

Introduction to Aerial Robotics

Lecture 6

Shaojie Shen
Associate Professor
Dept. of ECE, HKUST



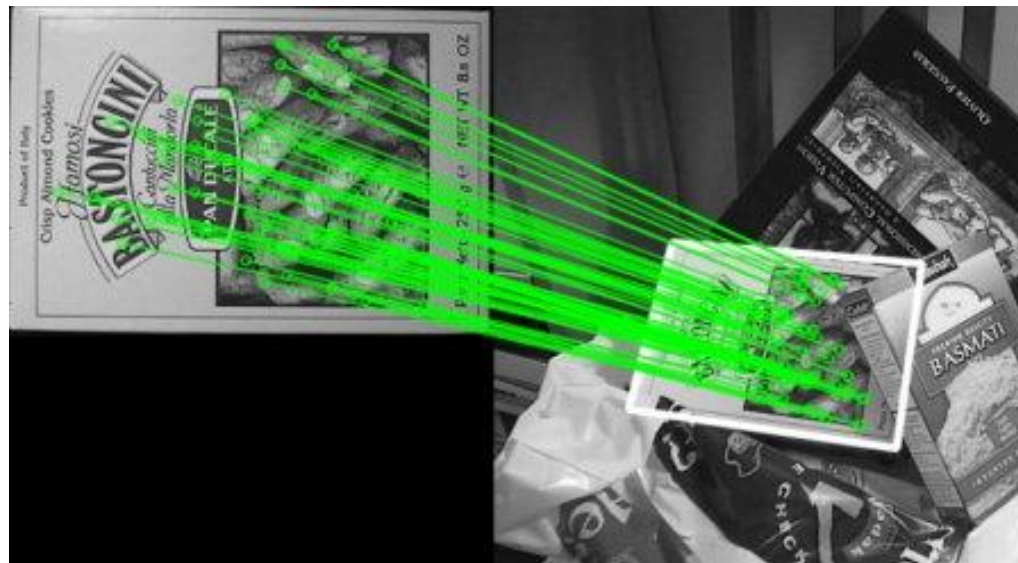
21 March 2023

Outline

- Feature Matching
- 3D-3D Pose Estimation
- 3D-2D Pose Estimation
- Outlier Rejection and Robust Estimation

Review: Feature Detection and Matching

- Detect corners features in both images
- Use image patch as feature description
 - Could be extended to color, texture, SIFT/HOG descriptor
- Find correspondences using descriptor matching



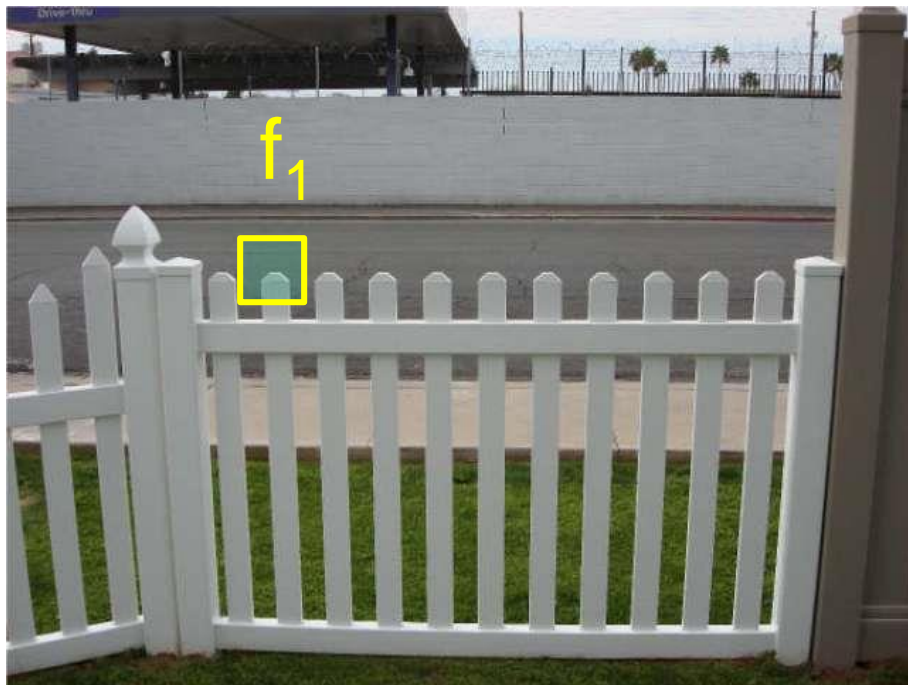
Feature matching

Given a feature in I_1 , how to find the best match in I_2 ?

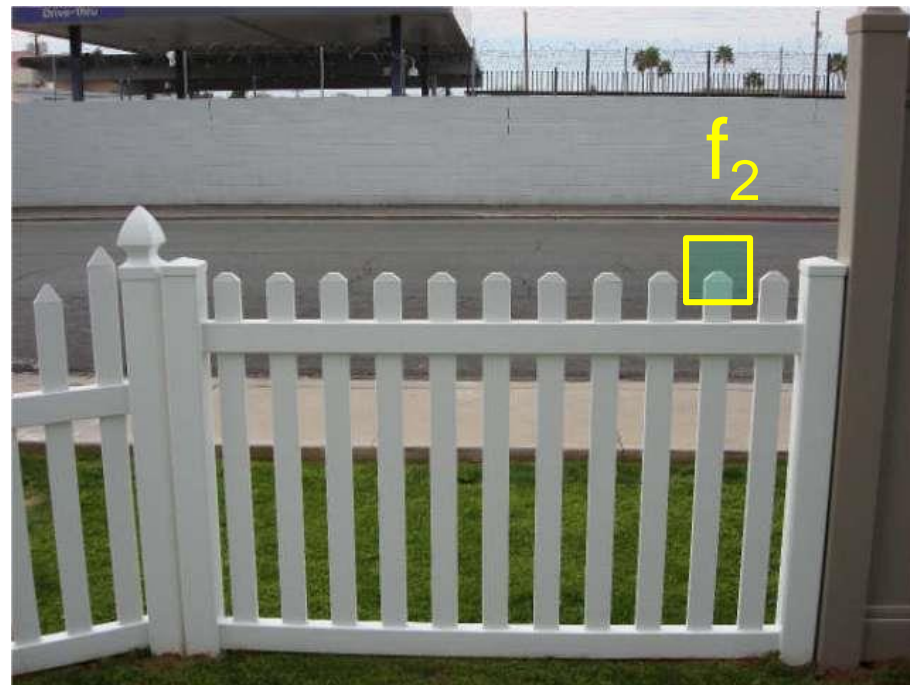
1. Define distance function that compares two descriptors
2. Test all the features in I_2 , find the one with min distance

Feature distance

- How to define the difference between two features f_1, f_2 ?
 - Simple approach is $SSD(f_1, f_2)$
 - sum of square differences between entries of the two descriptors
 - can give good scores to very ambiguous (bad) matches



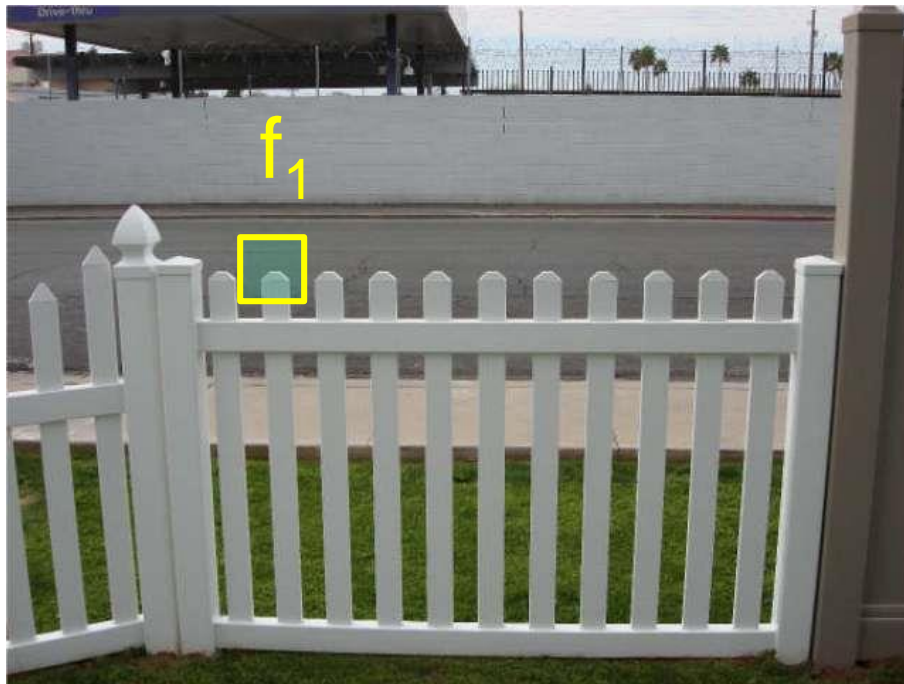
I_1



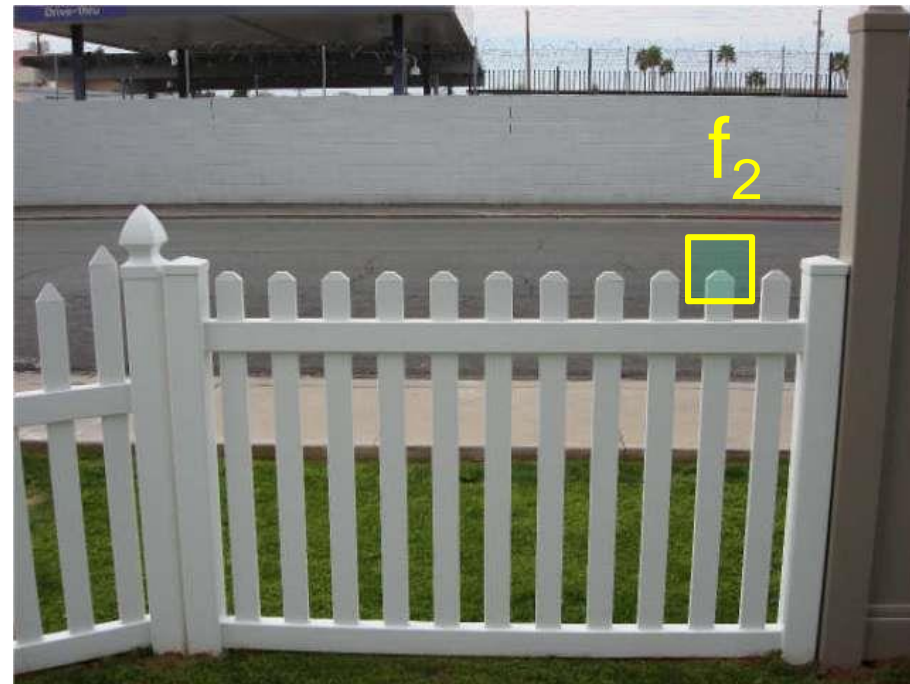
I_2

Feature distance

- How to define the difference between two features f_1, f_2 ?
 - Better approach: ratio distance = $\text{SSD}(f_1, f_2) / \text{SSD}(f_1, f_2')$
 - f_2 is best SSD match to f_1 in I_2
 - f_2' is 2nd best SSD match to f_1 in I_2
 - gives large values (close to 1) for ambiguous matches



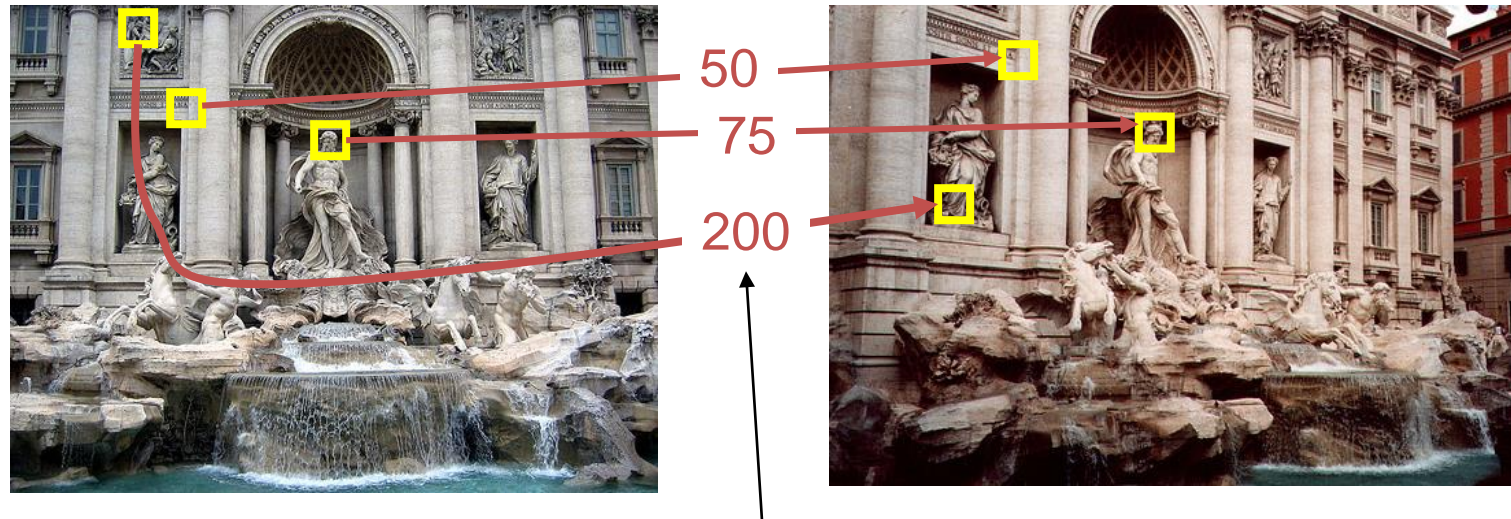
I_1



I_2

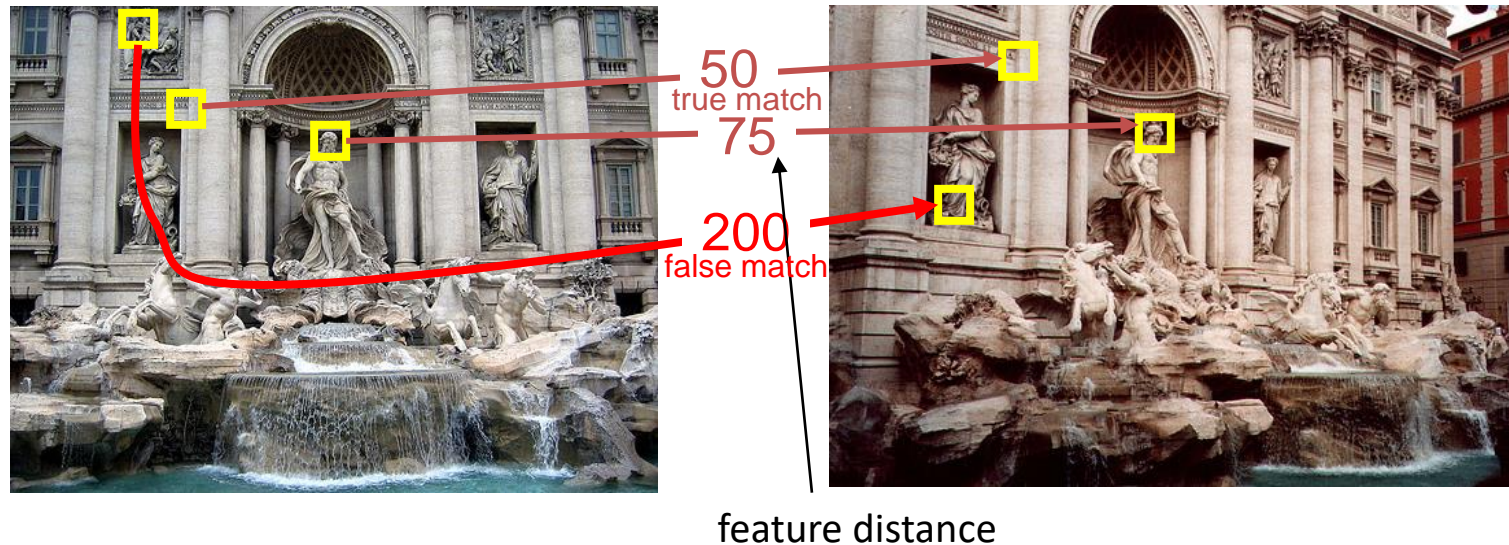
Evaluating the results

How can we measure the performance of a feature matcher?



