Visual SLAM
Course Slides

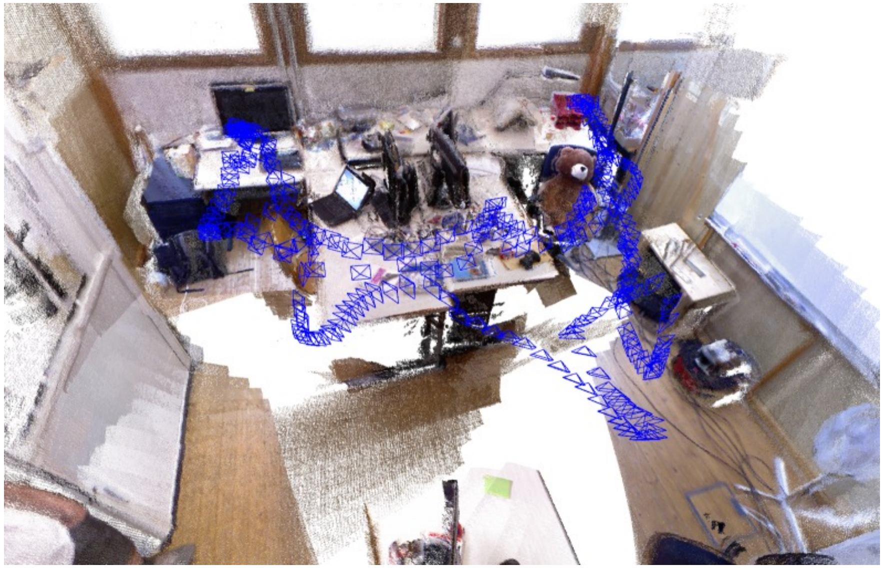
Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB

With contributions from:
Raúl Mur, Carlos Campos, Richard Elvira,
Juan J. Gómez Rodríguez and José M. M. Montiel

Content

Lesson 4. Visual SLAM

Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



Lesson 4. Visual SLAM

1. Introduction to Visual SLAM
2. Tracking
3. Mapping
4. Place Recognition
5. Accuracy

Readings

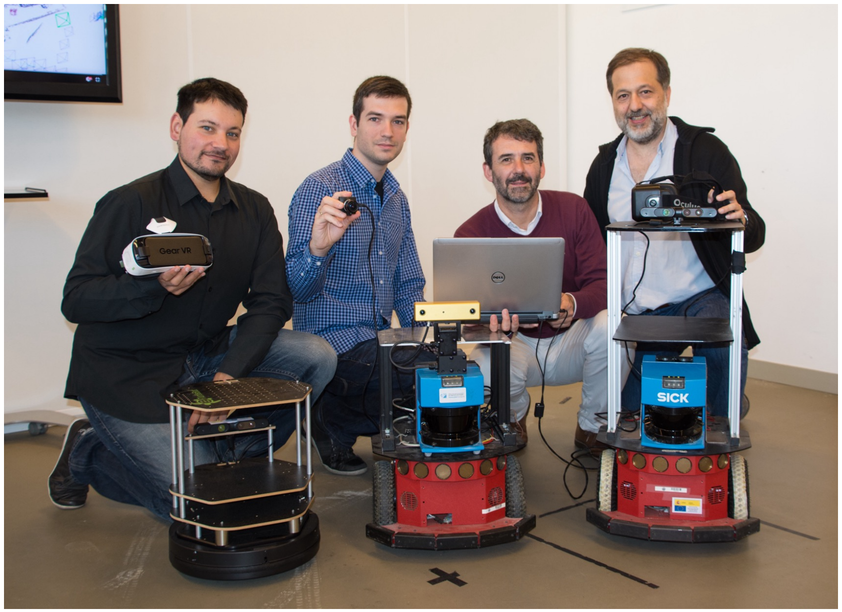
- Raúl Mur-Artal, J.M.M. Montiel and Juan D. Tardós
ORB-SLAM: A Versatile and Accurate **Monocular SLAM** System,
IEEE Trans. Robotics 31(5): 1147-1163, Oct. 2015

- Raúl Mur-Artal, and Juan D. Tardós.
ORB-SLAM2: an Open-Source SLAM System for Monocular, **Stereo** and **RGB-D** Cameras,
IEEE Trans. Robotics 33(5): 1255-1262, Oct. 2017

- C Campos, R Elvira, JJ Gómez Rodríguez, JMM Montiel, JD Tardós
ORB-SLAM3: An Accurate Open-Source Library for Visual, **Visual-Inertial and Multi-Map SLAM**.
IEEE Trans. Robotics, 37(6): 1874-1890, Dec. 2021

- D. Gálvez-López, J.D. Tardós
Bags of Binary Words for Fast **Place Recognition** in Image Sequences, IEEE Trans. Robotics 28(5):1188-1197, Oct. 2012

ORB-SLAM Team



Dorian Gálvez | Raúl Mur | José M.M. Montiel | Mingo Tardós

Richard Elvira | Carlos Campos | Juanjo Gómez

Lesson 4. Visual SLAM

1. Introduction to Visual SLAM
    a. Concept and Applications
    b. Feature-Based Visual SLAM
    c. A Complete System: ORB-SLAM3

Readings:

- Sections I, II & III of ORB-SLAM3 paper

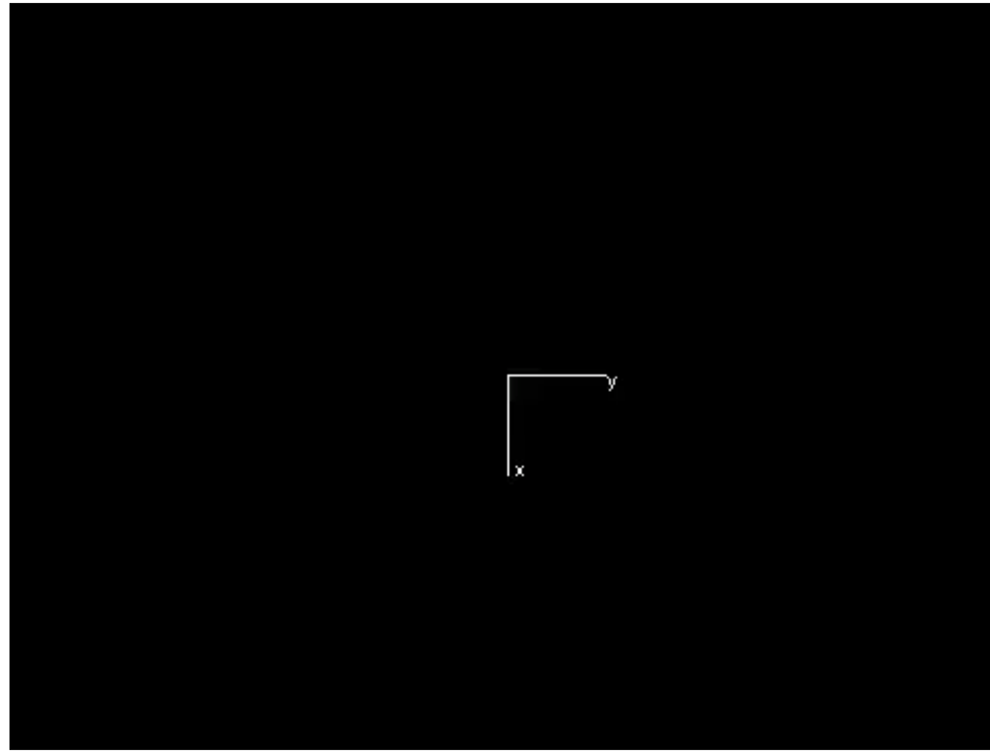
SLAM with Laser and EKF, 2002

The SLAM problem:

- a robot moving in an unknown environment

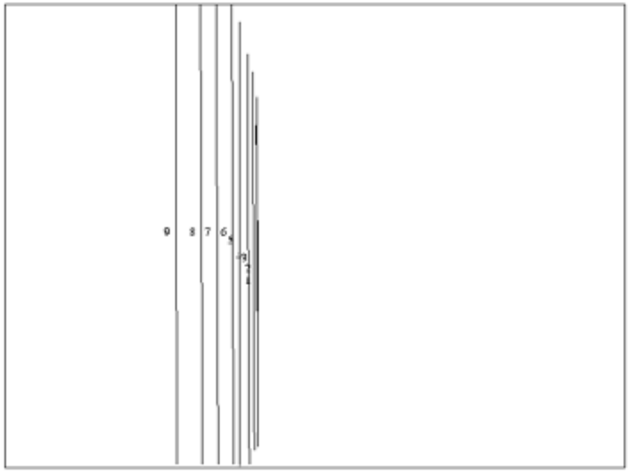
Use sensor data to:

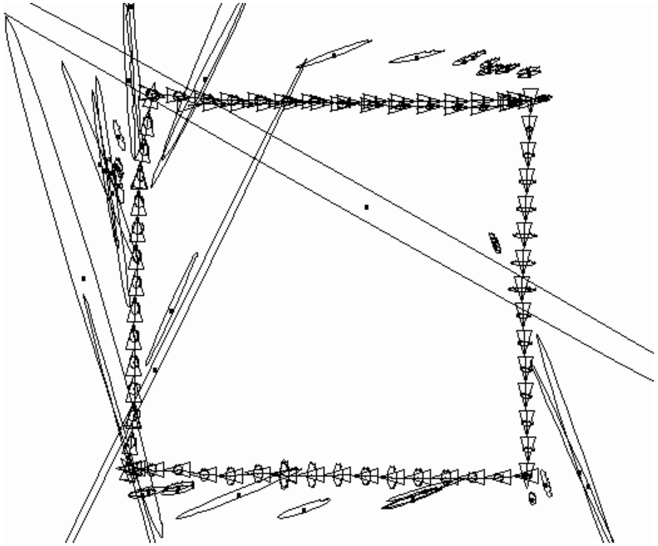
- **build a map** of the environment
- and **at the same time**
- use the map to compute the **robot location**



P. Newman, J.J Leonard, J.D. Tardos, J. Neira: **Explore and return: Experimental validation of real-time concurrent mapping and localization.** IEEE Int. Conf. Robotics and Automation, 2002

First Monocular SLAM: Lines and EKF, 1997



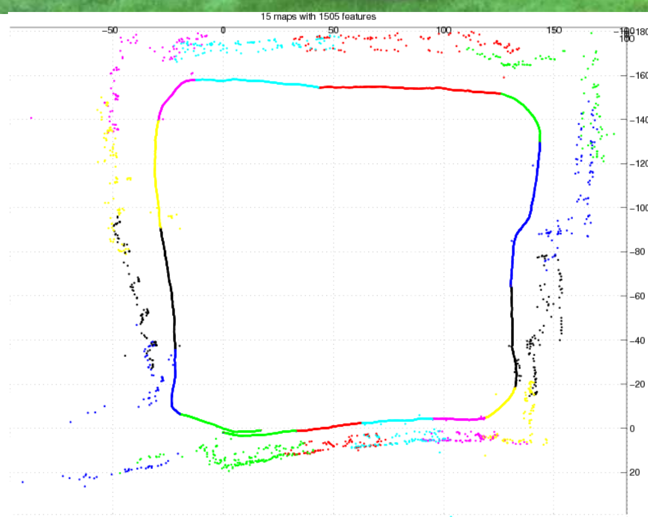


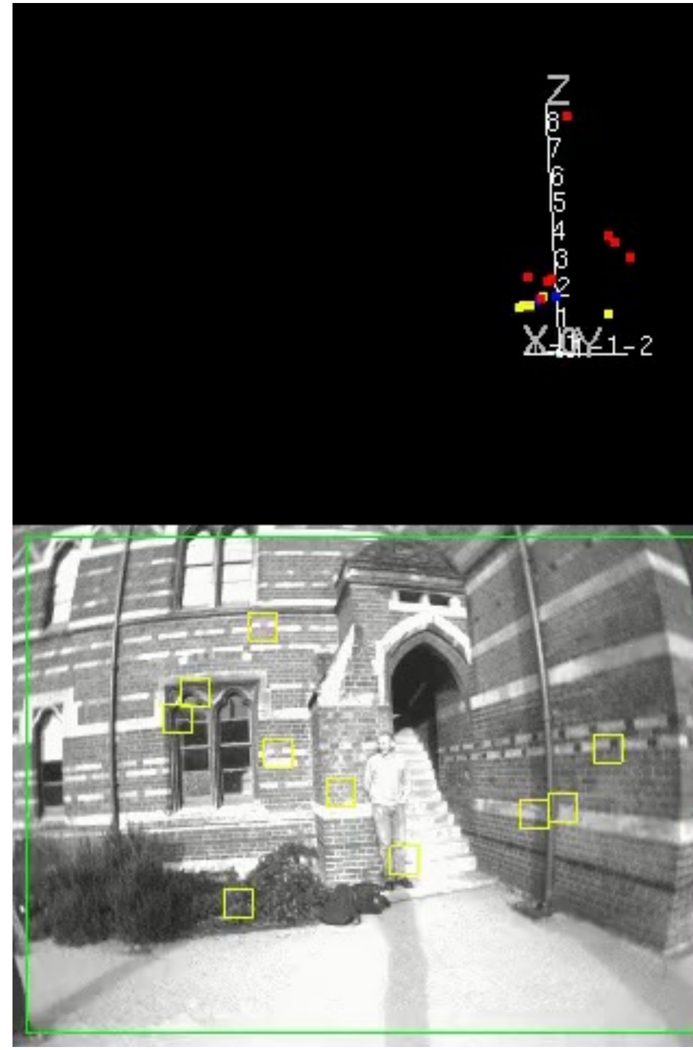
J. Neira, M.I. Ribeiro, J.D. Tardós,
Mobile Robot Localization and Map Building using Monocular Vision,
Symp. Intell. Robotics Systems, 1997

First Stereo SLAM with EKF, 2001

A.J. Davison, N. Kita: **3D Simultaneous Localisation and Map-Building using Active Vision for a Robot Moving on Undulating Terrain**, CVPR 2001.

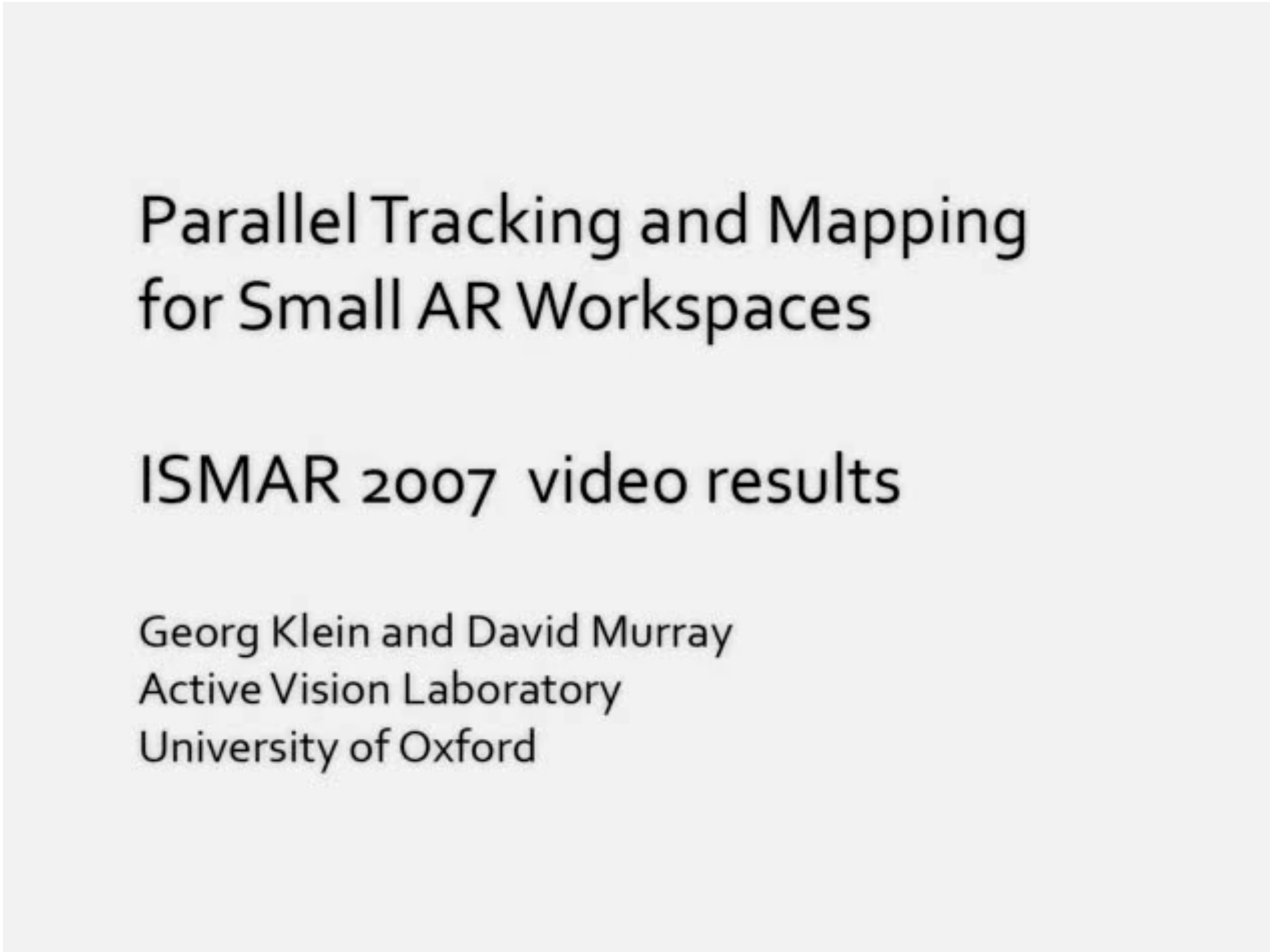
Monocular SLAM with Multi-Map EKF, 2007





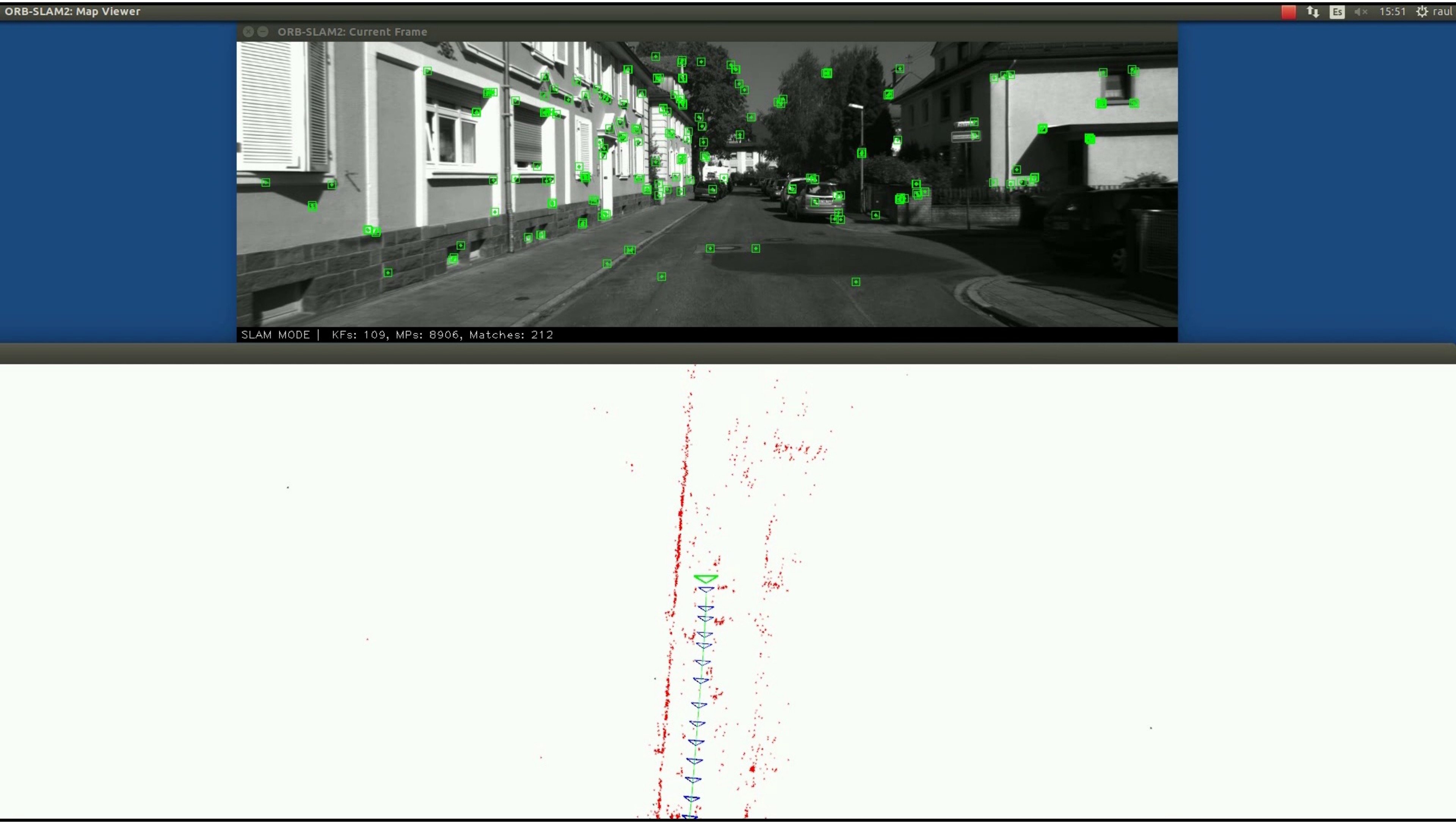
L.A. Clemente, A.J. Davison, I.D. Reid, J. Neira, J.D. Tardós, **Mapping Large Loops with a Single Hand-Held Camera,** RSS 2007

PTAM: Keyframe-Based SLAM, 2007



G. Klein and D. Murray, **Parallel Tracking and Mapping for Small AR Workspaces**, ISMAR 2007

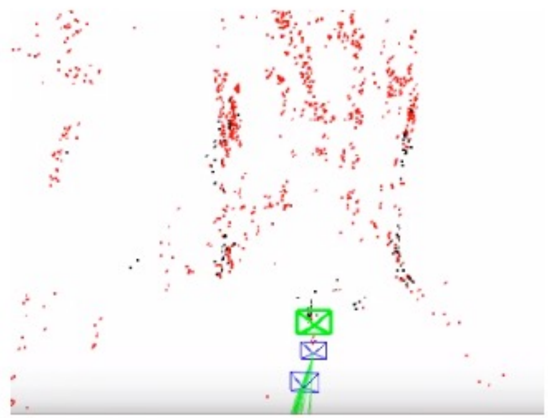
ORB-SLAM: Visual SLAM, 2015

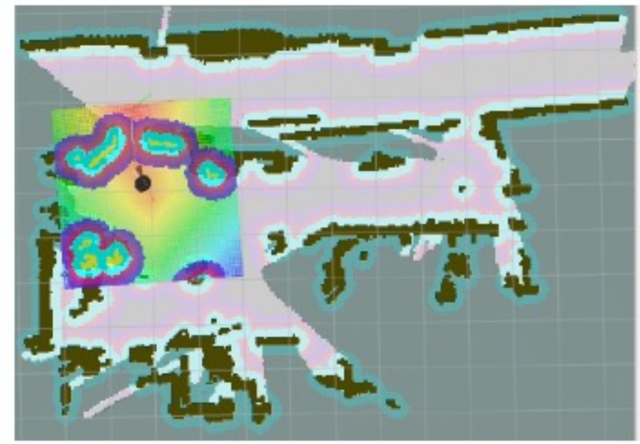


Applications: Autonomous Vehicles

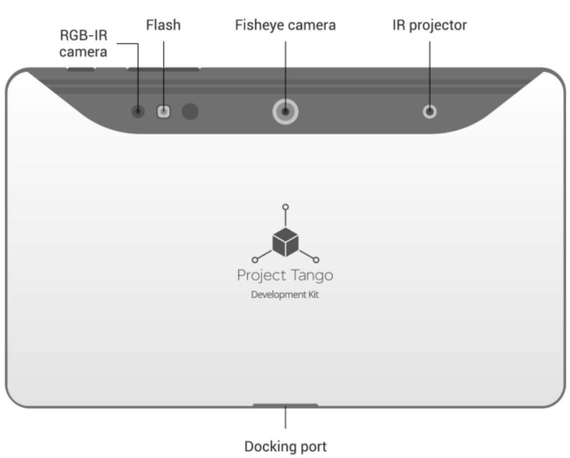
Applications: Robotics and 3D Modelling

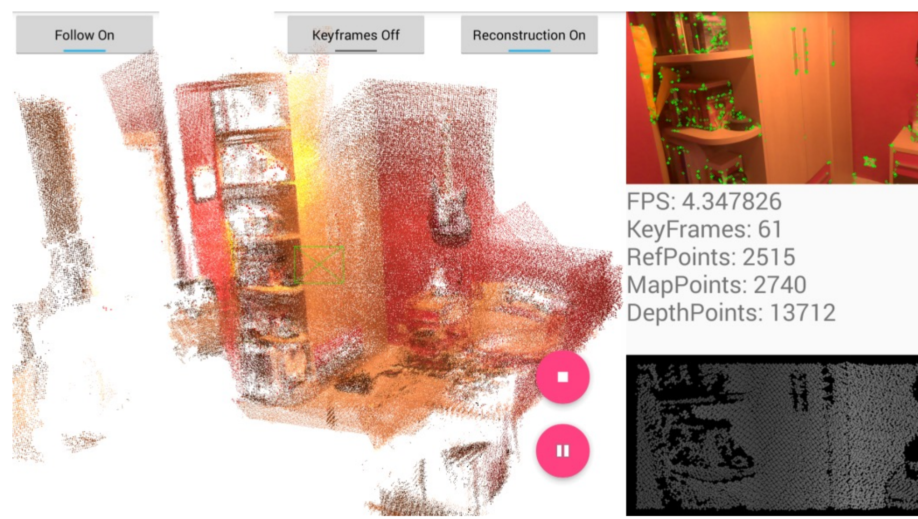
- Robot Navigation based on ORB-SLAM2





- ORB-SLAM2 on mobile devices





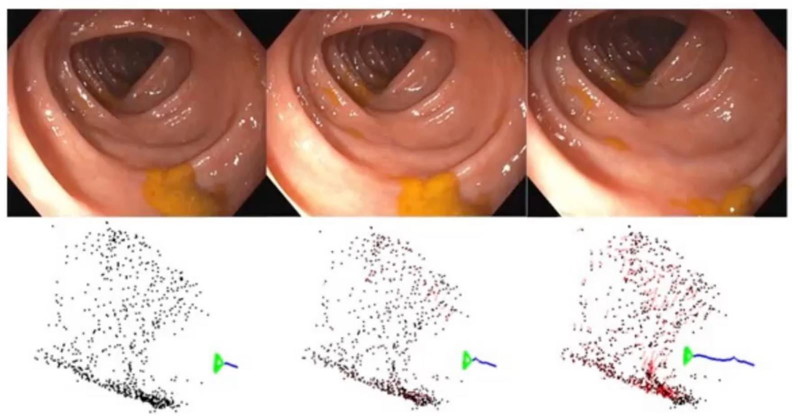
Applications: AR/VR

- Obtain in real time the camera trajectory
- And build a map of the environment
- To add virtual elements to the environment

Deformable SLAM Inside the Human Body

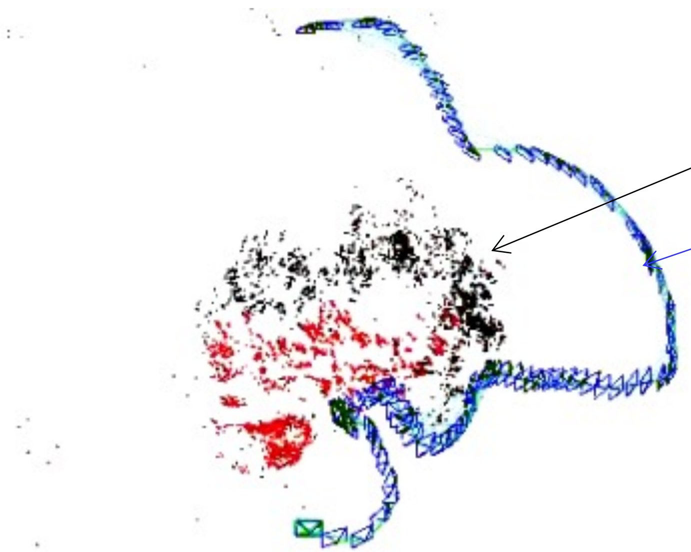
Tracking monocular camera pose and deformation for SLAM inside the human body

Juan J. Gómez Rodríguez, J.M.M. Montiel, Member, IEEE and Juan D. Tardós, Fellow, IEEE



Universidad Zaragoza 1542

4.1.b Feature-Based Visual SLAM



States

$$ {}^{w}\mathbf{x}_j = \mathbf{x}_{wj} \in \mathbb{R}^3 $$ Coordinates of point j

$$ \mathbf{T}_{iw} \in \mathrm{SE}(3) $$ Pose of camera i

Measurements

$$ \mathbf{u}_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix} $$ Observation of point j from camera i

Reprojection error

Projection Function

$$ \mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj}) $$

Projection of point j on camera i (1)

$$\mathbf{T}_{iw} \in \mathrm{SE}(3) \begin{cases} \mathbf{R}_{iw} \in \mathrm{SO}(3) \\ \mathbf{t}_{iw} \in \mathbb{R}^3 \end{cases}$$

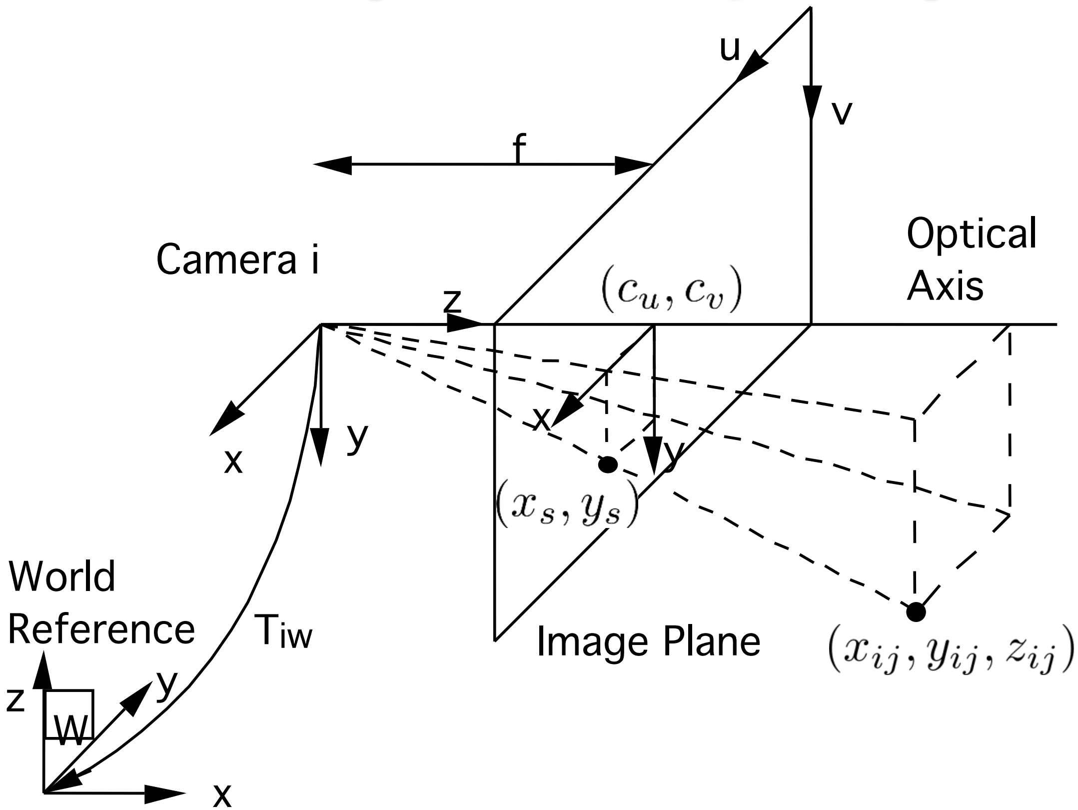
Rotation matrix

Translation vector

$$\mathbf{x}_{ij} = \mathbf{R}_{iw}\mathbf{x}_{wj} + \mathbf{t}_{iw}$$

Coordinates of point *j* w.r.t. camera *i*

Projection of point j on camera i (2)



focal length (mm)

$$x_s = f_i \frac{x_{ij}}{z_{ij}}$$

$$u = (s_u f_i) \frac{x_{ij}}{z_{ij}} + c_{i,u}$$

$$= f_{i,u} \frac{x_{ij}}{z_{ij}} + c_{i,u}$$

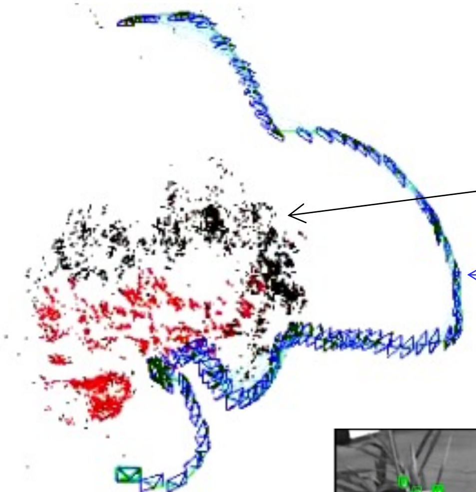
horizontal focal length (pixels)

principal point

- In summary:

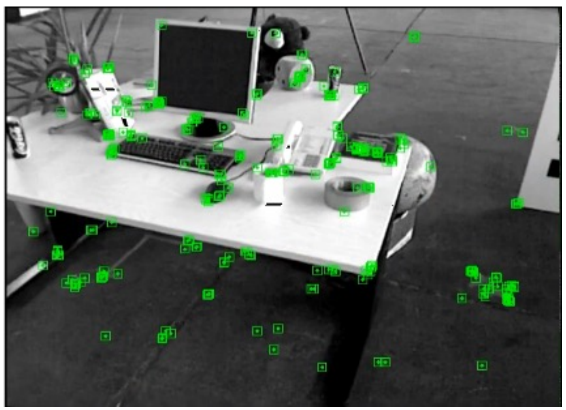
$$\pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj}) = \begin{bmatrix} f_{i,u} \frac{x_{ij}}{z_{ij}} + c_{i,u} \\ f_{i,v} \frac{y_{ij}}{z_{ij}} + c_{i,v} \end{bmatrix}$$

Feature-Based Visual SLAM as BA



States

$\mathbf{x}_{wj} \in \mathbb{R}^3$	Coordinates of point j
$\mathbf{R}_{iw} \in SO(3)$	Orientation of camera i
$\mathbf{t}_{iw} \in \mathbb{R}^3$	Position of camera i



Measurements

$$\mathbf{u}_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix}$$

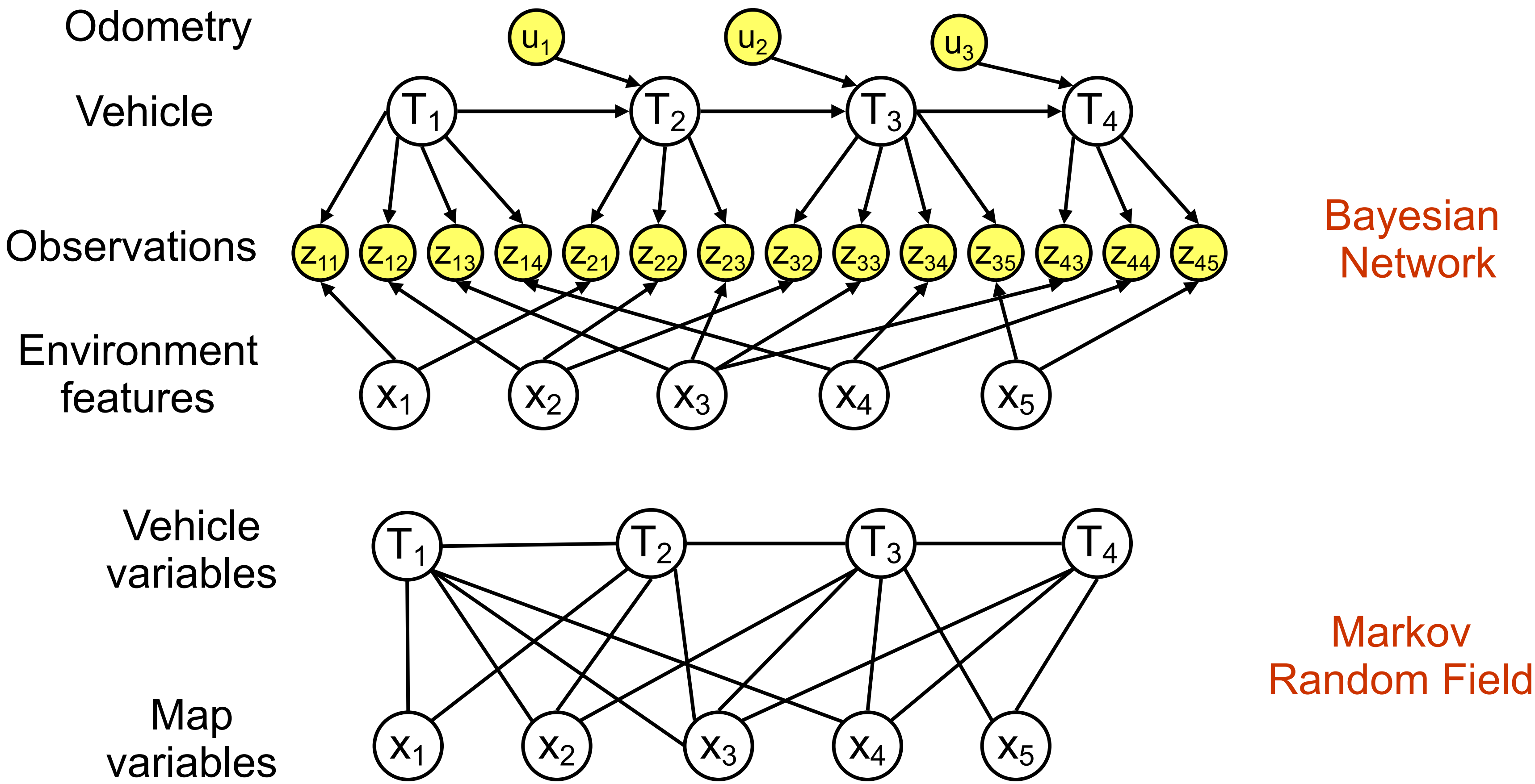
Observation of point *j* from camera *i*

Bundle Adjustment

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{C}, j \in \mathcal{P}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\underbrace{\| \mathbf{u}_{ij} - \pi_i \left(\mathbf{R}_{iw} \mathbf{x}_{wj} + \mathbf{t}_{iw} \right) \|^2_{\Sigma_{ij}}}_{\text{Reprojection error}} \right)$$

Non-Linear Optimization

Structure of the SLAM problem



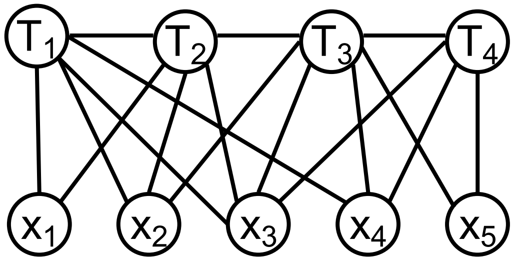
SLAM Problem

$$p(T_{1:k}, x_{1:n} \mid z_{1:k}, u_{1:k})$$

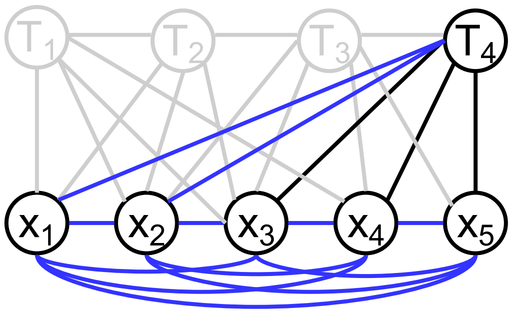
- The problem size grows with time
- The set of relationships is sparse

Maps with Thousands of Features?

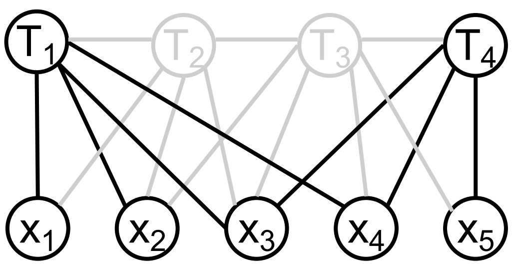
Sparse Problem



Dense Problem



Still Sparse



- Original SLAM problem
- EKF approach (MonoSLAM, 2003)
  - Marginalizes past poses
  - $O(n^2)$ with the number of features
  - Limited to 200-300 features in real-time
  - One-shot linearization → errors
- Keyframe approach (PTAM, 2007)
  - Keeps only a few poses: **keyframes**
  - Can handle thousands of points
  - Given the same computational effort is more accurate than EKF-SLAM
  - Non-linear optimization (BA)

Hauke Strasdat, J. M. M. Montiel, Andrew J. Davison, **Real-time Monocular SLAM: Why Filter?.** IEEE Int. Conf. Robotics and Automation, ICRA 2010.

Bundle Adjustment in Real Time?

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj} \,| i \in \mathcal{C}, j \in \mathcal{P}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\left\| \mathbf{u}_{ij} - \pi_i \left(\mathbf{R}_{iw} \mathbf{x}_{wj} + \mathbf{t}_{iw} \right) \right\|^2_{\Sigma_{ij}} \right)$$

- The problem is sparse
  - Not all cameras see all points!
- But still not feasible in real time
  - example: 500 Keyframes and 15k points → 6s
- Local BA or sliding-window BA
  - example: 30 Keyframes and 3k points → 200ms

- BA requires very good initial solutions to converge

Classical .vs. Keyframe-Based Visual SLAM

Classical Visual SLAM

- EKF
- Vehicle state: last pose
- Accumulates linearization errors
- Computes covariance (dense matrix)
- Matching using Mahalanobis distance
- Scaling: Local maps

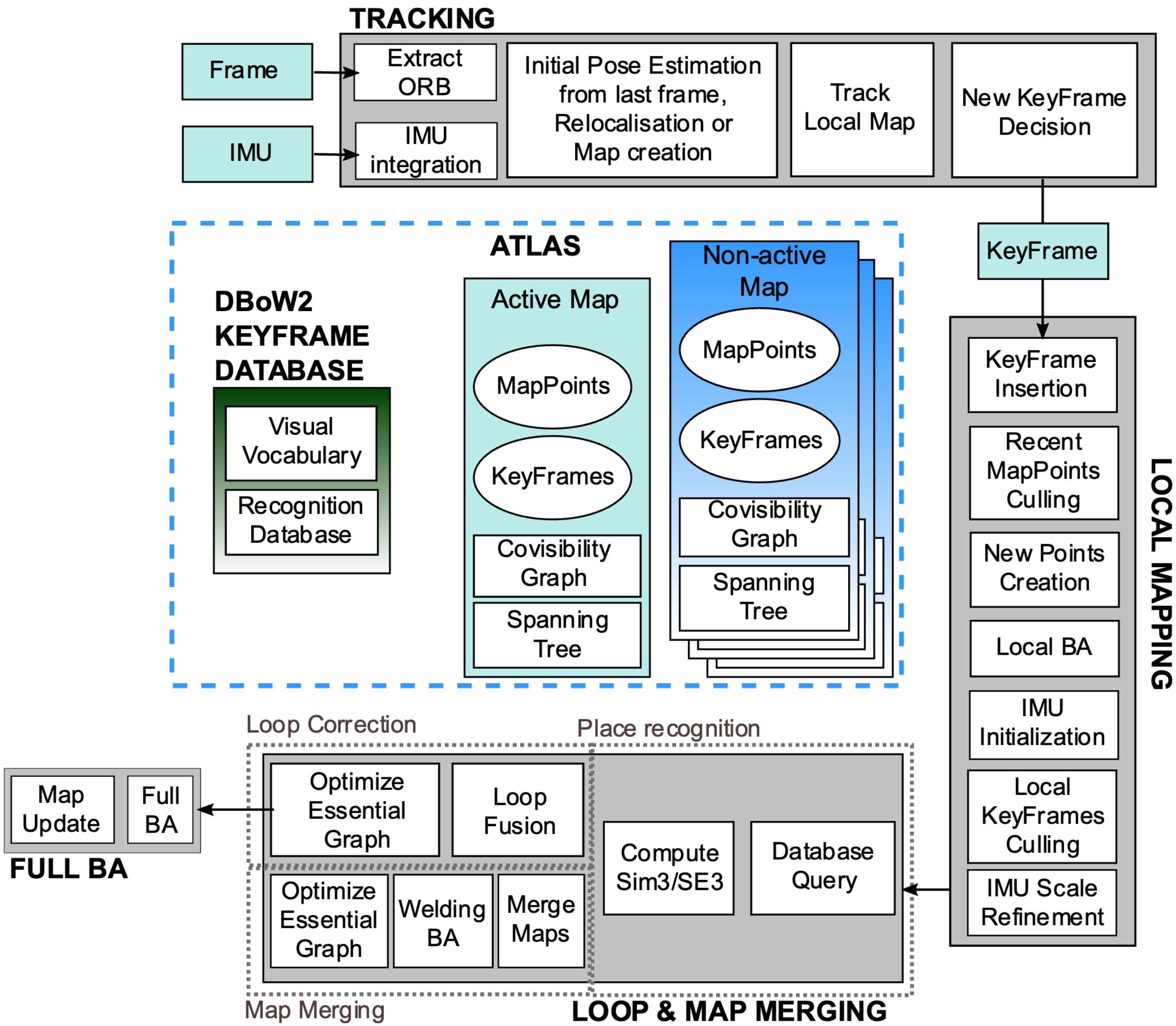
Keyframe Visual SLAM

- Non-linear Optimiz. (BA)
- Vehicle state: all poses
- Re-linearizes observations around current estimation
- Avoids covariance (maintains sparsity)
- Matching using visual appearance
- Scaling: Local BA

BA + Keyframes, what else do I need?

- Which features will I use?
- How to match them?
- How to start when the map is empty?
- How to track the camera pose?
- How to add new points to the map?
- How to make it run in real time?
  - Which information to keep, what to throw away?
- What if objects or people move?
- What if I get lost?
- How to detect a loop?
- How to correct drift after a loop?

4.1.c A Complete System: ORB-SLAM3



ORB-SLAM3 Highlights

- Visual and Visual-Inertial SLAM
- Pin-hole and Fish-eye lens models
- Multi-Map and Multi-Session
- Real-time operation in large environments
- Data association with ORB features:
  - Short term DA: match with previous images   Visual Odometry
  - Mid-term DA: match with local map
  - Long term DA: Relocation, Loop Closing & Map Merging

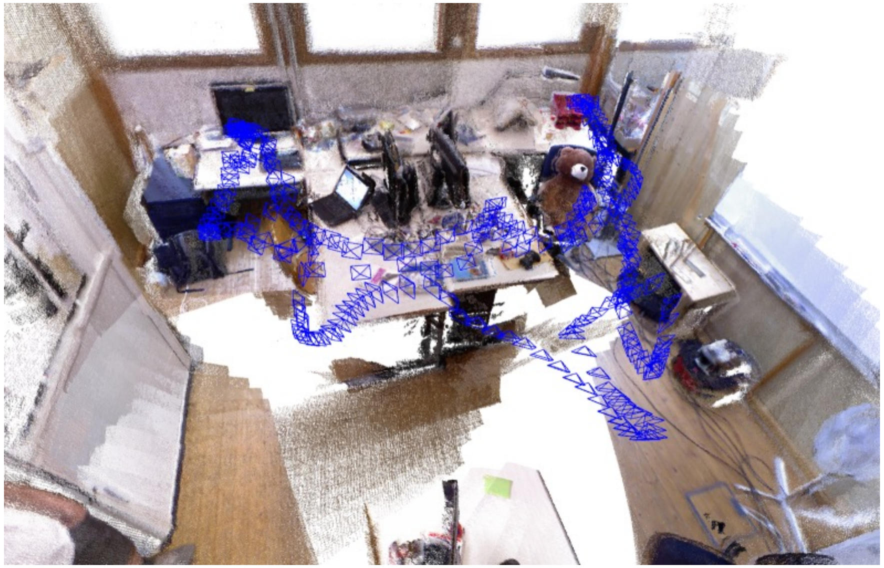
C Campos, R Elvira, JJ Gómez Rodríguez, JMM Montiel, JD Tardós
ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM.
IEEE Trans. Robotics, 37(6): 1874-1890, Dec. 2021

Visual Odometry and Visual SLAM Systems

	SLAM or VO	Pixels used	Data association	Estimation	Relocation	Loop closing	Multi Maps	Mono	Stereo	Mono IMU	Stereo IMU	Fisheye	Accuracy	Robustness	Open source
Mono-SLAM [13], [14]	SLAM	Shi Tomasi	Correlation	EKF	-	-	-	✓	-	-	-	-	Fair	Fair	[15][1]
PTAM [16]–[18]	SLAM	FAST	Pyramid SSD	BA	Thumbnail	-	-	✓	-	-	-	-	Very Good	Fair	[19]
LSD-SLAM [20], [21]	SLAM	Edgelets	Direct	PG	-	FABMAP PG	-	✓	✓	-	-	-	Good	Fair	[22]
SVO [23], [24]	VO	FAST+ Hi.grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Very Good	Very Good	[25][2]
ORB-SLAM2 [2], [3]	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	-	✓	✓	-	-	-	Exc.	Very Good	[26]
DSO [27]–[29]	VO	High grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Fair	Very Good	[30]
DSM [31]	SLAM	High grad.	Direct	Local BA	-	-	-	✓	-	-	-	-	Very Good	Very Good	[32]
MSCKF [33]–[36]	VO	Shi Tomasi	Cross correlation	EKF	-	-	-	✓	-	✓	✓	-	Fair	Very Good	[37][3]
OKVIS [38], [39]	VO	BRISK	Descriptor	Local BA	-	-	-	-	-	✓	✓	✓	Good	Very Good	[40]
ROVIO [41], [42]	VO	Shi Tomasi	Direct	EKF	-	-	-	-	-	✓	✓	✓	Good	Very Good	[43]
ORBSLAM-VI [4]	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	-	✓	-	✓	-	-	Very Good	Very Good	-
VINS-Fusion [7], [44]	VO	Shi Tomasi	KLT	Local BA	DBoW2	DBoW2 PG	✓	-	✓	✓	✓	✓	Good	Exc.	[45]
VI-DSO [46]	VO	High grad.	Direct	Local BA	-	-	-	-	-	✓	-	-	Very Good	Exc.	-
BASALT [47]	VO	FAST	KLT (LSSD)	Local BA	-	ORB BA	-	-	-	-	✓	✓	Very Good	Exc.	[48]
Kimera [8]	VO	Shi Tomasi	KLT	Local BA	-	DBoW2 PG	-	-	-	-	✓	-	Good	Exc.	[49]
ORB-SLAM3 (ours)	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	✓	✓	✓	✓	✓	✓	Exc.	Exc.	[5]

4.2 Visual SLAM: Tracking

Juan D. Tardós, Juan J. Gómez
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



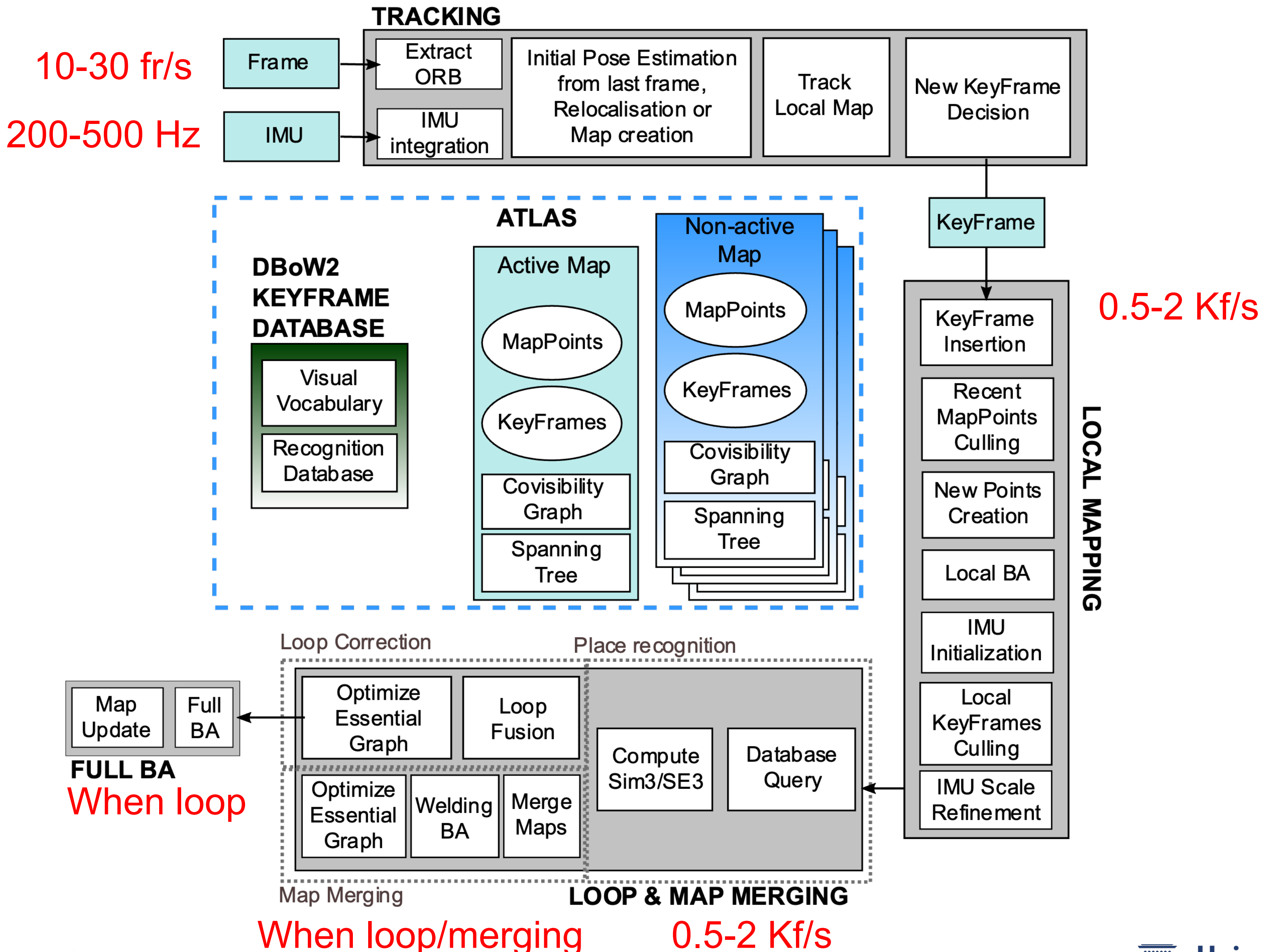
Lesson 4. Visual SLAM

2. Tracking
    a. Overview
    b. Feature Extraction
    c. Feature Matching
    d. Feature Tracking
    e. Camera Model
    f. Pose Tracking

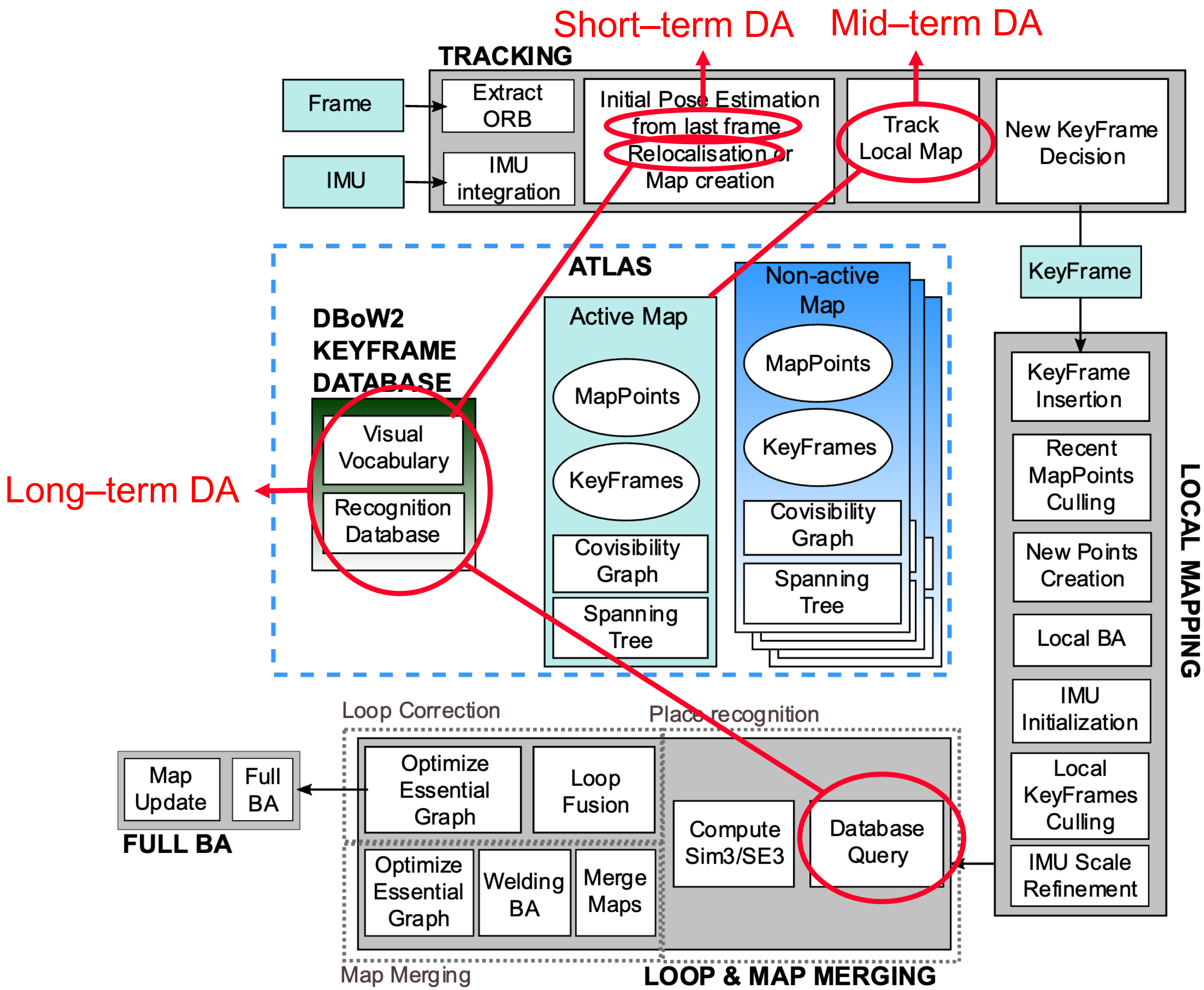
Readings:

- Sections III, IV & V of ORB-SLAM paper
- Section IV of ORB-SLAM3 paper

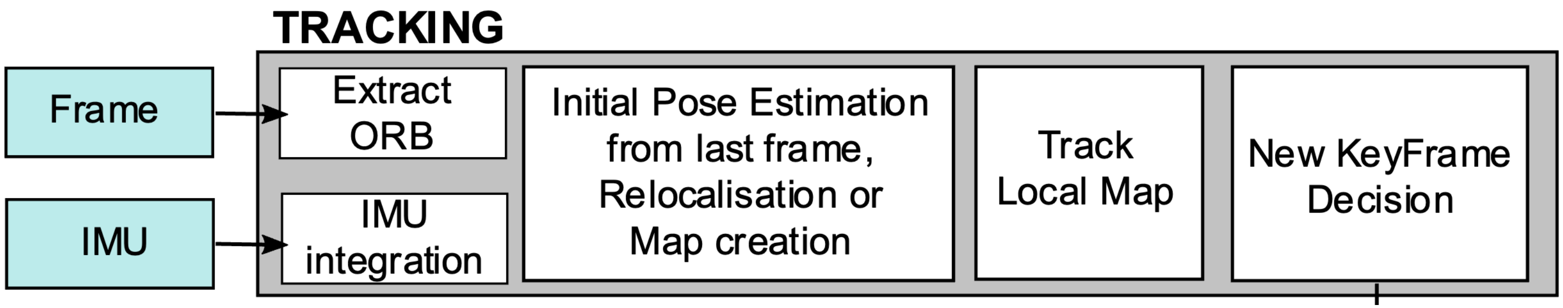
4.2.a Overview: ORB-SLAM3



Data Association



Camera Tracking



- Camera tracking is performed at the Tracking thread
- Works at frame rate (typ. 10-30 frames per second)

Main goals:

- Find feature matchings
- Compute the camera pose

Tracking Requirements

- Monocular, Stereo, RGB-D, Visual-inertial
- Pin-hole and Fisheye lenses
- Automatic map initialization
- Short-term and Mid-term data association
- Relocalization after tracking failure

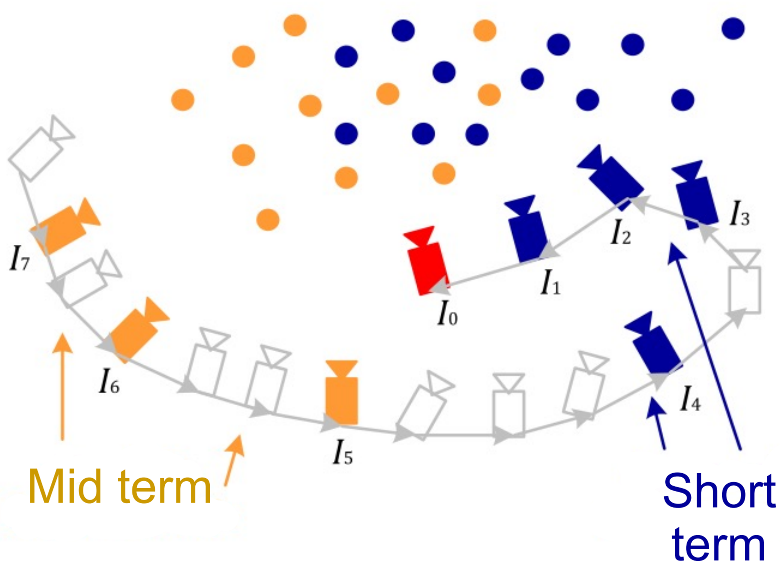
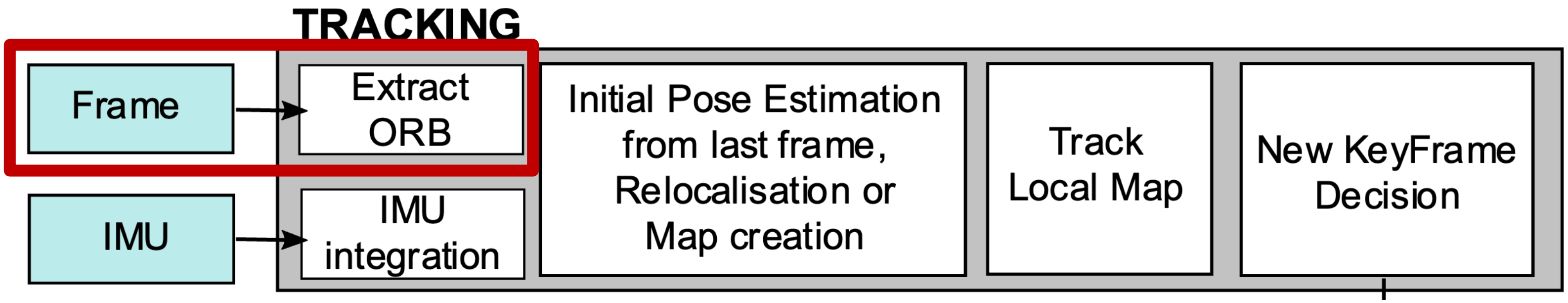


Image Processing



- Prepare each incoming image to be used in the SLAM pipeline (build a Frame object):
  - Convert images to grayscale (if needed)
  - Extract FAST features in an image pyramid
  - Compute ORB descriptors

4.2.b Feature Extraction

Local Features, Interest points, Keypoints

- Detector: find local maxima of a certain operator

original Image

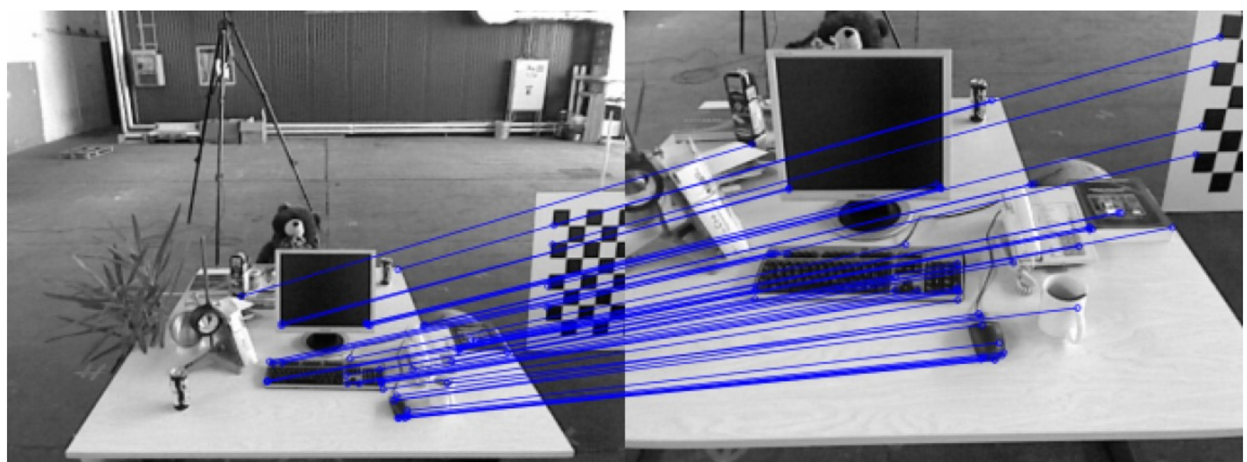
Harris detector
(corner-like)

DoG detector
(blob-like)

- Descriptor: to recognize the feature in new images

Feature Requirements

- Repeatability
- Accuracy
- Invariance
  - Illumination
  - Position
  - In-plane rotation
  - Viewpoint
  - Scale
- Efficiency



Corner detectors

- Harris Matrix or Moments Matrix:

$$A = \sum_u \sum_v w(u,v) \begin{bmatrix} I_x^2 & I_x I_y \\ I_x I_y & I_y^2 \end{bmatrix} = \begin{bmatrix} \langle I_x^2 \rangle & \langle I_x I_y \rangle \\ \langle I_x I_y \rangle & \langle I_y^2 \rangle \end{bmatrix}$$

  – I_x I_y: Image gradients
  – w: circular weights (uniform or Gaussian)
  – < >: sum over the image patch (*u*,*v*), weighted with *w*

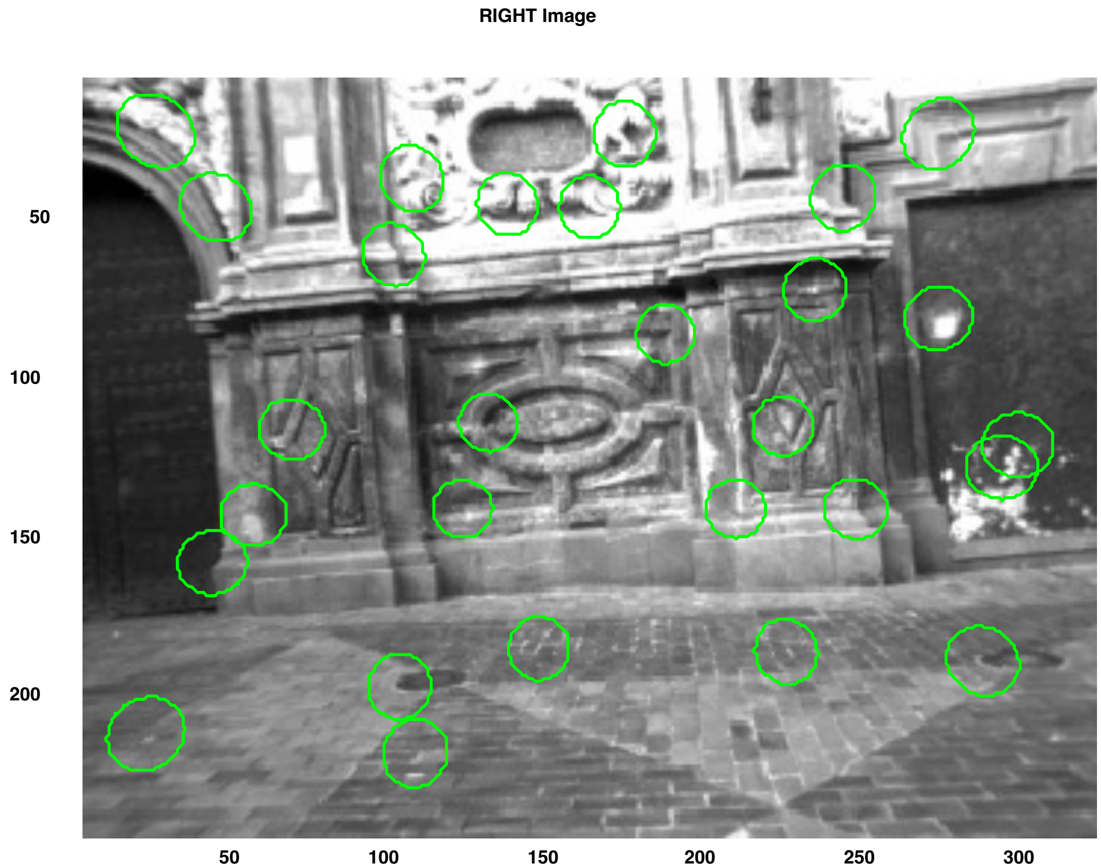
- Harris detector:

$$M_c = det\mathbf{A} - \alpha tr^2\mathbf{A} = \lambda_1\lambda_2 - \alpha(\lambda_1 + \lambda_2)^2 \qquad \alpha = 0.04 \,..\, 0.15$$

- Shi-Tomasi detector:

$$M_c = \min(\lambda_1, \lambda_2) \qquad (\lambda_1, \lambda_2) = eig(A)$$

Good for Tracking using Correlation

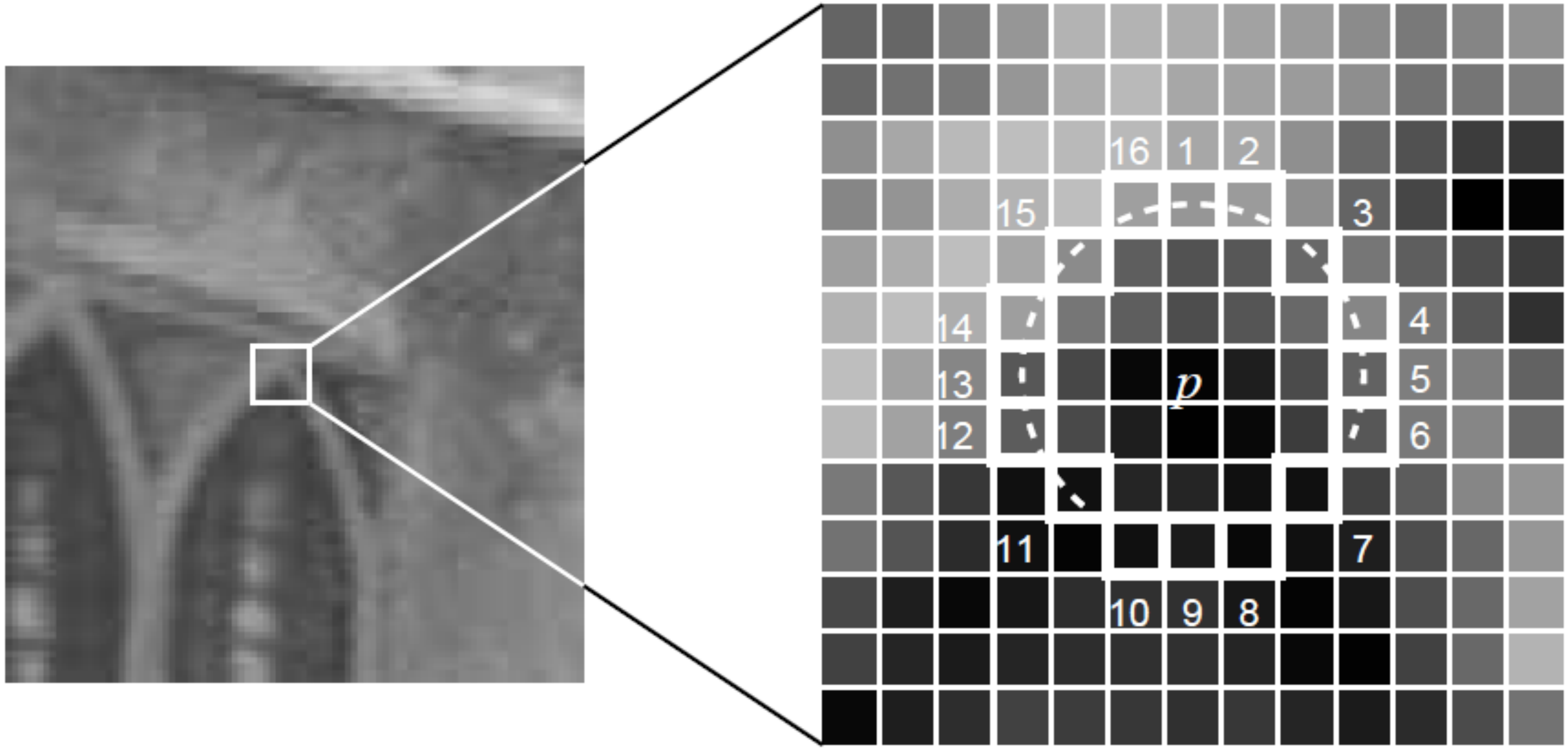


Shi-Tomasi points

Predict position in next image (@10-30 Hz)

Search by normalized correlation with a 11x11 patch

FAST corner detector



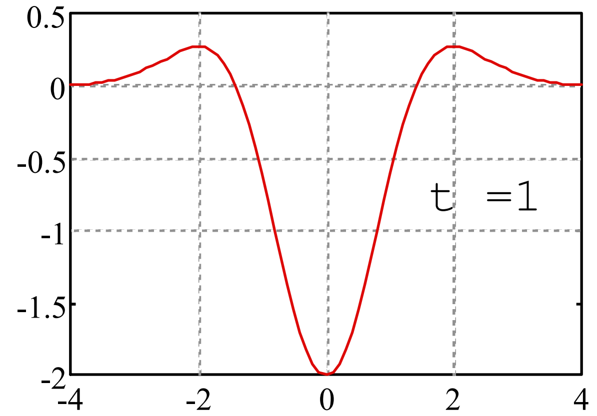
– Find pixel p surrounded by n consecutive pixels all brighter (or darker) than p

– Much faster than other detectors

E Rosten, T Drummond , Machine learning for high-speed corner detection, European Conf. on Computer Vision 2006

Blob detector using LoG

- Gaussian Filter (scale t) $L(x,y,t) = g(x,y,t) * f(x,y)$
- Laplacian of Gaussian (LoG) $\nabla^2 L = L_{xx} + L_{yy}$
- Normalized LoG $\nabla^2_{norm} L(x,y;t) = t(L_{xx} + L_{yy})$



- Feature detector:

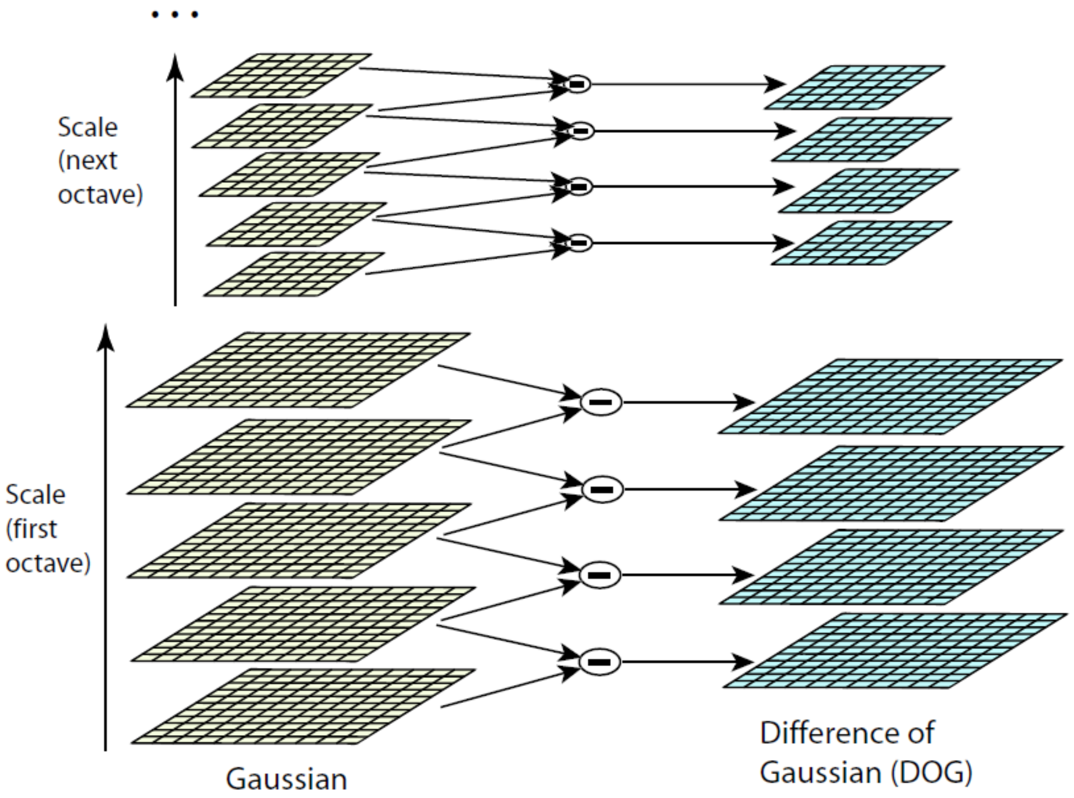
$$(\hat{x}, \hat{y}; \hat{t}) = \text{argmaxminlocal}_{(x,y;t)}(\nabla^2_{norm} L(x,y;t))$$

  – Strong response for blobs of size $\sqrt{t}$

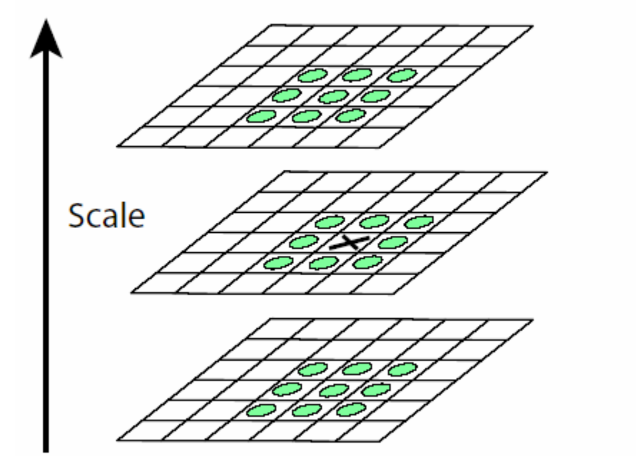
SIFT detector: Difference of Gaussians

- LoG ≈ Difference of Gaussians DoG:

$$\nabla^2 L(x,y;t) = \frac{1}{2\Delta t}\left(L(x,y;t+\Delta t) - L(x,y;t-\Delta t)\right)$$



Search for maxima in space and scale



Automatic Scale Selection

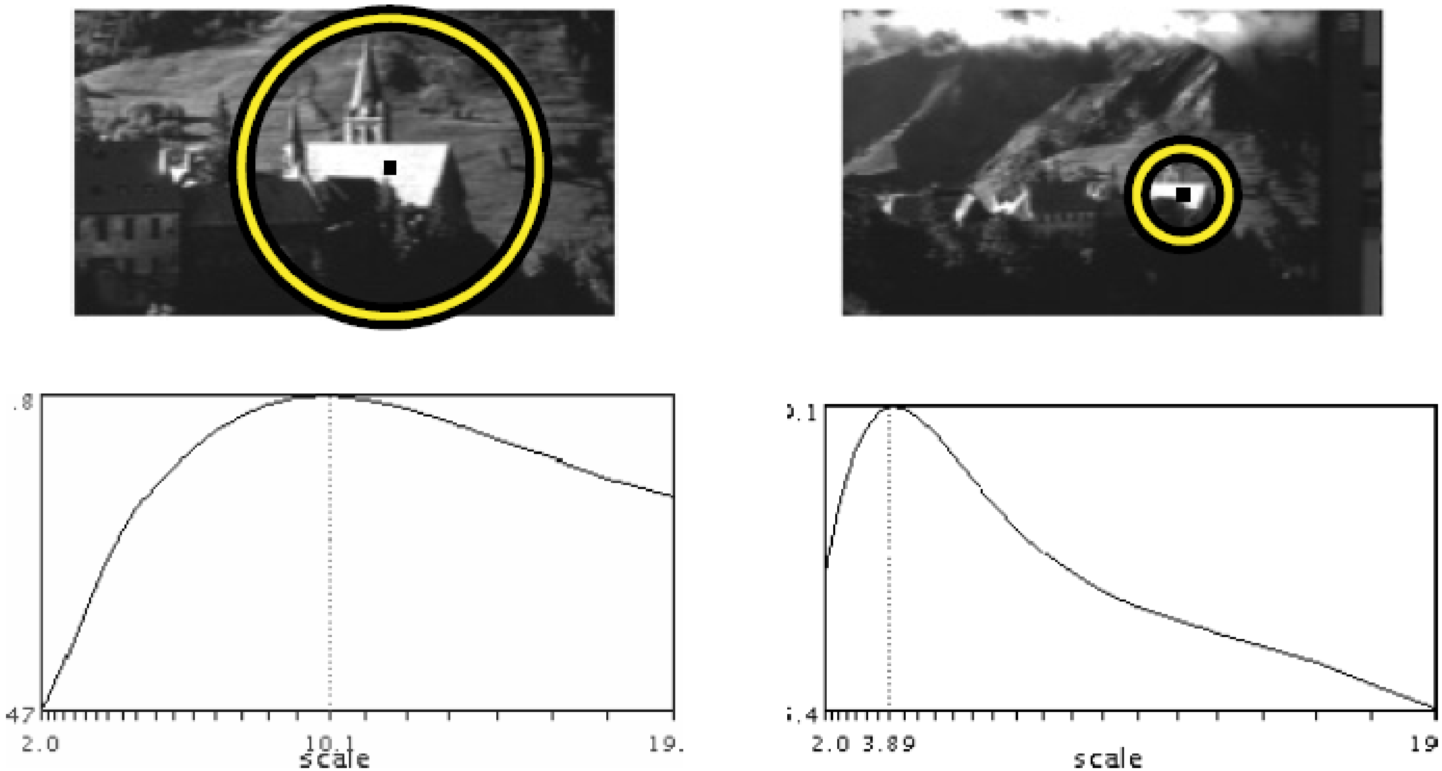
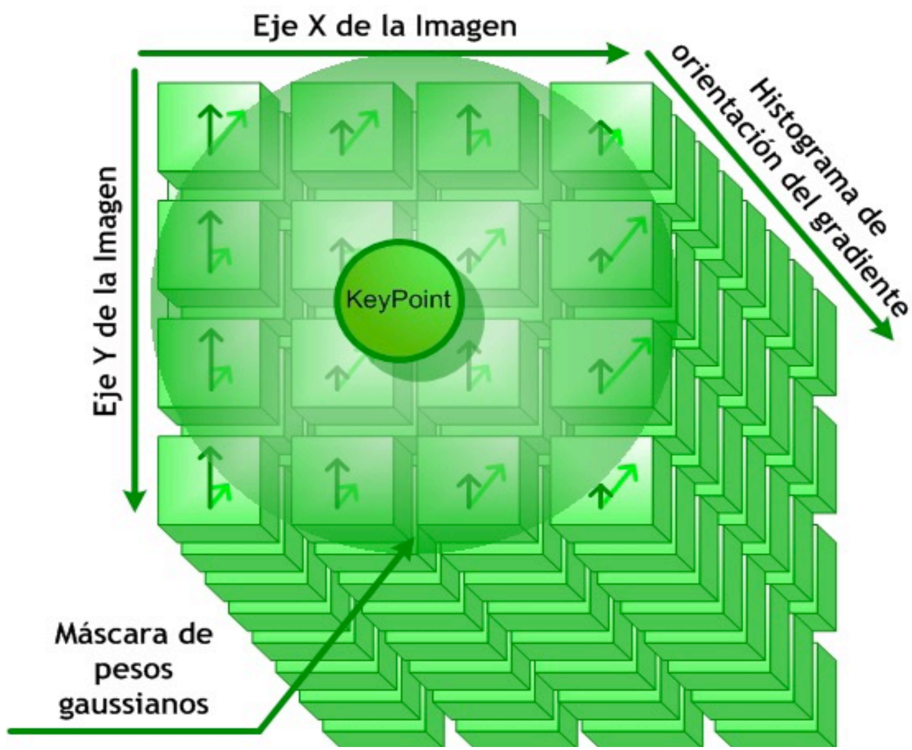


Fig. 3.5 Example of characteristic scales. The top row shows images taken with different zoom. The bottom row shows the responses of the Laplacian over scales for two corresponding points. The characteristic scales are 10.1 and 3.9 for the left and right images, respectively. The ratio of scales corresponds to the scale factor (2.5) between the two images. The radius of displayed regions in the top row is equal to 3 times the selected scales.

