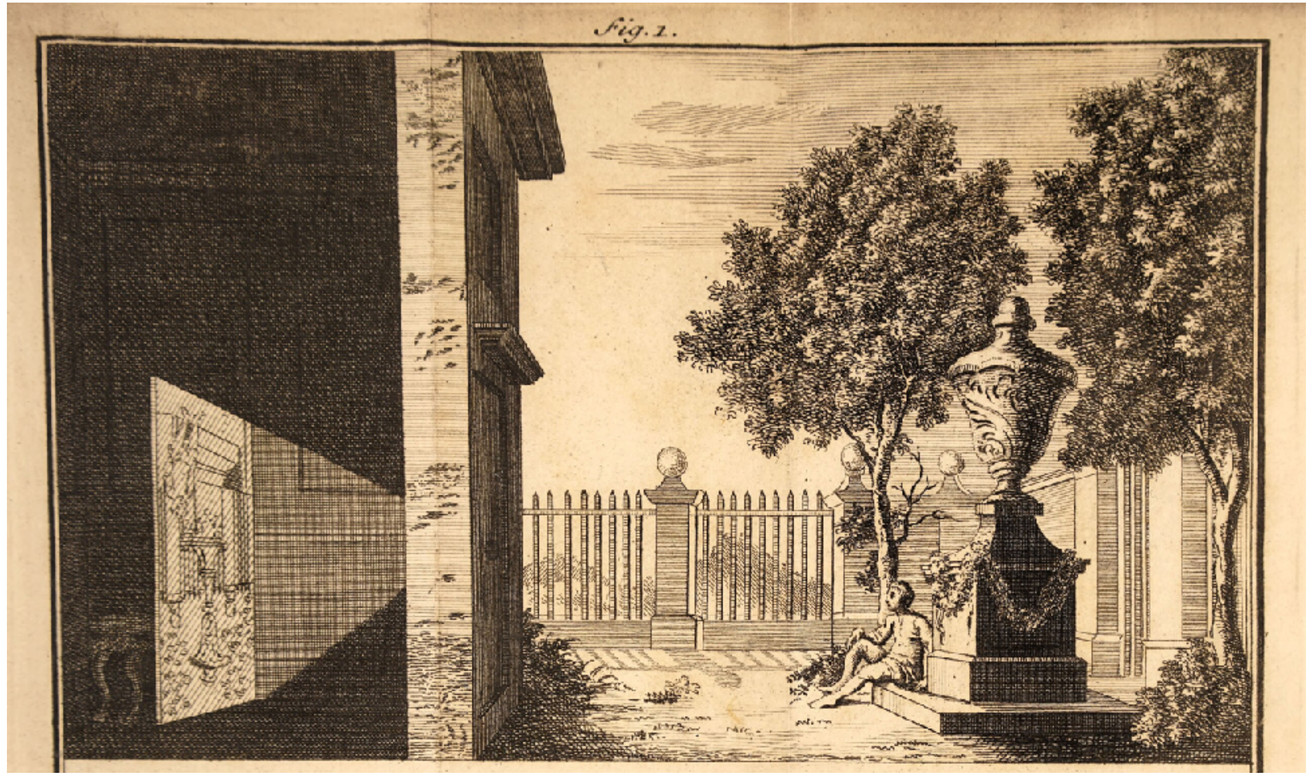
II: Multi-View Geometry

3D CV

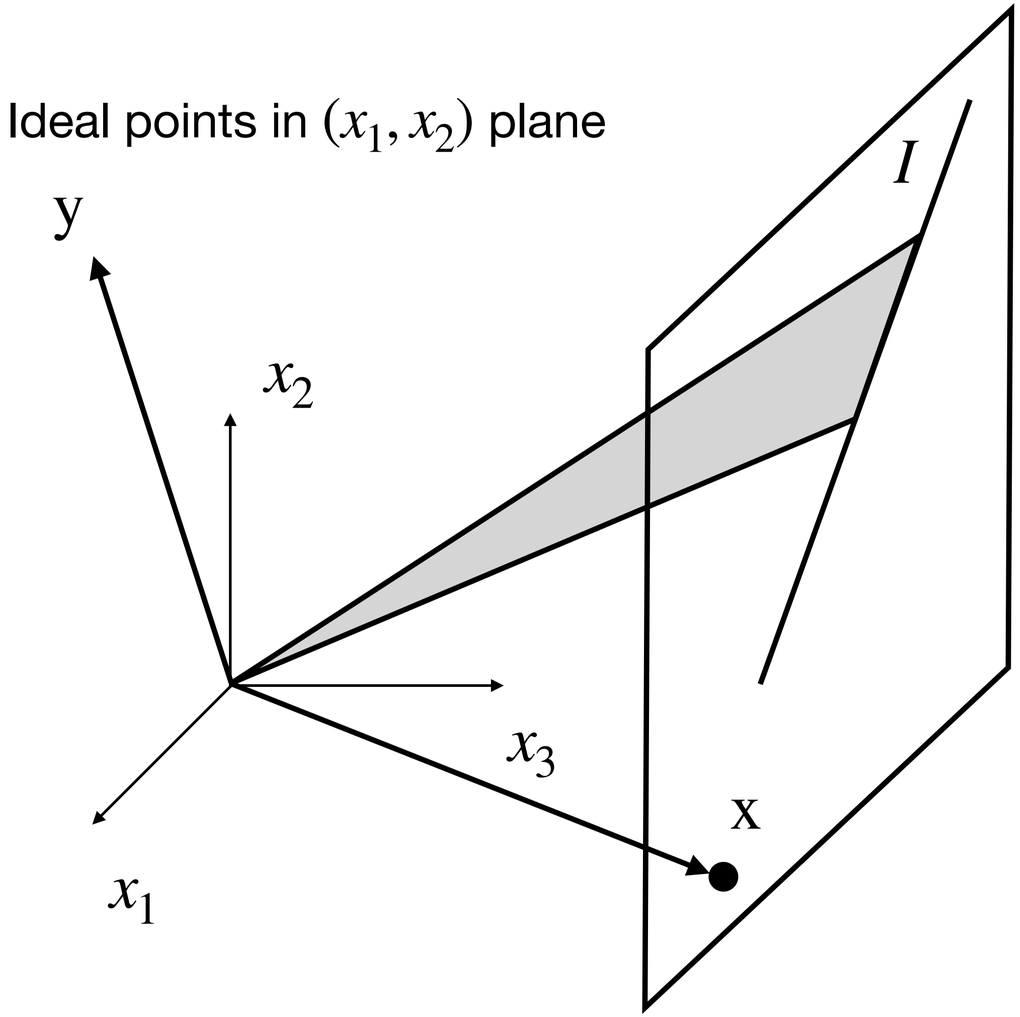
Kirill Struminsky

In the Previous Episode

- Image formation
- Homogeneous coordinates
- Projective transformations



Homogeneous Coordinates for $\mathbb{R}^n$

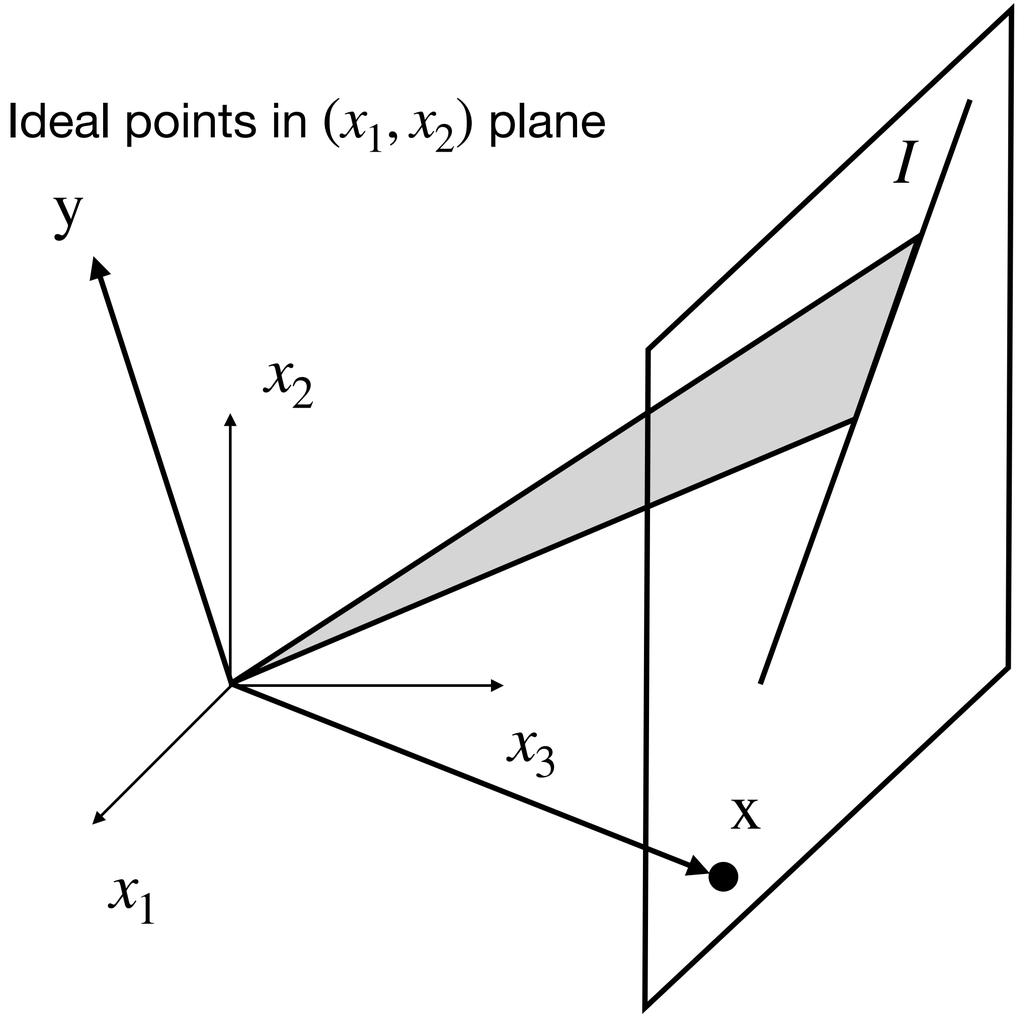


- Map $\mathrm{x} \in \mathbb{R}^n$ into $\mathbb{R}^{n+1}$

$$\tilde{\mathrm{x}} = (x_1, \ldots, x_n, 1)$$

- Say $\tilde{\mathrm{x}} \sim \tilde{\mathrm{y}}$ if $\exists \lambda : \mathrm{x} = \lambda \mathrm{y}$
- Projective space $\mathbb{P}^n$ is a set of equivalence classes in $\mathbb{R}^{n+1}$ w.r.t. $\sim$
- Added ideal points $(w_1, \ldots, w_n, 0)$

Homogeneous Coordinates for $\mathbb{R}^2$



- Point on the projective plane

$$\tilde{\mathrm{x}} = (x_1, x_2, x_3)$$

- Equation below defines proj. line for $\mathrm{I} = (a, b, c)$

$$\mathrm{I} \cdot \tilde{\mathrm{x}} = ax_1 + bx_2 + cx_3 = 0$$