feature distance

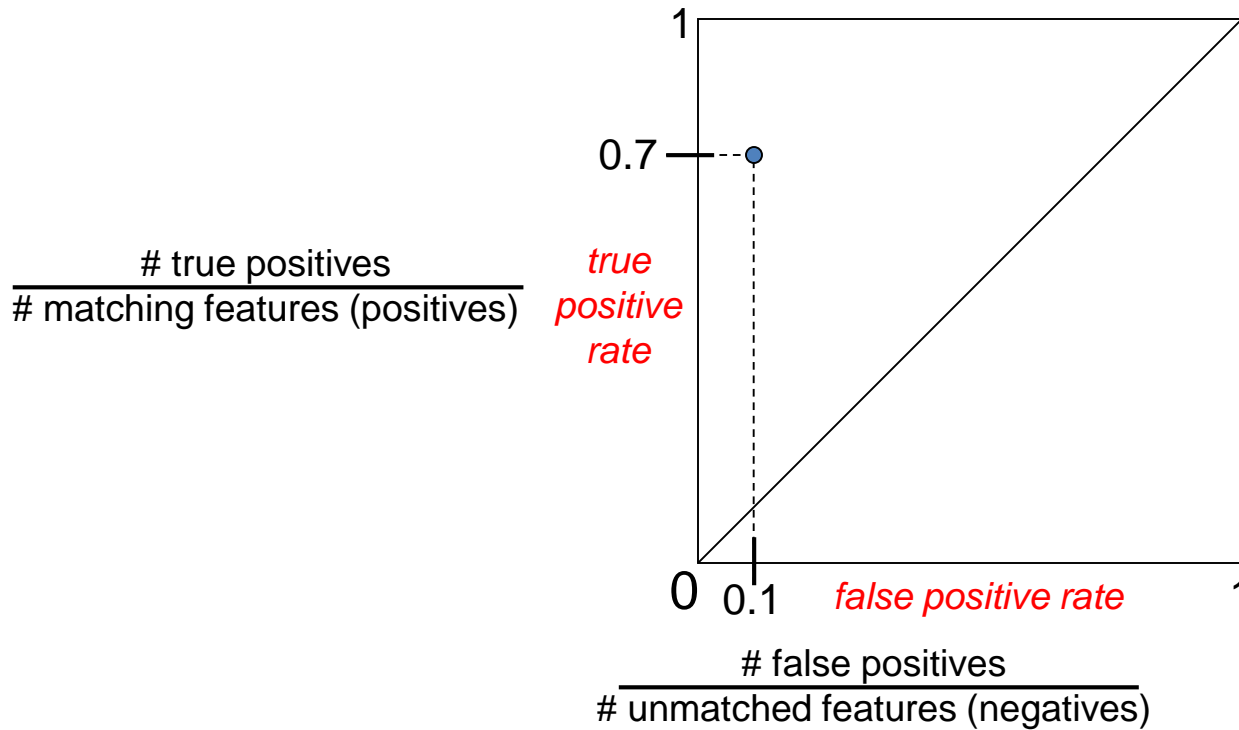
True/false positives



- The distance threshold affects performance
 - True positives = # of detected matches that are correct
 - Suppose we want to maximize these—how to choose threshold?
 - False positives = # of detected matches that are incorrect
 - Suppose we want to minimize these—how to choose threshold?

Evaluating the results

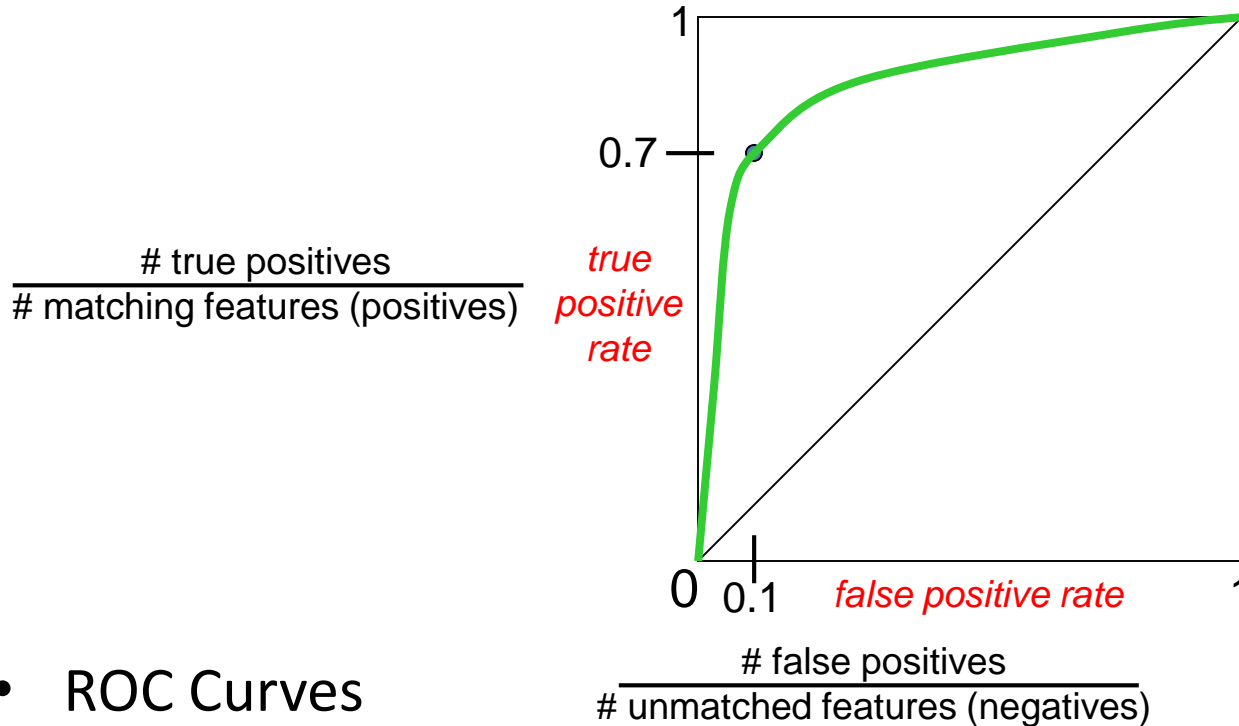
How can we measure the performance of a feature matcher?



Evaluating the results

- How can we measure the performance of a feature matcher?

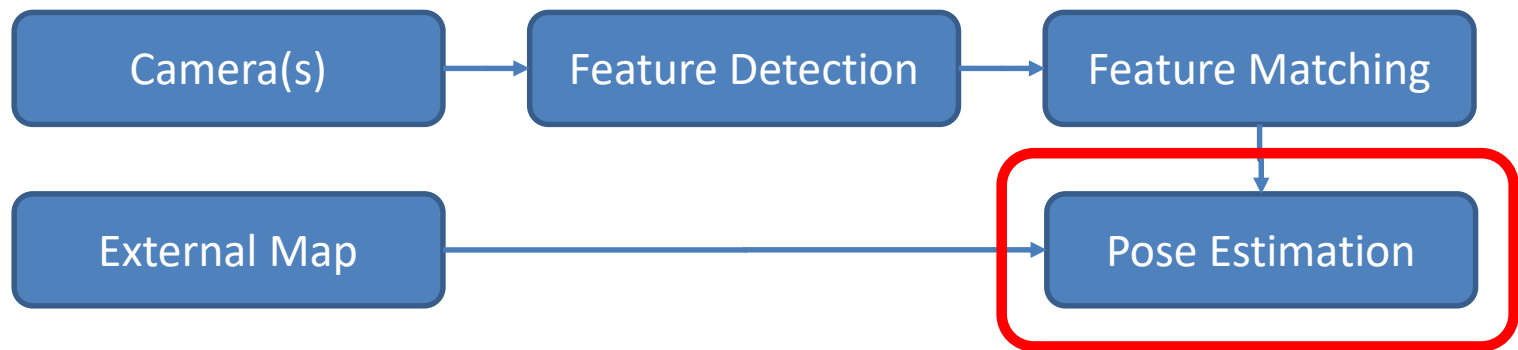
ROC curve ("Receiver Operator Characteristic")



ROC Curves

- Generated by counting # current/incorrect matches, for different thresholds
- Want to maximize area under the curve (AUC)
- Useful for comparing different feature matching methods

Vision-based Pose Estimation Pipeline (aka. Map-based Localization)

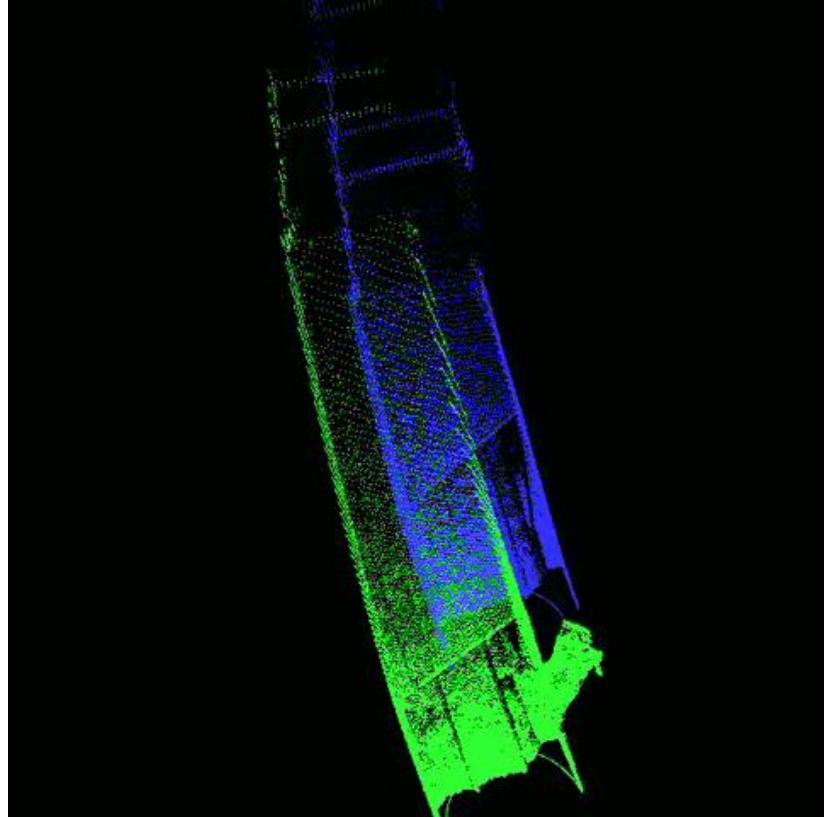
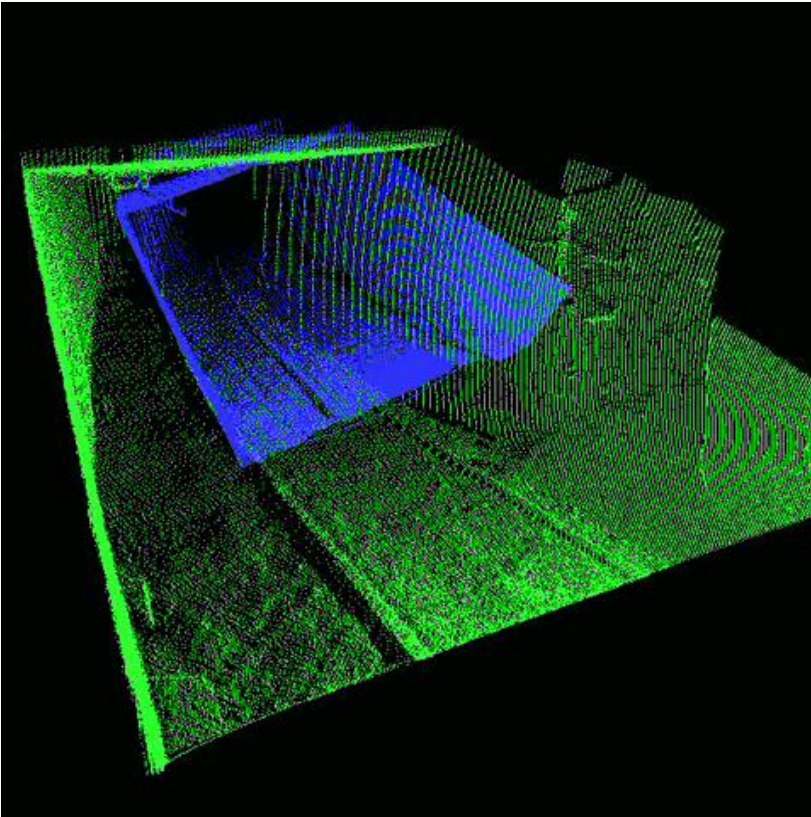


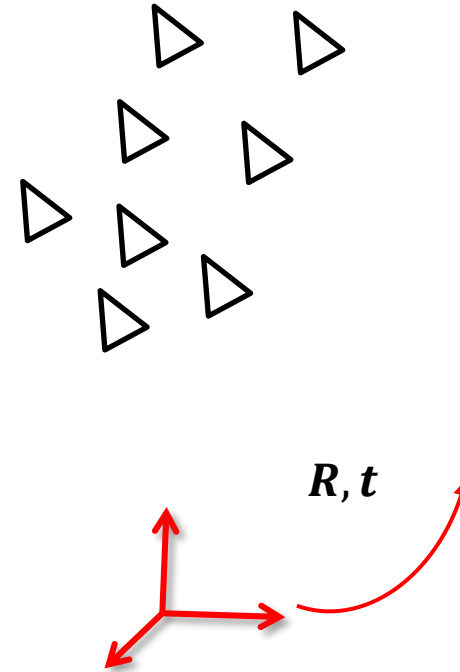
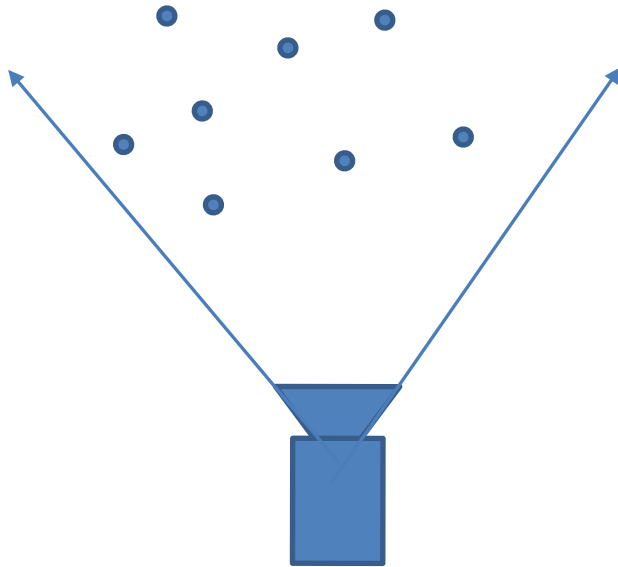
Vision-based Pose Estimation Pipeline (aka. Map-based Localization)

