Localization and SLAM II

Place Recognition and Localization

Topologic: $\{x_t, x_j\}$ or $x_t \in B$

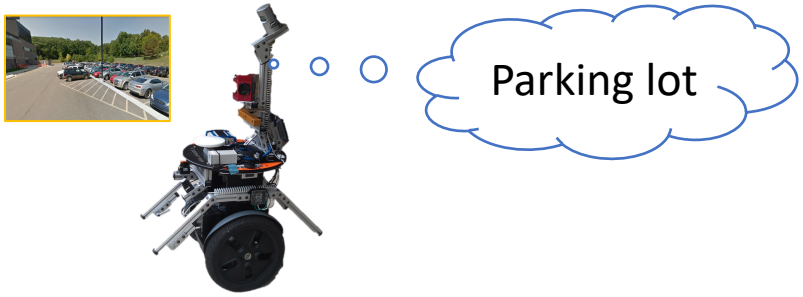
Metric: $x_t = x_j + \Delta_{tj}$

$x_t = (x, y, z, \varphi, \gamma, \vartheta)$

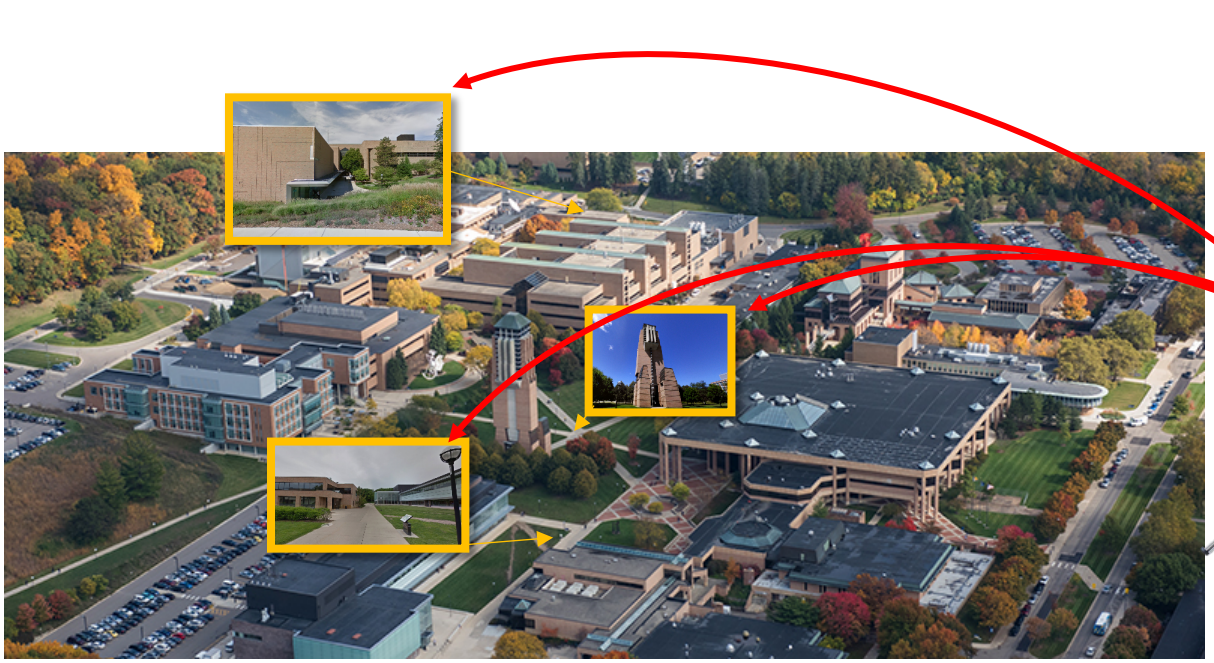
x_t: Current vehicle position

x_j: pervious vehicle positions in the map

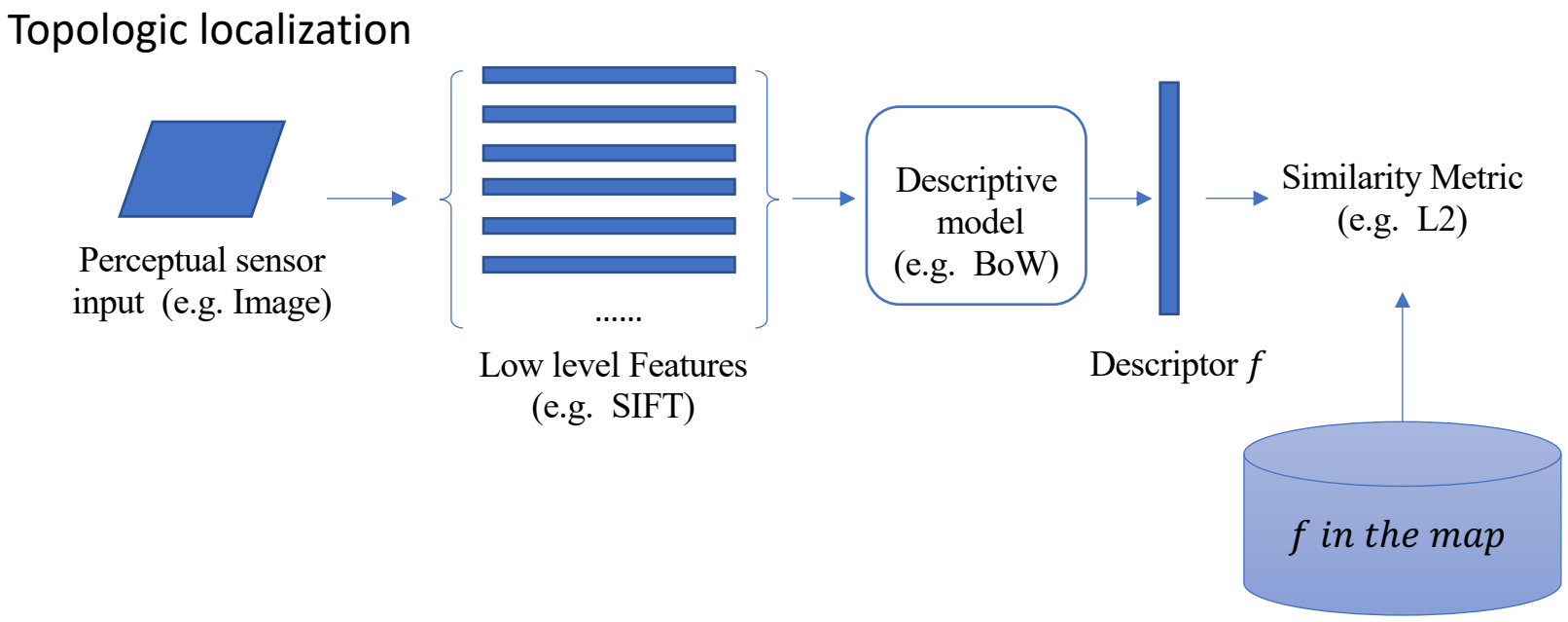
B : Location unit in a topologic map



Place recognition and Localization



Place recognition and Localization



Place Recognition and Localization

Topologic: $< x_t, x_j >$ or $x_t \in B$

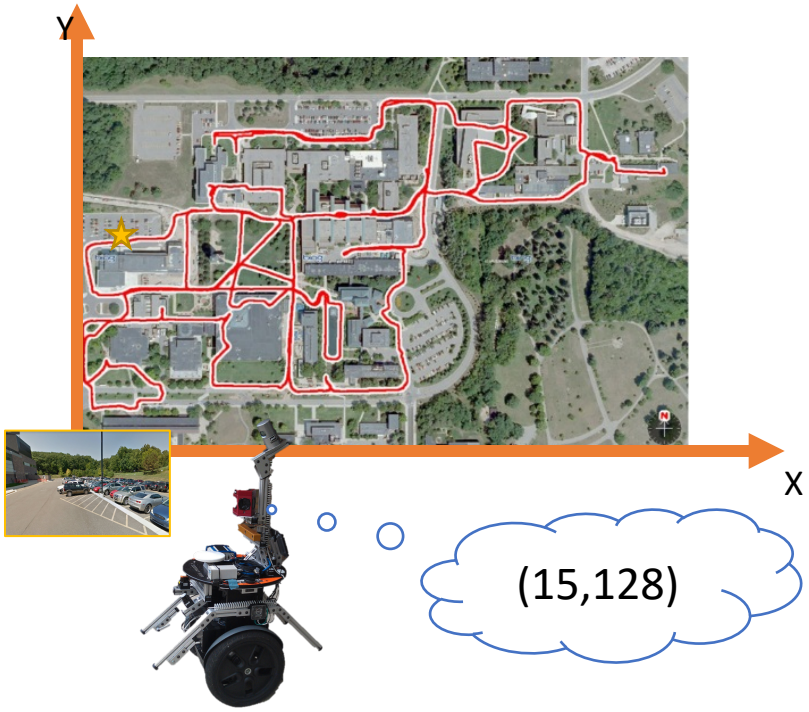
Metric: $x_t = x_j + \Delta_{tj}$

$x_t = (x, y, z, \varphi, \gamma, \vartheta)$

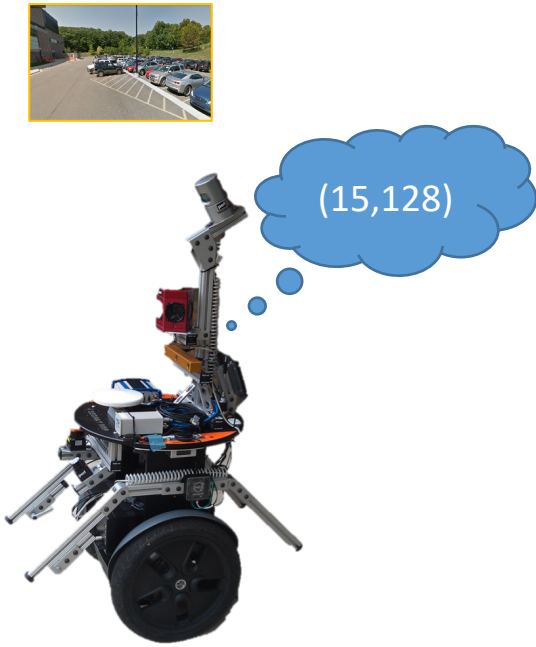
x_t: Current vehicle position

x_j: Pervious vehicle positions in the map

B: Location unit in a topologic map



Place recognition and Localization



Place Recognition and Localization

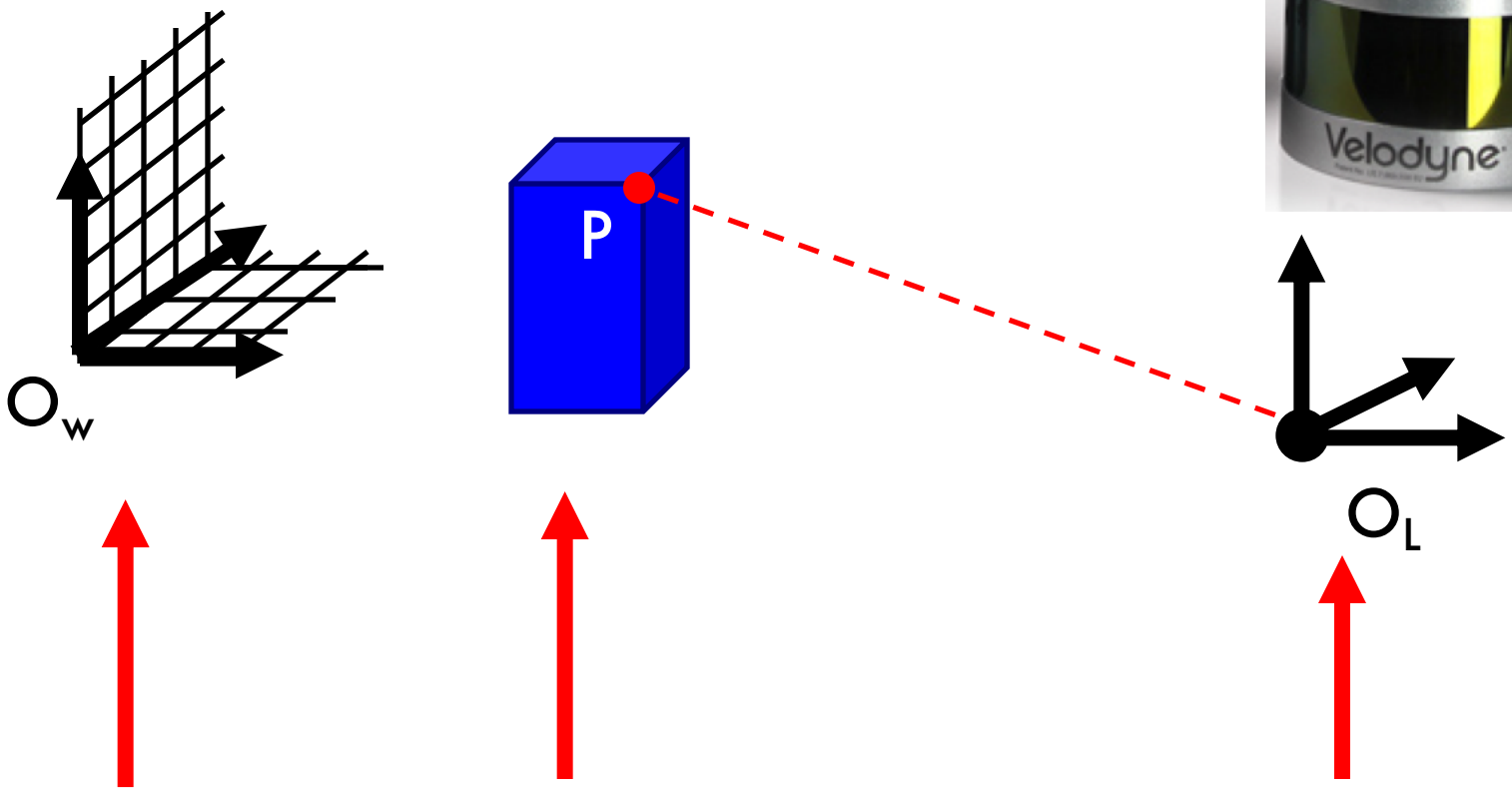
Metric localization

$$\mathrm{X}^{*} = \arg\max_{X} p(X \mid Z)$$

X: Vehicle locations
Z: Sensor inputs

Exteroceptive Sensors

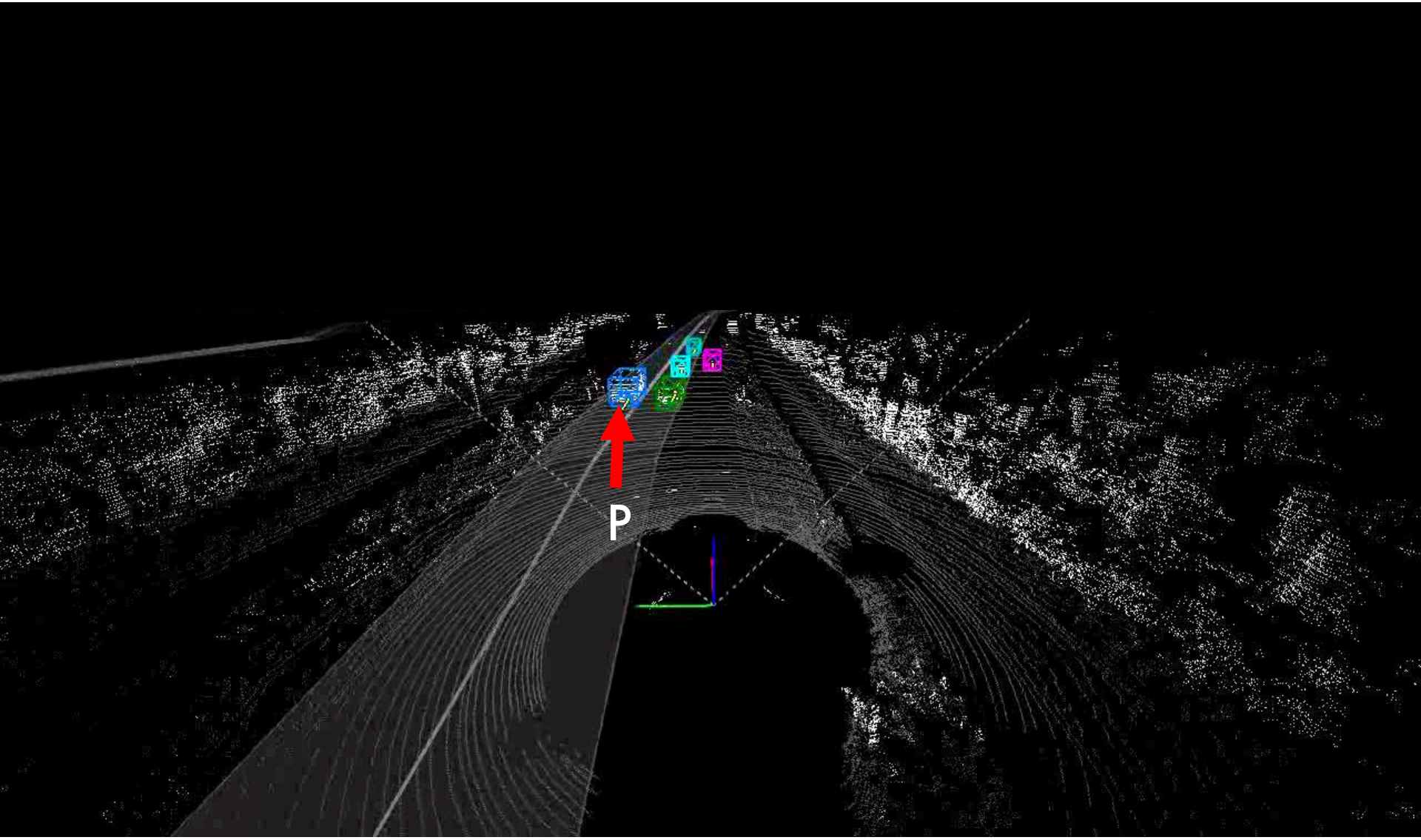
LIDAR



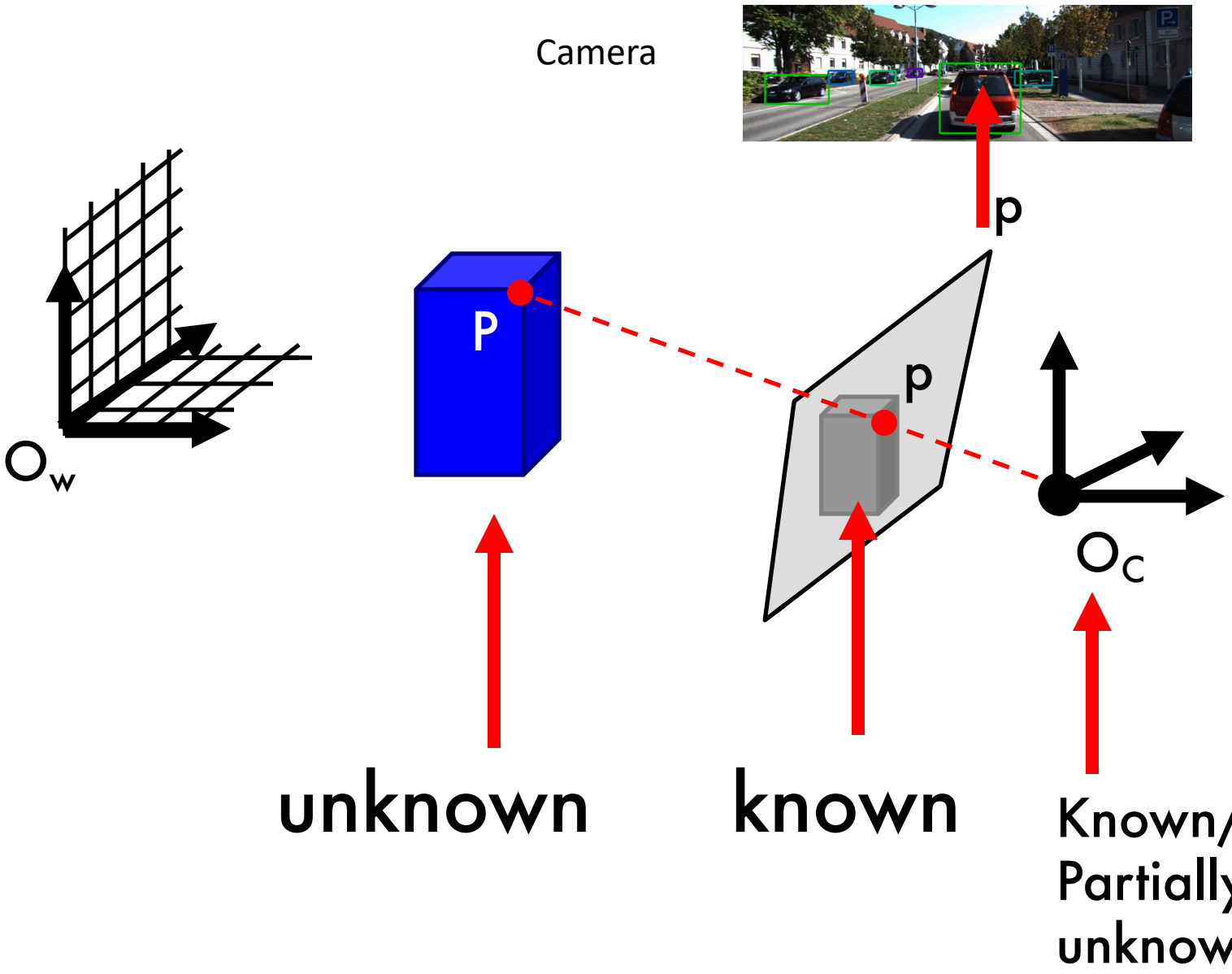
known

Known

$O_w = RT\ O_L$

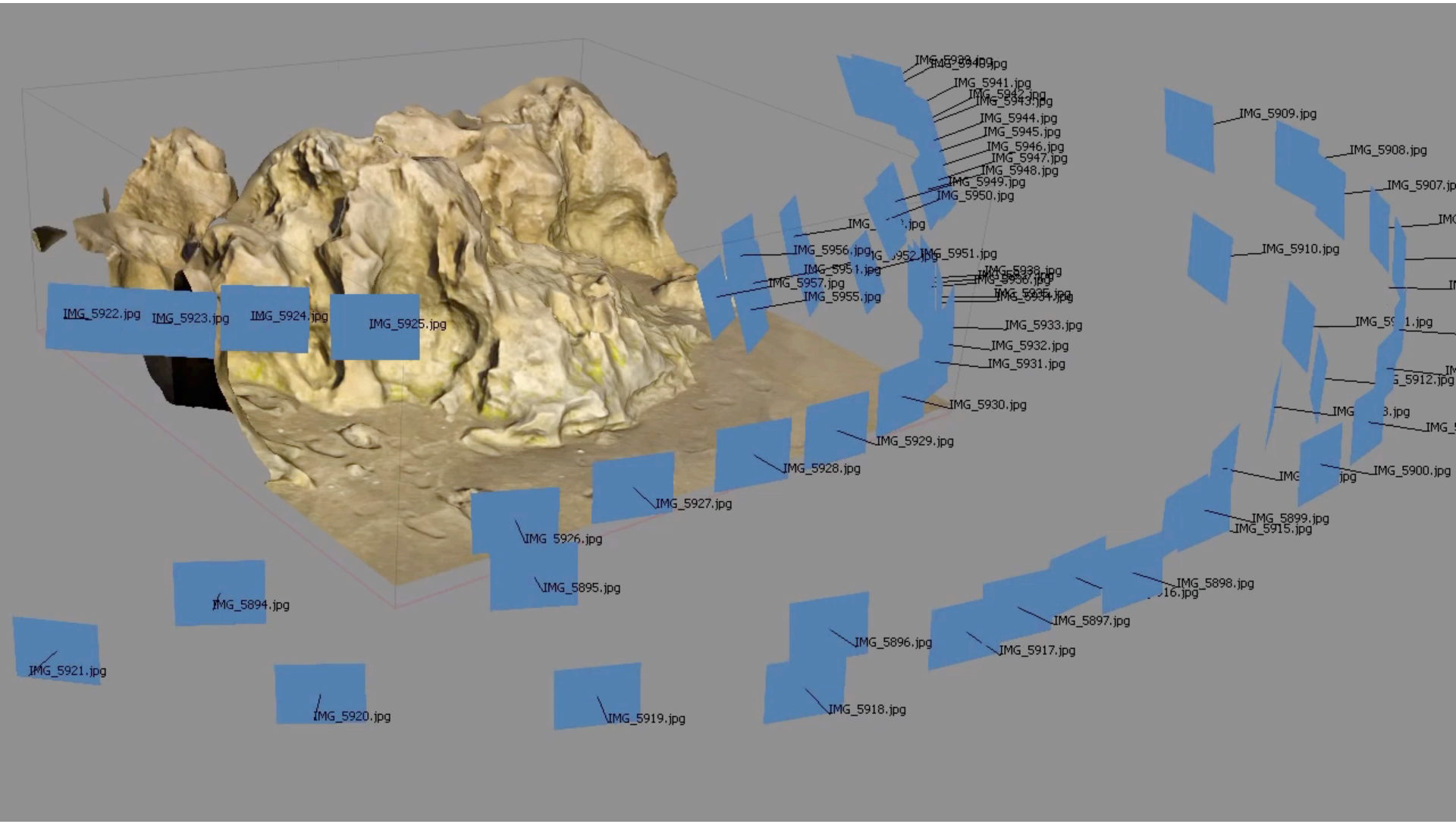

P

Point-based localization

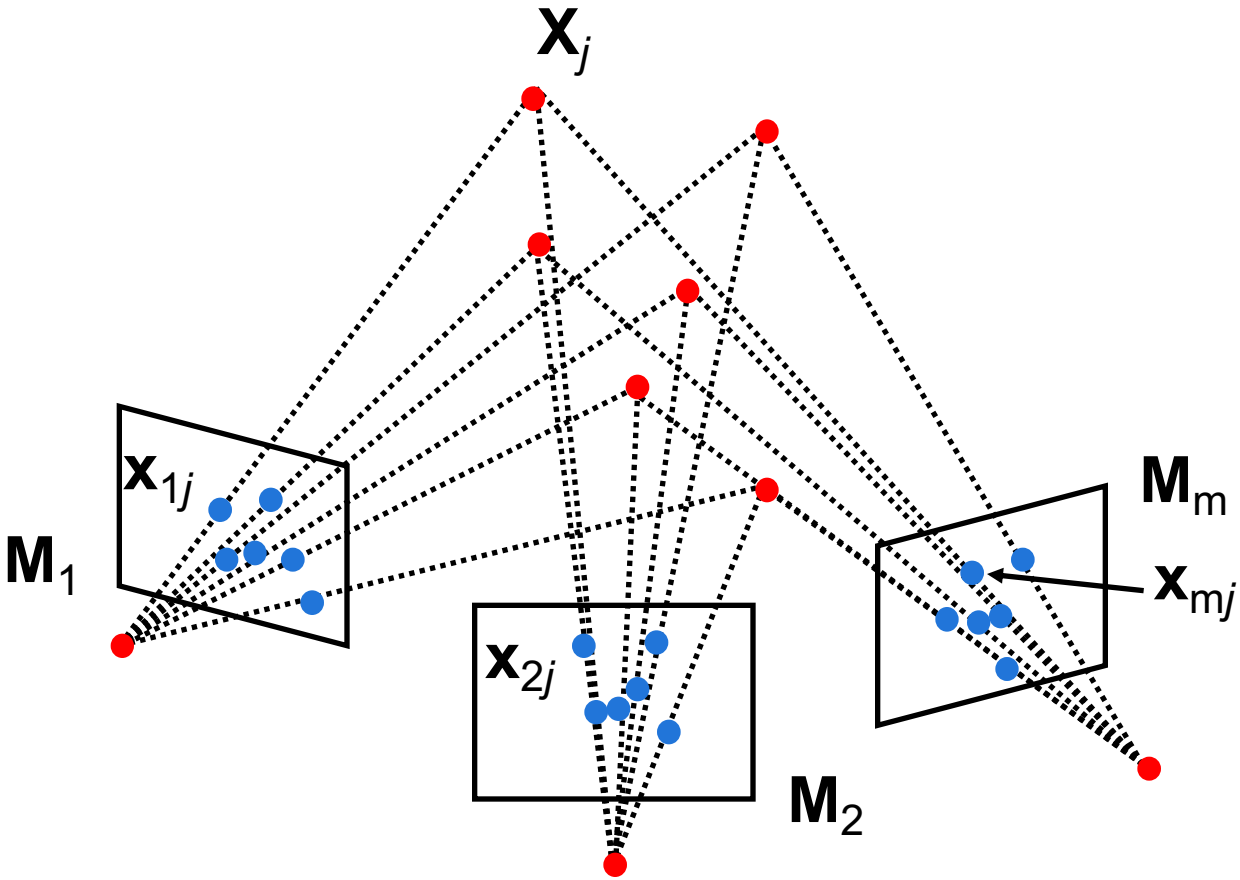

Camera
p
P
p
O_w
O_C
unknown
known
Known/
Partially known/
unknown

Camera-based localization

What if we want metric localization not topologic?


