Large Scale SLAM

Lina Maria Paz
University of Zaragoza
Spain

linapaz@unizar.es

Joint work with: Pedro Piniés, José Neira, Juan D. Tardós

Simultaneous Localization and Mapping

Is it possible to use a vehicle, starting at an

unknown initial location, in an

unknown environment,

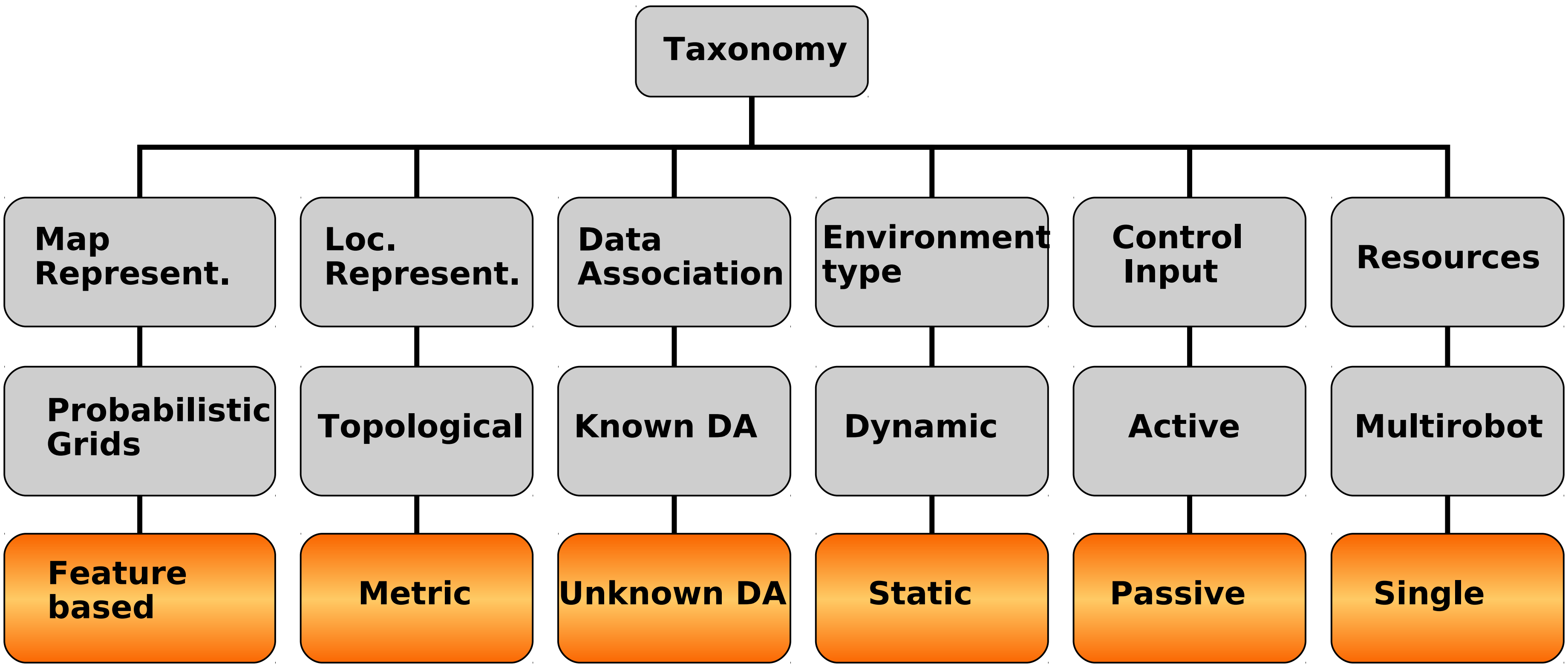
to **incrementally** build a map of the environment,

and **at the same time** use the map to determine the vehicle location?

(image: Paul Newman)

Chicken and egg problem?

Let's put things in context



Let's put things in context

Paradigms

EKF	Graph-based Optimization	Particle Filters
CLSF [Williams 2001]	Graph SLAM [Thrun 2006]	Rao-Blackwellized [Murphy, 2001]
Hierarchical SLAM [Estrada et al, 2005]	SAM [Dellaert 2001]	Fast SLAM [Montemerlo et al, 2003]

Outline

1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

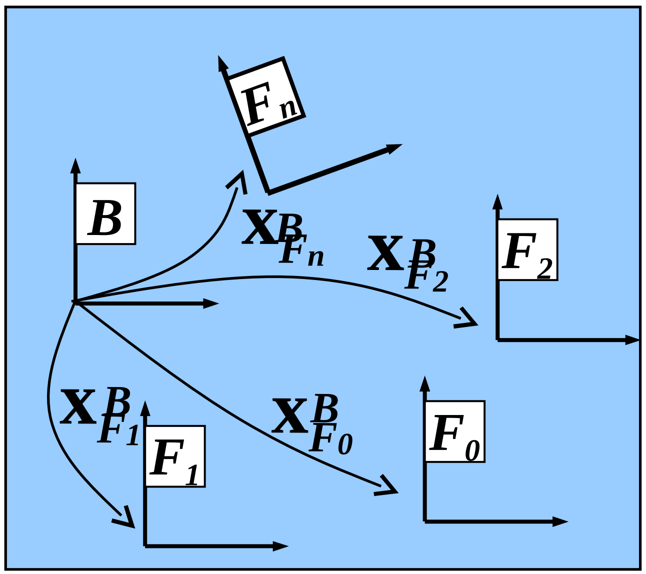
Outline

1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

SLAM approach

- Environment information related to a set of elements: $\mathcal{F} = \{B,\ F_0,\ F_1,\ \ldots,\ F_n\}$ F_0 = Robot

- represented by a map: $\mathcal{M}_{\mathcal{F}}^{B} = \left(\hat{\mathbf{x}}_{\mathcal{F}}^{B}, \mathbf{P}_{\mathcal{F}}^{B}\right)$

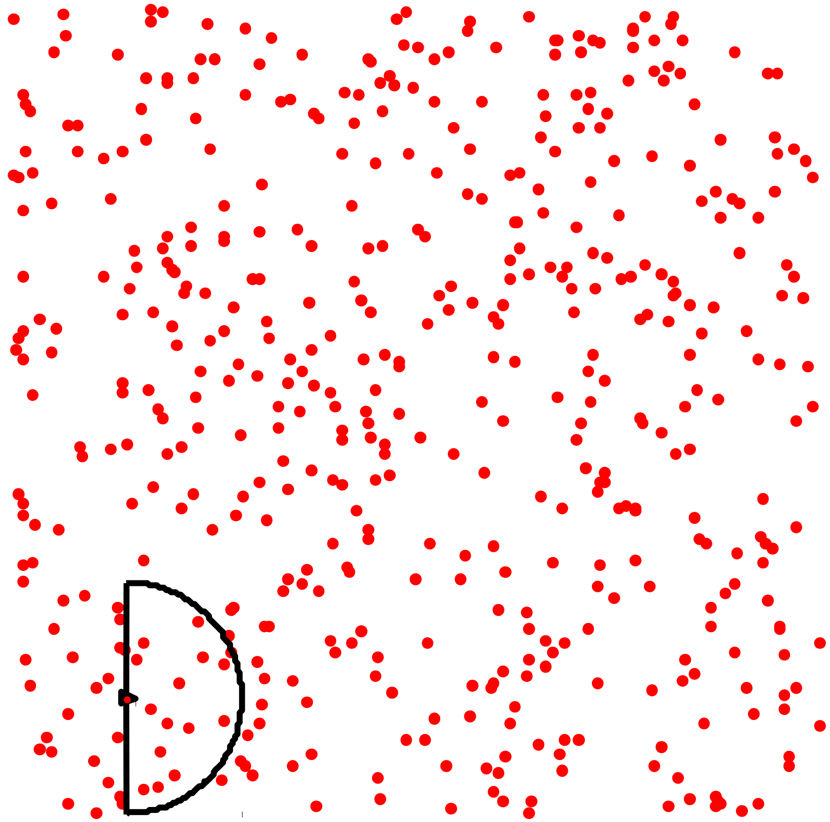


$$\hat{\mathbf{x}}_{\mathcal{F}}^{B} = \begin{bmatrix} \hat{\mathbf{x}}_{F_0}^{B} \\ \vdots \\ \hat{\mathbf{x}}_{F_n}^{B} \end{bmatrix}$$

$$\mathbf{P}_{\mathcal{F}}^{B} = \begin{bmatrix} \mathbf{P}_{F_0F_0}^{B} & \cdots & \mathbf{P}_{F_0F_n}^{B} \\ \vdots & \ddots & \vdots \\ \mathbf{P}_{F_nF_0}^{B} & \cdots & \mathbf{P}_{F_nF_n}^{B} \end{bmatrix}$$

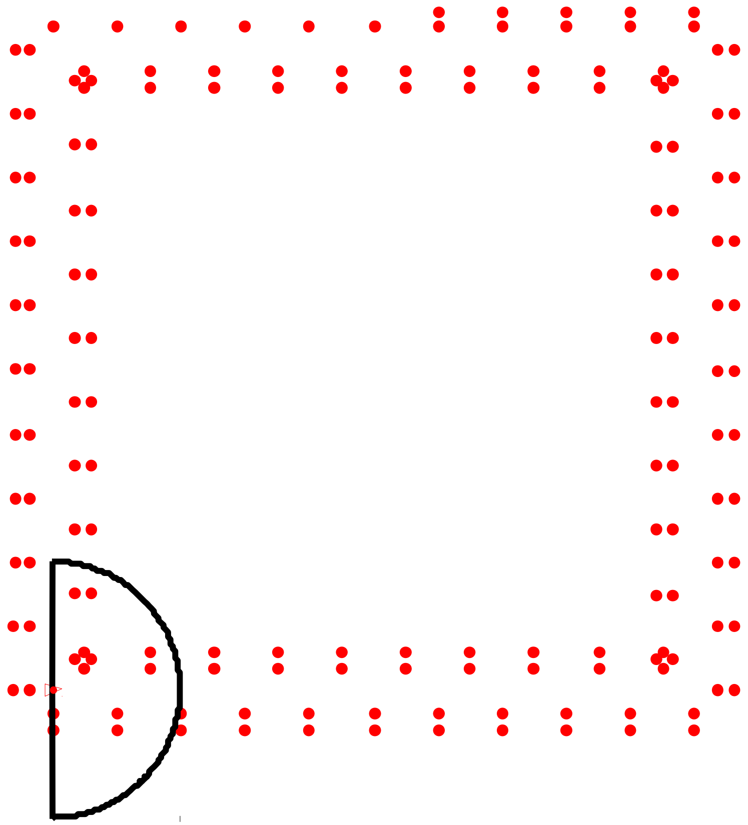
Without loss of generality...

- Environment to be mapped has more or less uniform density of features



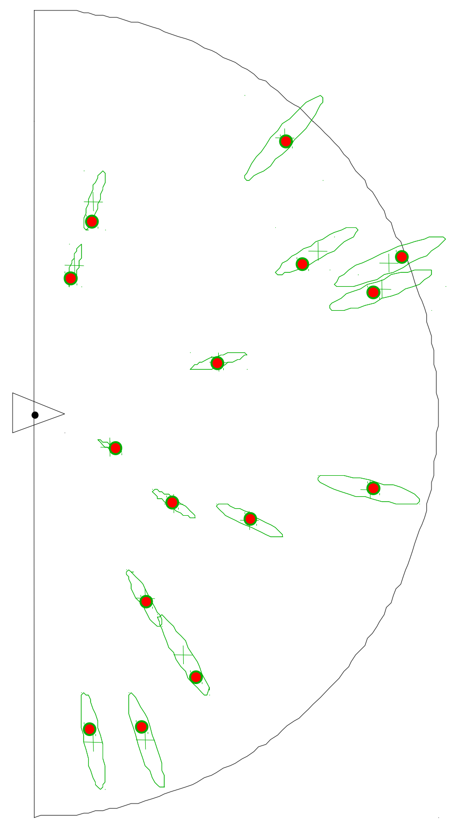
Without loss of generality...

