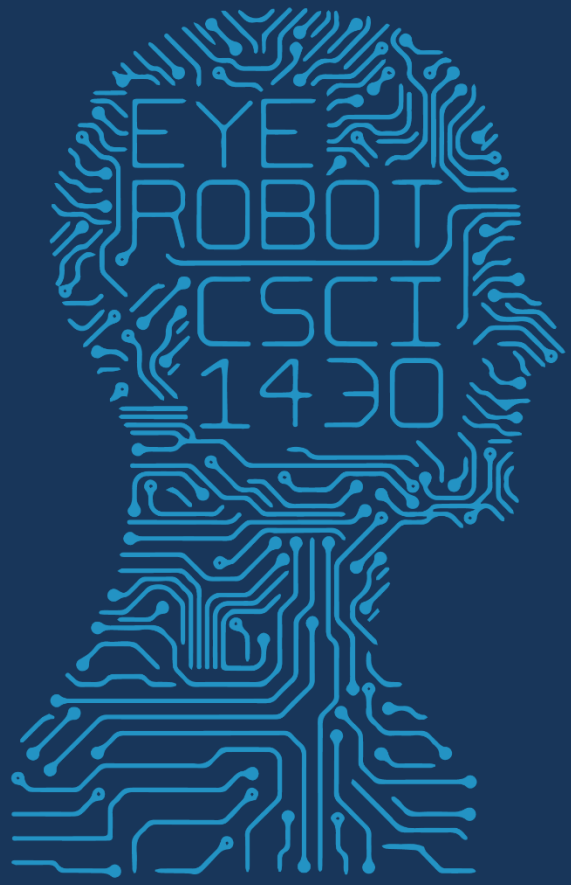
1950

Future Vision



2017 MWF 1pm 368

Computer Vision

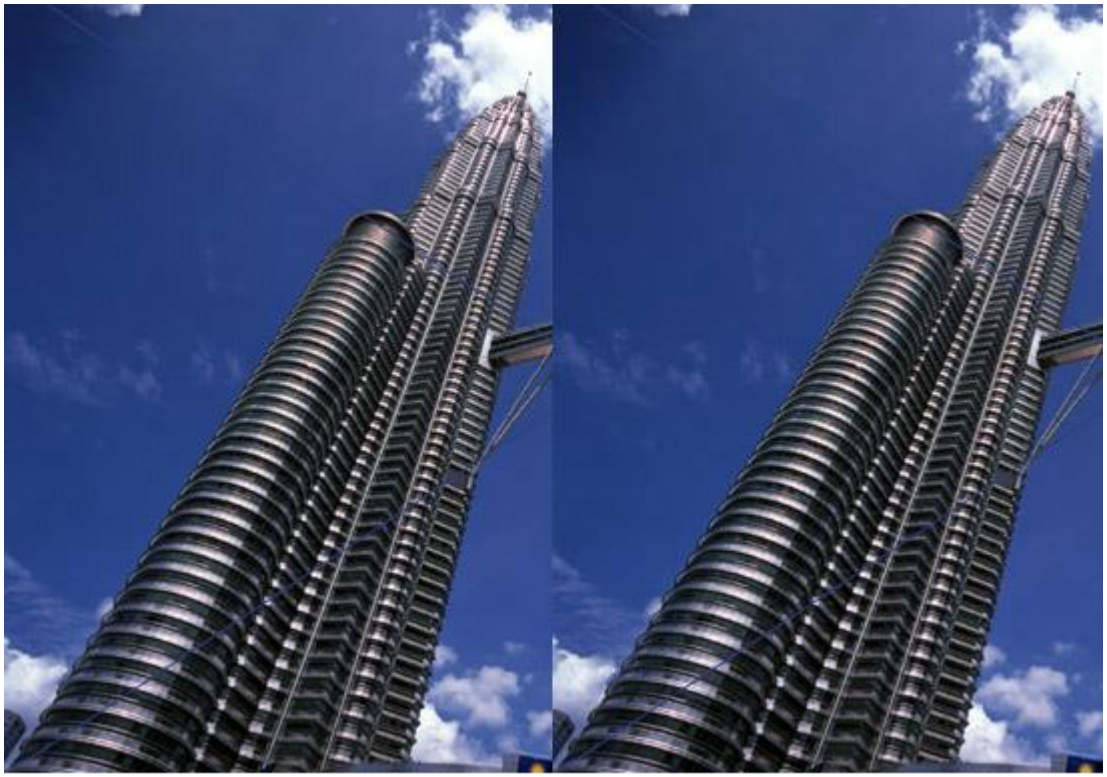
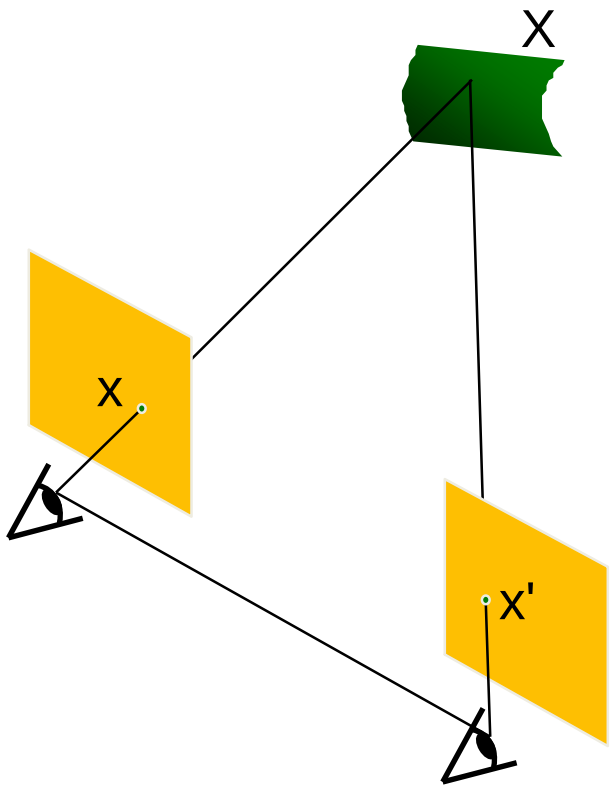
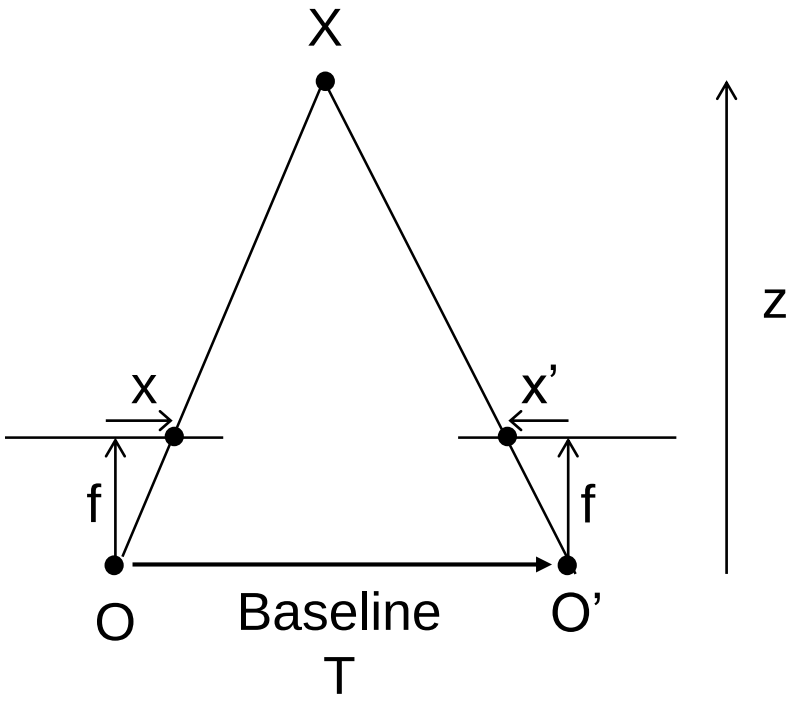



Photo: Georges Jansoon. Illusion: Frederick Kingdom, Ali Yoonessi and Elena

Depth from disparity

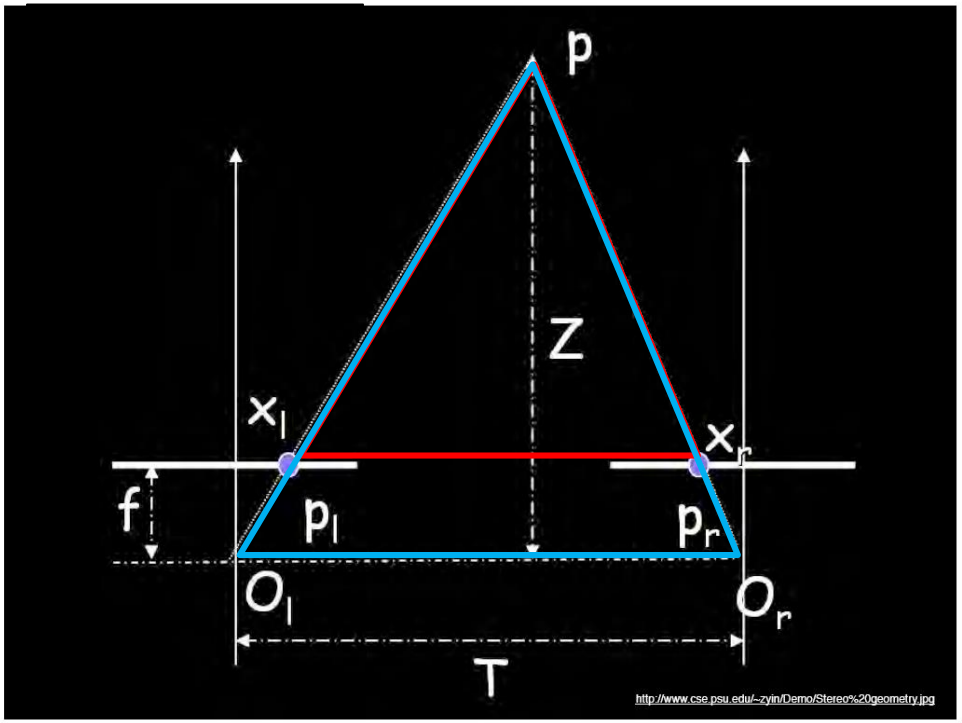
- Goal: recover depth by finding image coordinate x’ that corresponds to x





Geometry for a simple stereo system

- Assume parallel optical axes, known camera parameters (i.e., calibrated cameras). **What is expression for Z?**



Similar triangles (p_l, P, p_r) and (O_l, P, O_r):

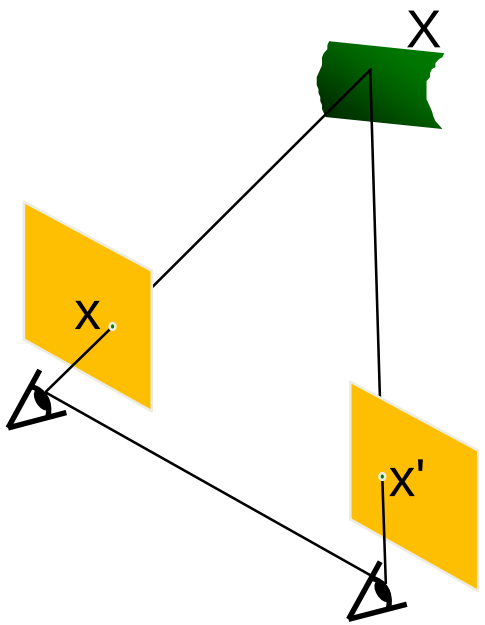
$$\frac{T + x_l - x_r}{Z - f} = \frac{T}{Z}$$

$$Z = f\frac{T}{x_r - x_l}$$

disparity

Depth from disparity

- Goal: recover depth by finding image coordinate x’ that corresponds to x
- Sub-Problems
  1. Calibration: How do we recover the relation of the cameras (if not already known)?
  2. Correspondence: How do we search for the matching point x’?



Depth from disparity

image I(x,y)

Disparity map D(x,y)

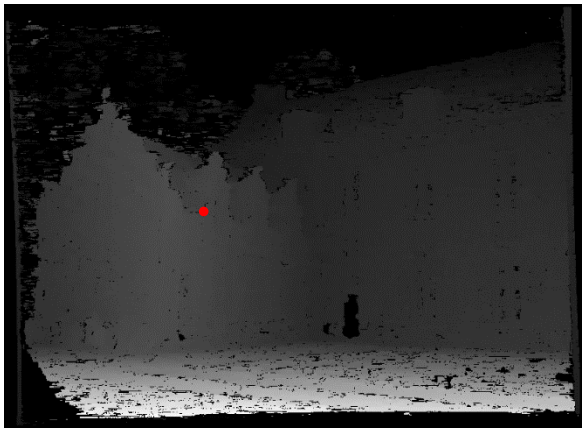


image I´(x´,y´)

$(x´,y´)=(x+D(x,y), y)$

If we could find the **corresponding points** in two images, we could **estimate relative depth**…

What do we need to know?

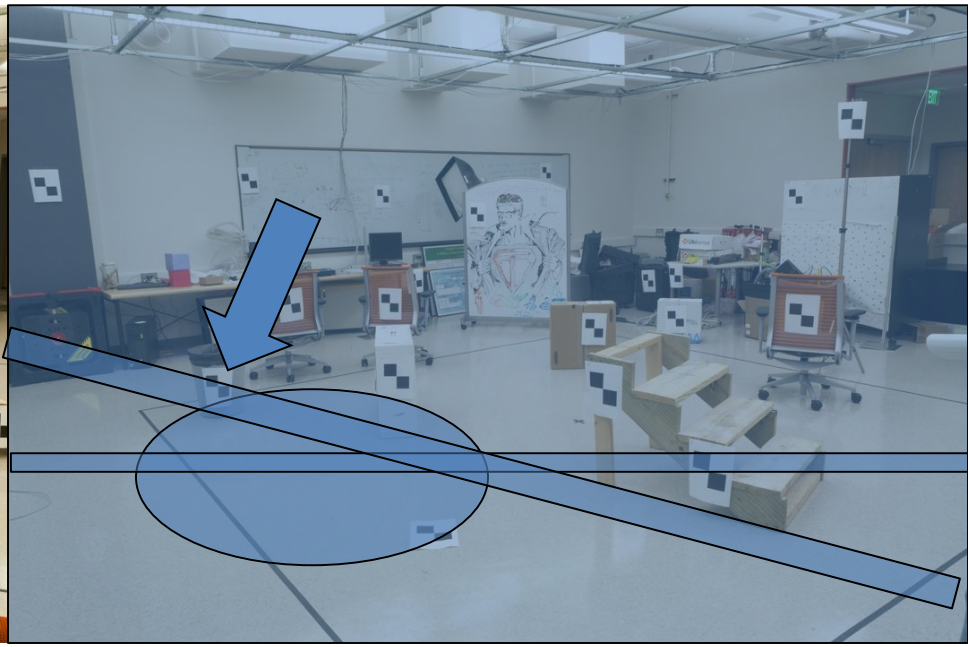
1. Calibration for the two cameras.
    1. Intrinsic matrices for both cameras (e.g., f)
    2. Baseline distance T in parallel camera case
    3. R, t in non-parallel case

2. Correspondence for every pixel.

    Like project 2, but project 2 is “sparse”.

    We need “dense” correspondence!

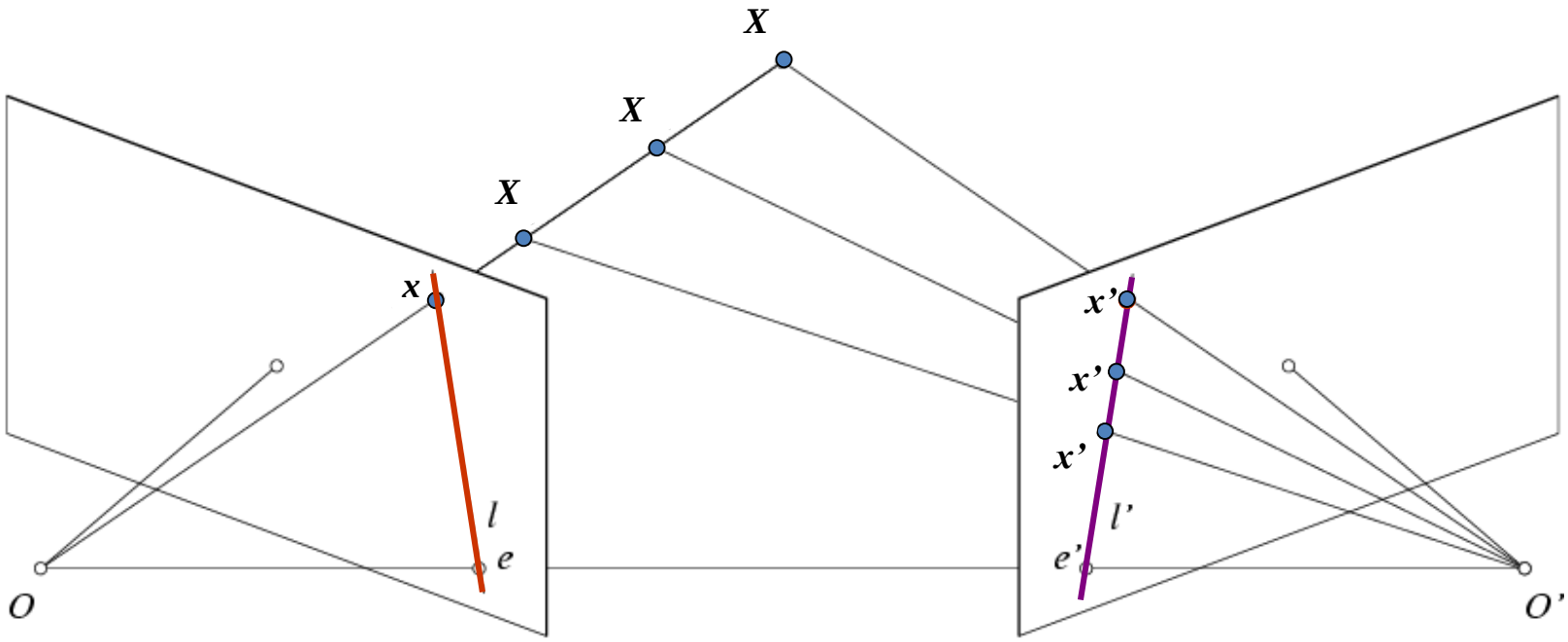
Correspondence for every pixel.
Where do we need to search?