- Setup
 - Applicable to different camera configurations (monocular, stereo, RGB-D, etc.)
 - Sufficient illumination and texture
 - Dominance of static scene
 - Known map
 - Sufficient observation of map features
 - Focus on global consistency


3D-3D Pose Estimation (Point Cloud Registration)


3D-3D Registration of Point Cloud



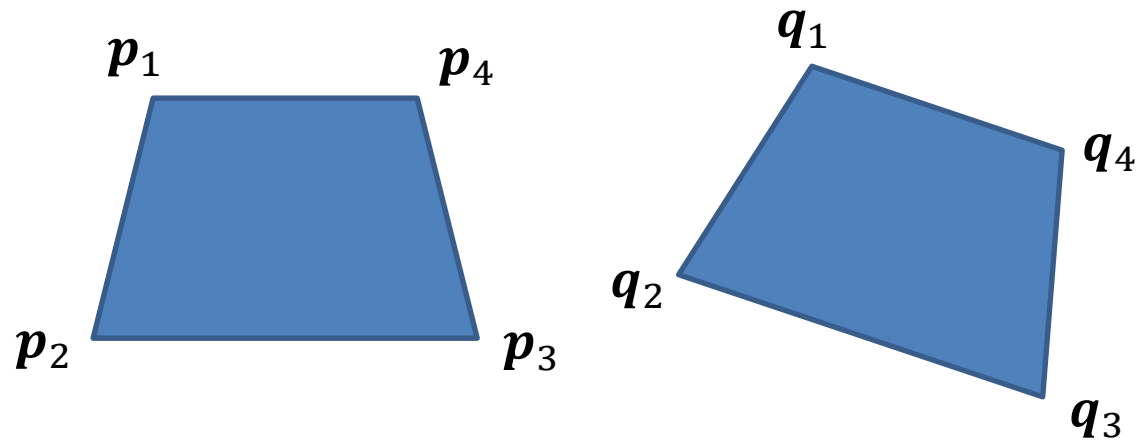


3D-3D


 3D points in frame 1
 (or known 3D points in world frame)


 3D points in frame 2
 (or 3D point observations in body frame)

3D-3D Pose Estimation

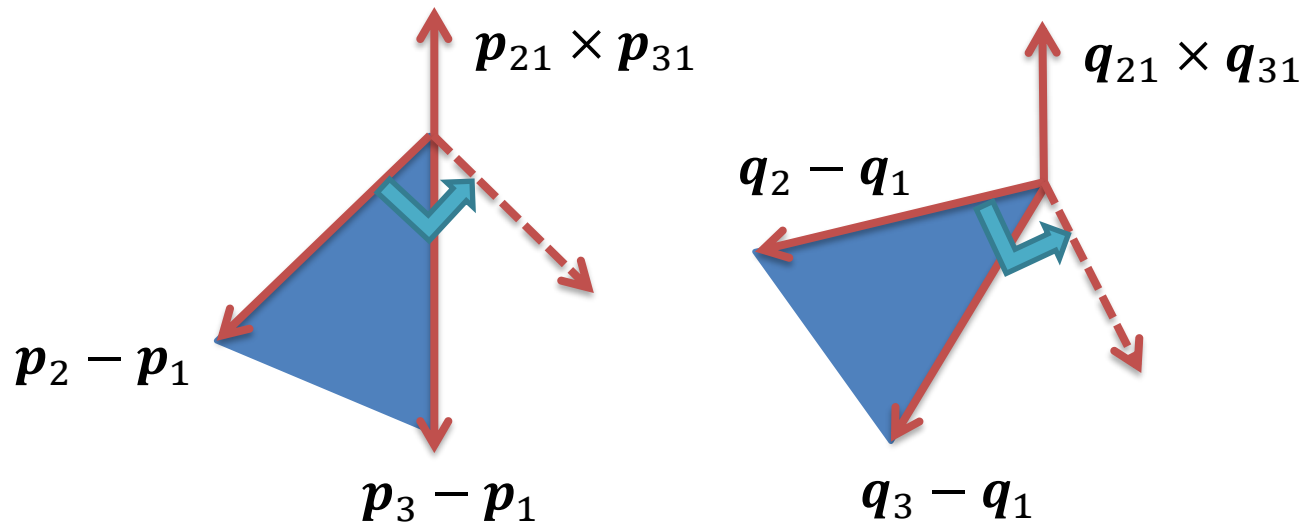


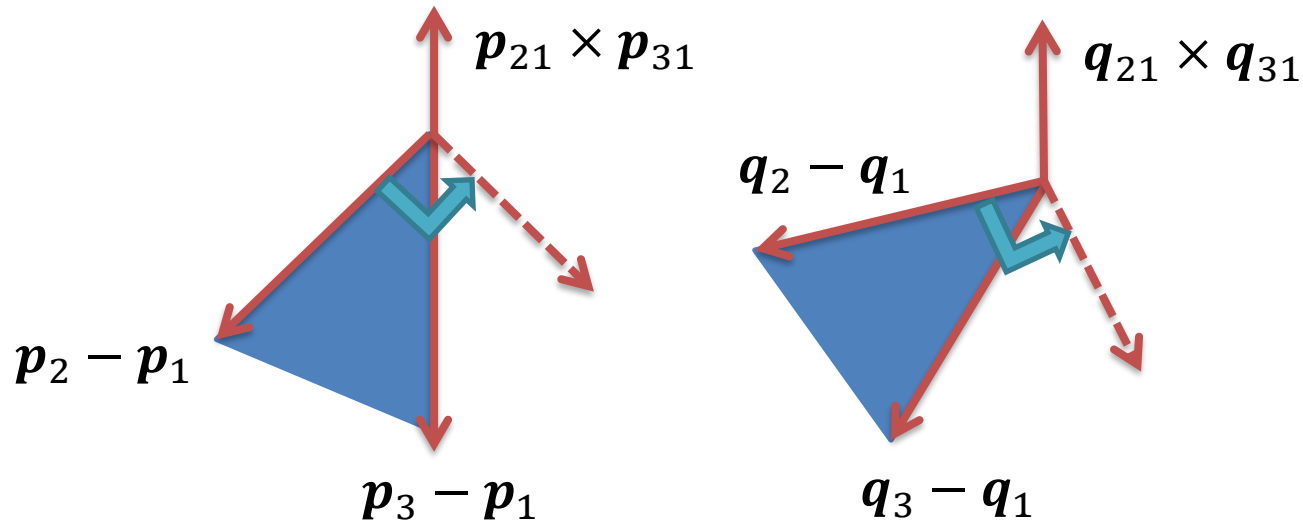
- How do we solve for R , t from point correspondences?

$$p_i = Rq_i + t$$

3D-3D Pose Estimation

- What is the minimal number of points needed?





- Three non-collinear points suffice: each triangle $\mathbf{p}_{i=1\dots 3}$ and $\mathbf{q}_{i=1\dots 3}$ make an orthogonal basis

$$(\mathbf{p}_{21} \quad (\mathbf{p}_{21} \times \mathbf{p}_{31}) \times \mathbf{p}_{21} \quad \mathbf{p}_{21} \times \mathbf{p}_{31})$$

and

$$(\mathbf{q}_{21} \quad (\mathbf{q}_{21} \times \mathbf{q}_{31}) \times \mathbf{q}_{21} \quad \mathbf{q}_{21} \times \mathbf{q}_{31})$$

- Rotation between two orthogonal bases is unique.



- We solve a minimization problem for $N \geq 3$ point correspondences:

$$\min_{\mathbf{R}, \mathbf{t}} \sum_i^N \|\mathbf{p}_i - (\mathbf{R}\mathbf{q}_i + \mathbf{t})\|^2$$

- After differentiating with respect to \mathbf{t} , we observe that the translation is the difference between the centroids:

$$\mathbf{t} = \frac{1}{N} \sum_i^N \mathbf{p}_i - \mathbf{R} \frac{1}{N} \sum_i^N \mathbf{q}_i = \bar{\mathbf{p}} - \mathbf{R}\bar{\mathbf{q}}$$

- We subtract the centroids $\bar{\mathbf{p}}$ and $\bar{\mathbf{q}}$ and rewrite the objective function as

$$\min_R \|\mathbf{P} - \mathbf{RQ}\|_F^2$$

where

$$\mathbf{Q} = [\mathbf{p}_1 - \bar{\mathbf{p}}, \dots, \mathbf{p}_N - \bar{\mathbf{p}}]$$

and

$$\mathbf{P} = [\mathbf{q}_1 - \bar{\mathbf{q}}, \dots, \mathbf{q}_N - \bar{\mathbf{q}}]$$

- Some useful mathematics

- Frobenius norm $\|\mathbf{A}\|_F = \sqrt{\sum \sum |a_{ij}|^2} \rightarrow \|\mathbf{A}\|_F = \sqrt{\text{tr}(\mathbf{A}\mathbf{A}^H)}$
- $\text{tr}(\mathbf{AB}) = \text{tr}(\mathbf{BA})$
- $\text{tr}(\mathbf{A}) = \text{tr}(\mathbf{A}^T)$
- $\text{tr}(\mathbf{A} + \mathbf{B}) = \text{tr}(\mathbf{A}) + \text{tr}(\mathbf{B})$



- We rewrite the Frobenius norm using the trace of the matrix
$$\|P - RQ\|_F^2 = \text{tr}(P^T P) + \text{tr}(Q^T Q) - \text{tr}(P^T RQ) - \text{tr}(Q^T R^T P)$$
- And observe that only the two last terms depend on the unknown R yielding a maximization problem.
- Even without using the properties of the trace we can see that both last terms are equal to

$$\sum_i^N R(q_i - \bar{q})(p_i - \bar{p})^T = \text{tr}(RQP^T)$$

- The 3D-3D pose problem reduced to

$$\max_R \text{tr}(RQP^T)$$

- If the SVD of \mathbf{QP}^T is \mathbf{USV}^T and let $\mathbf{Z} = \mathbf{V}^T \mathbf{R} \mathbf{U}$

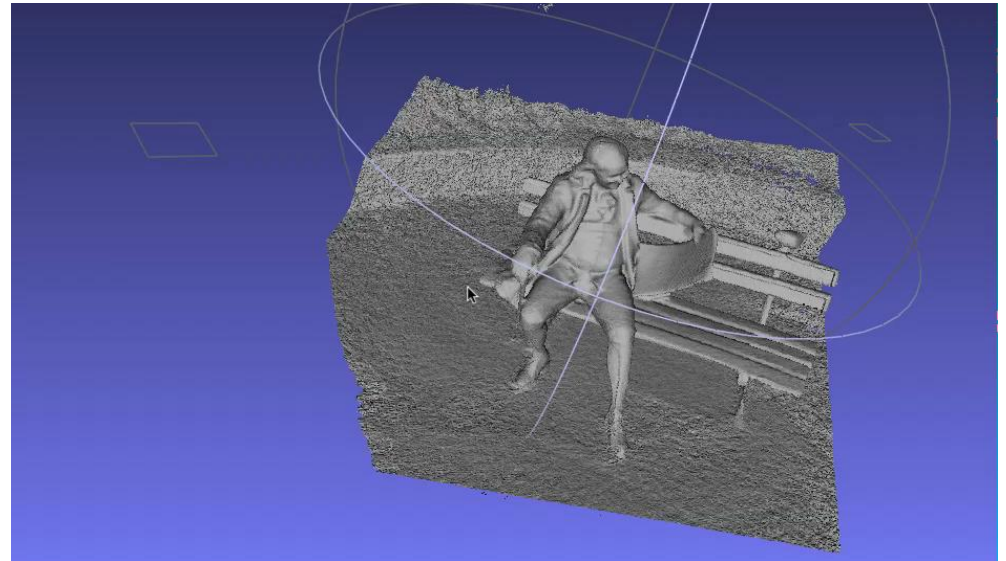
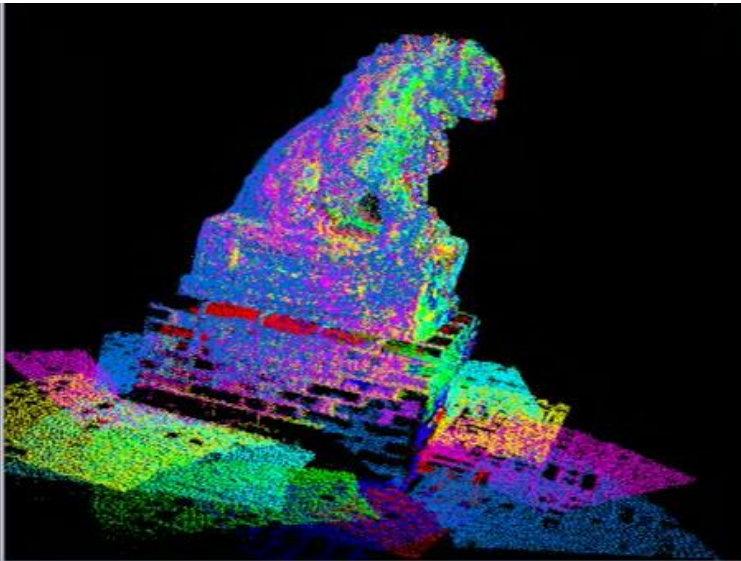
$$\text{tr}(\mathbf{RQP}^T) = \text{tr}(\mathbf{RUSV}^T) = \text{tr}(\mathbf{ZS}) = \sum_1^3 z_{ii} \sigma_i \leq \sum_1^3 \sigma_i$$

- The upper bound is obtained by setting

$$\mathbf{R} = \mathbf{V} \mathbf{U}^T$$

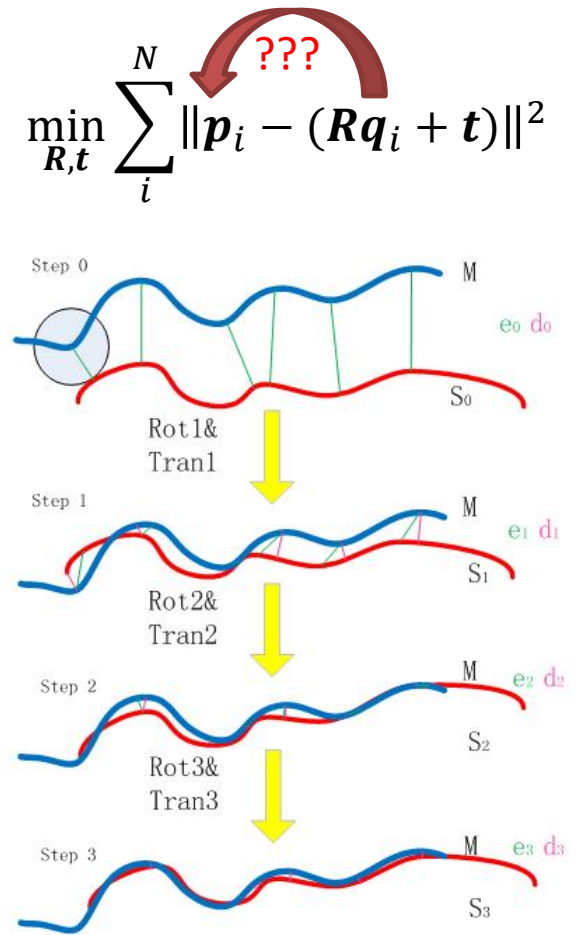
3D-3D Pose Estimation

- 3D-3D Registration enables the creation of 3D models from multiple point clouds:



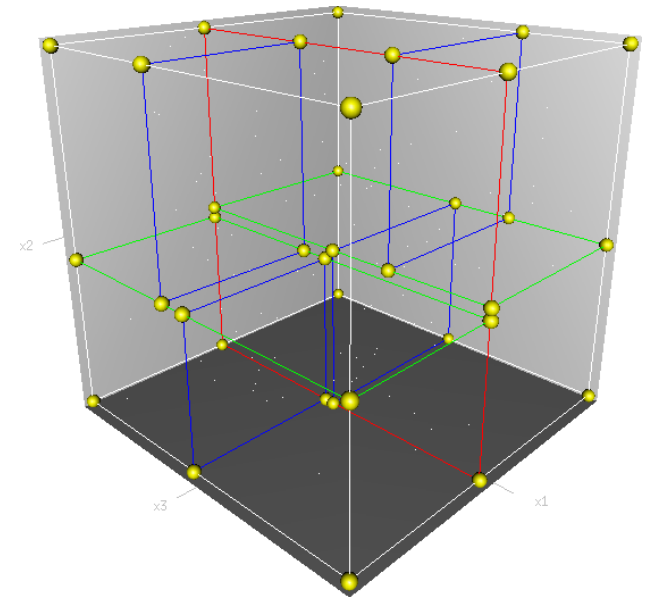
3D-3D Pose Estimation

- How to obtain 3D-3D data association?
 - “Soft” data association directly from point clouds
 - The Iterative Closest Point (ICP) algorithm
 1. Start with some initial guess of rotation and translation
 2. For each point in pointcloud1, find its **nearest neighbor** in pointcloud2 based on the current estimated rotation and translation
 3. Refine the rotation and translation based on the latest data association
 4. Iterate from step 2 until converge



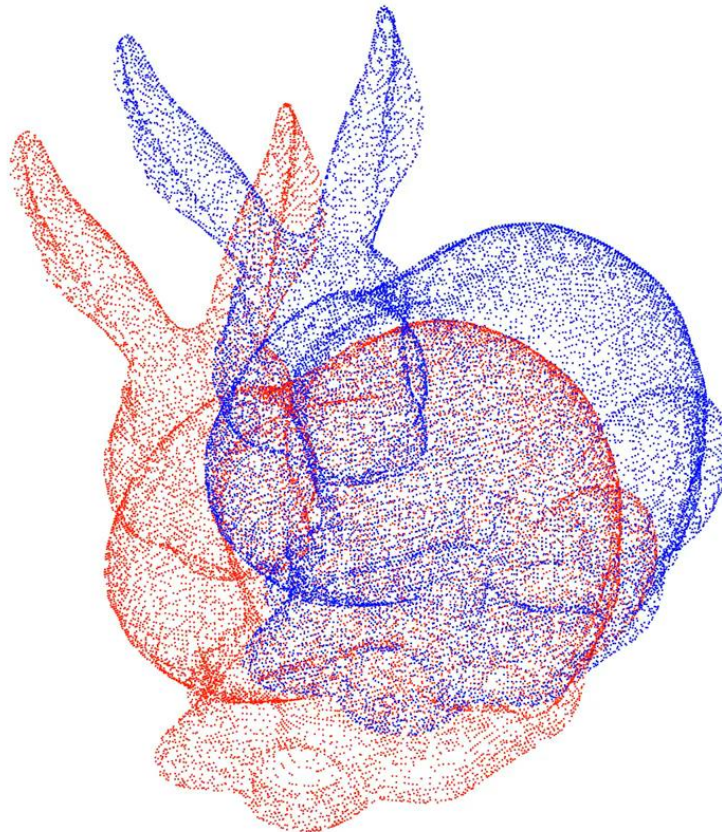
3D-3D Pose Estimation

- How to obtain 3D-3D data association?
 - “Soft” data association directly from point clouds
 - The Iterative Closest Point (ICP) algorithm
 - Need to speed up the search of nearest neighbors
 - Naive implementation: $O(N)$
 - K-d Tree: $O(\log N)$



Iterative Closest Point Algorithm

Iteration 0



3D Laser-Based Mapping

Online Quadrotor Trajectory Generation and Autonomous Navigation on Point Clouds

Fei Gao and Shaojie Shen

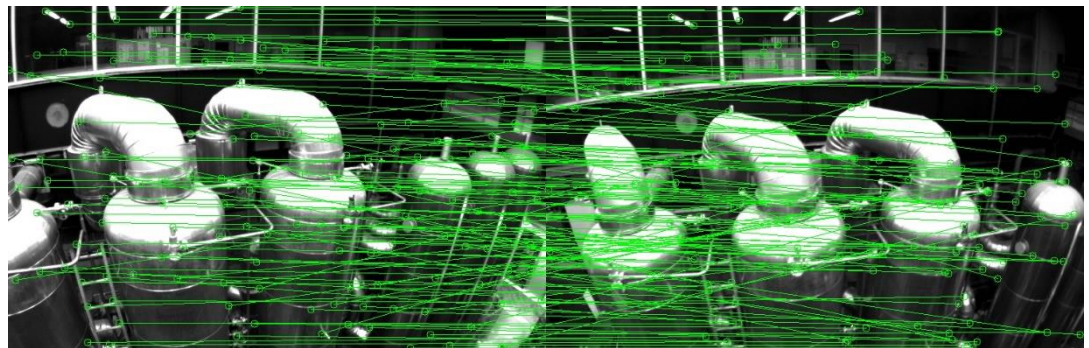
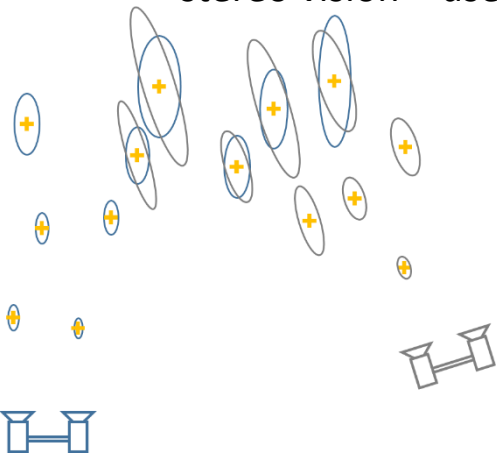


香港科技大學
THE HONG KONG
UNIVERSITY OF SCIENCE
AND TECHNOLOGY

**High resolution video available at
<http://www.ece.ust.hk/~eeshaojie/ssrr2016fei.mp4>**

3D-3D Pose Estimation

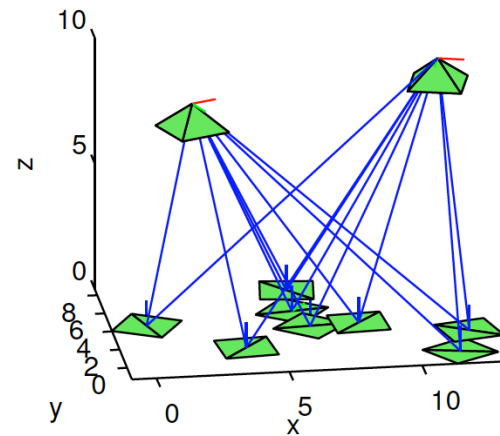
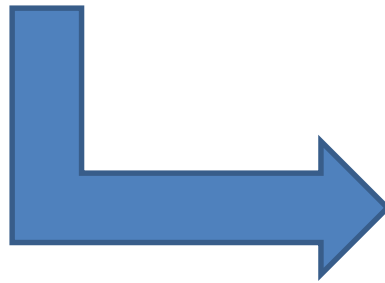
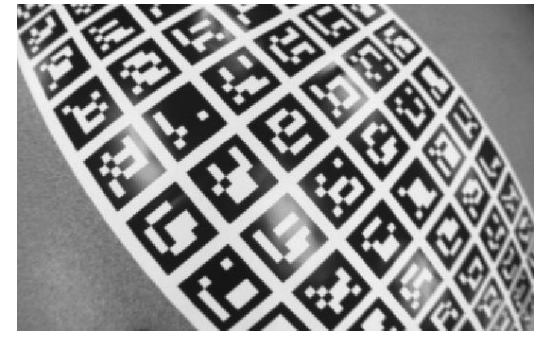
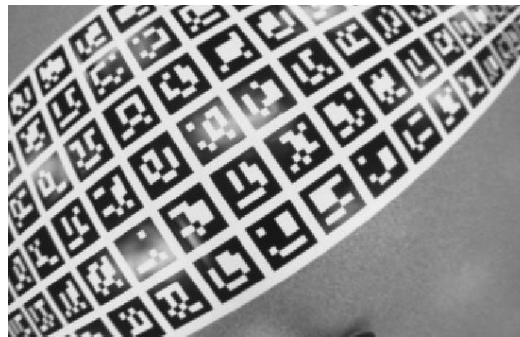
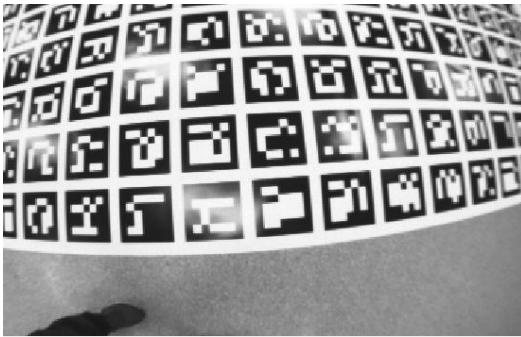
- How to obtain 3D-3D data association?
 - Feature matching using stereo images
 - Calibrated stereo image pairs as input – to be covered in the next lecture
 - Spatial matching – feature matching between stereo image pairs, for computation of 3D points
 - Temporal matching – feature matching between images captures at different times, for motion estimation
 - Need to address outlier removal – to be discussed soon
 - Usually poor performance due to increased ranging error at longer distance with stereo vision – use **3D-2D pose estimation** instead

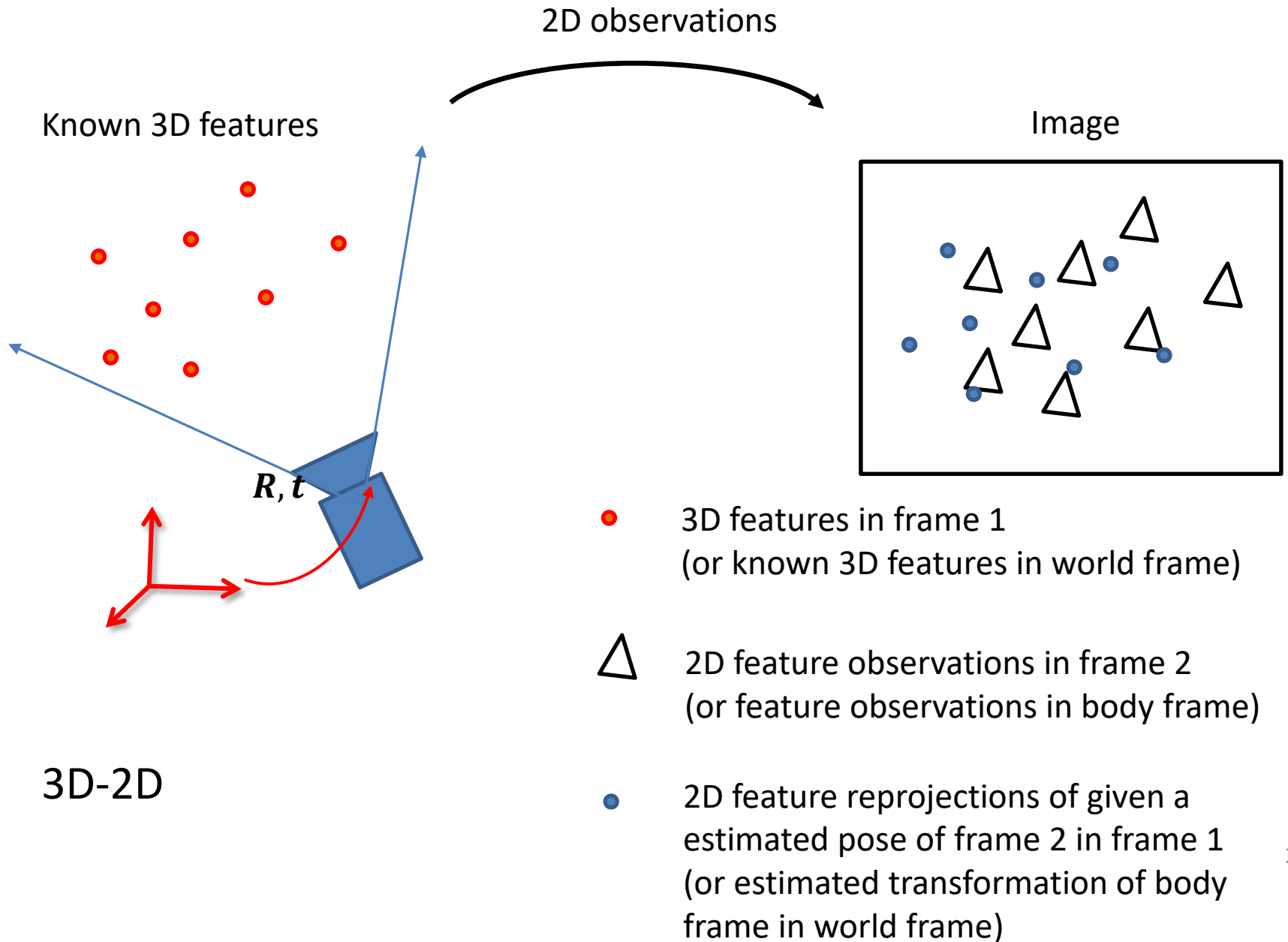


3D-2D Pose Estimation (Pose from Projective Transform) (The PnP Problem)

