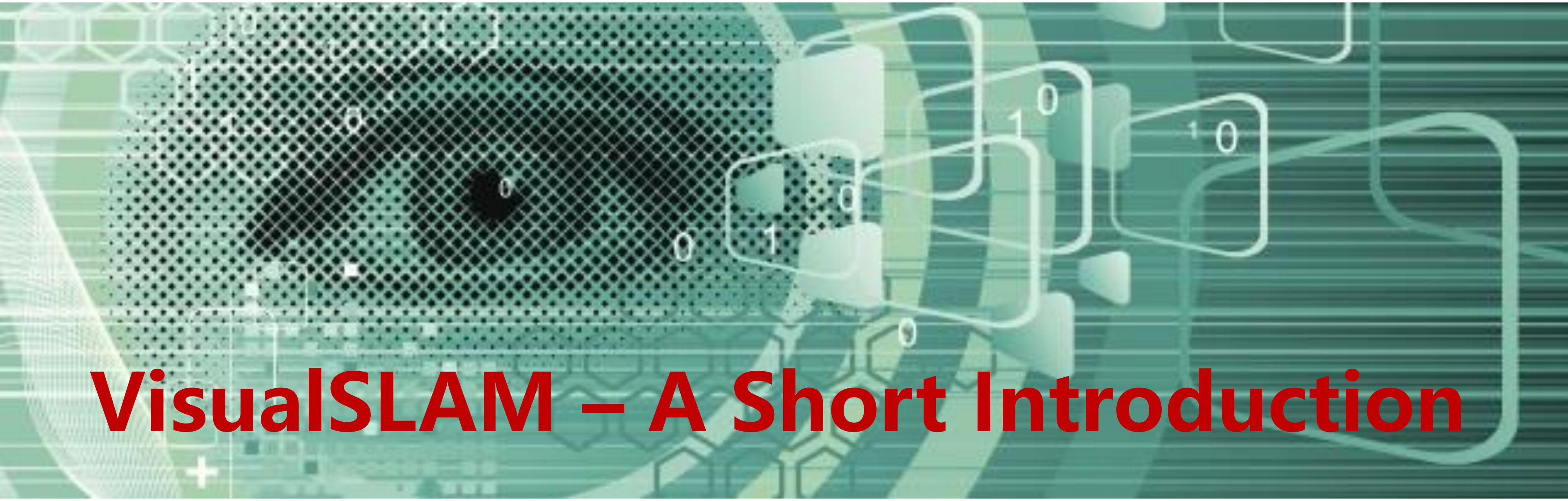


VisualSLAM – A Short Introduction

Danping Zou

dpzou@sjtu.edu.cn

http://drone.sjtu.edu.cn/dpzou

Lab of Navigation and Location-based Service

Shanghai Jiao Tong University

2016, 27th July, ROS Summer School @ECNU

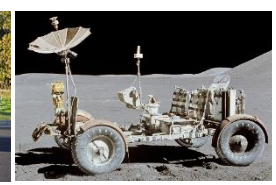
What's SLAM

- **SLAM** :**S**imultaneous **L**ocalization **A**nd **M**apping.

Constructing a map of an unknown environment while simultaneously keeping track of the robot's location.

- Autonomous navigation
- motion planning






SLAM with different kinds of sensors

2D laser rangefinder

3D liDar

RGB-D camera

- High precision
- Bulky

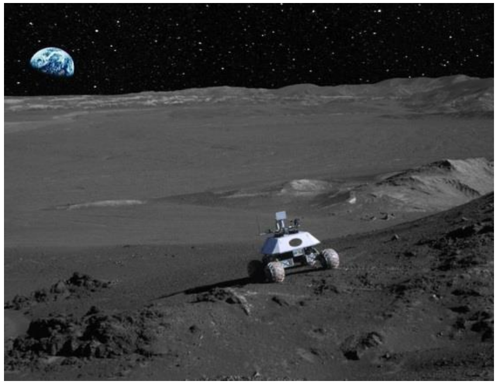
Since 2005, there has been intense research into VSLAM (**Visual SLAM**) using primarily visual (camera) sensors.

- Passive sensing
- Light & compact
- Energy saving
- Ubiquity

Visual SLAM

- Input sensor are video cameras
- build 3D map
- Estimate the self-pose of the camera
- both are processed in **real time**

1. A complementary to other sensors:
    - GPS
    - IMU
    - Laser range finder
2. Works in GPS-denied environments
    - Indoor
    - Cave
    - Mars, Moon

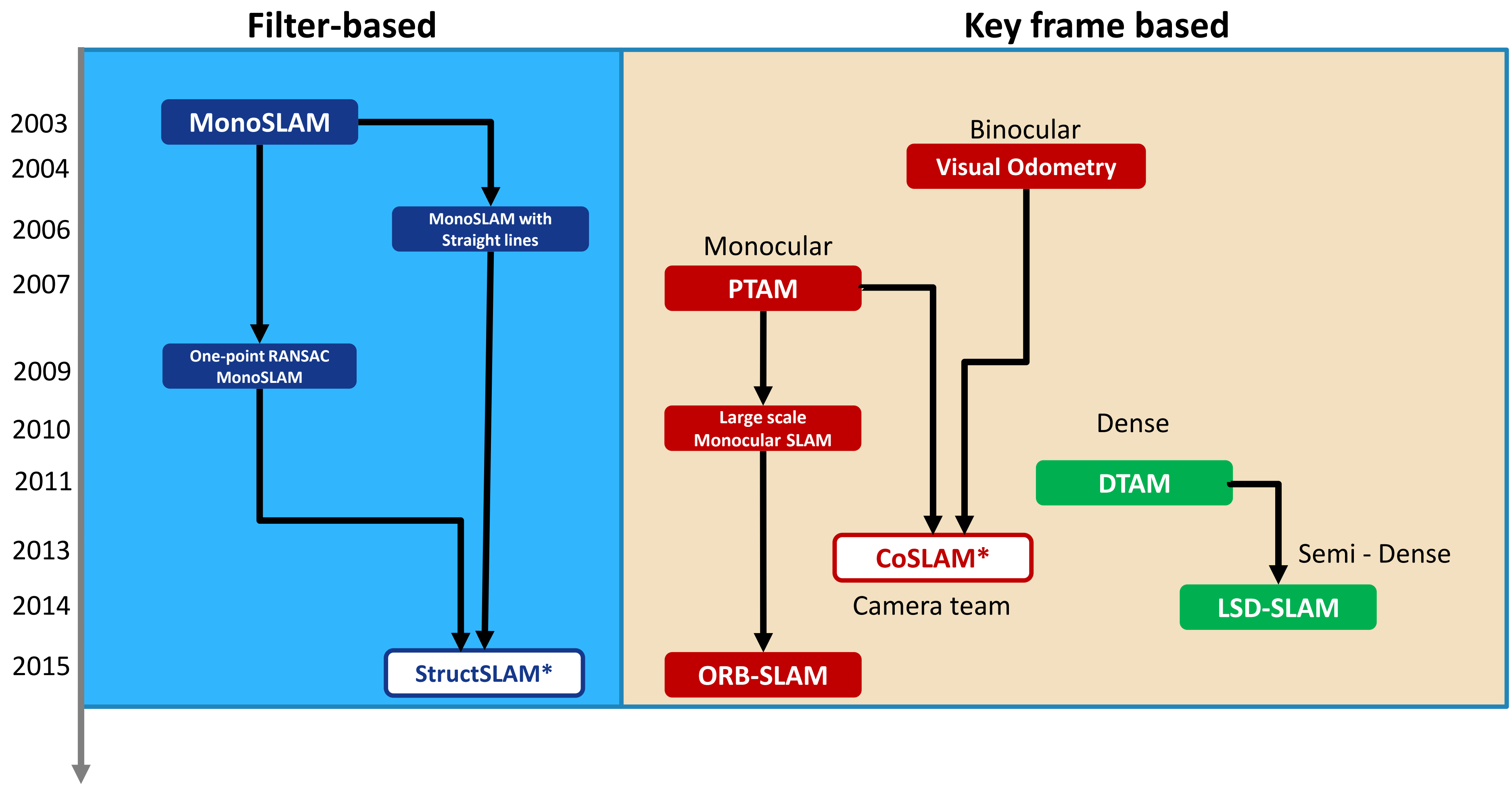


Other applications

- Motion capture in AR/VR

- Get the position and attitude of the camera
- Put the virtual object into the scene

Development of Visual SLAM in recent ten years



Open source software for VisualSLAM

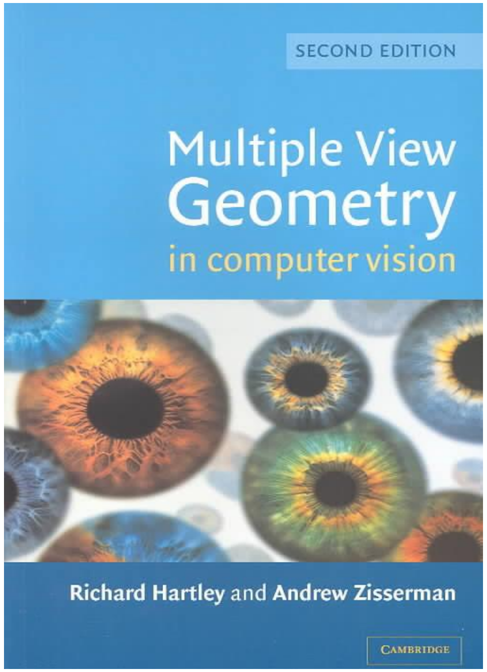
- PTAM(http://www.robots.ox.ac.uk/~gk/PTAM/)
- ORBSLAM(http://webdiis.unizar.es/~raulmur/orbslam/)
- MonoSLAM(http://webdiis.unizar.es/~jcivera/code/1p-ransac-ekf-monoslam.html)
- VisualSFM (http://ccwu.me/vsfm/)
- CoSLAM (https://github.com/danping/CoSLAM)

Outline

- Basic Theory
  - Pinhole camera model
  - Camera calibration
  - Two camera geometry

- Build a visual SLAM system
  - Structure-from-motion(SFM) approach:
    - PTAM
    - ORB SLAM
    - CoLSAM
  - Extended Kalman Filter approach:
    - MonoSLAM
    - StructSLAM

Multiple view geometry



Homogenous coordinates

- Homogenous representation
  - Represent an n-dimensional vector by a $n+1$ dimensional coordinate

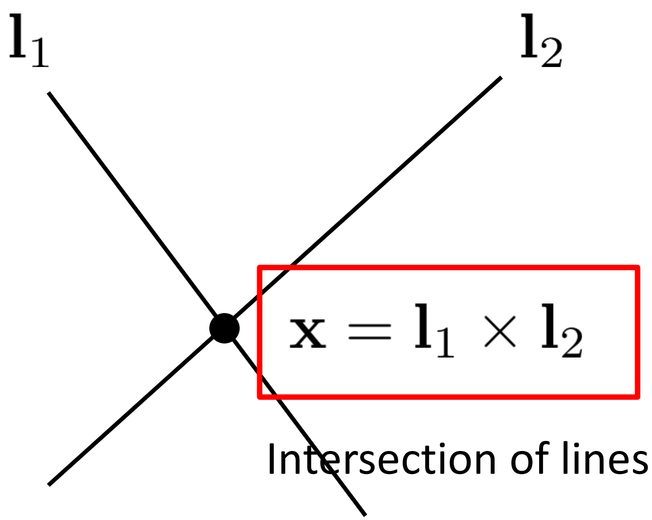
$$\mathbf{x} = \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \\ w \end{pmatrix} = \lambda \mathbf{x} \sim \begin{pmatrix} x_1/w \\ x_2/w \\ \dots \\ x_n/w \end{pmatrix}$$

Homogenous coordinate　　Euclidean coordinate

  - Can represent infinite points or lines

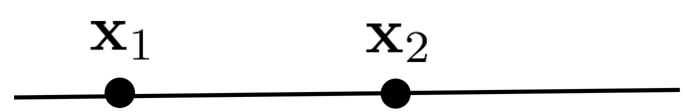
Homogenous coordinates

- 2D points and lines
  - A point is represented by $\mathbf{x} = (x, y, 1)^T$
  - A line is represented by $\mathbf{l} = (a, b, c)^T$
  - A line equation is $\mathbf{l}^T\mathbf{x} = 0$



$$\boxed{\mathbf{x} = \mathbf{l}_1 \times \mathbf{l}_2}$$

Intersection of lines



$$\boxed{\mathbf{l} = \mathbf{x}_1 \times \mathbf{x}_2}$$

Line across two points

Here, ‘ $\times$ ‘ represents cross production

Homogenous coordinates

- 2D points and lines
  - An infinite point is represented by $\mathbf{x} = (x, y, 0)^T$
  - All infinite points are on the infinite line

$$\mathbf{l} = (0, 0, 1)^T$$

- 3D points and planes
  - A point is represented by $\mathbf{x} = (x, y, z, w)^T$
  - A plane is represented by $\Pi = (\pi_1, \pi_2, \pi_3, \pi_4)^T$

$$\begin{pmatrix} \Pi_1^T \\ \Pi_2^T \\ \Pi_3^T \end{pmatrix} \mathbf{x} = 0 \qquad \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \end{pmatrix} \Pi = 0$$

Intersection of three planes **Plane across three points**

Homogenous coordinates

- 2D points and lines
  - An infinite point is represented by $\mathbf{x} = (x, y, 0)^T$
  - All infinite points are on the infinite line

$$\mathbf{l} = (0, 0, 1)^T$$

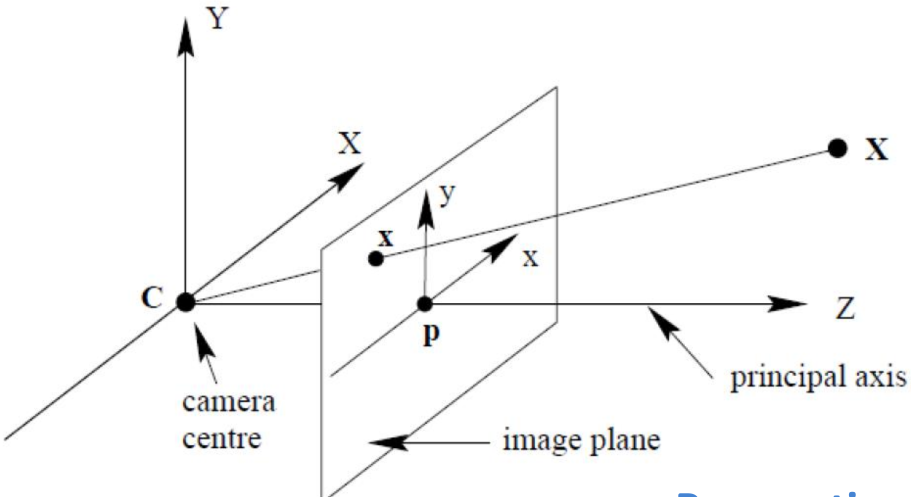
- 3D points and planes
  - A point is represented by $\mathbf{x} = (x, y, z, w)^T$
  - A plane is represented by $\Pi = (\pi_1, \pi_2, \pi_3, \pi_4)^T$

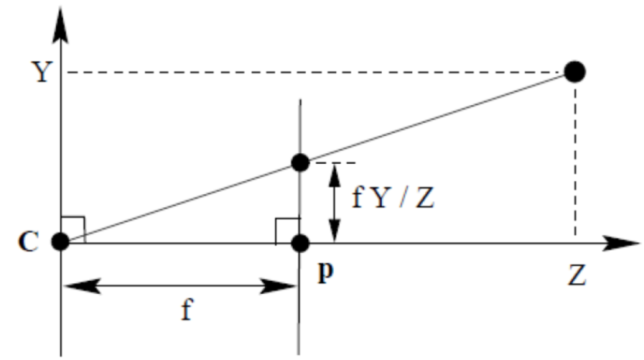
$$\begin{pmatrix} \Pi_1^T \\ \Pi_2^T \\ \Pi_3^T \end{pmatrix} \mathbf{x} = 0 \qquad \begin{pmatrix} \mathbf{x}_1^T \\ \mathbf{x}_2^T \\ \mathbf{x}_3^T \end{pmatrix} \Pi = 0$$

