Αντιστοίχιση Εικόνων

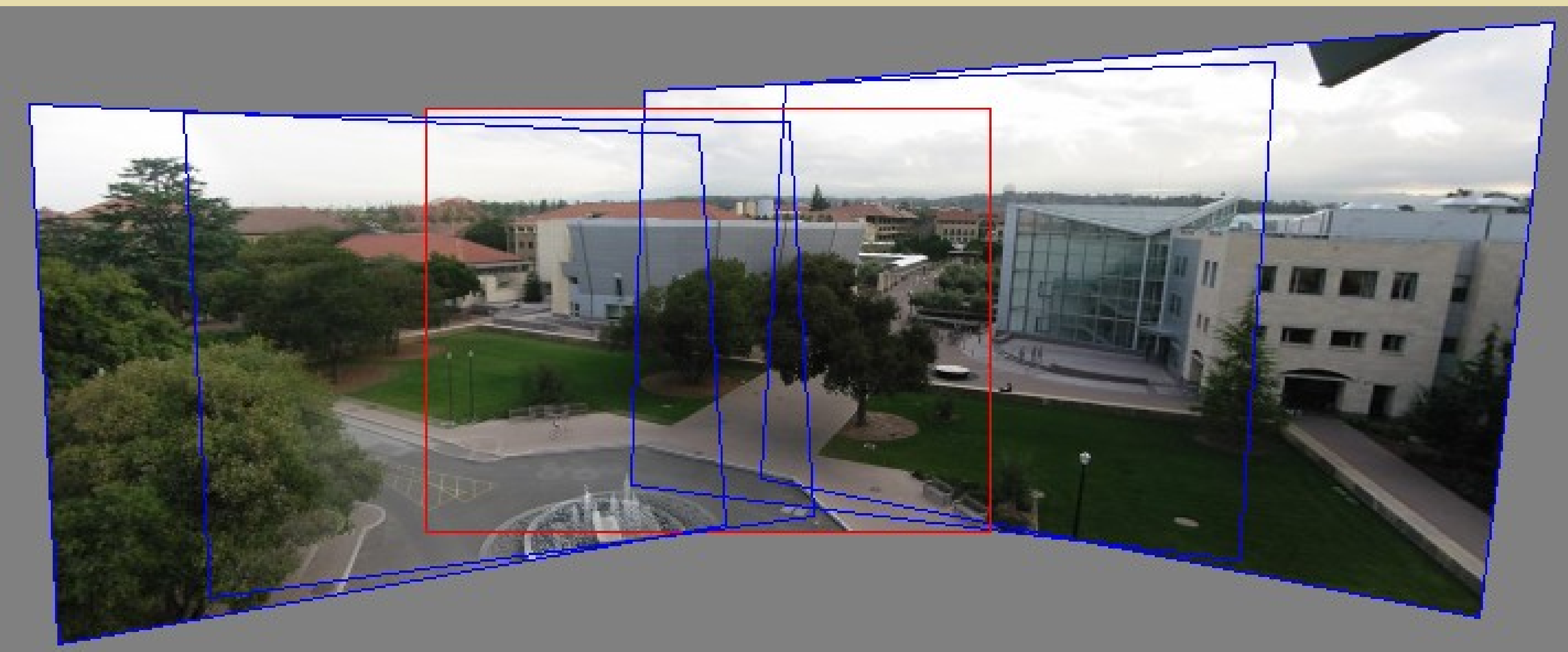
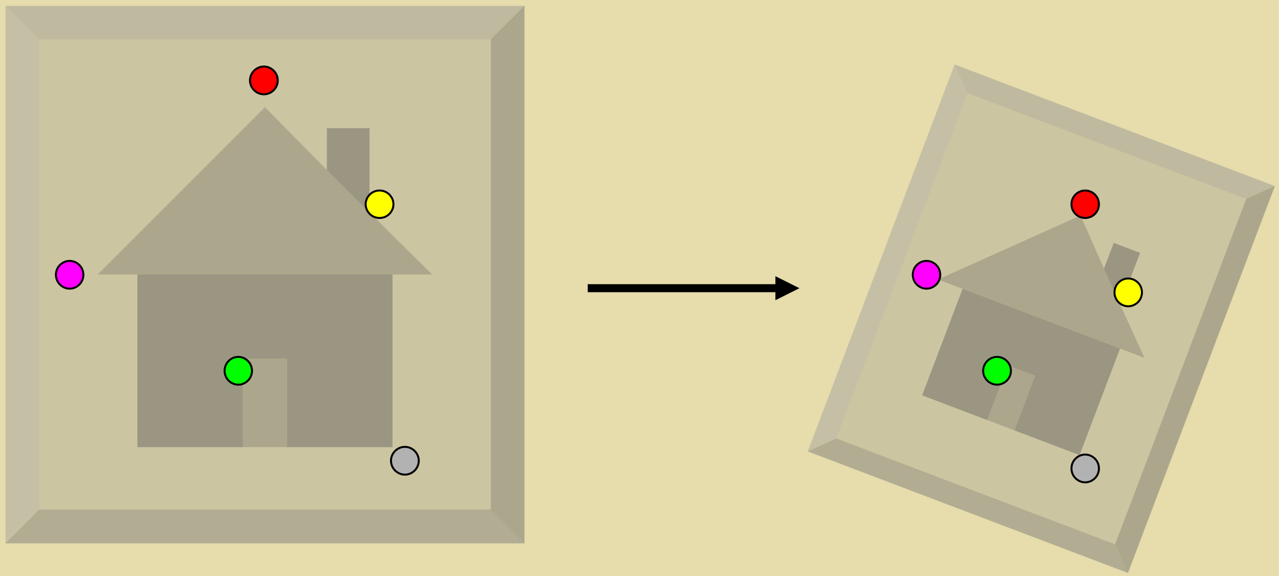
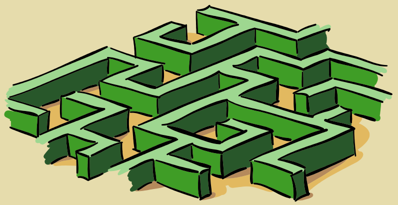


Image alignment

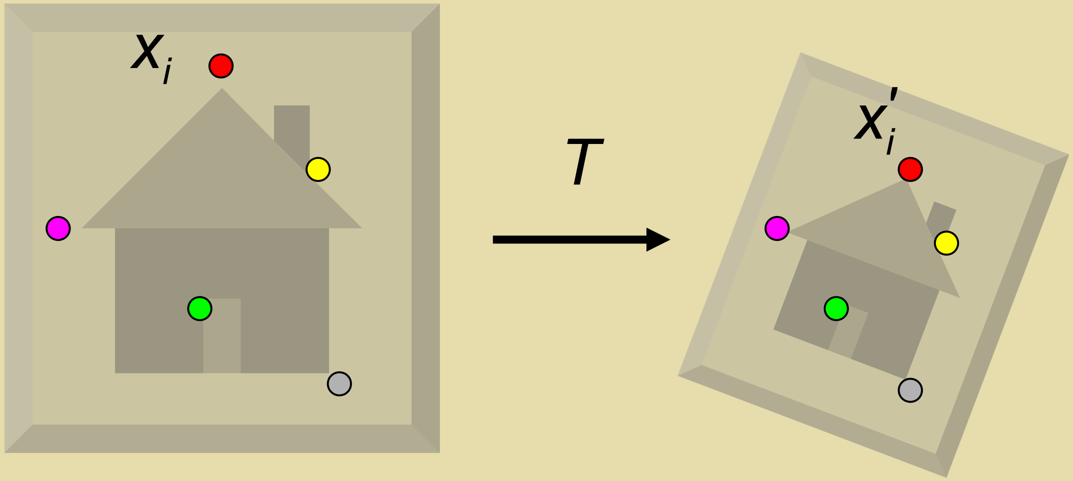


- Two broad approaches:
  - Direct (pixel-based) alignment
    - Search for alignment where most pixels agree
  - Feature-based alignment
    - Search for alignment where *extracted features* agree
    - Can be verified using pixel-based alignment



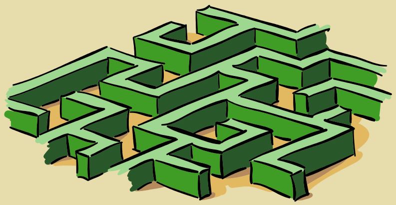
Alignment as fitting

- Alignment: fitting a model to a transformation between pairs of features (*matches*) in two images



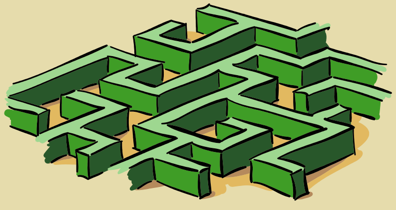
Find transformation T that minimizes

$$\sum_i \text{residual}(T(x_i), x_i')$$

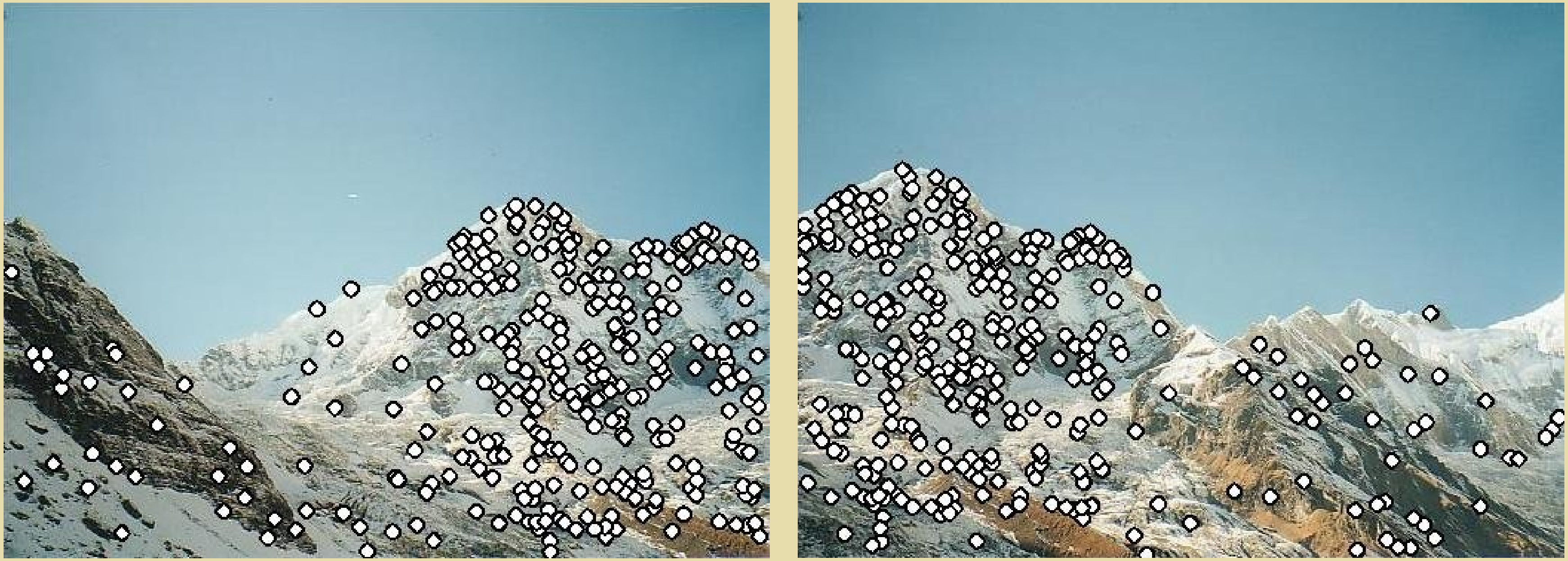


Feature-based alignment outline

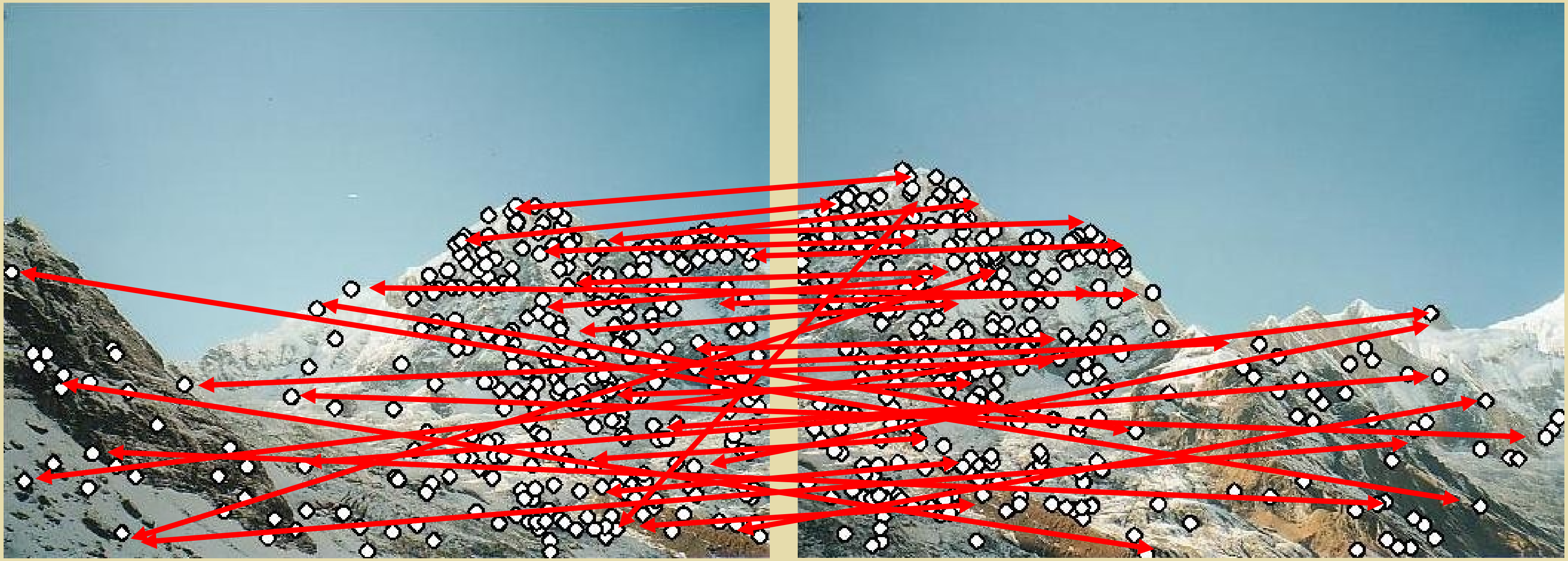


Feature-based alignment outline



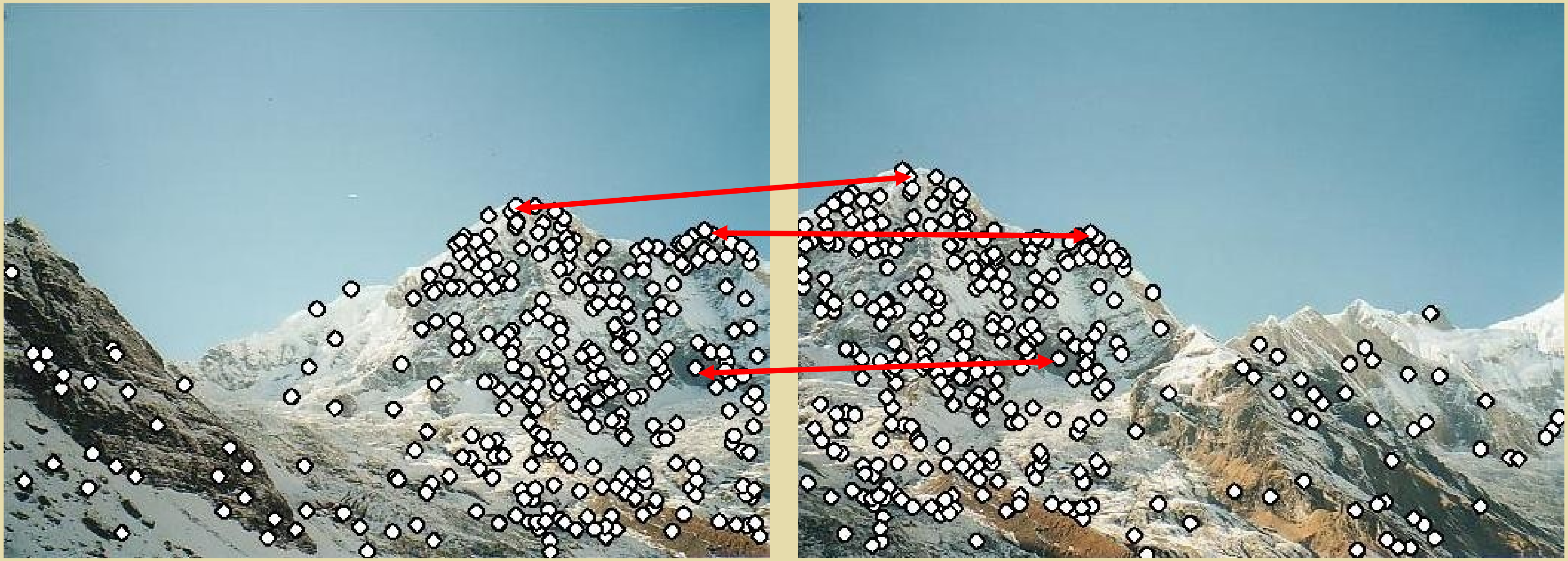
- Extract features

Feature-based alignment outline

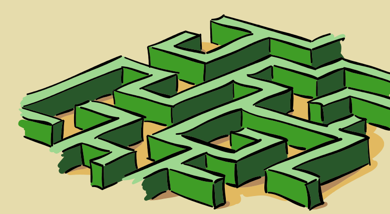


- Extract features
- Compute *matches*

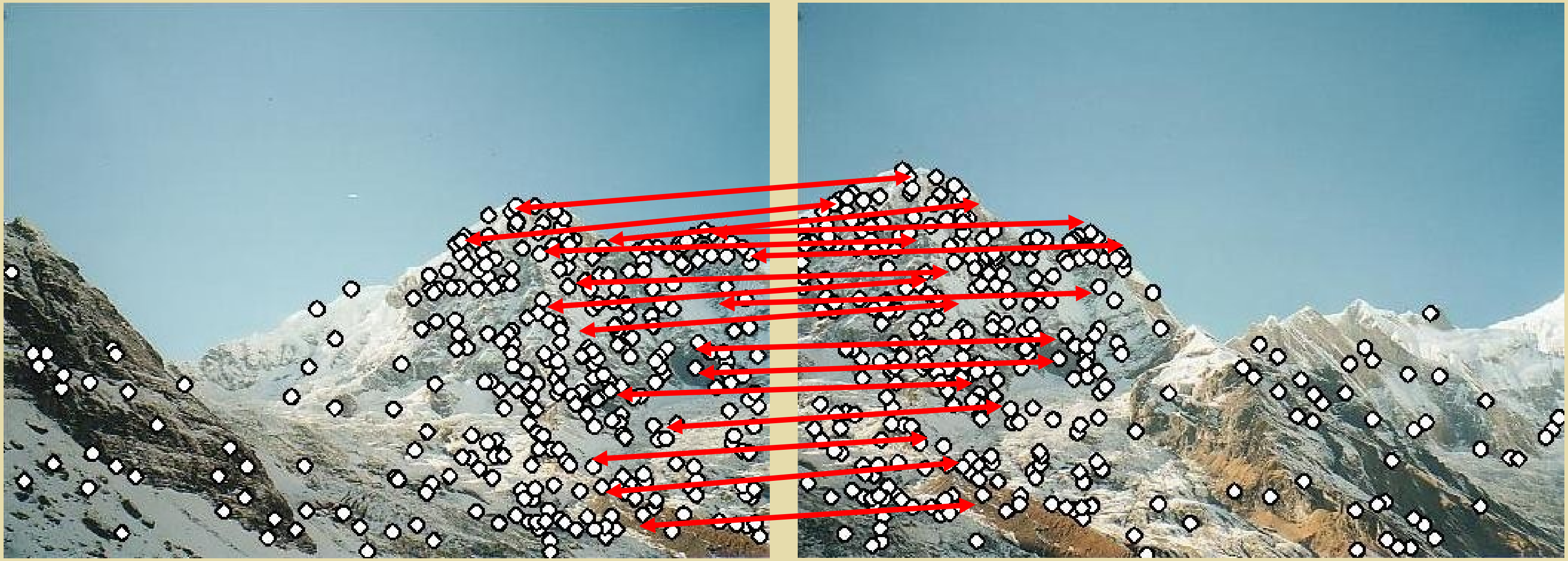
Feature-based alignment outline



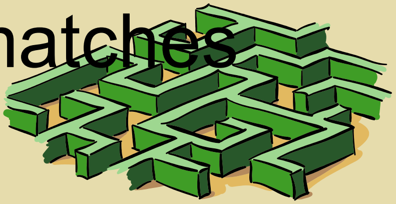
- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)