- Lines I_1 and I_2 intersect at $\mathrm{I}_1 \times \mathrm{I}_2$
- Points $\tilde{\mathrm{x}}$ and $\tilde{\mathrm{y}}$ lie on a line $\tilde{\mathrm{x}} \times \tilde{\mathrm{y}}$

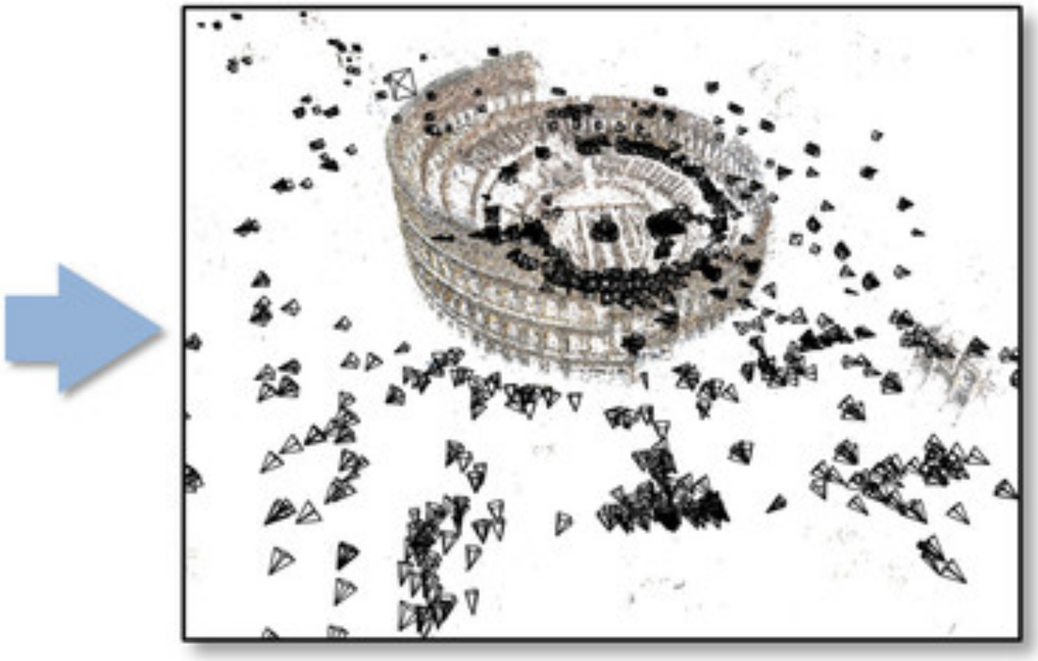
Projective Transformations

- In homogeneous coordinates, a 3×4 matrix defines a pinhole camera

$$\mathrm{H} = \left[\mathrm{K} \mid 0\right] \mathrm{C}^{W2C} = \begin{bmatrix} f_x & 0 & p_x & 0 \\ 0 & f_y & p_y & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathrm{R} & -\mathrm{Rt} \\ 0 & 1 \end{bmatrix}$$

- Matrix K defines ***intrinsic parameters*** of a camera
- Matrix C^{W2C} defines ***extrinsic parameters*** with $\mathrm{R} \in SO(3)$ and $\mathrm{t} \in \mathbb{R}^3$

Today's Lecture: Structure from Motion (SfM)



Structure From Motion is Still Relevant

- Fundamental 3D vision setup
- Has been under development for over 40 years
- ImageNet moment has not arrived (yet)
- Most advanced reconstruction techniques still rely on SfM for preprocessing