Wouldn’t it be nice to know where matches can live?

Epipolar geometry
Constrains 2D search to 1D

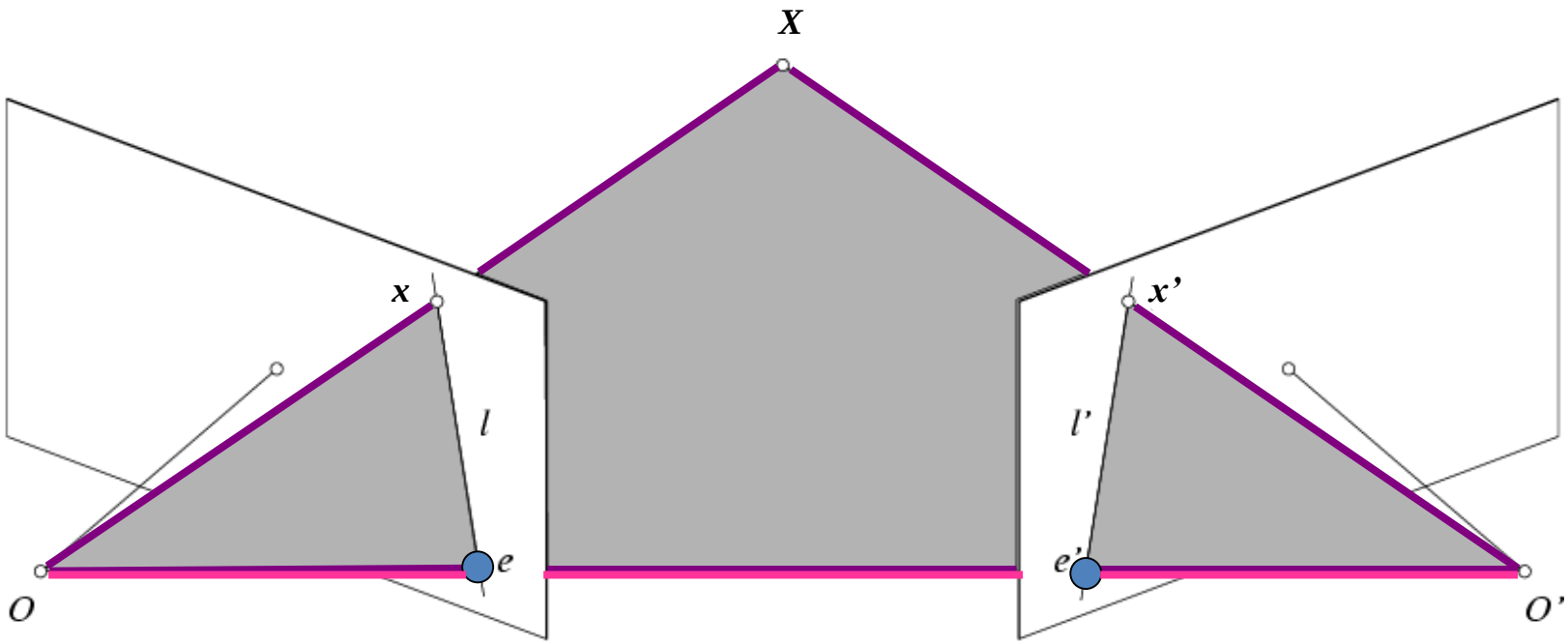
Key idea: Epipolar constraint



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

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

Epipolar geometry: notation



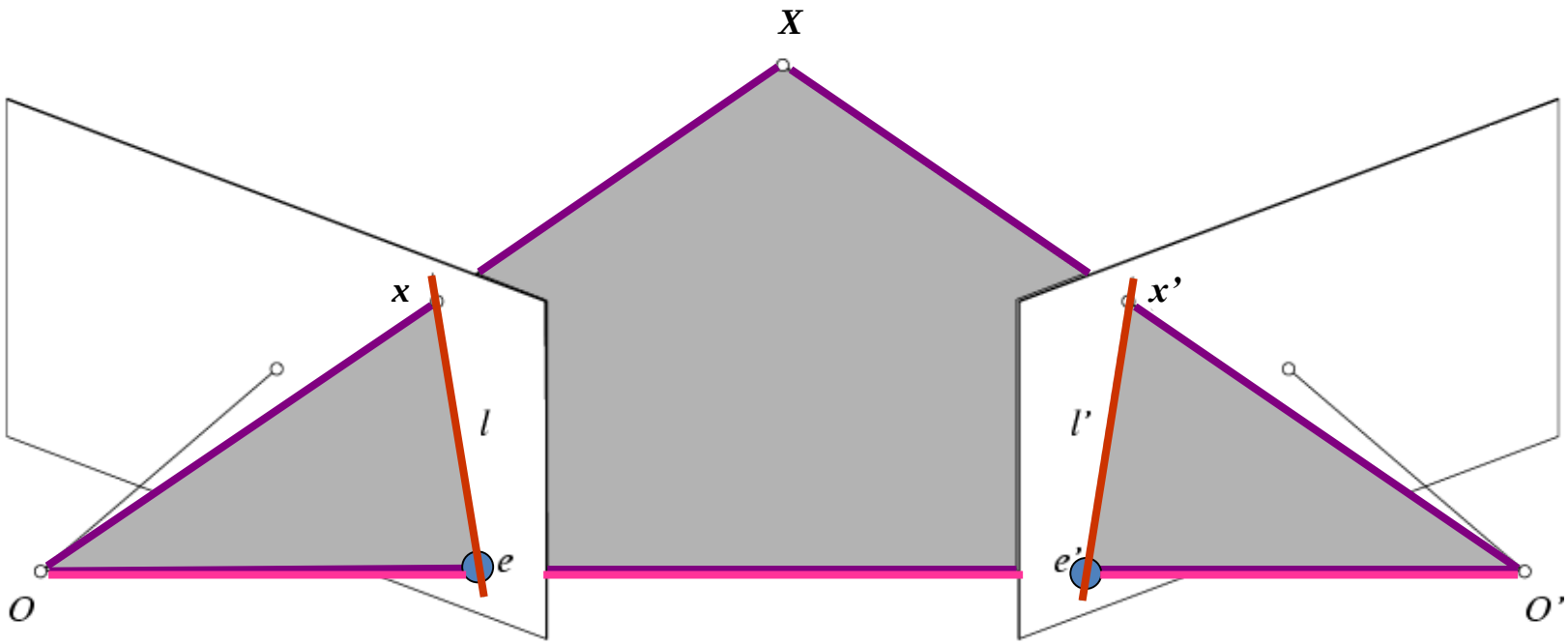
- **Baseline** – line connecting the two camera centers
- **Epipoles**

= intersections of baseline with image planes

= projections of the other camera center

- **Epipolar Plane** – plane containing baseline (1D family)

Epipolar geometry: notation



- **Baseline** – line connecting the two camera centers
- **Epipoles**

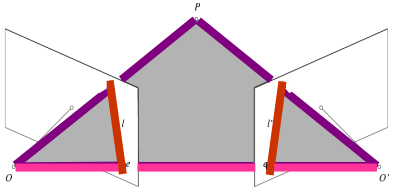
= intersections of baseline with image planes

= projections of the other camera center

- **Epipolar Plane** – plane containing baseline (1D family)
- **Epipolar Lines** - intersections of epipolar plane with image planes (always come in corresponding pairs)

Think Pair Share

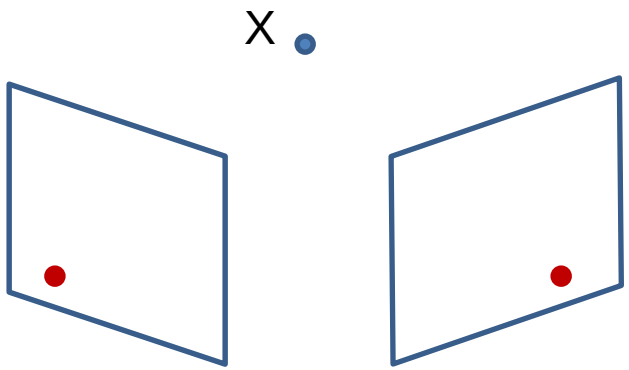
● = camera center



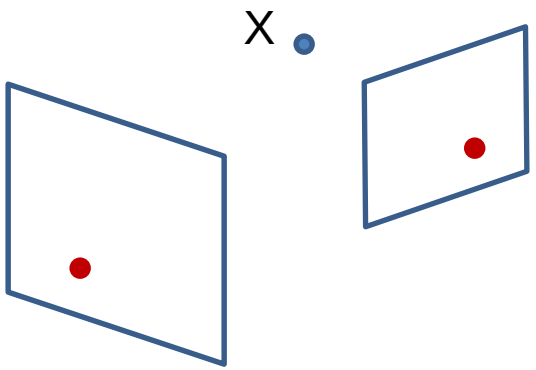
Where are the epipoles?

What do the epipolar lines look like?

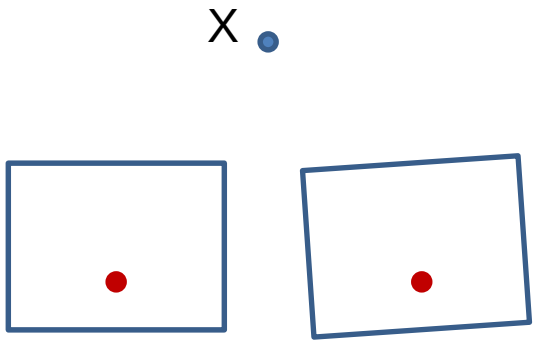
a)



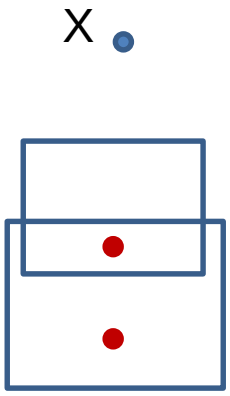
b)



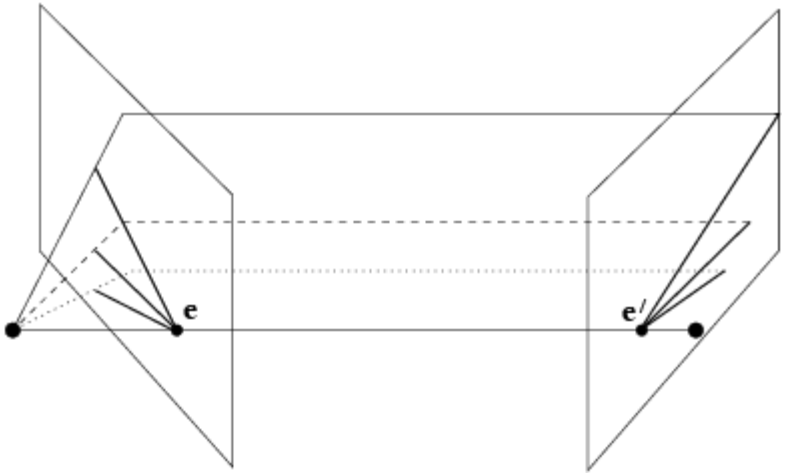
c)

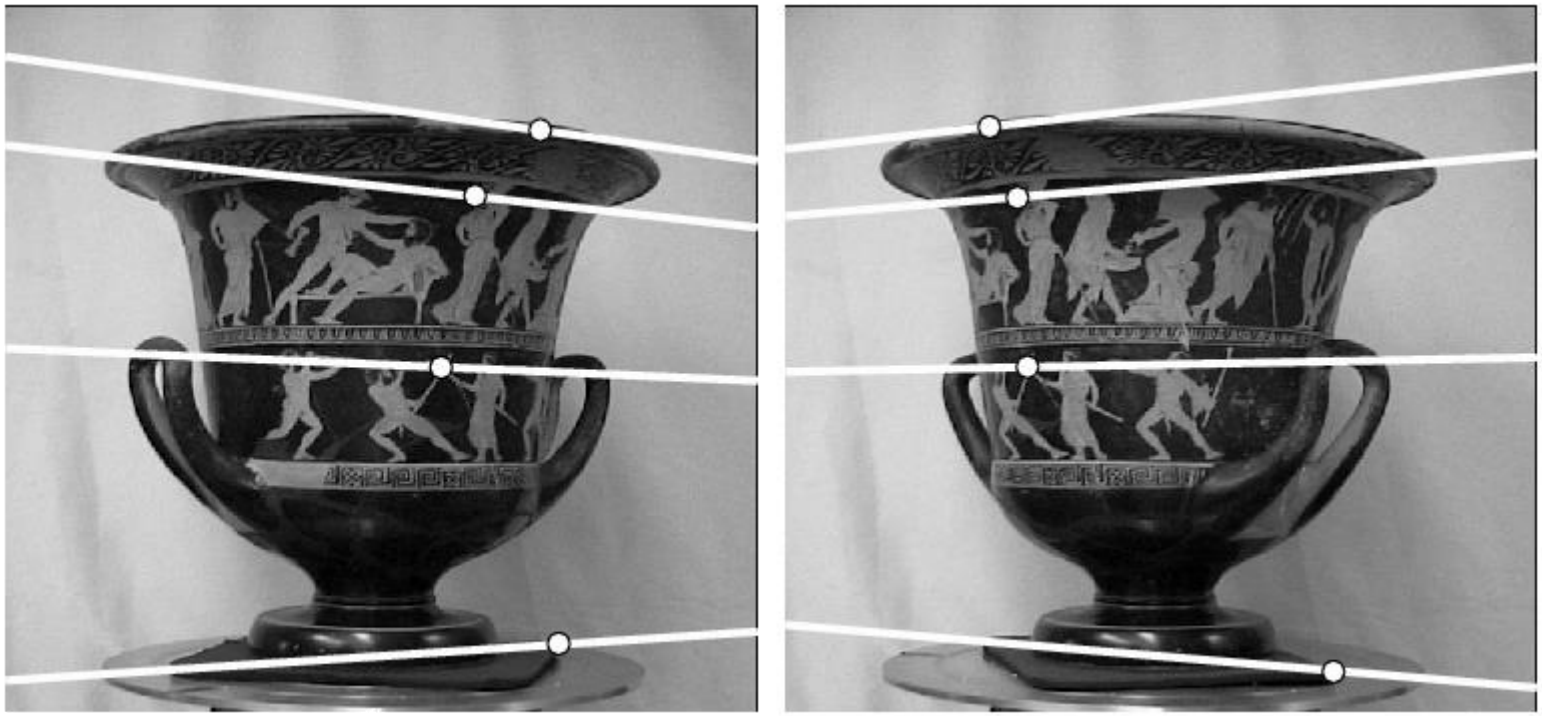


d)