SIFT Descriptor

- Histogram of 8 gradient orientations in 16 areas of 4x4 pixels around the detected keypoint

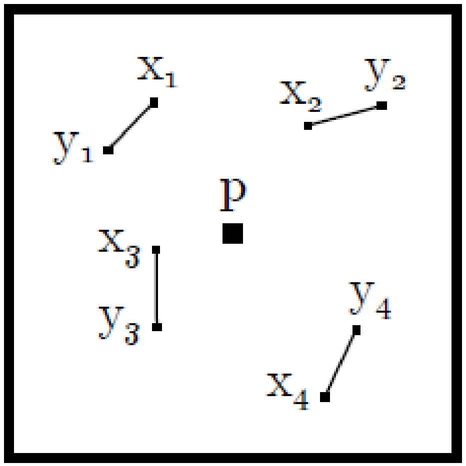


- 128 bytes (floats): 16 areas x 8 histogram bins

Binary Descriptors: BRIEF

- Computed around a FAST corner

BRIEF descriptor:



$$D_i(\mathbf{p}) = \begin{cases} 1 & \text{if } I(\mathbf{p} + \mathbf{x}_i) < I(\mathbf{p} + \mathbf{y}_i) \\ 0 & \text{otherwise} \end{cases}$$

$$\hookrightarrow D(\mathbf{p}) = [\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\ 1 \ldots\ ]$$

- Binary string, 256 bits in length.
- It is not invariant to scale or rotation.

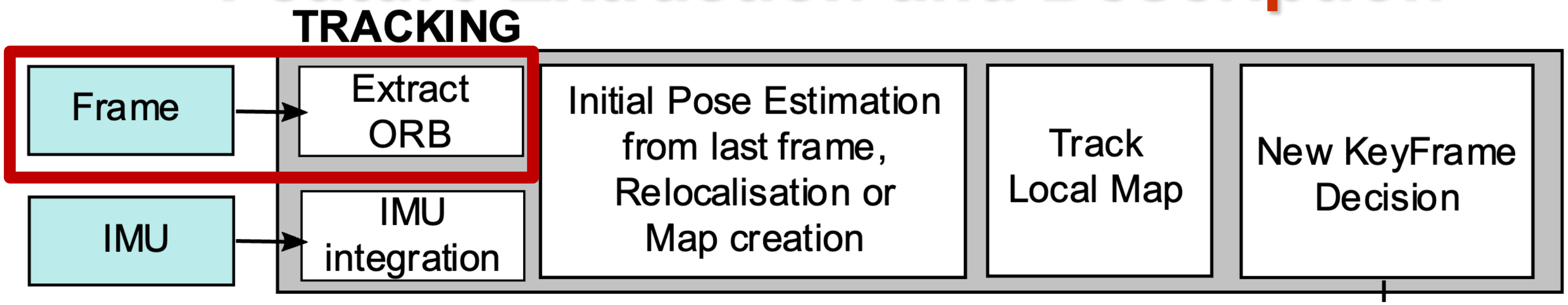
Popular Features for Visual SLAM

Detector	Descriptor	Rotation Invariant	Automatic Scale	Accuracy	Relocation & Loops	Efficiency
Harris	Patch	No	No	++++	-	++++
Shi-Tomasi	Patch	No	No	++++	-	++++
SIFT	SIFT	Yes	Yes	++	++++	+
SURF	SURF	Yes	Yes	++	++++	++
FAST	BRIEF	No	No	+++	+++	++++
FAST	ORB	Yes	No	+++	+++	++++

- ORB: Oriented FAST and Rotated Brief
  - Detect FAST and compute orientation (gradient direction)
  - Rotate the Brief pattern and obtain 256-bit binary descriptor
  - Fast to extract and match (Hamming distance)
  - Good for tracking, relocation and loop detection

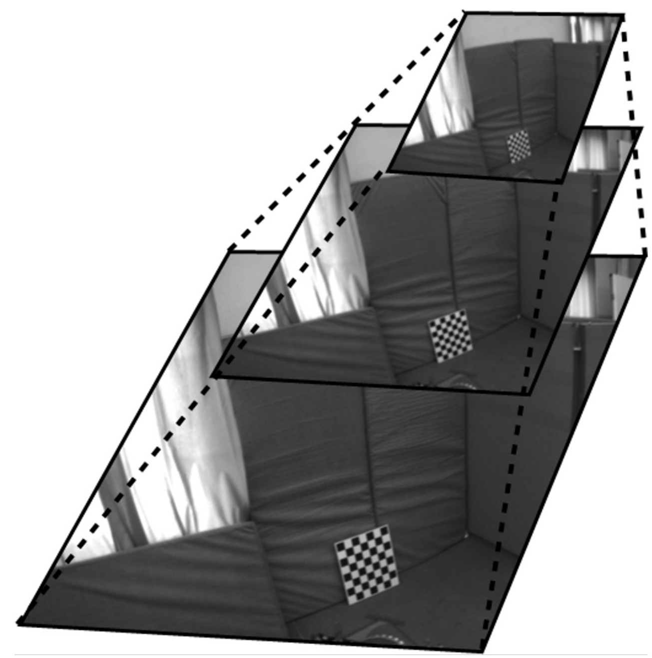
Rublee, E., Rabaud, V., Konolige, K., & Bradski, G.
ORB: an efficient alternative to SIFT or SURF, ICCV 2011

Feature Extraction and Description

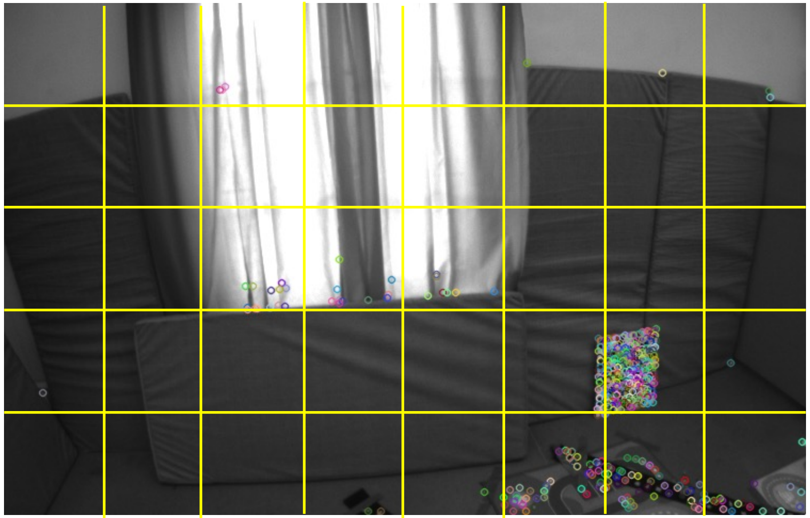


Multi-scale ORB detection

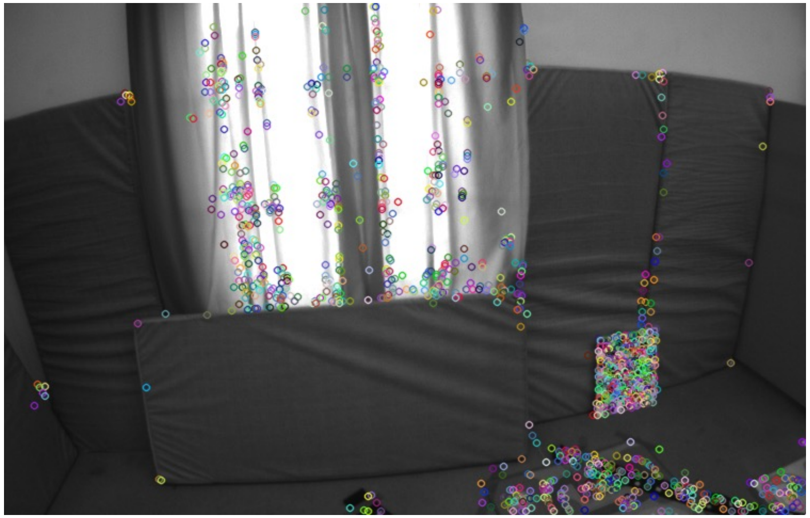
- Goal: match features when they move farther or closer to the camera
- Pyramid of scales (Gaussian filtered)
- 8 scales with a scale factor of 1.2 (aprox. 2 octaves)
- This can be parametrized in the calibration file
- ORB features may appear at the same pixel on several scales



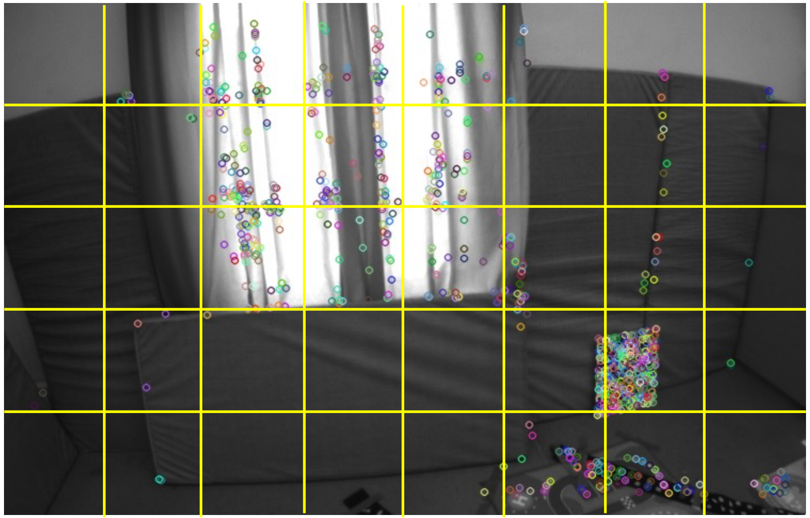
Distribute Features in the Image



FAST th=20, many empty areas

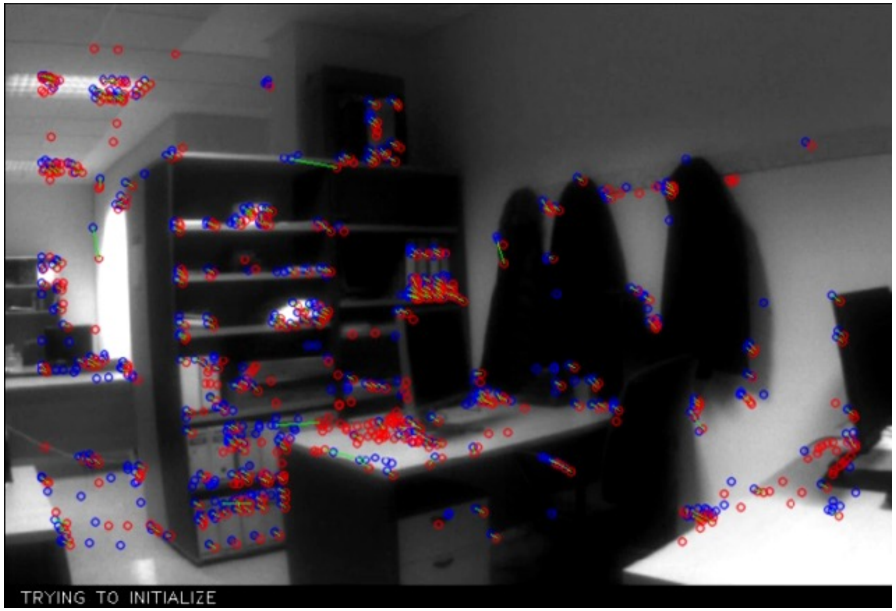


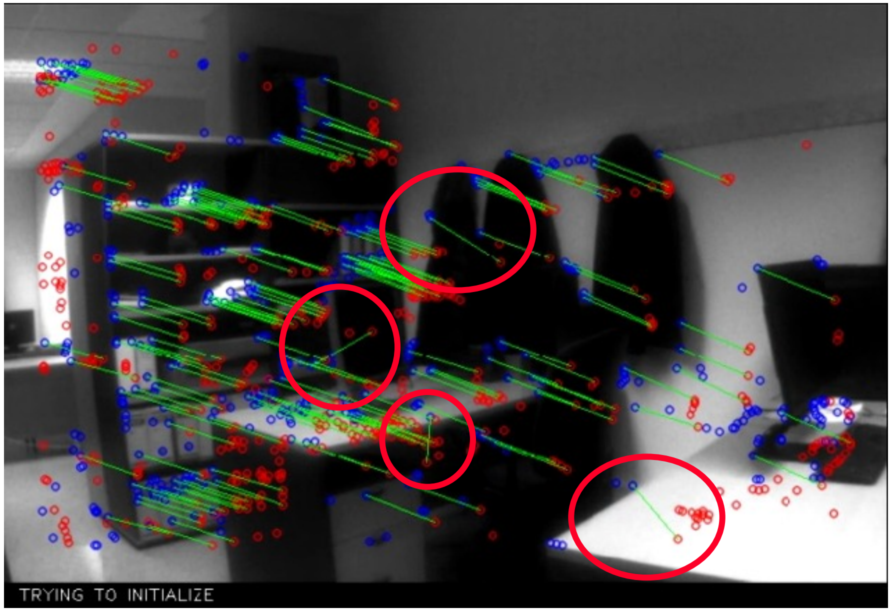
FAST th=7, too many features



FAST th=20-7, better distribution

4.2.c Feature Matching

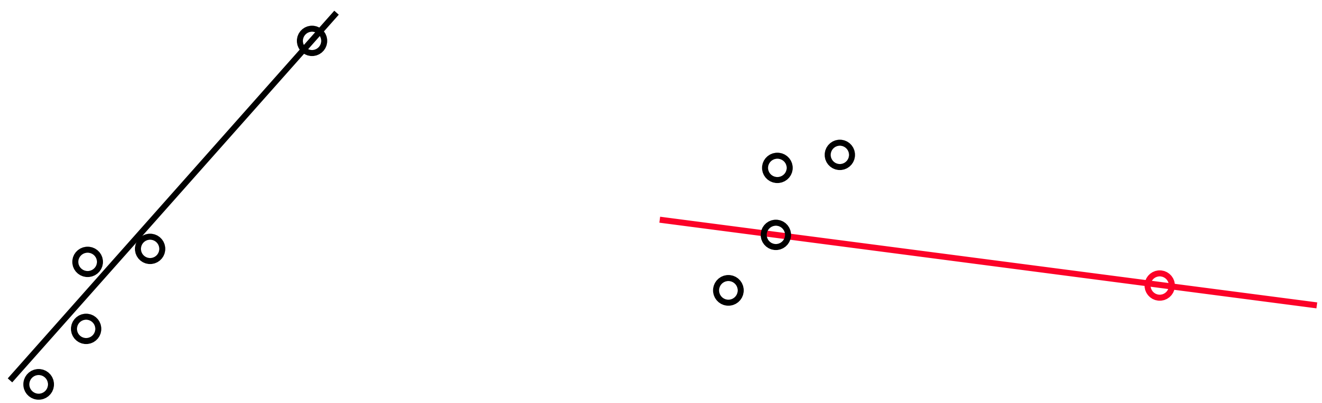




- Compare descriptors to get putative matchings
  - SIFT: Euclidean distance; ORB: Hamming distance
  - Improves using ratio to second neighbor
- Remove spurious matchings
  - Search for consensus with a robust technique: RANSAC

The Problem of Spurious Matchings

- Least-squares is very sensitive to spurious data
- A single spurious match may to ruin the estimation
- Leverage point:

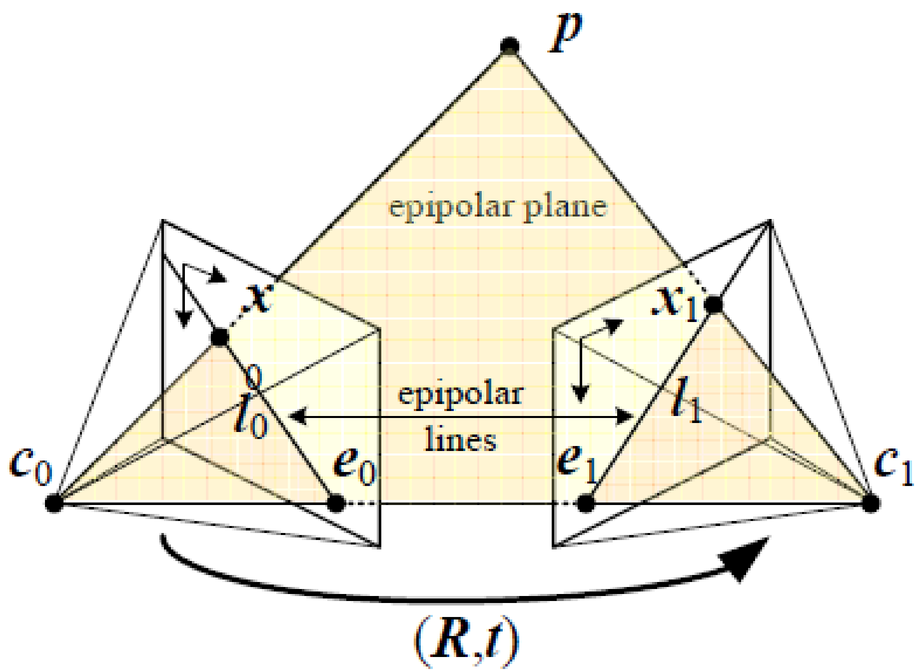


  - Removing the points with higher residuals DOES NOT SOLVE THE PROBLEM

RANSAC: RANdom SAmpling Consensus

```
RANSAC (P) return M and S
-- P: set of potential matches
-- M: alignment model found (requires at least k matchings)
-- S: set of supporting matches
for i = 1..max_attempts
      Si ← choose randomly k matchings from P
      Mi ← compute alignment model from Si
      Si* ← matchings in P that agree with Mi  (with tolerance ε )
      if  #(Si*) > consensus_threshold
         Mi* ← compute alignment model from Si*  (using least squares)
         return Mi* and Si*
      end if
endfor
return failure
```

Two View Model: Epipolar Constraint

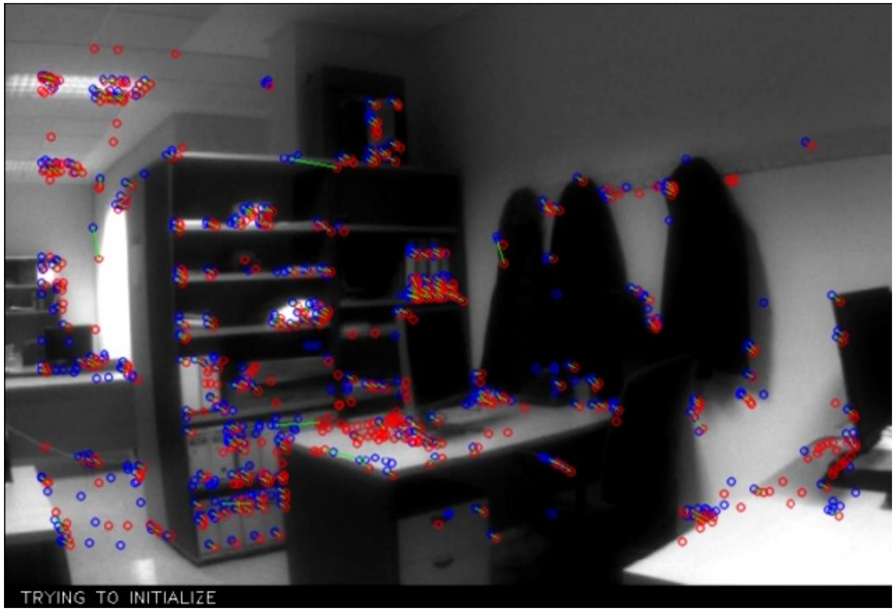


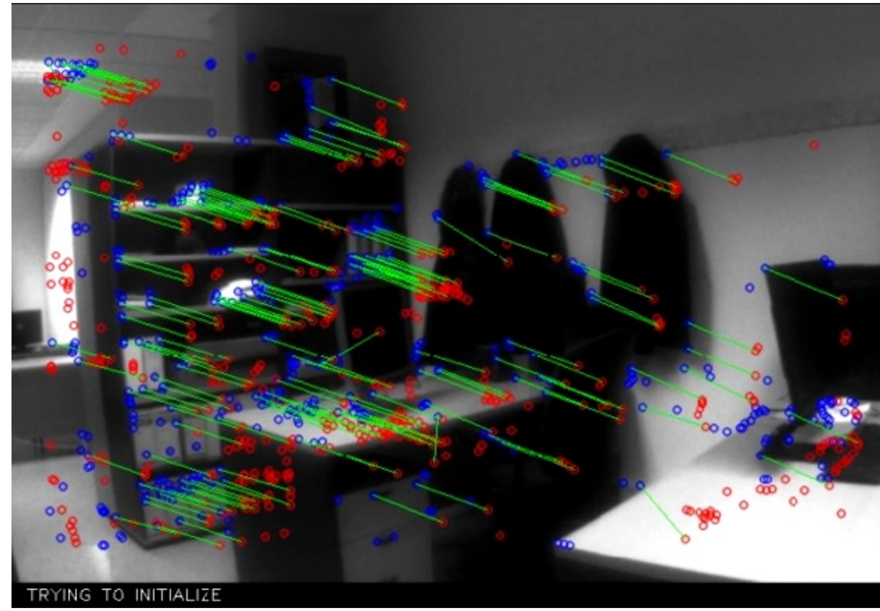
- Vectors **$\mathbf{t} = \mathbf{c}_1 - \mathbf{c}_0$, $\mathbf{p} - \mathbf{c}_0$, $\mathbf{p} - \mathbf{c}_1$** must be coplanar

- Epipolar constraint: $$\mathbf{x}_{c1}^T \mathbf{E}\, \mathbf{x}_{c0} = 0$$

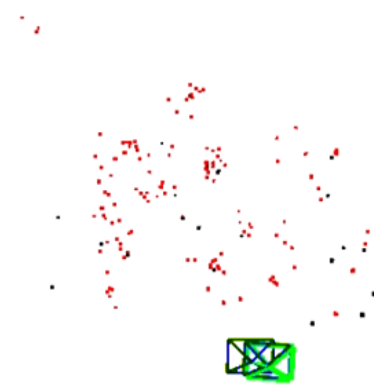
- Essential Matrix: $$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \mathbf{R}$$

Matchings in 2 Frames → 3D Points and Motion





- Find E:
  - 5pt or 8pt algorithm
- E → R_i, t_i
  - 4 solutions
- Triangulate points
  - Choose good solution



Monocular → Unknown Scale!

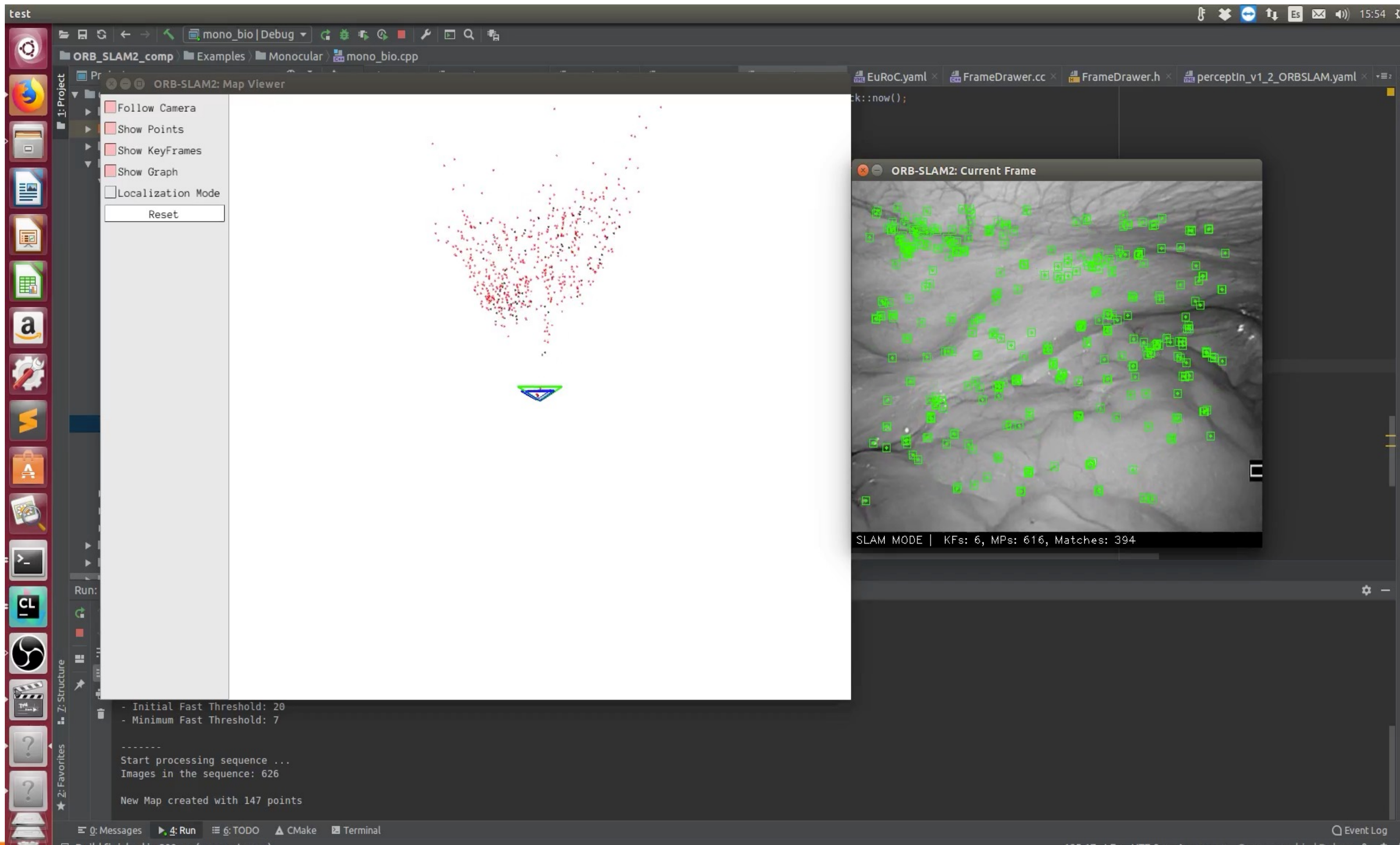
Matching Problems

Problem	Inputs	Model to find	Basic Equation	d.o.f.	Min. # of matches	Minimal solution
Camera Location	$\mathbf{u}_{ij}, \mathbf{x}_{wj}$	Pose $\mathbf{T}_{iw}$	$\pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj})$	6	3	P3P PnP
Initialize 3D scene	$\mathbf{u}_{1j}, \mathbf{u}_{2j}$	Essential Matrix $\mathbf{E}_{12} = [\mathbf{t}]_{\times} \mathbf{R}$	$\mathbf{u}_{1j}^T \mathbf{E}_{12} \mathbf{u}_{2j} = 0$	5	5	5-point 8-point
Initialize 2D scene	$\mathbf{u}_{1j}, \mathbf{u}_{2j}$	Homography $\mathbf{H}_{12}$	$\mathbf{u}_{1j} = \mathbf{H}_{12} \mathbf{u}_{2j}$	8	4	

Limitations of Feature Matching

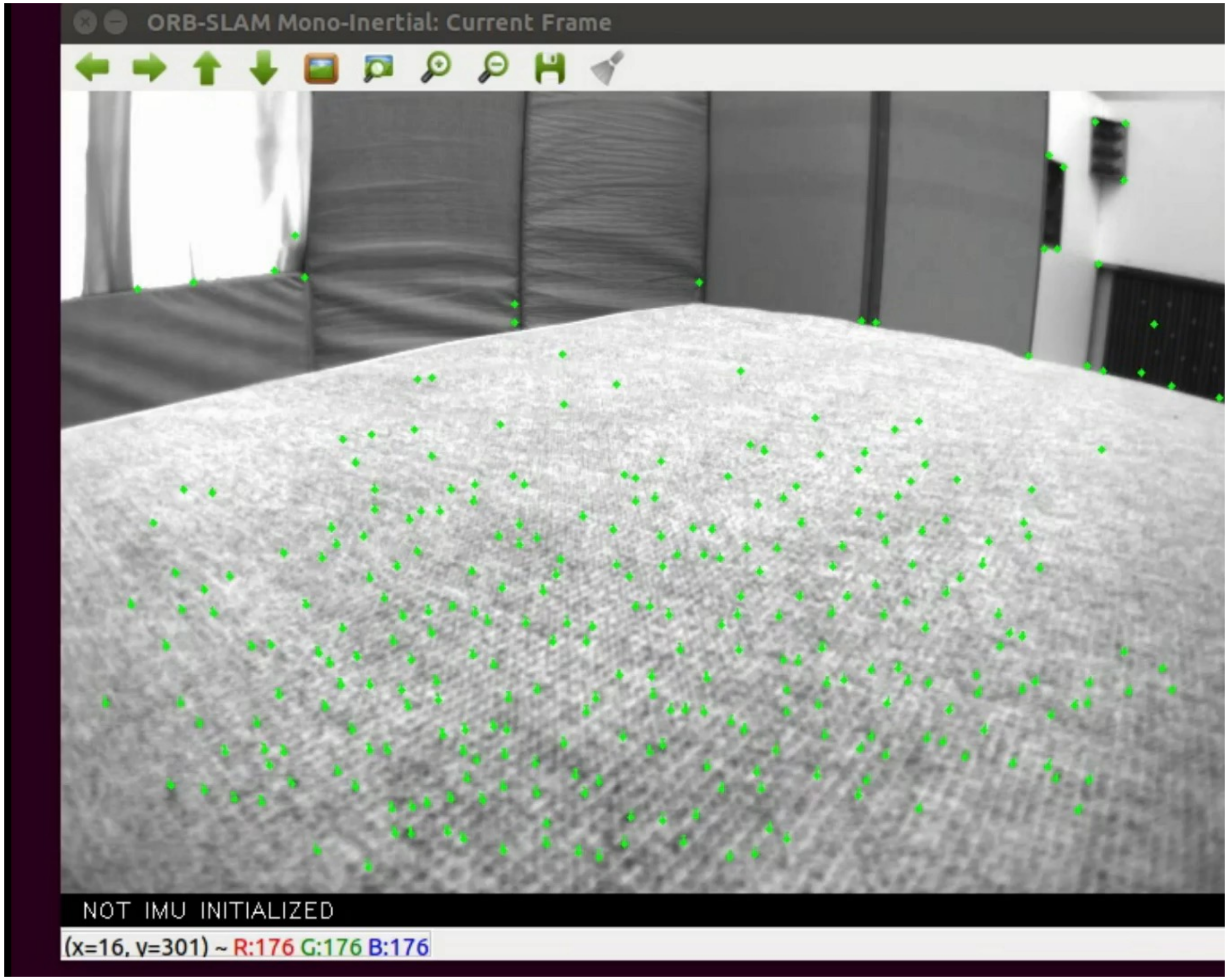
- Features extracted and matched in every frame
  - FAST features are not totally repeatable
  - ORB descriptors have some noise

→ Flickering matches



4.2.d Feature Tracking (Lucas-Kanade)

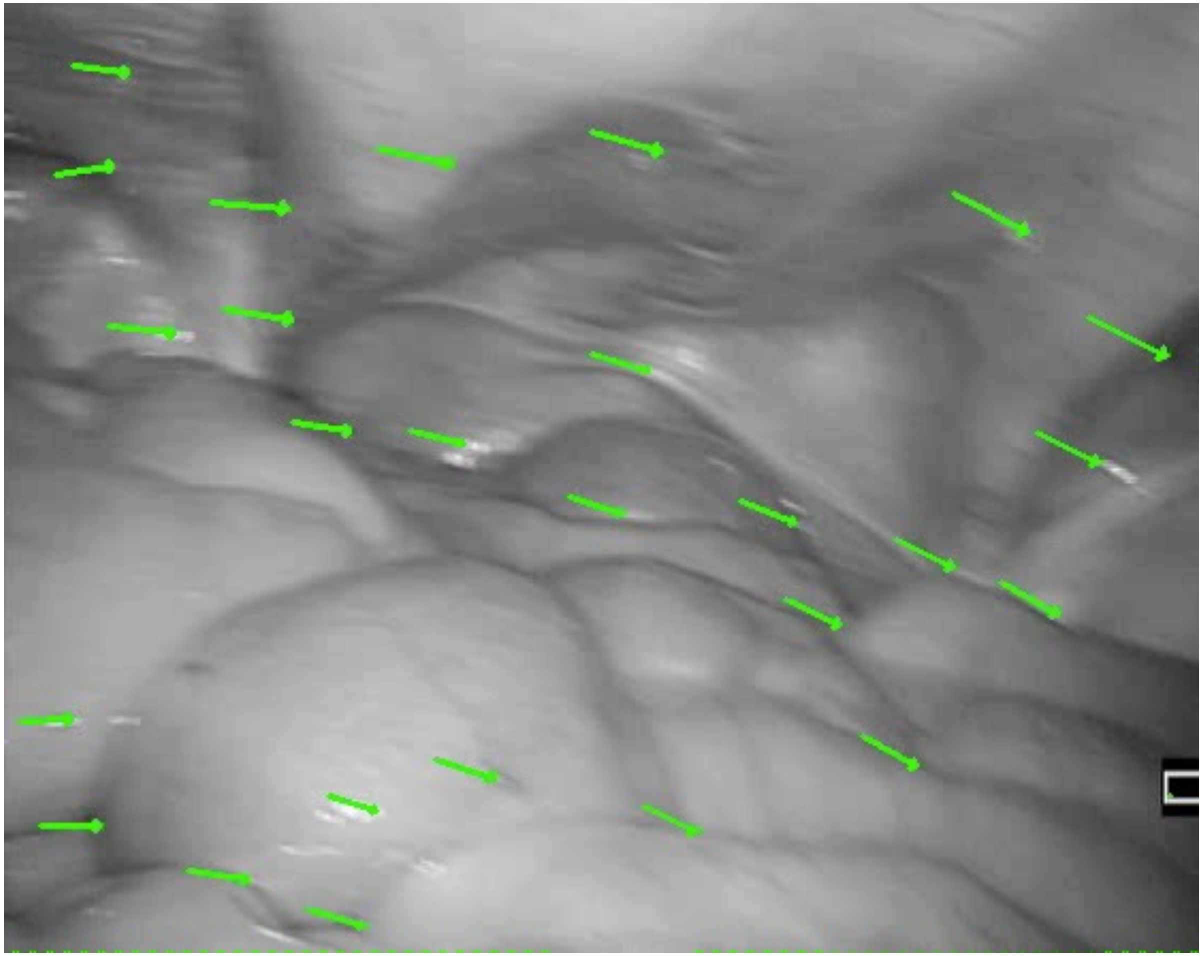
- Goal: robust and stable feature tracks



Optical Flow

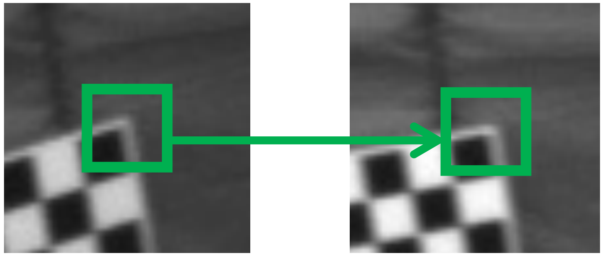
- Estimate apparent motion between succesive images:

$$d = (d_x, d_y)$$



Optical Flow

$$\underset{\mathbf{d}}{\operatorname{argmin}} \sum_{\mathbf{x} \in P(\mathbf{u})} (I(\mathbf{x}) - J(\mathbf{x}+\mathbf{d}))^2$$



- Assuming brightness constancy:

$$I(x, y, t) = I(x + \Delta x, y + \Delta y, t + \Delta t)$$

- Assuming small motion:

$$I(x + \Delta x, y + \Delta y, t + \Delta t) = I(x, y, t) + \frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t$$

- Optical Flow Equation:

$$\frac{\partial I}{\partial x}\Delta x + \frac{\partial I}{\partial y}\Delta y + \frac{\partial I}{\partial t}\Delta t = 0$$

- or:

$$\frac{\partial I}{\partial x}V_x + \frac{\partial I}{\partial y}V_y + \frac{\partial I}{\partial t} = 0$$

1 equation per pixel
2 unknowns

Lucas-Kanade Method

- Stack equations for n pixels in a window (q_i):
  - → Assumes same motion V_x V_y
  - → Assumes no camera rotation

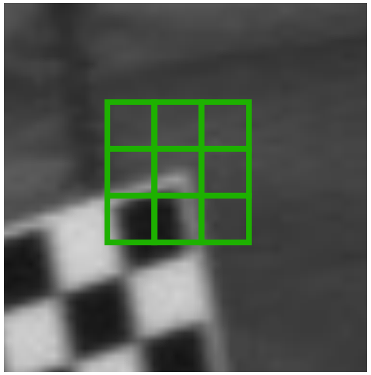
Gradient

$$I_x(q_1)V_x + I_y(q_1)V_y = -I_t(q_1)$$

$$I_x(q_2)V_x + I_y(q_2)V_y = -I_t(q_2)$$

$$\vdots$$

$$I_x(q_n)V_x + I_y(q_n)V_y = -I_t(q_n)$$



- System of equations:

$$A = \begin{bmatrix} I_x(q_1) & I_y(q_1) \\ I_x(q_2) & I_y(q_2) \\ \vdots & \vdots \\ I_x(q_n) & I_y(q_n) \end{bmatrix} \qquad v = \begin{bmatrix} V_x \\ V_y \end{bmatrix} \qquad b = \begin{bmatrix} -I_t(q_1) \\ -I_t(q_2) \\ \vdots \\ -I_t(q_n) \end{bmatrix}$$

Lucas-Kanade Method

- Solution:

$$\mathbf{v} = (A^T A)^{-1} A^T b$$

$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i I_x(q_i)^2 & \sum_i I_x(q_i) I_y(q_i) \\ \sum_i I_y(q_i) I_x(q_i) & \sum_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i I_x(q_i) I_t(q_i) \\ -\sum_i I_y(q_i) I_t(q_i) \end{bmatrix}$$

- With weighted window (typ. Gaussian weights) :

$$\mathbf{v} = (A^T W A)^{-1} A^T W b$$

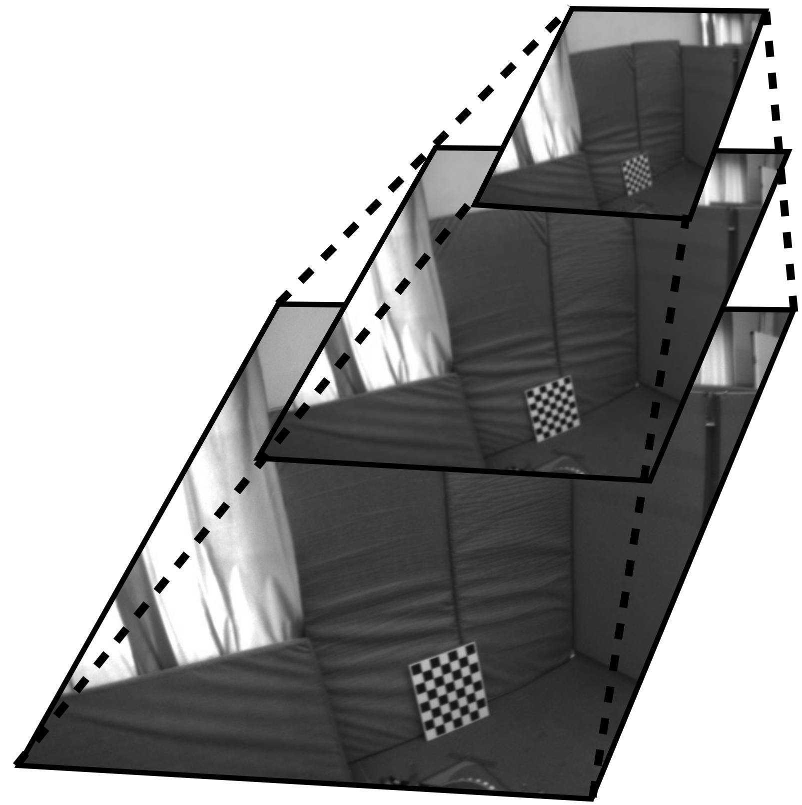
$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i w_i I_x(q_i)^2 & \sum_i w_i I_x(q_i) I_y(q_i) \\ \sum_i w_i I_x(q_i) I_y(q_i) & \sum_i w_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(q_i) I_t(q_i) \\ -\sum_i w_i I_y(q_i) I_t(q_i) \end{bmatrix}$$

Harris Matrix

$$I_t(\mathbf{x}) = I(\mathbf{x}) - J(\mathbf{x} + \mathbf{d})$$

Lucas-Kanade method

- Pyramidal processing to estimate large motions
- At each level, iterative solution for window flow
- Able to get subpixel accuracy

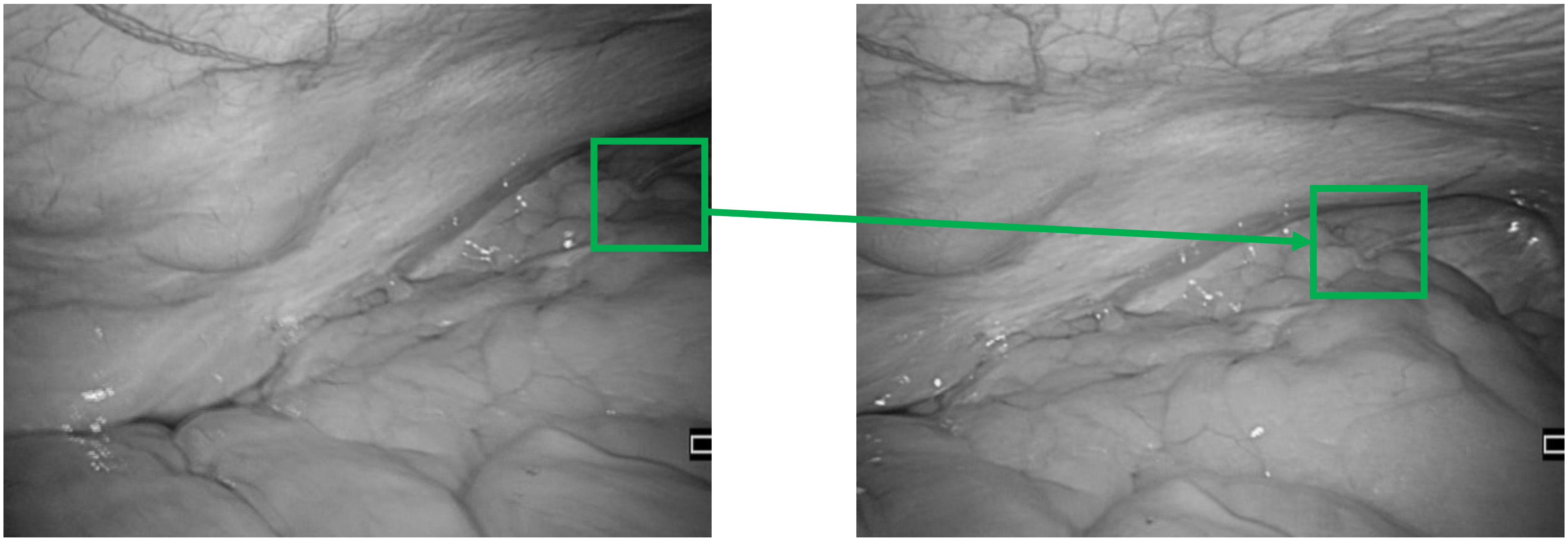


Lucas-Kanade for SLAM?

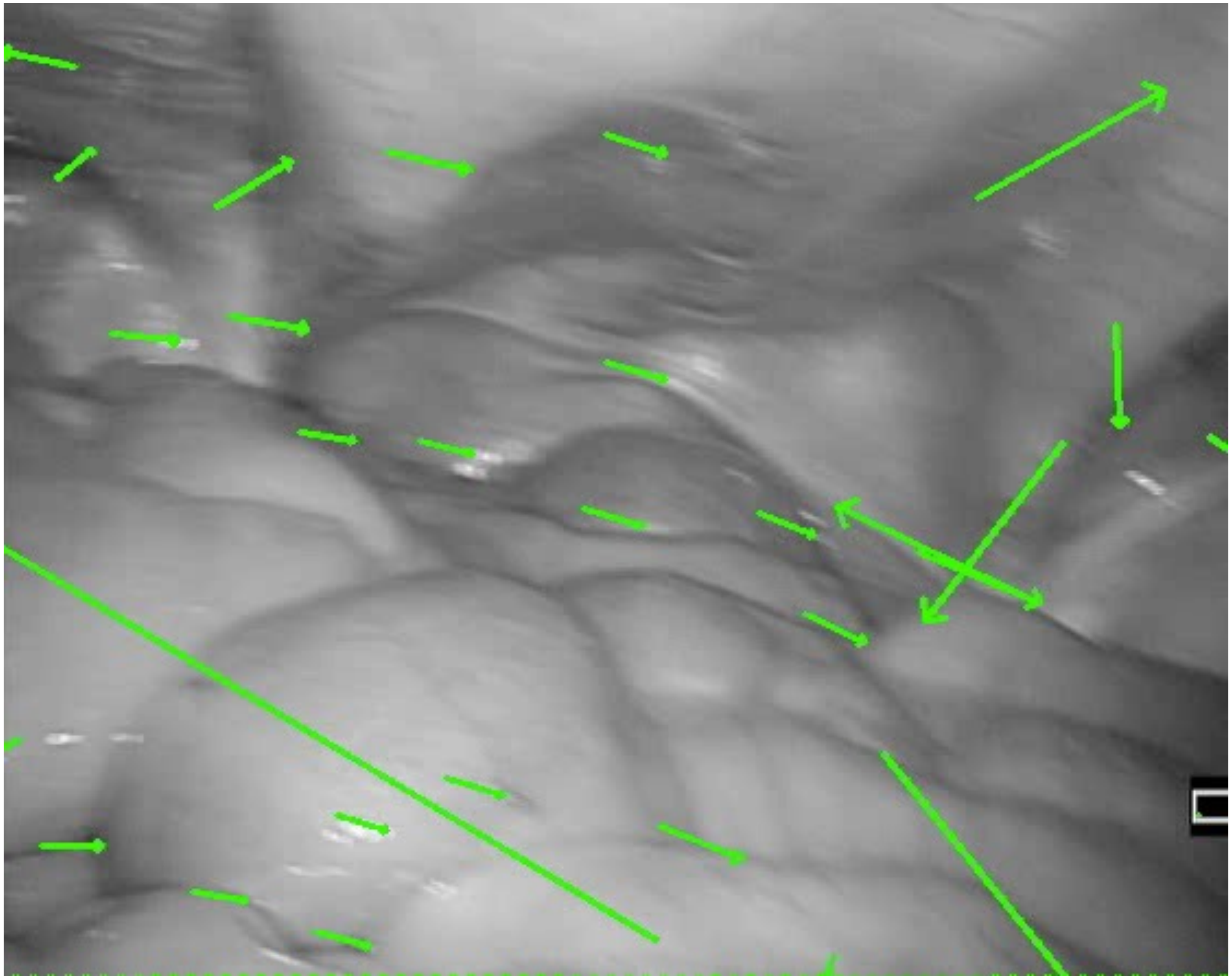
Main hypothesis

- Small displacements — KLT Pyramidal Processing
- Brightness constancy ✗
- No camera rotation ✗

Failure of Brightness Constancy



Example using KLT from OpenCV



Changes of Brightness, Orientation and Scale

- Basic optical flow

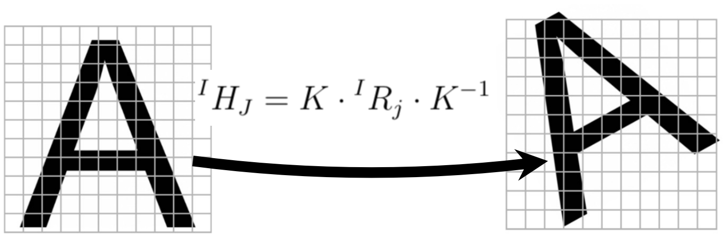
$$\underset{\mathbf{d}}{\operatorname{argmin}} \sum_{\mathbf{x} \in P(\mathbf{u})} \left(I(\mathbf{x}) - J\left(\mathbf{x} + \mathbf{d}\right)\right)^2$$

- Brightness changes:

$$\underset{\mathbf{d},\alpha,\beta}{\operatorname{argmin}} \sum_{\mathbf{x} \in P(\mathbf{u})} \left(I(\mathbf{x}) - \alpha J(\mathbf{x} + \mathbf{d}) - \beta\right)^2$$

$$\alpha = \frac{\sigma_I}{\sigma_J} \qquad \beta = \mu_I - \mu_J$$

- SLAM pose estimation → Homography
  - Compensates patch rotation and scale change



$$\underset{\mathbf{d},\alpha,\beta}{\operatorname{argmin}} \sum_{\mathbf{x} \in P(\mathbf{u})} \left(I(\mathbf{x}) - \alpha J\left(\mathbf{H}(\mathbf{x}) + \mathbf{d}\right) - \beta\right)^2$$

Modified Lukas-Kanade Method

- Modified optical flow

$$\operatorname*{argmin}_{\mathbf{d},\alpha,\beta} \sum_{\mathbf{x}\in P(\mathbf{u})} \left(I(\mathbf{x}) - \alpha J\left(\mathbf{H}(\mathbf{x}) + \mathbf{d}\right) - \beta\right)^2$$

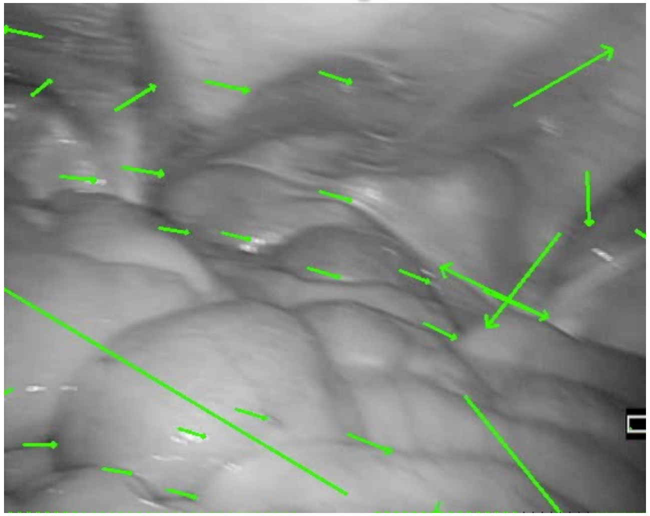
- Iterative solution:

$$\mathbf{v} = (A^T W A)^{-1} A^T W b$$

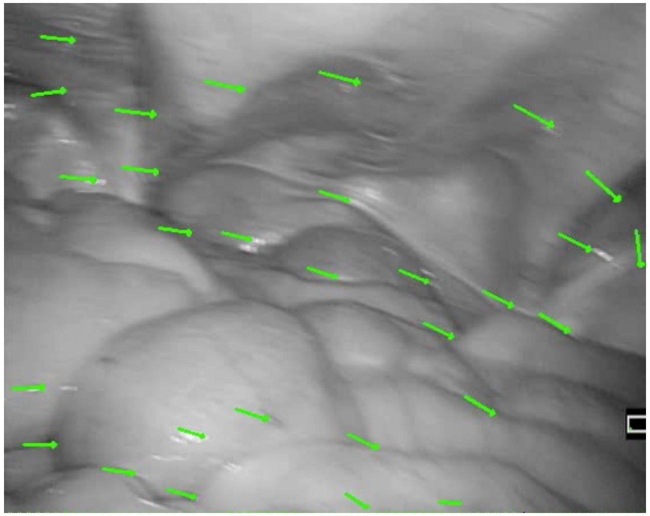
$$\begin{bmatrix} V_x \\ V_y \end{bmatrix} = \begin{bmatrix} \sum_i w_i I_x(q_i)^2 & \sum_i w_i I_x(q_i) I_y(q_i) \\ \sum_i w_i I_x(q_i) I_y(q_i) & \sum_i w_i I_y(q_i)^2 \end{bmatrix}^{-1} \begin{bmatrix} -\sum_i w_i I_x(q_i) I_t(q_i) \\ -\sum_i w_i I_y(q_i) I_t(q_i) \end{bmatrix}$$

$$\alpha = \frac{\sigma_I}{\sigma_J} \qquad \beta = \mu_I - \mu_J \qquad I_t(\mathbf{x}) = I(\mathbf{x}) - \alpha J\left(\mathbf{H}(\mathbf{x}) + \mathbf{d}\right) - \beta$$

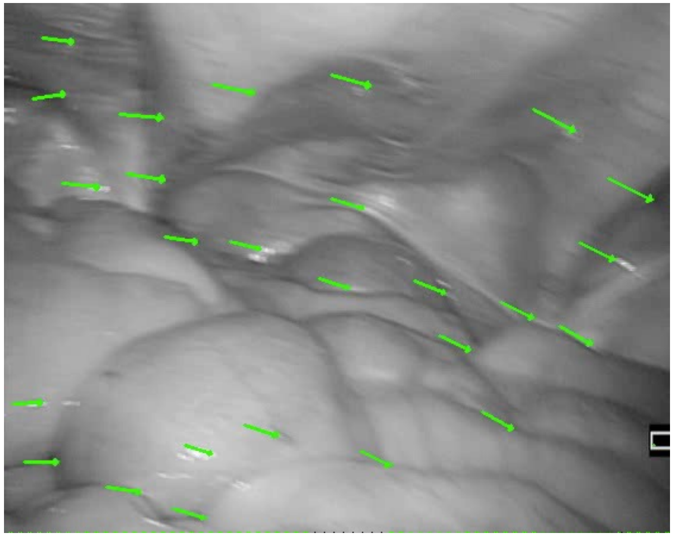
Example



OpenCV



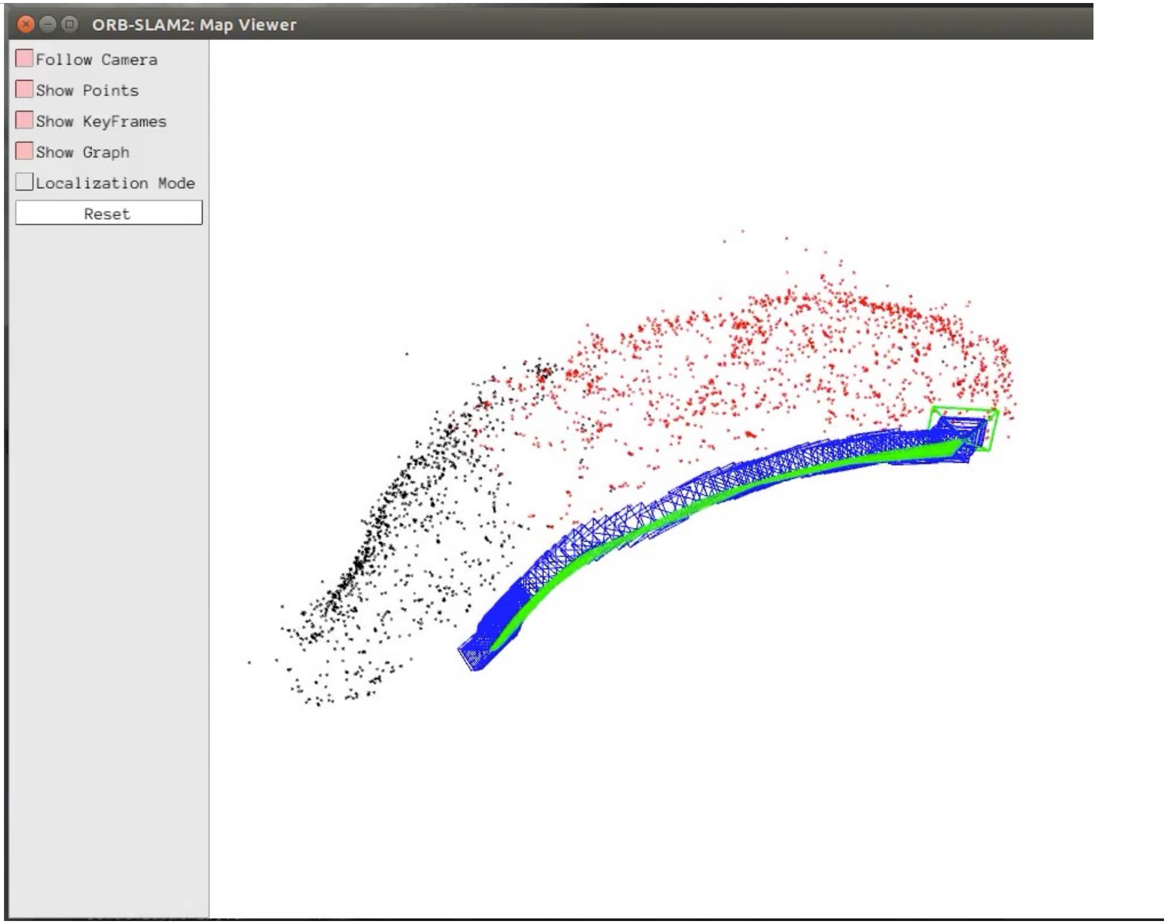
Brightness Compensation

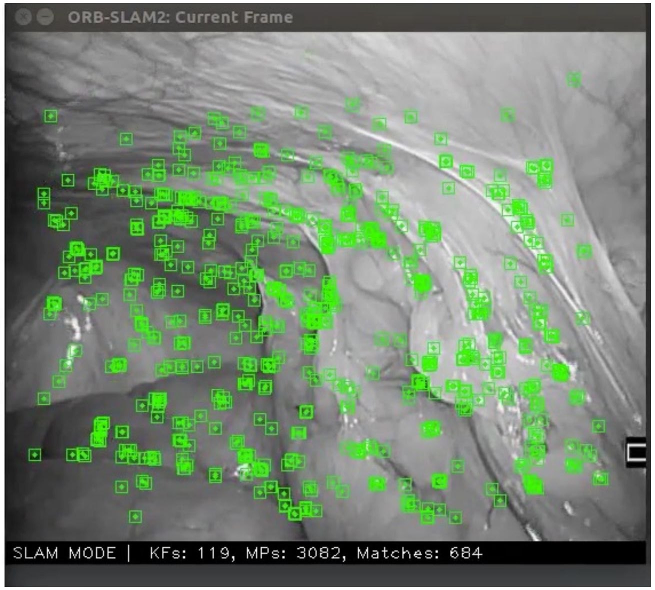


Brightness and Rotation

SLAM Example using KLT

Juan José Gómez, Robust tracking of visual features for medical image sequences in ORB-SLAM2, Ms. Thesis, Univ. Zaragoza, 2019





Features: Matching .vs. Tracking

- Feature Matching (using descriptors)
  - Extracts features in all frames
  - Some features fail to be extracted or matched
  - Good for short-term, mid-term and long-term data association

- Feature Tracking (optical flow)
  - Extracts features only in keyframes
  - Gives more stable tracks
  - Good for short-term data association → Visual Odometry

4.2.e Camera Model

1. Visual sensor setups:
   - Monocular cameras
   - Stereo cameras (rectified images)
   - RGB-D cameras (implemented as a virtual stereo)



2. Projection models:
   - Perspective (Pin-hole, or standard projection)
   - Fish-eye (Kannala-Brandt model)

Camera Model Abstraction

- Different Camera Models can be supported abstracting the functions required by the SLAM pipeline:
  - Projection, unprojection and Jacobians
- Example: Pinhole and Kannala-Brandt

GeometricCamera.h

```
project(point3D) : point2D
unproject(point2D) : point3D
projectJac(point3D) : matrix2x3
```

Interface

implementation

Pinhole.h/.cpp

```
project(point3D) : point2D
unproject(point2D) : point3D
projectJac(point3D) : matrix2x3
```

KannalaBrandt8.h/.cpp

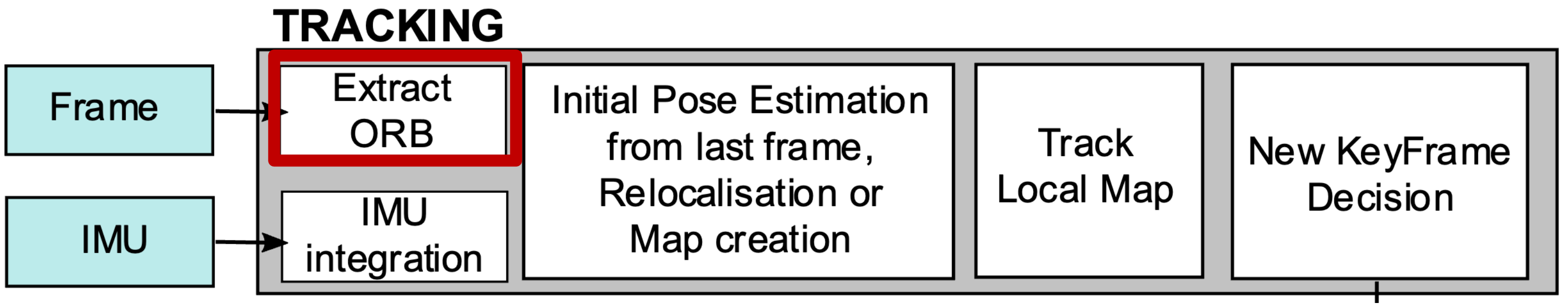
```
project(point3D) : point2D
unproject(point2D) : point3D
projectJac(point3D) : matrix2x3
```

$$\begin{pmatrix} u \\ v \end{pmatrix} = \pi \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

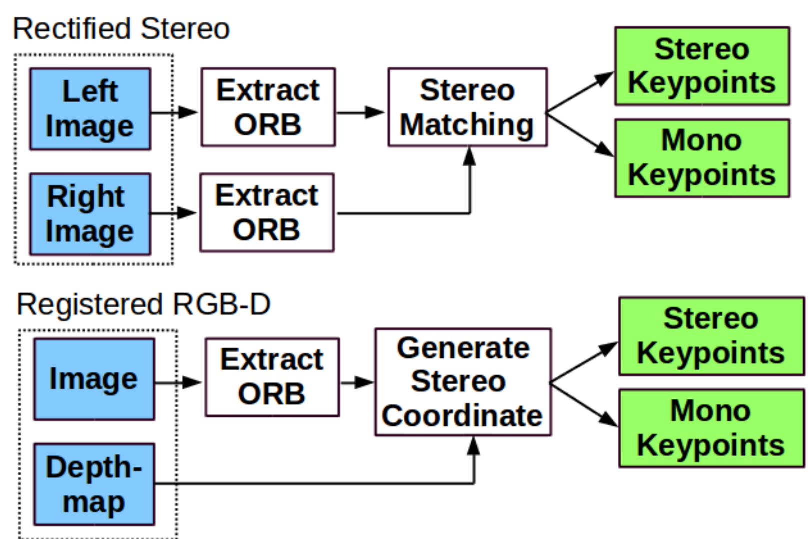
$$J_\pi = \begin{pmatrix} \frac{\partial u}{\partial x} & \frac{\partial u}{\partial y} & \frac{\partial u}{\partial z} \\ \frac{\partial v}{\partial x} & \frac{\partial v}{\partial y} & \frac{\partial v}{\partial z} \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} = \pi^{-1} \begin{pmatrix} u \\ v \end{pmatrix}$$

Monocular, Stereo and RGB-D



- **Monocular**: extract ORB features
- **Stereo**: extract ORB features in both images and match them. Compute disparity
- **RGB-D**: extract ORB features. Use Depth image to compute a virtual stereo image



Monocular, Stereo and RGB-D (ideal pin-hole)

- Monocular:

$$\mathbf{x} = \pi_m(\mathbf{X}_\mathrm{C}) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \end{bmatrix}, \quad \mathbf{X}_\mathrm{C} = [X, Y, Z]^T, \quad \mathbf{x} = [u, v]^T$$

- Stereo:

$$\mathbf{x} = \pi_s(\mathbf{X}_\mathrm{C}) = \begin{bmatrix} f_x \frac{X}{Z} + c_x \\ f_y \frac{Y}{Z} + c_y \\ f_x \frac{X-b}{Z} + c_x \end{bmatrix}, \quad \mathbf{X}_\mathrm{C} = [X, Y, Z]^T, \quad \mathbf{x} = [u_L, v_L, u_R]^T$$

- RGB-D:

$$u_r = u - \frac{f_x b_{rgbd}}{d}$$

- BA:

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj} \,|\, i \in \mathcal{C}, j \in \mathcal{P}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\left\| \mathbf{u}_{ij} - \pi_i(\mathbf{R}_{iw}\mathbf{x}_{wj} + \mathbf{t}_{iw}) \right\|^2_{\Sigma_{ij}} \right)$$

Some details

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj} \,| i \in \mathcal{C}, j \in \mathcal{P}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\| \mathbf{u}_{ij} - \pi_i \left(\mathbf{R}_{iw} \mathbf{x}_{wj} + \mathbf{t}_{iw} \right) \|^2_{\Sigma_{ij}} \right)$$

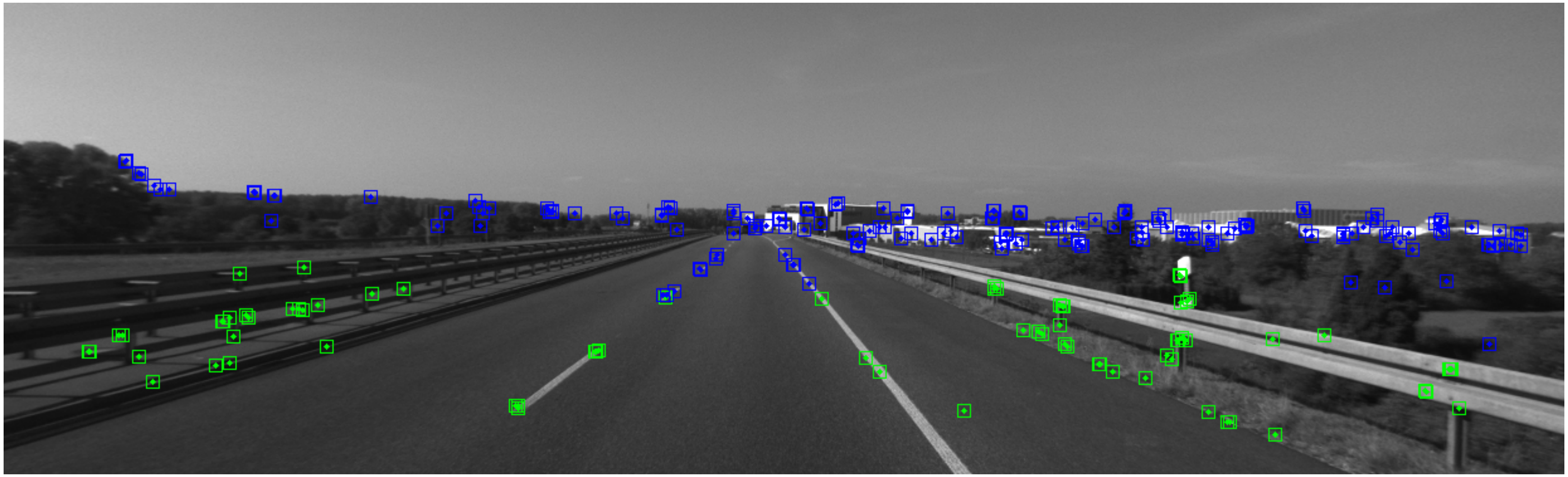
- Assumption: the camera has been calibrated
  - Focal lengths and principal point are known
  - Distortion parameters → Image rectification

Ideal Pin-hole

- $\rho_h(\ )$ robust cost function (i.e. Huber cost) to downweight wrong matchings

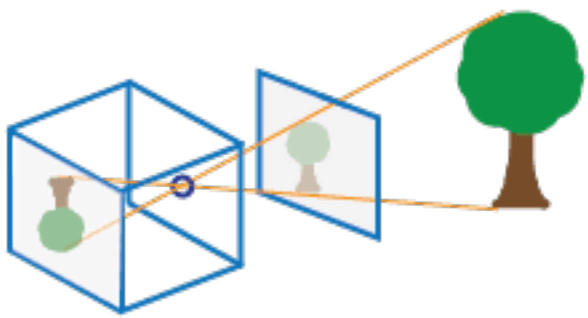
- $\Sigma_{ij} = \sigma_{ij}^2 \mathbf{I}_{2\times 2}$ std. dev. typically = 1 pixel * scale

Close and Far Points

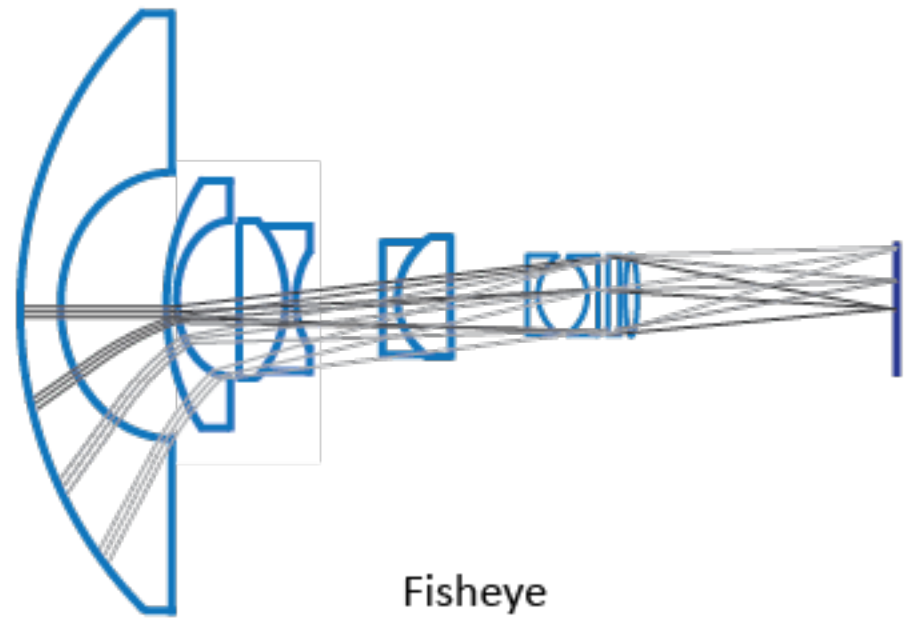


- Green points: depth <= 40 x baseline
  - Essential to compute camera translation
- Blue points: depth > 40 x baseline
  - Good to obtain camera orientation

Fish-Eye Lenses



Pinhole



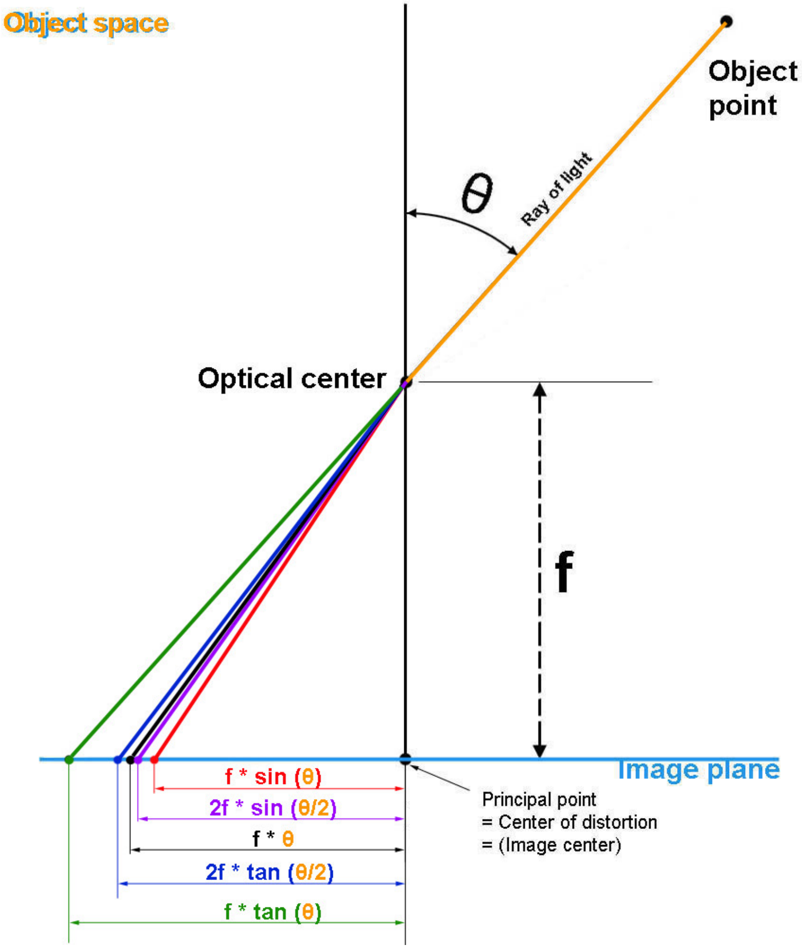
Fisheye

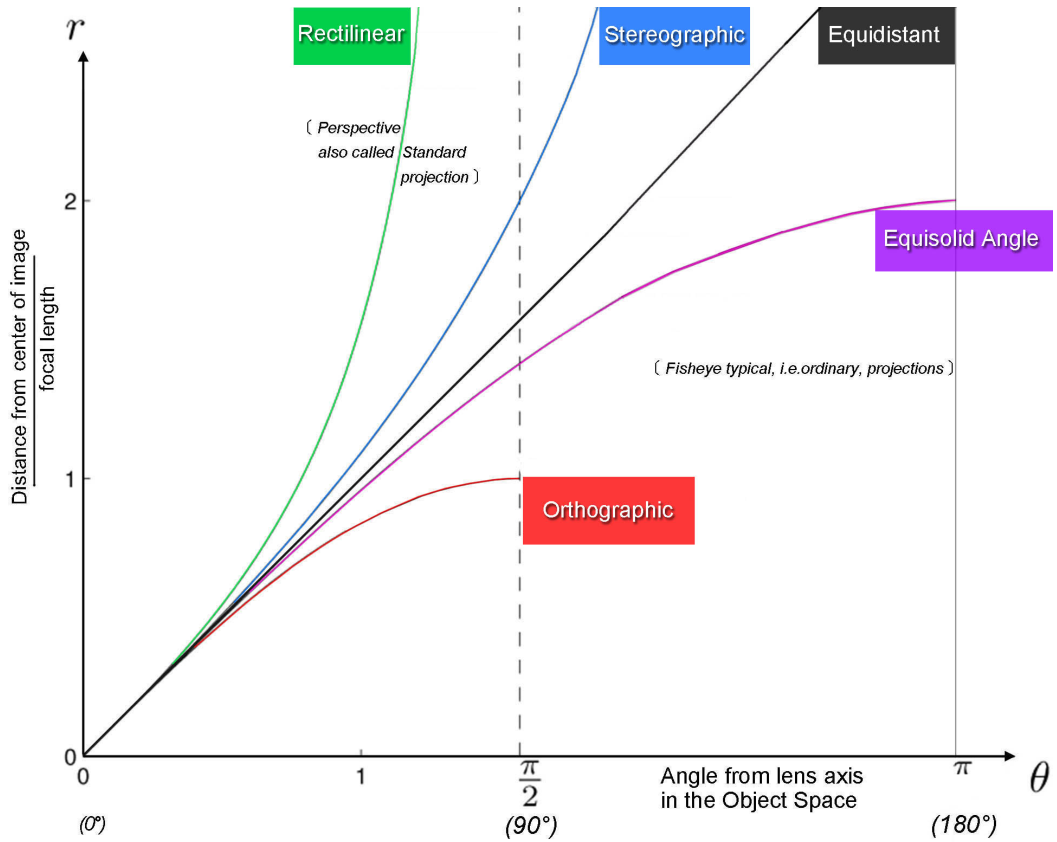
Typical FOV: 90º-100º

Can reach FOV > 180º

- Robust to occlusions
- Faster mapping

Fish-Eye Lenses





Fisheye Rectification?