Not in Today's Lecture

Correspondences, Random Sample Consensus, Incremental Initialisations

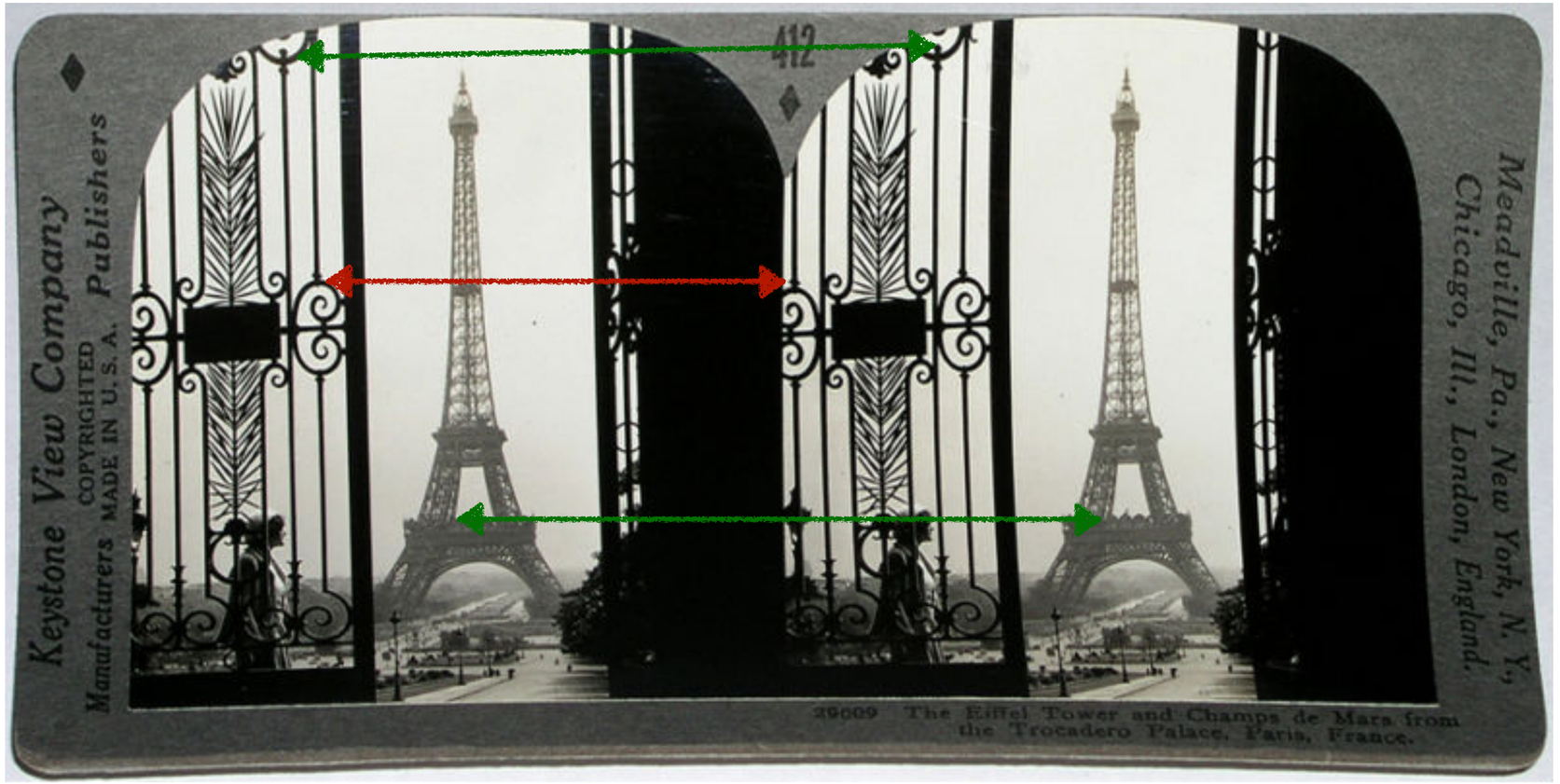
- Correspondences

$$\mathrm{x}_l \leftrightarrow \mathrm{x}_r$$

- For some x and projections H_l and H_r

$$\mathrm{x}_l = \mathrm{H}_l \mathrm{x}$$

$$\mathrm{x}_r = \mathrm{H}_r \mathrm{x}$$



Sic Parvis Magna

A More Detailed Plan

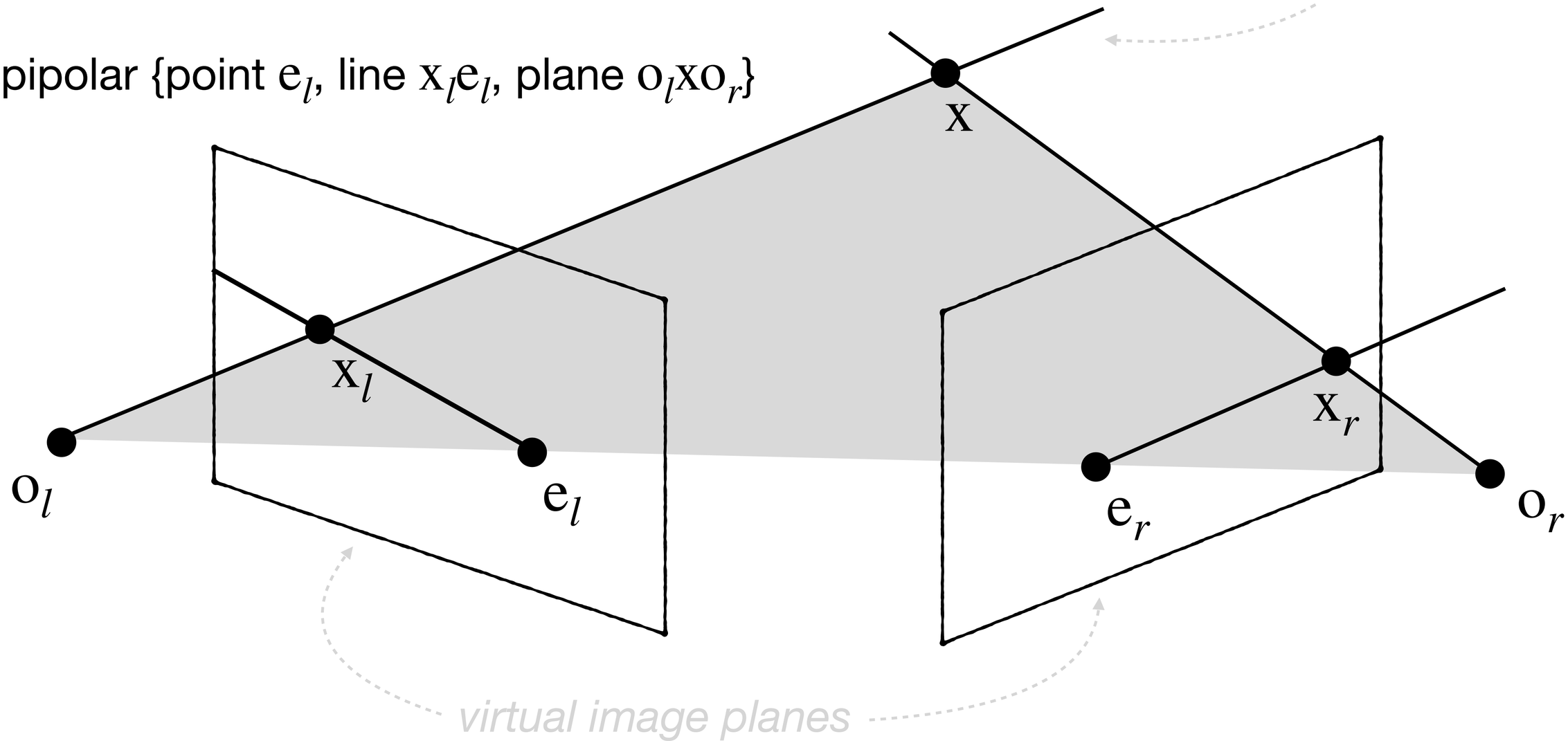
- SfM simultaneously recovers
  - Scene structure as a sparse point cloud
  - Positions of *multiple* cameras
- We will consider key basic components for a stereo-pair
  - How does the image change as we change location? ***Epipolar geometry***
  - Can we infer camera positions from images? ***Fundamental matrix***
  - How one reconstruct 3D structure given camera positions? ***Triangulation***

