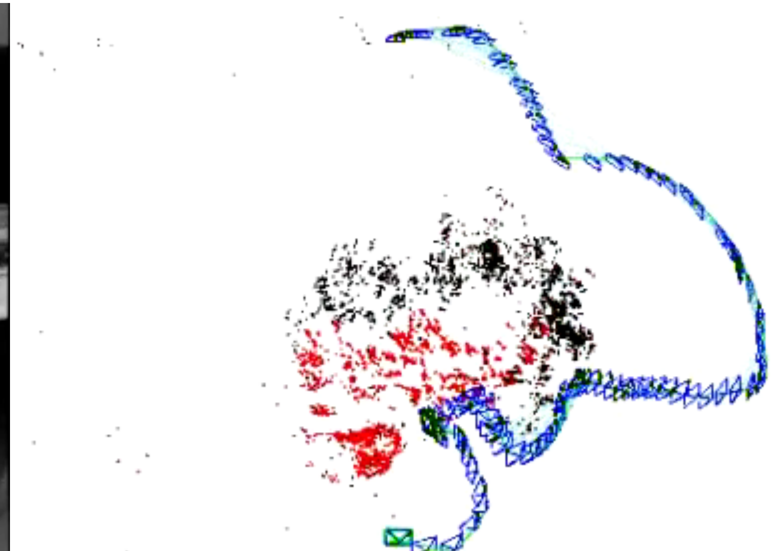
Master y Programa de Doctorado en “Ingeniería de Sistemas y de Control”
UNED, 11-Abril-2019

SLAM Visual Basado en Características

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



SLAMLAB: Members


Juan D. Tardós


José Neira


José A. Castellanos


José Mª M. Montiel


Javier Civera


Nader Mahmoud
Ing. Informática


Chema Fácil
Ing. Informática

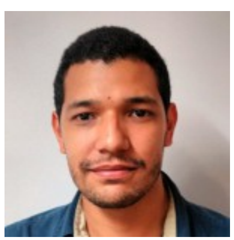

María L. Rodríguez
Matemáticas


José Lamarca
Ing. Industrial


Berta Bescós
Ing. Industrial


Javier Domínguez
Ing. Industrial


Juan José Gómez
Ing. Informática


Leonardo Fermír
Ing. Electrónica

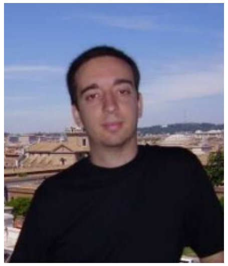

María Isabel Artigas
Ing. Mecánica


Richard Elvira
Ing. Informática


Carlos Campos
Ing. Industrial


Blanca Guillén
Ing. Industrial

Recent PhD Students


Pedro Piniés
Apple


Lina Mª Paz
Apple


Dorian Gálvez
intel


Marta Salas
accenture


Raúl Mur-Artal
oculus


Alejo Concha
Google


Yasir Latif
THE UNIVERSITY of ADELAIDE


César Cadena
ETH Zürich
Autonomous Systems Lab


Henry Carrillo
UNIVERSIDAD SERGIO ARBOLEDA


Antonio Agudo
Institut de Robòtica i Informàtica Industrial


Maite Lázaro
SAPIENZA UNIVERSITÀ DI ROMA

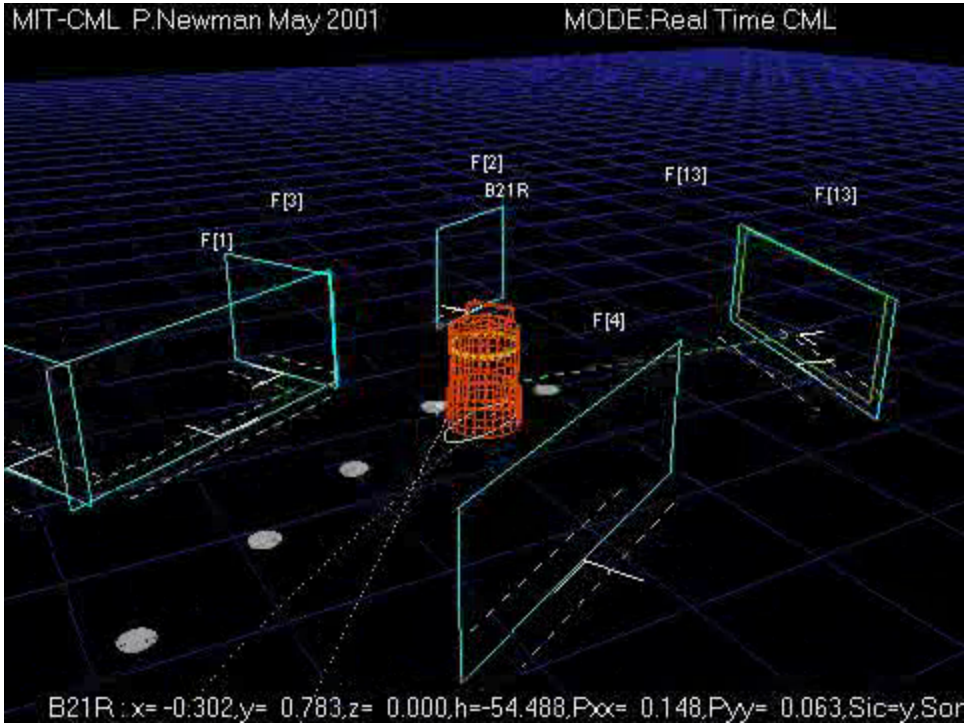
SLAM: Simultaneous Localization and Mapping

The SLAM problem:

- a robot moving in an unknown environment

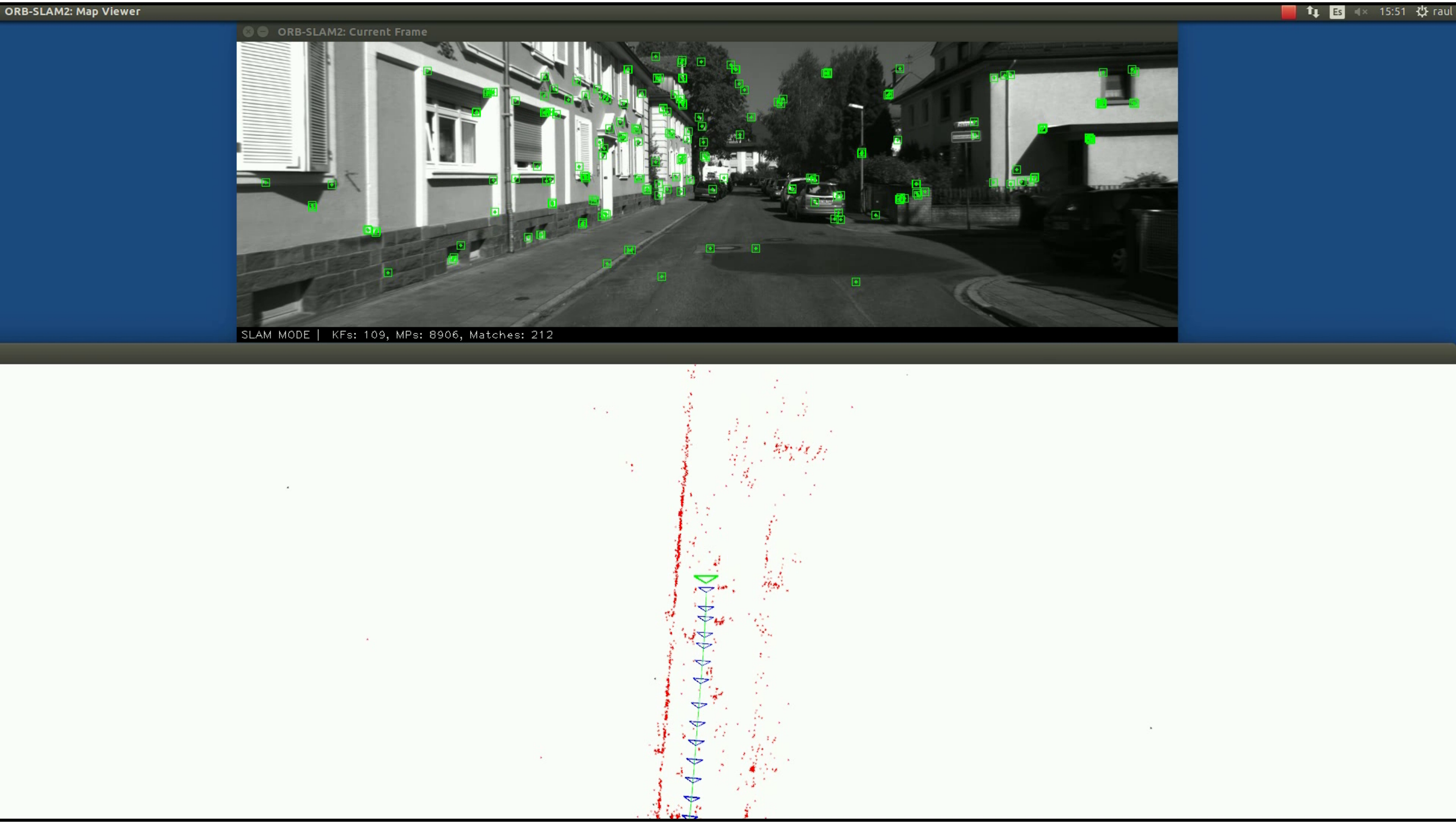
Use sensor data to:

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



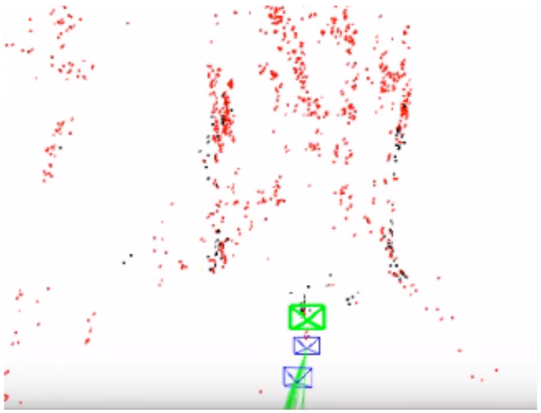
P. Newman, J.J Leonard, J.D. Tardos, J. Neira: **Explore and return: Experimental validation of real-time concurrent mapping and localization.** IEEE Int. Conf. Robotics and Automation, 2002

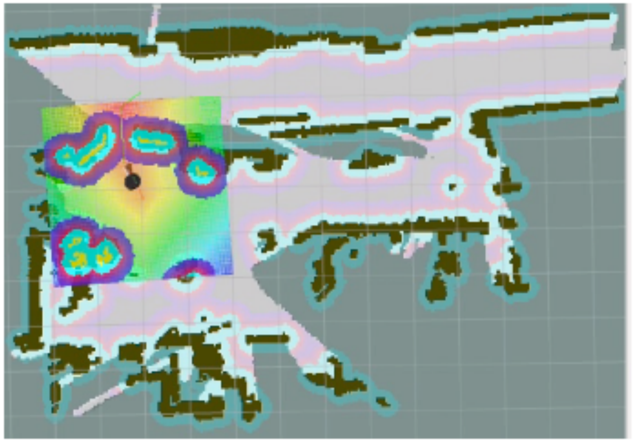
ORB-SLAM: Visual SLAM, 2015



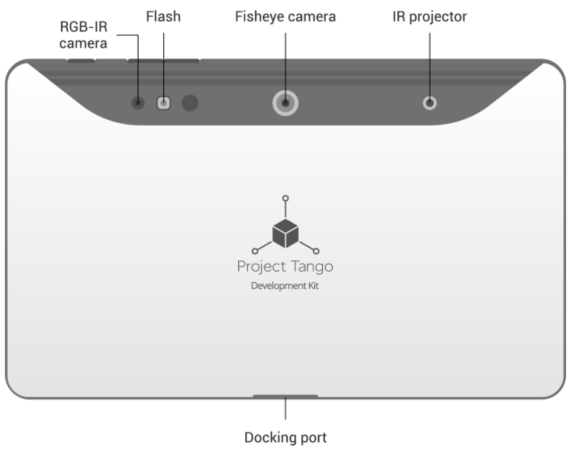
Applications: Robotics and 3D Modelling

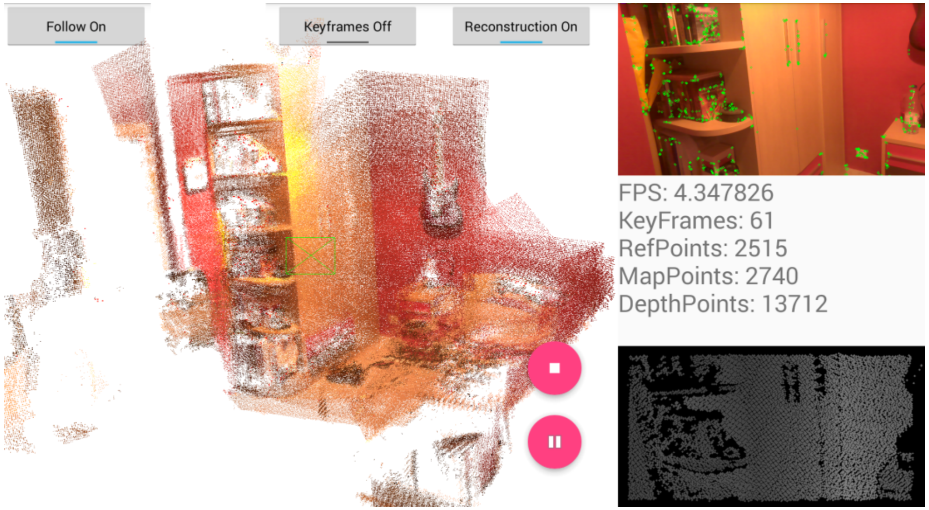
- Robot Navigation based on ORB-SLAM2





- ORB-SLAM2 on mobile devices

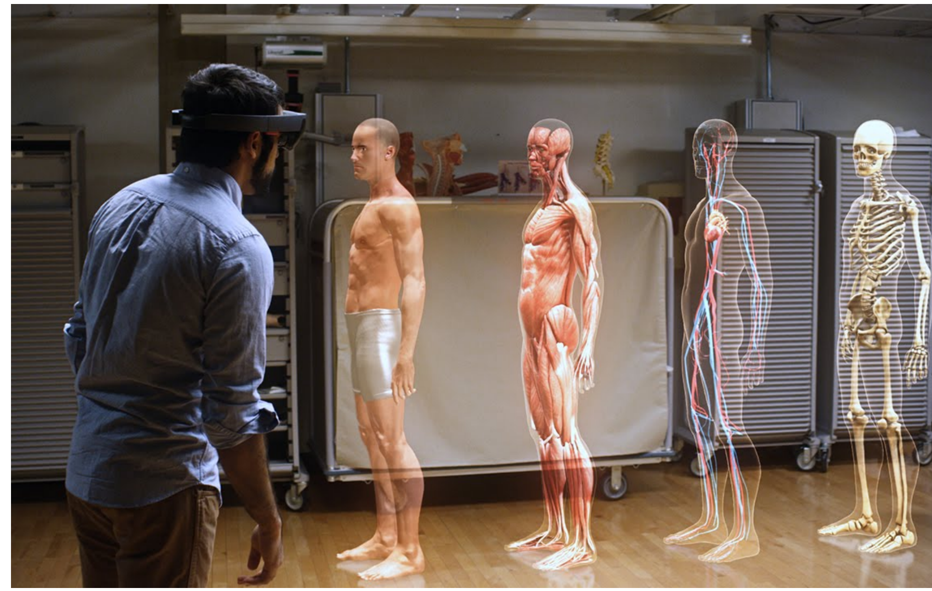




Applications: AR/VR

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

Applications: AR/VR



Oculus Rift
Facebook

Gear VR
Samsung

Meta 2
Metavision

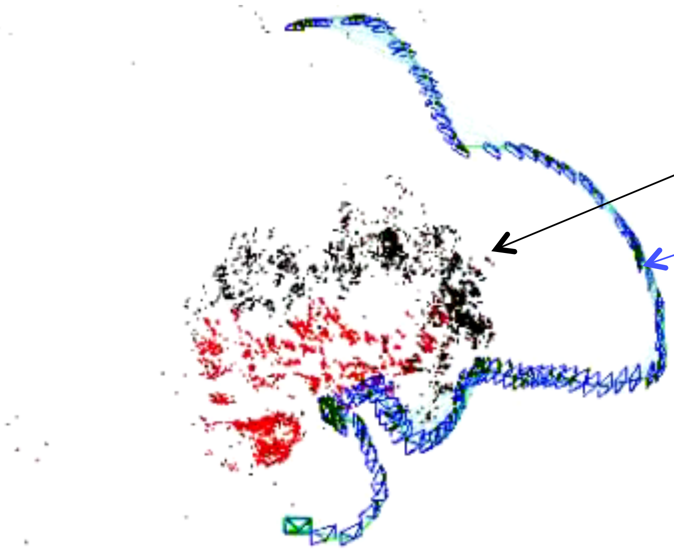
Hololens
Microsoft

- SLAM: User positional tracking

Outline

1. Feature-Based Visual SLAM
2. Features
3. Feature Matching
4. Relocation and Loop Closing
5. Putting all together: ORB-SLAM
6. ORB-SLAM2: Stereo and RGB-D
7. Visual-Inertial ORB-SLAM

1. Feature-Based Visual SLAM



States

$\mathbf{x}_{wj} \in \mathbb{R}^3$ 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

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

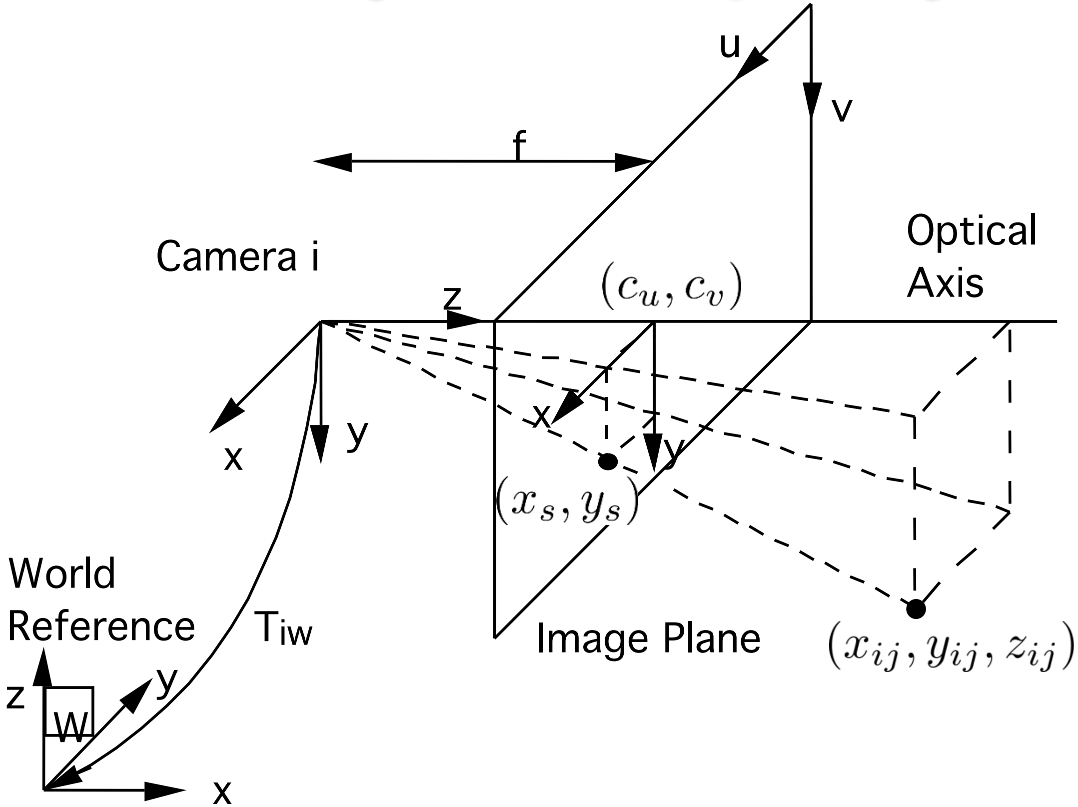
Projection of point j on camera i (1)

$$\mathbf{T}_{iw} \in \mathrm{SE}(3) \begin{cases} \mathbf{R}_{iw} \in \mathrm{SO}(3) & \text{Rotation matrix} \\ \mathbf{t}_{iw} \in \mathbb{R}^3 & \text{Translation vector} \end{cases}$$

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

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

Projection of point j on camera i (2)



focal length (mm)

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

$$\begin{aligned} u &= (s_u f_i) \frac{x_{ij}}{z_{ij}} + c_{i,u} \\ &= f_{i,u} \frac{x_{ij}}{z_{ij}} + c_{i,u} \end{aligned}$$

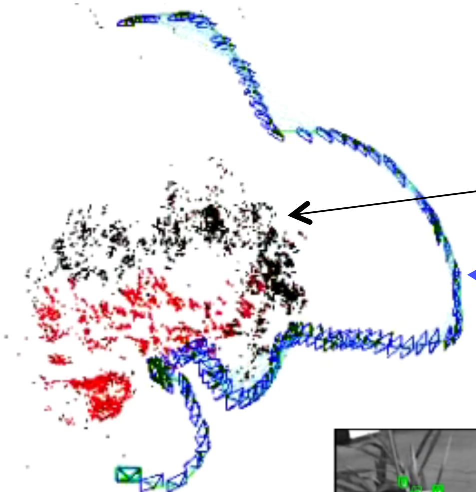
horizontal focal length (pixels)

principal point

- In summary:

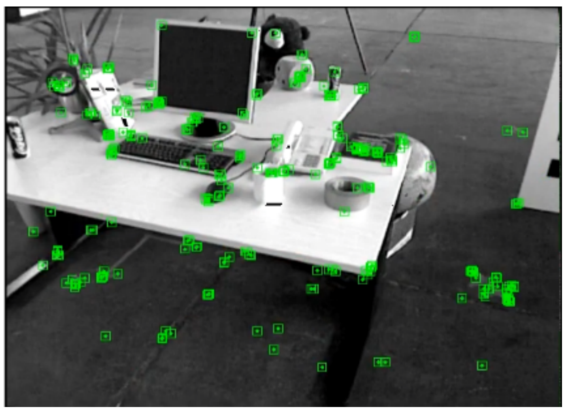
$$\pi_i(\mathbf{T}_{iw}, \mathbf{x}_{wj}) = \begin{bmatrix} f_{i,u} \frac{x_{ij}}{z_{ij}} + c_{i,u} \\ f_{i,v} \frac{y_{ij}}{z_{ij}} + c_{i,v} \end{bmatrix}$$

Feature-Based Visual SLAM



States

$\mathbf{x}_w^j \in \mathbb{R}^3$	Coordinates of point j
$\mathbf{R}_{iw} \in \mathrm{SO}(3)$	Orientation of camera i
$\mathbf{p}_{iw} \in \mathbb{R}^3$	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

$$\{\mathbf{R}_{iw}, \mathbf{p}_{iw}, \mathbf{x}_w^j \,|\, \forall i, \forall j\}^* = \underset{\mathbf{R},\mathbf{p},\mathbf{x}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\underbrace{\left\| \mathbf{u}_{ij} - \pi \left(\mathbf{R}_{iw} \mathbf{x}_w^j + \mathbf{p}_{iw} \right) \right\|}_{\text{Reprojection error}}{}^2_{\mathbf{\Sigma}_{ij}} \right)$$

Some details

$$\{\mathbf{R}_{iw}, \mathbf{p}_{iw}, \mathbf{x}_w^j | \forall i, \forall j\}^* = \underset{\mathbf{R},\mathbf{p},\mathbf{x}}{\operatorname{argmin}} \sum_{i,j} \rho \left(\left\| \mathbf{u}_{ij} - \pi \left(\mathbf{R}_{iw} \mathbf{x}_w^j + \mathbf{p}_{iw} \right) \right\|^2_{\mathbf{\Sigma}_{ij}} \right)$$

- Assumption: the camera has been calibrated
  - Focal lengths and principal point are known
  - Distortion can be corrected

- $\rho_h(\ )$ 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

Huber cost function

- L2 cost (quadratic)

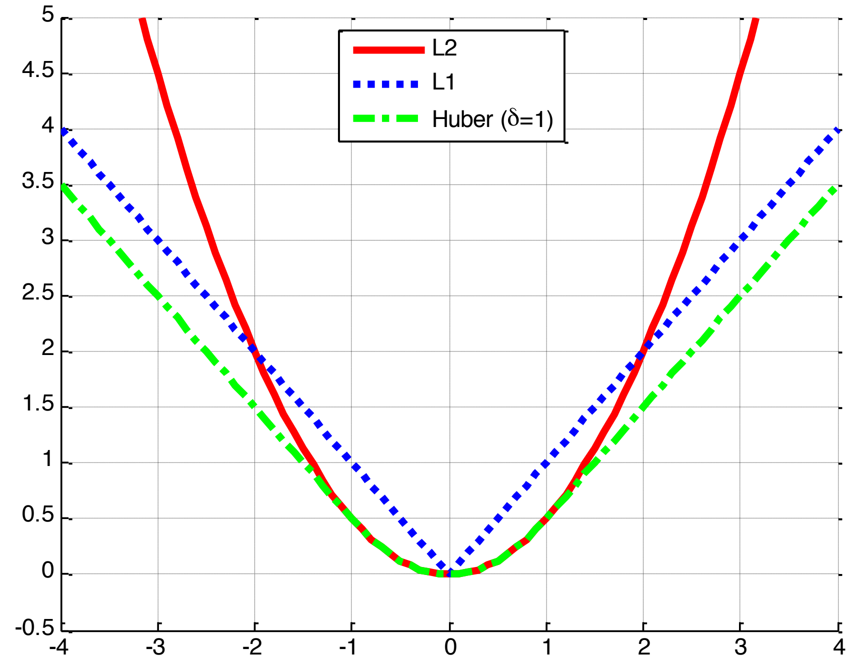
$$J_{L2}(\theta) = \frac{1}{2}\sum_{i=1}^{N}\left(h_\theta(\mathbf{x}^{(i)}) - y^{(i)}\right)^2$$

Differentiable ☺

- L1 cost (absolute value)

$$J_{L1}(\theta) = \sum_{i=1}^{N}\left|h_\theta(\mathbf{x}^{(i)}) - y^{(i)}\right|$$

Non differentiable ☹



$$r^{(i)} = h_\theta(\mathbf{x}^{(i)}) - y^{(i)}$$

- Huber cost:

$$L_H(r,\delta) = \begin{cases} r^2/2 & \text{if } |r| \le \delta \\ \delta|r| - \delta^2/2 & \text{if } |r| > \delta \end{cases}$$

Differentiable ☺

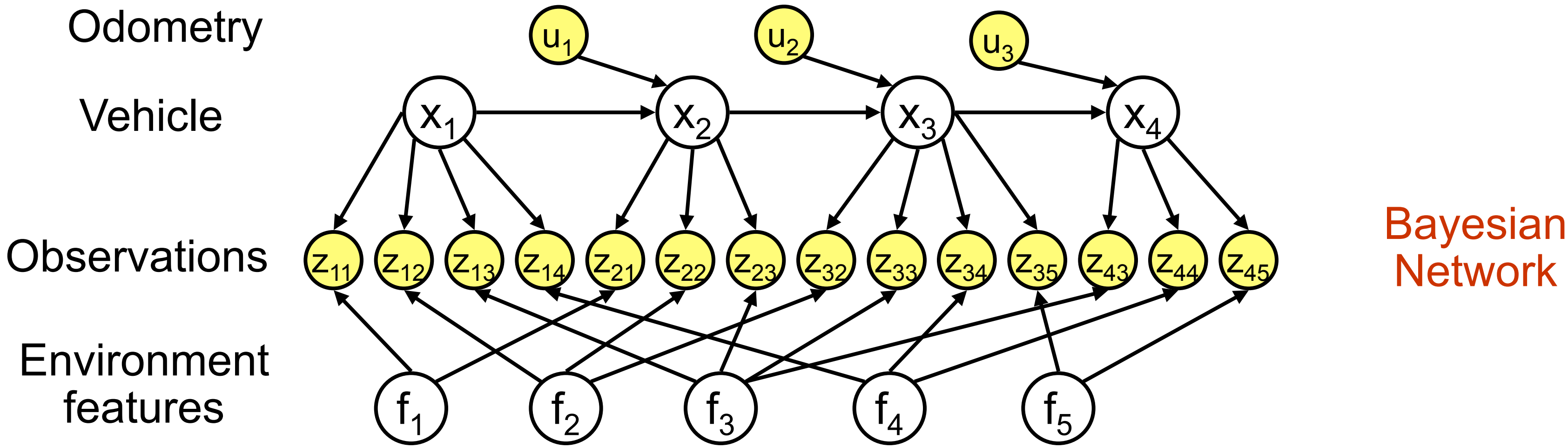
$$J_H(\theta) = \sum_{i=1}^{N} L_H(r^{(i)},\delta) = \sum_{\left|r^{(i)}\right| \le \delta} {r^{(i)}}^2/2 + \sum_{\left|r^{(i)}\right| > \delta} \delta\left|r^{(i)}\right| - \delta^2/2$$

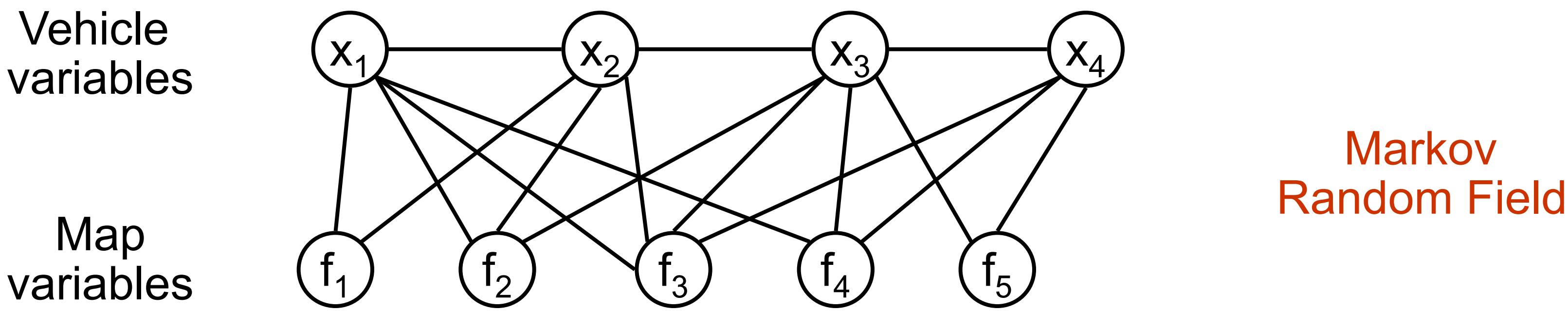
Full Bundle Adjustment in Real Time?

$$\{\mathbf{R}_{iw}, \mathbf{p}_{iw}, \mathbf{x}_w^j | \forall i, \forall j\}^* = \underset{\mathbf{R},\mathbf{p},\mathbf{x}}{\operatorname{argmin}} \sum_{i,j} \rho\left(\left\| \mathbf{u}_{ij} - \pi\left(\mathbf{R}_{iw}\mathbf{x}_w^j + \mathbf{p}_{iw}\right)\right\|^2_{\mathbf{\Sigma}_{ij}}\right)$$

- The problem is sparse
  - Not all cameras see all points!
- But still not feasible in real time
  - example: 1k images and 100k points → 1s per LM iteration
- Local BA or sliding-window BA
- BA requires very good initial solutions

Structure of the SLAM problem



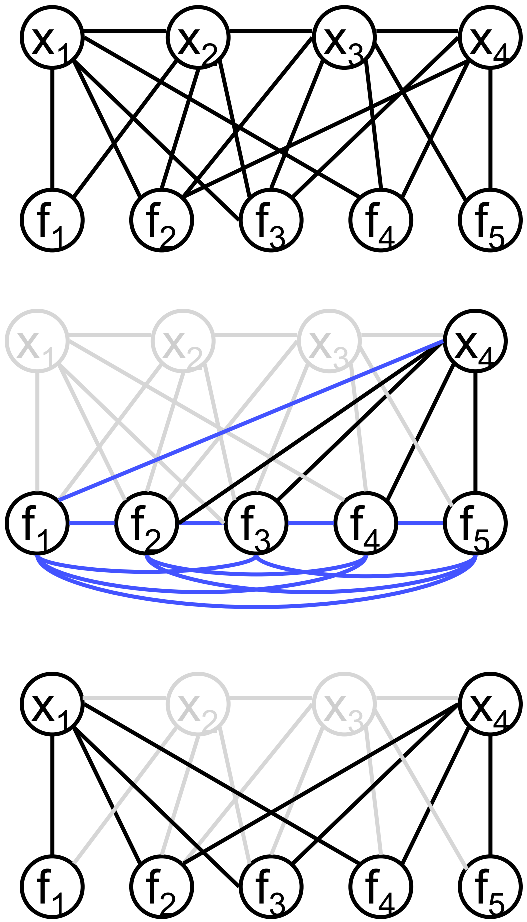


SLAM Problem

$$p(x_{1:k}\ ,\ f_{1:n}\ |\ z_{1:k}\ ,\ u_{1:k}\ )$$

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

Maps with Thousands of Features?



- Original SLAM problem
- EKF approach
  - Only keeps the last pose
  - $O(n^2)$ with the number of features
  - Limited to 200-300 features in real-time
- Keyframe approach (PTAM)
  - Uses only a few keyframes for map estimation with non-linear optimization
  - Can handle thousands of points
  - Given the same computational effort is more precise than EKF-SLAM

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

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?

2. Features

Local Features, Interest points, Keypoints

- Detector: find local maxima of a certain operator

original Image

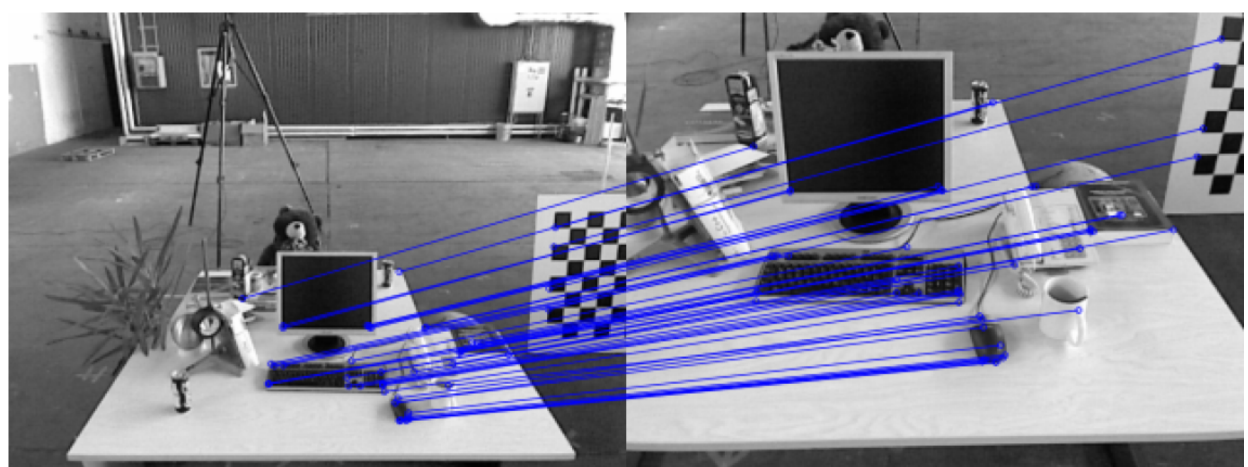
Harris detector
(corner-like)

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*

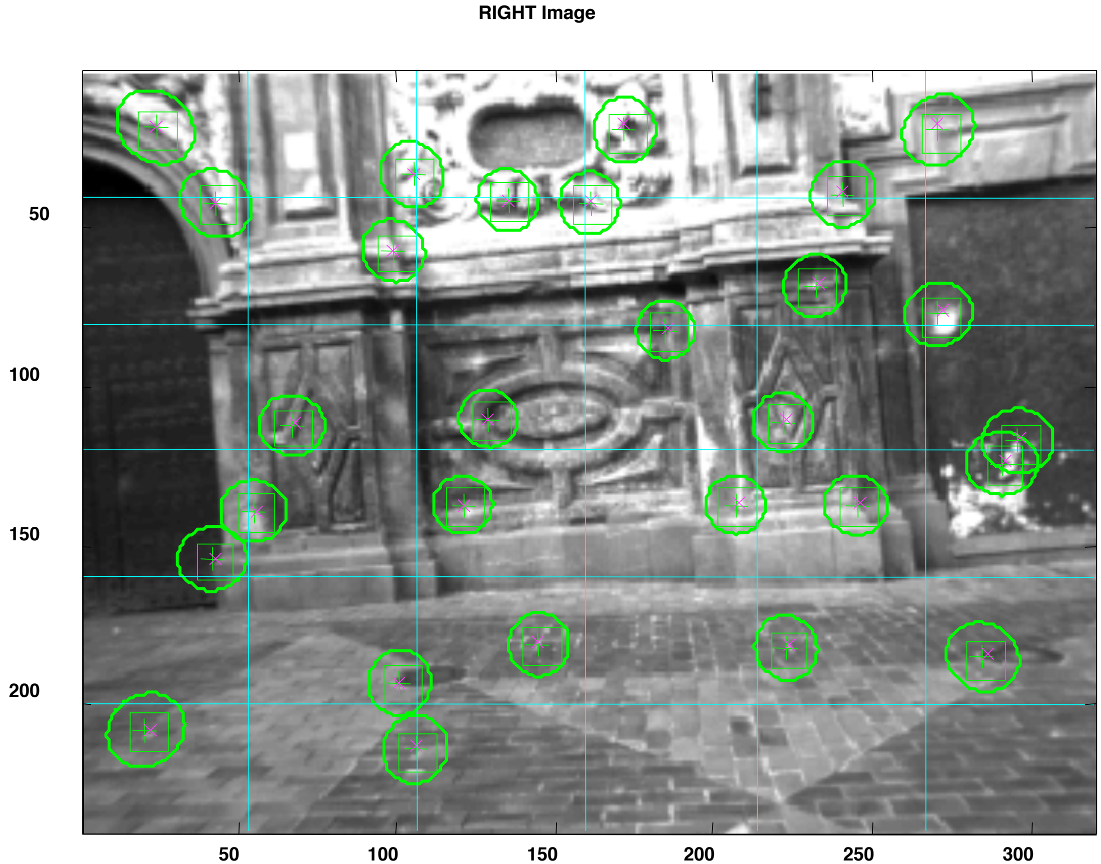
- Harris detector:

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

- Shi-Tomasi detector:

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

Good for Tracking using Correlation

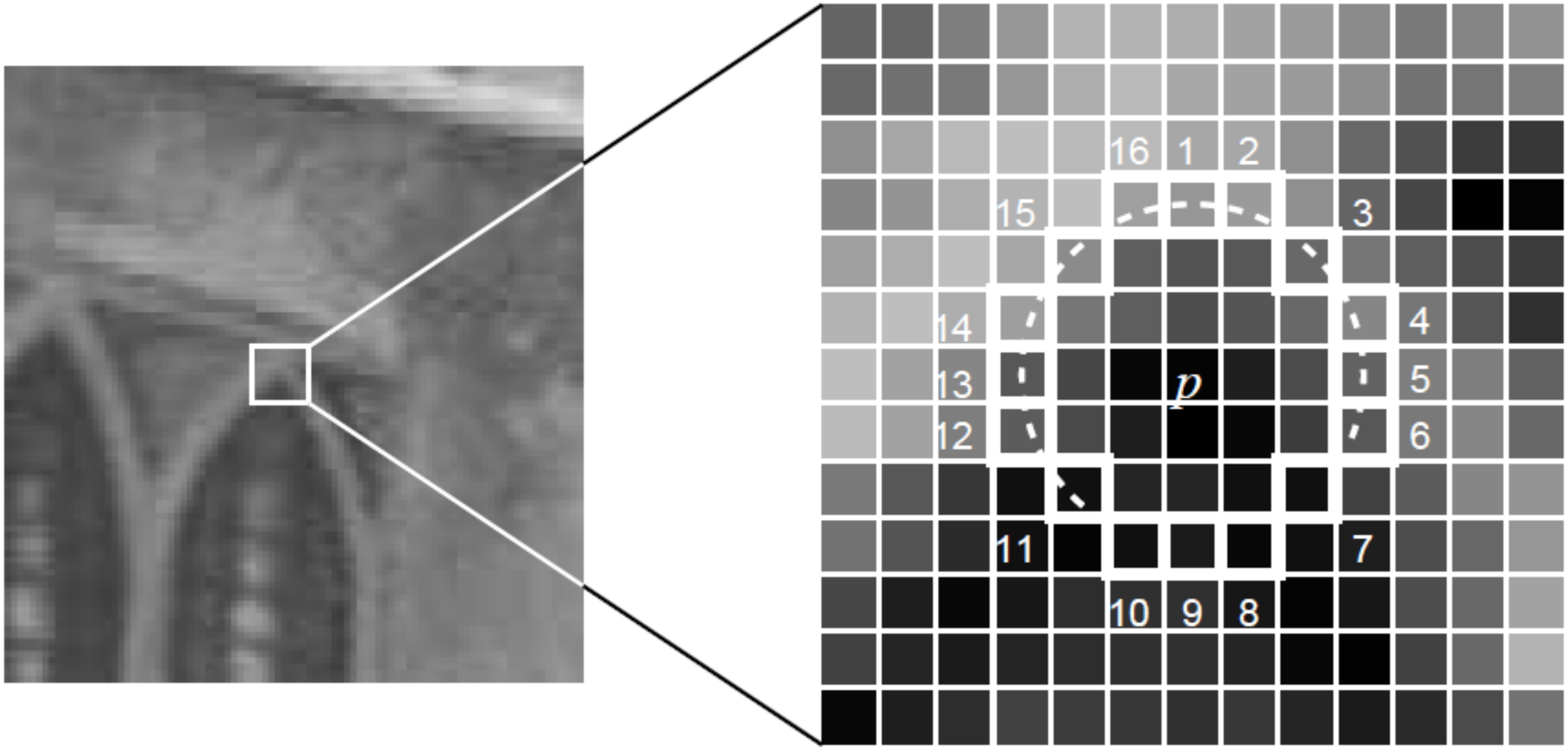


Shi-Tomasi points

Predict position in next image (@15-30 Hz)

Search by normalized correlation with a 11x11 patch

FAST corner detector

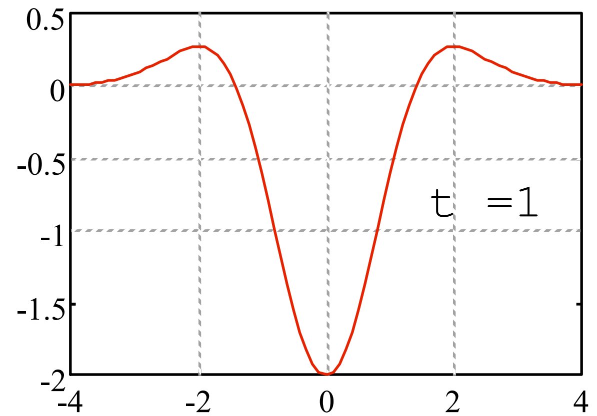


- Pixel p surrounded by n consecutive pixels all brighter (or darker) than p
- Much faster than other detectors