- Project2 phase1

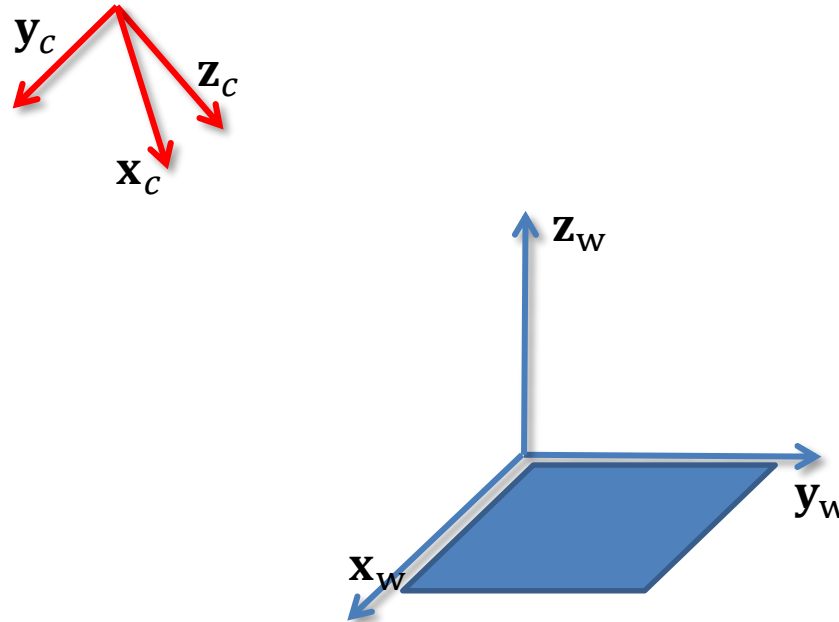
- Using the projective transformation to get the pose of a robot with respect to a planar pattern:





Linear 3D-2D Pose Estimation on Planar Scene

- Pose from reference points on plane $z_w = 0$



- Recall the projection from world to camera

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{K}(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{r}_3 \quad \mathbf{t}) \begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$$

- Assume all points in the world lie in the ground plane $z = 0$.
- Then the transformation reads

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \mathbf{K}(\mathbf{r}_1 \quad \mathbf{r}_2 \quad \mathbf{t}) \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

- Suppose $H = K(r_1 \ r_2 \ t)$

$$\lambda \begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & h_{33} \end{bmatrix} \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$\begin{aligned} \lambda u &= h_{11}x + h_{12}y + h_{13} \\ \lambda v &= h_{21}x + h_{22}y + h_{23} \\ \lambda &= h_{31}x + h_{32}y + h_{33} \end{aligned} \quad \longrightarrow \quad \begin{aligned} h_{31}xu + h_{32}yu + h_{33}u &= h_{11}x + h_{12}y + h_{13} \\ h_{31}xv + h_{32}yv + h_{33}v &= h_{21}x + h_{22}y + h_{23} \end{aligned}$$

Given a observation $(u, v) = (u_i, v_i)$ of
feature (x_i, y_i)

$$\begin{bmatrix} x_i & y_i & 1 & 0 & 0 & 0 & -x_i u_i & -y_i u_i & -u_i \\ 0 & 0 & 0 & x & y & 1 & -x_i v_i & -y_i v_i & -v_i \end{bmatrix} \begin{bmatrix} h_{11} \\ h_{12} \\ h_{13} \\ h_{21} \\ h_{22} \\ h_{23} \\ h_{31} \\ h_{32} \\ h_{33} \end{bmatrix} = 0$$

Solve this $Ax = 0$
problem using SVD

Stack pairs of observations

- Suppose we estimate an $\hat{\mathbf{H}}$ from $N \geq 4$ correspondences.
- Let us assume that we know the intrinsic parameters \mathbf{K} .
- Pose estimation means finding $\mathbf{R} \ \mathbf{t}$ given \mathbf{H} and intrinsic \mathbf{K} .
- We observe that $\mathbf{K}^{-1}\mathbf{H} = (\mathbf{r}_1 \ \mathbf{r}_2 \ \mathbf{t})$ has specific properties: its first two columns are orthogonal unit vectors.
- Nothing guarantees that the $\hat{\mathbf{H}}$ we computed will satisfy this condition.

- Let us name the columns of $K^{-1}\hat{H}$ be $[\tilde{h}_1 \quad \tilde{h}_2 \quad \tilde{h}_3]$
- since $K^{-1}H = (r_1 \quad r_2 \quad t)$
- We seek orthogonal r_1 and r_2 that are the closest to \tilde{h}_1 and \tilde{h}_2 . The solution to this position is given by the Singular Value Decomposition.
- We find the orthogonal matrix R that is the closest to $(\tilde{h}_1 \quad \tilde{h}_2 \quad \tilde{h}_1 \times \tilde{h}_2)$:

$$\arg \min_{R \in SO(3)} \|R - (\tilde{h}_1 \quad \tilde{h}_2 \quad \tilde{h}_1 \times \tilde{h}_2)\|_F^2$$

$$\arg \min_{R \in SO(3)} \|R - (\tilde{\mathbf{h}}_1 \quad \tilde{\mathbf{h}}_2 \quad \tilde{\mathbf{h}}_1 \times \tilde{\mathbf{h}}_2)\|_F^2$$

- If the SVD of $(\tilde{\mathbf{h}}_1 \quad \tilde{\mathbf{h}}_2 \quad \tilde{\mathbf{h}}_1 \times \tilde{\mathbf{h}}_2) = \mathbf{U}\mathbf{S}\mathbf{V}^T$, then the solution is

$$\mathbf{R} = \mathbf{U}\mathbf{V}^T$$

- To find the translation:

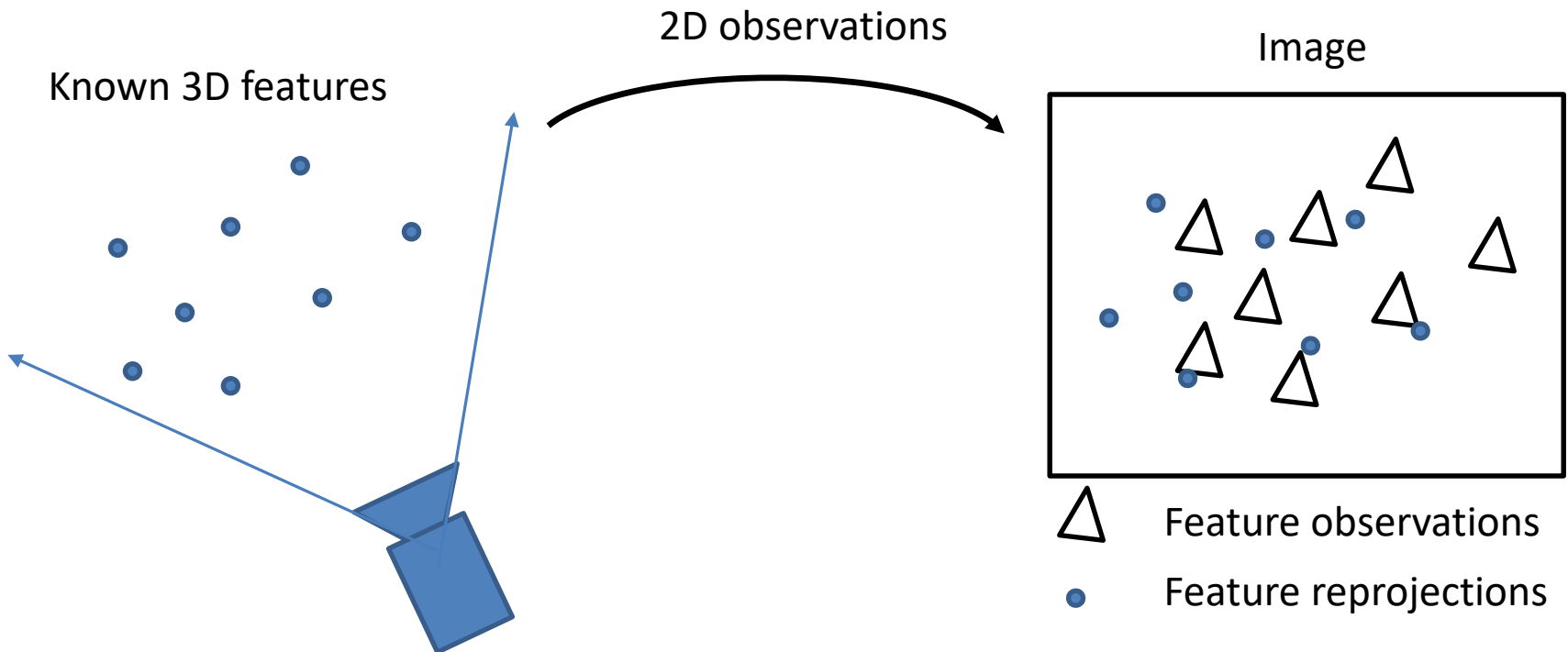
$$\mathbf{t} = \tilde{\mathbf{h}}_3 / \|\tilde{\mathbf{h}}_1\|$$

Nonlinear 3D-2D Pose Estimation

θ : Euler Angles $\in R^3$ t : Translation $\in R^3$ $\pi(\cdot)$: projection function

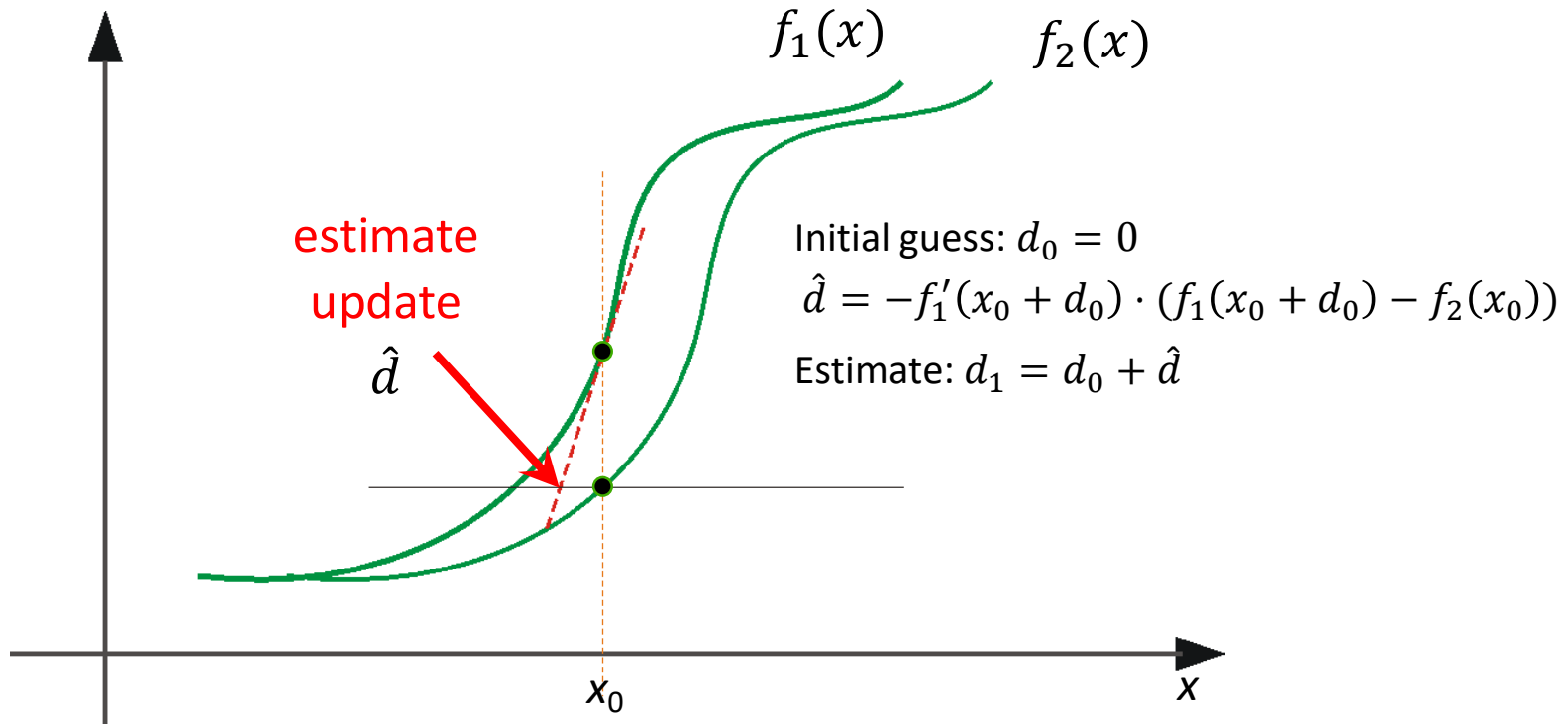
- Minimize the reprojection error w.r.t. camera pose
 - No need to assume planar scene

$$\min_{\theta, t} \sum_i \left\| \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \pi \left(K \cdot (R(\theta) \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + t) \right) \right\|^2$$