Intersection of three planes | **Plane across three points**

Pinhole camera model

- A pinhole camera model is illustrated as the follows





Perspective projection

$$\mathbf{X} = \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix} \Rightarrow \mathbf{x} = \begin{pmatrix} fX/Z \\ fY/Z \\ f \end{pmatrix} = \lambda \begin{pmatrix} u \\ v \\ f \end{pmatrix}$$

Pinhole camera model

- We have

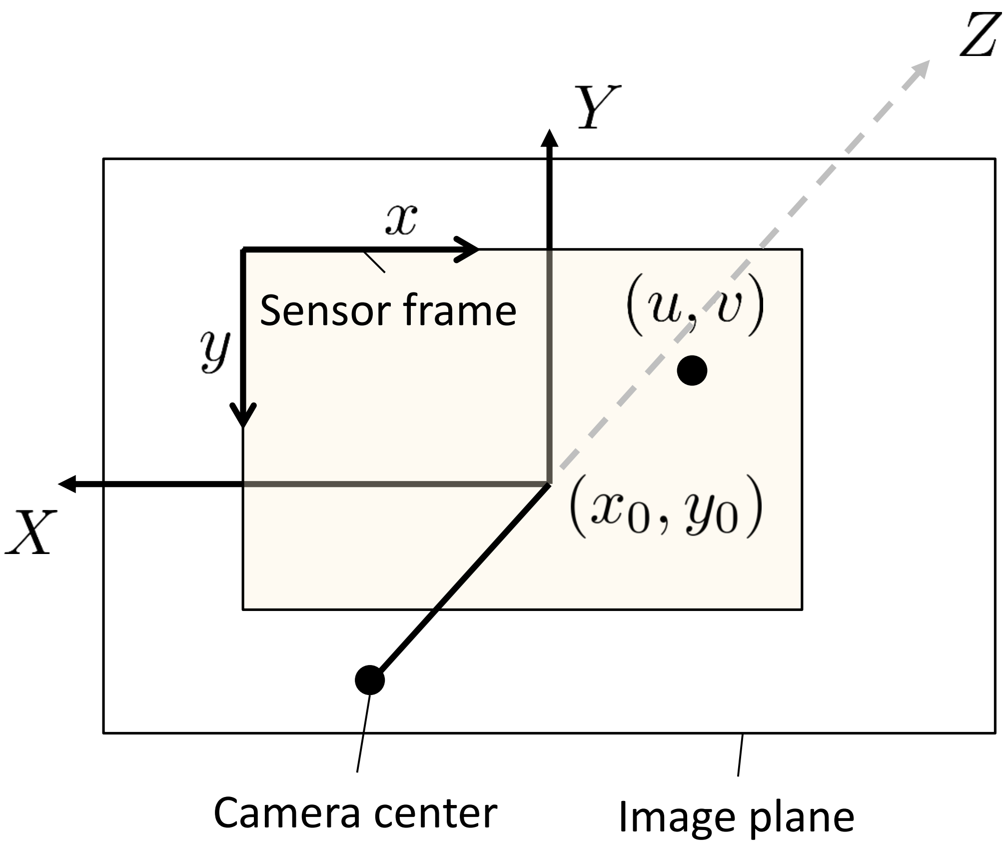
$$\begin{pmatrix} u \\ v \\ f \end{pmatrix} \propto \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{pmatrix} X \\ Y \\ Z \\ 1 \end{pmatrix}$$

image point **3x4 projection matrix** **scene point**

- Normalized image point will be then sensed by CCD or CMOS :
  – Next step: Image plane -> Sensor frame

Pinhole camera model

- Intrinsic camera parameters:
  - Image plane -> Sensor frame



$$x = k_x u + x0$$
$$y = k_y u + y0$$

where k_x and k_y are scaling factors, of which the units are ***pixels/length***

Nikon D610 camera:

- Maximum image resolution:
  6016 $\times$ 4016
- CMOS size:
  35.9 x 24 mm

We have :

$$k_x = 0.168 pixel/\mu m$$
$$k_y = 0.167 pixel/\mu m$$

Pinhole camera model

- Camera intrinsic matrix (or camera calibration matrix)

$$\mathbf{x} = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = \begin{bmatrix} fk_x & 0 & x_0 \\ 0 & fk_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \frac{1}{f} \begin{pmatrix} u \\ v \\ f \end{pmatrix}$$

$$= \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix}$$

$$= \mathbf{K} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix}$$

Pinhole camera model

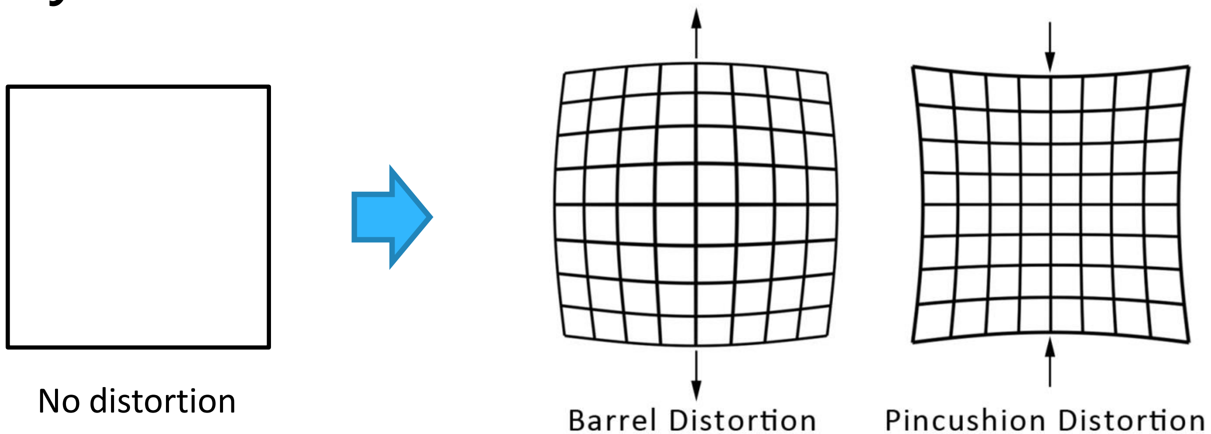
- $\mathbf{K}$ is a 3×3 upper triangle matrix, called camera intrinsic matrix (or camera calibration matrix)

$$\mathbf{K} = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

- There are four parameters:
  - **Principle point** (x_0, y_0) , which is point where the optical axis intersects the image plane
  - **Scaling factors** α_x, α_y in image x and y directions
  - In most cases the scaling factors are nearly the same, so sometimes only **three** parameters are used – two for principle point and one for scaling.

Image distortion

- Due to the imperfect imaging system, images are usually distorted.

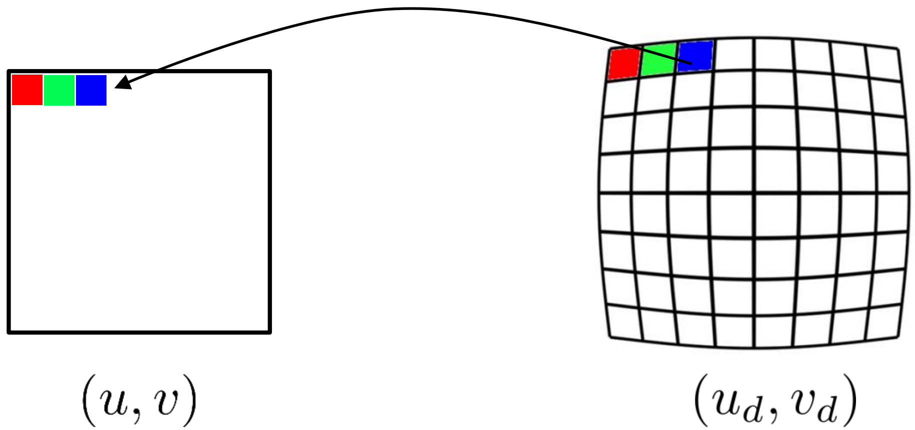


$$\begin{pmatrix} u_d \\ v_d \end{pmatrix} = \underbrace{(1 + D_1 r^2 + D_2 r^4 + D_5 r^6) \begin{pmatrix} u \\ v \end{pmatrix}}_{\text{Radial distortion}} + \underbrace{\begin{pmatrix} 2D_3 uv + D_4(r^2 + 2u^2) \\ D_3(r^2 + 2v^2) + 2D_4 uv \end{pmatrix}}_{\text{Tangential distortion}}$$

Distortion coefficients: $D = [D_1, D_2, D_3, D_4, D_5]$

Remove the distortion

- Rectify the distorted image:
  - For each pixel in the destination image (without distortion), find its corresponding pixel in the distorted image.
  - Fill the color of the corresponding pixel in the distorted image into the current pixel.
  - Repeat above steps until all pixels are filled.



Remove the distortion

- Compute the original coordinate from the distorted coordinate :

$$(u, v) \leftarrow (u_d, v_d)$$

  – Directly solve (u, v) from (u_d, v_d) is very difficult!

$$\begin{pmatrix} u_d \\ v_d \end{pmatrix} = \underbrace{(1 + D_1 r^2 + D_2 r^4 + D_5 r^6)}_{\tau} \begin{pmatrix} u \\ v \end{pmatrix} + \underbrace{\begin{pmatrix} 2D_3 uv + D_4(r^2 + 2u^2) \\ D_3(r^2 + 2v^2) + 2D_4 uv \end{pmatrix}}_{d\mathbf{x}}$$

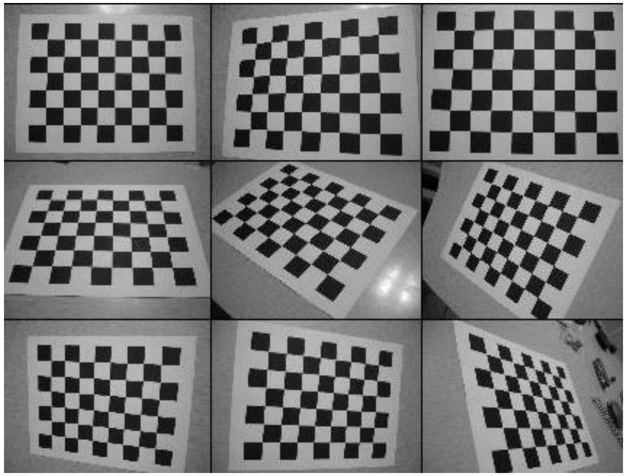
  – We can solve it iteratively :
    - Initially we let $(u_1, v_1) \leftarrow (u_d, v_d)$
    - Repeat until convergence :
      – Compute $\tau, d\mathbf{x}$ using (u_{i-1}, v_{i-1}).
      – We get (u_i, v_i) by

$$\begin{pmatrix} u_i \\ v_i \end{pmatrix} = \left(\begin{pmatrix} u_d \\ v_d \end{pmatrix} - d\mathbf{x} \right) / \tau$$

Usually, convergence can be achieved in 3~5 iterations

Camera calibration

- Calibration using checkerboard pattern
  - Use several snapshots of a checkerboard pattern to compute the intrinsic parameters.



Zhang, Zhengyou. "A flexible new technique for camera calibration." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 22.11 (2000): 1330-1334.

$$\mathbf{K} = \begin{bmatrix} \alpha_x & 0 & x_0 \\ 0 & \alpha_y & y_0 \\ 0 & 0 & 1 \end{bmatrix}$$

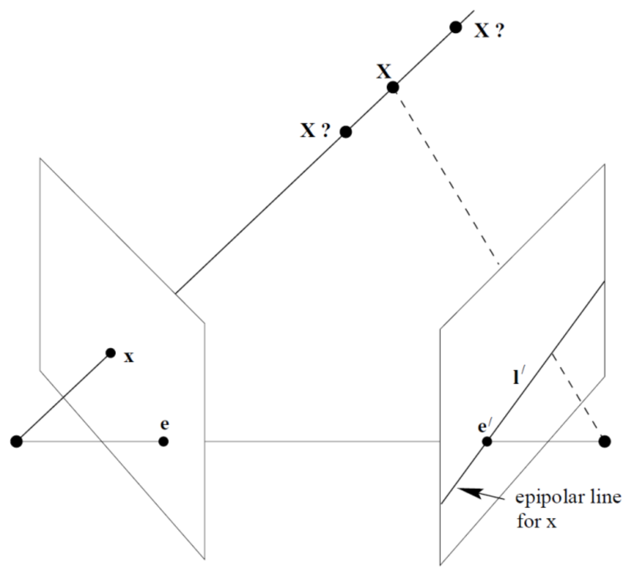
$$\mathbf{D} = [D_1, D_2, D_3, D_4, D_5]$$

Two view geometry

- Two view geometry tries to answer the following questions.
  - Given a image point in one view, where is its corresponding point in the other view?
    - **Epipolar constraint**
  - What is the relative pose between two views given a set of correspondences?
    - **Fundamental/Essential matrix estimation**
  - What is the 3D geometry of the scene?
    - **Triangulation**

Two view geometry

- Epiploar constraint:
  – Given a image point in the first view, how can we search its correspondence in the next view?



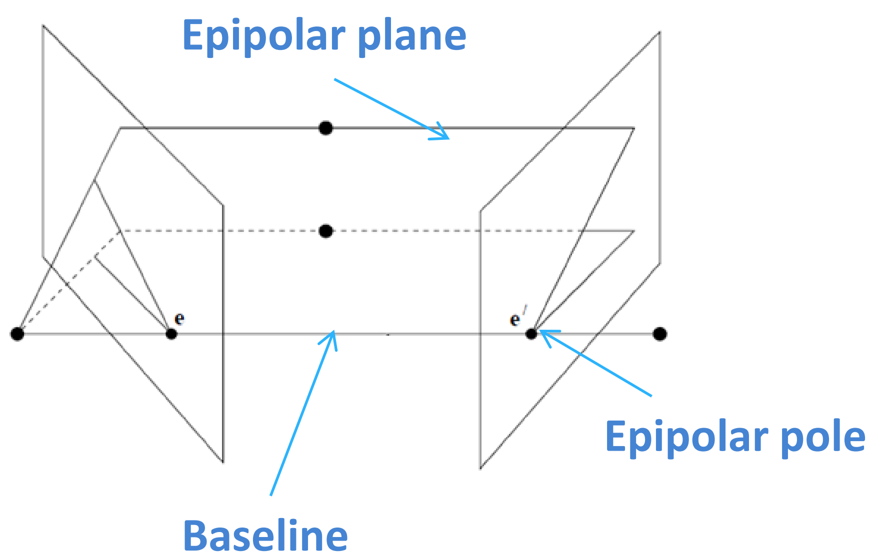
- Its correspondence lies in a line, which is named as '***epipolar line***' of $\mathbf{x}$.
- The geometry determining the epiplolar line is '***epipolar geometry***'
- The constraint that the correspondence of $\mathbf{x}$ should lie in the epipolar line is the ***epipolar constraint.***

Epiploar constraint

- The epipolar constraint is described mathematically as

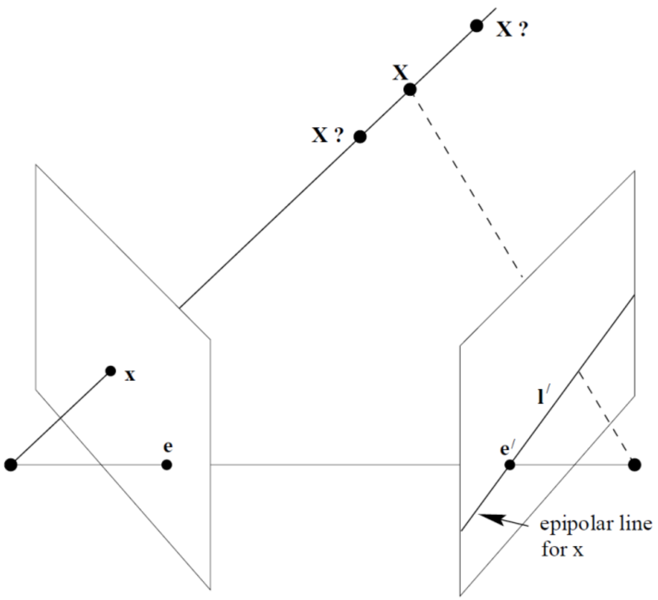
$$\mathbf{x}'^T F \mathbf{x} = 0$$

  - Here F is the **fundamental matrix**
  - $\mathbf{l} = F\mathbf{x}$ is the **epipolar line** of $\mathbf{x}$