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:

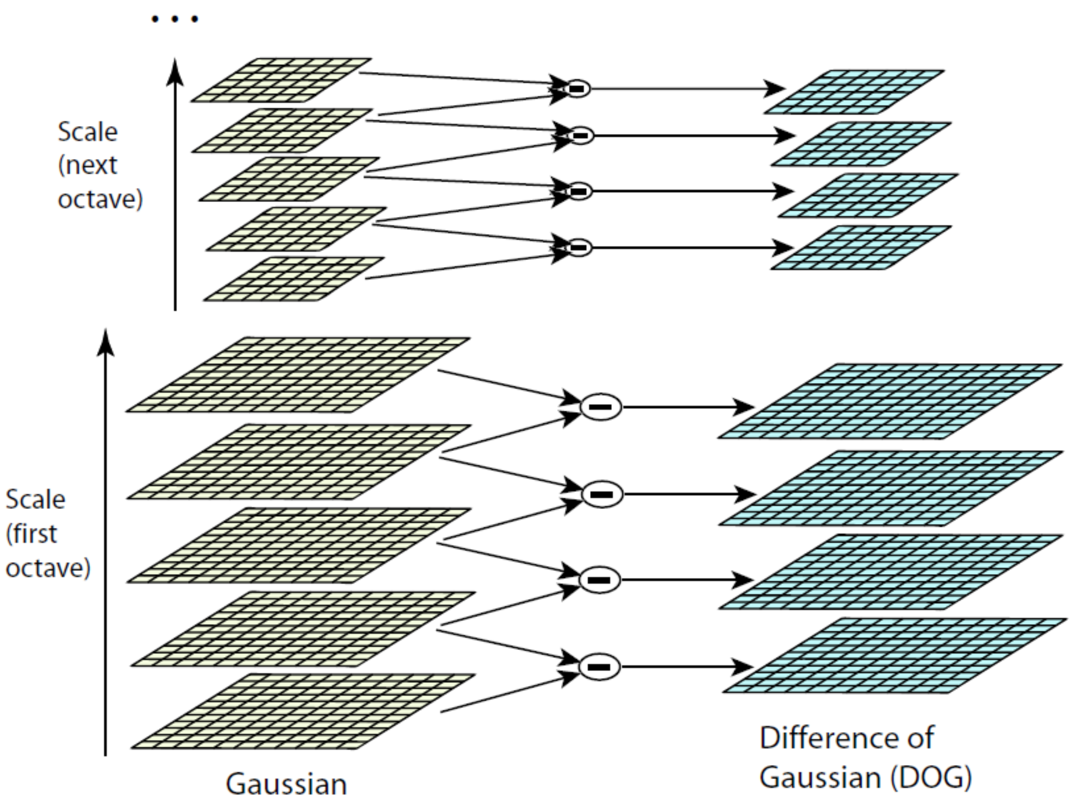
$$(\hat{x}, \hat{y}; \hat{t}) = \mathrm{argmaxminlocal}_{(x,y;t)}(\nabla^2_{norm} L(x,y;t))$$

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

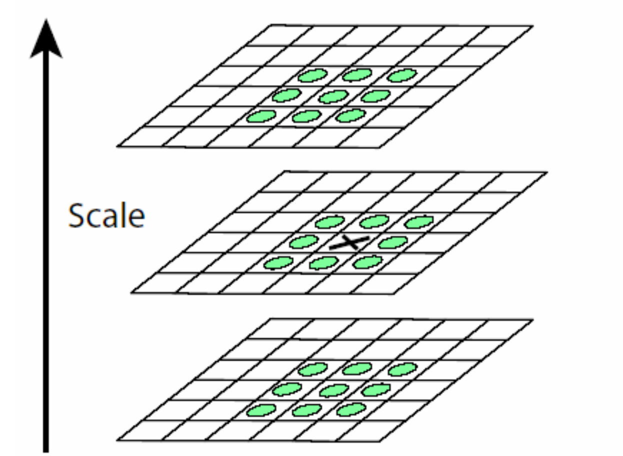
SIFT detector: Difference of Gaussians

- 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

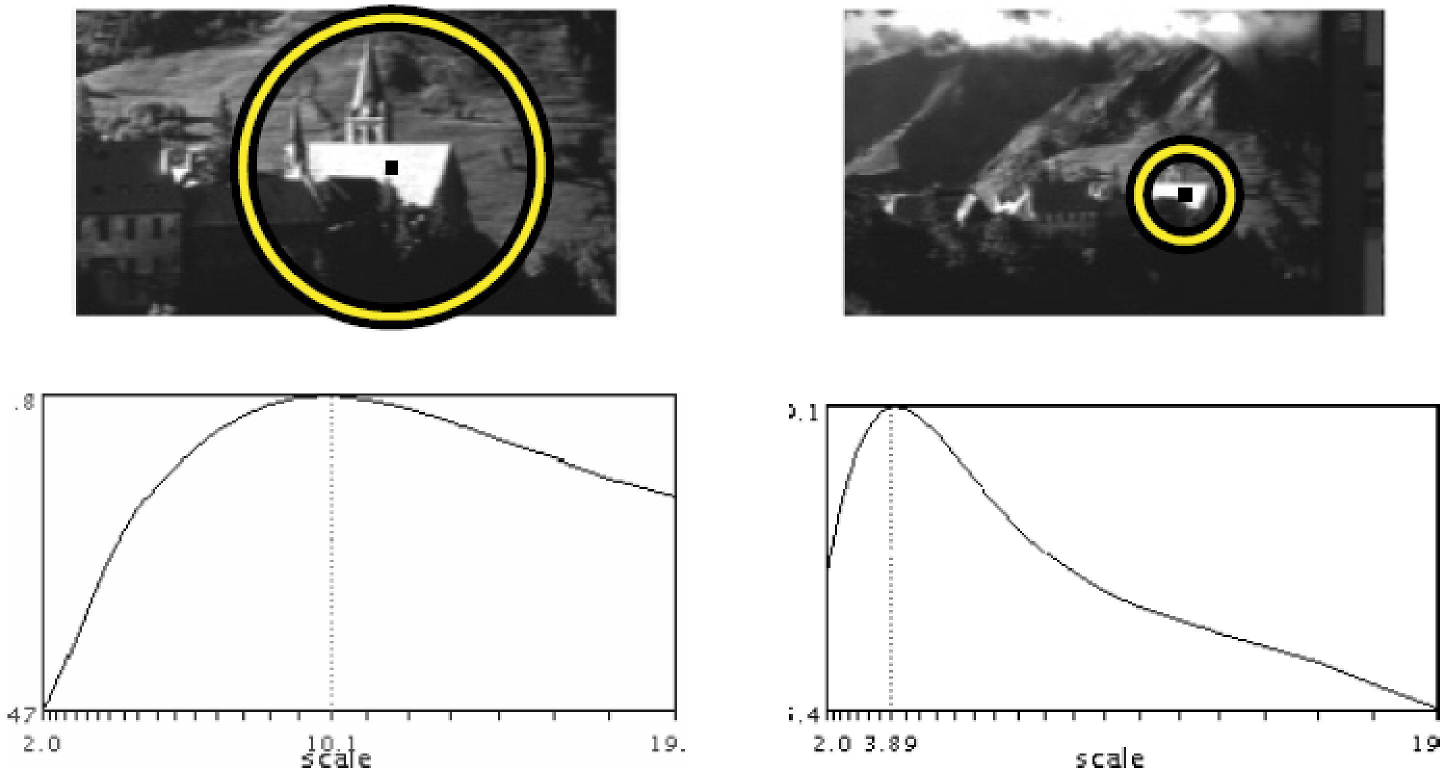
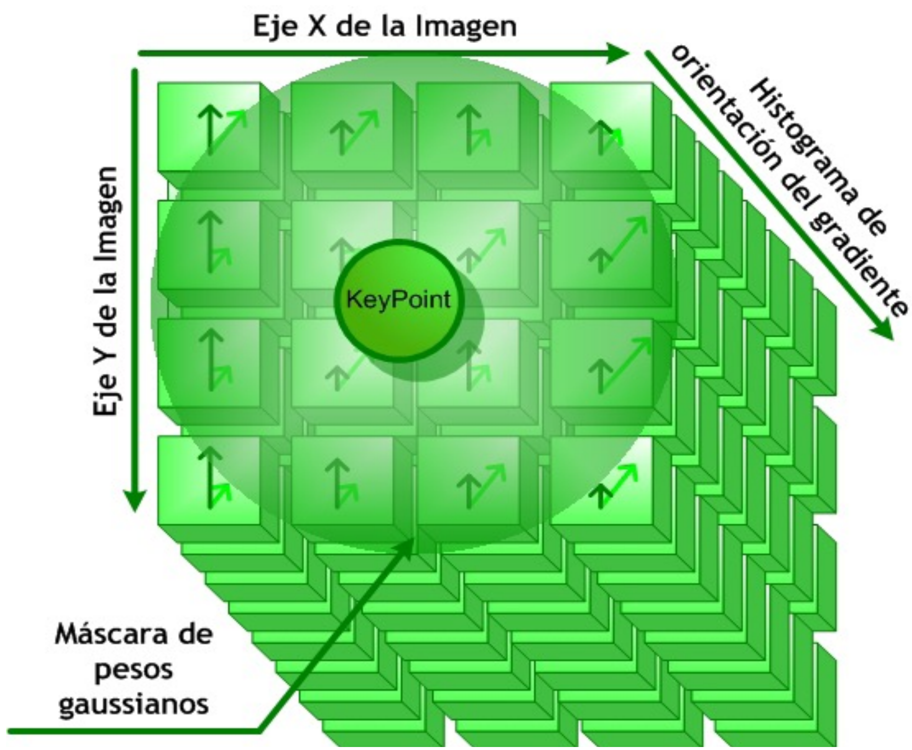


Fig. 3.5 Example of characteristic scales. The top row shows images taken with different zoom. The bottom row shows the responses of the Laplacian over scales for two corresponding points. The characteristic scales are 10.1 and 3.9 for the left and right images, respectively. The ratio of scales corresponds 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

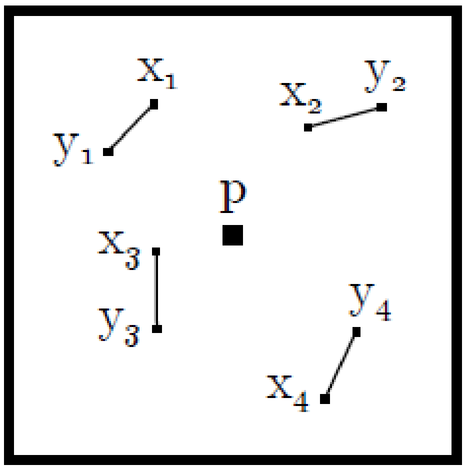


  - 128 bytes (floats): 16 areas x 8 histogram bins

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.

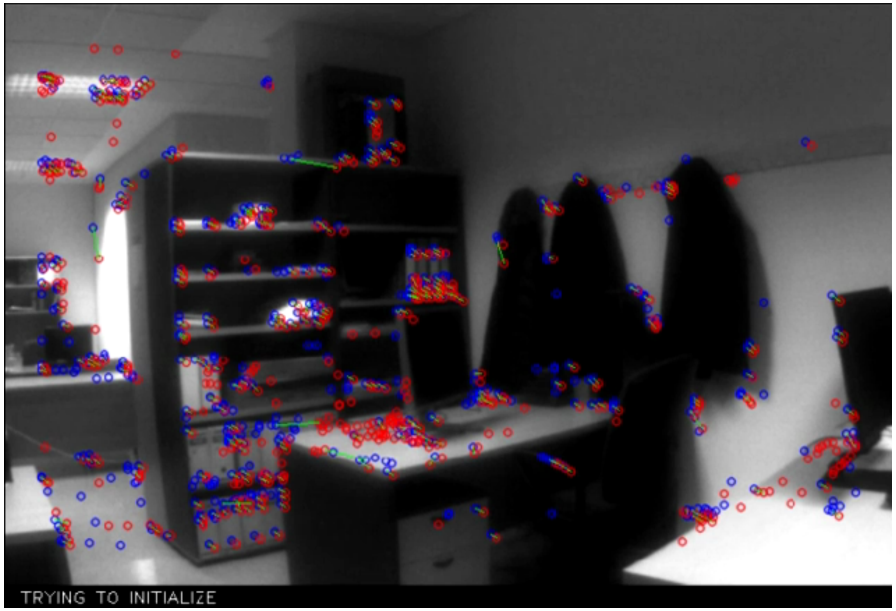
Popular Features for Visual SLAM

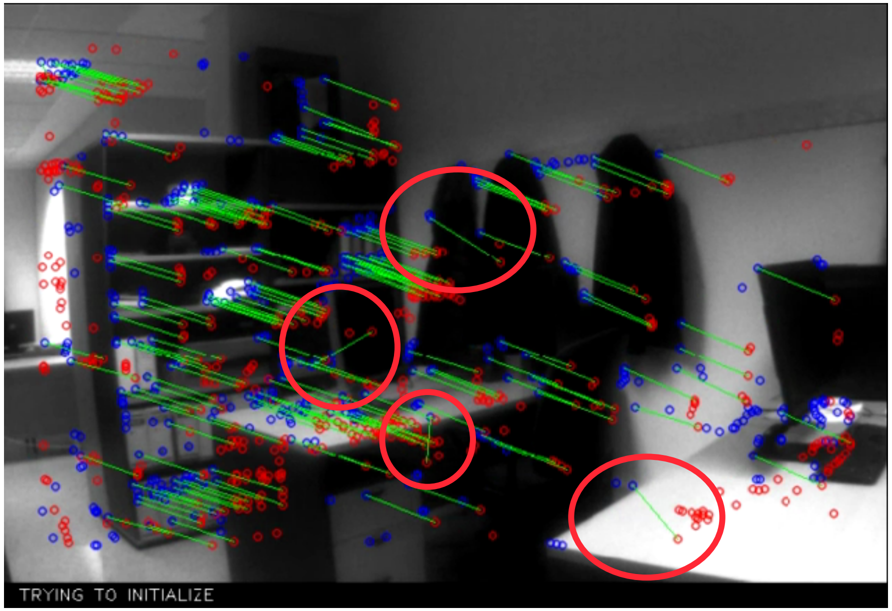
Detector	Descriptor	Rotation Invariant	Automatic Scale	Accuracy	Relocation & Loops	Efficiency
Harris	Patch	No	No	++++	-	++++
Shi-Tomasi	Patch	No	No	++++	-	++++
SIFT	SIFT	Yes	Yes	++	++++	+
SURF	SURF	Yes	Yes	++	++++	++
FAST	BRIEF	No	No	+++	+++	++++
ORB	ORB	Yes	No	+++	+++	++++

- ORB: Oriented FAST and Rotated Brief
  - 256-bit binary descriptor
  - Fast to extract and match (Hamming distance)
  - Good for tracking, relocation and Loop detection
  - Multi-scale detection → same point appears on several scales

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

3. Feature Matching

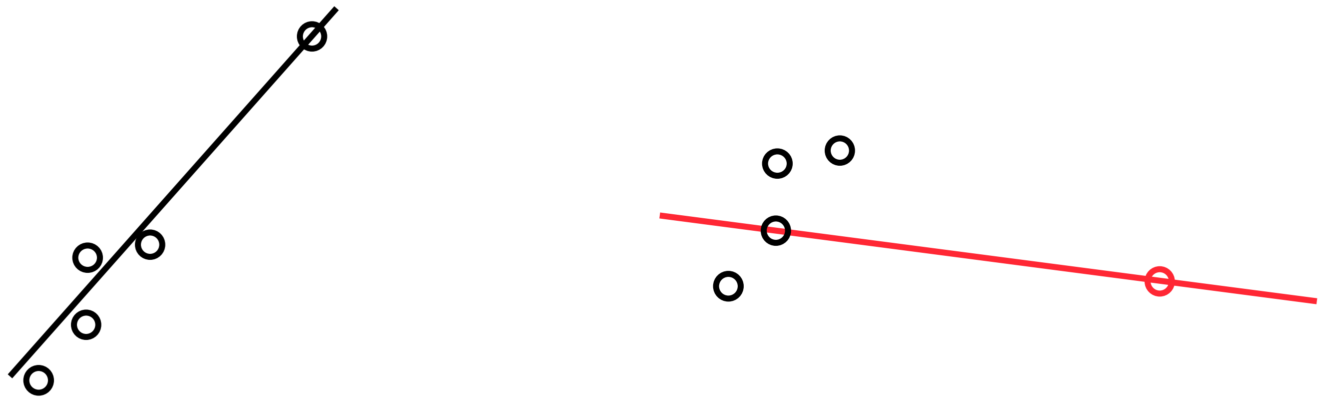




- Compare descriptors
- Spurious matchings
- Search for consensus with a robust technique: RANSAC

The problem of spurious matchings

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

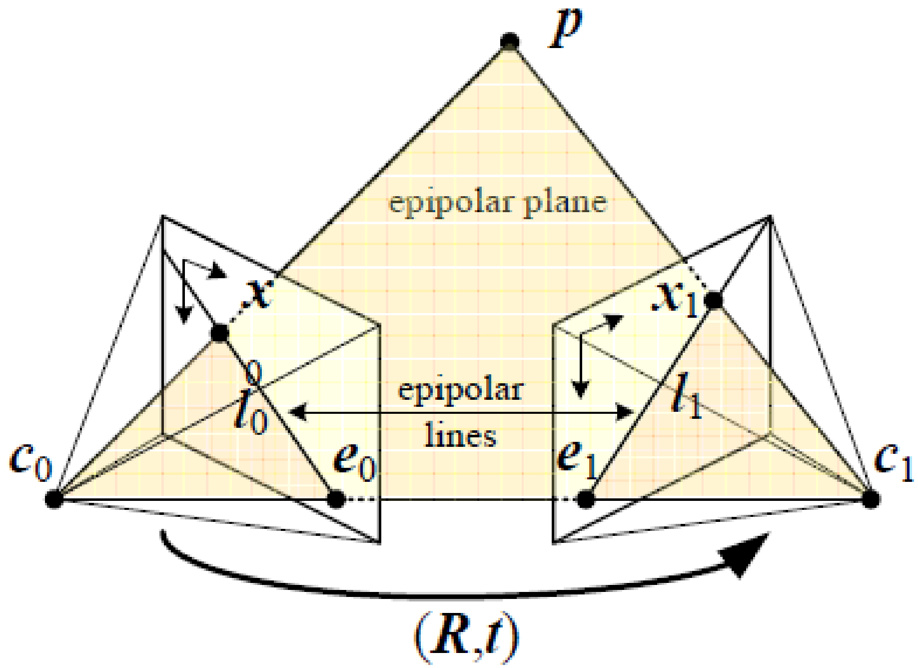


  - Removing the points with higher residuals DOES NOT SOLVE THE PROBLEM

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

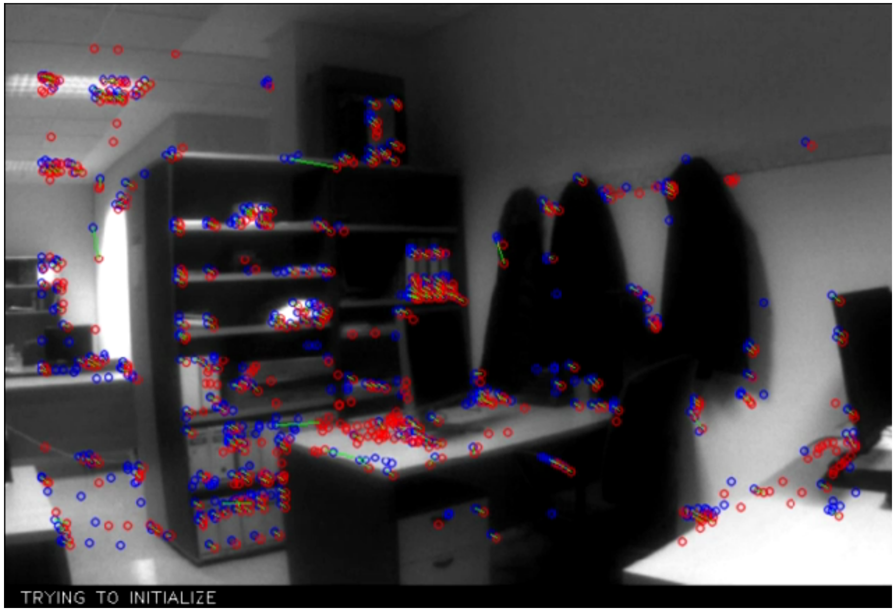


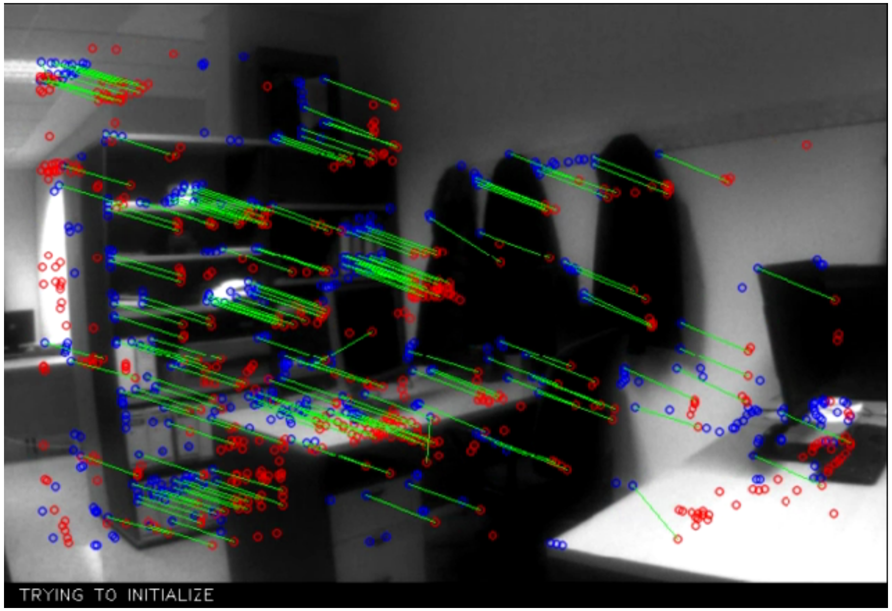
- Vectors **$\mathbf{t} = \mathbf{c}_1 - \mathbf{c}_0$, $\mathbf{p} - \mathbf{c}_0$, $\mathbf{p} - \mathbf{c}_1$** must be coplanar
- Epipolar constraint: $$\mathbf{x}_{c1}^T \mathbf{E}\, \mathbf{x}_{c0} = 0$$
- Essential Matrix: $$\mathbf{E} = [\mathbf{t}]_\times \mathbf{R} = \begin{bmatrix} 0 & -t_z & t_y \\ t_z & 0 & -t_x \\ -t_y & t_x & 0 \end{bmatrix} \mathbf{R}$$

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
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	

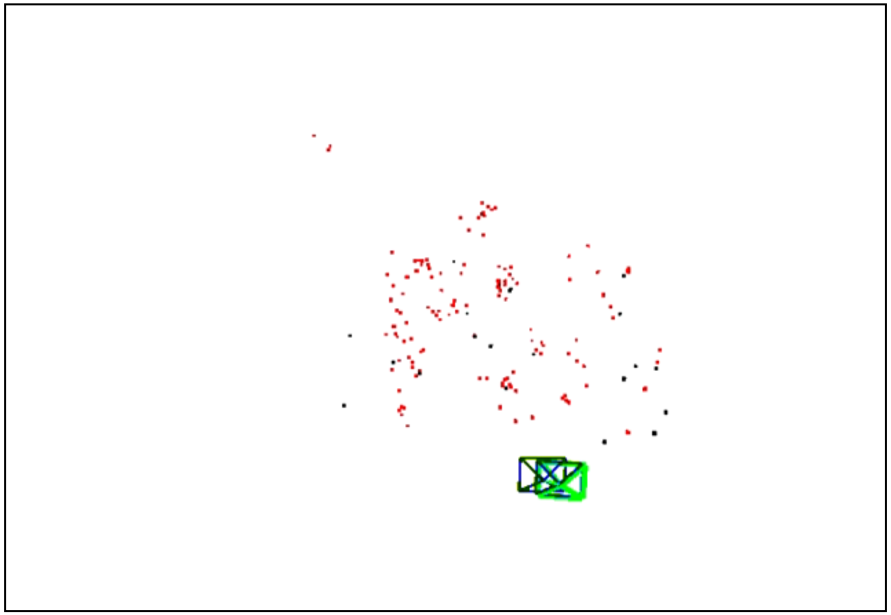
Matchings in 2 Frames → 3D Points and Motion





SFM:

- 5pt algorithm
- 8pt algorithm



Unknown Scale!