Epipolar Geometry

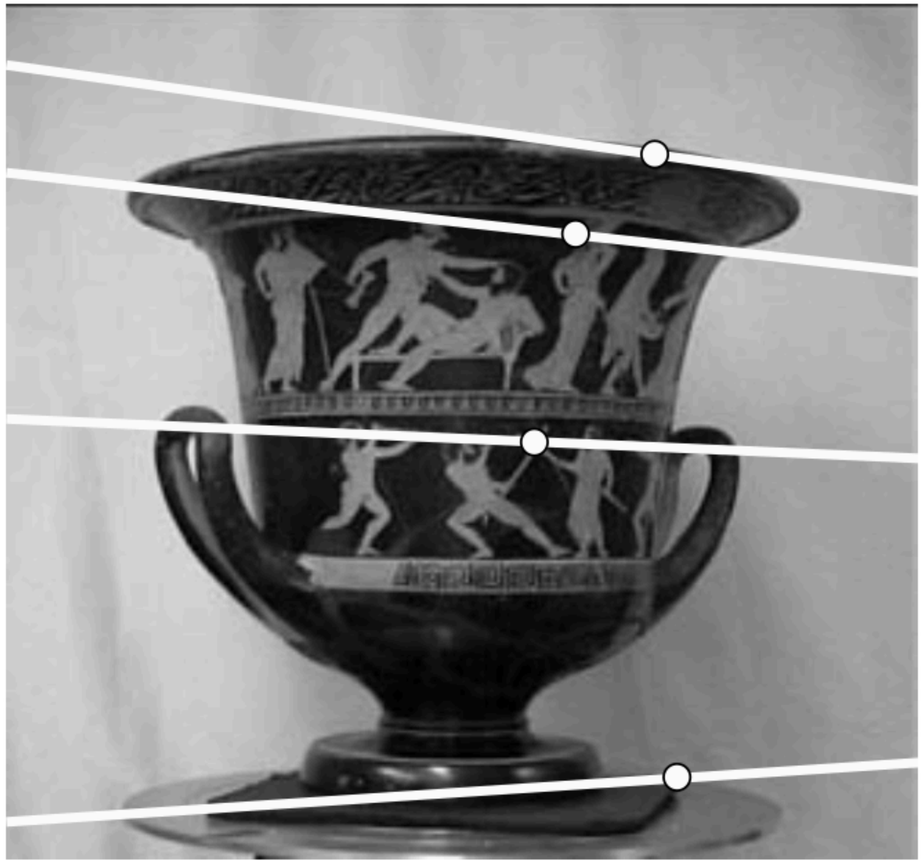
Epipolar Geometry

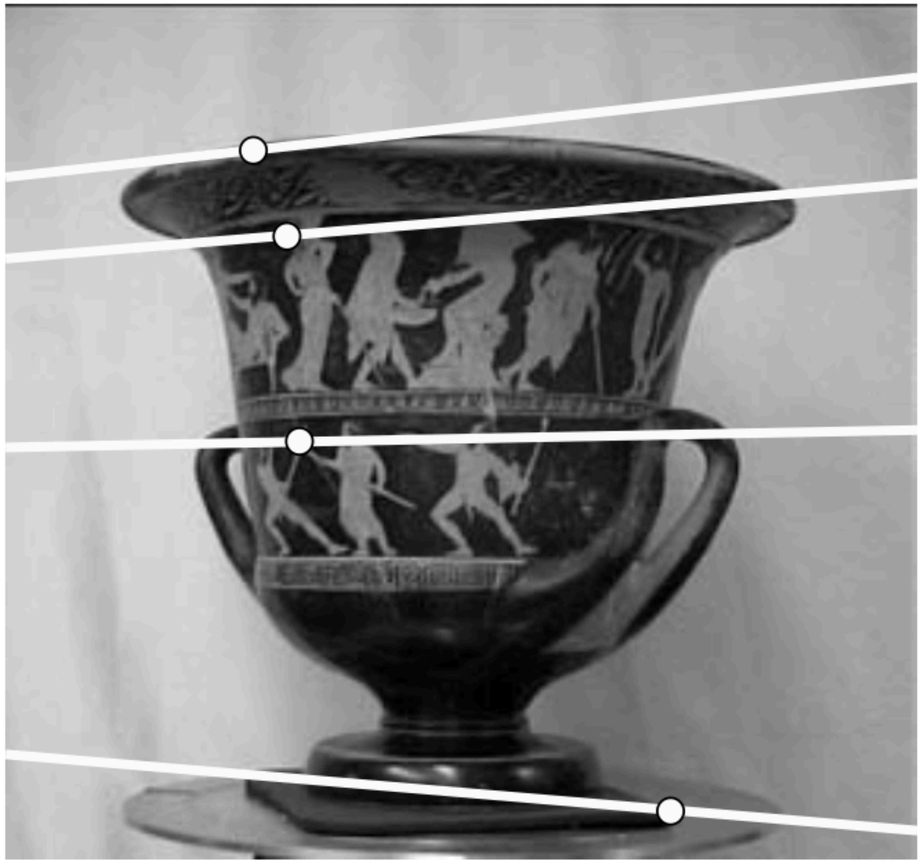
How does the image changes as we move?