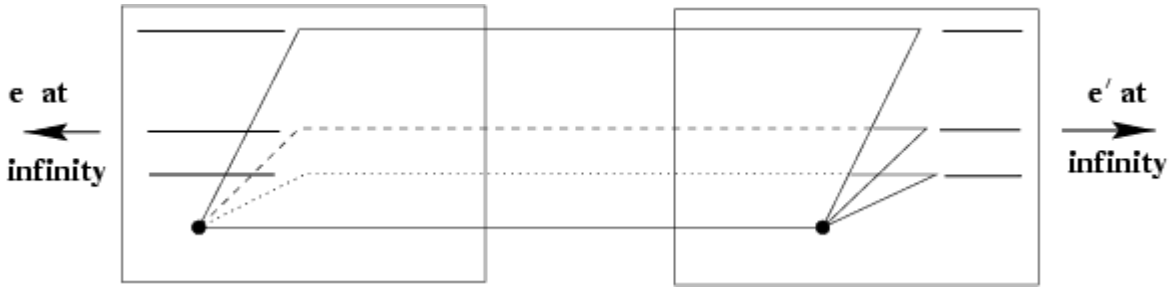


Example: Converging cameras

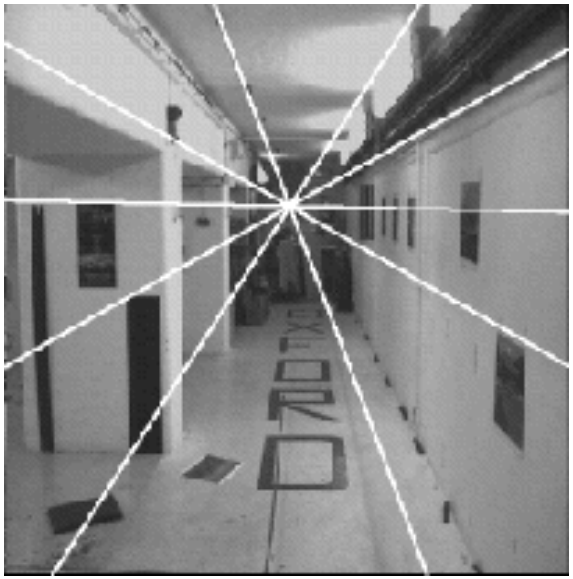


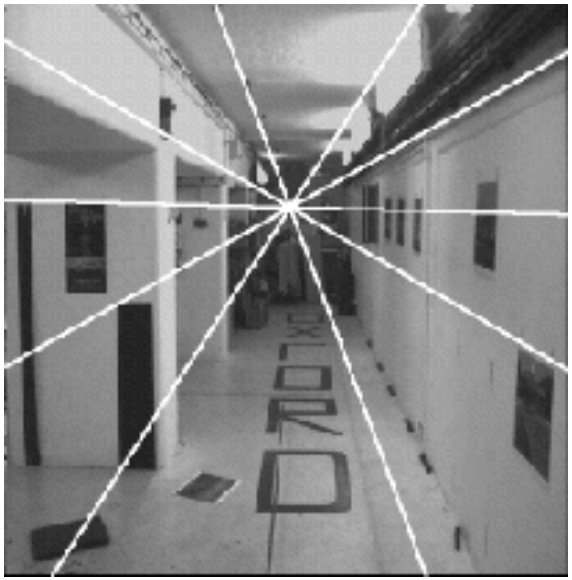


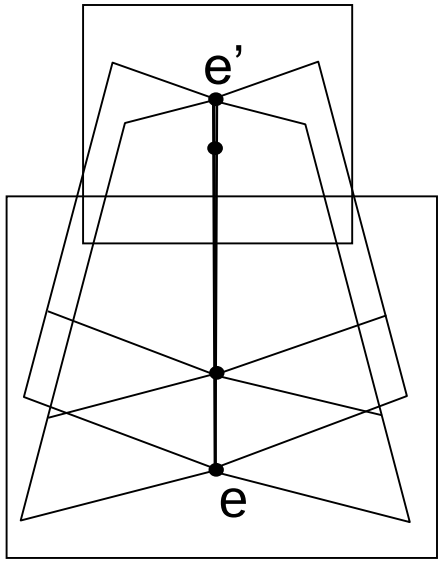
Example: Motion parallel to image plane



Example: Forward motion

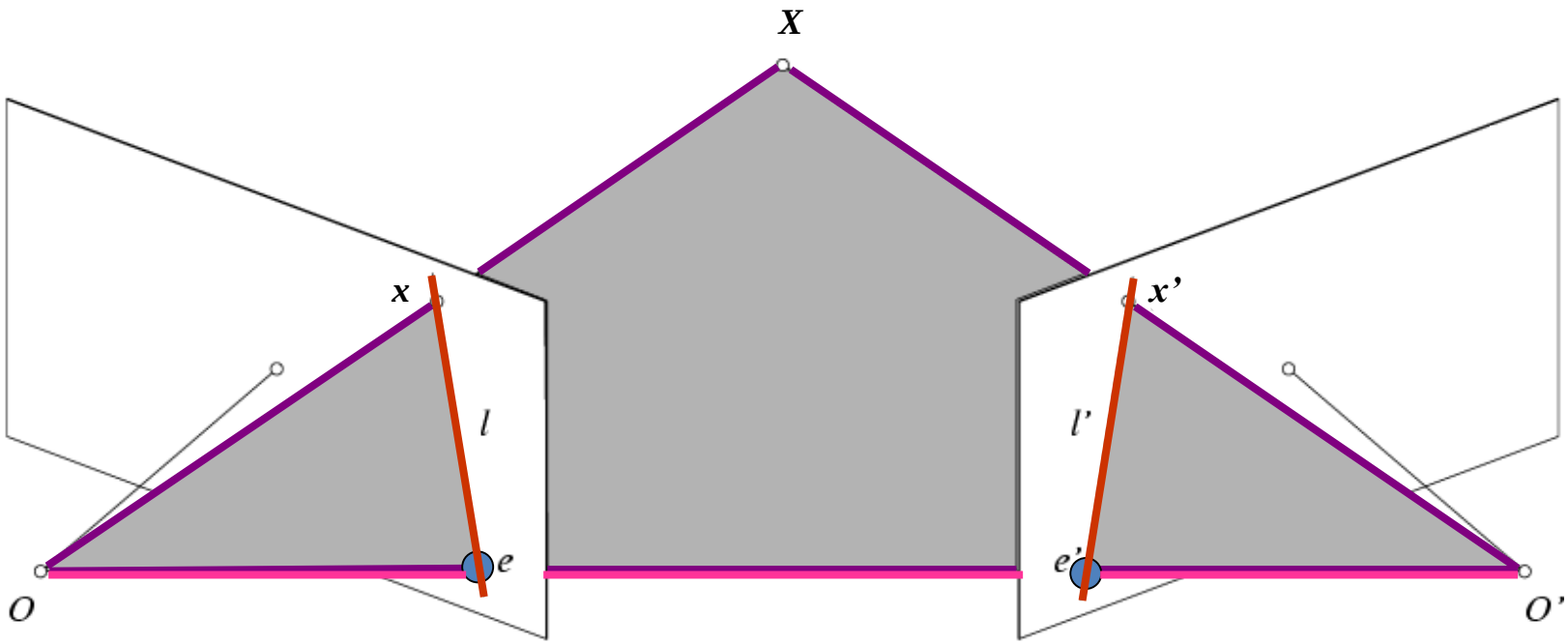






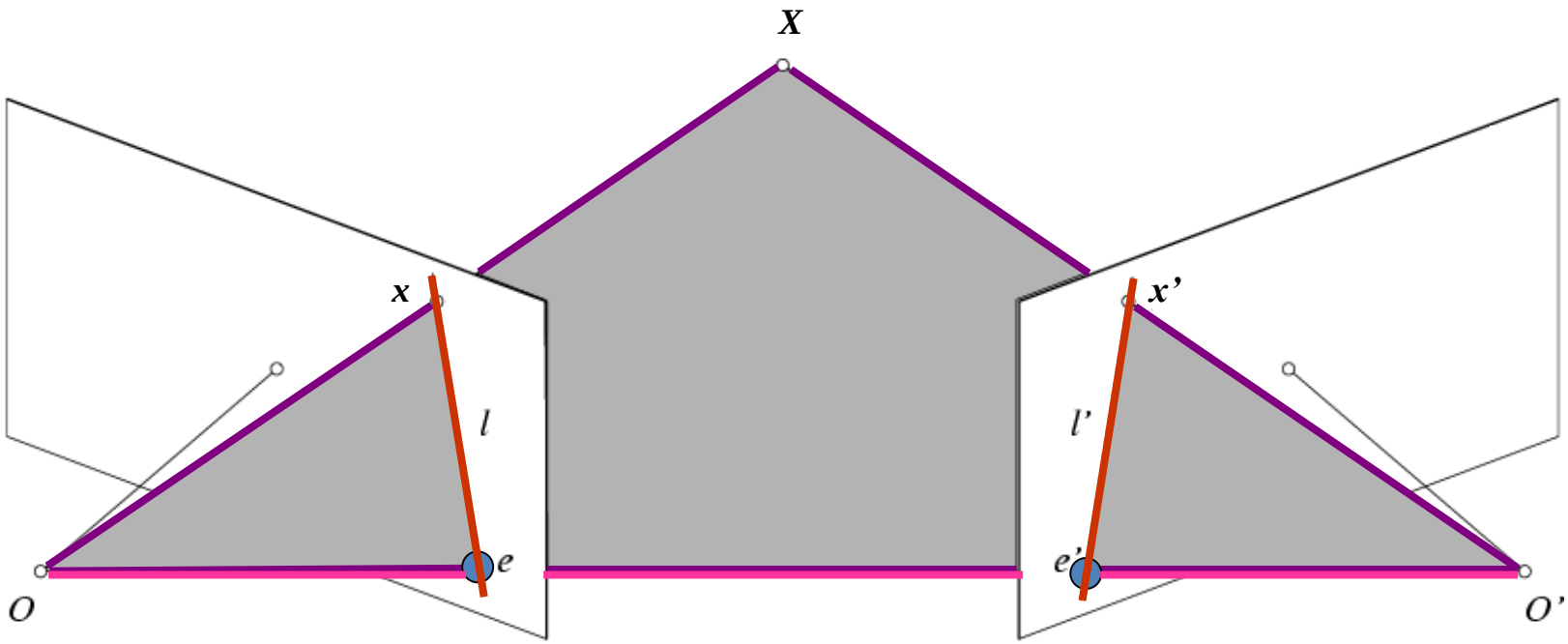
Epipole has same coordinates in both images.
Points move along lines radiating from e: “Focus of expansion”

What is this useful for?



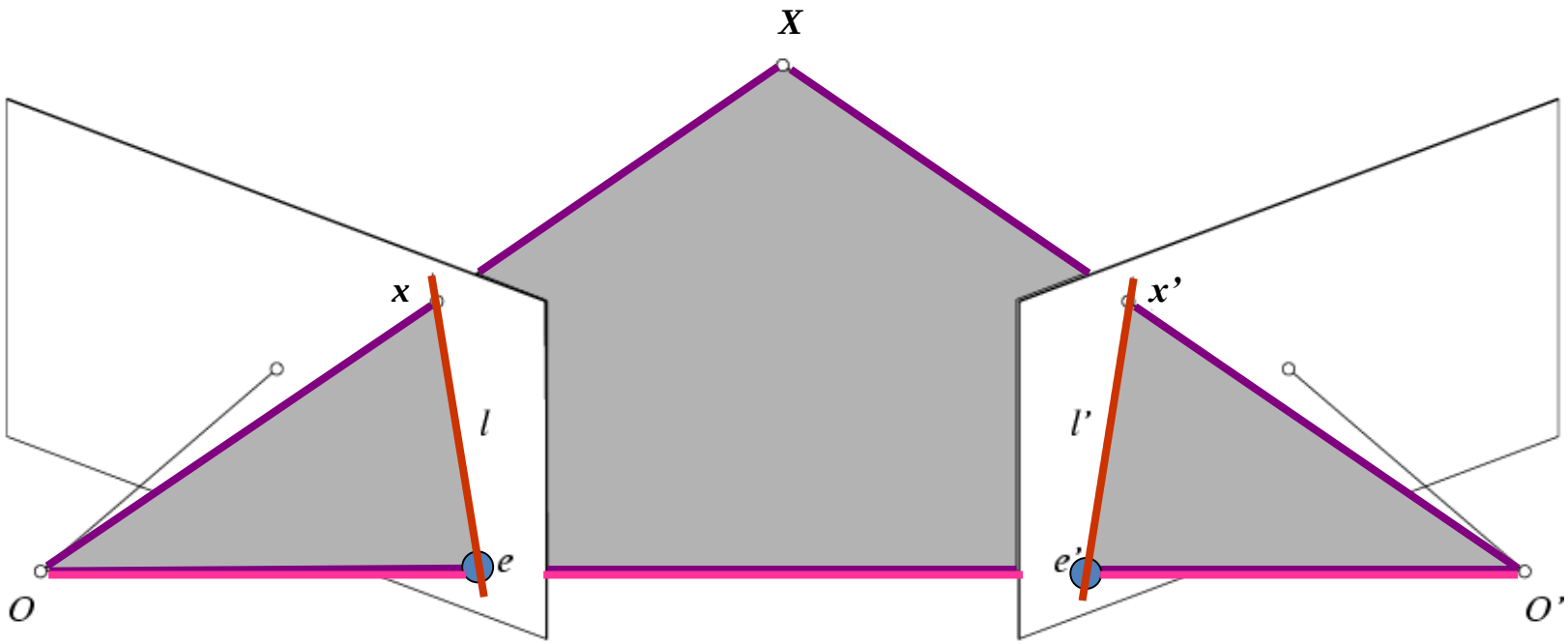
- Find *X*: If I know *x*, and have calibrated cameras (known intrinsics K,K' and extrinsic relationship), I can restrict *x'* to be along *l'.*
- Discover disparity for stereo.

What is this useful for?



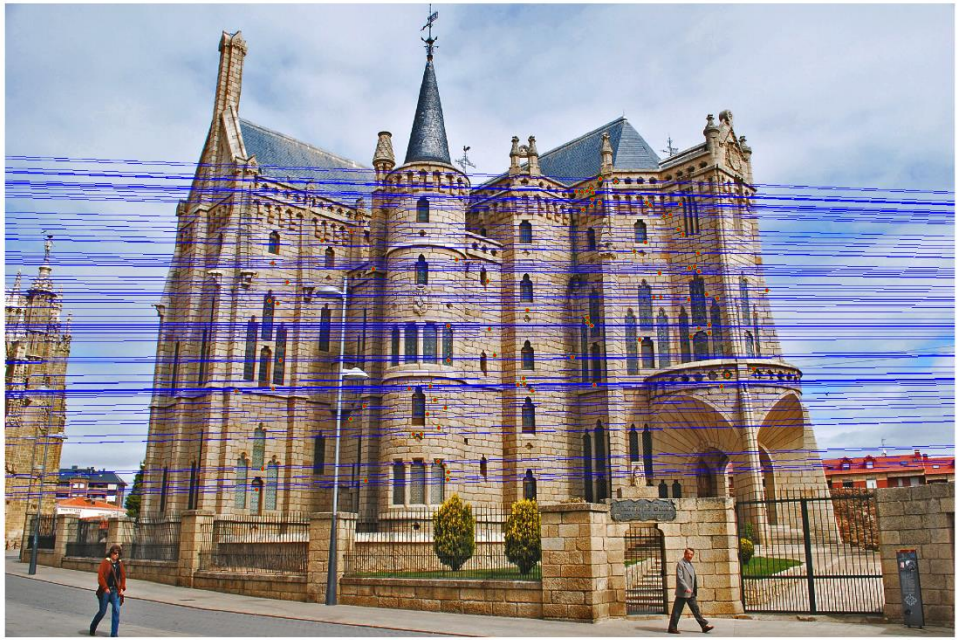
- Given candidate *x, x'* correspondences, estimate relative position and orientation between the cameras and the 3D position of corresponding image points.

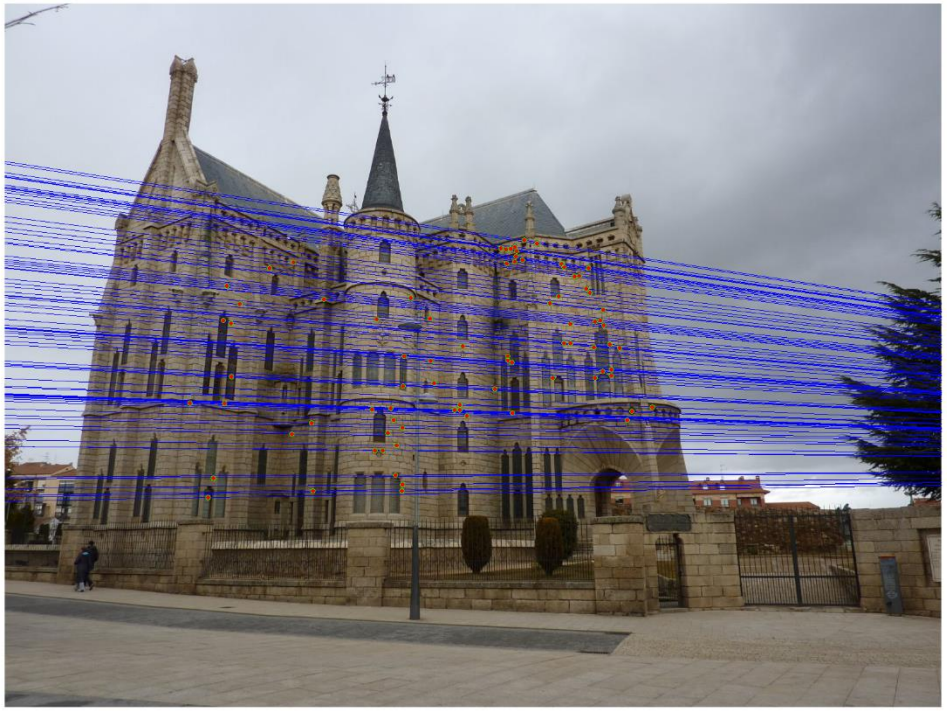
What is this useful for?



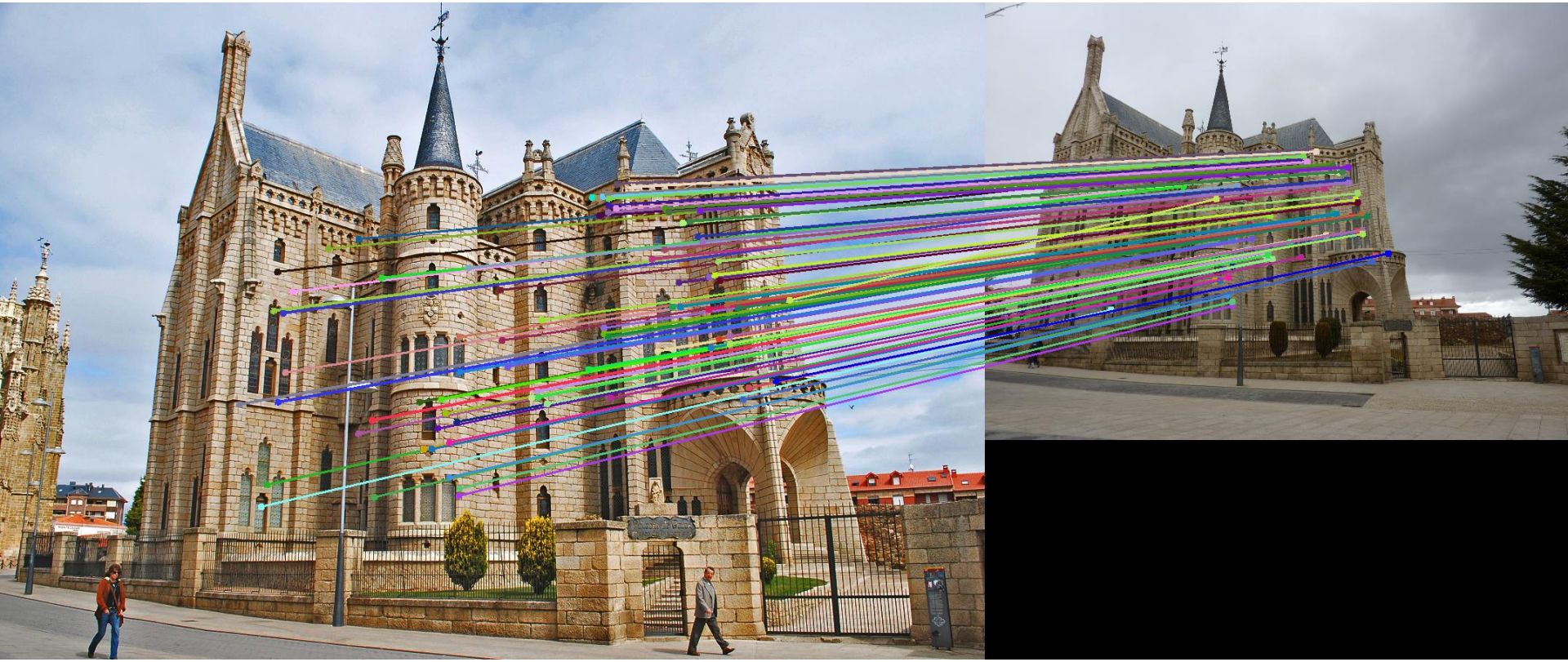
- Model fitting: see if candidate *x, x'* correspondences fit estimated projection models of cameras 1 and 2.