4. Relocation and Loop closing

- Relocation problem:

  During SLAM tracking can be lost: occlusions, low tecture, quick motions,…

  ➢ Re-acquire camera pose and continue

- Loop closing problem

  SLAM is working, and you come back to a previously mapped area

  ➢ Loop detection: to avoid map duplication

  ➢ Loop correction: to compensate the accumulated drift

- In both cases you need a place recognition technique

Why is Loop Detection Difficult?

- Is this a loop closure?

Likely algorithm answer:

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

Why is Loop Detection Difficult?

- Is this a loop closure?

Scene 1430

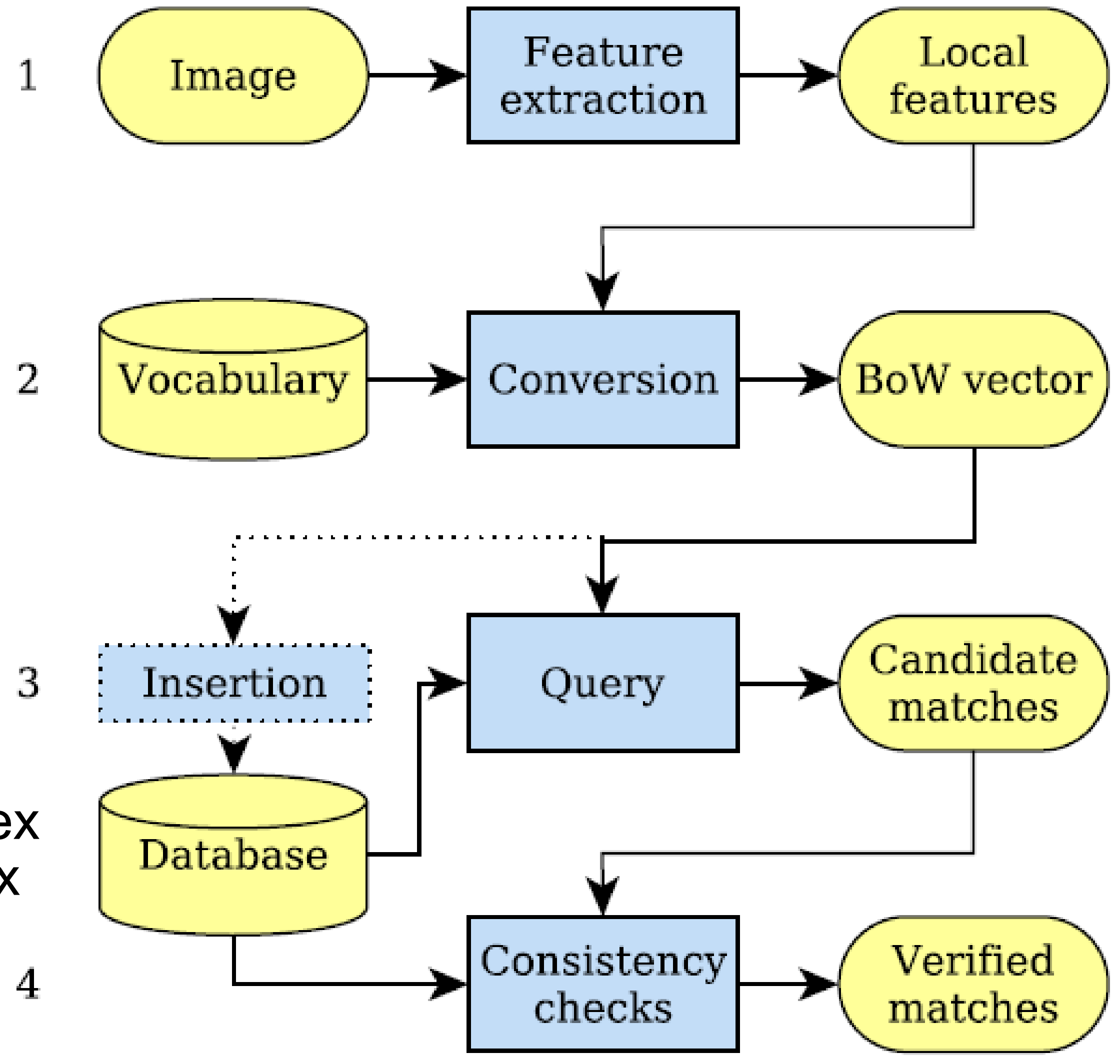
Scene 1244

Likely algorithm answer:

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

Perceptual aliasing is common in indoor scenarios

Bag of Words Approach

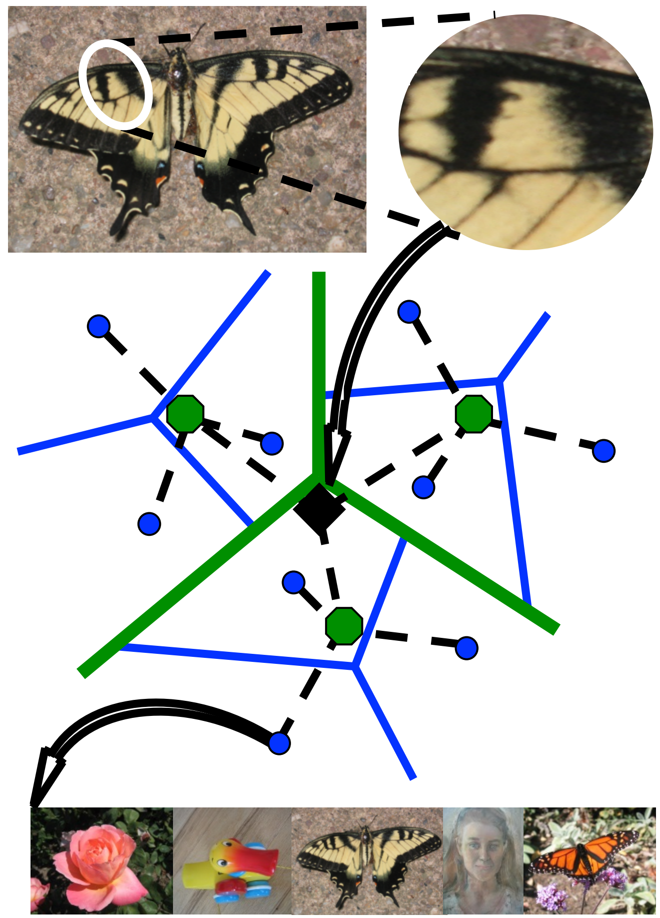


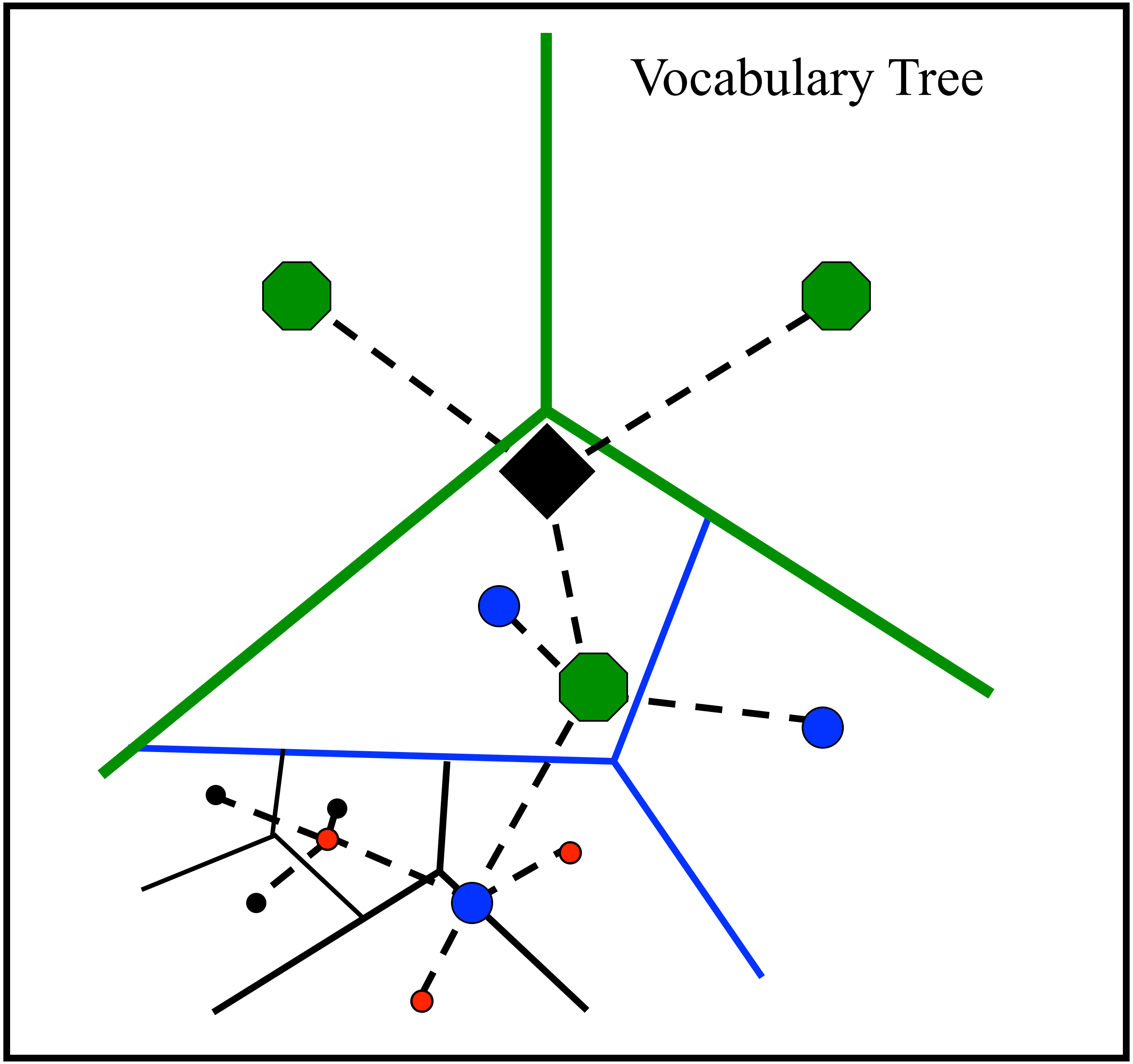
Binary Features
BRIEF, ORB

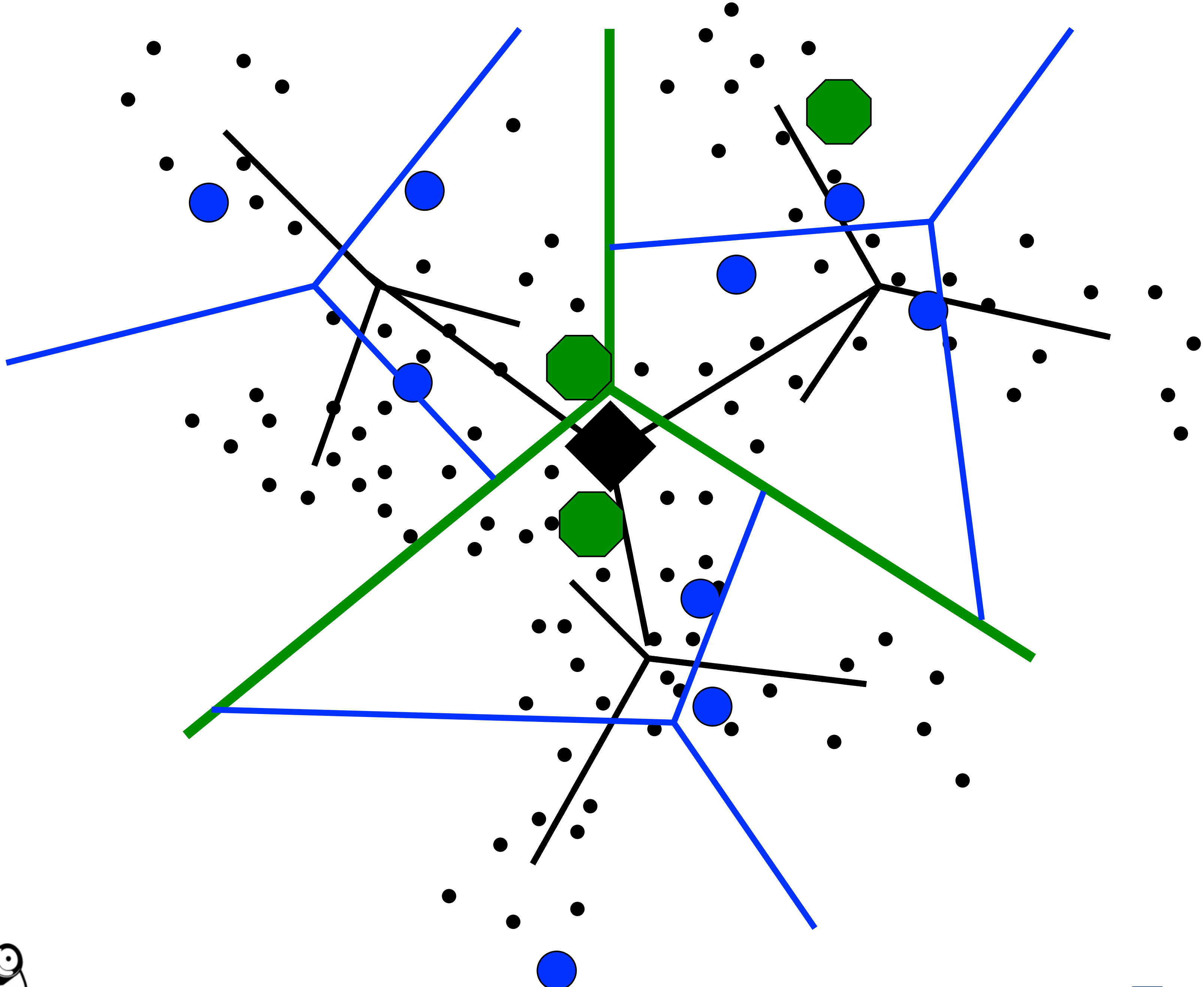
Inverse Index
Direct Index

Scalable Recognition with a Vocabulary Tree

David Nistér, Henrik Stewénius
CVPR 2006



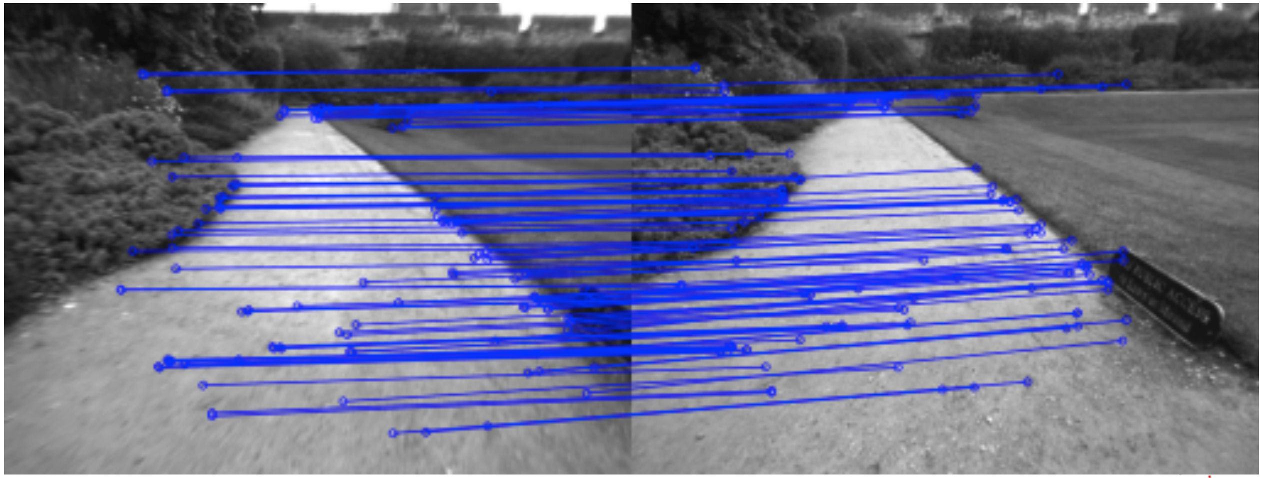

Vocabulary Tree

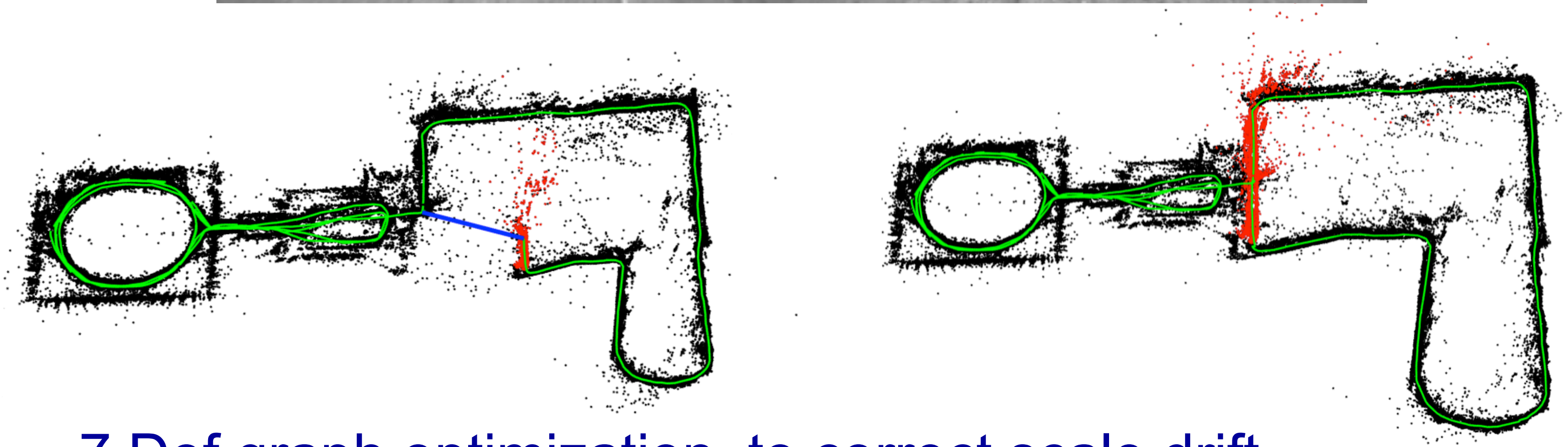


Examples with DBoW2 using ORB features

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)

Loop Correction





- 7 Dof graph optimization, to correct scale drift
- And optionally Full BA (little improvement, much slower)

Outline

1. Feature-Based Visual SLAM
2. Features
3. Feature Matching
4. Relocation and Loop Closing
5. Putting all together: ORB-SLAM
6. ORB-SLAM2: Stereo and RGB-D
7. Visual-Inertial ORB-SLAM