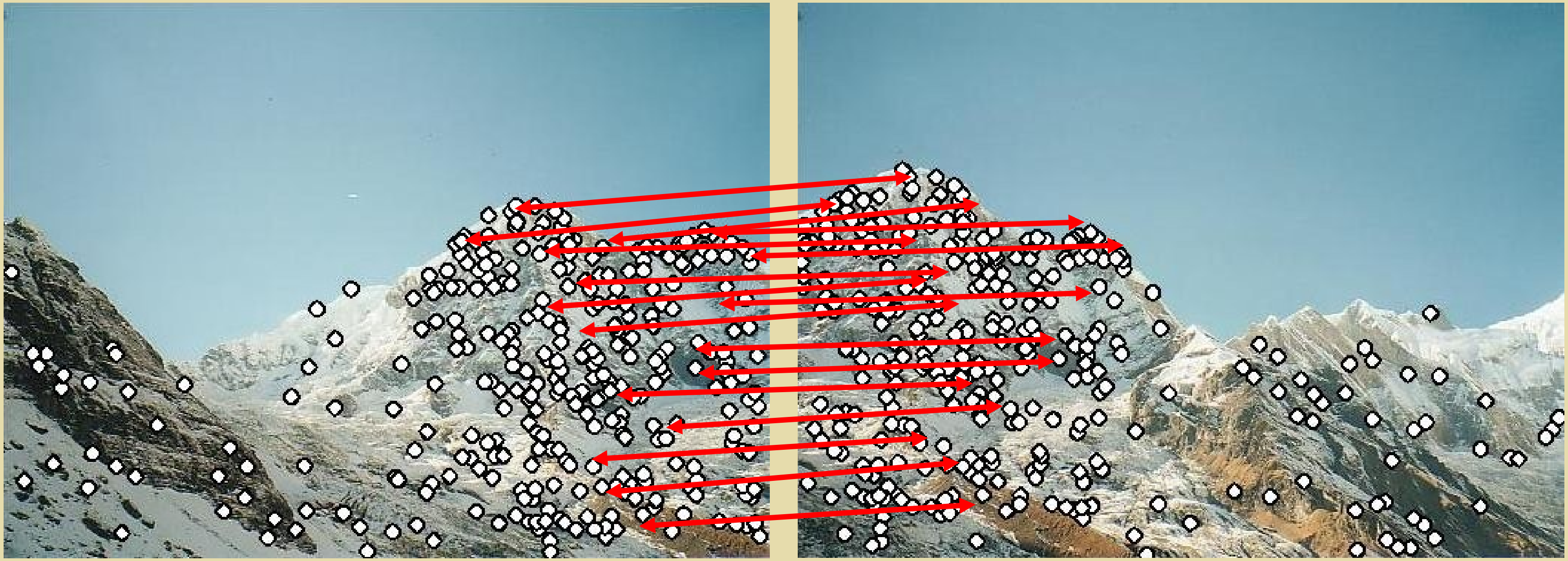
Feature-based alignment outline



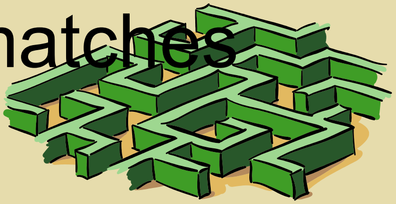
- Extract features
- Compute *putative matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of putative matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)



Feature-based alignment outline

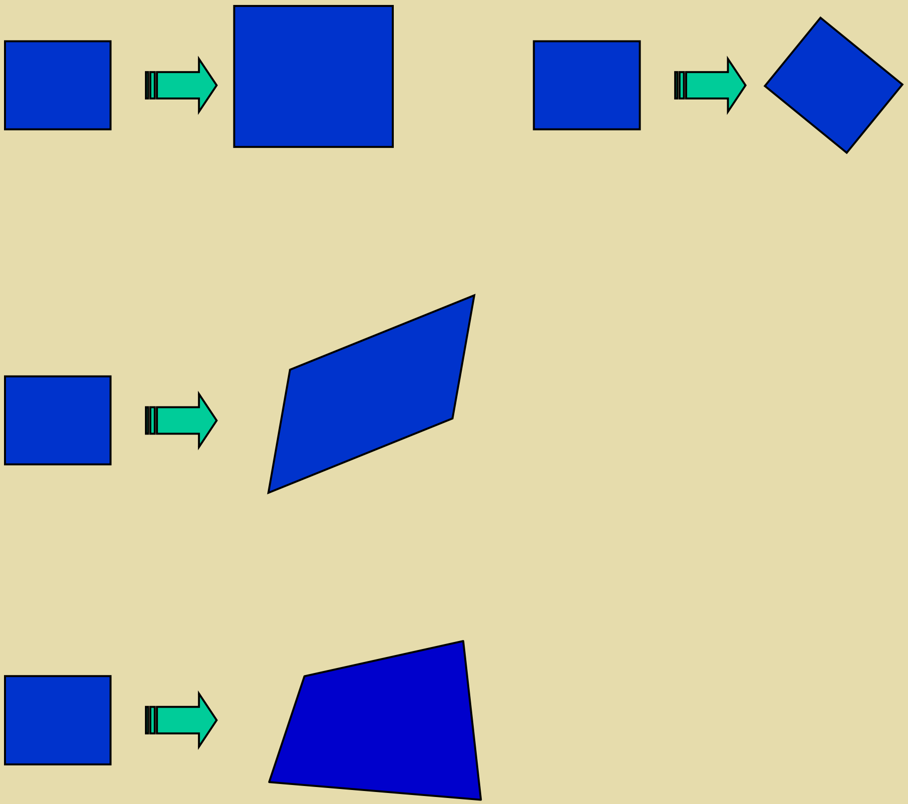


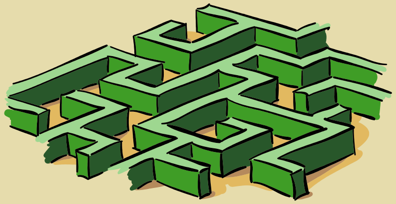
- Extract features
- Compute *matches*
- Loop:
  - *Hypothesize* transformation *T* (small group of matches that are related by *T*)
  - *Verify* transformation (search for other matches consistent with *T*)



2D transformation models

- Similarity (translation, scale, rotation)
- Affine
- Projective (homography)





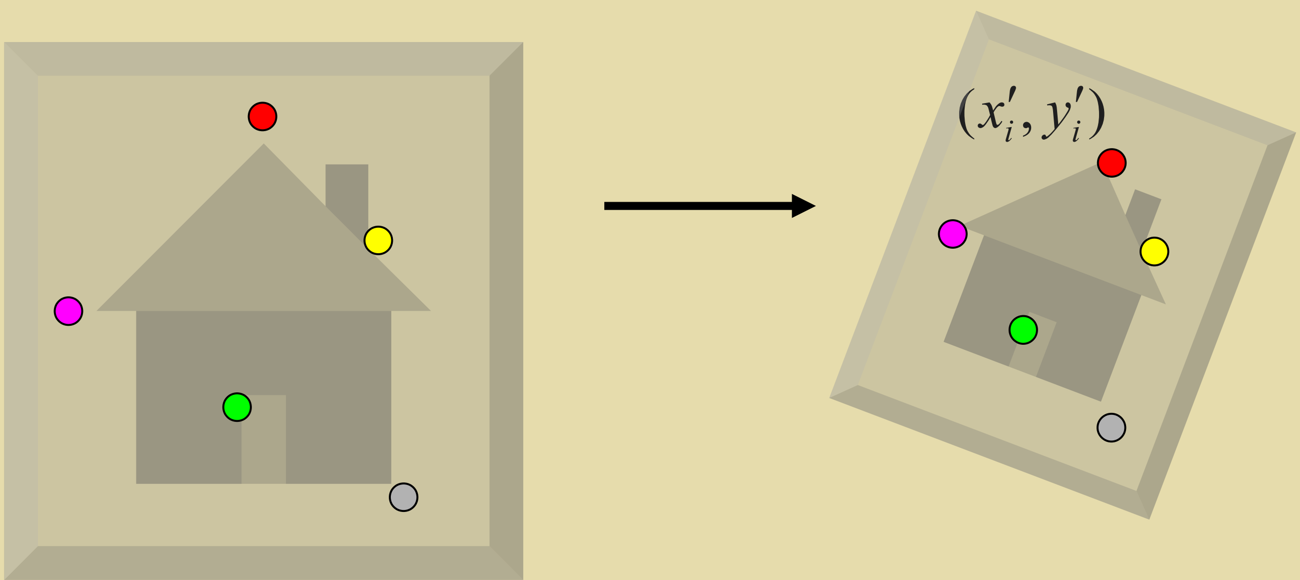
Fitting an affine transformation

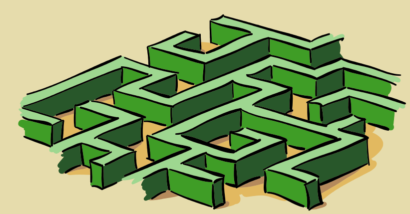
- Assume we know the correspondences, how do we get the transformation?

(x_i, y_i)

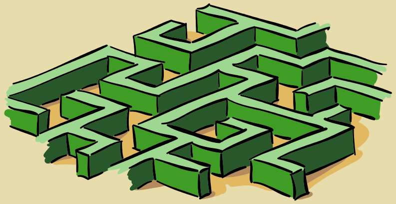
$$\begin{bmatrix} x_i' \\ y_i' \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} t_1 \\ t_2 \end{bmatrix}$$

$$\begin{bmatrix} & & \cdots & & & \\ x_i & y_i & 0 & 0 & 1 & 0 \\ 0 & 0 & x_i & y_i & 0 & 1 \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_1 \\ t_2 \end{bmatrix} = \begin{bmatrix} \cdots \\ x_i' \\ y_i' \\ \cdots \end{bmatrix}$$

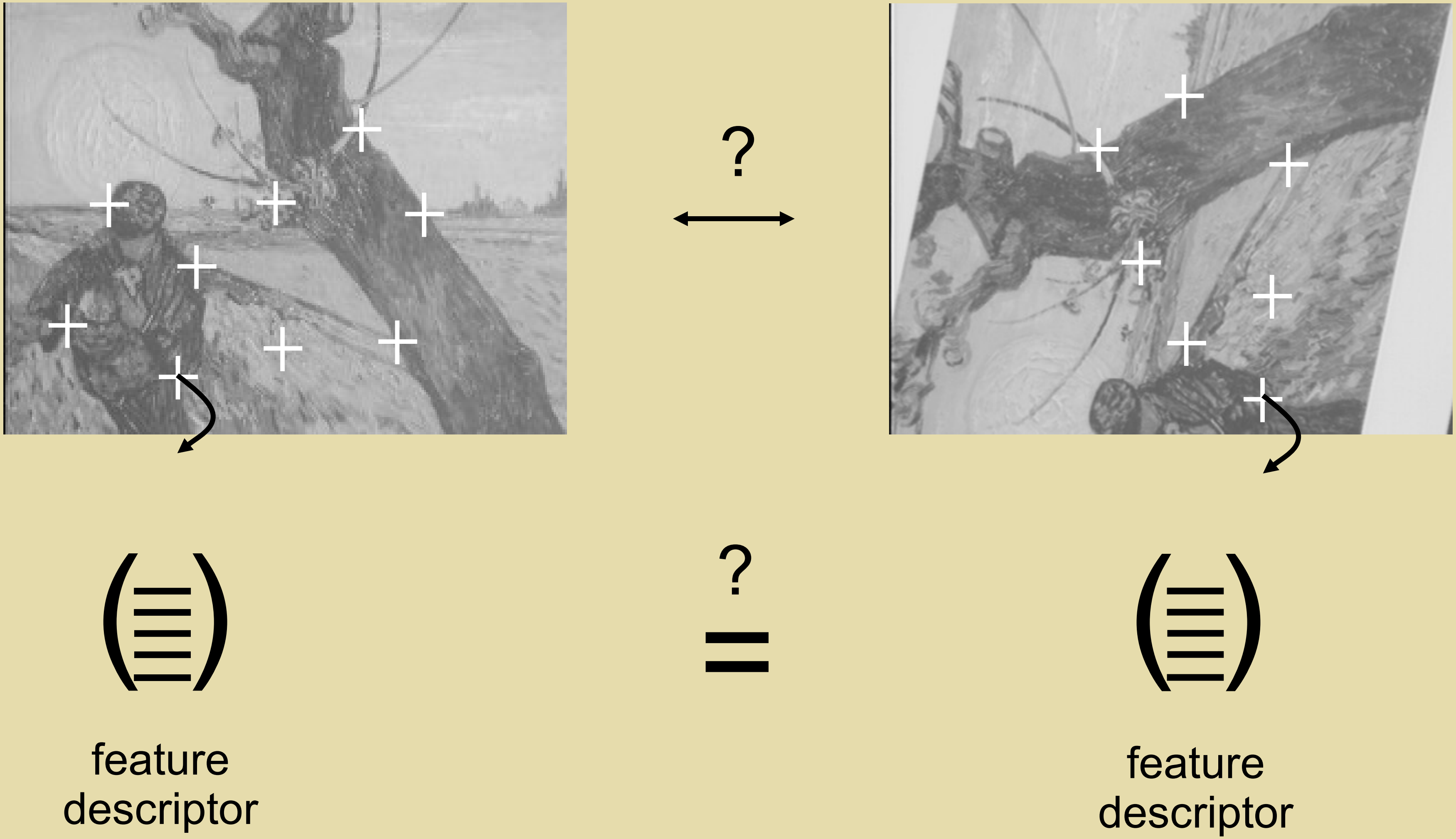




What if we don’t know the correspondences?



What if we don't know the correspondences?