Fish-eye image, FOV = 190º

Rectified to f = 220, FOV = 118º

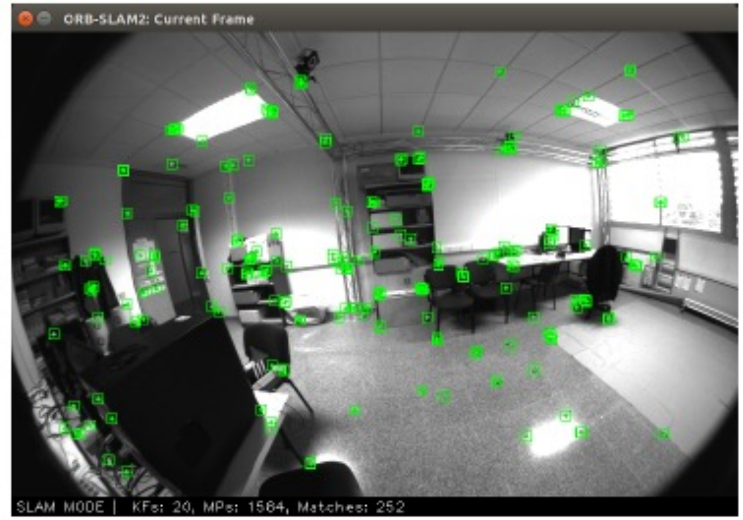
Rectified to f = 100, FOV = 150º

Problems of rectification:

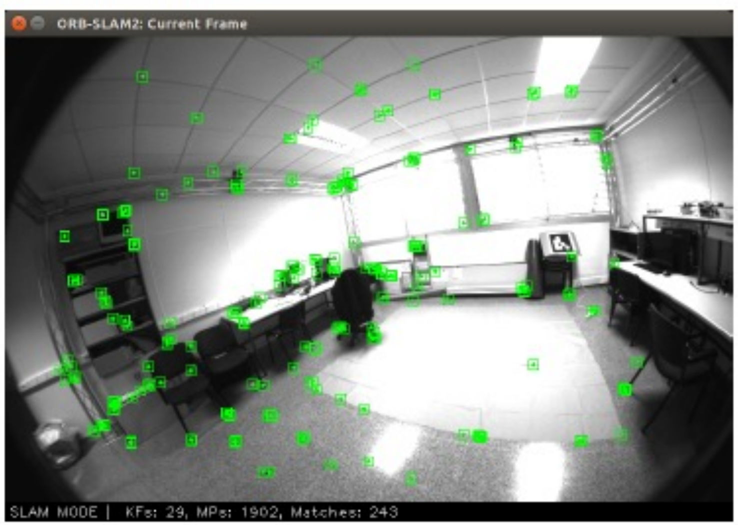
- Loss of effective FOV
- Difficult to match features
  - Objects near corners are enlarged!

SLAM with the Original Fisheye Images

(a) First *Frame* after map initialization.



(b) 100th *Frame* of the sequence.

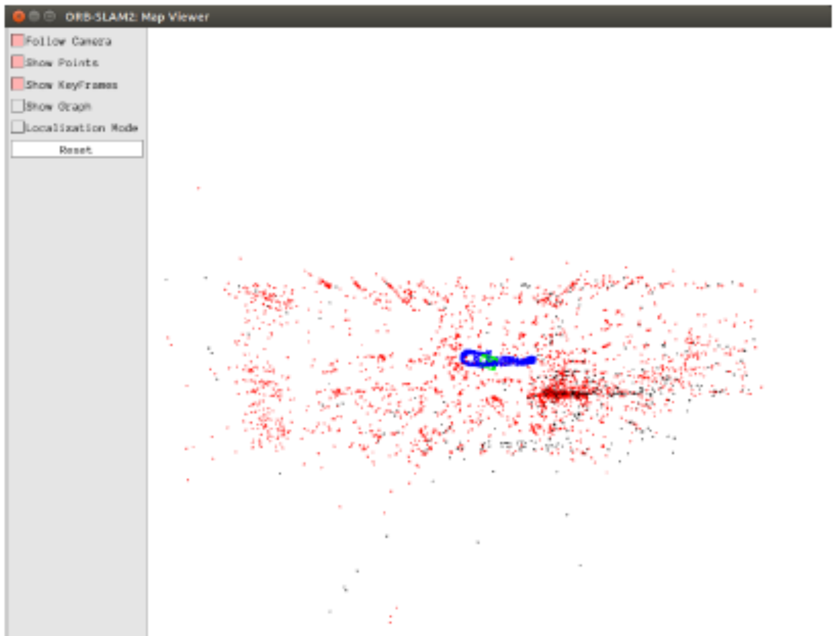


(c) 200th *Frame* of the sequence.

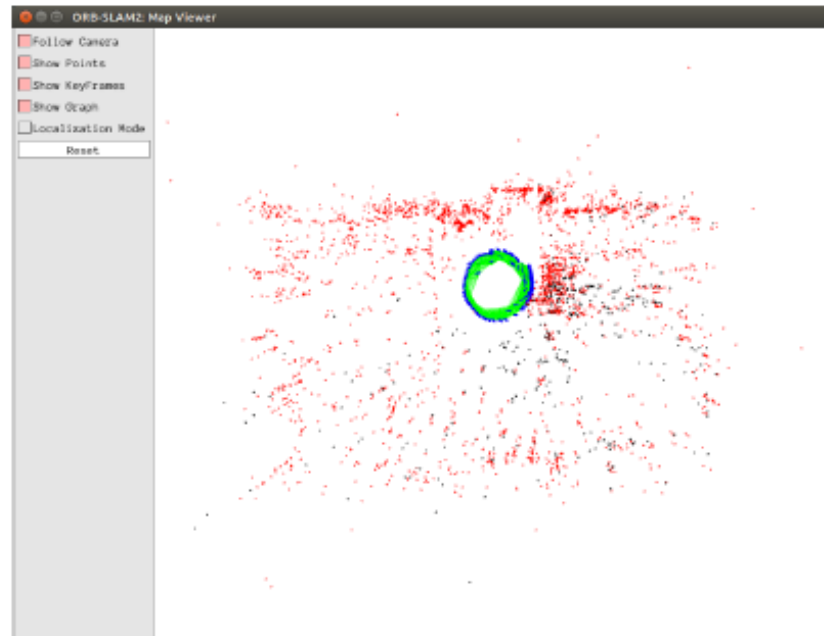
(d) 300th *Frame* of the sequence.

SLAM with the original Fisheye Images

- Faster and more accurate mapping



(a) Frontal view of the map



(b) Top view of the map

Kannala-Brandt Model

- Equidistant projection + polynomic angle distortion

Projection:

$$\pi(\mathbf{x},\mathbf{i}) = \begin{bmatrix} f_x\, d(\theta)\, \frac{x}{r} \\ f_y\, d(\theta)\, \frac{y}{r} \end{bmatrix} + \begin{bmatrix} c_x \\ c_y \end{bmatrix},$$

$$r = \sqrt{x^2 + y^2},$$

$$\theta = \operatorname{atan2}(r, z),$$

$$d(\theta) = \theta + k_1\theta^3 + k_2\theta^5 + k_3\theta^7 + k_4\theta^9$$

Unprojection:

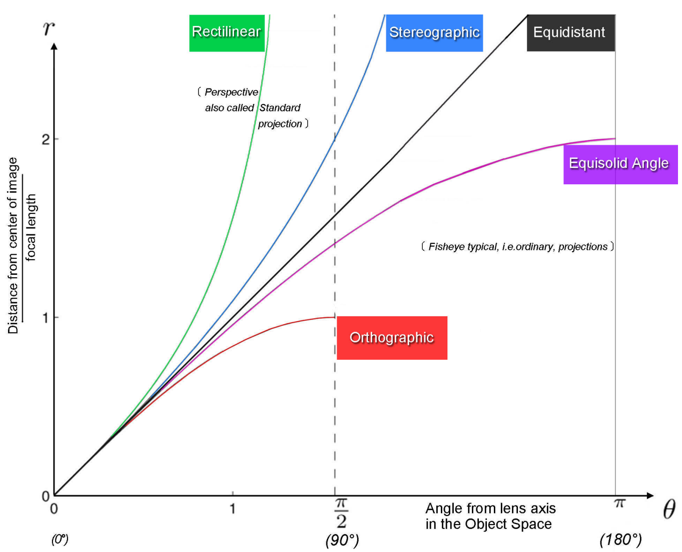
$$\pi^{-1}(\mathbf{u},\mathbf{i}) = \begin{bmatrix} \sin(\theta^*)\, \frac{m_x}{r_u} \\ \sin(\theta^*)\, \frac{m_y}{r_u} \\ \cos(\theta^*) \end{bmatrix}$$

$$m_x = \frac{u - c_x}{f_x},$$

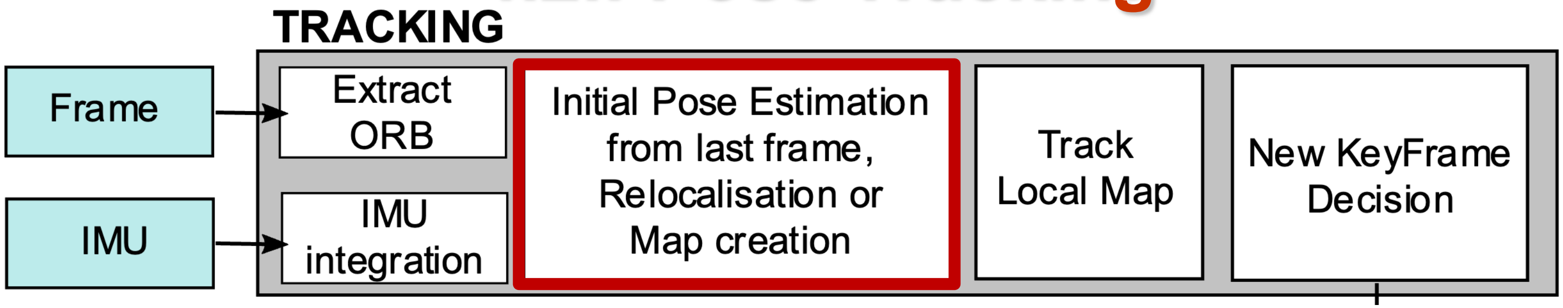
$$m_y = \frac{v - c_y}{f_y},$$

$$r_u = \sqrt{m_x^2 + m_y^2},$$

$$\theta^* = d^{-1}(r_u),$$

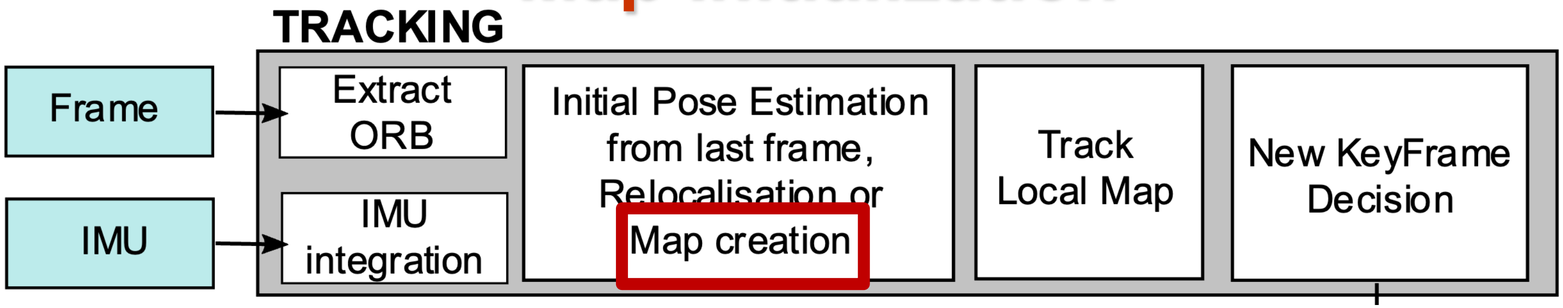


4.2.f Pose Tracking



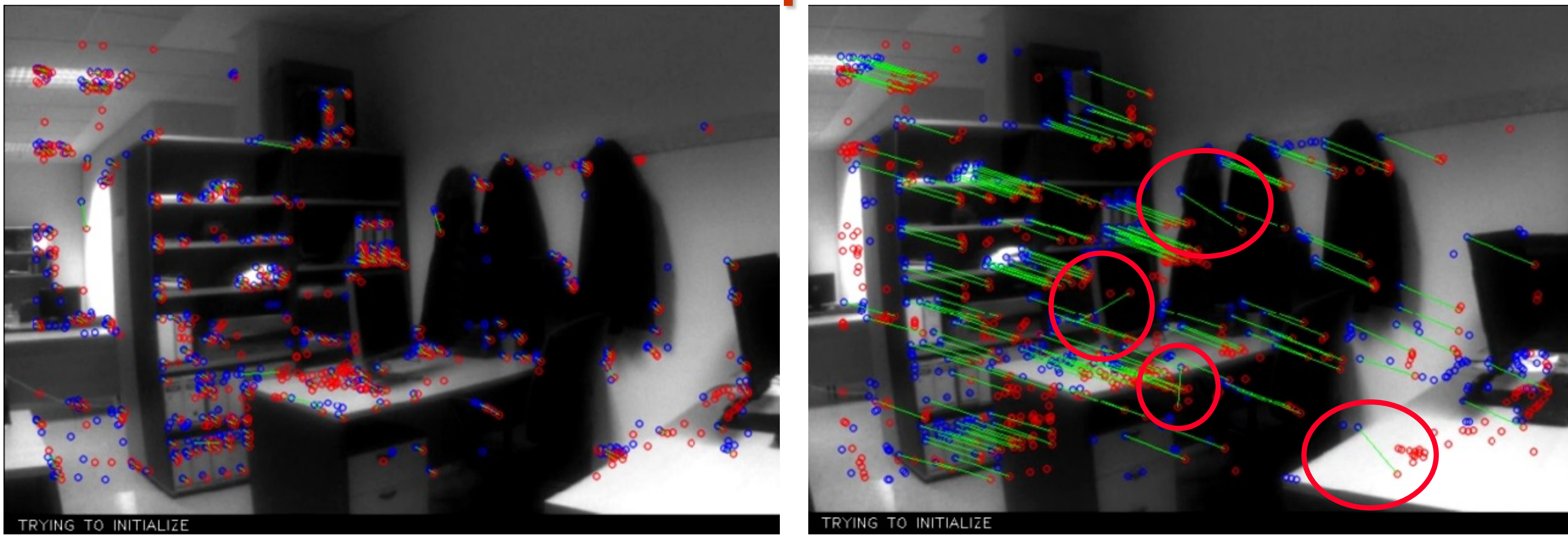
1. If SLAM is not initialized:
   - Automatic map creation
2. If tracking failed in last image:
   - Relocalisation or new map creation
3. Normal tracking:
   - Pose estimation from last frame (short-term DA)
   - Track local map: pose correction (mid-term DA)
   - New Keyframe decision

Map Initialization



- Monocular:
  - **Homography** / **Fundamental matrix** computed from **2 views**
  - Up-to-scale map
- Stereo and RGB-D :
  - Initialization from **1 image** with sensor depths
  - Real-scale map

Monocular Map Initialization



- From 2 Frames, match ORB features (brute force)
- Compute in parallel an Homography and a Fundamental matrix with RANSAC:

$$\mathbf{x}_c = \mathbf{H}_{cr}\,\mathbf{x}_r \qquad \mathbf{x}_c^T\,\mathbf{F}_{cr}\,\mathbf{x}_r = 0$$

- Select model according to the following score: $S_M = \sum_i \left(\rho_M\left(d^2_{cr}(\mathbf{x}^i_c, \mathbf{x}^i_r, M)\right) + \rho_M\left(d^2_{rc}(\mathbf{x}^i_c, \mathbf{x}^i_r, M)\right)\right)$
- Reconstruct cameras and map points
- Run a Bundle Adjustment

Monocular Map Initialization

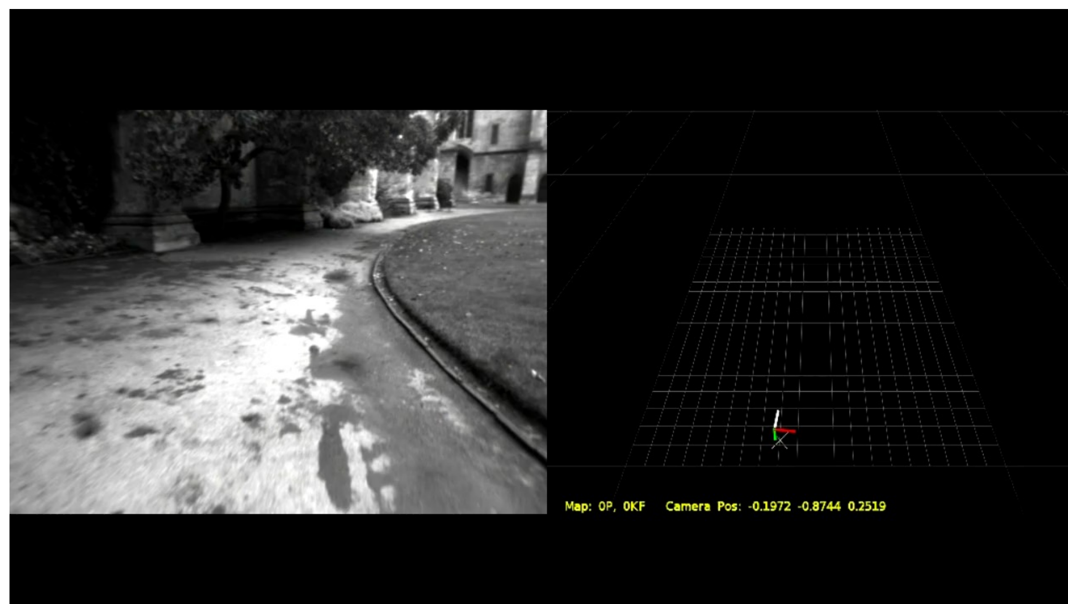
Model Selection

Homography
(Planar, Low Parallax)

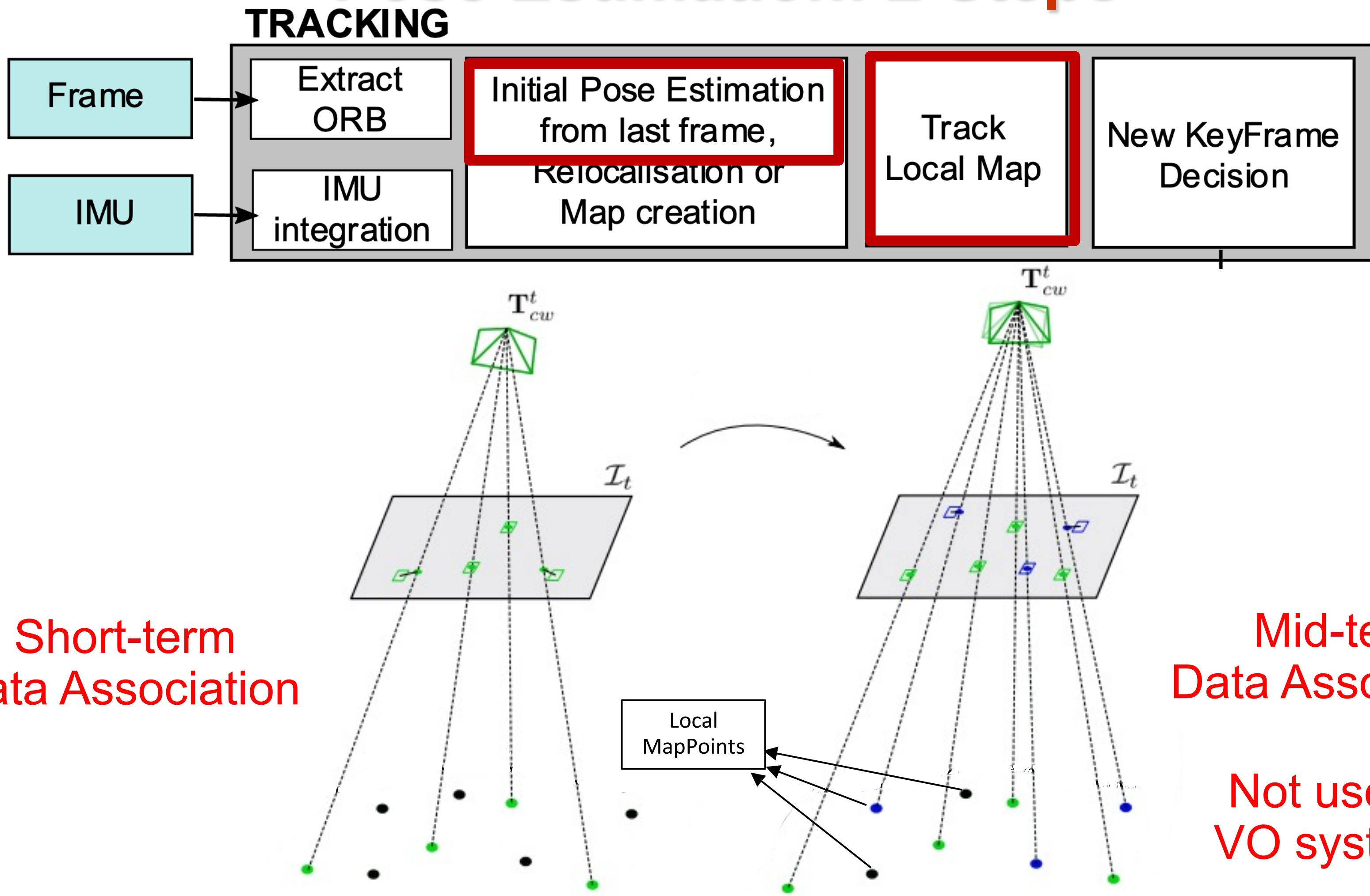
Fundamental Matrix
(General)

ORB-SLAM

PTAM



Pose Estimation: 2 Steps



Short-term
Data Association

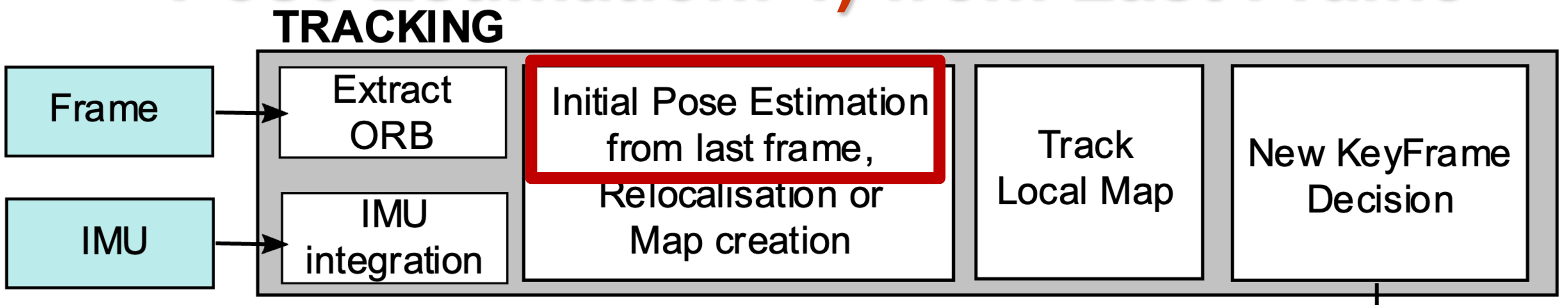
Mid-term
Data Association

Not used by
VO systems!!

First step:
Track from
previous Frame

Second step:
Match points from
the local map

Pose Estimation: 1) from Last Frame

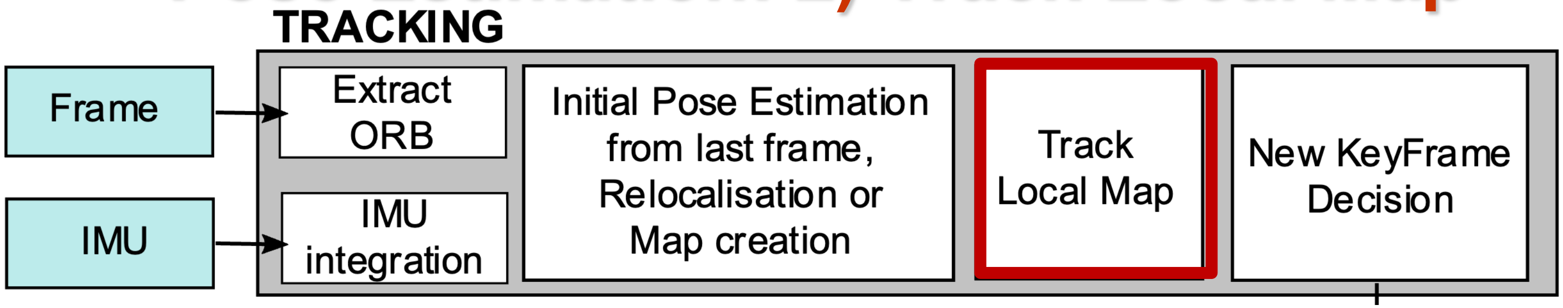


First estimation of the current camera pose:

- Predict the current camera pose using IMU integration or constant velocity model
- Project points seen in last frame and search for ORB matches (15-30 pixels window & Hamming < 100)
- Run first pose-only optimization

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}}{\operatorname{argmin}} \sum_j \rho\left(\left\|\mathbf{u}_{ij} - \pi_i\left(\mathbf{R}_{iw}\mathbf{x}_{wj} + \mathbf{t}_{iw}\right)\right\|^2_{\Sigma_{ij}}\right)$$

Pose Estimation: 2) Track Local Map



Second step to refine the camera pose:

- Find points in local map that can be visible in the current frame but have not been tracked
- Project them in the current frame and search for ORB matches (4 pixels window & Hamming < 100)
- Run second pose-only optimization

$$\{\mathbf{R}_{iw}, \mathbf{t}_{iw}\}^* = \underset{\mathbf{R}_{iw}, \mathbf{t}_{iw}}{\operatorname{argmin}} \sum_j \rho \left(\left\| \mathbf{u}_{ij} - \pi_i \left(\mathbf{R}_{iw} \mathbf{x}_{wj} + \mathbf{t}_{iw} \right) \right\|^2_{\Sigma_{ij}} \right)$$

- For inliers, update median ORB descriptor

Tracking: Occlusion Handling

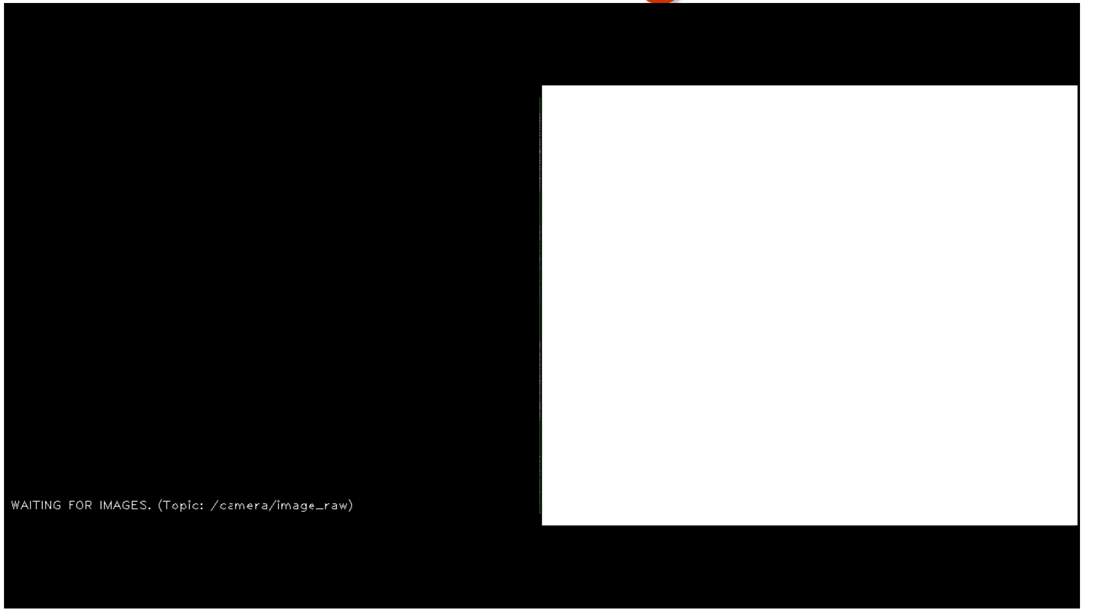
<u>Covisibility Graph</u>

Track only a local map (potentially visible)

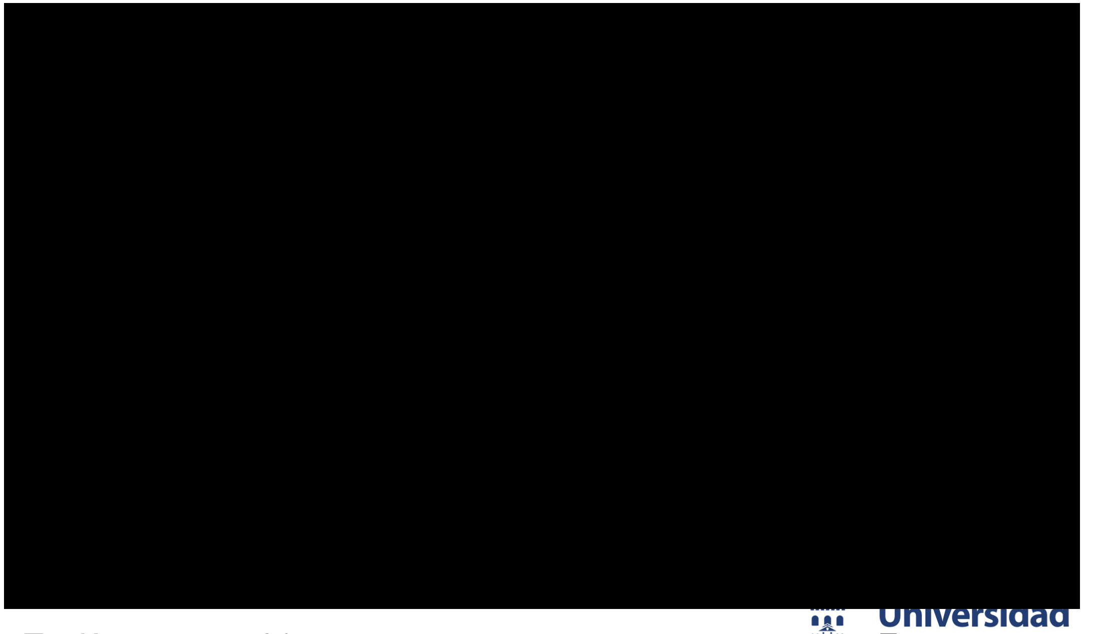
<u>Point Viewing Direction</u>

Do not project if further than 60º

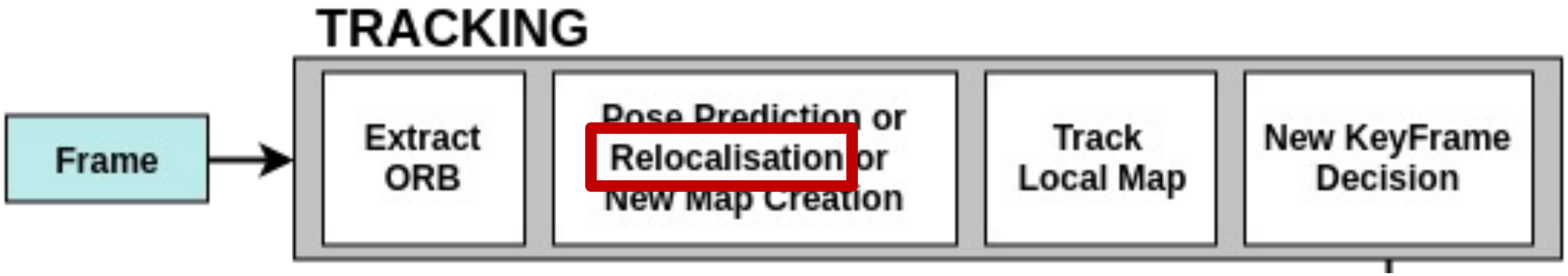
ORB-SLAM



PTAM



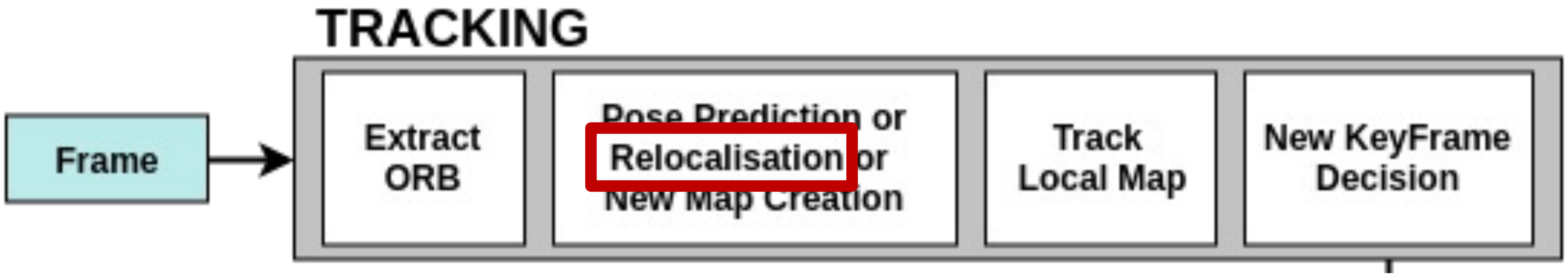
Tracking Failure: Relocalisation



When the camera tracking fails, enter relocalisation mode:

- Search the Bag-of-Words database with the current Frame, to find the most similar Keyframe
- Match ORB features between Frame (2D points) and Keyframe (3D points)
- Run PnP with RANSAC to recover the camera pose
- If a pose is found, run pose-only optimization
- And track local map

Tracking Failure: Relocalisation

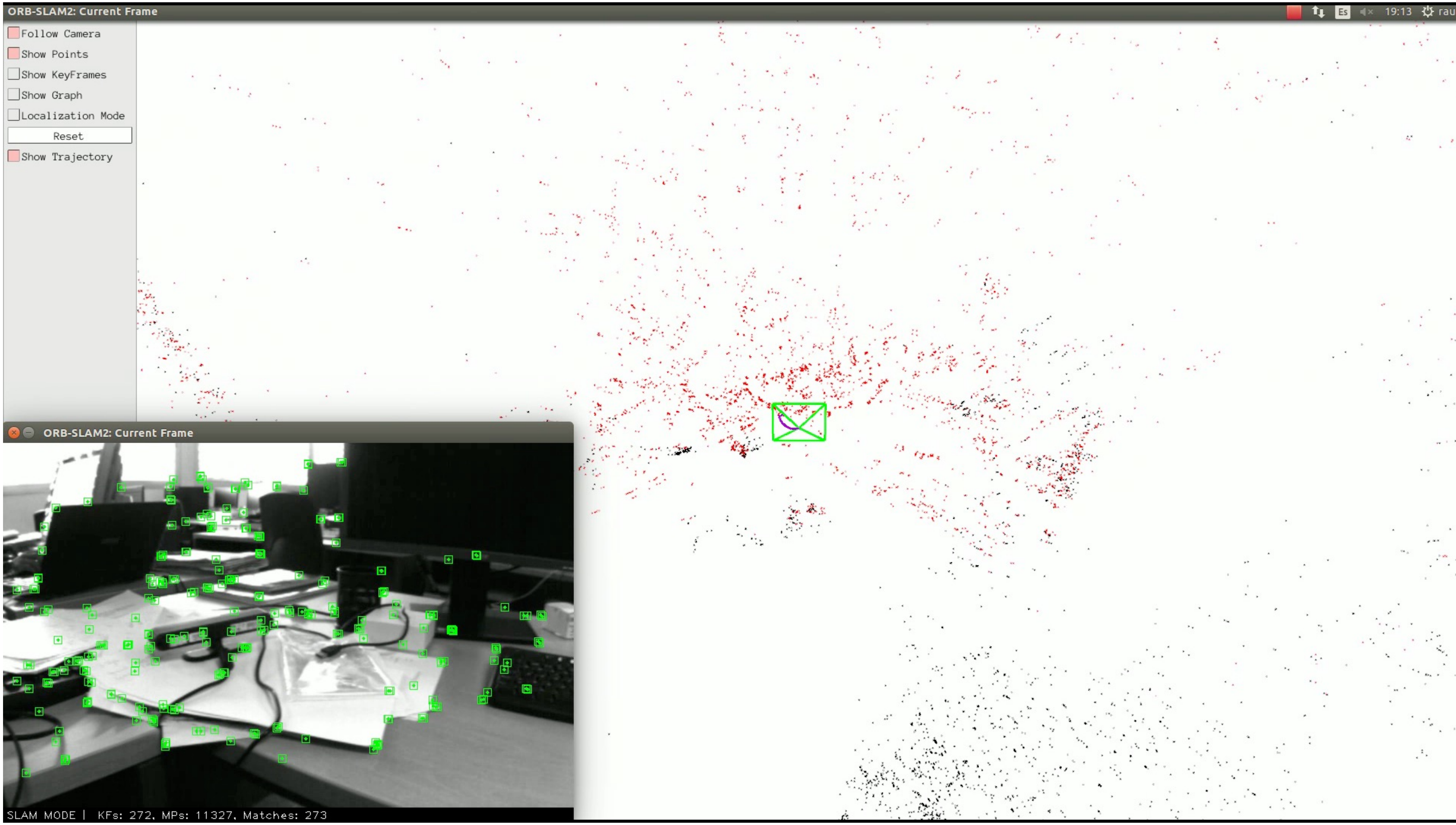


We solve de PnP problem with the MLPnP algorithm:

- Independent of the camera model
- Uses projection rays to solve the camera pose
  - Valid for pinhole and fisheye cameras

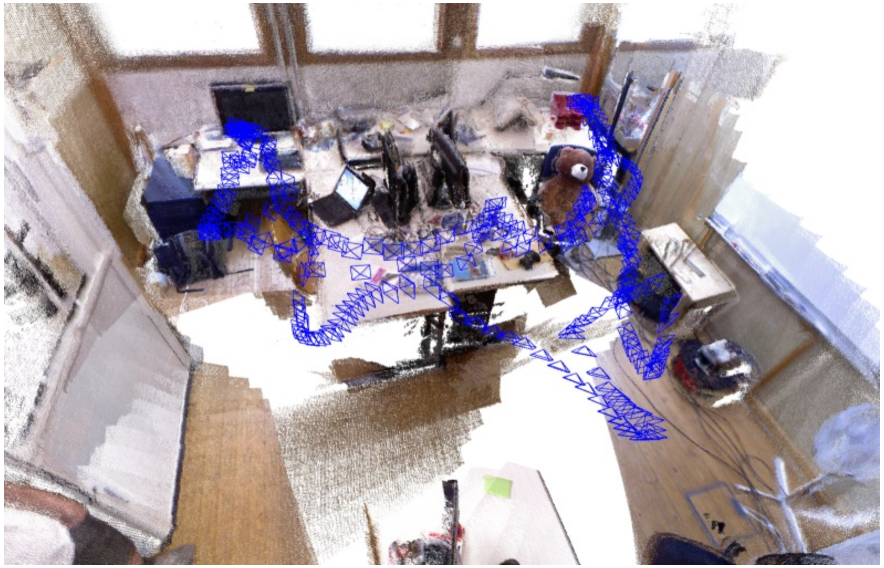
S. Urban, J. Leitloo, S. Ninz: MLPnP – A Real-Time Maximum Likelihood Solution to the Perspective-n-Point Problem, Arxiv preprint, 2016

ORBSLAM Robust Tracking



4.3 Visual SLAM: Mapping

Juan D. Tardós, Juan J. Gómez
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



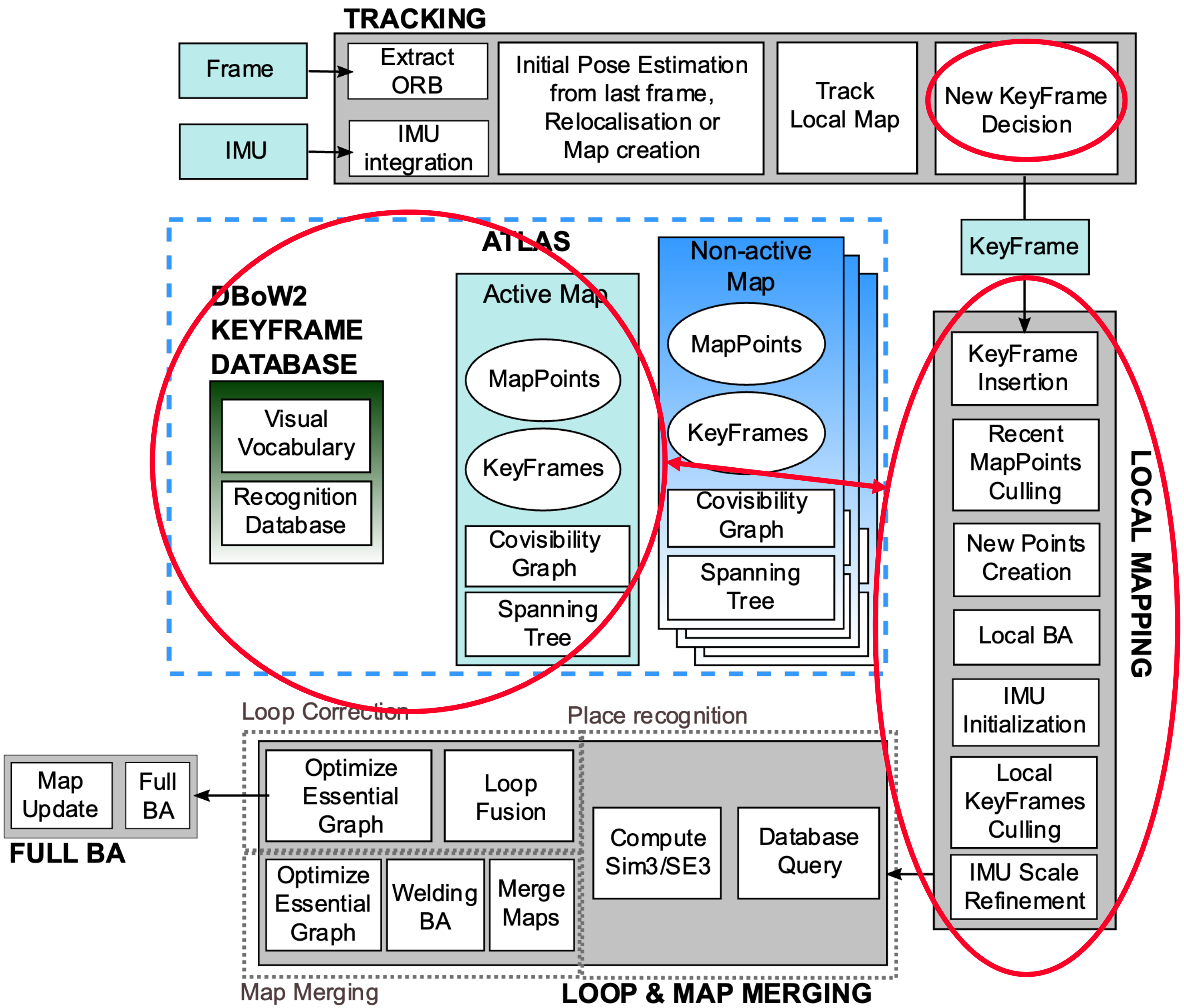
Lesson 4. Visual SLAM

3. Tracking
    a. Overview
    b. KeyFrame & MapPoint creation
    c. KF & Point management
    d. Local Bundle Adjustment
    e. Running and tuning

Readings:

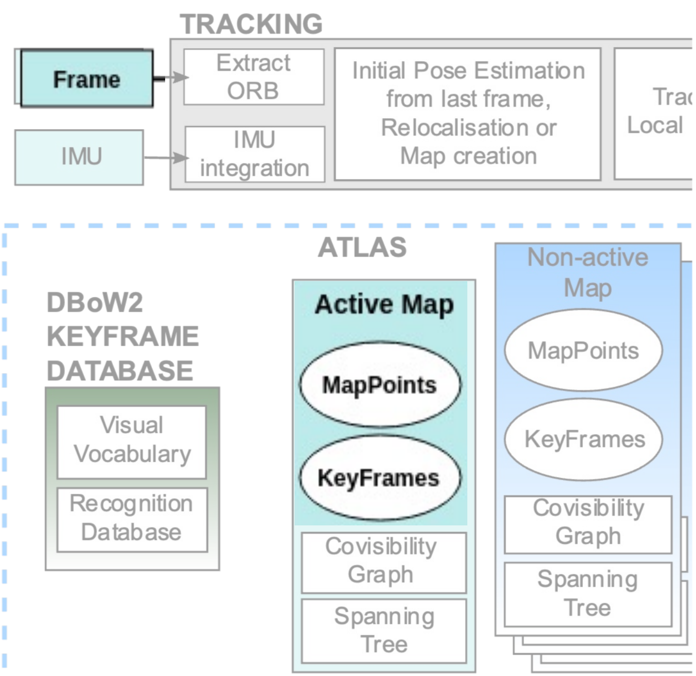
- Sections III, & VI of ORB-SLAM paper

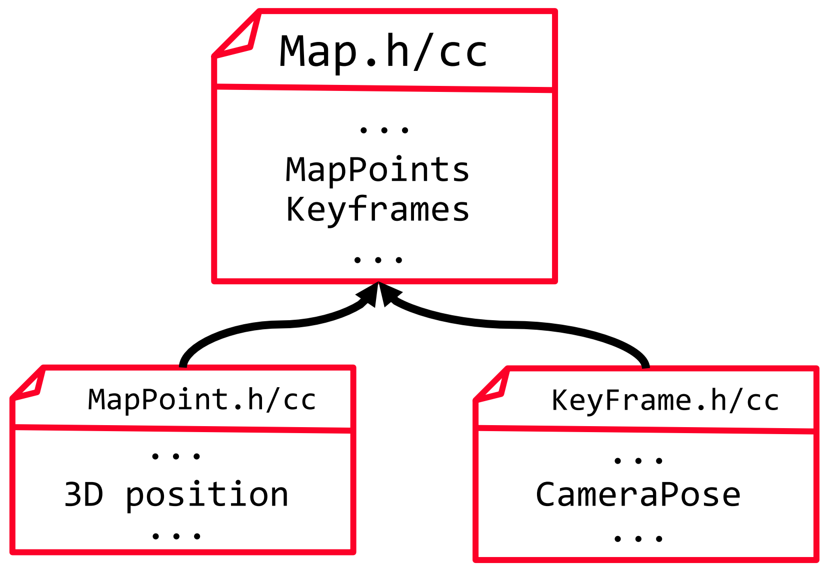
4.3.a Overview



Map Components

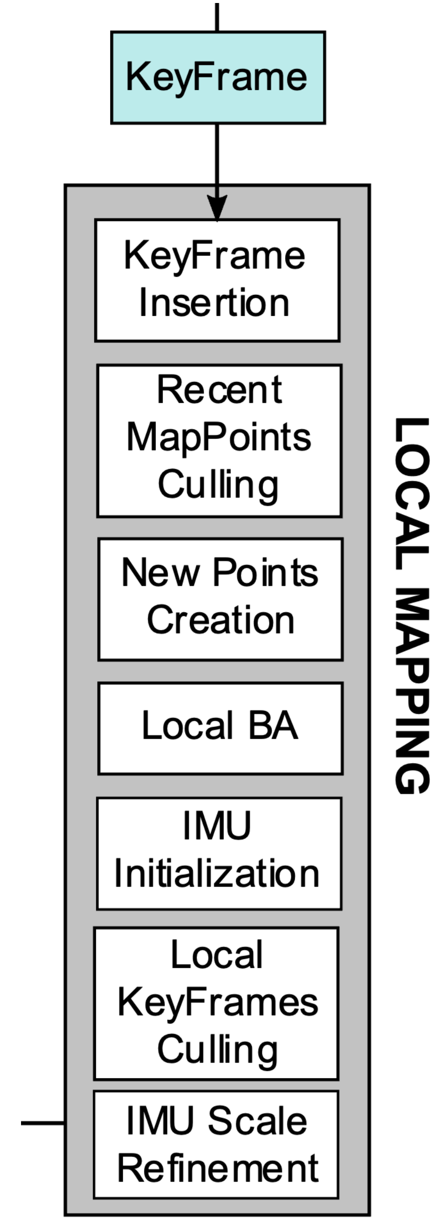
- Frames are the basic data structure to store image information: features, descriptors, camera pose, …
- The map is composed by two elements:
  - Map Points: represents a 3D landmark
  - Key Frames: a Frame that carries high visual innovation



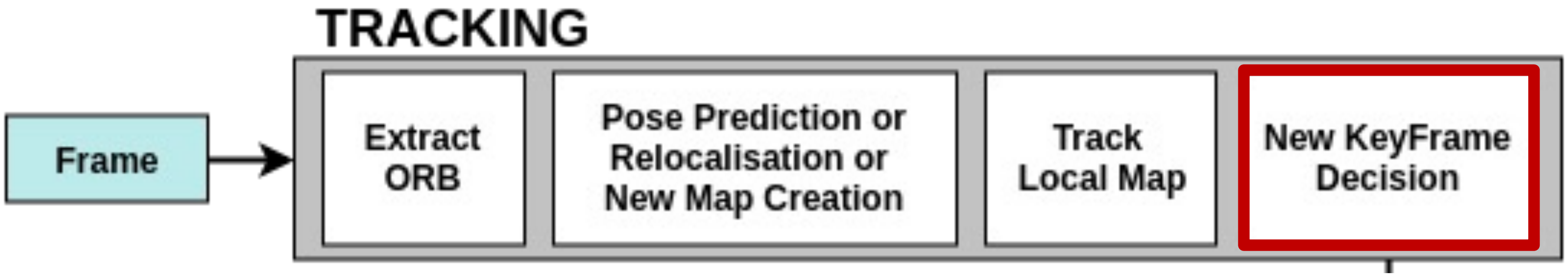


Local Mapping Overview

- Local Mapping is the task in charge of:
  - Inserting new KeyFrames
  - Triangulate new MapPoints
  - Some map managing operations:
    - » Recent MapPoints culling
    - » Local KeyFrame Culling
  - Refine the local map by running a local Bundle Adjustment
- This thread runs at KeyFrame frequency (typically 0.5 - 2 KF/s)



4.3.b KeyFrame & MapPoint Creation



If the tracking was successful, decide if promoting the current Frame to Keyframe

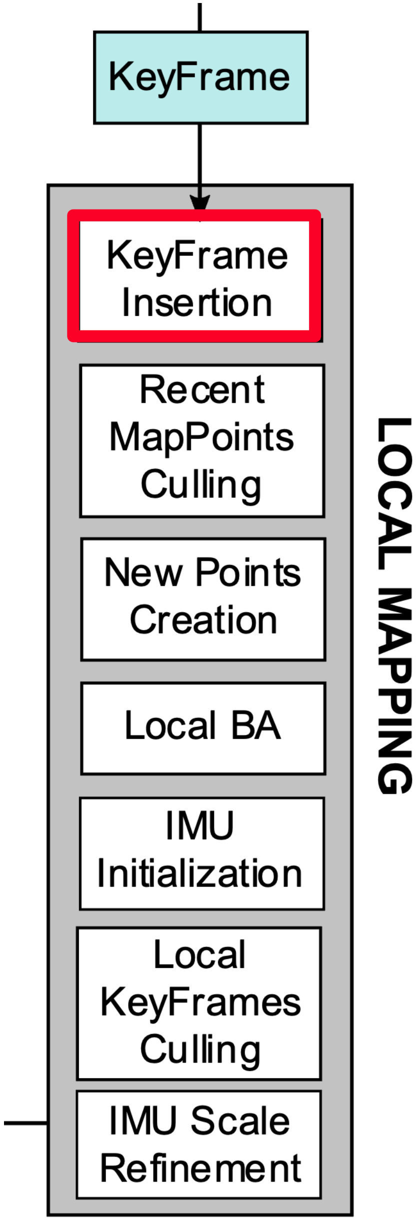
- Insert as many Keyframes as possible to make the tracking robust to fast camera motions
- Redundant Keyframes will be removed later

Insert a Keyframe if:

- Tracking is weak (low number of tracked points) **or**
- More than a certain number of frames have passed from last Keyframe **or**
- Local Mapping is idle

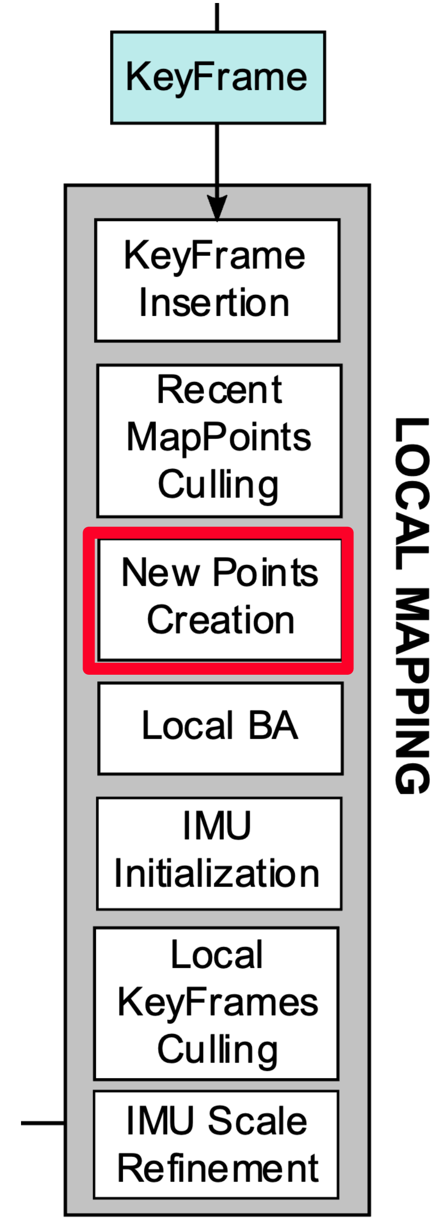
KeyFrame Insertion

- Add KeyFrame to the map
- Compute its Bag of Words and insert it into the recognition database to be used for future loop closing and relocation



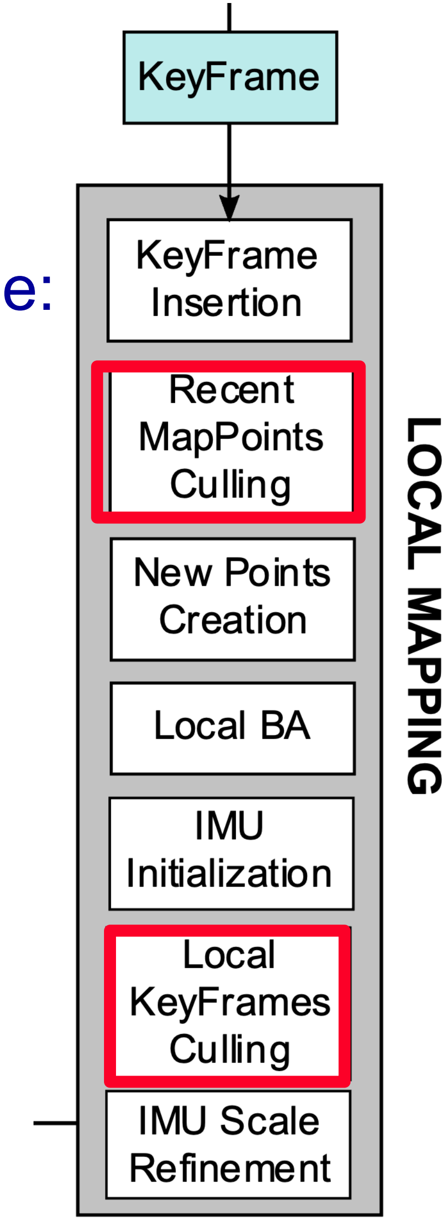
New MapPoints Creation

- As KeyFrames carry novel visual information, try to **add points to the map**
- **Match** features in the new Keyframe with features in covisible KeyFrames
- And **triangulate** new MapPoints
- Checks for new points:
  - enough parallax
  - positive depth
  - small reprojection error



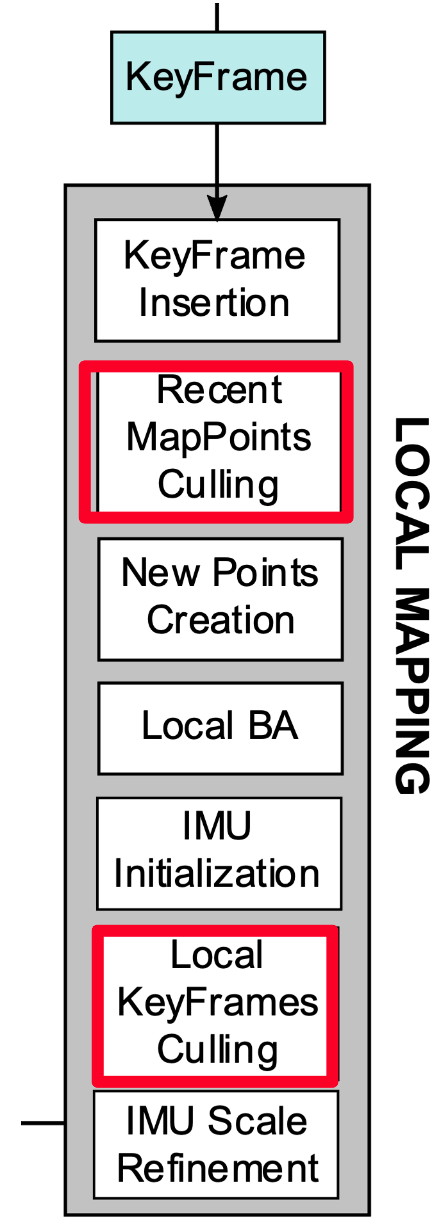
4.3.c Point & KF Managing

- Every time a new KeyFrame is inserted,…
- Try to keep the local map as small as possible:
  - Removing duplicated or unused MapPoints due to bad feature matching
  - Removing redundant KeyFrames due to the fast KeyFrame insertion policy



Point & KF Managing

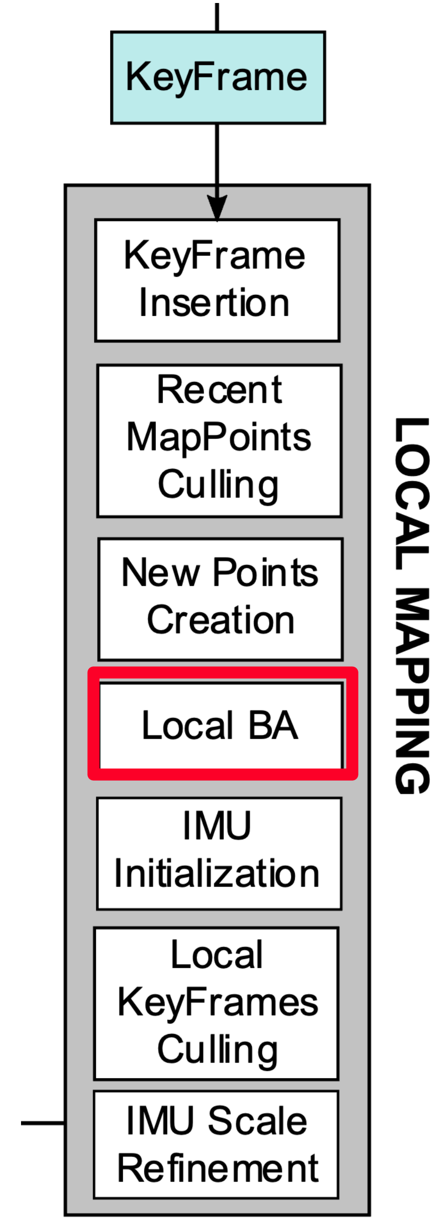
- A MapPoint is considered unused if:
  - It has not been seen in the 2 following KeyFrames after its triangulation
  - It has not been matched in at least 25% of Frames that should see the point (the point lies inside the Field of View of the camera)
- A KeyFrame is considered redundant if:
  - At least a 90% of its matched MapPoints are seen by at least 3 other KeyFrames in the same or finer scale



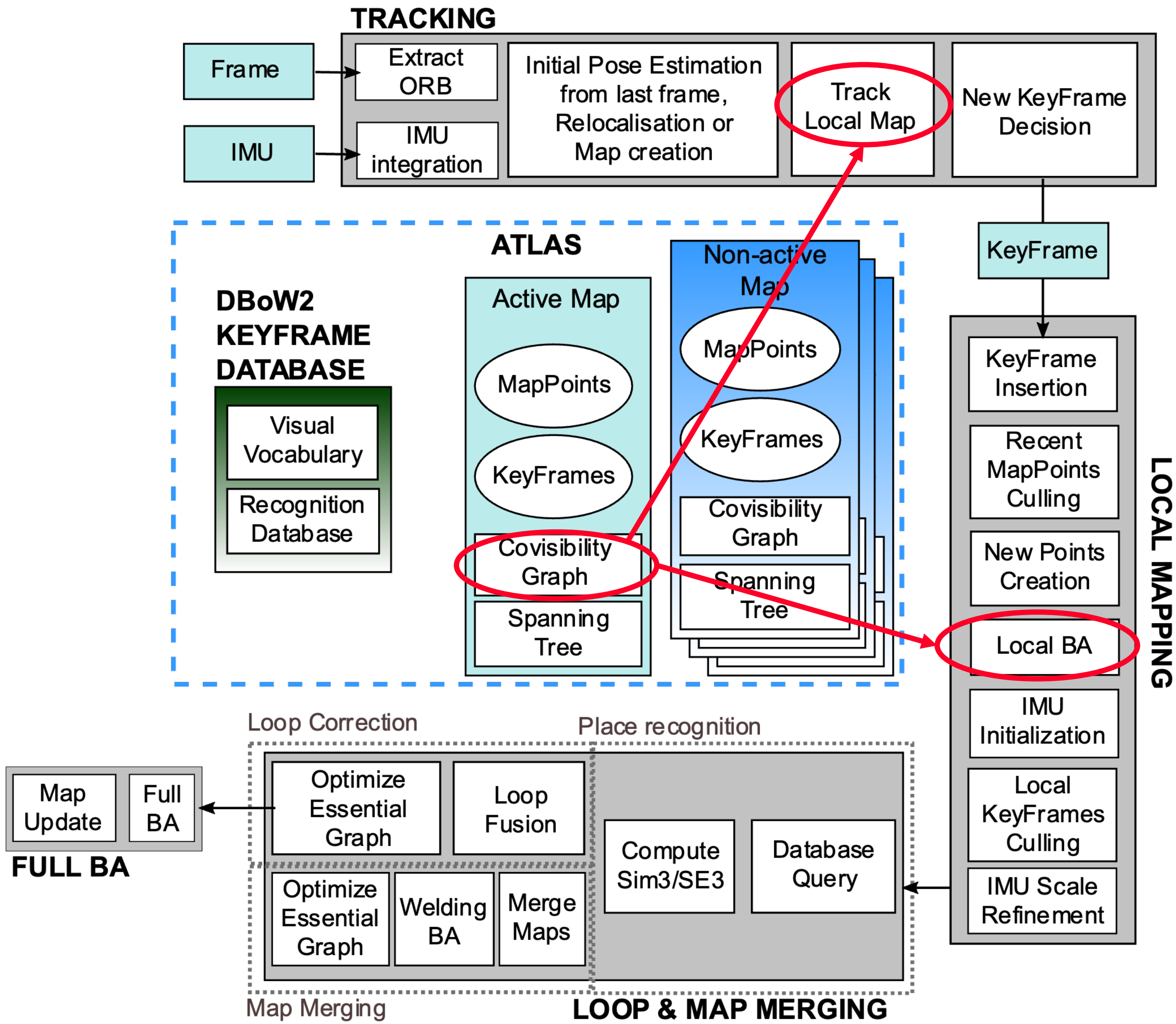
4.3.d Local Bundle Adjustment

- After point creation, perform a Local Bundle Adjustment to refine KeyFrame poses and point positions
- The local map is composed by the current KeyFrame, its **covisible** KeyFrames and all the points they see
- The rest of the map remains fixed to avoid drift

- Local BA complexity depends on KeyFrame density, not in the total map size!!

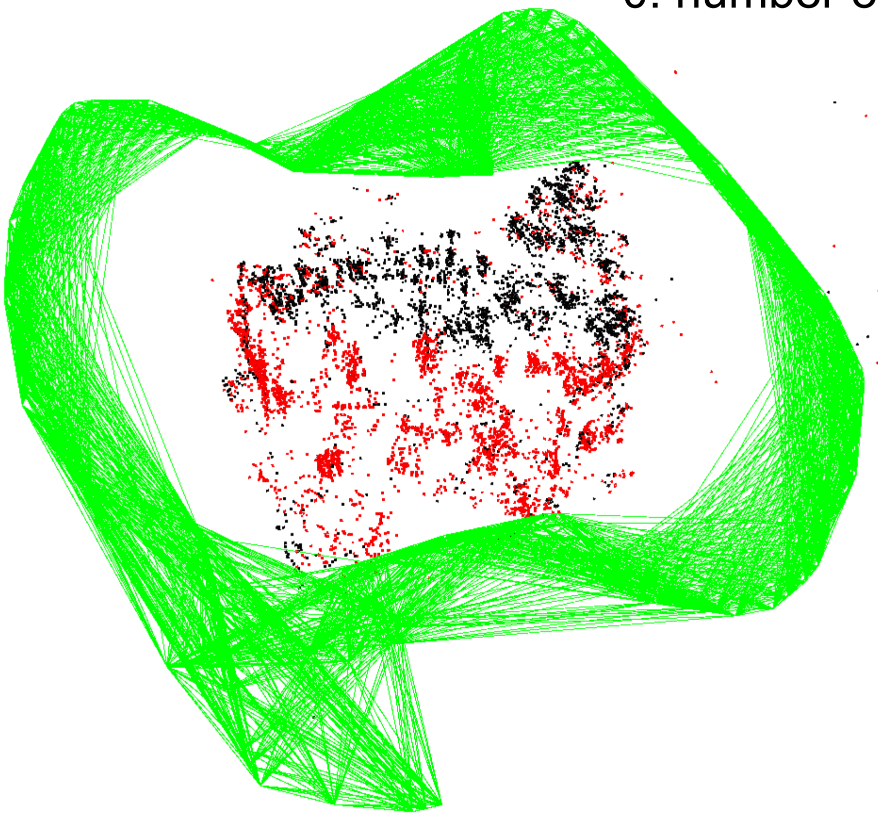


Local Mapping



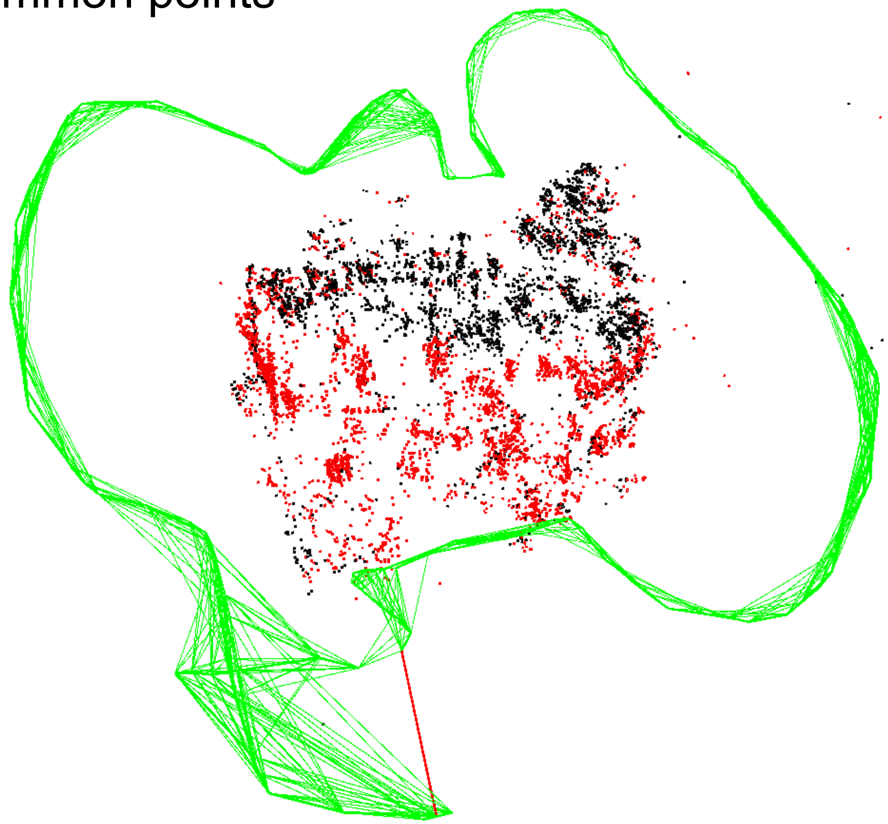
Covisibility Graph and Essential Graph

θ: number of common points



$\theta_{min} = 15$

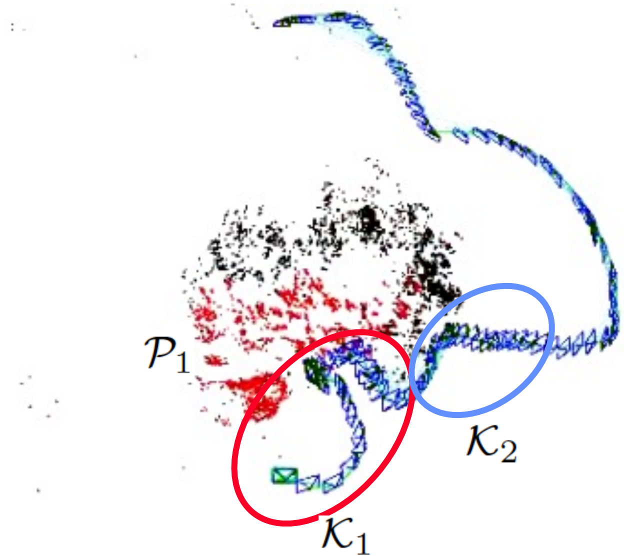
Used for Local BA,
instead of sliding window

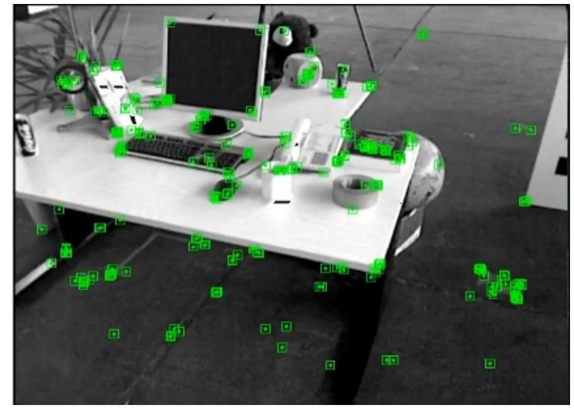


$\theta_{min} = 100$

Used for Loop Correction

Local Bundle Adjustment (BA)





Local Map:

- $\mathcal{K}_1$ Current Keyframe & Covisibles
- $\mathcal{P}_1$ Points seen by $\mathcal{K}_1$ (in red)
- $\mathcal{K}_2$ Rest of Keyframes that see $\mathcal{P}_1$

Local Bundle Adjustment

$$\{\mathbf{T}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{K}_1,\ j \in \mathcal{P}_1\}^* = \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{k \in \{\mathcal{K}_1, \mathcal{K}_2\},\, j \in \mathcal{P}_1} \rho\left(\left\| \mathbf{u}_{kj} - \pi_k\left(\mathbf{T}_{kw}\mathbf{x}_{wj}\right)\right\|^2_{\Sigma_{kj}} \right)$$

Variable

Constant, they anchor the map

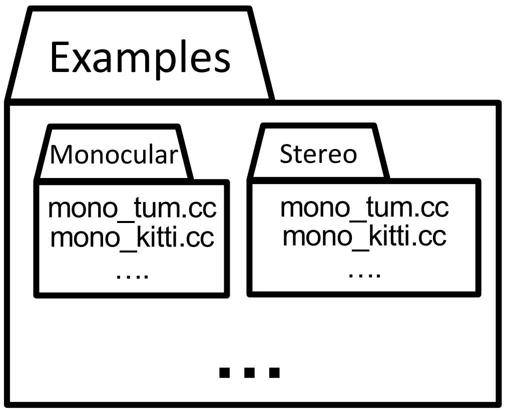
ORBSLAM3 Tracking & Mapping Times (ms)

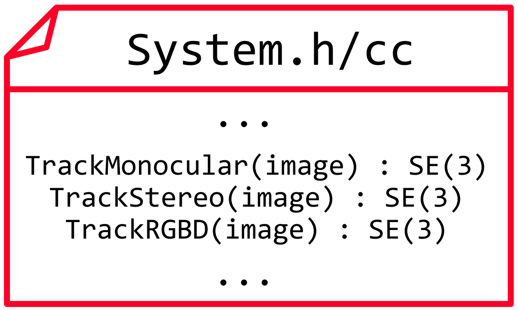
	System	ORB-SLAM2	ORB-SLAM3	ORB-SLAM3	ORB-SLAM3	ORB-SLAM3
Settings	Sensor	Stereo	Monocular	Stereo	Mono-Inertial	Stereo-Inertial
	Resolution	752×480	752×480	752×480	752×480	752×480
	Cam. FPS	20Hz	20Hz	20Hz	20Hz	20Hz
	IMU	-	-	-	200Hz	200HZ
	ORB Feat.	1200	1000	1200	1000	1200
	RMS ATE	0.035	0.029	0.028	0.021	0.014
Tracking	Stereo rect.	3.07±0.80	-	1.32±0.43	-	1.60±0.74
	ORB extract	11.20±2.00	12.40±5.10	15.68±4.74	11.98±4.78	15.22±4.37
	Stereo match	10.38±2.57	-	3.35±0.92	-	3.38±1.07
	IMU integr.	-	-	-	0.18±0.11	0.22±0.20
	Pose pred	2.20±0.72	1.87±0.68	2.69±0.85	0.09±0.41	0.15±0.71
	LM Track	9.89±4.95	4.98±1.65	6.31±2.85	8.22±2.52	11.51±3.33
	New KF dec	0.20±0.43	0.04±0.03	0.12±0.19	0.05±0.03	0.18±0.25
30-40 fps	Total	37.87±7.49	21.52±6.45	31.48±5.80	23.22±14.98	33.05±9.29
Mapping	KF Insert	8.72±3.60	9.25±4.62	8.03±2.96	13.17±7.43	8.53±2.17
	MP Culling	0.25±0.09	0.09±0.04	0.32±0.15	0.07±0.04	0.24±0.24
	MP Creation	36.88±14.53	22.78±8.80	18.23±9.84	30.19±12.95	23.88±9.97
	LBA	139.61±124.92	216.95±188.77	134.60±136.28	121.09±44.81	152.70±38.37
	KF Culling	4.37±4.73	18.88±12.217	5.49±5.09	26.25±17.08	11.15±7.67
3-5 KFs/s	Total	173.81±139.07	266.61±207.80	158.84±147.84	191.50±80.54	196.61±54.52
Map Size	KFs	278	272	259	332	135
	MPs	14593	9686	14245	10306	9761

Intel Core i7-7700 @3.6 GHz, 32 GB

4.3.e Running and Tuning

- Create a main file where you define the SLAM system:
  - Create an ORB_SLAM3::System object
  - Input images to the system with one of the following methods:
    - » TrackMonocular(image)
    - » TrackStereo(image)
    - » TrackRGBD(image)
  - These methods return the camera pose prediction
- You must provide a calibration file according to your camera calibration

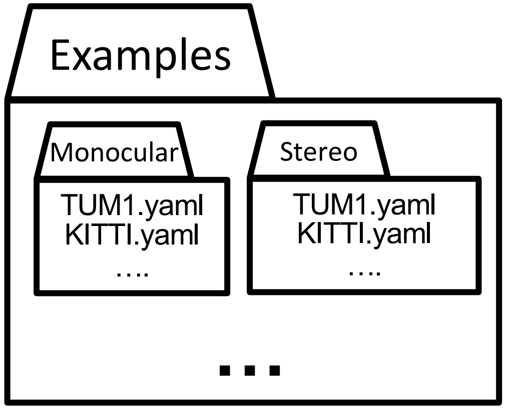




Calibration Files

These .yaml files contains:

- Camera parameters:
  - Type of camera model
  - Calibration parameters
  - Etc…
- ORB parameters:
  - Number of features to extract
  - Number of scales
  - Scale factor
- Some visualization parameters



Adjust them according to your needs!

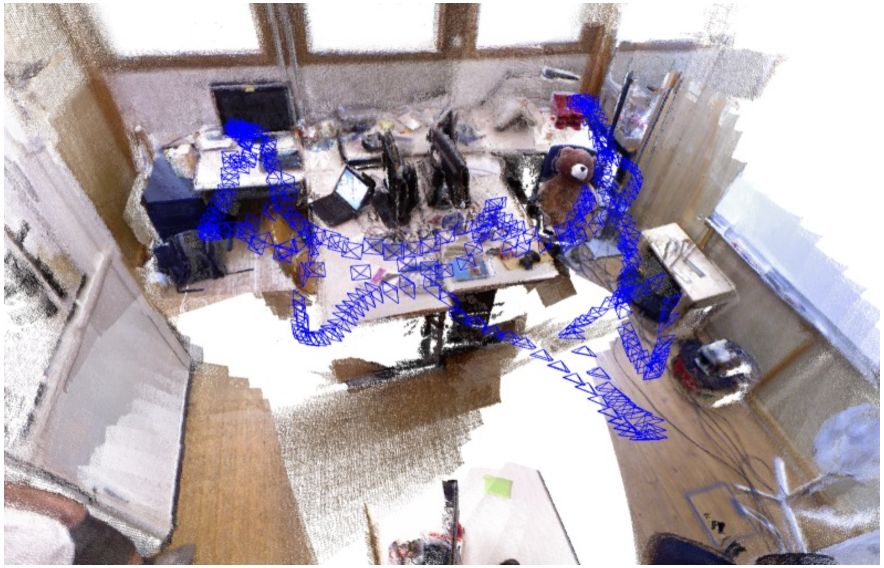
Things that can be Tuned up

Apart from the calibration file, you can tune up:

- Minimum parallax required to triangulate a point
- KeyFrame insertion policy
- ORB matching thresholds
- Number of iterations of the Bundle Adjustments
- Criteria for deleting MapPoints and KeyFrames
- …

4.4 Visual SLAM: Place Recognition

Juan D. Tardós, Richard Elvira
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



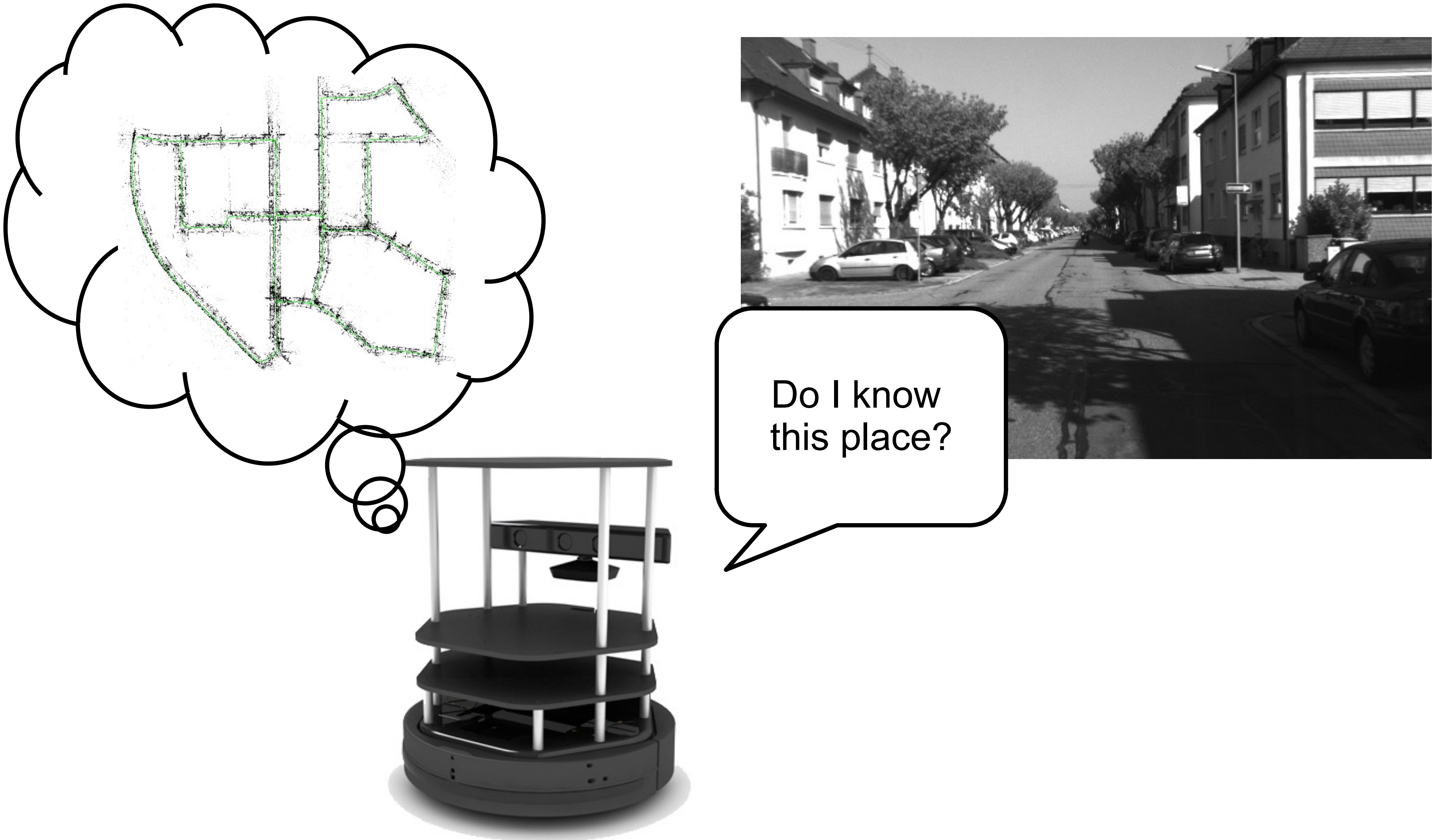
Lesson 4. Visual SLAM

4. Place Recognition
   a. Overview
   b. Bag of Words
   c. Relocation
   d. Loop closing

Readings:

- DBoW2 paper
- Section VII of ORB-SLAM paper
- Section VI of ORB-SLAM3 paper

4.4.a Place Recognition



Uses of Place Recognition

1. Relocalization (when tracking is lost)
   - Frame vs KeyFrame (**2D-3D** matches)
   - Compute camera pose with **PnP** algorithm
2. Loop Closing (correct the accumulated drift)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)
3. Map Merging (merge independent maps)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)

Why is Place Recognition Difficult?