ORB-SLAM: Feature-Based SLAM, 2015

- Use the same features for:
  - Tracking
  - Mapping
  - Loop closing
  - Relocation
- ORB: FAST corner + Oriented Rotated Brief descriptor
  - Binary descriptor
  - Very fast to compute and compare
- Real-time, large scale operation
- Survival of the fittest for points and keyframes

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

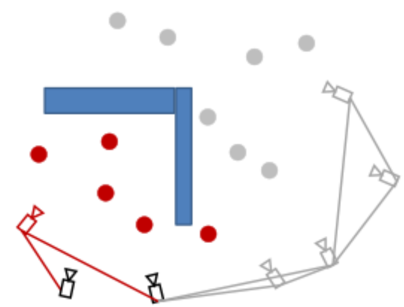
Recent Key Ideas

- Scale Drift-Aware Loop Closing

  H. Strasdat, J.M.M. Montiel and A.J. Davison
  Scale Drift-Aware Large Scale Monocular SLAM
  RSS 2010

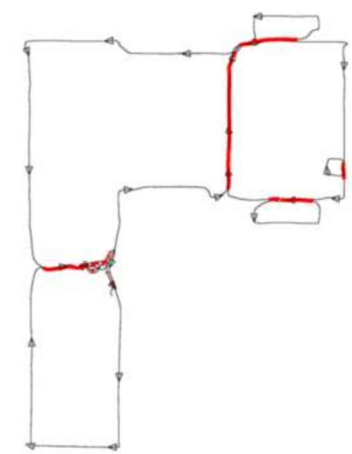
- Covisibility Graph

  H. Strasdat, A. J. Davison, J. M. M. Montiel , K. Konolige
  Double Window Optimization for Constant Time Visual SLAM
  ICCV 2011



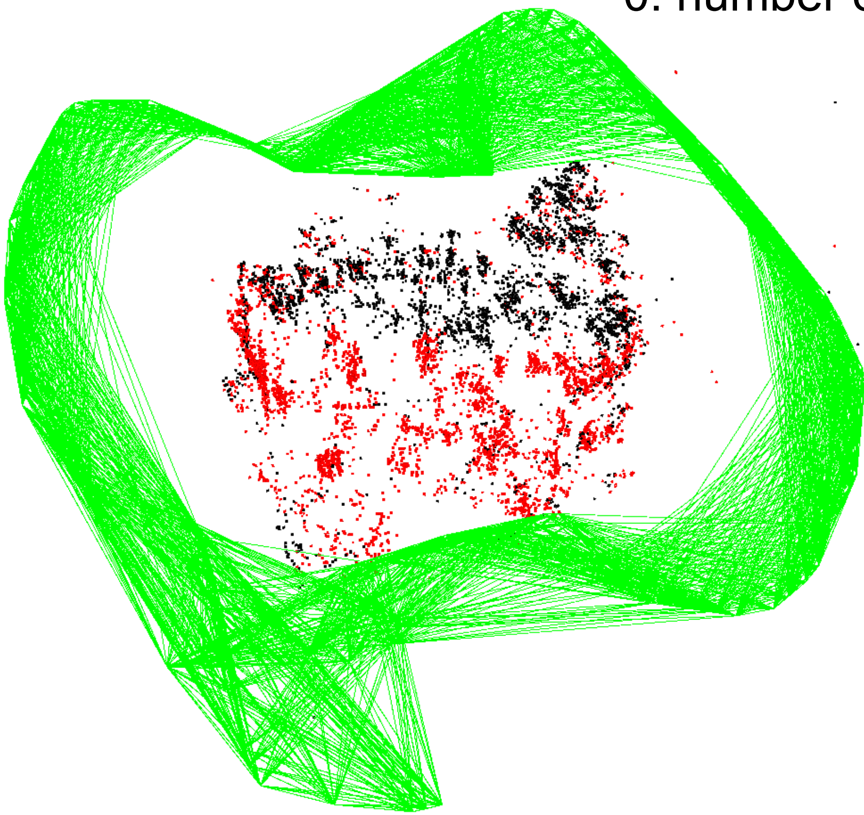
- Bags of Binary Words (DBoW)

  D. Gálvez-López and J. D. Tardós
  Bags of Binary Words for Fast Place Recognition in
  Image Sequences, IEEE Transactions on Robotics 2012