epipolar {point e_l, line $x_l e_l$, plane $o_l x o_r$}



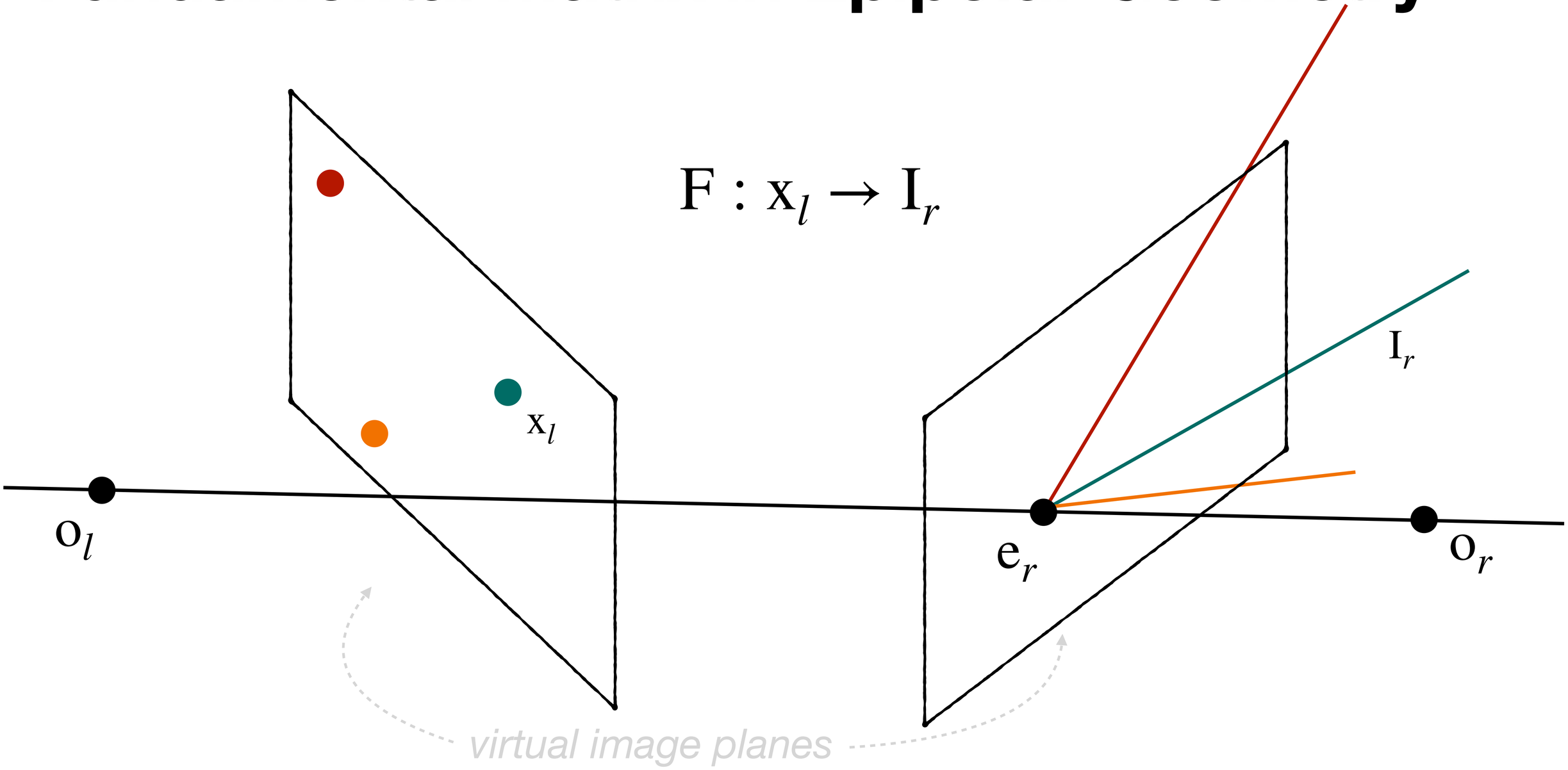
Epipolar Lines in Real Life





Hartley & Zisserman, Multiple View Geometry in CV, p. 241

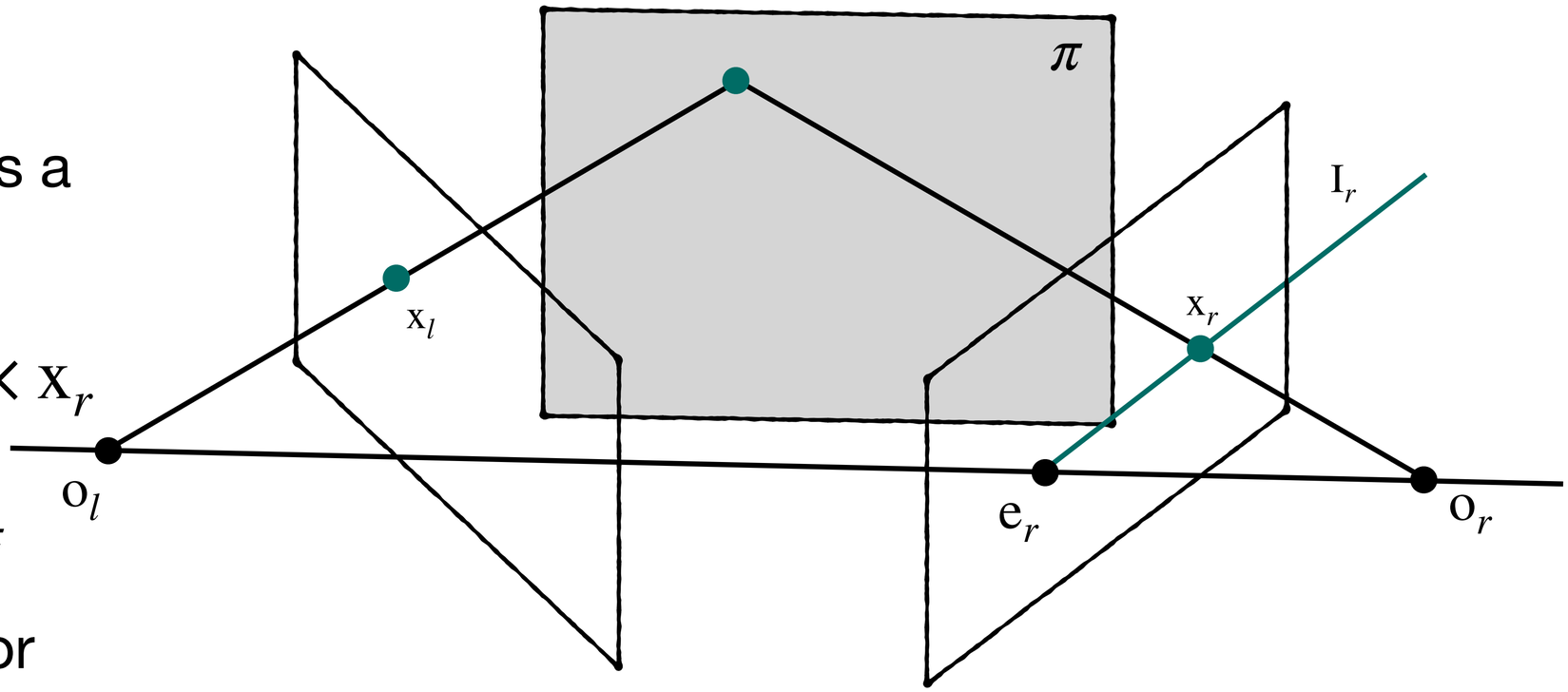
Fundamental Matrix in Epipolar Geometry



Fundamental Matrix

Fundamental Matrix: Geometric Viewpoint

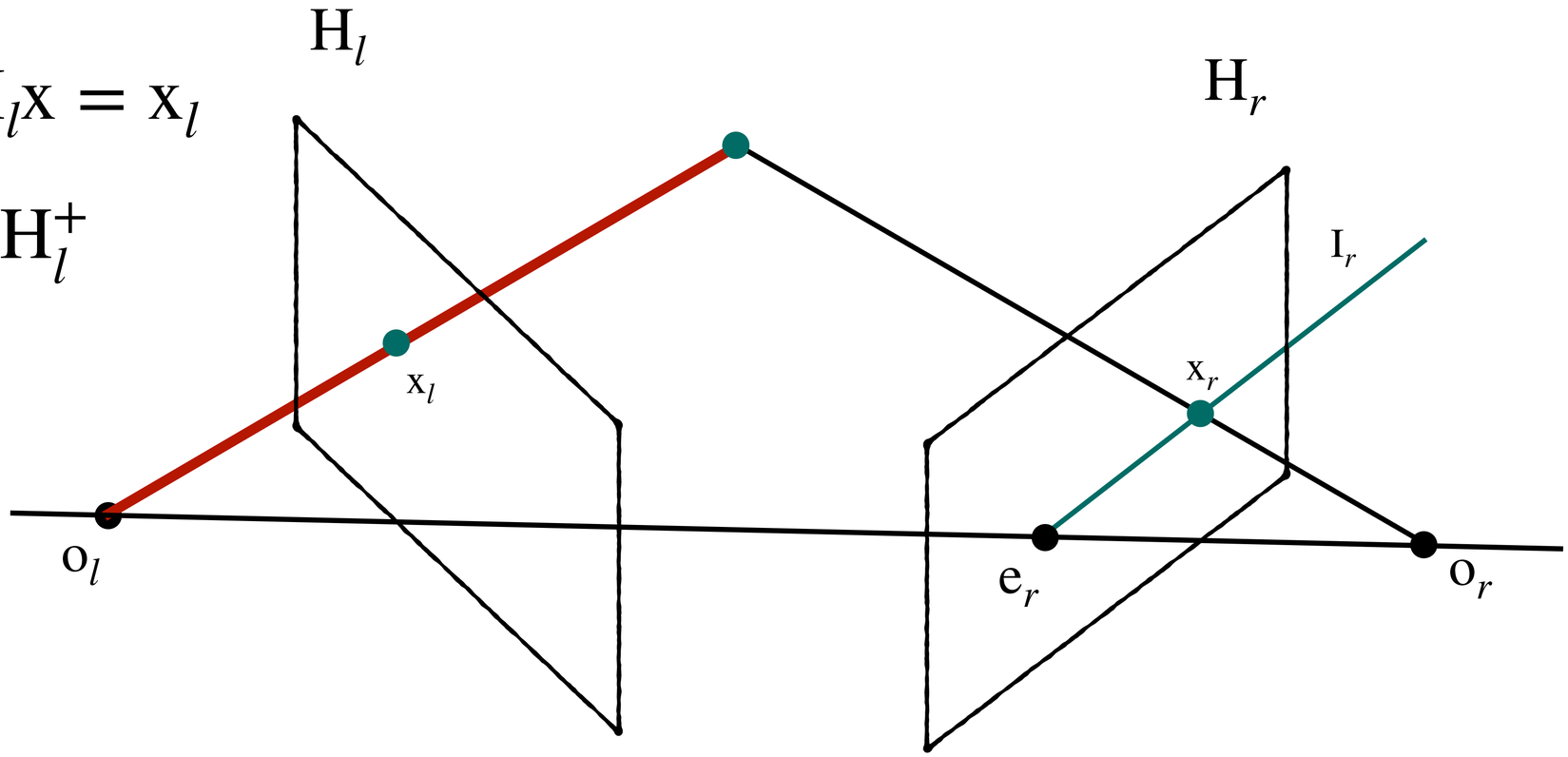
- Step 1: map x_l to x_r
  - Reprojection through π is a homography H_π
- Step 2: map x_r to $\mathrm{I}_r = \mathrm{e}_r \times \mathrm{x}_r$
- Composition: $\mathrm{F} = [\mathrm{e}_r]_\times \mathrm{H}_\pi$