- Is this a loop closure?

Likely algorithm answer:

YES **YES** **TRUE POSITIVE**

Why is Place Recognition Difficult?

- Is this a loop closure?

Scene 1430

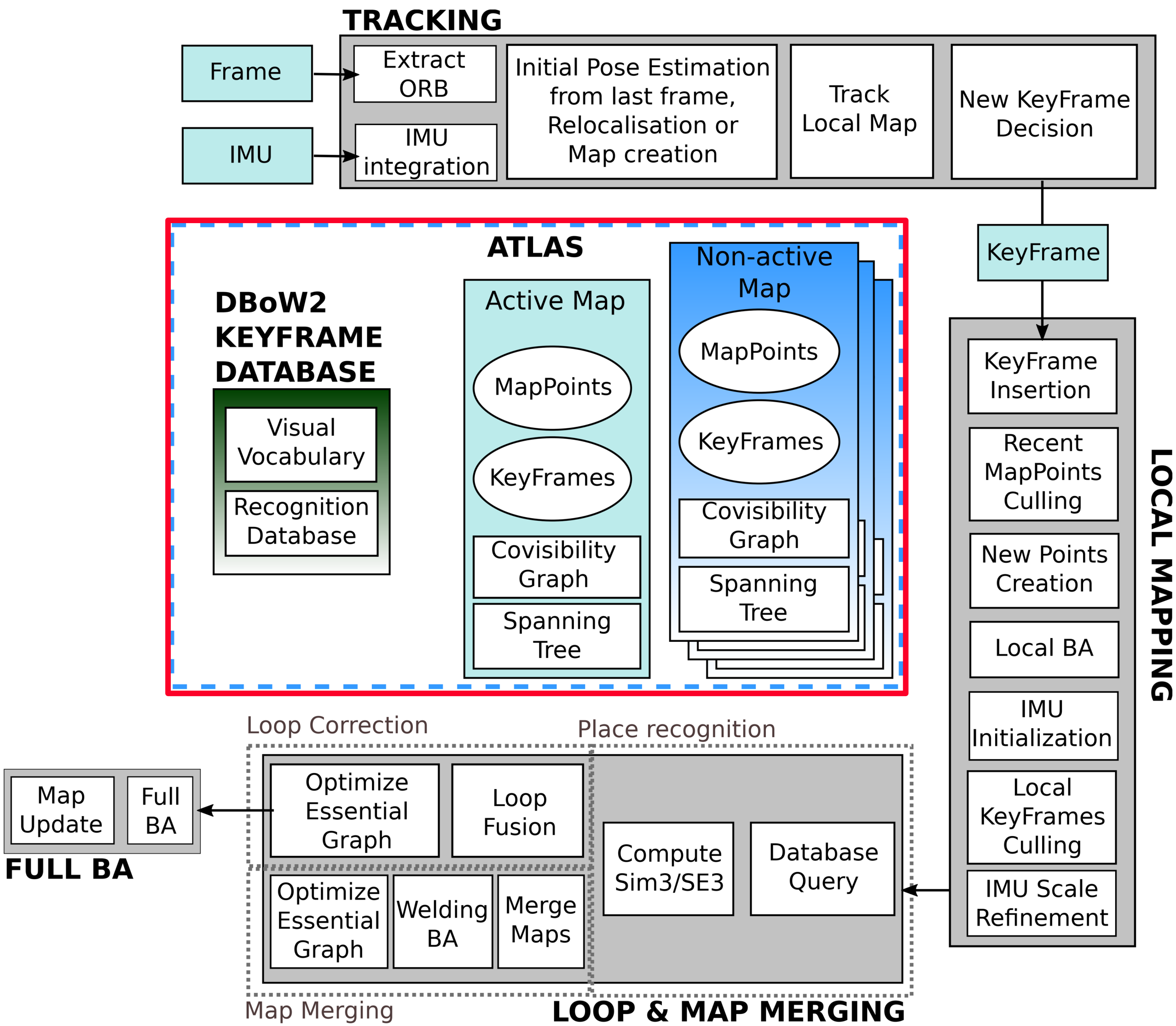
Scene 1244

Likely algorithm answer:

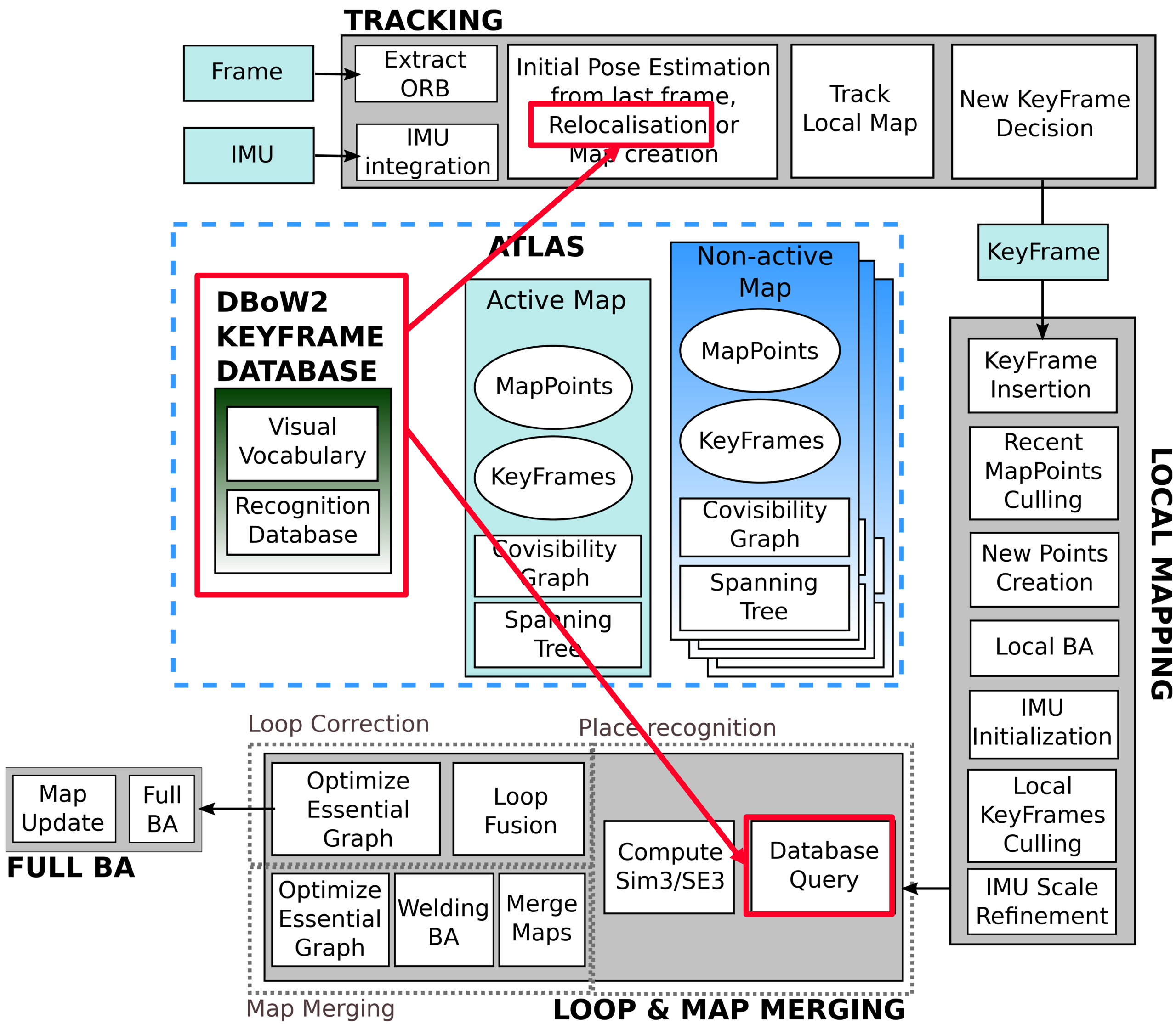
NO **YES** **FALSE POSITIVE**

Perceptual aliasing is common in indoor scenarios

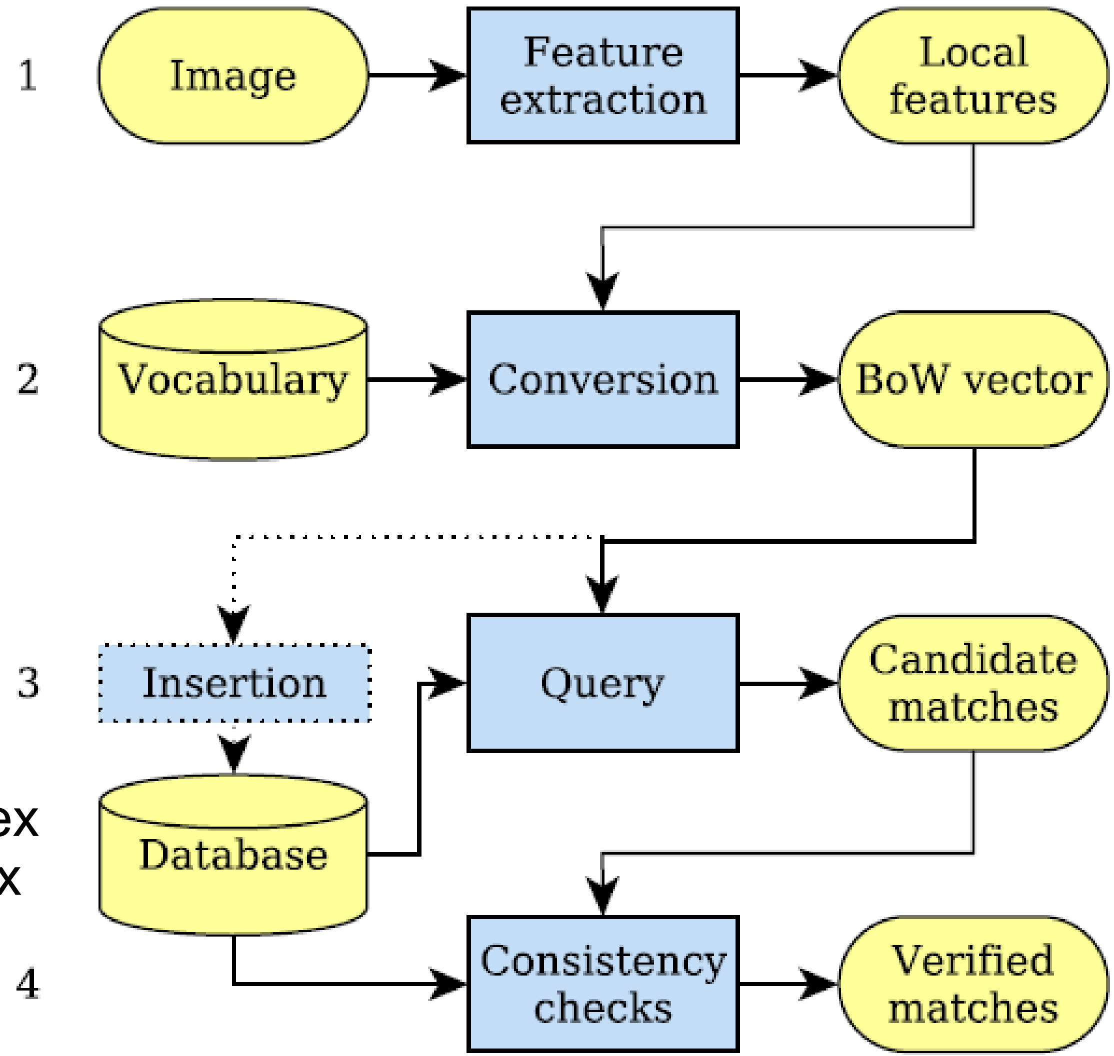
ORB-SLAM3: Multi-Map (Atlas)



ORB-SLAM3: Bag of Words



4.4.b Bag of Words Approach

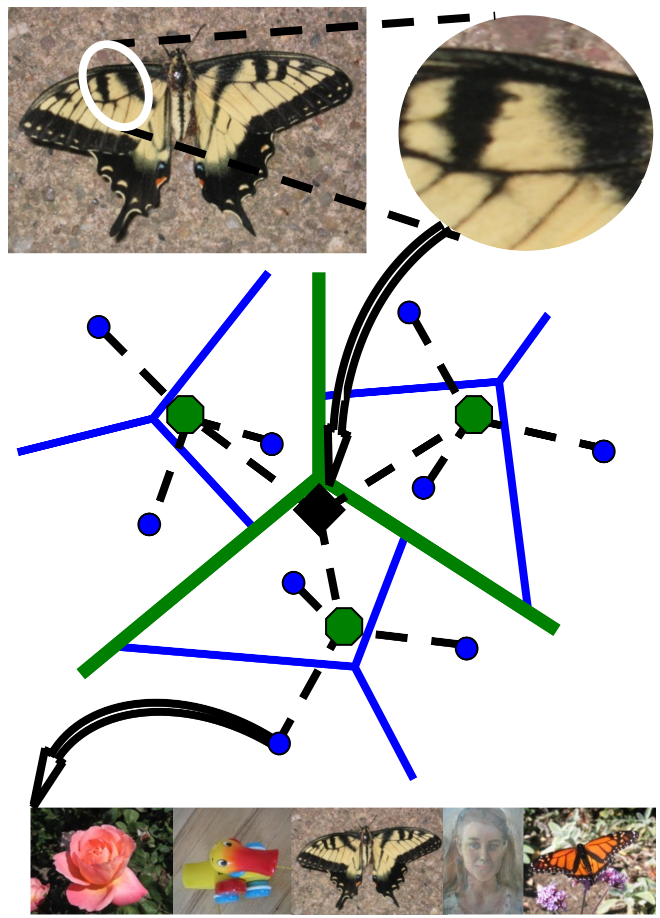


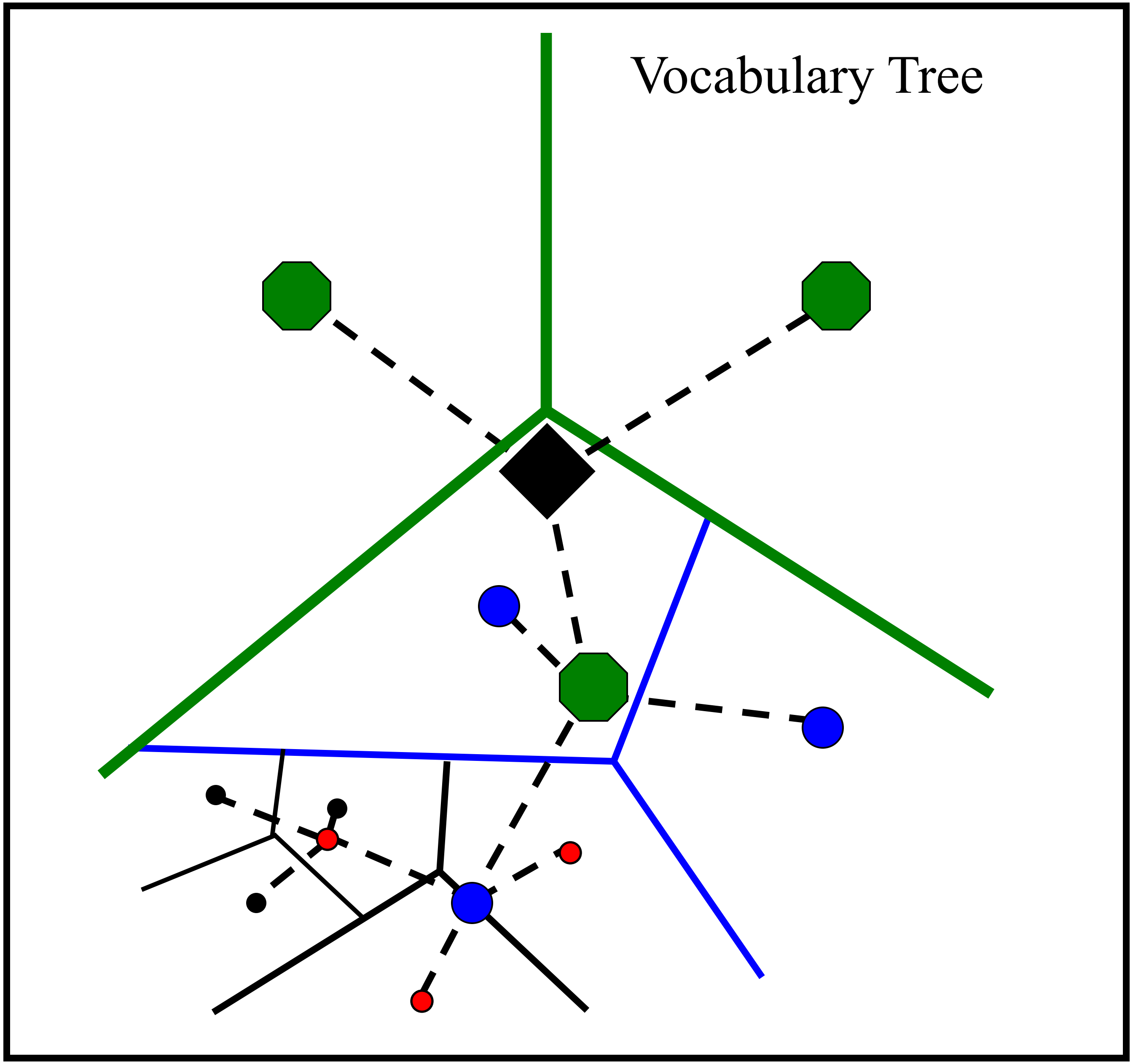
Binary Features
BRIEF, ORB

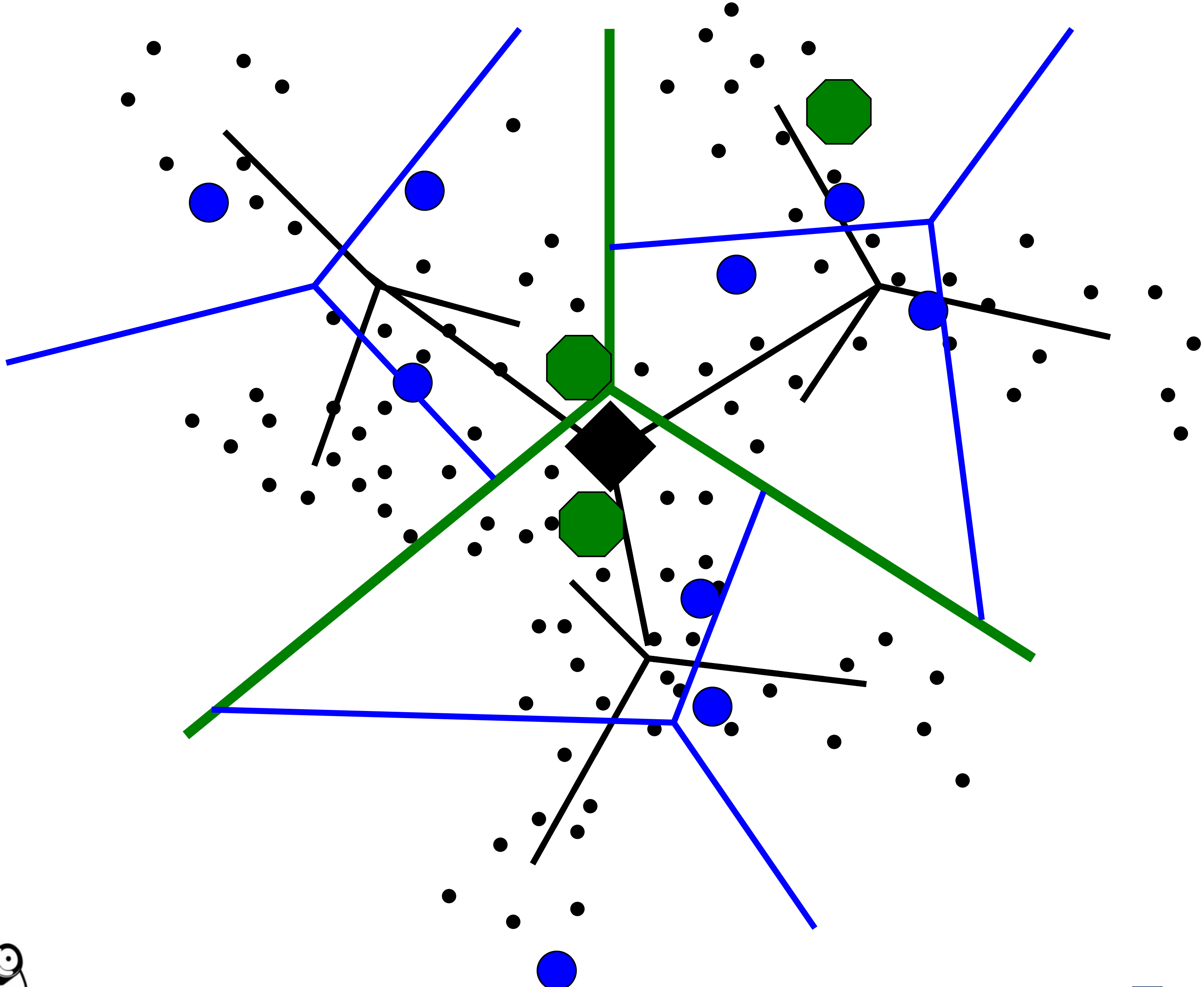
Inverse Index
Direct Index

Scalable Recognition with a Vocabulary Tree

David Nistér, Henrik Stewénius
CVPR 2006




Vocabulary Tree



DBoW2

- Similarity between two BOW vectors:

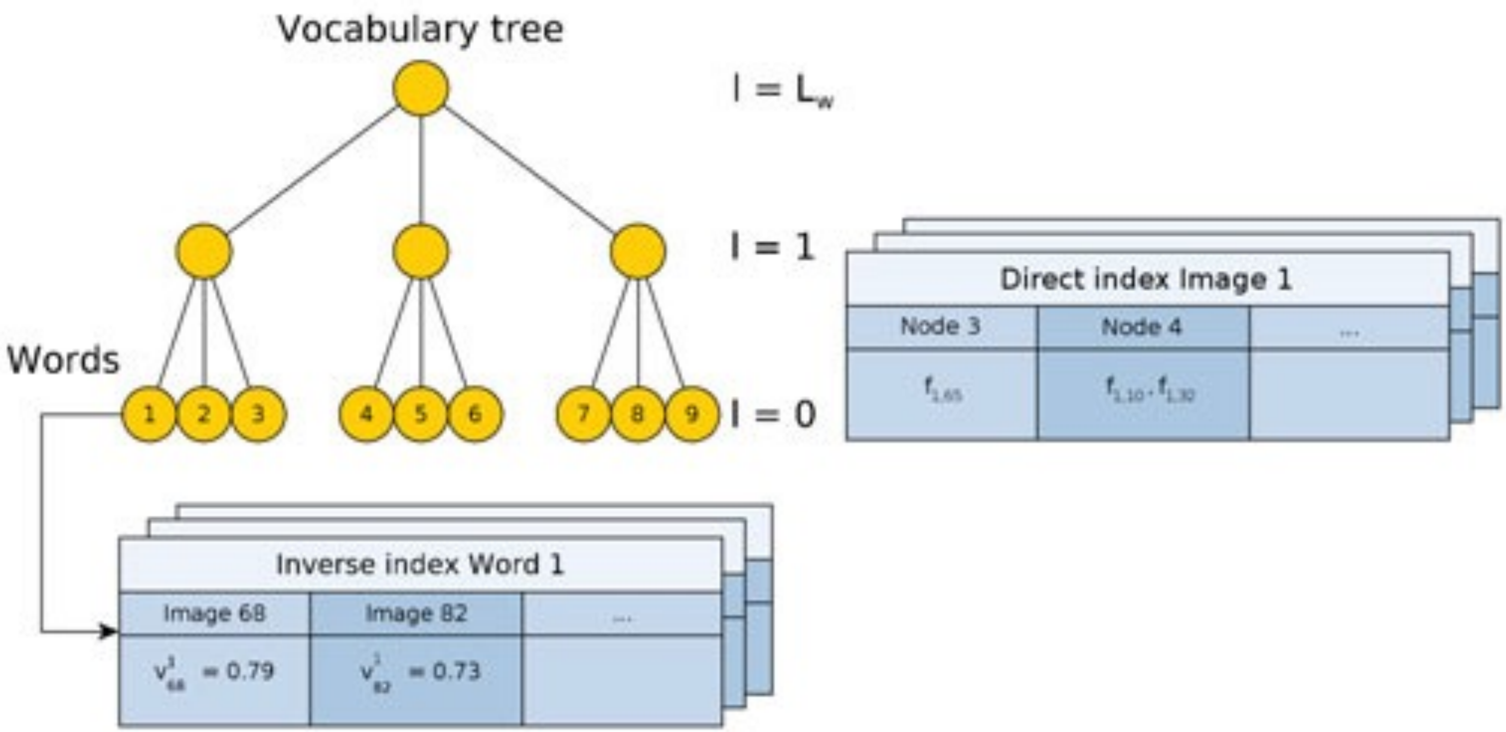
$$s(\mathbf{v}_1, \mathbf{v}_2) = 1 - \frac{1}{2}\left|\frac{\mathbf{v}_1}{|\mathbf{v}_1|} - \frac{\mathbf{v}_2}{|\mathbf{v}_2|}\right|$$

- Normalized similarity (comparised with previous image):

$$\eta(\mathbf{v}_t, \mathbf{v}_{t_j}) = \frac{s(\mathbf{v}_t, \mathbf{v}_{t_j})}{s(\mathbf{v}_t, \mathbf{v}_{t-\Delta t})}$$

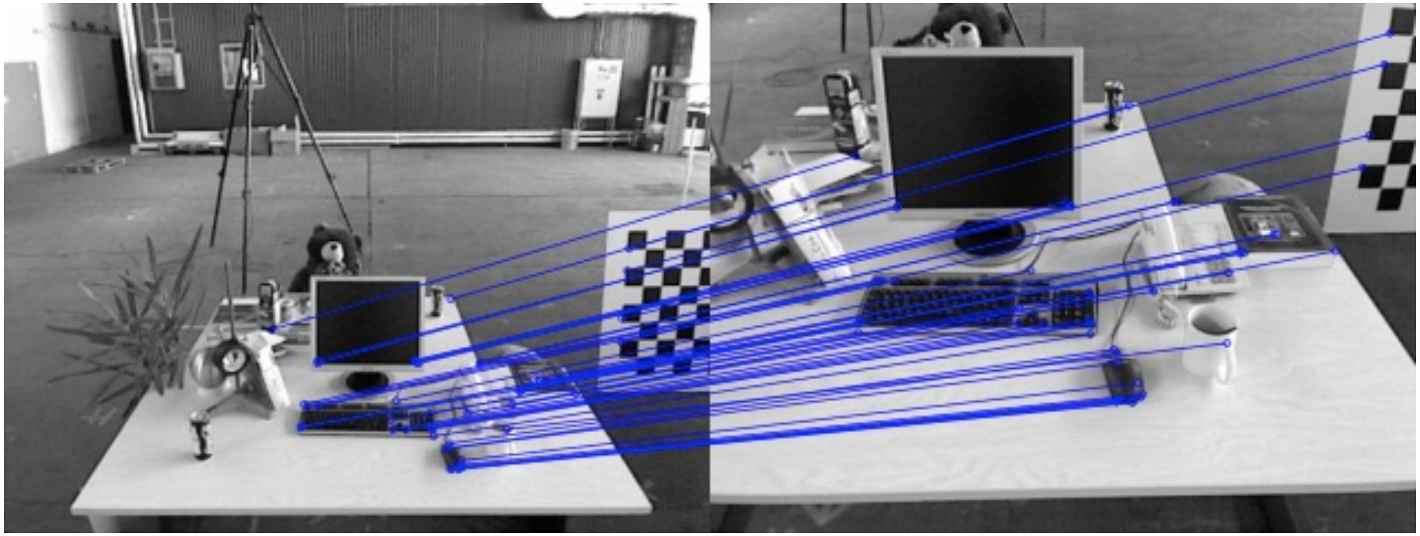
D. Gálvez-López, J.D. Tardós: *Bags of Binary Words for Fast Place Recognition in Image Sequences*, IEEE Trans. Robotics 28(5):1188-1197, 2012 (DBow2 software)

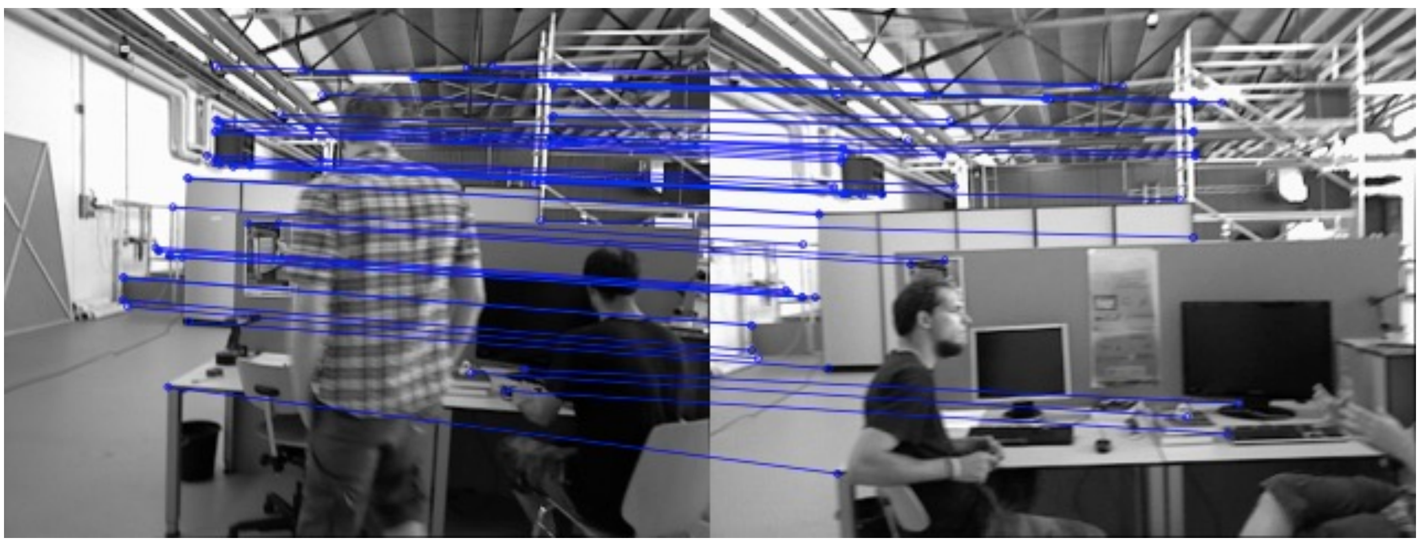
Vocabulary + Direct & Inverse Indexes



- ORB-SLAM predefined vocabulary:
  - Tree levels: 6
  - Branching factor: 10
  - 10^6 words

Examples with DBoW2 using ORB features



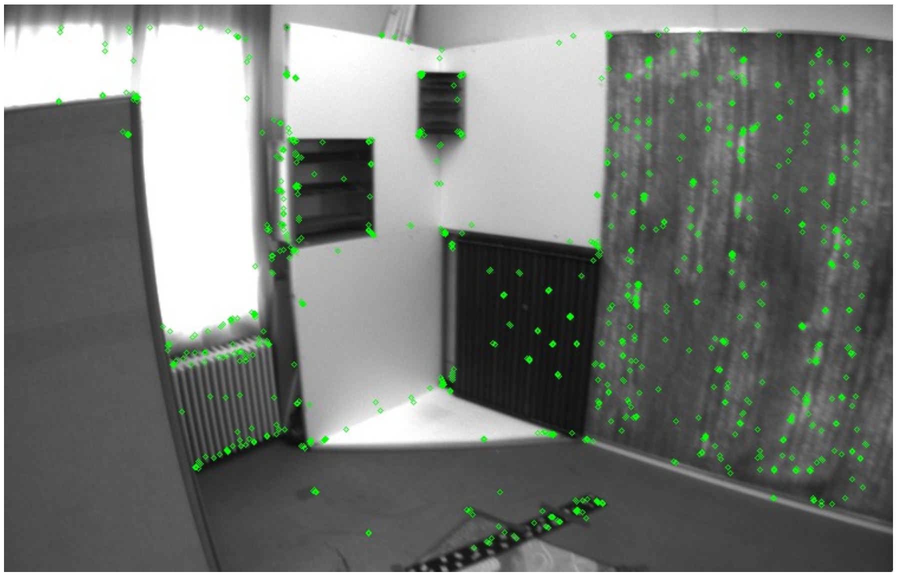


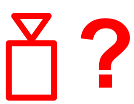
Uses of Place Recognition

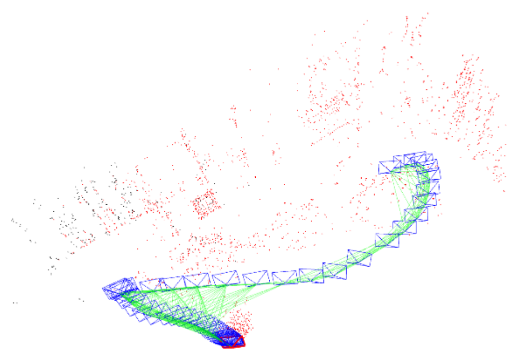
1. Relocalization (when tracking is lost)
   - Frame vs KeyFrame (**2D-3D** matches)
   - Compute camera pose with **PnP** algorithm
2. Loop Closing (correct the accumulated drift)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)
3. Map Merging (merge independent maps)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)

4.4.c Relocalization

0.- Camera tracking was lost
1.- Extract ORB features in current frame

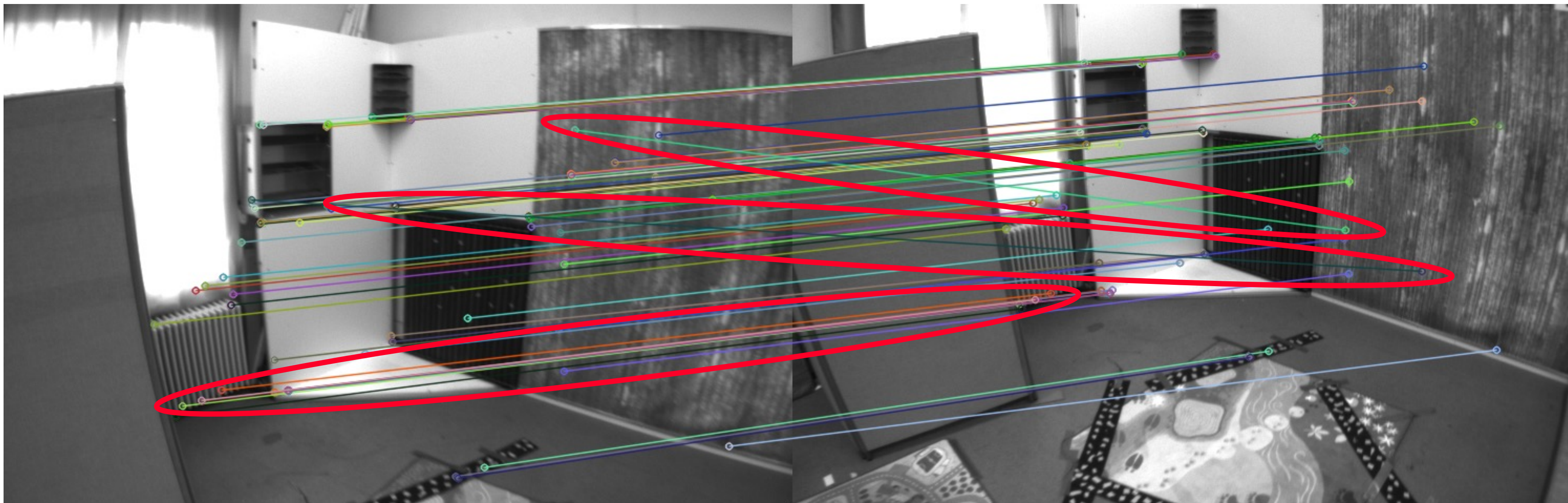


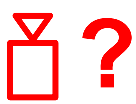


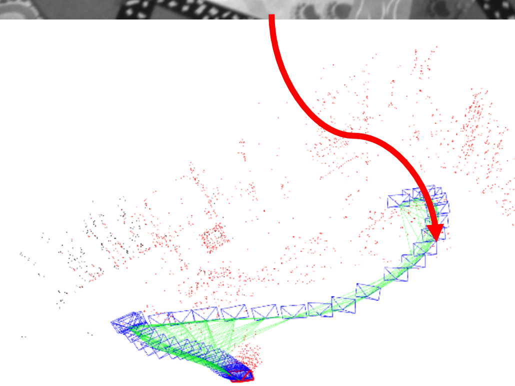


Relocalization

2.- Retrieve most similar KF using DBoW2 and a set of candidate matches



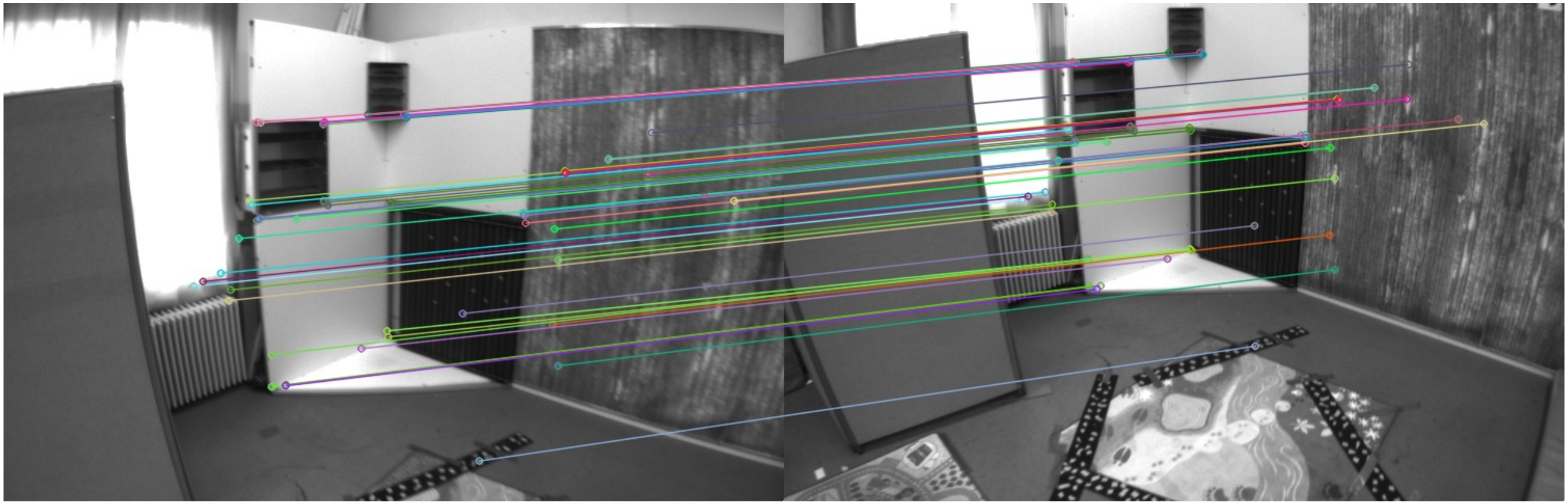


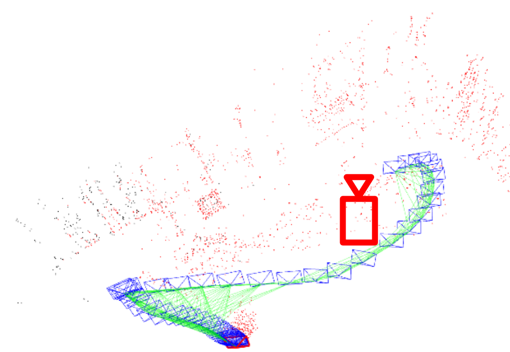


3D pose in map

Relocalization

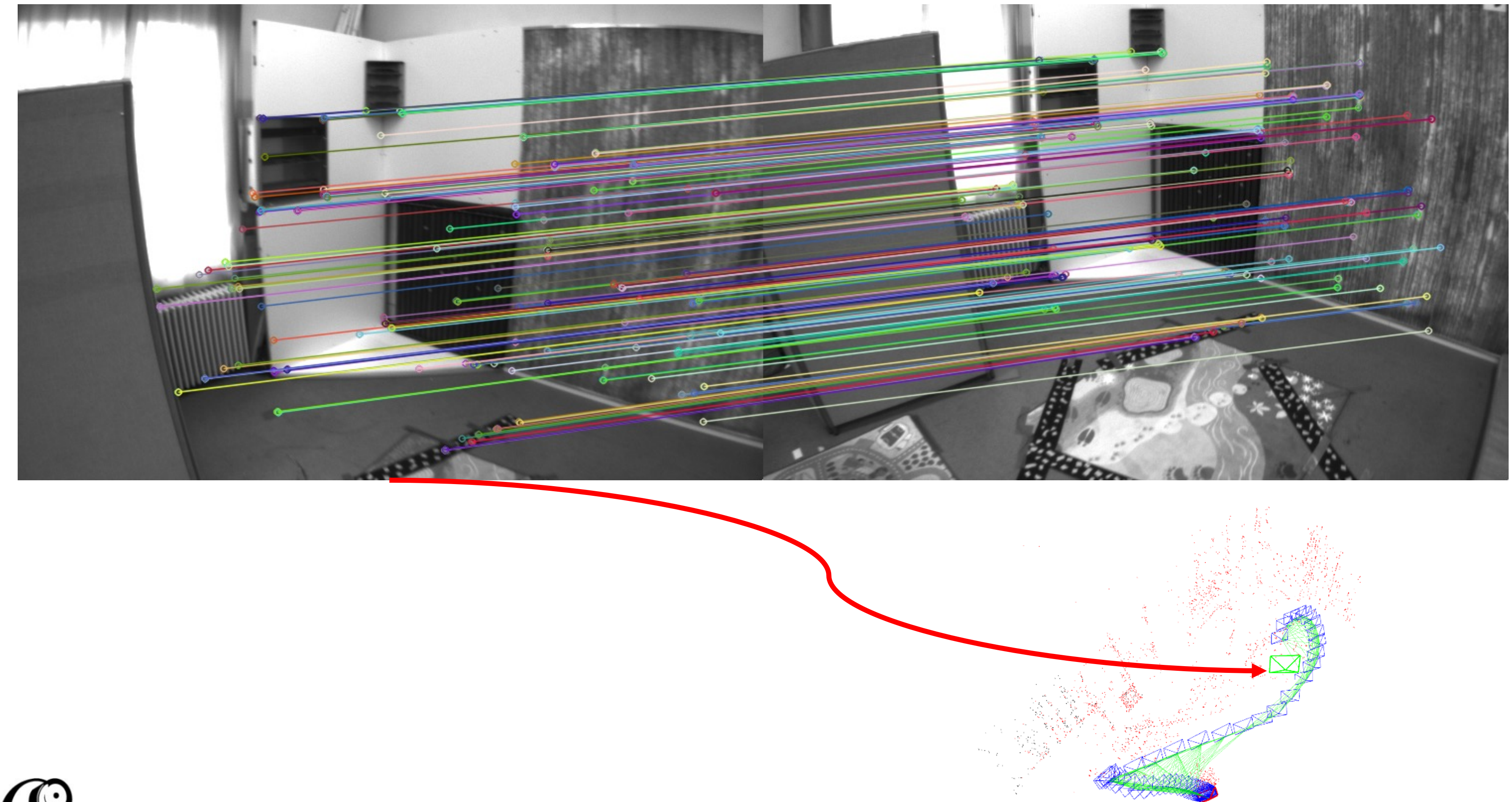
3.- Find inliers and pose from DBoW2 matches
(PnP with RANSAC)



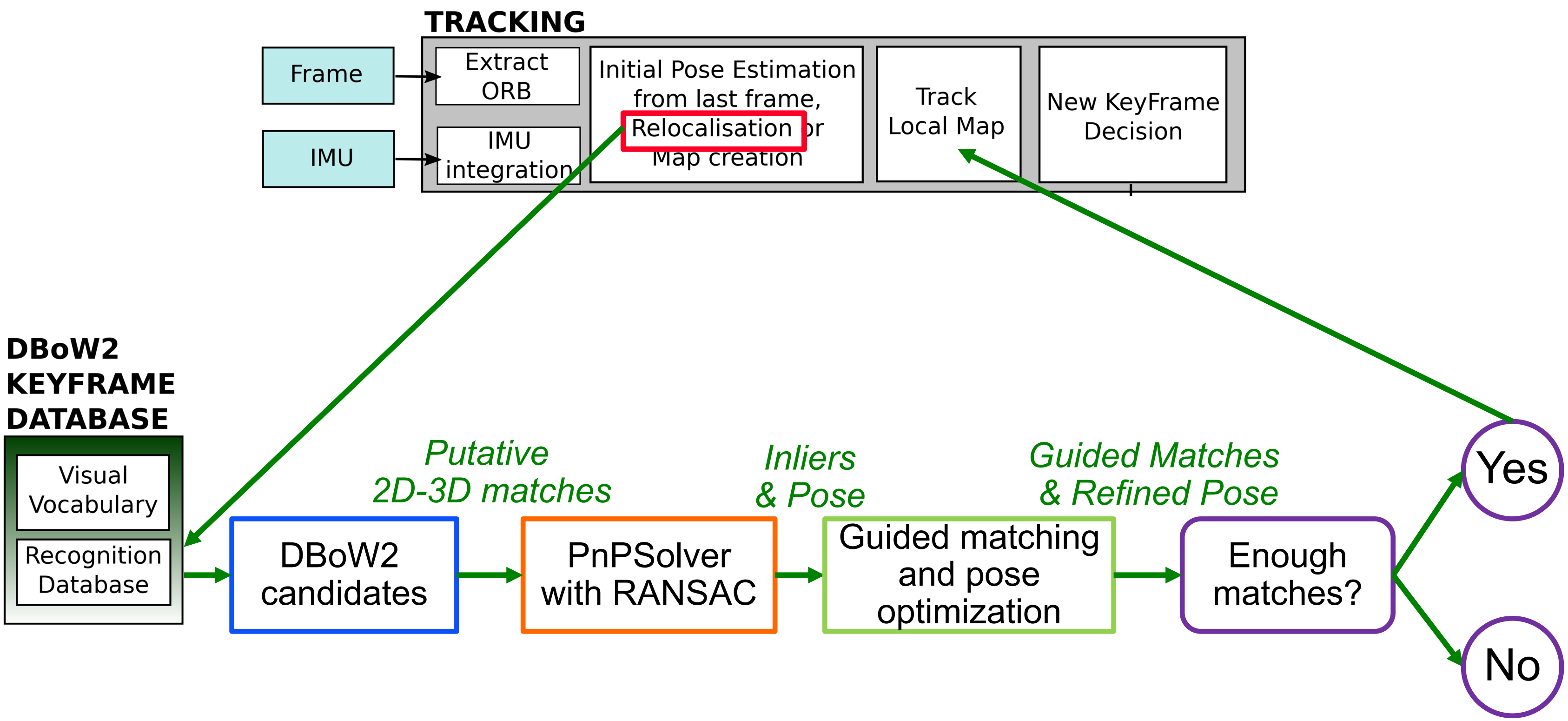


Relocalization

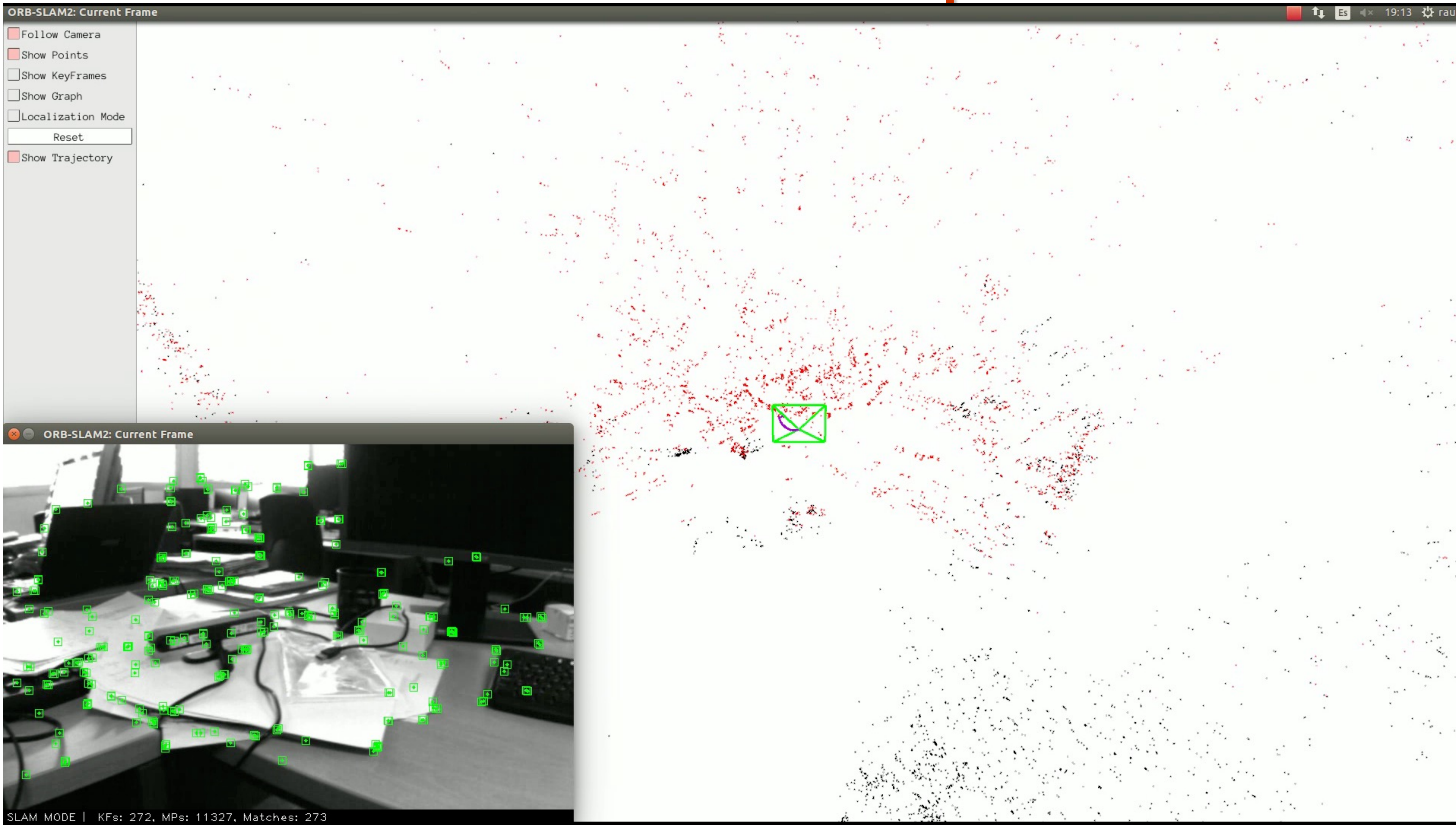
4.- Guided matching with coarse pose estimation
5.- Non-linear camera pose optimization



Relocalization



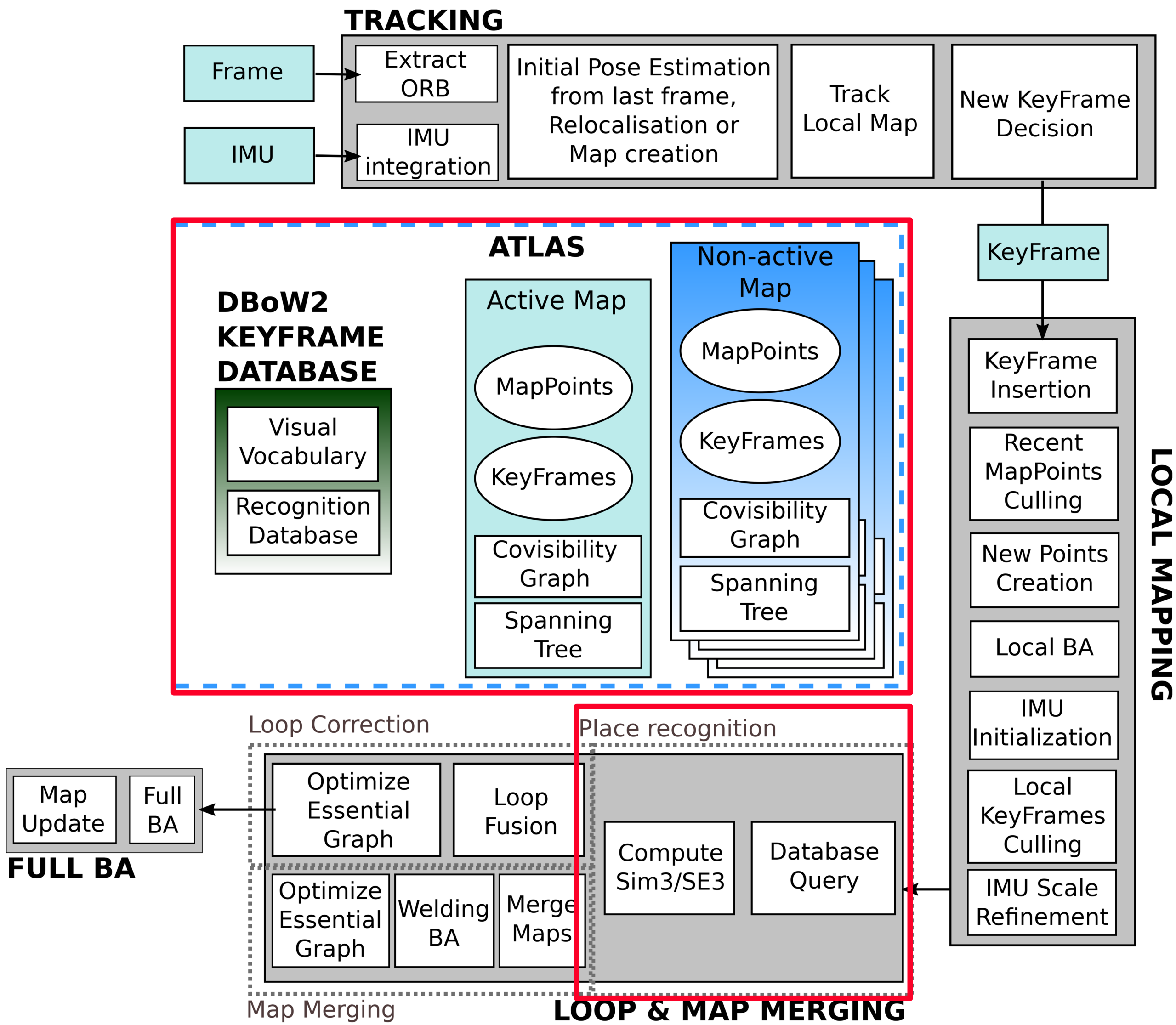
Relocalization Examples



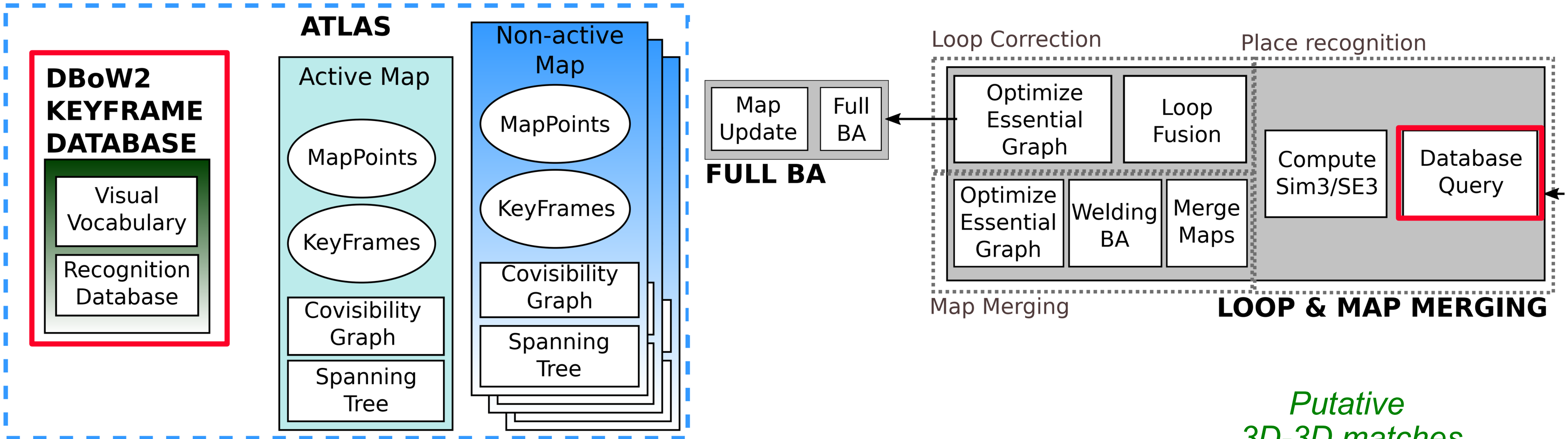
Uses of Place Recognition

1. Relocalization (when tracking is lost)
   - Frame vs KeyFrame (**2D-3D** matches)
   - Compute camera pose with **PnP** algorithm
2. Loop Closing (correct the accumulated drift)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)
3. Map Merging (merge independent maps)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - If monocular align KeyFrames with **Sim3** (scale and pose)
   - Else, align KeyFrames with **SE3** (pose)

4.4.d Loop & Merging detection



Loop & Merging detection



Putative 3D-3D matches

LoopClosing.h/cc

...
NewDetectCommonRegions() : bool
...

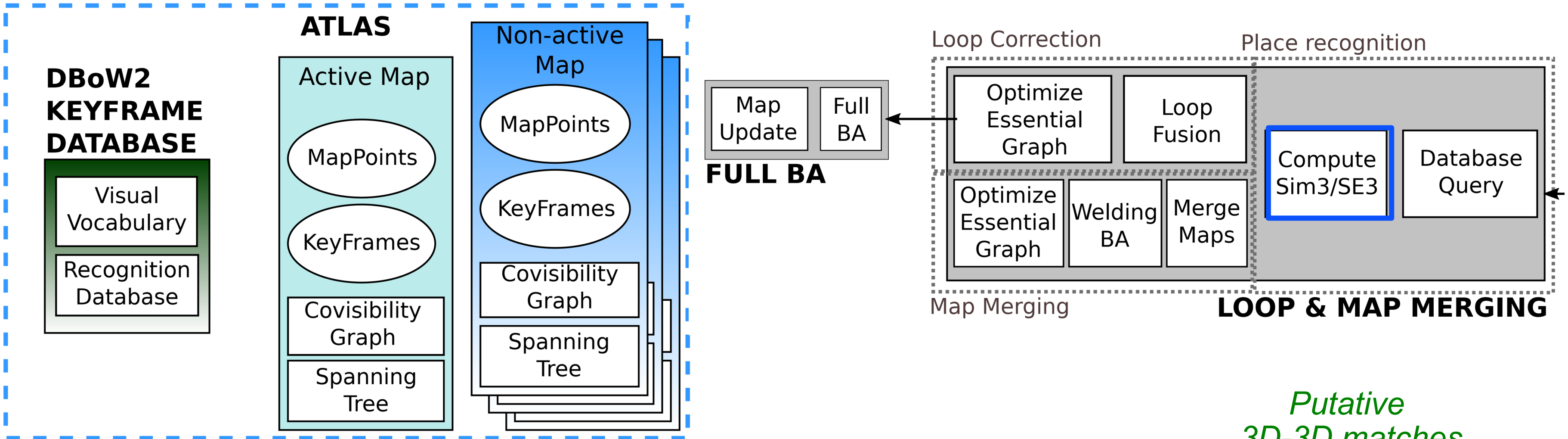
KeyFrameDatabase.h/cc

...
DetectNBestCandidates(KeyFrame*, vector<KeyFrame*>, vector<KeyFrame*>, int) : void
...

ORBMatcher.h/cc

...
SearchByBoW(KeyFrame*, KeyFrame*, vector<MapPoint*>&) : int
...

Loop & Merging Detection



Putative 3D-3D matches

LoopClosing.h/cc

```
...
NewDetectCommonRegions() : bool
...
```

KeyFrameDatabase.h/cc

```
...
DetectNBestCandidates(KeyFrame*, vector<KeyFrame*>, vector<KeyFrame*>, int) : void
...
```

ORBMatcher.h/cc

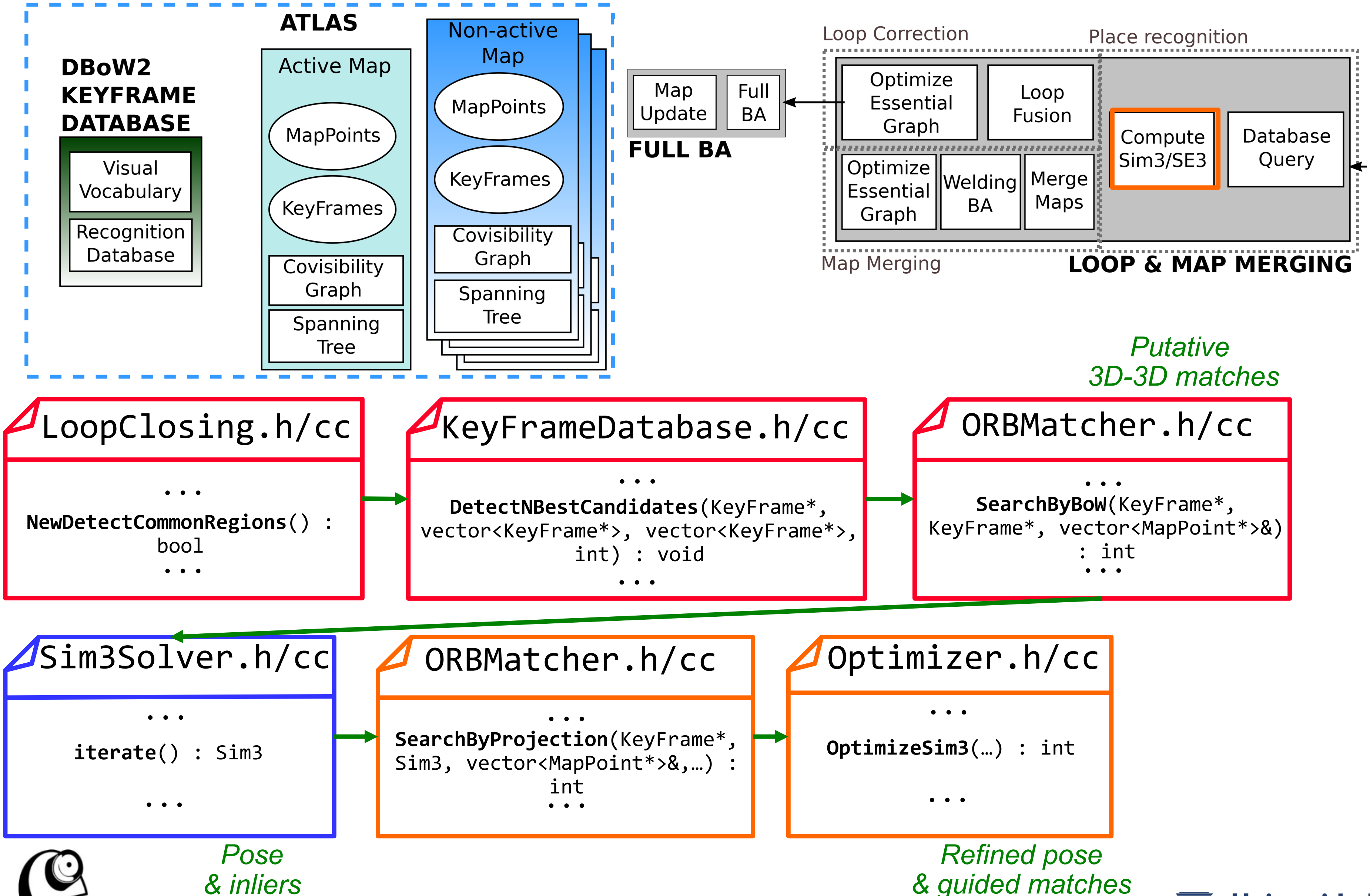
```
...
SearchByBoW(KeyFrame*, KeyFrame*, vector<MapPoint*>&) : int
...
```

Sim3Solver.h/cc

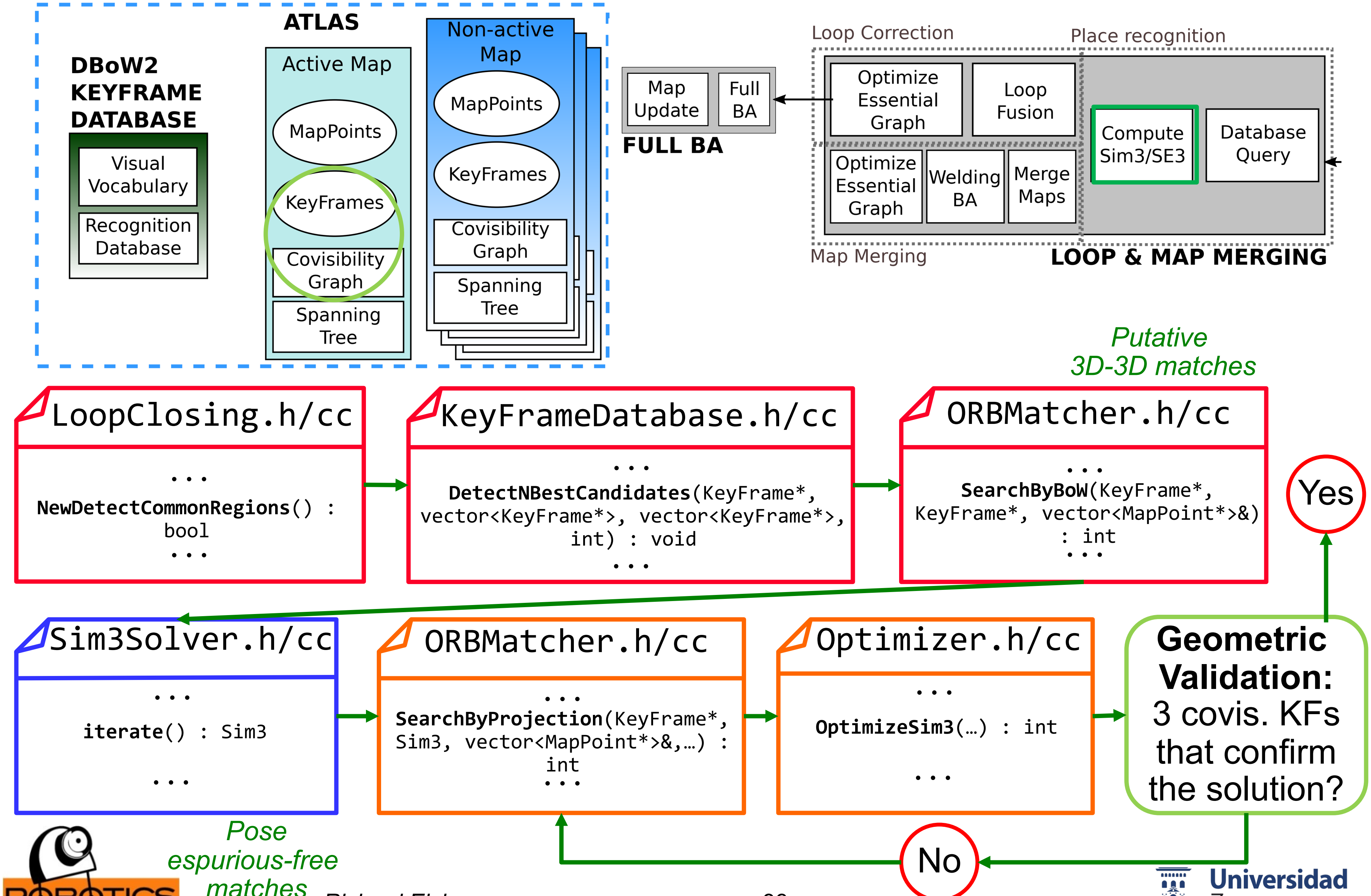
```
...
iterate() : Sim3
...
```

Pose & inliers

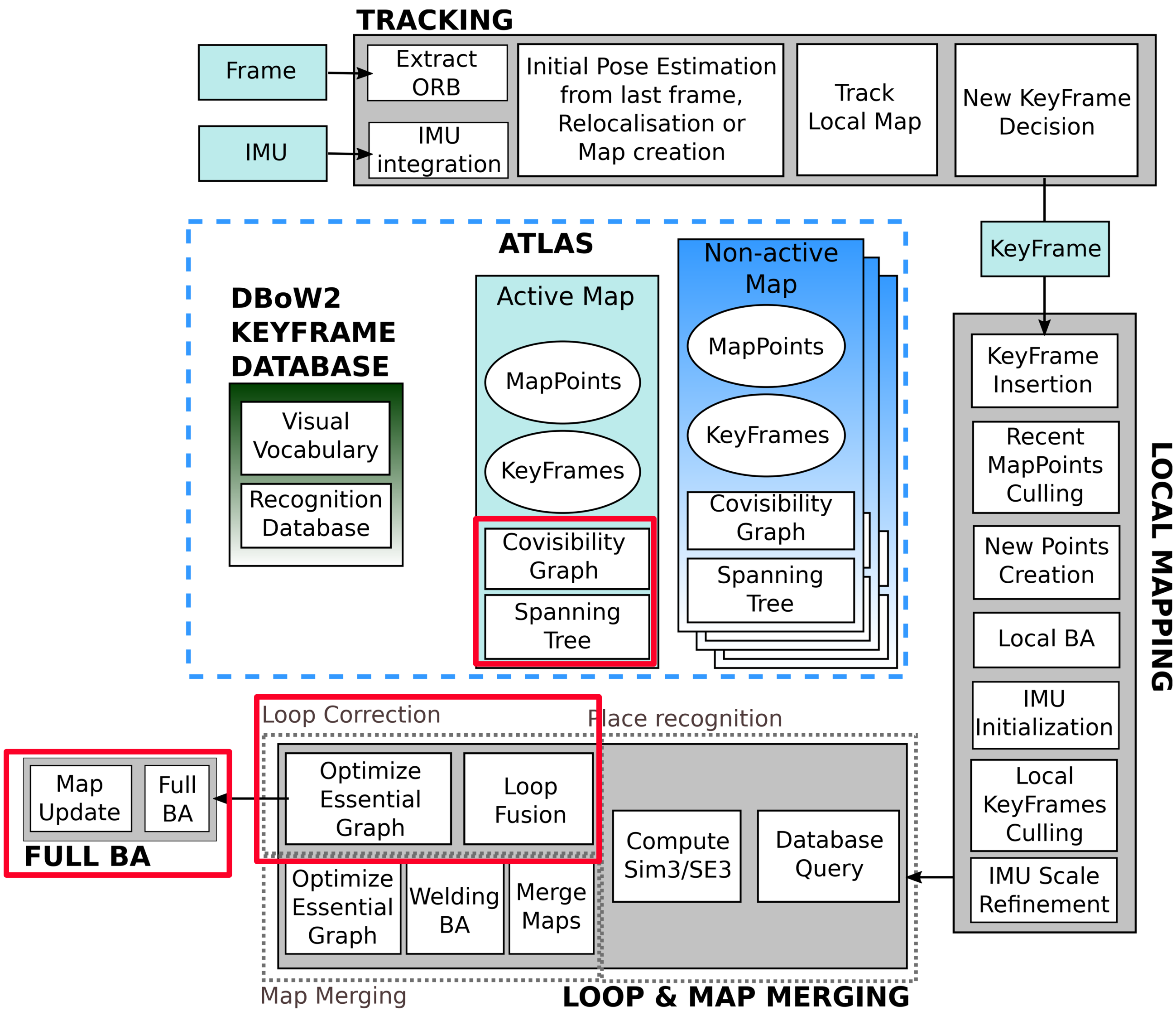
Loop & Merging Detection



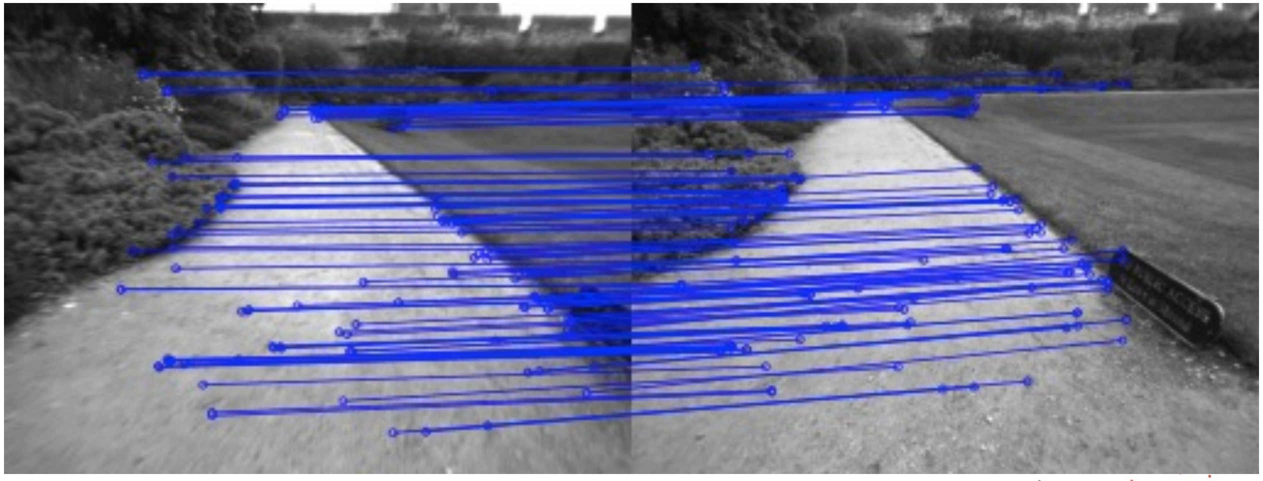
Loop & Merging Detection

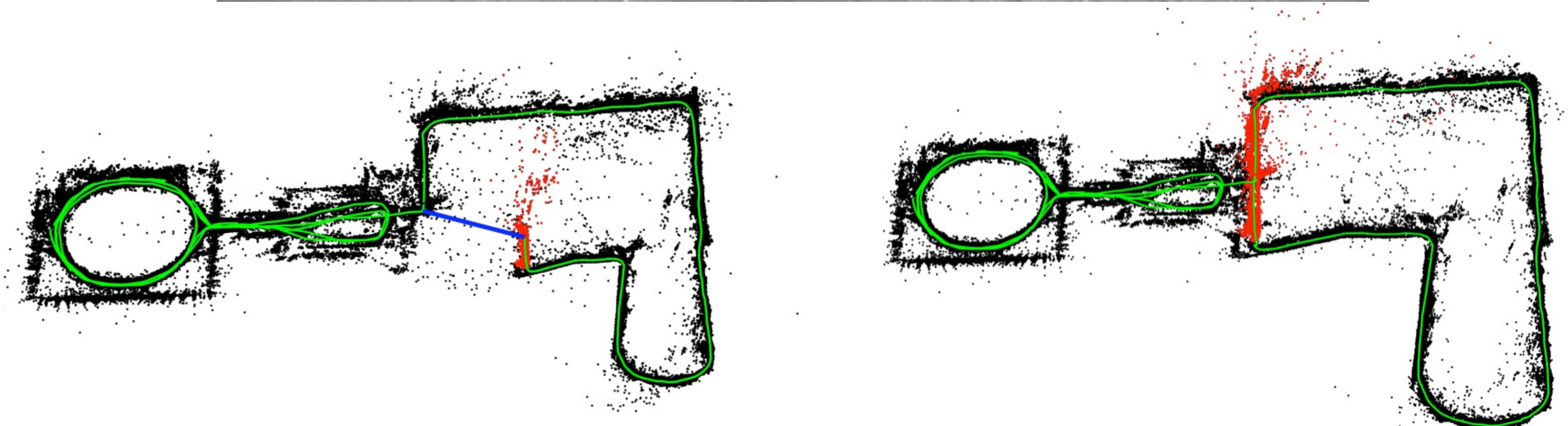


Loop Correction



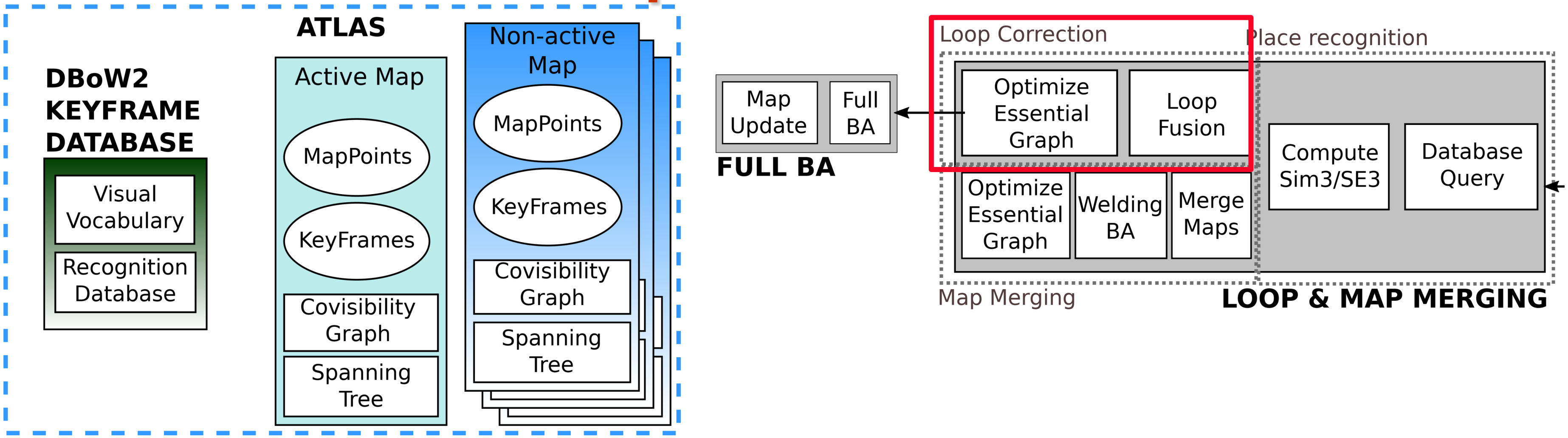
Loop Correction

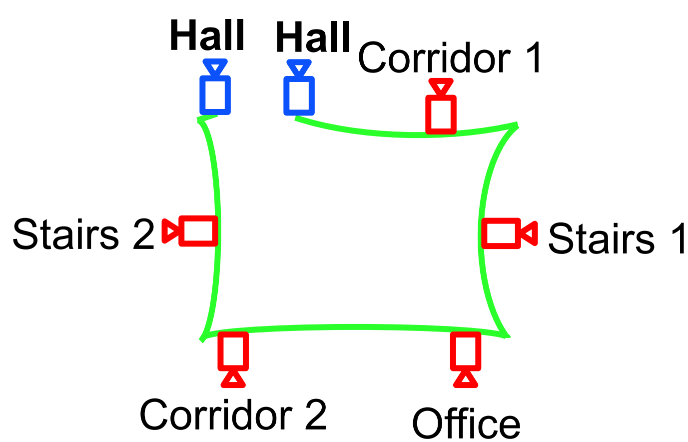




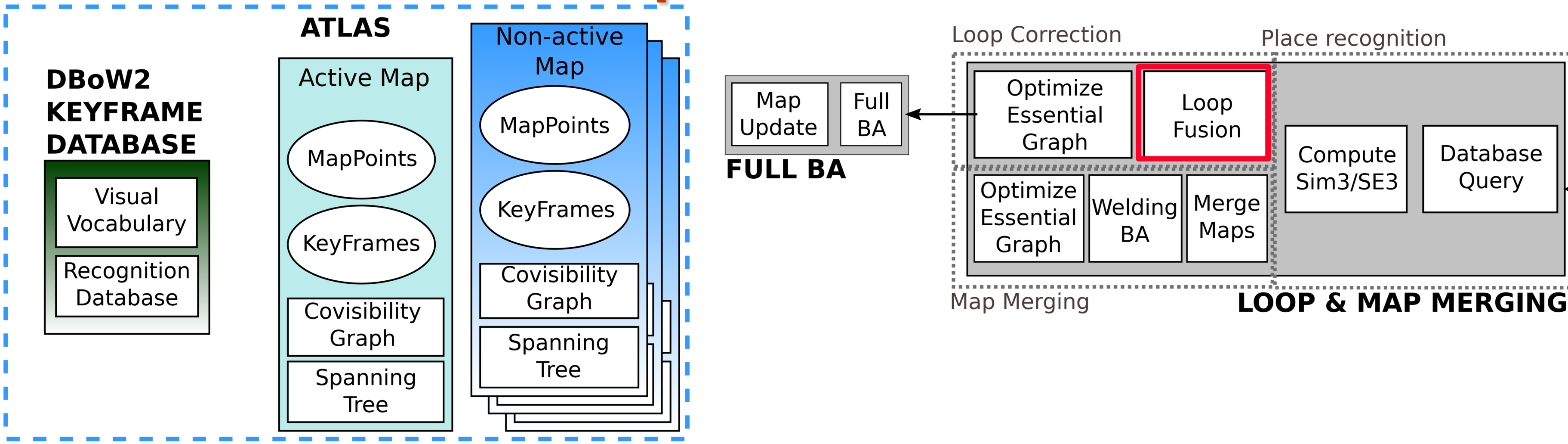
- Mono: 7 DoF graph optimization, to correct scale drift
- Stereo and Visual-Inertial: 6 DoF graph optimization
- And Full BA in a separated thread

Loop Correction



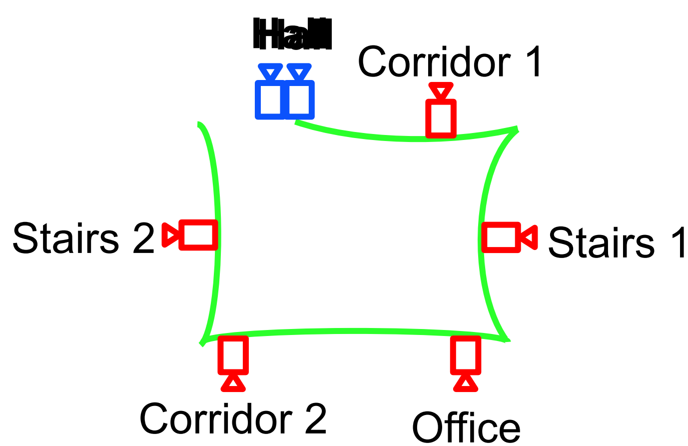


Loop Correction

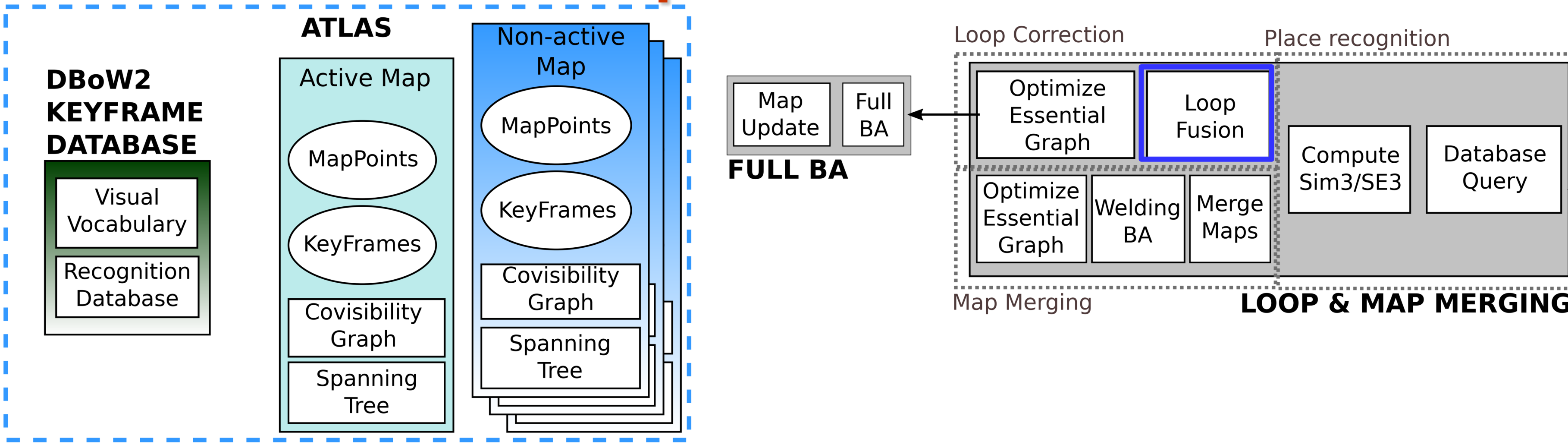


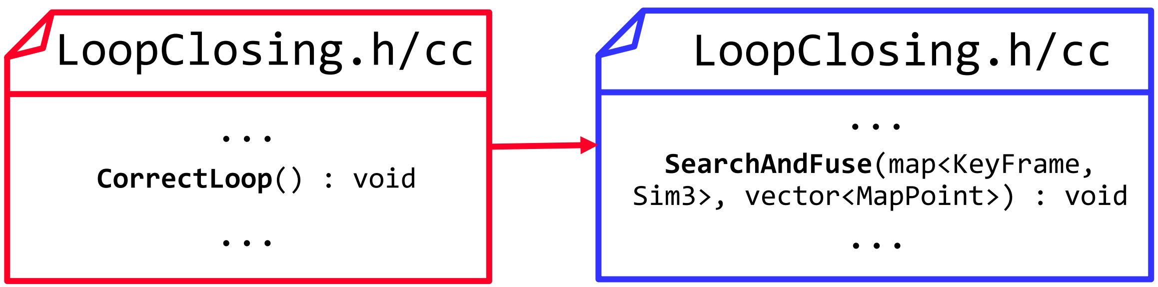
LoopClosing.h/cc

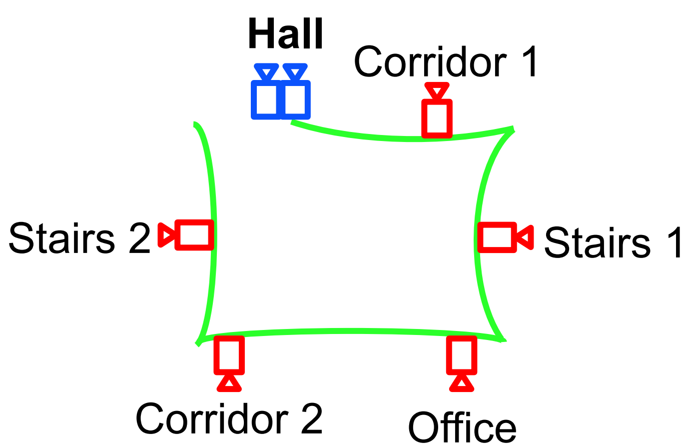
```
...
CorrectLoop() : void
...
```



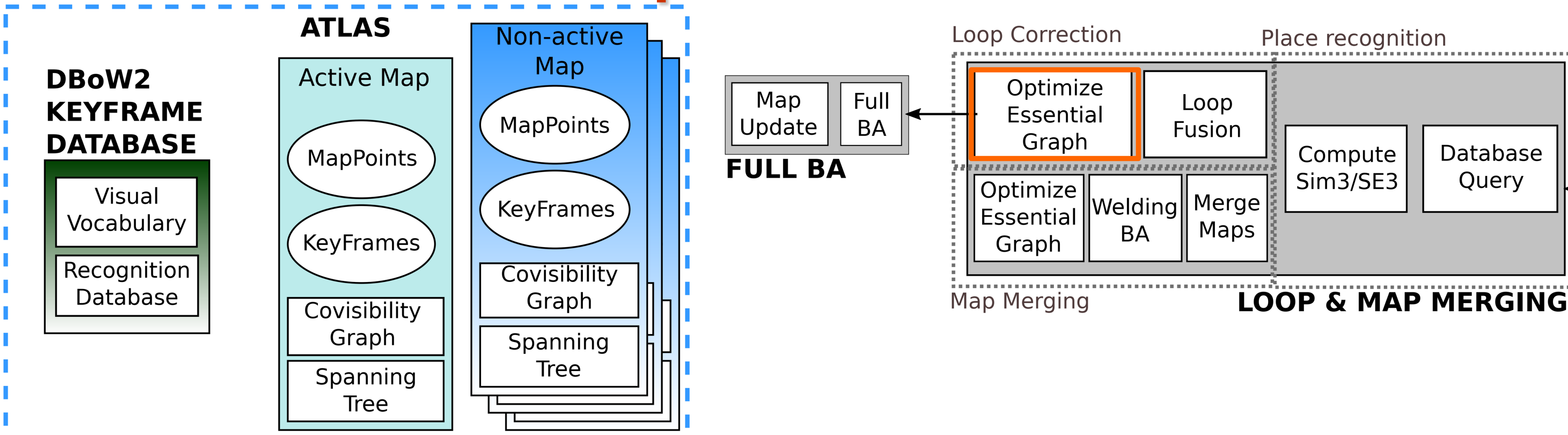
Loop Correction







Loop Correction



LoopClosing.h/cc

...