- Environment to be mapped has more or less uniform density of features



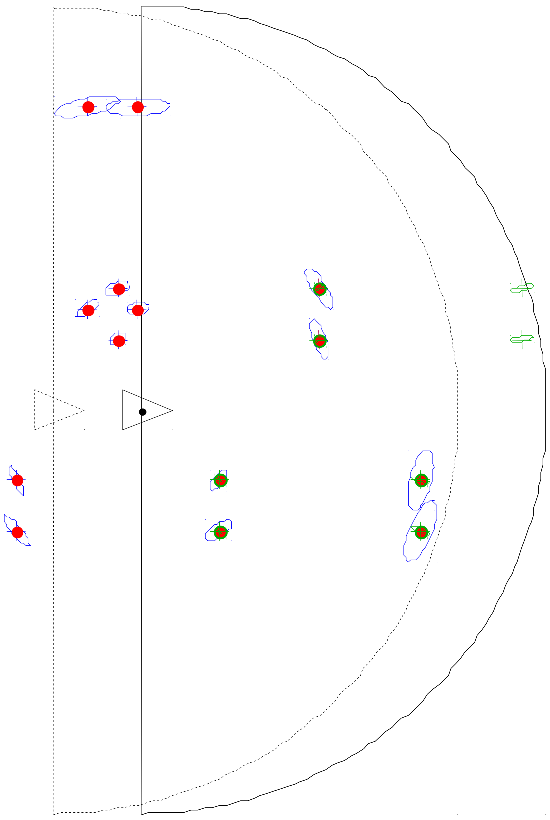
Without loss of generality...

- Onboard range and bearing sensor obtains ***m*** measurements



Without loss of generality...

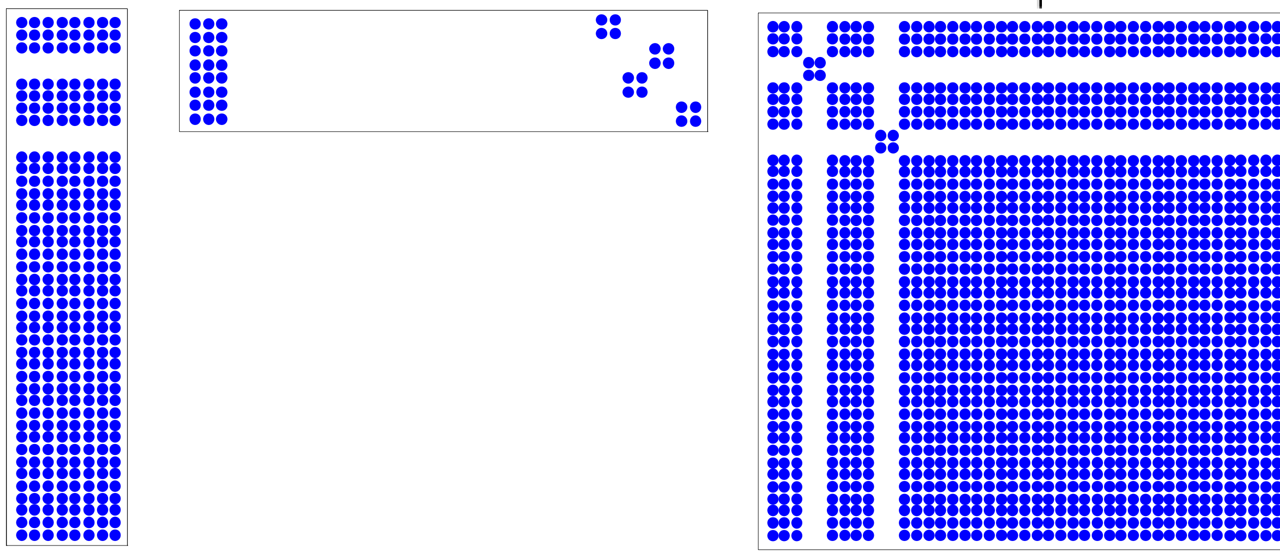
- Vehicle performs an exploratory trajectory, re-observing ***r*** features, and seeing ***s = m – r*** new features.



The EKF-covariance matrix update

$$\mathbf{P}_k = (\mathbf{I} - \mathbf{K}_k \mathbf{H}_k)\mathbf{P}_{k|k-1}$$

..... $=$ ..... $\mathbf{K}_k$ $\mathbf{H}_k$ $\mathbf{P}_{k|k-1}$



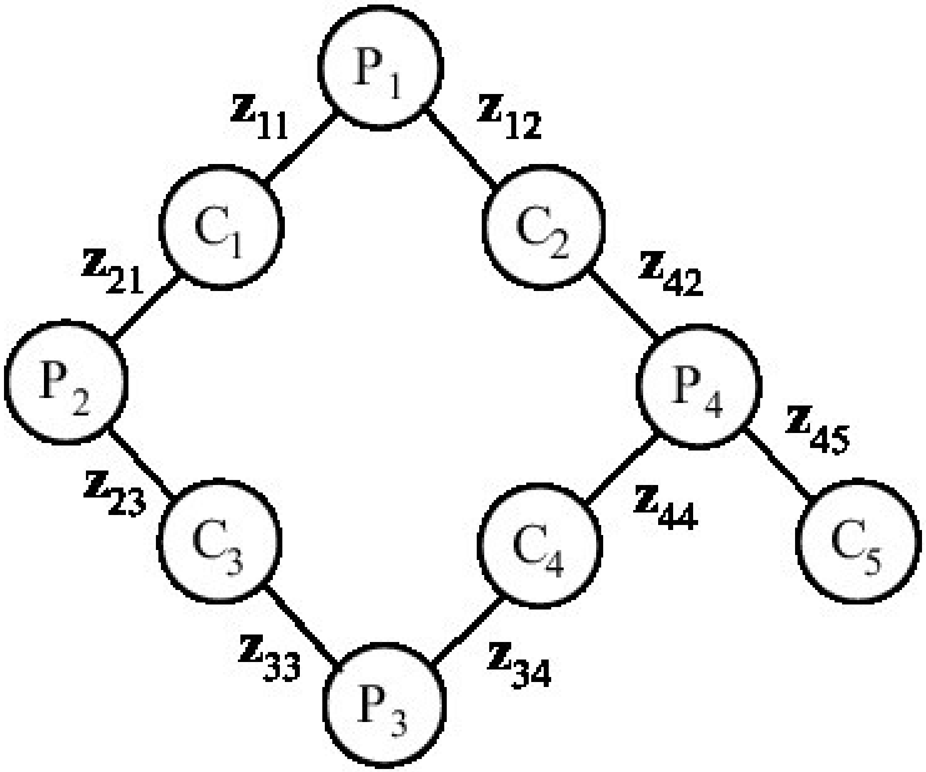
....