Epipolar lines

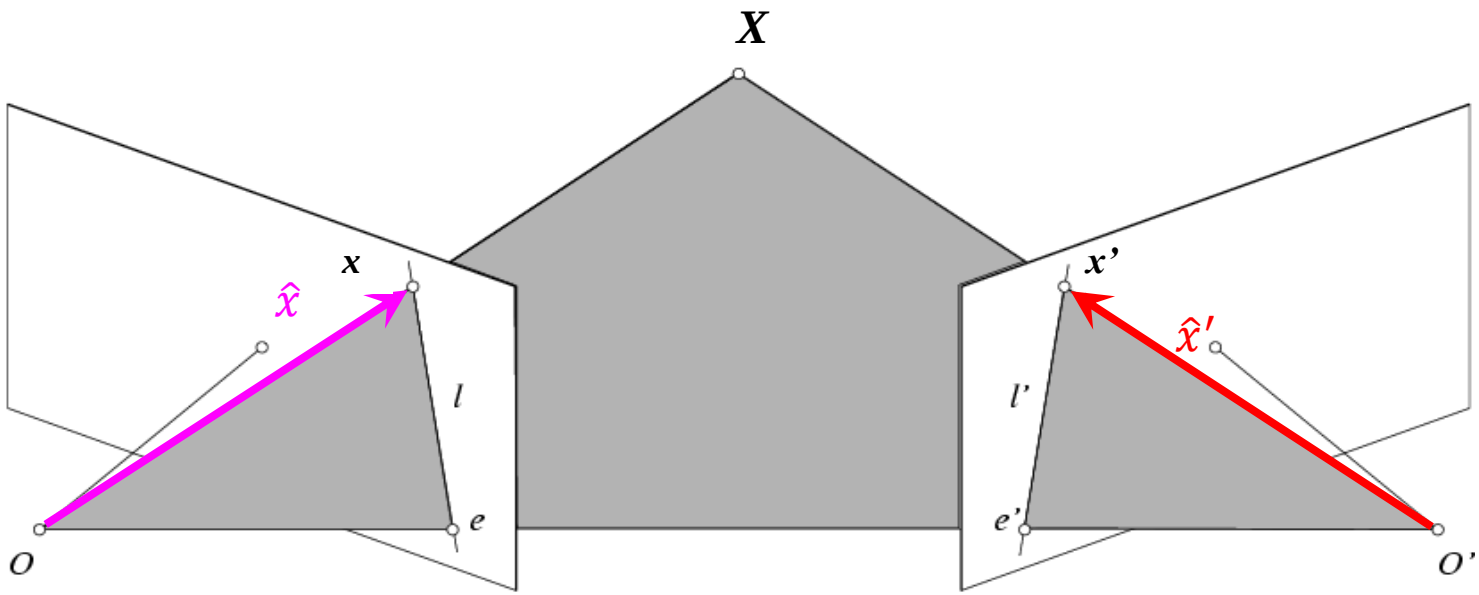




Keep only the matches at are “inliers” with respect to the “best” fundamental matrix



Epipolar constraint: Calibrated case



$$\hat{x} = K^{-1}x = X$$

Homogeneous 2d point (3D ray towards X)

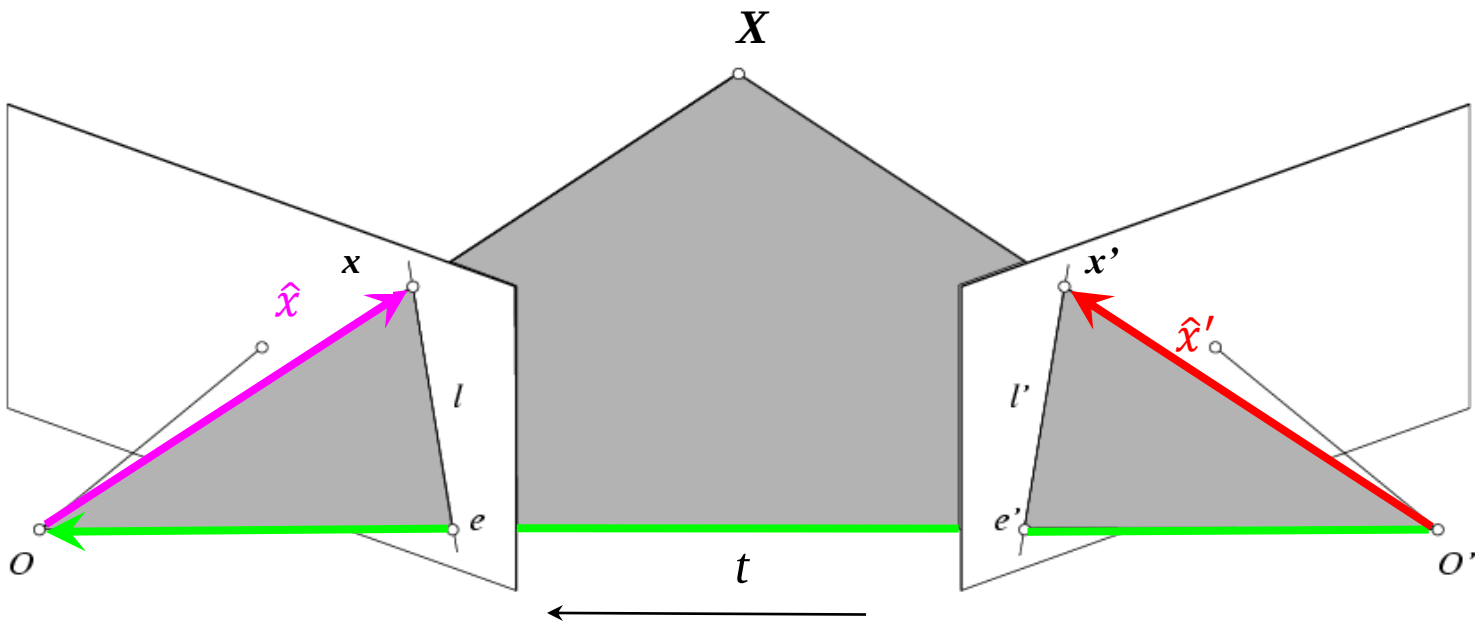
2D pixel coordinate (homogeneous)

3D scene point

$$\hat{x}' = K'^{-1}x' = X'$$

3D scene point in 2nd camera's 3D coordinates

Epipolar constraint: Calibrated case



$$\hat{x} = K^{-1}x = X$$

Homogeneous 2d point (3D ray towards X)

2D pixel coordinate (homogeneous)

3D scene point

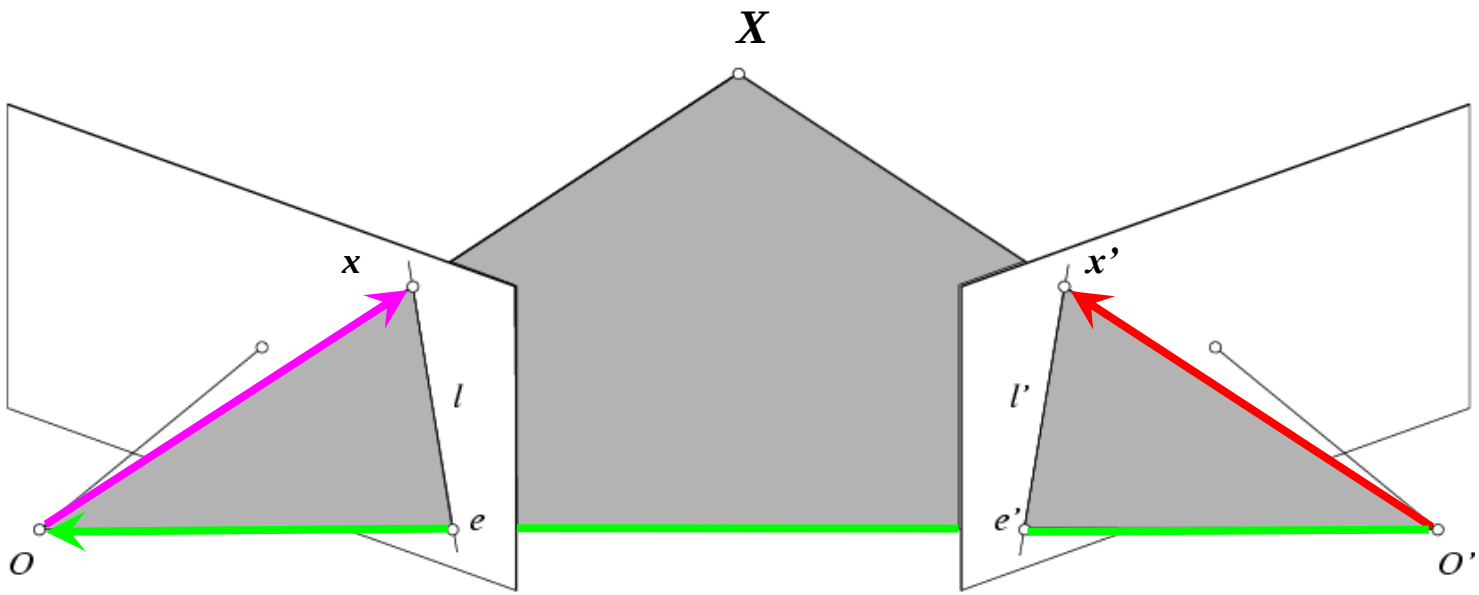
$$\hat{x}' = K'^{-1}x' = X'$$

3D scene point in 2nd camera's 3D coordinates

$$\hat{x} \cdot [t \times (R\hat{x}')] = 0$$

(because $\hat{x}, R\hat{x}'$, and t are co-planar)

Essential matrix



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

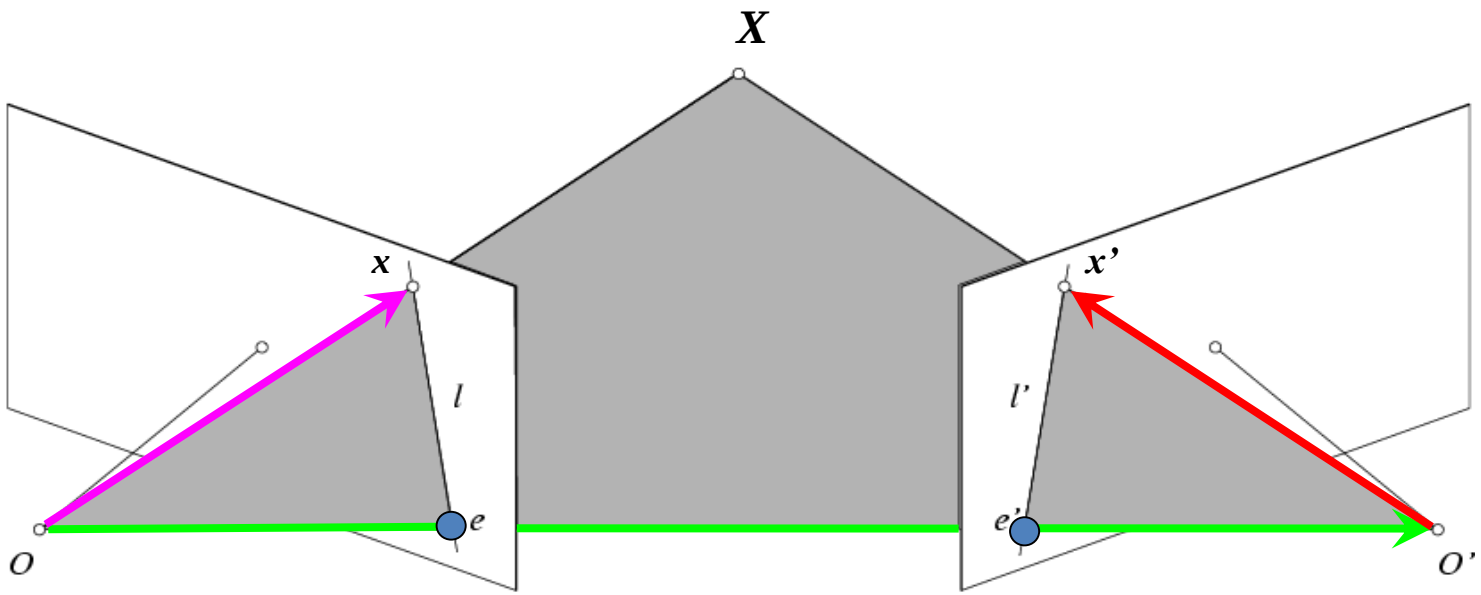
E is a 3x3 matrix which relates corresponding pairs of normalized homogeneous image points across pairs of images – for *K* calibrated cameras.

Estimates relative position/orientation.

Essential Matrix
(Longuet-Higgins, 1981)

Note: $[t]_\times$ is matrix representation of cross product

Epipolar constraint: Uncalibrated case



- If we don't know *K* and *K'*, then we can write the epipolar constraint in terms of *unknown* normalized coordinates:

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

The Fundamental Matrix

Without knowing K and K', we can define a similar relation using *unknown* normalized coordinates

$$\hat{x}^T E \hat{x}' = 0$$

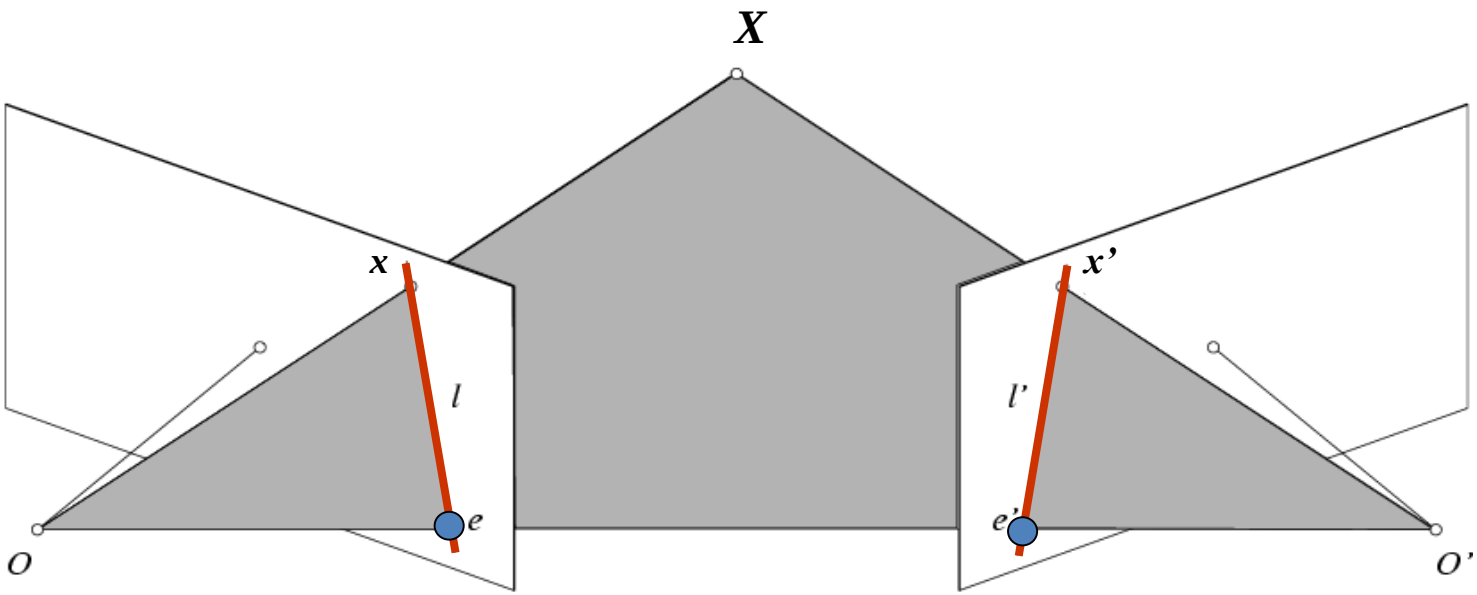
$$\hat{x} = K^{-1} x$$

$$\hat{x}' = K'^{-1} x'$$

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

Fundamental Matrix
(Faugeras and Luong, 1992)

Properties of the Fundamental matrix



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

- $F\,x' = 0$ is the epipolar line l associated with x'
- $F^T x = 0$ is the epipolar line l' associated with x

- F is singular (rank two): det(F)=0
- $F\,e' = 0$ and $F^T e = 0$ (nullspaces of F = e'; nullspace of F^T = e')
- F has seven degrees of freedom: 9 entries but defined up to scale, det(F)=0

F in more detail

- F is a 3x3 matrix
- Rank 2 -> projection; one column is a linear combination of the other two.
- Determined up to scale.
- 7 degrees of freedom

$$\begin{bmatrix} a & b & \alpha a + \beta b \\ c & d & \alpha c + \beta d \\ e & f & \alpha e + \beta f \end{bmatrix}$$ where *a* is scalar; e.g., can normalize out.

Given x projected from X into image 1, F constrains the projection of x’ into image 2 to an epipolar line.

Estimating the Fundamental Matrix

- 8-point algorithm
  - Least squares solution using SVD on equations from 8 pairs of correspondences
  - Enforce det(F)=0 constraint using SVD on F

Note: estimation of F (or E) is degenerate for a planar scene.