CorrectLoop() : void

...

LoopClosing.h/cc

...

SearchAndFuse(map<KeyFrame, Sim3>, vector<MapPoint>) : void

...

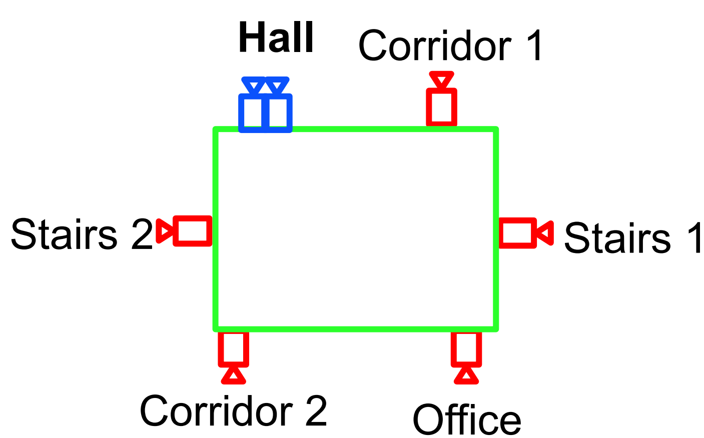
Optimizer.h/cc

...

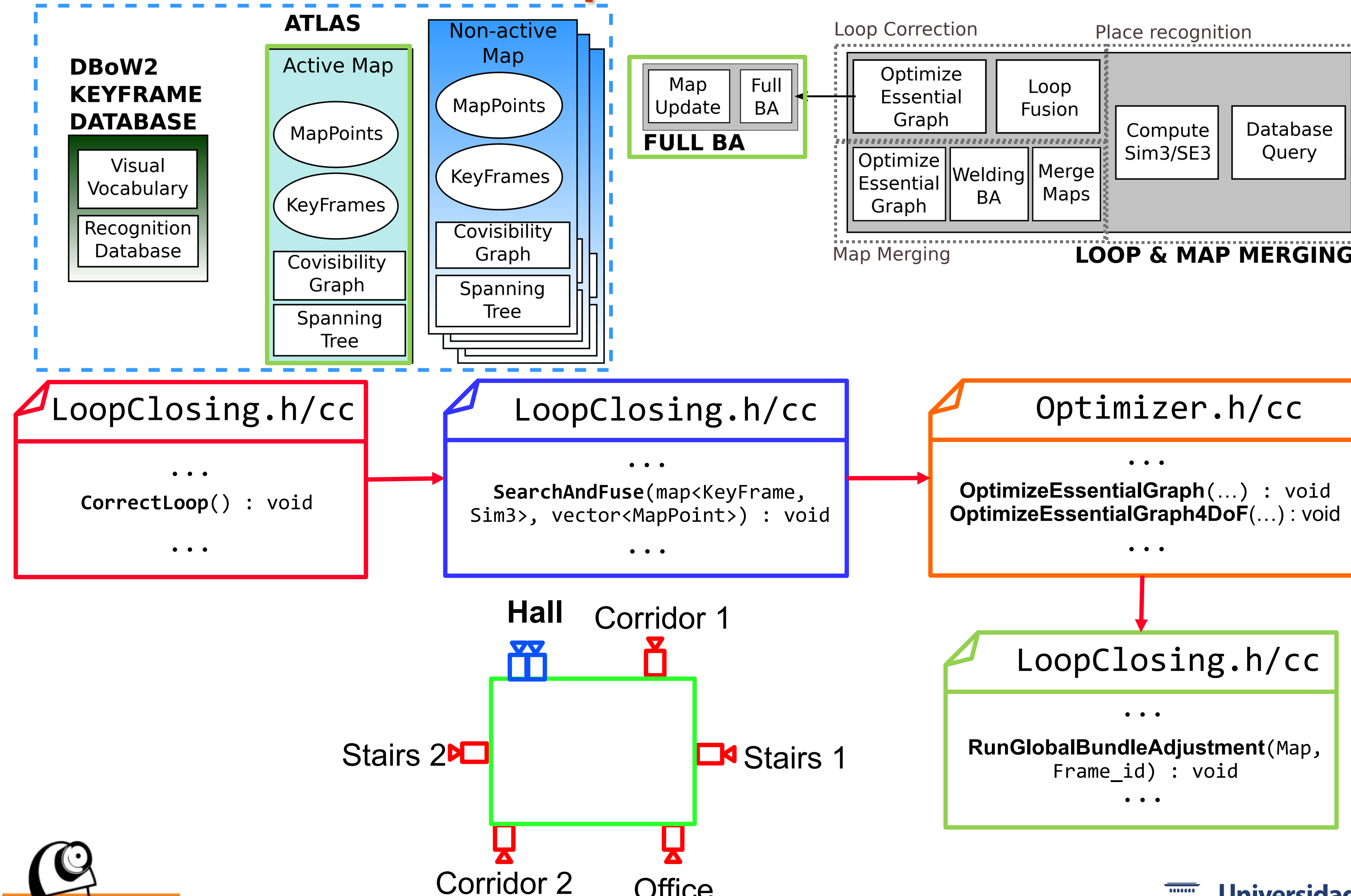
OptimizeEssentialGraph(...) : void

OptimizeEssentialGraph4DoF(...) : void

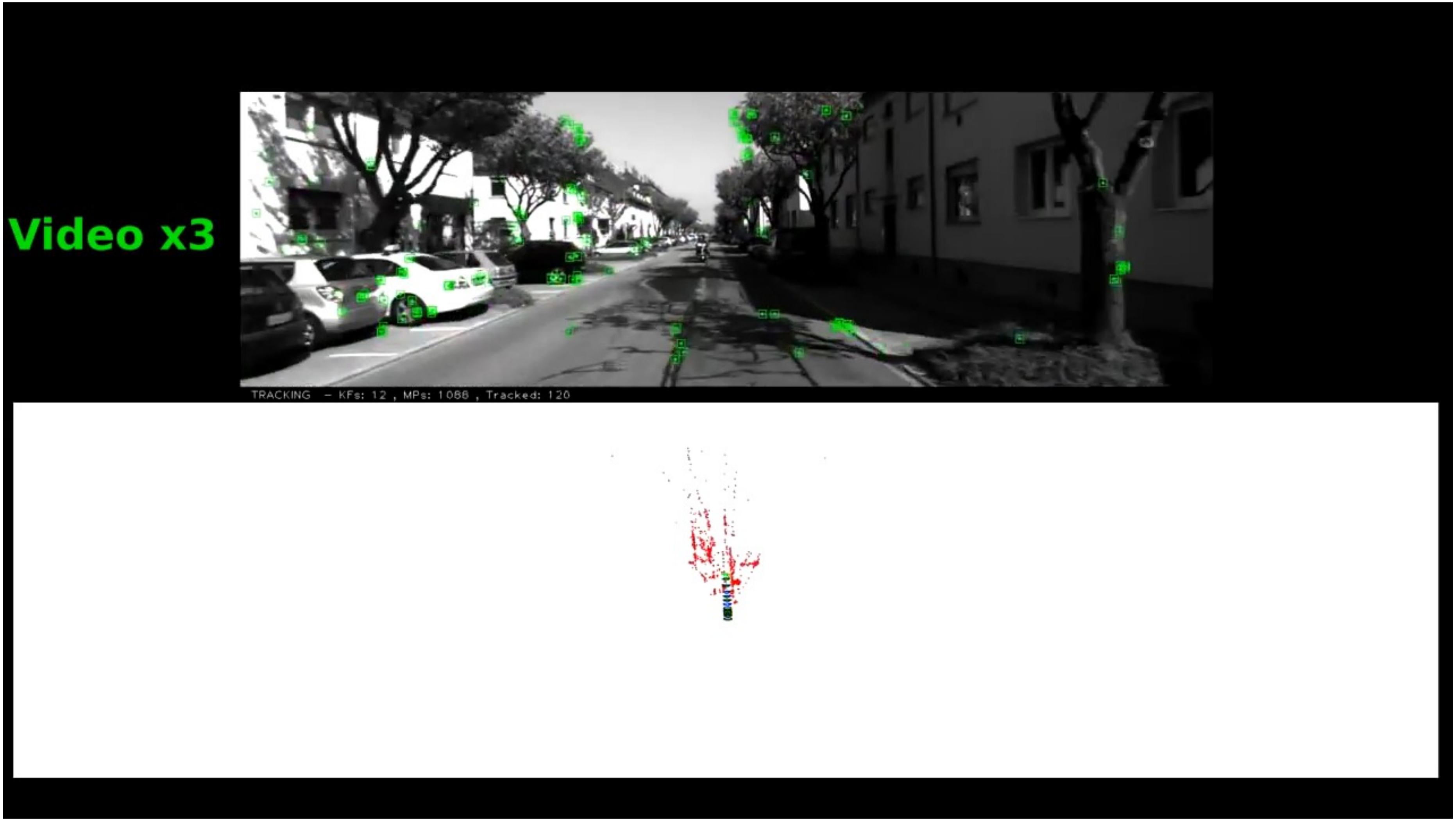
...



Loop Correction



Examples of Loop Correction



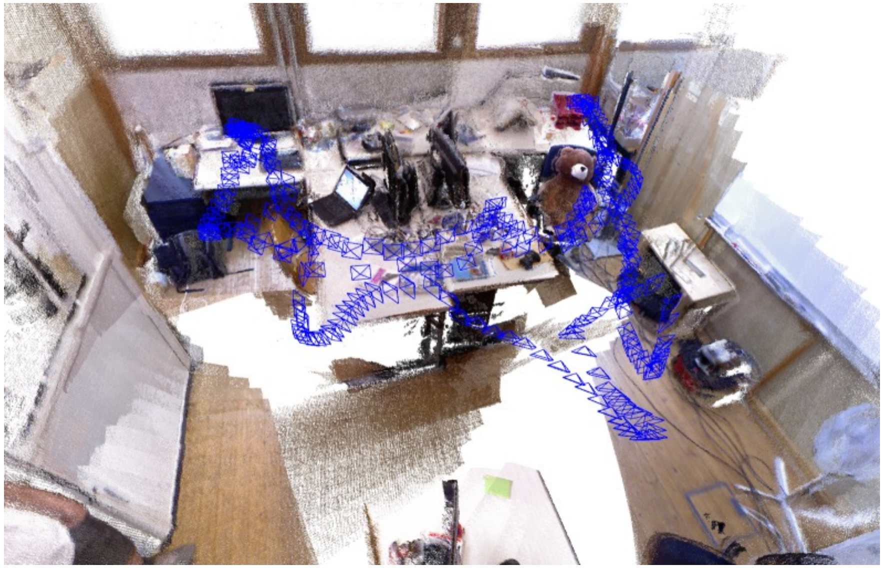
ORBSLAM3 Loop Closing Times (ms)

Settings	Sensor	Monocular	Stereo	Mono-Inertial	Stereo-Inertial
	Resolution	752×480	752×480	752×480	752×480
	Cam. FPS	20Hz	20Hz	20Hz	20Hz
	IMU	-	-	200Hz	200HZ
	ORB Feat.	1000	1200	1000	1200
	RMS ATE	0.	0.	0.	0.
Place Recognition	Database query	0.96±0.58	1.06±0.58	1.04±0.59	1.02±0.60
	Compute Sim3/SE3	3.61±2.81	5.26±3.79	2.98±2.26	5.71±3.54
Once per KF	Total	3.92±3.28	5.26±4.39	3.45±2.81	5.89±4.29
Map Merging	Merge Maps	152.03±45.85	68.56±13.56	129.08±8.26	91.07±5.56
	Welding BA	52.09±14.08	35.57±7.94	103.14±6.08	58.15±4.84
	Opt. Essential Graph	5.82±3.01	10.98±9.79	52.83±17.81	36.08±17.95
	Total	221.90±58.73	120.63±16.23	287.33±15.58	187.82±6.38
Merge info	# Detected merges	5	4	2	2
	Merge size (# keyframes)	31±1	31±3	25±1	25±0
	Merge size (# map points)	2476±207	2697±718	2425±88	4260±160
Loop	Loop Fusion	311.82±333.49	29.07±23.64	-	25.67
	Opt. Essential Graph	254.84±87.03	84.36±37.56	-	95.13
Once per Loop	Total	570.39±420.77	118.62±59.93	-	124.77
Loop info	# Detected loops	3	4	0	1
	Loop size (# keyframes)	58±60	27±9	-	60
Loop Full BA	Full BA	4010.14±1835.85	1118.54±563.75	-	1366.64
	Map Update	124.80±6.07	13.65±12.86	-	163.06
	Total	4134.94±1829.78	1132.19±572.28	-	1529.69
4th Thread	BA size (# keyframes)	345±147	220±110	-	151
	BA size (# map points)	13511±3778	12297±4572	-	14397

Intel Core i7-7700 @3.6 GHz, 32 GB

4.5 Visual SLAM: Accuracy

Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



Lesson 4. Visual SLAM

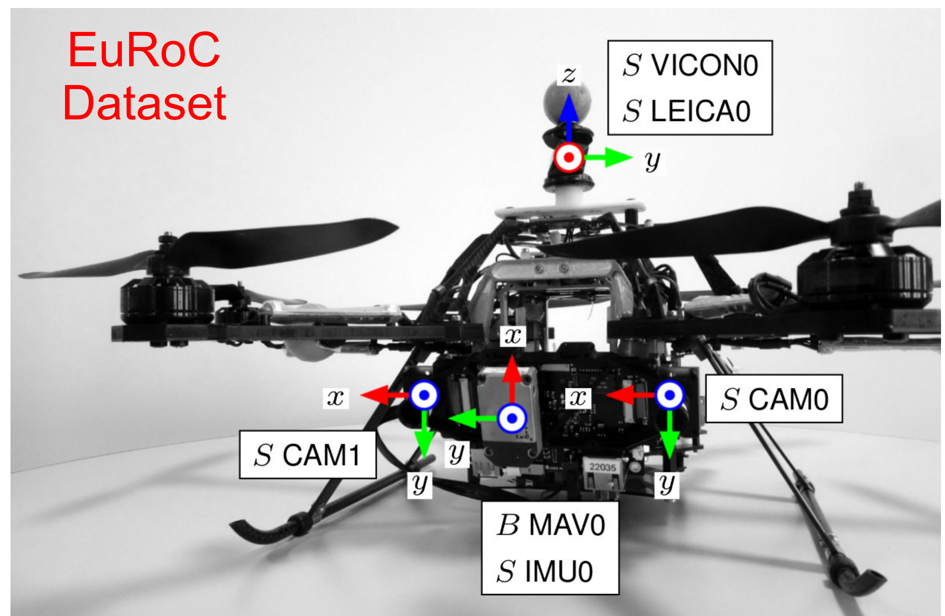
5. Accuracy
    a. Measuring accuracy
    b. Monocular results
    c. Stereo & RGB-D results
    d. Conclusions

Readings:

- ORB-SLAM paper
- ORB-SLAM2 paper
- ORB-SLAM3 paper

4.5.a Measuring SLAM Accuracy

- Requires ground-truth (GT) from some external sensor
- Map accuracy
  - Useful in dense SLAM methods
  - Ground truth: dense 3D model of the environment
    - » Stereo or multi-view reconstruction
    - » 3D Laser scanner (very expensive!)
- Trajectory accuracy
  - Bad trajectory => bad map
  - Useful in all SLAM methods
  - Trajectory ground-truth
    - » Outdoors: GPS
    - » Indoors: motion capture system, total station,…



ATE: Absolute Translation Error (RMSE)

SLAM trajectory

$$\{\mathbf{R}_{0i}, \mathbf{t}_{0i}, t_i \mid i \in \mathcal{C}\} \; ; \; i = \{1..N\}$$

GT trajectory

$$\{\mathbf{R}_{wj}^{GT}, \mathbf{t}_{wj}^{GT}, t_j \mid j \in \mathcal{GT}\}$$

Different reference

Different timestamps

Find the best 6-DOF alignment between SLAM and GT

$$ATE_{RMSE} = \min_{\mathbf{R}_{w0}, \mathbf{t}_{w0}} \sqrt{\frac{1}{N} \sum_{i=1}^{N} \left\| \mathbf{t}_{wj}^{GT}(t_i) - \mathbf{R}_{w0}\mathbf{t}_{0i} - \mathbf{t}_{w0} \right\|^2}$$

Interpolate GT at SLAM timestamps

ATE: Monocular case

SLAM trajectory

$$\{\mathbf{R}_{0i}, \mathbf{t}_{0i}, t_i \mid i \in \mathcal{C}\}\ ;\ i = \{1..N\}$$

GT trajectory

$$\{\mathbf{R}_{wj}^{GT}, \mathbf{t}_{wj}^{GT}, t_j \mid j \in \mathcal{GT}\}$$

Different reference AND SCALE

Different timestamps

Find the best 7-DOF alignment between SLAM and GT

$$ATE_{RMS}^{MONO} = \min_{s,\mathbf{R}_{w0},\mathbf{t}_{w0}} \sqrt{\frac{1}{N}\sum_{i=1}^{N} \left\| \mathbf{t}_{wj}^{GT}(t_i) - s\mathbf{R}_{w0}\mathbf{t}_{0i} - \mathbf{t}_{w0} \right\|^2}$$

Interpolate GT at SLAM timestamps

ATE in Keyframe-Based SLAM

- Keyframes are better estimated than frames
  - Computing ATE with the frame trajectory is more realistic

$$\{\mathbf{R}_{0i}, \mathbf{t}_{0i}, t_i \mid i \in \mathcal{C}\} \; ; \; i = \{1..N\}$$

- During tracking, store the pose of each frame relative to its previous KF:

$$\mathbf{T}_{ki} = \mathbf{T}_{0k}^{-1}\mathbf{T}_{0i}$$

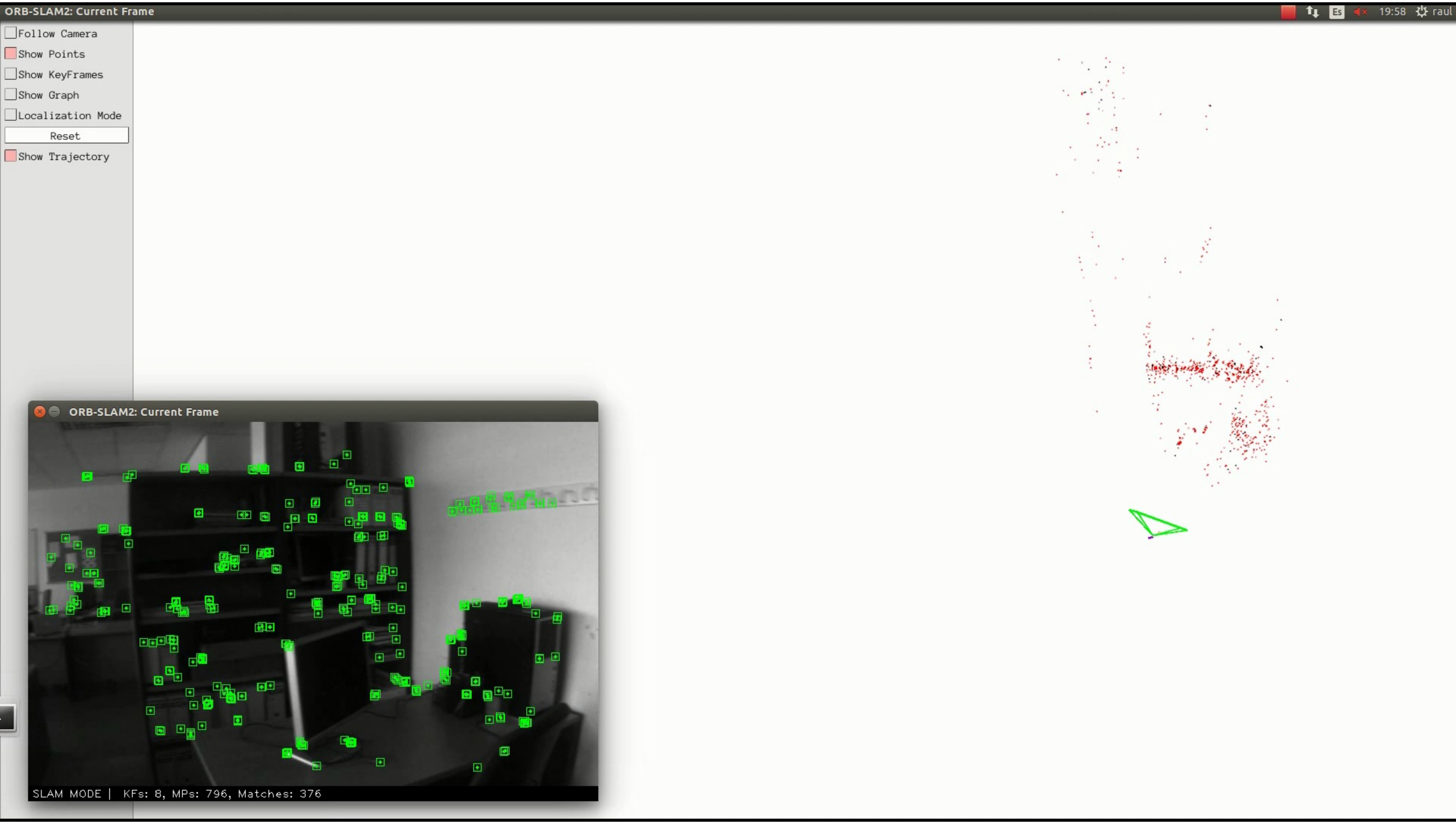
- KFs are optimized during mapping and loop closing.
- At the end of SLAM, save the corrected frame trajectory:

$$\mathbf{T}^*_{0i} = \mathbf{T}^*_{0k}\mathbf{T}_{ki}$$

RPE: Relative Pose Error

- Translation: position error / trajectory length (%)
- Rotation: angular error / trajectory length (degrees / m)

4.5.b Monocular Results, Indoors



ORB-SLAM Indoors: TUM RGB-D Dataset

ORB-SLAM

Raúl Mur-Artal, J. M. M. Montiel and Juan D. Tardós

{raulmur, josemari, tardos} @unizar.es

Instituto Universitario de Investigación
en Ingeniería de Aragón
Universidad Zaragoza

Universidad
Zaragoza
1542

	ORB-SLAM	PTAM	RGBD-SLAM
fr1_xyz	**0.90**	1.15	1.34
fr2_xyz	0.30	**0.20**	1.42
fr1_floor	**2.99**		3.51
fr1_desk	**1.69**		2.52
fr2_360 _kidnap	3.81	**2.63**	100.5
fr2_desk	**0.88**		3.94
fr3_long _office	**3.45**		-
fr3_nstr_ tex_near	**1.39**	2.74	-
fr3_str_ tex_far	**0.77**	0.93	-
fr3_str_ tex_near	1.58	**1.04**	-
fr2_desk _person	**0.63**		2.00
fr3_sit_ xyz	**0.79**	0.83	-
fr3_sit_ _halfsph	**1.34**		-
fr3_walk _xyz	**1.24**		-
fr3_walk _halfsph	**1.74**		-

TUM RGB-D Benchmark

ATE RMS (cm)

RGB-D SLAM results taken from the benchmark website

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza

Monocular Outdoors: Kitti Dataset

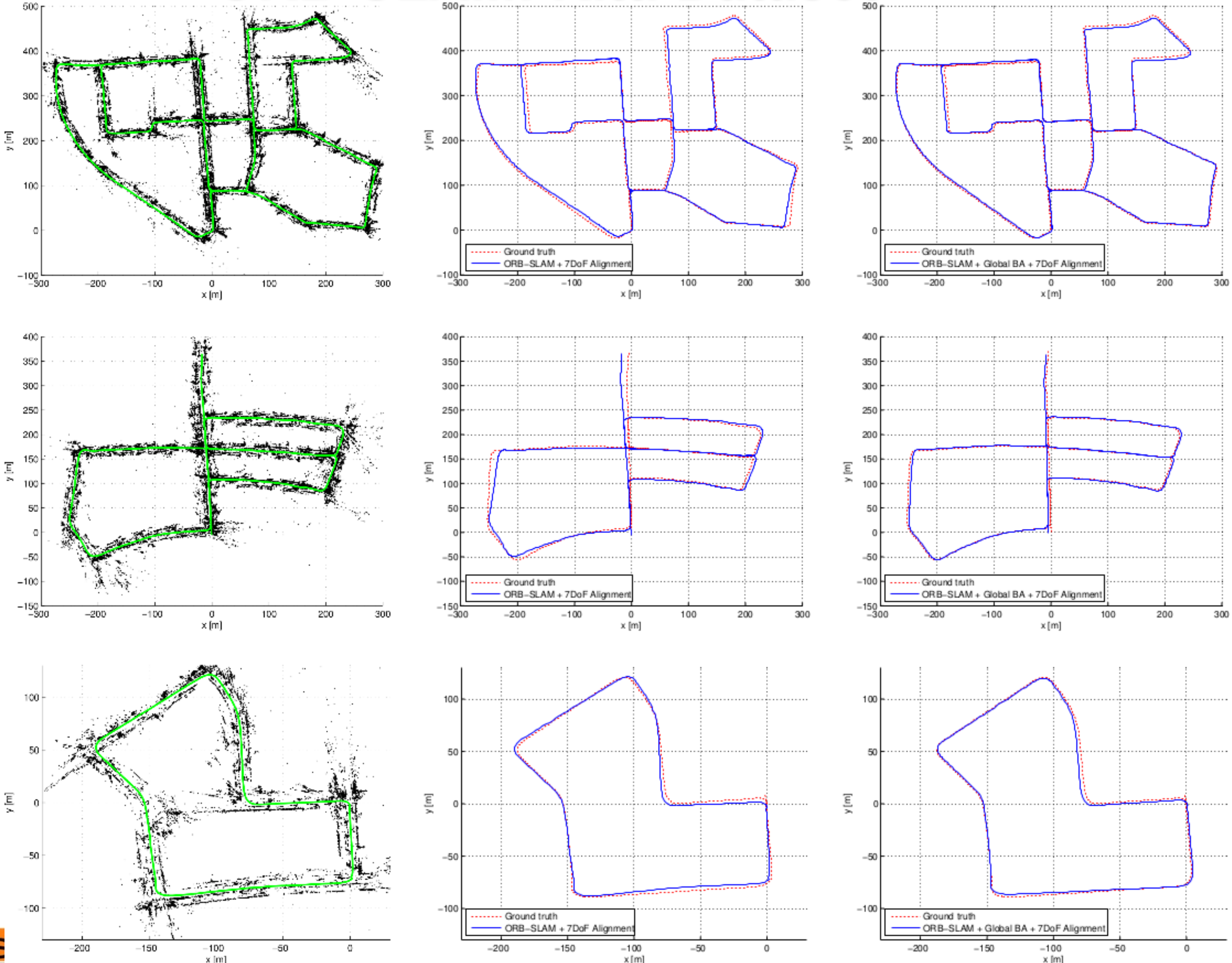
ORB-SLAM

Raúl Mur-Artal, J. M. M. Montiel and Juan D. Tardós

{raulmur, josemari, tardos} @unizar.es

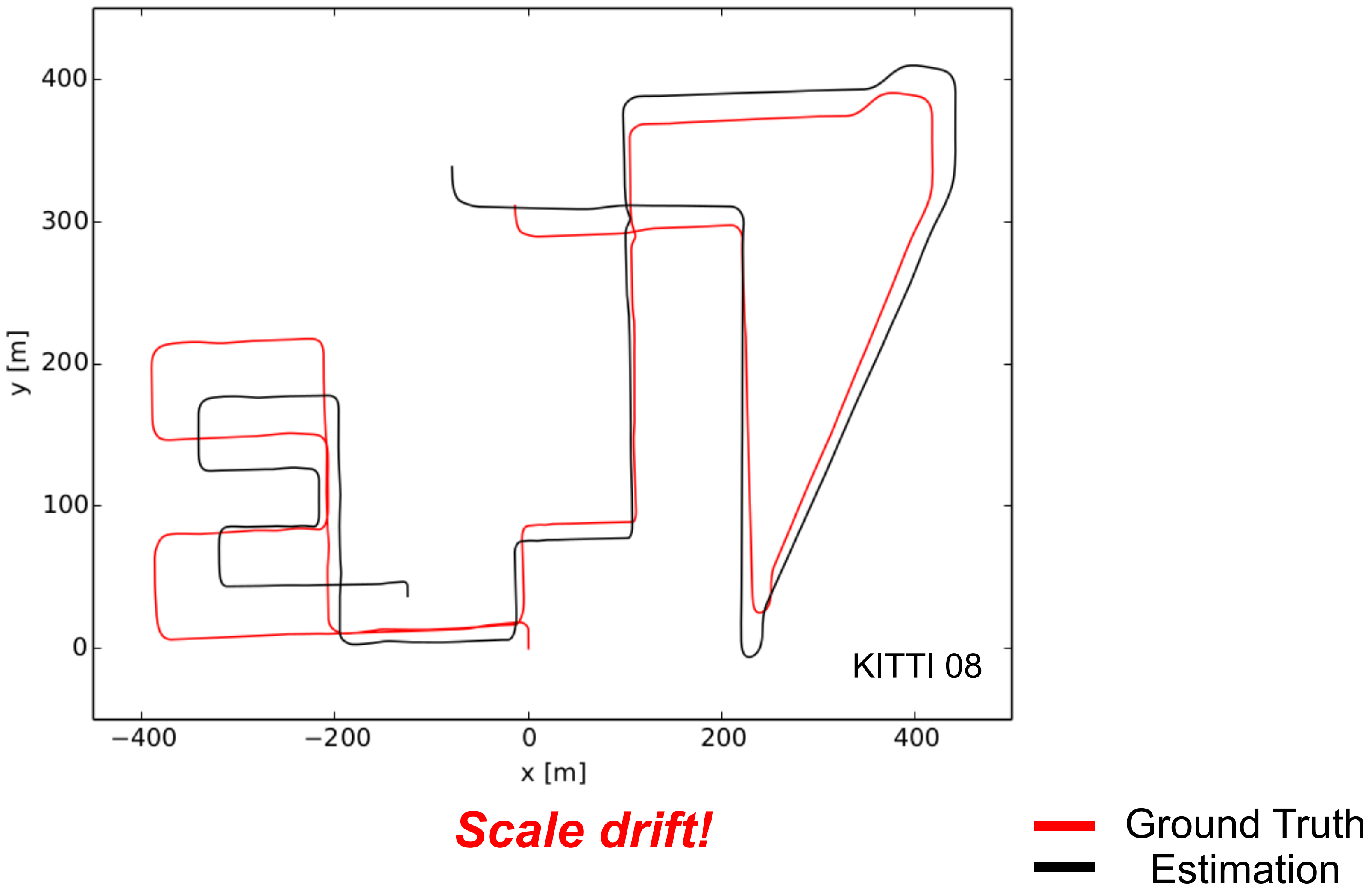
Trajectory and Map Obtained

ORB-SLAM: Kitti Dataset



ORB-SLAM Monocular

- With pure monocular, scale is not observable



Scale drift!

4.5.c Stereo and RGB-D

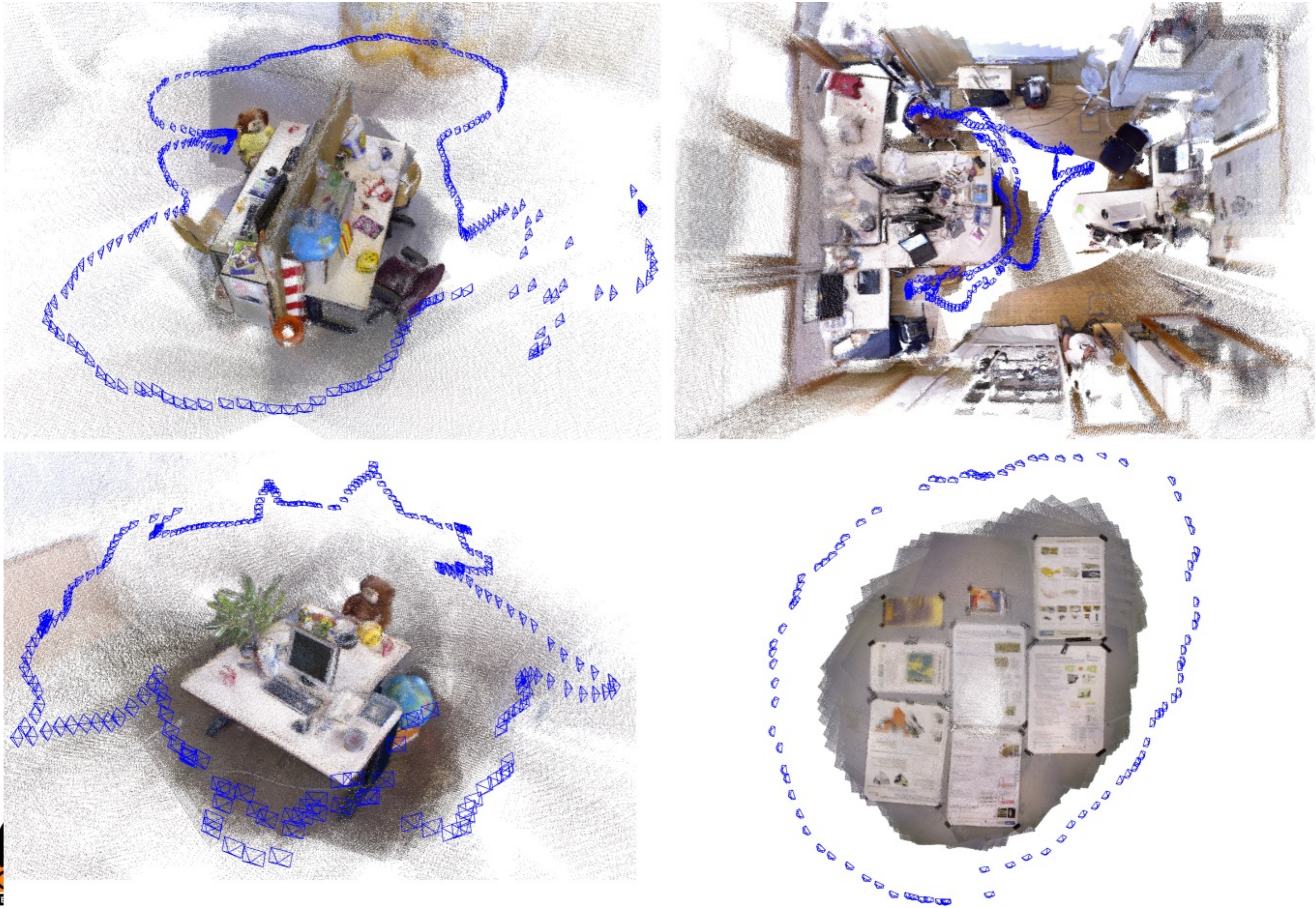
Instituto Universitario de Investigación en Ingeniería de Aragón
Universidad Zaragoza

ORB-SLAM2: an Open-Source SLAM System for Monocular, Stereo and RGB-D Cameras

Raúl Mur-Artal and Juan D. Tardós

raulmur@unizar.es tardos@unizar.es

RGB-D Point Cloud Reconstructions



RGB-D Point Cloud Reconstruction

Accuracy in TUM RGB-D Dataset

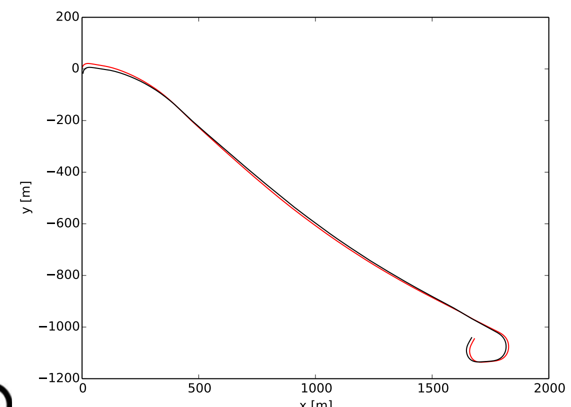
Table 2: TUM RGB-D Dataset. Comparison of Translation RMSE (m).

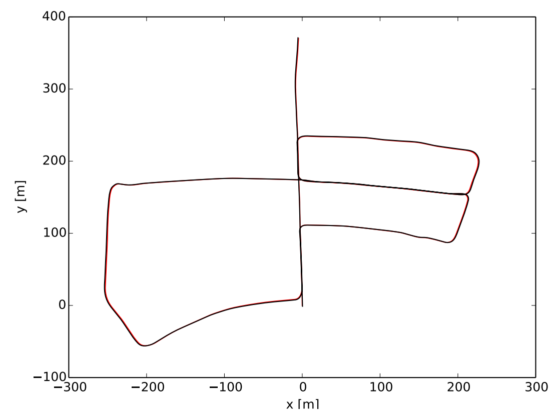
Sequence	ORB-SLAM2 (RGB-D)	Elastic-Fusion	Kintinuous	DVO SLAM	RGBD SLAM
fr1/desk	**0.016**	0.020	0.037	0.021	0.026
fr1/desk2	**0.022**	0.048	0.071	0.046	-
fr1/room	0.047	0.068	0.075	**0.043**	0.087
fr2/desk	**0.009**	0.071	0.034	0.017	0.057
fr2/xyz	**0.004**	0.011	0.029	0.018	-
fr3/office	**0.010**	0.017	0.030	0.035	-
fr3/nst	0.019	**0.016**	0.031	0.018	-

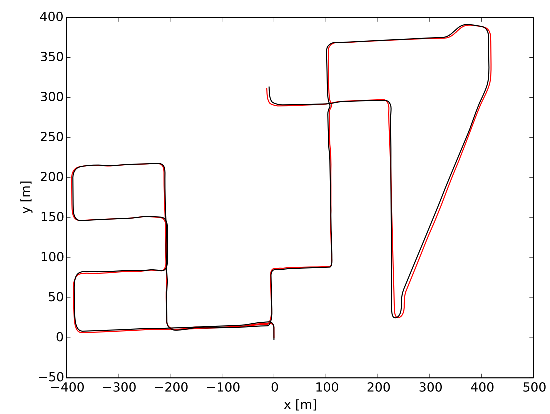
- BA gives better accuracy than ICP
- And is less computationally expensive

Accuracy in the KITTI Dataset

	ORB-SLAM2 (Stereo)			Stereo LSD-SLAM		
Error (Units)	t_{rel} (%)	r_{rel} (deg/100m)	t_{abs} (m)	t_{rel} (%)	r_{abs} (deg/100m)	t_{abs} (m)
00	0.70	**0.25**	1.3	**0.63**	0.26	**1.0**
01	**1.39**	**0.21**	10.4	2.36	0.36	**9.0**
02	**0.76**	0.23	5.7	0.79	0.23	**2.6**
03	**0.71**	**0.18**	**0.6**	1.01	0.28	1.2
04	0.48	**0.13**	0.2	**0.38**	0.31	0.2
05	**0.40**	**0.16**	**0.8**	0.64	0.18	1.5
06	**0.51**	**0.15**	**0.8**	0.71	0.18	1.3
07	**0.50**	**0.28**	0.5	0.56	0.29	0.5
08	**1.05**	0.32	**3.6**	1.11	**0.31**	3.9
09	**0.87**	0.27	**3.2**	1.14	**0.25**	5.6
10	**0.60**	**0.27**	**1.0**	0.72	0.33	1.5





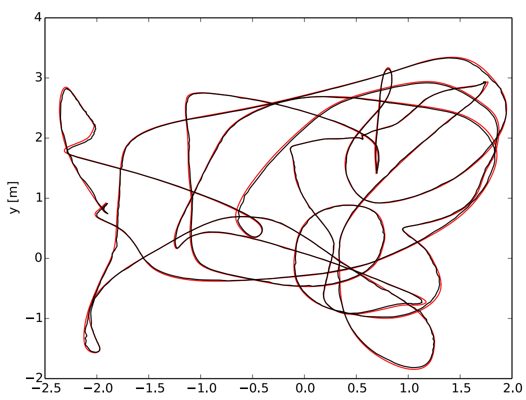


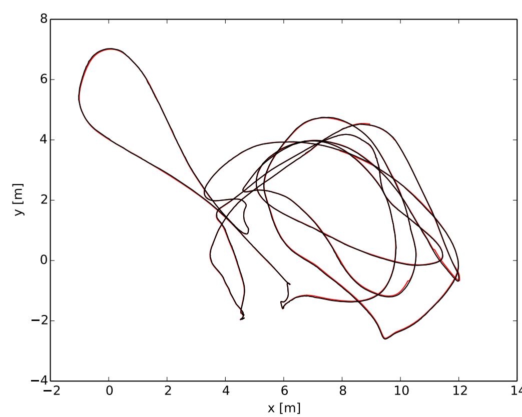
ORB-SLAM2 Stereo: Challenging Lighting

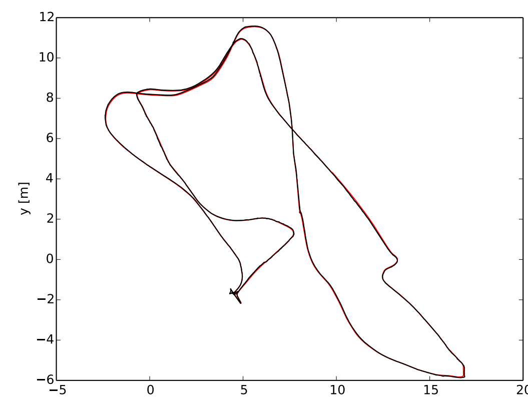
ORB-SLAM2 Stereo in EuRoC dataset

Accuracy in the EuRoC Dataset

	ORB-SLAM2 (Stereo)	Stereo LSD-SLAM
V1_01_easy	**0.035**	0.066
V1_02_medium	**0.020**	0.074
V1_03_difficult	**0.048**	0.089
V2_01_easy	0.037	-
V2_02_medium	0.035	-
V2_03_difficult	X	-
MH_01_easy	0.035	-
MH_02_easy	0.018	-
MH_03_medium	0.028	-
MH_04_difficult	0.119	-
MH_05_difficult	0.060	-







ORB-SLAM3: Visual Results

			MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg[1]
Monocular	ORB-SLAM [4]	ATE[2,3]	0.071	0.067	0.071	0.082	0.060	**0.015**	0.020	-	**0.021**	**0.018**	-	0.047*
	DSO [27]	ATE	0.046	0.046	0.172	3.810	0.110	0.089	0.107	0.903	0.044	0.132	1.152	0.601
	SVO [24]	ATE	0.100	0.120	0.410	0.430	0.300	0.070	0.210	-	0.110	0.110	1.080	0.294*
	DSM [31]	ATE	0.039	0.036	0.055	**0.057**	0.067	0.095	0.059	0.076	0.056	0.057	**0.784**	**0.126**
	ORB-SLAM3 (ours)	ATE	**0.017**	**0.017**	**0.031**	0.066	**0.044**	0.033	**0.016**	**0.037**	**0.021**	0.022	-	0.030*
Stereo	ORB-SLAM2 [3]	ATE	0.035	**0.018**	0.028	0.119	0.060	**0.035**	**0.020**	**0.048**	0.037	0.035	-	0.044*
	VINS-Fusion [44]	ATE	0.540	0.460	0.330	0.780	0.500	0.550	0.230	-	0.230	0.200	-	0.424*
	SVO [24]	ATE	0.040	0.070	0.270	0.170	0.120	0.040	0.040	0.070	0.050	0.090	0.790	0.159
	ORB-SLAM3 (ours)	ATE (m)	**0.025**	0.022	**0.027**	**0.089**	**0.058**	**0.035**	0.021	0.049	**0.032**	**0.027**	**0.361**	**0.068**

	SLAM or VO	Pixels used	Data association	Estimation	Relocation	Loop closing	Multi Maps	Mono	Stereo	Mono IMU	Stereo IMU	Fisheye	Accuracy	Robustness	Open source
SVO [23], [24]	VO	FAST+ Hi.grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Very Good	Very Good	[25][2]
ORB-SLAM2 [2], [3]	SLAM	ORB	Descriptor	Local BA	DBoW2	DBoW2 PG+BA	-	✓	✓	-	-	-	Exc.	Very Good	[26]
DSO [27]–[29]	VO	High grad.	Direct	Local BA	-	-	-	✓	✓	-	-	✓	Good	Very Good	[30]
DSM [31]	SLAM	High grad.	Direct	Local BA	-	-	-	✓	-	-	-	-	Very Good	Very Good	[32]
VINS-Fusion [7], [44]	VO	Shi Tomasi	KLT	Local BA	DBoW2	DBoW2 PG	✓	-	✓	✓	✓	✓	~~Very Good~~	~~Exc.~~	[45]

Short & Mid-term DA

Long-term DA

With IMU

ORB-SLAM3: Comparison

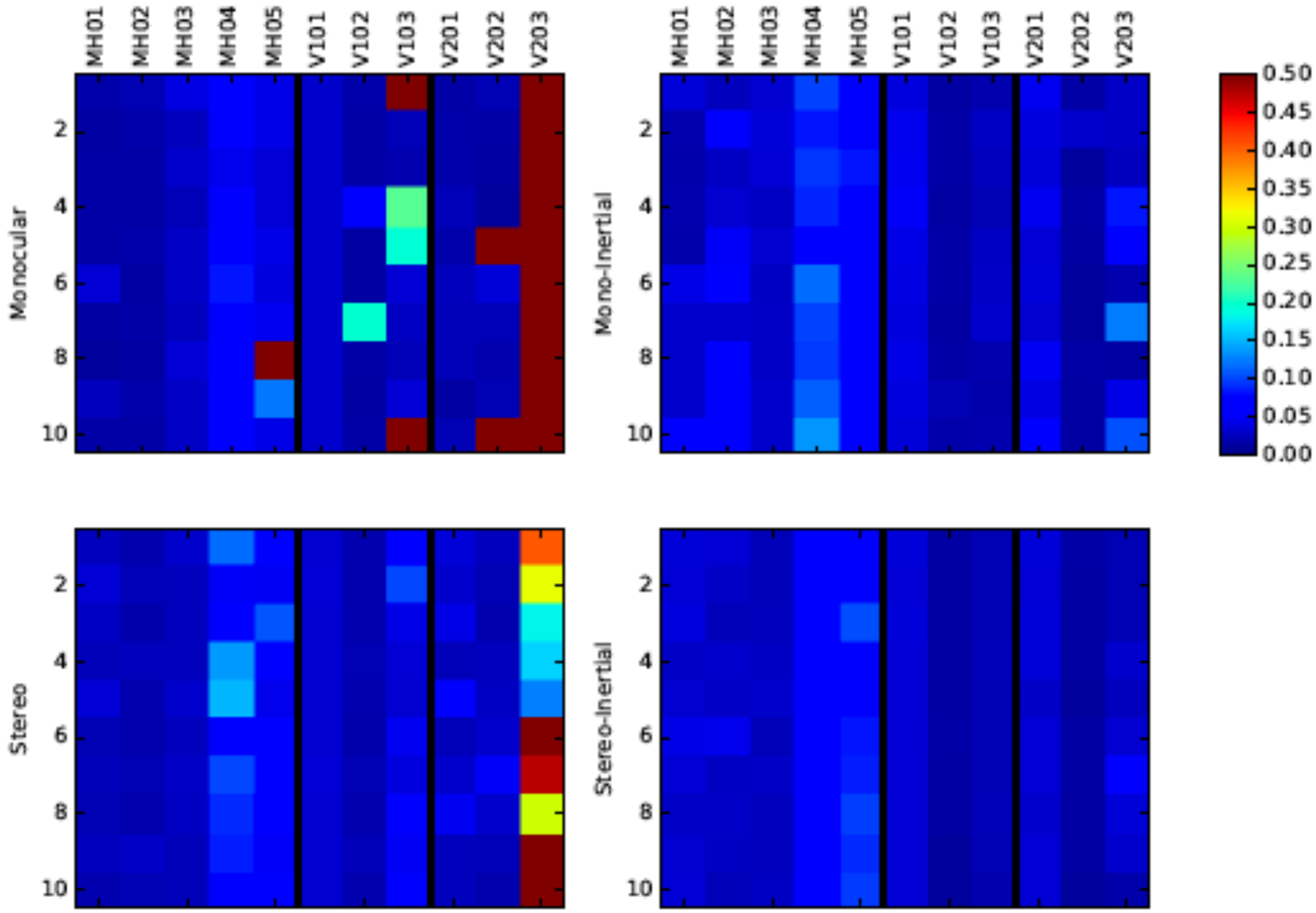


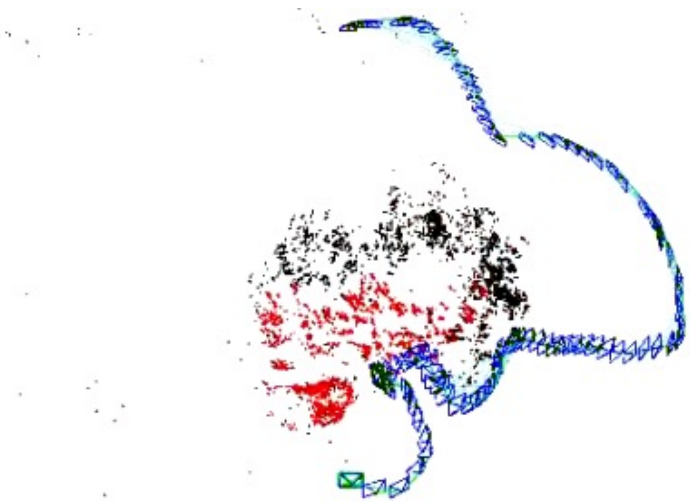
Figure 4: Colored squares represent the RMS ATE for ten different execution in each sequence of the EuRoC dataset.

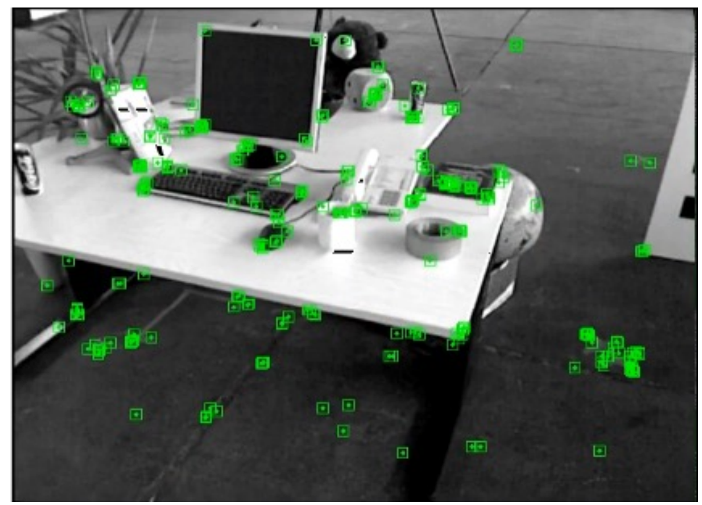
4.5.d Conclusions

- Monocular SLAM
  - Scale is unobservable
  - Scale drift produces innacuracy
  - Less robust: needs two views with parallax to initialize points
    - » Avoid pure rotation during exploration
    - » Avoid fast motions
- Stereo & RGB-D SLAM
  - Obtains true scale
  - More accurate
  - More robust: point initialization from one view
- Mono-Inertial & Stereo Inertial (see Lesson 6)
  - Obtains true scale
  - Excellent accuracy and robustness

Lesson 5. Optimization in SLAM

Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



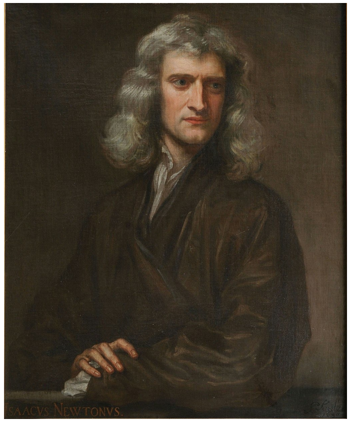


$$\{\mathbf{T}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{K}_1,\ j \in \mathcal{P}_1\}^* = \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{k \in \{\mathcal{K}_1, \mathcal{K}_2\},\, j \in \mathcal{P}_1} \rho\left(\left\| \mathbf{u}_{kj} - \pi_k\left(\mathbf{T}_{kw}\mathbf{x}_{wj}\right) \right\|^2_{\Sigma_{kj}} \right)$$

Lesson 5. Optimization in SLAM

1. Optimization algorithms
2. Lie groups
3. Optimization in Visual SLAM
   - Bundle Adjustment
   - Pose-graph

Standing on the Shoulders of Giants



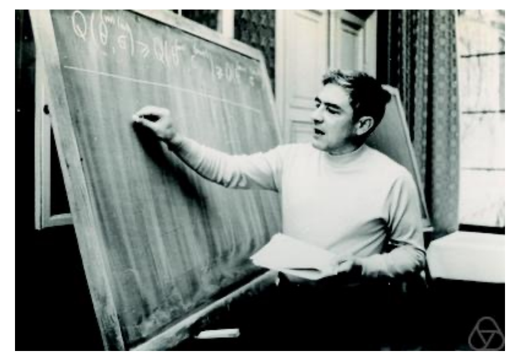
Isaac Newton
1643 – 1727

Carl F. Gauss
1777 – 1875

Sophus Lie
1842 – 1899

Kenneth Levenberg
1919 – 1973

Donald W. Marquardt
1929 – 1997



Peter J. Huber
1934 –

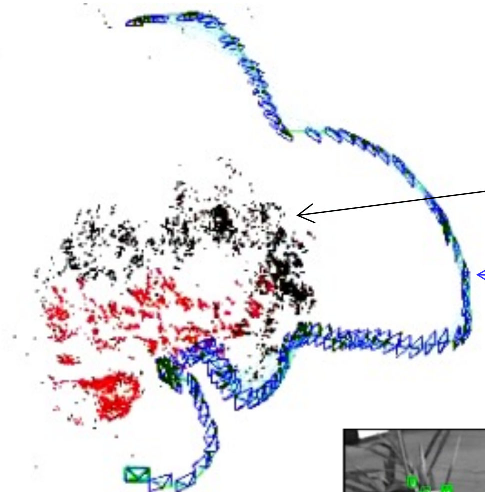
Lesson 5. Optimization in SLAM

1. Optimization algorithms
    a. Newton method
        - Levenberg-Marquardt
    b. Gauss-Newton
    c. Robust cost functions
        - Huber, Cauchy

Readings:

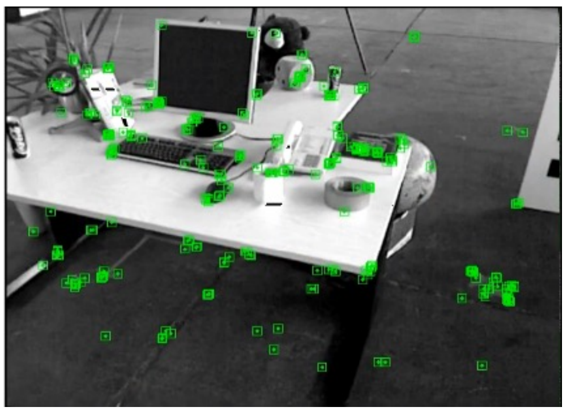
- B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon
  Bundle Adjustment — A Modern Synthesis,
  in Vision Algorithms: Theory andPractice, Springer, 2000

Feature-Based Visual SLAM as BA



States

$\mathbf{x}_{wj} \in \mathbb{R}^3$	Coordinates of point j
$\mathbf{R}_{iw} \in SO(3)$	Orientation of camera i
$\mathbf{t}_{iw} \in \mathbb{R}^3$	Position of camera i



Measurements

$$\mathbf{u}_{ij} = \begin{bmatrix} u_{ij} \\ v_{ij} \end{bmatrix}$$

Observation of point *j* from camera *i*

Bundle Adjustment

$$\{\mathbf{T}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{K}_1,\ j \in \mathcal{P}_1\}^* = \underbrace{\operatorname*{argmin}_{\mathbf{T}_{iw}, \mathbf{x}_{wj}}}_{\text{Non-Linear Optimization}} \sum_{k \in \{\mathcal{K}_1, \mathcal{K}_2\},\, j \in \mathcal{P}_1} \rho\left(\underbrace{\left\| \mathbf{u}_{kj} - \pi_k\left(\mathbf{T}_{kw}\mathbf{x}_{wj}\right)\right\|^2_{\Sigma_{kj}}}_{\text{Reprojection error}} \right)$$

5.1 Optimization Algorithms: Notation

- Residual: $\triangle z(x) \equiv \underline{z} - z(x)$

Observations

Predicted Observations

Current State Estimation

  » with m = dim(z), n = dim(x)

- Cost function: $f(x)$ dim(f) = 1
  - Gradient $g \equiv \frac{df}{dx}$ n x 1
  - Hessian $H \equiv \frac{d^2f}{dx^2}$ n x n
- Jacobian: $J \equiv \frac{dz}{dx}$ m x n

5.1.a Newton Method

- 2^{nd} order approximation of the cost function:

$$f(x+\delta x) \approx f(x) + g^\top \delta x + \tfrac{1}{2}\delta x^\top H\, \delta x$$
quadratic local model

$$g \equiv \frac{df}{dx}(x)$$
gradient vector

$$H \equiv \frac{d^2 f}{dx^2}(x)$$
Hessian matrix

- At the minimum: $\frac{df}{dx}(x+\delta x) \approx H\,\delta x + g = 0$

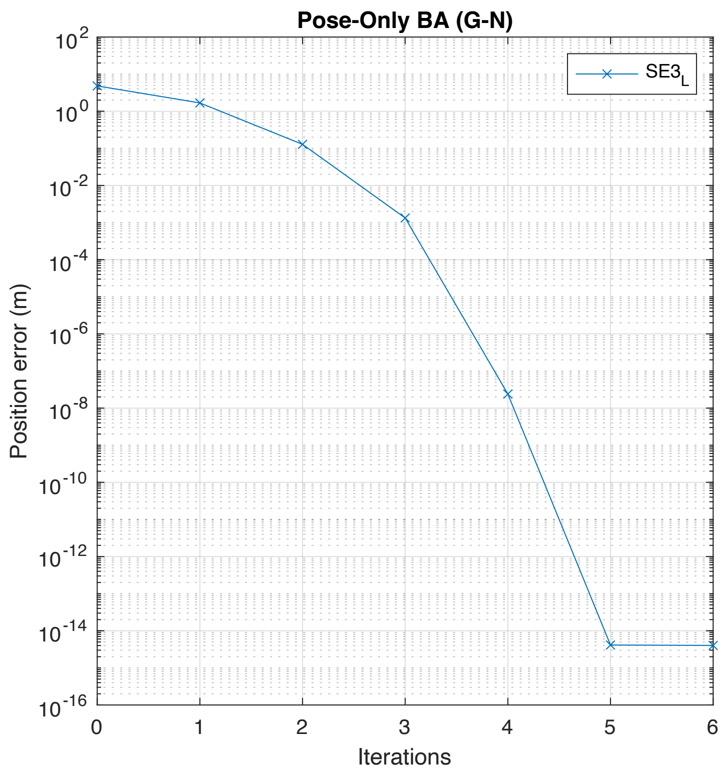
- Newton Step:

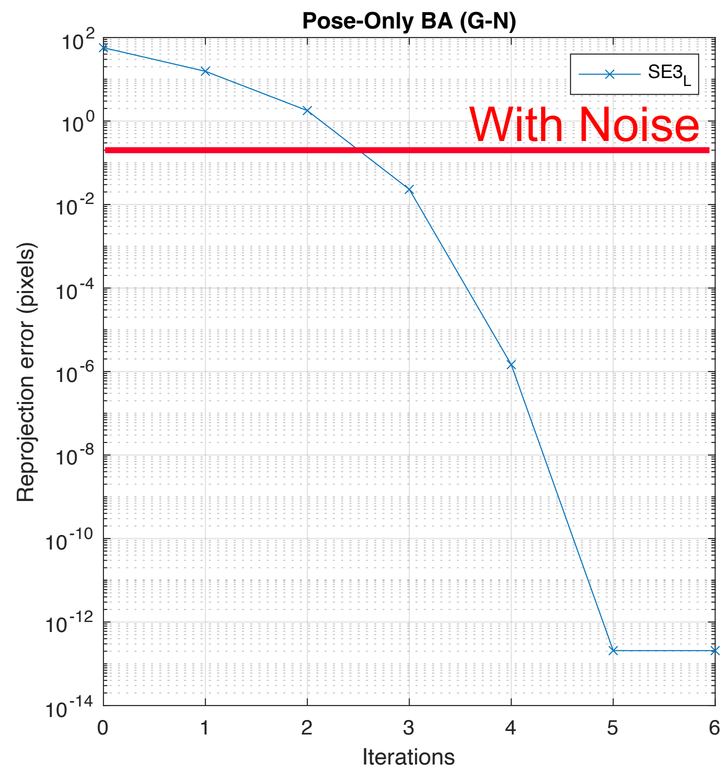
$$\boxed{\delta x = -H^{-1} g}$$

$$x \rightarrow x + \delta x$$

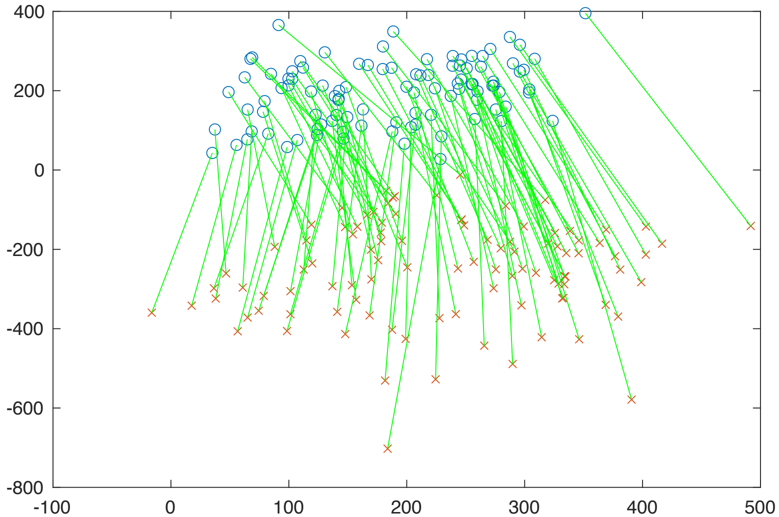
Convergence of Newton Method

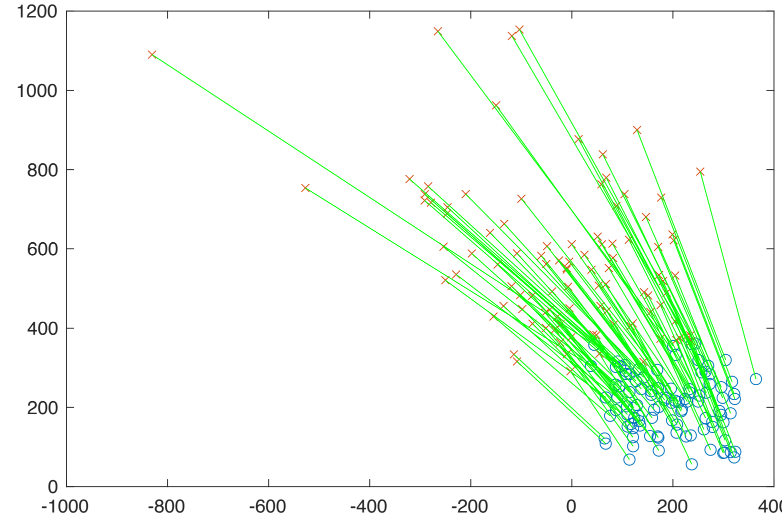
- If the cost is exactly a quadratic function of the state, it converges in one step (as in the Kalman Filter)
- Quadratic asymptotic convergence:
  - Close to the optimum, the state error is squared at each iteration
  - It converges in 5-6 steps to double accuracy (3 steps in practice)

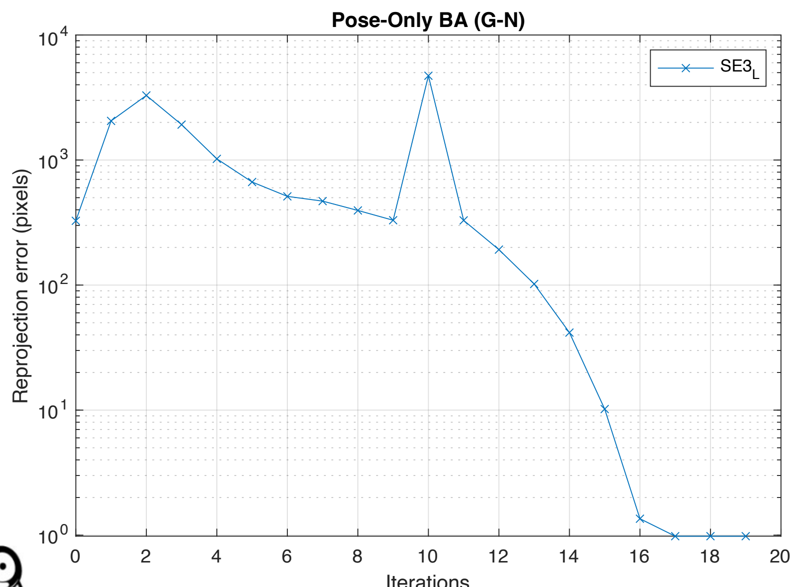


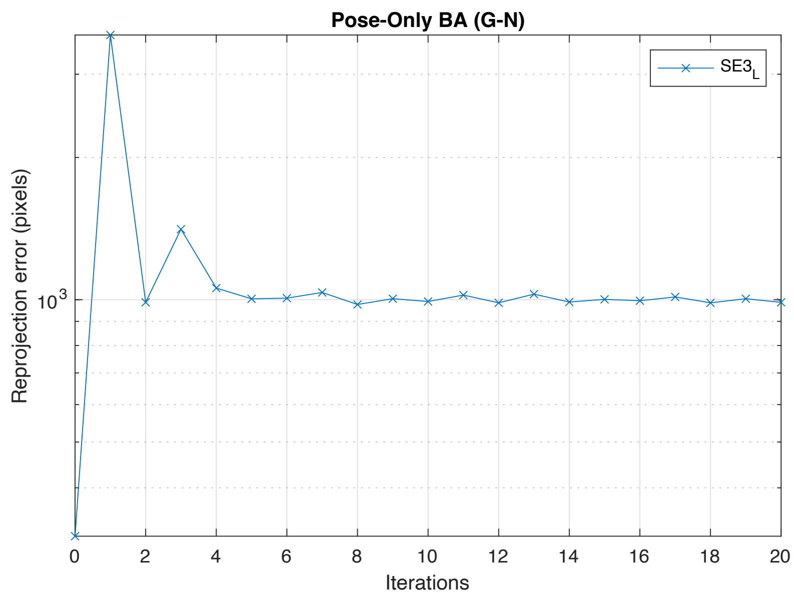


But, if we are Far from the Minimum….









