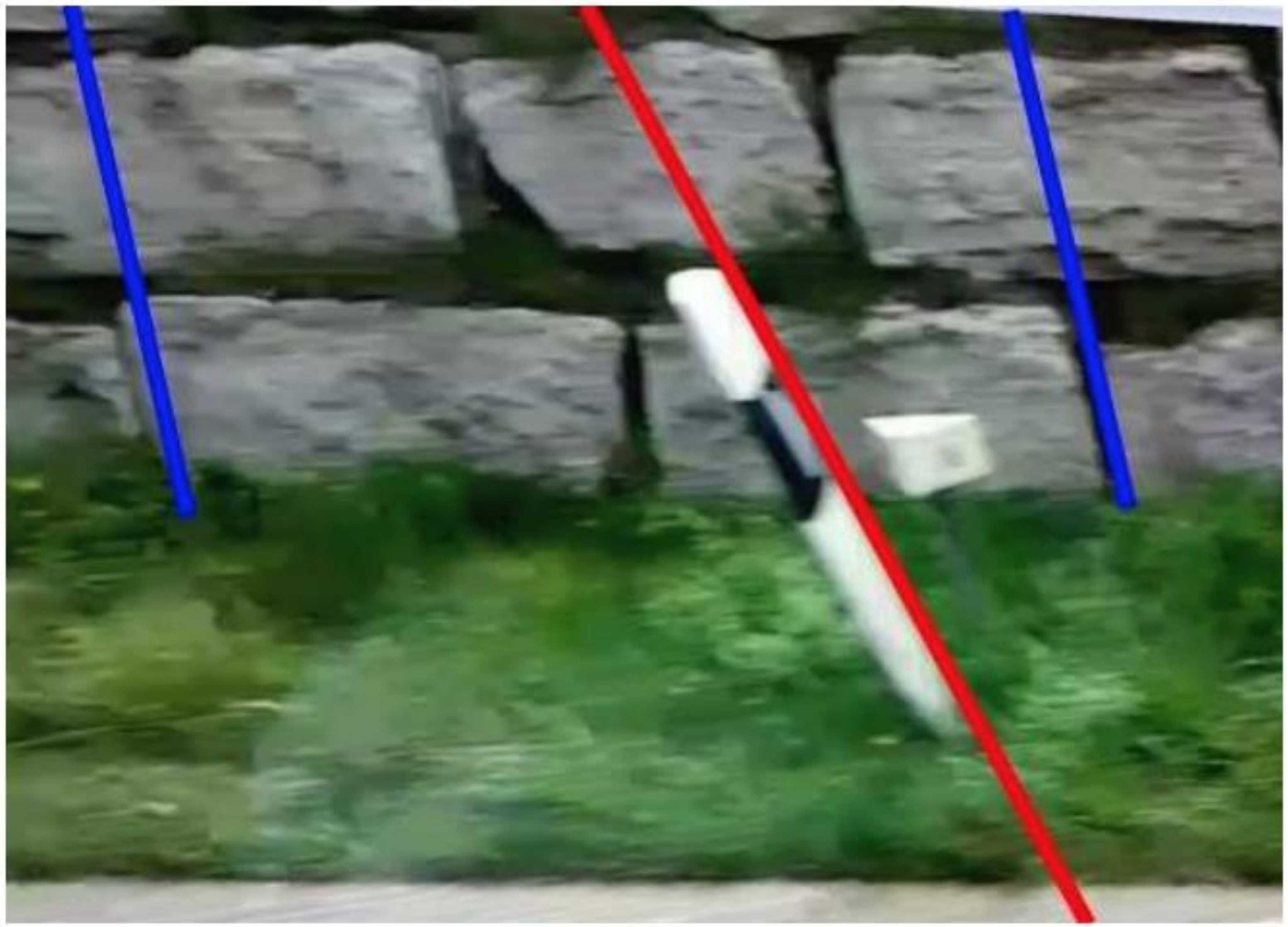
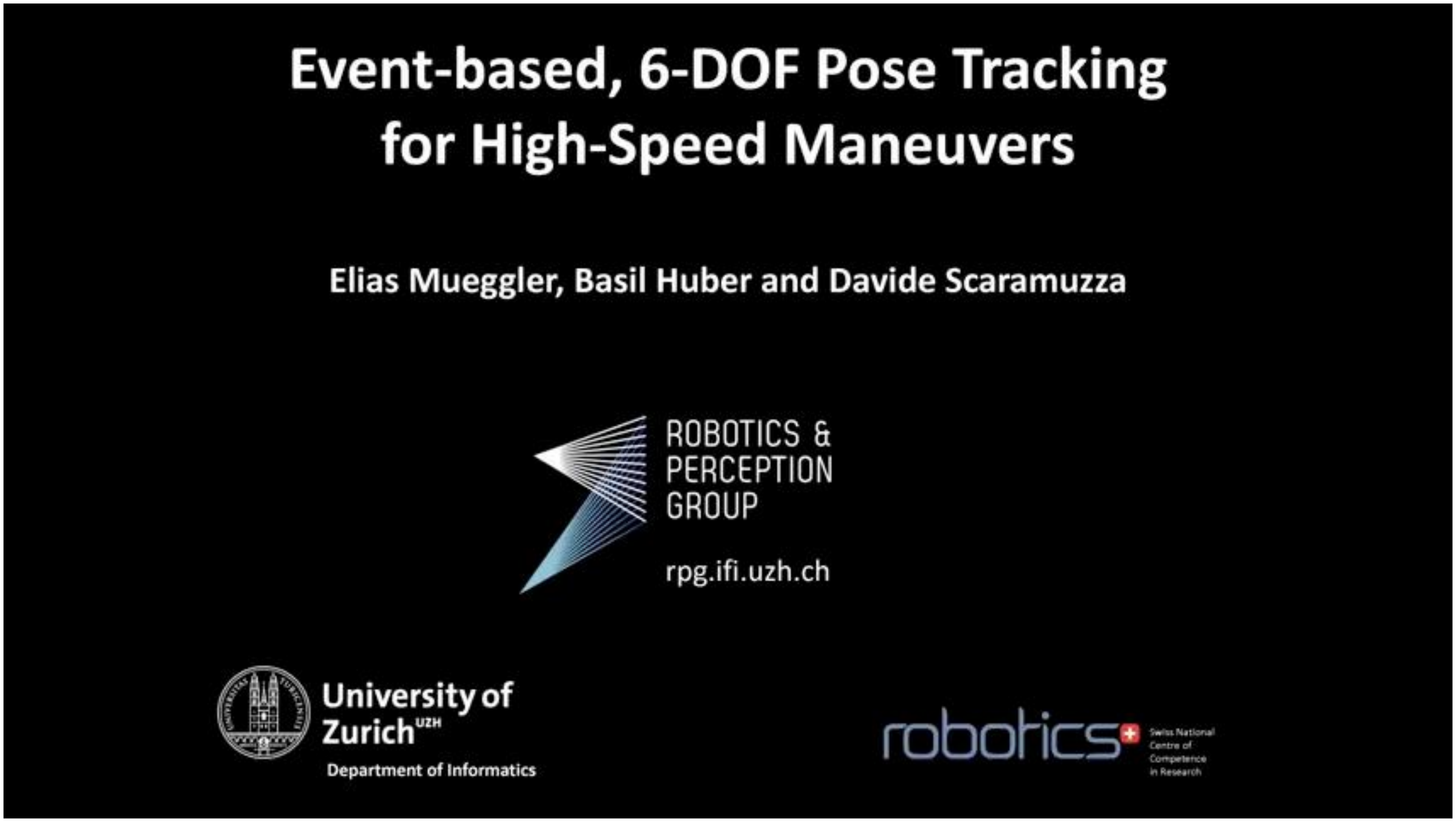


Image is exposed row by row

Saurer, Köser, Bouguet, Pollefeys. *Rolling Shutter Stereo*, ICCV 2013

Event Cameras



video

Lessons Learned

- Main lessons from this lecture
  - 2D projective transformations via homogeneous coordinates
  - 3D-to-2D projection via homogenous coordinates
  - Transformation from global to camera coordinates
  - What are intrinsic and extrinsic camera parameters?
  - What is a projection matrix?
  - How to compute the different types of transformations?
- Next lecture: More 3D Geometry

Next Lecture

Jan. 20	Introduction, Linear classifiers and filtering	
Jan. 23	Filtering, gradients, scale	Lab 1
Jan. 27	Local features	
Jan. 30	Learning a classifier	Lab 2
Feb. 3	Convolutional neural networks	
Feb. 6	More convolutional neural networks	
Feb. 10	Robust model fitting and RANSAC	Lab 3
Feb. 13	Image registration	
Feb. 17	Camera Geometry	Lab 4
Feb. 20	**More camera geometry**	
Feb. 24	Generative neural networks	
Feb. 27	Generative neural networks	
Mar. 2	TBA	
Mar. 9	TBA	