Epiploar constraint

- Epipolar constraint - Derivation



$$\mathbf{x} = [I\ \mathbf{0}]\mathbf{X} \qquad \mathbf{x}' = [R\ t]\mathbf{X}$$

$$\mathbf{l}' = \mathbf{e}' \times \mathbf{x}'$$

$$\mathbf{l}' = t \times \mathbf{x}' \qquad (\mathbf{e}' = [R\ t](0,0,0,1)^T)$$

$$\begin{aligned} \mathbf{l}' &= t \times (R\mathbf{x} + t)\lambda \qquad (\mathbf{X} = \lambda(\mathbf{x}, 1)^T) \\ &= t \times R\mathbf{x} \\ &= [t]_\times R\mathbf{x} \end{aligned}$$

$$\boxed{\mathbf{x}'^T E \mathbf{x} = 0} \qquad (\mathbf{x}'^T \mathbf{l} = 0)$$

$E = [t]_\times R$ is the **essential matrix**

Epiploar constraint

- Essential matrix and fundamental matrix

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

$$\Downarrow$$

$$\mathbf{m}'^T K'^{-T} E K^{-1} \mathbf{m} = 0 \qquad (\mathbf{m}' = K'\mathbf{x}', \mathbf{m} = K\mathbf{x})$$

$$\Downarrow$$

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

$$\Downarrow$$

$$\boxed{F = K'^{-T} E K^{-1}}$$

Fundamental matrix

Fundamental matrix estimation

- Given a set of point correspondences $\{\mathbf{x}_i \leftrightarrow \mathbf{x}'_i\}$, solve

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

  - Fundamental matrix is a rank-2 matrix and has seven degree of freedom
  - At least 7 points are required to solve the fundamental matrix, where each point provides one equation.
  - **Eight point algorithm**
  - **Seven point algorithm**

Fundamental matrix estimation

- **Eight point algorithm**
  - For each correspondence $\mathbf{x} \leftrightarrow \mathbf{x}'$, we have the equation

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

which can be written as

$$(x'x, x'y, x', y'x, y'y, y', x, y, 1) \begin{pmatrix} f_{11} \\ f_{12} \\ f_{13} \\ f_{21} \\ f_{22} \\ f_{23} \\ f_{31} \\ f_{32} \\ f_{33} \end{pmatrix} = 0$$

where $\mathbf{f} = (f_{11}, f_{12}, f_{13}, f_{21}, f_{22}, f_{23}, f_{31}, f_{32}, f_{33})^\top$ holds the entities of $\boldsymbol{F}$

Two view geometry

- The entries of fundamental matrix can be solved by stacking all equations together.

$$\mathbf{Af} = 0$$

- Since $\boldsymbol{F}$ is determined up to scale only, at least eight points are required to solve $\boldsymbol{F}$.
- The solution can also be obtained by SVD decomposition.

Least-squares solution

(i) Form equations $\mathtt{A}\mathbf{f} = \mathbf{0}$.
(ii) Take SVD : $\mathtt{A} = \mathtt{UDV}^\top$.
(iii) Solution is last column of $\mathtt{V}$ (corresp : smallest singular value)
(iv) Minimizes $||\mathtt{A}\mathbf{f}||$ subject to $||\mathbf{f}|| = 1$.

Fundamental matrix estimation

- Singularity correction
  - The solution by 8 point algorithm does not satisfy the singularity condition.
    - F is rank-2 matrix or mathematically, $det(F) = 0$
  - SVD approximation
    - Decompose F by SVD $F = U\Sigma V^T$
    - Here $\Sigma = diag(r, s, t)$.
    - The SVD approximation of F is

$$F' = U diag(r, s, 0) V^T$$

F' is the 'closest' singular matrix to F in Frobenius norm!

Fundamental matrix estimation

- **Seven point algorithm**
  - If we impose the singularity condition on the unknowns, we get another equation.
  - So we can solve the fundamental matrix by 7 points
- Steps:
  - 1. Get the null space solution of $\mathbf{A}\mathbf{f} = 0$

$$\mathbf{f} = \lambda \mathbf{f}_0 + \mu \mathbf{f}_1$$

  - 2. Rewrite it into the matrix form

$$\mathrm{F} = \lambda \mathrm{F}_0 + \mu \mathrm{F}_1$$

  - 3. Condition $\det(F) = 0$ gives a cubic equation of λ, μ
  - 4. Solve λ, μ and get F

Essential matrix estimation

- Compute essential matrix
  - Once the camera has been calibrated.
  - Only five points are required to solve essential matrix since there is only five degree of freedom in essential matrix

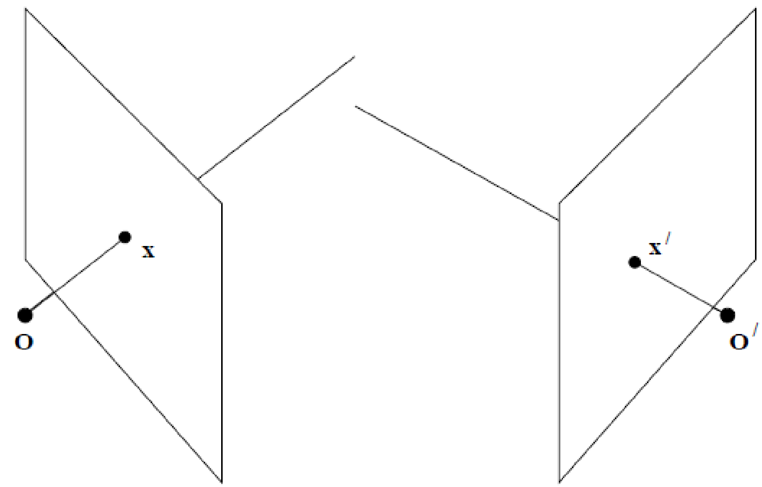
$$E = [t]_{\times} R$$

  - Nister’s five point algorithm

Nistér, David. "An efficient solution to the five-point relative pose problem."*Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.6 (2004): 756-770.

Triangulation

- Knowing P and P'
- Knowing $\boldsymbol{x}$ and $\boldsymbol{x}'$
- Compute $\boldsymbol{X}$



$$\mathbf{x} = P\mathbf{X}$$

$$\mathbf{x}' = P'\mathbf{X}$$

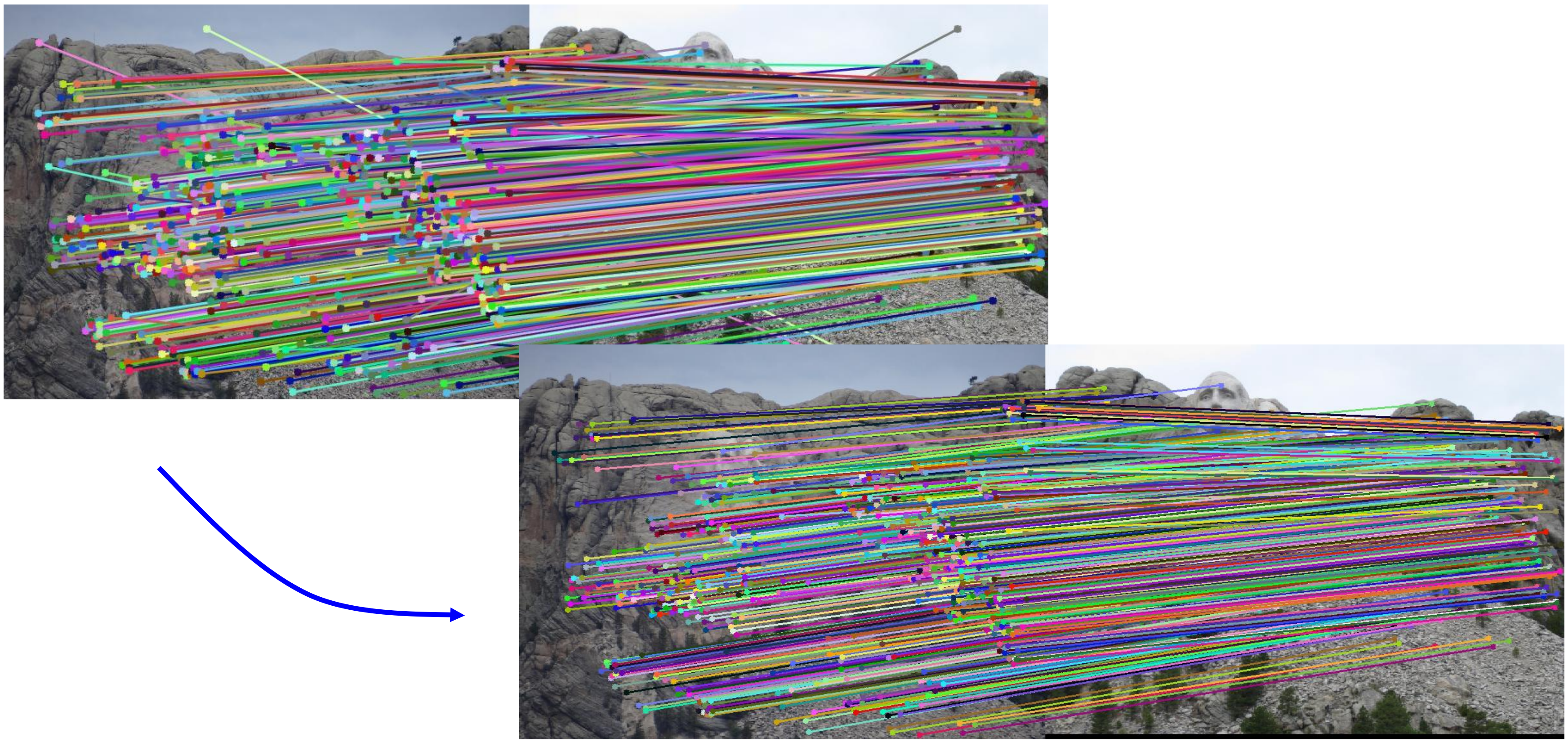
Refinement

- Minimizing the re-projection errors $\|x - f(P, X)\|^2 + \|x' - f(P', X)\|^2$

Here $f(P, X) = \begin{pmatrix} \frac{p_1 x + p_2 y + p_3 z + p_4}{p_9 x + p_{10} y + p_{11} z + p_{12}} \\ \frac{p_5 x + p_6 y + p_7 z + p_8}{p_9 x + p_{10} y + p_{11} z + p_{12}} \end{pmatrix}$. It is a nonlinear least square problem and can be solved by Levenberg-Marquardt algorithm efficiently.

RANSAC algorithm

- **RAN**dom **S**ample **A**nd **C**onsensus
  - Robust estimation under the presence of outliers



RANSAC algorithm

- Randomly select a small subset of correspondences and solve the Fundamental/Essential matrix

- Evaluate the error residuals for the rest of the correspondences. The **Consensus set** is the set of correspondences within the error threshold

- Repeat above steps and finally select solution that yields the largest consensus set.

A quick way to learn all about this

- Write a simple program to reconstruct 3D points from two snapshots. For example, use your phone.
- The pipeline
  - 1. calibrate the camera intrinsic parameters
  - 2. take two pictures by your phone
  - 3. Match feature points (SIFT, SURF)
  - 4. Use RANSAC algorithm to estimate the fundamental matrix and remove the outlier
  - 5. use Nister’s code to estimate the essential matrix from the inlier corresponding points
  - 6. extract the R and t from the essential matrix
  - 7. triangulate the 3D points

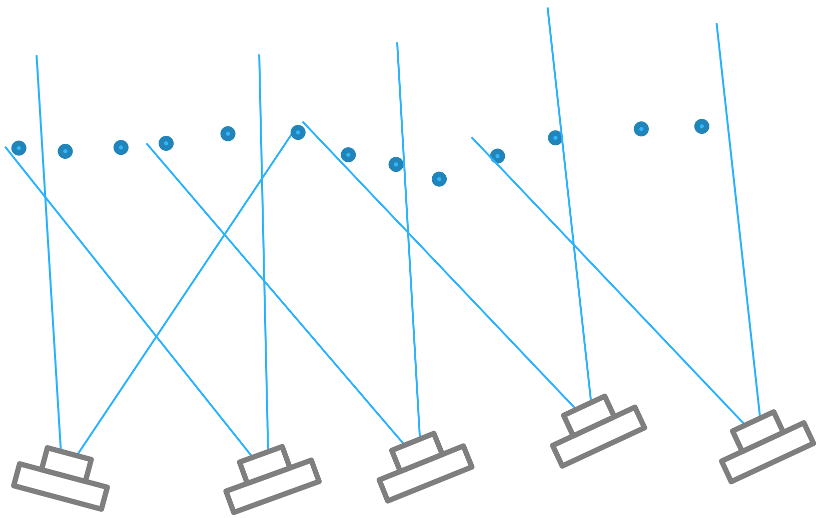
Outline

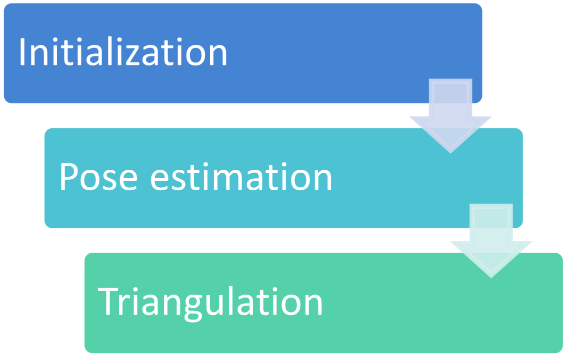
- Basic Theory
  - Pinhole camera model
  - Camera calibration
  - Two camera geometry

- Build a visual SLAM system
  - Structure-from-motion(SFM) approach:
    - **PTAM**
    - **ORB SLAM**
    - **CoSLAM**
  - Extended Kalman Filter approach:
    - **MonoSLAM**
    - **StructSLAM**

A quick overview of a SFM system

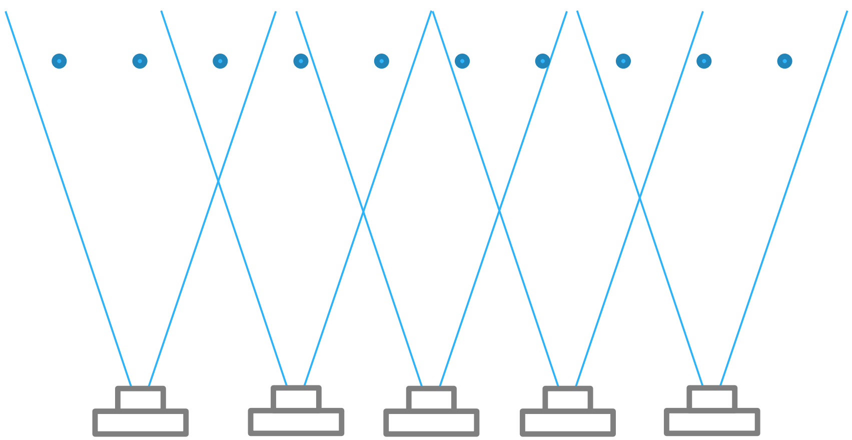
- A typical pipeline of incremental structure-from-motion (one camera case)