- Need to compare *feature descriptors* of local patches surrounding interest points

Feature descriptors

- Assuming the patches are already normalized (i.e., the local effect of the geometric transformation is factored out), how do we compute their similarity?
- Want invariance to intensity changes, noise, perceptually insignificant changes of the pixel pattern

Feature descriptors

- Simplest descriptor: vector of raw intensity values
- How to compare two such vectors?
  - Sum of squared differences (SSD)

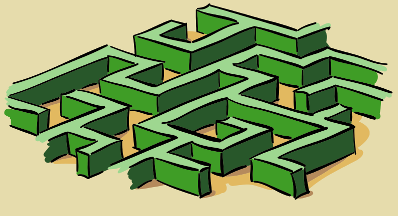
$$\mathrm{SSD}(u,v) = \sum_i \left(u_i - v_i\right)^2$$

Not invariant to intensity change

Normalized correlation

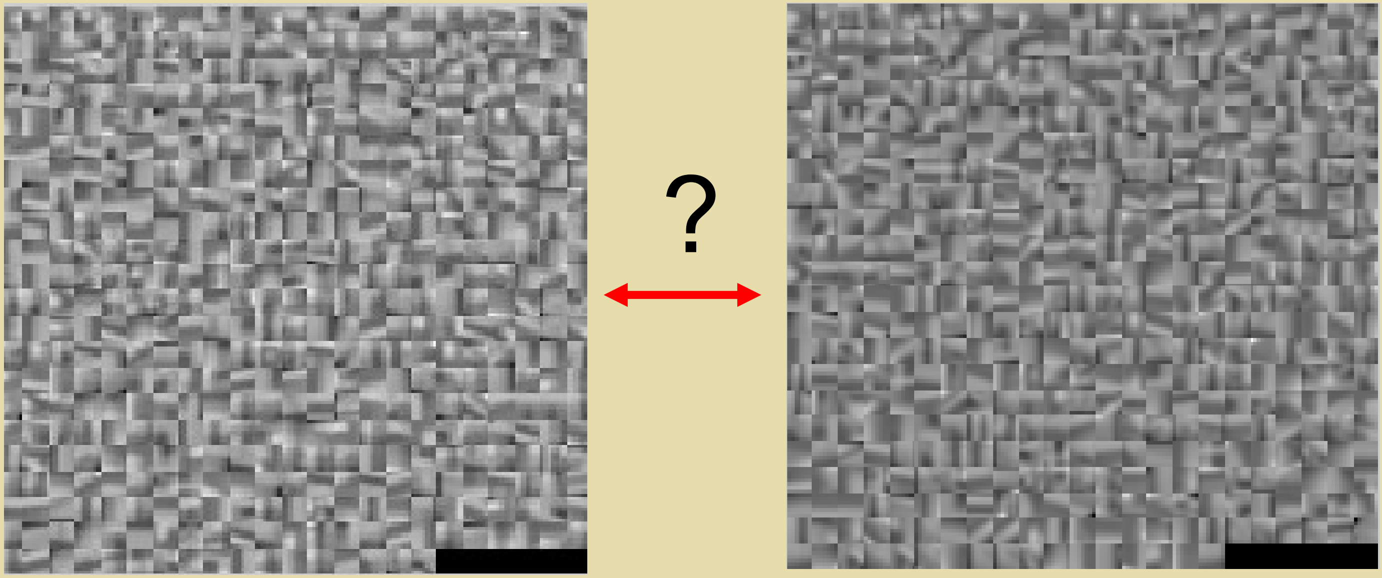
$$\rho(u,v) = \frac{\sum_i (u_i - \bar{u})(v_i - \bar{v})}{\sqrt{\left(\sum_j (u_j - \bar{u})^2\right)\left(\sum_j (v_j - \bar{v})^2\right)}}$$

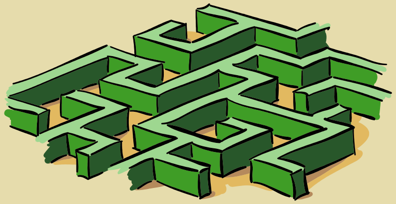
Invariant to affine intensity change



Feature matching

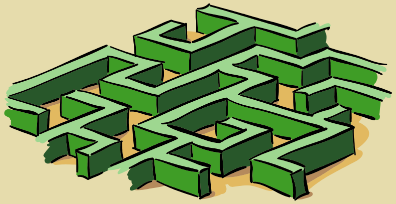
- Generating *putative matches*: for each patch in one image, find a short list of patches in the other image that could match it based solely on appearance





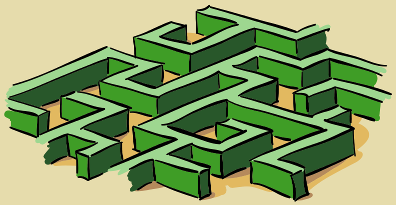
Feature matching

- Generating *putative matches*: for each patch in one image, find a short list of patches in the other image that could match it based solely on appearance
  - Exhaustive search
    - For each feature in one image, compute the distance to *all* features in the other image and find the "closest" ones (threshold or fixed number of top matches)
  - Fast approximate nearest neighbor search
    - Hierarchical spatial data structures (kd-trees, vocabulary trees)
    - Hashing



Dealing with outliers

- The set of putative matches contains a very high percentage of outliers
- Heuristics for feature-space outlier rejection
- Geometric fitting strategies:
  - RANSAC
  - Incremental alignment
  - Hough transform
  - Hashing

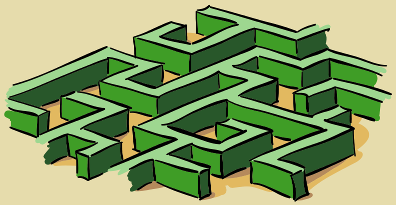


Strategy 1: RANSAC

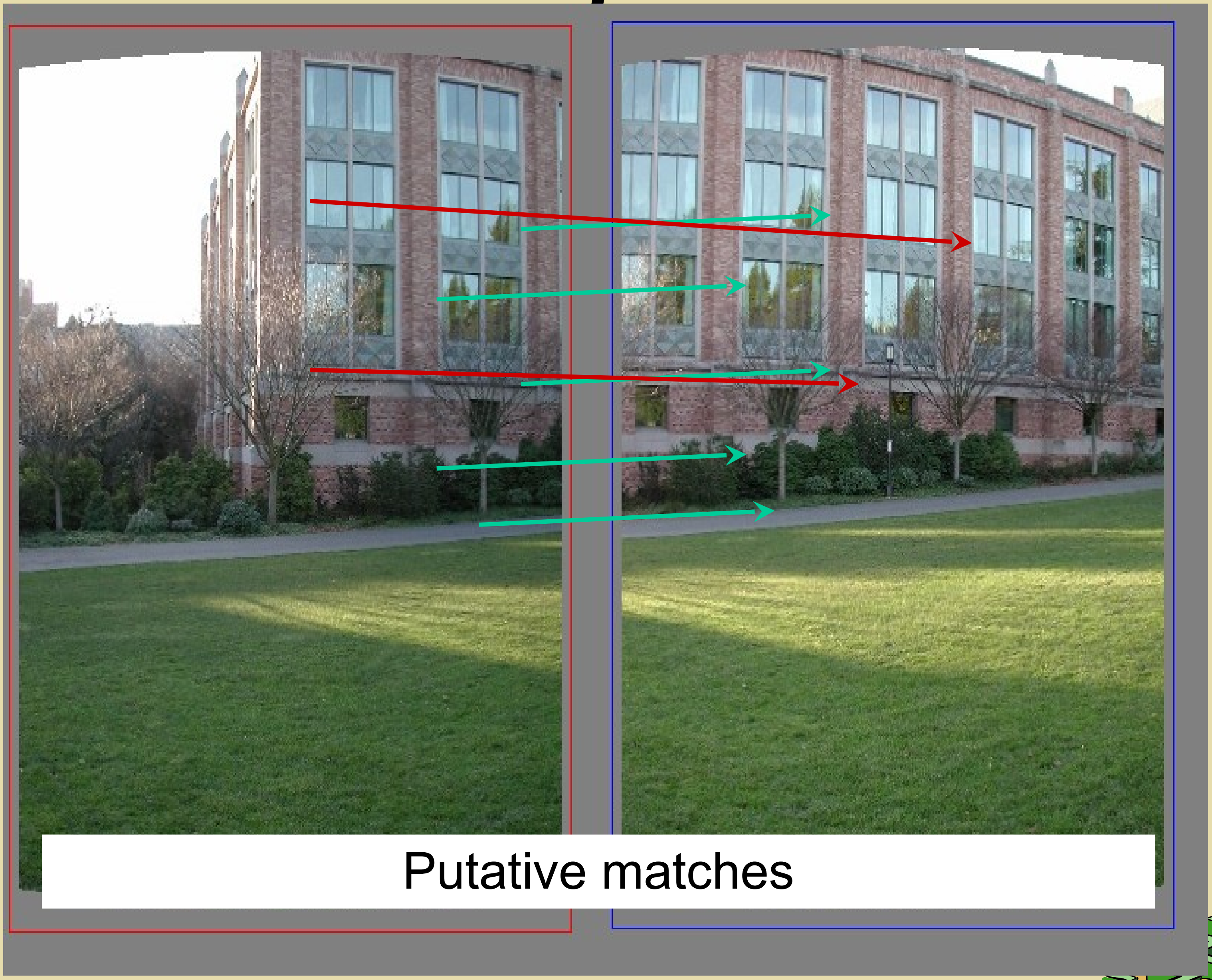
- RANSAC loop:

1. Randomly select a *seed group* of matches
2. Compute transformation from seed group
3. Find *inliers* to this transformation
4. If the number of inliers is sufficiently large, re-compute least-squares estimate of transformation on all of the inliers

- Keep the transformation with the largest number of inliers

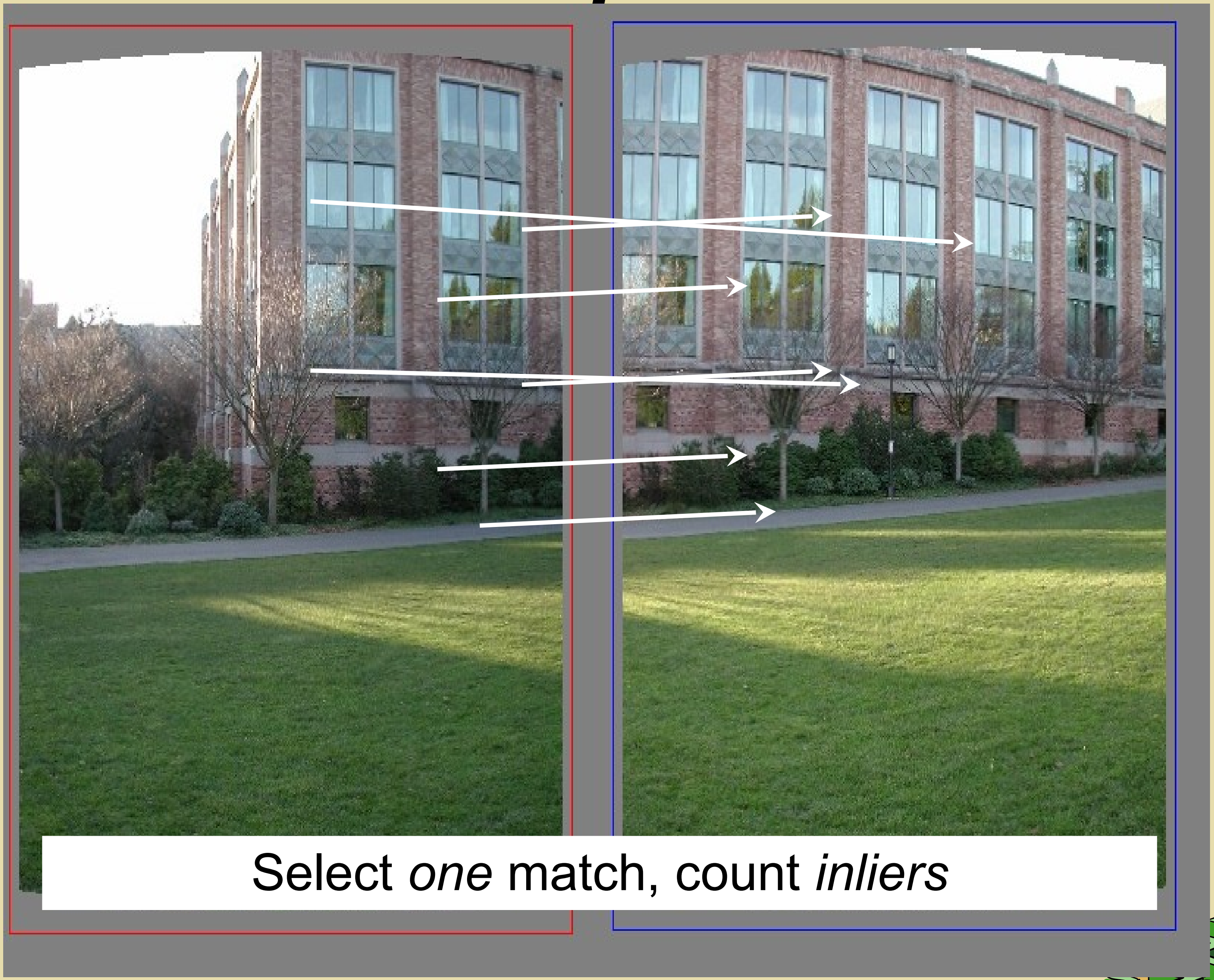


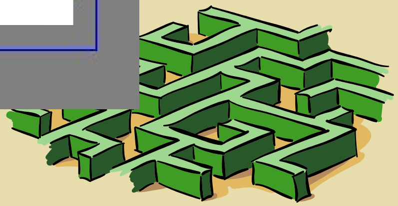
RANSAC example: Translation



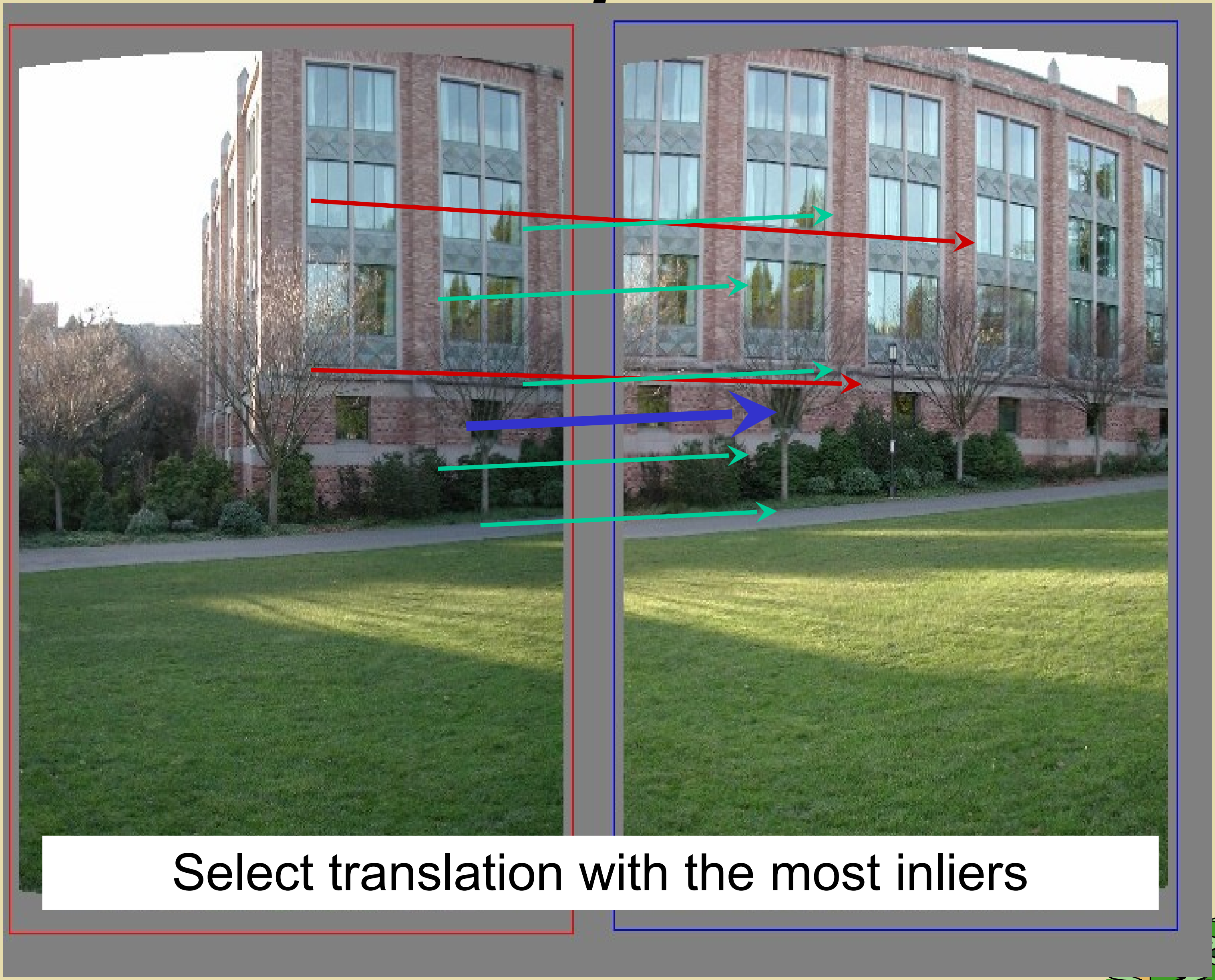
Putative matches

RANSAC example: Translation



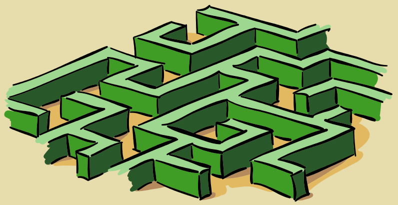


RANSAC example: Translation



Problem with RANSAC

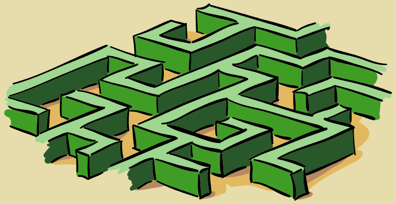
- In many practical situations, the percentage of outliers (incorrect putative matches) is often very high (90% or above)
- Alternative strategy: restrict search space by using strong locality constraints on seed groups and inliers
  - Incremental alignment



Strategy 2: Incremental alignment

- Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

S. Lazebnik, C. Schmid and J. Ponce, "Semi-local affine parts for object recognition," BMVC 2004.



Strategy 2: Incremental alignment

- Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

Strategy 2: Incremental alignment

- Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

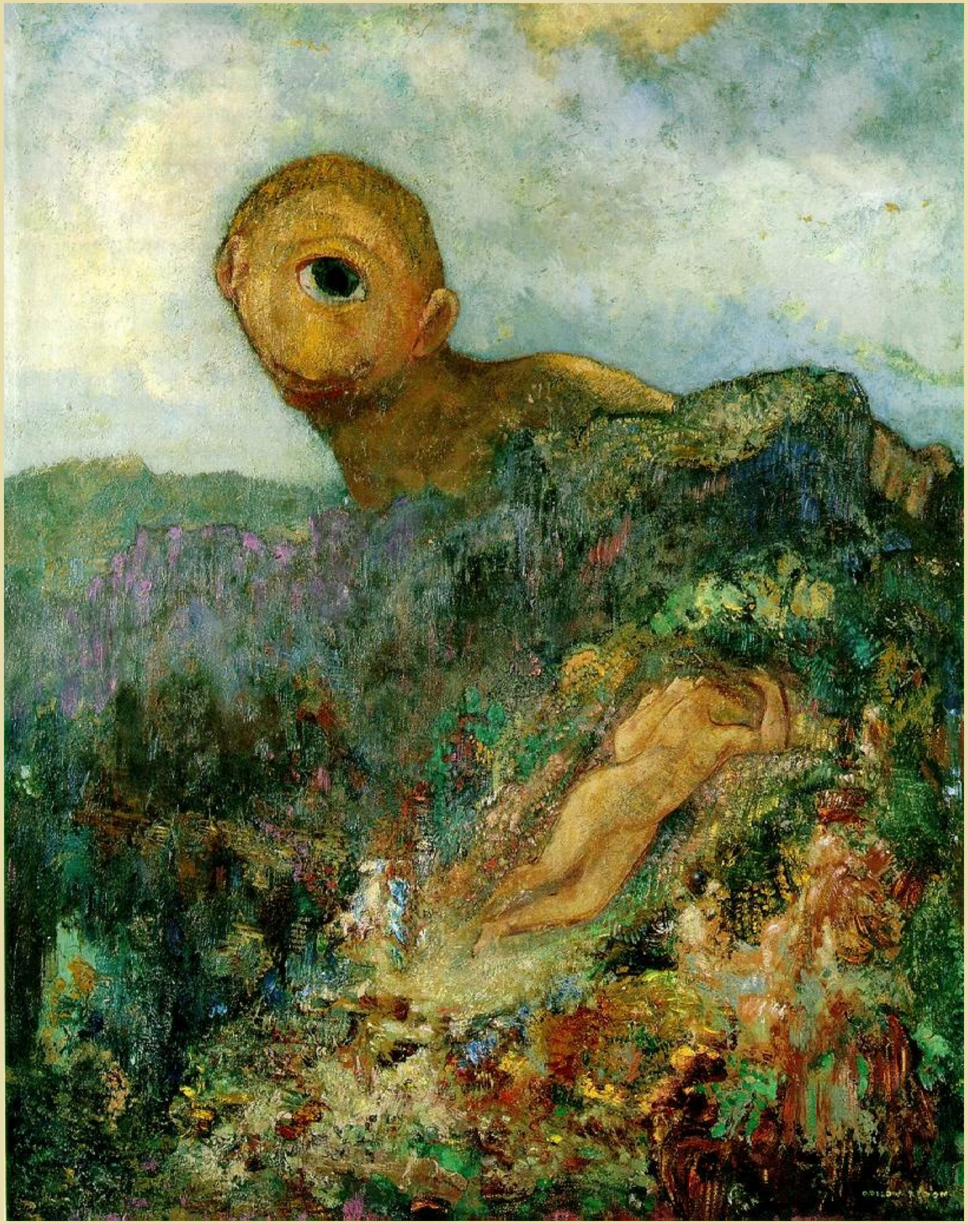
Strategy 2: Incremental alignment

- Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

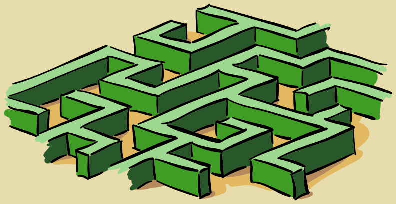
Strategy 2: Incremental alignment

- Take advantage of strong locality constraints: only pick close-by matches to start with, and gradually add more matches in the same neighborhood