Properties of Newton Method

- Needs a suitable step control policy to guarantee convergence when far from the minimum
- In a dense system, inverting the Hessian is $O(n^3)$
  - But Visual SLAM and BA are sparse problems!!!
  - Take profit from the sparsity of Hessian!!!
- For very large problems, approximate 1st order systems can be used, but they have linear convergence.
- Computing Hessian is not trivial (high implementation and computational costs)
  - Gauss-Newton method (analytical approximation of H)
  - Numerical approximations of H (Krylov methods)

Step Control

- Newton method can fail:
  - It can converge to a saddle point
  - For large steps, the 2^{nd} order cost prediction is inaccurate
  - No guarantee that the actual cost will decrease
- To guarantee cost decrease:
  - Follow a gradient descent direction
  - And make reasonable progress in that direction (step control)
    - » Too little, and it will converge slowly
    - » Too much, and it will overshoot the minimum

Damped Newton Methods

- Instead of: $\boxed{\delta \mathrm{x} = -\mathrm{H}^{-1}\mathrm{g}}$

- Solve: $\boxed{(\mathrm{H} + \lambda \mathrm{W})\, \delta \mathrm{x} = -\mathrm{g}}$ W: weight matrix, identity or diag(H)

  – Small lambda → Newton direction (faster)
  – Big lambda → Gradient descent direction (safer)

- Trust-region methods: choose lambda to limit the step to a maximum dynamic size

- Levenberg-Marquardt (L-M) methods: perform heuristic changes of lambda:

  – Bad convergence: increase lambda → Gradient descent
  – Good convergence: decrease lambda → Newton direction

An Implentation of Levenberg-Marquardt

Algorithm 2.1 The Levenberg-Marquardt algorithm

function LM$(g(), X^0)$ ▷ quadratic cost function $g()$, ▷ initial estimate X^0
    $\lambda = 10^{-4}$
    $t = 0$
    repeat
        $A, b \leftarrow$ linearize $g(X)$ at X^t
        $\Delta \leftarrow$ solve $\left(A^\top A + \lambda\, \mathrm{diag}(A^\top A)\right) \Delta = A^\top b$
        if $\mathrm{g}(X^t + \Delta) < \mathrm{g}(X^t)$ **then**
            $X^{t+1} = X^t + \Delta$ ▷ accept update
            $\lambda \leftarrow \lambda / 10$
        else
            $X^{t+1} = X^t$ ▷ reject update
            $\lambda \leftarrow \lambda * 10$
        $t \leftarrow t + 1$
    until convergence
    return X^t ▷ return latest estimate

From F. Dellaert & M Kaess: Factor Graphs for Robot Perception, 2017

Line-Search Methods

- Once you have obtained a descent direction solving:

$$\boxed{(\mathsf{H} + \lambda \mathsf{W})\, \delta\mathsf{x} \;=\; -\mathsf{g}}$$

- Perform a 1D search in that direction to minimize the cost function:

$$\boxed{\mathsf{x} + \alpha\, \delta\mathsf{x}}$$

  – Most techniques use a quadratic or cubic 1D cost model

5.1.b Gauss-Newton (Weighted Least Squares)

- Non linear weighted SSE cost function:

$$\boxed{f(x) \equiv \frac{1}{2} \triangle z(x)^{\top} W \triangle z(x)}$$

$$\triangle z(x) = \underline{z} - z(x)$$

weight matrix W

$$\boxed{J \equiv \frac{dz}{dx}}$$

$$g \equiv \frac{df}{dx} = -\triangle z^{\top} W J$$

$$H \equiv \frac{d^2 f}{dx^2} = J^{\top} W J + \sum_i (\triangle z^{\top} W)_i \frac{d^2 z_i}{dx^2}$$

- Gauss-Newton approximation: $\boxed{H \approx J^{\top} W J}$
  - Good if prediction error is small
  - Or if the observation is almost linear $\frac{d^2 z_i}{dx^2} \approx 0$
- Gauss-Newton or **normal equation**:

$$\boxed{(J^{\top} W J)\, \delta x = +J^{\top} W \triangle z}$$

Relationship to EKF

- Measurement equation:

$$\mathbf{z}_i = \mathbf{h}_i(\mathbf{x}) + \mathbf{w}_i \quad ; \quad \mathbf{R}_i = Cov(\mathbf{w}_i)$$

- Stacking measurements:

$$\mathbf{z} = \begin{pmatrix} \mathbf{z}_1 \\ \mathbf{z}_2 \\ \dots \\ \mathbf{z}_m \end{pmatrix} \qquad \mathbf{h}(\mathbf{x}) = \begin{pmatrix} \mathbf{h}_1(\mathbf{x}) \\ \mathbf{h}_2(\mathbf{x}) \\ \dots \\ \mathbf{h}_m(\mathbf{x}) \end{pmatrix} \qquad \mathbf{R} = \begin{pmatrix} \mathbf{R}_1 & 0 & \dots & 0 \\ 0 & \mathbf{R}_2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \mathbf{R}_m \end{pmatrix}$$

- Using as weight matrix:

$$\mathbf{W} = \mathbf{R}^{-1} = \begin{pmatrix} \mathbf{R}_1^{-1} & 0 & \dots & 0 \\ 0 & \mathbf{R}_2^{-1} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & 0 & \mathbf{R}_m^{-1} \end{pmatrix}$$

- G-N cost function = Mahalanobis distance

$$\boxed{\mathsf{f}(\mathsf{x}) = \tfrac{1}{2} \triangle \mathsf{z}(\mathsf{x})^\top \mathsf{W} \triangle \mathsf{z}(\mathsf{x})}$$

Relationship to EKF

State prediction: $\hat{\mathbf{x}} \ ; \ \mathbf{P}$

Measurements: $\mathbf{z} = \mathbf{h}(\mathbf{x}) + \mathbf{w} \ ; \ \mathbf{R} = Cov(\mathbf{w})$ $\qquad \mathbf{H} = \left.\frac{d\mathbf{h}(\mathbf{x})}{d\mathbf{x}}\right|_{\hat{\mathbf{x}}}$

Cost function: $f(\mathbf{x}) = \frac{1}{2}(\mathbf{x} - \hat{\mathbf{x}})^\top \mathbf{P}^{-1}(\mathbf{x} - \hat{\mathbf{x}}) + \frac{1}{2}(\mathbf{z} - \mathbf{h}(\mathbf{x}))^\top \mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\mathbf{x}))$

$$\mathbf{g} = \mathbf{P}^{-1}(\mathbf{x} - \hat{\mathbf{x}}) - \mathbf{H}^\top \mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\mathbf{x}))$$

$$\mathbf{Hess} = \mathbf{P}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1}\mathbf{H} = \mathbf{P}_{k+1}^{-1}$$

Hessian = Information Matrix

Gauss-Newton equation:

$$\Delta\mathbf{x} = -\mathbf{Hess}^{-1}\left.\mathbf{g}\right|_{\hat{\mathbf{x}}} = \left(\mathbf{P}^{-1} + \mathbf{H}^\top \mathbf{R}^{-1}\mathbf{H}\right)^{-1}\mathbf{H}^\top \mathbf{R}^{-1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}))$$

Using matrix inv. lema: $(A + BD^{-1}C)BD^{-1} = A^{-1}B(D + CA^{-1}B)^{-1}$

We get EKF update: $\Delta\mathbf{x} = \mathbf{P}\mathbf{H}^\top\left(\mathbf{R} + \mathbf{H}\mathbf{P}\mathbf{H}^\top\right)^{-1}(\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}))$

$$Cov(\mathbf{x}_{k+1}) = \mathbf{P}_{k+1} = \mathbf{Hess}^{-1}$$

EKF .vs. Gauss-Newton

One-shot:

$$\boxed{\Delta \mathbf{x} = \mathbf{P}\mathbf{H}^\top \left(\mathbf{R} + \mathbf{H}\mathbf{P}\mathbf{H}^\top\right)^{-1} (\mathbf{z} - \mathbf{h}(\hat{\mathbf{x}}))}$$

Iterate:

$$\boxed{\begin{aligned} \delta \mathsf{x} &= -\mathsf{H}^{-1}\mathsf{g} \\ \mathsf{x} &\rightarrow \mathsf{x} + \delta \mathsf{x} \end{aligned}}$$

- EKF performs just one Gauss-Newton iteration
- Followed by past state marginalization
  - Reduces the state dimension
  - But correlates all the features, losing sparsity
- Iterated EKF re-linearizes the current observation, but cannot correct linearization errors in past observations
- The Hessian of Gauss-Newton can be inverted to obtain the covariances of the state variables
  - But it’s expensive, usually avoided
  - Use visual clues for data association, instead of covariances

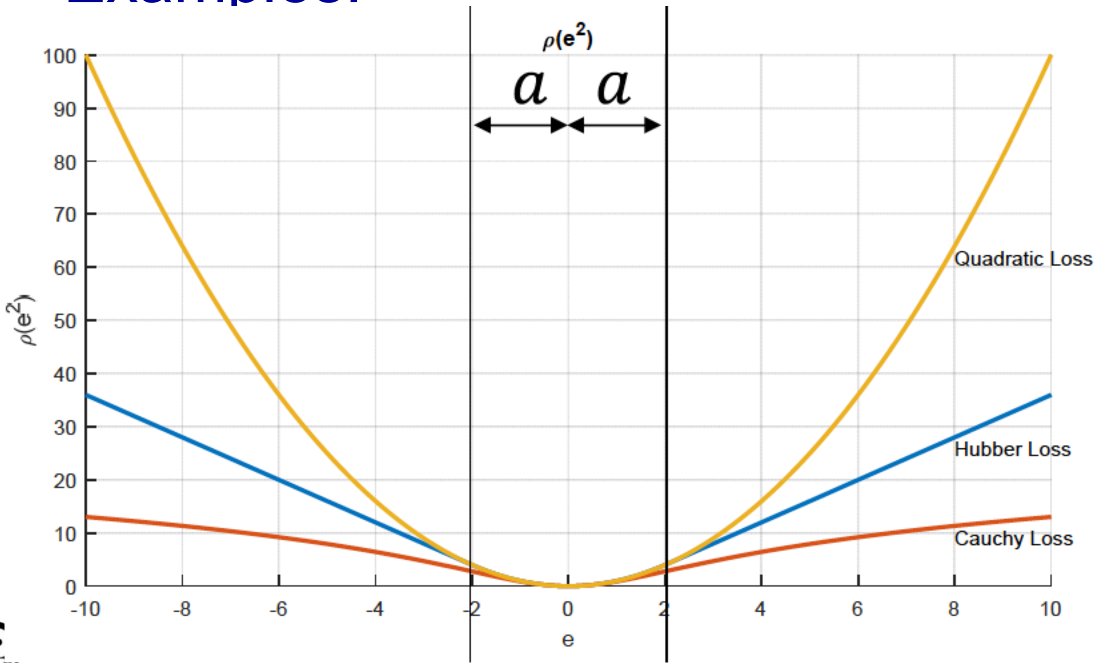
5.1.c Robust Cost Functions

- To down-weight large errors (spurious data)

$$\boxed{\mathsf{f}_i(\mathsf{x}) \equiv \frac{1}{2}\rho_i(\,\triangle\mathsf{z}_i(\mathsf{x})^\top \mathsf{W}_i \triangle\mathsf{z}_i(\mathsf{x})\,)}$$

  - where: $\rho_i(s)$ is an increasing function
  - with: $\rho_i(0) = 0$ and $\frac{d}{ds}\rho_i(0) = 1$
  - for standard SSE: $\rho_i(s) = s$

- Examples:



Huber

$$\rho(s) = \begin{cases} s & s \le a \\ 2a^2\sqrt{\dfrac{s}{a^2}} - 1 & s > a \end{cases}$$

Cauchy

$$\rho(s) = a^2\left(\log\left(\frac{s}{a^2}\right) + 1\right)$$

Robustified Gauss-Newton

$$\boxed{\mathsf{f}_i(\mathsf{x}) \equiv \tfrac{1}{2}\,\rho_i(\,\triangle\mathsf{z}_i(\mathsf{x})^\top \mathsf{W}_i\,\triangle\mathsf{z}_i(\mathsf{x})\,)}$$

$$\boxed{\delta\mathsf{x} = -\mathsf{H}^{-1}\mathsf{g}}$$

- Robustified Gauss-Newton approximation:

$$\mathsf{g}_i = -\rho_i'\,\mathsf{J}_i^\top \mathsf{W}_i\,\triangle\mathsf{z}_i$$

$$\mathsf{H}_i \approx \mathsf{J}_i^\top\left(\rho_i'\,\mathsf{W}_i + 2\,\rho_i''\,(\mathsf{W}_i\,\triangle\mathsf{z}_i)\,(\mathsf{W}_i\,\triangle\mathsf{z}_i)^\top\right)\mathsf{J}_i$$

- Some methods just use:

$$\mathsf{H}_i \approx \mathsf{J}_i^\top\left(\rho_i'\,\mathsf{W}_i + \cancel{2\,\rho_i''\,(\mathsf{W}_i\,\triangle\mathsf{z}_i)\,(\mathsf{W}_i\,\triangle\mathsf{z}_i)^\top}\right)\mathsf{J}_i$$

  – Reweighted Least-Squares: using $\rho_i'\,\mathsf{W}_i$ instead of W_i

5.2 Lie Groups

Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB

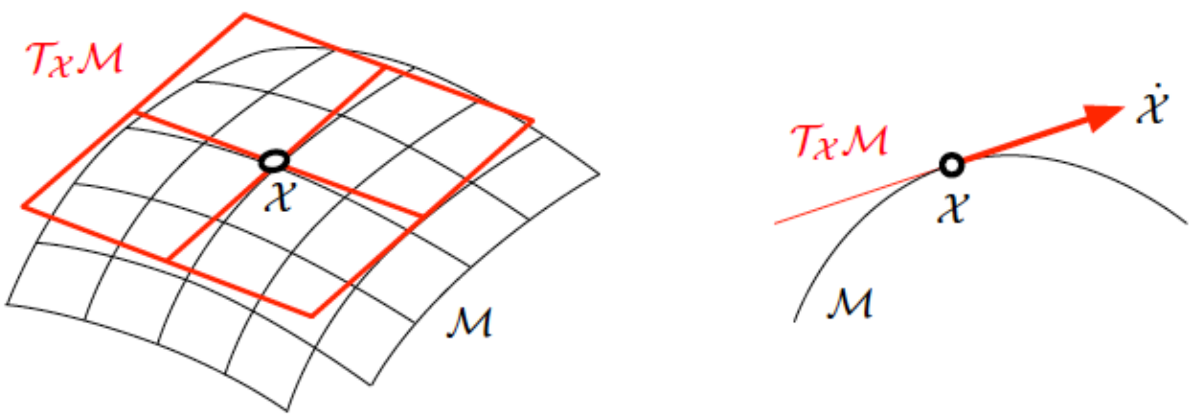


Figure credit: (Sola et al. 2020)

Lesson 5. Optimization in SLAM

2. Lie Groups
    a. Lie Group theory
    b. Rotations: SO(3)
    c. Motions: SE(3)
    d. Jacobians in SE(3)
    e. Sim(3)

Readings:

- Joan Solà, Jeremie Deray, Dinesh Atchuthan
  A micro Lie theory for state estimation in robotics, arXiv 2020
- Hauke Strasdat
  Local accuracy and global consistency for efficient visual SLAM, PhD Thesis, Imperial College, 2012 (chapters 2, A, B)

5.2.a Lie Group theory

- Group: a set with an operation $(\mathcal{G}, \circ)$ such that:

$$
\begin{array}{rcl}
\text{Closure under `}\circ\text{'} & : & \mathcal{X} \circ \mathcal{Y} \in \mathcal{G} \\
\text{Identity } \mathcal{E} & : & \mathcal{E} \circ \mathcal{X} = \mathcal{X} \circ \mathcal{E} = \mathcal{X} \\
\text{Inverse } \mathcal{X}^{-1} & : & \mathcal{X}^{-1} \circ \mathcal{X} = \mathcal{X} \circ \mathcal{X}^{-1} = \mathcal{E} \\
\text{Associativity} & : & (\mathcal{X} \circ \mathcal{Y}) \circ \mathcal{Z} = \mathcal{X} \circ (\mathcal{Y} \circ \mathcal{Z})
\end{array}
$$

- Some examples:
  - Translations R^3: $t_{AC} = t_{AB} + t_{BC}$
  - Rotation Matrices SO(3): $R_{AC} = R_{AB}\ R_{BC}$
  - Rigid Motions SE(3): $T_{AC} = T_{AB}\ T_{BC}$

Manifolds and Tangent Space

- A Manifold is a smooth surface that is locally similar to R^n

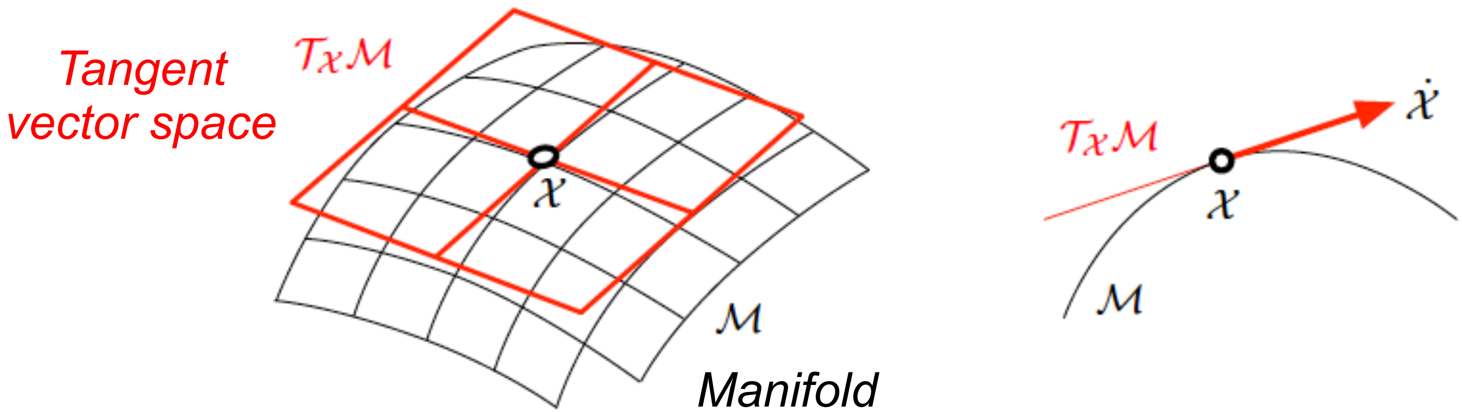


Figure 2. A manifold $\mathcal{M}$ and the vector space $T_\mathcal{X}\mathcal{M}$ (in this case $\cong \mathbb{R}^2$) tangent at the point $\mathcal{X}$, and a convenient side-cut. The velocity element, $\dot{\mathcal{X}} = \partial\mathcal{X}/\partial t$, does not belong to the manifold $\mathcal{M}$ but to the tangent space $T_\mathcal{X}\mathcal{M}$.

Credit: (Sola et al. 2020)

- A 3D sphere belongs to R^3, but is locally similar to R^2
- A rotation matrix belongs to R^9, but is locally similar to R^3
- A rigid motion matrix belongs to R^{16}, but is locally similar to R^6

State Estimation in Manifolds

- In robotics and computer vision, many state variables live in manifolds, that are groups: orientations, poses,…
- If we update their estimations using a vectorial sum:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}_k$$

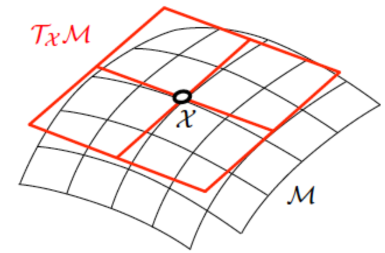
  the estimation will "fly away" from the manifold

- Solution:
  - Compute updates in a local tangent space,
  - Map the update to the manifold,
  - And update the state using the manifold's composition.
    - » As it is a group, the state will always stay in the manifold!

$$\boxed{\mathbf{x}_{k+1} = \mathbf{x}_k \oplus \mathbf{d}_k}$$

Lie Group

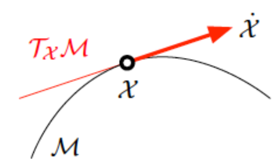
- A **Lie Group** is a *group* that is also a *smooth manifold*



$$\mathcal{X} \in \mathcal{M}$$

- Velocities belong to the *tangent space*:

$$\mathcal{X}(t) \in \mathcal{M} \qquad \dot{\mathcal{X}} = \partial\mathcal{X}/\partial t \in T_\mathcal{X}\mathcal{M}$$



- Examples of Lie groups:
  - SO(3): Special Orthogonal Group 3D (Rotation Matr. 3x3)
  - SE(3): Special Euclidean Group 3D (Homog. Transf. 4x4)
  - Sim(3): Similarity Transformation (motion + scale change)
  - S^3: 4D Unit sphere (quaternions)
  - Their 2D versions: SO(2), SE(2), Sim(2)

Lie Algebra

- The tangent space of a Lie Group represents velocities

$$\mathcal{X}(t) \in \mathcal{M} \qquad\qquad \dot{\mathcal{X}} = \partial\mathcal{X}/\partial t \in T_{\mathcal{X}}\mathcal{M}$$

- The **Lie Algebra** is the tangent space at the identity $\mathcal{E}$

$$\boxed{\mathfrak{m} \triangleq T_{\mathcal{E}}\mathcal{M}}$$

  - The Lie Algebra is a vector space, isomorphic to $\boxed{\mathfrak{m} \cong \mathbb{R}^m}$
    - » m : degrees of freedom of the group

Basis of $\mathfrak{m}$

$$\text{Hat}: \quad \mathbb{R}^m \to \mathfrak{m}; \qquad \boldsymbol{\tau} \mapsto \boldsymbol{\tau}^\wedge = \sum_{i=1}^{m} \tau_i E_i$$

Basis of $\mathbb{R}^m$

$$\text{Vee}: \quad \mathfrak{m} \to \mathbb{R}^m; \quad \boldsymbol{\tau}^\wedge \mapsto (\boldsymbol{\tau}^\wedge)^\vee = \boldsymbol{\tau} = \sum_{i=1}^{m} \tau_i \mathbf{e}_i$$

  - The Lie Algebra is used to represent velocities or displacements

$$\mathbf{v}^\wedge \qquad\qquad \boldsymbol{\tau}^\wedge = (\mathbf{v}t)^\wedge = \mathbf{v}^\wedge t$$

Form of the Lie Algebra

- The Lie Algebra of a Lie Group can be obtained by differentiating the group constraint

$$\boxed{\mathfrak{m} \triangleq T_{\mathcal{E}}\mathcal{M}}$$

- For a multiplicative group:

$$\mathcal{X}^{-1}\mathcal{X} = \mathcal{E} \quad \Longrightarrow \quad \mathcal{X}^{-1}\dot{\mathcal{X}} + \dot{\mathcal{X}^{-1}}\mathcal{X} = 0$$

- The elements of the Lie Algebra are:

$$\mathbf{v}^\wedge = \mathcal{X}^{-1}\dot{\mathcal{X}} = -\dot{\mathcal{X}^{-1}}\mathcal{X}$$

$$\boxed{\dot{\mathcal{X}}\Big|_{\mathcal{X}=\mathcal{E}} = \mathcal{E}\mathbf{v}^\wedge = \mathbf{v}^\wedge \in \mathfrak{m}}$$

- Differential equation:

$$\dot{\mathcal{X}} = \mathcal{X}\mathbf{v}^\wedge$$

$$\dot{x}(t) = a x(t) \implies x(t) = x_0\, e^{at}$$

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) \implies \mathbf{x}(t) = \mathbf{x}_0\, e^{\mathbf{A}t}$$

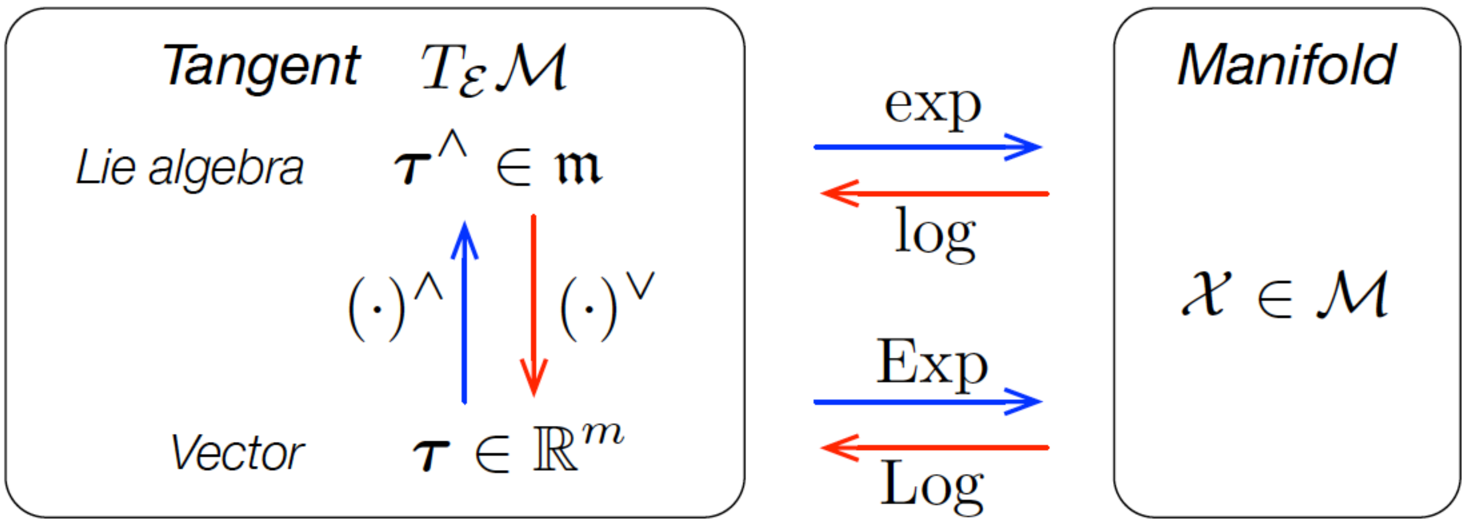
$$e^{\mathbf{X}} = \sum_{k=0}^{\infty} \frac{\mathbf{X}^k}{k!}$$

- Solution:

$$\mathcal{X}(t) = \mathcal{X}(0)\underbrace{\exp(\mathbf{v}^\wedge t)}_{\text{Belongs to the Lie Group}}$$

Exp and Log Maps

$$\mathcal{X} = \mathrm{Exp}(\boldsymbol{\tau}) \triangleq \exp(\boldsymbol{\tau}^\wedge)$$
$$\boldsymbol{\tau} = \mathrm{Log}(\mathcal{X}) \triangleq \log(\mathcal{X})^\vee$$



Credit: (Sola et al. 2020)

Adjoint Matrix

- Vectors of the tangent space at $\mathcal{X} \in \mathcal{M}$ can be transformed to the tangent space at the identity $\mathcal{E}$

$$\mathrm{Ad}_{\mathcal{X}} : \mathfrak{m} \to \mathfrak{m}; \quad \boldsymbol{\tau}^\wedge \mapsto \mathrm{Ad}_{\mathcal{X}}(\boldsymbol{\tau}^\wedge) \triangleq \mathcal{X}\boldsymbol{\tau}^\wedge\mathcal{X}^{-1}$$

- As it is a linear operation, we will use its matrix form:

$$\mathbf{Ad}_{\mathcal{X}} : \mathbb{R}^m \to \mathbb{R}^m; \quad {}^{\mathcal{X}}\boldsymbol{\tau} \mapsto {}^{\mathcal{E}}\boldsymbol{\tau} = \mathbf{Ad}_{\mathcal{X}}\,{}^{\mathcal{X}}\boldsymbol{\tau}$$

$$\mathcal{X} \circ \mathrm{Exp}({}^{\mathcal{X}}\boldsymbol{\tau}) = \mathrm{Exp}({}^{\mathcal{E}}\boldsymbol{\tau}) \circ \mathcal{X} = \mathrm{Exp}(\mathbf{Ad}_{\mathcal{X}}\,{}^{\mathcal{X}}\boldsymbol{\tau}) \circ \mathcal{X}$$

  – Or, defining $\oplus$ as composition with Exp

$$\mathcal{X} \oplus {}^{\mathcal{X}}\boldsymbol{\tau} = {}^{\mathcal{E}}\boldsymbol{\tau} \oplus \mathcal{X} = (\mathbf{Ad}_{\mathcal{X}}\,{}^{\mathcal{X}}\boldsymbol{\tau}) \oplus \mathcal{X}$$

5.2.b Rotation Group SO(3)

- Lie Group SO(3): $\mathbf{R} \in SO(3)$ $\mathbf{R}^T\mathbf{R} = \mathbf{I}_3$
  - Differentiating the constraint:

$$\mathbf{R}^T\dot{\mathbf{R}} + \dot{\mathbf{R}}^T\mathbf{R} = \mathbf{0} \implies \mathbf{R}^T\dot{\mathbf{R}} = -\left(\mathbf{R}^T\dot{\mathbf{R}}\right)^T$$

Must be anti-symmetric (a skew matrix)

- Lie Algebra so(3):

$$\mathbf{R}^T\dot{\mathbf{R}} = [\boldsymbol{\omega}]_\times = \begin{pmatrix} 0 & -\omega_z & \omega_y \\ \omega_z & 0 & -\omega_x \\ -\omega_y & \omega_x & 0 \end{pmatrix} \in so(3)$$

- Is isomorphic to the vector space:

$$\boldsymbol{\omega} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \in \mathbb{R}^3$$

Angular Velocity

Lie Algebra so(3)

- so(3) is the tangent space of SO(3) at the origin:

$$\dot{\mathbf{R}} = \mathbf{R}\,[\boldsymbol{\omega}]_\times \quad \Longrightarrow \quad \boxed{\left.\dot{\mathbf{R}}\right|_{\mathbf{R}=\mathbf{I}} = [\boldsymbol{\omega}]_\times}$$

- It's elements are:

$$\boldsymbol{\theta}^\wedge = [\boldsymbol{\theta}]_\times = \begin{pmatrix} 0 & -\theta_z & \theta_y \\ \theta_z & 0 & -\theta_x \\ -\theta_y & \theta_x & 0 \end{pmatrix} = \theta_x \mathbf{E}_1 + \theta_y \mathbf{E}_2 + \theta_z \mathbf{E}_3$$

$$\mathbf{E}_1 = \begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & -1 \\ 0 & 1 & 0 \end{pmatrix} \; ; \; \mathbf{E}_2 = \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 0 \\ -1 & 0 & 0 \end{pmatrix} \; ; \; \mathbf{E}_3 = \begin{pmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}$$

- They represent angular velocity or a (small) rotation

Understanding the Exponential Map

- Differential equation:

$$\mathbf{R}^T \dot{\mathbf{R}} = [\boldsymbol{\omega}]_\times \quad \Longrightarrow \quad \dot{\mathbf{R}} = \mathbf{R}\,[\boldsymbol{\omega}]_\times$$

- It's solution is:

$$\mathbf{R}(t) = \mathbf{R}_0 \exp\left([\boldsymbol{\omega}]_\times\, t\right) = \mathbf{R}_0 \exp\left([\boldsymbol{\theta}]_\times\right) = \mathbf{R}_0\,\mathrm{Exp}(\boldsymbol{\theta})$$

  – Note that successive rotations are expressed in the rotated frames, not in W:

$$\mathbf{R}_{W2} = \mathbf{R}_{W0}\,\mathbf{R}_{01}\,\mathbf{R}_{12} = \mathbf{R}_{W0}\,\mathrm{Exp}(\boldsymbol{\theta}_1)\,\mathrm{Exp}(\boldsymbol{\theta}_2)$$

Exponential and Logarithmic Maps

- Taking into account: $$e^{\mathbf{X}} = \sum_{k=0}^{\infty} \frac{\mathbf{X}^k}{k!}$$

- Exponential map (Rodrigues formula):

$$\boxed{\mathbf{R} = \mathrm{Exp}(\boldsymbol{\theta}) = \mathrm{Exp}(\mathbf{u}\theta) = \mathbf{I} + \sin\theta\,[\mathbf{u}]_\times + (1-\cos\theta)\,[\mathbf{u}]_\times^2}$$

- Logarithmic map:

$$\boxed{\theta = \cos^{-1}\left(\frac{\mathrm{trace}(\mathbf{R})-1}{2}\right)} \qquad \boxed{\boldsymbol{\theta} = \mathrm{Log}(\mathbf{R}) = \frac{\theta}{2\sin\theta}\left(\mathbf{R}-\mathbf{R}^T\right)^\vee}$$

- Derivative of Exp:

$$\frac{\partial\,\mathrm{Exp}(\boldsymbol{\theta})}{\partial\theta_i} = \frac{\partial e^{\boldsymbol{\theta}^\wedge}}{\partial\theta_i} = e^{\boldsymbol{\theta}^\wedge}\frac{\partial\boldsymbol{\theta}^\wedge}{\partial\theta_i} = \mathrm{Exp}(\boldsymbol{\theta})\mathbf{E}_i \qquad \boxed{\left.\frac{\partial\,\mathrm{Exp}(\boldsymbol{\theta})}{\partial\theta_i}\right|_{\boldsymbol{\theta}=\mathbf{0}} = \mathbf{E}_i}$$

$$\boldsymbol{\theta}^\wedge = \theta_x\mathbf{E}_1 + \theta_y\mathbf{E}_2 + \theta_z\mathbf{E}_3$$

5.2.c Rigid Motion Group SE(3)

- Lie Group SE(3):

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \qquad \begin{array}{c} \mathbf{R} \in SO(3) \\ \mathbf{t} \in \mathbb{R}^3 \end{array}$$

$$\mathbf{T}_1\mathbf{T}_2 = \begin{pmatrix} \mathbf{R}_1\mathbf{R}_1 & \mathbf{R}_1\mathbf{t}_2 + \mathbf{t}_1 \\ \mathbf{0} & 1 \end{pmatrix} \qquad \mathbf{T}^{-1} = \begin{pmatrix} \mathbf{R}^T & -\mathbf{R}^T\mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix}$$

- Lie Algebra se(3):

$$\mathbf{v}^\wedge = \begin{pmatrix} [\boldsymbol{\omega}]_\times & \boldsymbol{v} \\ \mathbf{0} & 0 \end{pmatrix} \in se(3)$$

- Isomorphic to:

$$\mathbf{v} = \begin{pmatrix} \boldsymbol{v} \\ \boldsymbol{\omega} \end{pmatrix} \in \mathbb{R}^6 \qquad \boldsymbol{v} = \begin{pmatrix} v_x \\ v_y \\ v_z \end{pmatrix} \in \mathbb{R}^3 \qquad \boldsymbol{\omega} = \begin{pmatrix} \omega_x \\ \omega_y \\ \omega_z \end{pmatrix} \in \mathbb{R}^3$$

Linear Velocity

Angular Velocity

Lie Algebra se(3)

- se(3) is the tangent space of SE(3) at the origin
- Its vector basis is:

$$\mathbf{G}_1 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \; ; \; \mathbf{G}_2 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \; ; \; \mathbf{G}_3 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{G}_4 = \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \; ; \; \mathbf{G}_5 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \; ; \; \mathbf{G}_6 = \begin{pmatrix} 0 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$$

$$\mathbf{d}^{\wedge} = \begin{pmatrix} 0 & -\theta_z & \theta_y & d_x \\ \theta_z & 0 & -\theta_x & d_y \\ -\theta_y & \theta_x & 0 & d_z \\ 0 & 0 & 0 & 0 \end{pmatrix} = d_x\mathbf{G}_1 + d_y\mathbf{G}_2 + d_z\mathbf{G}_3 + \theta_x\mathbf{G}_4 + \theta_y\mathbf{G}_5 + \theta_z\mathbf{G}_6$$

$$\mathbf{d} = (d_x \, d_y \, d_z \, \theta_x \, \theta_y \, \theta_z)^T$$

  – se(3) represents linear and angular velocities or (small) motions

Exponential Map

- Solution of differential equation:

$$\mathbf{T}(t) = \mathbf{T}_0 \exp\left((\mathbf{v}\, t)^\wedge\right) = \mathbf{T}_0 \exp\left(\mathbf{d}^\wedge\right) = \mathbf{T}_0 \, \mathrm{Exp}(\mathbf{d})$$

  - Note that successive motions are expressed in the transformed frames, not in W:

$$\mathbf{T}_{W2} = \mathbf{T}_{W0}\, \mathbf{T}_{01}\, \mathbf{T}_{12} = \mathbf{T}_{W0}\, \mathrm{Exp}(\mathbf{d}_1)\, \mathrm{Exp}(\mathbf{d}_2)$$

Exponential and Logarithmic Maps in SE(3)

- Vector space: $$\mathbf{d} = \begin{pmatrix} \boldsymbol{d} \\ \boldsymbol{\theta} \end{pmatrix} = (d_x \, d_y \, d_z \, \theta_x \, \theta_y \, \theta_z)^T$$

Watch out!

- Exponential map:

$$\left. \frac{\partial \operatorname{Exp}(\mathbf{d})}{\partial d_i} \right|_{\mathbf{d}=\mathbf{0}} = \mathbf{G}_i$$

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} = \operatorname{Exp}(\mathbf{d}) = \begin{pmatrix} \operatorname{Exp}(\boldsymbol{\theta}) & \mathbf{V}(\boldsymbol{\theta})\, \boldsymbol{d} \\ \mathbf{0} & 1 \end{pmatrix}$$

$$\mathbf{R} = \operatorname{Exp}(\boldsymbol{\theta}) = \operatorname{Exp}(\mathbf{u}\theta) = \mathbf{I} + \sin\theta \, [\mathbf{u}]_\times + (1 - \cos\theta)\, [\mathbf{u}]_\times^2$$

$$\mathbf{V}(\boldsymbol{\theta}) = \mathbf{I} + \frac{1 - \cos\theta}{\theta} [\mathbf{u}]_\times + \frac{\theta - \sin\theta}{\theta} [\mathbf{u}]_\times^2$$

- Logarithmic map:

$$\theta = \cos^{-1}\left(\frac{\operatorname{trace}(\mathbf{R}) - 1}{2} \right)$$

$$\boldsymbol{\theta} = \operatorname{Log}(\mathbf{R}) = \frac{\theta}{2\sin\theta} \left(\mathbf{R} - \mathbf{R}^T\right)^\vee$$

$$\boldsymbol{d} = \left(\mathbf{V}(\boldsymbol{\theta})\right)^{-1} \mathbf{t}$$

A Simplified Notation

$$\mathbf{T} = \begin{pmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \qquad \mathbf{d} \in \mathbb{R}^6 \cong se(3) \qquad \mathbf{x} \in \mathbb{R}^3$$

- Overloading the composition $\oplus$ with implicit Exp / Log:

$$\begin{aligned}
\mathbf{T}_{AB} \oplus \mathbf{T}_{BC} &\triangleq \mathbf{T}_{AB}\,\mathbf{T}_{BC} = \mathbf{T}_{AC} \\
\mathbf{T}_{AB} \oplus \mathbf{d}_{B} &\triangleq \mathbf{T}_{AB} \oplus \mathrm{Exp}(\mathbf{d}_B) \\
\mathbf{d}_{A} \oplus \mathbf{T}_{AB} &\triangleq \mathrm{Exp}(\mathbf{d}_A) \oplus \mathbf{T}_{AB} \\
\\
\mathbf{d} = \mathbf{T}_1 \oplus \mathbf{T}_2 &\triangleq \mathrm{Log}\left(\mathbf{T}_1 \mathbf{T}_2\right) \\
\\
\mathbf{T}_{AB} \oplus \mathbf{x}_{BP} &\triangleq \mathbf{R}_{AB}\,\mathbf{x}_{BP} + \mathbf{t}_{AB} = \mathbf{x}_{AP} \\
\mathbf{d}_{B} \oplus \mathbf{x}_{BP} &\triangleq \mathrm{Exp}(\mathbf{d}_B) \oplus \mathbf{x}_{BP}
\end{aligned}$$

Other ways to Represent Perturbations

$$\mathbf{M} = \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{bmatrix} \in SE(3) \qquad \boldsymbol{\tau} = \begin{bmatrix} \boldsymbol{\rho} \\ \boldsymbol{\theta} \end{bmatrix} \in \mathbb{R}^6$$

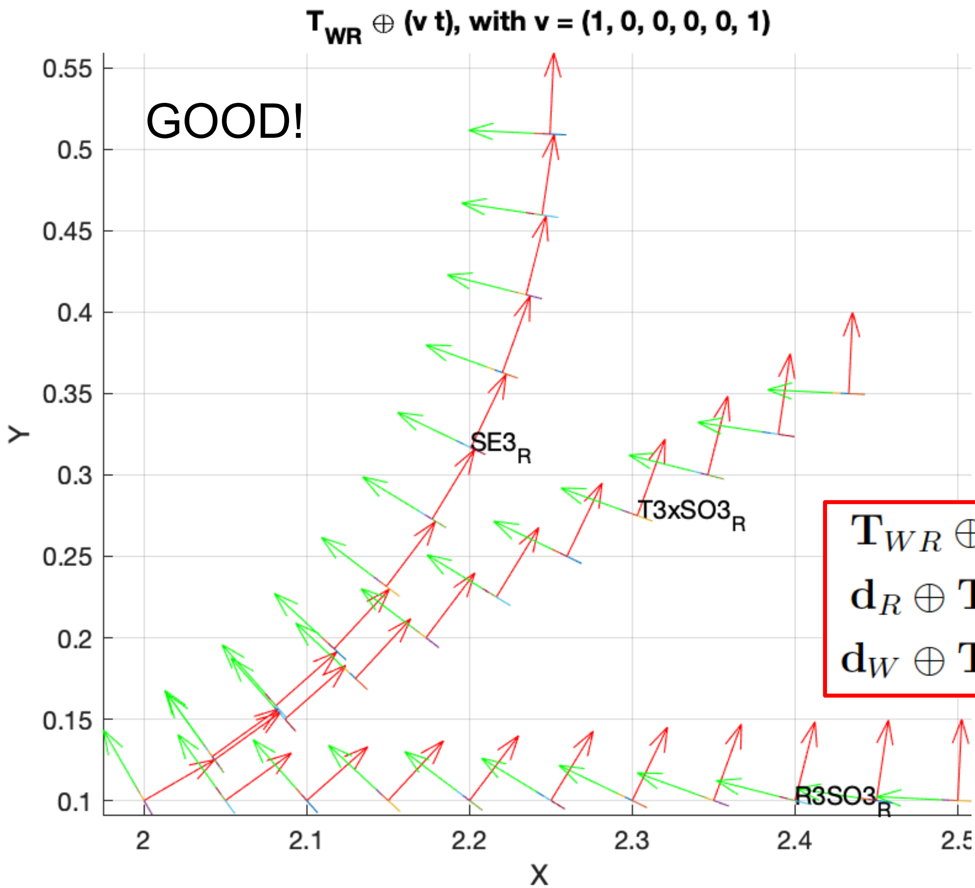
- Same rotational part, different translations:

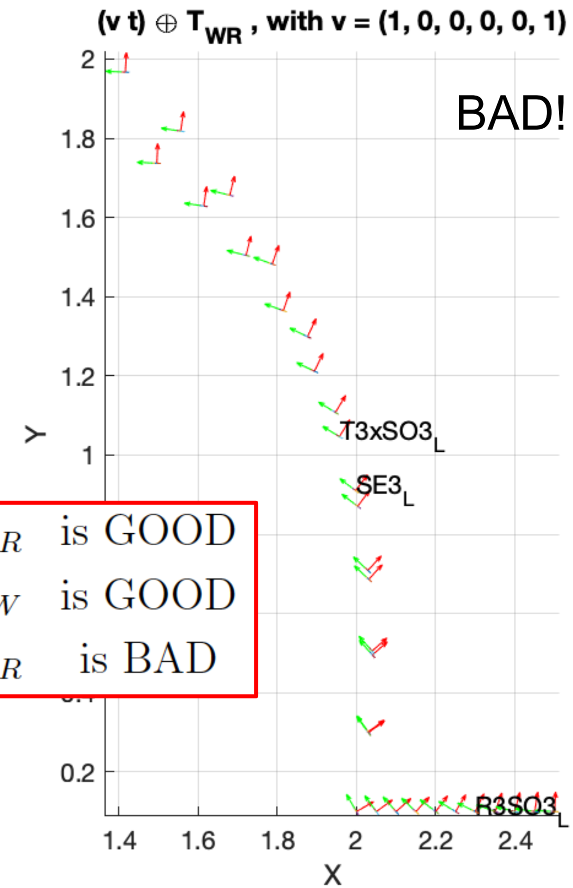
$$SE(n): \quad \mathbf{M} \oplus \boldsymbol{\tau} = \begin{bmatrix} \mathbf{R}\,\mathrm{Exp}(\boldsymbol{\theta}) & \mathbf{t} + \mathbf{R}\mathbf{V}(\boldsymbol{\theta})\boldsymbol{\rho} \\ \mathbf{0} & 1 \end{bmatrix}$$

$$T(n) \times SO(n): \quad \mathbf{M} \oplus \boldsymbol{\tau} = \begin{bmatrix} \mathbf{R}\,\mathrm{Exp}(\boldsymbol{\theta}) & \mathbf{t} + \mathbf{R}\mathbf{p} \\ \mathbf{0} & 1 \end{bmatrix}$$

$$\langle \mathbb{R}^n, SO(n) \rangle: \quad \mathbf{M} \diamondplus \boldsymbol{\tau} = \begin{bmatrix} \mathbf{t} + \mathbf{p} \\ \mathbf{R}\,\mathrm{Exp}(\boldsymbol{\theta}) \end{bmatrix}$$

Perturbations on the Right or on the Left? Wrong question!





$$\mathbf{T}_{WR} \oplus \mathbf{d}_R \quad \text{is GOOD}$$
$$\mathbf{d}_R \oplus \mathbf{T}_{RW} \quad \text{is GOOD}$$
$$\mathbf{d}_W \oplus \mathbf{T}_{WR} \quad \text{is BAD}$$

Define perturbations in a **LOCAL REFERENCE**: robot, camera, IMU,…

NEVER AT THE ORIGIN.
Rotations around the origin produce big translations when far from the origin!!!

5.2.d Jacobians
Jacobian of a Transformation (Adjoint Matrix)

- Allows to change velocities or perturbations from one side of a transformation to the other side:

$$\mathbf{J}_{AB} \triangleq \mathbf{Ad}_{\mathbf{T}_{AB}} = \begin{pmatrix} \mathbf{R}_{AB} & [\mathbf{t}_{AB}]_\times \mathbf{R}_{AB} \\ 0 & \mathbf{R}_{AB} \end{pmatrix}$$

$$\mathbf{T}_{AB} \oplus \mathbf{d}_B = \mathbf{d}_A \oplus \mathbf{T}_{AB} = \mathbf{J}_{AB}\mathbf{d}_B \oplus \mathbf{T}_{AB}$$

$$\begin{aligned} \mathbf{d}_A &= \mathbf{J}_{AB}\,\mathbf{d}_B \\ \mathbf{d}_B &= \mathbf{J}_{BA}\,\mathbf{d}_A \end{aligned}$$

Jacobians: Composition

- Left:
- Right:

$$\mathbf{J}_{1\oplus}(\mathbf{T}_{AB}) \equiv \mathbf{J}_l(\mathbf{T}_{AB}) \triangleq \left.\frac{\partial(\mathbf{d}_A \oplus \mathbf{T}_{AB})}{\partial \mathbf{d}_A}\right|_{\mathbf{d}_A=\mathbf{0}}$$

$$\mathbf{J}_{2\oplus}(\mathbf{T}_{AB}) \equiv \mathbf{J}_r(\mathbf{T}_{AB}) \triangleq \left.\frac{\partial(\mathbf{T}_{AB} \oplus \mathbf{d}_B)}{\partial \mathbf{d}_B}\right|_{\mathbf{d}_B=\mathbf{0}}$$

- Relationship:

$$\mathbf{J}_{2\oplus}(\mathbf{T}_{AB}) = \left.\frac{\partial(\mathbf{J}_{AB}\mathbf{d}_B \oplus \mathbf{T}_{AB})}{\partial \mathbf{d}_B}\right|_{\mathbf{d}_B=\mathbf{0}} = \mathbf{J}_{1\oplus}(\mathbf{T}_{AB})\mathbf{J}_{AB}$$

- Computation:

$$\mathbf{J}_{1\oplus}(\mathbf{T}) = \begin{pmatrix} \mathbf{V}(\boldsymbol{\theta}) & \mathbf{Q}(\boldsymbol{\rho},\boldsymbol{\theta}) \\ \mathbf{0} & \mathbf{V}(\boldsymbol{\theta}) \end{pmatrix}$$

$$\boxed{\mathrm{Log}(\mathbf{T}) = \begin{pmatrix} \boldsymbol{\rho} \\ \boldsymbol{\theta} \end{pmatrix}}$$

$$\begin{aligned}
\mathbf{Q}(\boldsymbol{\rho},\boldsymbol{\theta}) = & \frac{1}{2}\boldsymbol{\rho}_\times + \frac{\theta - \sin\theta}{\theta^3}(\boldsymbol{\theta}_\times\boldsymbol{\rho}_\times + \boldsymbol{\rho}_\times\boldsymbol{\theta}_\times + \boldsymbol{\theta}_\times\boldsymbol{\rho}_\times\boldsymbol{\theta}_\times) \\
& - \frac{1 - \frac{\theta^2}{2} - \cos\theta}{\theta^4}(\boldsymbol{\theta}_\times^2\boldsymbol{\rho}_\times + \boldsymbol{\rho}_\times\boldsymbol{\theta}_\times^2 - 3\boldsymbol{\theta}_\times\boldsymbol{\rho}_\times\boldsymbol{\theta}_\times) \\
& - \frac{1}{2}\left(\frac{1 - \frac{\theta^2}{2} - \cos\theta}{\theta^4} - 3\frac{\theta - \sin\theta - \frac{\theta^3}{6}}{\theta^5}\right) \\
& \times (\boldsymbol{\theta}_\times\boldsymbol{\rho}_\times\boldsymbol{\theta}_\times^2 + \boldsymbol{\theta}_\times^2\boldsymbol{\rho}_\times\boldsymbol{\theta}_\times) .
\end{aligned}$$

(Barfoot & Furgale, TRO 2014)
(Sola et al., arXiv 2020)

Jacobians: Point Composition

- Point composition:

$$\mathbf{d} \oplus \mathbf{T} \oplus \mathbf{x} = \mathrm{Exp}(\mathbf{d}) \oplus \mathbf{y} = \mathbf{S}_{1:3}\,\mathrm{Exp}(\mathbf{d}) \begin{pmatrix} \mathbf{y} \\ 1 \end{pmatrix}$$

$$\boxed{\begin{aligned} \mathbf{y} &\triangleq \mathbf{T} \oplus \mathbf{x} \\ \mathbf{S}_{1:3} &\triangleq \begin{pmatrix} \mathbf{I}_3 & \mathbf{0}_{3\times 1} \end{pmatrix} \end{aligned}}$$

- Lie groups make differentiation easy:

$$\left.\frac{\partial(\mathbf{d} \oplus \mathbf{T} \oplus \mathbf{x})}{\partial d_i}\right|_{\hat{\mathbf{x}},\mathbf{d}=\mathbf{0}} = \mathbf{S}_{1:3} \left.\frac{\partial\,\mathrm{Exp}(\mathbf{d})}{\partial d_i}\right|_{\mathbf{d}=\mathbf{0}} \begin{pmatrix} \hat{\mathbf{y}} \\ 1 \end{pmatrix} = \mathbf{S}_{1:3}\,\mathbf{G}_i \begin{pmatrix} \hat{\mathbf{y}} \\ 1 \end{pmatrix}$$

- Jacobians:

$$\begin{aligned} \mathbf{J}_d(\hat{\mathbf{y}}) &\triangleq \left.\frac{\partial(\mathbf{d} \oplus \mathbf{T} \oplus \mathbf{x})}{\partial \mathbf{d}}\right|_{\hat{\mathbf{x}},\mathbf{d}=\mathbf{0}} = \begin{pmatrix} \mathbf{I}_3 & [-\hat{\mathbf{y}}]_\times \end{pmatrix} \\ \mathbf{J}_x(\hat{\mathbf{T}}) &\triangleq \left.\frac{\partial(\mathbf{d} \oplus \mathbf{T} \oplus \mathbf{x})}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}},\mathbf{d}=\mathbf{0}} = \left.\frac{\partial(\mathbf{R}\mathbf{x} + \mathbf{t})}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}},\mathbf{d}=\mathbf{0}} = \hat{\mathbf{R}} \end{aligned}$$

Jacobians of Point Observation

- A point observed by a camera:

$$\mathbf{u}_{ij} = \boldsymbol{\pi}\left(\mathbf{x}_{C_iP_j}\right) = \boldsymbol{\pi}\left(\mathbf{T}_{C_iW} \oplus \mathbf{x}_{WP_j}\right) = \boldsymbol{\pi}\left(\mathbf{d}_{C_i} \oplus \mathbf{T}_{C_iW} \oplus \mathbf{x}_{WP_j}\right)$$

- Jacobian of projection:

$$\mathbf{J}_\pi(\hat{\mathbf{x}}) \triangleq \left.\frac{\partial \pi\left(\mathbf{x}\right)}{\partial \mathbf{x}}\right|_{\hat{\mathbf{x}}}$$

- Jacobians:

$$\mathbf{J}_{Cij} \triangleq \left.\frac{\partial \mathbf{u}_{ij}}{\partial \mathbf{d}_{C_i}}\right|_{\hat{\mathbf{x}}_{C_iP_j}} = \mathbf{J}_\pi(\hat{\mathbf{x}}_{C_iP_j})\, \mathbf{J}_d(\hat{\mathbf{x}}_{C_iP_j}) = \mathbf{J}_\pi(\hat{\mathbf{x}}_{C_iP_j}) \begin{pmatrix} \mathbf{I}_3 & \left[-\hat{\mathbf{x}}_{C_iP_j}\right]_\times \end{pmatrix}$$

$$\mathbf{J}_{Pij} \triangleq \left.\frac{\partial \mathbf{u}_{ij}}{\partial \mathbf{x}_{WP_j}}\right|_{\hat{\mathbf{x}}_{C_iP_j}} = \mathbf{J}_\pi(\hat{\mathbf{x}}_{C_iP_j})\, \mathbf{J}_x(\hat{\mathbf{T}}_{C_iW}) = \mathbf{J}_\pi(\hat{\mathbf{x}}_{C_iP_j})\, \hat{\mathbf{R}}_{C_iW}$$

- If the state is the robot pose instead of the camera pose:

$$\mathbf{x}_{C_iP_j} = \mathbf{T}_{CR} \oplus \mathbf{d}_{R_i} \oplus \mathbf{T}_{R_iW} \oplus \mathbf{x}_{WP_j} = \mathbf{J}_{CR}\mathbf{d}_{R_i} \oplus \mathbf{T}_{CR} \oplus \mathbf{T}_{R_iW} \oplus \mathbf{x}_{WP_j}$$

$$\mathbf{J}_{Rij} \triangleq \left.\frac{\partial \mathbf{u}_{ij}}{\partial \mathbf{d}_{R_i}}\right|_{\hat{\mathbf{x}}_{C_iP_j}} = \mathbf{J}_\pi(\hat{\mathbf{x}}_{C_iP_j}) \begin{pmatrix} \mathbf{I}_3 & \left[-\hat{\mathbf{x}}_{C_iP_j}\right]_\times \end{pmatrix} \mathbf{J}_{CR}$$

5.2.e Similarity Transformations Sim(3)

- Useful in monocular SLAM (to optimize scale drift)
- Lie group Sim(3): $\mathbf{T} = \begin{pmatrix} s\mathbf{R} & \mathbf{t} \\ \mathbf{0} & 1 \end{pmatrix} \in Sim(3) \qquad s \in \mathbb{R}^{+}$
- Lie algebra sim(3): $\mathbf{v} = \begin{pmatrix} \boldsymbol{\upsilon} \\ \boldsymbol{\omega} \\ \sigma \end{pmatrix} \in \mathbb{R}^{7} \qquad \mathbf{G}_7 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix}$
- Exp:

$$\exp_{\mathbf{Sim}(3)}(\boldsymbol{\upsilon}, \boldsymbol{\omega}, \sigma) := \exp\begin{pmatrix} [\boldsymbol{\omega}]_\times + \sigma \mathtt{I}_{3\times3} & \boldsymbol{\upsilon} \\ \mathtt{0}_{1\times3} & 0 \end{pmatrix} = \begin{pmatrix} e^\sigma \exp([\boldsymbol{\omega}]_\times) & \mathtt{W}\boldsymbol{\upsilon} \\ \mathtt{0}_{1\times3} & 1 \end{pmatrix}$$

$$\mathtt{W} = \left(\frac{e^\sigma - 1}{\sigma}\right)\mathtt{I} + \frac{A\sigma + (1-B)\theta}{\theta(\sigma^2+\theta^2)}[\boldsymbol{\omega}]_\times + \left(\frac{e^\sigma - 1}{\sigma} - \frac{(B-1)\sigma + A\theta}{\sigma^2+\theta^2}\right)\frac{[\boldsymbol{\omega}]_\times^2}{\theta^2}$$

$$A = e^\sigma \sin(\theta),\ B = e^\sigma \cos(\theta) \text{ and } \theta = ||\boldsymbol{\omega}||_2$$

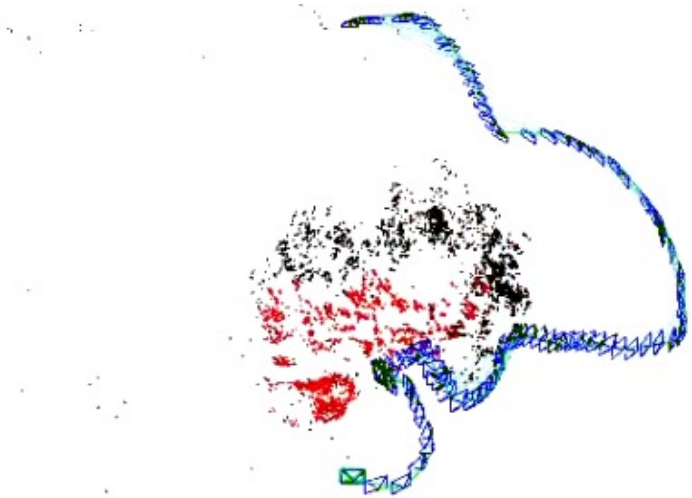
- Log:

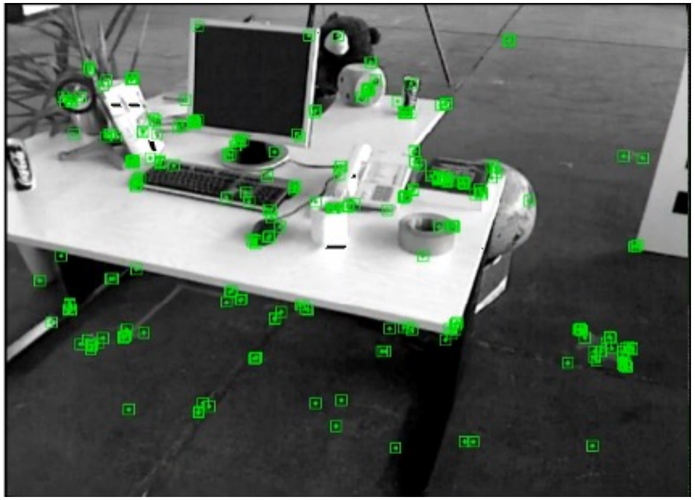
$$\log\begin{bmatrix} s\mathtt{R} & \mathbf{t} \\ \mathtt{0}_{1\times3} & 1 \end{bmatrix} = \begin{bmatrix} \log(\mathtt{R}) + \ln(s)\mathtt{I} & \mathtt{W}^{-1}\mathbf{t} \\ \mathtt{0}_{1\times3} & 0 \end{bmatrix}$$

(Strasdat PhD 2012, Chapter 5)

5.3 Optimization in Visual SLAM

Juan D. Tardós
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB





$$\{\mathbf{T}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{K}_1,\ j \in \mathcal{P}_1\}^* = \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{k \in \{\mathcal{K}_1, \mathcal{K}_2\},\, j \in \mathcal{P}_1} \rho\left(\left\| \mathbf{u}_{kj} - \pi_k\left(\mathbf{T}_{kw}\mathbf{x}_{wj}\right) \right\|^2_{\Sigma_{kj}} \right)$$

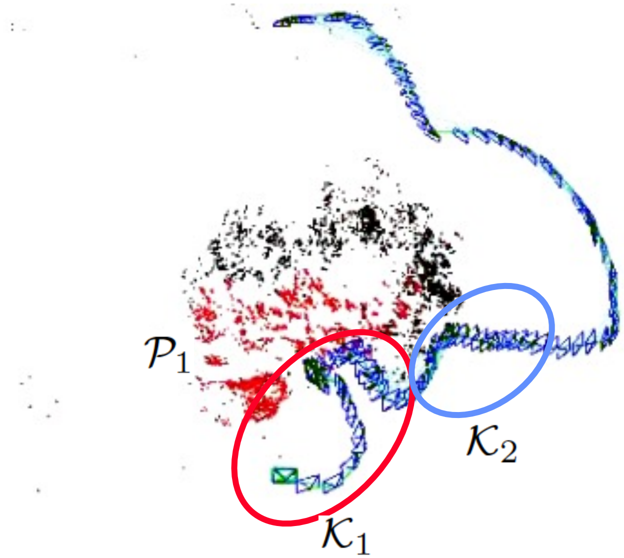
Lesson 5. Optimization in SLAM

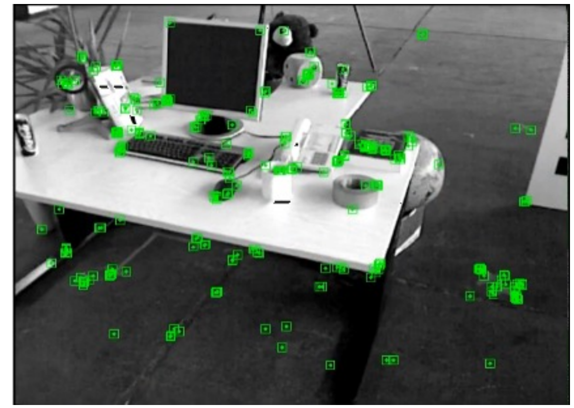
3. Optimizations in Visual SLAM
    a. Bundle Adjustment
    b. Pose-graph

Readings:

- B. Triggs, P. McLauchlan, R. Hartley, A. Fitzgibbon Bundle Adjustment — A Modern Synthesis, in Vision Algorithms: Theory andPractice, Springer, 2000
- R Kümmerle, G Grisetti, H Strasdat, K Konolige, W Burgard, g^{2}o: A general framework for graph optimization, ICRA 2011
- Hauke Strasdat Local Accuracy and Global Consistency for Efficient Visual SLAM, PhD Thesis, Imperial College, 2012 (chapters 5 and B.6)

5.3.a Local Bundle Adjustment (BA)





Local Map:
- $\mathcal{K}_1$ Current Keyframe & Covisibles
- $\mathcal{P}_1$ Points seen by $\mathcal{K}_1$ (in red)
- $\mathcal{K}_2$ Rest of Keyframes that see $\mathcal{P}_1$

Local Bundle Adjustment

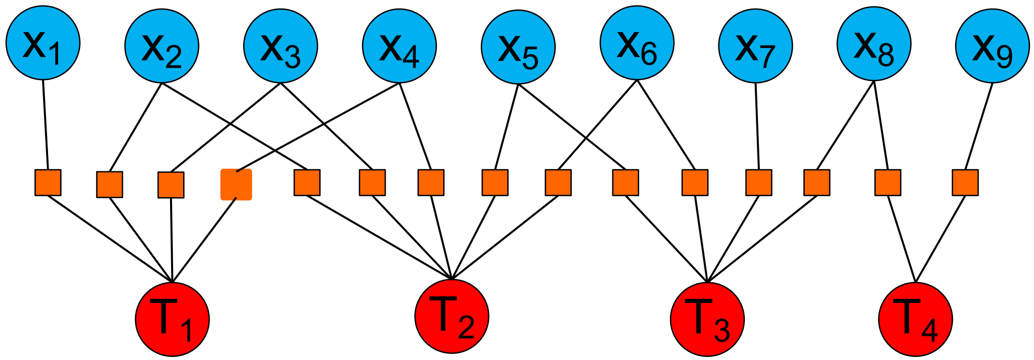
$$\{\mathbf{T}_{iw}, \mathbf{x}_{wj} \mid i \in \mathcal{K}_1,\ j \in \mathcal{P}_1\}^* = \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{k \in \{\mathcal{K}_1, \mathcal{K}_2\},\, j \in \mathcal{P}_1} \rho\left(\left\|\mathbf{u}_{kj} - \pi_k\left(\mathbf{T}_{kw}\mathbf{x}_{wj}\right)\right\|^2_{\Sigma_{kj}}\right)$$

Variable

Constant, they anchor the map

Representation with a Factor Graph

- Bipartite graph: variables (cameras and points) and constraints between them (observations or priors):



$$\mathbf{e}_{ij} = \mathbf{u}_{ij} - \boldsymbol{\pi}_i(\mathbf{T}_{iw} \oplus \mathbf{x}_{wj})$$

- Represents the map posterior pdf as a product of factors:

$$\begin{aligned} p\left(\{\mathbf{T}_{iw}, \mathbf{x}_{wj}\} \,|\, \{\mathbf{u}_{ij}\}\right) \;&\propto\; \prod_{ij} p(\mathbf{u}_{ij} \,|\, \mathbf{T}_{iw}, \mathbf{x}_{wj}) * \cancel{\prod_{i} p(\mathbf{T}_{iw})} * \cancel{\prod_{j} p(\mathbf{x}_{wj})} \\ &\propto\; \prod_{ij} \exp\left(-\mathbf{e}_{ij}^{\top} \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij}\right) \end{aligned}$$

Assuming Gaussian noise

- The Maximum a Posteriori (MAP) solution is:

$$\begin{aligned} \{\mathbf{T}_{iw}, \mathbf{x}_{wj}\}^{*} \;&=\; \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmax}}\, p\left(\{\mathbf{T}_{iw}, \mathbf{x}_{wj}\} \,|\, \{\mathbf{u}_{ij}\}\right) \\ &=\; \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \mathbf{e}_{ij}^{\top} \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij} \end{aligned}$$

Putting Everything Together

se(3)

- Observations: $\mathbf{u}_{ij} = \boldsymbol{\pi}_i(\mathbf{T}_{iw} \oplus \mathbf{x}_{wj}) + \mathbf{w}_{ij} = \boldsymbol{\pi}_i(\mathbf{d}_i \oplus \mathbf{T}_{iw} \oplus \mathbf{x}_{wj}) + \mathbf{w}_{ij}$

$$\mathbf{w}_{ij} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{ij}) \ ; \ \boldsymbol{\Sigma}_{ij} = \sigma_{ij}^2 \mathbf{I}_2 \ ; \ \sigma_{ij} \approx 1pixel * scale_{ij}$$

- Reproject. error: $\mathbf{e}_{ij} = \mathbf{u}_{ij} - \boldsymbol{\pi}_i(\mathbf{T}_{iw} \oplus \mathbf{x}_{wj})$
- Robust cost: $\{\mathbf{T}_{iw}, \mathbf{x}_{wj}\}^* = \underset{\mathbf{T}_{iw}, \mathbf{x}_{wj}}{\operatorname{argmin}} \sum_{i,j} \rho\left(\mathbf{e}_{ij}^\top \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij}\right)$

$$\mathbf{W}_{ij} = \rho'(\mathbf{e}_{ij}^\top \boldsymbol{\Sigma}_{ij}^{-1} \mathbf{e}_{ij})\, \boldsymbol{\Sigma}_{ij}^{-1}$$

- Estate: $\mathbf{d} \triangleq (\mathbf{d}_1 \ldots \mathbf{d}_i \ldots \mathbf{d}_n\, \delta\mathbf{x}_{w1} \ldots \delta\mathbf{x}_{wj} \ldots \delta\mathbf{x}_{wm})^\top$
- Jacobians:

$$\begin{aligned}
\mathbf{J}_{ij} &= (0 \ldots 0\, \mathbf{J}_{Cij}\, 0 \ldots\ldots\ldots\ldots 0\, \mathbf{J}_{Pij}\, 0 \ldots\ldots 0) \\
\mathbf{J}_{Cij} &\triangleq \left.\frac{\partial \mathbf{u}_{ij}}{\partial \mathbf{d}_i}\right|_{\hat{\mathbf{x}}_{ij}} = \mathbf{J}_\pi(\hat{\mathbf{x}}_{ij}) \left(\mathbf{I}_3 \ \ [-\hat{\mathbf{x}}_{ij}]_\times\right) \\
\mathbf{J}_{Pij} &\triangleq \left.\frac{\partial \mathbf{u}_{ij}}{\partial \mathbf{x}_{wj}}\right|_{\hat{\mathbf{x}}_{ij}} = \mathbf{J}_\pi(\hat{\mathbf{x}}_{ij})\, \hat{\mathbf{R}}_{iw}
\end{aligned}$$

- Levengerg-Marquardt:

$$\begin{aligned}
(\mathbf{J}^\top \mathbf{W} \mathbf{J} + \lambda \mathbf{I})\ \mathbf{d} &= \mathbf{J}^\top \mathbf{W}\, \mathbf{e} \\
\left(\sum_{i,j} \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{J}_{ij} + \lambda \mathbf{I}\right) \mathbf{d} &= \sum_{i,j} \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{e}_{ij}
\end{aligned}$$

Algorithm 1 Local BA

Require: $\mathcal{P}_1 = \{\mathbf{x}_{wj}\}, j = 1..m$: points to estimate
Require: $\mathcal{K}_1 = \{\mathbf{T}_{iw}\}, i = 1..n$: keyframes to estimate
Require: $\mathcal{K}_2 = \{\mathbf{T}_{iw}\}, i = n+1..n2$: other keyframes seeing $\mathcal{P}_1$
Require: $\{\mathbf{u}_{ij}\}$: observations of $\mathcal{P}_1$ from $\mathcal{K}_1$ and $\mathcal{K}_2$

repeat
  $\mathbf{H}, \mathbf{JWe} = \mathbf{0}$
  for all ij **do**
    $\mathbf{x}_{ij} = \mathbf{T}_{iw} \oplus \mathbf{x}_{wj}$
    $\mathbf{e}_{ij} = \mathbf{u}_{ij} - \pi_i(\mathbf{x}_{ij})$
    $d2_{ij} = \mathbf{e}_{ij}^\top \mathbf{\Sigma}_{ij}^{-1} \mathbf{e}_{ij}$
    $\mathbf{W}_{ij} = \rho'(d2_{ij}) \mathbf{\Sigma}_{ij}^{-1}$
    $\mathbf{J}_{Pij} = \mathbf{J}_\pi(\mathbf{x}_{ij})\ \mathbf{R}_{iw}$
    if $i \leq n$ **then**
      $\mathbf{J}_{Cij} = \mathbf{J}_\pi(\mathbf{x}_{ij})\ \left(\mathbf{I}_3\ [-\mathbf{x}_{ij}]_\times\right)$
      $\mathbf{J}_{ij} = (0...0\ \mathbf{J}_{Cij}\ 0...0\ \mathbf{J}_{Pij}\ 0...0)$
    else
      $\mathbf{J}_{ij} = (0......0\ \mathbf{J}_{Pij}\ 0...0)$
    end if
    $\mathbf{H} = \mathbf{H} + \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{J}_{ij}$
    $\mathbf{JWe} = \mathbf{JWe} + \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{e}_{ij}$
  end for
  $\mathbf{d} \leftarrow solve : (\mathbf{H} + \lambda \mathbf{I})\mathbf{d} = \mathbf{JWe}$
  for $i = 1..n$ **do**
    $\mathbf{T}_{iw} = \mathbf{d}(6i-5 : 6i) \oplus \mathbf{T}_{iw}$
  end for
  for $j = 1..m$ **do**
    $\mathbf{x}_{wj} = \mathbf{d}(6n+3j-2 : 6n+3j) + \mathbf{x}_{wj}$
  end for
until converged

Reprojection error
Mahalanobis distance
Robust cost funtion

Compute Jacobians

Build linear system

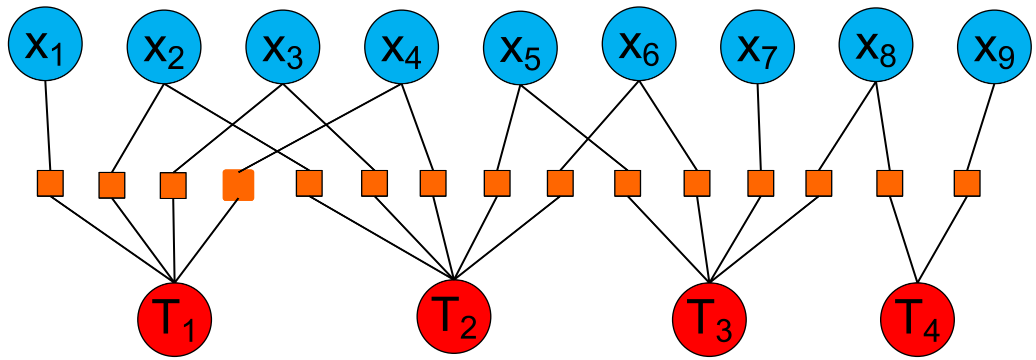
Solve system (Levengerg-Marquardt)

Update Keyframes using
Lie Group SE(3)

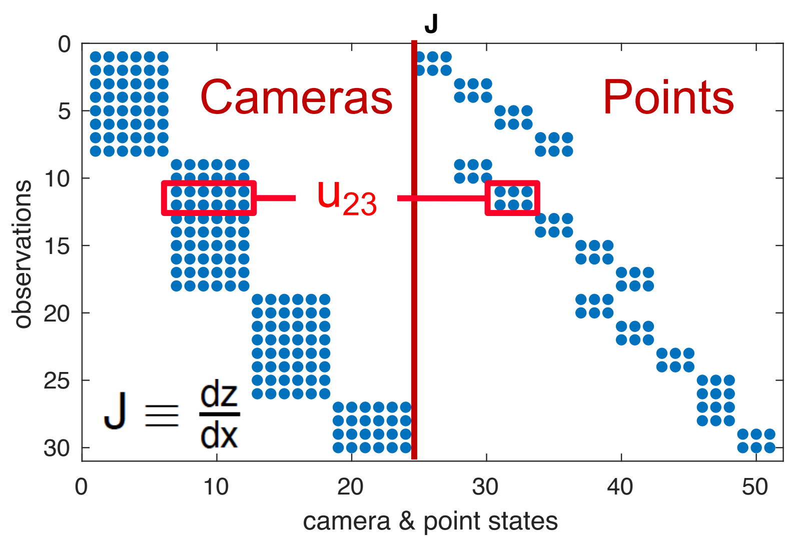
Update Points in R^3

Sparsity of Visual SLAM

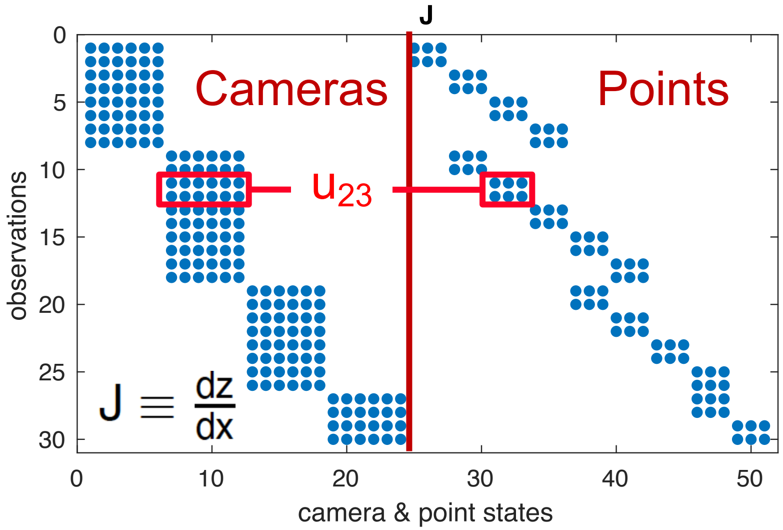
- Each camera does not see all the points

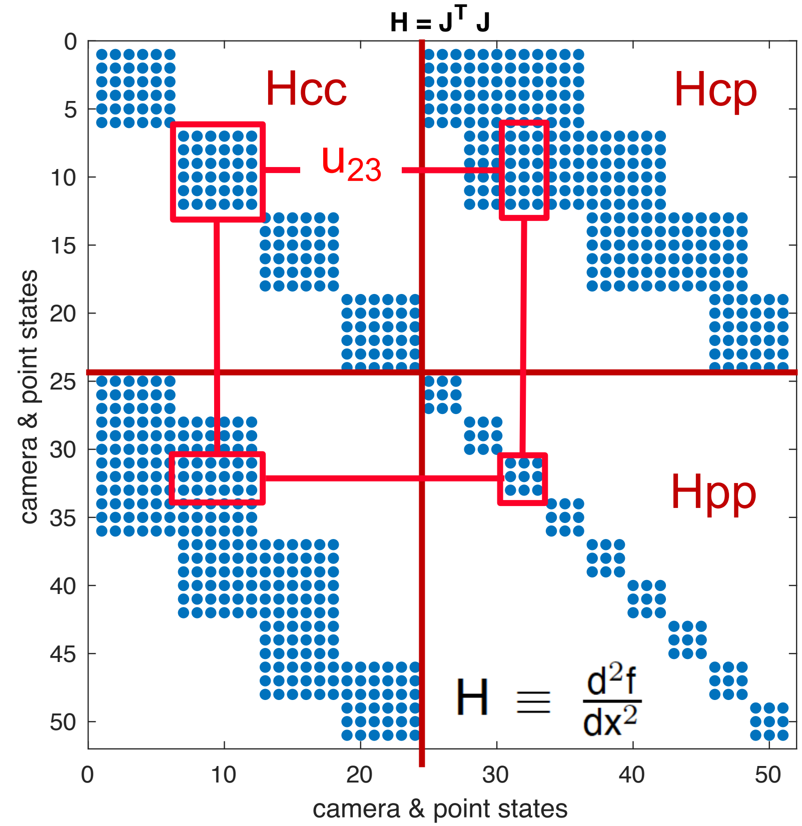


- The Jacobian is very sparse:



Sparsity of Visual SLAM





You don't need to build J, instead, build H and b:

$$\begin{aligned}\mathbf{H} &= \mathbf{J}^\top \mathbf{W} \mathbf{J} = \sum_{i,j} \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{J}_{ij} \\ \mathbf{b} &= \mathbf{J}^\top \mathbf{W}\, \mathbf{e} = \sum_{i,j} \mathbf{J}_{ij}^\top \mathbf{W}_{ij} \mathbf{e}_{ij}\end{aligned}$$

And only some blocks of H:

$$\begin{aligned}\mathbf{H}_{cc}(i) &= \mathbf{H}_{cc}(i) + \mathbf{J}_{Cij}^\top \mathbf{W}_{ij} \mathbf{J}_{Cij} \\ \mathbf{H}_{pp}(j) &= \mathbf{H}_{pp}(j) + \mathbf{J}_{Pij}^\top \mathbf{W}_{ij} \mathbf{J}_{Pij} \\ \mathbf{H}_{cp}(i,j) &= \mathbf{J}_{Cij}^\top \mathbf{W}_{ij} \mathbf{J}_{Pij}\end{aligned}$$

The Schur Complement

$$\begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{I} & -\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{I} & -\mathbf{B}\mathbf{D}^{-1} \\ \mathbf{0} & \mathbf{I} \end{pmatrix} \begin{pmatrix} \mathbf{b}_1 \\ \mathbf{b}_2 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C} & \mathbf{0} \\ \mathbf{C} & \mathbf{D} \end{pmatrix} \begin{pmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \end{pmatrix} = \begin{pmatrix} \mathbf{b}_1 - \mathbf{B}\mathbf{D}^{-1}\mathbf{b}_2 \\ \mathbf{b}_2 \end{pmatrix}$$

Schur complement of D

$$\begin{aligned} (\mathbf{A} - \mathbf{B}\mathbf{D}^{-1}\mathbf{C})\,\mathbf{x}_1 &= \mathbf{b}_1 - \mathbf{B}\mathbf{D}^{-1}\mathbf{b}_2 \\ \mathbf{D}\,\mathbf{x}_2 &= \mathbf{b}_2 - \mathbf{C}\mathbf{x}_1 \end{aligned}$$

Solve reduced system for x_1

And then, solve for x_2

Reduced Camera System

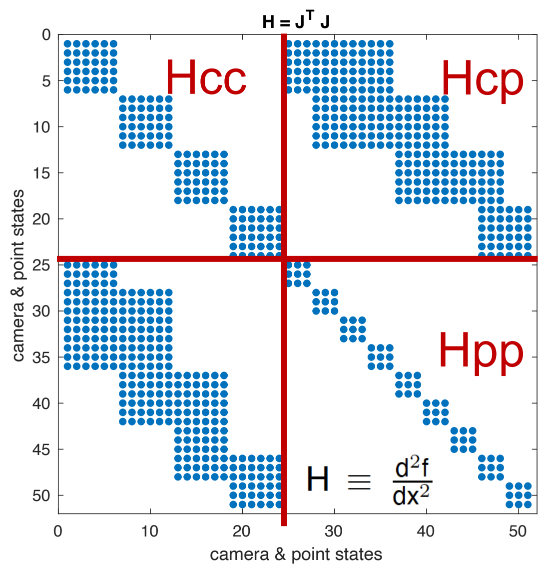
$$(\mathbf{J}^\top \mathbf{W} \mathbf{J} + \lambda \mathbf{I})\ \mathbf{d} \ = \ \mathbf{J}^\top \mathbf{W}\, \mathbf{e} \longrightarrow \begin{pmatrix} \mathbf{H}_{cc} & \mathbf{H}_{cp} \\ \mathbf{H}_{cp}^\top & \mathbf{H}_{pp} \end{pmatrix} \begin{pmatrix} \mathbf{d}_c \\ \mathbf{d}_p \end{pmatrix} = \begin{pmatrix} \mathbf{b}_c \\ \mathbf{b}_p \end{pmatrix}$$

$$(\mathbf{H}_{cc} - \mathbf{H}_{cp}\mathbf{H}_{pp}^{-1}\mathbf{H}_{cp}^\top)\ \mathbf{d}_c \ = \ \mathbf{b}_c - \mathbf{H}_{cp}\mathbf{H}_{pp}^{-1}\mathbf{b}_p$$

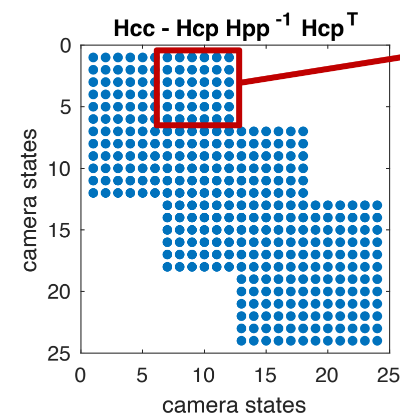
1) Solve reduced camera system

$$\mathbf{H}_{pp}\ \mathbf{d}_p \ = \ \mathbf{b}_p - \mathbf{H}_{cp}^\top \mathbf{d}_c$$

2) Substitute d_c to find d_p



Schur complement of the points



Reduced Camera Matrix (6n x 6n)

Solving the Reduced Camera System

- Cholesky decomposition of a positive definite matrix A:

$$\mathbf{A} = \mathbf{L}\mathbf{L}^\top$$

L: lower triangular matrix

- Linear system:

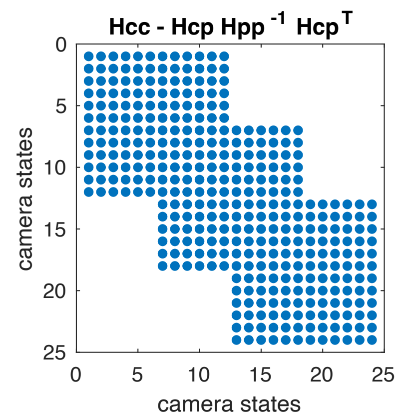
$$\mathbf{A}\,\mathbf{x} = \mathbf{b}$$
$$\mathbf{L}\mathbf{L}^\top\mathbf{x} = \mathbf{b}$$

- Solution:

$$\mathbf{L}\,\mathbf{y} = \mathbf{b}$$ Forward substitution

$$\mathbf{L}^\top\mathbf{x} = \mathbf{y}$$ Back substitution

- The camera matrix is positive definite and sparse
  - Use a sparse Cholesky library
    - » CSparse, CHOLMOD,…
  - Best case: long and thin problems → O(n)
  - Worst case: high covisibility → O(n^3)



Gauge Freedoms

SLAM is based on relative measurements of features from the on-board sensors

- The absolute pose of the map is NOT observable
- 6 gauge freedoms: 3 position + 3 orientation
- Hessian has a rank deficiency of 6

- Solution: select an arbitrary pose for the 1st camera, and exclude it from the set of variables to estimate
  - Full rank Hessian (except in degenerate cases)

In pure monocular SLAM, scale is also not observable

- 7 gauge freedoms: 3 position + 3 orientation + 1 scale
- Fix the 1st camera, and an arbitrary scale

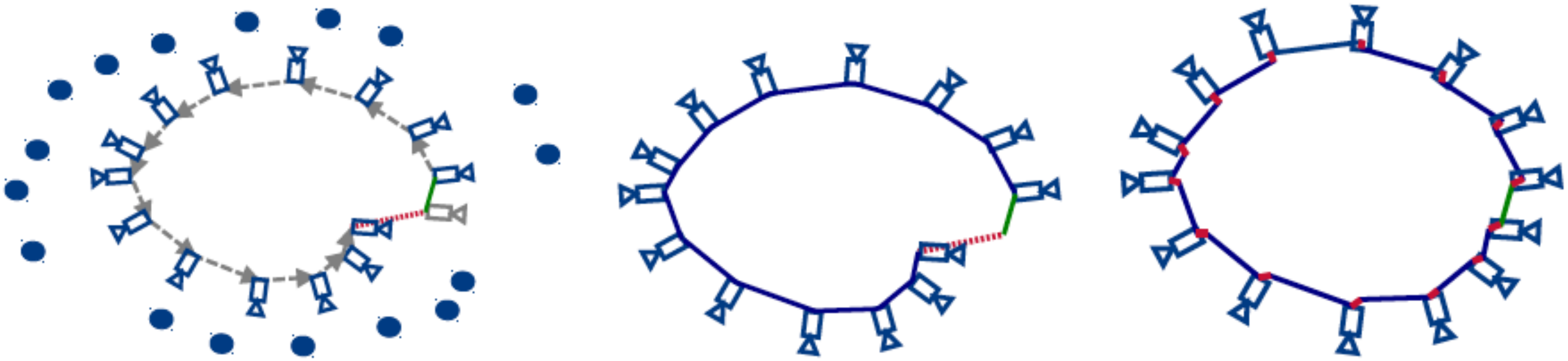
- Solution:
  - Initialize with arbitrary scale and use L-M: $\left(\mathbf{J}^\top \mathbf{W} \mathbf{J} + \lambda \mathbf{I}\right) \mathbf{d} = \mathbf{J}^\top \mathbf{W}\, \mathbf{e}$

    Avoids rank-deficiency
  - Or fix the distance between 1st and 2nd camera

5.3.b Pose-Graph Optimization

- When performing SLAM, the relative pose between neighboring keyframes is very accurate
- But drift is accumulative
- In a long loop, the accumulated drift will be large
- With large errors, BA is problematic:
  - BA is non-convex, and can converge to local minima
  - Far from the optimum, convergence can be very slow
- Solution: correct loop with pose-graph optimization
  - Optionally followed by full BA in a separate thread

Pose-Graph Optimization



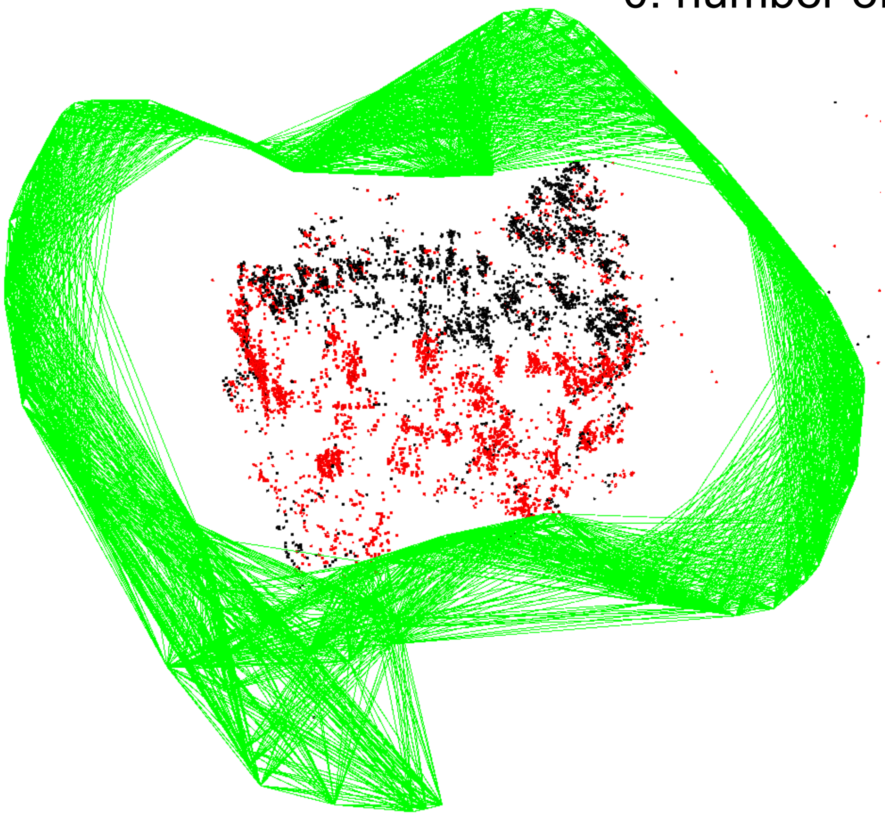
(a) Poses and points (b) Pose-graph (c) Corrected pose-graph

Figure 5.1: The loop closure problem. (a) illustrates the error (dotted red line) between the final camera pose (in grey) and the drifted pose estimate. The loop closure constraint is shown in green. (b) illustrates the pose-graph representation, where point observations are replaced by relative pose graph constraints (blue line). We correct the pose graph (c) and close the loop by distributing the error over all relative constraints.