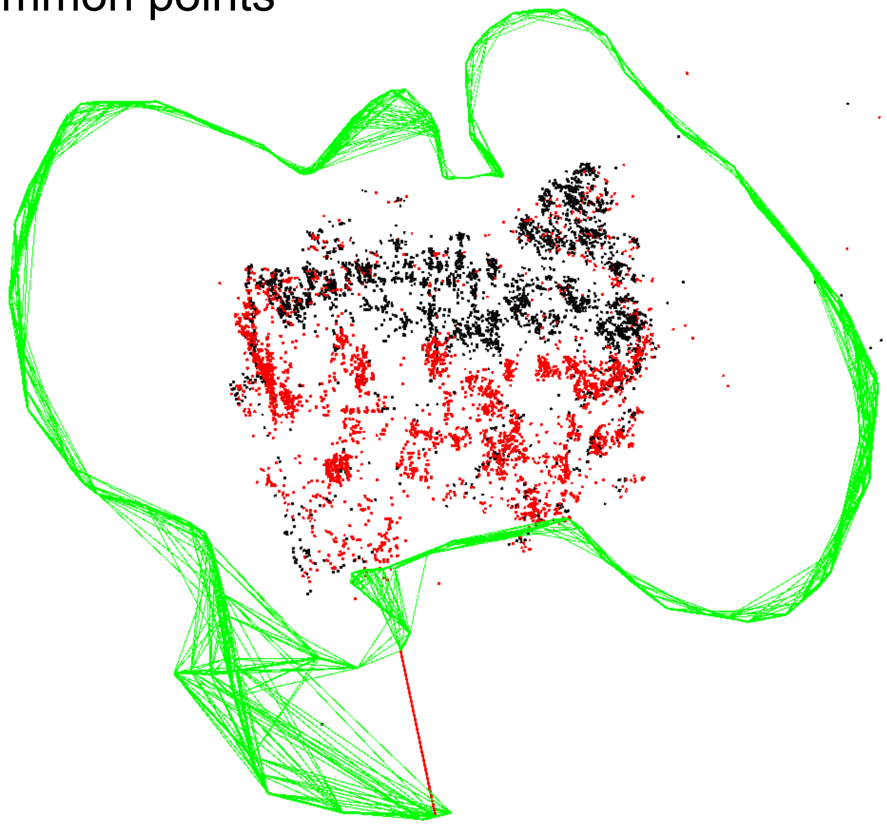
Covisibility Graph and Essential Graph

θ: number of common points



$\theta_{min} = 15$

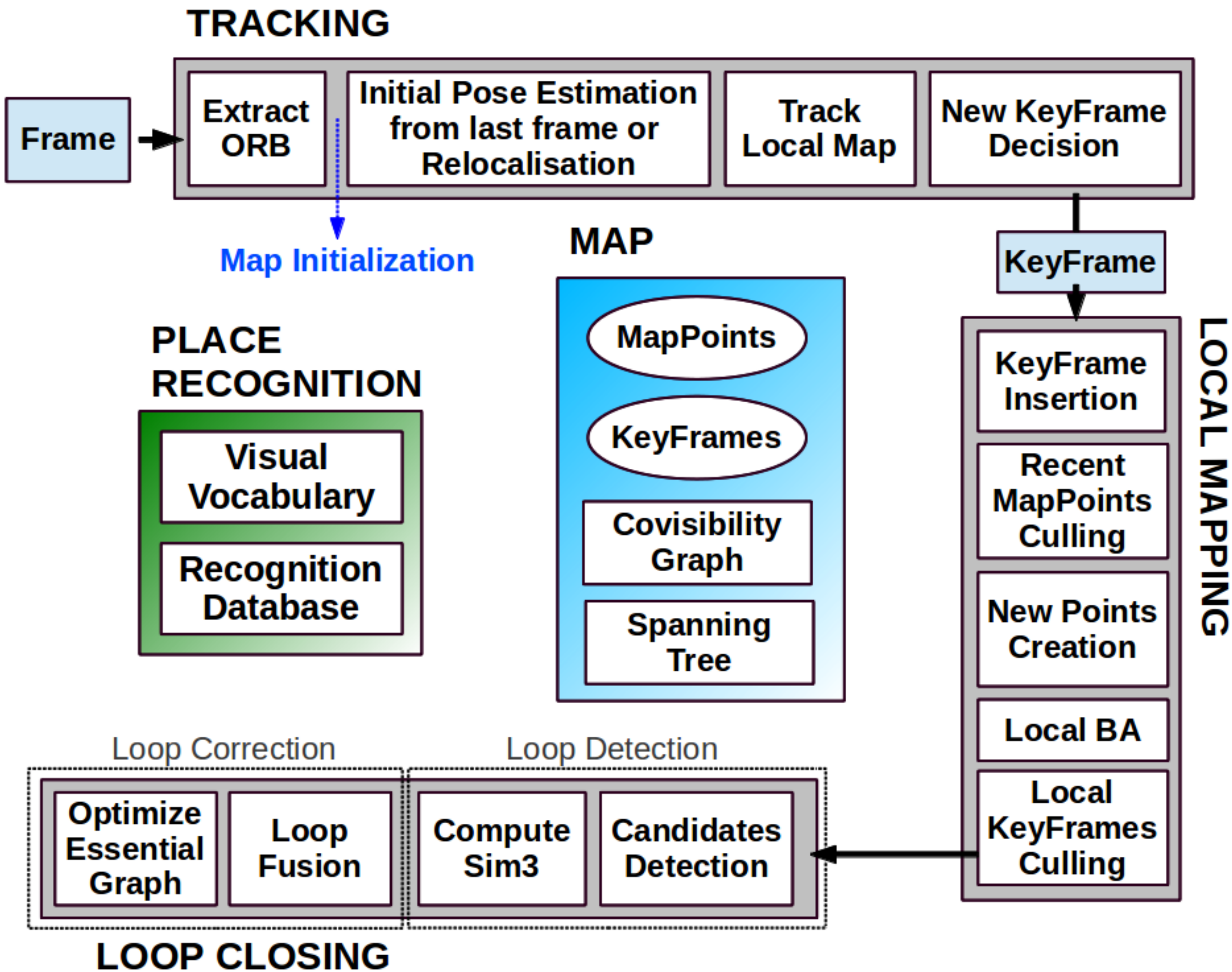
Used for Local BA



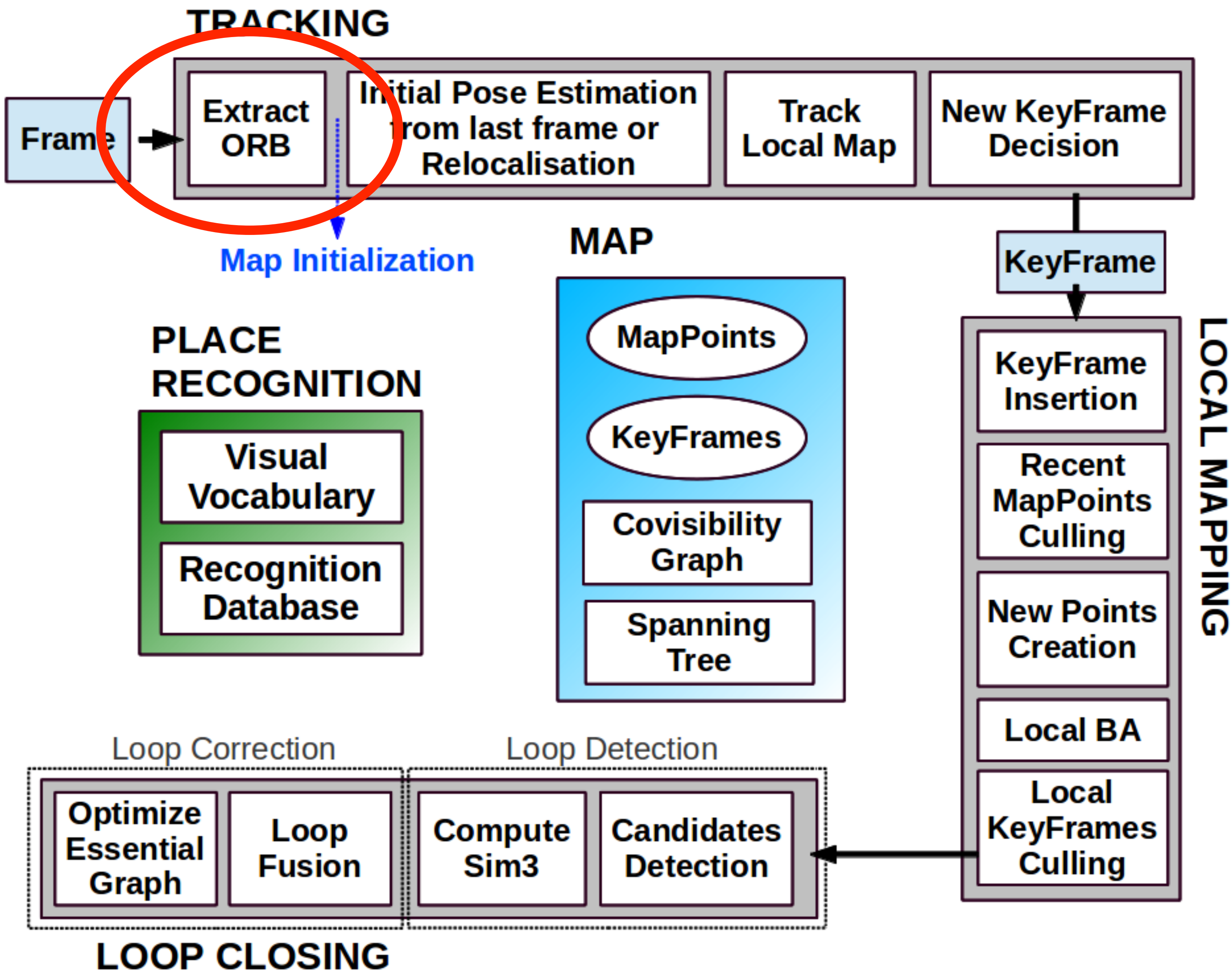
$\theta_{min} = 100$

Used for Loop Correction

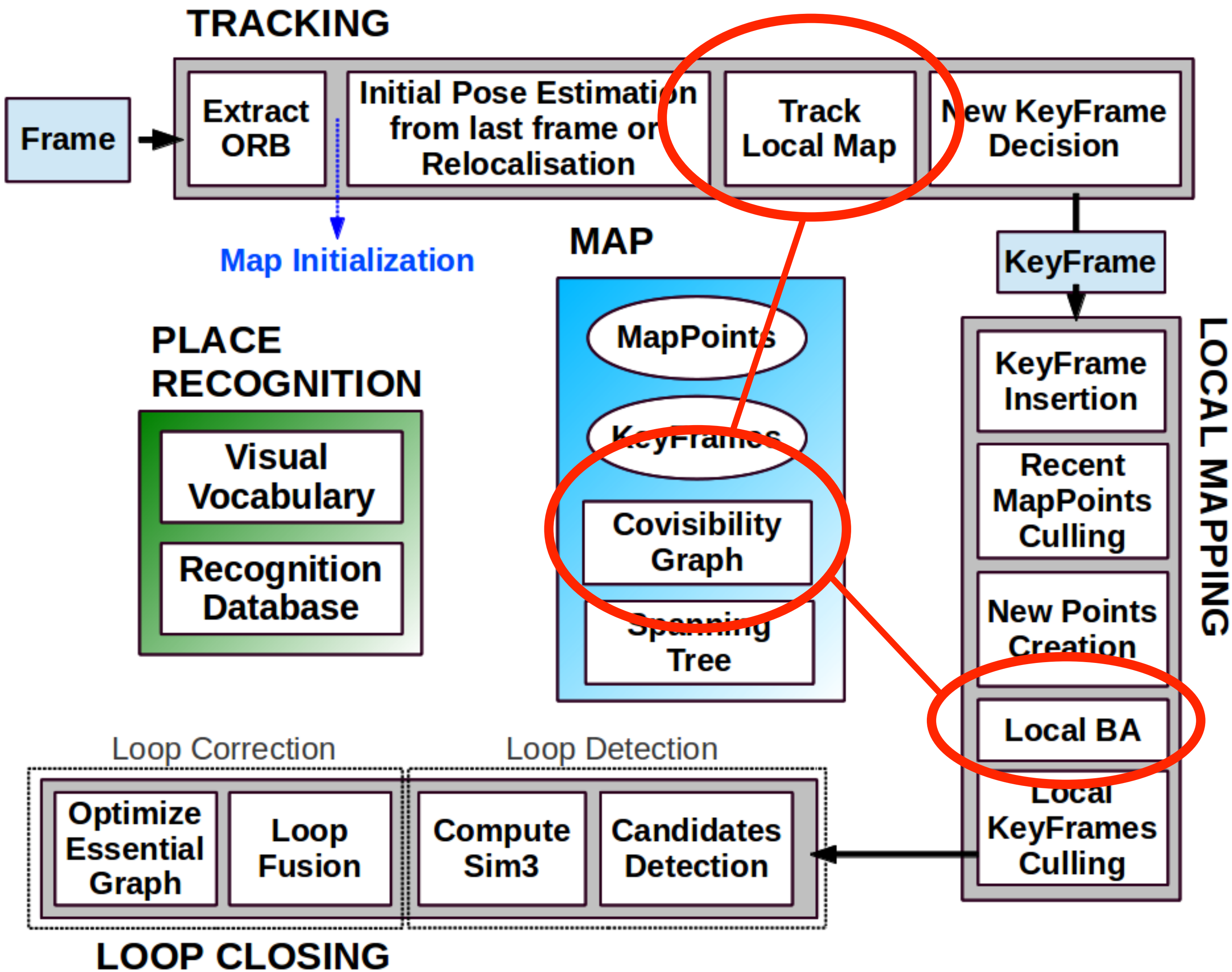
ORB-SLAM: Real-Time Monocular SLAM



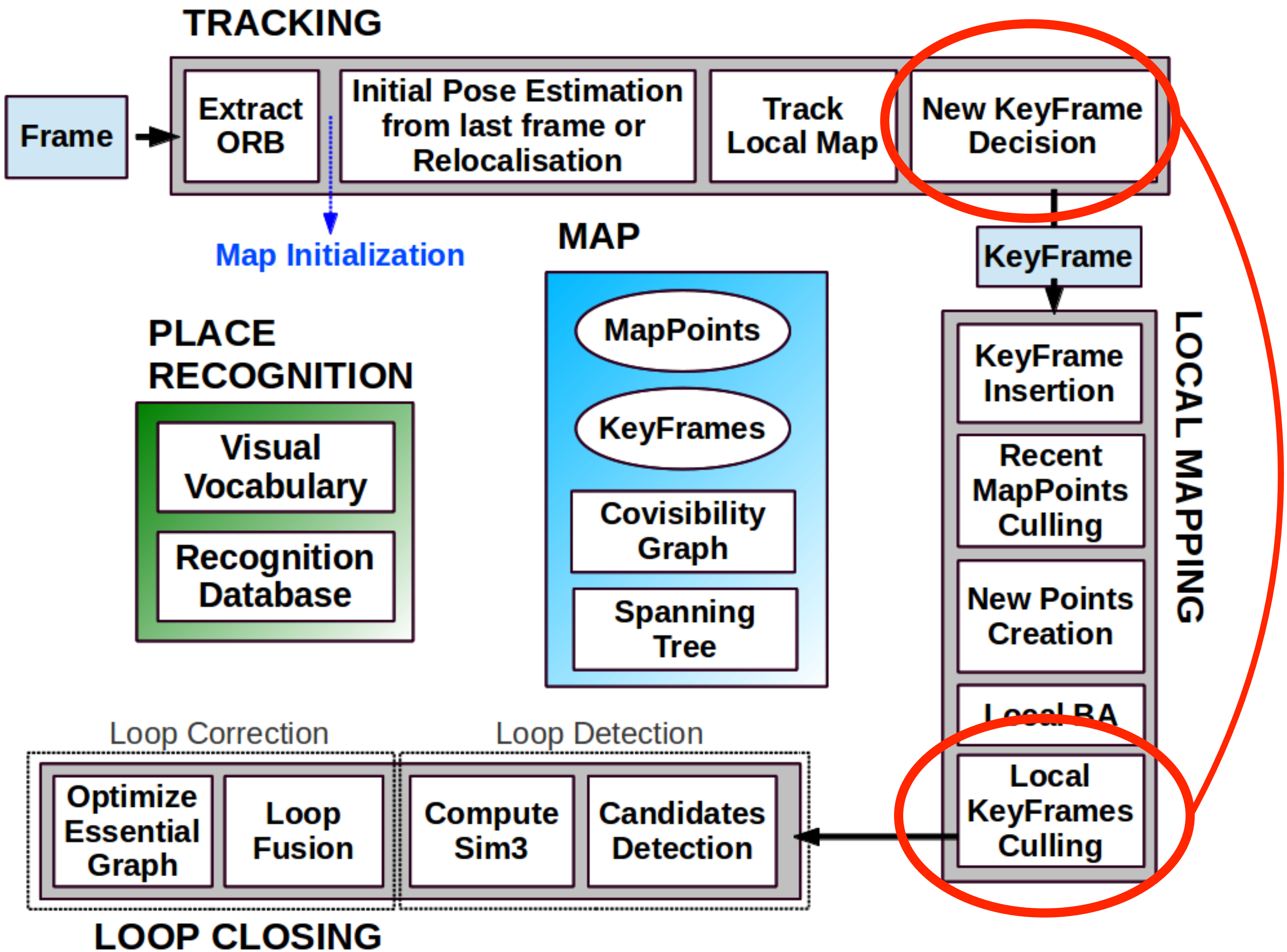
ORB-SLAM: Real-Time Monocular SLAM



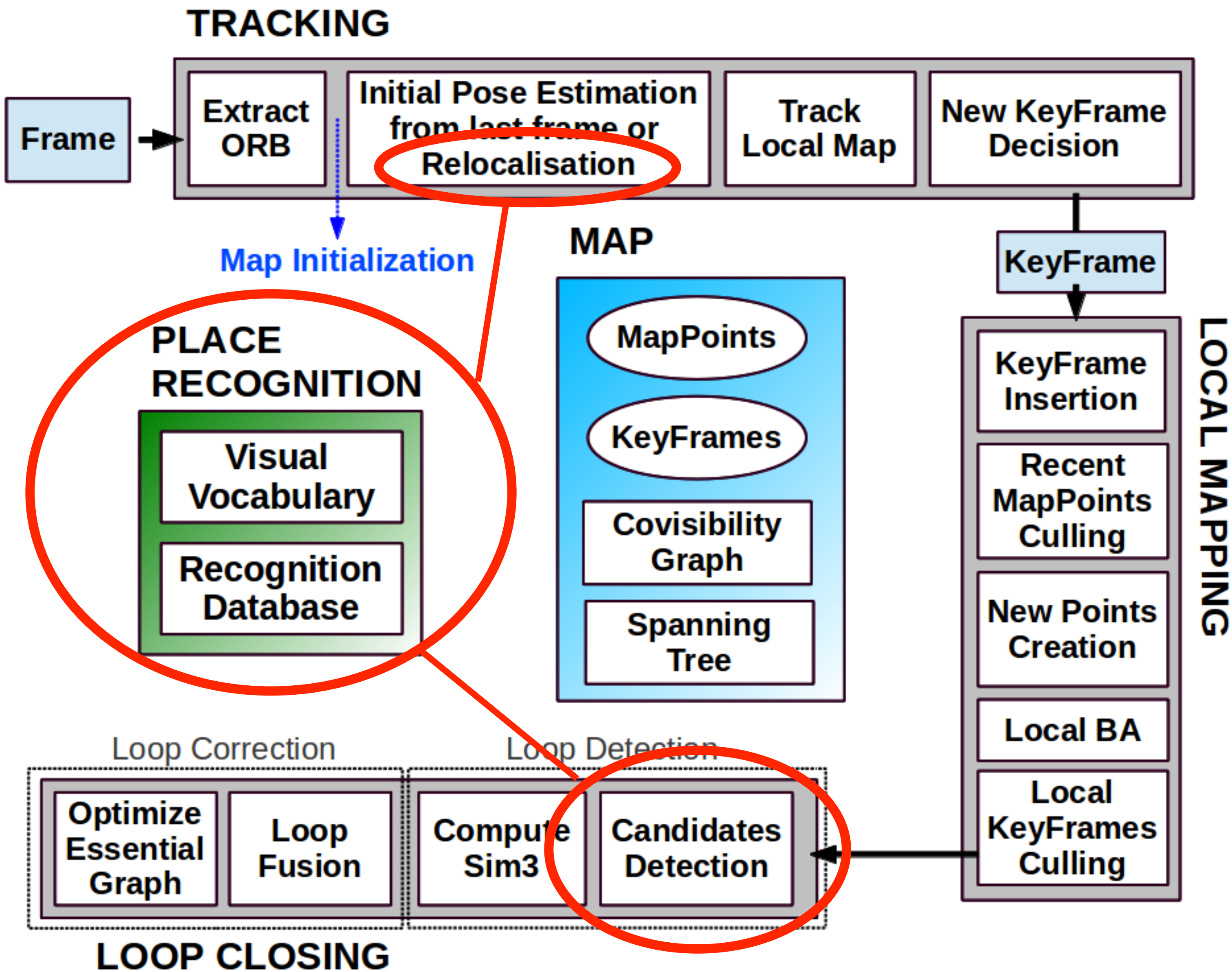
ORB-SLAM: Real-Time Monocular SLAM



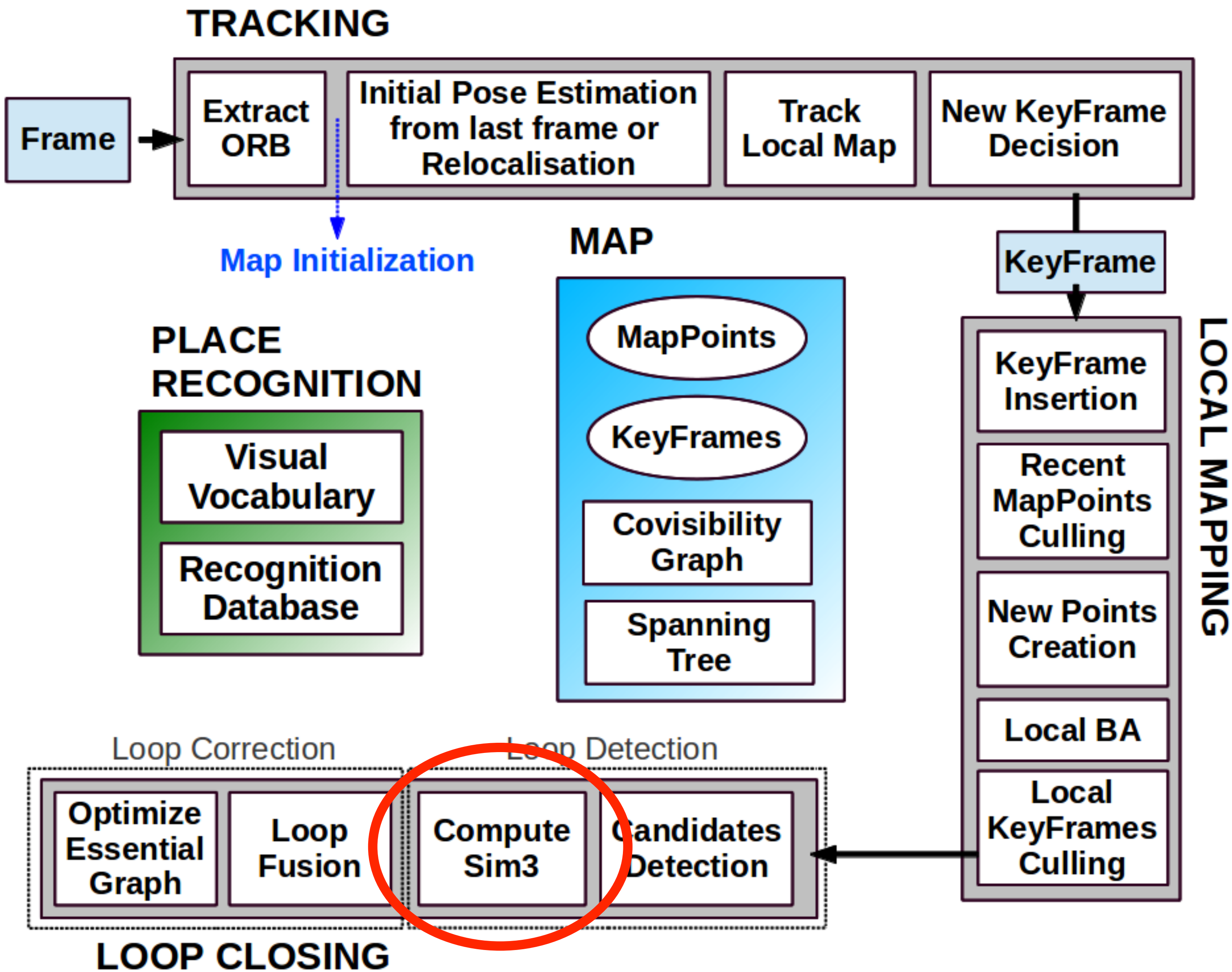
ORB-SLAM: Real-Time Monocular SLAM



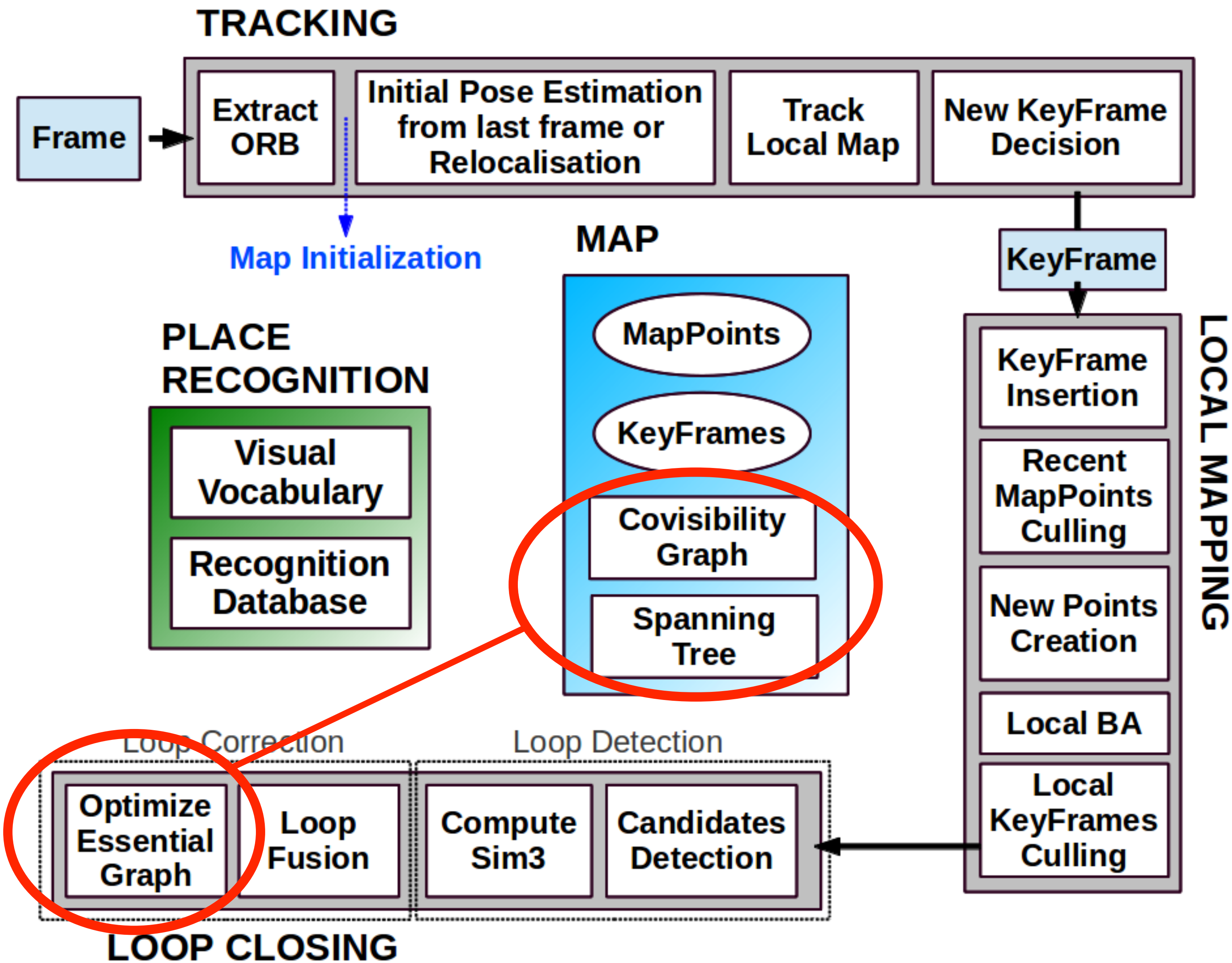
ORB-SLAM: Real-Time Monocular SLAM



ORB-SLAM: Real-Time Monocular SLAM



ORB-SLAM: Real-Time Monocular SLAM



ORB-SLAM indoors: TUM RGB-D dataset

ORB-SLAM

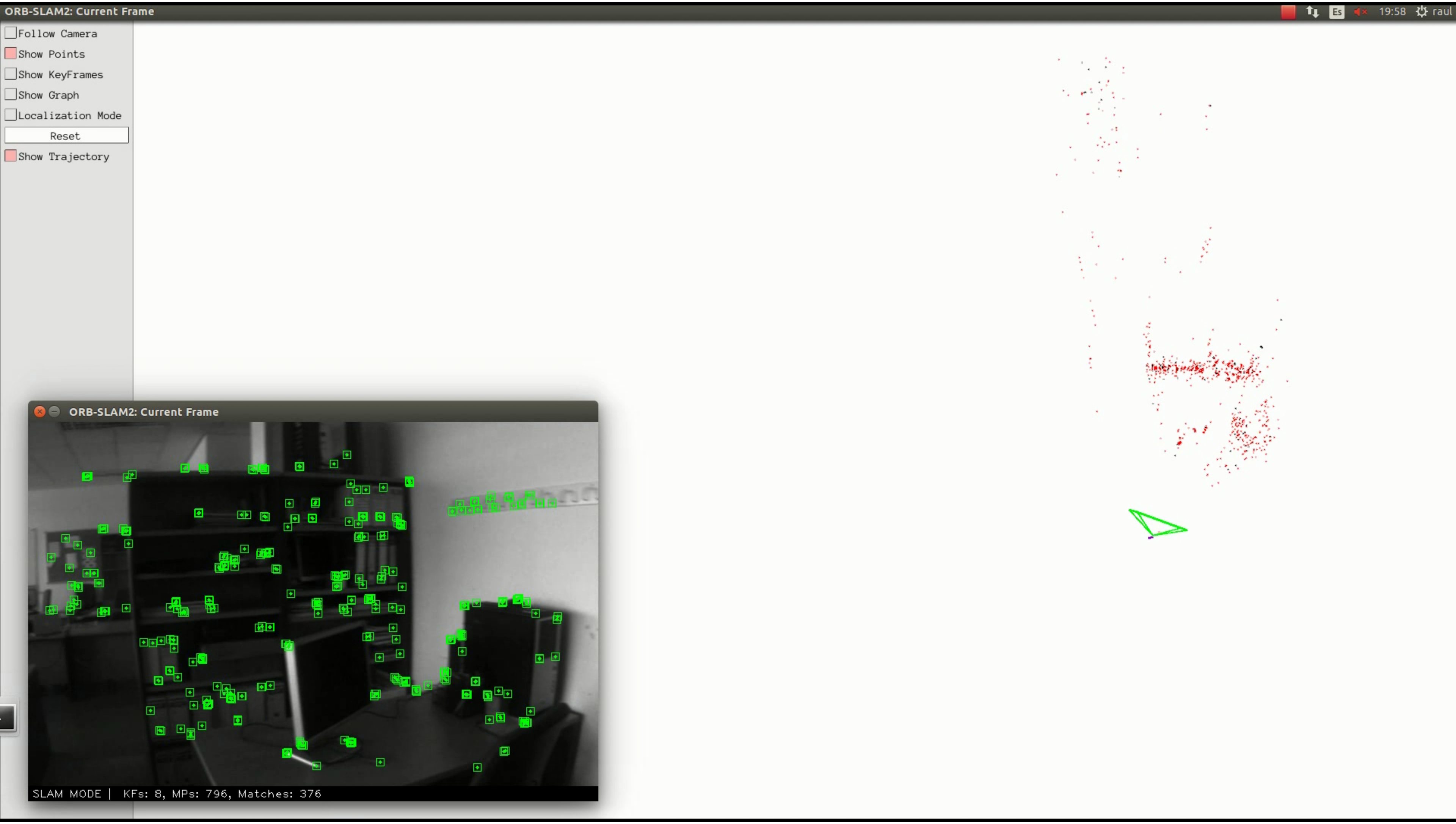
Raúl Mur-Artal, J. M. M. Montiel and Juan D. Tardós

{raulmur, josemari, tardos} @unizar.es

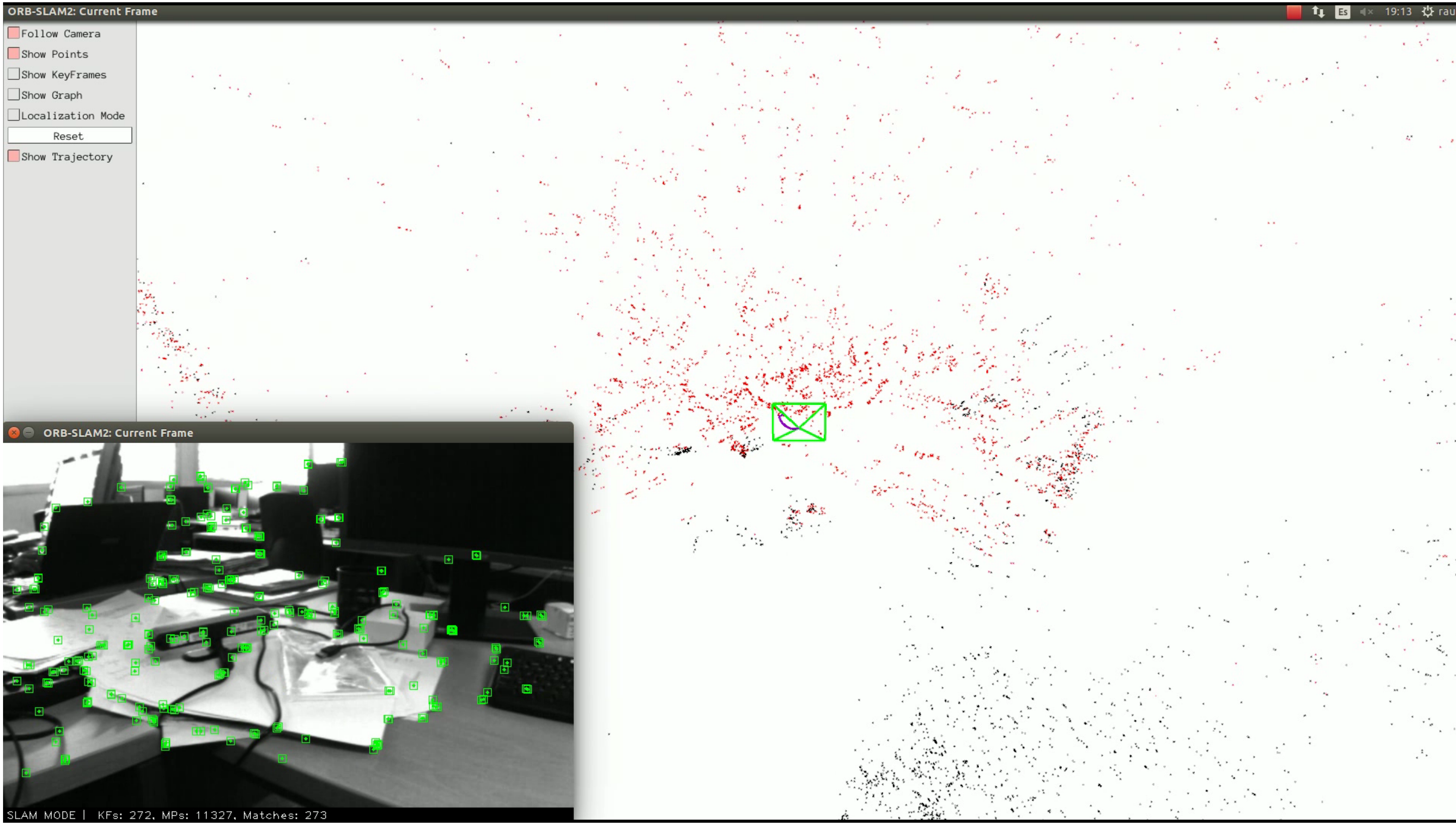
Instituto Universitario de Investigación
en Ingeniería de Aragón
Universidad Zaragoza