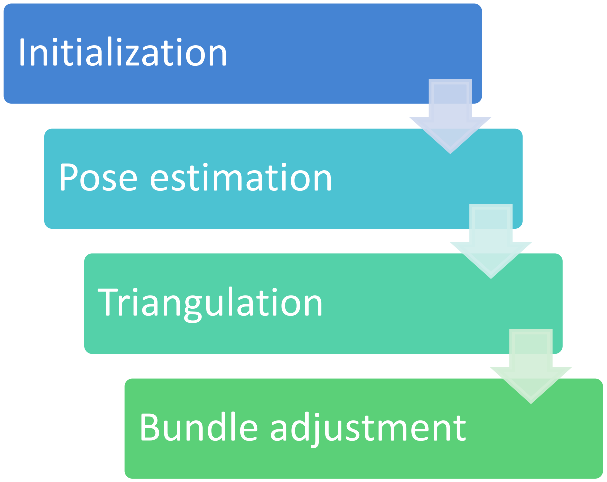




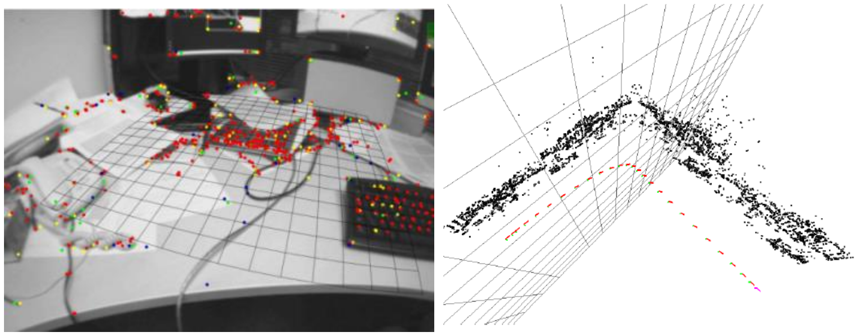
A quick overview of a SFM system

- A typical pipeline of incremental structure-from-motion (one camera case)



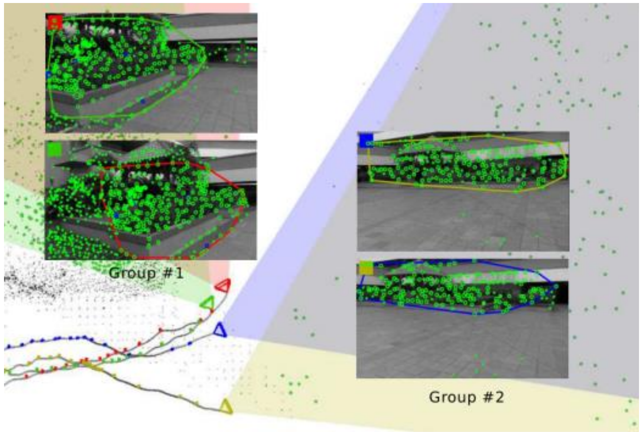


Existing systems



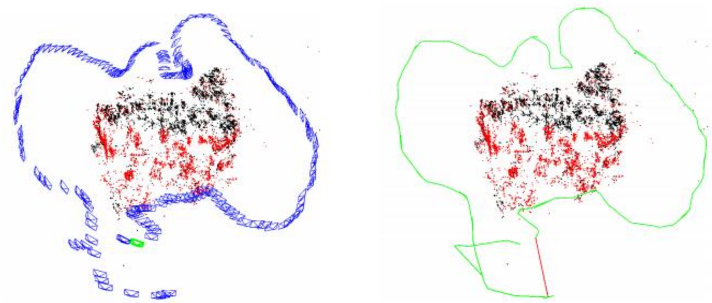
2007, PTAM

Klein, Georg, and David Murray. "Parallel tracking and mapping for small AR workspaces." *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*. IEEE, 2007.



2013, CoSLAM

Zou, Danping, and Ping Tan. "Coslam: Collaborative visual slam in dynamic environments." IEEE transactions on pattern analysis and machine intelligence 35.2 (2013): 354-366.

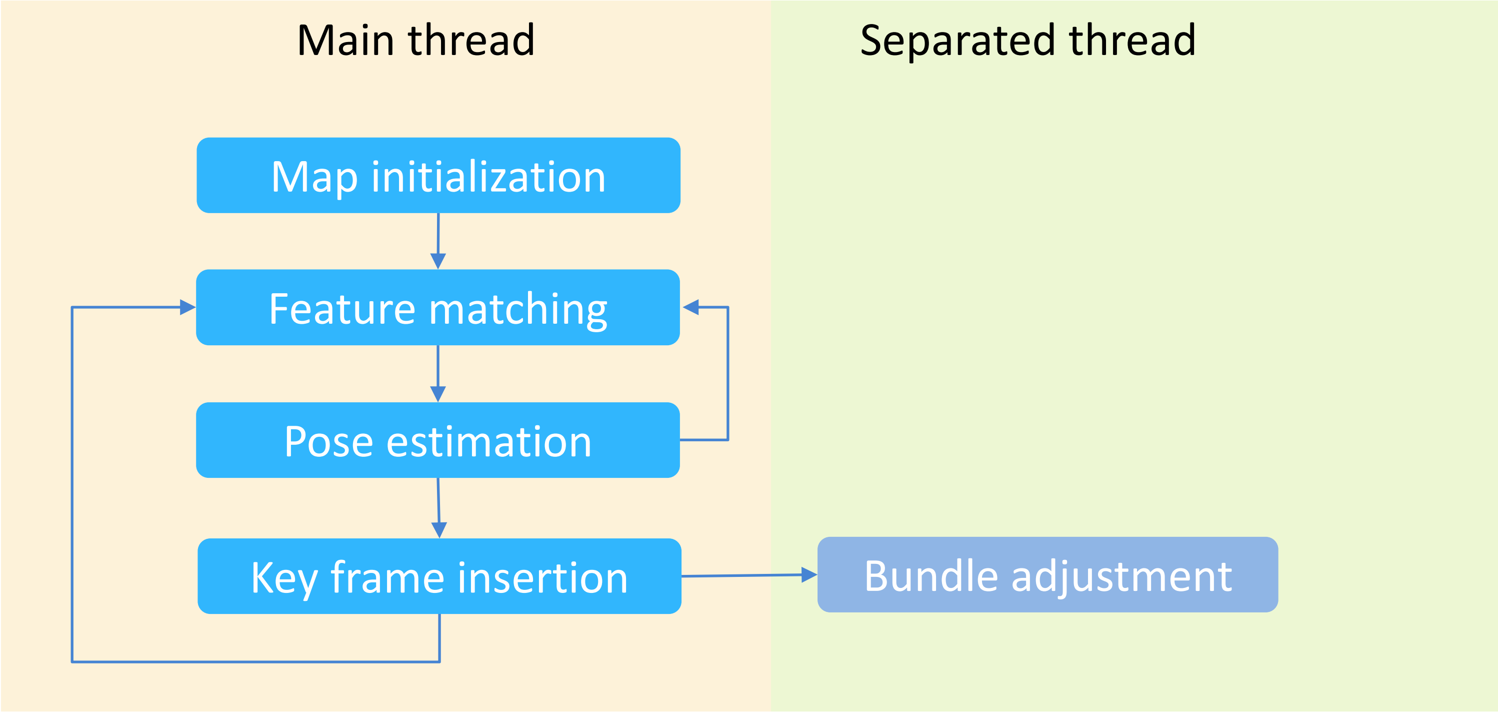


2015,ORB-SLAM

Mur-Artal, Raul, J. M. M. Montiel, and Juan D. Tardós. "Orb-slam: a versatile and accurate monocular slam system." IEEE Transactions on Robotics 31, no. 5 (2015): 1147-1163.

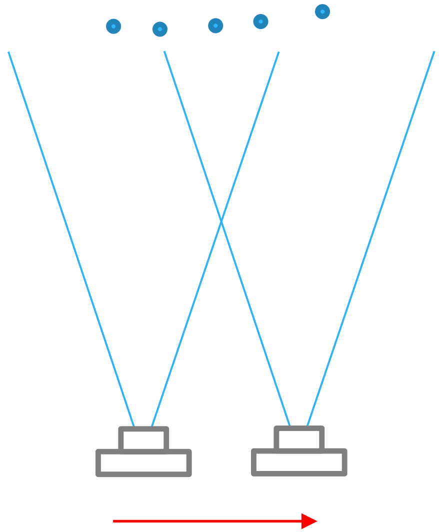
Single camera SLAM- overview

- Camera Tracking (Localization)
  - Feature matching
  - Pose estimation
- Mapping
  - Initialization
  - Key frame insertion
  - Bundle adjustment



Map initialization

- Map initialization
  - Use two images to get the **initial poses** and generate **seed map points**



Step1:Estimate the essential matrix

↓

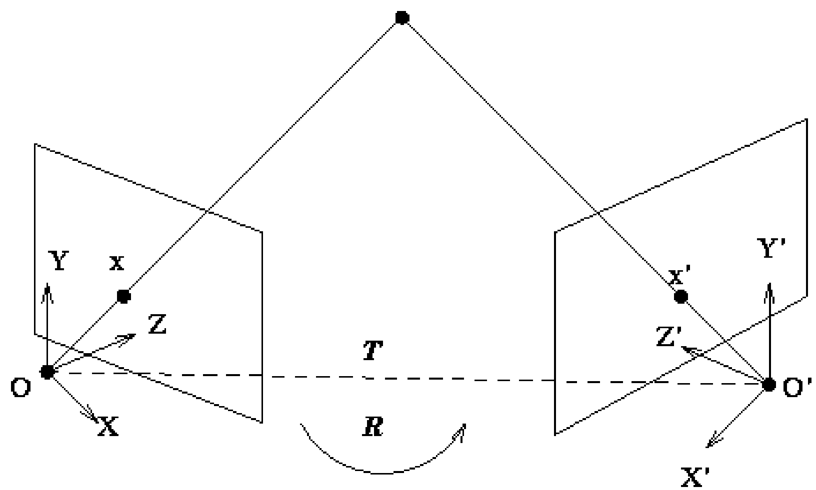
Step2:Decompose the relative pose

↓

Step3:Triangulation

Map initialization

- Estimate the essential matrix (five point algorithm)



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

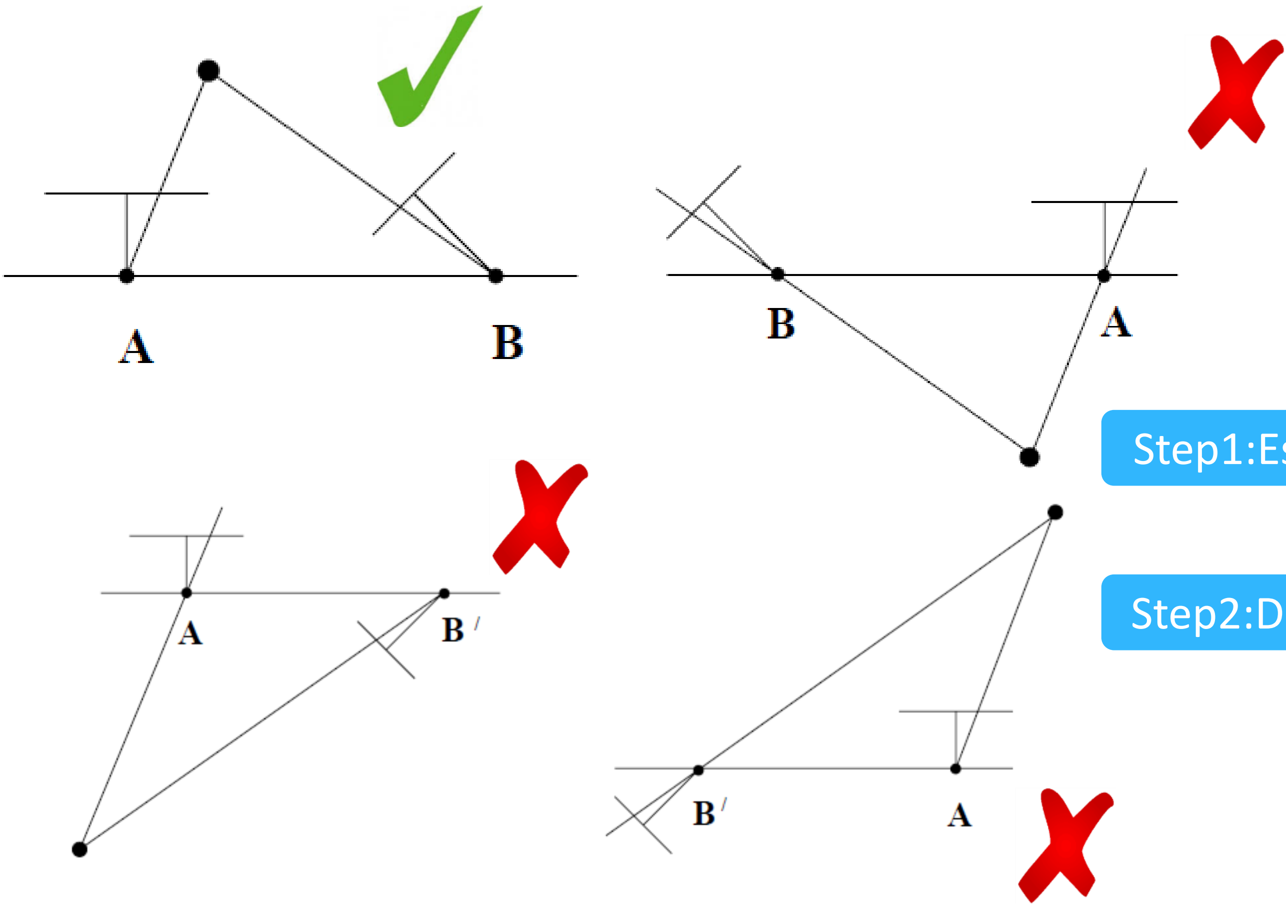
$$E = [t]_\times R$$

Step1:Estimate the essential matrix

Nistér, David. "An efficient solution to the five-point relative pose problem."*Pattern Analysis and Machine Intelligence, IEEE Transactions on* 26.6 (2004): 756-770.