Credit: (H. Strasdat PhD, 2012)

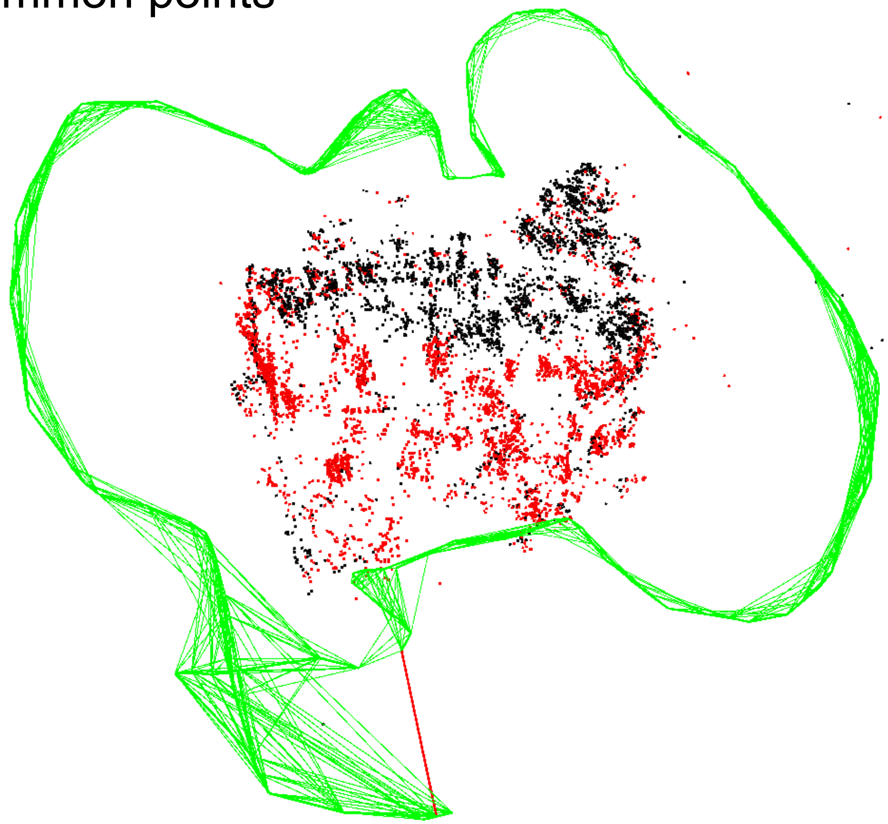
ORB-SLAM uses the Essential Graph

θ: number of common points



$\theta_{min} = 15$

Used for Local BA,
instead of sliding window



$\theta_{min} = 100$

Used for Loop Correction

Pose-Graph Optimization

- Use the relative pose between pairs of close keyframes before the loop closure as an observation (constant):

$$\bar{\mathbf{T}}_{ji} \triangleq \mathbf{T}_{jw}\mathbf{T}_{iw}^{-1}$$

- After loop correction the residual of this observation will be:

$$\begin{aligned} \mathbf{E}_{ji} &\triangleq \bar{\mathbf{T}}_{ji}\mathbf{T}_{iw}\mathbf{T}_{jw}^{-1} \in SE(3) \\ \mathbf{d}_{ji} &\triangleq \mathrm{Log}\left(\bar{\mathbf{T}}_{ji}\mathbf{T}_{iw}\mathbf{T}_{jw}^{-1}\right) \in se(3) \end{aligned}$$

- Pose-Graph optimization:

Can be obtained by point marginalization, or simply use I_6

$$\{\mathbf{T}_{2w}^*, ..\mathbf{T}_{mw}^*\} = \underset{\mathbf{T}_{2w},..\mathbf{T}_{mw}}{\mathrm{argmin}} \sum_{ji \in G} \mathbf{d}_{ji}^\top \boldsymbol{\Sigma}_{ji}^{-1} \mathbf{d}_{ji}$$

Jacobians of Pose-Graph

- We will estimate perturbations for each pose:

$$\begin{aligned}
\mathbf{d}_{ji} &= \operatorname{Log}\left(\bar{\mathbf{T}}_{ji} \oplus (\mathbf{d}_i \oplus \mathbf{T}_{iw})(\mathbf{d}_j \oplus \mathbf{T}_{jw})^{-1}\right) \\
&= \operatorname{Log}\left(\mathbf{J}_{\bar{\mathbf{T}}_{ji}} \mathbf{d}_i \oplus \bar{\mathbf{T}}_{ji} \oplus \mathbf{T}_{iw} \oplus \mathbf{T}_{jw}^{-1} \oplus -\mathbf{d}_j\right) \\
&= \operatorname{Log}\left(\mathbf{J}_{\bar{\mathbf{T}}_{ji}} \mathbf{d}_i \oplus \mathbf{E}_{ji} \oplus -\mathbf{d}_j\right)
\end{aligned}$$

- The required Jacobians are:

$$\begin{aligned}
\left.\frac{\partial \mathbf{d}_{ji}}{\partial \mathbf{d}_i}\right|_{\mathbf{d}_i, \mathbf{d}_j = 0} &= \left.\frac{\partial \operatorname{Log}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{E}_{ji}} \mathbf{J}_{1\oplus}(\mathbf{E}_{ji})\, \mathbf{J}_{\bar{\mathbf{T}}_{ji}} \\
\left.\frac{\partial \mathbf{d}_{ji}}{\partial \mathbf{d}_j}\right|_{\mathbf{d}_i, \mathbf{d}_j = 0} &= -\left.\frac{\partial \operatorname{Log}(\mathbf{x})}{\partial \mathbf{x}}\right|_{\mathbf{x}=\mathbf{E}_{ji}} \mathbf{J}_{2\oplus}(\mathbf{E}_{ji})
\end{aligned}$$

More details:
(Strasdat PhD, 2012)

- At the end, correct the cameras and the map points:

$$\mathbf{x}_{ij} = \mathbf{T}_{iw} \oplus \mathbf{x}_{wj}$$ Compute point j relative to KF i

$$\mathbf{T}_{i'w} = \mathbf{d}_i \oplus \mathbf{T}_{iw}$$ Correct KF i

$$\mathbf{x}_{wj} = \mathbf{T}_{i'w}^{-1} \oplus \mathbf{x}_{ij}$$ Correct point j

Monocular Pose-Graph

- Correcting scale drift in a monocular loop closure:

$$\mathbf{T}_{iw} = \begin{pmatrix} \mathbf{R}_{iw} & \mathbf{t}_{iw} \\ \mathbf{0} & 1 \end{pmatrix} \in SE(3) \xrightarrow{s_{iw}=1} \mathbf{S}_{iw} = \begin{pmatrix} s_{iw}\mathbf{R}_{iw} & \mathbf{t}_{iw} \\ \mathbf{0} & 1 \end{pmatrix} \in Sim(3)$$

- Pose-graph optimization in sim(3):

$$\bar{\mathbf{S}}_{ji} \triangleq \mathbf{S}_{jw}\mathbf{S}_{iw}^{-1}$$

$$\begin{aligned} \mathbf{E}_{ji} &\triangleq \bar{\mathbf{S}}_{ji}\mathbf{S}_{iw}\mathbf{S}_{jw}^{-1} \in Sim(3) \\ \mathbf{d}_{ji} &\triangleq \operatorname{Log}\left(\bar{\mathbf{S}}_{ji}\mathbf{S}_{iw}\mathbf{S}_{jw}^{-1}\right) \in sim(3) \end{aligned}$$

$$\{\mathbf{S}_{2w}^*, ..\mathbf{S}_{mw}^*\} = \underset{\mathbf{S}_{2w},..\mathbf{S}_{mw}}{\operatorname{argmin}} \sum_{ji \in G} \mathbf{d}_{ji}^\top \mathbf{\Sigma}_{ji}^{-1} \mathbf{d}_{ji}$$

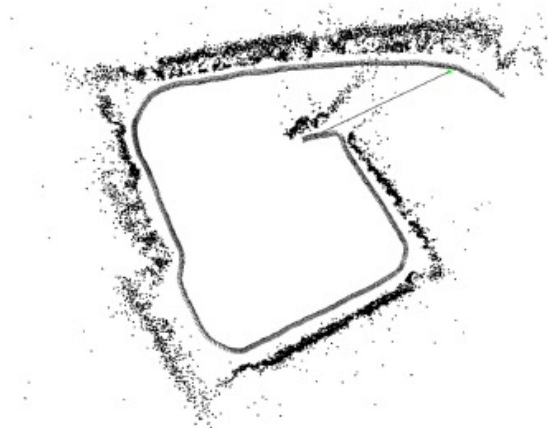
- Loop correction:

$$\mathbf{x}_{ij} = \mathbf{T}_{iw} \oplus \mathbf{x}_{wj}$$

$$\left.\begin{aligned} \mathbf{S}_{i'w} &= \mathbf{d}_i \oplus \mathbf{S}_{iw} \\ \mathbf{x}_{wj} &= \mathbf{S}_{i'w}^{-1} \oplus \mathbf{x}_{ij} \end{aligned}\right\} \text{Corrects scale drift for poses and map points}$$

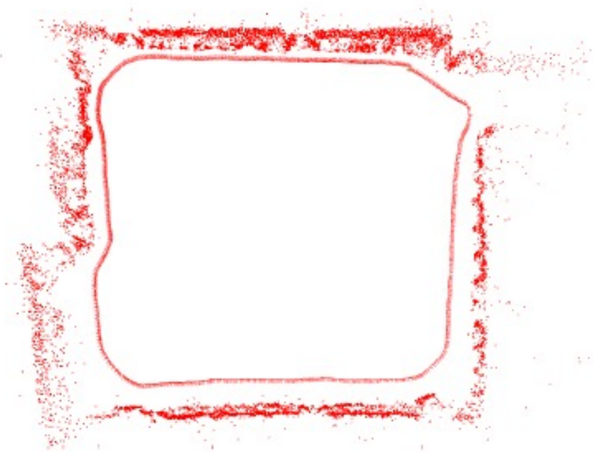
(Strasdat-PhD 2012, Chapter 5)

Example of Sim(3) vs. SE(3) Optimization



(a) before optimisation

(b) SE(3) optimisation



(c) Sim(3) optimisation

Imagery ©2012 GeoEye, Getmapping plc, Infoterra Ltd & Bluesky, The GeoInformation Group, Map data ©2012 Google

(d) aerial photo

Figure 5.3: Keble college data set

(Strasdat-PhD 2012)

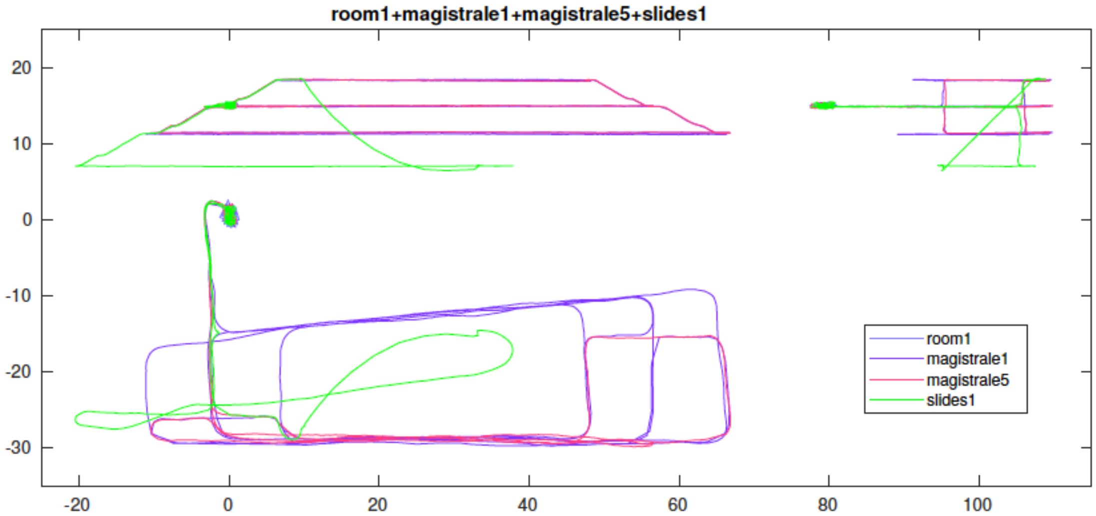
Take-Home Messages

- SLAM is a *non-linear* least-squares problem
- With *good seeds*, G-N or L-M converge in a few iterations
- *Robust cost* functions take care of spurious matchings
- Optimization in a manifold
  - It is essential to use local perturbations
  - *Lie groups* provide a nice theoretical framework
- SLAM is a *sparse* problem
  - Use Schur complement and sparse Cholesky decomposition
- Loops can be corrected with *pose-graph*
  - With 6 dof in stereo or visual-inertial settings
  - With 7 dof in pure monocular setting, to correct scale drift
- Libraries for SLAM optimization: g2o, Ceres, GTSAM

69156 - Simultaneous Localization and Mapping (SLAM)

Lesson 6: Visual-Inertial and Multi-Map SLAM

Juan D. Tardós, Carlos Campos, Richard Elvira
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



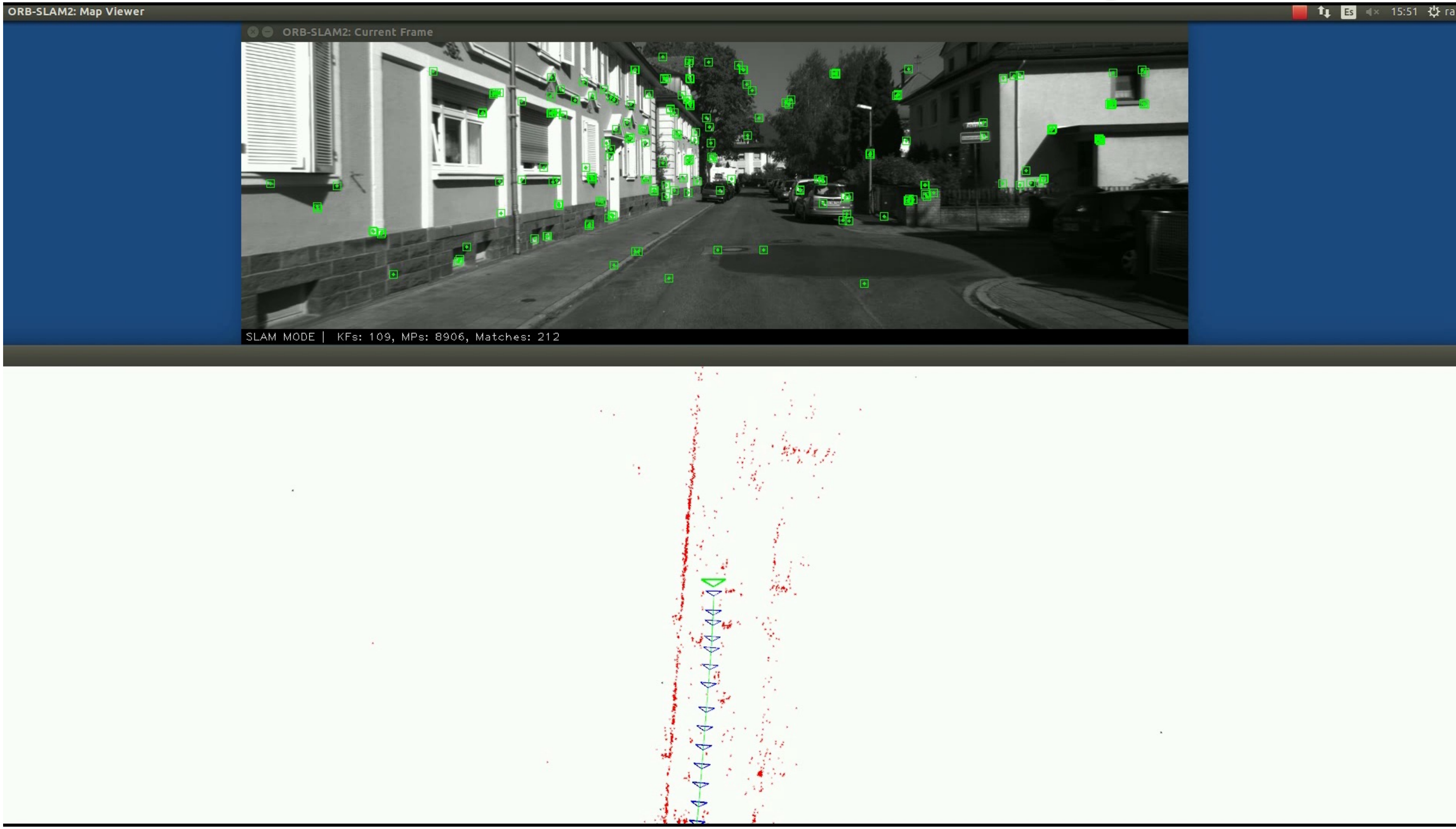
Lesson 6. Visual-Inertial and Multi-Map

1. Visual-Inertial SLAM
   a) IMU Sensor & Preintegration
   b) IMU initialization
   c) Visual-Inertial Tracking
   d) Visual-Inertial Mapping
   e) Loop-Closing and Map Merging with IMU
   f) Visual-Inertial SLAM result

Readings:

- C Campos, R Elvira, JJ Gómez Rodríguez, JMM Montiel, JD Tardós: ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM. IEEE Trans. Robotics, 2021.
- C Campos, JMM Montiel, JD Tardós, Inertial-Only Optimization for Visual-Inertial Initialization, ICRA 2020

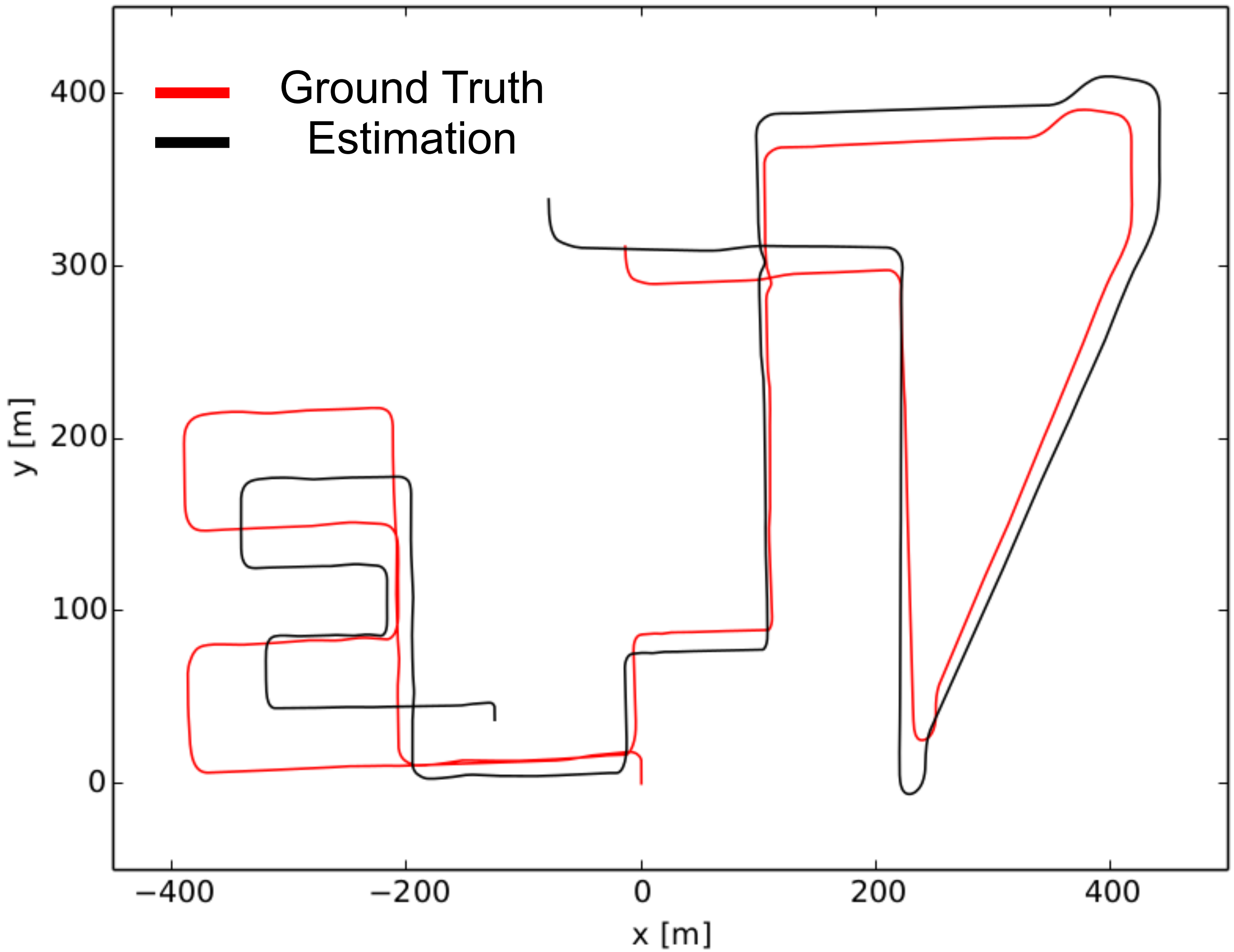
Monocular SLAM Works Nicely…



ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

…but Scale is NOT Observable



Scale drift!

Visual-Inertial SLAM

- Camera:
  - 3D localization
  - Environment Mapping
  - Place Recognition

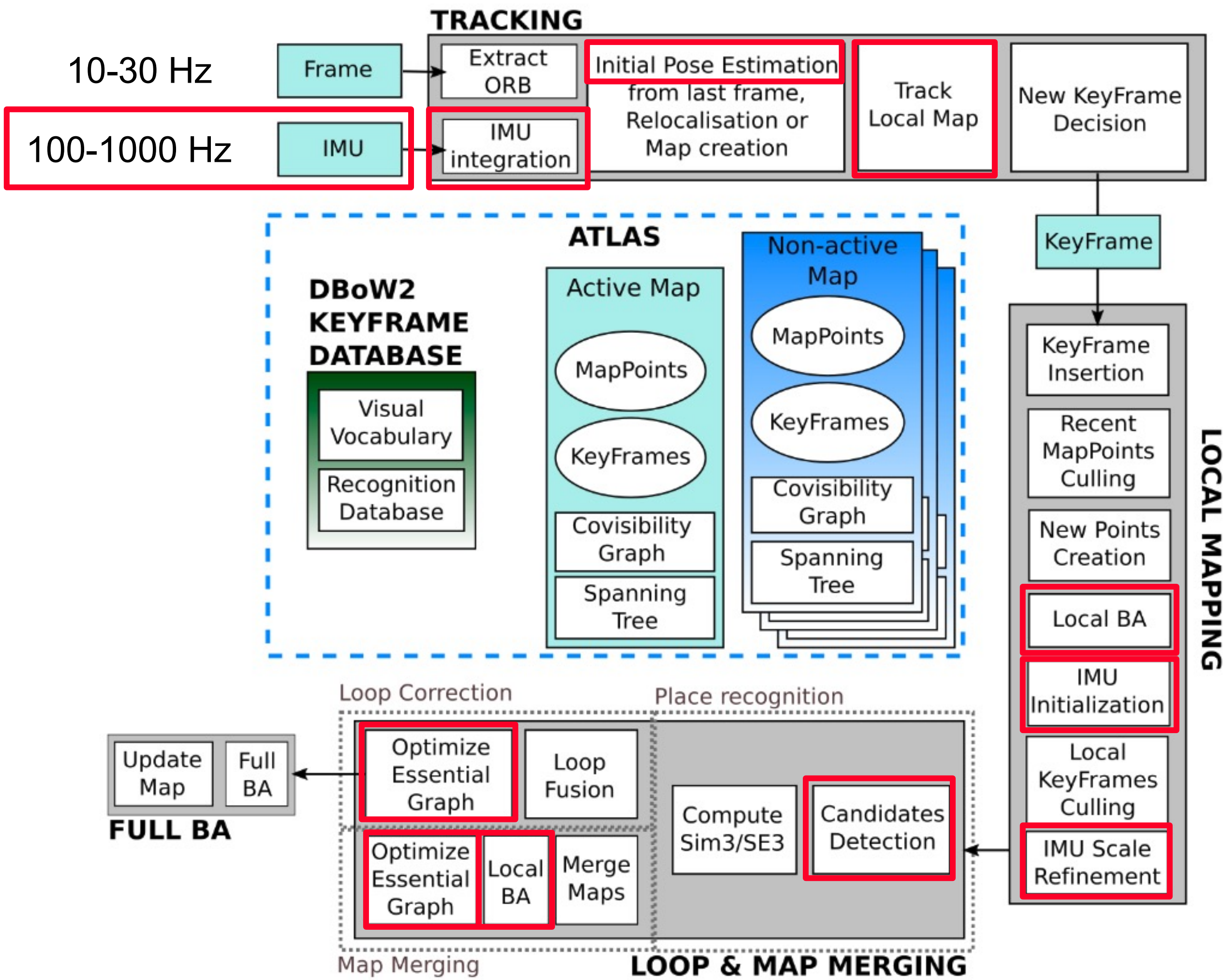
- IMU (Inertial Measurement Unit)
  - Angular velocity
  - Linear acceleration + gravity

→ Short-term motion prediction

→ Absolute roll and pitch

→ True scale estimation

IMU in Tracking, Mapping and Loop Closing



6.1.a IMU Sensor

- IMU measures angular velocity and linear acceleration in the body reference

Noise Bias

$$\begin{aligned}
\mathrm{R}_{\mathtt{WB}_{i+1}} &= \mathrm{R}_{\mathtt{WB}_i} \operatorname{Exp}\left(\left(\tilde{\boldsymbol{\omega}}_{\mathrm{B}_i} - \boldsymbol{b}_i^g\right)\Delta t\right) \\
\mathbf{v}_{\mathtt{WB}_{i+1}} &= \mathbf{v}_{\mathtt{WB}_i} + \mathbf{g}_{\mathtt{W}}\Delta t + \mathrm{R}_{\mathtt{WB}_i}\left(\tilde{\boldsymbol{a}}_{\mathrm{B}_i} - \boldsymbol{b}_i^a\right)\Delta t \\
\mathbf{p}_{\mathtt{WB}_{i+1}} &= \mathbf{p}_{\mathtt{WB}_i} + \mathbf{v}_{\mathtt{WB}_i}\Delta t + \frac{1}{2}\mathbf{g}_{\mathtt{W}}\Delta t^2 + \frac{1}{2}\mathrm{R}_{\mathtt{WB}_i}\left(\tilde{\boldsymbol{a}}_{\mathrm{B}_i} - \boldsymbol{b}_i^a\right)\Delta t^2
\end{aligned}$$

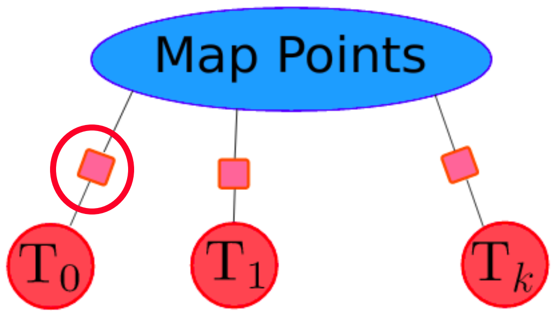
Initial Velocity Gravity

- Difficulties:
  - Measurement noise
  - Accelerometer and gyroscope biases
  - Direction of gravity unknown
  - Initial velocity unknown

Additional states to be estimated

Visual-Inertial SLAM: The Big Difference

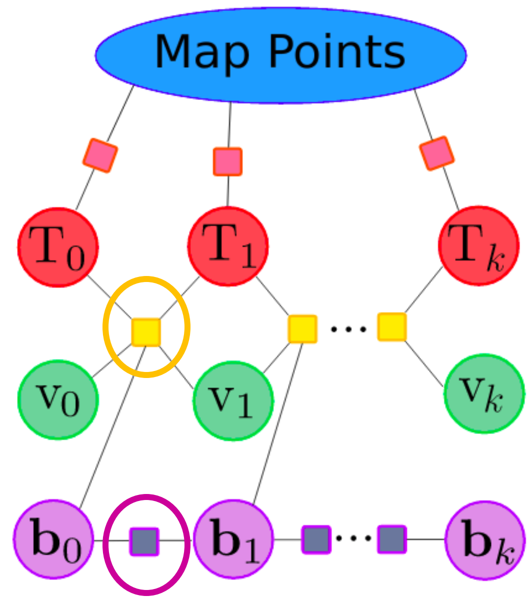
Visual Bundle Adjustment



Variables:
1. Camera poses
2. Map-Point positions

Residuals:
1. Reprojection error

Visual-Inertial Bundle Adjustment



Extra variables:
1. Camera velocities
2. IMU Biases

Extra residuals:
1. IMU measurements

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

IMU Sensor: References

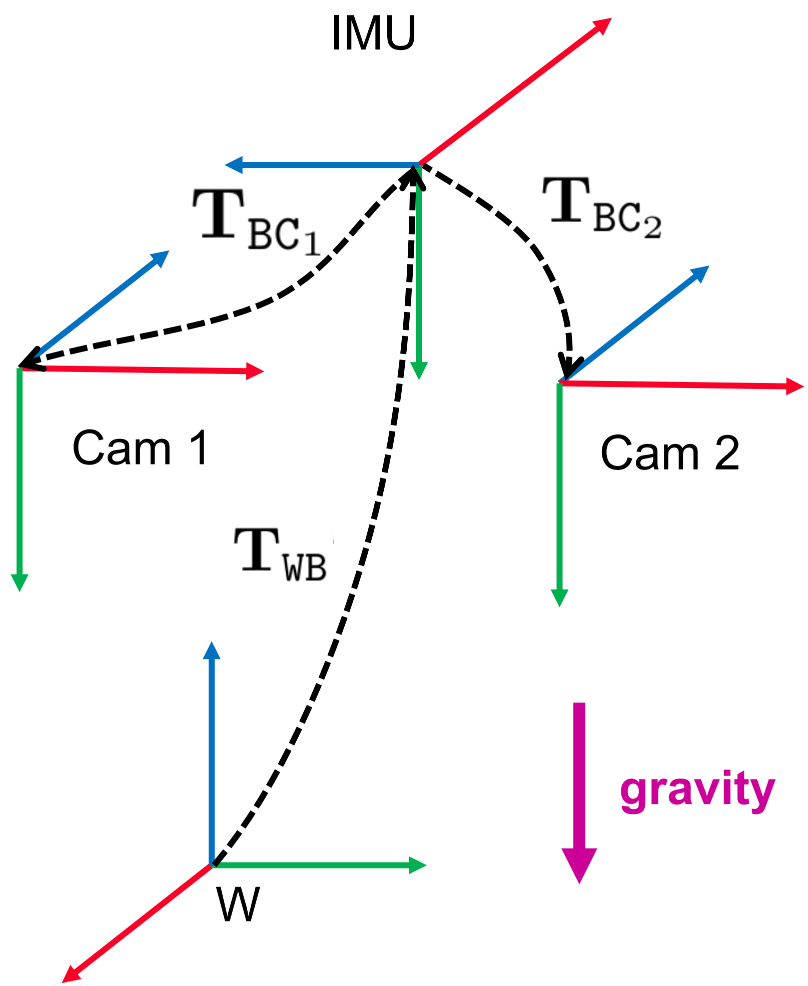
- Several sensors references
- A single optimizable reference: (IMU/Body)

State

$$\mathbf{T}_{\mathtt{WC}_1} = \mathbf{T}_{\mathtt{WB}}\mathbf{T}_{\mathtt{BC}_1}$$

$$\mathbf{T}_{\mathtt{WC}_2} = \mathbf{T}_{\mathtt{WB}}\mathbf{T}_{\mathtt{BC}_2}$$

From calibration



Intrinsic and Extrinsic parameters can be calibrated with Kalibr:

https://github.com/ethz-asl/kalibr

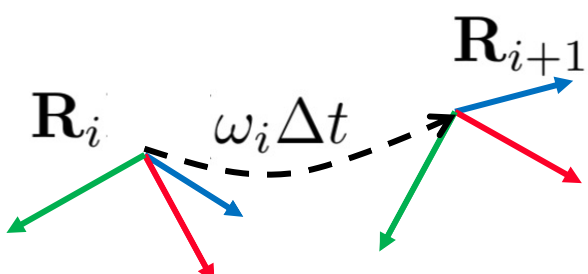
IMU Sensor: Gyroscope

- Measures angular velocity (rad/s) in body frame:

$$\tilde{\boldsymbol{\omega}}_{\mathrm{B}_i} = \boldsymbol{\omega}_{\mathrm{B}_i} + \boldsymbol{b}_i^g + \boldsymbol{\eta}_i^g \quad ; \quad \boldsymbol{\eta}_i^g \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}^g)$$ Measurement noise

$$\boldsymbol{b}_{i+1}^g = \boldsymbol{b}_i^g + \boldsymbol{\eta}_{rw,i}^g \quad ; \quad \boldsymbol{\eta}_{rw,i}^g \sim \mathcal{N}(\mathbf{0}, \boldsymbol{\Sigma}_{rw}^g)$$ Bias random walk

Two consecutive IMU orientations are related by:



$$\mathbf{R}_{i+1} = \mathbf{R}_i \mathrm{Exp}\left(\left(\tilde{\omega}_i - \mathbf{b}_i^g \right) \Delta t \right)$$

Exp : Takes 3 angles and returns a rotation matrix

Dropping W&B subindexes

```
EuRoC.yaml
IMU.NoiseGyro: 1.7e-4
IMU.GyroWalk: 1.93e-5
```

IMU Sensor: Accelerometer

- Measures linear acceleration (m/s²) in body frame, affected gravity, bias and noise:

$$\tilde{\boldsymbol{a}}_{\mathtt{B}_i} = \mathrm{R}_{\mathtt{B}_i\mathtt{W}}(\boldsymbol{a}_{\mathtt{W}} - \mathrm{g}_{\mathtt{W}}) + \boldsymbol{b}_i^a + \boldsymbol{\eta}_i^a \quad ; \quad \boldsymbol{\eta}_i^a \sim \mathcal{N}(0, \Sigma^a)$$ Measurement noise

$$\boldsymbol{b}_{i+1}^a = \boldsymbol{b}_i^a + \boldsymbol{\eta}_{rw,i}^a \quad ; \quad \boldsymbol{\eta}_{rw,i}^a \sim \mathcal{N}(0, \Sigma_{rw}^a)$$ Bias random walk

$$\boldsymbol{a}_{\mathtt{W}} = \mathrm{R}_{\mathtt{WB}_i}(\tilde{\boldsymbol{a}}_{\mathtt{B}_i} - \boldsymbol{b}_i^a) + \mathrm{g}_{\mathtt{W}} + \boldsymbol{\eta}_i^a$$

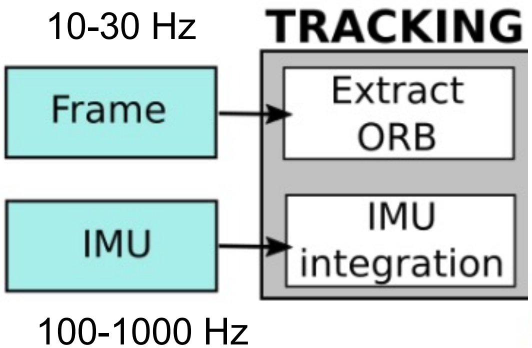
```
EuRoC.yaml
IMU.NoiseAcc: 2.0e-3
IMU.AccWalk: 3.0e-3
```

- Two consecutive velocities and positions:

$$\boxed{\begin{aligned} \mathbf{v}_{i+1} &= \mathbf{v}_i + \mathbf{g}\Delta t + \mathrm{R}_i\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^a\right)\Delta t \\ \mathbf{p}_{i+1} &= \mathbf{p}_i + \mathbf{v}_i\Delta t + \frac{1}{2}\mathbf{g}\Delta t^2 + \frac{1}{2}\mathrm{R}_i\left(\tilde{\boldsymbol{a}}_i - \boldsymbol{b}_i^a\right)\Delta t^2 \end{aligned}}$$

Dropping W&B

IMU Preintegration: Why?



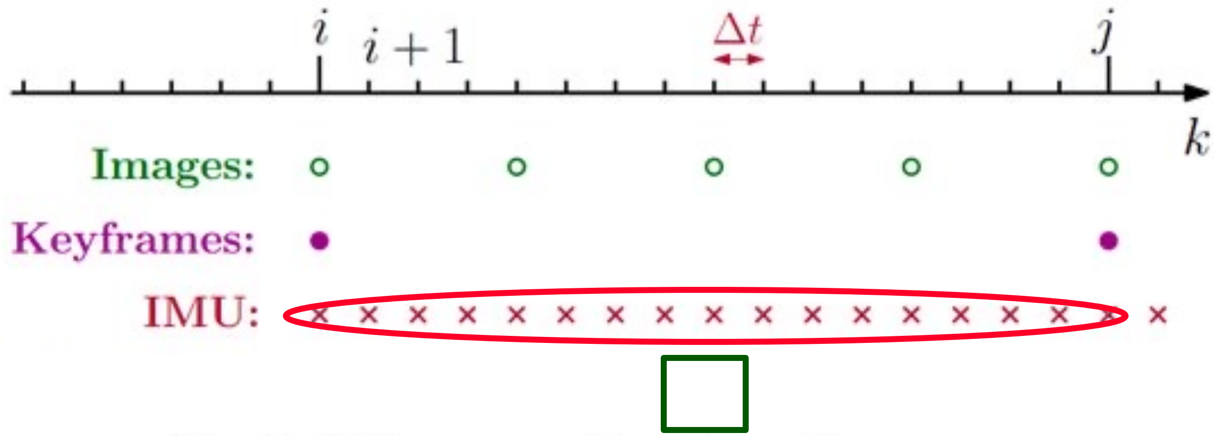


Fig. 4: Different rates for IMU and camera.

A lot of IMU measurement between Frames/KF

Computations for each measurement when Frame/KF states change

High computational burden

SOLUTION:
Summarize IMU measurements in a single precomputed term
PREINTEGRATION

When Frame/KF state change, we only need to **update a single preintegrated term**

"Visual-inertial-aided navigation...", Lupton et al, TRO 2011

"On-Manifold Preintegration..." Forster et al, TRO 2016

IMU Preintegration: Details

Preintegrated terms $\Delta \mathbf{R}_{ij}$ $\Delta \mathbf{v}_{ij}$ $\Delta \mathbf{p}_{ij}$ are computed at:

```
ImuTypes.h/.cc

Preintegrated::dR;
Preintegrated::dV;
Preintegrated::dP;
Preintegrated::IntegrateNewMeasurement()
```

Expressions and further details: http://rpg.ifi.uzh.ch/docs/RSS15_Forster_Supplementary.pdf

Preintegrated terms relate (i,j) frames/keyframes as follows:

1. Preintegrated rotation $\Delta \mathbf{R}_{ij}$: $\mathbf{R}_j = \mathbf{R}_i \Delta \mathbf{R}_{ij}$
2. Preintegrated velocity $\Delta \mathbf{v}_{ij}$: $\mathbf{v}_j = \mathbf{v}_i + \mathbf{g}\Delta t_{ij} + \mathbf{R}_i \Delta \mathbf{v}_{ij}$
3. Preintegrated position $\Delta \mathbf{p}_{ij}$: $\mathbf{p}_j = \mathbf{p}_i + \mathbf{v}_i \Delta t_{ij} + \frac{1}{2}\mathbf{g}\Delta t_{ij}^2 + \mathbf{R}_i \Delta \mathbf{p}_{ij}$

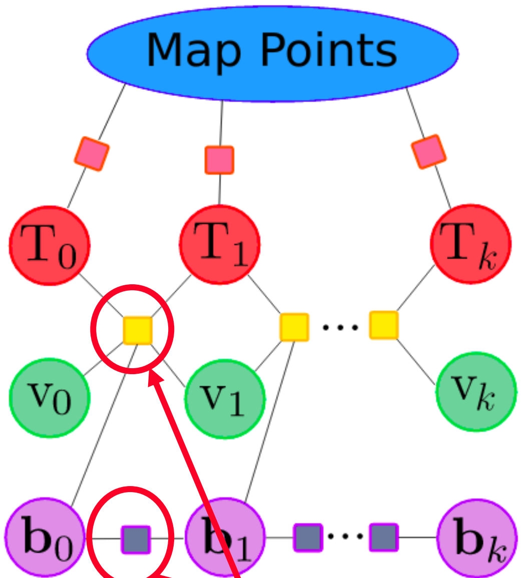
Remark: Preintegrated terms depend on bias

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza

IMU Preintegration: Residuals

Moving back to our Visual-Inertial BA...



How do we use preintegrated measurements to define residuals?

j-KF estimation		j-KF prediction from i-KF estimation and IMU preintegration
$\mathbf{R}_j$	$=$	$\mathbf{R}_i \Delta \mathbf{R}_{ij}$
$\mathbf{v}_j$	$=$	$\mathbf{v}_i + \mathbf{g}\Delta t_{ij} + \mathbf{R}_i \Delta \mathbf{v}_{ij}$
$\mathbf{p}_j$	$=$	$\mathbf{p}_i + \mathbf{v}_i \Delta t_{ij} + \frac{1}{2}\mathbf{g}\Delta t_{ij}^2 + \mathbf{R}_i \Delta \mathbf{p}_{ij}$

Preintegrated residual terms:

$$\mathbf{r}_{\Delta \mathbf{R}_{ij}} = \mathrm{Log}\left(\mathbf{R}_j^{\mathrm{T}} \mathbf{R}_i \Delta \mathbf{R}_{ij}\right)$$

$$\mathbf{r}_{\Delta \mathbf{v}_{ij}} = \mathbf{v}_j - \left(\mathbf{v}_i + \mathbf{g}\Delta t_{ij} + \mathbf{R}_i \Delta \mathbf{v}_{ij}\right)$$

$$\mathbf{r}_{\Delta \mathbf{p}_{ij}} = \mathbf{p}_j - \left(\mathbf{p}_i + \mathbf{v}_i \Delta t_{ij} + \frac{1}{2}\mathbf{g}\Delta t_{ij}^2 + \mathbf{R}_i \Delta \mathbf{p}_{ij}\right)$$

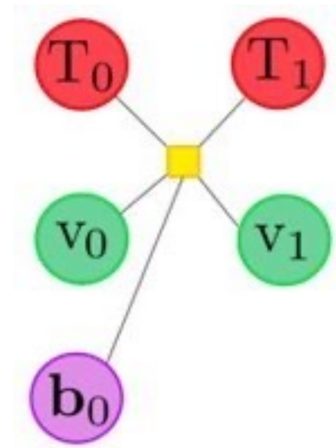
Log Maps rotation matrix to 3 angles vector

Bias random walk residual: $\mathbf{r}_{\mathbf{ba}_{ij}} = \mathbf{b}_j^a - \mathbf{b}_i^a \quad \mathbf{r}_{\mathbf{bg}_{ij}} = \mathbf{b}_j^g - \mathbf{b}_i^g$

IMU Preintegration: Residuals in g2o

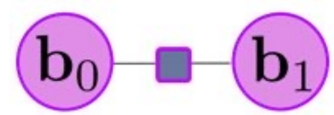
Edge for Inertial preintegration

```
G2oTypes.h/cc

EdgeInertial;
VertexPose;
VertexVelocity;
VertexGyroBias;
VertexAccBias;
```



Edge for bias random walk

```
G2oTypes.h/cc

EdgeGyroRW;
EdgeAccRW;
VertexGyroBias;
VertexAccBias;
```

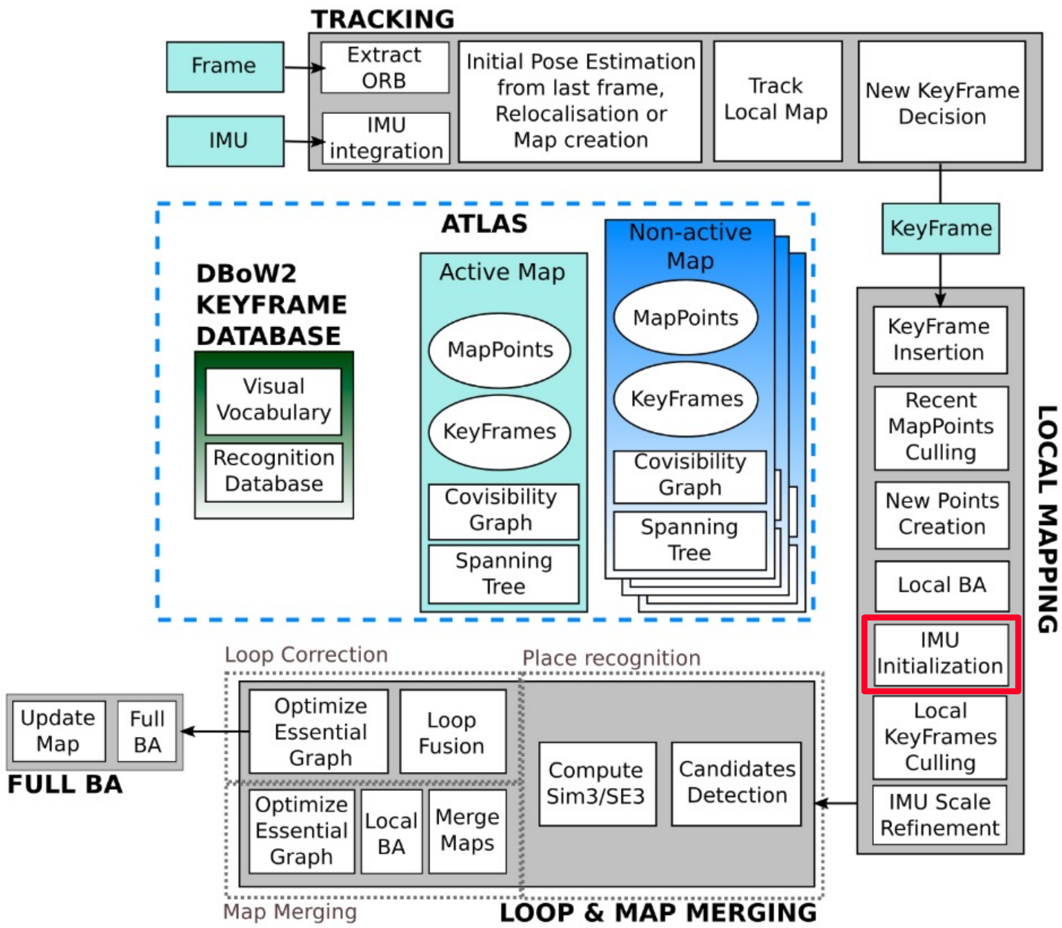


Expressions and further details: http://rpg.ifi.uzh.ch/docs/RSS15_Forster_Supplementary.pdf

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza

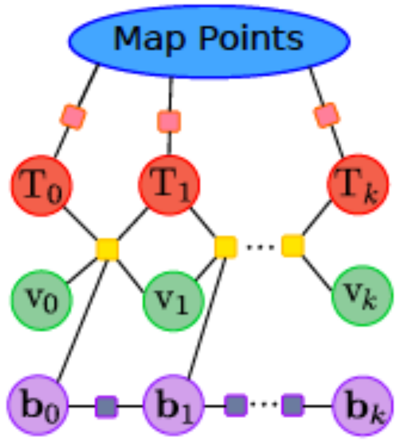
6.1.b IMU Initialization



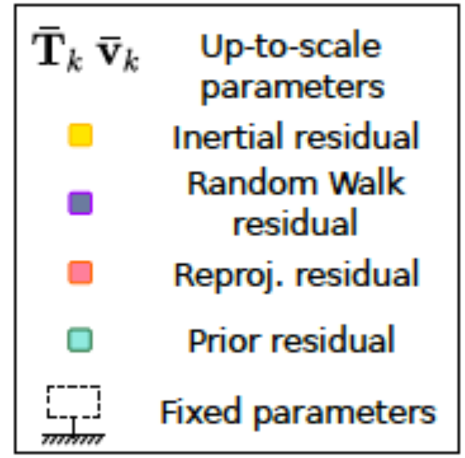
Goal: Find initial estimation for inertial parameters:

1. Velocities
2. Biases
3. Map scale
4. Gravity direction

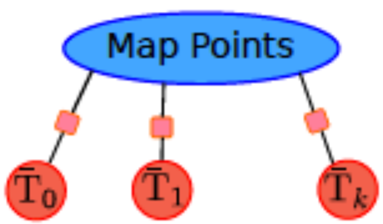
ORB-SLAM3: Visual-Inertial SLAM



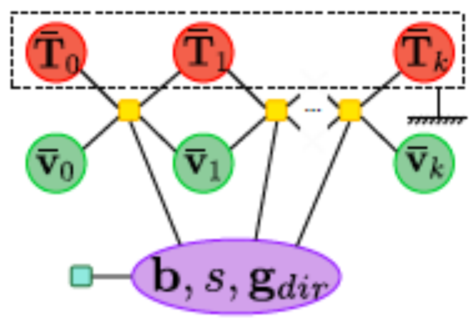
(a) Visual-Inertial



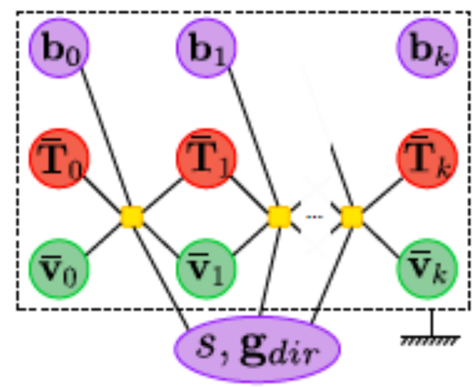
Initialization steps:



(b) Visual-Only



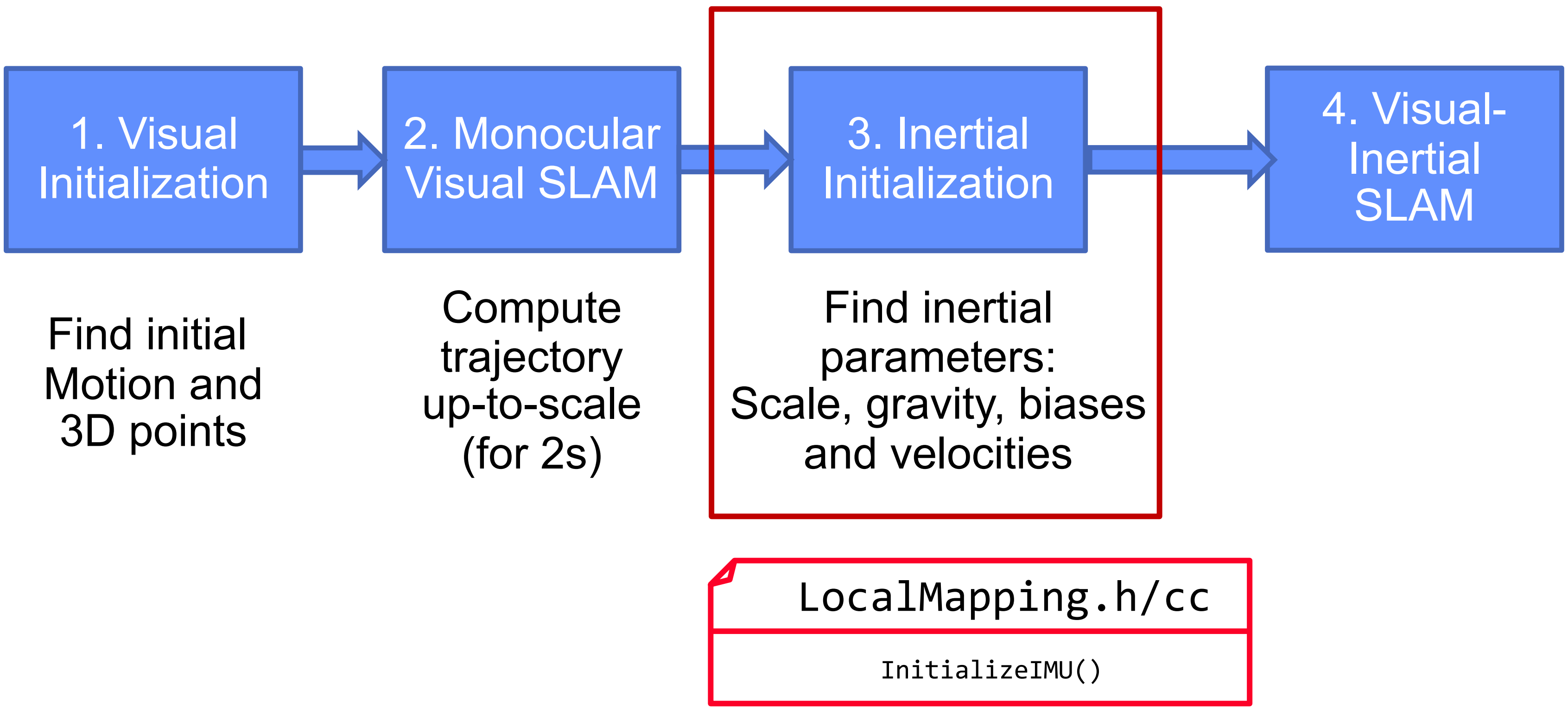
(c) Inertial-Only



(d) Scale and Gravity

IMU Initialization

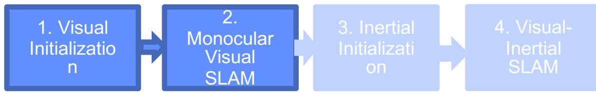
Approach: Split initialization process into small blocks.



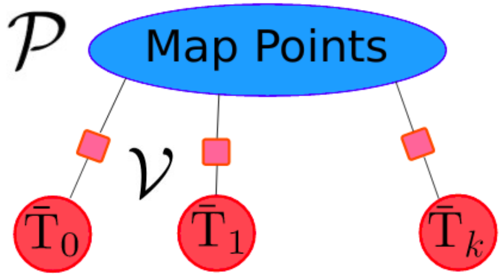
ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

IMU Initialization: MAP Approach



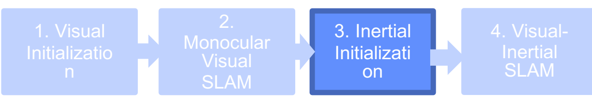
Visual Initialization and SLAM



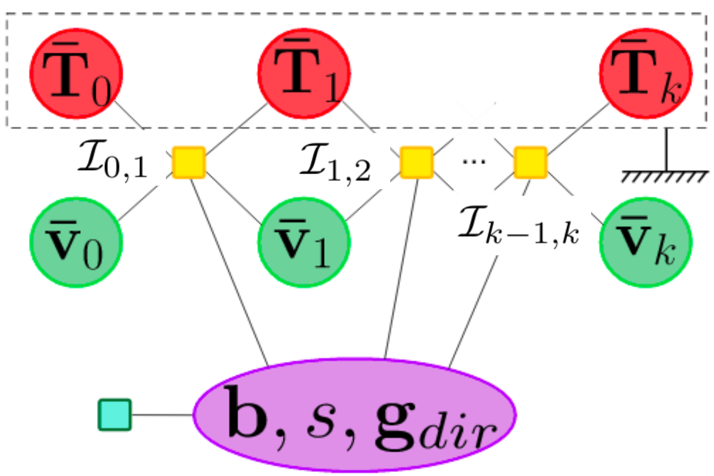
Equivalent MAP problem:

Visual observations

$$\{\bar{\mathbf{T}}_{0:k}, \mathcal{P}\}^* = \arg \max_{\bar{\mathbf{T}}_{0:k}, \mathcal{P}} p(\bar{\mathbf{T}}_{0:k}, \mathcal{P} | \mathcal{V})$$



IMU Initialization: Inertial-Only optimization



Equivalent MAP problem:

Inertial observations

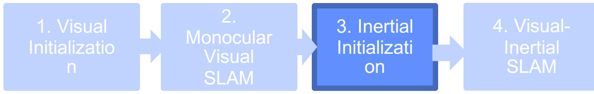
$$\mathcal{X}_k^* = \arg \max_{\mathcal{X}_k} p(\mathcal{X}_k | \mathcal{I}_{0:k})$$

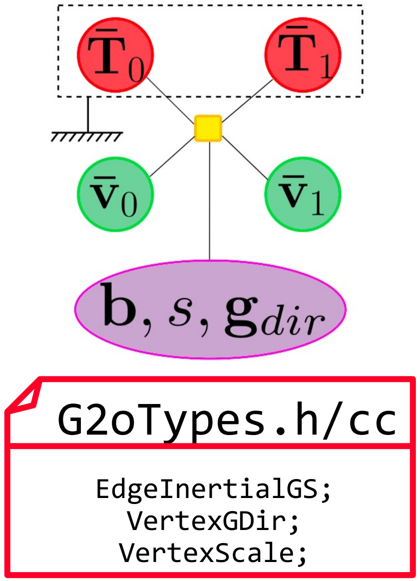
$$\mathcal{X}_k = \{s, \mathbf{g}, \mathbf{b}, \bar{\mathbf{v}}_{0:k}\}$$

Optimizer.h/cc
`InertialOptimization()`

- Only inertial measurements
- Only inertial variables
- Scale not estimated for stereo

IMU Initialization: Inertial-Only Optimization





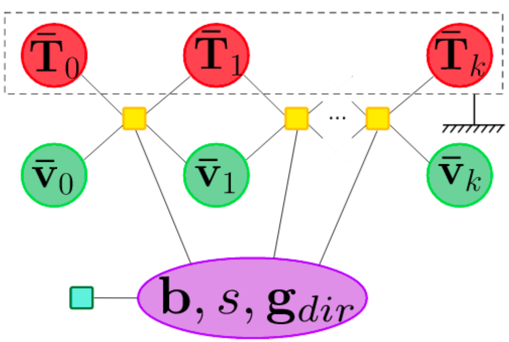
Residuals w.r.t up-to-scale poses and velocities, scale and gravity direction:

$$\mathbf{r}_{\Delta \mathrm{R}_{ij}} = \mathrm{Log}\left(\Delta \mathbf{R}_{ij}(\mathbf{b}^g)^{\mathrm{T}} \mathbf{R}_i^{\mathrm{T}} \mathbf{R}_j\right)$$

$$\mathbf{r}_{\Delta \mathrm{v}_{ij}} = \mathbf{R}_i^{\mathrm{T}}\left(s\bar{\mathbf{v}}_j - s\bar{\mathbf{v}}_i - \mathbf{R}_{\mathrm{w}g}\mathbf{g}_{\mathrm{I}}\Delta t_{ij}\right) - \Delta \mathbf{v}_{ij}$$

$$\mathbf{r}_{\Delta \mathrm{p}_{ij}} = \mathbf{R}_i^{\mathrm{T}}\left(s\bar{\mathbf{p}}_j - s\bar{\mathbf{p}}_i - s\bar{\mathbf{v}}_i \Delta t_{ij} - \frac{1}{2}\mathbf{R}_{\mathrm{w}g}\mathbf{g}_{\mathrm{I}}\Delta t_{ij}^2\right) - \Delta \mathbf{p}_{ij}$$

Solve optimization:

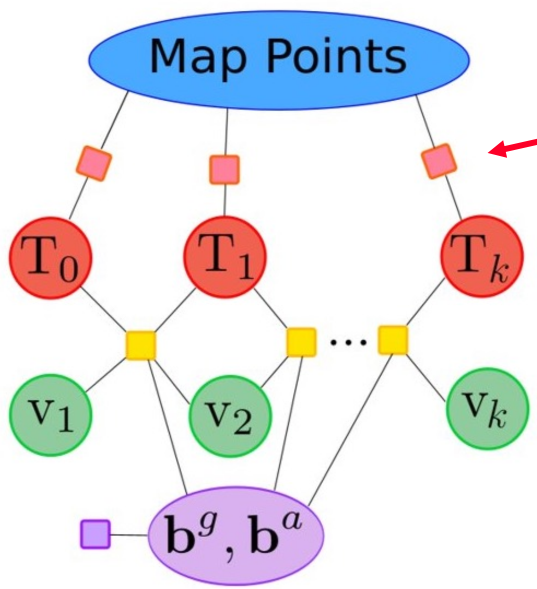


- Levenberg-Marquardt
- Optimization converges in 10 ms
- Map is rotated and scaled after opt.

IMU Initialization: Inertial-Only Optimization

1. Visual Initialization → 2. Monocular Visual SLAM → 3. Inertial Initialization → 4. Visual-Inertial SLAM

Solve final joint visual-Inertial Optimization.



Visual measurements in the body reference:

$$\mathbf{r}_{ij} = \mathbf{u}_{ij} - \Pi\left(\mathbf{T}_{CB}\mathbf{T}_i^{-1}\mathbf{x}_j\right)$$

```
G2oTypes.h/cc

EdgeMono;
EdgeStereo;
```

Remark: IMU measurements do not use robust cost function since missassociations do not exist.

Finally, just after 2 seconds, launch Visual-Inertial SLAM

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

IMU initialization Example

Inertial-Only Optimization for Visual-Inertial Initialization

Carlos Campos, José M. M. Montiel and Juan Tardós

Sequence: V1_02_medium
Dataset: EuRoC MAV Dataset

IMU Initialization: Practical Tips

IMU needs to be excited during initialization (first 2 seconds) to properly estimate initial parameters:

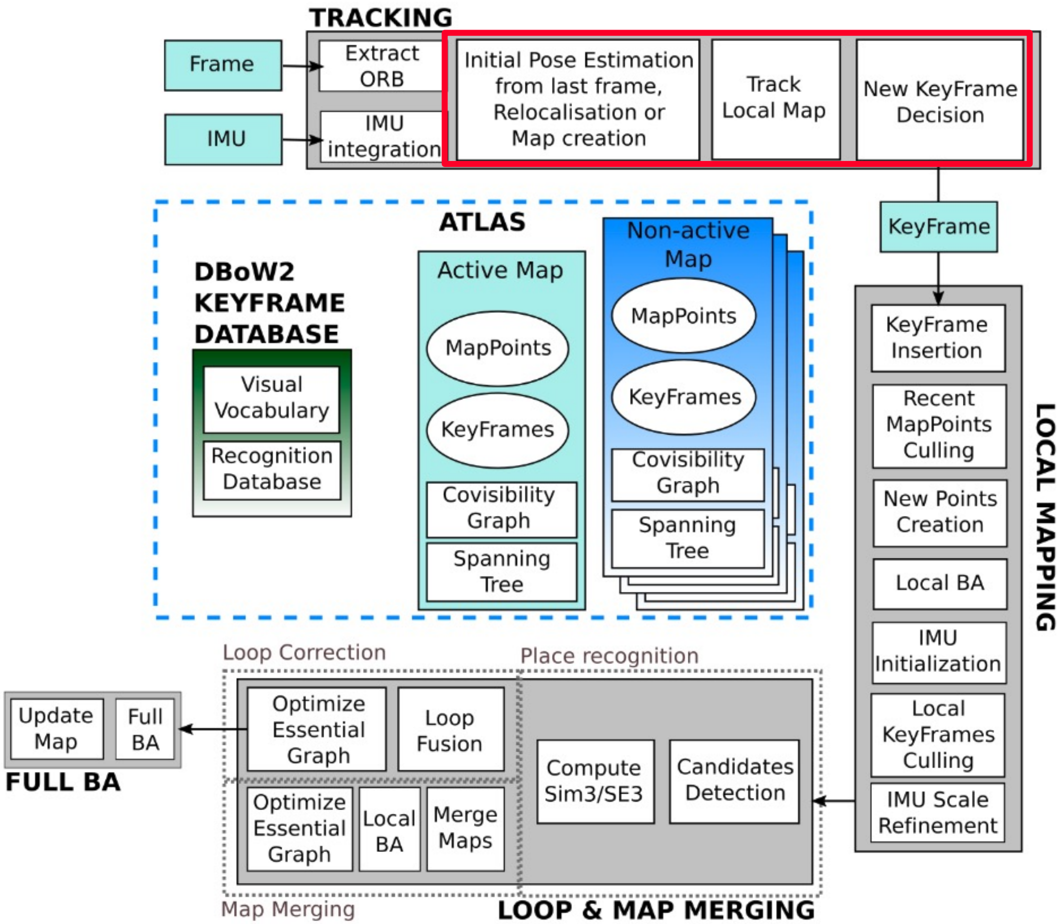
AVOID:

- No motion
- Pure rotational motion
- Pure translational motion
- Constant velocity motion
- Rotation only around one axis

TRY TO:

- Combine rotations with translations
- Make sure gravity points in different directions in IMU reference
- 8-shaped trajectories are a good choice

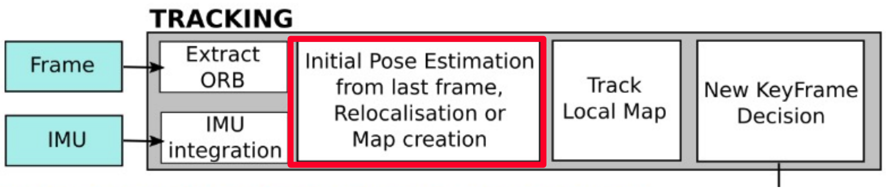
6.1.c Visual-Inertial Tracking



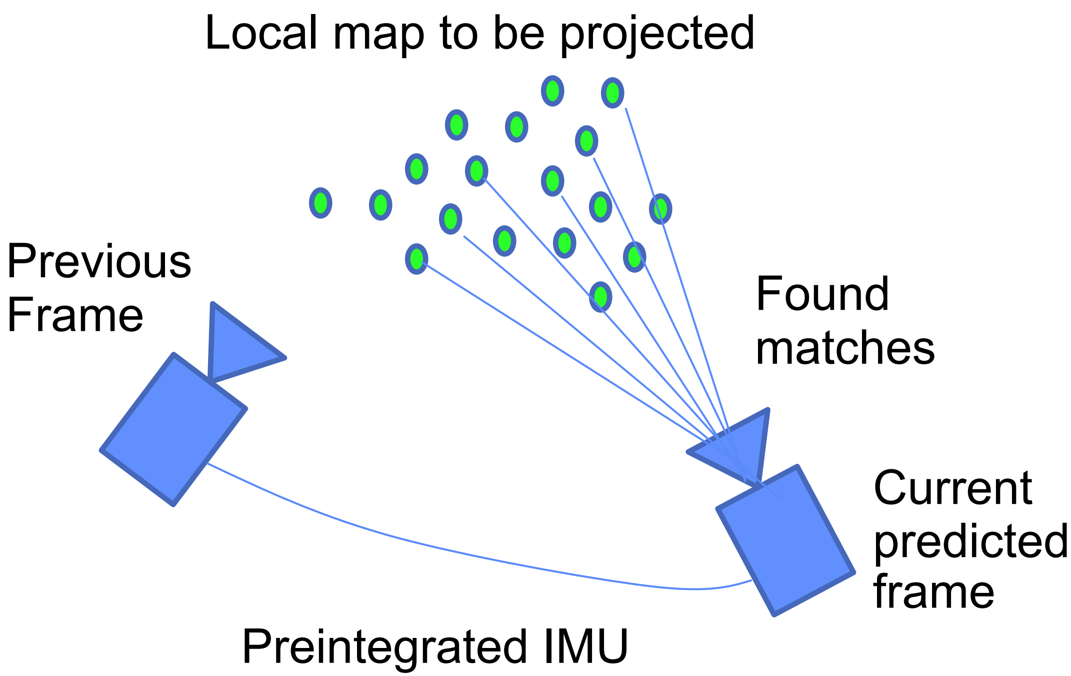
Main differences w.r.t. visual tracking:

- Initial pose estimation
- Local map tracking
- Lost case & new Keyframe decision

VI Tracking: Initial Pose Estimation

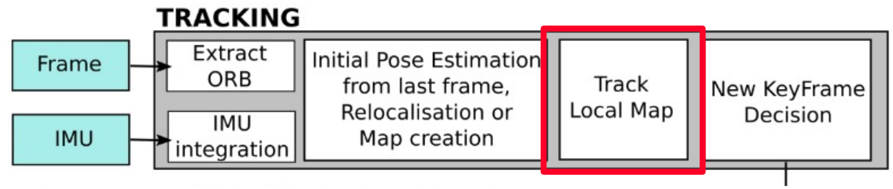


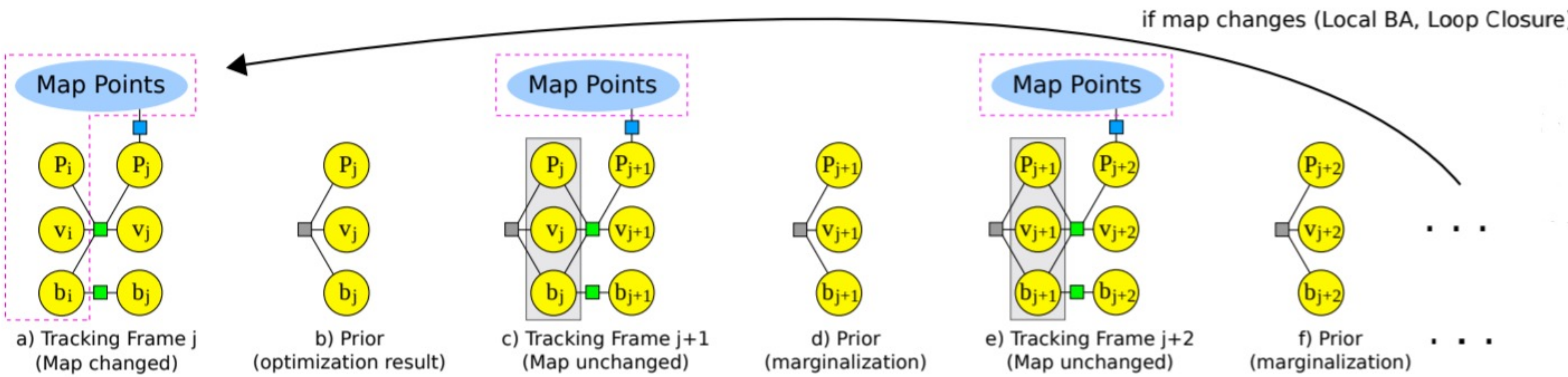
- Predict pose using IMU preintegration instead of using pose-only BA
- Project into smaller windows



VI Tracking: Local Map Tracking

Optimize current frame state from found visual matches and preintegrated IMU:





"ORB-SLAM VI..." Mur et al. 2017

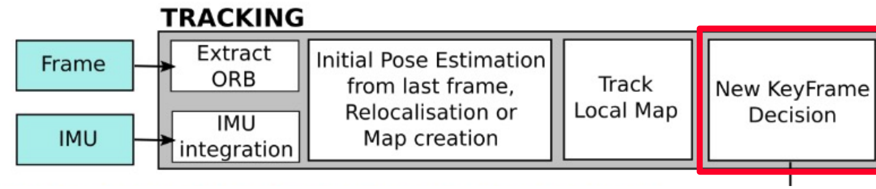
- If map is not updated, optimize w.r.t. previous frame and marginalize it.
- If map is updated, optimize w.r.t. previous key-frame

Optimizer.h/cc

```
PoseInertialOptimizationLastKeyFrame();
PoseInertialOptimizationLastFrame();
```

VI Tracking: Tracking Loss & Keyframe Decision

When tracked Map Points drop below 15, system passes to RECENTLY_LOST state:



- Continue estimating pose with IMU for 5 seconds.

•Project local map points to find new matches (larger search window)
•Create keyframes for triangulating new points.