IMG_5922.jpg
IMG_5923.jpg
IMG_5924.jpg
IMG_5925.jpg
IMG_5926.jpg
IMG_5927.jpg
IMG_5928.jpg
IMG_5929.jpg
IMG_5930.jpg
IMG_5931.jpg
IMG_5932.jpg
IMG_5933.jpg
IMG_5955.jpg
IMG_5957.jpg
IMG_5956.jpg
IMG_5950.jpg
IMG_5949.jpg
IMG_5948.jpg
IMG_5947.jpg
IMG_5946.jpg
IMG_5945.jpg
IMG_5944.jpg
IMG_5941.jpg
IMG_5909.jpg
IMG_5908.jpg
IMG_5910.jpg
IMG_5912.jpg
IMG_5900.jpg
IMG_5899.jpg
IMG_5915.jpg
IMG_5898.jpg
IMG_5897.jpg
IMG_5917.jpg
IMG_5896.jpg
IMG_5918.jpg
IMG_5895.jpg
IMG_5894.jpg
IMG_5921.jpg
IMG_5920.jpg
IMG_5919.jpg

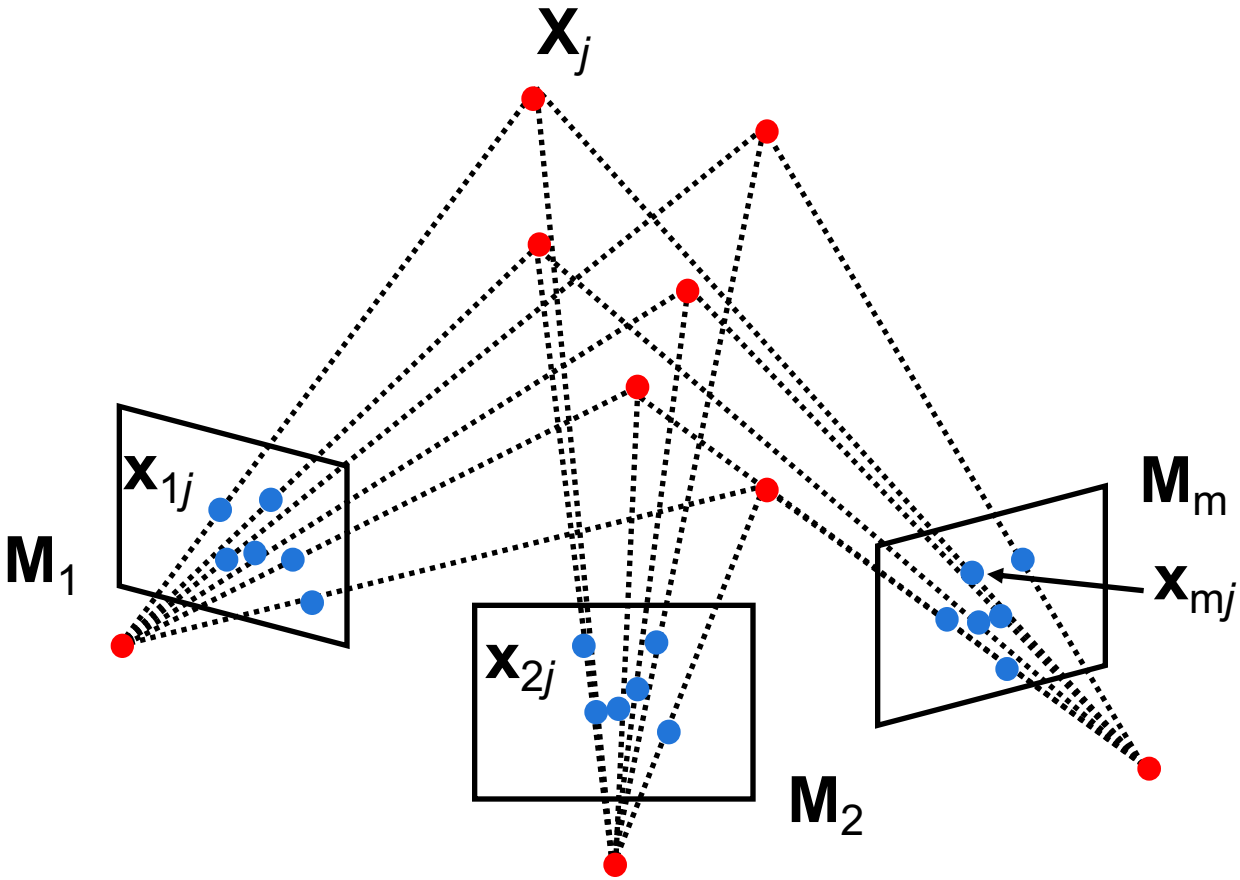
Structure from motion problem



Given *m* images of *n* fixed 3D points

- $\mathbf{x}_{ij} = \mathbf{M}_i \mathbf{X}_j, \quad i = 1, \ldots, m, \quad j = 1, \ldots, n$

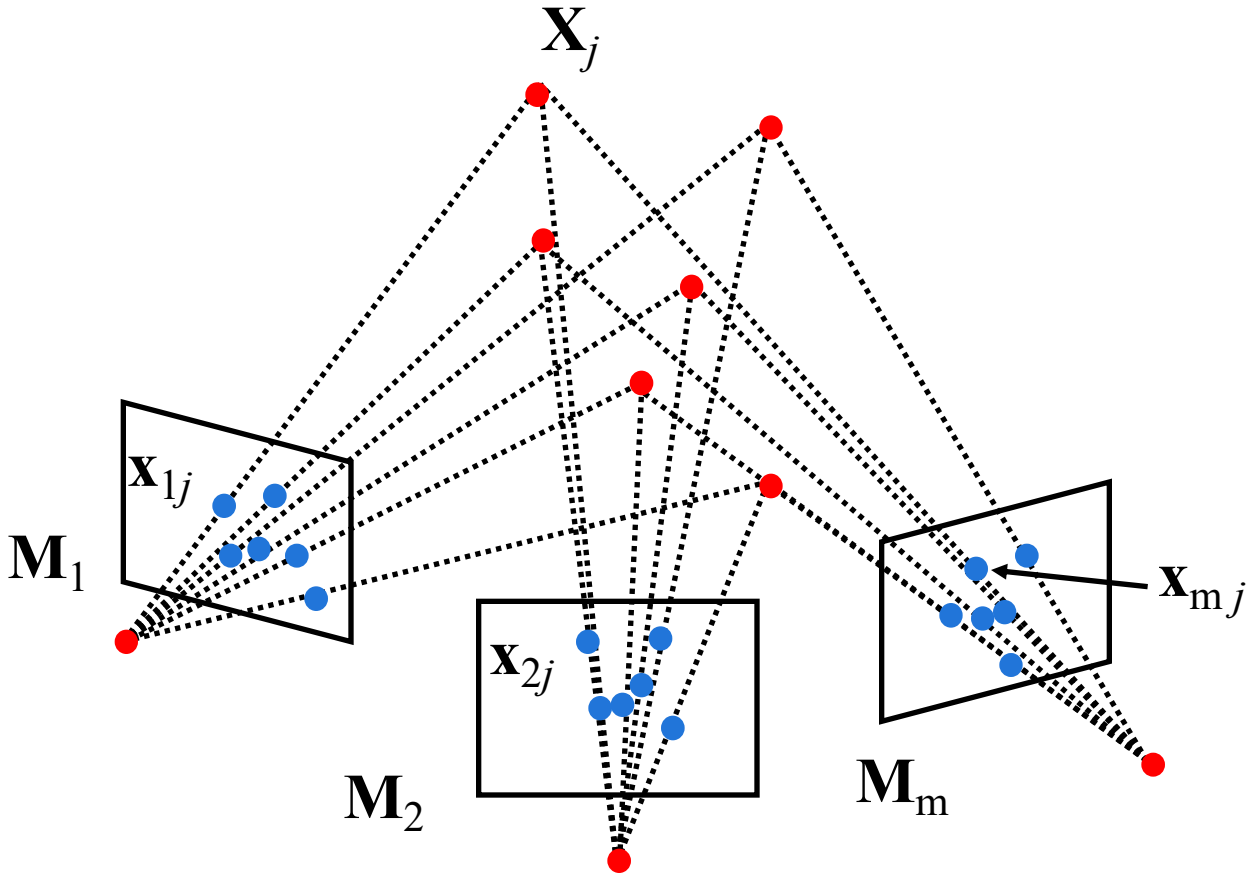
Structure from motion problem



From the mxn correspondences $\mathbf{x}_{ij}$, estimate:

- *m* projection matrices $\mathbf{M}_i$ — motion
- *n* 3D points $\mathbf{X}_j$ — structure

Structure from motion problem



m cameras $M_1 \ldots M_m$

$$M_i = \begin{bmatrix} a_{11} & a_{12} & a_{13} & b_1 \\ a_{21} & a_{22} & a_{23} & b_2 \\ a_{31} & a_{32} & a_{33} & 1 \end{bmatrix}$$

The Structure-from-Motion Problem

Given *m* images of *n* fixed points X_j we can write

$$x_{ij} = M_i X_j \quad \text{for} \quad i = 1, \ldots, m \quad \text{and} \quad j = 1, \ldots, n.$$

Problem: estimate the m 3×4 matrices M_i and the n positions X_j from the $m\times n$ correspondences x_{ij} .

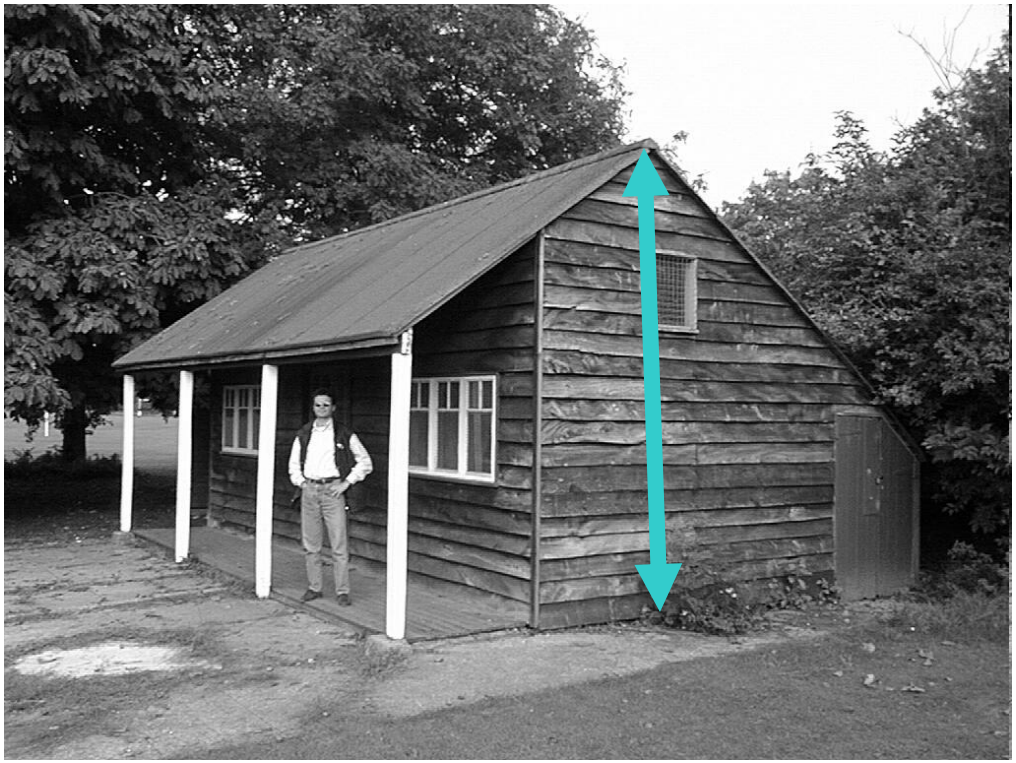
- With no calibration info, cameras and points can only be recovered up to a 4x4 projective (15 parameters)
- Given two cameras, how many points are needed?
- How many equations and how many unknown?

$2m \times n$ equations in $11m+3n - 15$ unknowns

So 7 points! [2x2x7 = 28; 11x2 + 3x7 – 15 = 28]

Structure from motion ambiguity

-Scale ambiguity: it is impossible based on the images alone to estimate the absolute scale of the scene (i.e. house height)



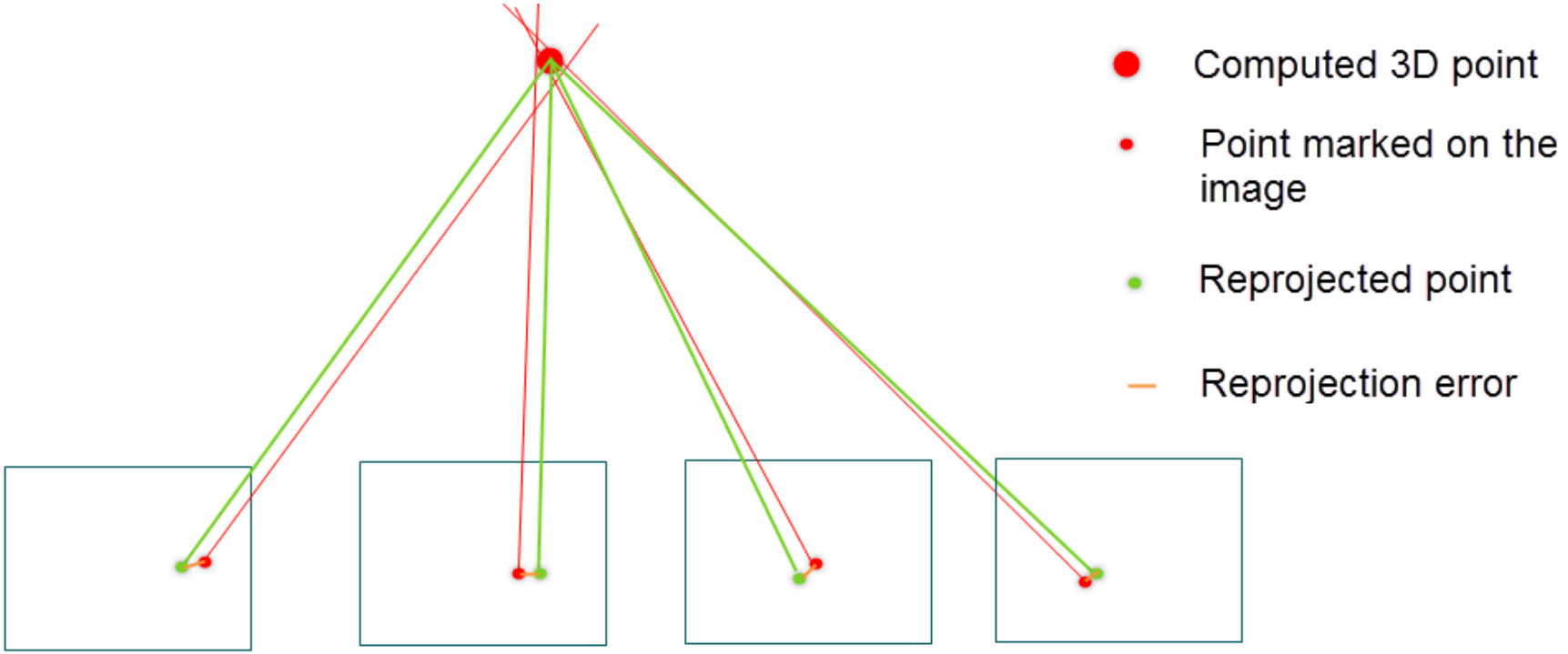
Structure from motion ambiguity

- If we scale the entire scene by some factor *k* and, at the same time, scale the camera matrices by the factor of 1/*k*, the projections of the scene points in the image remain exactly the same:

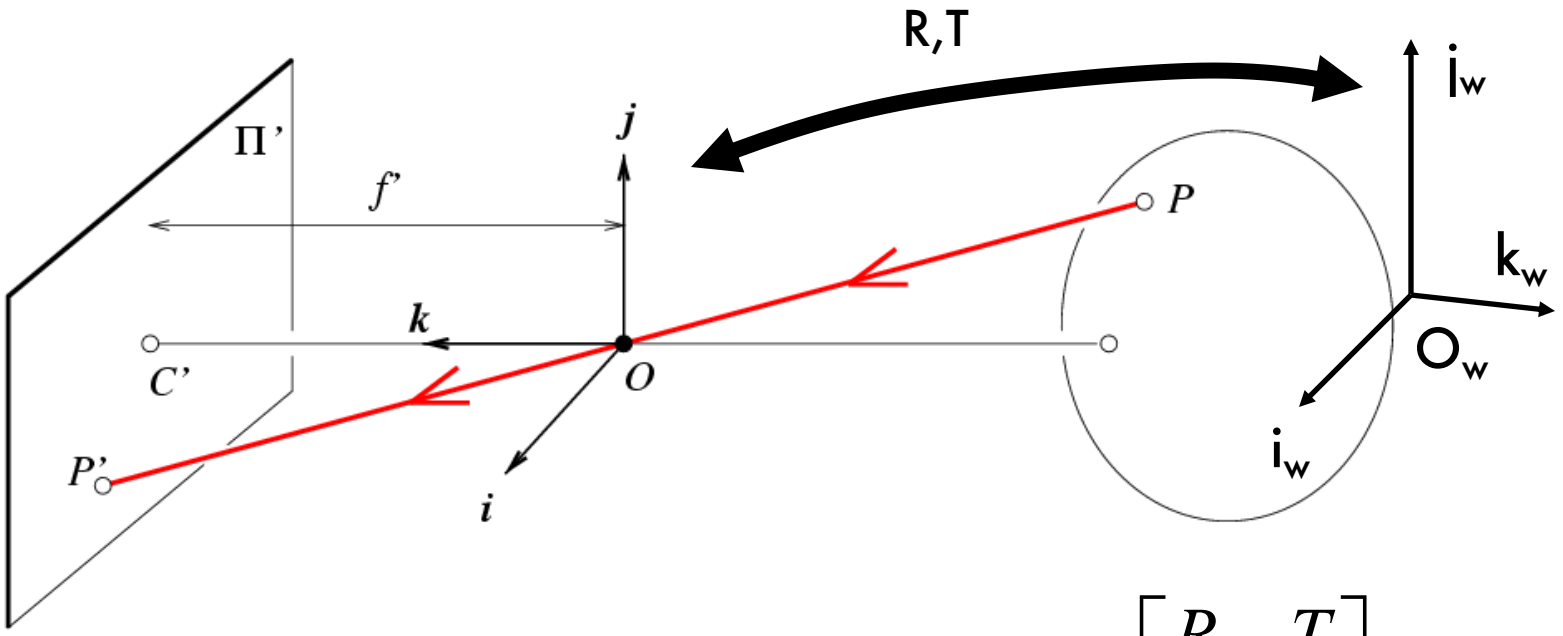
$$\mathbf{x} = \mathbf{M}\mathbf{X} = \left(\frac{1}{k}\mathbf{M}\right)(k\mathbf{X})$$

It is impossible to recover the absolute scale of the scene!

Reprojection error



World reference system



In 4D homogeneous coordinates: $X = \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}_{4\times4} X_w$

Internal parameters

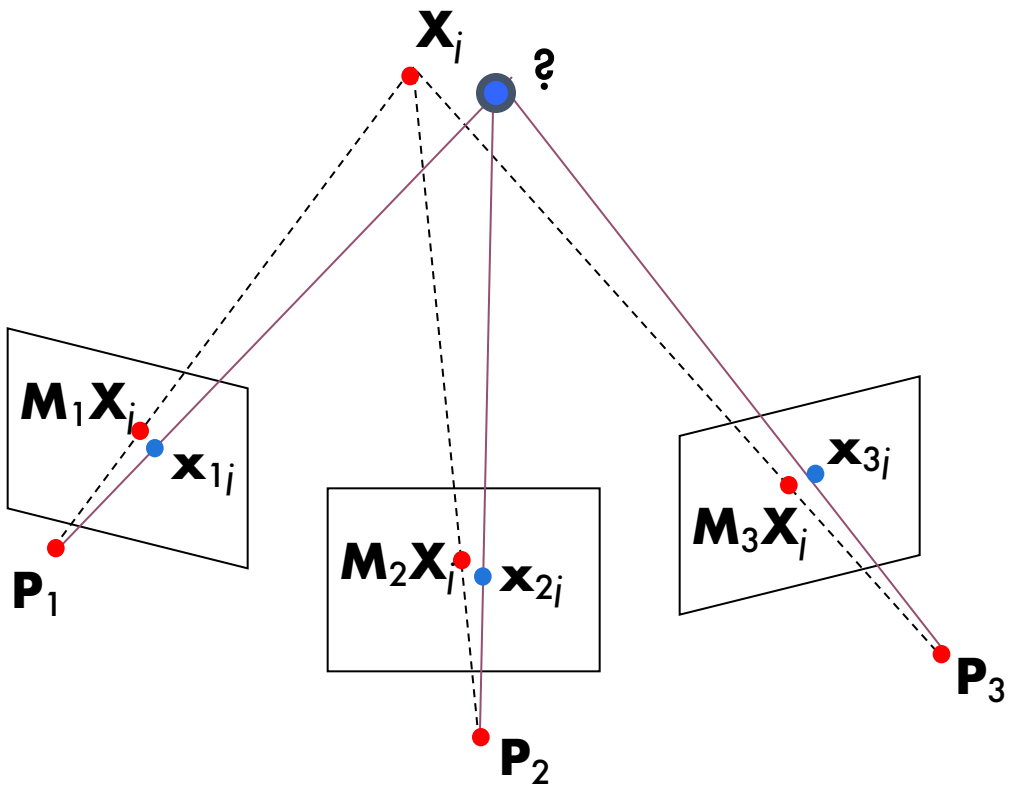
External parameters

$$X' = K\begin{bmatrix} I & 0 \end{bmatrix} X = K\begin{bmatrix} I & 0 \end{bmatrix} \begin{bmatrix} R & T \\ 0 & 1 \end{bmatrix}_{4\times4} X_w = \underbrace{K\begin{bmatrix} R & T \end{bmatrix}}_{M} X_w$$

Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$



Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$

- Advantages
  - Handle large number of views
  - Handle missing data

- Limitations
  - Large minimization problem (parameters grow with number of views)
  - requires good initial condition

Used as the final step of SFM

Applications

M. Brown and D. G. Lowe. Unsupervised 3D Object Recognition and Reconstruction in Unordered Datasets. *(3DIM2005)*

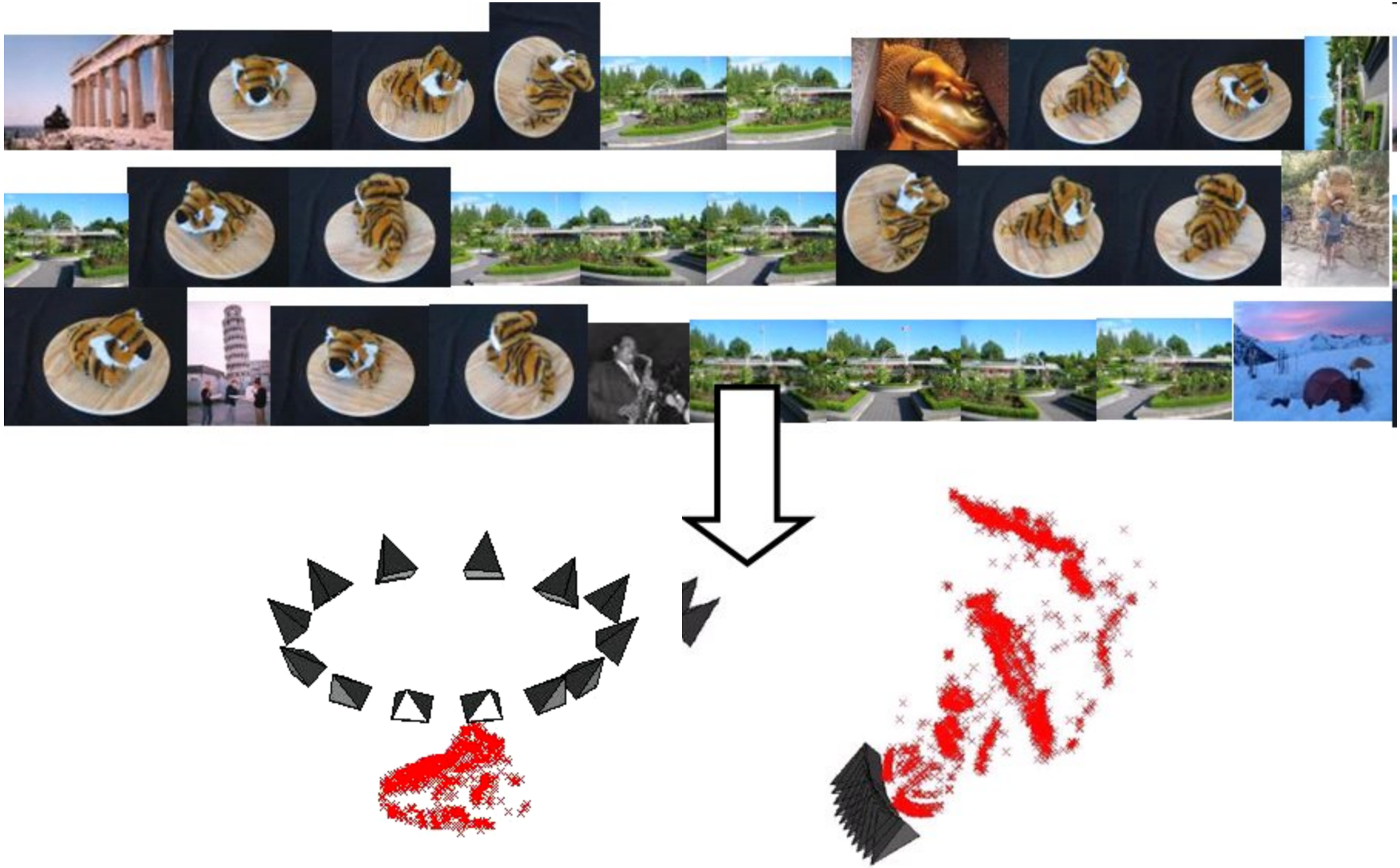
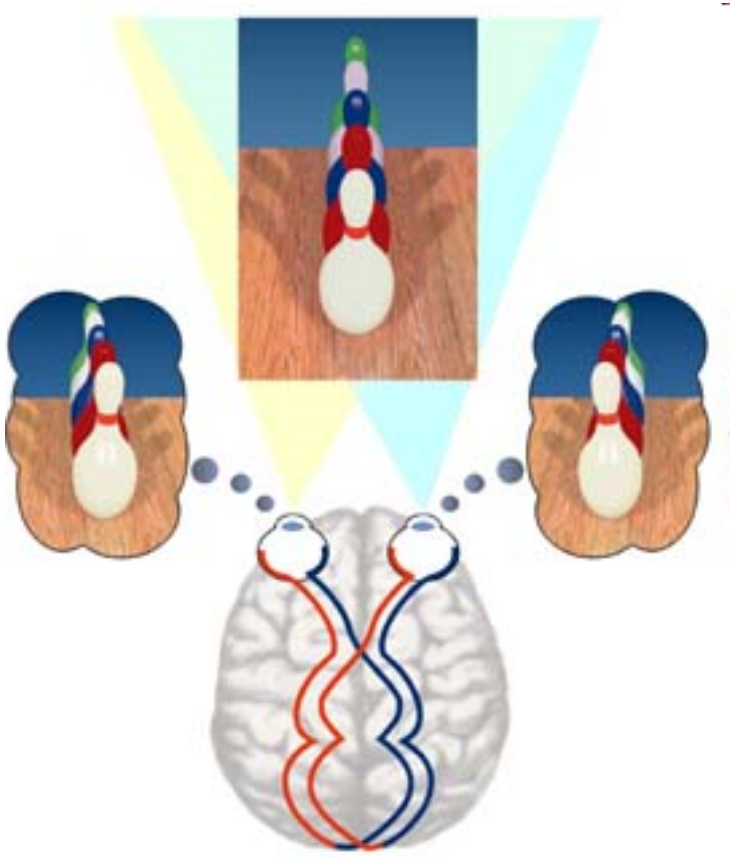