Map initialization

- Relative pose decomposition. As I explained previous there are four possible solutions:



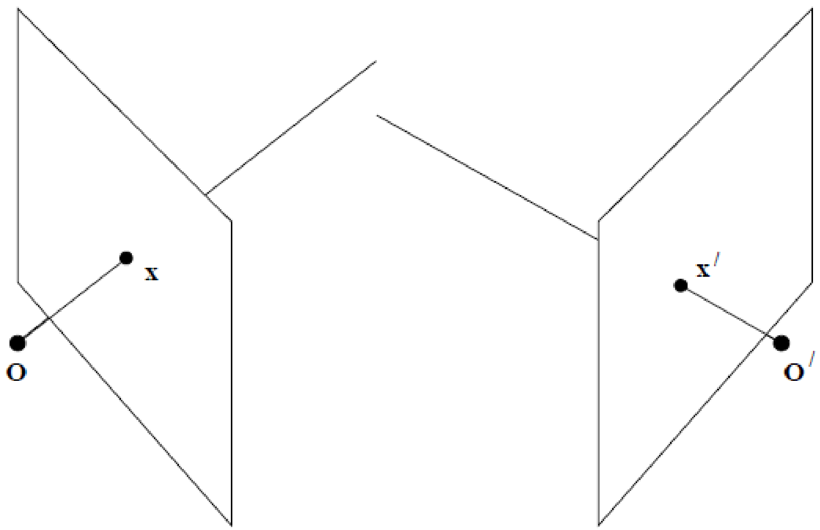
Step1:Estimate the essential matrix

Step2:Decompose the relative pose

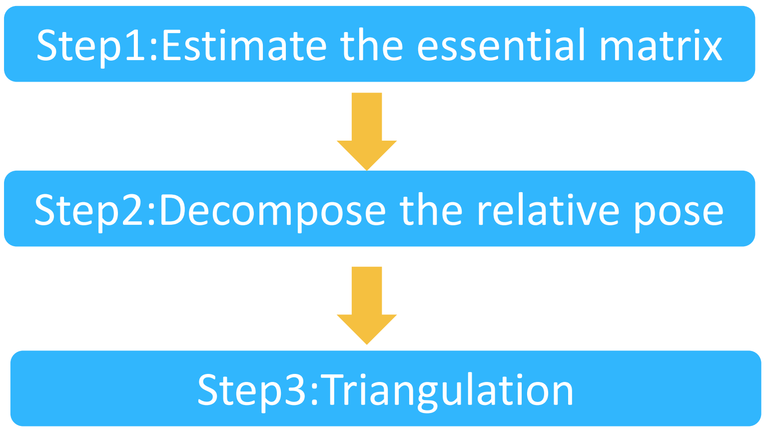
Map initialization

- Triangulation - generate 3D points

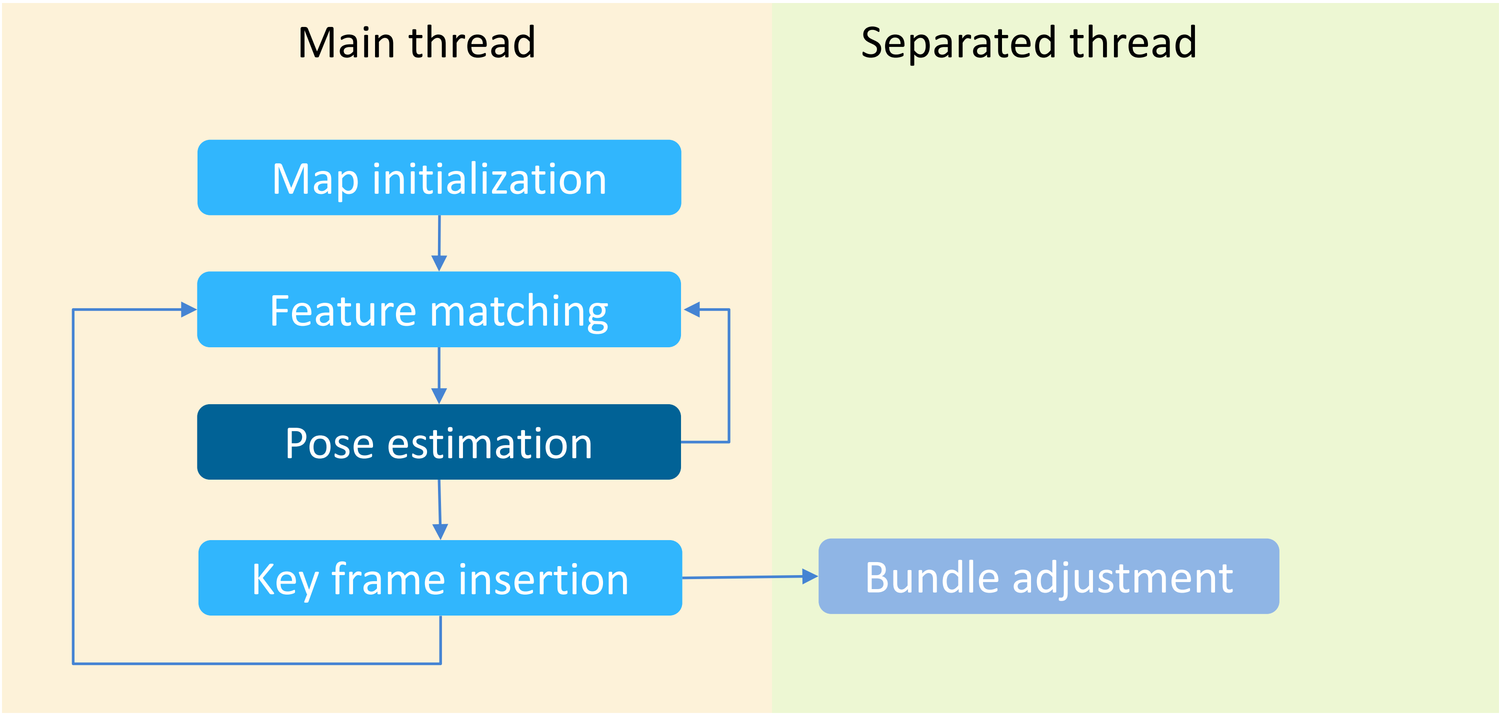
$$\mathbf{x} = P\mathbf{X} \qquad \mathbf{x}' = P'\mathbf{X}$$



$\Rightarrow \mathbf{X}$

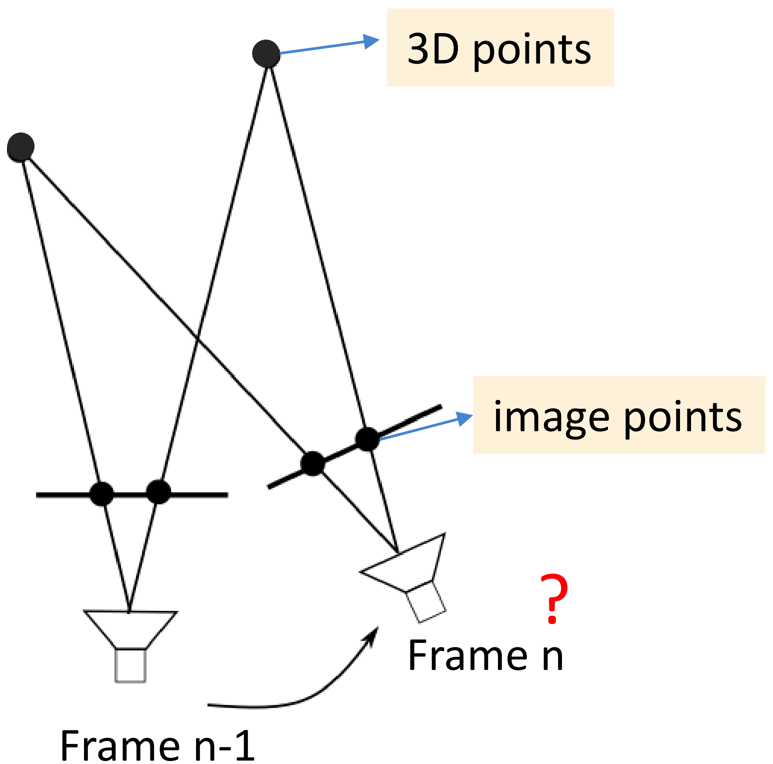


Pose estimation



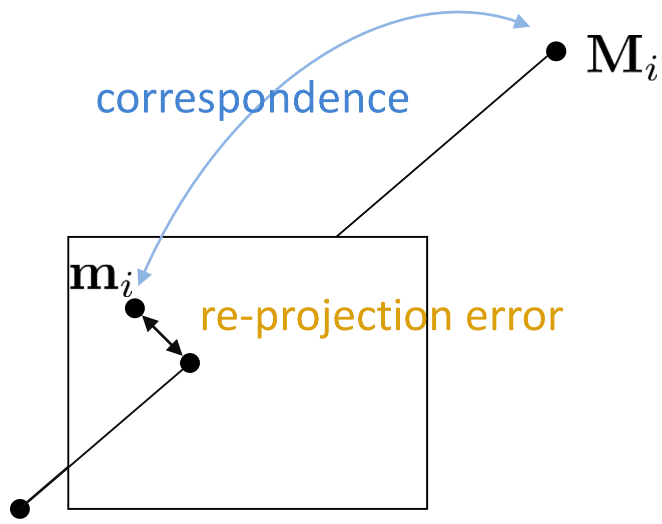
Pose estimation

- The problem:
  - Given 3D points and their corresponding images, how do we compute the camera pose ? (given that the camera is calibrated)



Pose estimation

- Denote 3D points by $\{\mathbf{M}_i\}$ and their corresponding images by $\{\mathbf{m}_i\}$.
- The re-projection error of a 3D point is defined as the distance between the image point and its projection.



Pose estimation

- Re-projection error:

$$r_i(\theta) = \mathbf{m}_i - Proj(\mathbf{M}_i, \theta)$$

Camera pose

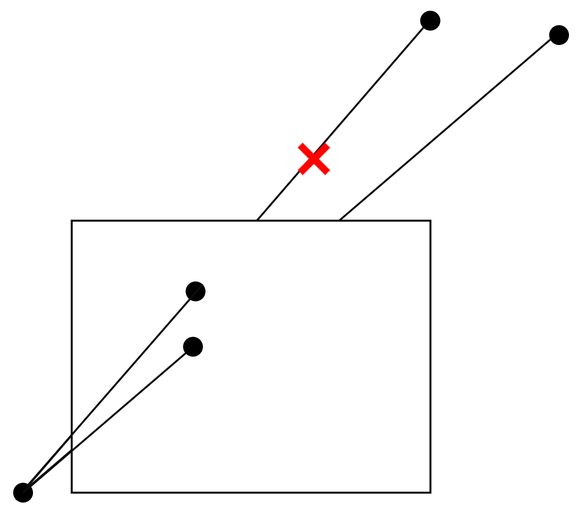
- We want find a pose that minimizes

$$\boxed{\theta^* = \arg\min_\theta \sum r_i^2(\theta)}$$

This is a standard non-linear least square problem, which can be solved by **Levenberg-Marquardt** algorithm.

Pose estimation

- How about if we get noisy correspondences?
  – Feature matching is not always correct!



$$\sum r_i^2(\theta)$$

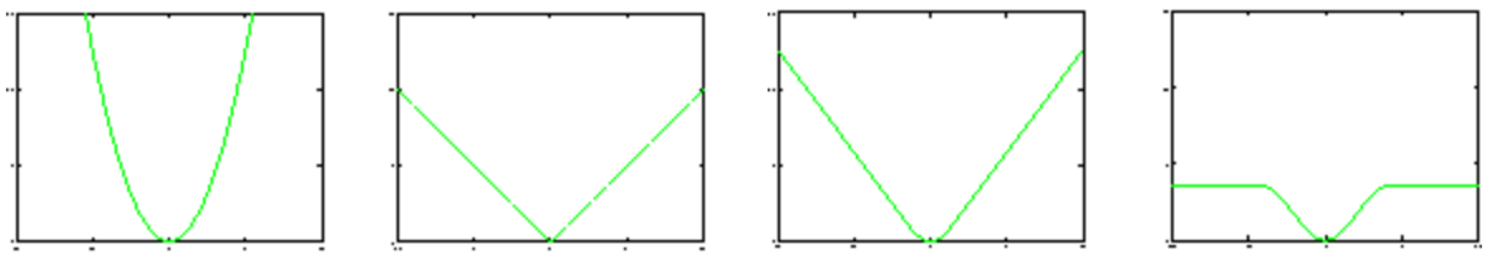
Pose estimation

- Another robust method : **M-estimator**

The **M-estimators** try to reduce the effect of outliers by replacing the squared residuals with another function.

$$\sum r_i^2(\theta) \Rightarrow \sum \rho(r_i(\theta))$$

$\rho(\cdot)$



L2 L1 Huber Tukey

Pose estimation

- **Least square V.S. M-estimator**

$$\sum r_i^2(\theta + \Delta\theta) \quad \leftrightarrow \quad \sum \rho(r_i(\theta + \Delta\theta))$$

$$\sum r_i \frac{\partial r_i}{\partial \Delta\theta} = 0 \qquad \sum \rho'(r_i) \frac{\partial r_i}{\partial \Delta\theta} = 0$$

$$\sum \underbrace{\boxed{\frac{\rho'(r_i)}{r_i}}}_{w(r_i)} r_i \frac{\partial r_i}{\partial \Delta\theta} = 0$$

This is a weighted least square problem!

M-estimator

- Reweighted least square algorithm:
  - Solve the weighted least square problem using initial weights (1s)
  - Evaluate the residual and update the weights
  - Repeat above steps for several times

$$\min \sum \rho(r_i(\theta))$$

Initialization: $\theta^{(0)} \leftarrow \min \sum r_i^2(\theta)$

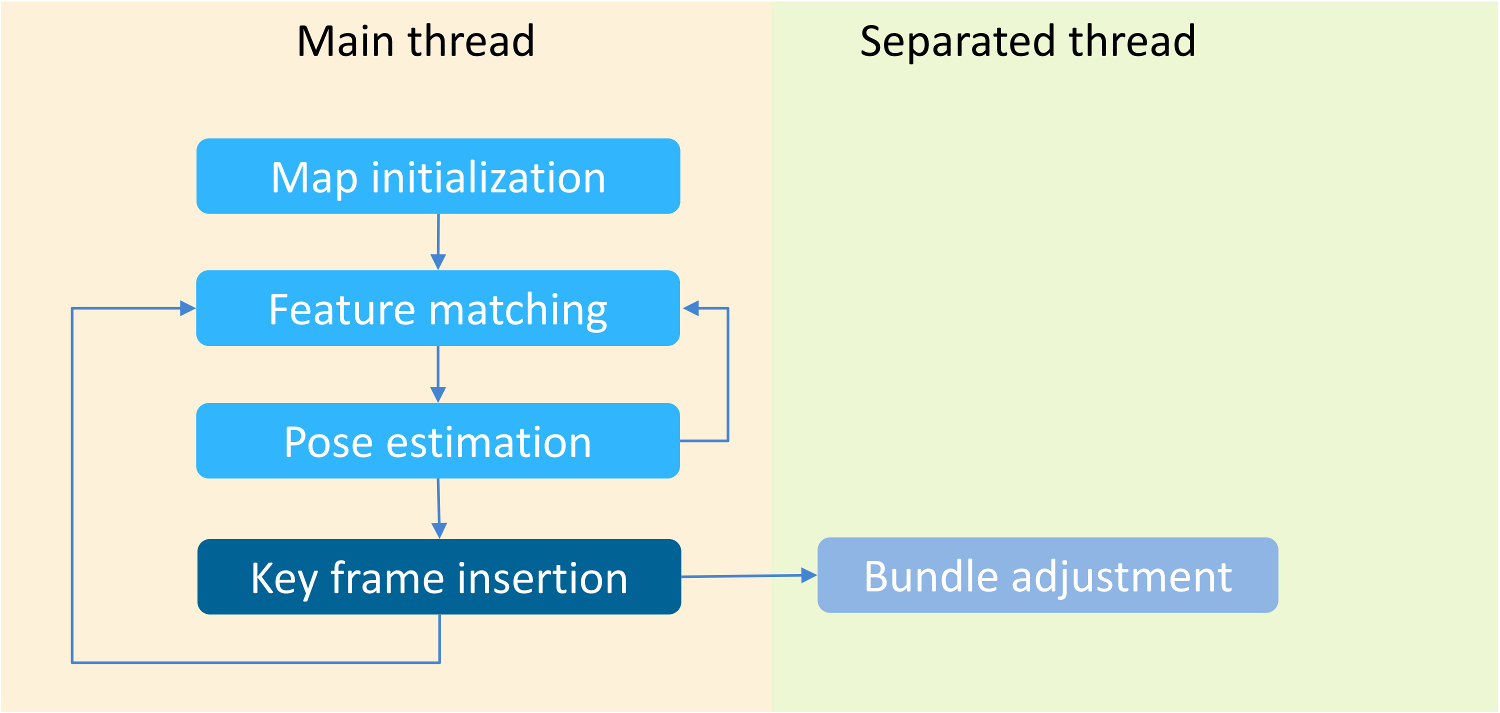
Compute new weights: $w^{(k)} \leftarrow \frac{\rho'(r_i^{(k-1)})}{r_i^{(k-1)}}$

Solve for new parameters $\theta^{k+1} \leftarrow \min \sum w^{(k+1)} r_i^2(\theta)$

M-estimator tutorial by Zhengyou Zhang

http://research.microsoft.com/en-us/um/people/zhang/INRIA/Publis/Tutorial-Estim/node24.html

Key frame selection



Key frame selection

- What is key frame ?
  - An structure storing:
    - current **camera pose**
    - current **3D points** and their **image correspondences**

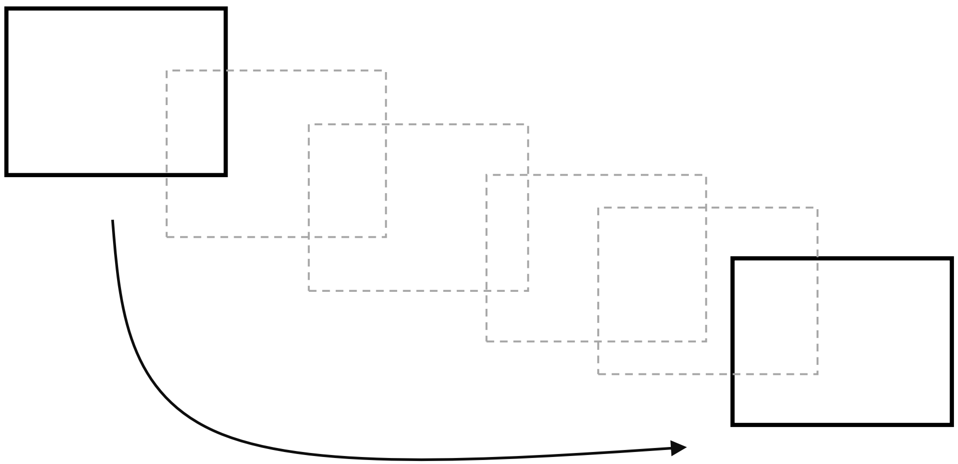
Question: Why not select all video frames as key frames?