8-point algorithm

1. Solve a system of homogeneous linear equations
   a. Write down the system of equations

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

$$uu'f_{11} + uv'f_{12} + uf_{13} + vu'f_{21} + vv'f_{22} + vf_{23} + u'f_{31} + v'f_{32} + f_{33} = 0$$

$$\mathrm{A}\boldsymbol{f} = \begin{bmatrix} u_1u_1' & u_1v_1' & u_1 & v_1u_1' & v_1v_1' & v_1 & u_1' & v_1' & 1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ u_nu_v' & u_nv_n' & u_n & v_nu_n' & v_nv_n' & v_n & u_n' & v_n' & 1 \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ \vdots \\ f_{33} \end{bmatrix} = \mathbf{0}$$

8-point algorithm

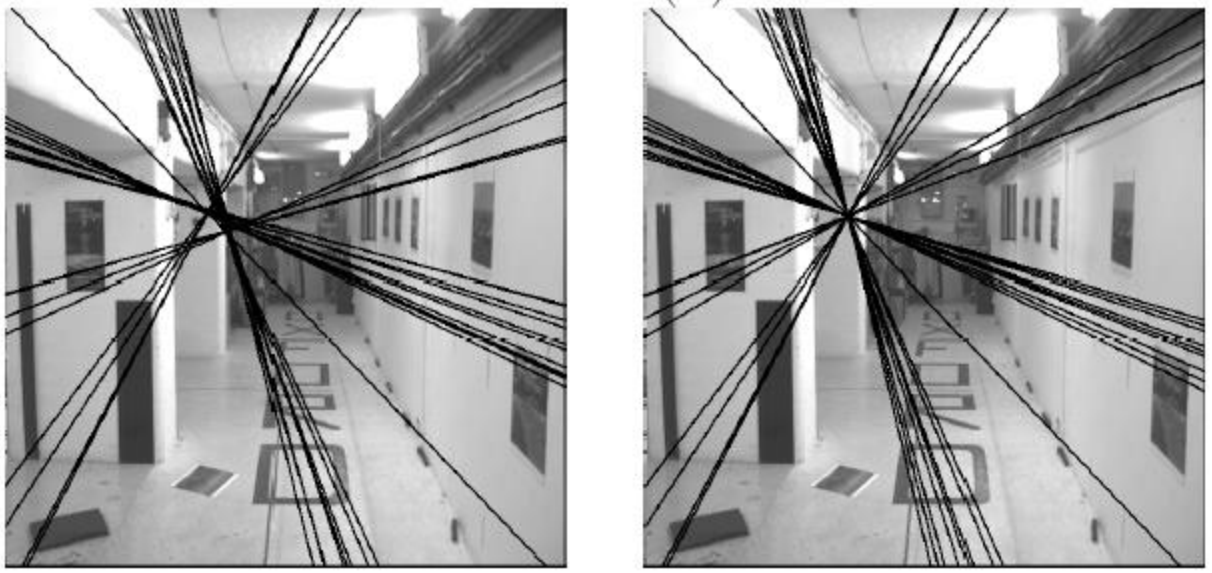
1. Solve a system of homogeneous linear equations
   a. Write down the system of equations
   b. Solve **f** from A**f**=**0** using SVD

Matlab:

```
[U, S, V] = svd(A);
f = V(:, end);
F = reshape(f, [3 3])';
```

Need to enforce singularity constraint

Fundamental matrix has rank 2 : $\det(\mathbf{F}) = 0$.



Left : Uncorrected $\mathbf{F}$ – epipolar lines are not coincident.

Right : Epipolar lines from corrected $\mathbf{F}$.

8-point algorithm

1. Solve a system of homogeneous linear equations
   a. Write down the system of equations
   b. Solve **f** from A**f**=**0** using SVD

   Matlab:
   ```
   [U, S, V] = svd(A);
   f = V(:, end);
   F = reshape(f, [3 3])';
   ```

2. Resolve det(F) = 0 constraint using SVD

   Matlab:
   ```
   [U, S, V] = svd(F);
   S(3,3) = 0;
   F = U*S*V';
   ```

Problem with eight-point algorithm

$$\begin{bmatrix} u'u & u'v & u' & v'u & v'v & v' & u & v \end{bmatrix} \begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \end{bmatrix} = -1$$

Problem with eight-point algorithm

250906.36	183269.57	921.81	200931.10	146766.13	738.21	272.19	198.81
2692.28	131633.03	176.27	6196.73	302975.59	405.71	15.27	746.79
416374.23	871684.30	935.47	408110.89	854384.92	916.90	445.10	931.81
191183.60	171759.40	410.27	416435.62	374125.90	893.65	465.99	418.65
48988.86	30401.76	57.89	298604.57	185309.58	352.87	846.22	525.15
164786.04	546559.67	813.17	1998.37	6628.15	9.86	202.65	672.14
116407.01	2727.75	138.89	169941.27	3982.21	202.77	838.12	19.64
135384.58	75411.13	198.72	411350.03	229127.78	603.79	681.28	379.48

$$\begin{bmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \end{bmatrix} = -1$$

Poor numerical conditioning

Can be fixed by rescaling the data

The normalized eight-point algorithm

(Hartley, 1995)

- Center the image data at the origin, and scale it so the mean squared distance between the origin and the data points is 2 pixels
- Use the eight-point algorithm to compute ***F*** from the normalized points
- Enforce the rank-2 constraint (for example, take SVD of ***F*** and throw out the smallest singular value)
- Transform fundamental matrix back to original units: if ***T*** and ***T'*** are the normalizing transformations in the two images, than the fundamental matrix in original coordinates is $\boldsymbol{T'^T F T}$

Comparison of estimation algorithms

	8-point	Normalized 8-point	Nonlinear least squares
Av. Dist. 1	2.33 pixels	0.92 pixel	0.86 pixel
Av. Dist. 2	2.18 pixels	0.85 pixel	0.80 pixel

From epipolar geometry to camera calibration

- If we know the calibration matrices of the two cameras, we can estimate the essential matrix: $E = K^T F K'$
- The essential matrix gives us the relative rotation and translation between the cameras, or their extrinsic parameters.
- Fundamental matrix lets us compute relationship up to scale for cameras with unknown intrinsic calibrations.
- Estimating the fundamental matrix is a kind of “weak calibration”

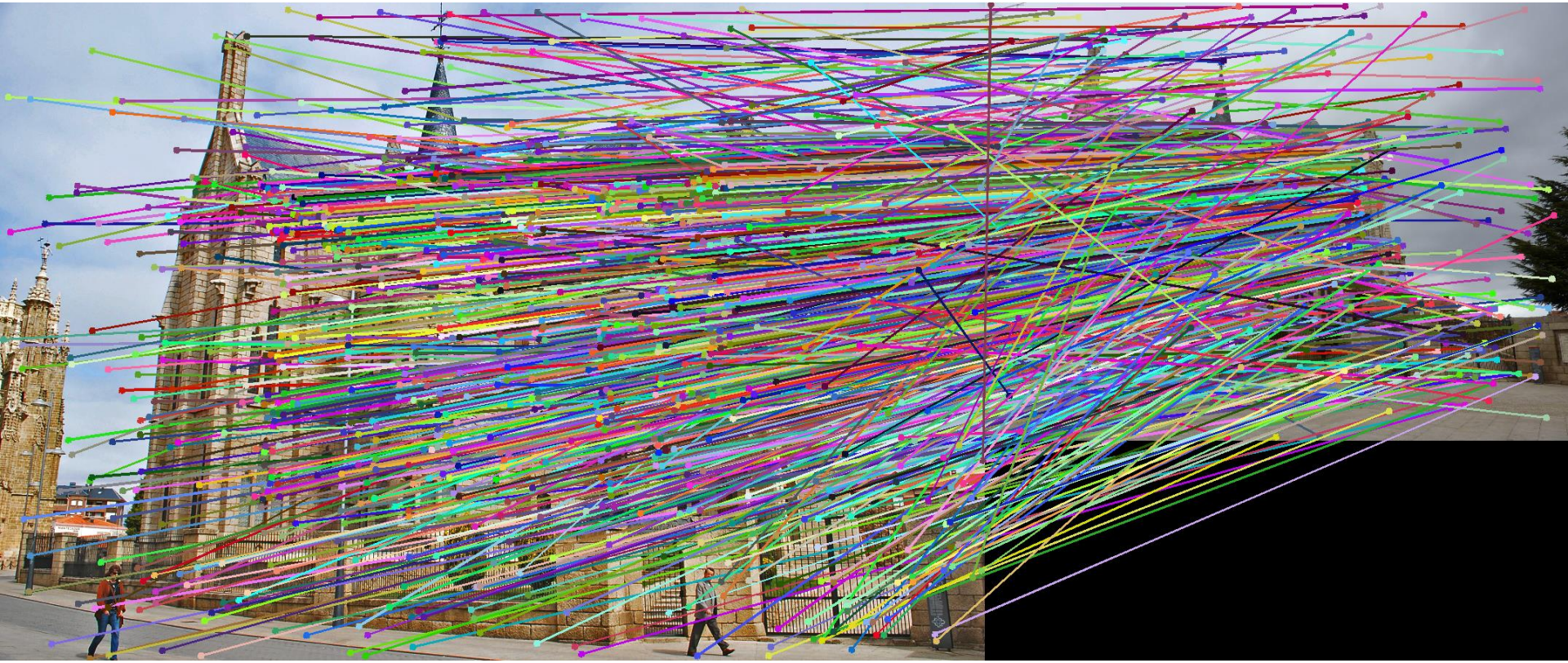
Let’s recap…

- Fundamental matrix song

- http://danielwedge.com/fmatrix/

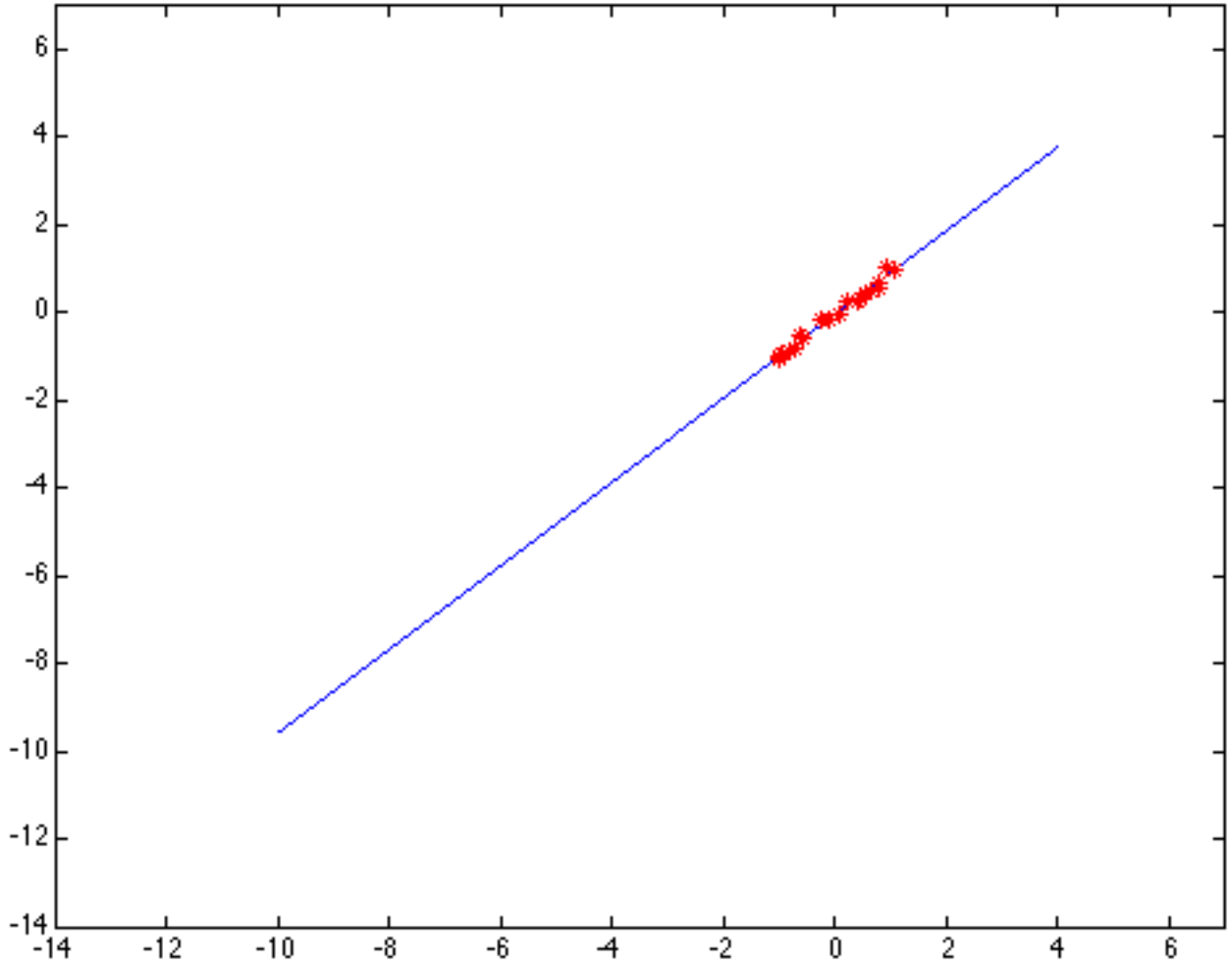
Among all my matches, how do I know which ones are good?

VLFeat's 800 most confident matches among 10,000+ local features.



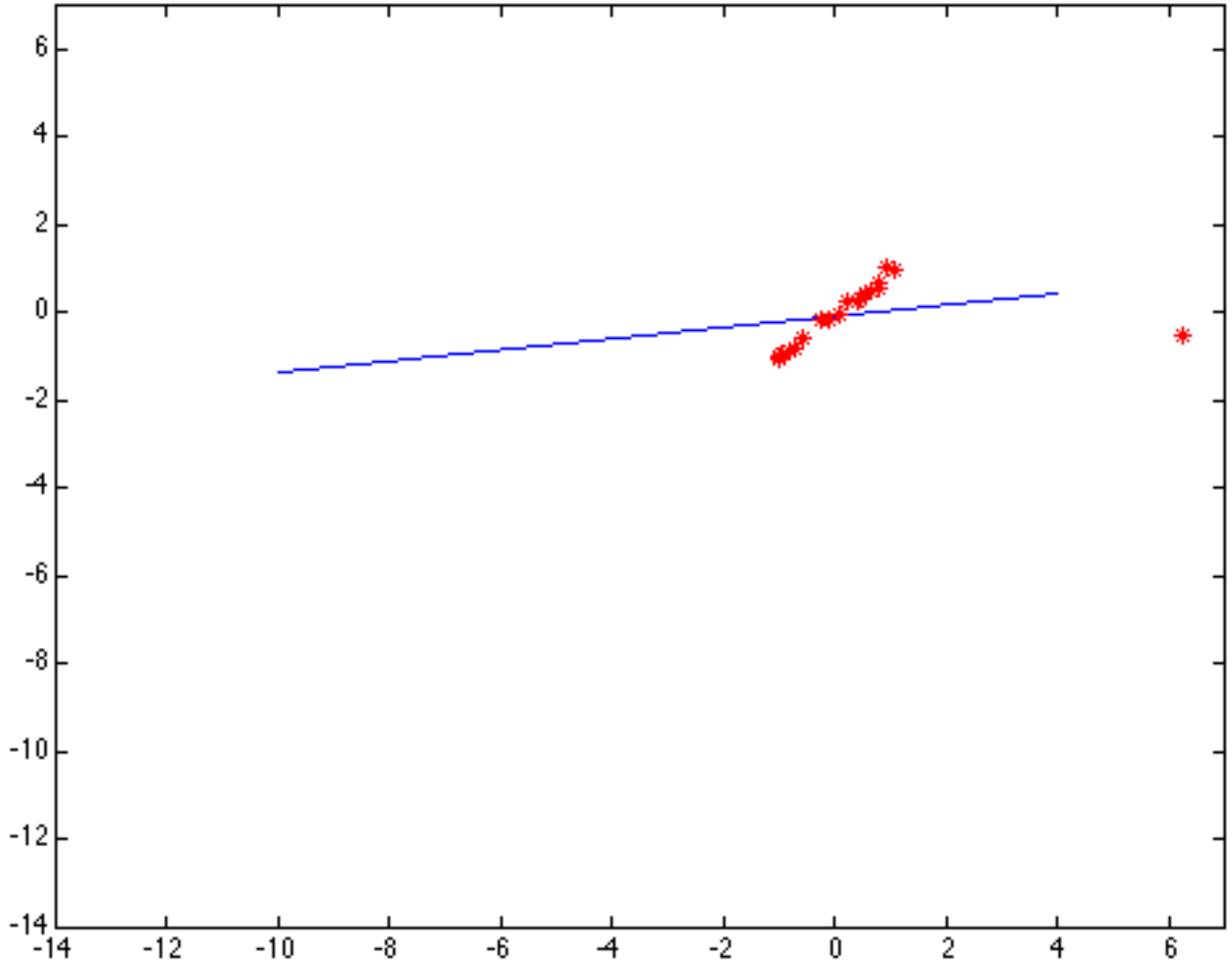
Least squares: Robustness to noise

- Least squares fit to the red points:



Least squares: Robustness to noise

- Least squares fit with an outlier:

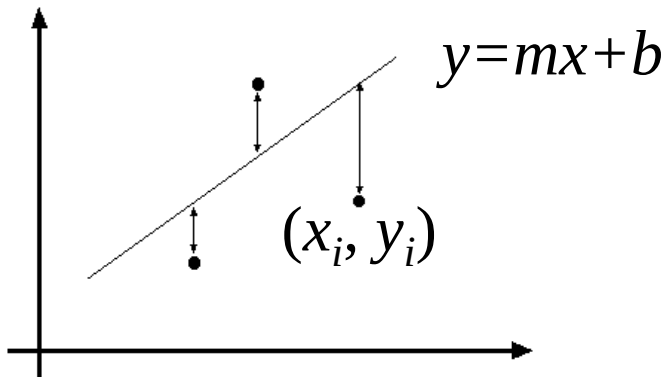


Problem: squared error heavily penalizes outliers

Least squares line fitting

- Data: $(x_1, y_1), \ldots, (x_n, y_n)$
- Line equation: $y_i = m x_i + b$
- Find (m, b) to minimize

$$E = \sum_{i=1}^{n} (y_i - mx_i - b)^2$$



Matlab: `p = A \ y;`

(Closed form solution)