Γεωμετρική ανακατασκευή χώρου

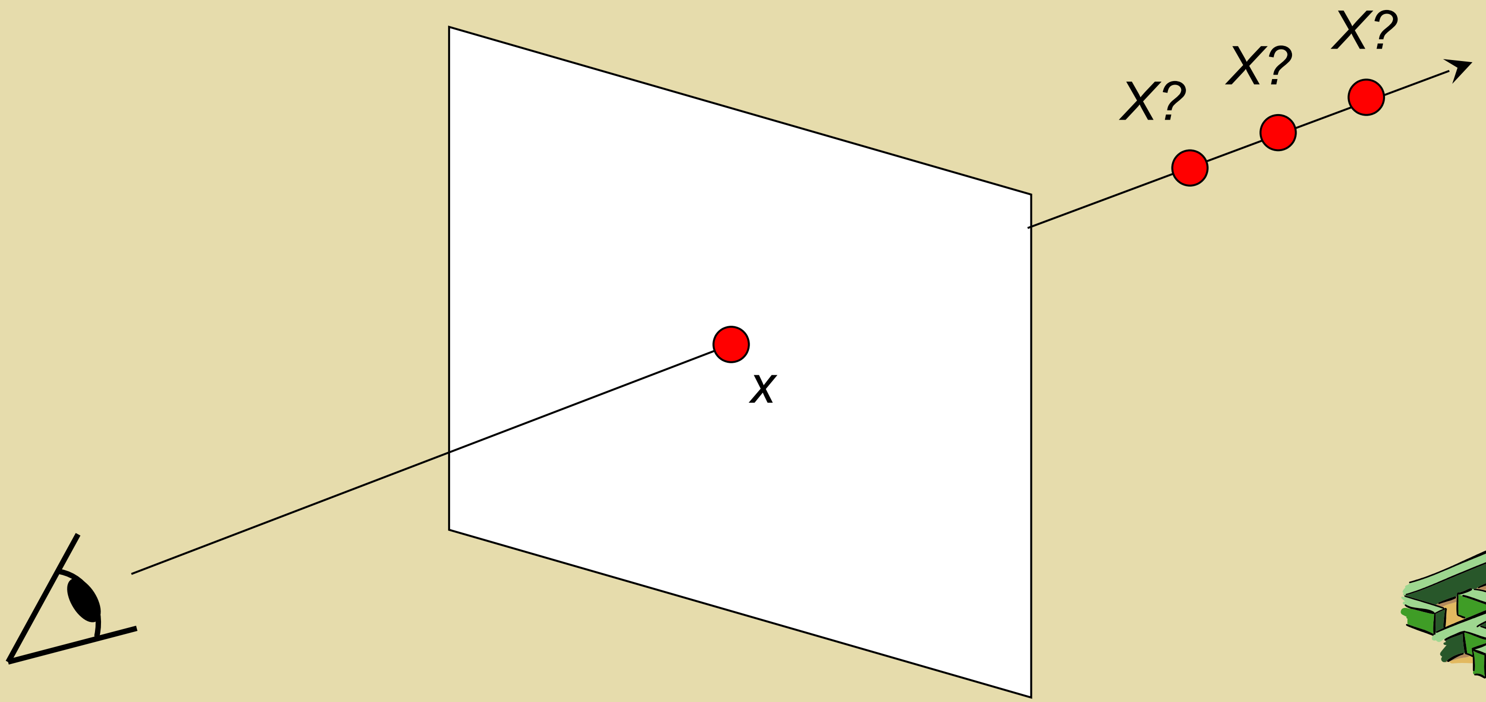


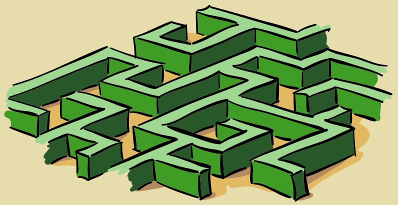
Odilon Redon, Cyclops, 1914



Recovery of 3D structure

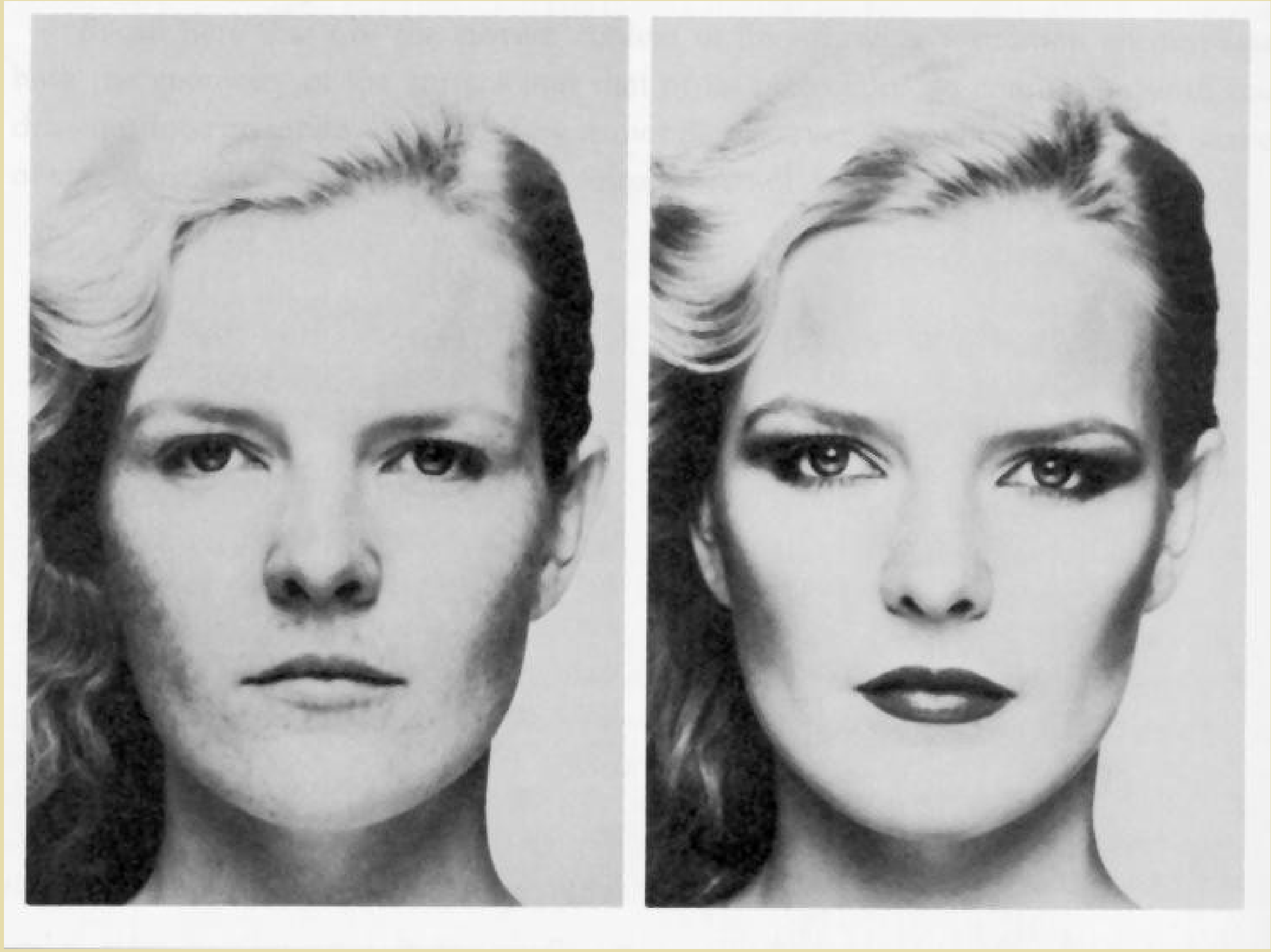
- We will focus on perspective and motion
- We need *multi-view geometry* because recovery of structure from one image is inherently ambiguous





Visual cues

- Shading



Merle Norman Cosmetics, Los Angeles

Slide credit: S. Seitz

Visual cues

- Focus

From *The Art of Photography*, Canon

Slide credit: S. Seitz

Visual cues

- Perspective

Slide credit: S. Seitz

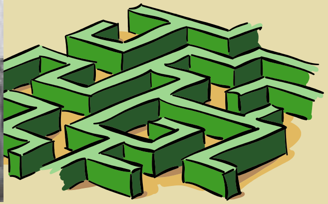
Visual cues

- Motion

Slide credit: S. Seitz

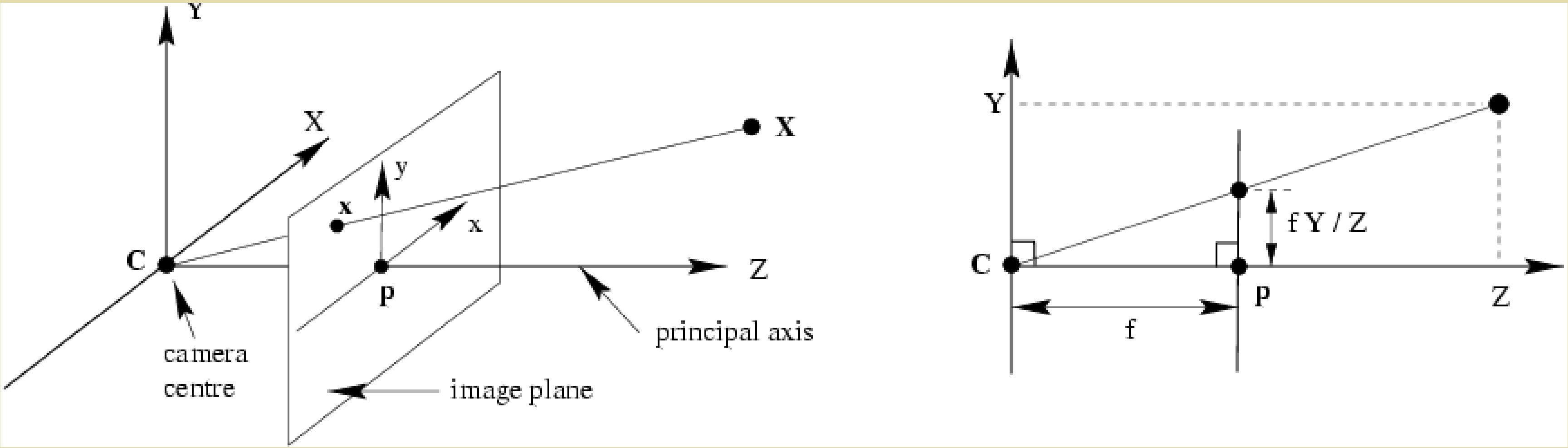
Recovery of 3D structure

- We will focus on perspective and motion
- We need *multi-view geometry* because recovery of structure from one image is inherently ambiguous



Recovery of 3D structure

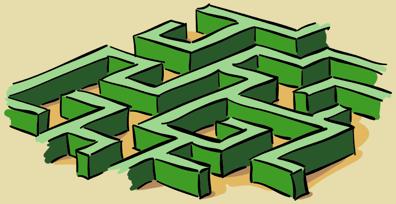
Pinhole camera model



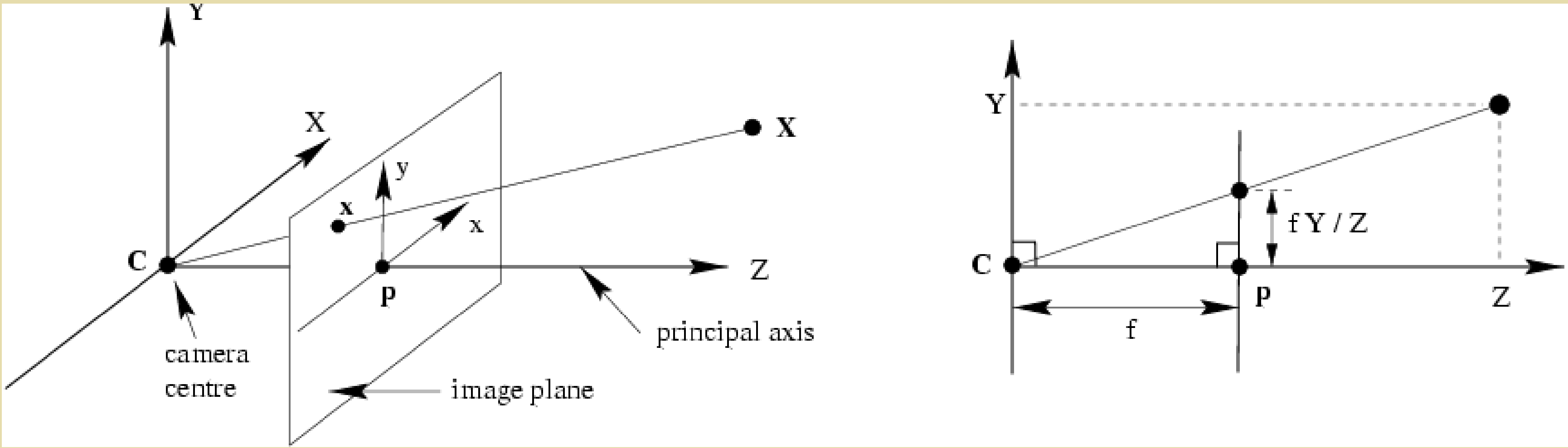
$$(X, Y, Z) \mapsto (fX/Z, fY/Z)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & & 0 \\ & f & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$\mathrm{x} = \mathrm{PX}$$

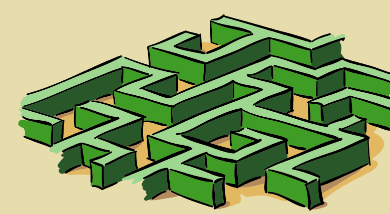


Pinhole camera model

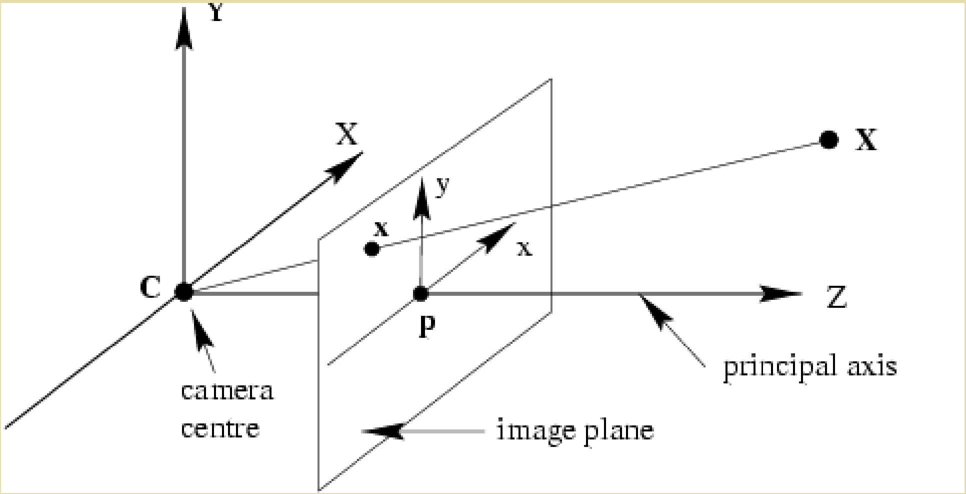


$$\begin{pmatrix} fX \\ fY \\ Z \end{pmatrix} = \begin{bmatrix} f & & \\ & f & \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

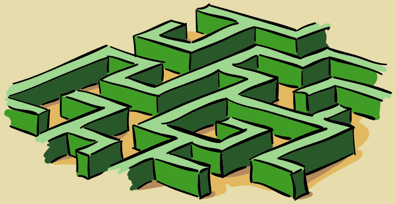
$$\mathrm{x} = \mathrm{PX} \qquad \mathrm{P} = \mathrm{diag}(f, f, 1)\left[\mathrm{I} \mid 0\right]$$



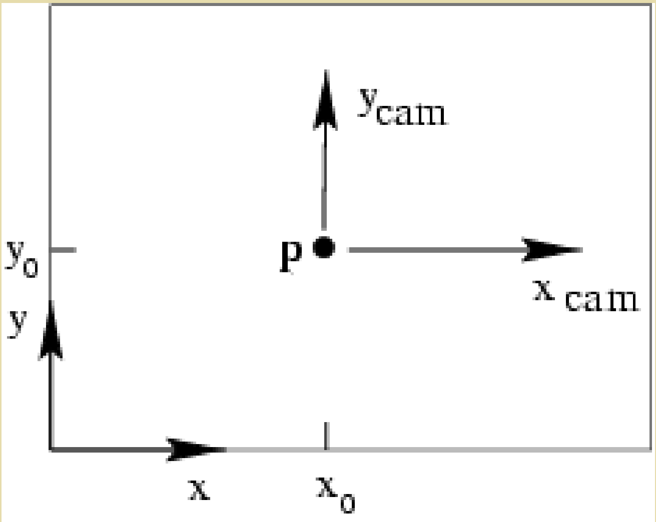
Camera coordinate system



- **Principal axis:** line from the camera center perpendicular to the image plane
- **Normalized (camera) coordinate system:** camera center is at the origin and the principal axis is the z-axis
- **Principal point (p):** point where principal axis intersects the image plane (origin of normalized coordinate system)