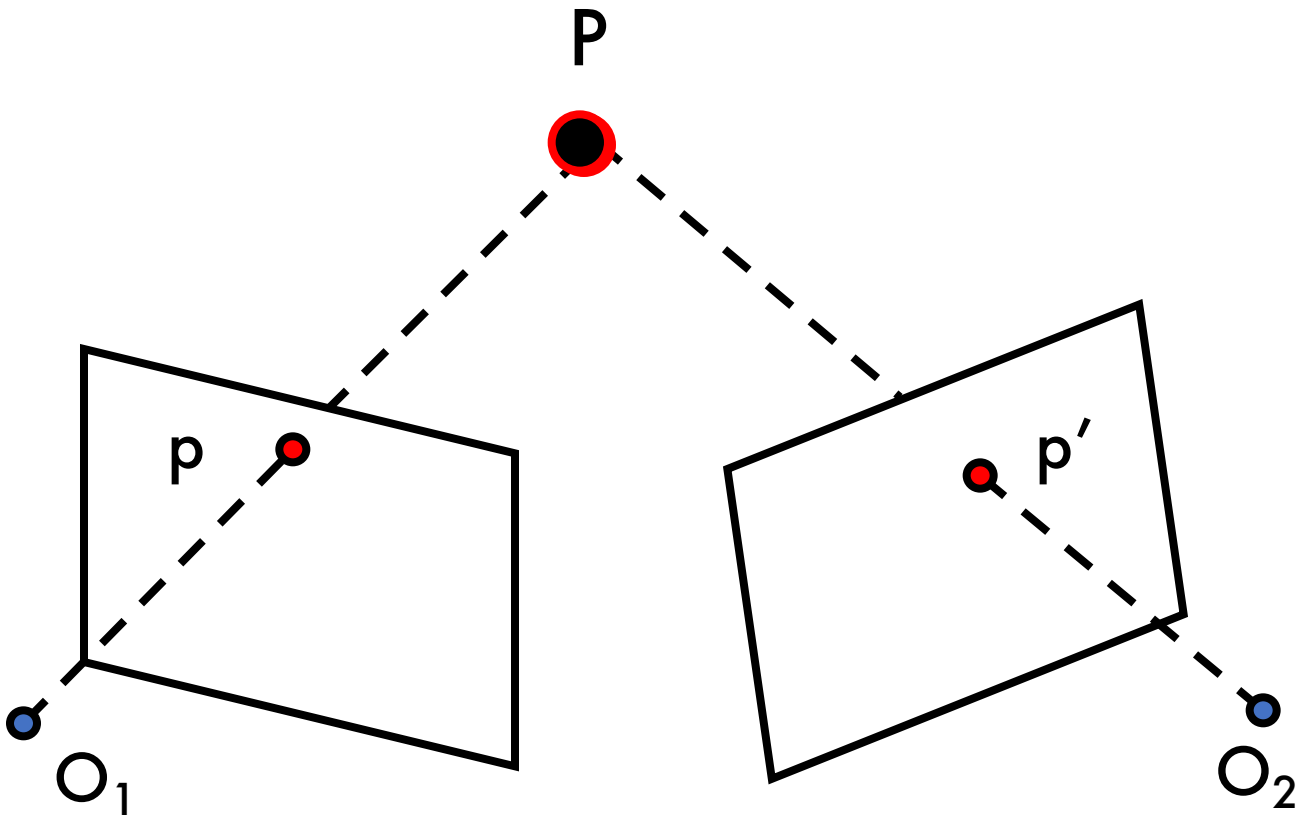
Photo synth

Noah Snavely, Steven M. Seitz, Richard Szeliski, "Photo tourism: Exploring photo collections in 3D," ACM Transactions on Graphics (SIGGRAPH Proceedings),2006,

Two eyes help!



Stereo vision

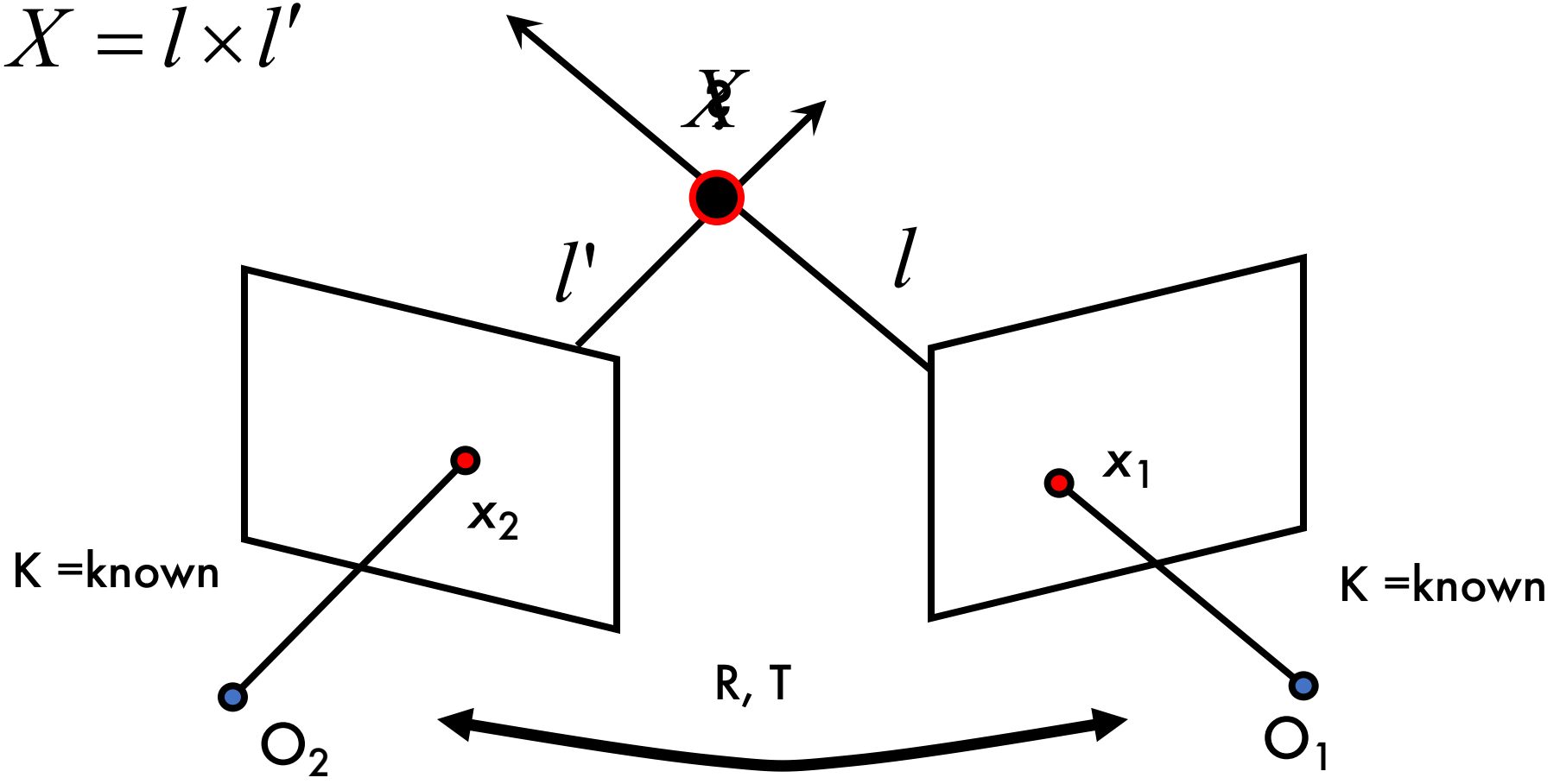


Goal: estimate the position of P given the observation of P from two view points

Assumptions: known camera parameters and position (K, R, T)

Two eyes help!

$$X = l \times l'$$

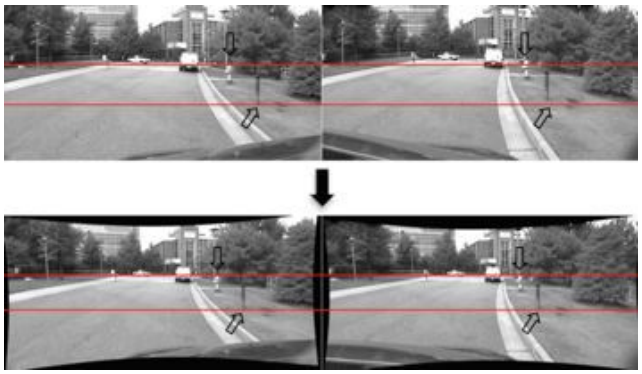


This is called triangulation

How can we use this to compute pose?

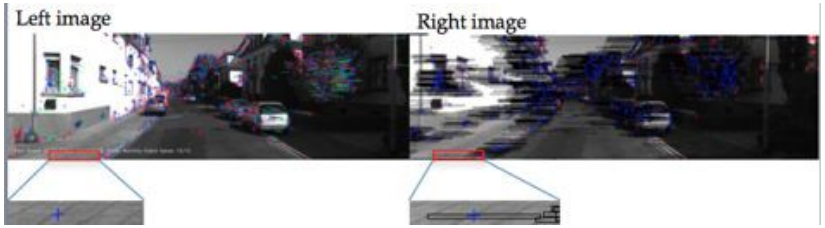
Algorithm Overview

1. Rectification



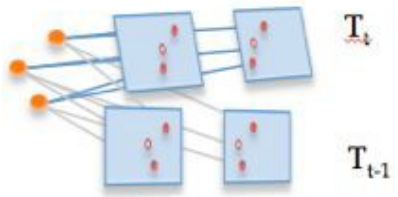
2. Feature Extraction

3. Stereo Feature Matching



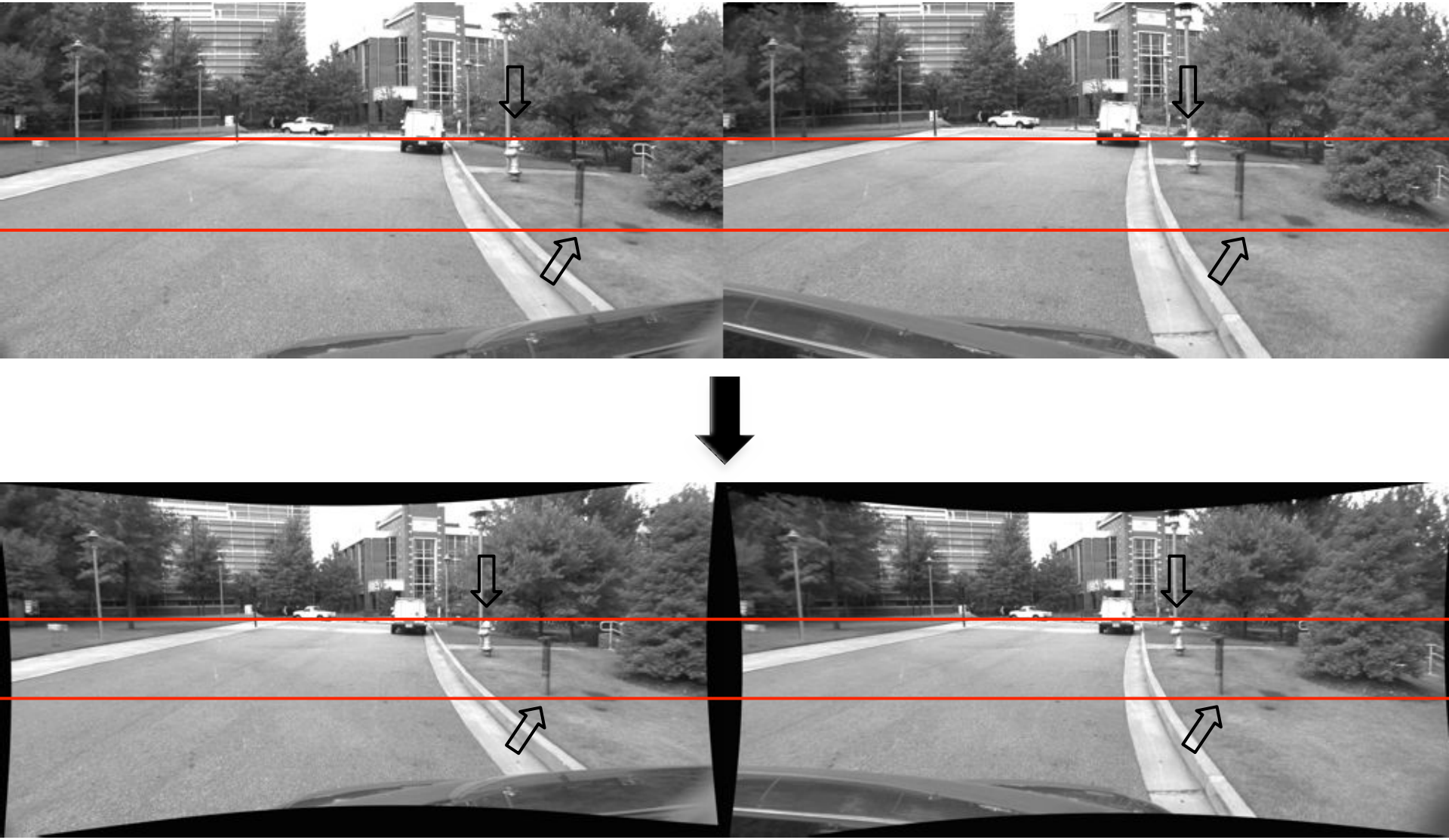
4. Temporal Feature Matching

5. Incremental Pose Recovery/RANSAC



Slides:C. Beall

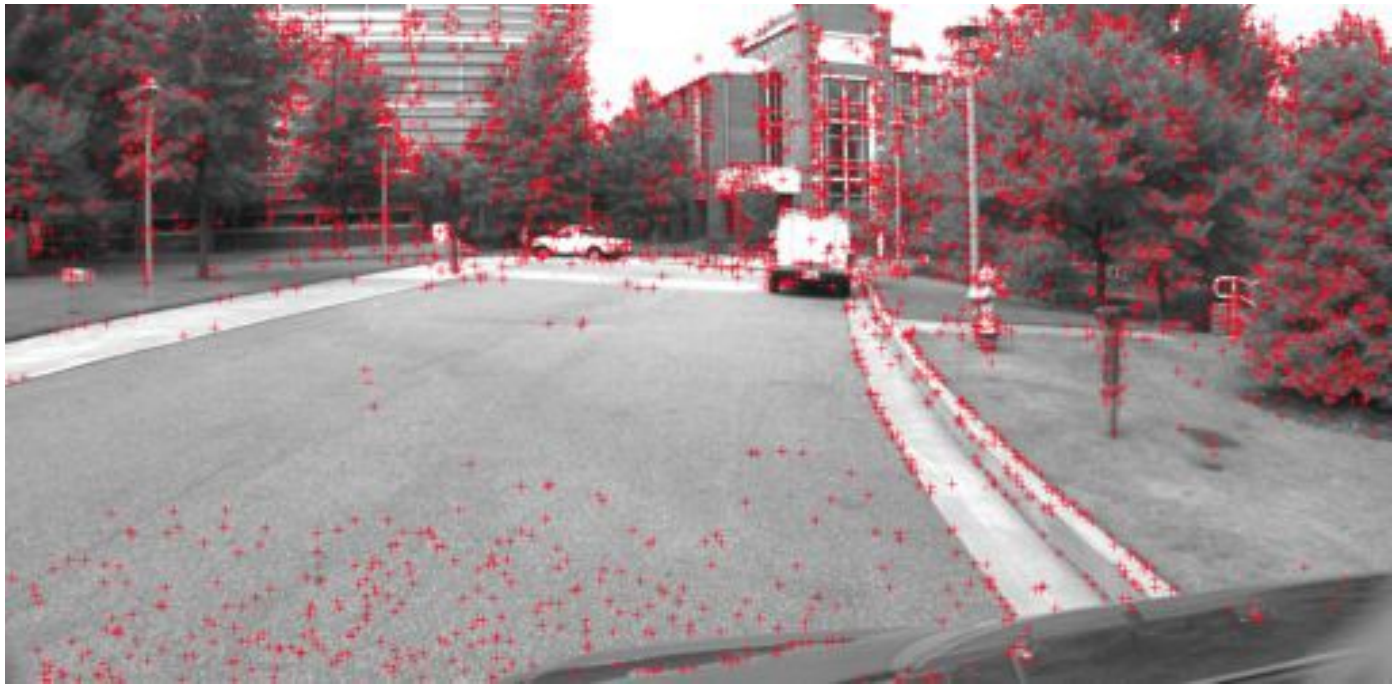
Undistortion and Rectification



Slides:C. Beall

Feature Extraction

- Detect local features in each image
  - SIFT gives good results (can also use SURF, Harris, etc.)



Lowe ICCV 1999

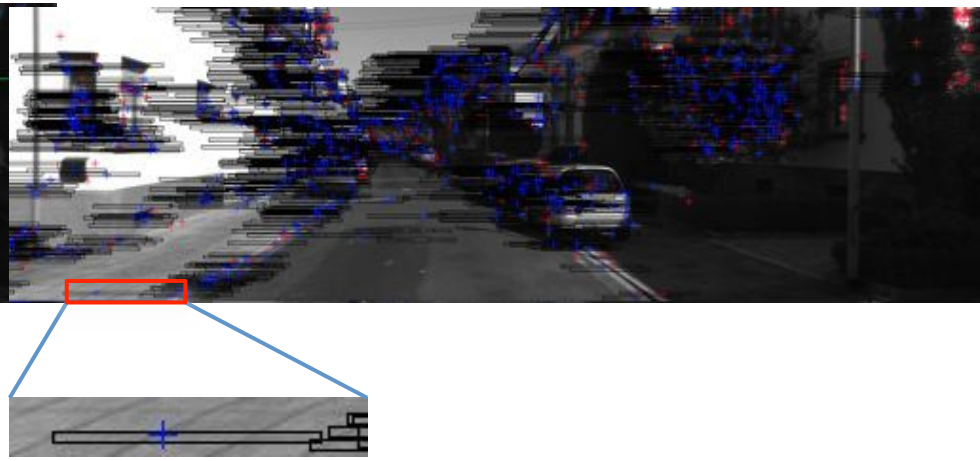
Slides:C. Beall

Stereo Matching

- Match features between left/right images
- Since the images are rectified, we can restrict the search to a bounding box on the same scan-line

Left image

Right image



Slides:C. Beall

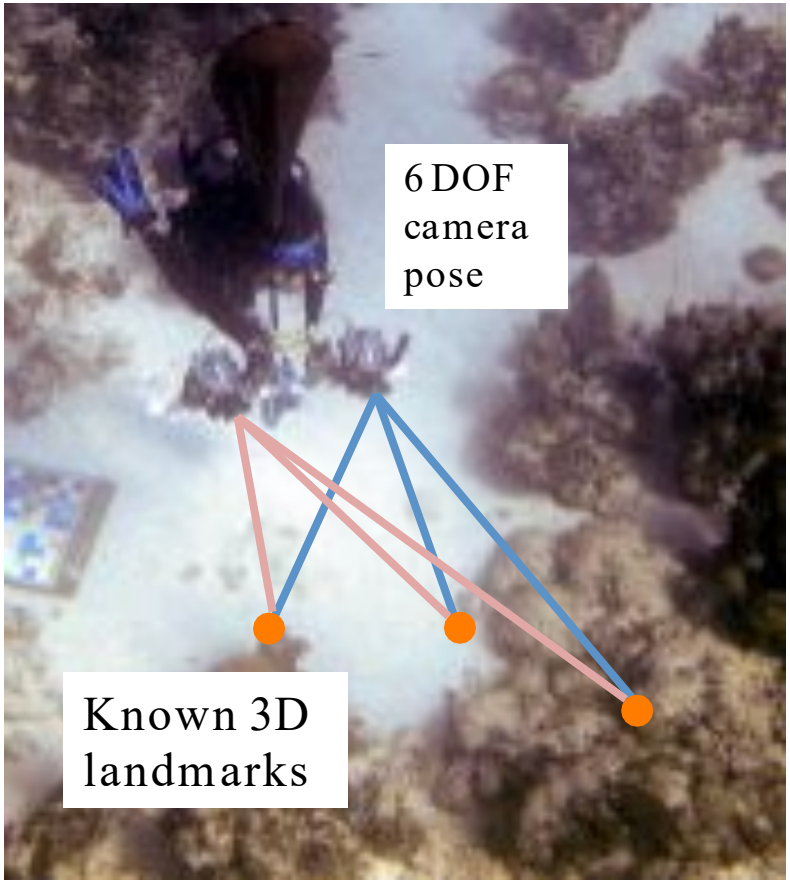
Temporal Matching

- Temporally match features between frame t and t-1

Slides:C. Beall

Relative Pose Estimation

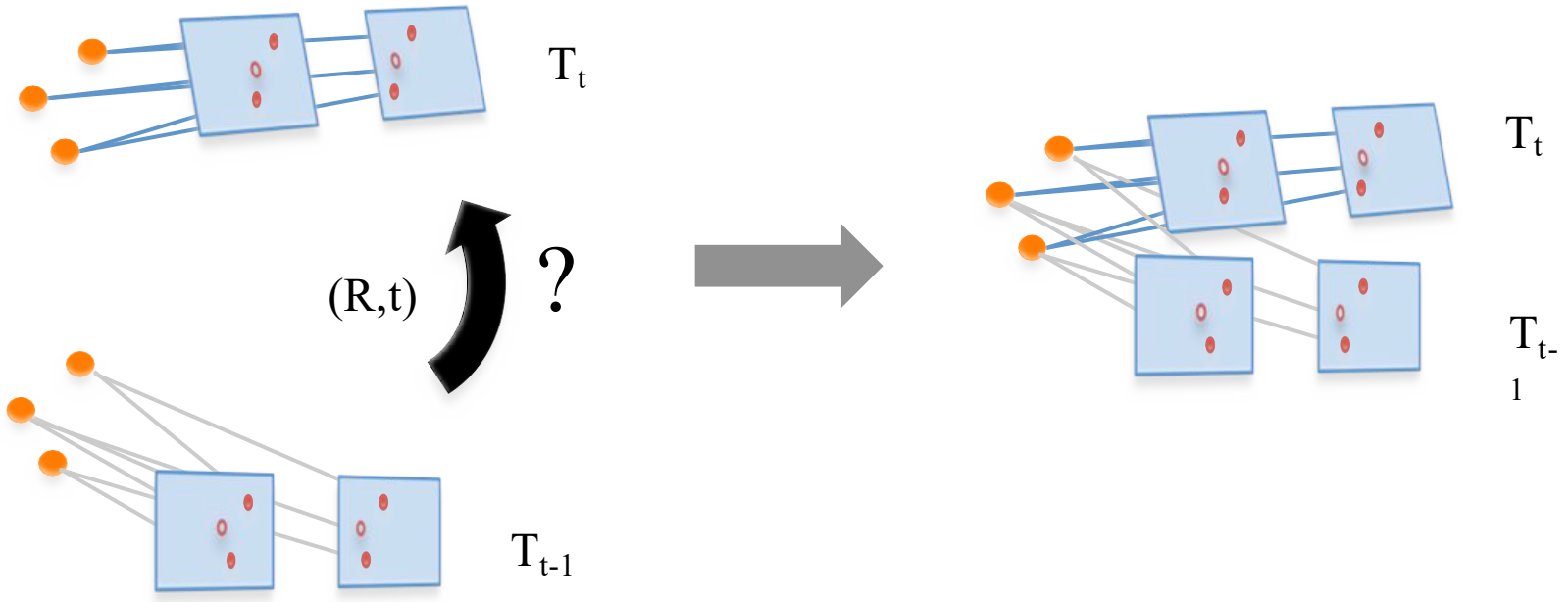
- Camera pose can be recovered given at least three known landmarks in a non-degenerate configuration
- In the case of stereo VO, landmarks can simply be triangulated
- Two ways to recover pose:
  - Absolute orientation
  - Reprojection error minimization



Slides: C. Beall

Absolute Orientation

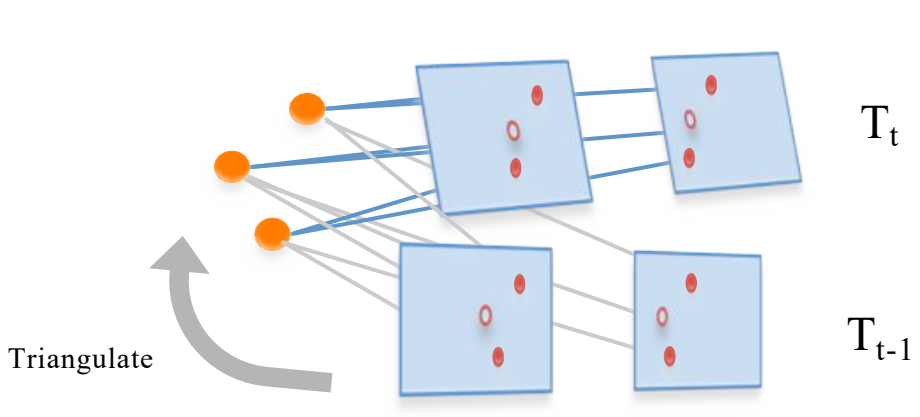
- Estimate relative camera motion by computing relative transformation between 3D landmarks which were triangulated from stereo-matched features