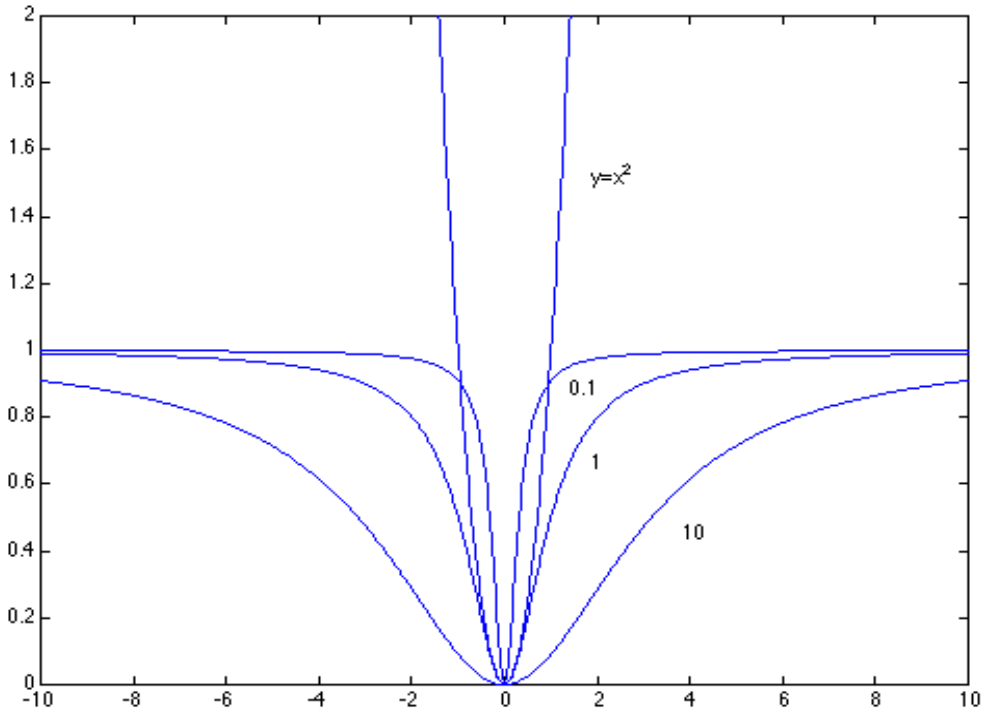
Robust least squares (to deal with outliers)

General approach:

minimize $\sum_i \rho\left(u_i\left(x_i, \theta\right); \sigma\right)$ $\qquad u^2 = \sum_{i=1}^{n} (y_i - mx_i - b)^2$

$u_i(x_i, \theta)$ – residual of ith point w.r.t. model parameters θ

ρ – robust function with scale parameter σ



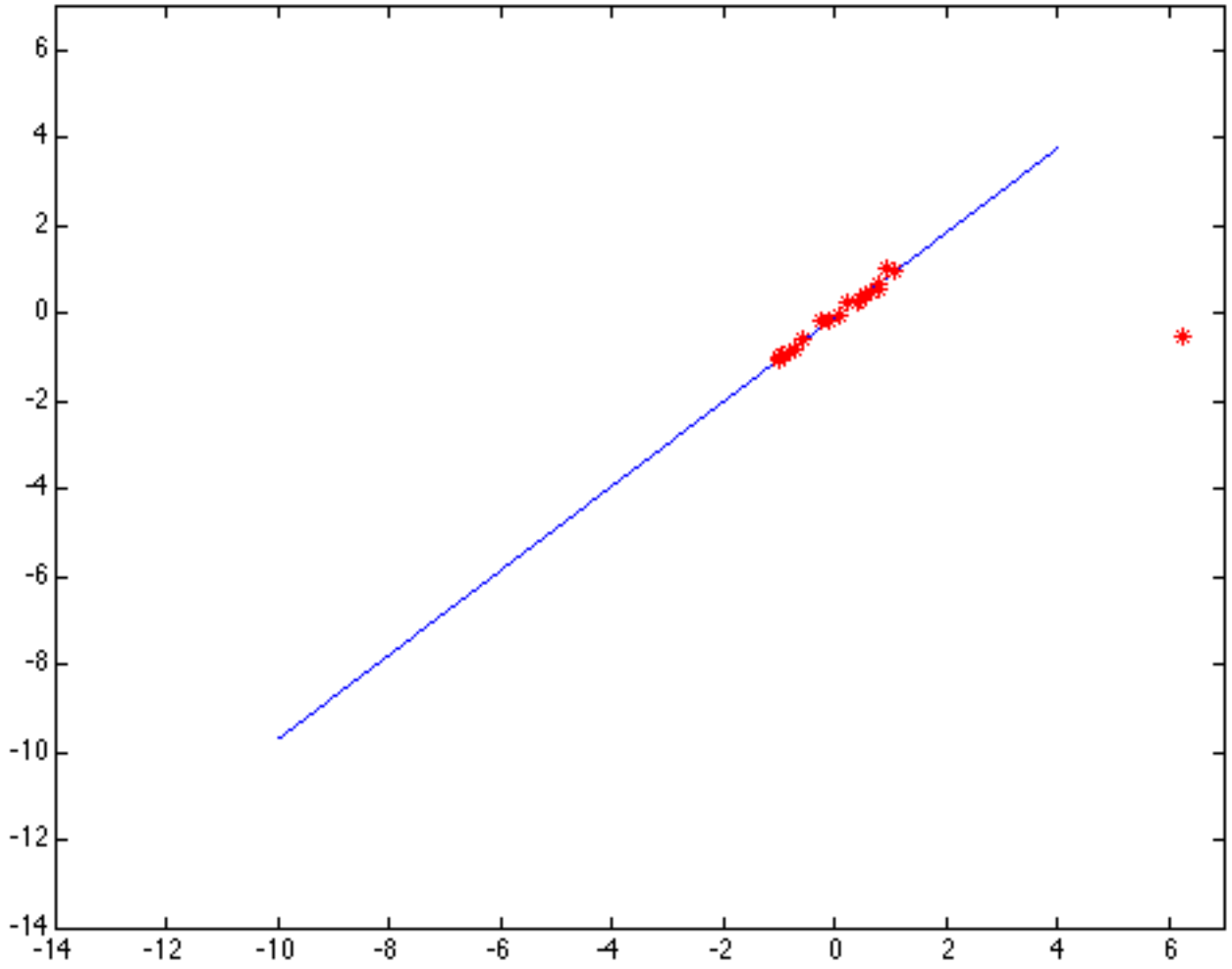
The robust function ρ

- Favors a configuration with small residuals
- Constant penalty for large residuals

$$\rho(u; \sigma) = \frac{u^2}{\sigma^2 + u^2}$$

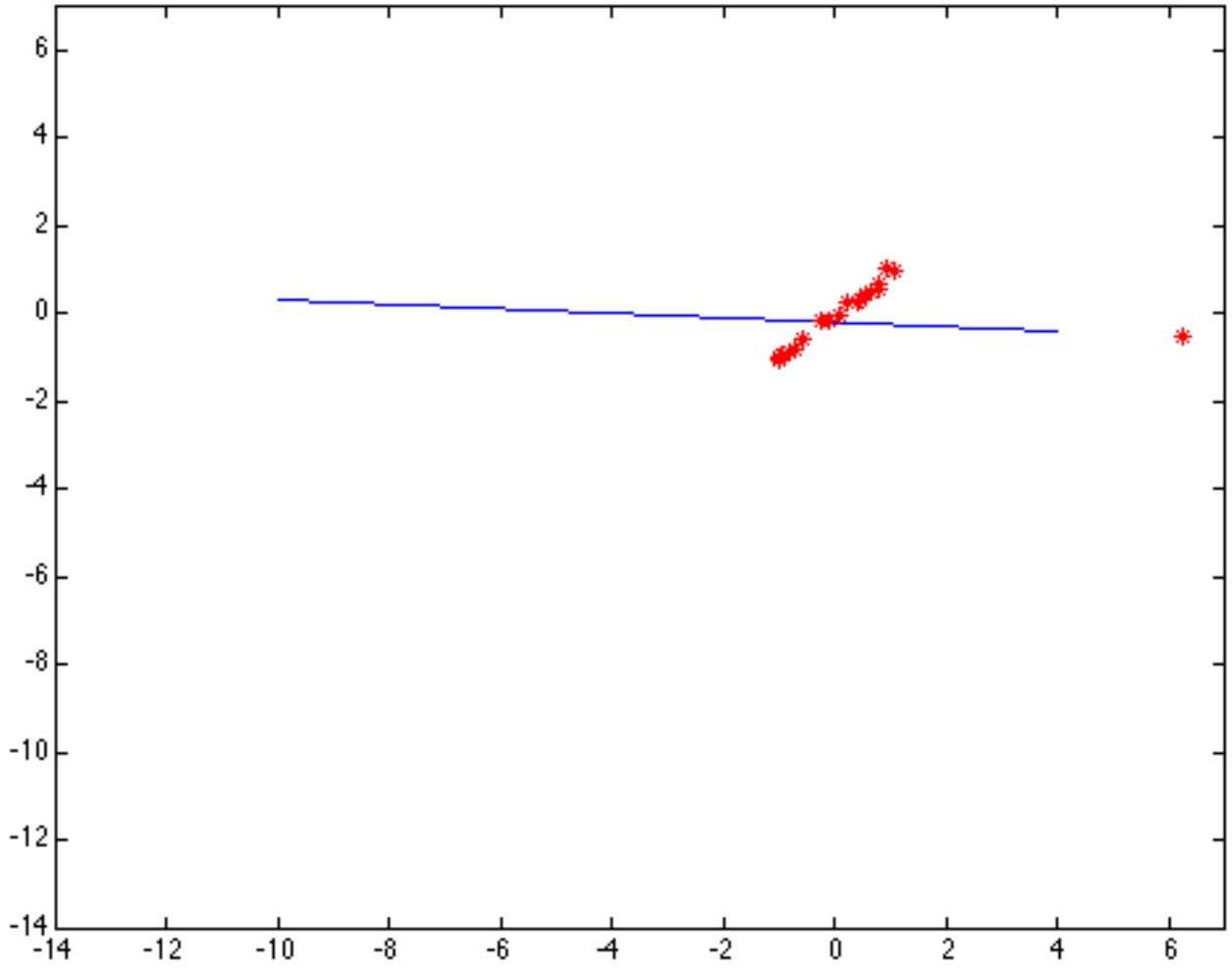
Slide from S. Savarese

Choosing the scale: Just right



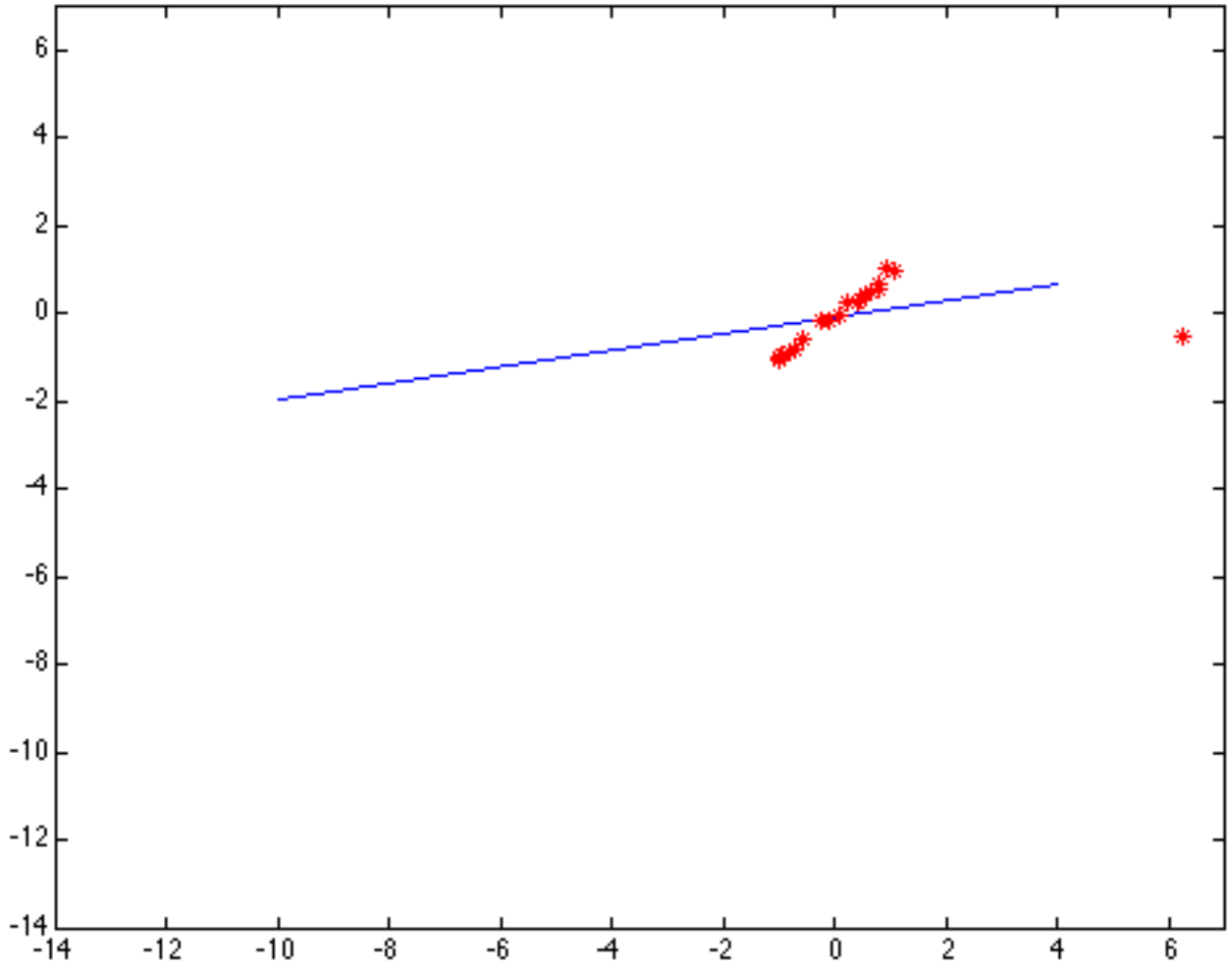
The effect of the outlier is minimized

Choosing the scale: Too small



The error value is almost the same for every point and the fit is very poor

Choosing the scale: Too large

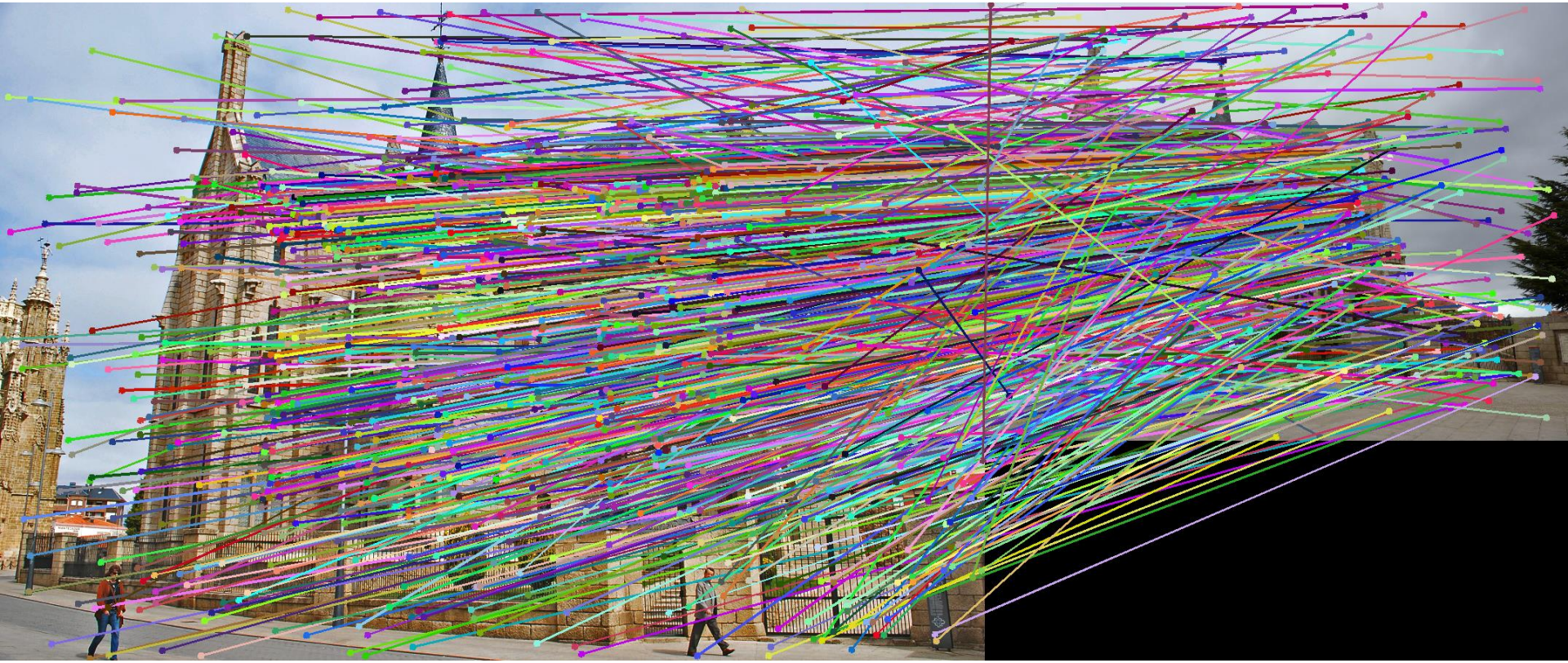


Behaves much the same as least squares

Robust estimation: Details

- Robust fitting is a nonlinear optimization problem that must be solved iteratively
- Scale of robust function should be chosen adaptively based on median residual
- Least squares solution can be used for initialization

VLFeat's 800 most confident matches among 10,000+ local features.