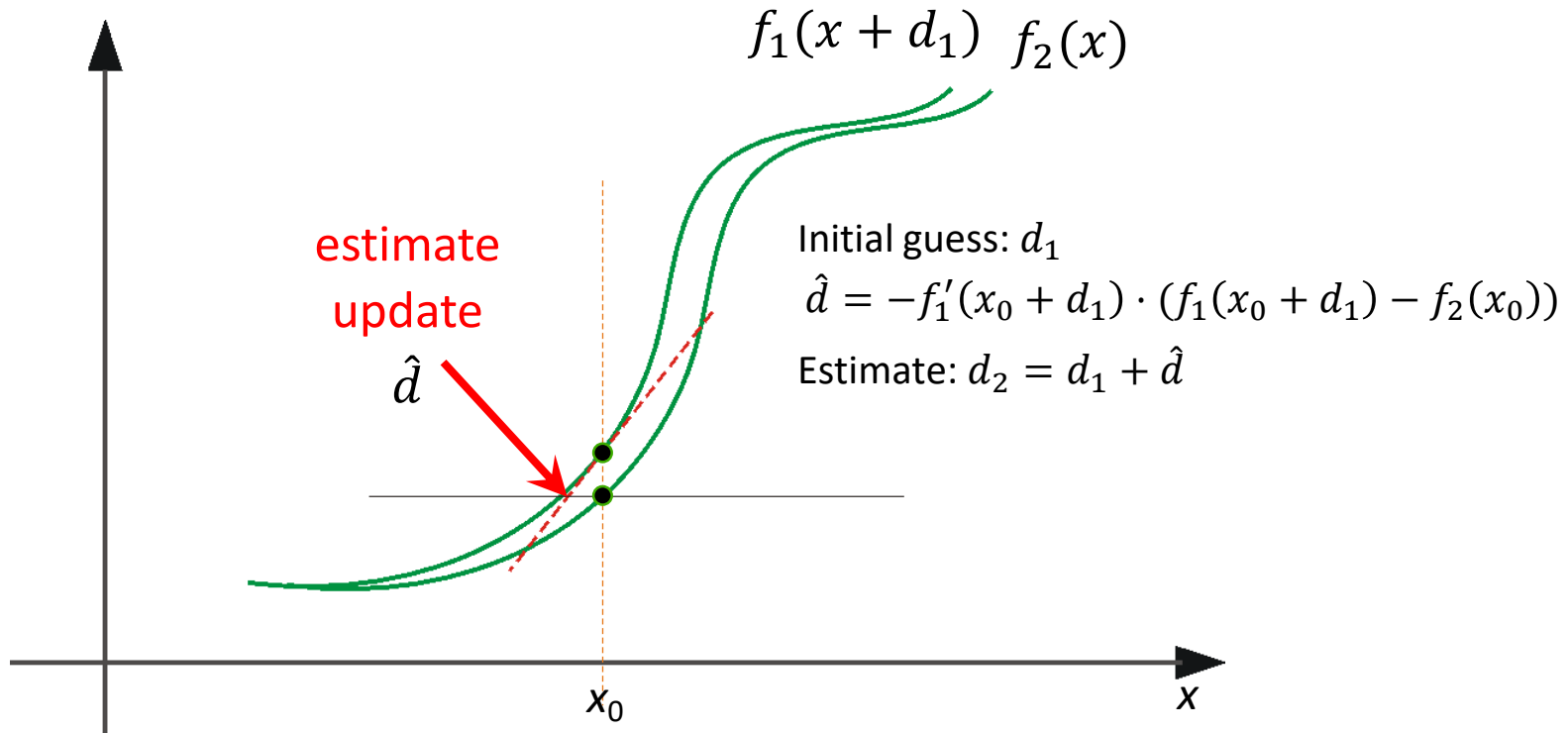
Gauss Newton Method

Compute d to minimize $\|f_1(x + d) - f_2(x)\|^2$



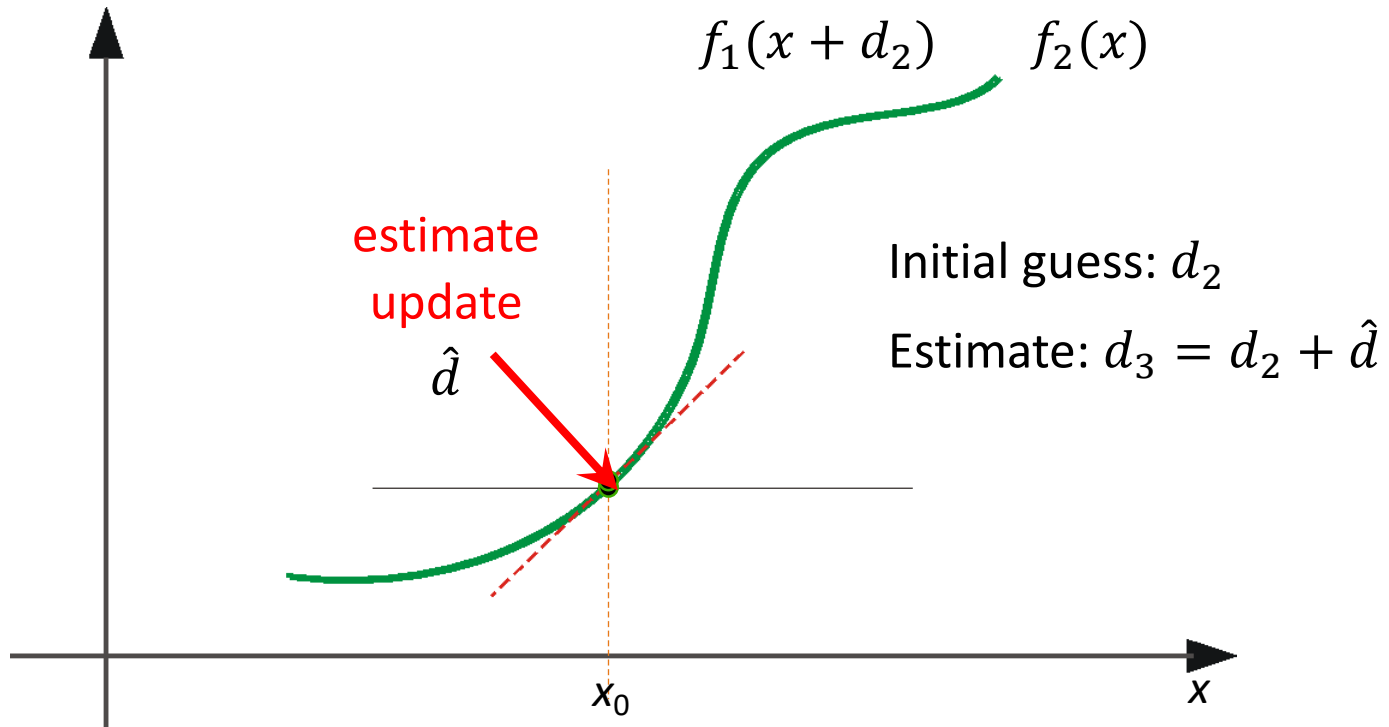
Gauss Newton Method

Compute d to minimize $\|f_1(x + d) - f_2(x)\|^2$



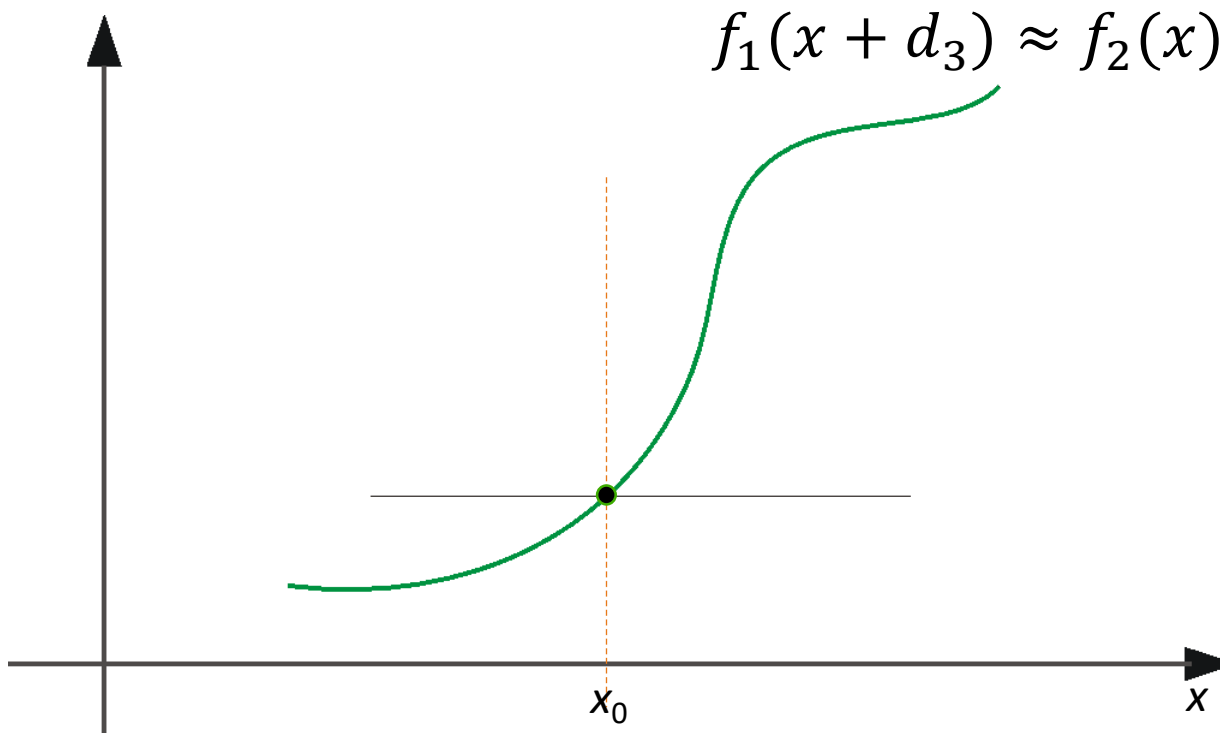
Gauss Newton Method

Compute d to minimize $\|f_1(x + d) - f_2(x)\|^2$



Gauss Newton Method

Compute d to minimize $\|f_1(x + d) - f_2(x)\|^2$



Nonlinear 3D-2D Pose Estimation

θ : Euler Angles $\in R^3$ t : Translation $\in R^3$ $\pi(\cdot)$: projection function

1) Nonlinear least square

$$\min_{\theta, t} \sum_i \left\| \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \pi \left(K \cdot (R(\theta) \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + t) \right) \right\|^2 = \min_{\theta, t} \sum_i \|\gamma_i(\theta, t)\|^2$$

2) Linearization **Initial values**

$$\gamma_i(\theta, t) \approx \gamma_i(\underbrace{\theta_0, t_0}_{\text{Initial values}}) + \underbrace{\frac{\partial \gamma_i}{\partial \theta, t} \big|_{\theta_0, t_0}}_{2 \times 6 \text{ matrix}} \cdot \begin{bmatrix} \delta \theta \\ \delta t \end{bmatrix} = \gamma_i(\theta_0, t_0) + \underbrace{J_i}_{2 \times 6 \text{ matrix}} \begin{bmatrix} \delta \theta \\ \delta t \end{bmatrix}$$

The problem becomes:

$$\min_{\delta \theta, \delta t} \sum_i \left\| \gamma_i(\theta_0, t_0) + J_i \begin{bmatrix} \delta \theta \\ \delta t \end{bmatrix} \right\|^2$$

Nonlinear 3D-2D Pose Estimation

θ : Euler Angles $\in R^3$ t : Translation $\in R^3$ $\pi(\cdot)$: projection function

3) Take derivative and set it to zero

$$\sum_i J_i^T J_i \begin{bmatrix} \delta\theta \\ \delta t \end{bmatrix} + \sum_i J_i^T \gamma_i(\theta_0, t_0) = 0 \Rightarrow \overbrace{\sum_i J_i^T J_i}^{\mathbf{A}} \begin{bmatrix} \delta\theta \\ \delta t \end{bmatrix} = - \overbrace{\sum_i J_i^T \gamma_i(\theta_0, t_0)}^{\mathbf{b}}$$


4) Solve for the incremental states and update the optimization variables

$$\begin{bmatrix} \delta\theta \\ \delta t \end{bmatrix} = \mathbf{A}^{-1} \mathbf{b} \Rightarrow \begin{bmatrix} \theta \\ t \end{bmatrix} = \begin{bmatrix} \theta_0 \\ t_0 \end{bmatrix} + \begin{bmatrix} \delta\theta \\ \delta t \end{bmatrix}$$

5) Iterate from step (1) with updated states as initial values

Nonlinear State Estimation

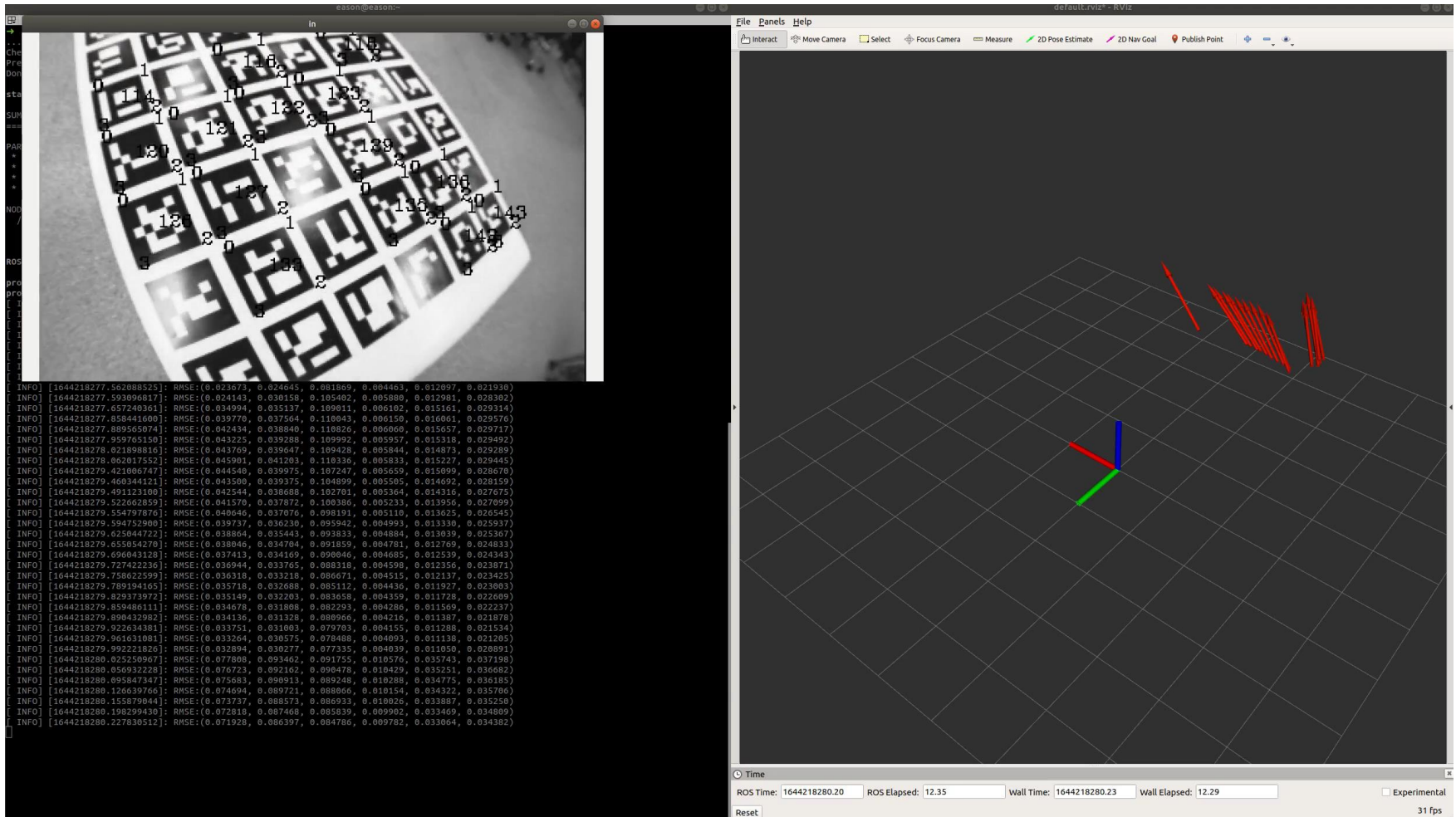
- The nonlinear Gauss Newton method also works for 3D-3D pose estimation.
- In fact, it works for all nonlinear optimization problems, if provided with sufficiently **good initial values**.
- Simultaneous localization and mapping (SLAM) problems (with only cameras or with heterogeneous sensors) are solved in essentially the same way.
- Anything missing?



$$\min_{\theta, t} \sum_i \left\| \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \pi \left(\mathbf{K} \cdot (\mathbf{R}(\theta) \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + \mathbf{t}) \right) \right\|^2$$

???