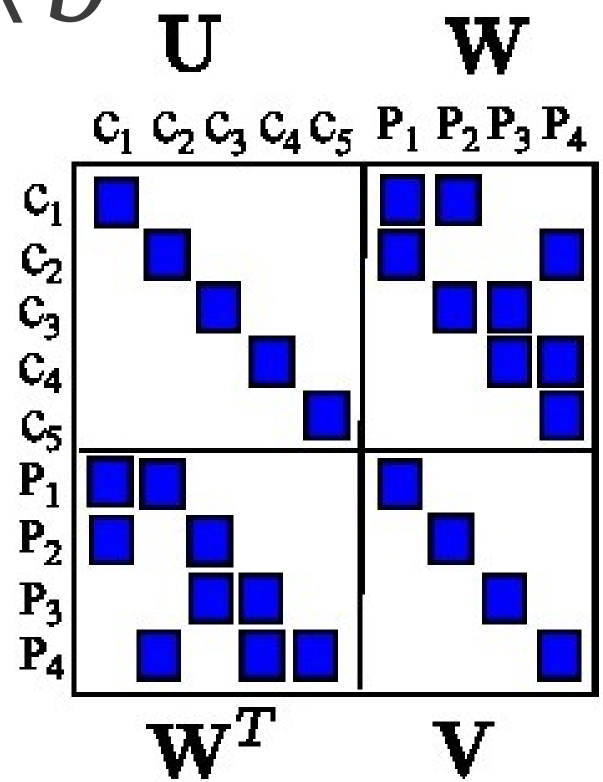
nxr *cxn* *nxn*

EKF update step is *O(n²)*

BA Primary structure

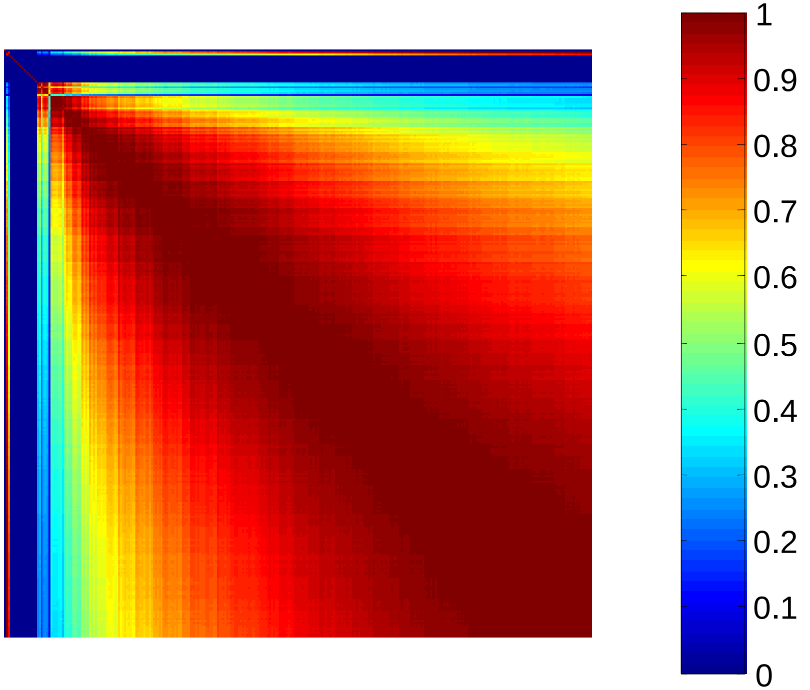
$$x = A \backslash b$$





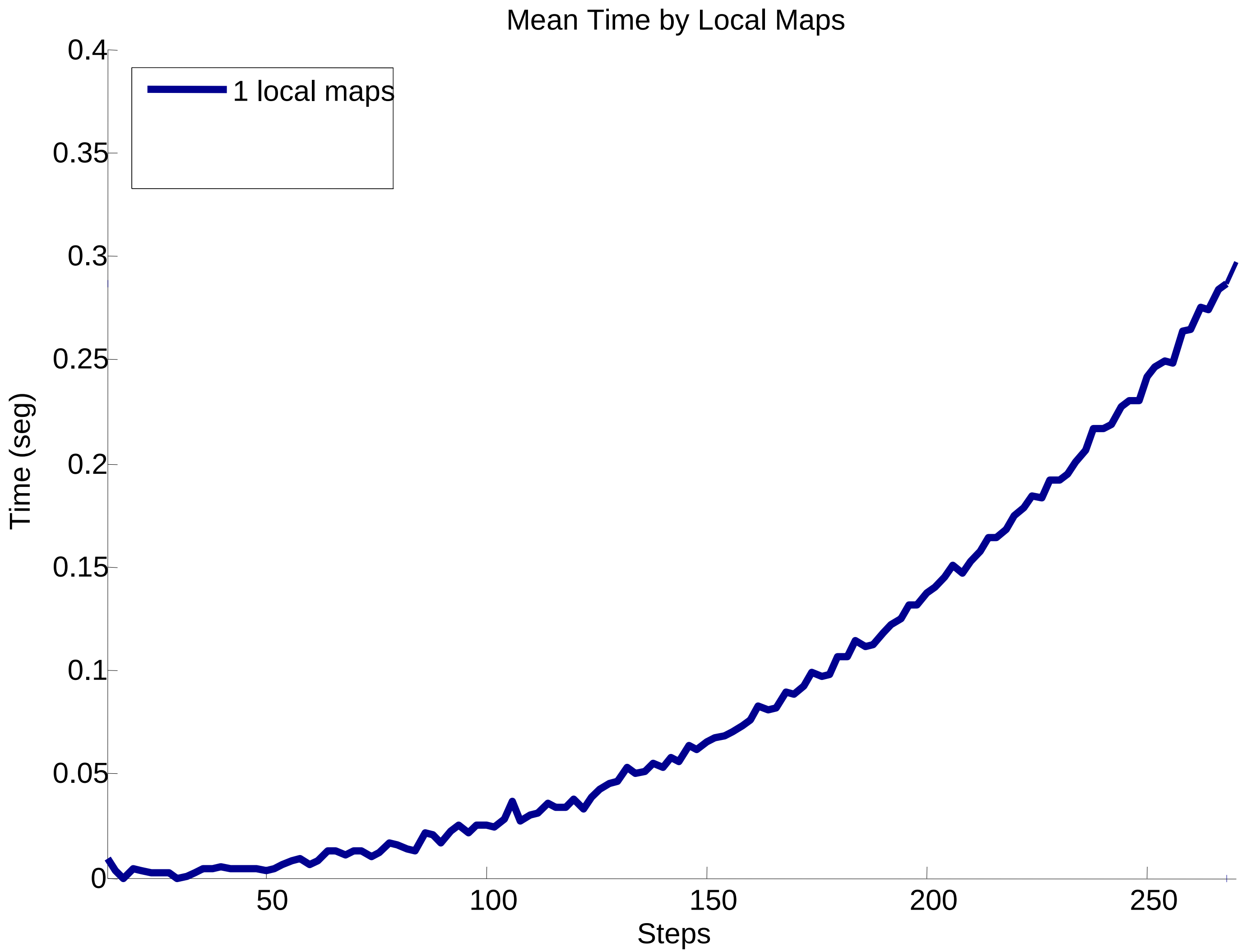
BA solution is *O(n+p)* ? *O((n+p)^3)*

The mixed blessing of Covariance

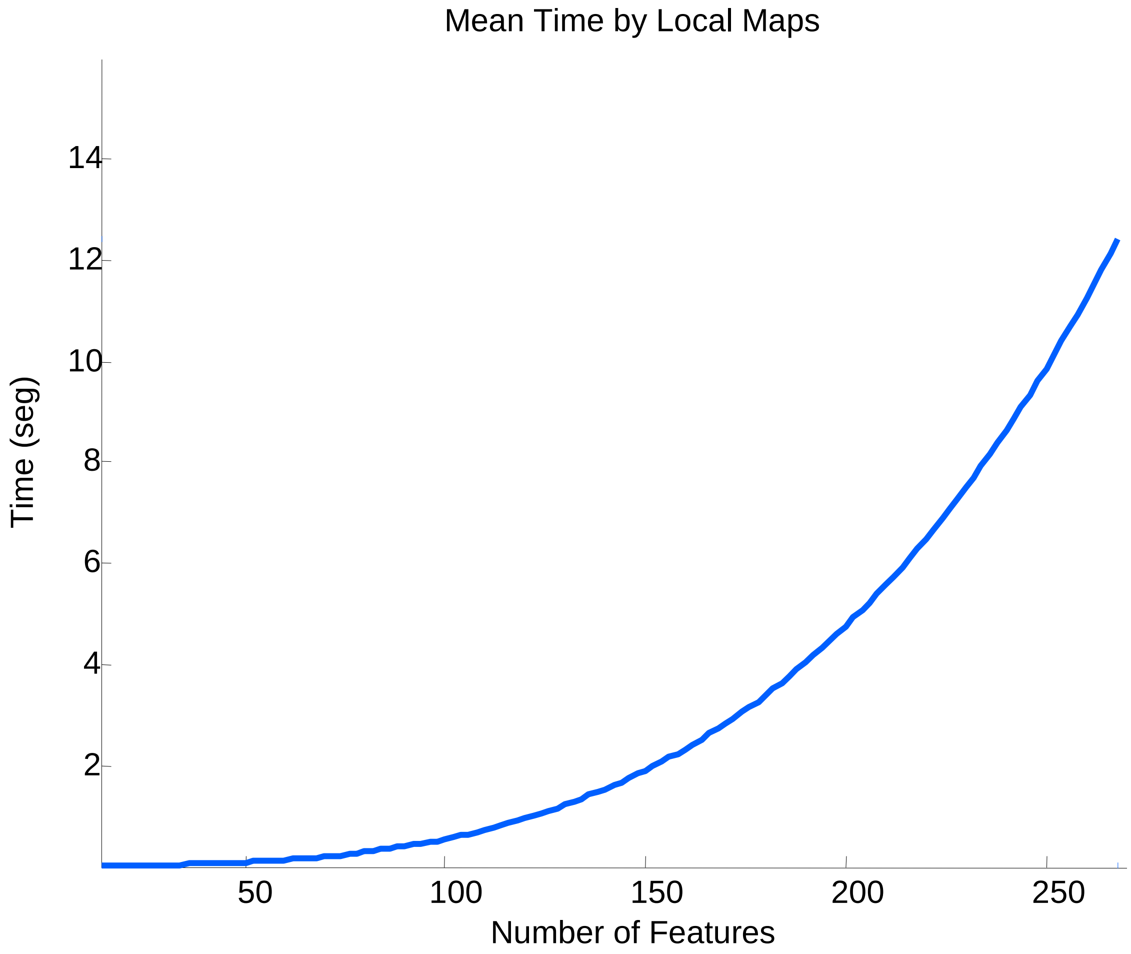


- Covariance provides data association based on statistical tests
- Covariance-based criteria speeds up convergence
- But the covariance matrix is full (e.g. EKF)