Because it is not efficient (computation time + memory request)

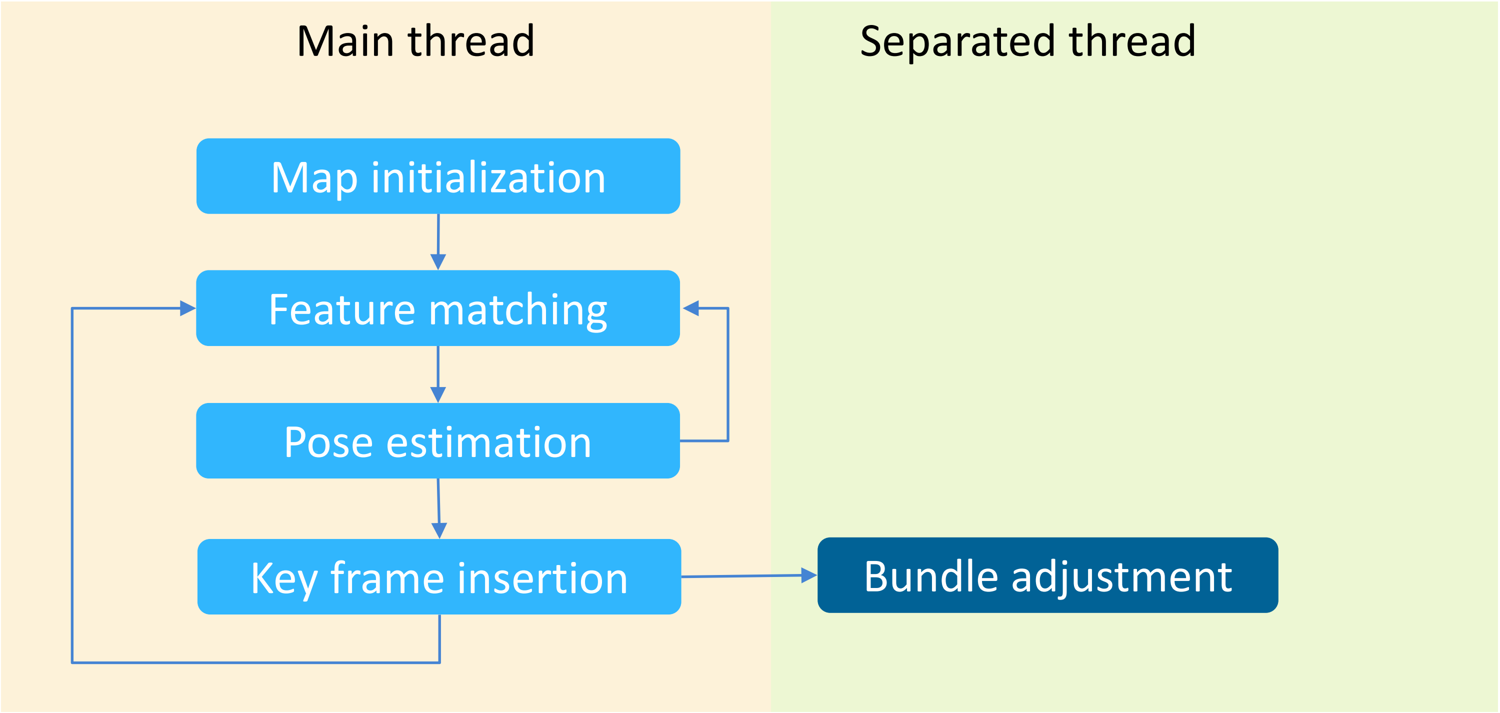
- two many points
- tw0 many camera poses

Key frame selection

- Some strategy to select key frame
  - A sufficient moving distance
  - Good quality of image
  - Maintain the number of features tracked



Bundle adjustment



Bundle adjustment

- What is bundle adjustment?
  - Bundle adjustment is to minimize re-projection errors in all views with respect to all 3D points and all camera poses

$$\min \sum_i \sum_j (\mathbf{m}_{ij} - Proj(\theta_i, \mathbf{M}_j))^2$$

  - This is still a non-linear least square problem

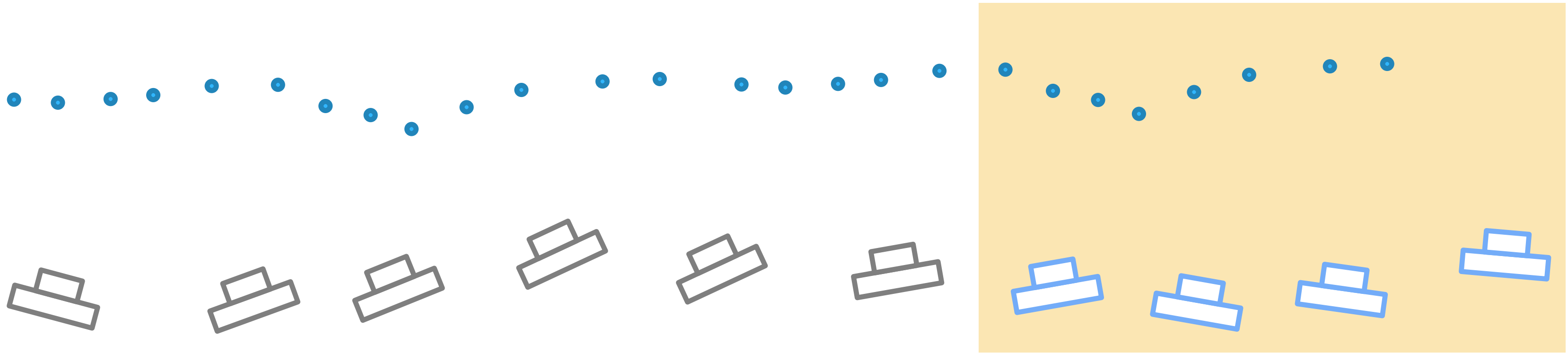
$$\min \sum_i \sum_j r_{ij}(\mathbf{x})^2$$

$\mathbf{x}$ is a vector containing all camera poses and 3D points

Software
sba: http://users.ics.forth.gr/~lourakis/sba/
mcba: http://grail.cs.washington.edu/projects/mcba/

Bundle adjustment

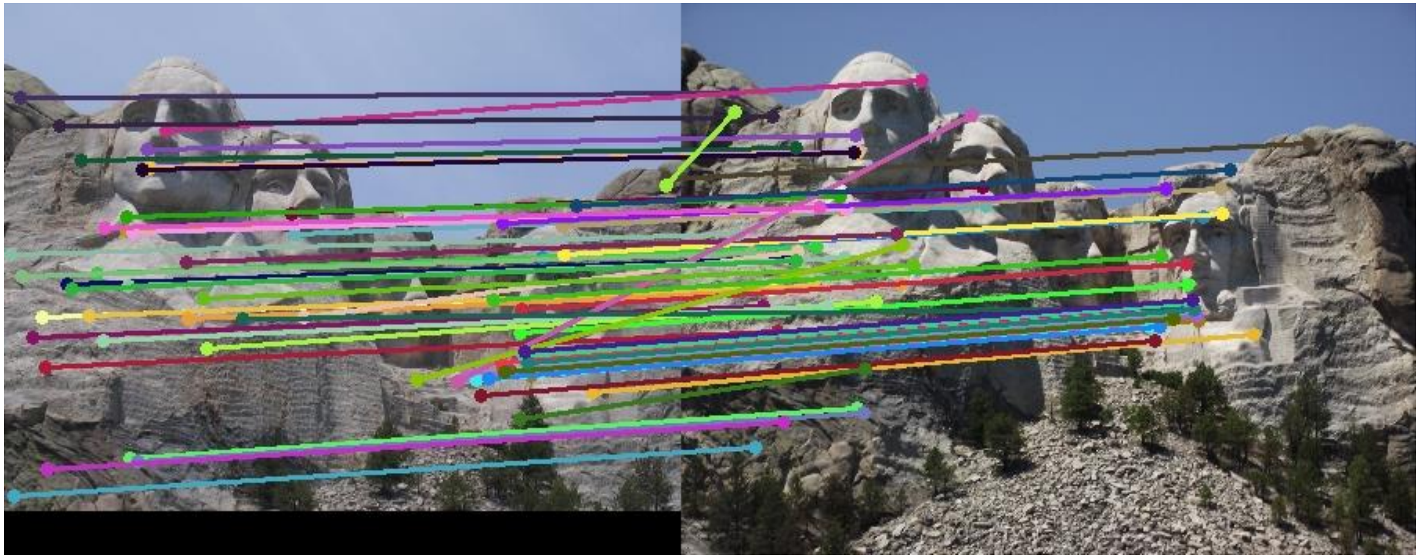
- Bundle adjustment with all parameters involved costs a lot of time.
  - A alternative solution is selecting only a subset of parameters to optimize. This approach is so called ***local bundle adjustment***.



Keyframes	2-49	50-99	100-149
Local Bundle Adjustment	170ms	270ms	440ms
Global Bundle Adjustment	380ms	1.7s	6.9s

Feature matching

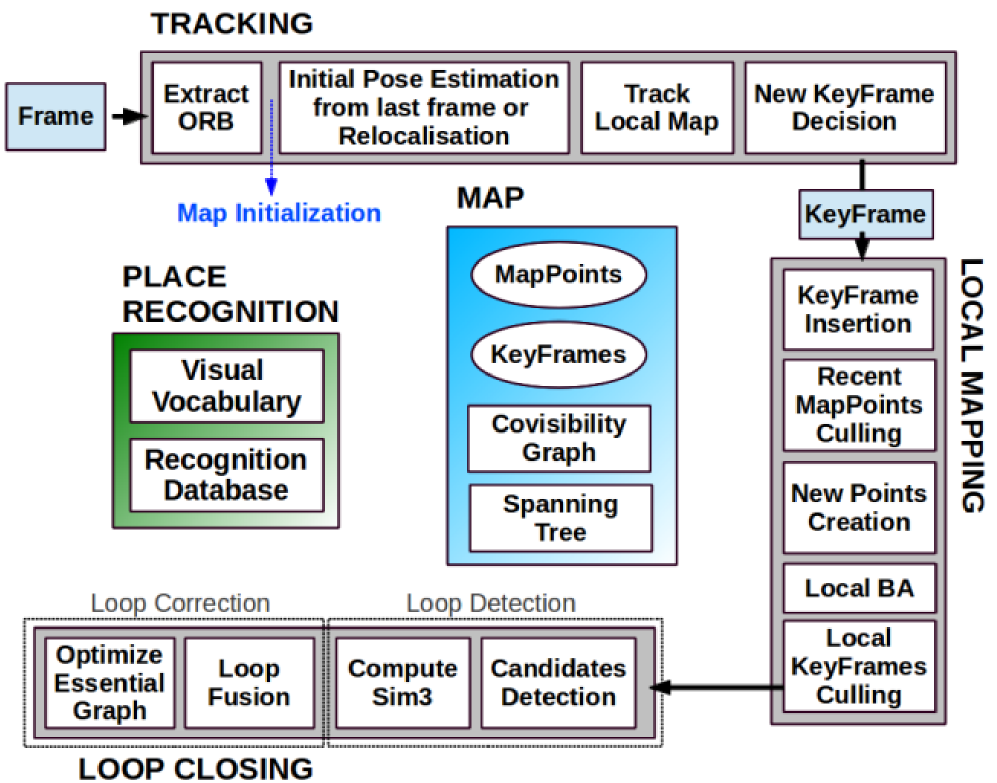
- PTAM : Fast corner (PTAM) &ZNCC matching
- ORB-SLAM : ORB feature & ORB matching
- CoSLAM :
  - Intra-camera : Harris corner & KLT tracking
  - Inter-camera : ZNCC matching