3D-2D Vision-Based Localization with Markers



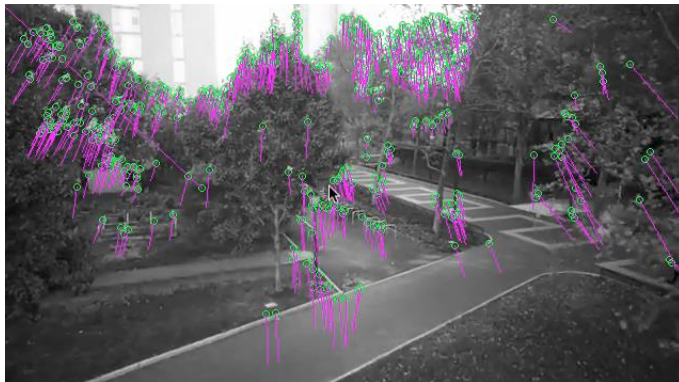
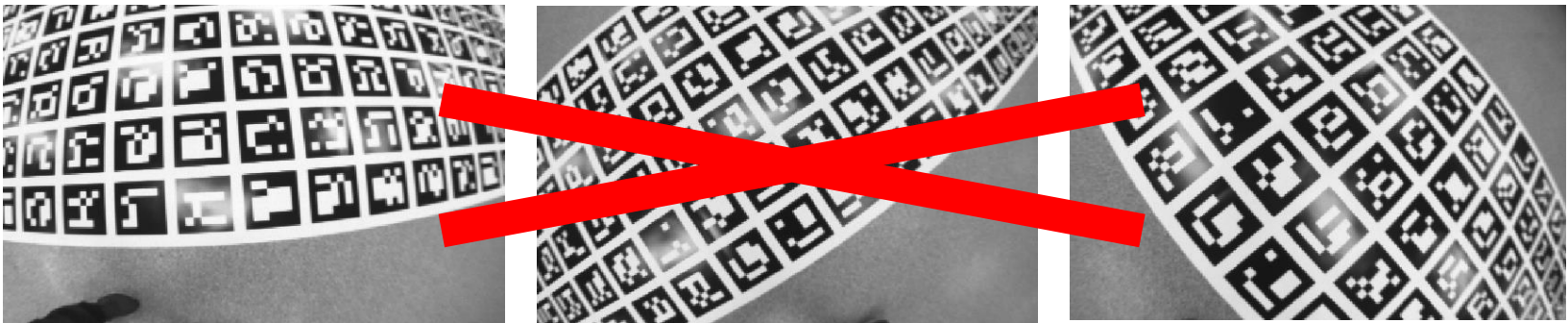
Outlier Rejection and Robust Estimation

3D-2D Pose Estimation in Unstructured Environments

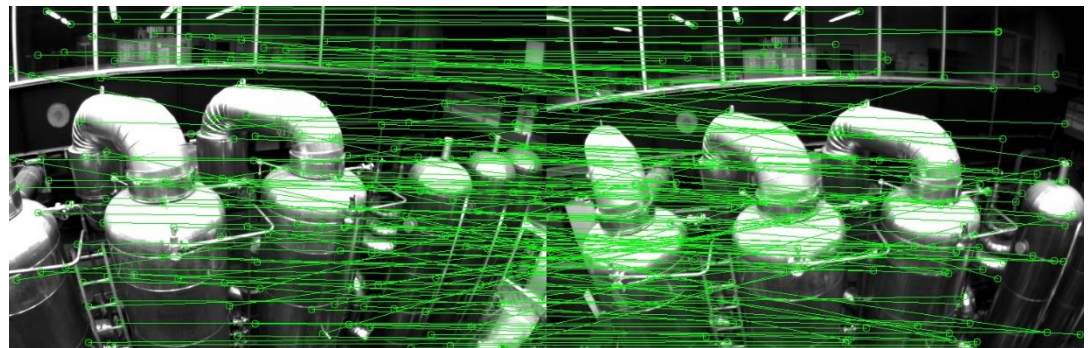
- Why do you need markers?
 - For data association
 - What if you do not have them?

$$\min_{\theta, t} \sum_i \left\| \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \pi \left(K \cdot (R(\theta) \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + t) \right) \right\|^2$$

???



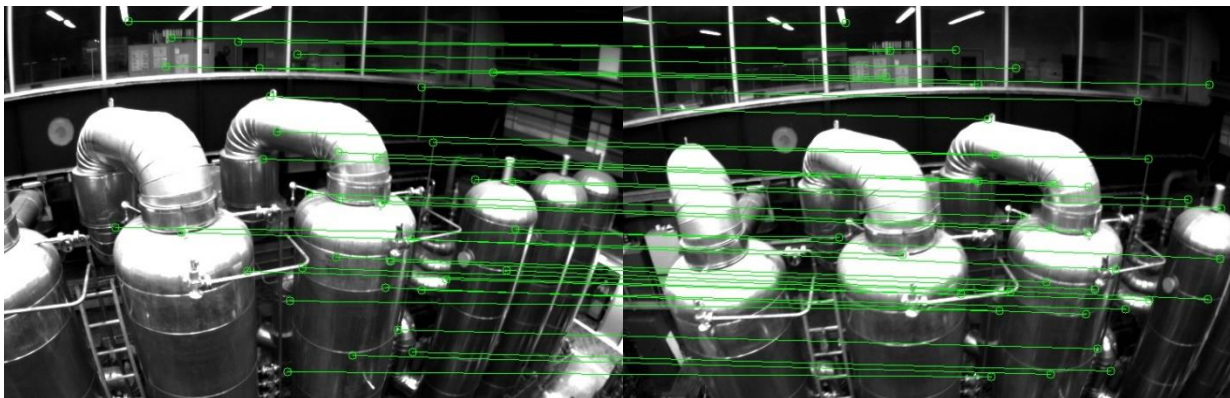
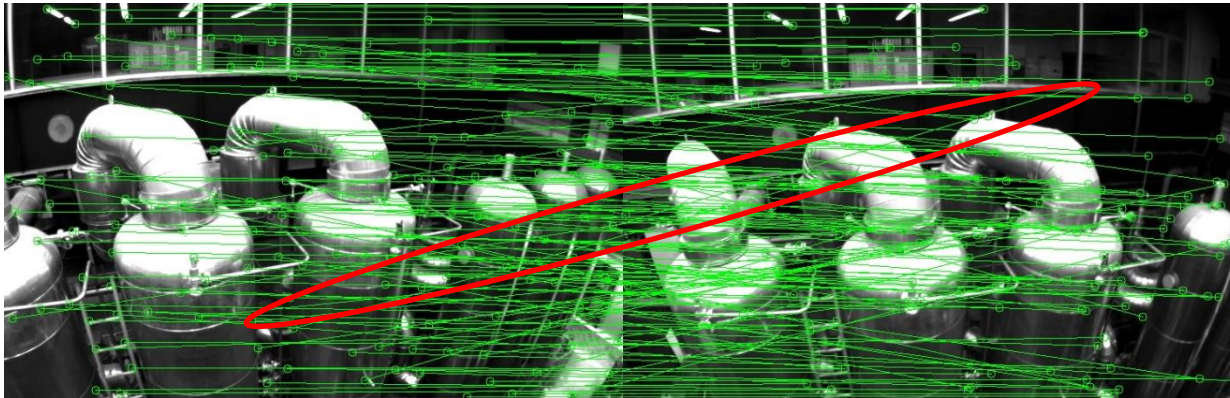
Optical flow (next lecture)



Feature matching

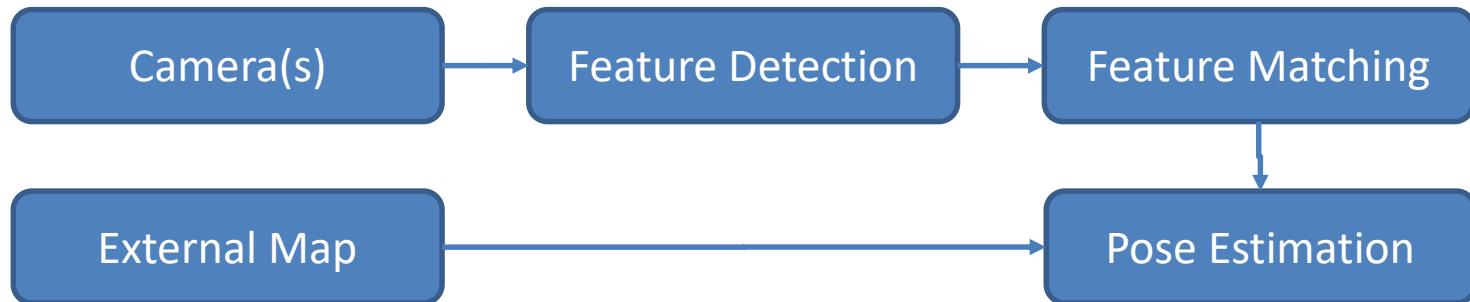
3D-2D Pose Estimation in Unstructured Environments

- What if you do not have markers?
 - Outlier rejection



3D-2D Pose Estimation in Unstructured Environments

- What if you do not have markers?
 - Map representation – features stored in a database
 - Data association – descriptor matching
 - Outlier rejection – RANSAC

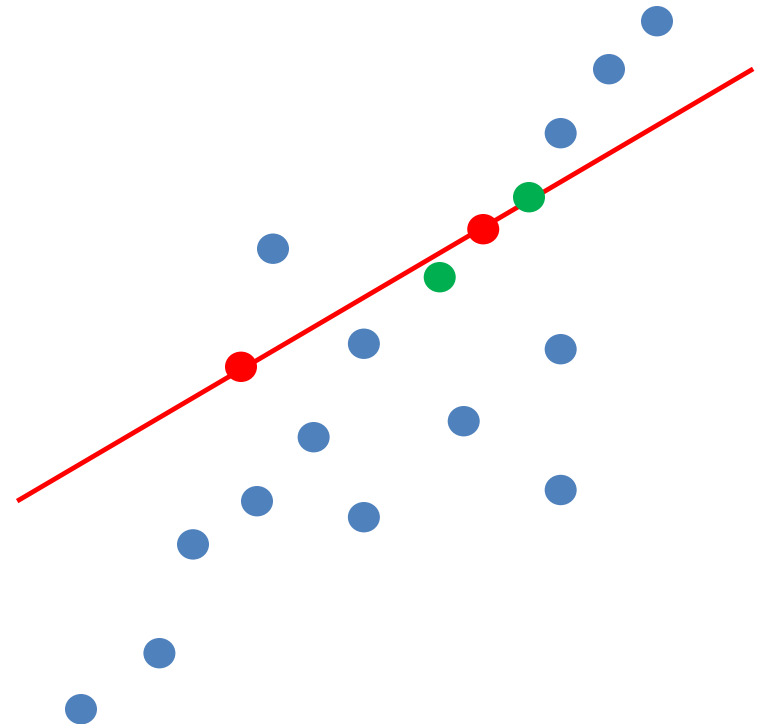


RANdom SAmple Consensus (RANSAC)

- Model fitting with outlier rejection
 - The 6-DOF pose you are trying to estimation is a model
- Algorithm:
 - Loop:
 - Randomly select a small amount of (or minimum) data points to find a model
 - See the error between the model and all other data points
 - Find the data points with error smaller than a threshold as inliers
 - If the current model has more inliers than all previous ones, record all inliers
 - Repeat
 - Use all inliers to find the best estimate of the model

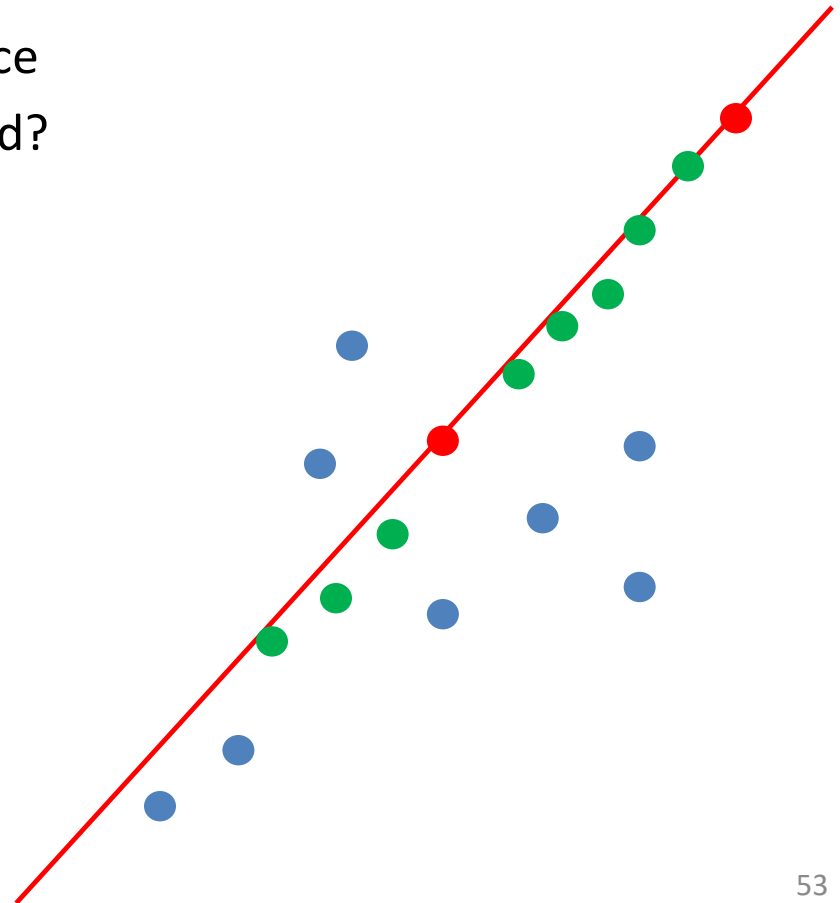
RANdom SAmple Consensus (RANSAC)

- RANSAC for 2D line fitting
 - Minimum number of points to define a 2D line: 2
 - Error metric: point to line distance
 - How many iterations are required?
- Iteration 1: 4 Inliers



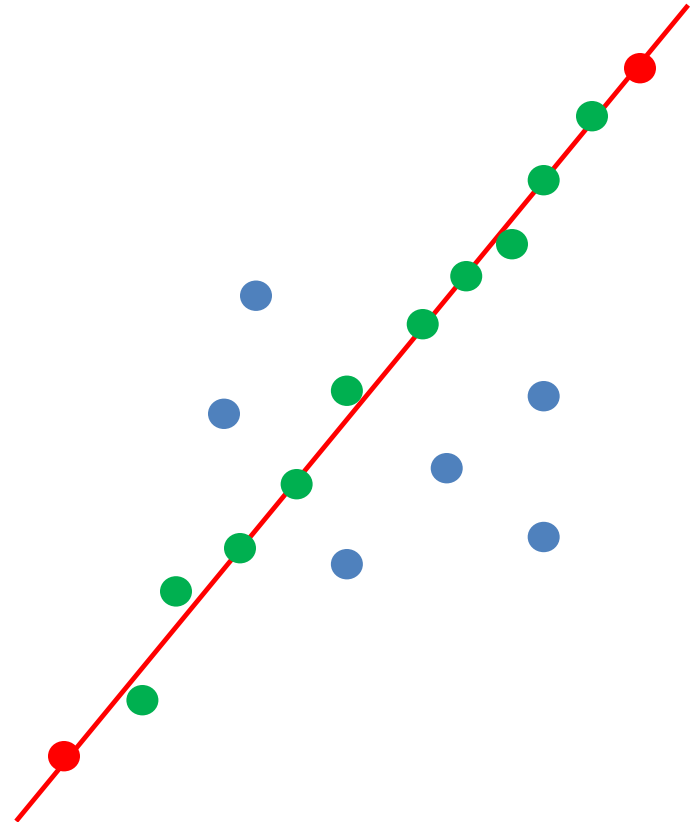
RANdom SAmple Consensus (RANSAC)

- RANSAC for 2D line fitting
 - Minimum number of points to define a 2D line: 2
 - Error metric: point to line distance
 - How many iterations are required?
- Iteration 2: 10 Inliers



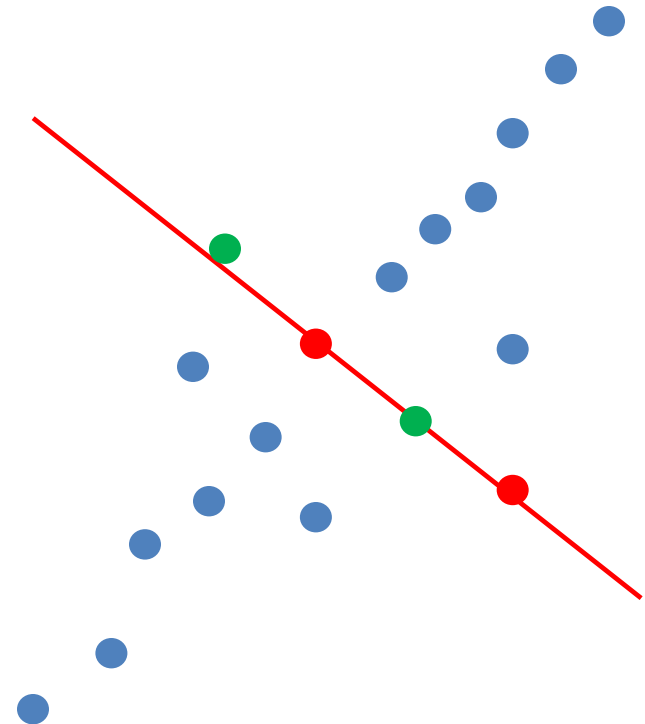
RANdom SAmple Consensus (RANSAC)

- RANSAC for 2D line fitting
 - Minimum number of points to define a 2D line: 2
 - Error metric: point to line distance
 - How many iterations are required?
- Iteration 3: 12 Inliers



RANdom SAmple Consensus (RANSAC)

- RANSAC for 2D line fitting
 - Minimum number of points to define a 2D line: 2
 - Error metric: point to line distance
 - How many iterations are required?
- Iteration 4: 4 Inliers



Pose Estimation in Unstructured Environments

- How many feature correspondences are required to create a model?
 - 3D-3D: 3
 - 3D-2D: 3
 - It is OK to use more points to find the model (our 3D-2D requires 4)
 - But few number of points is better (Why?)
- How to find the model from feature correspondences?
- How to define the error metric?
 - 3D-3D: point distance
 - 3D-2D: reprojection error
- How many iterations are required?