Universidad
Zaragoza
1542

ORB-SLAM indoors: 2cm precision



ORBSLAM Robust Tracking



ORB-SLAM outdoors: Kitti Dataset

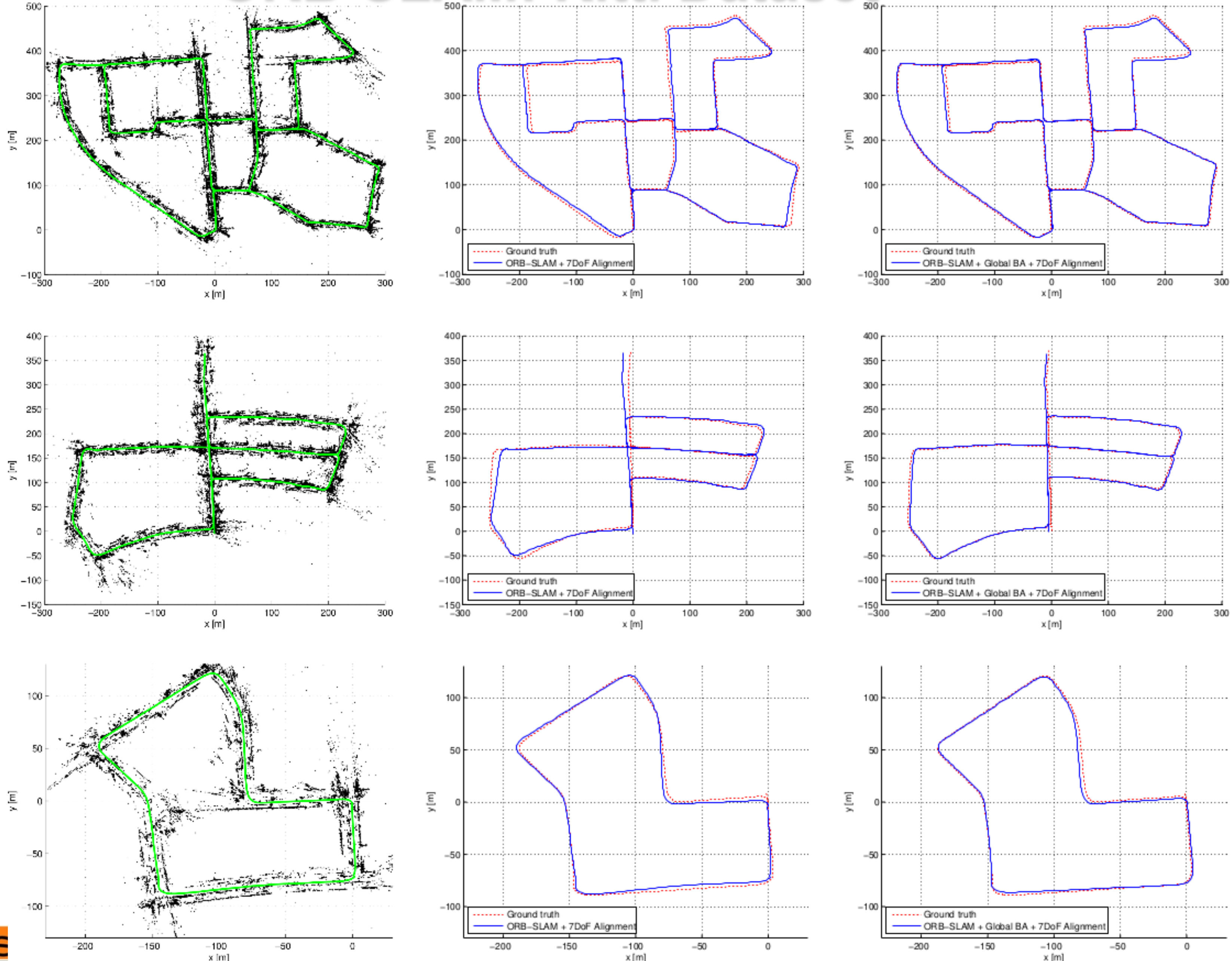
ORB-SLAM

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

{raulmur, josemari, tardos} @unizar.es

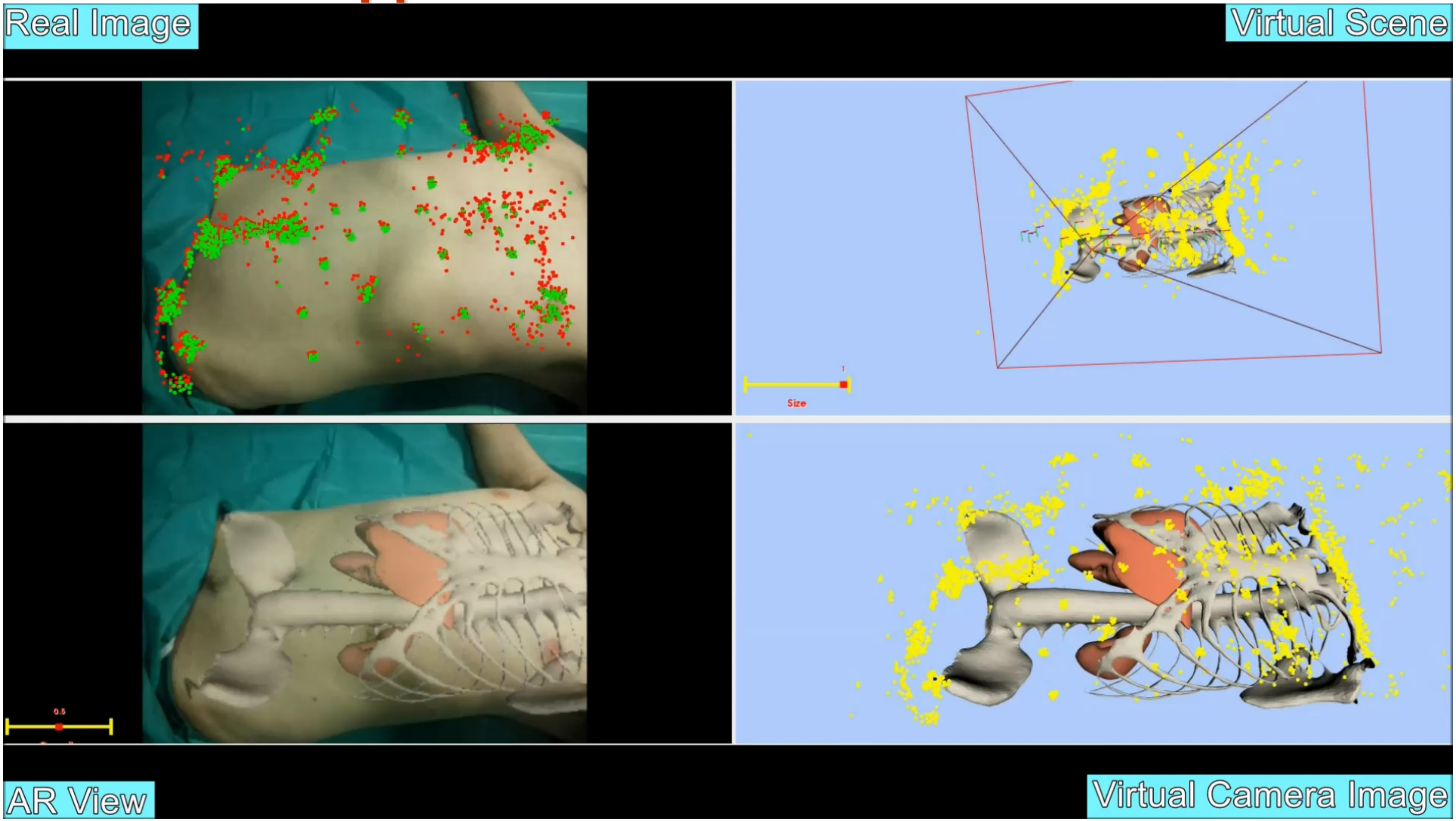
Trajectory and Map Obtained

ORB-SLAM: Kitti Dataset



ROBOTICS
UNIVERSIDAD DE ZARAGOZA

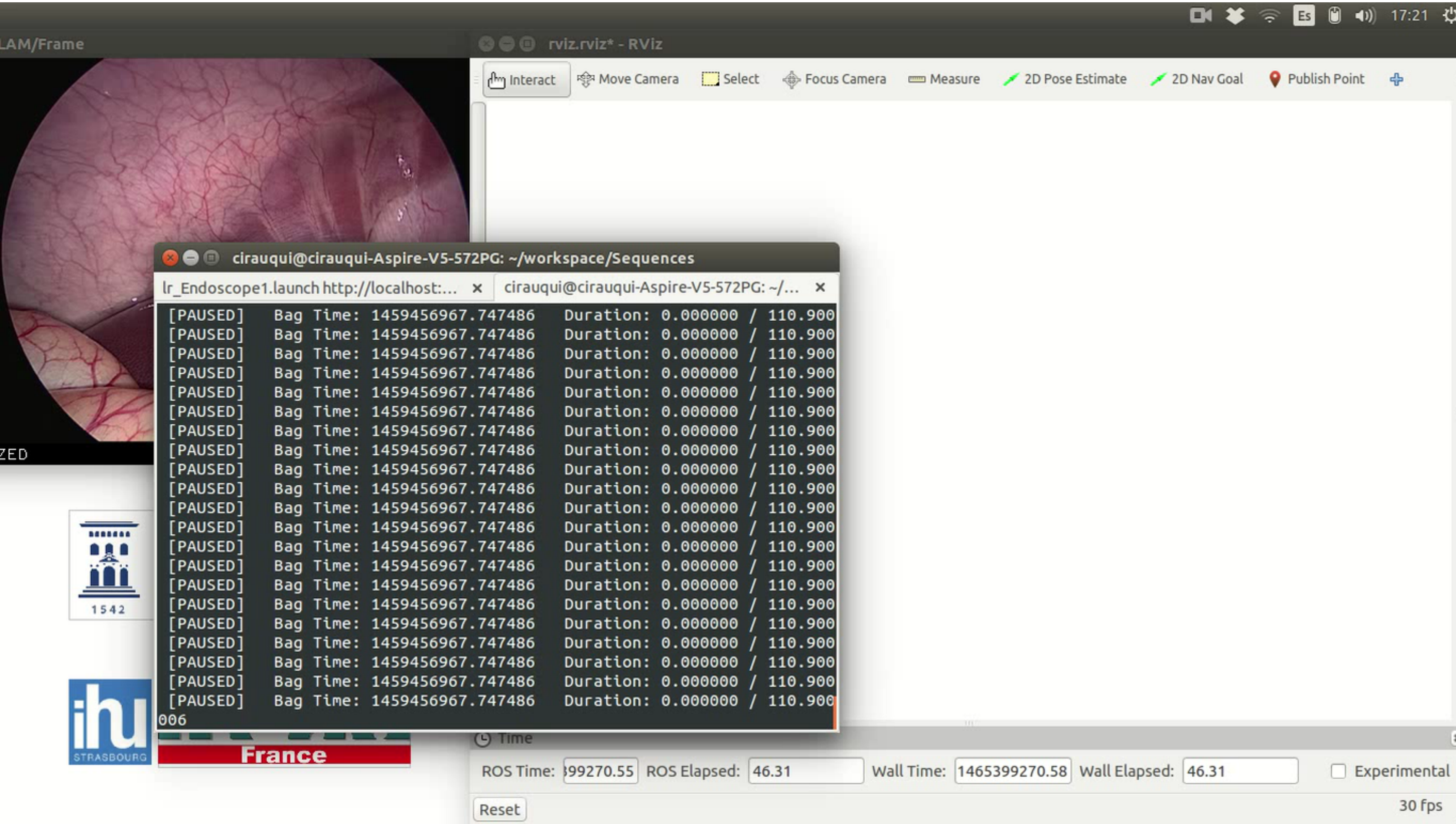
Applications: AR for Medicine



Cooperation with:

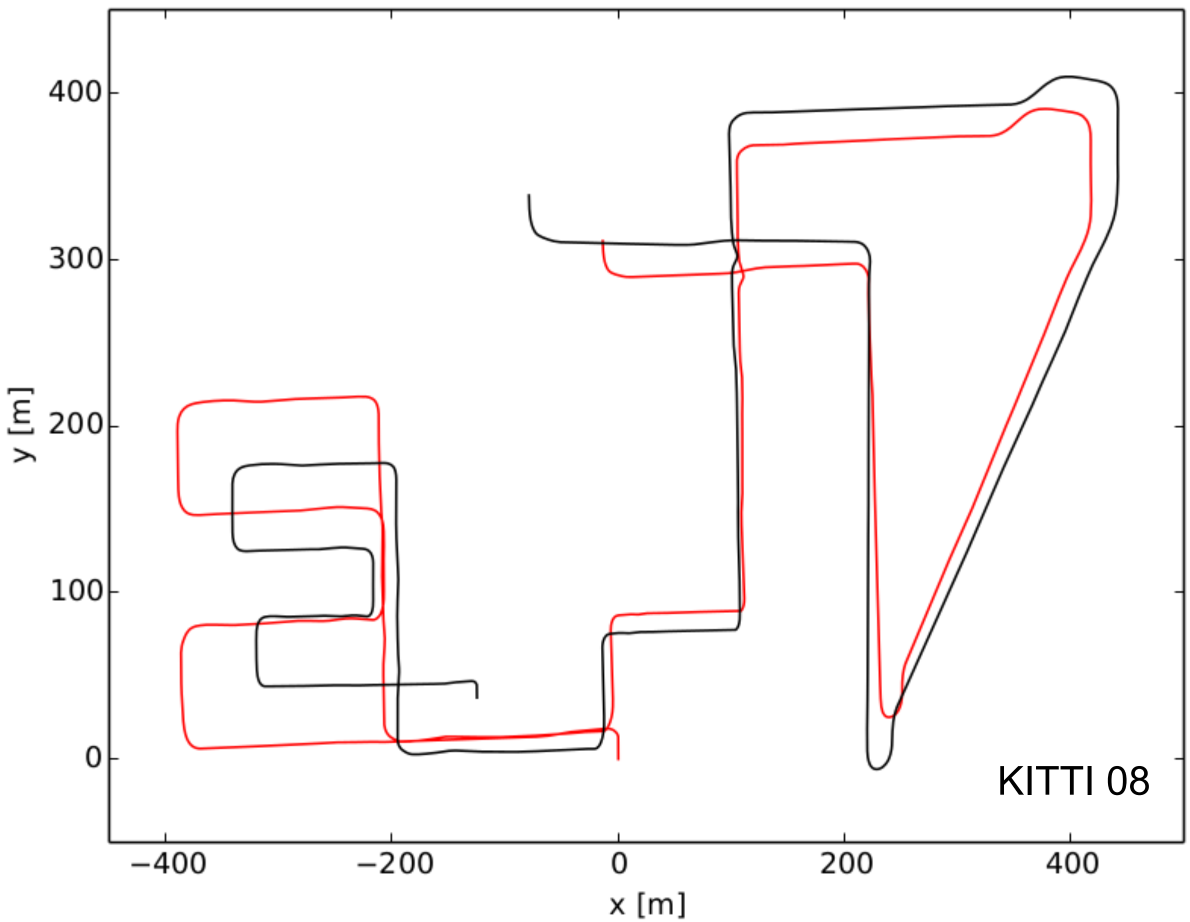
Institut de Recherche Contre les Cancers de l'Appareil Digéstif, Strasbourg, France

ORB-SLAM Inside the Body



ORB-SLAM Monocular

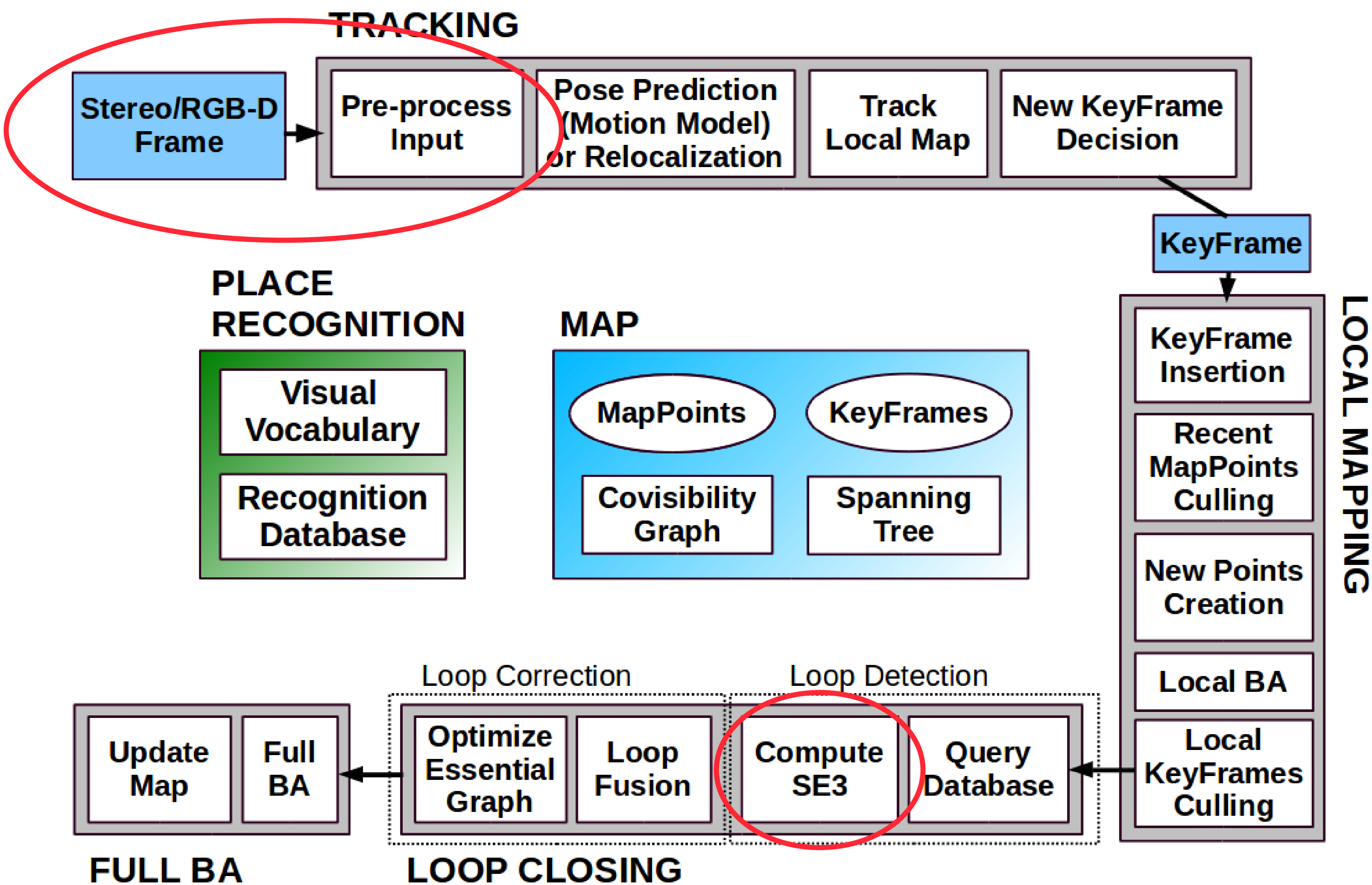
- With monocular scale is not observable



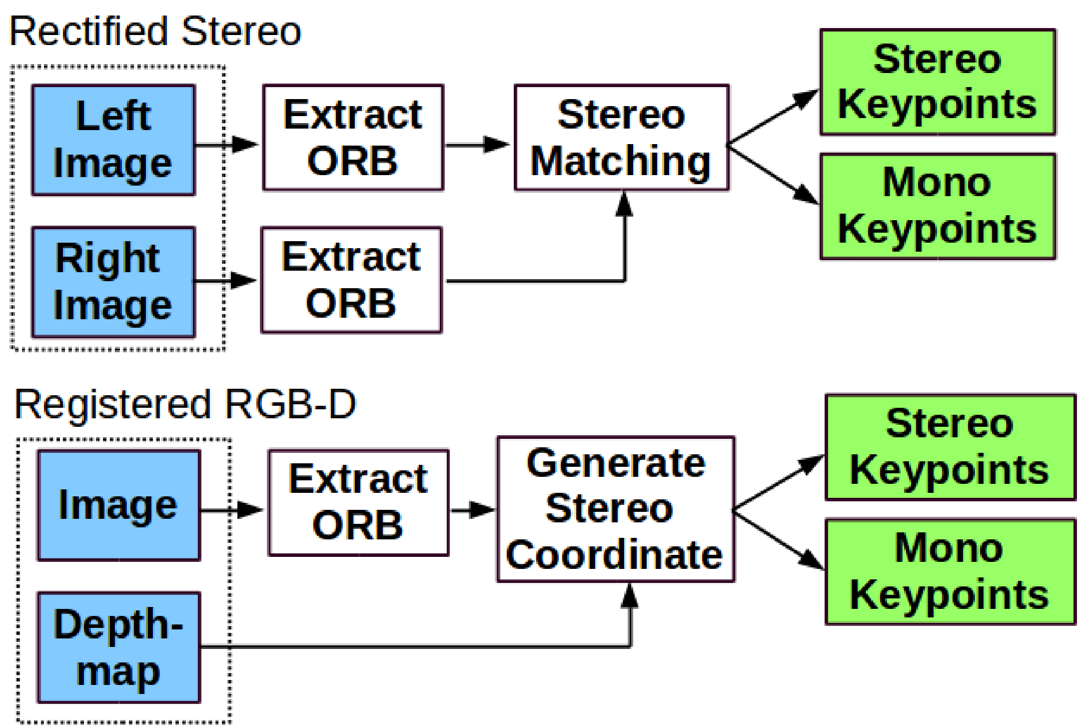
Scale drift!