EKF-SLAM updates are $O(n^2)$

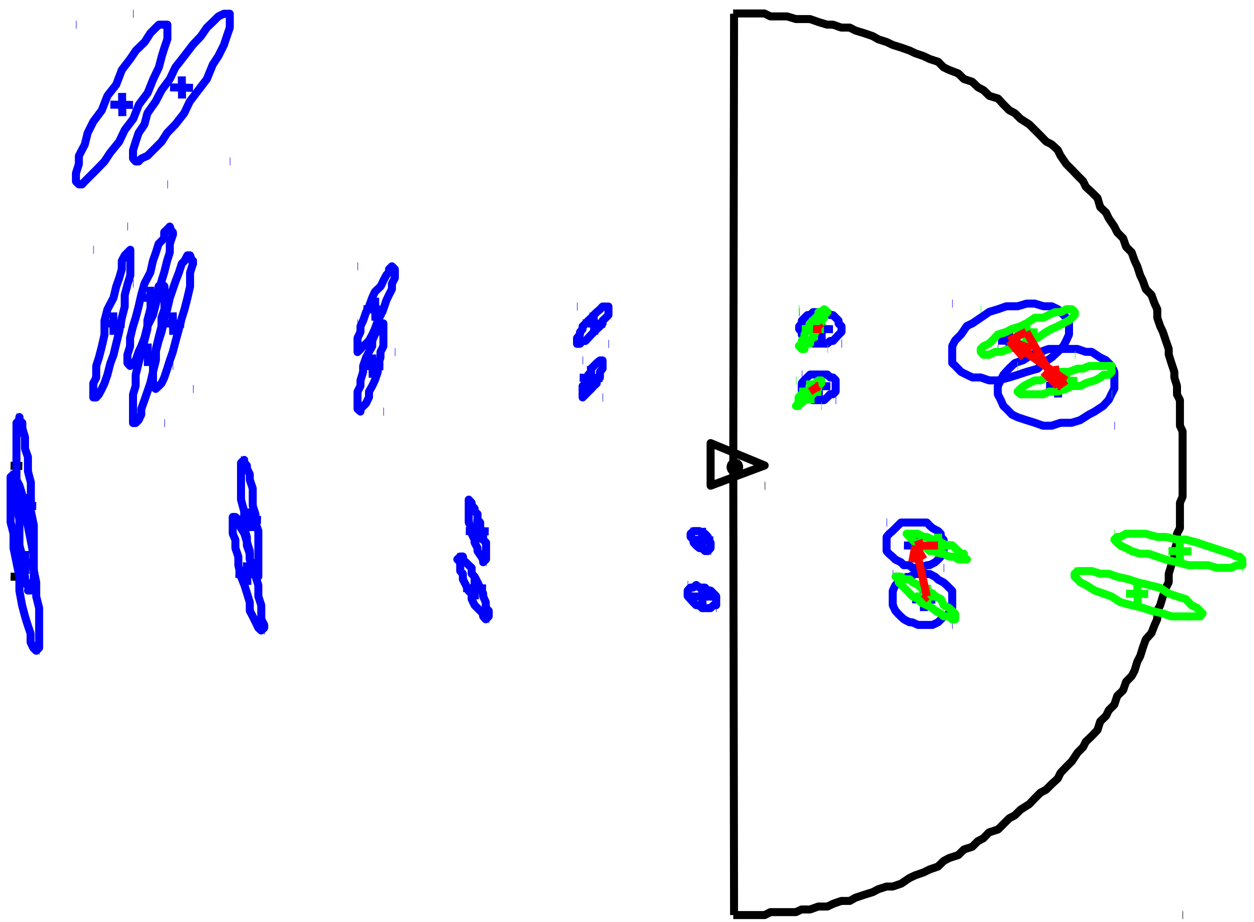


Total cost of EKF-SLAM



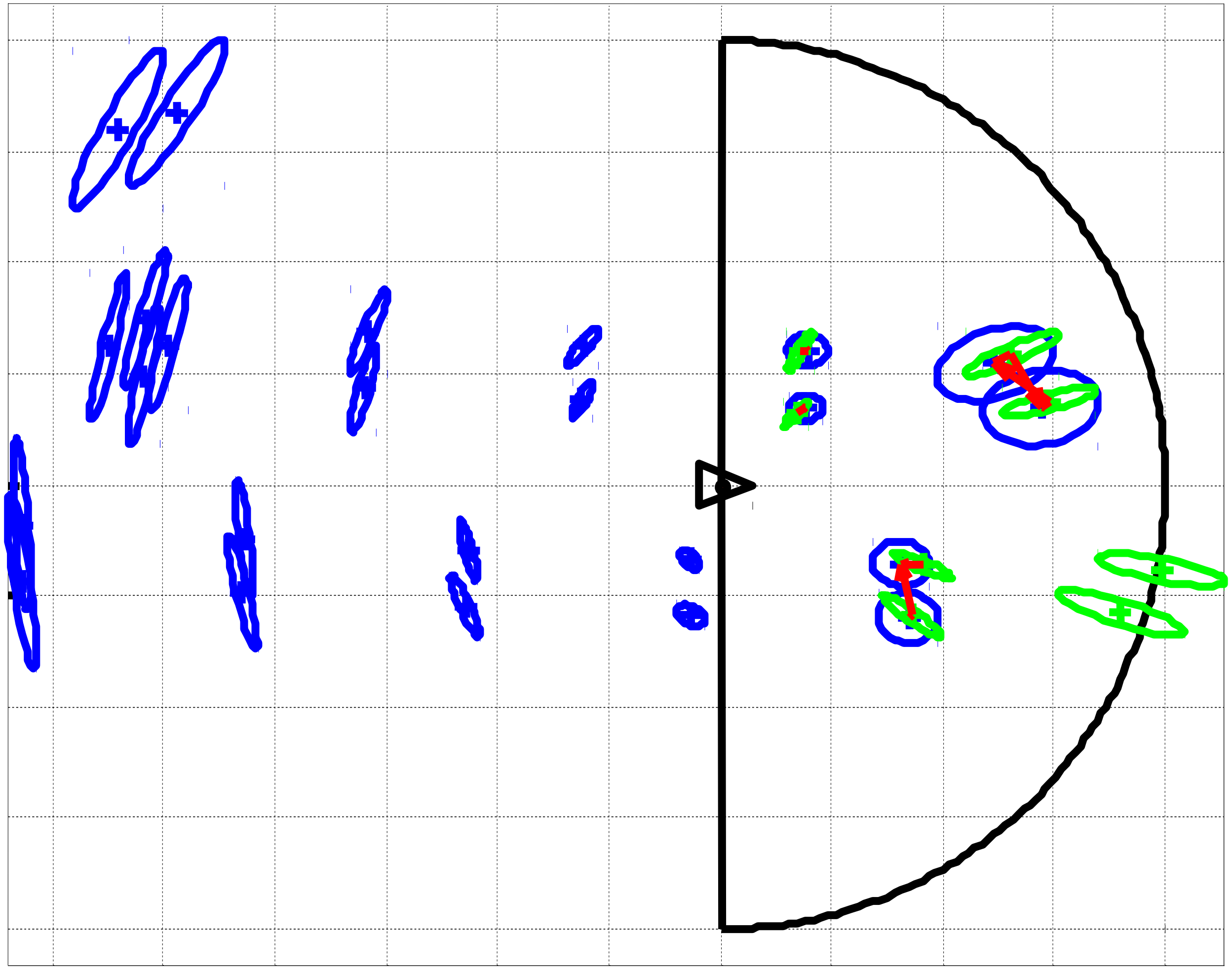
Total SLAM is $O(n^3)$

Continuous data association



Individual compatibility is $O(nm) = O(n)$

Map tessellation



Individual compatibility can be *O(1)*

Efforts to reduce complexity

- Decoupled Stochastic Mapping (Leonard and Feder, 2000) (Jensfelt 2001) ***O(1)***
- Suboptimal SLAM (Guivant and Nebot 2001) ***O(n)***
- Sparse Weight Filter (Julier 2001) ***O(n)***
- Sparse Information Filter (Thrun et al 2004) ***O(1) amort.***

Aproximate, or pessimistic solutions

- Postponement (Knight, Davidson and Reed 2001)
- Compressed Filter (Guivant and Nebot 2001)
- Constrained Local Submap Filter (Williams 2001)
- Map Joining (Tardós et al, 2002)

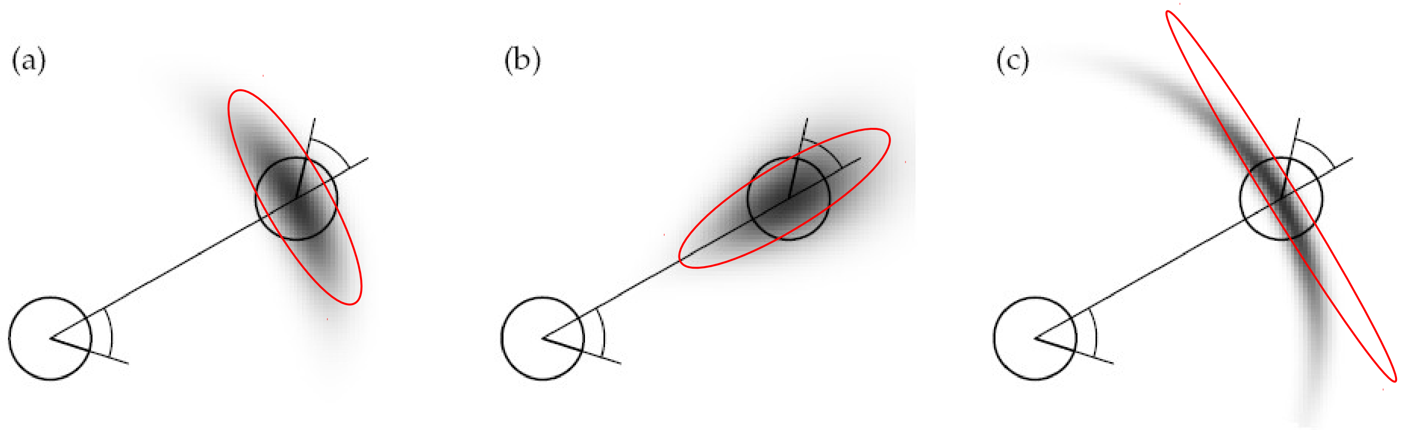
Exact solutions that delay global map updating, and strongly reduce cost.

But still *O(n^2)*

Consistency of EKF-SLAM

$$\mathbf{x}_{R_k}^{R_{k-1}} = (0,0,\phi_1)^t \oplus (x_1,0,0)^t \oplus (0,0,\phi_2)^t$$

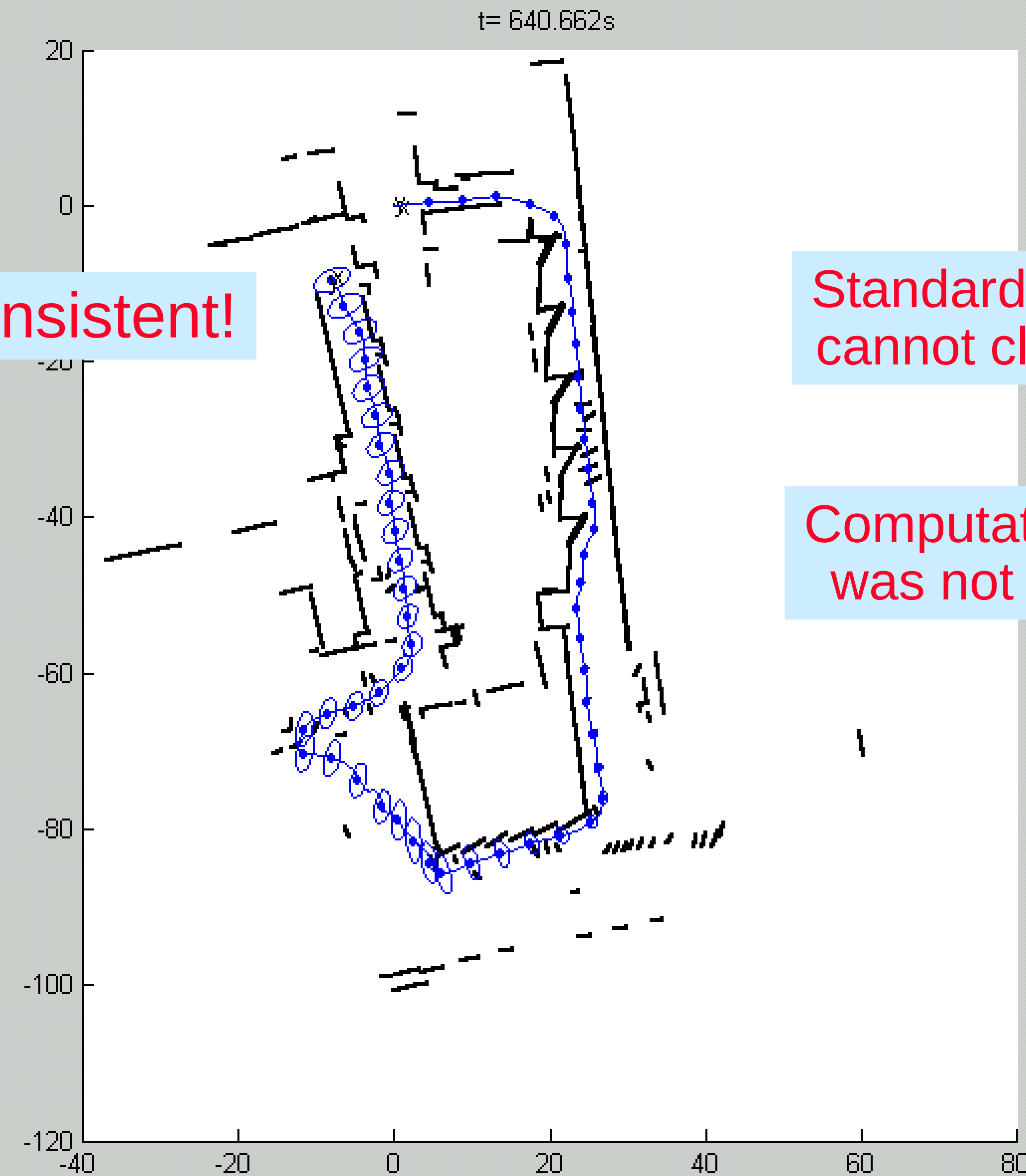
We are linearizing errors!



(image: Thrun, Burgard, Fox)

EKF-SLAM: Real Example

EKF-SLAM: Real Example



Inconsistent!

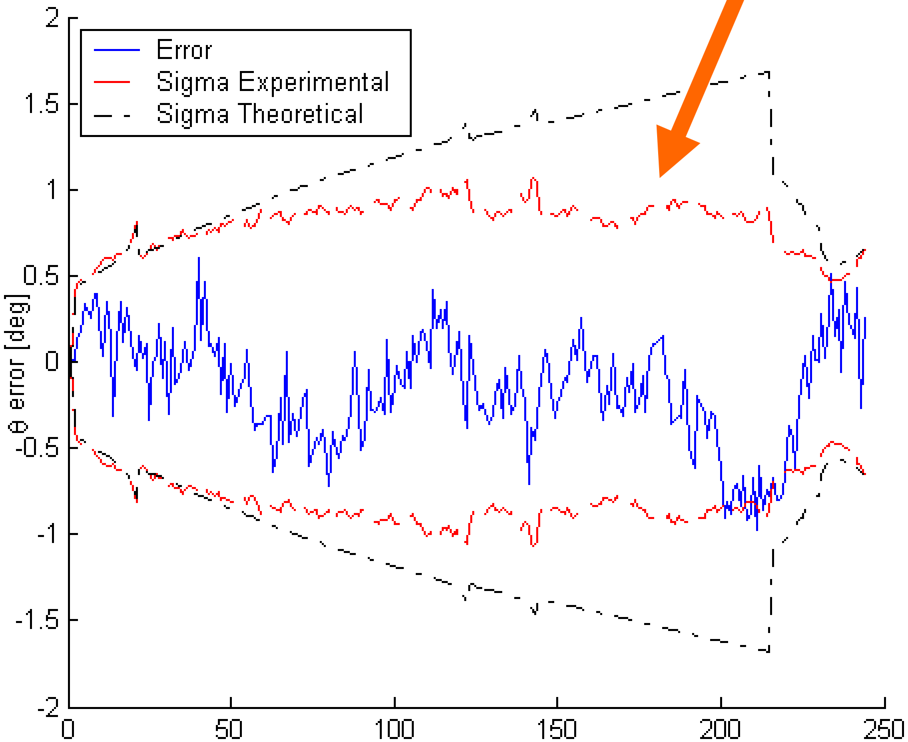
Standard data association cannot close a 250m loop

Computational complexity was not a problem here!