  - $[\mathrm{e}_r]_\times$ denotes a matrix for vector product



Fundamental Matrix: Algebraic Viewpoint

Ray back-projection

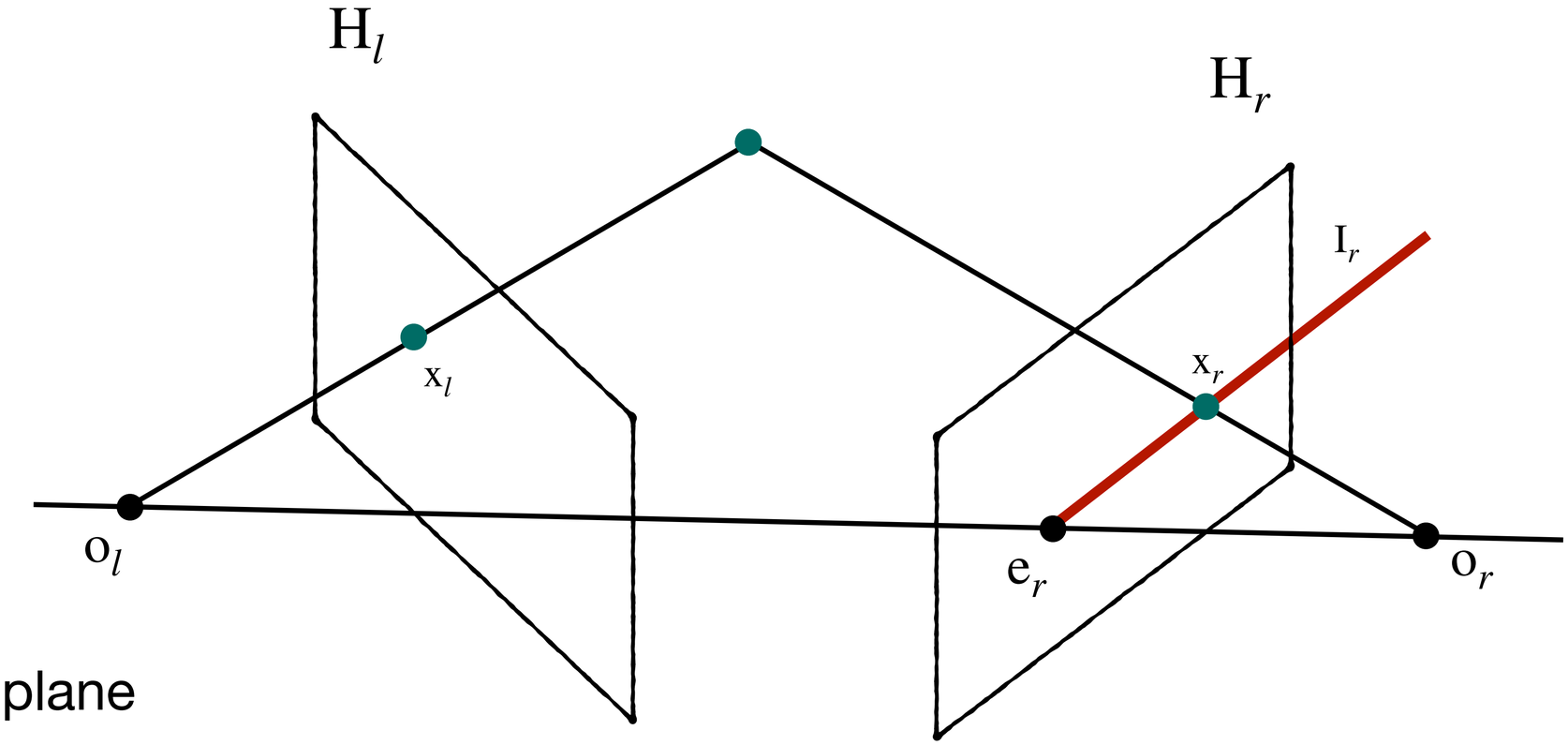
- Ray $\mathrm{o}_l\mathrm{x}_l$ is a solution to $\mathrm{H}_l\mathrm{x} = \mathrm{x}_l$
- Consider pseudo-inverse H_l^+
  - $\mathrm{H}_l\mathrm{H}_l^+ = \mathrm{I}$
- Explicit parameterisation:
  - $\mathrm{x} = \mathrm{H}_l^+\mathrm{x}_l + \lambda\mathrm{o}_l$
  - Camera center solves $\mathrm{H}_l\mathrm{o}_l = 0$
  - Pseudo-inverse: $\mathrm{H}_l(\mathrm{H}_l^+\mathrm{x}_l) = \mathrm{x}_l$