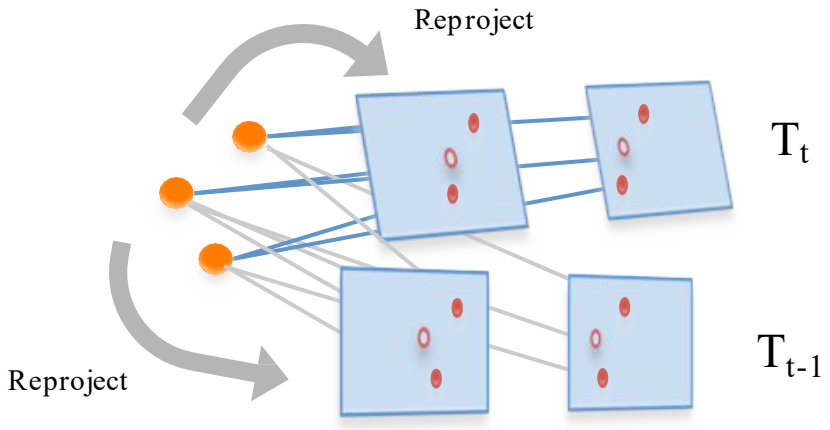


Slides:C. Beall

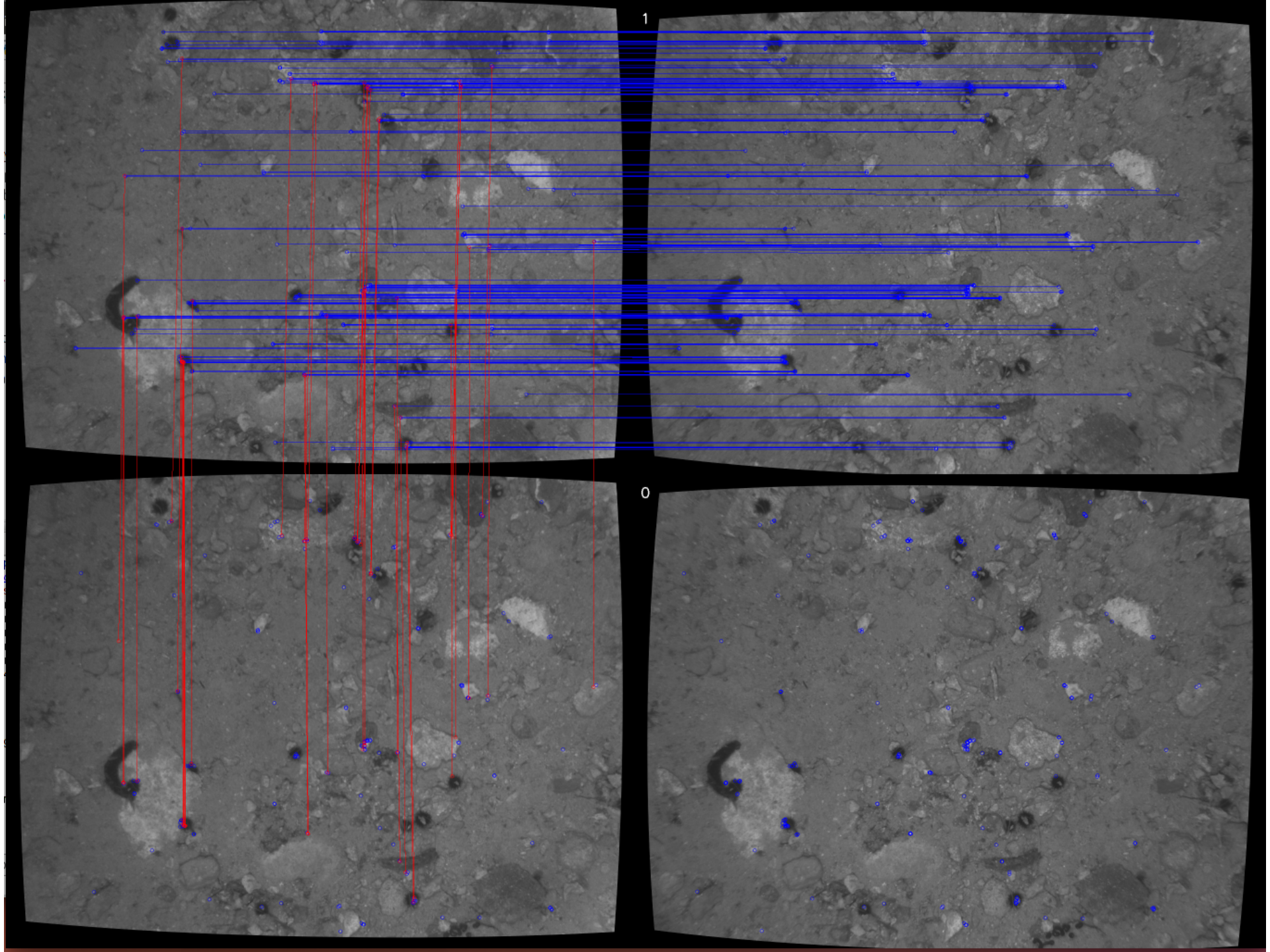
Reprojection Error Minimization

- A better approach is to estimate relative pose by minimizing reprojection error of 3D landmarks into images at time t and t-1

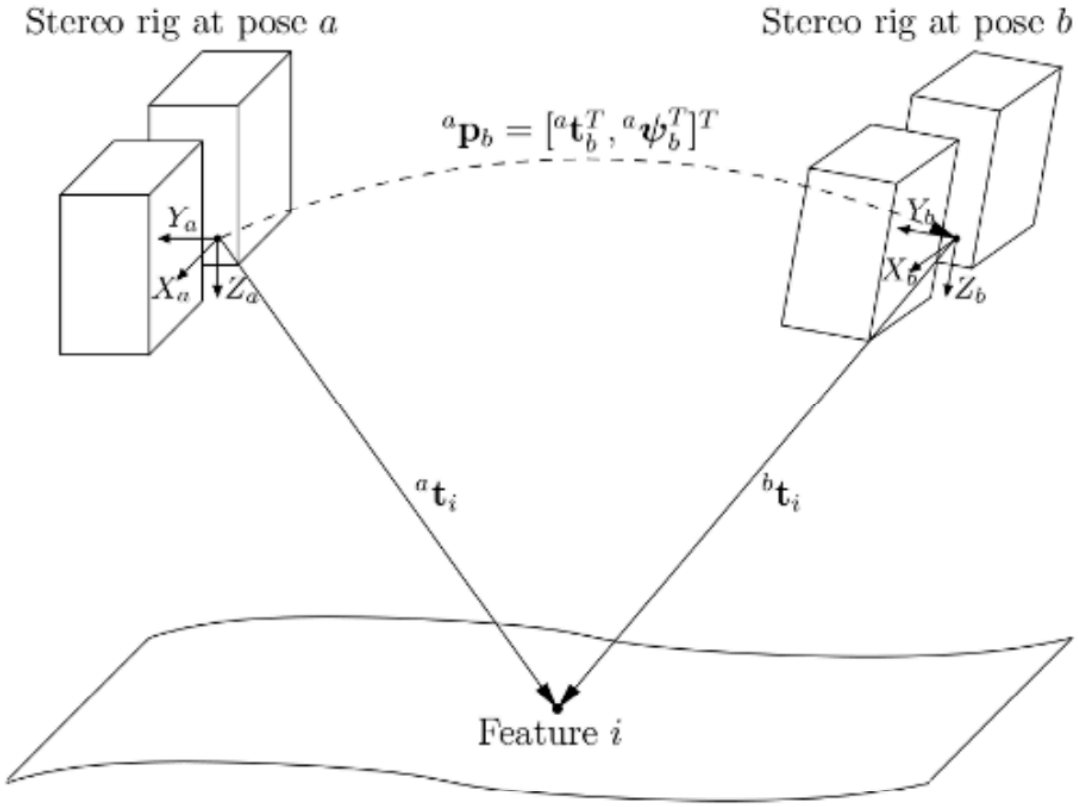


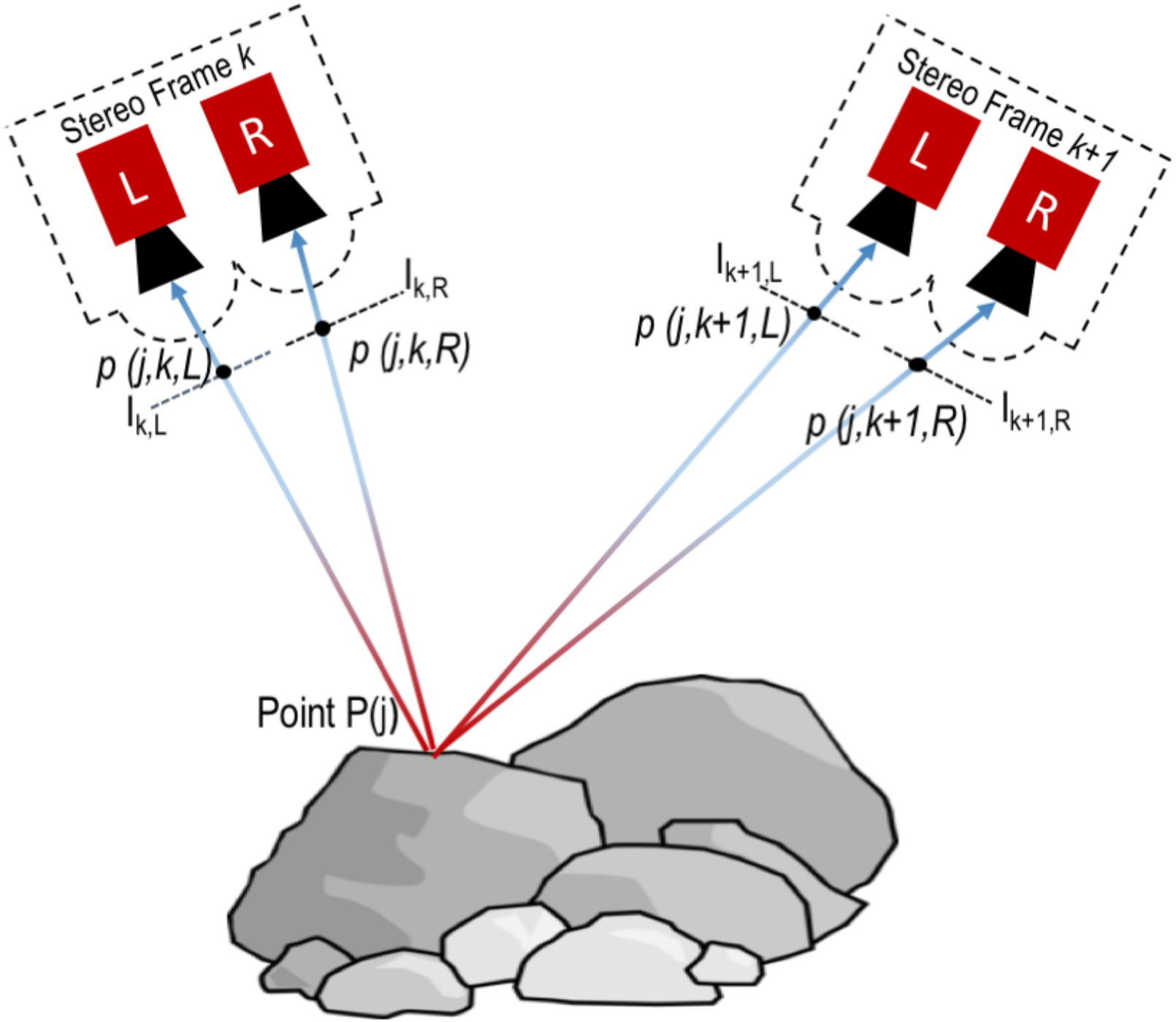


Slides:C. Beall


1
0

Relative pose estimation with 3D point registration




Stereo Frame k
L
R
$I_{k,R}$
$p\ (j,k,L)$
$p\ (j,k,R)$
$I_{k,L}$
Stereo Frame k+1
L
R
$I_{k+1,L}$
$p\ (j,k+1,L)$
$p\ (j,k+1,R)$
$I_{k+1,R}$
Point P(j)

Practical/Robustness Considerations

- The presented algorithm works very well in feature-rich, static environments, but…
- A few tricks for better results in challenging conditions:
  - Feature binning to cope with bias due to uneven feature distributions
  - Keyframing to cope with dynamic scenes, as well as reducing drift of a stationary camera

Slides:C. Beall

Feature Binning

- Incremental pose estimation yields poor results when features are concentrated in one area of the image
- Solution: Draw a grid and keep k strongest features in each cell

Slides:C. Beall

Challenges: Dynamic Scenes

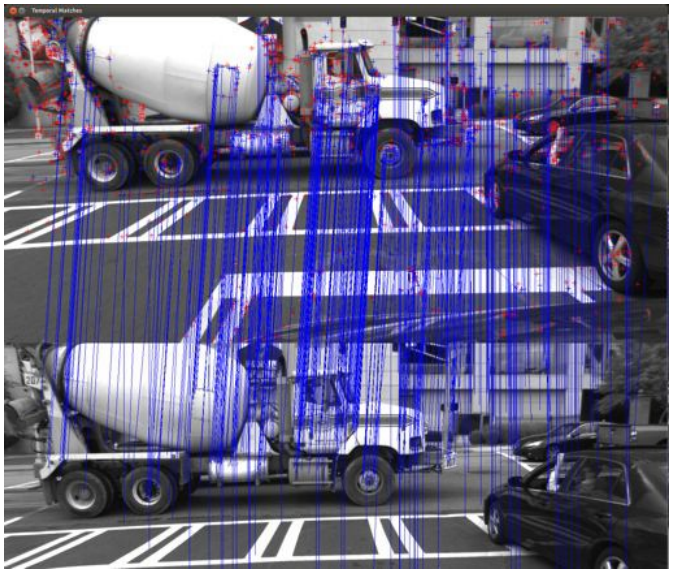
- Dynamic, crowded scenes present a real challenge
- Cannot depend on RANSAC to always recover the correct inlier set
- Example 1: Large van “steals” inlier set in passing

Inliers Outliers

Slides:C. Beall

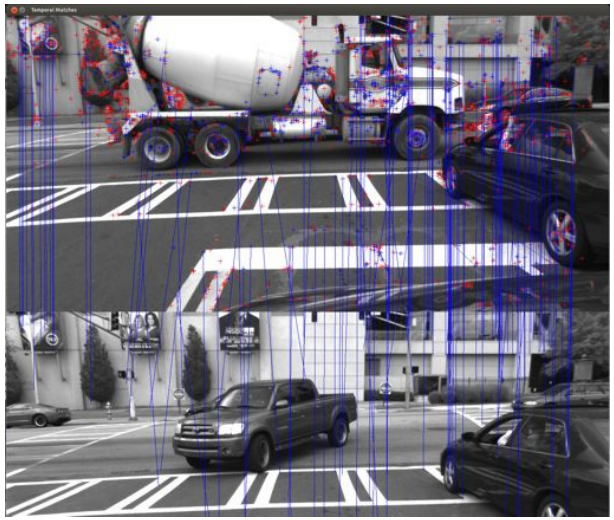
Challenges: Dynamic Scenes

- Example 2: Cross-traffic while waiting to turn left at light

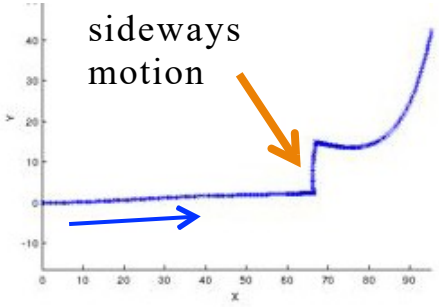


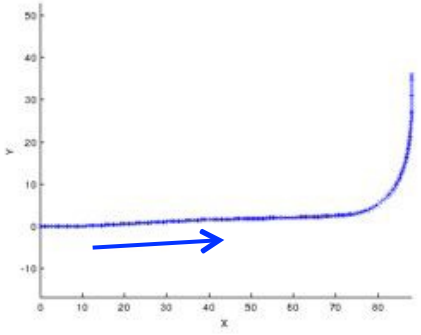
Only accept incremental pose if:

- translation > 0.5m
- Dominant direction is forward



Incorrect sideways motion





Without keyframing

With keyframing

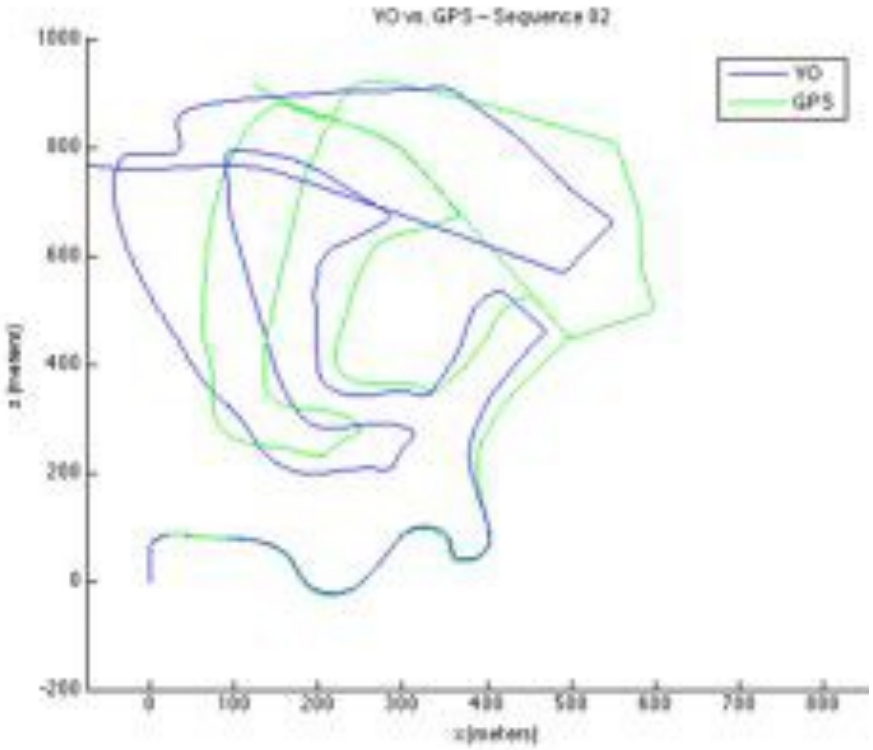
Slides:C. Beall

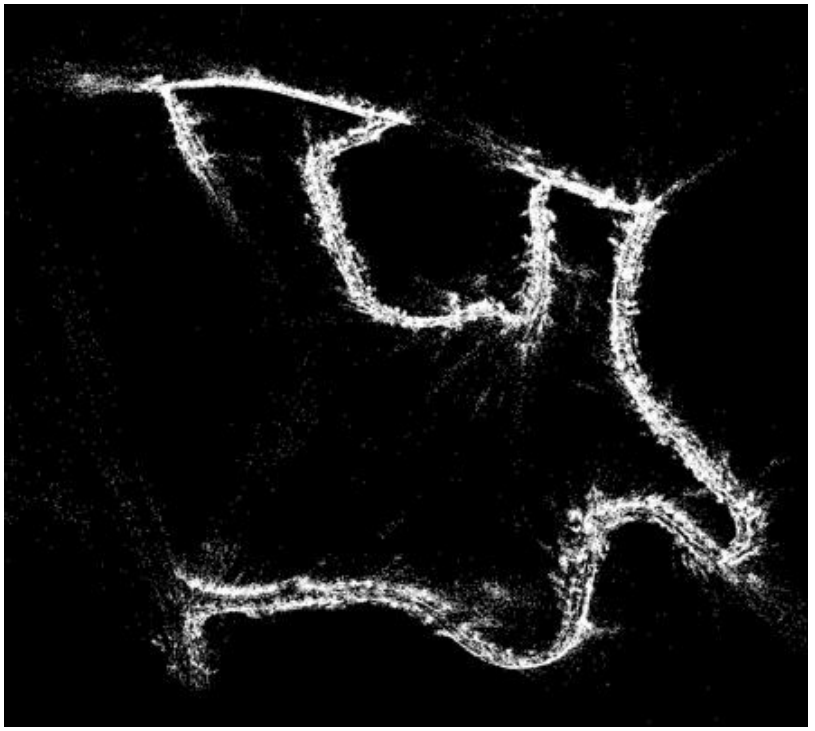
What else do you get?

- Stereo Visual Odometry yields more than just a camera trajectory!
- Tracked landmarks form a sparse 3D point cloud of the environment
  - Can be used as the basis for localization

What else do you get?

- 3D Point cloud on KITTI Benchmark, Sequence 2

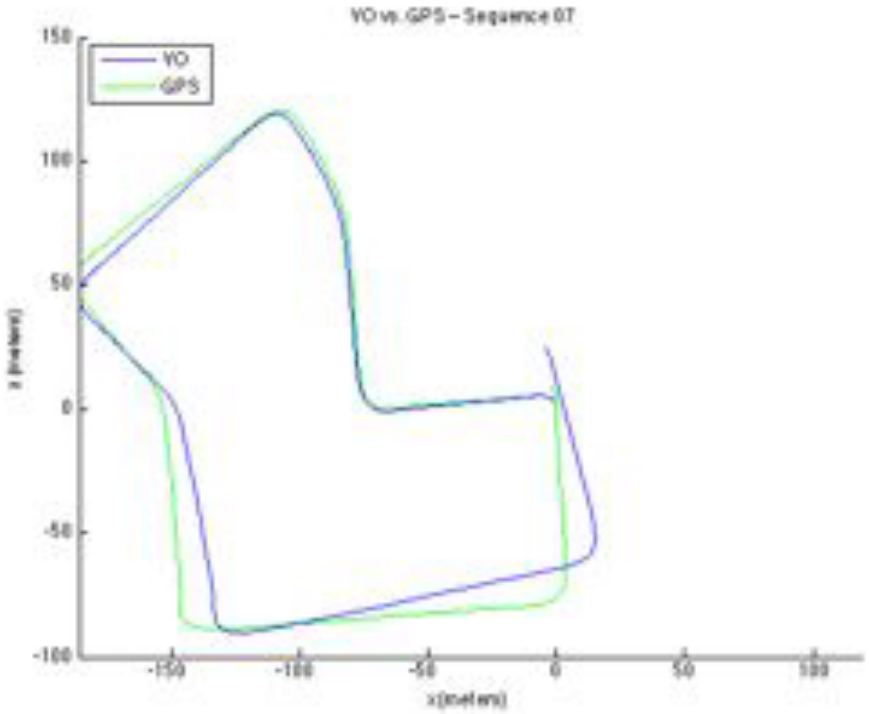


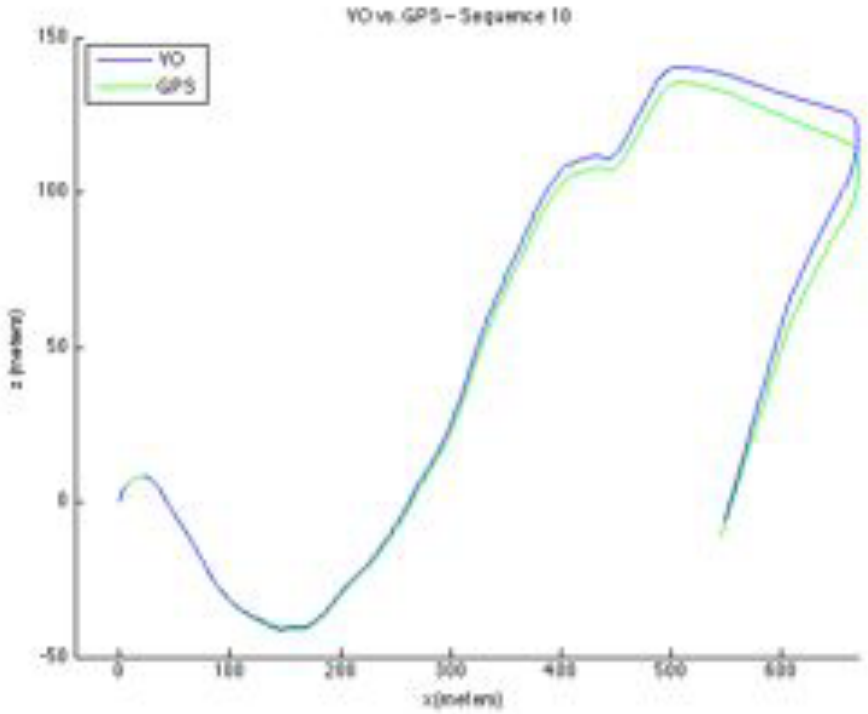


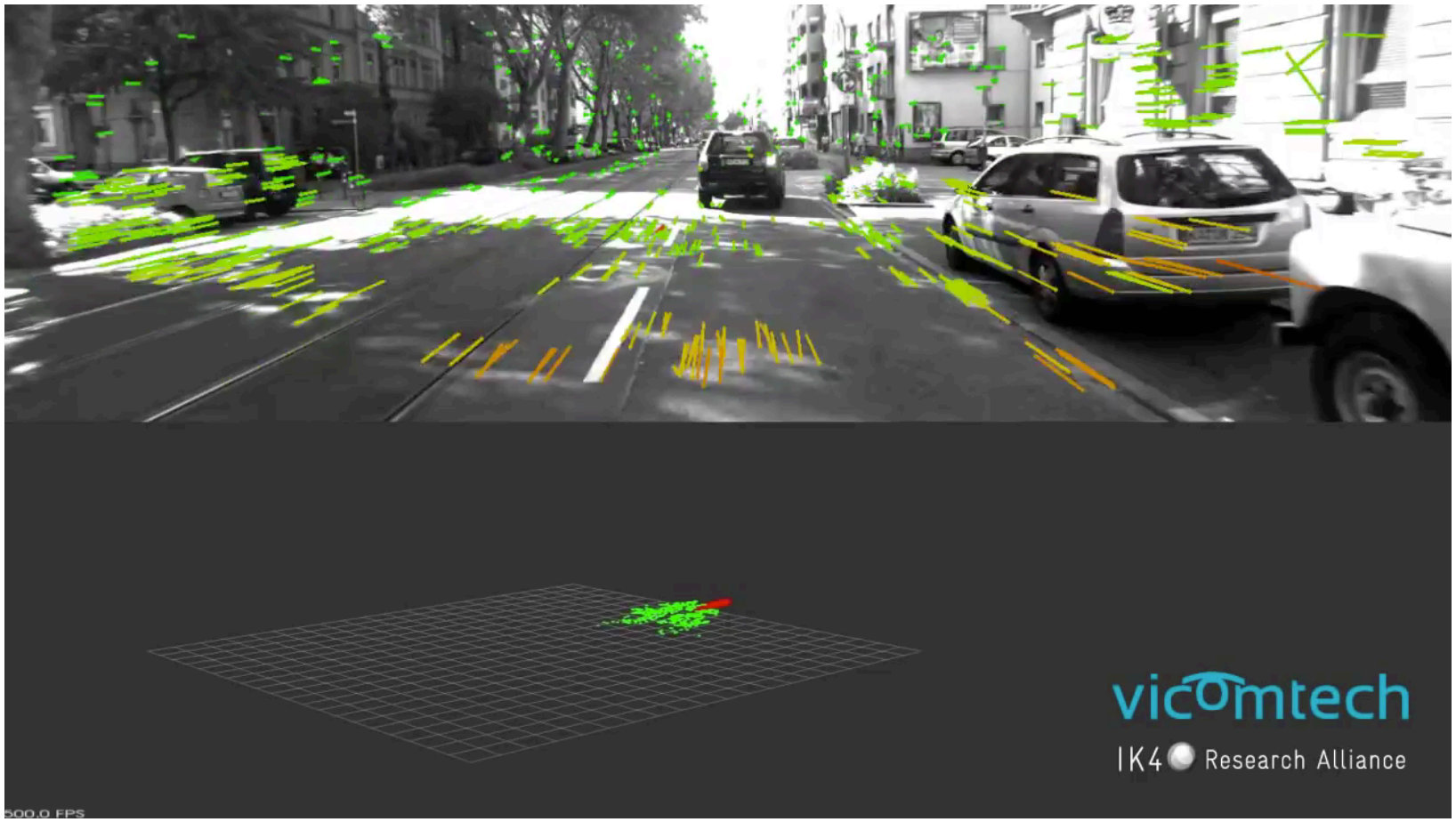
http://www.cvlibs.net/datasets/kitti/

Results on KITTI Benchmark

- Representative results on KITTI VO Benchmark
  - Average translational/rotational errors are very small
  - Accumulate over time, resulting in drift