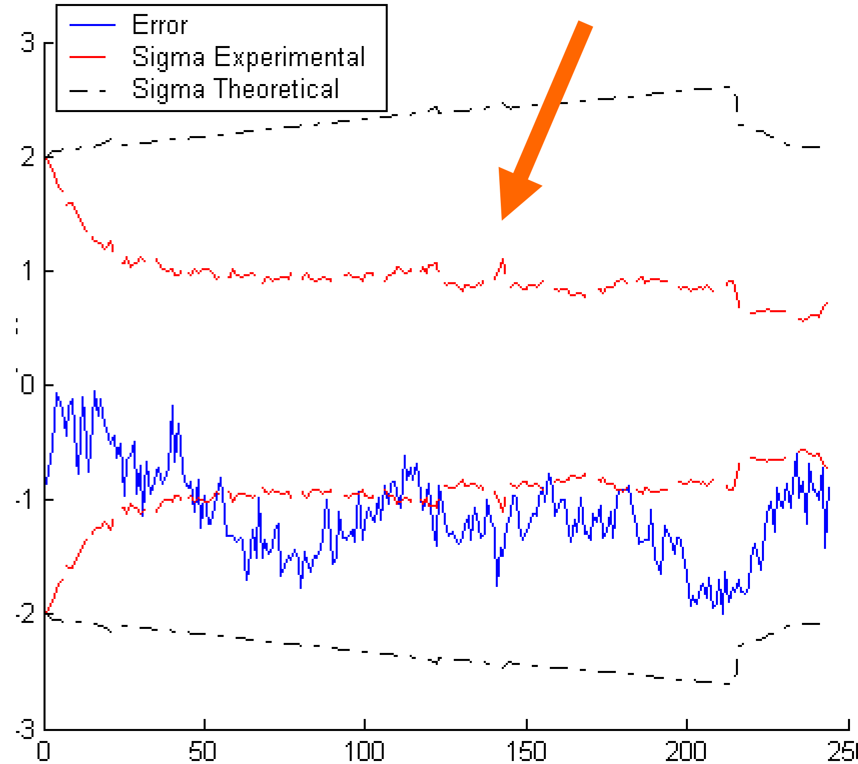
EKF-SLAM: Covariance

Optimistic

Violates lower bound



Initial uncertainty = 0



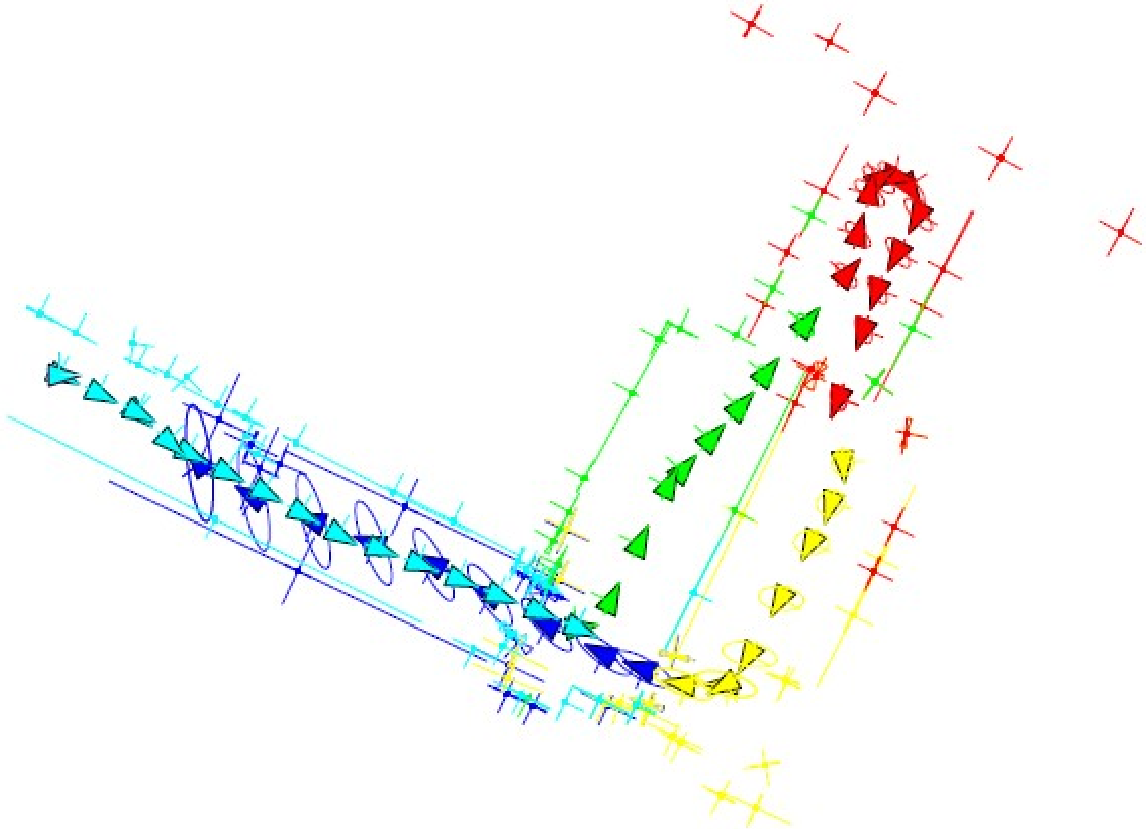
Initial uncertainty > 0

J.A. Castellanos, J. Neira, J.D. Tardós, **Limits to the Consistency of EKF-based SLAM**, 5th IFAC Symposium on Intelligent Autonomous Vehicles, Lisbon, July 2004

Outline

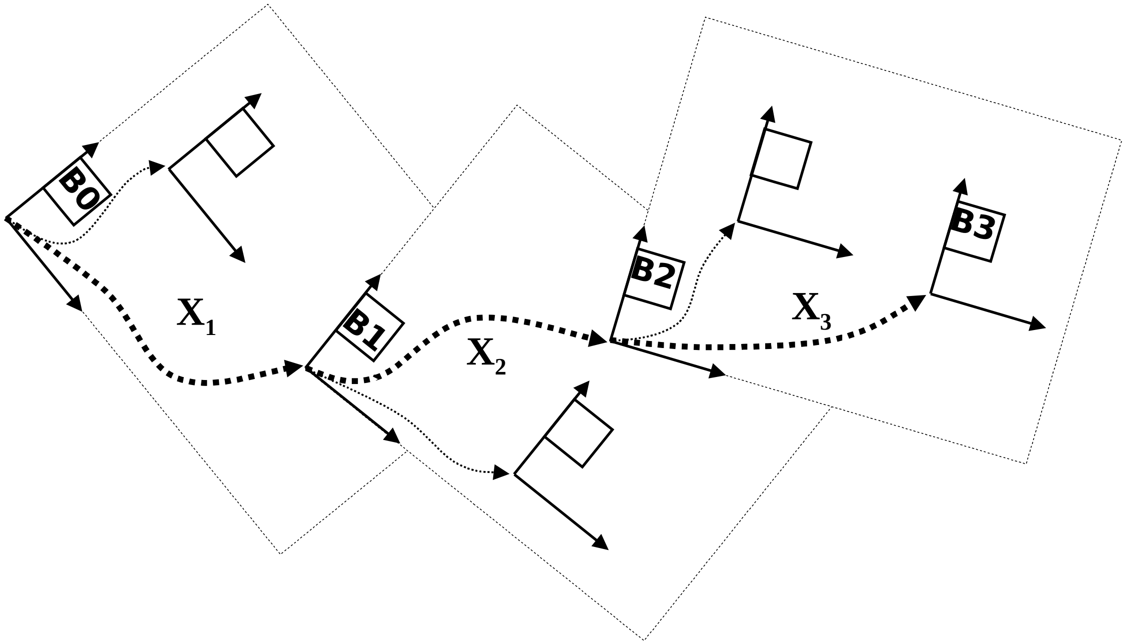
1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

Scalable EKF SLAM: Independent Local Maps



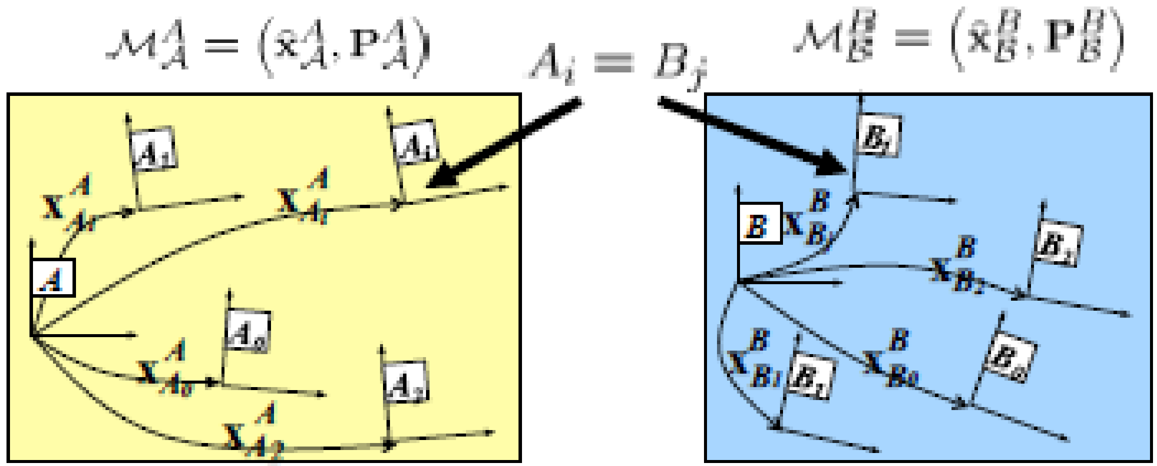
- Local Map Joining (Tardós et. al. 2002)
- Atlas (Bosse et. al. 2003)
- Constant time SLAM (Newman et. al 2003)

Scalable EKF SLAM: Independent Local Maps

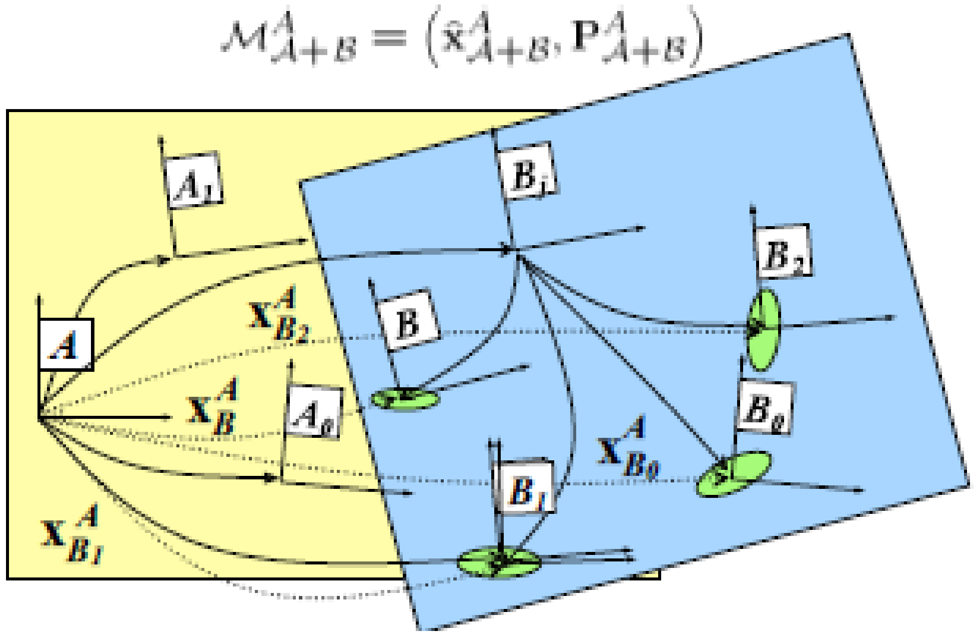


- Local Map Joining (Tardós et. al. 2002)
- Atlas (Bosse et. al. 2003)
- Constant time SLAM (Newman et. al 2003)