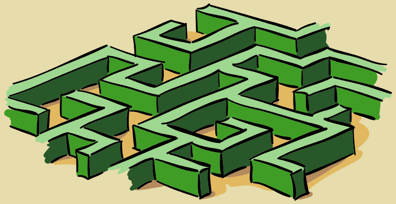


Principal point offset

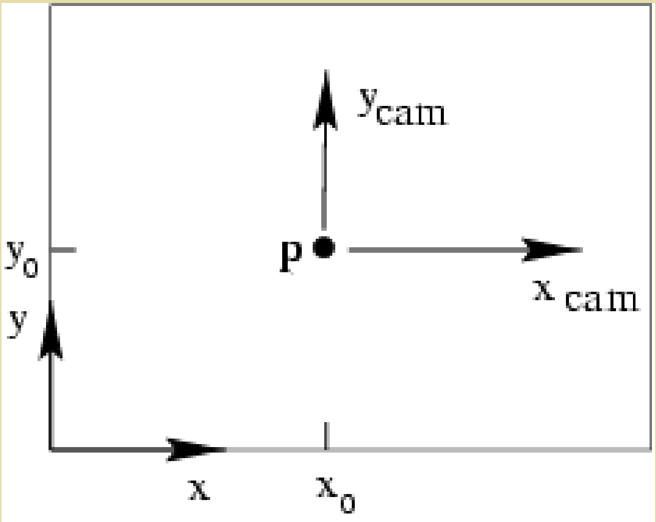


principal point: (p_x, p_y)

- Camera coordinate system: origin is at the prinicipal point
- Image coordinate system: origin is in the corner



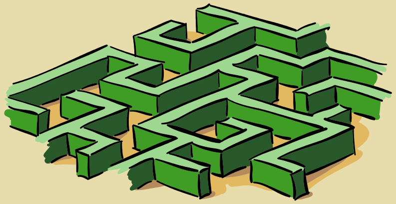
Principal point offset



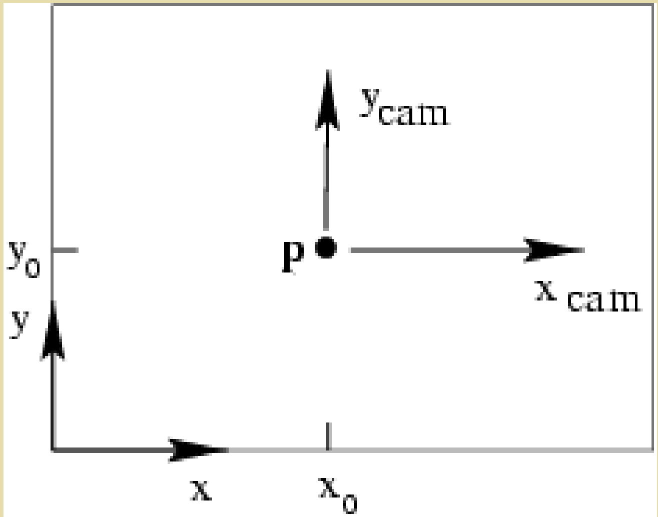
principal point: (p_x, p_y)

$$(X, Y, Z) \mapsto (fX/Z + p_x, fY/Z + p_y)$$

$$\begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x & 0 \\ & f & p_y & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$



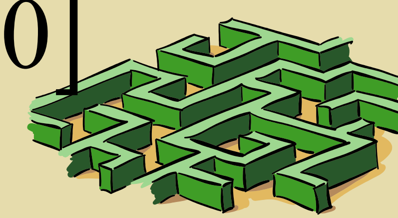
Principal point offset



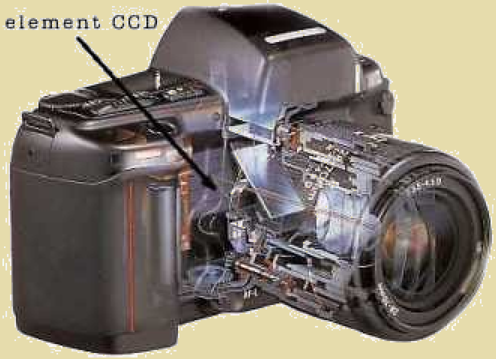
principal point: (p_x, p_y)

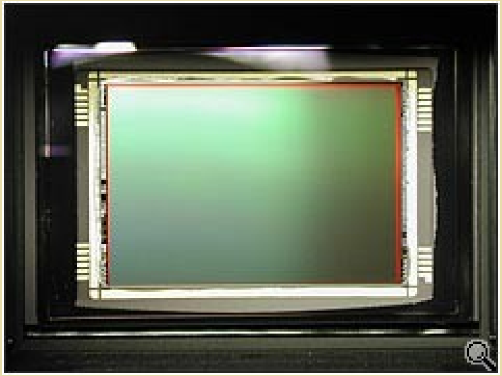
$$\begin{pmatrix} fX + Zp_x \\ fY + Zp_y \\ Z \end{pmatrix} = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & 0 \\ & 1 & & 0 \\ & & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

$$K = \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$ calibration matrix $$\mathrm{P} = \mathrm{K}[\mathrm{I} \mid 0]$$



Pixel coordinates



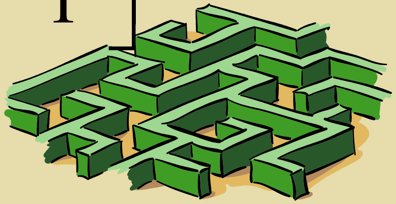


Pixel size: $$\frac{1}{m_x} \times \frac{1}{m_y}$$

- m_x pixels per meter in horizontal direction, m_y pixels per meter in vertical direction

$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}$$

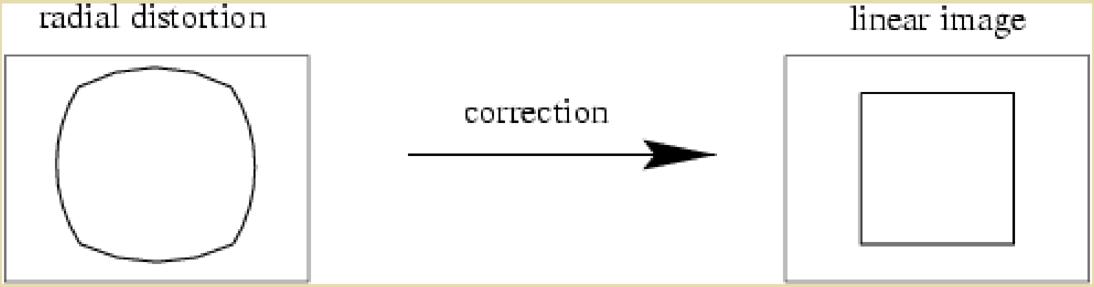
pixels/m m pixels

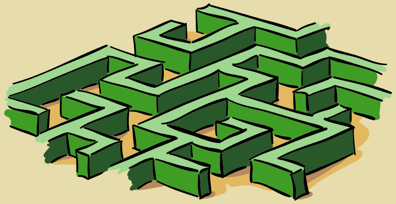


Camera parameters

- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*

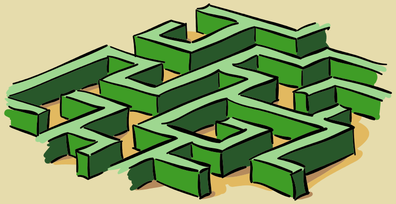
$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix} = \begin{bmatrix} \alpha_x & & \beta_x \\ & \alpha_y & \beta_y \\ & & 1 \end{bmatrix}$$





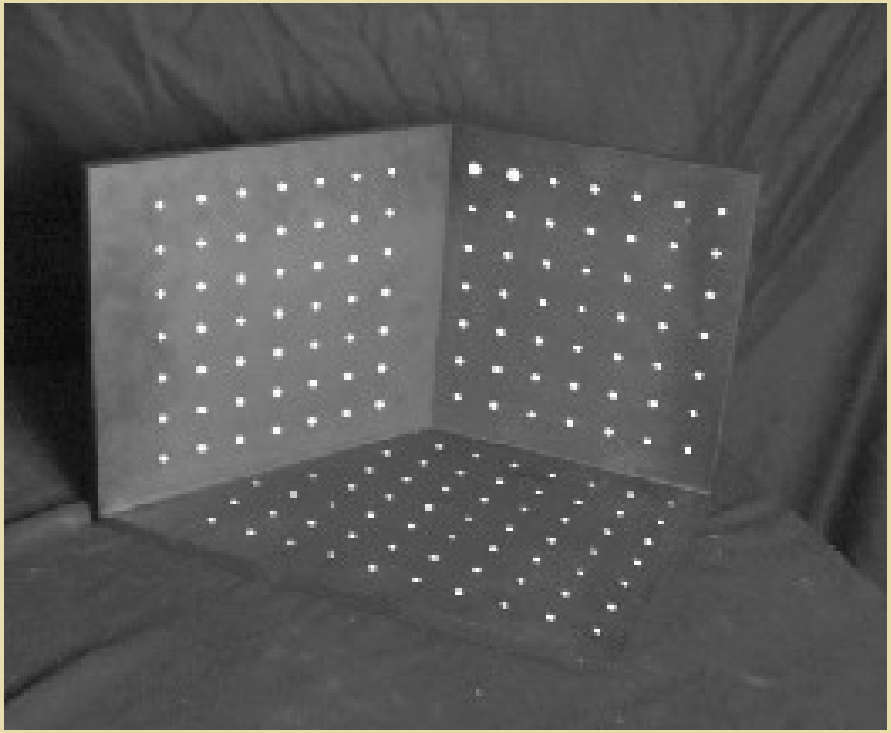
Camera parameters

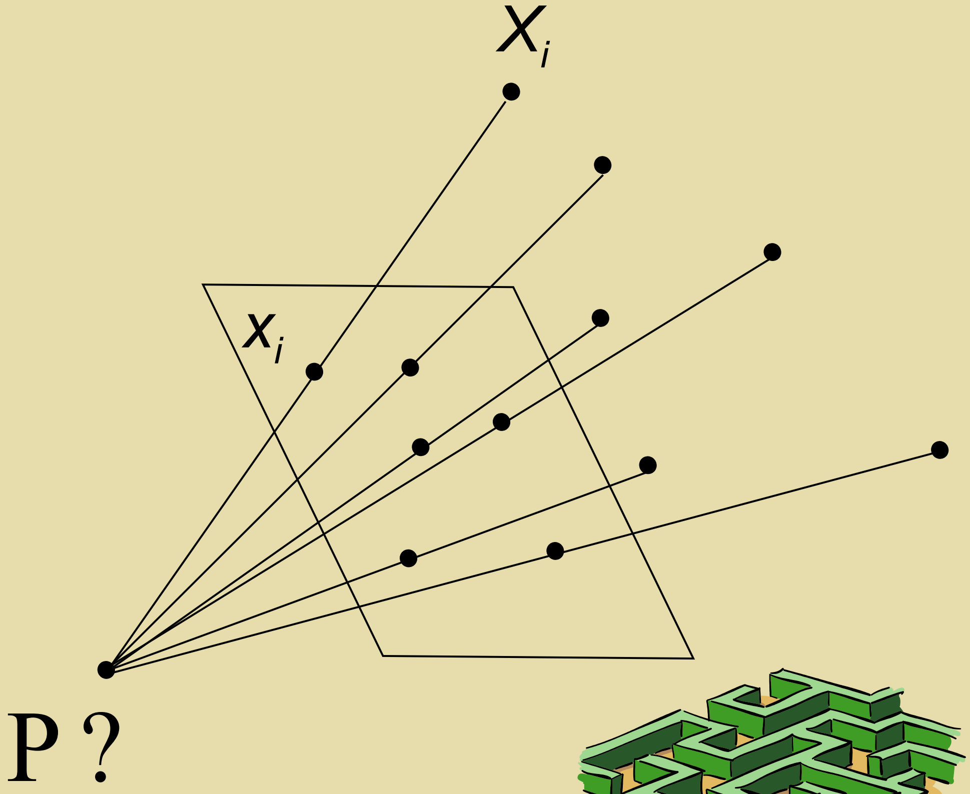
- Intrinsic parameters
  - Principal point coordinates
  - Focal length
  - Pixel magnification factors
  - *Skew (non-rectangular pixels)*
  - *Radial distortion*
- Extrinsic parameters
  - Rotation and translation relative to world coordinate system



Camera calibration

- Given n points with known 3D coordinates X_i and known image projections x_i, estimate the camera parameters



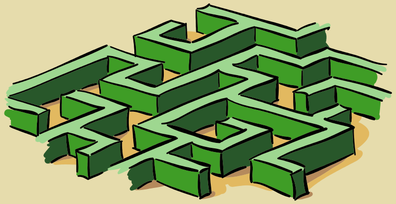


Camera calibration

$$\lambda \mathrm{x}_i = \mathrm{P}\mathrm{X}_i \qquad \lambda \begin{bmatrix} x_i \\ y_i \\ 1 \end{bmatrix} = \begin{bmatrix} \mathrm{P}_1^T \\ \mathrm{P}_2^T \\ \mathrm{P}_3^T \end{bmatrix} \mathrm{X}_i \qquad \mathrm{x}_i \times \mathrm{P}\mathrm{X}_i = 0$$

$$\begin{bmatrix} 0 & -\mathrm{X}_i^T & y_i \mathrm{X}_i^T \\ \mathrm{X}_i^T & 0 & -x_i \mathrm{X}_i^T \\ -y_i \mathrm{X}_i^T & x_i \mathrm{X}_i^T & 0 \end{bmatrix} \begin{pmatrix} \mathrm{P}_1 \\ \mathrm{P}_2 \\ \mathrm{P}_3 \end{pmatrix} = 0$$

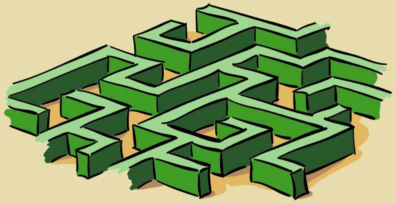
Two linearly independent equations



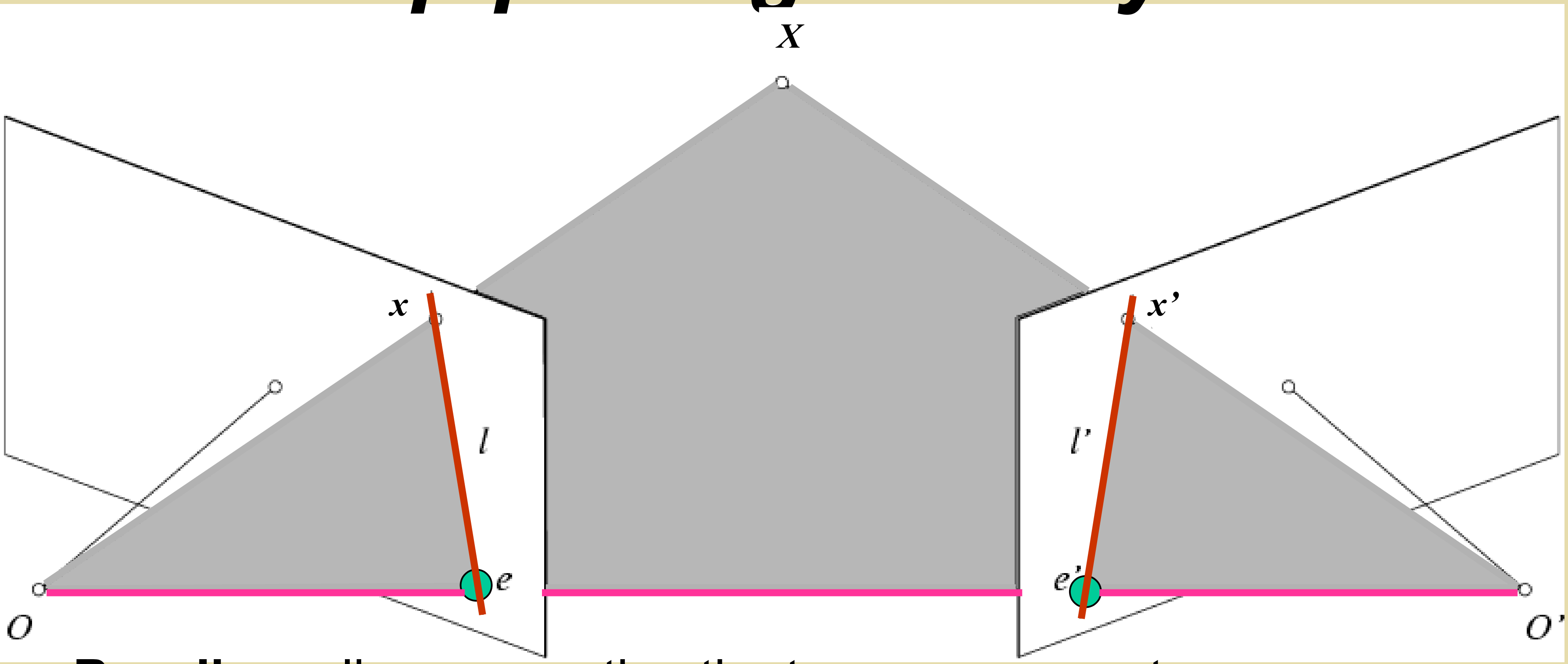
Camera calibration

$$\begin{bmatrix} 0^T & X_1^T & -y_1 X_1^T \\ X_1^T & 0^T & -x_1 X_1^T \\ \cdots & \cdots & \cdots \\ 0^T & X_n^T & -y_n X_n^T \\ X_n^T & 0^T & -x_n X_n^T \end{bmatrix} \begin{pmatrix} P_1 \\ P_2 \\ P_3 \end{pmatrix} = 0$$