vicomtech
IK4 Research Alliance
500.0 FPS

How to deal with outliers? RANSAC

Fitting

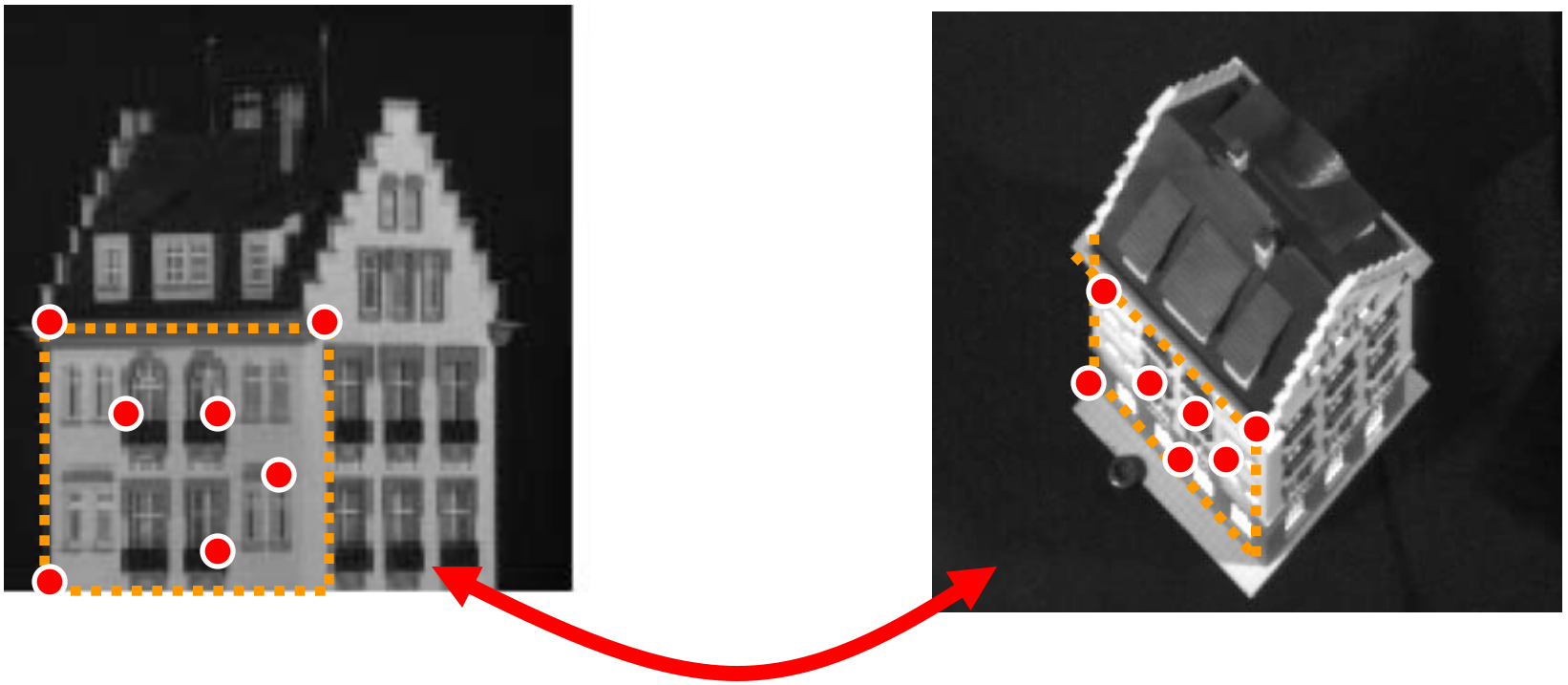
Goals:

- Choose a parametric model to fit a certain quantity from data
- Estimate model parameters

- Lines
- Curves
- Homographic transformation
- Fundamental matrix
- Shape model

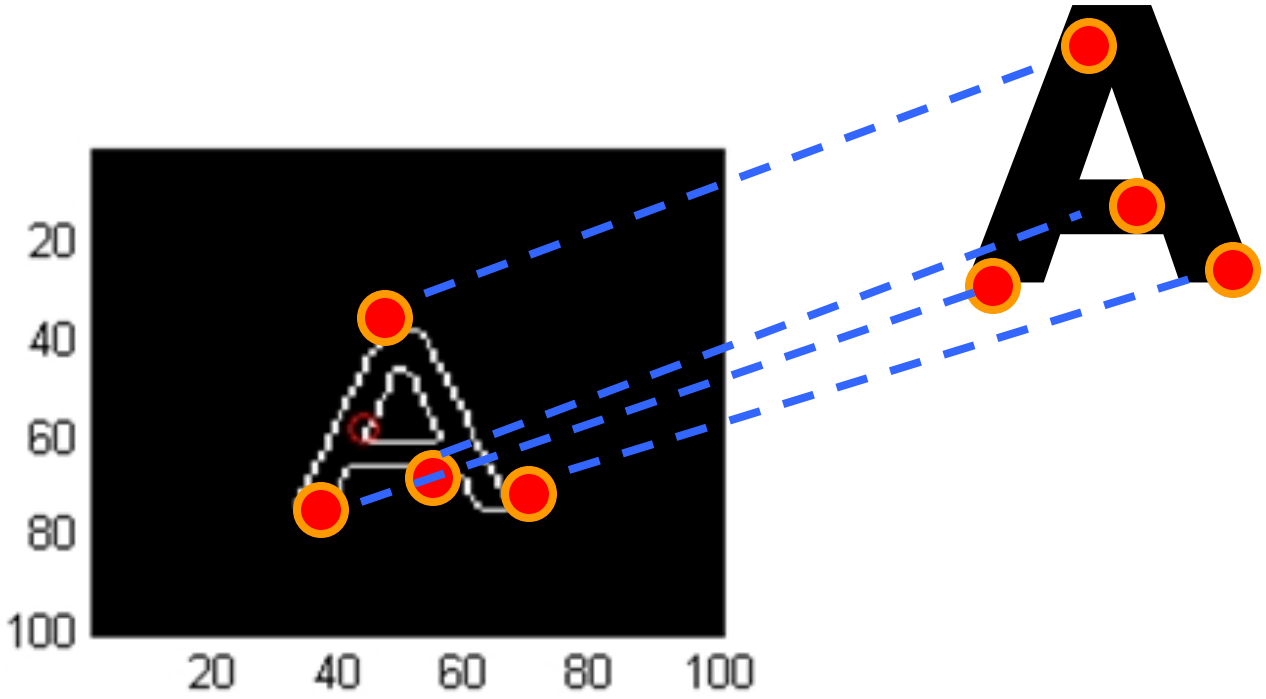
Example: fitting lines
(for computing vanishing points)

Example: Estimating an homographic transformation

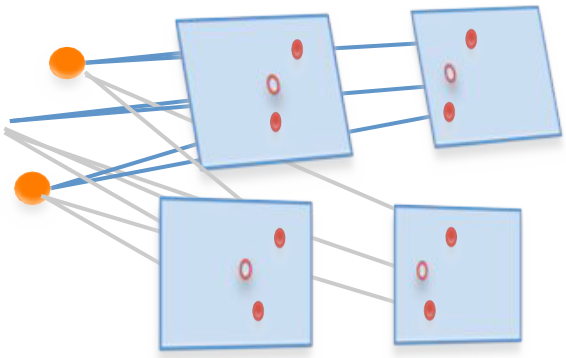


H

Example: fitting a 2D shape template



Example: fitting stereo relative pose



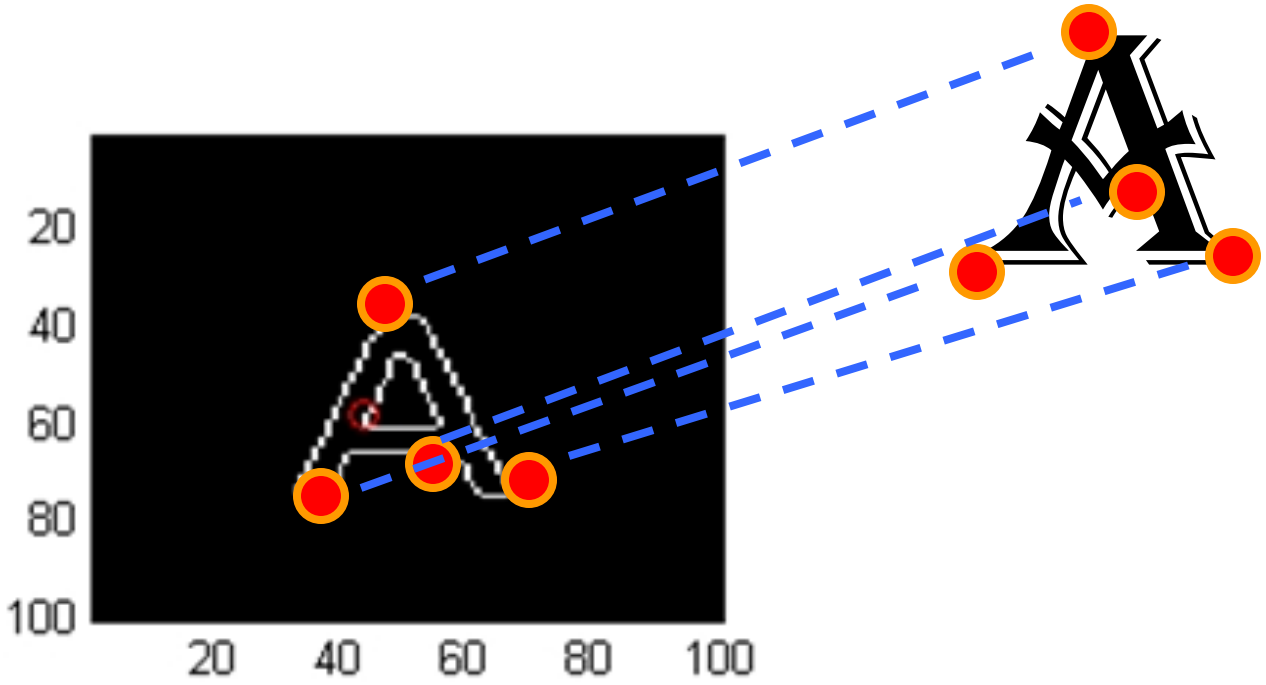
Fitting

Critical issues:

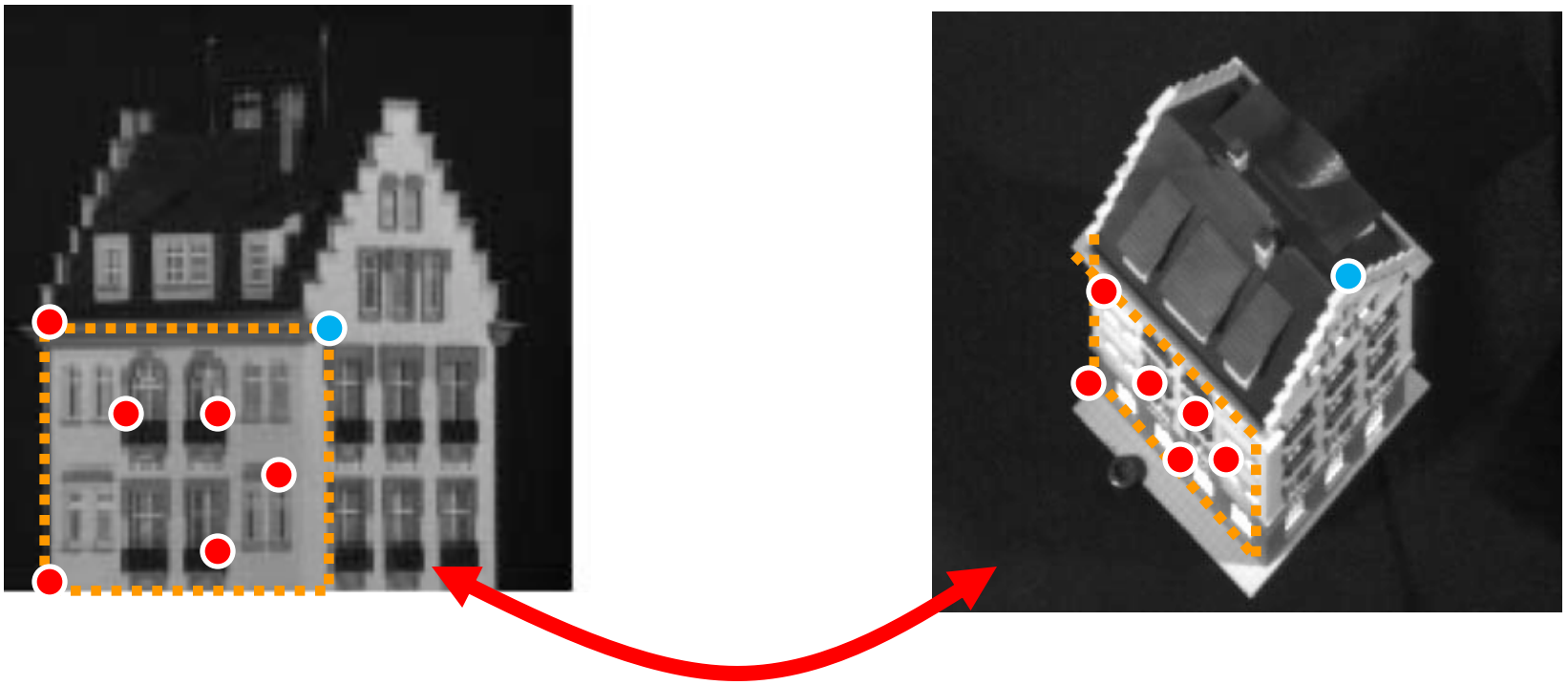
- noisy data
- outliers
- missing data

Critical issues: noisy data

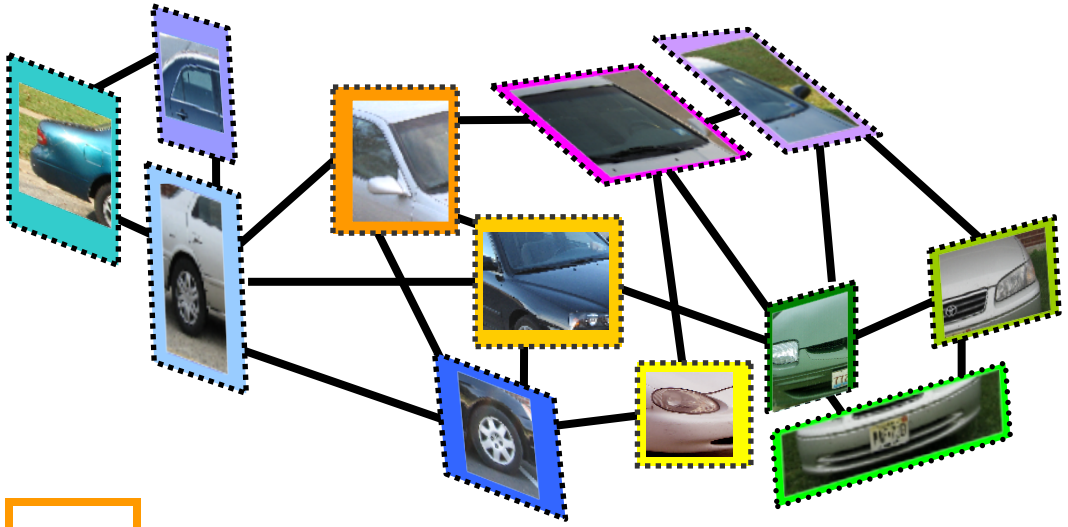
Critical issues: noisy data (intra-class variability)

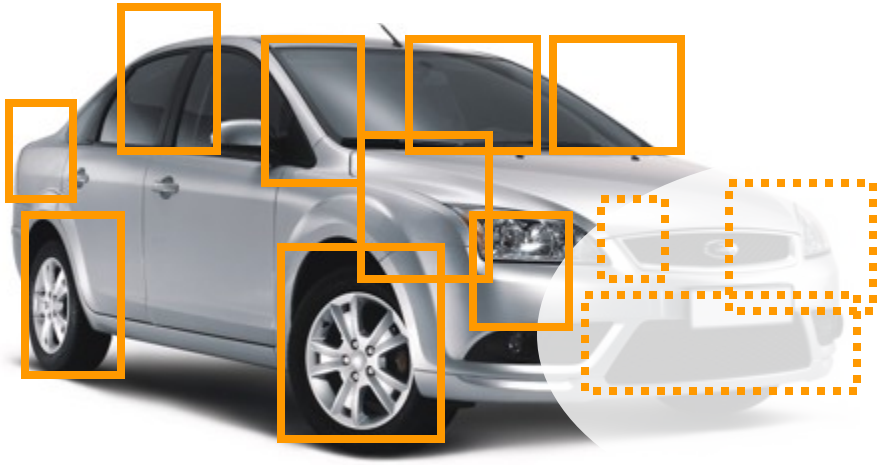


Critical issues: outliers



Critical issues: missing data (occlusions)





Fitting

Goal: Choose a parametric model to fit a certain quantity from data

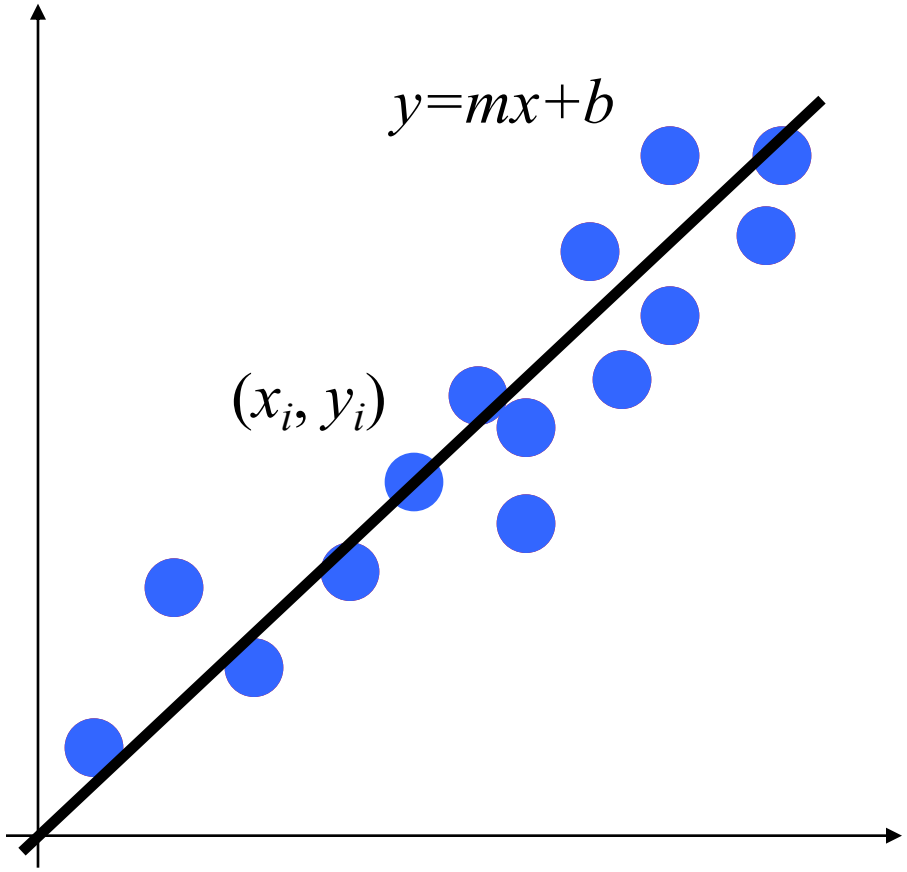
Techniques:

- Least square methods
- RANSAC
- Hough transform
- EM (Expectation Maximization) [not covered]

Least squares methods
- fitting a line -

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

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



Least squares methods

- fitting a line -

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

$$\mathrm{E} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \left(\mathrm{y_i} - \begin{bmatrix} \mathrm{x_i} & 1 \end{bmatrix} \begin{bmatrix} \mathrm{m} \\ \mathrm{b} \end{bmatrix} \right)^2 = \left\| \begin{bmatrix} \mathrm{y_1} \\ \vdots \\ \mathrm{y_n} \end{bmatrix} - \begin{bmatrix} \mathrm{x_1} & 1 \\ \vdots & \vdots \\ \mathrm{x_n} & 1 \end{bmatrix} \begin{bmatrix} \mathrm{m} \\ \mathrm{b} \end{bmatrix} \right\|^2 = \|\mathrm{Y} - \mathrm{XB}\|^2$$

$$= (\mathrm{Y} - \mathrm{XB})^{\mathrm{T}} (\mathrm{Y} - \mathrm{XB}) = \mathrm{Y}^{\mathrm{T}}\mathrm{Y} - 2(\mathrm{XB})^{\mathrm{T}}\mathrm{Y} + (\mathrm{XB})^{\mathrm{T}}(\mathrm{XB})$$

Find (m, b) that minimize E

$$\frac{dE}{dB} = -2X^T Y + 2X^T XB = 0$$

$$\mathrm{X}^{\mathrm{T}}\mathrm{XB} = \mathrm{X}^{\mathrm{T}}\mathrm{Y}$$

Normal equation

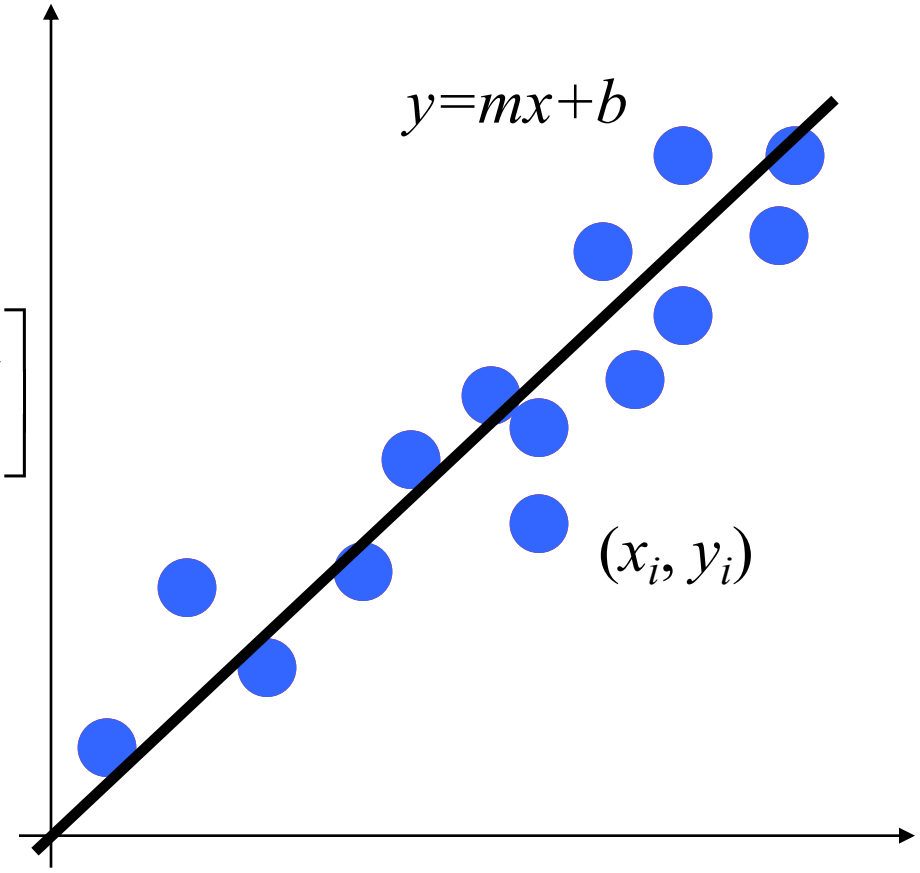
$$\mathrm{B} = \left(\mathrm{X}^{\mathrm{T}}\mathrm{X}\right)^{-1}\mathrm{X}^{\mathrm{T}}\mathrm{Y}$$

Least squares methods

- fitting a line -

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

$$B = \left(X^T X\right)^{-1} X^T Y \qquad B = \begin{bmatrix} m \\ b \end{bmatrix}$$