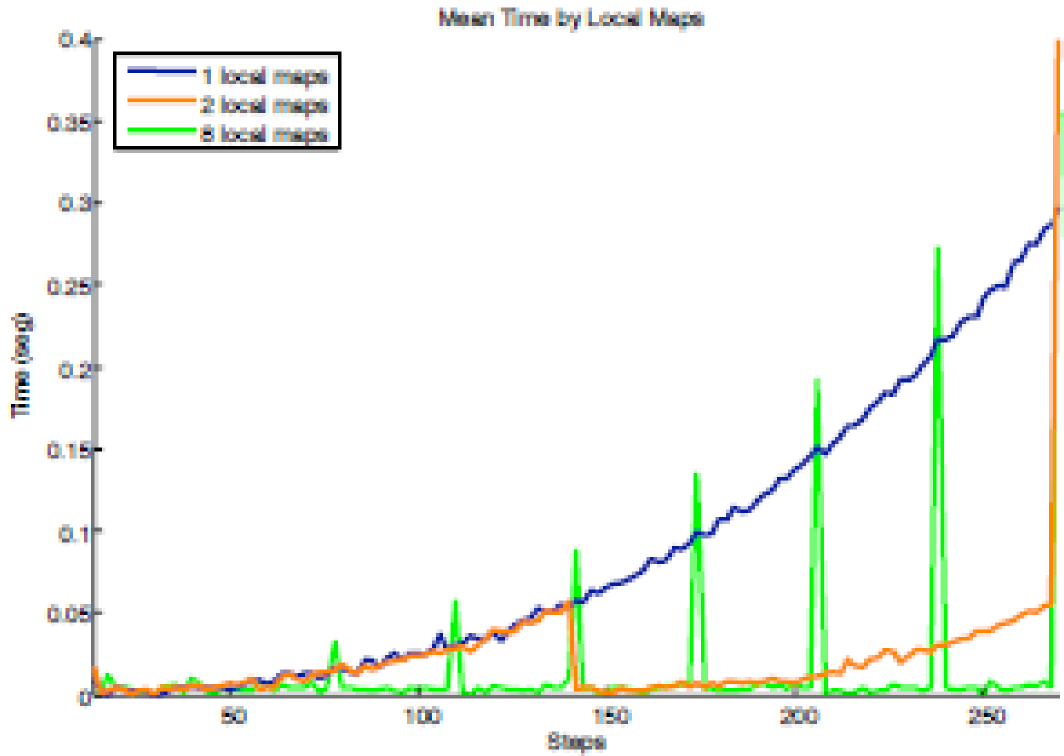
Scalable EKF SLAM: Independent Local Maps



Scalable EKF SLAM: Independent Local Maps

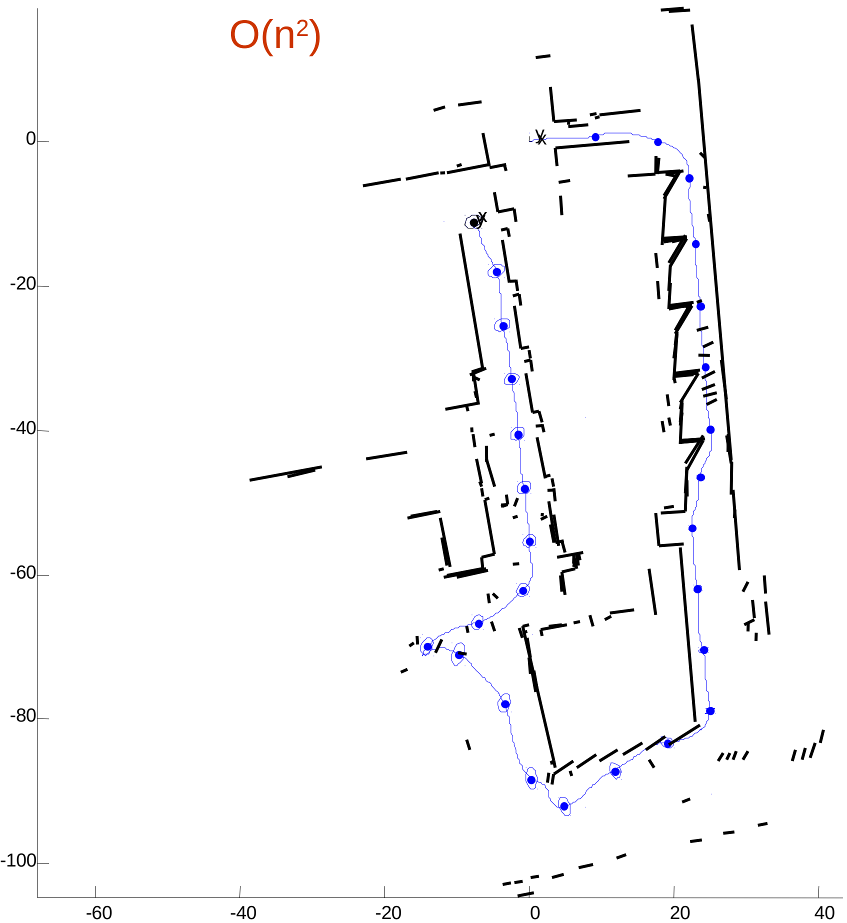


Local Maps Improve Complexity

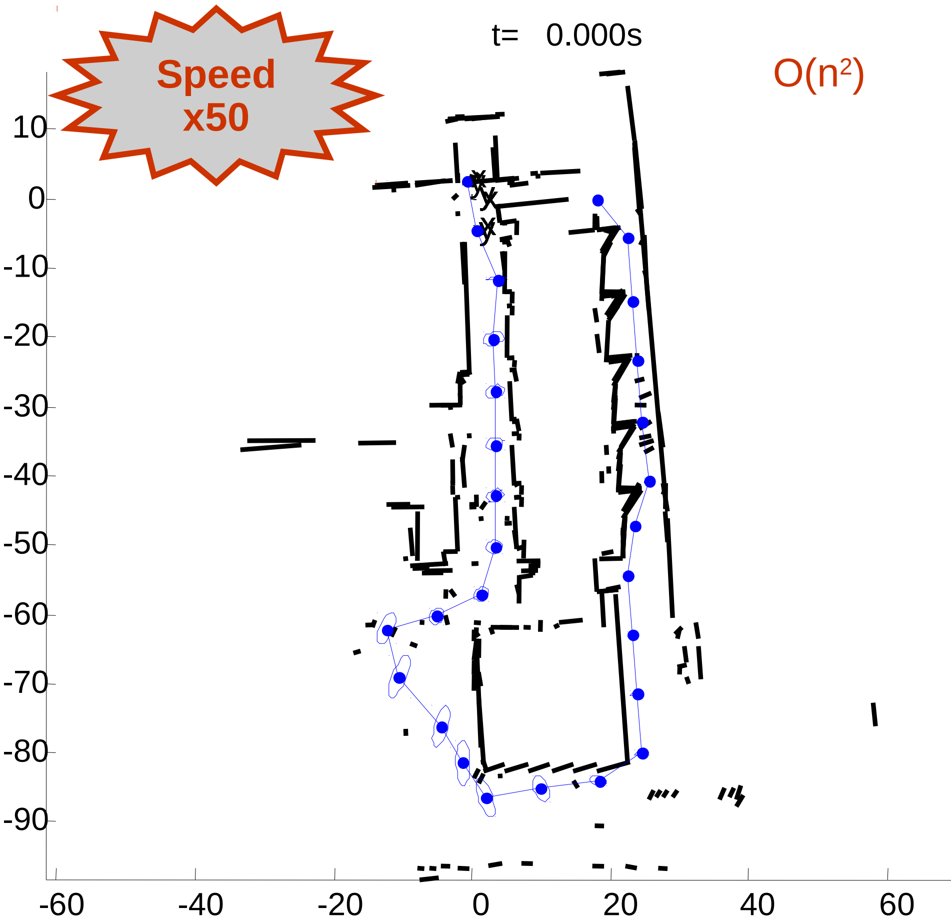


Local Maps Improve Consistency

- Classic EKF-SLAM

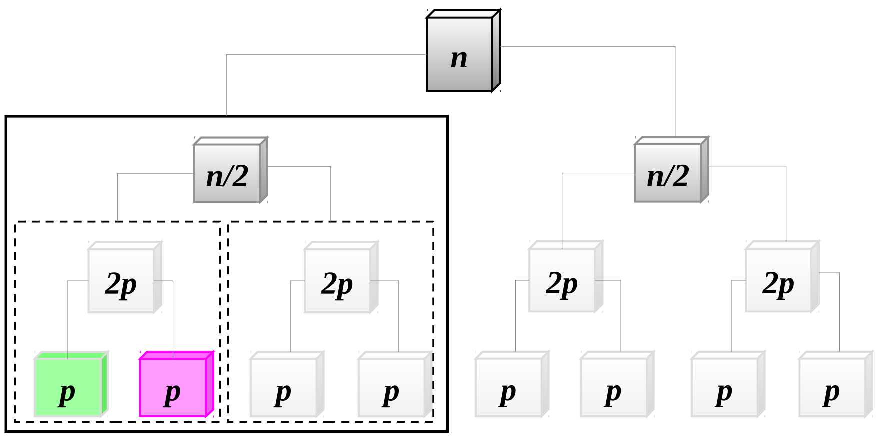


- Map joining: 28 local maps

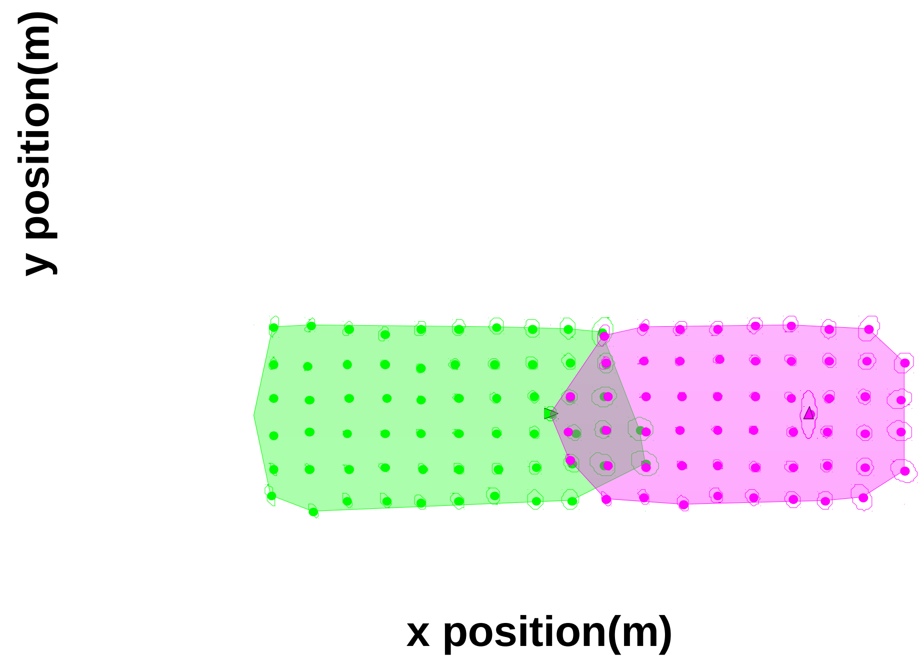


J.A. Castellanos, R. Martinez-Cantin, J.D. Tardós and J. Neira, "**Robocentric Map Joining: Improving the Consistency of EKF-SLAM**", Robotics and Autonomous Systems, Vol. 55, pp. 21-29, 2007