ORB-SLAM

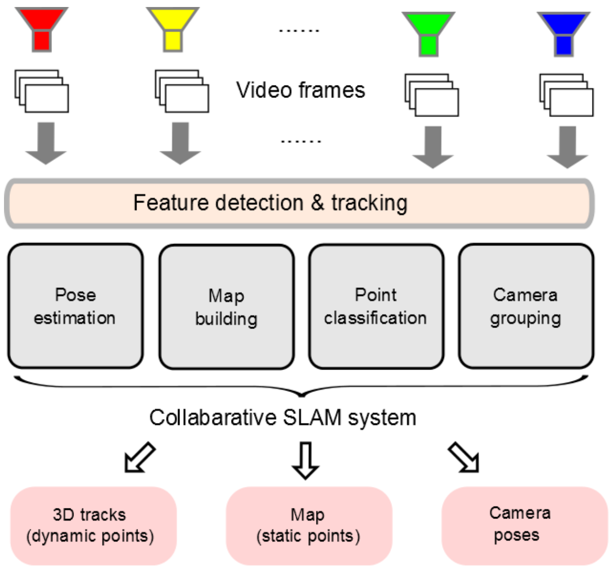
A extension from PTAM

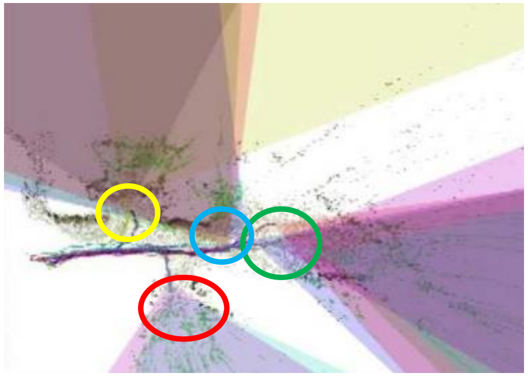
- Robust initialization
- Loop closing



CoSLAM

- Visual SLAM system for robot swarms

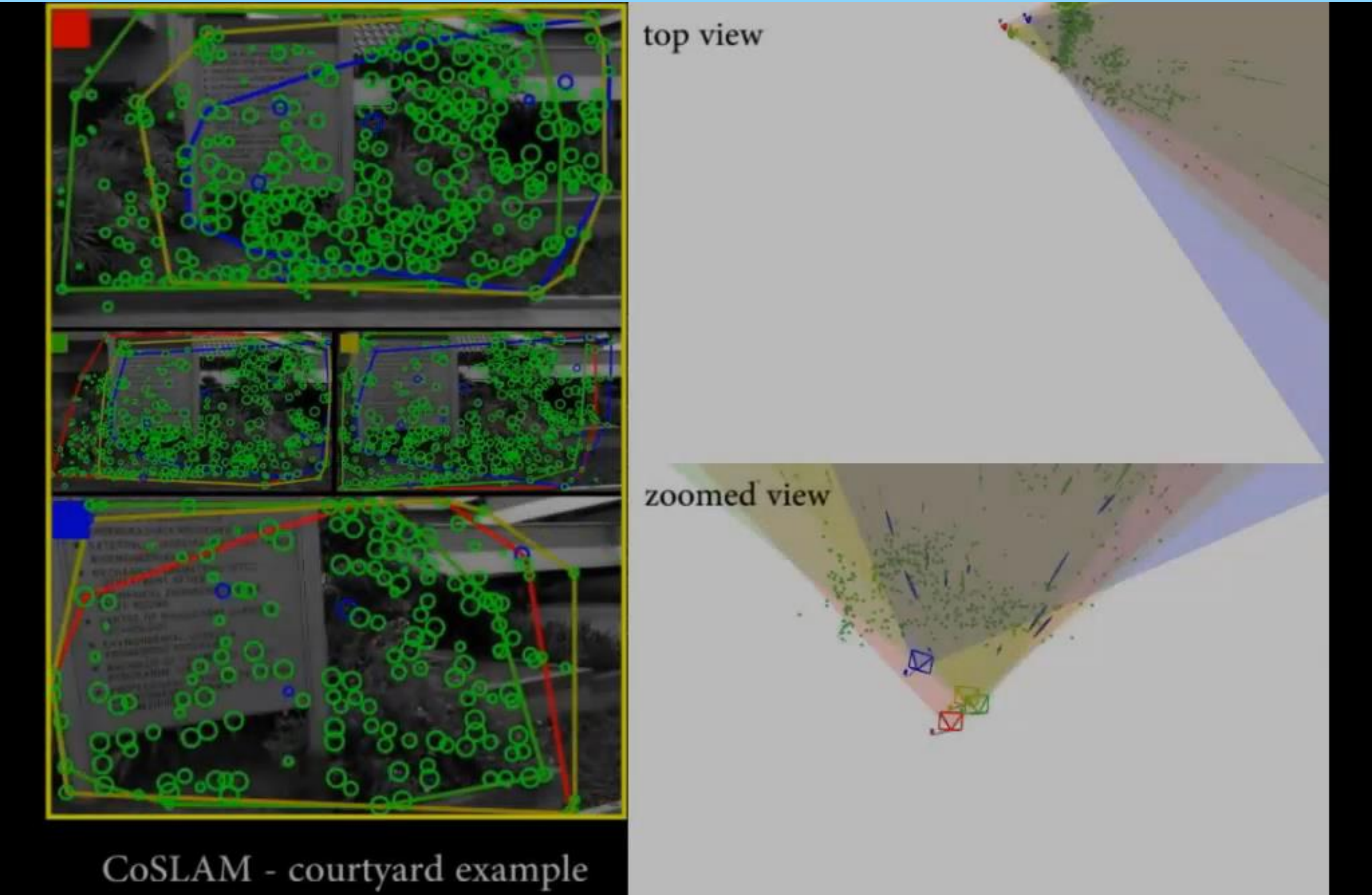




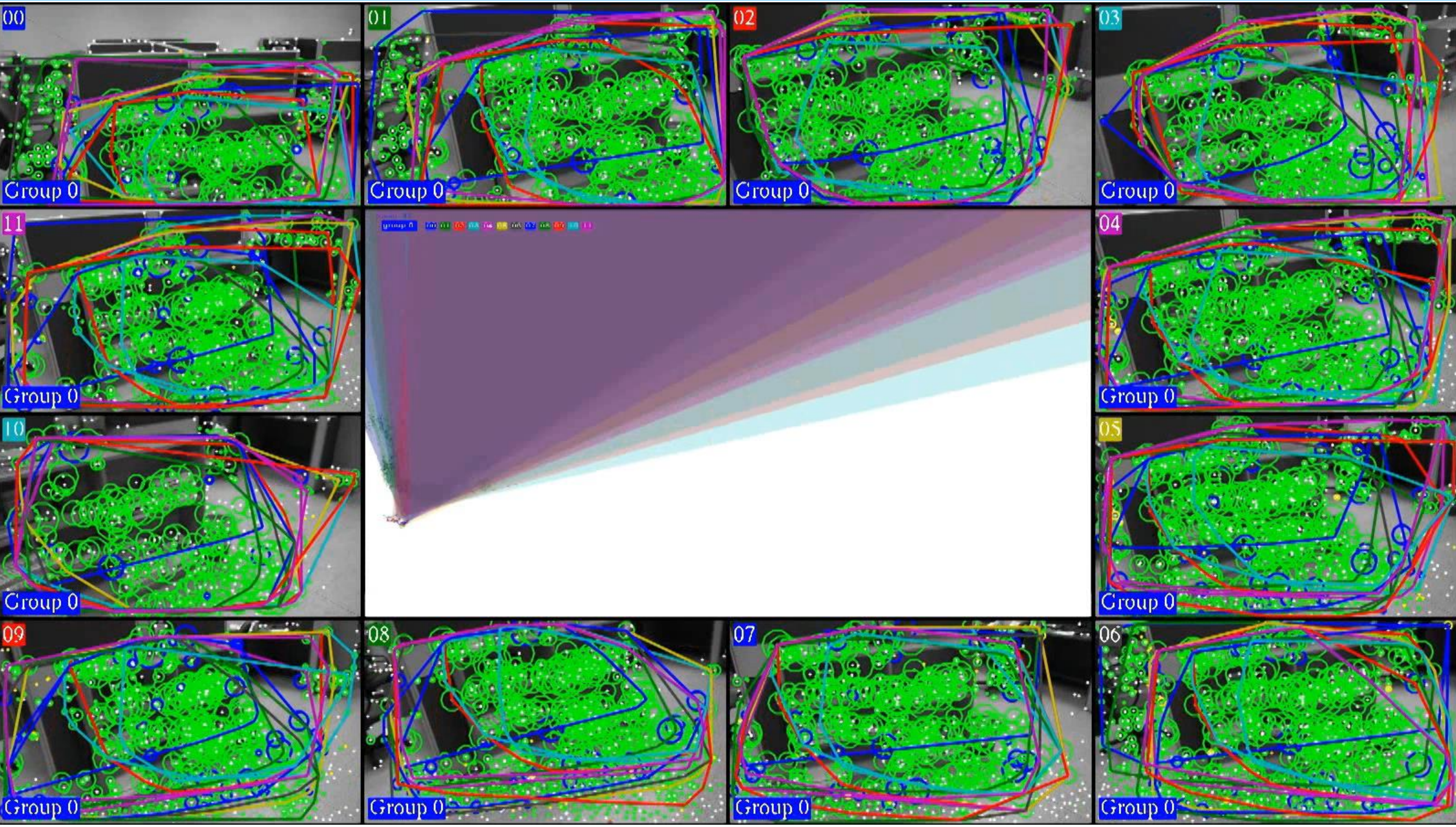
Collaborative localization & mapping

- Inter/intra-camera pose estimation
- Inter/intra-camera mapping
- Identification of moving points
- Spanning tree for dividing camera into groups
- Collaboration in group

CoSLAM



CoSLAM – 12 cameras in a room

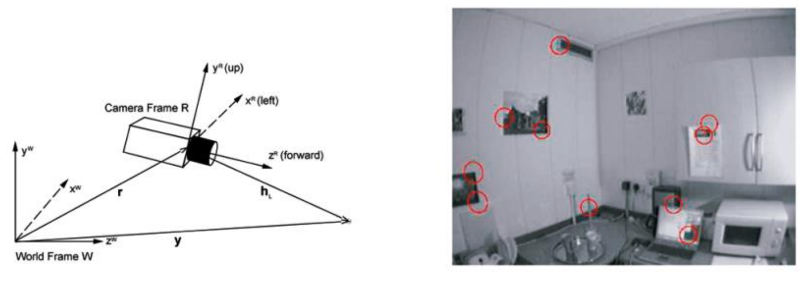


Outline

- Basic Theory
  - Pinhole camera model
  - Camera calibration
  - Two camera geometry

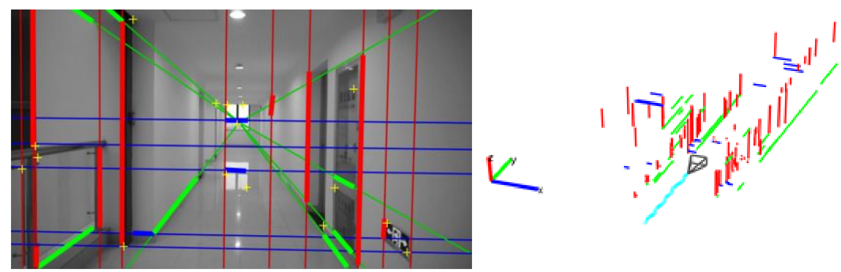
- Build a visual SLAM system
  - Structure-from-motion(SFM) approach:
    - **PTAM**
    - **ORB SLAM**
    - **CoSLAM**
  - Extended Kalman Filter approach:
    - **MonoSLAM**
    - **StructSLAM**

Extended Kalman Filter approach



MonoSLAM,2003

Davison, Andrew J., Ian D. Reid, Nicholas D. Molton, and Olivier Stasse. "MonoSLAM: Real-time single camera SLAM." IEEE transactions on pattern analysis and machine intelligence 29, no. 6 (2007): 1052-1067.



StructSLAM, 2015

Zhou, Huizhong, Danping Zou, Ling Pei, Rendong Ying, Peilin Liu, and Wenxian Yu. "StructSLAM: Visual SLAM With Building Structure Lines."Vehicular Technology, IEEE Transactions on 64, no. 4 (2015): 1364-1375.

Kalman filter

- What is Kalman filter?
  - A Kalman filter is an **estimator** – i.e. infers parameters of interest from indirect, inaccurate and uncertain observations
  - It is **recursive** so that new measurements can be processed as they arrive.
  - It is **optimal** – i.e. if the noise is Gaussian, Kalman filter minimizes the mean square error of the estimated parameters.

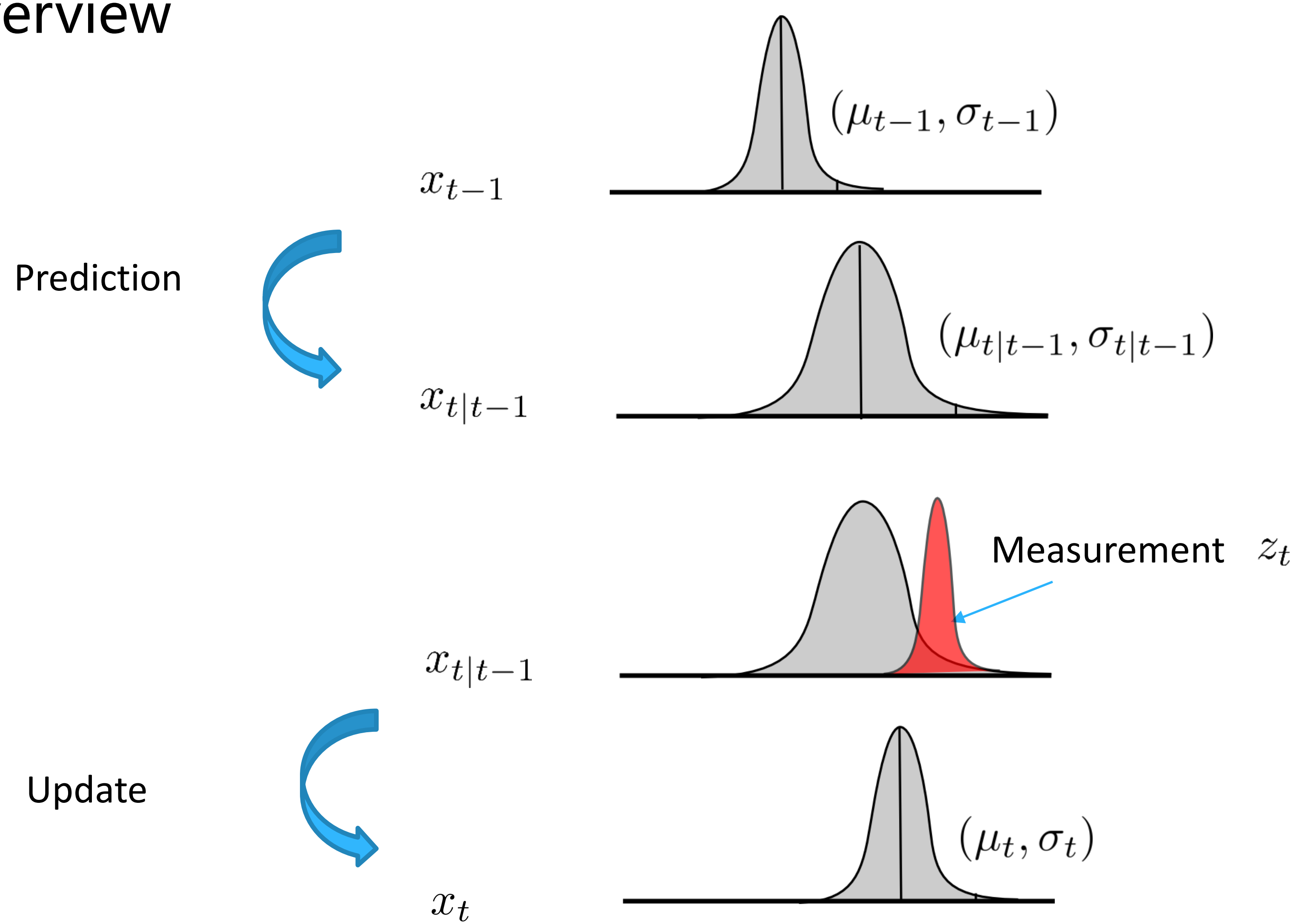
Rudolf Emil Kálmán, co-inventor and developer of the Kalman filter.

Kalman filter

- Why is Kalman filter so popular?
  - Good results in practice due to optimality and structure.
  - Convenient form for online real time processing.
  - Easy to formulate and implement given a basic understanding.
  - Measurement equations need not be inverted.

Kalman filter

- Overview



Kalman filter

- Linear dynamic model

$$x_t = F_t x_{t-1} + B_t u_t + w_t$$

- F_t is the state transition model which is applied to the previous state x_{t-1};
- B_t is the control-input model which is applied to the control vector u_t;
- w_t is the process noise which is assumed to be drawn from a zero mean multivariate normal distribution with covariance Q_t.

$$w_t \sim \mathcal{N}(0, Q_t)$$

Kalman filter

- Observation model (measurement model)

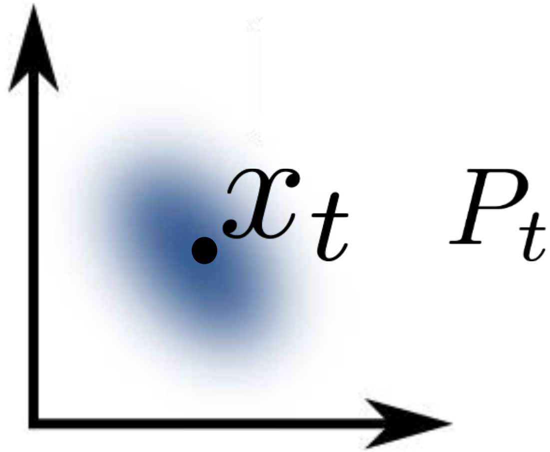
$$z_t = H_t x_t + n_t$$

  - H_t is the observation model which maps the true state space into the observed space.
  - n_t is the observation noise which is assumed to be zero mean Gaussian white noise with covariance R_t

$$n_t \sim \mathcal{N}(0, R_t)$$

Kalman filter

- At each time step, Kalman filter try to compute both the state estimation and the state covariance



Kalman filter

- Prediction
  - State prediction - use the dynamic model to predict the state in the next time step:

$$x_{t|t-1} = F_t x_{t-1} + B_t u_t$$

  - Uncertainty of prediction – propagate the covariance

$$P_{t|t-1} = F_t P_{t-1} F_t^T + Q_t$$

Kalman filter

- Correction/Update
  - Compute innovation (measurement residual)

$$y_t = z_t - H_t x_{t|t-1}$$

  - Get innovation covariance

$$S_t = H_t P_{t|t-1} H_t^T + R_t$$

Kalman filter

- Correction/Update
  - Compute Kalman Gain

$$K_t = P_{t|t-1} H_t^T S_t^{-1}$$

  - Update state estimate

$$x_t = x_{t|t-1} + K_t y_t$$

  - Update state covariance

$$P_t = (I - K_t H_t) P_{t|t-1}$$

Extended Kalman filter

- Nonlinear dynamic model
- Nonlinear observation model

$$x_t = f(x_{t-1}, u_t) + w_t$$
$$z_t = h(x_t) + v_t$$

Prediction

$$x_{t|t-1} = f(x_{t-1}, u_t)$$

$$P_{t|t-1} = F_t P_{t-1} F_t^T + Q_t$$

$$F_t = \frac{\partial f}{\partial x}\Big|_{x_t, u_t}$$

Observation

$$y_t = z_t - h(x_{t|t-1})$$

$$S_t = H_t P_{t|t-1} H_t^T + R_t$$

$$H_t = \frac{\partial h}{\partial x}\Big|_{x_t}$$

Extended Kalman Filter approach

- EKF VSLAM tries to estimate *a state variable* that contains current robot state(orientation, position, velocity) and all landmarks .