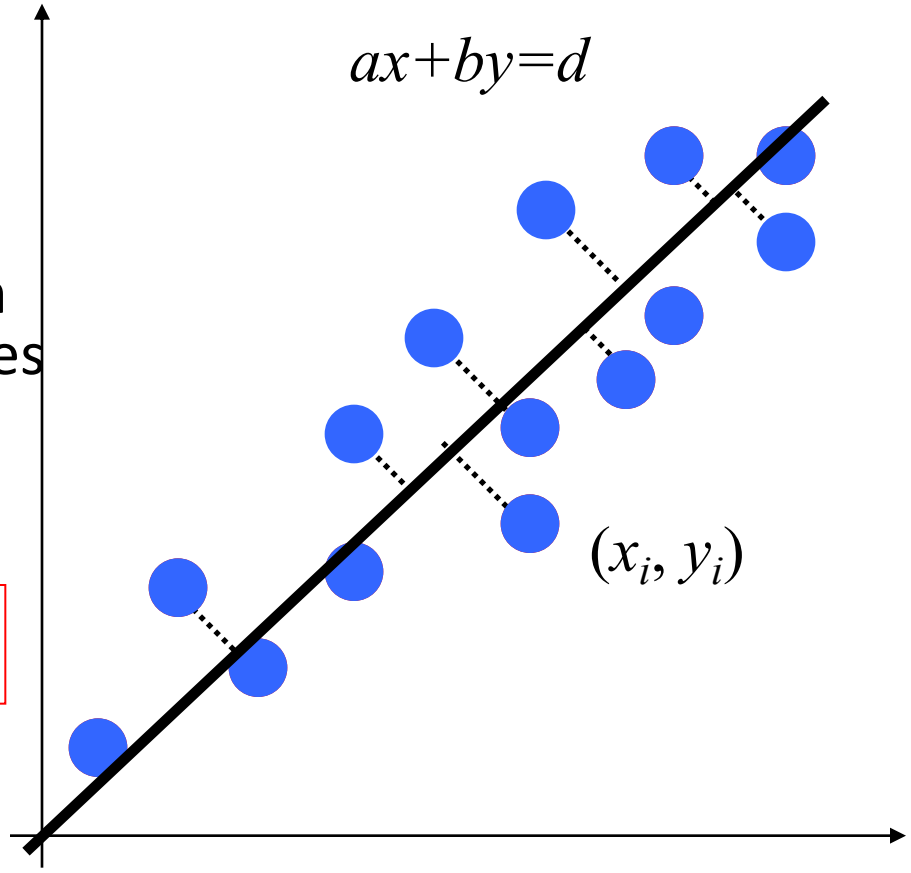
Limitations

- Fails completely for vertical lines

Least squares methods
- fitting a line -

- Distance between point
  (x_n, y_n) and line $ax+by=d$

- Find (a, b, d) to minimize the sum of squared perpendicular distances



$$E = \sum_{i=1}^{n} (ax_i + by_i - d)^2$$

$$UN = 0$$

data model parameters

Least squares methods

- fitting a line -

$$\mathrm{A\,h} = 0$$

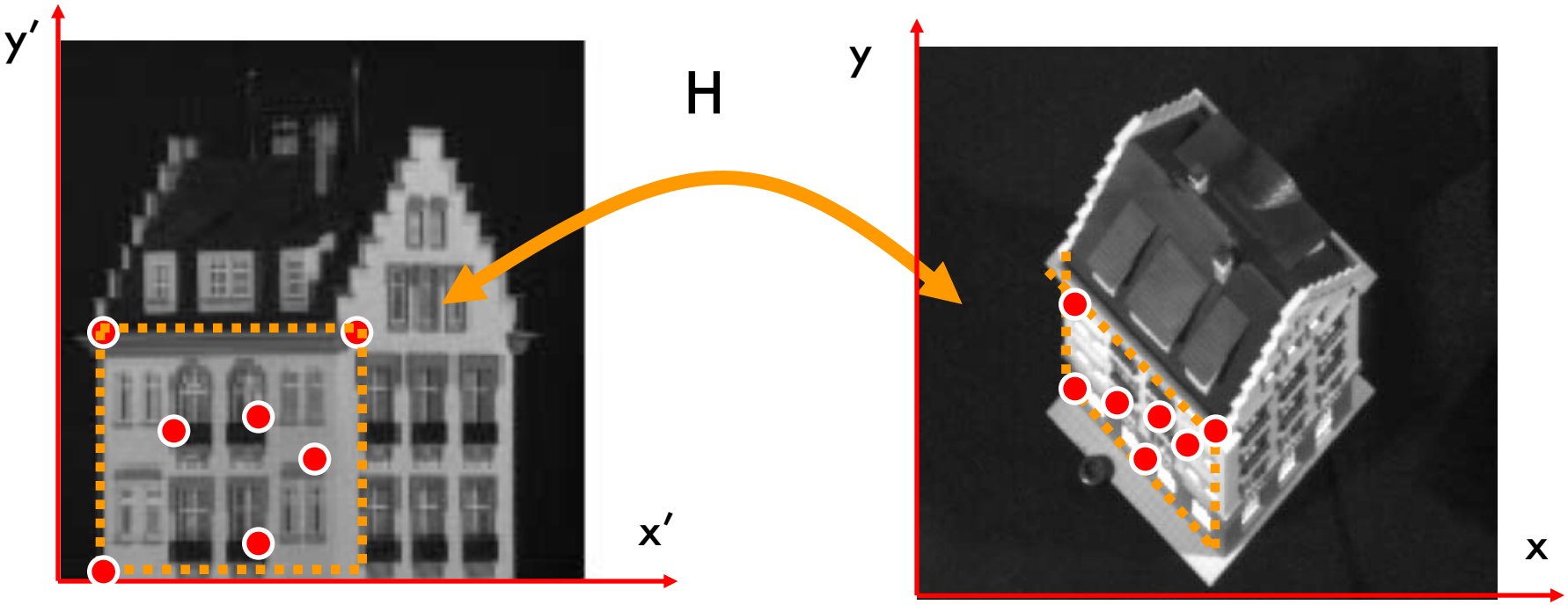
$$\text{Minimize} \| \mathrm{A\,h} \| \quad \text{subject to} \quad \| \mathrm{h} \| = 1$$

$$A = UDV^T$$

$$\mathrm{h} = \text{last column of } \mathrm{V}$$

Least squares methods

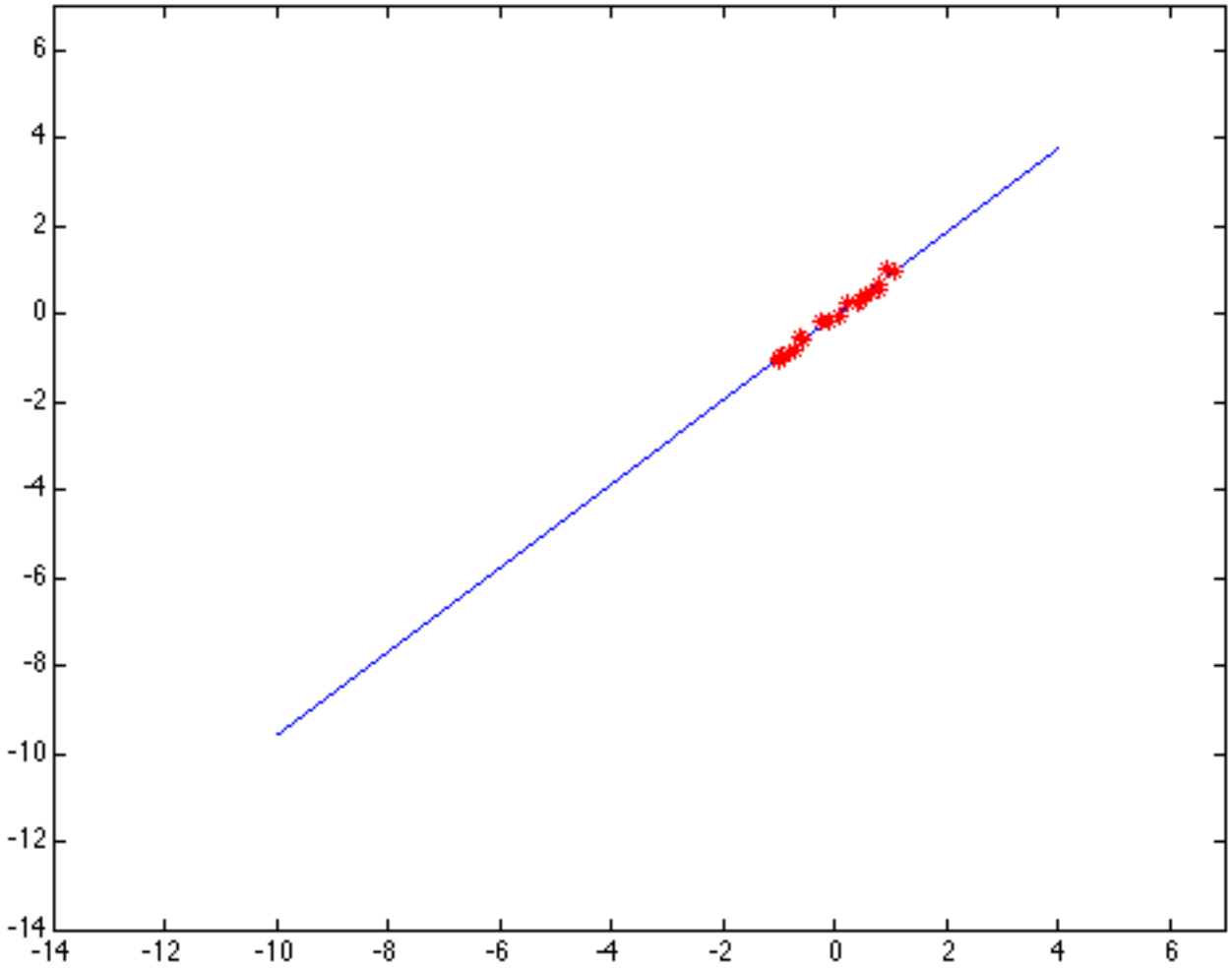
- fitting an homography -



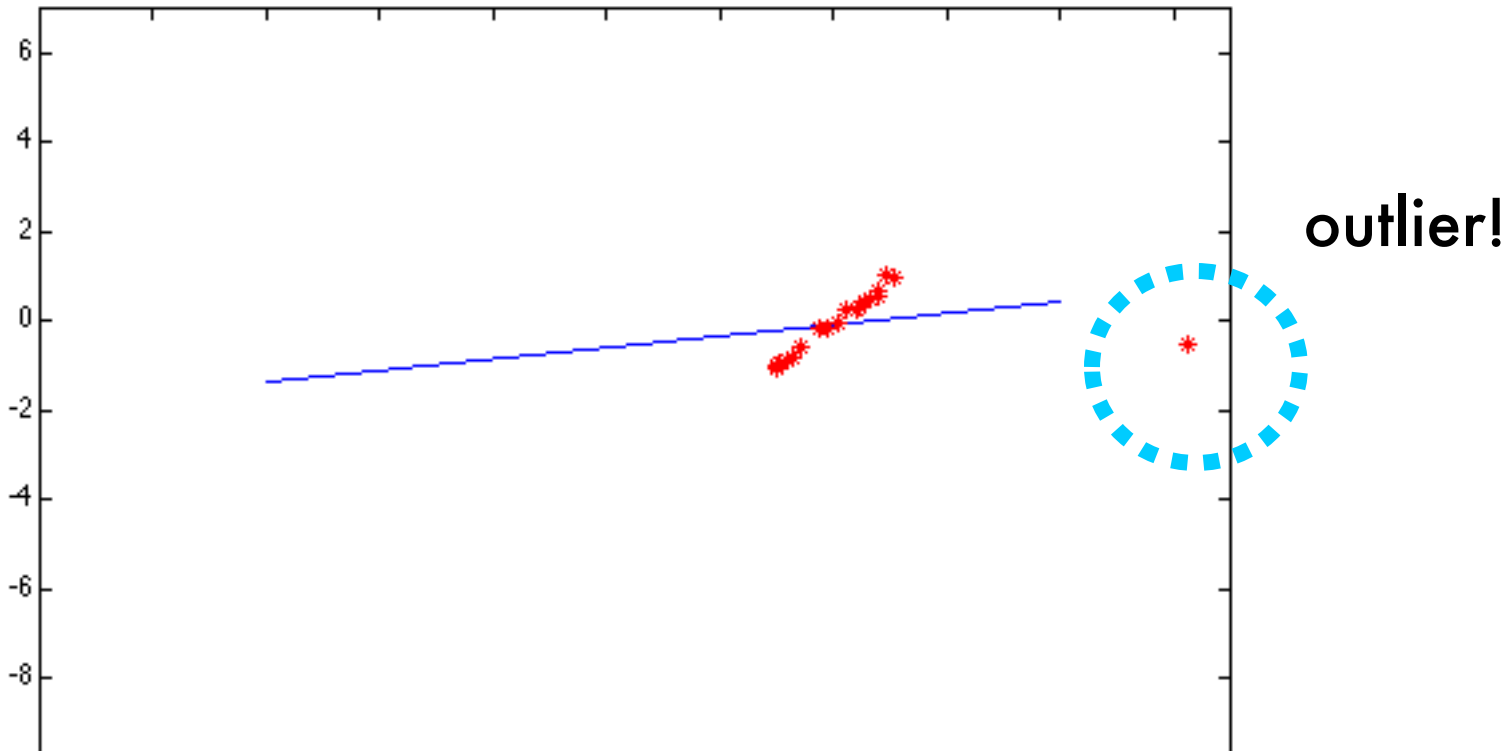
$$U\,N = 0$$

data model parameters

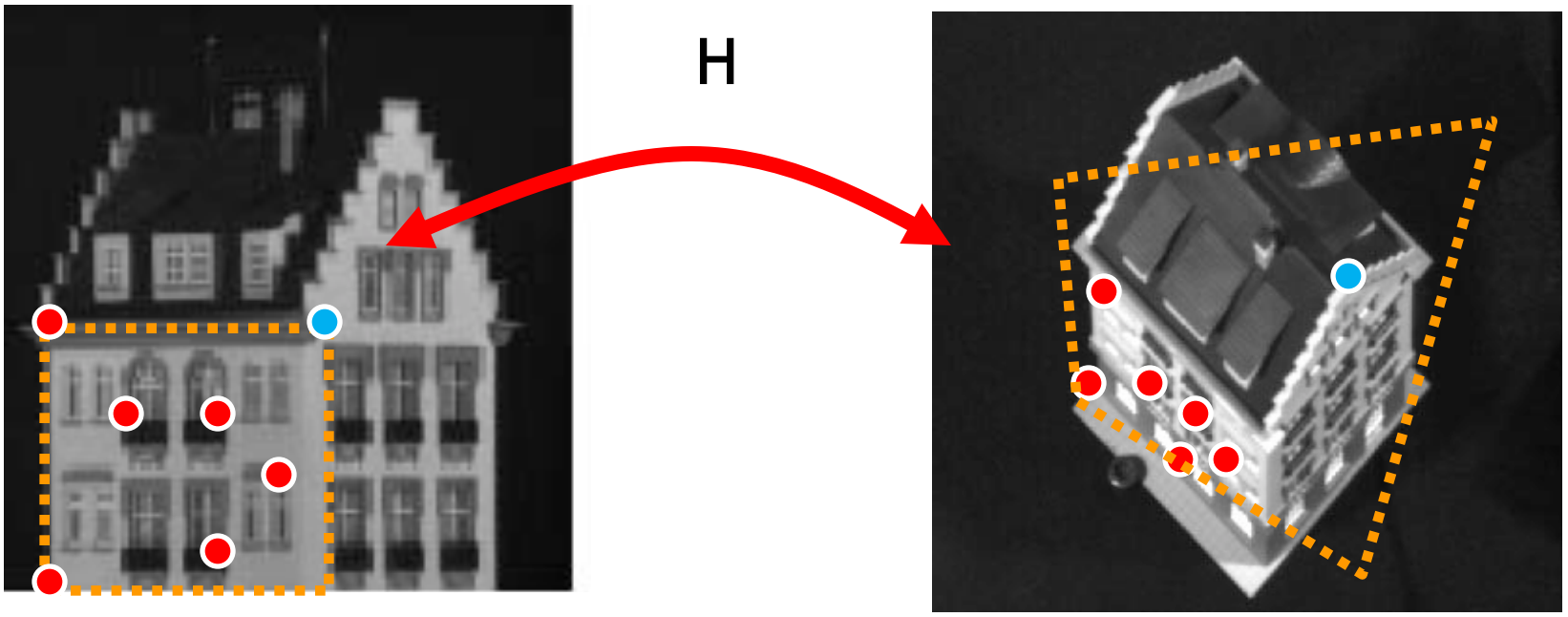
Least squares: Robustness to noise



Least squares: Robustness to noise



Critical issues: outliers



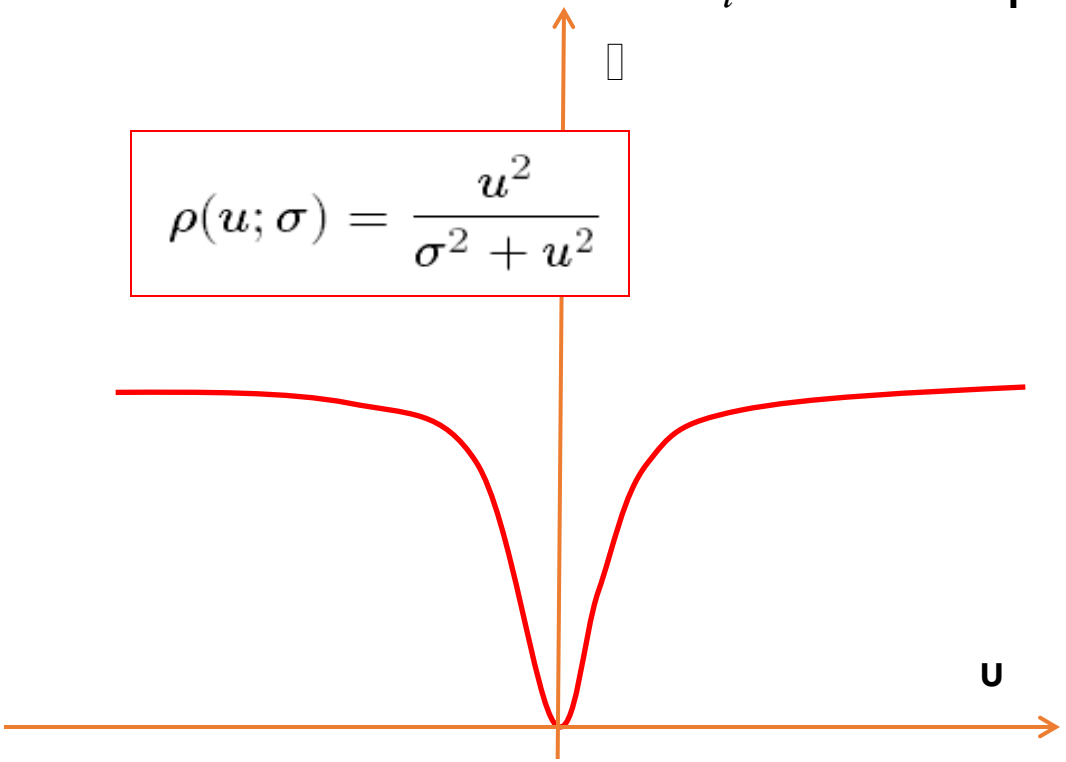
CONCLUSION: Least square is not robust w.r.t. outliers

Least squares: Robust estimators

Instead of minimizing $E = \sum_{i=1}^{n} (ax_i + by_i - d)^2$

We minimize $E = \sum_i \rho(u_i ; \sigma)$ $\quad u_i = ax_i + by_i - d$

- u_i = error (residual) of i[th] point w.r.t. model parameters β = (a,b,d)
- □ = robust function of u_i with scale parameter □



The robust function □

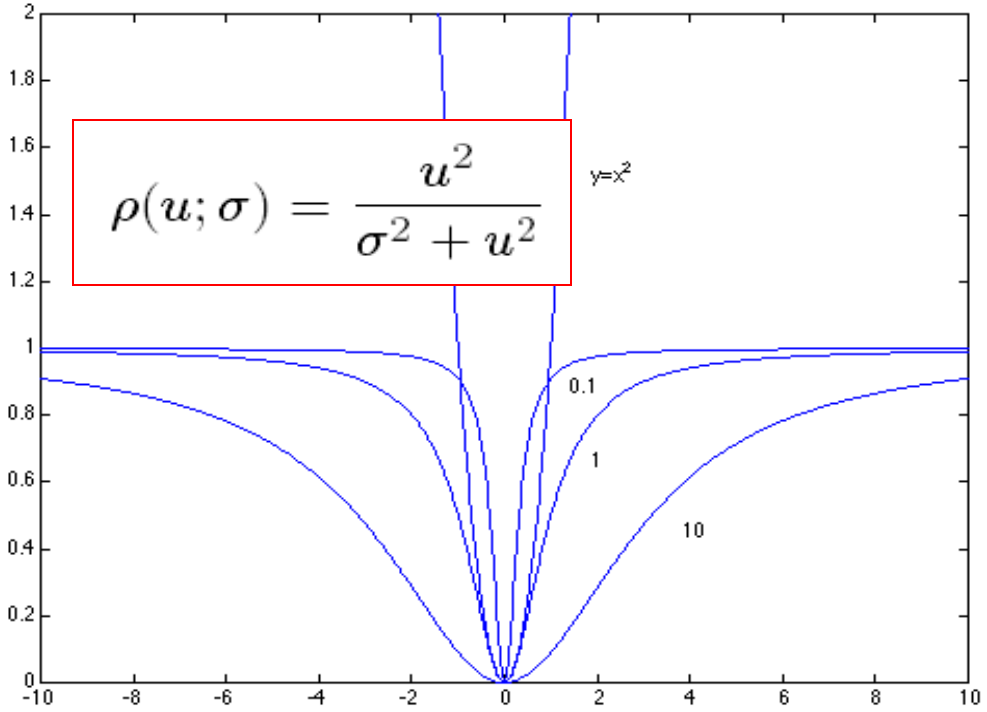
- Favors a configuration with small residuals
- Penalizes large residuals

Least squares: Robust estimators

Instead of minimizing $E = \sum_{i=1}^{n} (a x_i + b y_i - d)^2$

We minimize $E = \sum_i \rho(u_i ; \sigma)$ $\quad u_i = a x_i + b y_i - d$

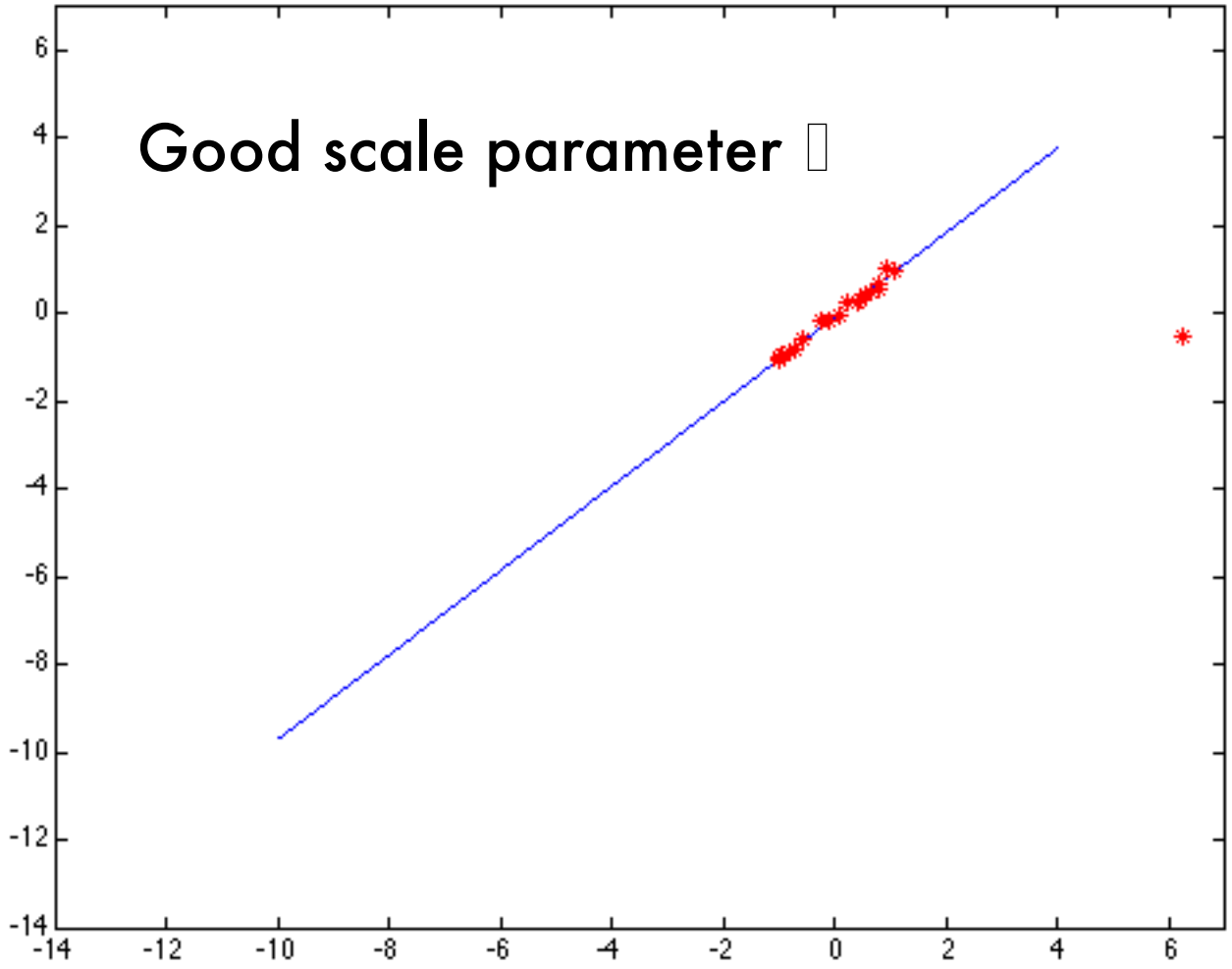
- u_i = error (residual) of i[th] point w.r.t. model parameters β = (a,b,d)
- ρ = robust function of u_i with scale parameter σ



The robust function ρ

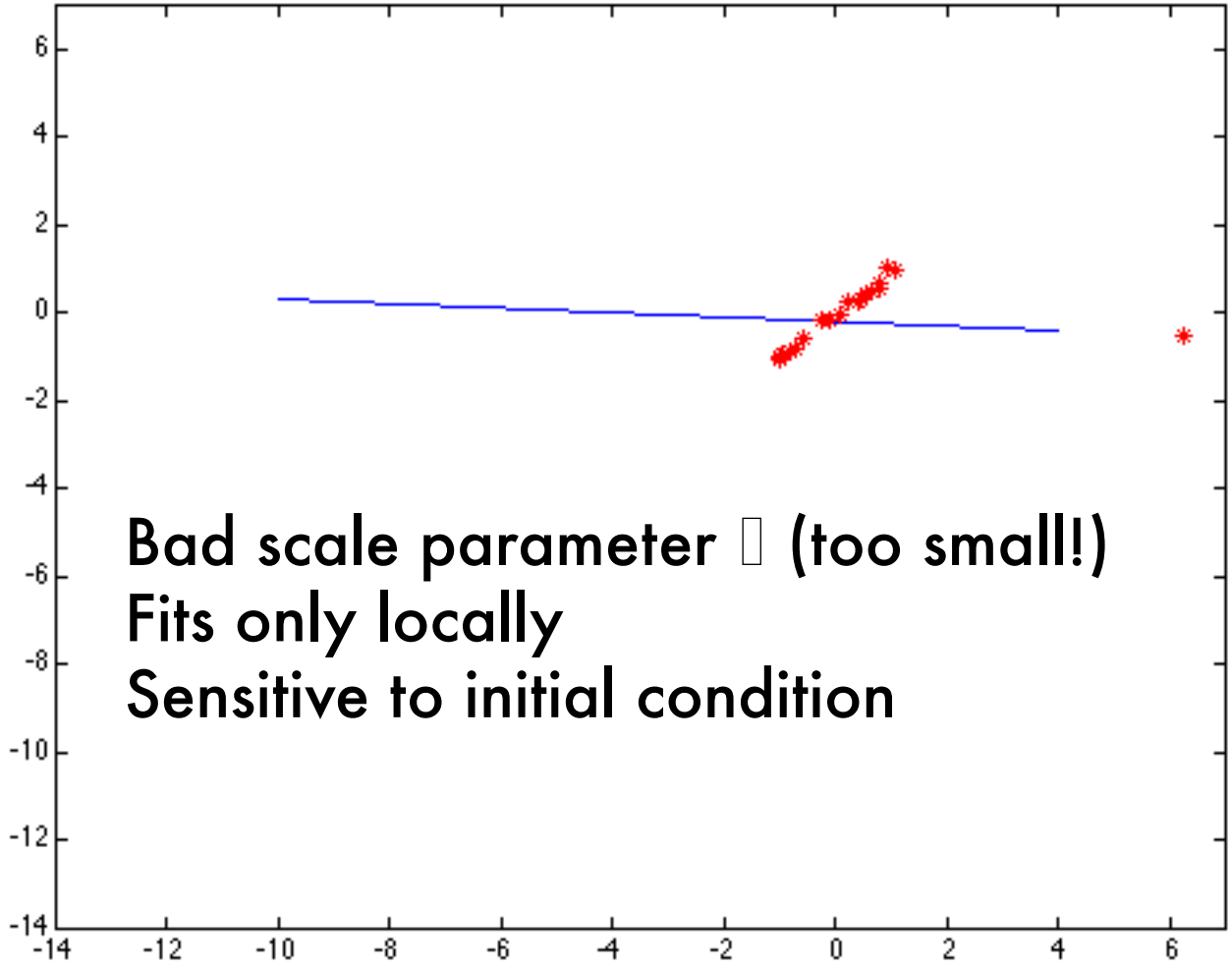
- Favors a configuration with small residuals
- Penalizes large residuals