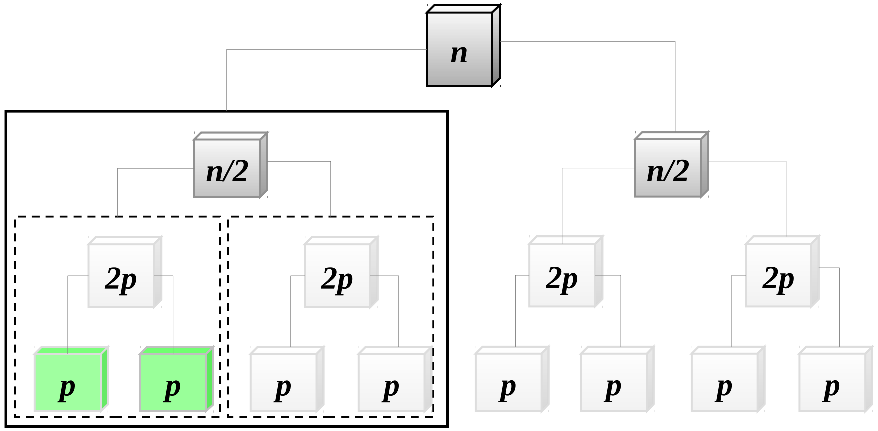
Divide & Conquer SLAM



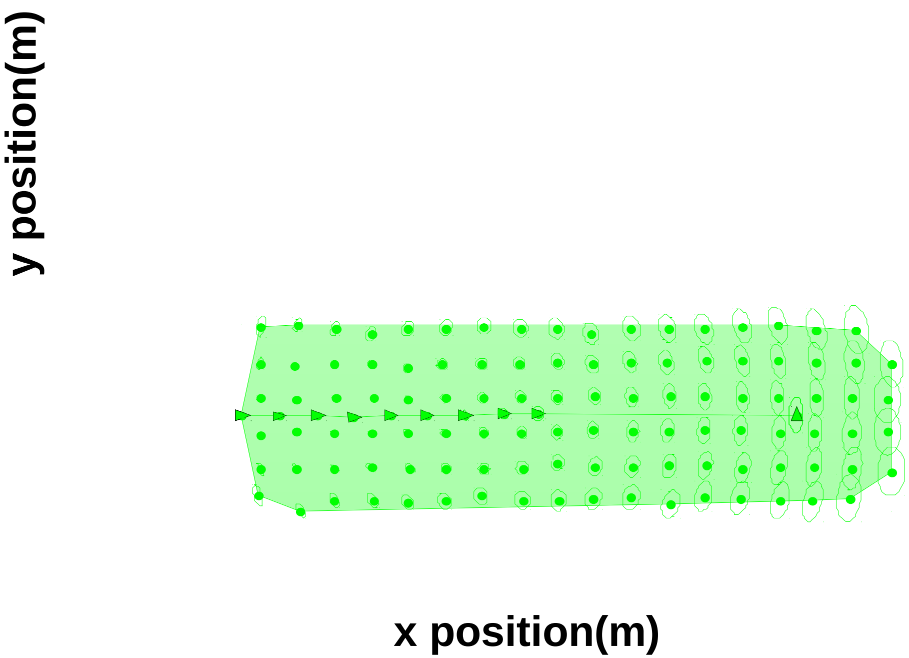
Number of Maps : 2



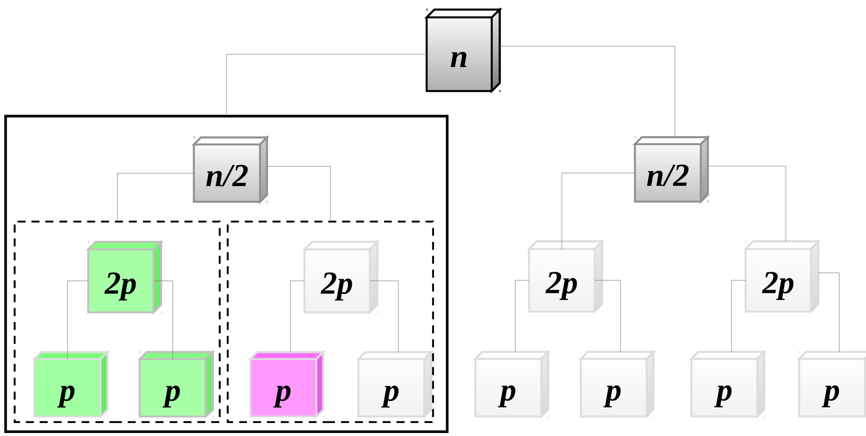
Divide & Conquer SLAM



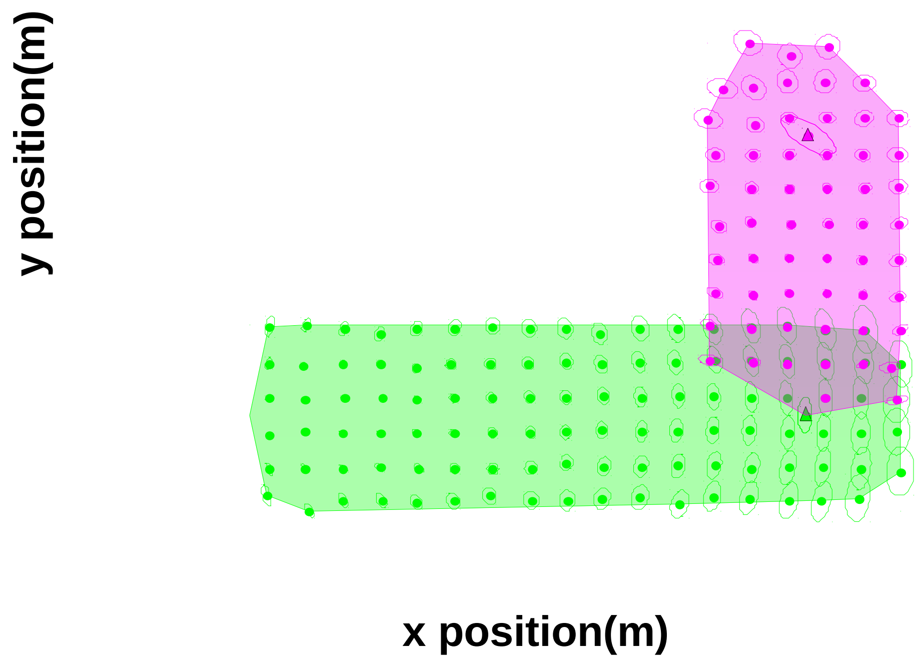
Number of Maps : 1



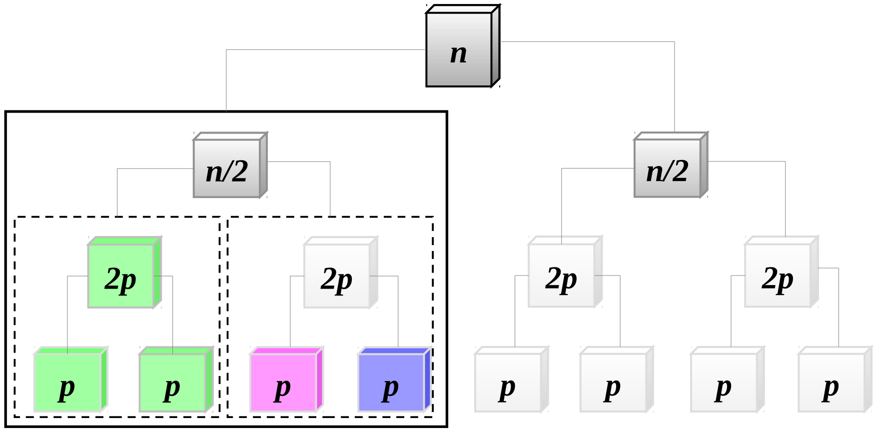
Divide & Conquer SLAM

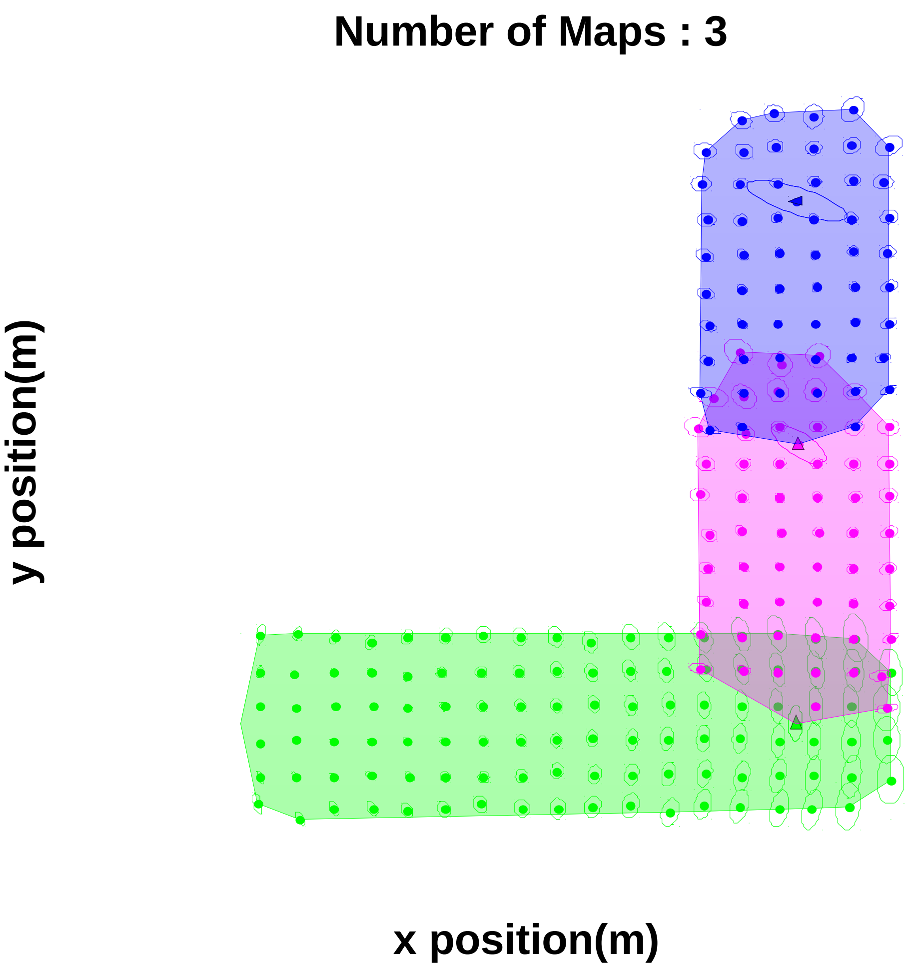


Number of Maps : 2