$$\mathbf{x} = \begin{bmatrix} \mathcal{R} \\ \mathcal{M} \end{bmatrix} = \begin{bmatrix} \mathcal{R} \\ \mathcal{L}_1 \\ \vdots \\ \mathcal{L}_n \end{bmatrix}$$

Robot state

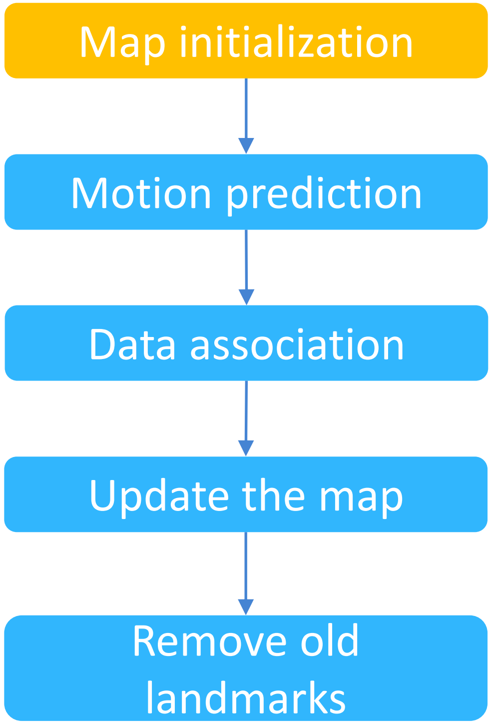
Landmarks (points, lines)

This is also called ***map***

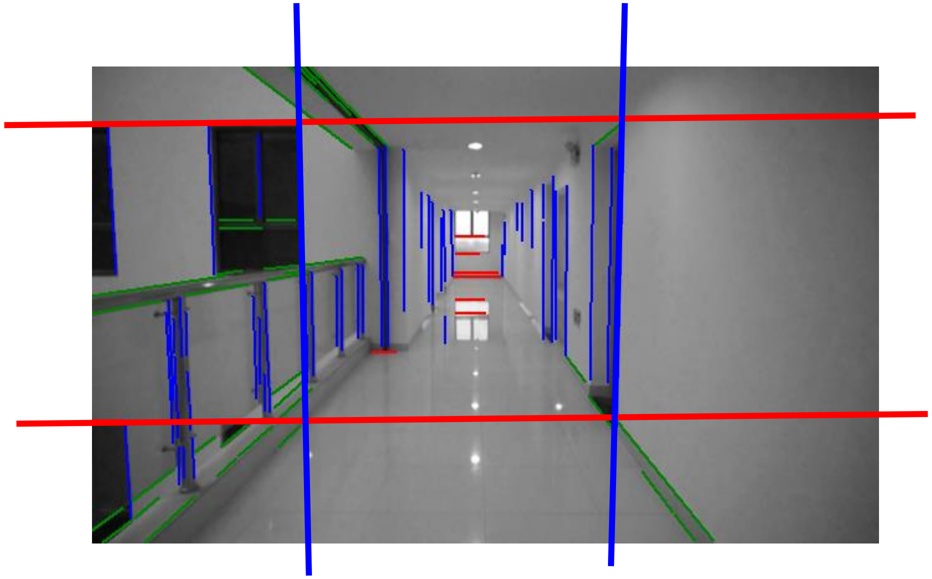
$$\mathbf{P} = \begin{bmatrix} \mathbf{P}_{\mathcal{R}\mathcal{R}} & \mathbf{P}_{\mathcal{R}\mathcal{M}} \\ \mathbf{P}_{\mathcal{M}\mathcal{R}} & \mathbf{P}_{\mathcal{M}\mathcal{M}} \end{bmatrix} = \begin{bmatrix} \mathbf{P}_{\mathcal{R}\mathcal{R}} & \mathbf{P}_{\mathcal{R}\mathcal{L}_1} & \cdots & \mathbf{P}_{\mathcal{R}\mathcal{L}_n} \\ \mathbf{P}_{\mathcal{L}_1\mathcal{R}} & \mathbf{P}_{\mathcal{L}_1\mathcal{L}_1} & \cdots & \mathbf{P}_{\mathcal{L}_1\mathcal{L}_n} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{P}_{\mathcal{L}_n\mathcal{R}} & \mathbf{P}_{\mathcal{L}_n\mathcal{L}_1} & \cdots & \mathbf{P}_{\mathcal{L}_n\mathcal{L}_n} \end{bmatrix}$$

Extended Kalman Filter approach

- The workflow of a typical EKF vSLAM system.



- Initialize the state variable and the covariance



Extended Kalman Filter approach

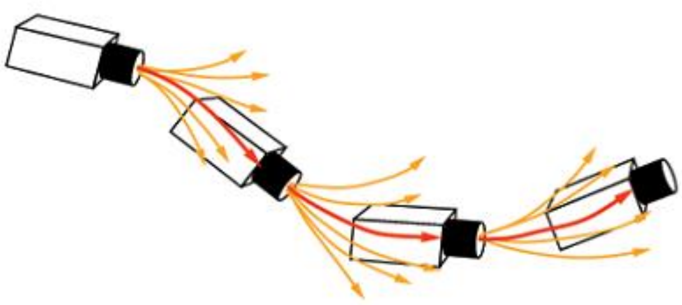
- The workflow of a typical EKF vSLAM system.

Map initialization → Motion prediction → Data association → Update the map → Remove old landmarks

- Predict the state and propagate the covariance

$$\bar{\mathbf{x}} \leftarrow f(\bar{\mathbf{x}}, \mathbf{u}, 0)$$

$$\mathbf{P} \leftarrow \mathbf{F_x} \mathbf{P} \mathbf{F_x}^\top + \mathbf{F_n} \mathbf{N} \mathbf{F_n}^\top$$

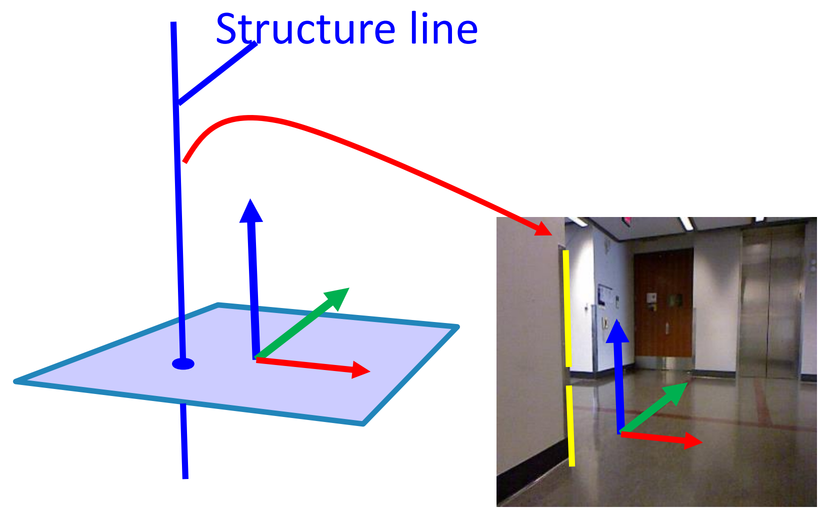


Extended Kalman Filter approach

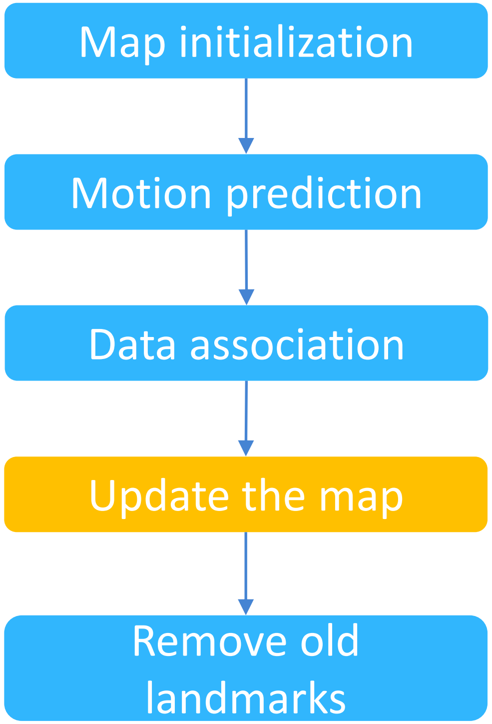
- The workflow of a typical EKF vSLAM system.

Map initialization
↓
Motion prediction
↓
Data association
↓
Update the map
↓
Remove old landmarks

- Find the corresponding features of the landmarks in the image



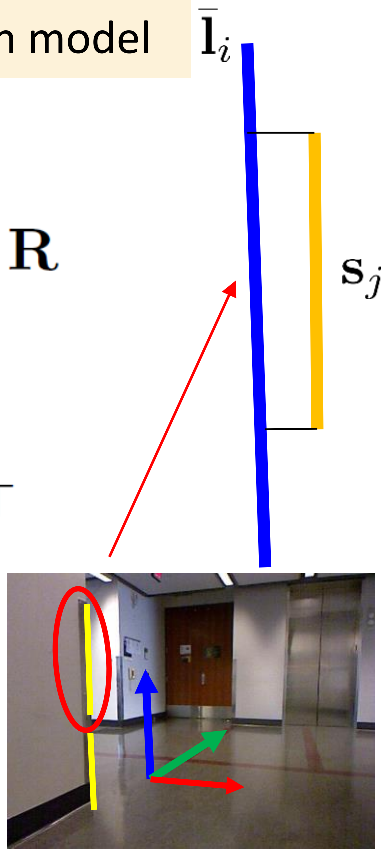
Extended Kalman Filter approach



- Compute Kalman Gain according to the observation model and use it to update the state.

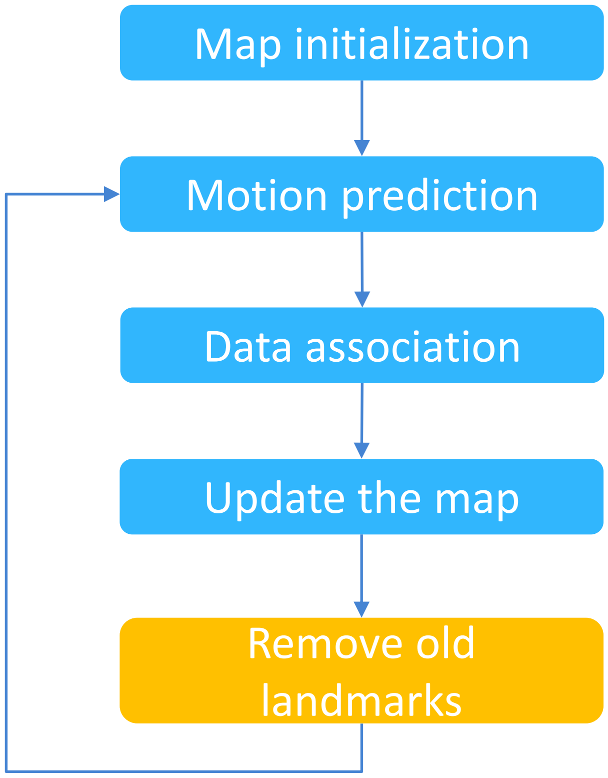
$h(\bar{\mathbf{x}})$ is the observation model

$$
\begin{aligned}
\bar{\mathbf{z}} &= \mathbf{y} - \boxed{h(\bar{\mathbf{x}})} \\
\mathbf{Z} &= \mathbf{H}_{\mathbf{x}}\mathbf{P}\mathbf{H}_{\mathbf{x}}^{\top} + \mathbf{R} \\
\mathbf{K} &= \mathbf{P}\mathbf{H}_{\mathbf{x}}^{\top}\mathbf{Z}^{-1} \\
\bar{\mathbf{x}} &\leftarrow \bar{\mathbf{x}} + \mathbf{K}\bar{\mathbf{z}} \\
\mathbf{P} &\leftarrow \mathbf{P} - \mathbf{K}\mathbf{Z}\mathbf{K}^{\top}
\end{aligned}
$$



Extended Kalman Filter approach

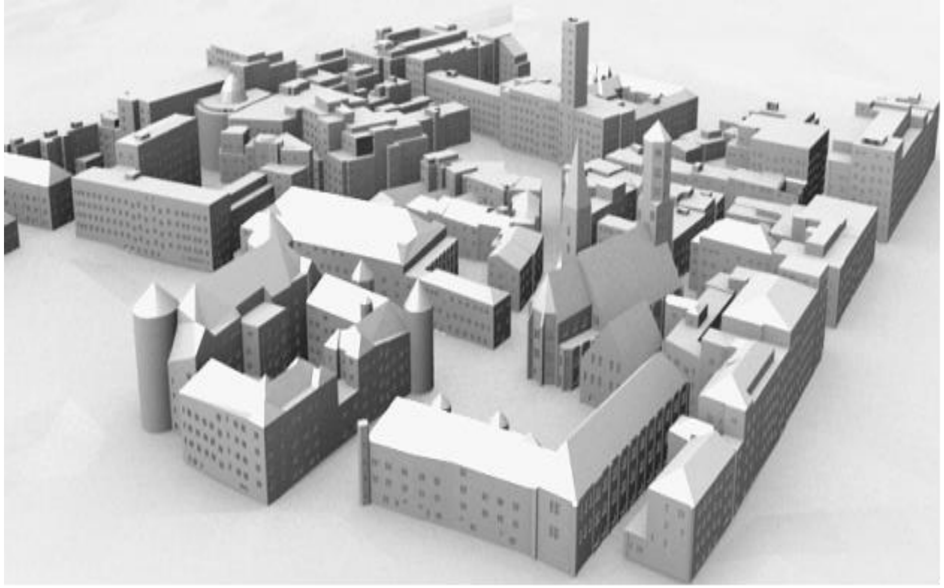
- The workflow of a typical EKF vSLAM system.



- To limit the dimension of the map without growing to a very large value.

StructSLAM

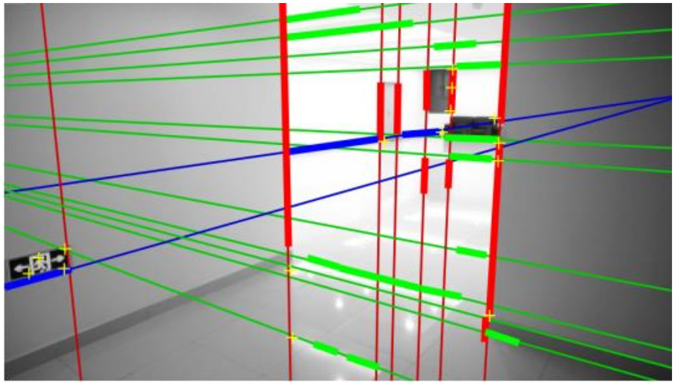
- Basic idea: Most man-made scenes exhibit strong regularity in structures, especially the indoor spaces.
- This regularity can be simply described as 'Manhattan world'.



- Perpendicular surfaces
- Have several dominant directions

StructSLAM

- A new kind of line features named as *Structure Lines*



- **Structure Lines here are those lines who are aligned with x,y,z axes.**

- Motivations:
  - Structure lines encode the **global orientation information** in the image
  - Lines are better landmarks in texture-less scenes (like many indoor scenes with only white walls) than points.

StructSLAM:
Visual SLAM with Building Structure Lines

Hui Zhong Zhou, Danping Zou et al.

Shanghai Key Laboratory of Navigation and Location Based Services
Shanghai Jiao Tong University
Apirl,2014


Q&A
You have
Questions
We have
Answers