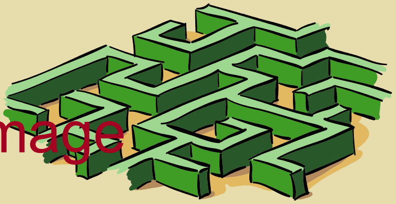
- P has 11 degrees of freedom (12 parameters, but scale is arbitrary)
- 6 correspondences needed for a minimal solution
- Homogeneous least squares



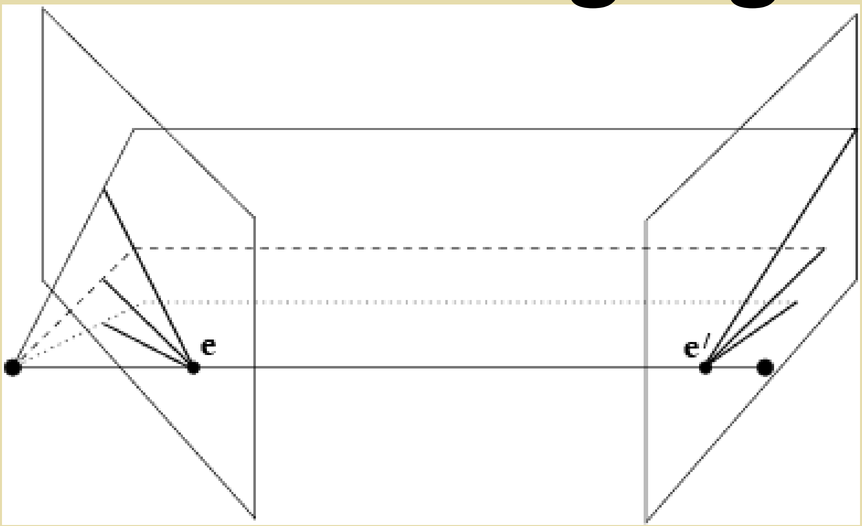
Epipolar geometry

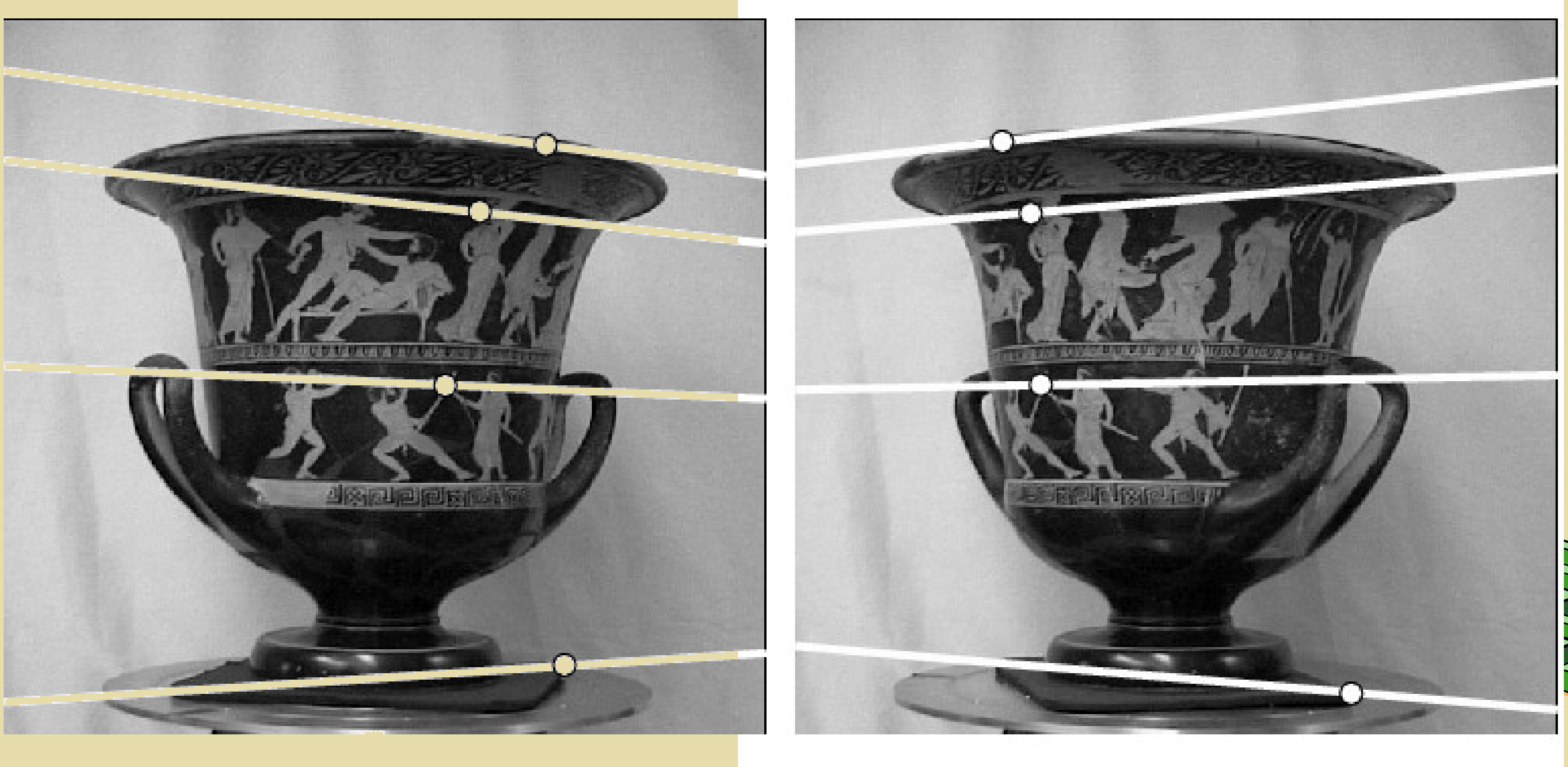


- **Baseline** – line connecting the two camera centers
- **Epipolar Plane** – plane containing baseline (1D family)
- **Epipoles**
  = intersections of baseline with image planes
  = projections of the other camera center
  = vanishing points of camera motion direction
- **Epipolar Lines** - intersections of epipolar plane with image planes (always come in corresponding pairs)

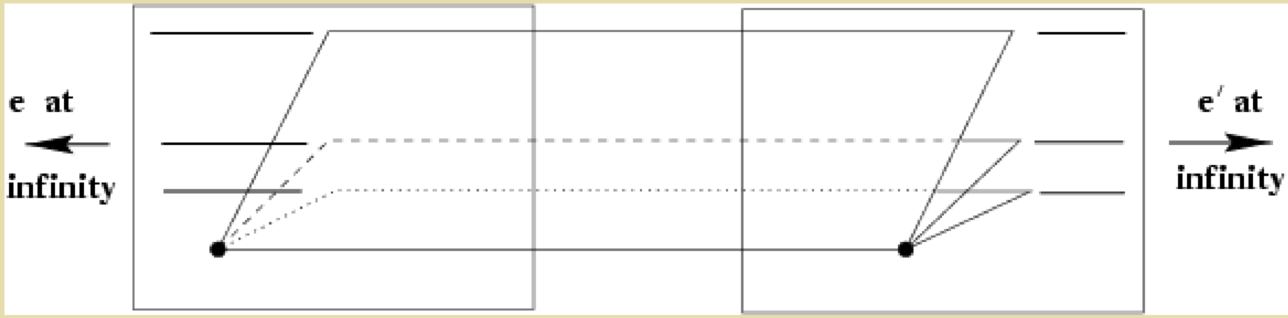


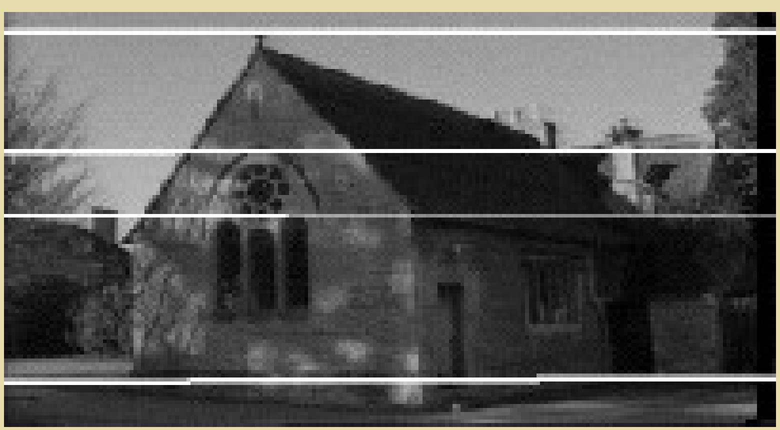
Example: Converging cameras

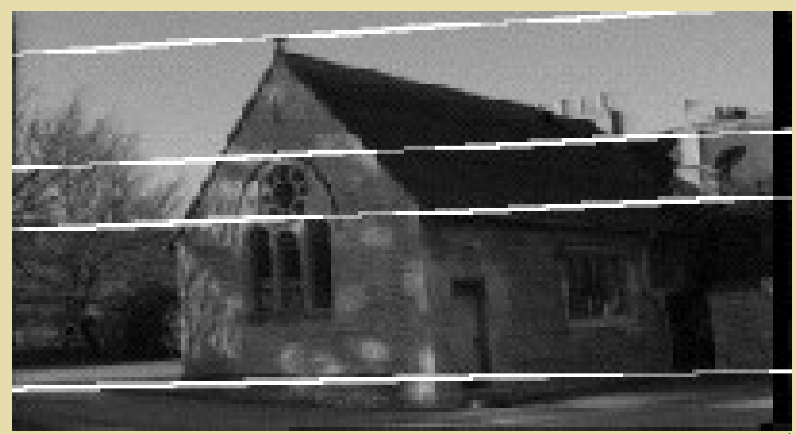




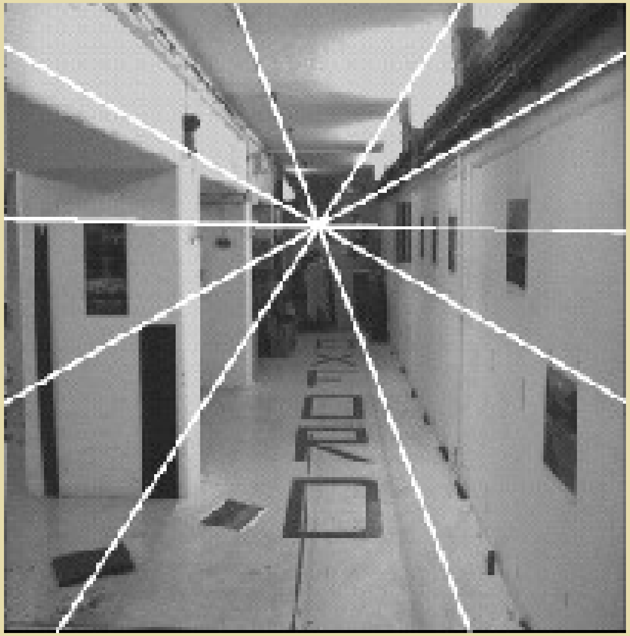
Example: Motion parallel to image plane

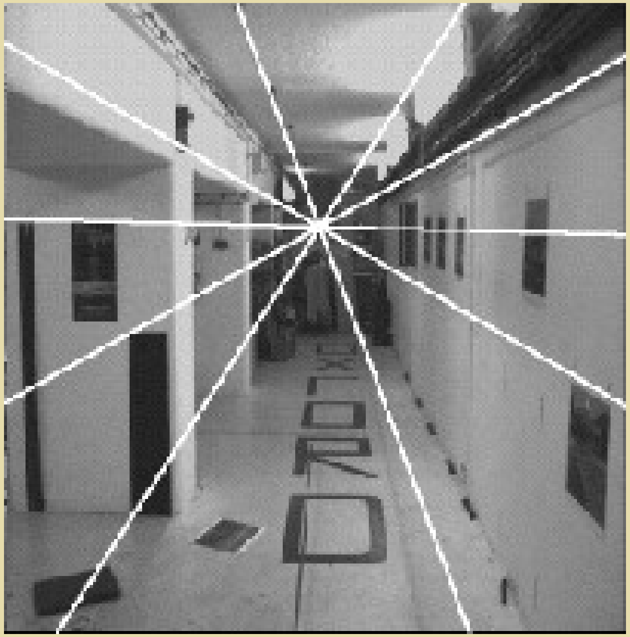


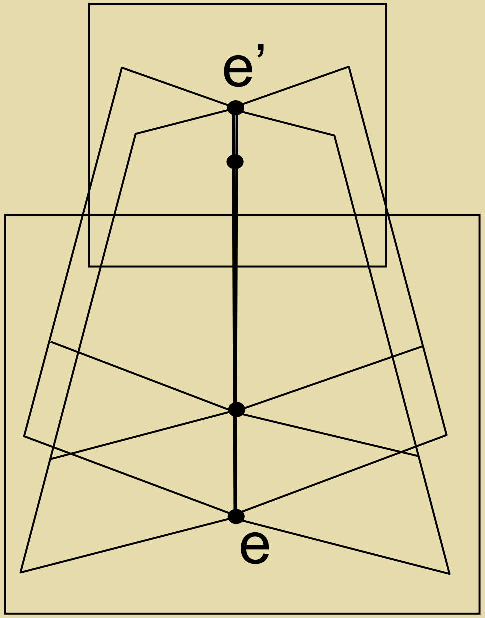




Example: Forward motion

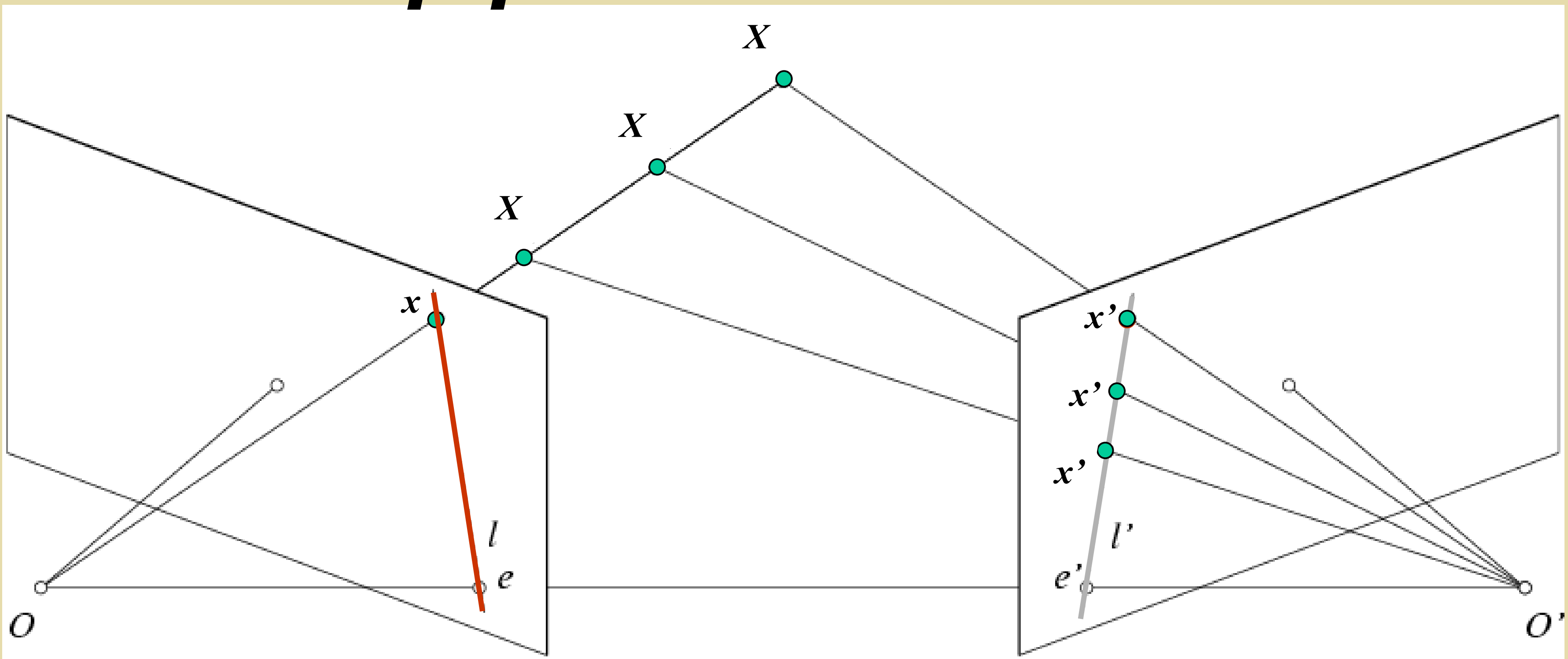






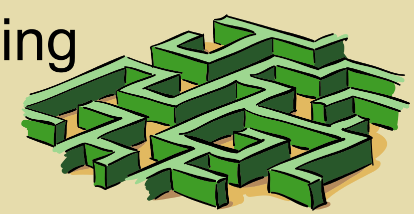
Epipole has same coordinates in both images.
Points move along lines radiating from e: “Focus of expansion”

Epipolar constraint

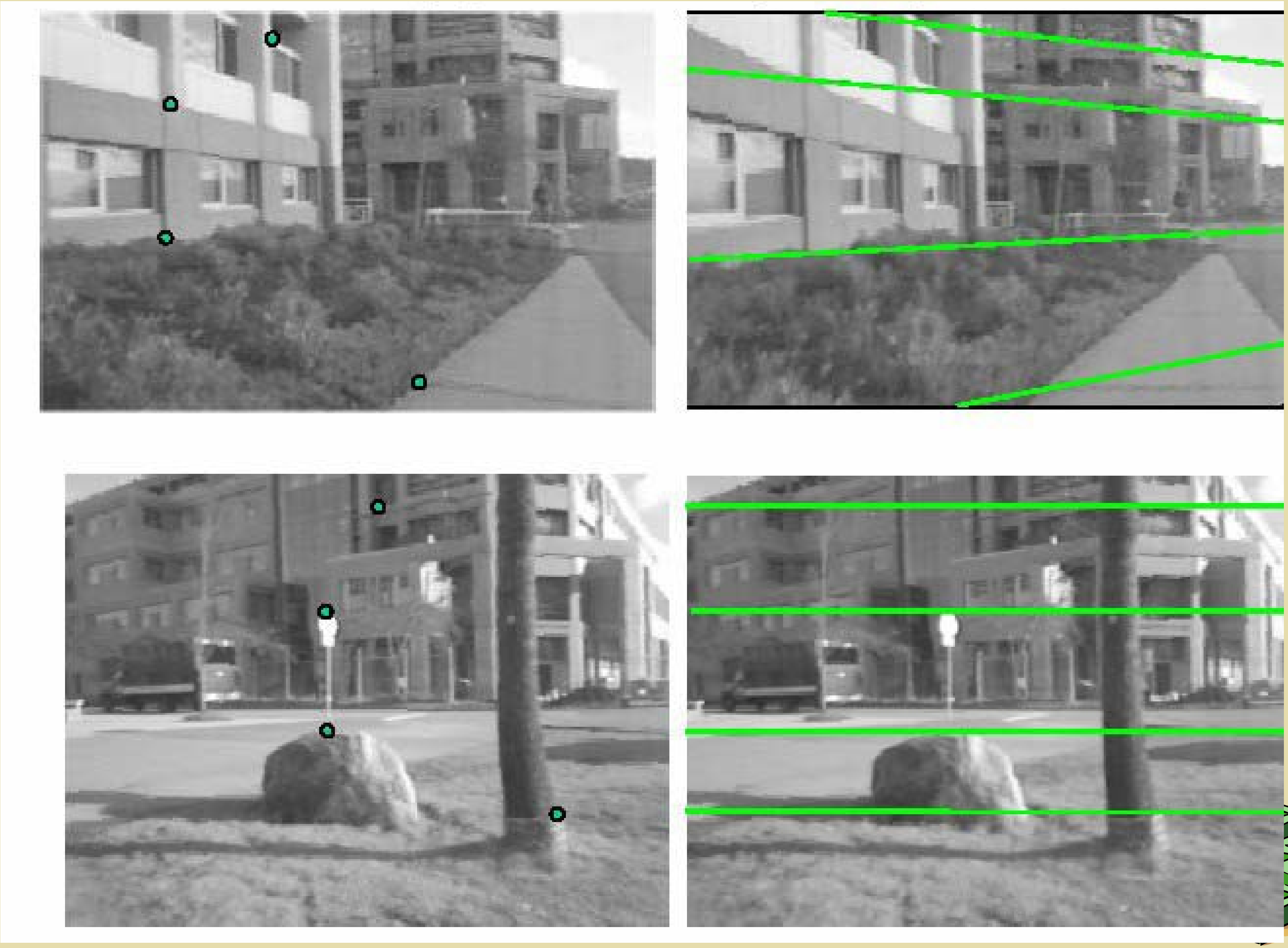


- Potential matches for *x* have to lie on the corresponding epipolar line *l'*.