Least squares: Robust estimators

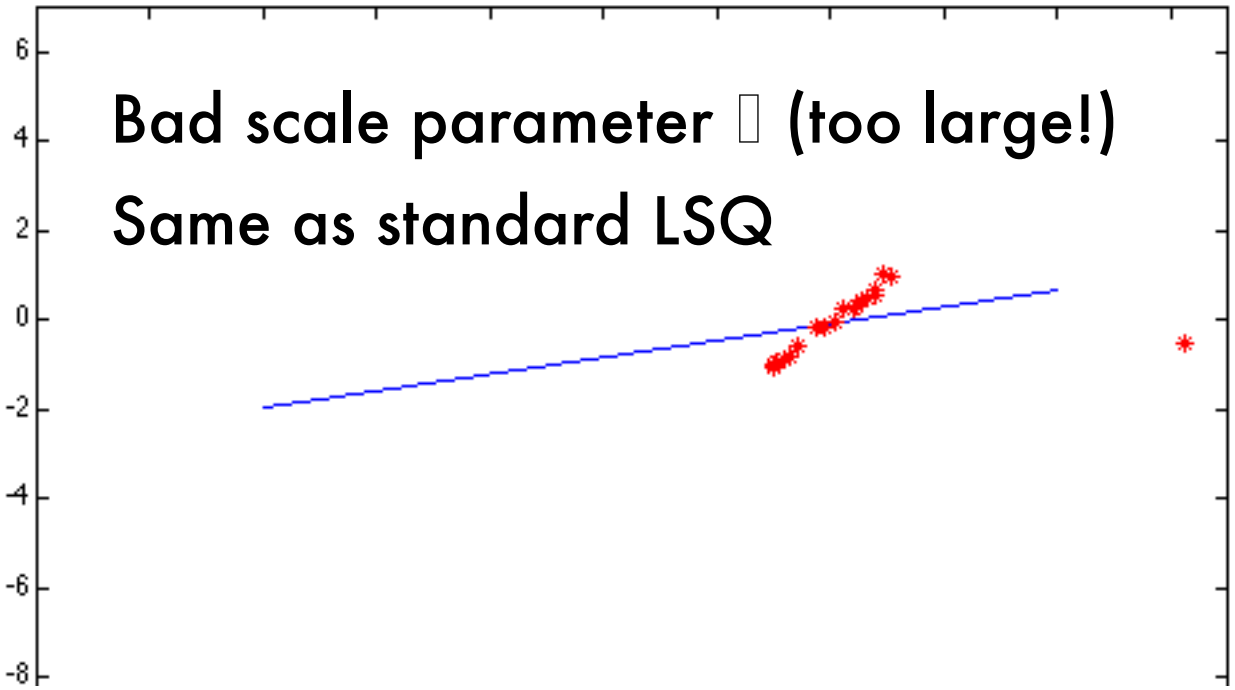


The effect of the outlier is eliminated

Least squares: Robust estimators



Least squares: Robust estimators



- CONCLUSION: Robust estimator useful if prior info about the distribution of points is known

- Robust fitting is a nonlinear optimization problem (iterative solution)
- Least squares solution provides good initial condition

Fitting

Goal: Choose a parametric model to fit a certain quantity from data

Techniques:

- Least square methods
- RANSAC
- Hough transform

Basic philosophy
(voting scheme)

- Data elements are used to vote for one (or multiple) models

- Robust to outliers and missing data

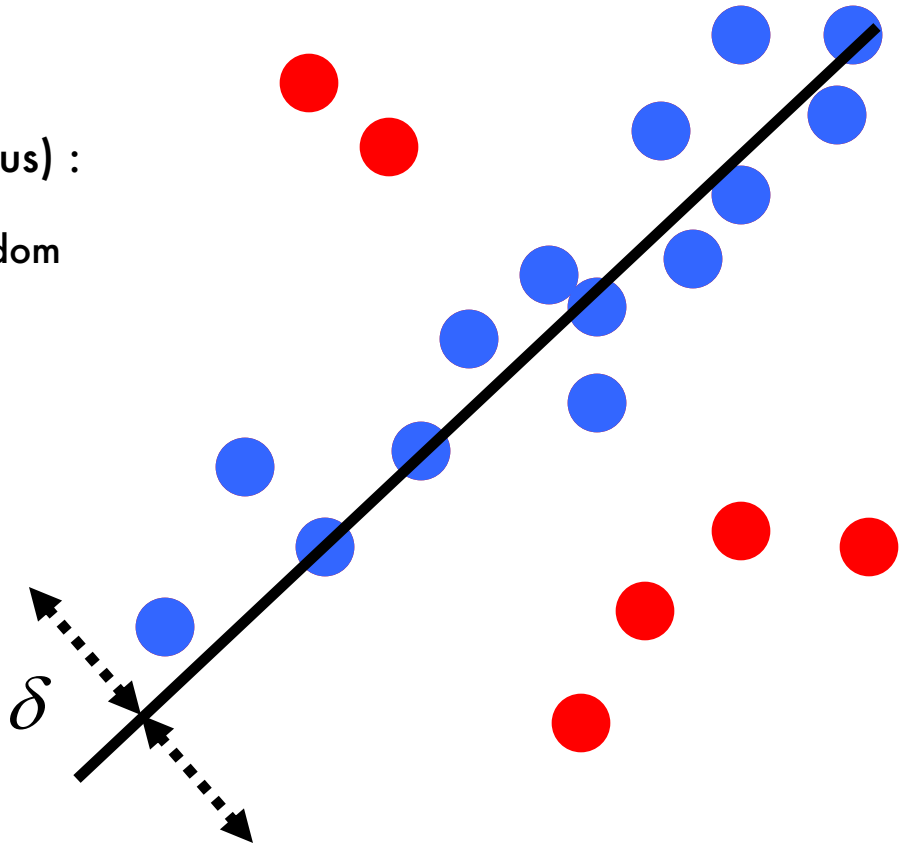
- Assumption1: Noise features will not vote consistently for any single model ("few" outliers)

- Assumption2: there are enough features to agree on a good model ("few" missing data)

RANSAC

(RANdom SAmple Consensus) :
Learning technique to estimate parameters of a model by random sampling of observed data

Fischler & Bolles in '81.

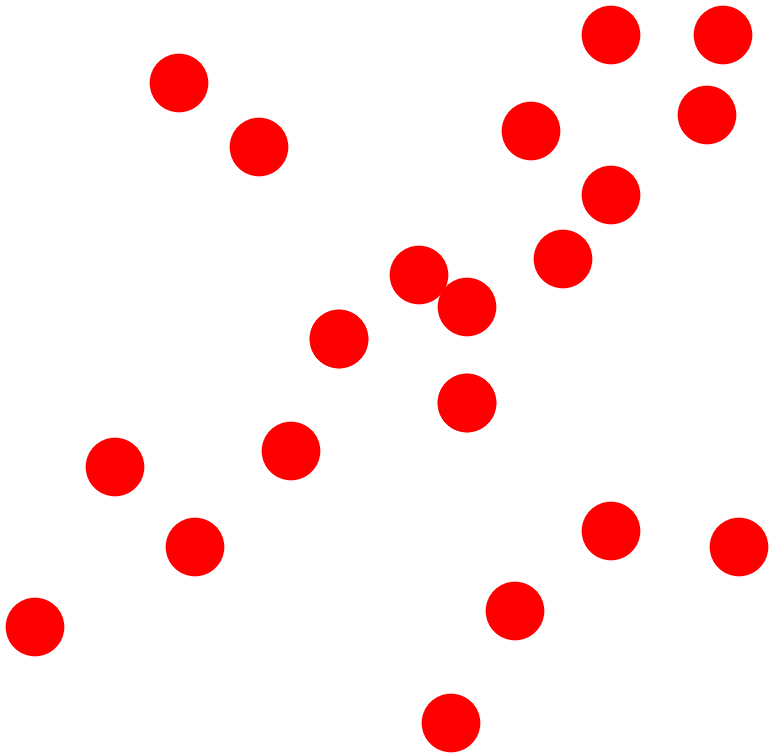


Model parameters

Evaluate

$$f(\boldsymbol{P}, \beta) < \delta$$

RANSAC



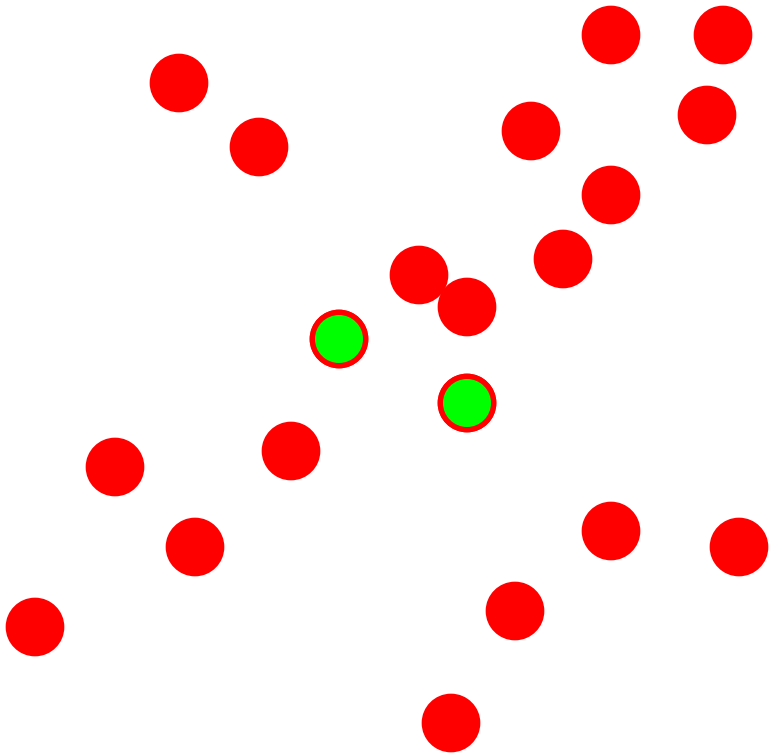
Sample set = set of points in 2D

Algorithm:

1. Select random sample of minimum required size to fit model
2. Compute a putative model from sample set
3. Compute the set of inliers to this model from whole data set

Repeat 1-3 until model with the most inliers over all samples is found

RANSAC



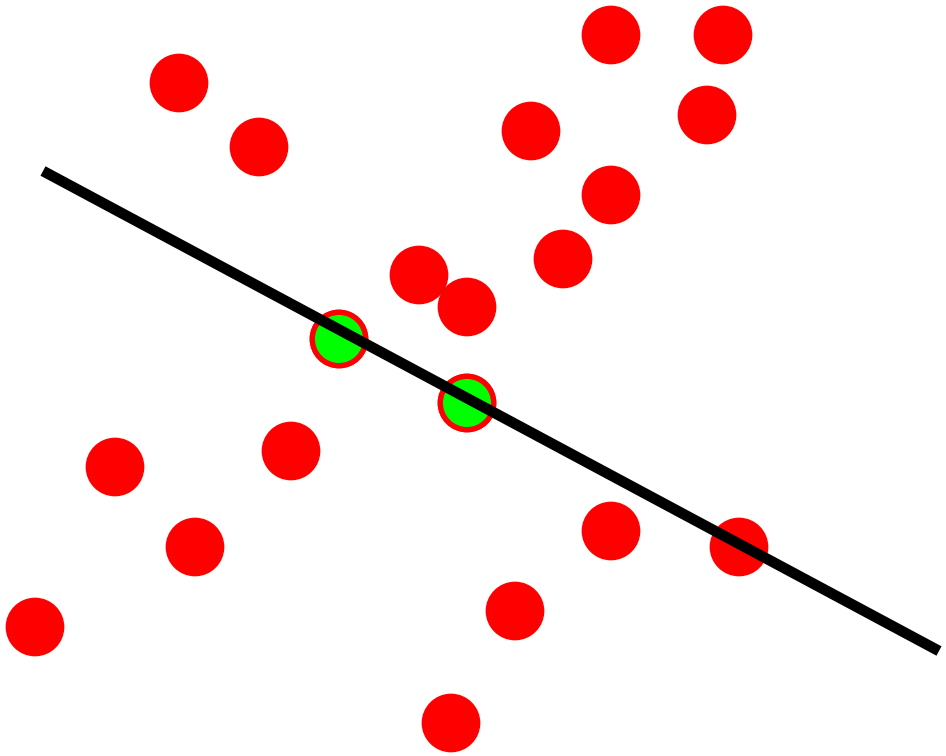
Sample set = set of points in 2D

Algorithm:

1. Select random sample of minimum required size to fit model [?]
2. Compute a putative model from sample set
3. Compute the set of inliers to this model from whole data set

Repeat 1-3 until model with the most inliers over all samples is found

RANSAC



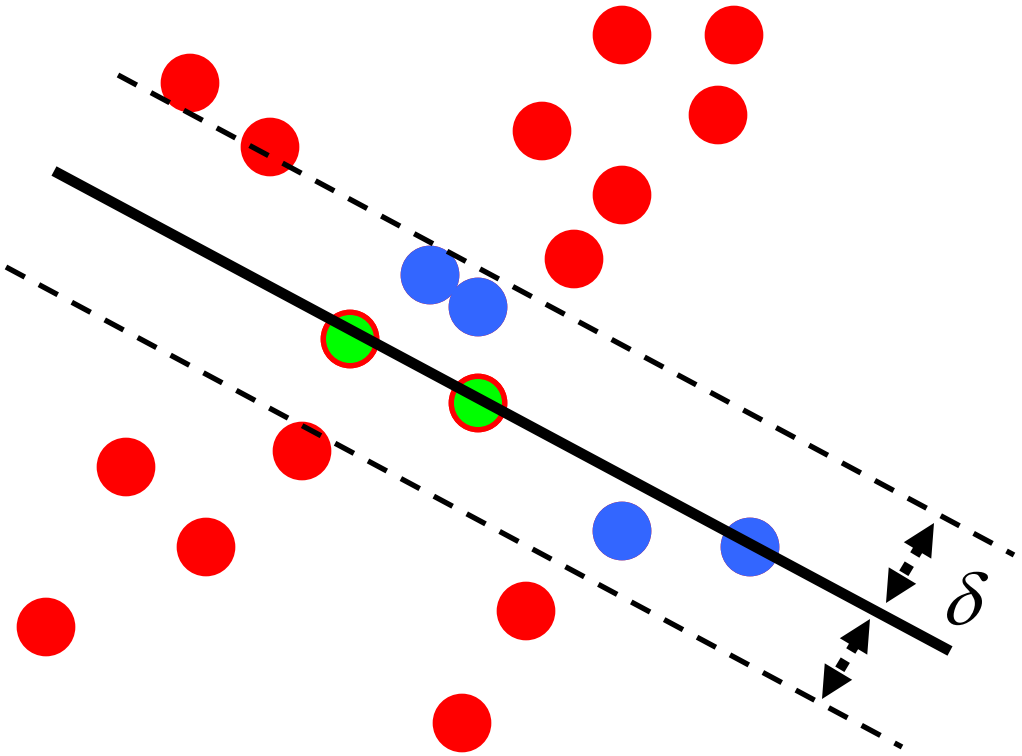
Sample set = set of points in 2D

Algorithm:

1. Select random sample of minimum required size to fit model [?]
2. Compute a putative model from sample set
3. Compute the set of inliers to this model from whole data set

Repeat 1-3 until model with the most inliers over all samples is found

RANSAC



Sample set = set of points in 2D

$|O| = 14$

Algorithm:

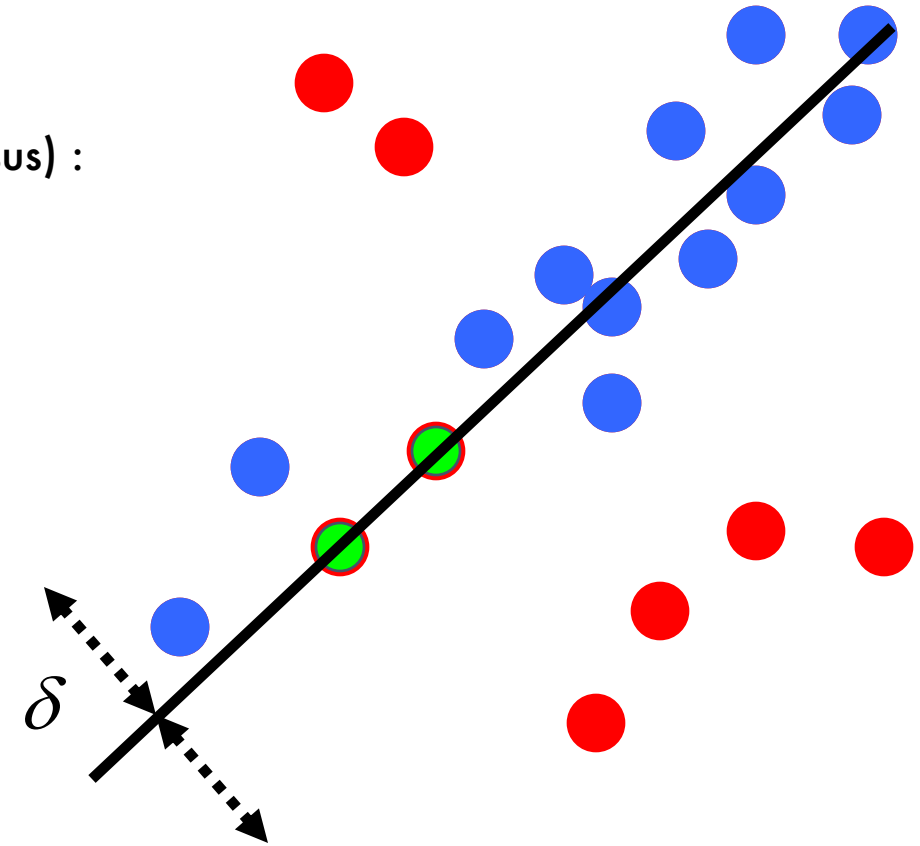
1. Select random sample of minimum required size to fit model [?]
2. Compute a putative model from sample set
3. Compute the set of inliers to this model from whole data set

Repeat 1-3 until model with the most inliers over all samples is found

RANSAC

(RANdom SAmple Consensus) :

Fischler & Bolles in '81.



$|O| = 6$

Algorithm:

1. Select random sample of minimum required size to fit model [?]
2. Compute a putative model from sample set
3. Compute the set of inliers to this model from whole data set

Repeat 1-3 until model with the most inliers over all samples is found

How many samples?

- Number of samples *N*
  - p = probability at least one random sample is free from outliers (e.g. *p*=0.99)
  - *e* = outlier ratio
  - s = minimum number needed to fit the model

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

	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

Estimating relative pose by RANSAC

Algorithm:

1. Select a random sample of minimum required size [?]
2. Compute a putative model from these
3. Compute the set of inliers to this model from whole sample space

Repeat 1-3 until model with the most inliers over all samples is found

Temporal Matching

- Temporally match features between frame t and t-1

Slides:C. Beall

Relative Pose Estimation/RANSAC

- Want to recover the incremental camera pose using the tracked features and triangulated landmarks
- There will be some erroneous stereo and temporal feature associations ! Use RANSAC
  - Select N out of M data items at random (the minimal set here is 3)
  - Estimate parameter (incremental pose from t-1 to t)
  - Find the number K of data items that fit the model (called inliers) within a given tolerance
  - Repeat S times
  - Compute refined model using full inlier set

Slides:C. Beall

RANSAC - conclusions

Good:

- Simple and easily implementable
- Successful in different contexts

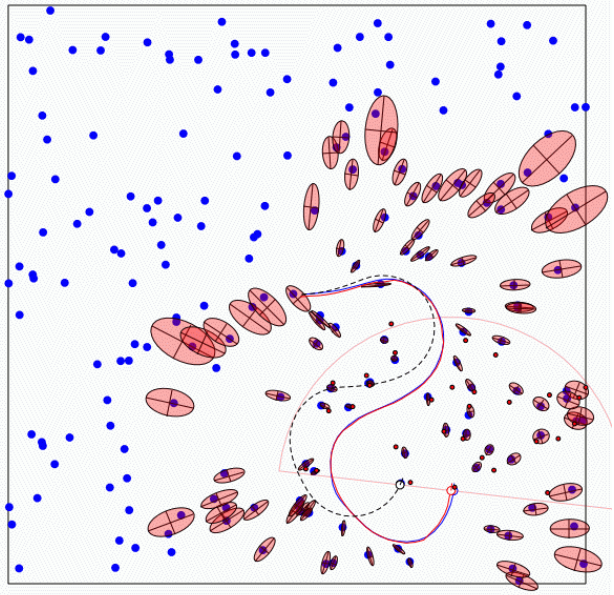
Bad:

- Many parameters to tune
- Trade-off accuracy-vs-time
- Cannot be used if ratio inliers/outliers is too small

SLAM

The SLAM Problem

A robot is exploring an unknown, static environment.



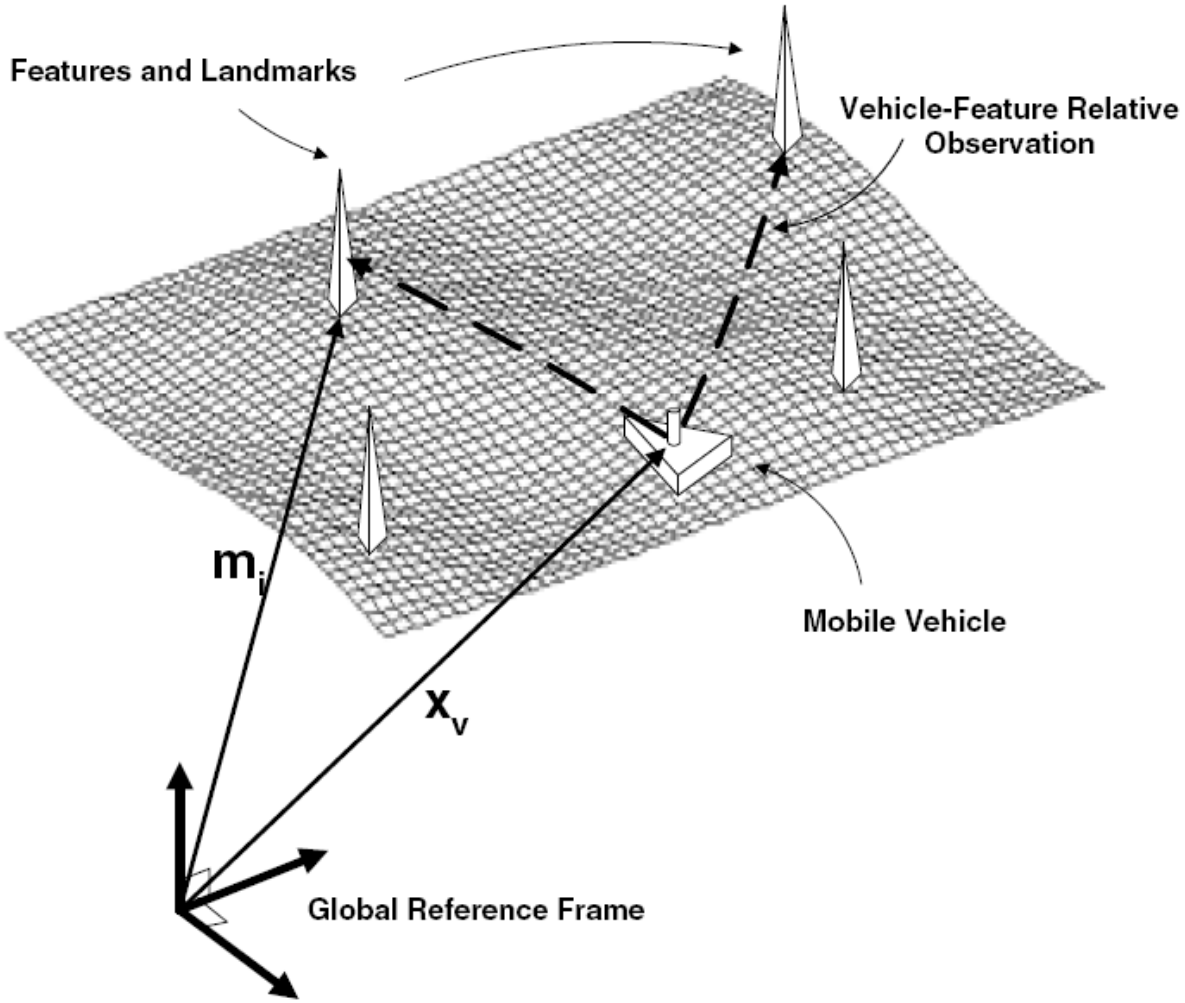
Given:

- The robot's controls
- Observations of nearby features

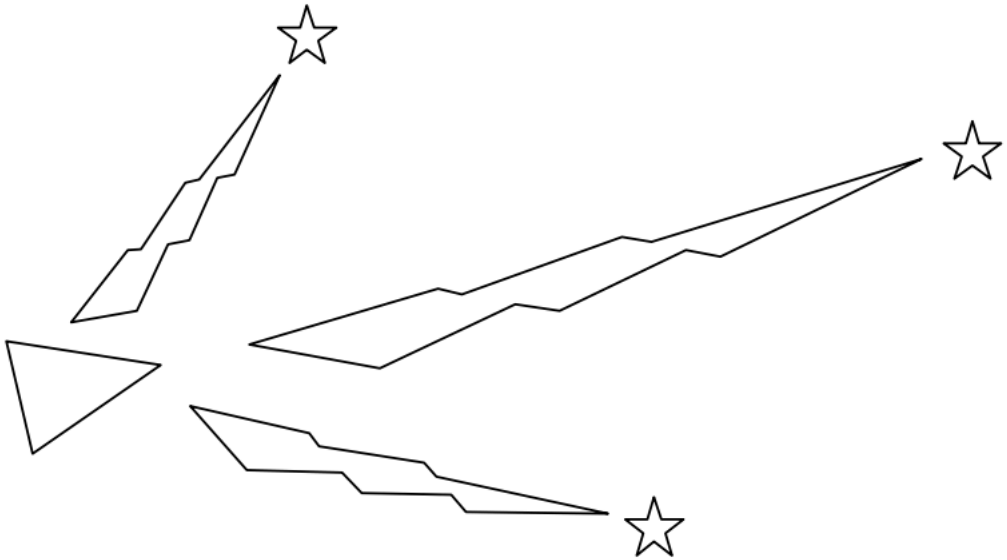
Estimate:

- Map of features
- Path of the robot

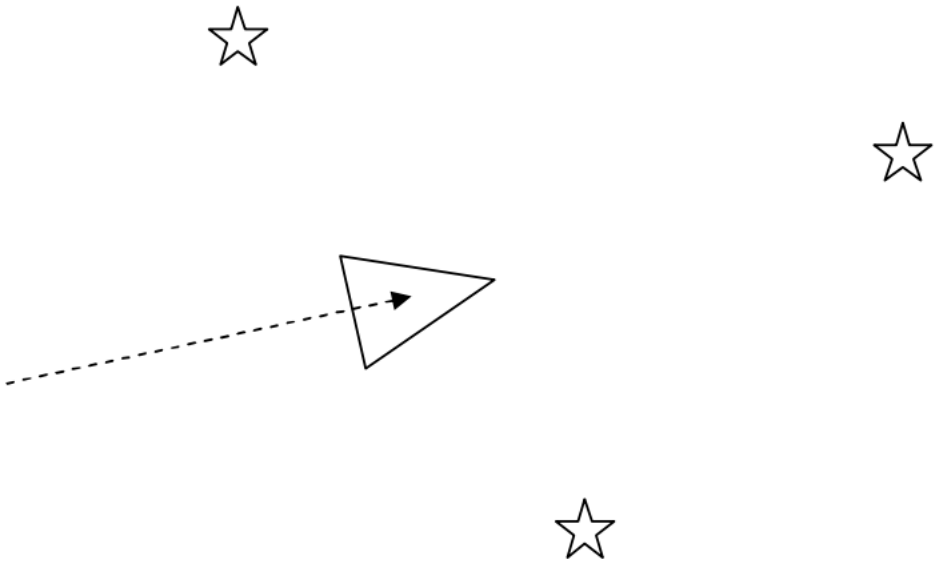
Structure of the Landmark-based SLAM-Problem



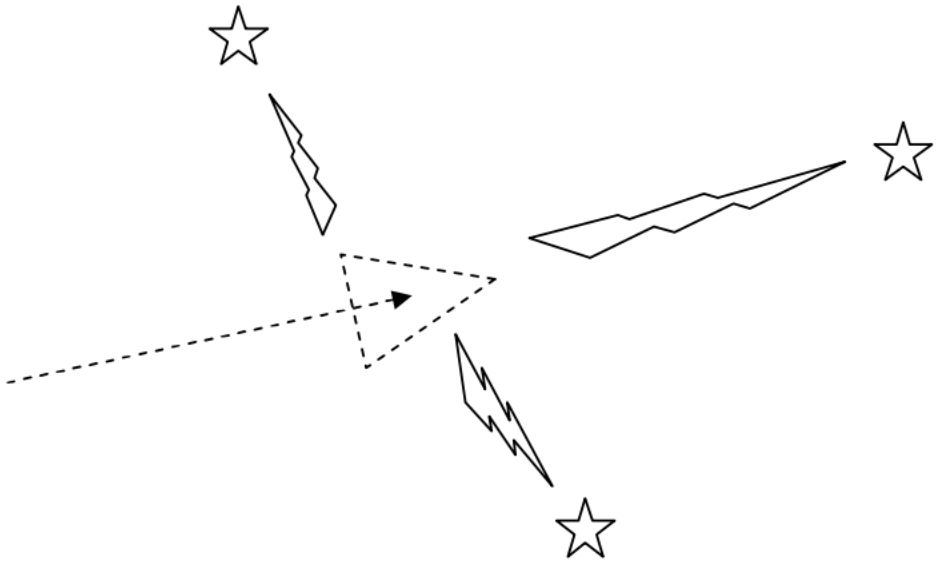
Sensor observations

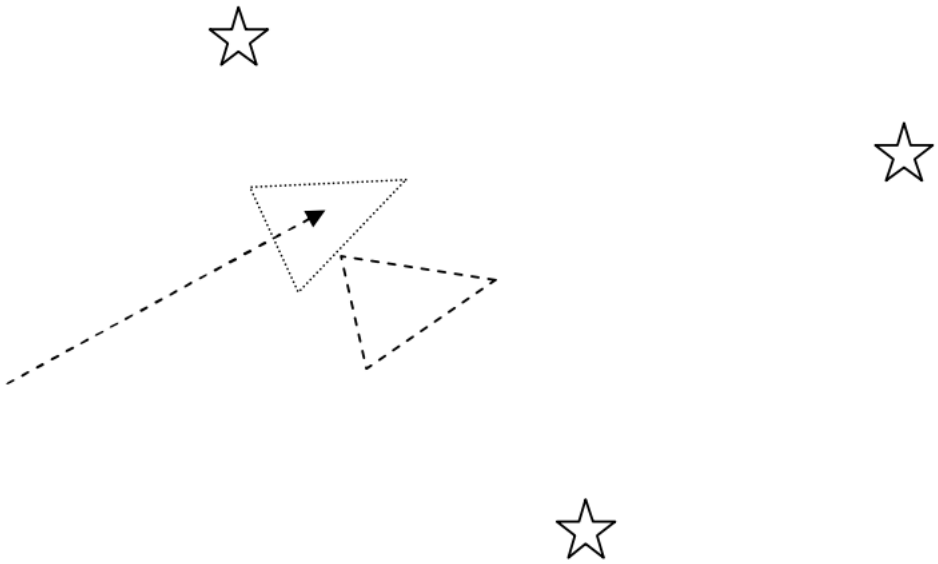


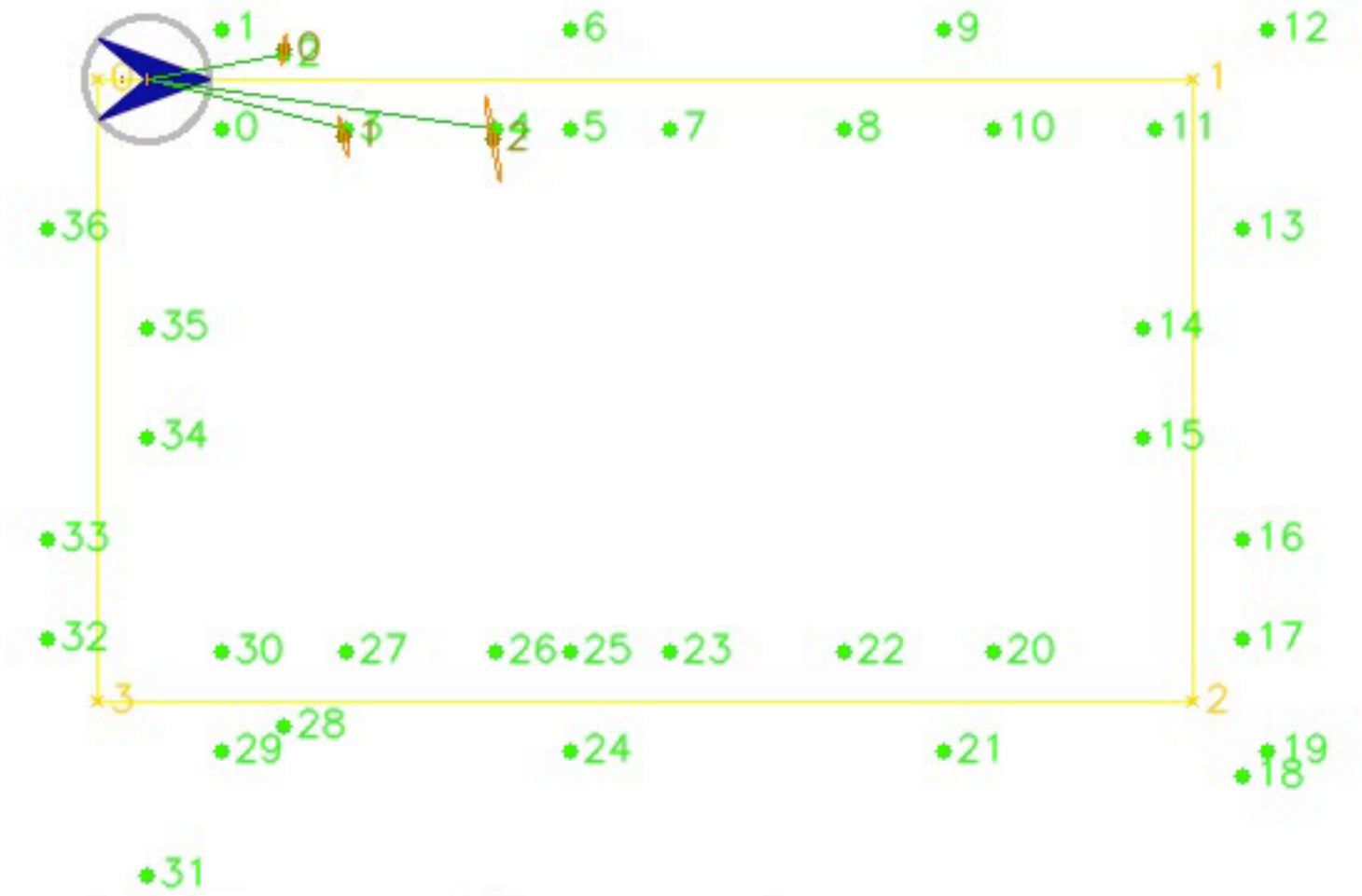
Wheel Odometry



Sensor observations







[Q]uit. [P]redict step ON Robot speed: 10
[H]alt. [C]orrect step ON [F]aster [S]lowdown

Self-Driving SLAM

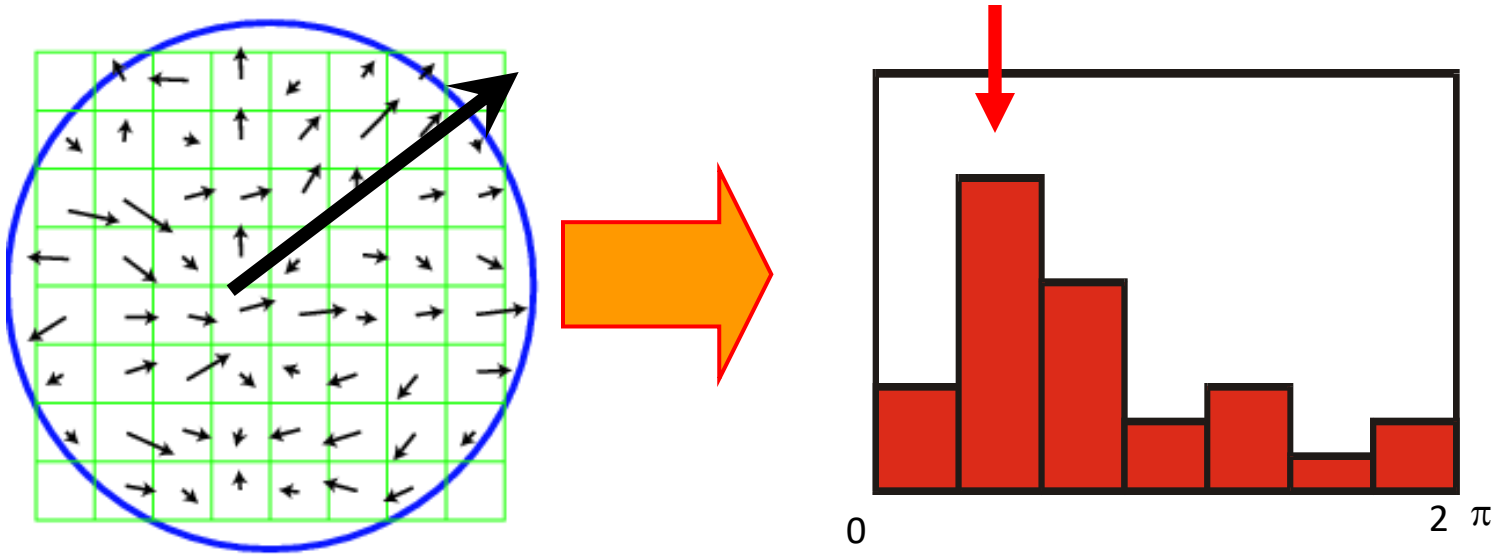
- Wheel odometry
- LIDAR
- Camera

Loop closure

Slides:C. Beall

Rotational invariance

- Find dominant orientation by building smoothed orientation histogram
- Rotate all orientations by the dominant orientation

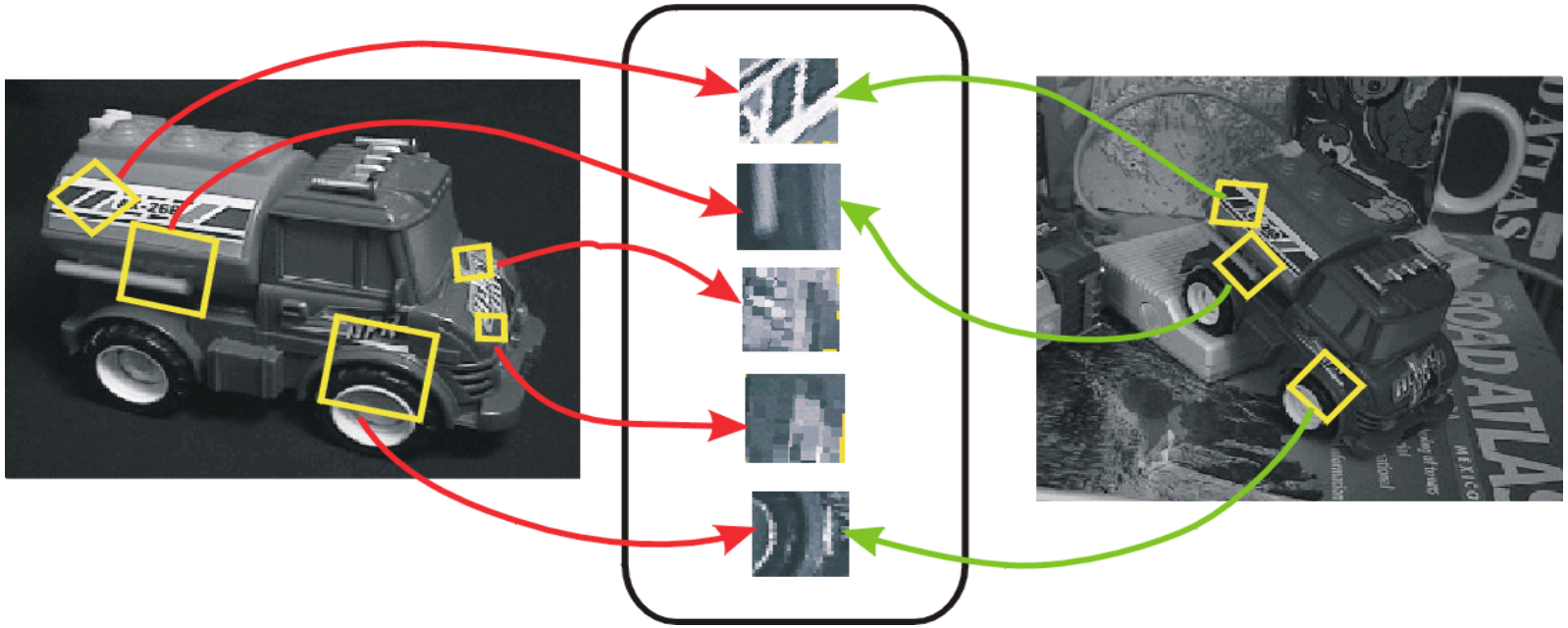


This makes the SIFT descriptor rotational invariant

Rotational invariance

(b)

Rotational invariance

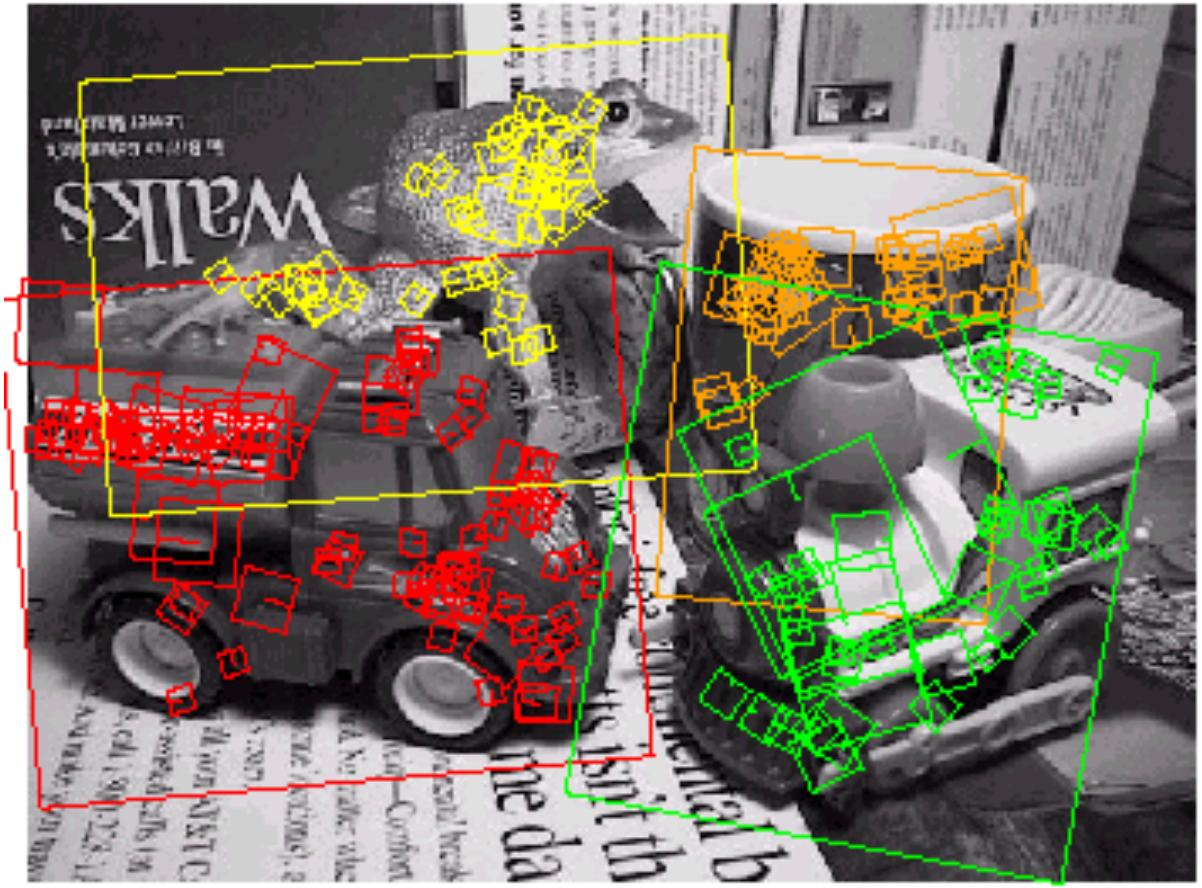


Matching using SIFT

David G. Lowe. **"Distinctive image features from scale-invariant keypoints."** *IJCV* 60 (2), 04

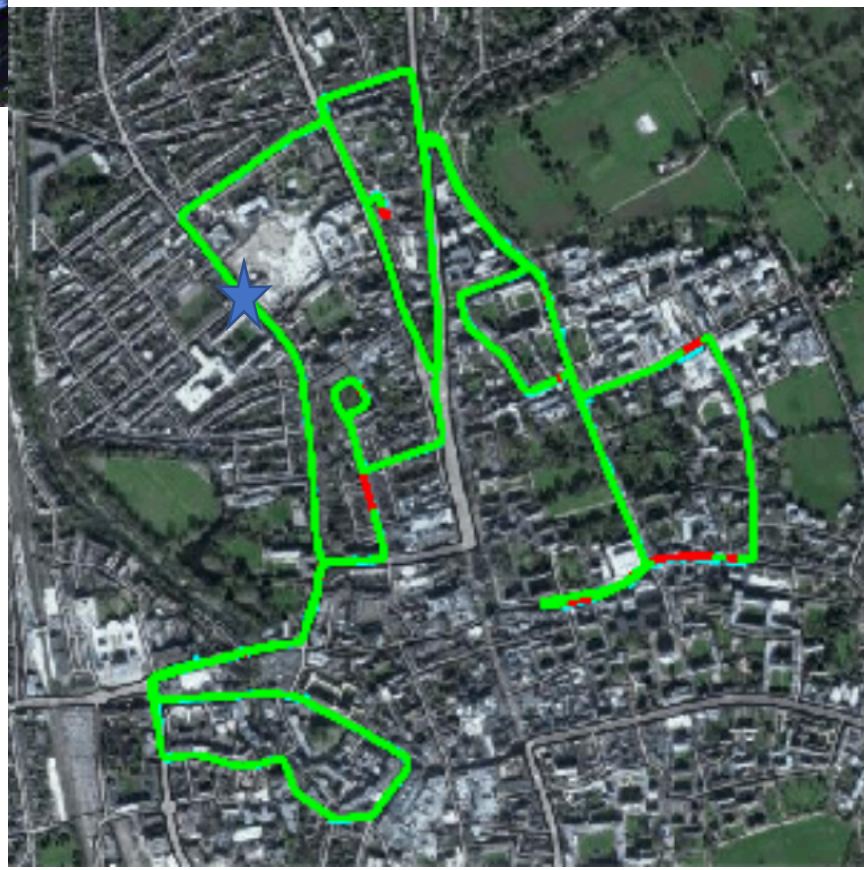
Matching using SIFT

David G. Lowe. **"Distinctive image features from scale-invariant keypoints."** *IJCV* 60 (2), 04



Slides:C. Beall

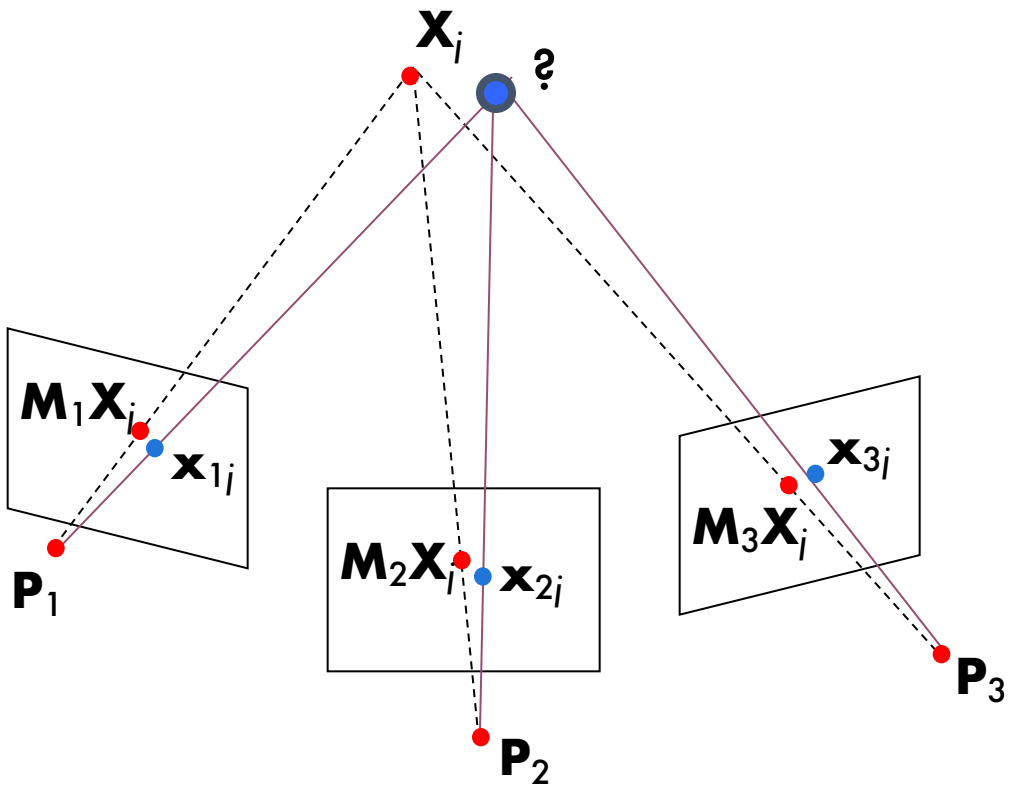
Object → Bag of ‘words’

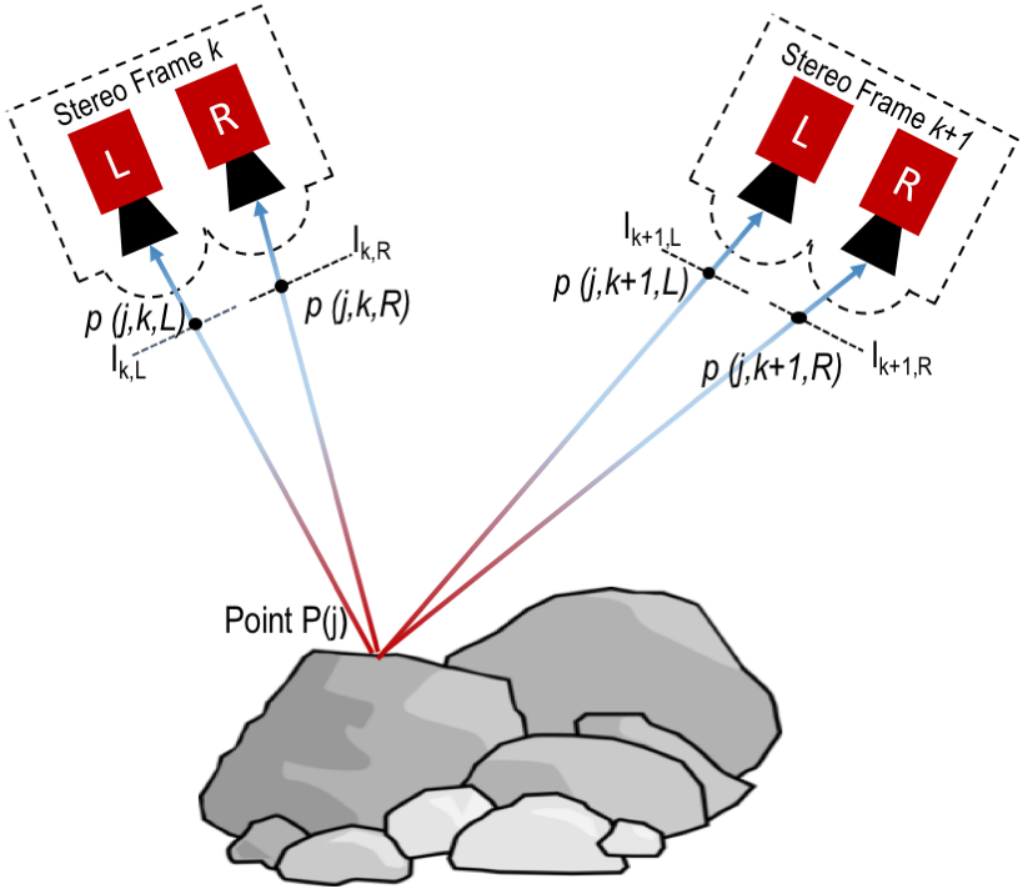


Bundle adjustment

- Non-linear method for refining structure and motion
- Minimizing re-projection error

$$E(M, \mathbf{X}) = \sum_{i=1}^{m} \sum_{j=1}^{n} D\left(\mathbf{x}_{ij}, M_i \mathbf{X}_j\right)^2$$




Stereo Frame k
L
R
$I_{k,R}$
$p\ (j,k,L)$
$p\ (j,k,R)$
$I_{k,L}$
Stereo Frame k+1
L
R
$I_{k+1,L}$
$p\ (j,k+1,L)$
$p\ (j,k+1,R)$
$I_{k+1,R}$
Point P(j)

Next time machine learning…