Ground Truth
Estimation

6. ORB-SLAM2: Stereo and RGB-D



ORB-SLAM2: Input pre-processing



- ORB-SLAM2 is agnostic to the type of sensor

ORB-SLAM2: Monocular, Stereo and RGB-D

- Monocular:

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

- Stereo:

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

- RGB-D:

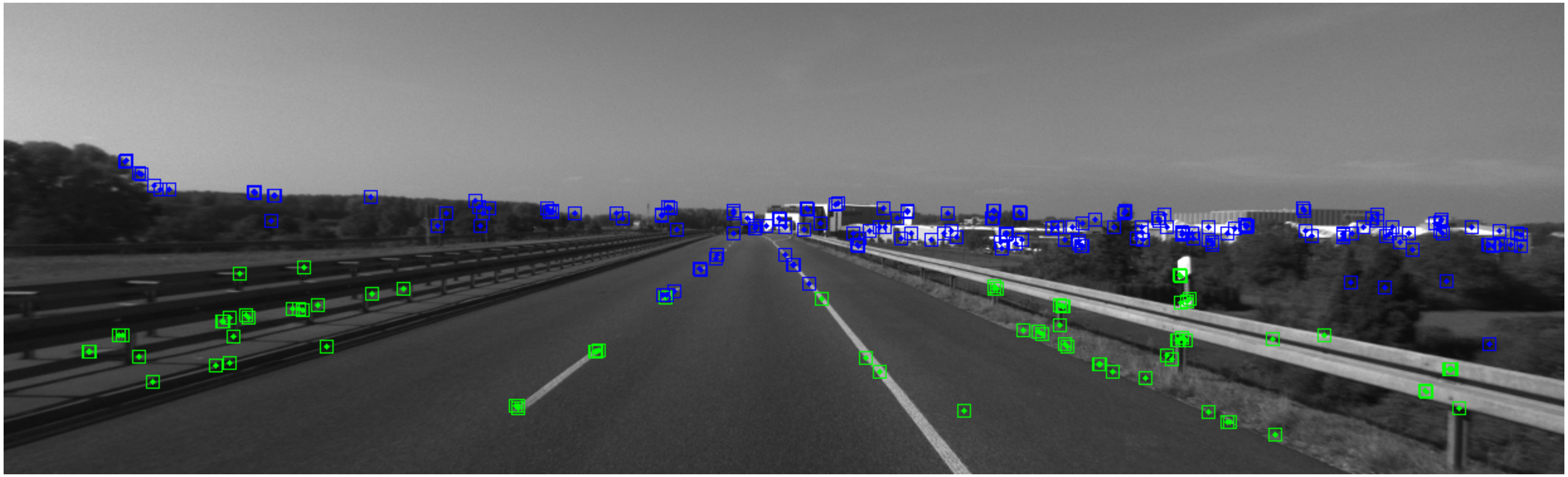
$$u_r = u - \frac{f_x b_{rgbd}}{d}$$

- BA:

$$\theta = \{\mathbf{X}^{\mathbf{j}}{}_{\mathtt{W}}, \mathbf{R}_{\mathtt{iW}}, {}_{\mathtt{i}}\mathbf{p}_{\mathtt{W}} \mid \forall j \in \mathcal{P}, \forall \mathtt{i} \in \mathcal{C}\}$$

$$\theta = \underset{\theta}{\operatorname{argmin}} \sum_{\mathtt{i},j} \rho \left(\left\| \mathbf{x}_{\mathtt{i}}^{j} - \pi_m \left(\mathbf{R}_{\mathtt{iW}} \mathbf{X}_{\mathtt{W}}^{j} + {}_{\mathtt{i}}\mathbf{p}_{\mathtt{W}} \right) \right\|_{\mathbf{\Sigma}_{\mathtt{i}}^{\mathtt{j}}}^{2} \right)$$

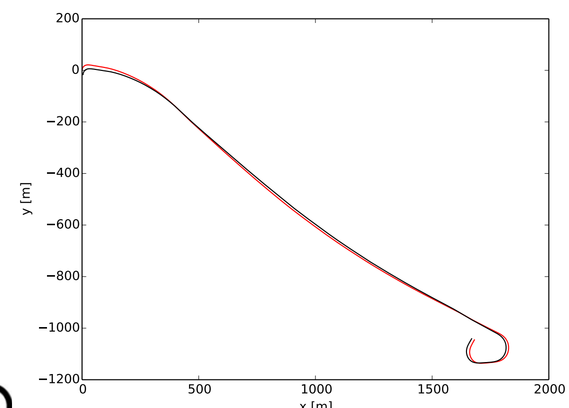
Close and Far Points

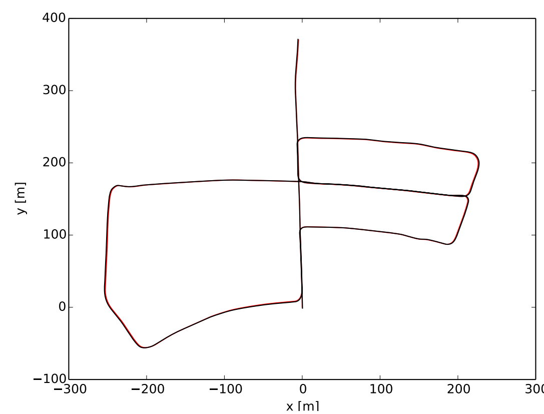


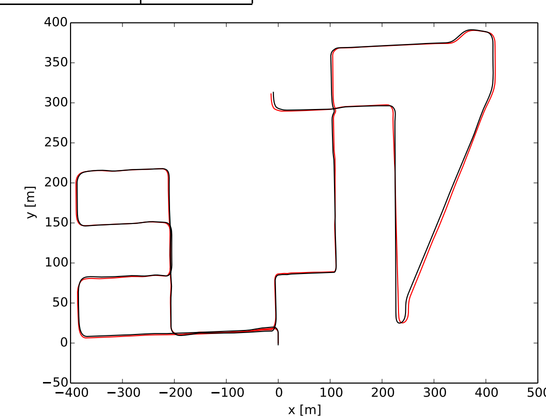
- Green points: depth <= 40 x baseline
  - Essential to compute camera translation
- Blue points: depth > 40 x baseline
  - Good to obtain camera orientation

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: Monocular, Stereo and RGB-D

Universidad Zaragoza

ROBOTICS UNIVERSIDAD DE ZARAGOZA

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

Dense Point Cloud Reconstruction

7. Visual-Inertial ORB-SLAM

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

$$\mathbf{R}_{\mathtt{WB}}^{k+1} = \mathbf{R}_{\mathtt{WB}}^{k} \operatorname{Exp}\left(\left(\boldsymbol{\omega}_{\mathtt{B}}^{k} - \boldsymbol{b}_{g}^{k}\right)\Delta t\right)$$

$${}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{k+1} = {}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{k} + \mathbf{g}_{\mathtt{W}}\Delta t + \mathbf{R}_{\mathtt{WB}}^{k}\left(\boldsymbol{a}_{\mathtt{B}}^{k} - \boldsymbol{b}_{a}^{k}\right)\Delta t$$

$${}_{\mathtt{W}}\mathbf{p}_{\mathtt{B}}^{k+1} = {}_{\mathtt{W}}\mathbf{p}_{\mathtt{B}}^{k} + {}_{\mathtt{W}}\mathbf{v}_{\mathtt{B}}^{k}\Delta t + \frac{1}{2}\mathbf{g}_{\mathtt{W}}\Delta t^{2} + \frac{1}{2}\mathbf{R}_{\mathtt{WB}}^{k}\left(\boldsymbol{a}_{\mathtt{B}}^{k} - \boldsymbol{b}_{a}^{k}\right)\Delta t^{2}$$

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

Visual-Inertial ORB-SLAM: IMU Initialization

Goal: Gravity, IMU Biases, Velocities, Scale

Divide and Conquer Solution

1. Run Monocular ORB-SLAM for 10-20s

Keyframe orientation and up-to-scale translation

2. Optimize Gyroscope Bias

Rotate accelerometer measurements

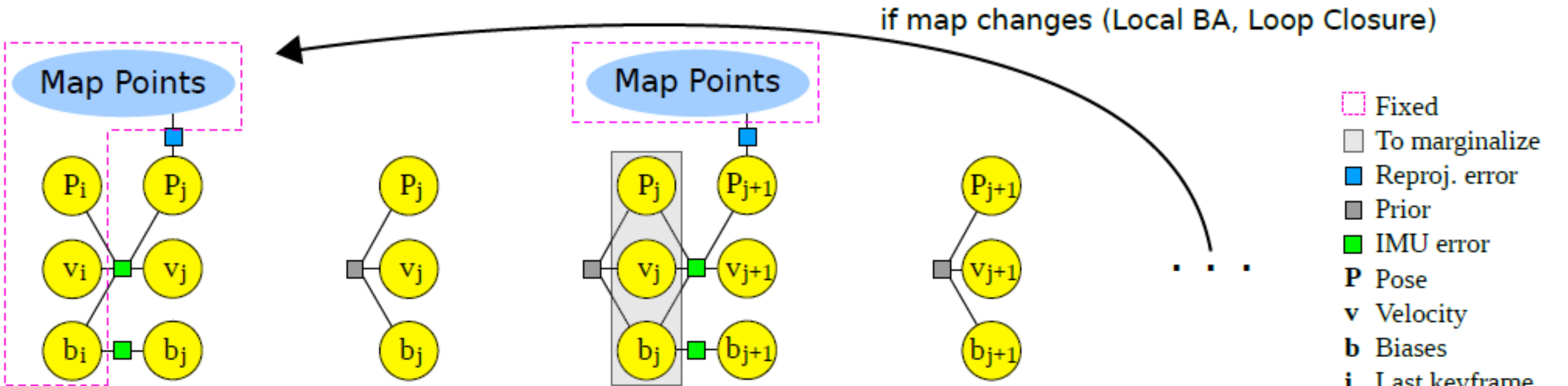
3. Estimate Gravity Vector (no Acc. Bias)

Initial seed for gravity direction

4. Optimize Gravity Direction, Acc. Bias and Scale

5. Compute Velocities

Visual-Inertial ORB-SLAM: Tracking



(a) Tracking Frame j (Map changed)

(b) Prior (optimization result)

(c) Tracking Frame j+1 (Map unchanged)

(d) Prior (marginalization)

. . .

$$\theta = \left\{ \mathrm{R}^j_{\mathtt{WB}}, {}_{\mathtt{W}}\mathrm{p}^j_{\mathtt{B}}, {}_{\mathtt{W}}\mathrm{v}^j_{\mathtt{B}}, \mathrm{b}^j_g, \mathrm{b}^j_a \right\}$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left(\sum_k \mathrm{E}_{\mathrm{proj}}(k, j) + \mathrm{E}_{\mathrm{IMU}}(i, j) \right)$$

$$\theta = \left\{ \mathrm{R}^j_{\mathtt{WB}}, \mathrm{p}^j_{\mathtt{W}}, \mathrm{v}^j_{\mathtt{W}}, \mathrm{b}^j_g, \mathrm{b}^j_a, \mathrm{R}^{j+1}_{\mathtt{WB}}, \mathrm{p}^{j+1}_{\mathtt{W}}, \mathrm{v}^{j+1}_{\mathtt{W}}, \mathrm{b}^{j+1}_g, \mathrm{b}^{j+1}_a \right\}$$

$$\theta^* = \underset{\theta}{\operatorname{argmin}} \left(\sum_k \mathrm{E}_{\mathrm{proj}}(k, j+1) + \mathrm{E}_{\mathrm{IMU}}(j, j+1) + \mathrm{E}_{\mathrm{prior}}(j) \right)$$

$$\mathrm{E}_{\mathrm{IMU}}(i, j) = \rho\left(\left[\mathrm{e}_R^T \, \mathrm{e}_v^T \, \mathrm{e}_p^T \right] \Sigma_I \left[\mathrm{e}_R^T \, \mathrm{e}_v^T \, \mathrm{e}_p^T \right]^T \right) + \rho\left(\mathrm{e}_b^T \Sigma_R \mathrm{e}_b \right)$$

$$\mathrm{e}_R = \operatorname{Log}\left(\left(\Delta\mathrm{R}_{ij} \operatorname{Exp}\left(\mathrm{J}^g_{\Delta R} \mathrm{b}^j_g \right) \right)^T \mathrm{R}^i_{\mathtt{BW}} \mathrm{R}^j_{\mathtt{WB}} \right)$$

$$\mathrm{e}_v = \mathrm{R}^i_{\mathtt{BW}} \left({}_{\mathtt{W}}\mathrm{v}^j_{\mathtt{B}} - {}_{\mathtt{W}}\mathrm{v}^i_{\mathtt{B}} - \mathrm{g}_{\mathtt{W}} \Delta t_{ij} \right) - \left(\Delta\mathrm{v}_{ij} + \mathrm{J}^g_{\Delta v} \mathrm{b}^j_g + \mathrm{J}^a_{\Delta v} \mathrm{b}^j_a \right)$$

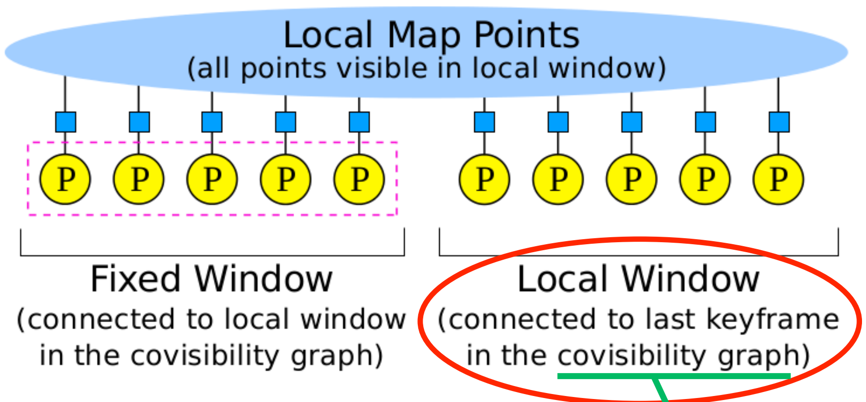
$$\mathrm{e}_p = \mathrm{R}^i_{\mathtt{BW}} \left({}_{\mathtt{W}}\mathrm{p}^j_{\mathtt{B}} - {}_{\mathtt{W}}\mathrm{p}^i_{\mathtt{B}} - {}_{\mathtt{W}}\mathrm{v}^i_{\mathtt{B}} \Delta t_{ij} - \frac{1}{2} \mathrm{g}_{\mathtt{W}} \Delta t^2_{ij} \right) - \left(\Delta\mathrm{p}_{ij} + \mathrm{J}^g_{\Delta p} \mathrm{b}^j_g + \mathrm{J}^a_{\Delta p} \mathrm{b}^j_a \right)$$

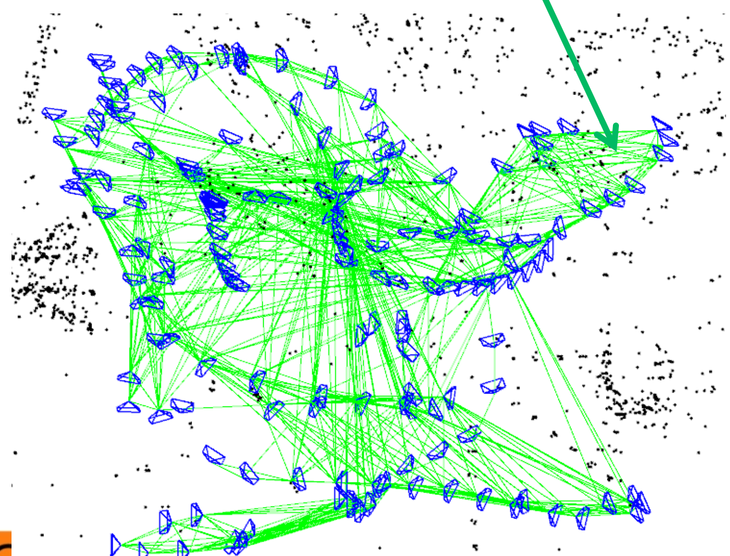
$$\mathrm{e}_b = \mathrm{b}^j - \mathrm{b}^i$$

Visual-Inertial ORB-SLAM: Mapping

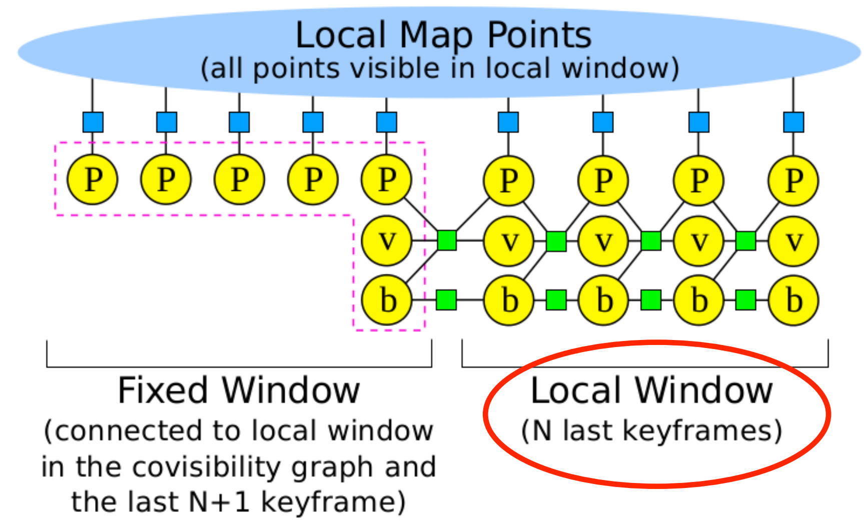
Local Bundle Adjustment

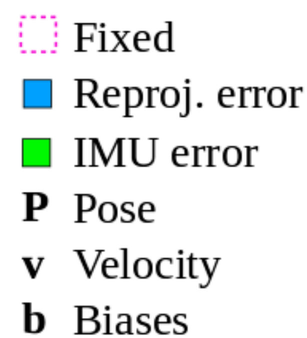
ORB-SLAM's Local BA





Visual-Inertial ORB-SLAM's Local BA





Visual-Inertial ORB-SLAM: Results

Universidad Zaragoza 1542

ROBOTICS UNIVERSIDAD DE ZARAGOZA

Instituto Universitario de Investigación en Ingeniería de Aragón Universidad Zaragoza

Visual-Inertial Monocular SLAM with Map Reuse

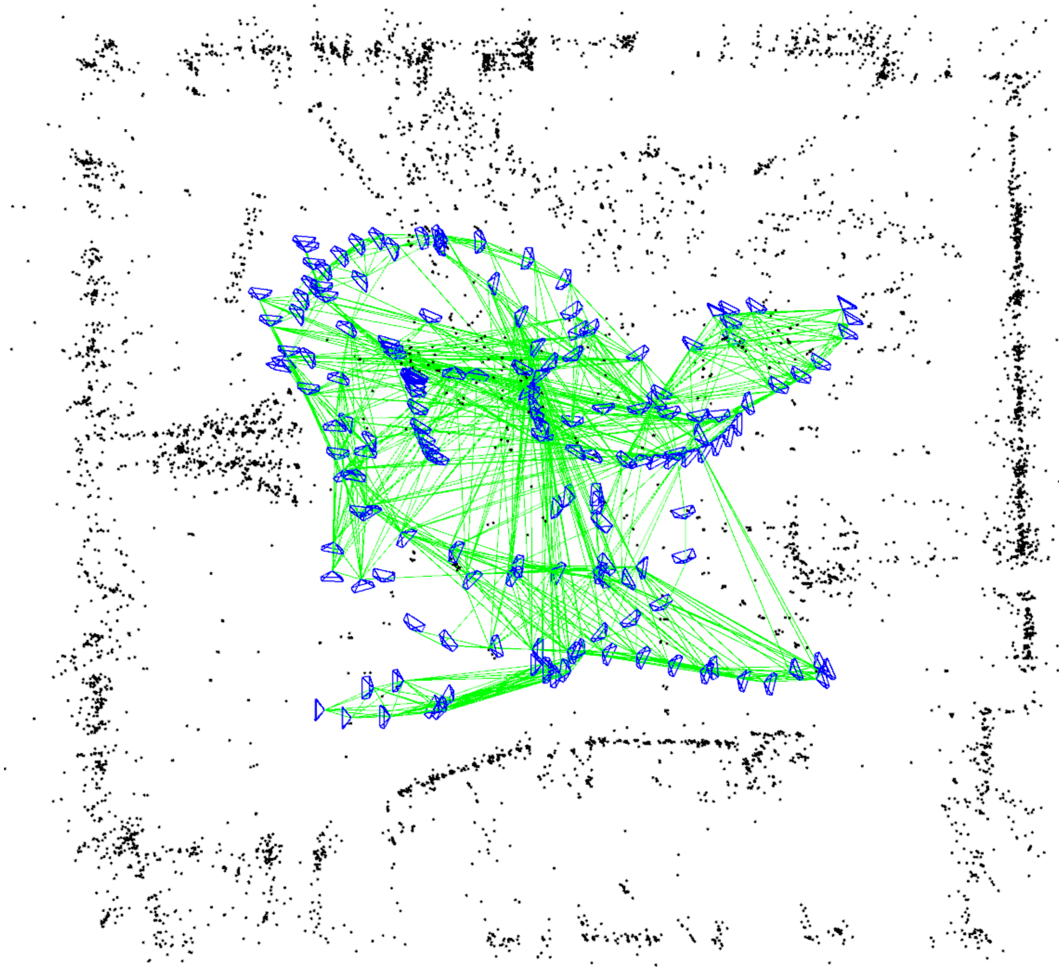
Raúl Mur-Artal and Juan D. Tardós

Visual-Inertial ORB-SLAM

Sequence: V1_02_medium
Dataset: EuRoC MAV Dataset

True scale (1% error) and centimeter precision

Results on EuRoC dataset



Visual-Inertial Odometry: Keeps accumulating drift

Visual-Inertial ORB-SLAM: Zero drift in mapped areas

Visual-Inertial ORB-SLAM: Results

TABLE I
EUROC DATASET. COMPARISON OF TRANSLATION RMSE (m).

Sequence	ORB-SLAM Monocular (with GT scale)	ORB-SLAM Visual Inertial	ORB-SLAM2 Stereo	LSD-SLAM Stereo
V1_01_easy	0.015	0.027	0.035	0.066
V1_02_medium	0.020	0.028	0.020	0.074
V1_03_difficult	X	X	0.048	0.089
V2_01_easy	0.021	0.032	0.037	-
V2_02_medium	0.018	0.041	0.035	-
V2_03_difficult	X	0.074	X	-
MH_01_easy	0.071	0.075	0.035	-
MH_02_easy	0.067	0.084	0.018	-
MH_03_medium	0.071	0.087	0.028	-
MH_04_difficult	0.082	0.217	0.119	-
MH_05_difficult	0.060	0.082	0.060	-

Raúl Mur-Artal, Juan D. Tardós,
ORB-SLAM2: An Open-Source SLAM System for Monocular, Stereo and RGB-D cameras,
IEEE Trans. on Robotics, Oct. 2017

Summary

- Monocular: excellent accuracy, but scale?
- Stereo: excellent accuracy and robustness
- Tightly-coupled Visual-Inertial SLAM
  - Recovers the true scale within 1% of error
- SLAM allows loop closing and map reuse
  - More accurate than Visual Odometry

- Future work:
  - Visual-inertial stereo SLAM
  - Direct SLAM
  - Deformable SLAM

More Information

- 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
- Raúl Mur-Artal, and Juan D. Tardós.
  Visual-Inertial Monocular SLAM with Map Reuse
  IEEE Robotics and Automation Letters 2(2): 798-803, Jan 2017
- Carlos Campos, José M. M. Montiel, Juan D. Tardós
  Fast and Robust Initialization for Visual-Inertial SLAM
  IEEE Int. Conf. Robotics and Automation, May 2019

- https://github.com/uz-slamlab

https://github.com/uz-slamlab