RANSAC

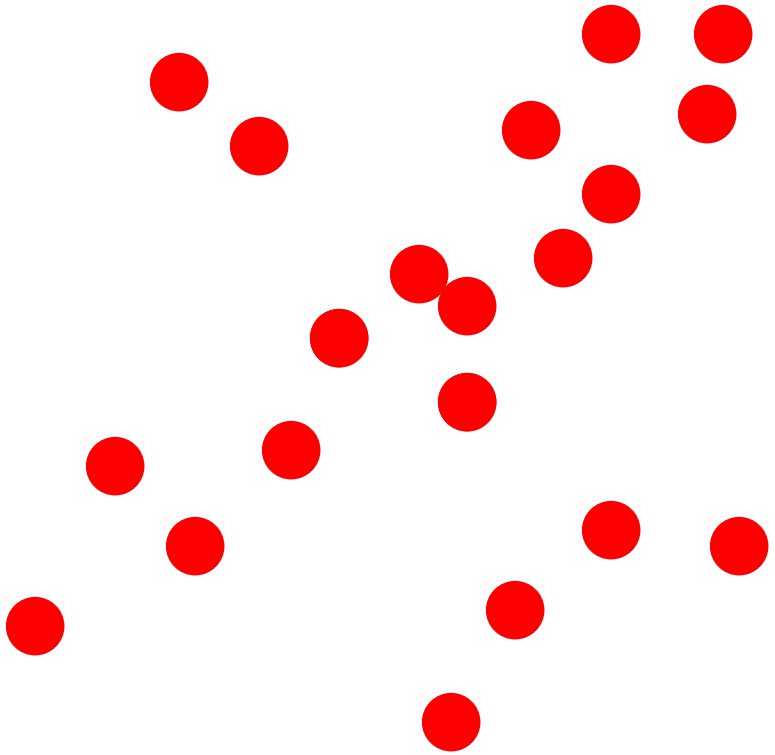
(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.

RANSAC

(RANdom SAmple Consensus) :

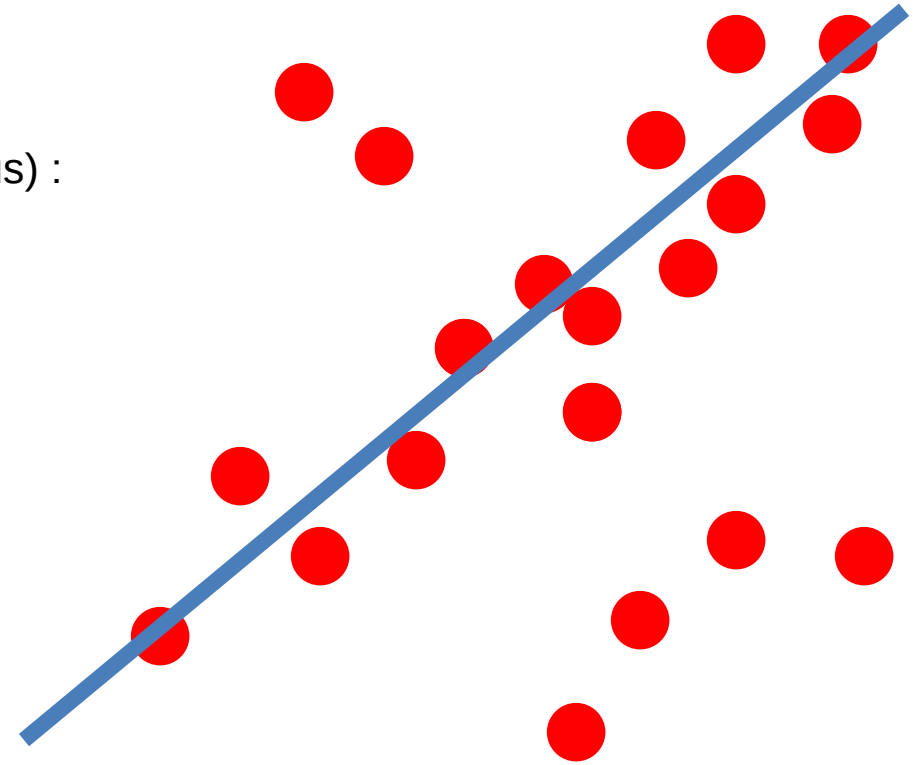
Fischler & Bolles in ‘81.



RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.

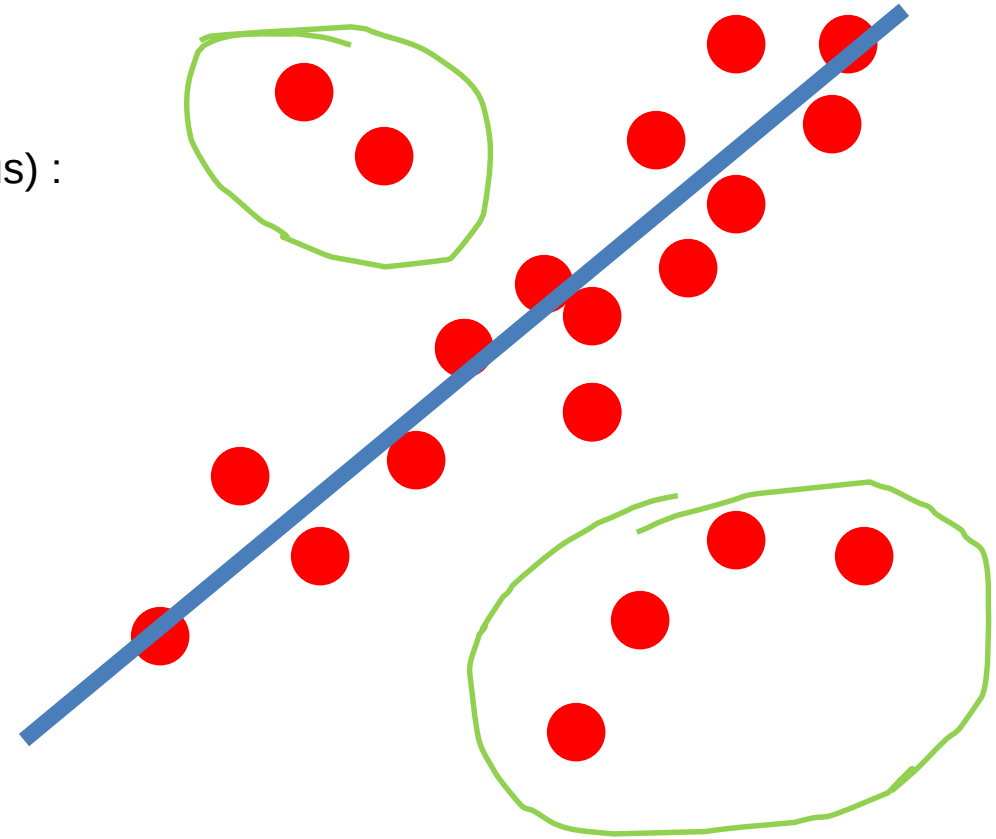


This data is noisy, but we expect a good fit to a known model.

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.



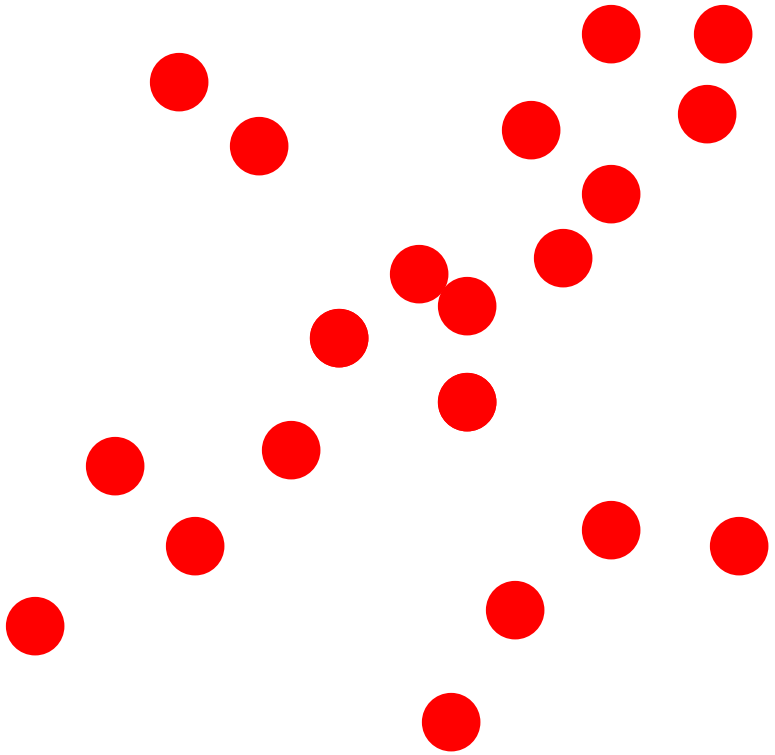
This data is noisy, but we expect a good fit to a known model.

Here, we expect to see a line, but least-squares fitting will produce the wrong result due to strong outlier presence.

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in ‘81.



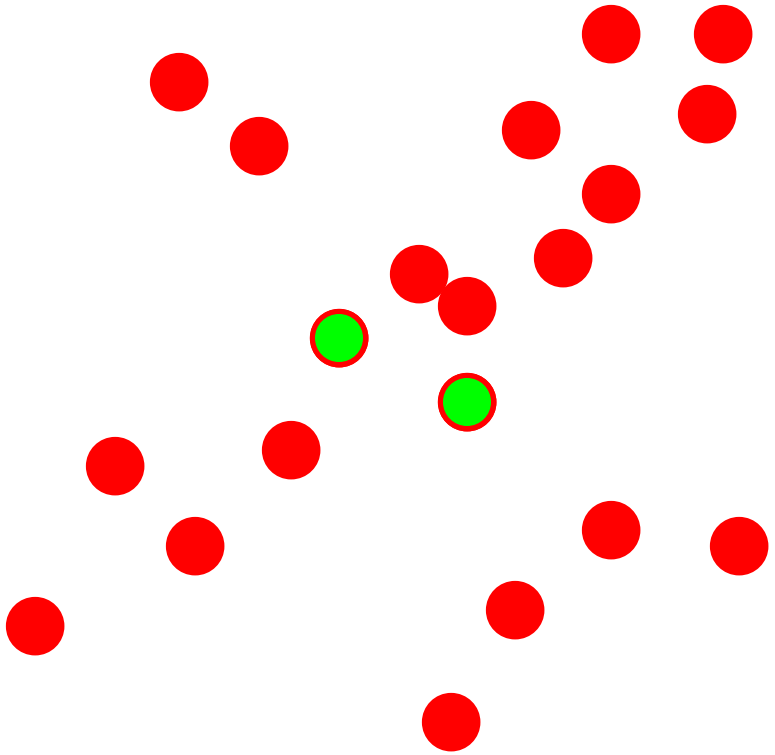
Algorithm:

1. **Sample** (randomly) the number of points *s* required to fit the model
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example



Algorithm:

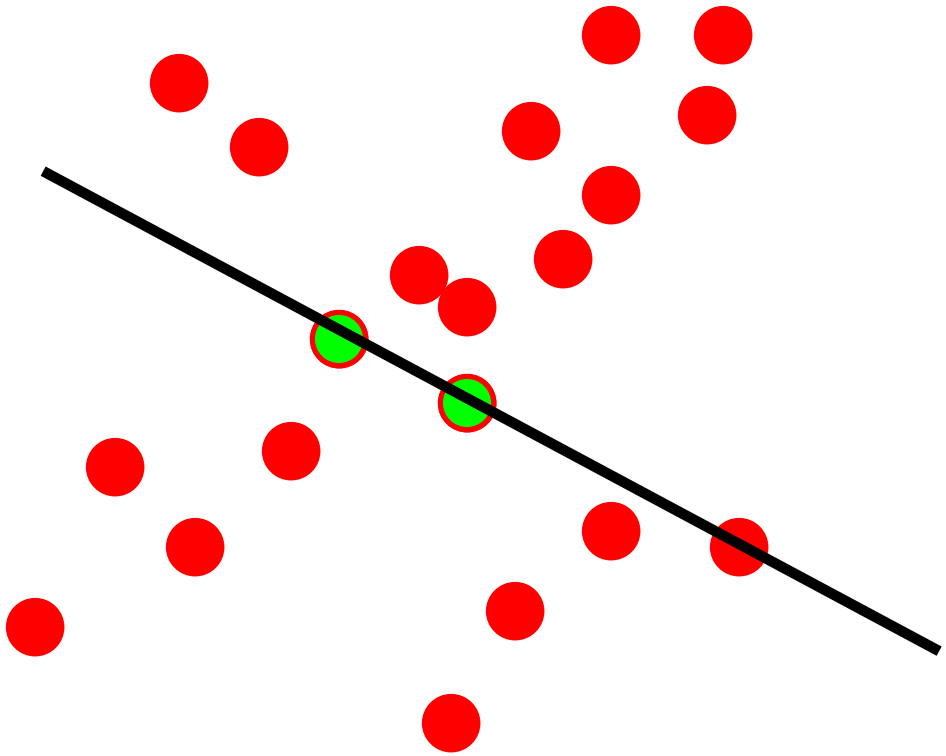
1. **Sample** (randomly) the number of points required to fit the model (s=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

Illustration by Savarese

RANSAC

Line fitting example



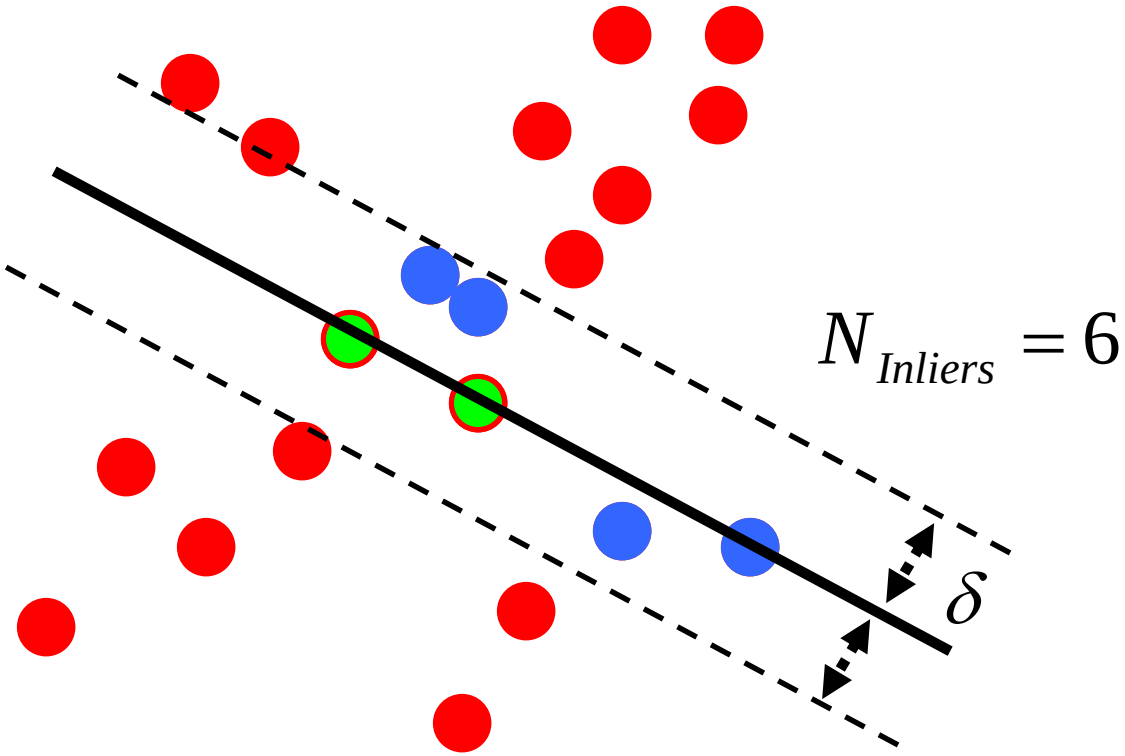
Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (*s*=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC

Line fitting example

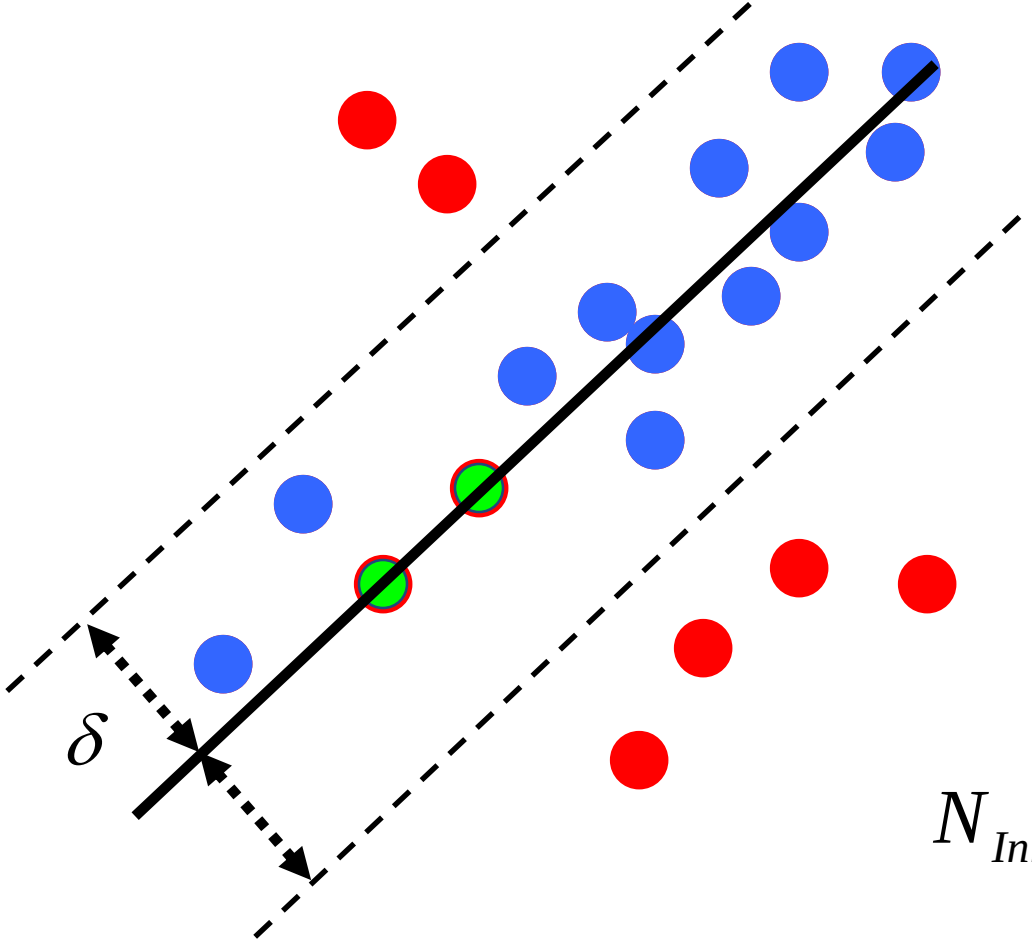


Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (*s*=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

RANSAC



$$N_{Inliers} = 14$$

Algorithm:

1. **Sample** (randomly) the number of points required to fit the model (*s*=2)
2. **Solve** for model parameters using samples
3. **Score** by the fraction of inliers within a preset threshold of the model

Repeat 1-3 until the best model is found with high confidence

How to choose parameters?