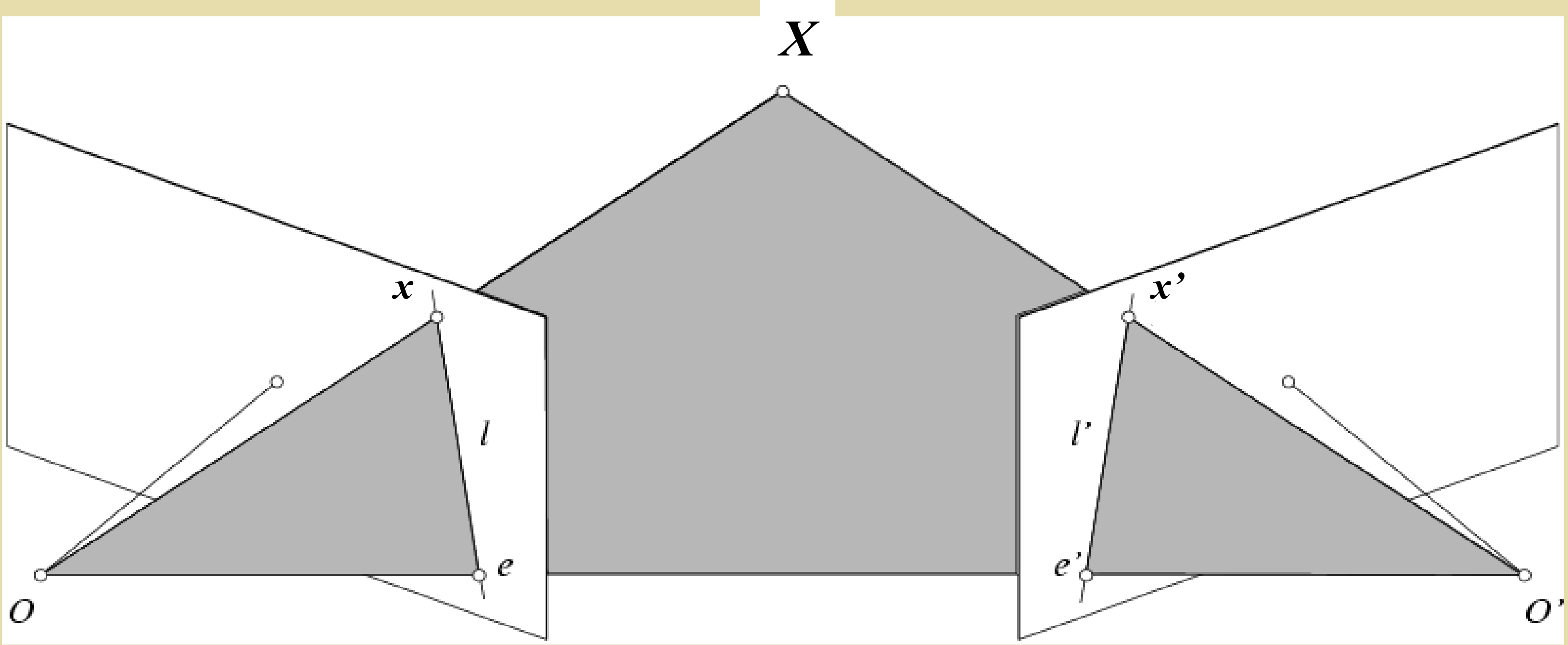
- Potential matches for *x'* have to lie on the corresponding epipolar line *l*.



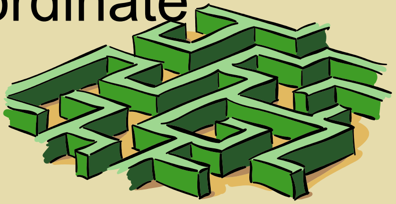
Epipolar constraint example



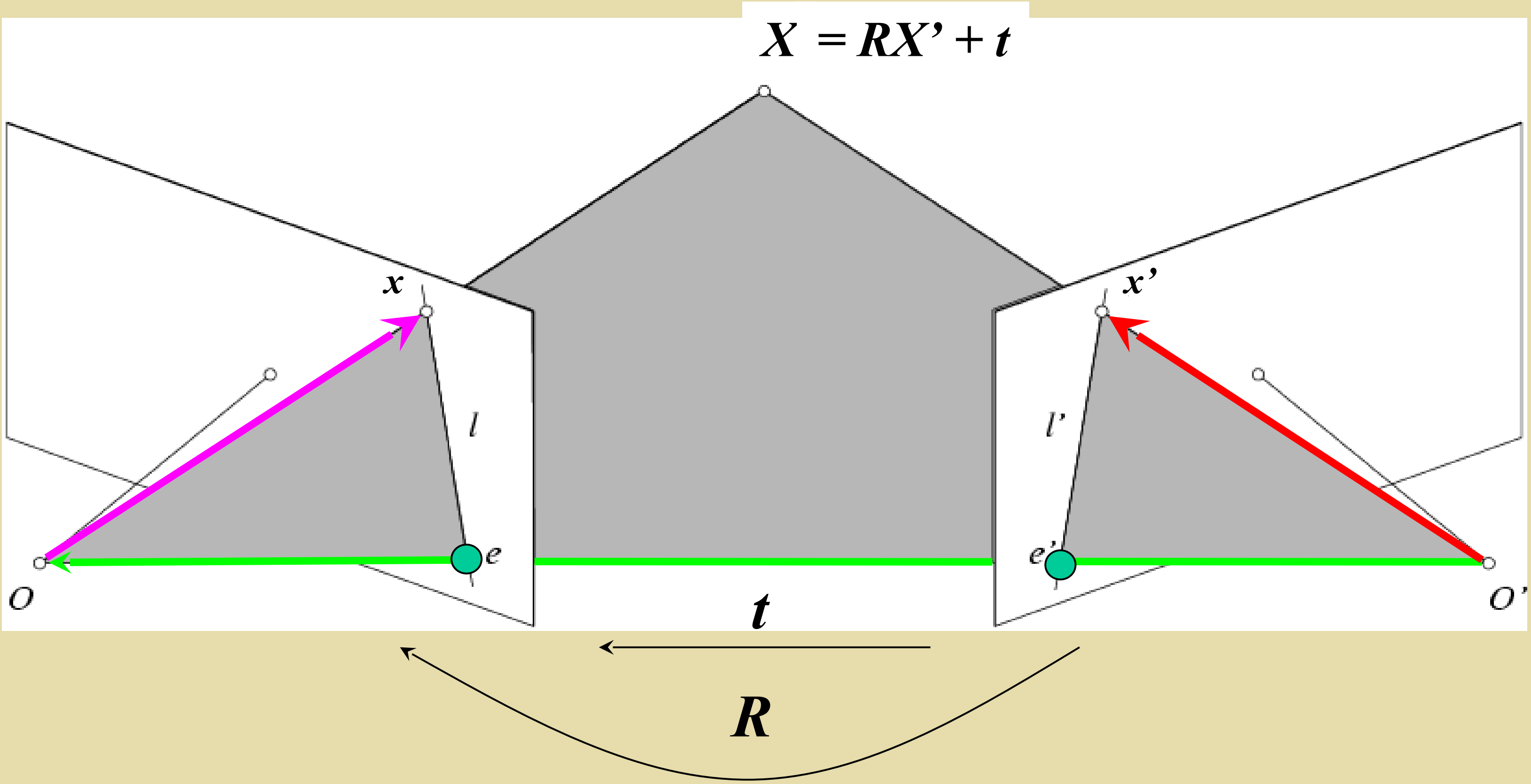
Epipolar constraint: Calibrated case



- Assume that the intrinsic and extrinsic parameters of the cameras are known
- We can multiply the projection matrix of each camera (and the image points) by the inverse of the calibration matrix to get *normalized* image coordinates
- We can also set the global coordinate system to the coordinate system of the first camera

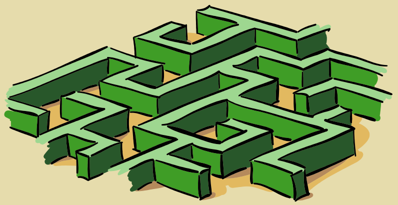


Epipolar constraint: Calibrated case



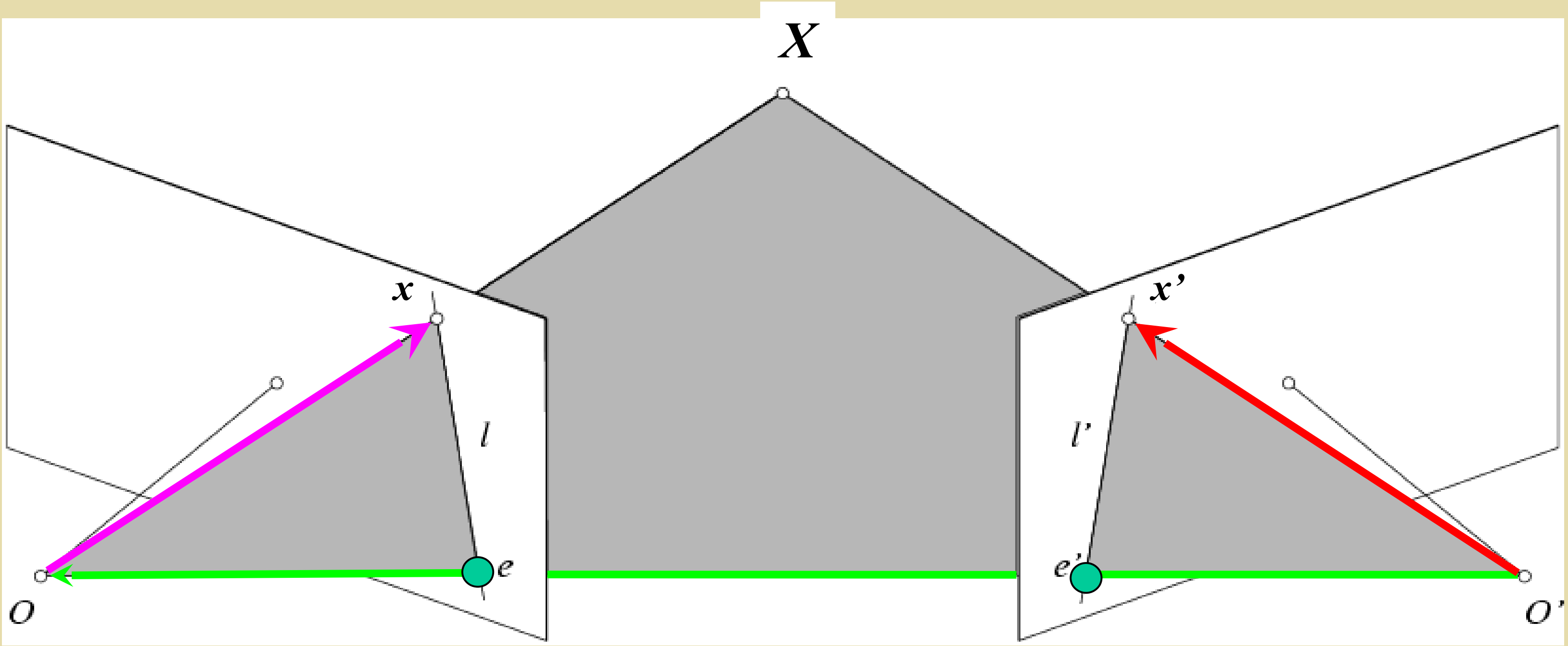
Camera matrix: $[\mathrm{I}|0]$
$\boldsymbol{X} = (u, v, w, 1)^T$
$\boldsymbol{x} = (u, v, w)^T$

Camera matrix: $[\boldsymbol{R}^T | -\boldsymbol{R}^T\boldsymbol{t}]$
Vector $\boldsymbol{x}'$ in second coord. system has coordinates $\boldsymbol{Rx}'$ in the first one



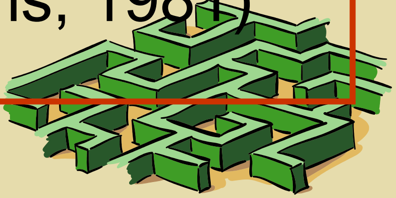
The vectors $\boldsymbol{x}$, $\boldsymbol{t}$, and $\boldsymbol{Rx}'$ are coplanar

Epipolar constraint: Calibrated case



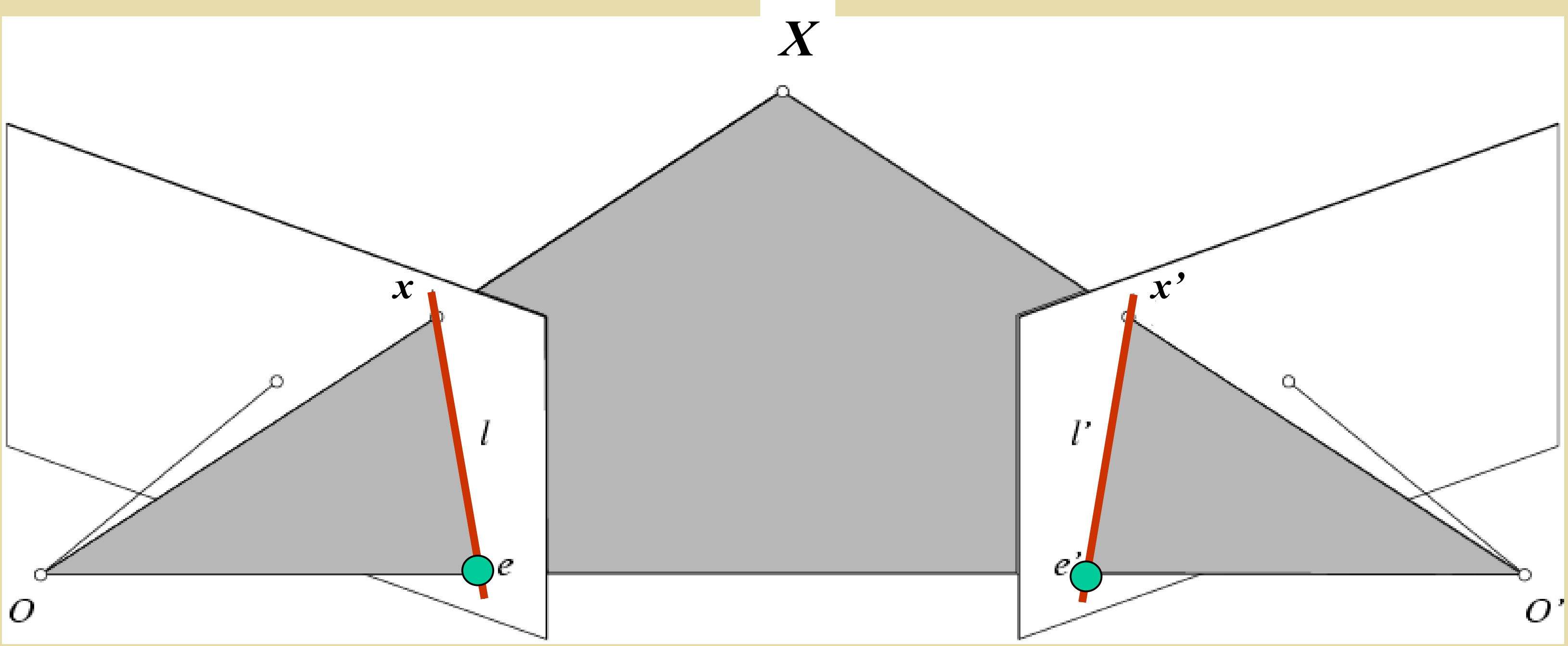
$$x \cdot [t \times (Rx')] = 0 \quad \Rightarrow \quad x^T E x' = 0 \quad \text{with} \quad E = [t_\times]R$$

Essential Matrix
(Longuet-Higgins, 1981)