RANdom SAmple Consensus (RANSAC)

- How many iterations are required - the probability
 - Probability of outlier: X ($X < 1$)
 - M number of data points to create a model
 - N iterations
- RANSAC failure: all random sample contains at least 1 outlier
 - Failure probability = $(1 - (1 - X)^M)^N$
 - 2D line fitting example
 - 30% outliers
 - 2 data points to create a model
 - Failure probability for 5 iterations: 3.45%
 - Failure probability for 10 iterations: 0.12%

RANdom SAmple Consensus (RANSAC)

- How many iterations are required - the probability
 - Probability of outlier: X ($X < 1$)
 - M number of data points to create a model
 - N iterations
- RANSAC failure: all random sample contains at least 1 outlier
 - Failure probability = $(1 - (1 - X)^M)^N$
 - 3D-3D pose estimation
 - 30% outliers
 - 3 data points to create a model
 - Failure probability for 5 iterations: 12.24%
 - Failure probability for 10 iterations: 1.49%
 - Failure probability for 20 iterations: 0.02%

RANdom SAmple Consensus (RANSAC)

- How many iterations are required - the probability
 - Probability of outlier: X ($X < 1$)
 - M number of data points to create a model
 - N iterations
- RANSAC failure: all random sample contains at least 1 outlier
 - Failure probability = $(1 - (1 - X)^M)^N$
 - 3D-3D pose estimation
 - 30% outliers
 - 20 data points to create a model (bad example)
 - Failure probability for 5 iterations: 99.6%
 - Failure probability for 10 iterations: 99.2%
 - Failure probability for 20 iterations: 98.4%
 - Failure probability for 1000 iterations: 45%
 - Failure probability for 10000 iterations: 0.03%

Robust M-Estimator

$\boldsymbol{\theta}$: Euler Angles $\in R^3$ \mathbf{t} : Translation $\in R^3$ $\pi(\cdot)$: projection function

- Recall the nonlinear 3D-2D pose estimation problem

$$\min_{\boldsymbol{\theta}, \mathbf{t}} \sum_i \left\| \begin{bmatrix} u_i \\ v_i \end{bmatrix} - \pi \left(\mathbf{K} \cdot (\mathbf{R}(\boldsymbol{\theta}) \cdot \begin{bmatrix} x_i \\ y_i \\ z_i \end{bmatrix} + \mathbf{t}) \right) \right\|^2 = \min_{\boldsymbol{\theta}, \mathbf{t}} \sum_i \|\gamma_i(\boldsymbol{\theta}, \mathbf{t})\|^2$$

- The original least square problem is very sensitive to outliers, due to the **squared** influence of data terms
- Even after RANSAC, there may still be outlier (or relatively bad “inliers”) exists
- We want a method to reduce the impact of outliers

Robust M-Estimator

- Rewrite in a general form for minimizing w.r.t parameter p

$$\min_{\theta, t} \sum_i \|\gamma_i(\theta, t)\|^2 \Rightarrow \min_p \sum_i \rho(\gamma_i(p))$$

- ρ is a symmetric, positive-definite function with unique minimum at 0
- Take the derivative and set to 0

$$\sum_i \frac{\partial \rho}{\partial \gamma_i} \frac{\partial \gamma_i}{\partial p} = \sum_i \psi(\gamma_i) \frac{\partial \gamma_i}{\partial p} = 0$$

- $\psi(x) = \partial \rho / \partial x$ is called the influence function
 - We can also define a weight function $\omega(x) = \psi(x)/x$
- We then have:

$$\sum_i \omega(\gamma_i) \gamma_i \frac{\partial \gamma_i}{\partial p} = 0$$

- This is exactly the system we obtain if we solve the following iterative re-weighted least square problem:

$$\min_p \sum_i \omega(\gamma_i^{k-1}) \|\gamma_i(p)\|^2$$

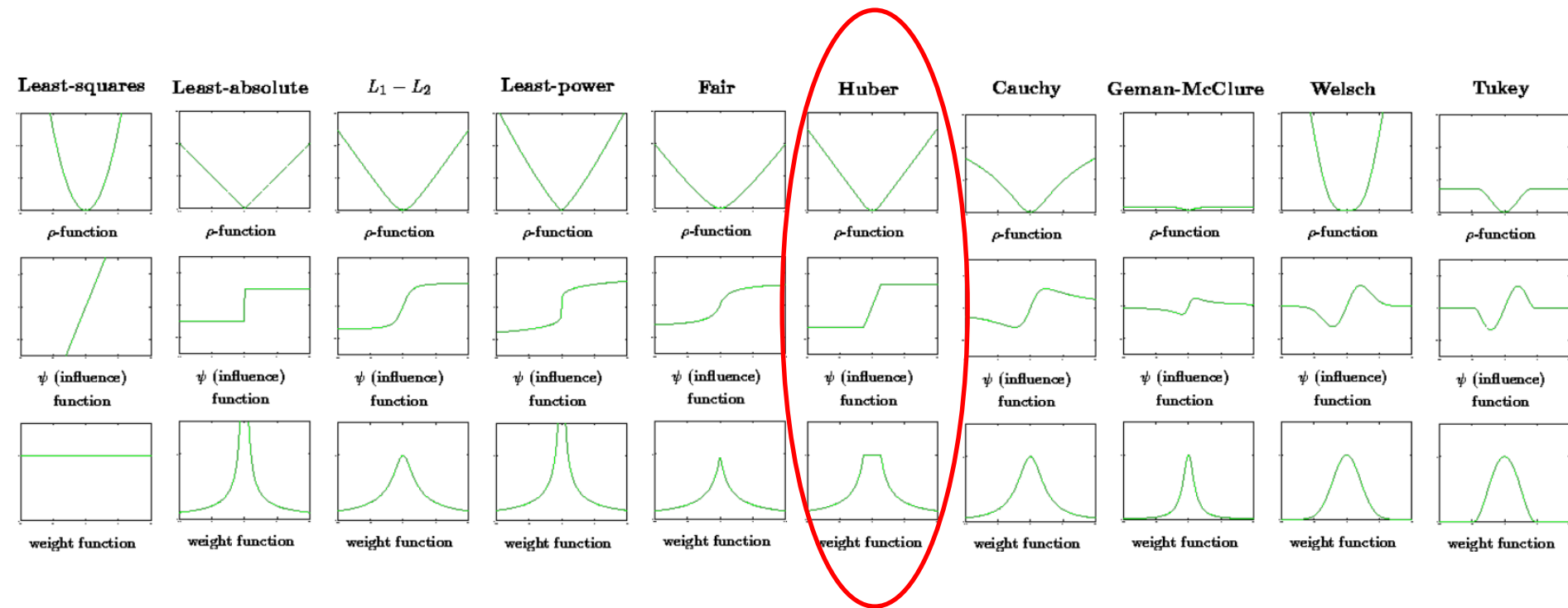
Robust M-Estimator

- The influence function $\psi(\cdot)$ measures the influence of a datum on the value of parameter estimation
- For least square with $\rho(x) = x^2/2$, the influence is $\psi(x) = x$. The influence of a datum increases w.r.t. the size of error
- When an estimator is robust, we want influence of any single datum is insufficient to yield any significant offsets, provided following constraints are met:
 - Bounded influence function
 - Individual $\rho(\cdot)$ function is convex in parameter p

Robust M-Estimator

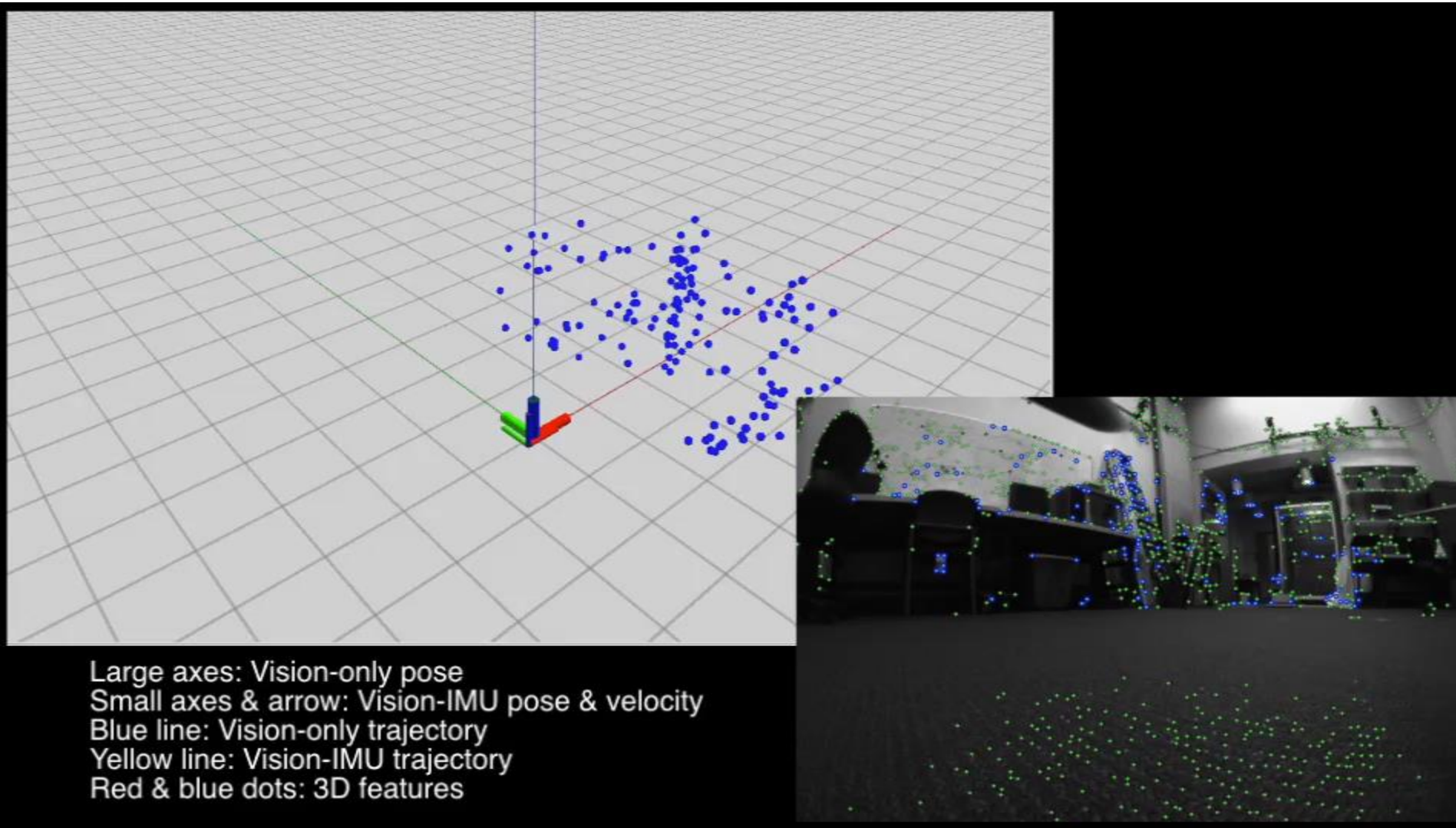
type	$\rho(x)$	$\psi(x)$	$w(x)$
L_2	$x^2/2$	x	1
L_1	$ x $	$\text{sgn}(x)$	$\frac{1}{ x }$
$L_1 - L_2$	$2(\sqrt{1 + x^2/2} - 1)$	$\frac{x}{\sqrt{1 + x^2/2}}$	$\frac{1}{\sqrt{1 + x^2/2}}$
L_p	$\frac{ x ^\nu}{\nu}$	$\text{sgn}(x) x ^{\nu-1}$	$ x ^{\nu-2}$
“Fair”	$c^2[\frac{ x }{c} - \log(1 + \frac{ x }{c})]$	$\frac{x}{1 + x /c}$	$\frac{1}{1 + x /c}$
Huber $\begin{cases} \text{if } x \leq k \\ \text{if } x \geq k \end{cases}$	$\begin{cases} x^2/2 \\ k(x - k/2) \end{cases}$	$\begin{cases} x \\ k \text{sgn}(x) \end{cases}$	$\begin{cases} 1 \\ k/ x \end{cases}$
Cauchy	$\frac{c^2}{2} \log(1 + (x/c)^2)$	$\frac{x}{1 + (x/c)^2}$	$\frac{1}{1 + (x/c)^2}$
Geman-McClure	$\frac{x^2/2}{1 + x^2}$	$\frac{x}{(1 + x^2)^2}$	$\frac{1}{(1 + x^2)^2}$
Welsch	$\frac{c^2}{2} [1 - \exp(-(x/c)^2)]$	$x \exp(-(x/c)^2)$	$\exp(-(x/c)^2)$
Tukey $\begin{cases} \text{if } x \leq c \\ \text{if } x > c \end{cases}$	$\begin{cases} \frac{c^2}{6} (1 - [1 - (x/c)^2]^3) \\ (c^2/6) \end{cases}$	$\begin{cases} x[1 - (x/c)^2]^2 \\ 0 \end{cases}$	$\begin{cases} [1 - (x/c)^2]^2 \\ 0 \end{cases}$

Robust M-Estimator



Most commonly used

3D-2D Vision-Based Localization without Markers



Logistics

- Project 2, phase 1 is released (03/21), due next Friday (03/31)
- Project 1, phase 4 will also due next Friday (03/31)