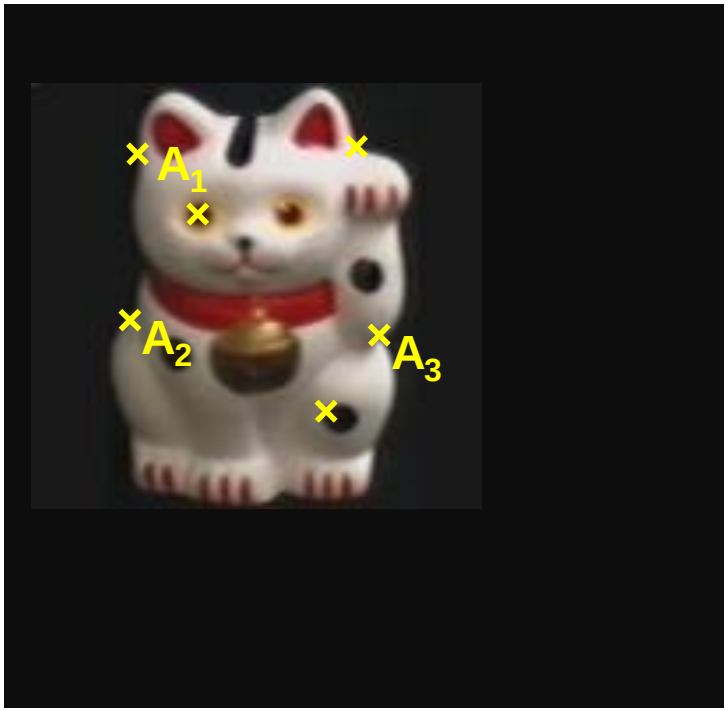
- Number of algorithm iterations *N*
  - Choose *N* so that, with probability *p*, at least one random sample is free from outliers (e.g., *p*=0.99) (outlier ratio: *e*)
- Number of sampled points *s*
  - Minimum number needed to fit the model
- Distance threshold δ
  - Choose δ so that a good point with noise is likely (e.g., prob=0.95) within threshold
  - Zero-mean Gaussian noise with std. dev. σ: $t^2=3.84\sigma^2$

$$N = \log(1-p)/\log\left(1-(1-e)^s\right)$$

	Proportion of outliers *e*						
s	5%	10%	20%	25%	30%	40%	50%
2	2	3	5	6	7	11	17
3	3	4	7	9	11	19	35
4	3	5	9	13	17	34	72
5	4	6	12	17	26	57	146
6	4	7	16	24	37	97	293
7	4	8	20	33	54	163	588
8	5	9	26	44	78	272	1177

For *p* = 0.99

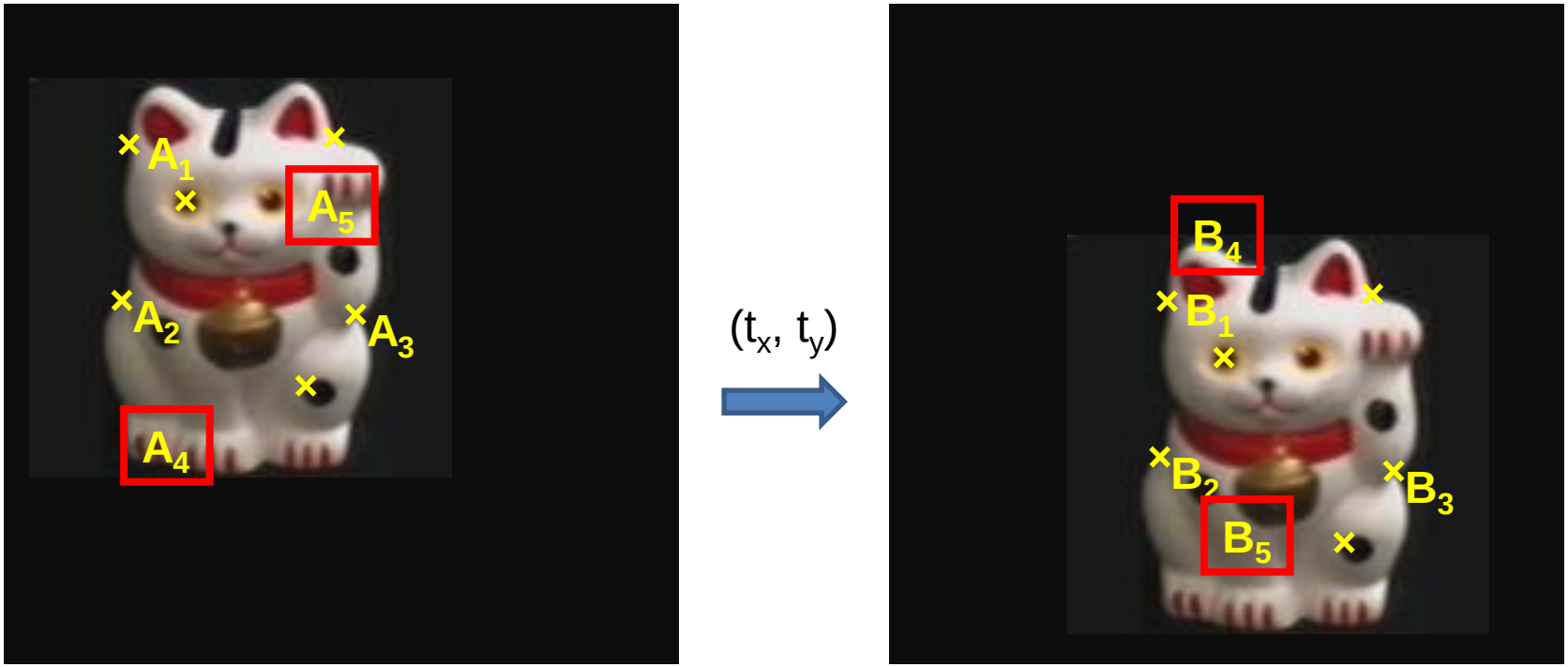
Example: solving for translation



Given matched points in {A} and {B}, estimate the translation of the object

$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Example: solving for translation



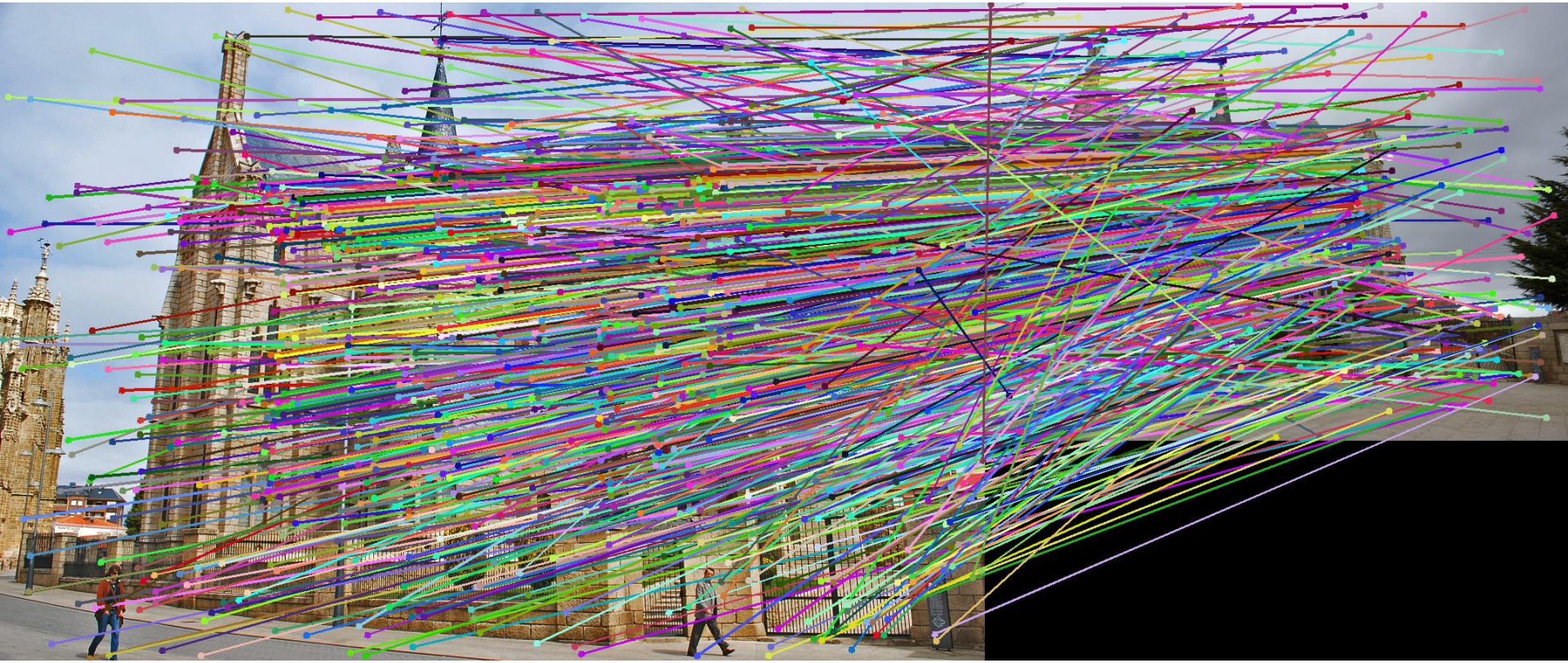
Problem: outliers

RANSAC solution

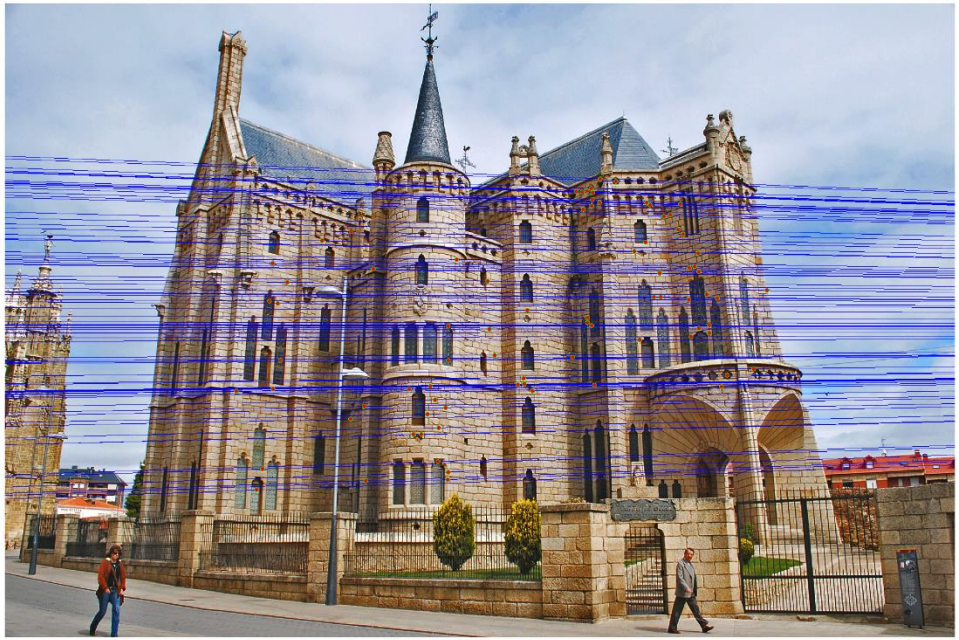
1. Sample a set of matching points (1 pair)
2. Solve for transformation parameters
3. Score parameters with number of inliers
4. Repeat steps 1-3 N times

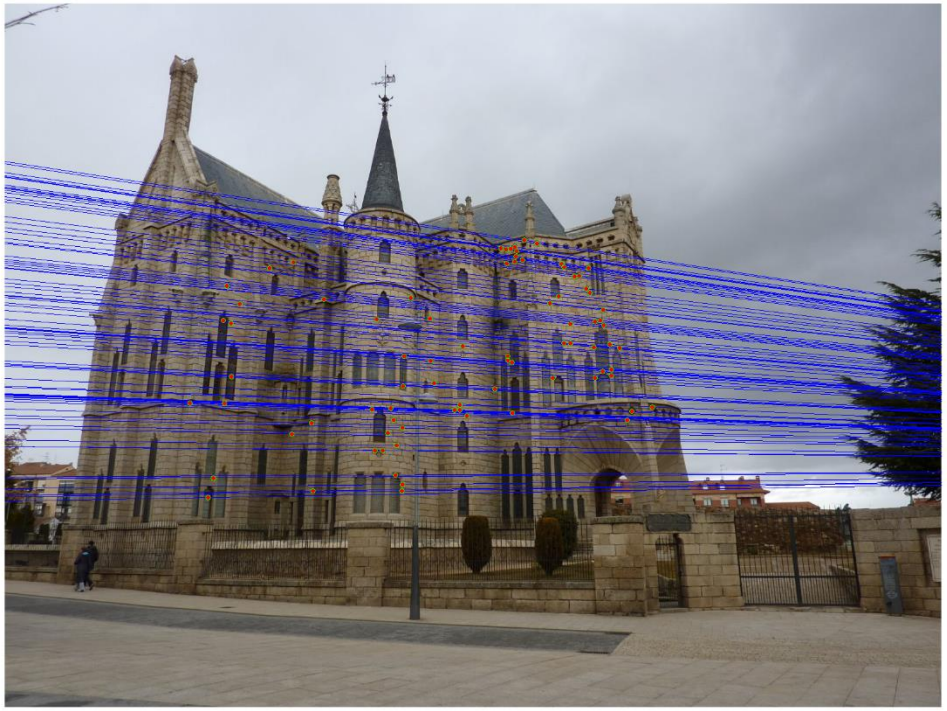
$$\begin{bmatrix} x_i^B \\ y_i^B \end{bmatrix} = \begin{bmatrix} x_i^A \\ y_i^A \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

VLFeat's 800 most confident matches among 10,000+ local features.

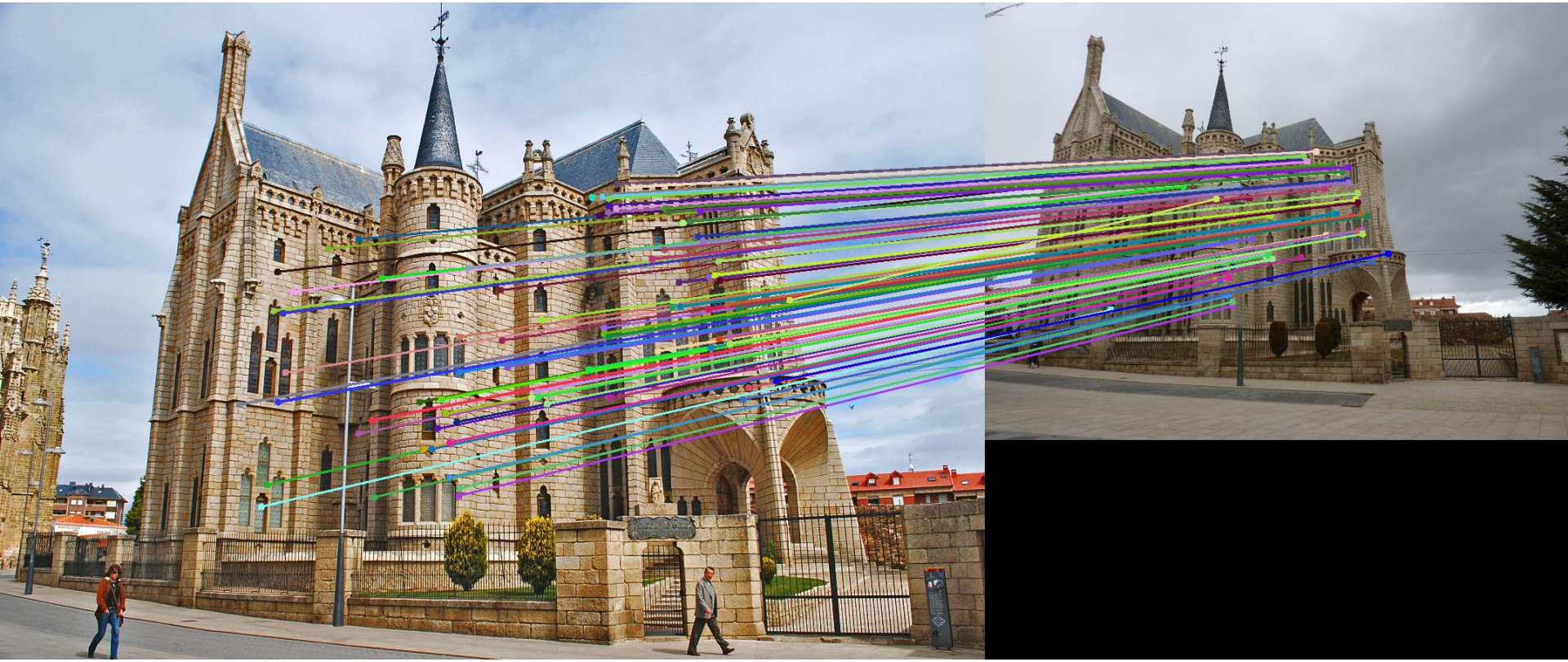


Epipolar lines





Keep only the matches at are “inliers” with respect to the “best” fundamental matrix



RANSAC conclusions

Good

- Robust to outliers
- Applicable for large number of objective function parameters (than Hough transform)
- Optimization parameters are easier to choose (than Hough transform)

Bad

- Computational time grows quickly with fraction of outliers and number of parameters
- Not good for getting multiple fits

Common applications

- Estimating fundamental matrix (relating two views)
- Computing a homography (e.g., image stitching)