The vectors ***x***, ***t***, and ***Rx'*** are coplanar

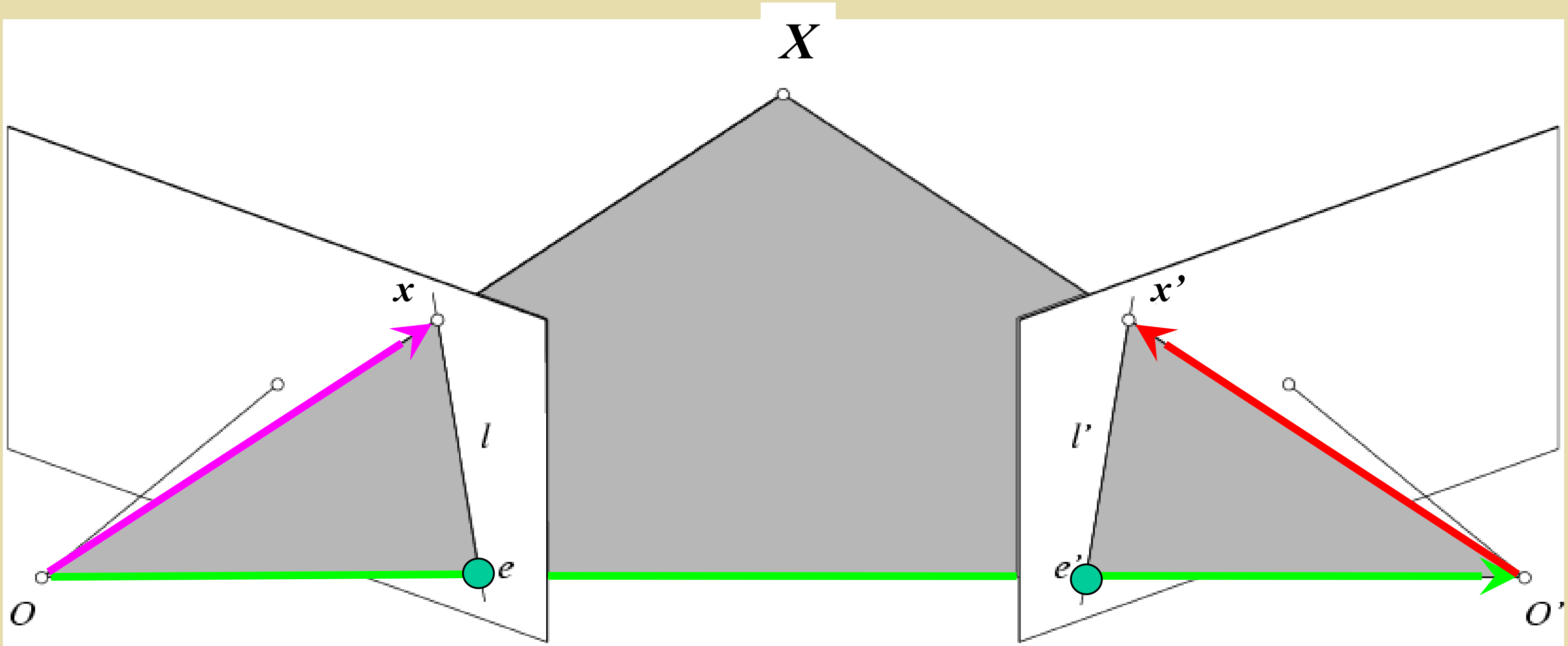
Epipolar constraint: Calibrated case



$$x \cdot [t \times (R x')] = 0 \quad \Rightarrow \quad x^T E x' = 0 \quad \text{with} \quad E = [t_\times] R$$

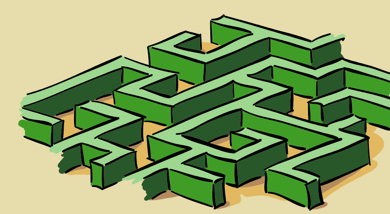
- $E\,x'$ is the epipolar line associated with x' ($l = E\,x'$)
- $E^T x$ is the epipolar line associated with x ($l' = E^T x$)
- $E\,e' = 0$ and $E^T e = 0$
- E is singular (rank two)
- E has five degrees of freedom

Epipolar constraint: Uncalibrated case

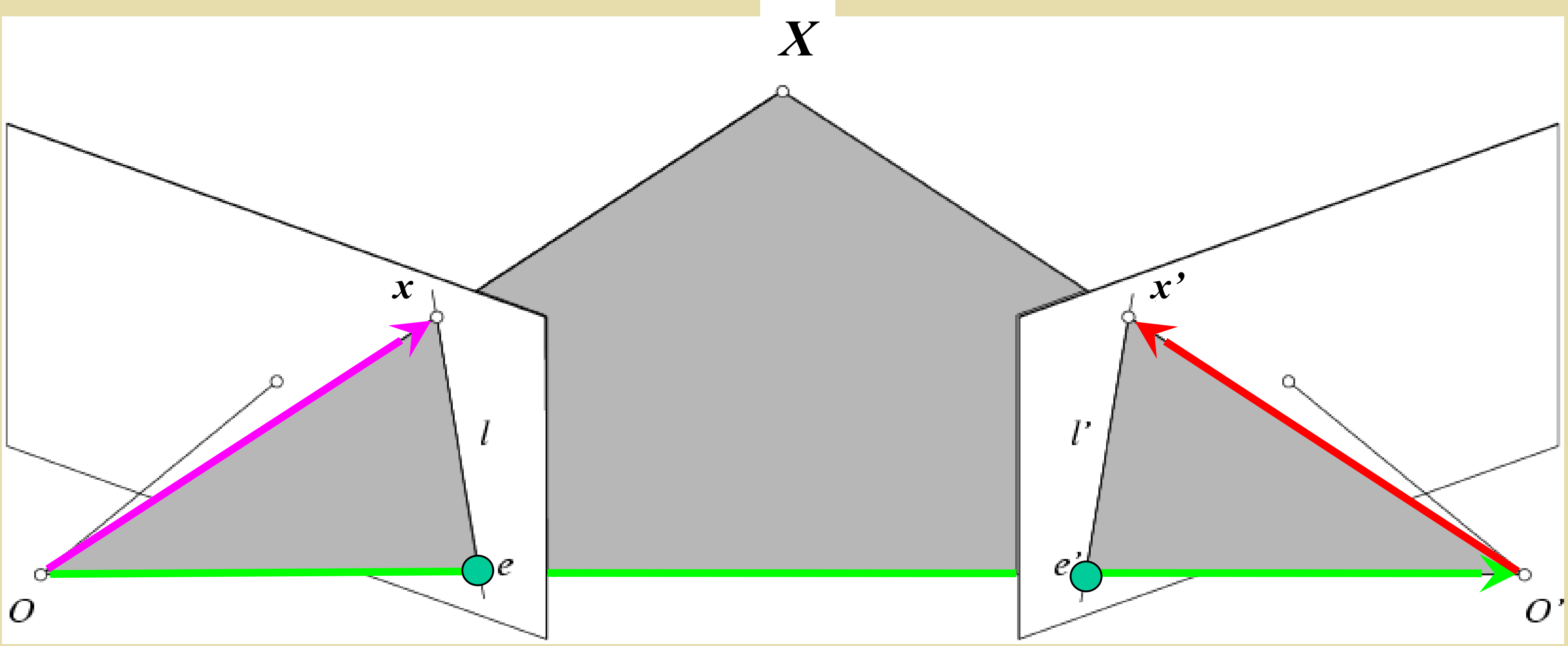


- The calibration matrices *K* and *K'* of the two cameras are unknown
- We can write the epipolar constraint in terms of *unknown* normalized coordinates:

$$\hat{x}^T E \hat{x}' = 0 \qquad x = K\hat{x}, \quad x' = K'\hat{x}'$$



Epipolar constraint: Uncalibrated case

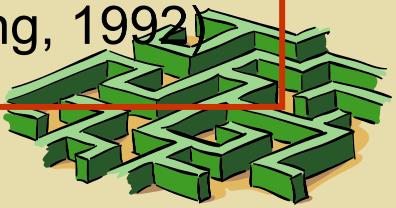


$$\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

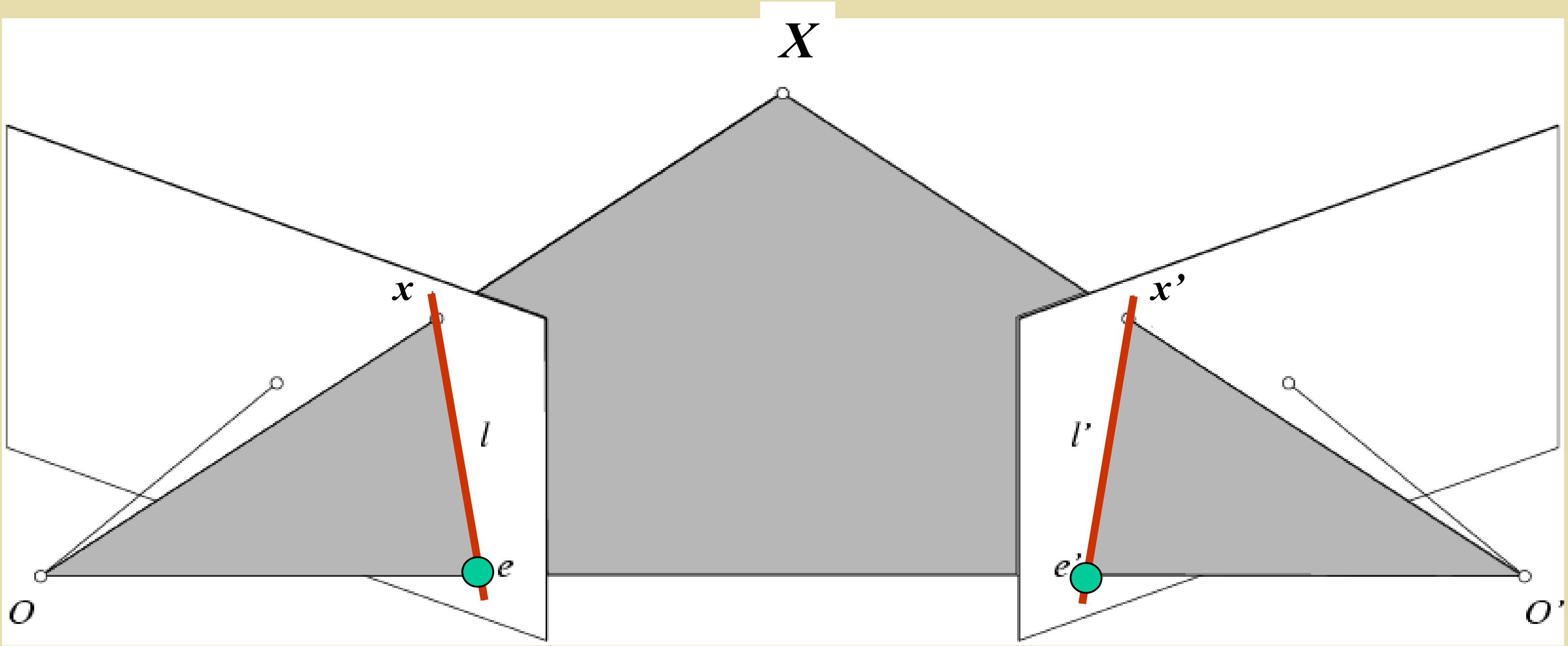
$$x = K \hat{x}$$

$$x' = K' \hat{x}'$$

Fundamental Matrix
(Faugeras and Luong, 1992)

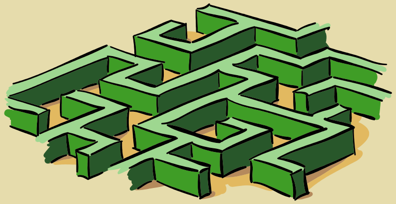


Epipolar constraint: Uncalibrated case



$$\hat{x}^T E \hat{x}' = 0 \quad \Rightarrow \quad x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

- $F x'$ is the epipolar line associated with x' ($l = F x'$)
- $F^T x$ is the epipolar line associated with x ($l' = F^T x$)
- $F e' = 0$ and $F^T e = 0$
- F is singular (rank two)
- F has seven degrees of freedom



The eight-point algorithm

$\boldsymbol{x} = (u, v, 1)^T, \quad \boldsymbol{x'} = (u', v', 1)^T$

$$(u, v, 1)\begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix}\begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0$$

$$(uu', uv', u, vu', vv', v, u', v', 1)\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \\ F_{33} \end{pmatrix} = 0$$

$$\begin{pmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' \end{pmatrix}\begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix} = -\begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

Minimize:

$$\sum_{i=1}^{N} (x_i^T F x_i')^2$$

under the constraint

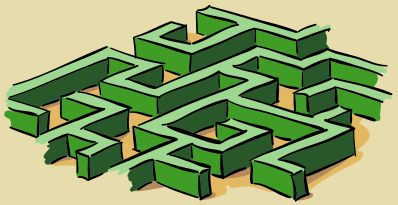
$|F|^2 = 1$

The eight-point algorithm

- Meaning of error $\sum_{i=1}^{N} (x_i^T F x_i')^2$:

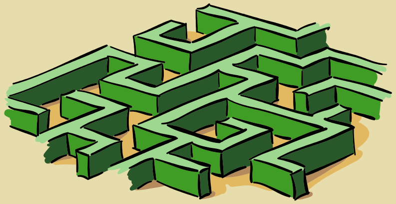
- Nonlinear approach: minimize

$$\sum_{i=1}^{N} \left[\mathrm{d}^2(x_i, F x_i') + \mathrm{d}^2(x_i', F^T x_i) \right]$$



Problem with eight-point algorithm

$$\begin{pmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' \\ u_2u_2' & u_2v_2' & u_2 & v_2u_2' & v_2v_2' & v_2 & u_2' & v_2' \\ u_3u_3' & u_3v_3' & u_3 & v_3u_3' & v_3v_3' & v_3 & u_3' & v_3' \\ u_4u_4' & u_4v_4' & u_4 & v_4u_4' & v_4v_4' & v_4 & u_4' & v_4' \\ u_5u_5' & u_5v_5' & u_5 & v_5u_5' & v_5v_5' & v_5 & u_5' & v_5' \\ u_6u_6' & u_6v_6' & u_6 & v_6u_6' & v_6v_6' & v_6 & u_6' & v_6' \\ u_7u_7' & u_7v_7' & u_7 & v_7u_7' & v_7v_7' & v_7 & u_7' & v_7' \\ u_8u_8' & u_8v_8' & u_8 & v_8u_8' & v_8v_8' & v_8 & u_8' & v_8' \end{pmatrix} \begin{pmatrix} F_{11} \\ F_{12} \\ F_{13} \\ F_{21} \\ F_{22} \\ F_{23} \\ F_{31} \\ F_{32} \end{pmatrix} = - \begin{pmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$



Εκτίμηση Εξωτερικών και εσωτερικών παραμέτρων καμερών

Ορισμοί - Ιδιότητες

$\boldsymbol{X} = (u, v, w, 1)^T$

$\boldsymbol{x} = (u, v, w)^T$

C = $[\mathrm{I}|0]$

C’= $[\boldsymbol{R}^T \,|\, -\boldsymbol{R}^T\boldsymbol{t}]$

$$x = PX$$

$$\mathrm{P} = \mathrm{KC}$$

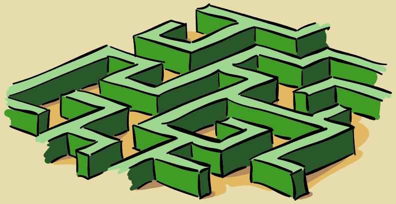
$$K = \begin{bmatrix} m_x & & \\ & m_y & \\ & & 1 \end{bmatrix} \begin{bmatrix} f & & p_x \\ & f & p_y \\ & & 1 \end{bmatrix}$$

$$x^T F x' = 0 \quad \text{with} \quad F = K^{-T} E K'^{-1}$$

$$E = [t_\times]R \Rightarrow F = K^{-T}[t_\times]RK'^{-1}$$

Ιδιες κάμερες:

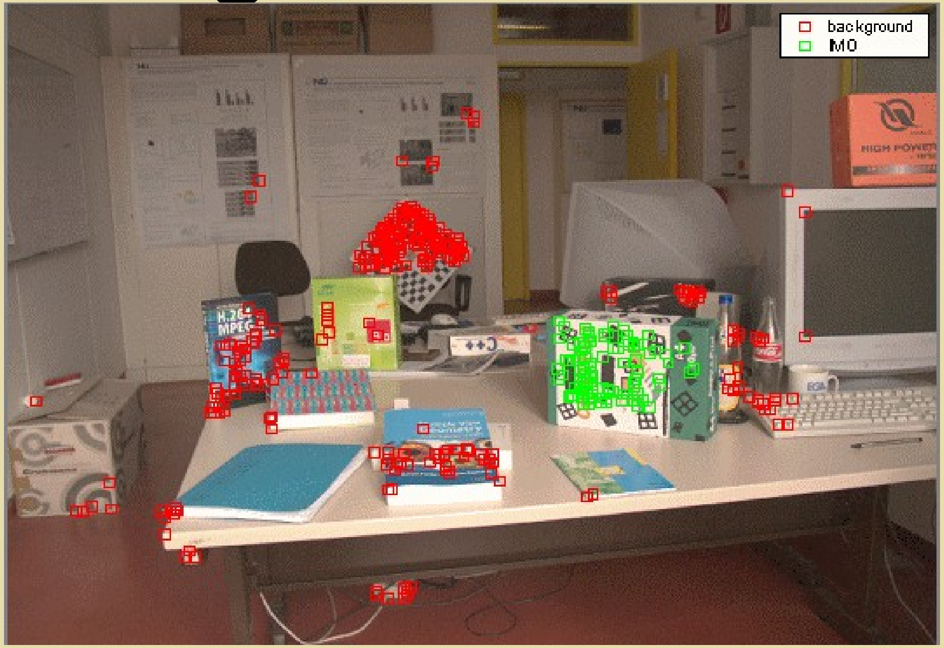
$$F = K^{-T}[t_\times]RK^{-1}$$



Εκτίμηση F: Motion segmentation

For each independent motion in the sequence, there exists a corresponding F-matrix, F_i, which fulfills the epipolar constraint

$$\mathbf{x}_1^T \mathbf{F}_i \mathbf{x}_2 = 0$$



- F-matrix estimation for consecutive keyframes RANSAC ➔ labeling of background and independent moving objects

EUSIPCO 2006

Εκτίμηση Θέσης στον τρισδιάστατο χώρο

Ορισμοί - Ιδιότητες

$\boldsymbol{X} = (u, v, w, 1)^T$
$\boldsymbol{x} = (u, v, w)^T$

$$x = PX \qquad x' = P'X$$

C = $[I|0]$

$$P = KC \qquad P' = K'C'$$

C’= $[\boldsymbol{R}^T \mid -\boldsymbol{R}^T\boldsymbol{t}]$

$$x = PX$$

$$P = KC$$

$$x = PX$$

Επίλυση:

$$x' = P'X$$