System robust to fast rotations or short occlusions

- If tracking remains for more than 5 seconds in this state, system changes to **LOST** state

•Relocalization
•Create new map

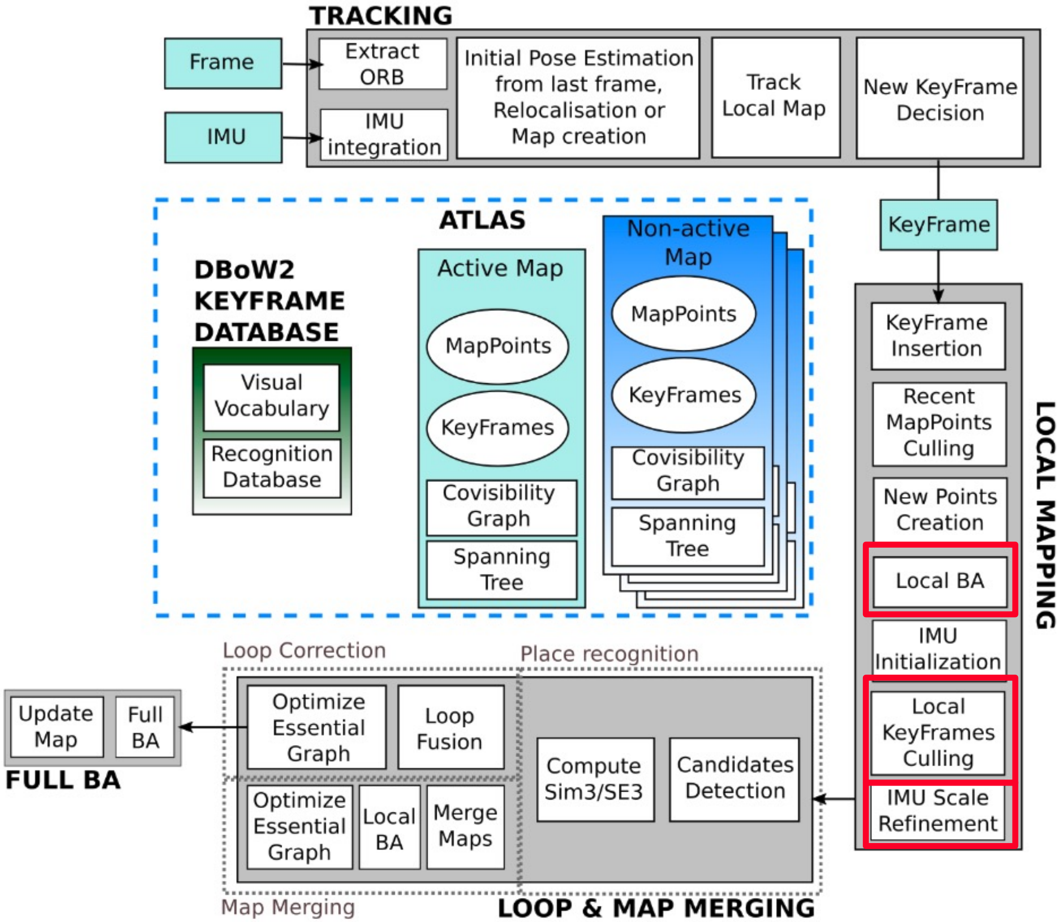
System robust to long losses (Multimap system)

New keyframe decision:
•Same as visual
•Insert a keyframe at least every 0.5s

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza

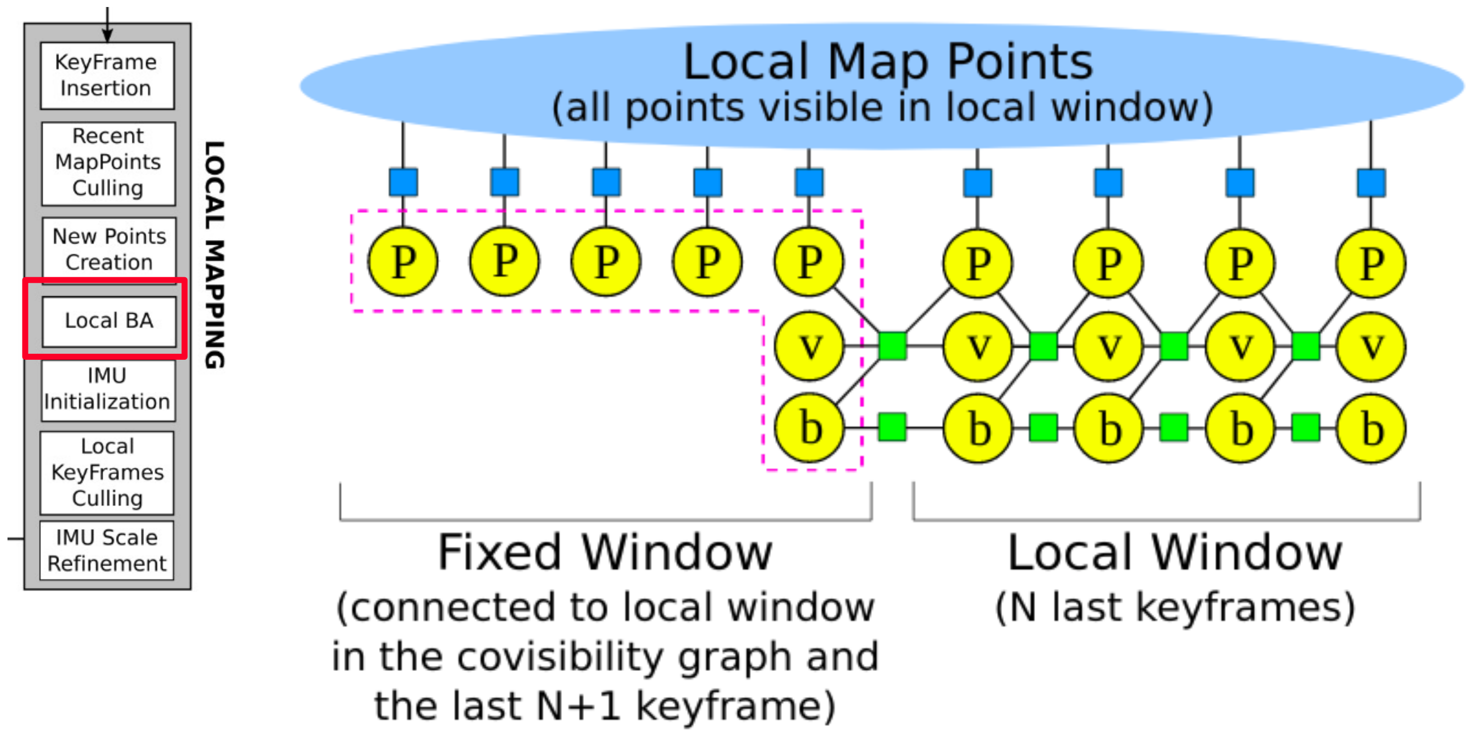
6.1.d Visual-Inertial Mapping



Differences w.r.t. visual mapping:

- Local Visual-Inertial BA
- Local Keyframe Culling
- Scale/Gravity refinement

VI Mapping: Local Visual-Inertial BA

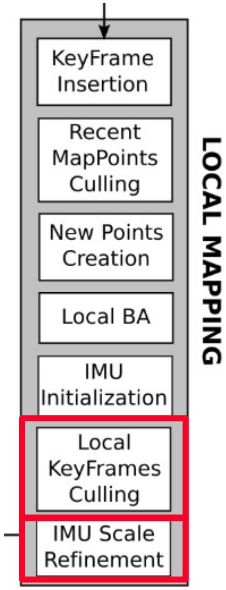


- Covisible keyframes to local window are added but remain fixed during optimization

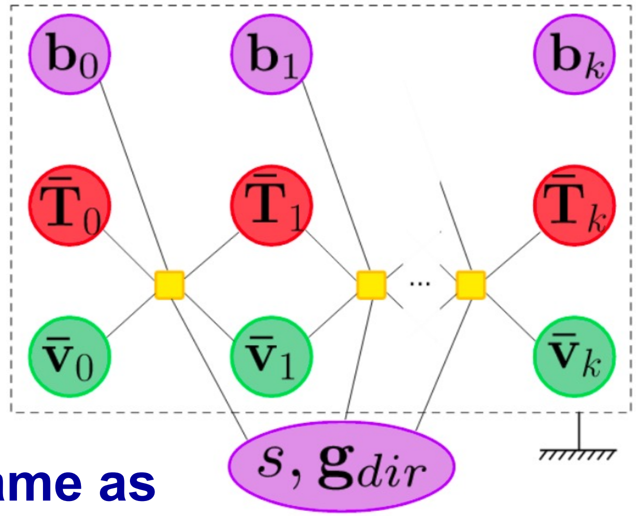
LocalMapping.h/cc

```
LocalInertialBA();
```

VI Mapping: Scale/Gravity Refinement and Keyframe Culling



Since we are using a temporal window, scale or gravity direction errors outside the window may not be corrected by Local-Inertial BA



All keyframes included, but fixed.

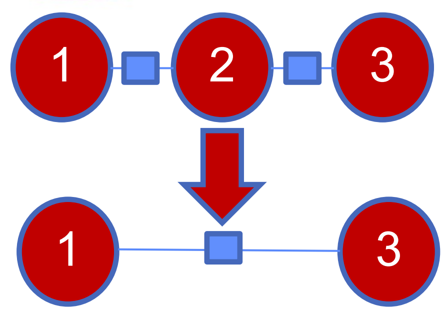
Very efficient, only 3 variables.

```
LocalMapping.h/cc

ScaleRefinement();
```

Keyframe culling: same as visual, but…

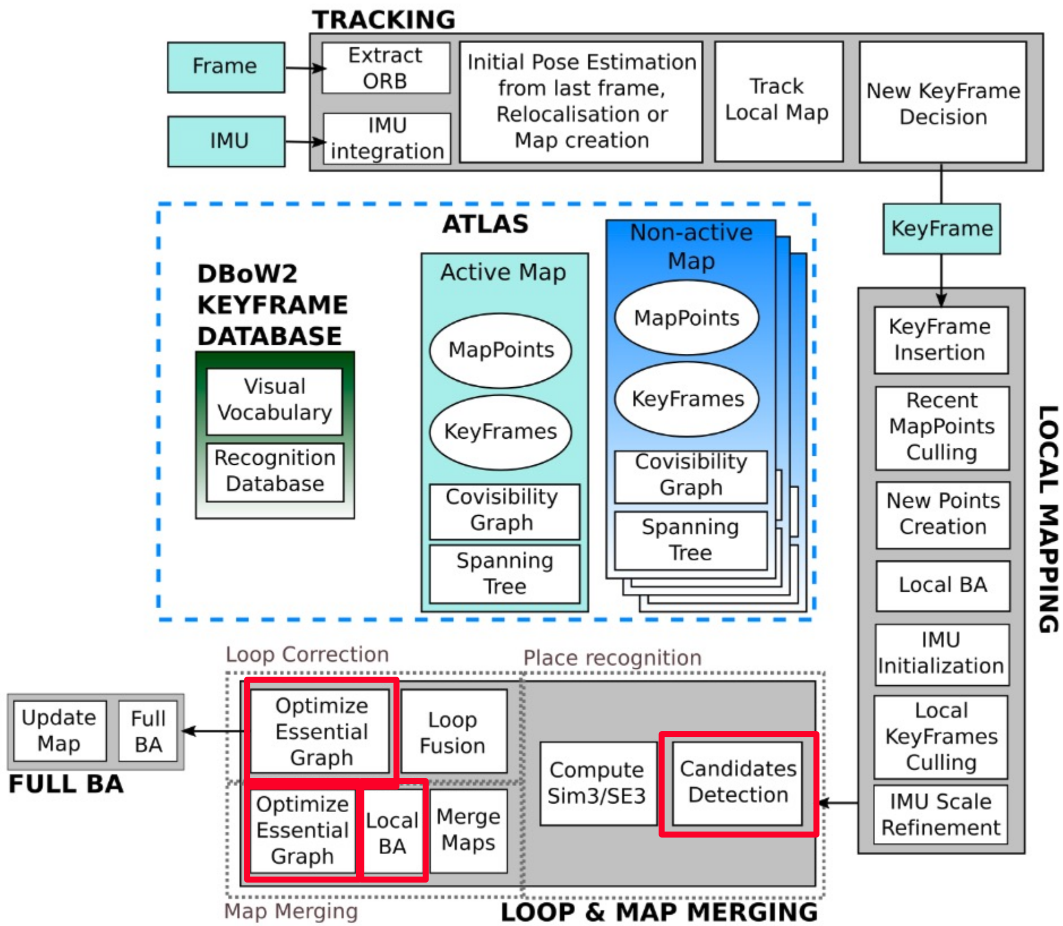
- Two consecutive KF are not more distant than 3s.

- Merge inertial preintegrations:

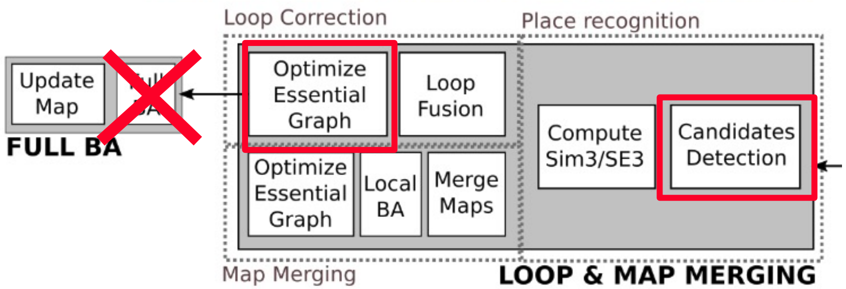



```
ImuTypes.h/cc

Preintegrated::MergePrevious();
```

6.1.e Loop-Closing and Map-Merging with IMU



Loop Closing with IMU



1. IMU increases robustness of loop-closing detection:

- When loop closing candidates are found, check gravity direction change:
  - If > 0.5 degree → Discard loop

2. IMU makes pose-graph more efficient:

- Pitch and Roll are observable
- Optimize poses with only 4 Degrees of Freedom

3. Thanks to lower visual-inertial drift, Global BA after loop-closing is not required

Optimizer.h/cc

```
OptimizeEssentialGraph4DoF;
```

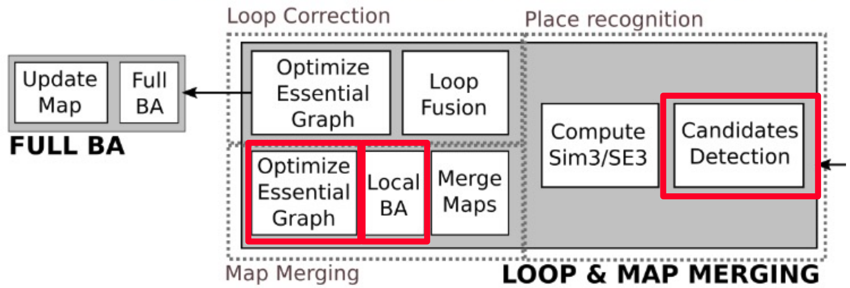
G2oTypes.h/cc

```
Edge4DoF;
Vertex4DoF;
```


ROBOTICS UNIVERSIDAD DE ZARAGOZA

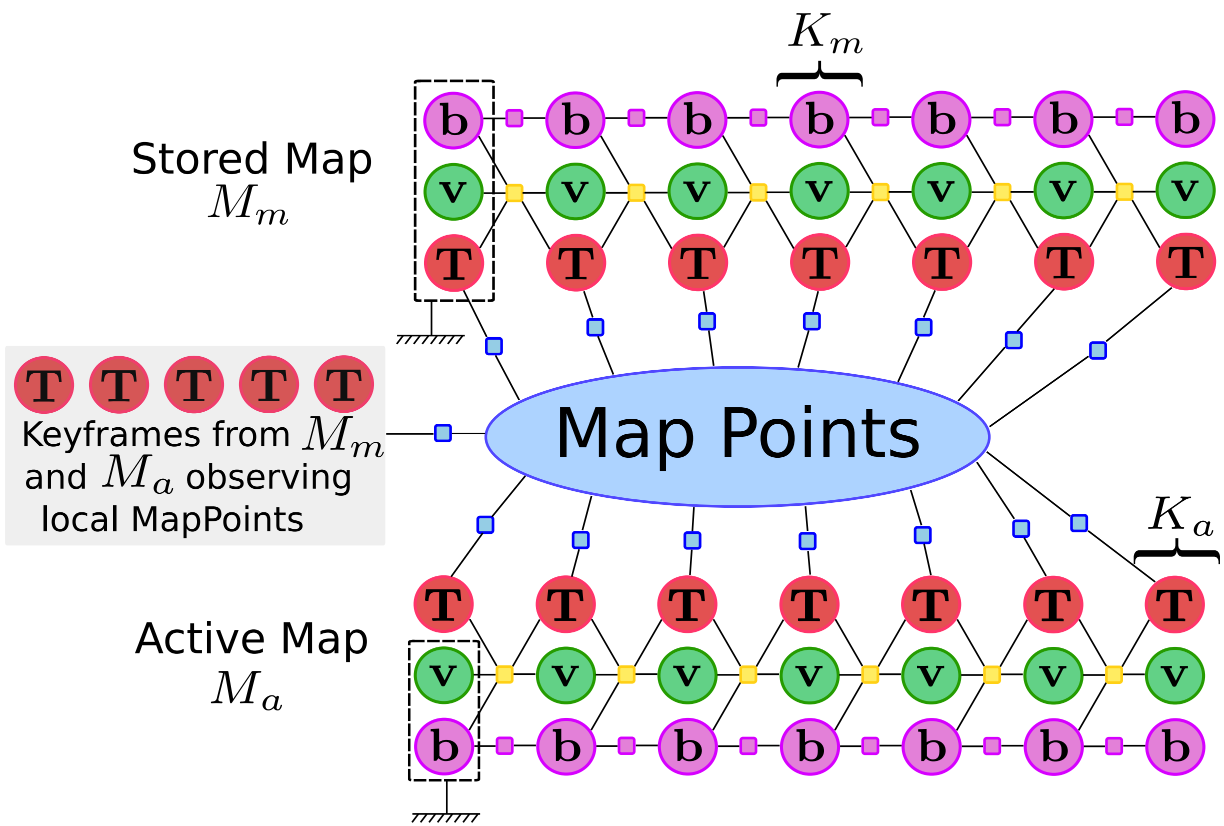
Universidad Zaragoza 1542

Map Merging with IMU



Similar to visual case, but…

- Gravity check
- 4DoF optimizations
- Temporal windows for stored and active map in merging BA.



Optimizer.h/cc

```
MergeInertialBA();
```

6.1.f Results: EuRoc Dataset

Inertial-Only Optimization for Visual-Inertial Initialization

Carlos Campos, José M. M. Montiel and Juan Tardós

Sequence: V1_02_medium
Dataset: EuRoC MAV Dataset

ORB-SLAM3: Visual Results

			MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg[1]
Monocular	ORB-SLAM [4]	ATE[2,3]	0.071	0.067	0.071	0.082	0.060	**0.015**	0.020	-	**0.021**	**0.018**	-	0.047*
	DSO [27]	ATE	0.046	0.046	0.172	3.810	0.110	0.089	0.107	0.903	0.044	0.132	1.152	0.601
	SVO [24]	ATE	0.100	0.120	0.410	0.430	0.300	0.070	0.210	-	0.110	0.110	1.080	0.294*
	DSM [31]	ATE	0.039	0.036	0.055	**0.057**	0.067	0.095	0.059	0.076	0.056	0.057	**0.784**	**0.126**
	ORB-SLAM3 (ours)	ATE	**0.017**	**0.017**	**0.031**	0.066	**0.044**	0.033	**0.016**	**0.037**	**0.021**	0.022	-	0.030*
Stereo	ORB-SLAM2 [3]	ATE	0.035	**0.018**	0.028	0.119	0.060	**0.035**	**0.020**	**0.048**	0.037	0.035	-	0.044*
	VINS-Fusion [44]	ATE	0.540	0.460	0.330	0.780	0.500	0.550	0.230	-	0.230	0.200	-	0.424*
	SVO [24]	ATE	0.040	0.070	0.270	0.170	0.120	0.040	0.040	0.070	0.050	0.090	0.790	0.159
	ORB-SLAM3 (ours)	ATE (m)	**0.025**	0.022	**0.027**	**0.089**	**0.058**	**0.035**	0.021	0.049	**0.032**	**0.027**	**0.361**	**0.068**

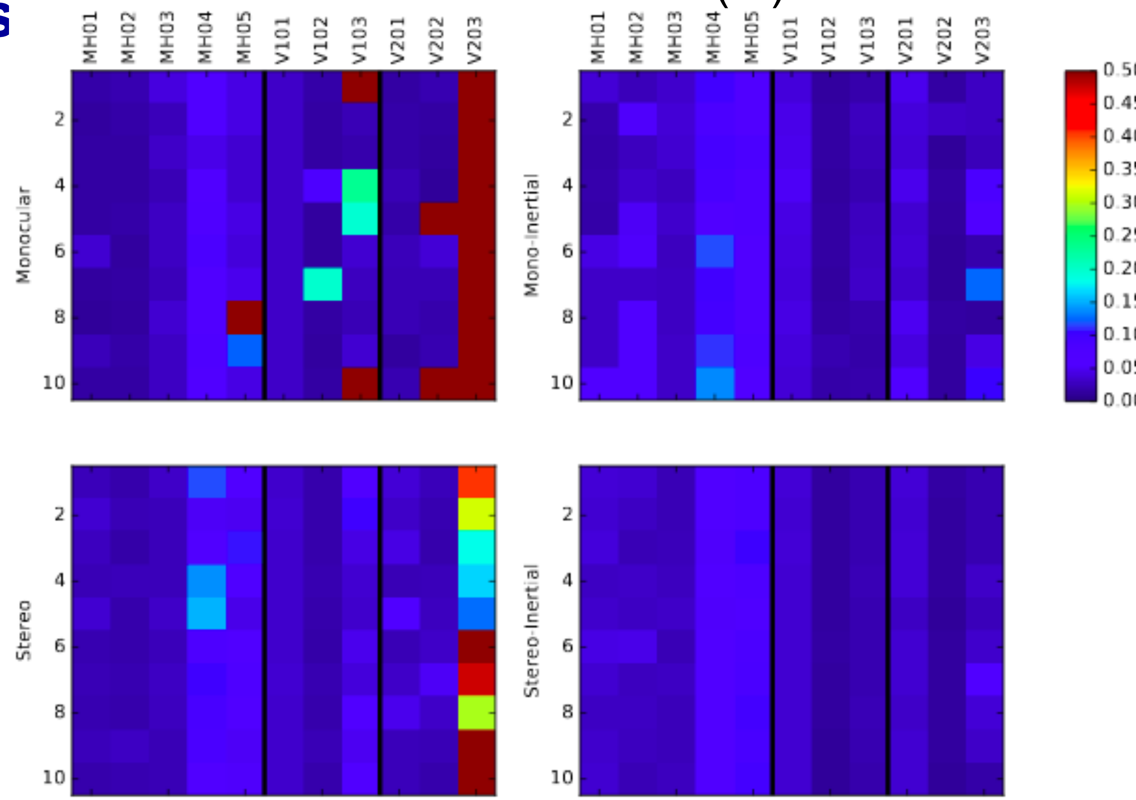
ORB-SLAM3: Visual-Inertial Results

			MH01	MH02	MH03	MH04	MH05	V101	V102	V103	V201	V202	V203	Avg[1]
Monocular Inertial	MCSKF [80]	ATE[5]	0.420	0.450	0.230	0.370	0.480	0.340	0.200	0.670	0.100	0.160	1.130	0.414
	OKVIS [39]	ATE[5]	0.160	0.220	0.240	0.340	0.470	0.090	0.200	0.240	0.130	0.160	0.290	0.231
	ROVIO [42]	ATE[5]	0.210	0.250	0.250	0.490	0.520	0.100	0.100	0.140	0.120	0.140	0.140	0.224
	ORBSLAM-VI [4]	ATE[2,3] scale error[2,3]	0.075 0.5	0.084 0.8	0.087 1.5	0.217 3.5	0.082 0.5	**0.027** 0.9	0.028 0.8	- -	**0.032** 0.2	0.041 1.4	0.074 0.7	0.075* 1.1*
	VINS-Mono [7]	ATE[4]	0.084	0.105	0.074	0.122	0.147	0.047	0.066	0.180	0.056	0.090	0.244	0.110
	VI-DSO [46]	ATE scale error	0.062 1.1	**0.044** 0.5	0.117 0.4	0.132 0.2	0.121 0.8	0.059 1.1	0.067 1.1	0.096 0.8	0.040 1.2	0.062 0.3	0.174 0.4	0.089 0.7
	ORB-SLAM3 (ours)	ATE scale error	**0.032** 0.7	0.053 1.0	**0.033** 0.3	**0.099** 1.0	**0.071** 0.6	0.043 1.5	**0.016** 0.5	**0.025** 1.1	0.041 0.5	**0.015** 0.3	**0.037** 0.9	**0.042** 0.8
Stereo Inertial	VINS-Fusion [44]	ATE[4]	0.166	0.152	0.125	0.280	0.284	0.076	0.069	0.114	0.066	0.091	0.096	0.138
	BASALT [47]	ATE[3]	0.080	0.060	0.050	0.100	**0.080**	0.040	0.020	0.030	**0.030**	0.020	-	0.051*
	Kimera [8]	ATE	0.080	0.090	0.110	0.150	0.240	0.050	0.110	0.120	0.070	0.100	0.190	0.119
	ORB-SLAM3 (ours)	ATE scale error	**0.037** 0.7	**0.031** 0.2	**0.026** 0.2	**0.059** 0.4	0.086 1.0	**0.037** 0.6	**0.014** 0.6	**0.023** 0.6	0.037 1.4	**0.014** 0.2	**0.029** 0.8	**0.036** 0.6

Visual-Inertial Results: EuRoC

- **Comparison for different sensor configurations**
  - IMU renders system **more accurate**
  - IMU renders system **more robust**

EuRoC Dataset
RMSE ATE (m)



Visual

Visual-Inertial

Visual-Inertial Results: TUM VI Benchmarck

Universidad Zaragoza 1542

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Instituto Universitario de Investigación en Ingeniería de Aragón Universidad Zaragoza

ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM

Carlos Campos*, Richard Elvira*, Juan J. Gómez Rodríguez,
José M. M. Montiel and Juan D. Tardós

Dataset: TUM VI Benchmark
Sequences: Room1+Slides1+Slides2+Slides3
Setup: Monocular-Inertial

Visual-Inertial Results: TUM VI Benchmarck

- **Comparison against SOTA (mono/stereo) in a public dataset.**
- **ORB-SLAM3 gets best results for indoor** sequences:
  - Mid-term data association.
- **ORB-SLAM3 gets best results for outdoor** sequences:
  - Exploratory trajectories (No loops)
- **ORB-SLAM3 gets competitive results for challenging** sequences.

TUM-VI Dataset

Seq.	Mono-Inertial		Stereo-Inertial				Length (m)	LC
	VINS-Mono	ORB-SLAM3	OKVIS	ROVIO	BASALT	ORB-SLAM3		
corridor1	0.63	**0.04**	0.33	0.47	0.34	**0.03**	305	✓
corridor2	0.95	**0.02**	0.47	0.75	0.42	**0.02**	322	✓
corridor3	1.56	**0.31**	0.57	0.85	0.35	**0.02**	300	✓
corridor4	0.25	**0.17**	0.26	**0.13**	0.21	0.21	114	
corridor5	0.77	**0.03**	0.39	2.09	0.37	**0.01**	270	✓
magistrale1	2.19	**0.56**	3.49	4.52	1.20	**0.24**	918	✓
magistrale2	3.11	**0.52**	2.73	13.43	1.11	**0.52**	561	✓
magistrale3	**0.40**	4.89	1.22	14.80	**0.74**	1.86	566	
magistrale4	5.12	**0.13**	0.77	39.73	1.58	**0.16**	688	✓
magistrale5	**0.85**	1.03	1.62	3.47	**0.60**	1.13	458	✓
magistrale6	2.29	**1.30**	3.91	X	3.23	**0.97**	771	
outdoors1	74.96	**70.79**	X	101.95	255.04	**32.23**	2656	
outdoors2	133.46	**14.98**	73.86	21.67	64.61	**10.42**	1601	
outdoors3	**36.99**	39.63*	32.38	**26.10**	38.26	54.77	1531	
outdoors4	**16.46**	25.26	19.51	X	17.53	**11.61**	928	
outdoors5	130.63	**14.87**	13.12	54.32	**7.89**	8.95	1168	✓
outdoors6	133.60	**16.84**	96.51	149.14	65.50	**10.70**	2045	
outdoors7	21.90	**7.59**	13.61	49.01	**4.07**	4.58	1748	✓
outdoors8	83.36	**27.88**	16.31	36.03	13.53	**11.02**	986	
room1	0.07	**0.01**	0.06	0.16	0.09	**0.01**	146	✓
room2	0.07	**0.02**	0.11	0.33	0.07	**0.01**	142	✓
room3	0.11	**0.04**	0.07	0.15	0.13	**0.01**	135	✓
room4	0.04	**0.01**	0.03	0.09	0.05	**0.01**	68	✓
room5	0.20	**0.02**	0.07	0.12	0.13	**0.01**	131	✓
room6	0.08	**0.01**	0.04	0.05	0.02	**0.01**	67	✓
slides1	**0.68**	0.97	0.86	13.73	**0.32**	0.41	289	
slides2	**0.84**	1.06	2.15	0.81	**0.32**	0.49	299	
slides3	**0.69**	**0.69**	2.58	4.68	0.89	**0.47**	383	

Computational Cost

Table VI: Running time of the main parts of our tracking and mapping threads compared to ORB-SLAM2, on EuRoC V202 (mean time and standard deviation in ms).

	System	ORB-SLAM2	ORB-SLAM3	ORB-SLAM3	ORB-SLAM3	ORB-SLAM3
Settings	Sensor	Stereo	Monocular	Stereo	Mono-Inertial	Stereo-Inertial
	Resolution	752×480	752×480	752×480	752×480	752×480
	Cam. FPS	20Hz	20Hz	20Hz	20Hz	20Hz
	IMU	-	-	-	200Hz	200HZ
	ORB Feat.	1200	1000	1200	1000	1200
	RMS ATE	0.035	0.022	0.027	0.015	0.014
Tracking	Stereo rect.	3.07±0.80	-	1.30±0.42	-	1.26±0.30
	ORB extract	11.20±2.00	11.82±4.37	14.80±4.12	11.28±3.76	14.42±3.11
	Stereo match	10.38±2.57	-	3.16±0.83	-	3.10±0.72
	IMU integr.	-	-	-	0.89±0.20	0.77±0.14
	Pose pred	2.20±0.72	2.31±0.91	2.89±0.99	0.11±0.39	0.17±0.74
	LM Track	9.89±4.95	5.30±2.25	6.52±2.80	8.51±2.33	10.58±2.58
	New KF dec	0.20±0.43	0.05±0.03	0.28±0.73	0.05±0.03	0.34±0.64
	Total	37.87±7.49	24.10±11.41	31.94±8.85	25.30±17.21	32.67±10.18
Mapping	KF Insert	8.72±3.60	12.04±5.39	10.88±4.68	16.16±7.89	12.08±3.39
	MP Culling	0.25±0.09	0.09±0.03	0.31±0.12	0.08±0.03	0.22±0.12
	MP Creation	36.88±14.53	47.96±20.93	52.24±22.53	63.69±25.10	58.06±22.40
	LBA	139.61±124.92	244.33±245.52	189.44±152.56	152.43±62.21	160.91±45.61
	KF Culling	4.37±4.73	16.82±13.88	7.53±5.89	24.61±17.27	14.14±13.54
	Total	173.81±139.07	318.18±277.79	241.48±177.53	258.33±110.64	246.07±77.07
Map Size	KFs	278	271	245	284	143
	MPs	14593	9879	12925	8876	10175

Better performance for free

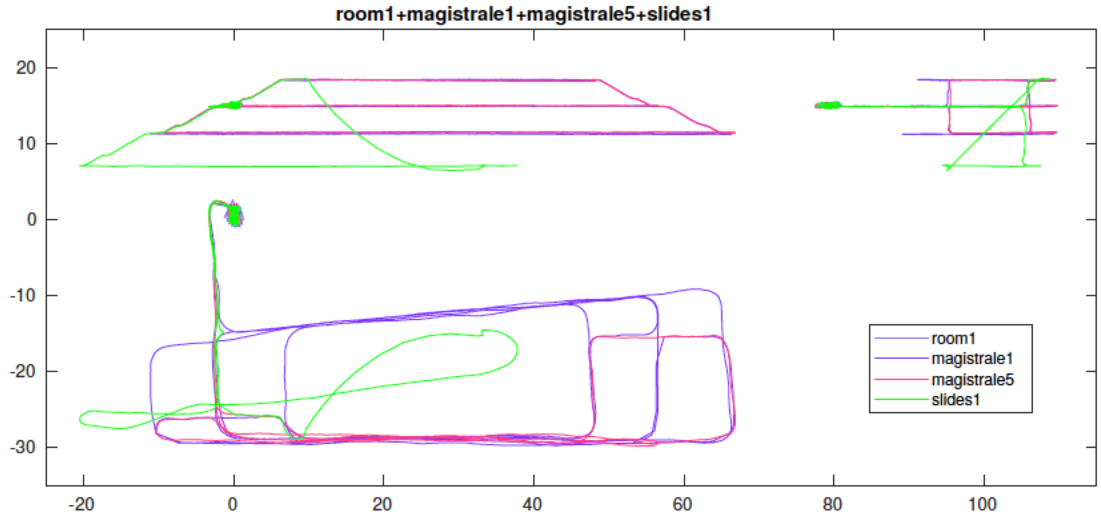
Visual-Inertial SLAM: Conclusions

Using a **small**, **cheap** and **simple** IMU sensor improves visual SLAM:

- **Robust to short occlusions** or lack of visual features
- **More accurate**
- Gets **true scale in the monocular case**
- Gets body **pose at a higer rate** than camera
- **No extra computational cost**

6.2: Multi-Map SLAM

Juan D. Tardós, Richard Elvira
Universidad de Zaragoza, Spain
robots.unizar.es/SLAMLAB



Lesson 6. Visual-Inertial and Multi-Map

2. Multi-Map SLAM

a) New Map Creation

b) Map Merge Detection

c) Map Merging

d) Multi-Map Results

Readings:

- C Campos, R Elvira, JJ Gómez Rodríguez, JMM Montiel, JD Tardós: ORB-SLAM3: An Accurate Open-Source Library for Visual, **Visual-Inertial and Multi-Map** SLAM. IEEE Trans. Robotics, 2021.
- R Elvira, JD Tardós, JMM Montiel: ORBSLAM-Atlas: A Robust and Accurate Multi-map System, IROS 2019

ROBOTICS UNIVERSIDAD DE ZARAGOZA

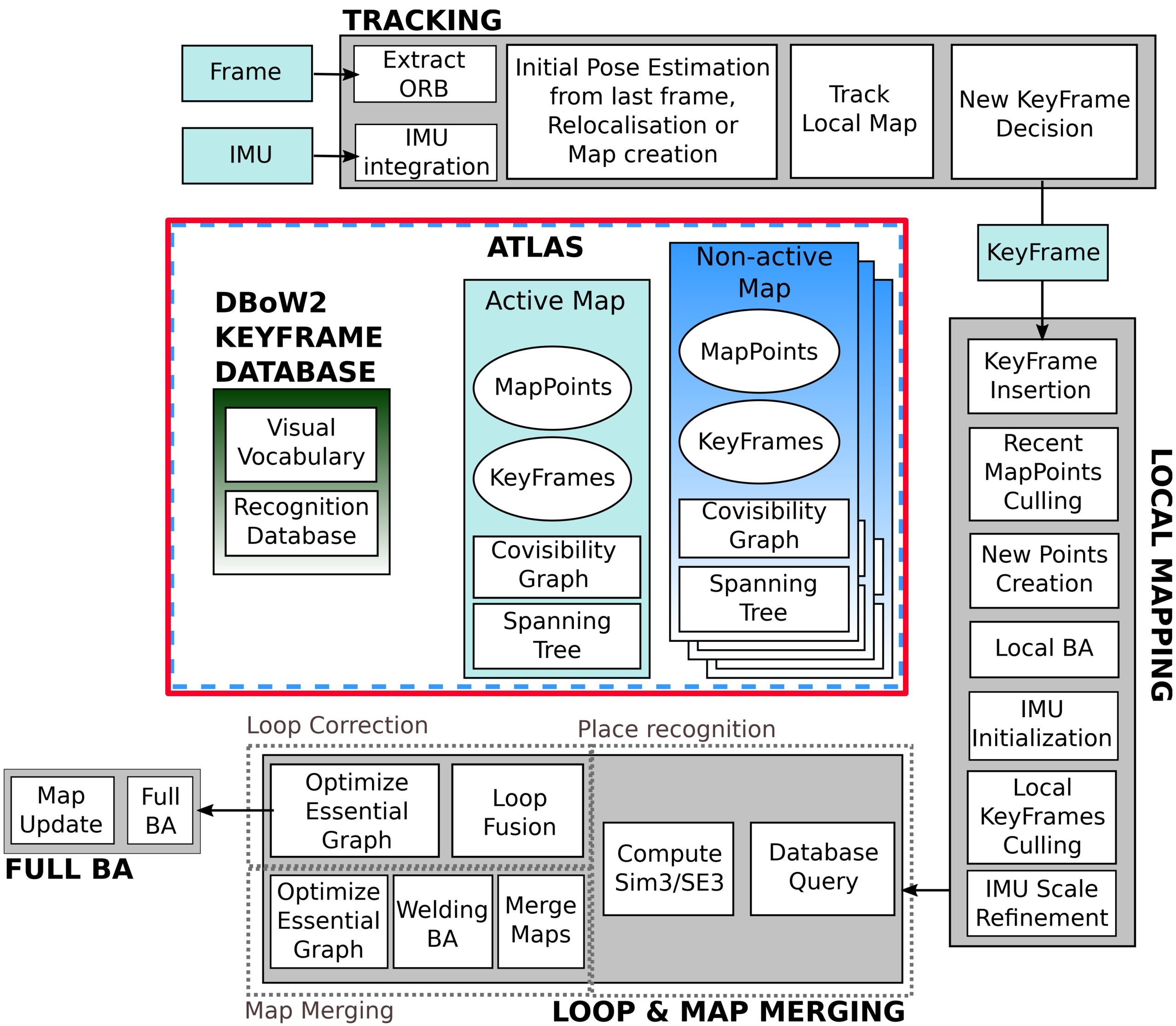
Universidad Zaragoza 1542

ORB-SLAM3 Highlights

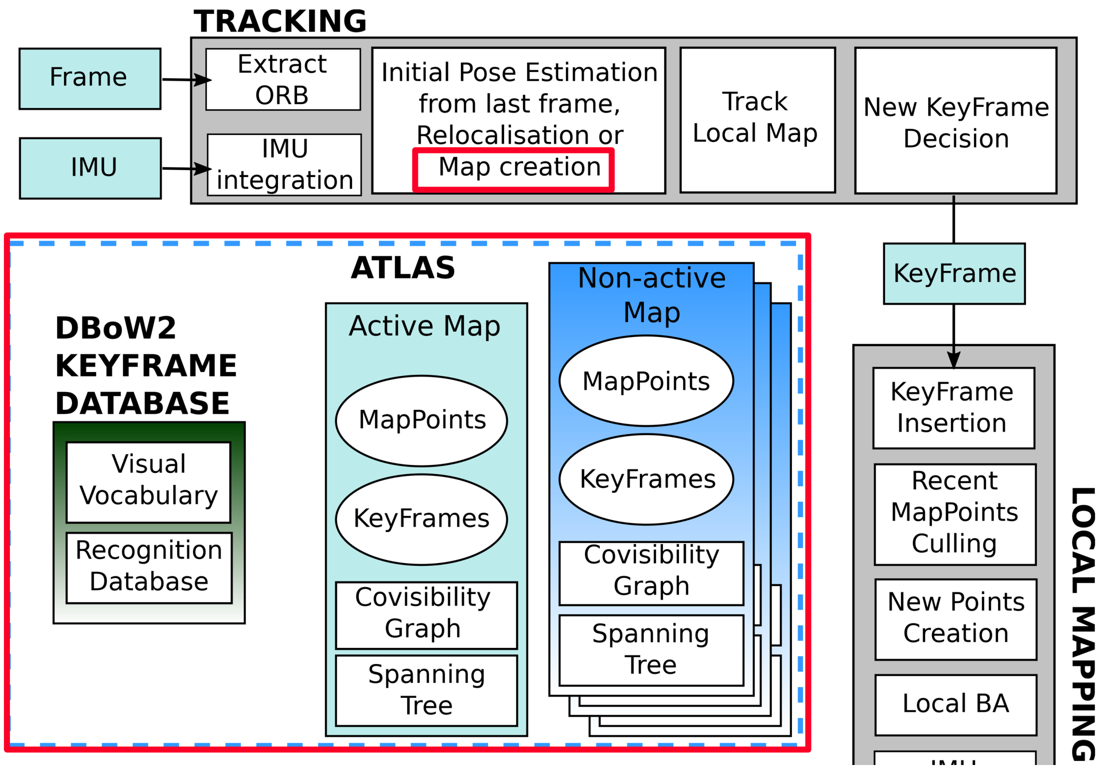
- Visual and Visual-Inertial SLAM
- Pin-hole and Fisheye lens models
- Multi-Map and Multi-Session
- Real-time operation in large environments
- Data association with ORB features:
  - Short term: from previous images
  - Mid-term: from local map
  - Long term: for Relocalization, Loop closing and Map Merging

C Campos, R Elvira, JJ Gómez Rodríguez, JMM Montiel, JD Tardós
ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM.
IEEE Transactions on Robotics 37 (6), 1874-1890, Dec 2021

ORB-SLAM3: Multi-Map (Atlas)



6.2.a Map Creation



- At the start of a mapping sessions
- If camera tracking is lost for more that 5 seconds

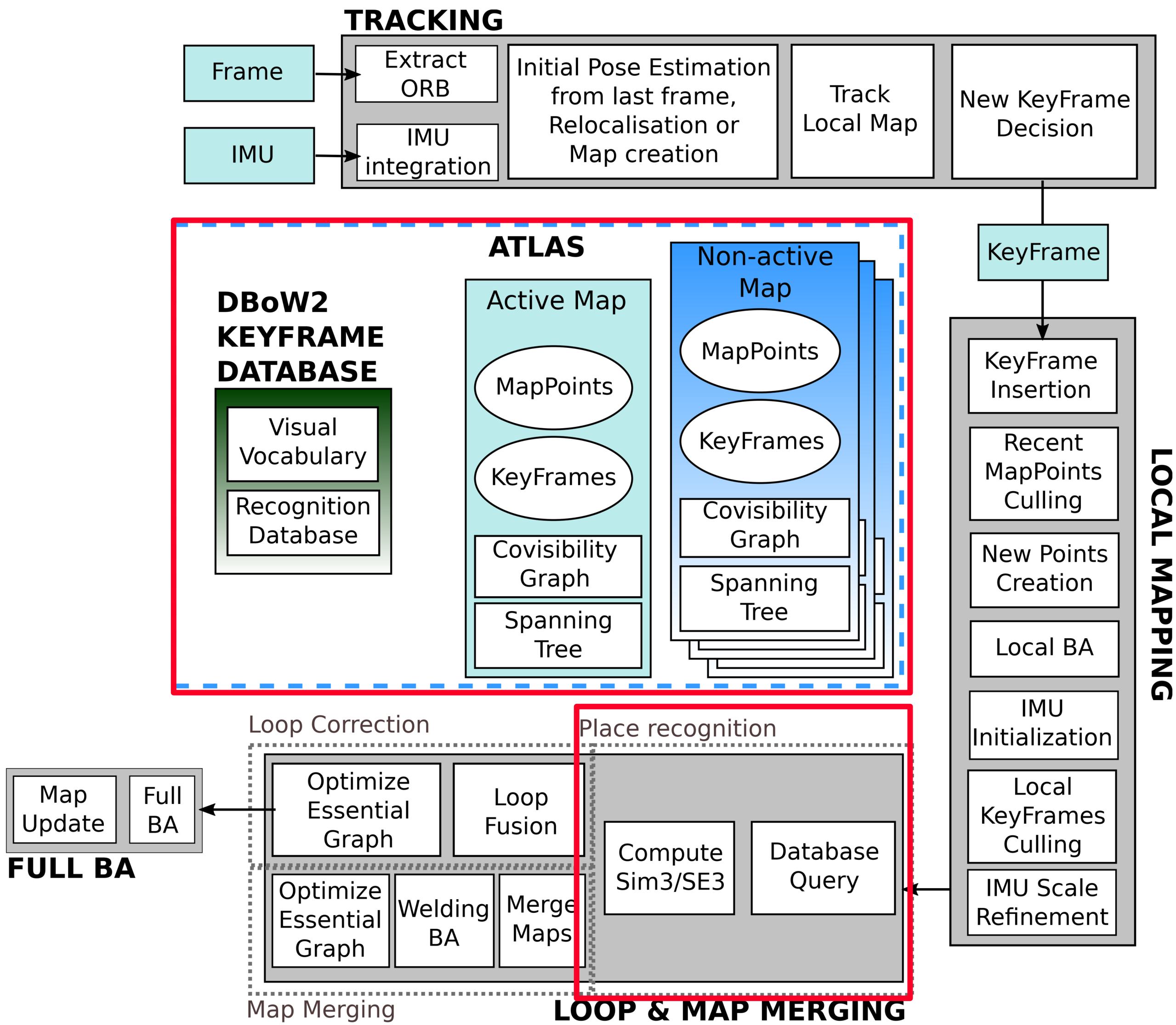
6.2.b Map Merge Detection: Place Recognition

1. Relocalization problem (tracking lost)
   - Frame vs KeyFrame (**2D-3D** matches)
   - Compute camera pose with **PnP** algorithm
2. Loop Closing problem (correct the accumulated drift)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - Align KeyFrames with **Sim3** (scale, rotation and position)
   - Align KeyFrames with **SE3** (rotation and position)
3. Map Merging problem (merge independent maps)
   - KeyFrame vs KeyFrame (**3D-3D** matches)
   - Align KeyFrames with **Sim3** (scale, rotation and position)
   - Align KeyFrames with **SE3** (rotation and position)

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

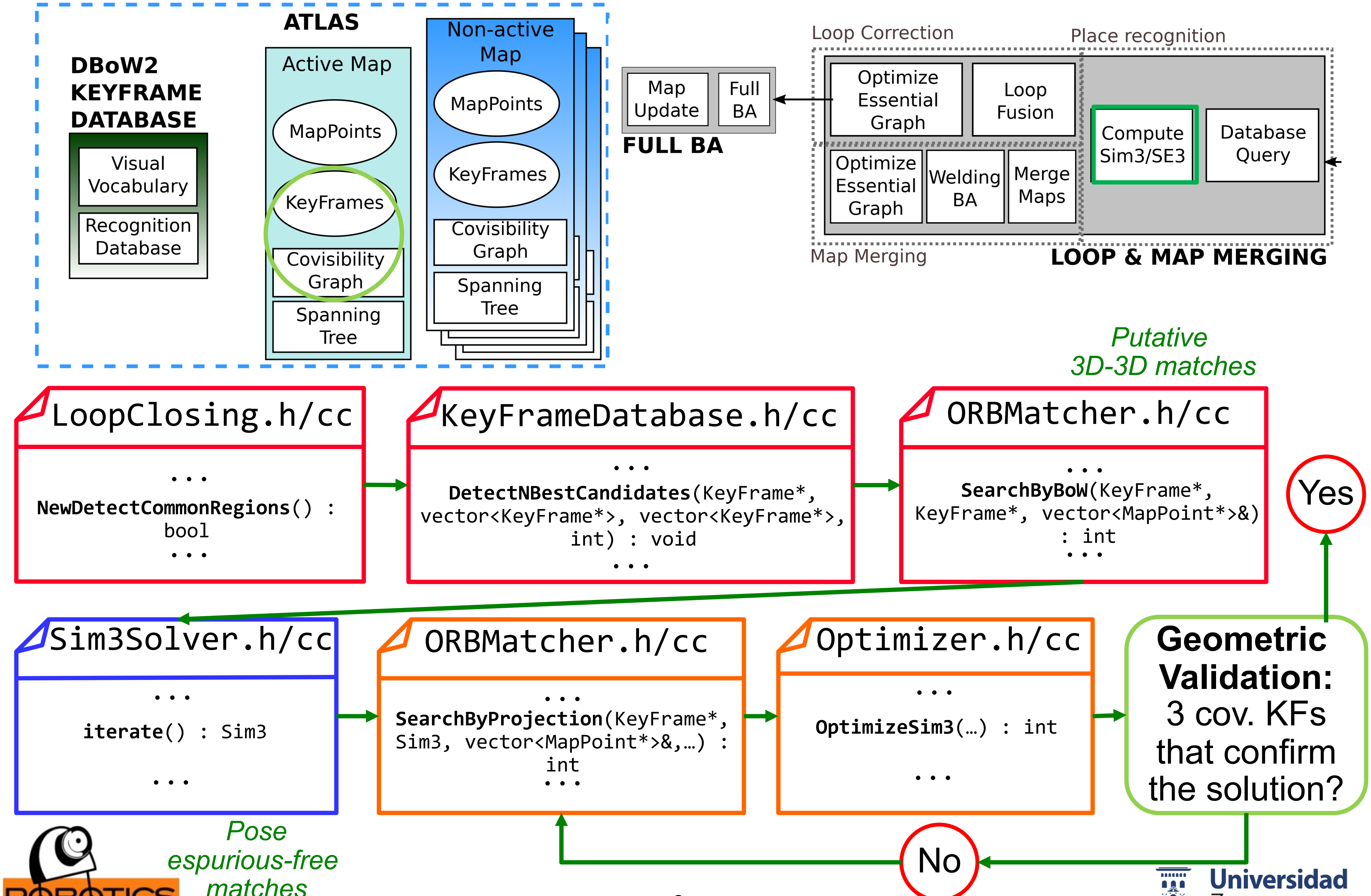
ORB-SLAM3: Loop & Merging Detection



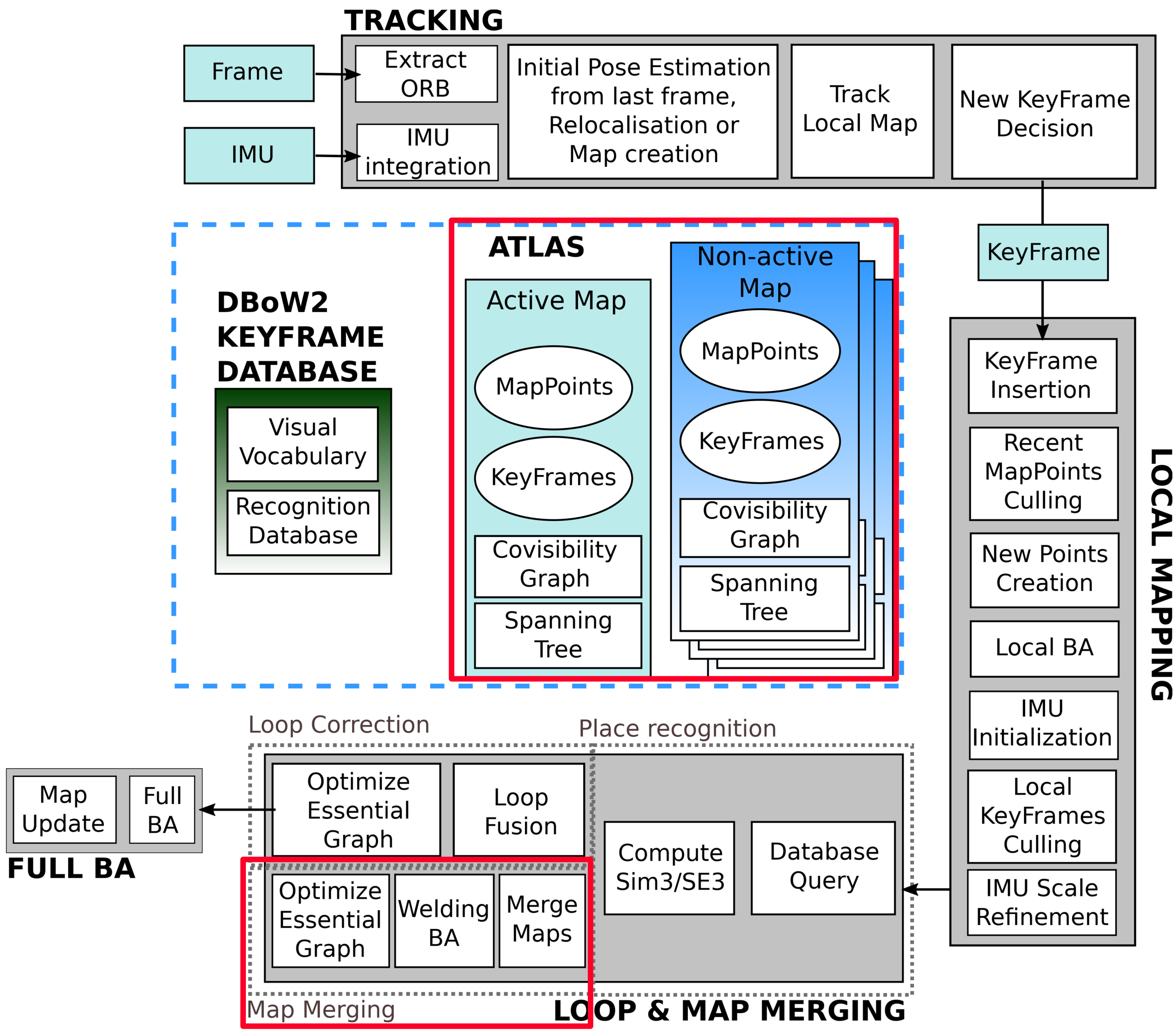
ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza

ORB-SLAM3: Loop & Merging Detection



6.2.c Map Merging in ORB-SLAM3

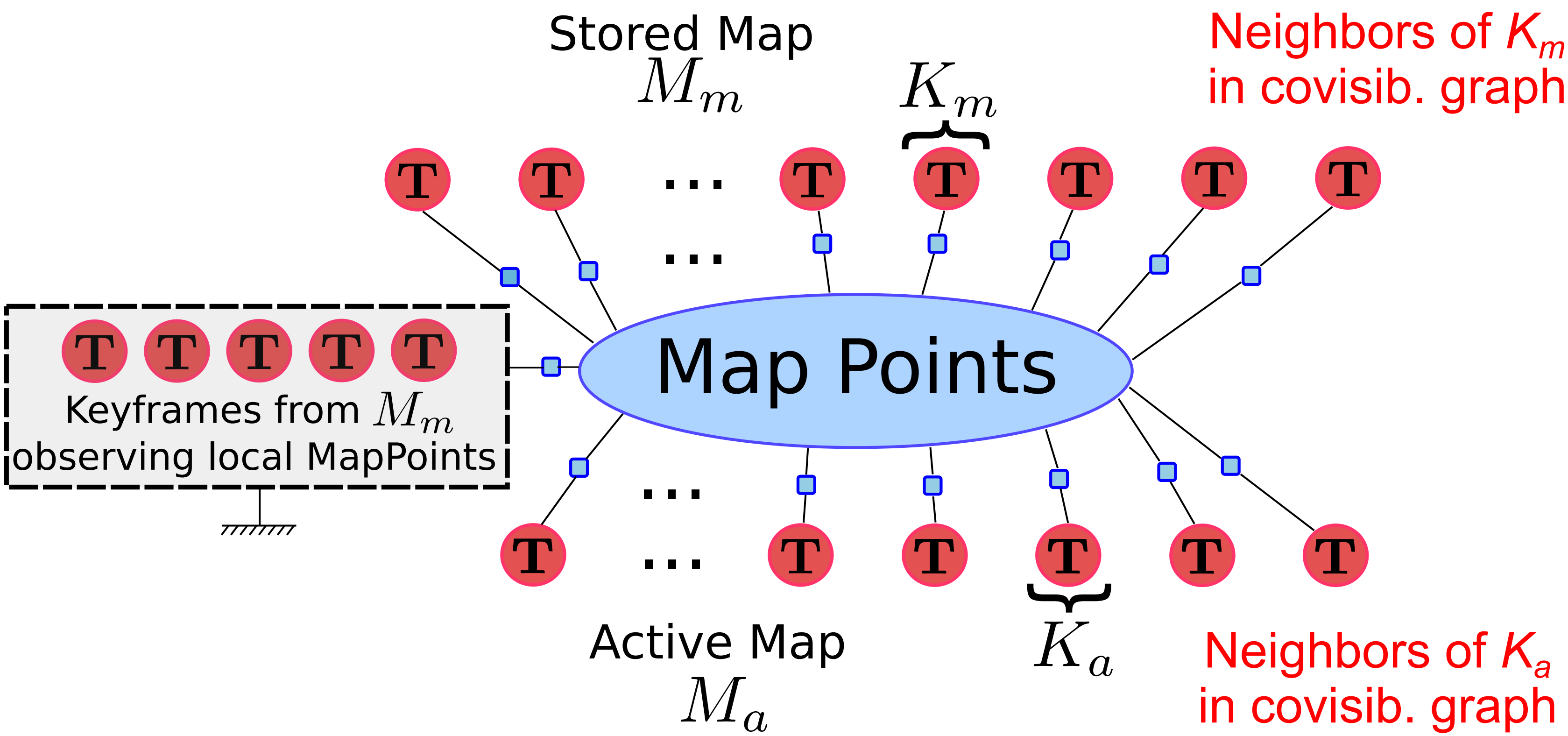


Map Merging in ORB-SLAM3

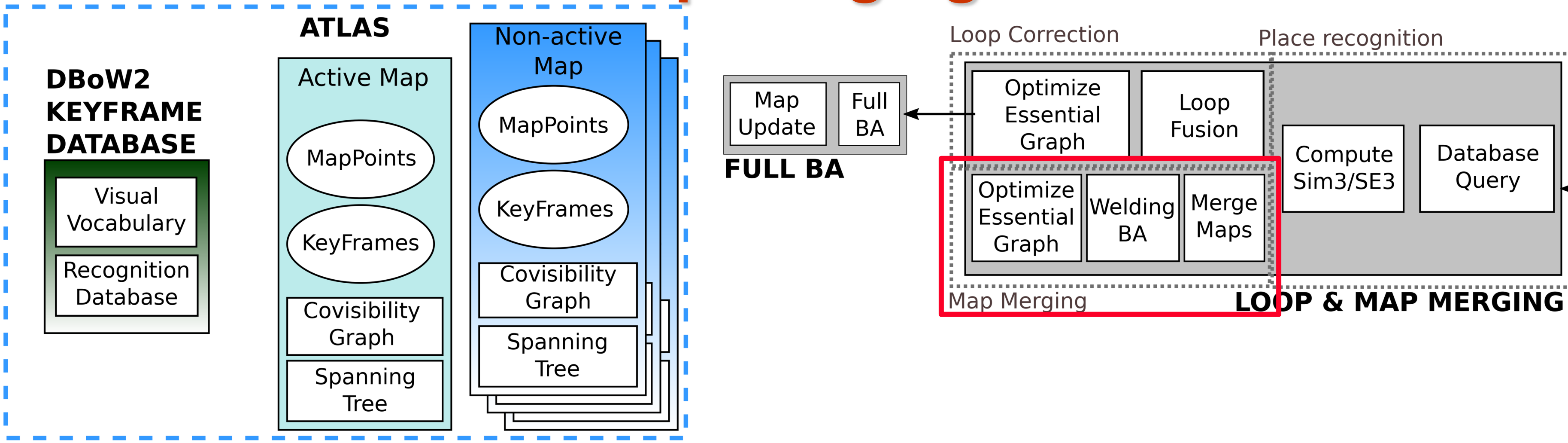
Place recognition has matched K_a in the active map M_a, with K_m in a stored map M_m, and has computed T_{ma}

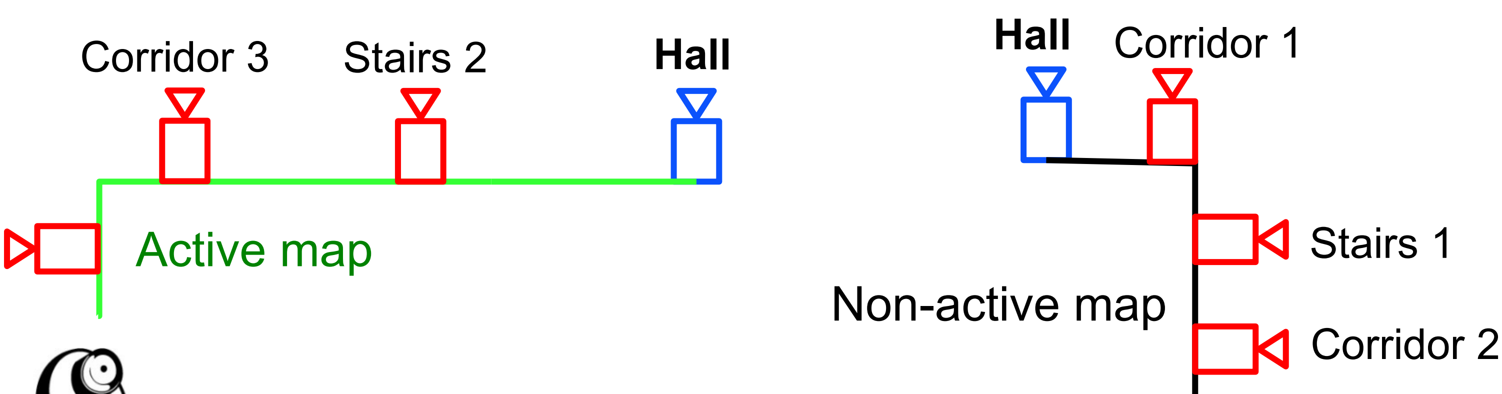
1. Transform K_a, its points and its neighbors to M_m
2. Fuse duplicated points, covisibility and essential graphs
3. Perform a Welding BA in the local area of both maps
4. The stored map becomes the active map, and starts being used for tracking
5. Transform the rest of M_a to M_m and perform pose-graph optimization in the essential graph of the fused map
6. Launch full BA in a separated thread

Welding Bundle Adjustment

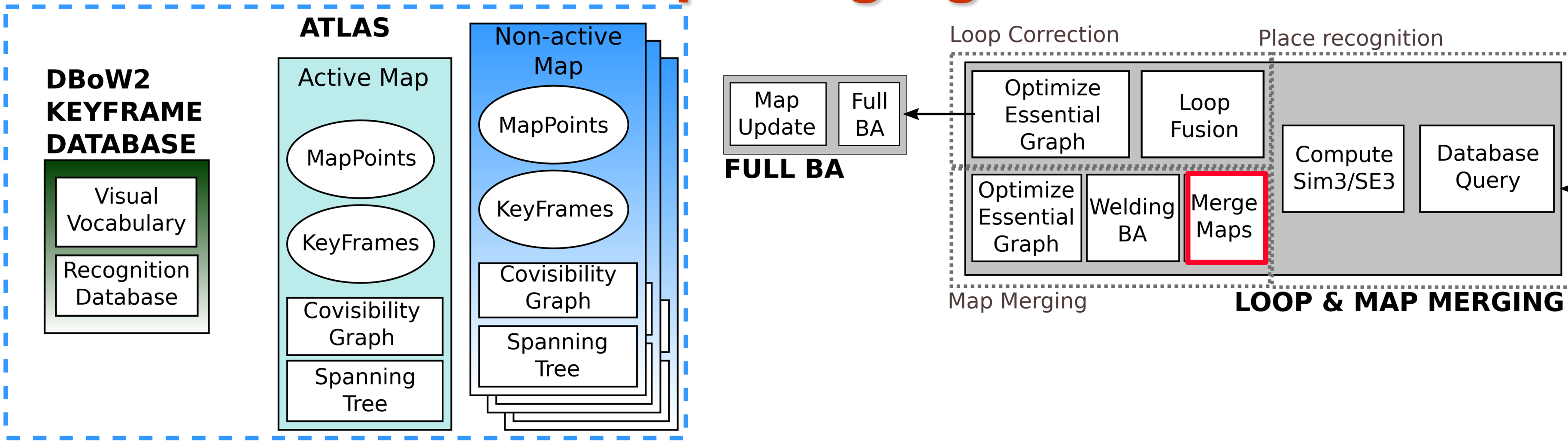


Map Merging


ATLAS
DBoW2 KEYFRAME DATABASE
Visual Vocabulary
Recognition Database
Active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Non-active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Map Update
Full BA
FULL BA
Loop Correction
Place recognition
Optimize Essential Graph
Loop Fusion
Compute Sim3/SE3
Database Query
Optimize Essential Graph
Welding BA
Merge Maps
Map Merging
LOOP & MAP MERGING


Corridor 3
Stairs 2
Hall
Active map
Hall
Corridor 1
Stairs 1
Non-active map
Corridor 2

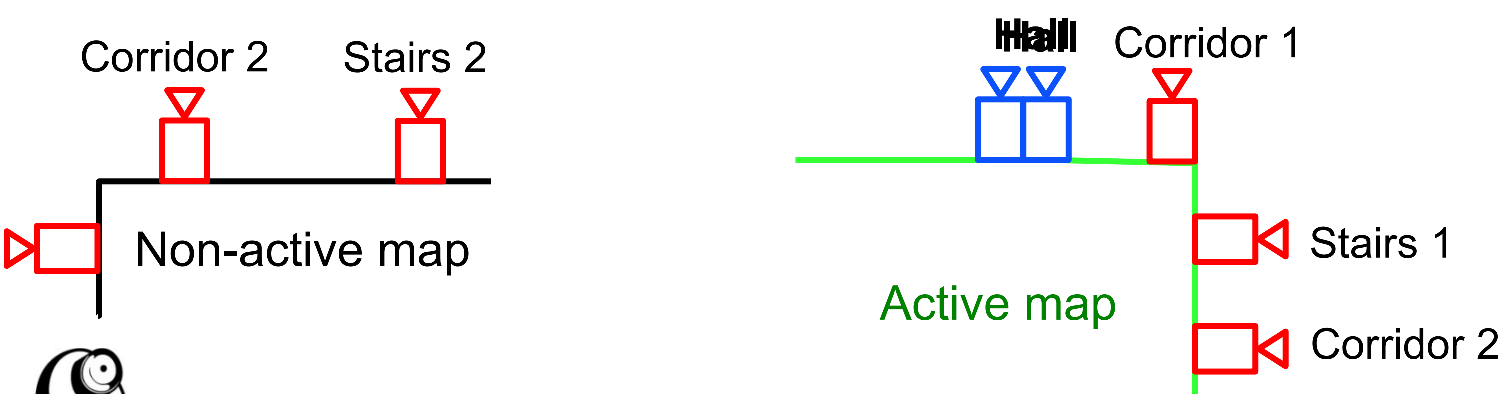
Map Merging



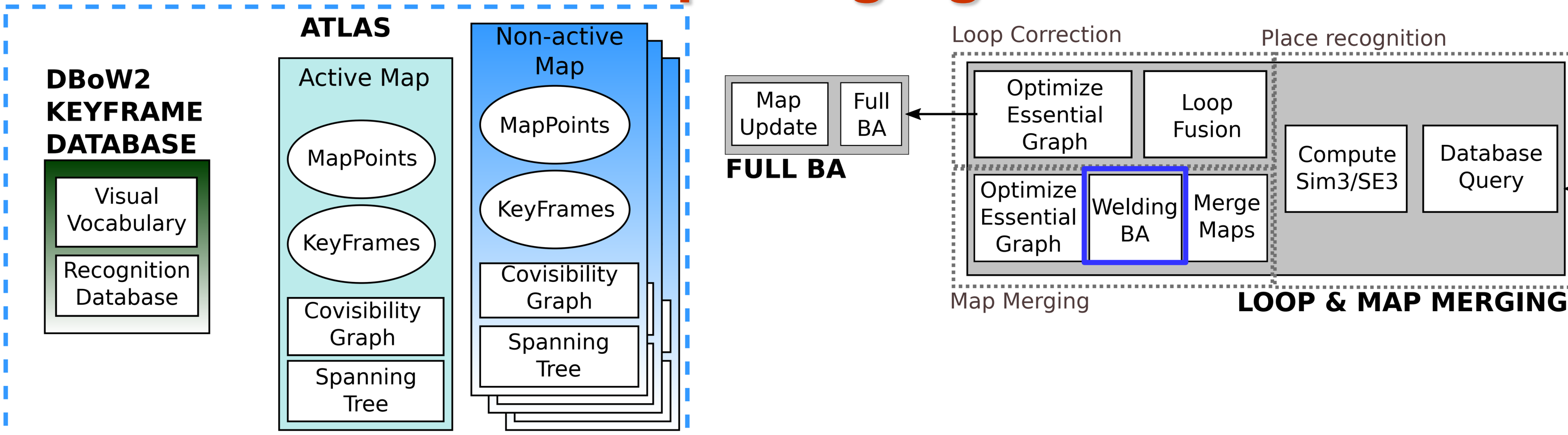
LoopClosing.h/cc

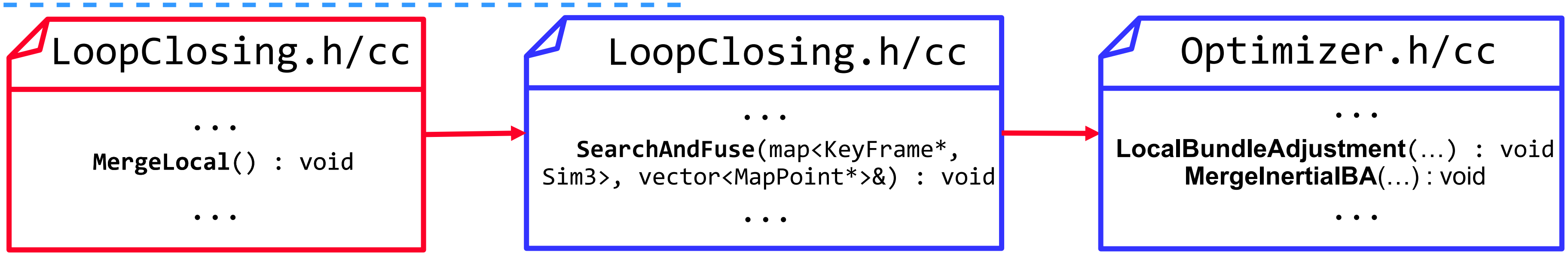
...

MergeLocal() : void



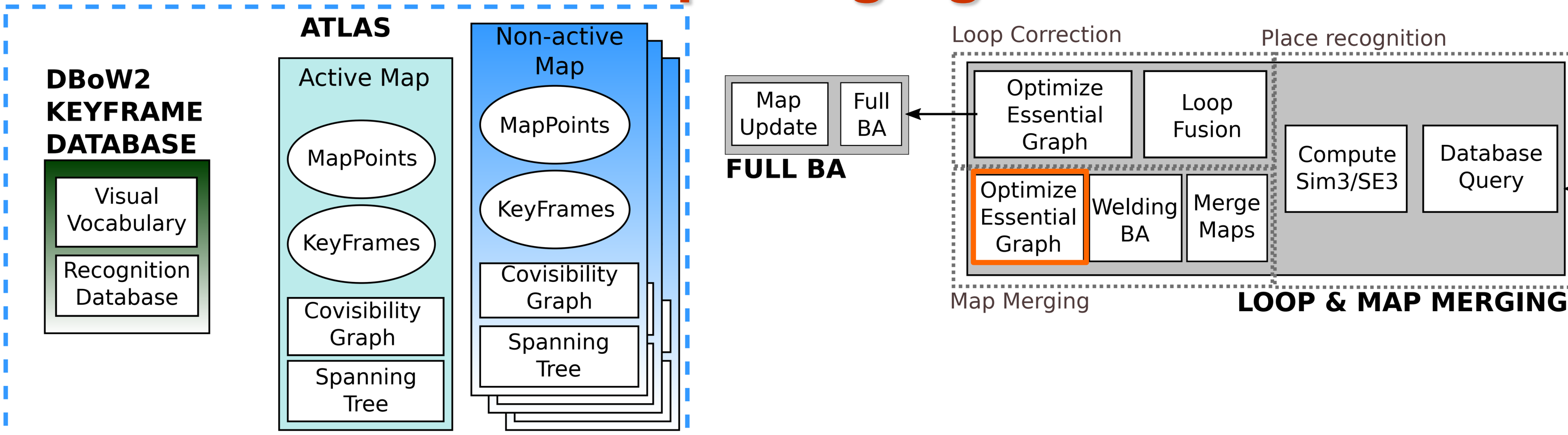
Map Merging


ATLAS
DBoW2 KEYFRAME DATABASE
Visual Vocabulary
Recognition Database
Active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Non-active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Map Update
Full BA
FULL BA
Loop Correction
Place recognition
Optimize Essential Graph
Loop Fusion
Compute Sim3/SE3
Database Query
Optimize Essential Graph
Welding BA
Merge Maps
Map Merging
LOOP & MAP MERGING

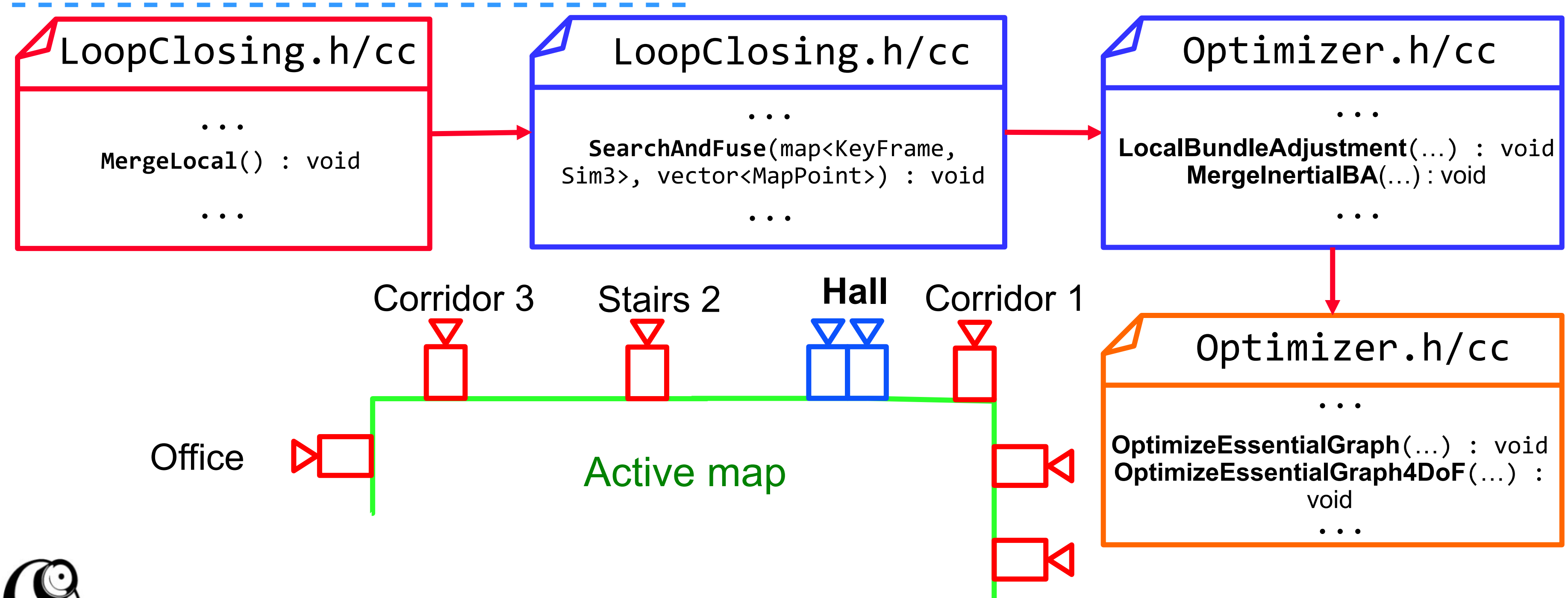

LoopClosing.h/cc
...
MergeLocal() : void
...
LoopClosing.h/cc
...
SearchAndFuse(map<KeyFrame*, Sim3>, vector<MapPoint*>&) : void
...
Optimizer.h/cc
...
LocalBundleAdjustment(...) : void
MergeInertialBA(...) : void
...


Corridor 3
Stairs 2
Non-active map
Hall
Corridor 1
Stairs 1
Active map
Corridor 2

Map Merging



ATLAS
DBoW2 KEYFRAME DATABASE
Visual Vocabulary
Recognition Database
Active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Non-active Map
MapPoints
KeyFrames
Covisibility Graph
Spanning Tree
Map Update
Full BA
FULL BA
Loop Correction
Place recognition
Optimize Essential Graph
Loop Fusion
Compute Sim3/SE3
Database Query
Optimize Essential Graph
Welding BA
Merge Maps
Map Merging
LOOP & MAP MERGING



LoopClosing.h/cc
...
MergeLocal() : void
...
LoopClosing.h/cc
...
SearchAndFuse(map<KeyFrame, Sim3>, vector<MapPoint>) : void
...
Optimizer.h/cc
...
LocalBundleAdjustment(...) : void
MergeInertialBA(...) : void
...
Optimizer.h/cc
...
OptimizeEssentialGraph(...) : void
OptimizeEssentialGraph4DoF(...) : void
...
Corridor 3
Stairs 2
Hall
Corridor 1
Office
Active map

6.2.d Multi-Map Results

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Universidad Zaragoza 1542

ORB-SLAM3: Map Merging Timing

	Sensor	Monocular	Stereo	Mono-Inertial	Stereo-Inertial
Settings	Resolution	752×480	752×480	752×480	752×480
	Cam. FPS	20Hz	20Hz	20Hz	20Hz
	IMU	-	-	200Hz	200HZ
	ORB Feat.	1000	1200	1000	1200
	RMS ATE	0.284	0.163	0.048	0.046
Place Recognition	Database query	0.99±0.65	1.01±0.60	0.90±0.49	0.88±0.56
	Compute Sim3/SE3	5.94±9.47	9.16±11.20	3.45±3.41	8.00±8.02
	Total	6.83±9.65	10.00±11.41	4.29±3.59	8.81±8.19
Map Merging	Merge Maps	135.04±38.18	143.27±10.84	231.33±28.04	425.58±57.65
	Welding BA	118.86±30.93	98.50±9.33	258.70±21.15	181.43±30.34
	Opt. Essential Graph	86.89±126.62	19.03±19.72	37.47±38.48	37.51±39.68
	Total	349.15±164.31	281.29±39.32	555.16±10.16	669.93±51.84
Merge info	# Detected merges	4	3	2	2
	Merge size (# keyframes)	28±2	31±3	36±1	38±1
	Merge size (# map points)	1840±405	2280±507	2955±344	5467±692
Loop	Loop Fusion	90.47±32.59	71.80±31.77	-	61.38
	Opt. Essential Graph	152.52±114.22	176.39±30.61	-	331.20
	Total	268.35±90.46	253.37±30.35	-	394.84
Loop info	# Detected loops	4	4	0	1
	Loop size (# keyframes)	122±43	55±6	-	42
Loop Full BA	Full BA	2348.53±3339.16	1633.65±1363.88	-	1913.39
	Map Update	4.59±2.58	9.30±1.65	-	10.55
	Total	2416.63±3395.28	1702.57±1347.56	-	1978.65
	BA size (# keyframes)	170±68	186±82	-	143
	BA size (# map points)	6186±3092	10342±3256	-	12245

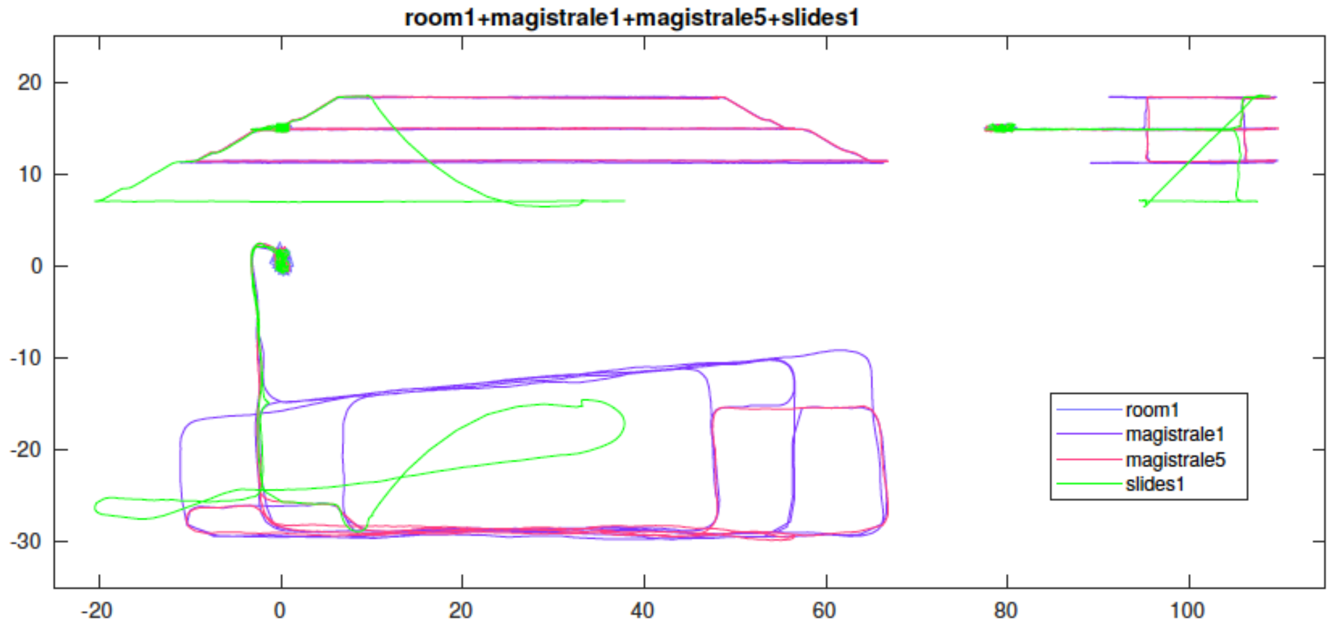
Visual-Inertial and Multi-Map Example

ORB-SLAM3: An Accurate Open-Source Library for Visual, Visual-Inertial and Multi-Map SLAM

Carlos Campos*, Richard Elvira*, Juan J. Gómez Rodríguez,
José M. M. Montiel and Juan D. Tardós

Dataset: TUM VI Benchmark
Sequences: Room1+Magistrale1+Magistrale5+Slides1
Setup: Stereo-Inertial

Visual-Inertial and Multi-Map Example



Multi-Map Summary

- Improves robustness to tracking failures
  - When lost, start a new map
  - Fuse with the previous map when revisiting it
- Allows multi-session mapping
  - Effective reuse of previous maps (the goal of SLAM!)
- Maps are fused seam-less
  - Similar results if mapping in one or several sessions
- Open issues
  - Map maintenance (long-term mapping)

Summary

- Visual SLAM is better than Visual Odometry
  - Mid-term DA: Zero drift in mapped areas
  - Long-term DA: Relocation, Loop Closure and Multi-Mapping
- ORB-SLAM3 is the most complete SLAM system
  - Mono, stereo, mono-inertial and stereo-inertial
  - Pinhole and fisheye camera models
  - Multi-map and Multi-session
- Visual SLAM accuracy
  - Mono: robustness issues and scale unknown
  - Stereo: good robustness and accuracy
  - Mono-inertial: excellent robustness and accuracy
  - Stereo-inertial: excellent+ robustness and accuracy