Fundamental Matrix: Algebraic Viewpoint

Epipolar Line Construction

- Back-projected ray:
  - $\mathrm{x} = \mathrm{H}_l^+ \mathrm{x}_l + \lambda \mathrm{o}_l$
- Consider two points:
  - $\lambda = 0 : \mathrm{H}_l^+ \mathrm{x}_l$
  - $\lambda = \infty : \mathrm{o}_l$
- Project them onto the right plane
  - $\mathrm{e}_r = \mathrm{H}_r \mathrm{o}_l, \;\; \mathrm{H}_r \mathrm{H}_l^+ \mathrm{x}_l$
- Construct the line $\mathrm{I}_r = [e_r]_\times \mathrm{H}_r \mathrm{H}_l^+ \mathrm{x}_l = \mathrm{F}\mathrm{x}_l$



Fundamental Matrix: Calibrated Cameras

- Consider $\mathrm{H}_l = \begin{bmatrix}\mathrm{K}_l | 0\end{bmatrix} \begin{bmatrix}\mathrm{I} & 0 \\ 0 & 1\end{bmatrix}$ and $\mathrm{H}_l = \begin{bmatrix}\mathrm{K}_r | 0\end{bmatrix} \begin{bmatrix}\mathrm{R} & \mathrm{t} \\ 0 & 1\end{bmatrix}$
  - Matrix $\mathrm{R} \in SO(3)$ and vector $\mathrm{t} \in \mathbb{R}^3$ parameterise relative positions of cameras
- Compute fundamental matrix
  - We have $\mathrm{H}_l^+ = \begin{bmatrix}\mathrm{K}_l^{-1} \\ 0^T\end{bmatrix}$ and $o_l = \begin{pmatrix}0 \\ 0 \\ 1\end{pmatrix}$

$$\mathrm{F} = \begin{bmatrix}\mathrm{e}_r\end{bmatrix}_\times \mathrm{H}_r \mathrm{H}_l^+ = \begin{bmatrix}\mathrm{K}_r \mathrm{t}\end{bmatrix}_\times \mathrm{K}_r \mathrm{R} \mathrm{K}_l^{-1} = \ldots = \mathrm{K}_r^{-T} [\mathrm{t}]_\times \mathrm{R} \mathrm{K}_l^{-1}$$

Fundamental Matrix: Relative Positions

- Consider $\mathrm{H}_l = \left[\mathrm{K}_l \,|\, 0\right] \begin{bmatrix} \mathrm{I} & 0 \\ 0 & 1 \end{bmatrix}$ and $\mathrm{H}_l = \left[\mathrm{K}_r \,|\, 0\right] \begin{bmatrix} \mathrm{R} & \mathrm{t} \\ 0 & 1 \end{bmatrix}$
- Fundamental matrix encodes R and t:

$$\mathrm{F} = \mathrm{K}_r^{-T}[\mathrm{t}]_\times \mathrm{R} \mathrm{K}_l^{-1}$$

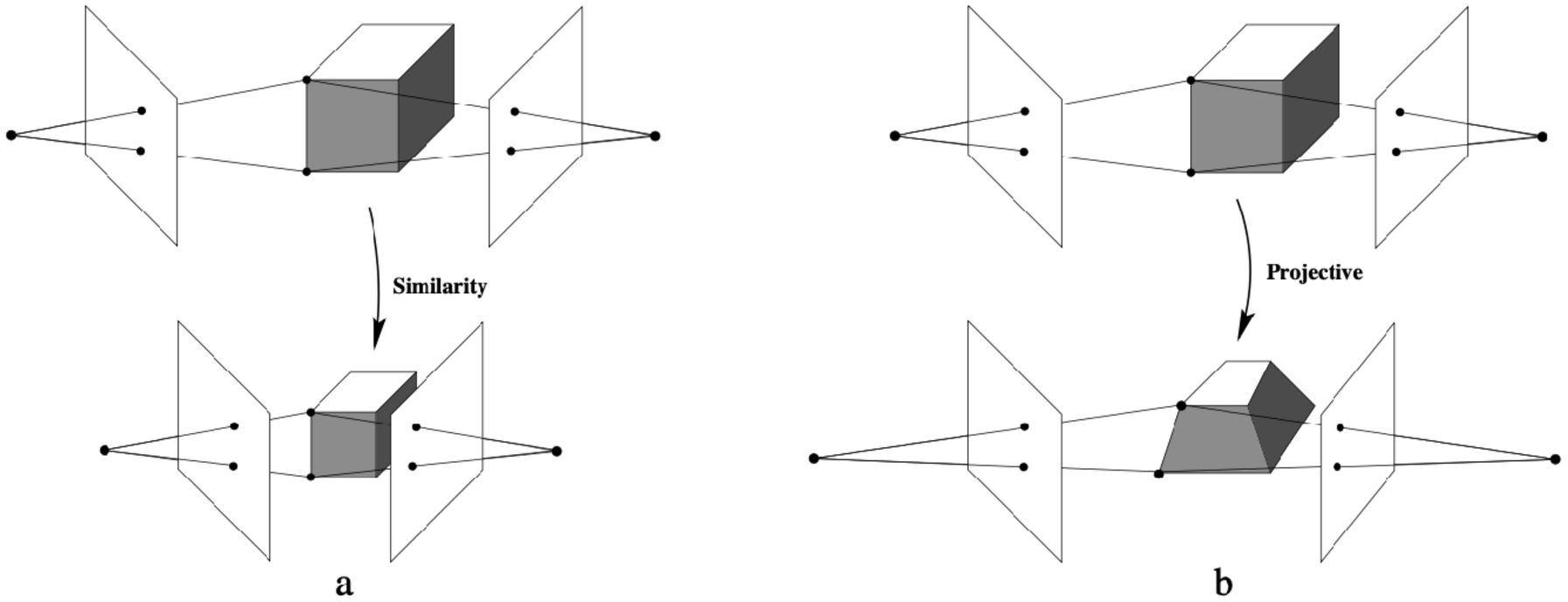
- Usually, we know K_r and K_l in advance
- Matrix $\mathrm{E} = [\mathrm{t}]_\times \mathrm{R}$ is known as **essential matrix**
- One can recover t and R from E *(up to scale scale)*

Estimation of Fundamental Matrix

- Note that $\mathrm{x}_r^T \mathrm{F} \mathrm{x}_l = 0$ for any corresponding pair $\mathrm{x}_l \leftrightarrow \mathrm{x}_r$
- Equation $\mathrm{x}_r^T \mathrm{F} \mathrm{x}_l = 0$ is a linear equation in F
- To recover F *(up to scale factor)* we need 8 point correspondences
- ***Eight-point algorithm:***
  - finds F gives points correspondences for a stereo-pair
- *How many points do we actually need? Hint:* $\mathrm{F} = \mathrm{K}_r^{-T} [\mathrm{t}]_\times \mathrm{R} \mathrm{K}_l^{-1}$

Structure Recovery

Reconstruction Ambiguity

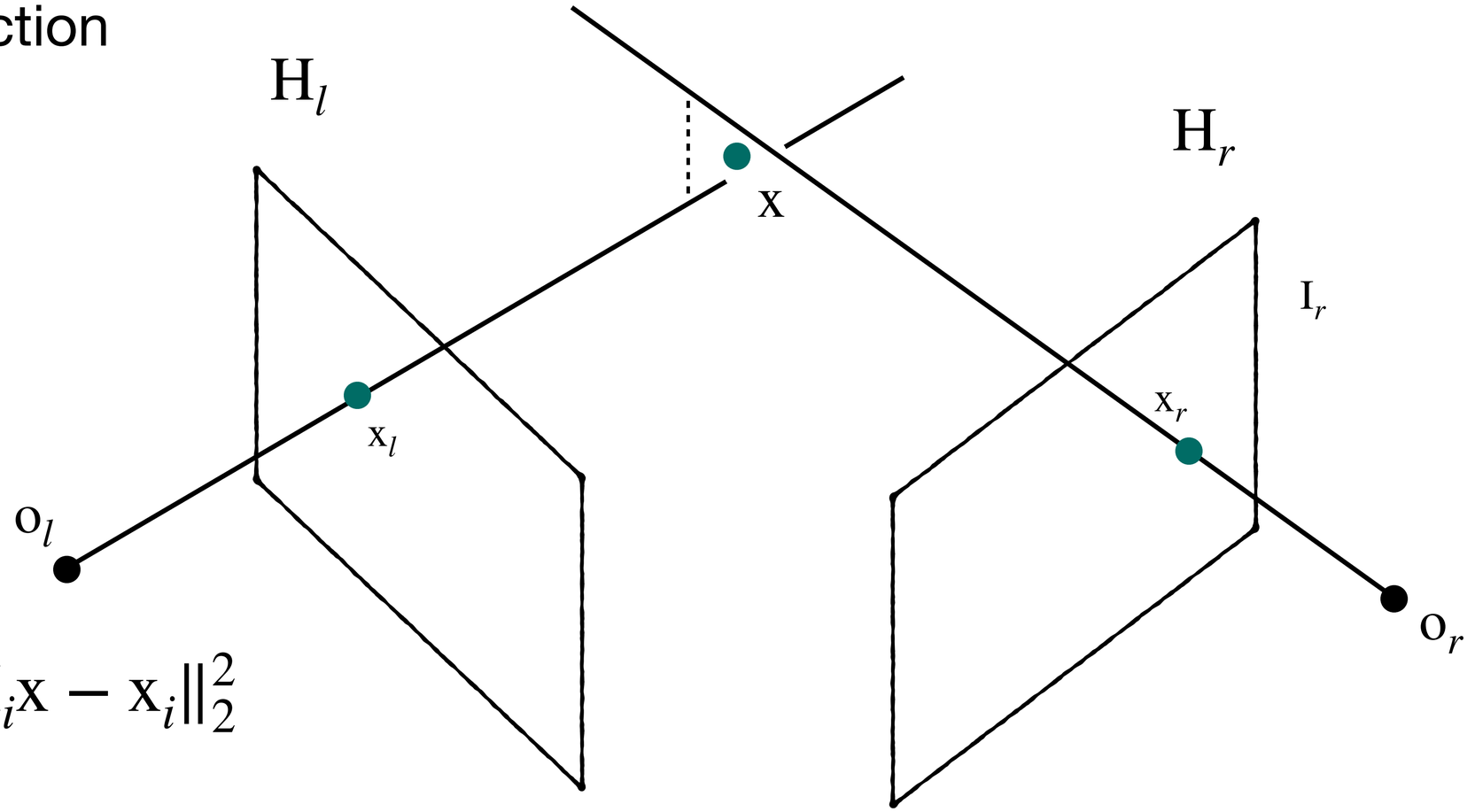


Hartley & Zisserman, Multiple View Geometry in CV, p. 265

Structure Recovery

- Naive: find the intersection
  - $\mathrm{H}_l^+ \mathrm{x}_l + \lambda \mathrm{o}_l$
  - $\mathrm{H}_r^+ \mathrm{x}_r + \mu \mathrm{o}_l$
- More practical:

$$\mathrm{x} = \arg\min \sum_{i \in \{l,r\}} \|\mathrm{H}_i \mathrm{x} - \mathrm{x}_i\|_2^2$$

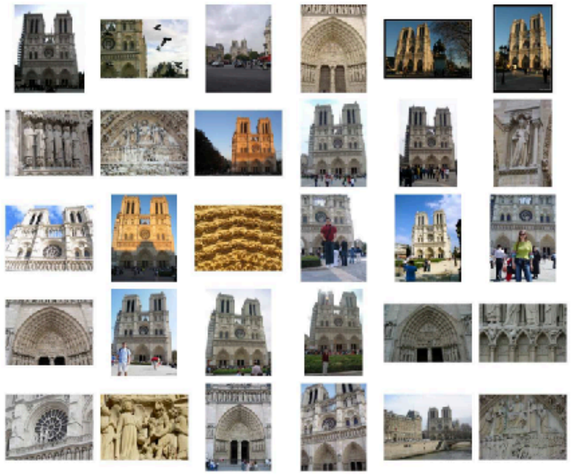


Bundle Adjustment

Bundle Adjustment

Goal

- Bundle Adjustment is a step in Structure From Motion Pipeline
- Input: images and point correspondences
- Aims to
    1. estimate camera parameters
    2. recover structure
- Does (1) and (2) simultaneously



Snavely, Noah, Steven M. Seitz, and Richard Szeliski. "Photo tourism: exploring photo collections in 3D." ACM SIGGRAPH 2006

Bundle Adjustment

Problem Formulation (with a slight shift in notation)

- Input: $\mathcal{X}_s = \{\mathrm{x}_{ip}^s\}$
  - i is the camera index
  - p is the point cloud index
- Parameters:
  - Cameras $\Pi = \{\pi_i\}, \quad \pi_i = \left(\mathrm{K}_i, \mathrm{R}_i, \mathrm{t}_i\right)$
  - Point cloud $\mathcal{X}_p = \{\mathrm{x}_p^w\}$

Bundle Adjustment

Problem Formulation (with a slight shift in notation)

- Input: $\mathcal{X}_s = \{\mathrm{x}_{ip}^s\}$
- Parameters: cameras $\Pi = \{\pi_i\}$ and point cloud $\mathcal{X}_p = \{\mathrm{x}_p^w\}$
- Objective: reprojection error

$$\Pi^*, \mathcal{X}_w^* = \arg\min_{\Pi, \mathcal{X}_w} \sum_i \sum_p w_{ip} \|\pi_i(\mathrm{x}_p^w) - \mathrm{x}_{ip}^s\|_2^2$$

where π_i denotes the projection of i-th screen and w_{ip} indicates visibility

Bundle Adjustment

In Practice

- Objective: reprojection error

$$\Pi^*, \mathcal{X}_w^* = \arg\min_{\Pi, \mathcal{X}_w} \sum_i \sum_p w_{ip} \| \pi_i(\mathrm{x}_p^w) - \mathrm{x}_{ip}^s \|_2^2$$

- Does not converge with an arbitrary initialisation
- Modern solution construct initialisation incrementally
  - Consider pair of images
  - Recover relative positions, approximate structure with triangulation etc.

Key takeaways

- *Fundamental matrix*
  - Helps inferring relative camera positions
  - Can be estimated based on point correspondences
- *Triangulation* recovers sparse structure given relative camera positions & correspondences
- Next lecture: depth estimation