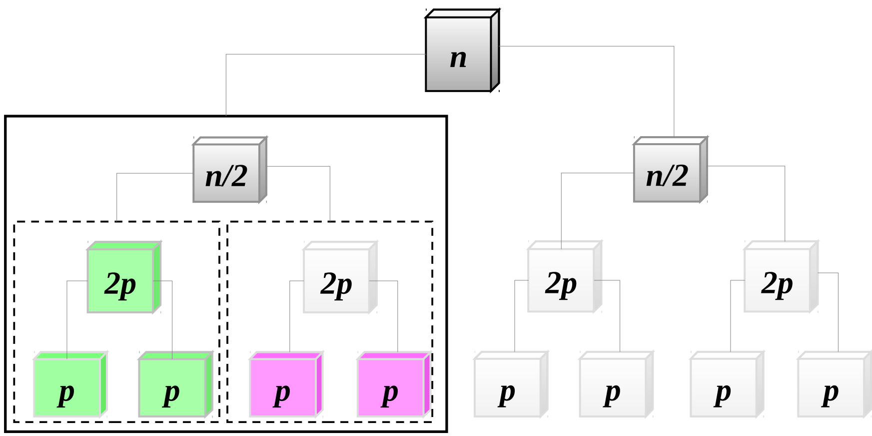


Divide & Conquer SLAM

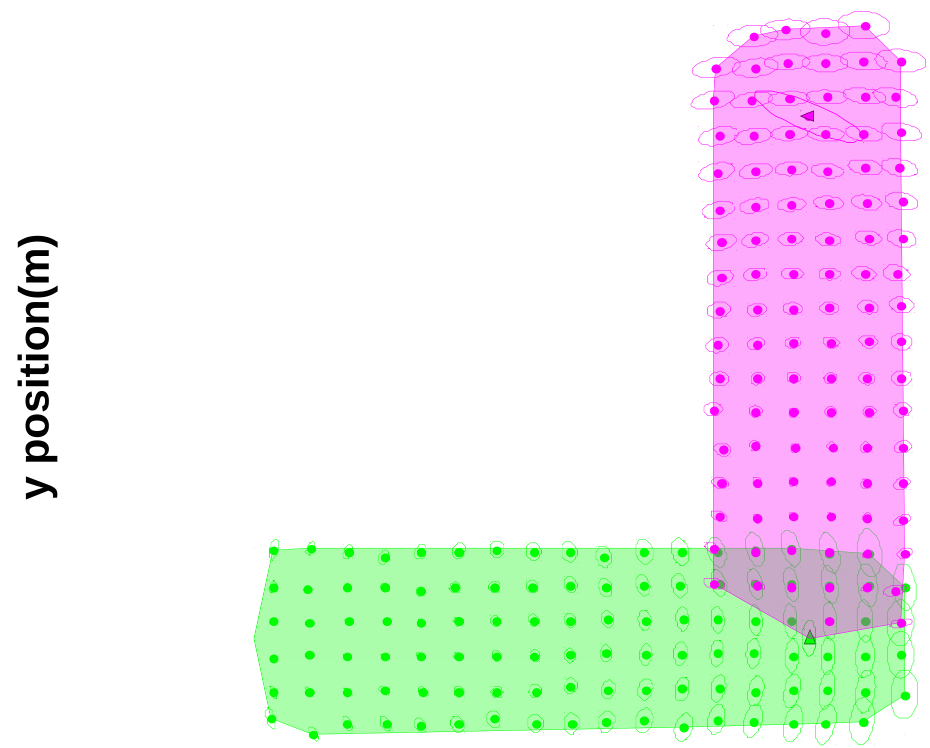




Divide & Conquer SLAM

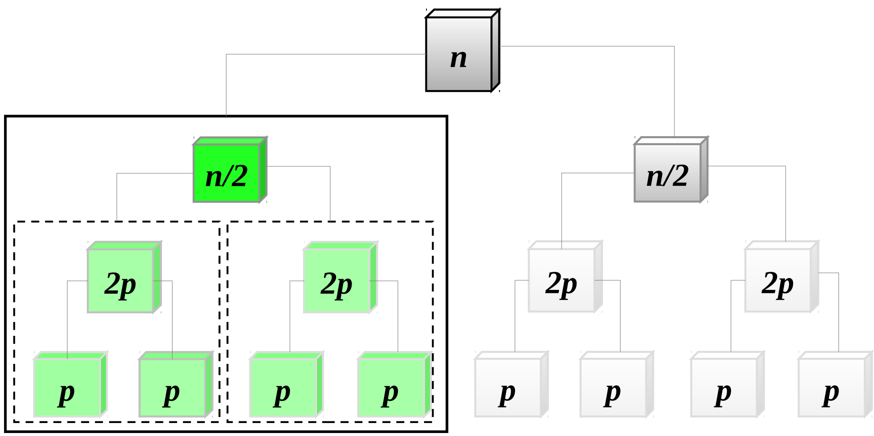


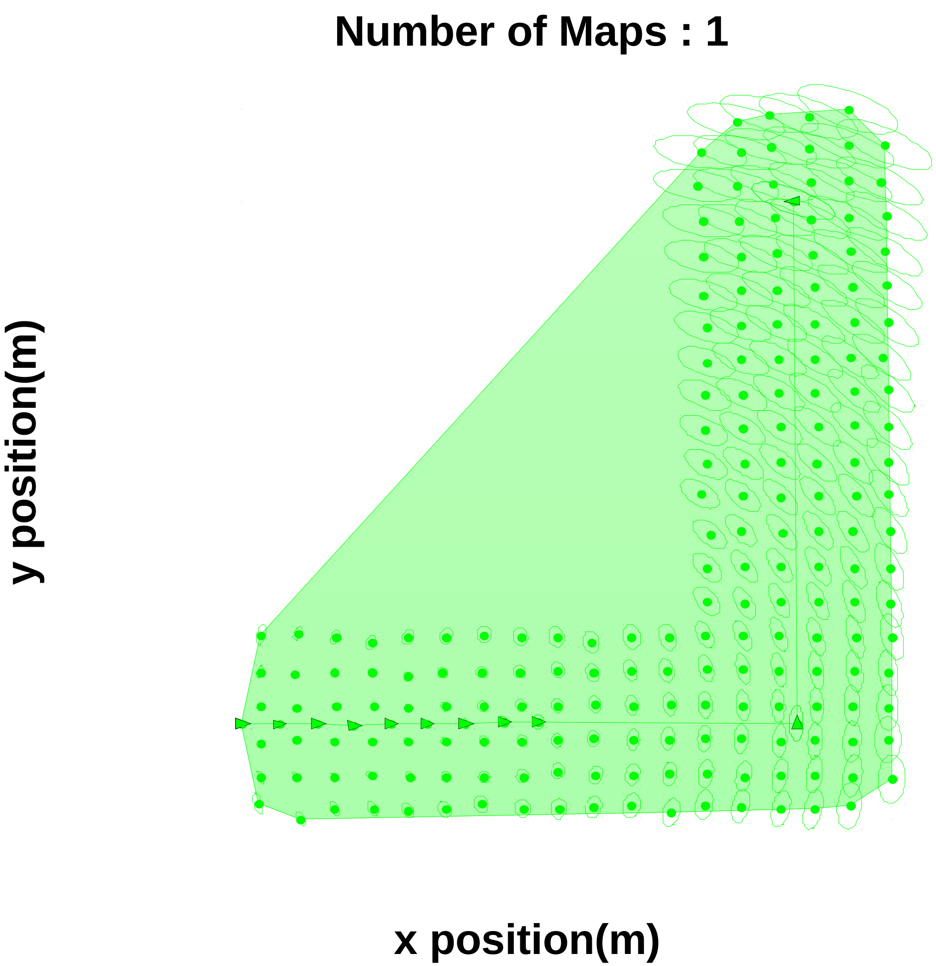
Number of Maps : 2



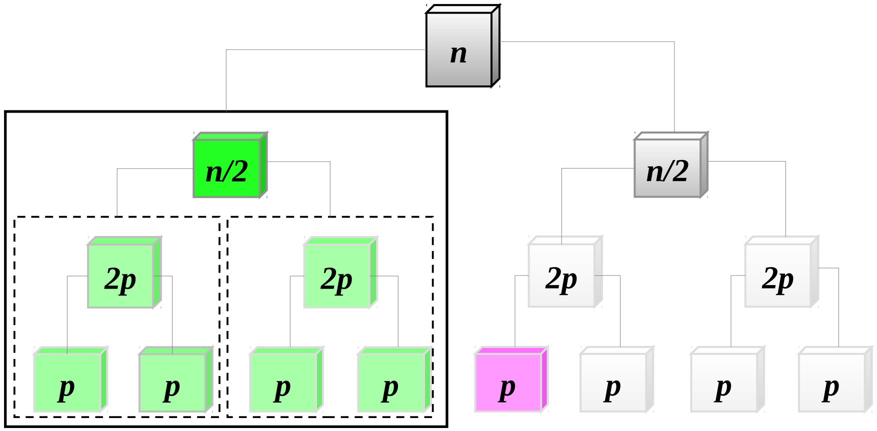
x position(m)

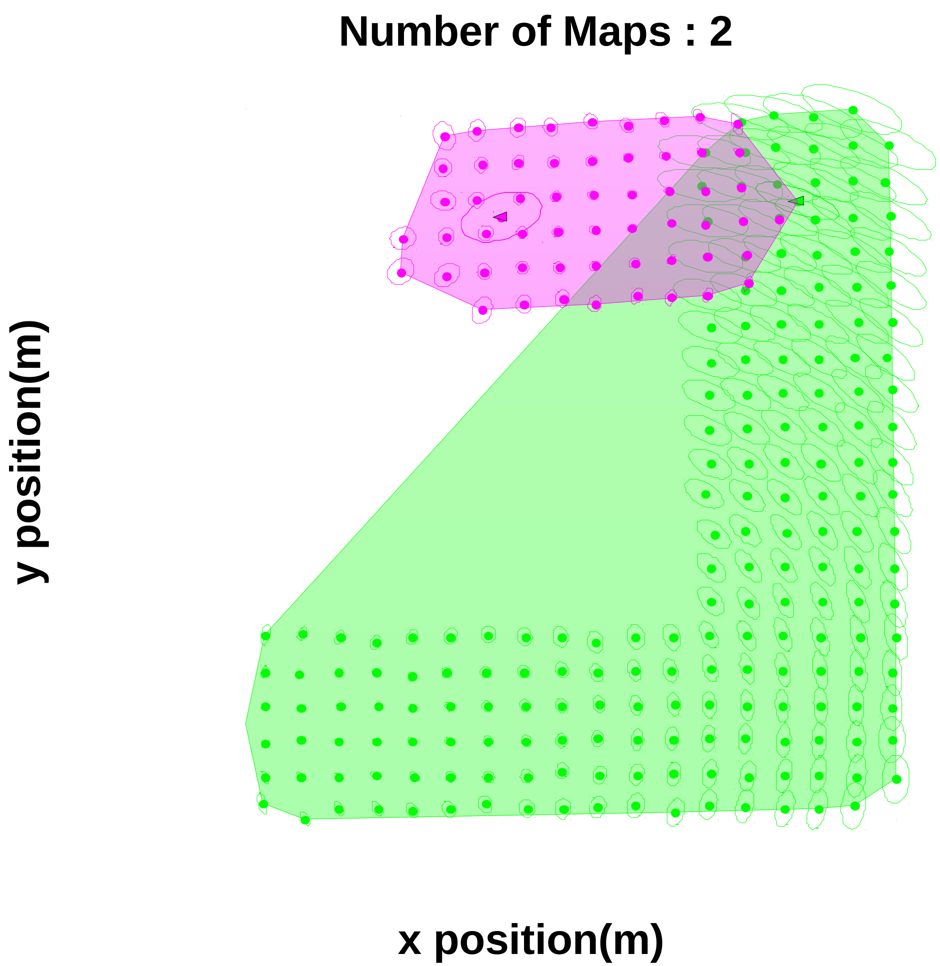
Divide & Conquer SLAM



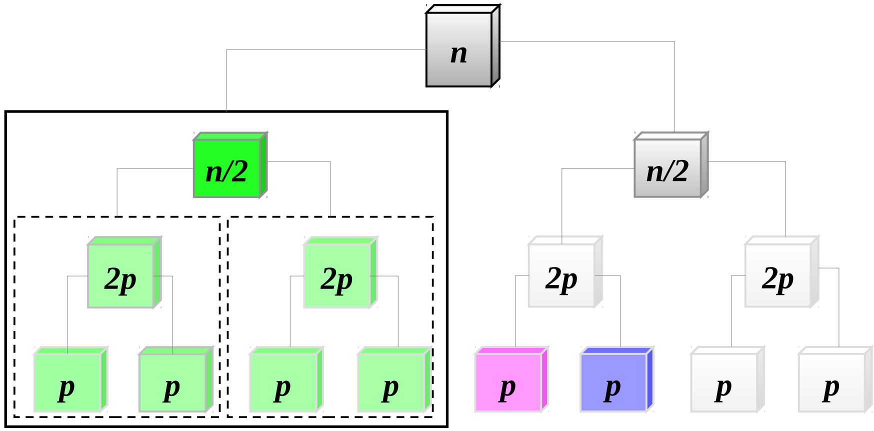


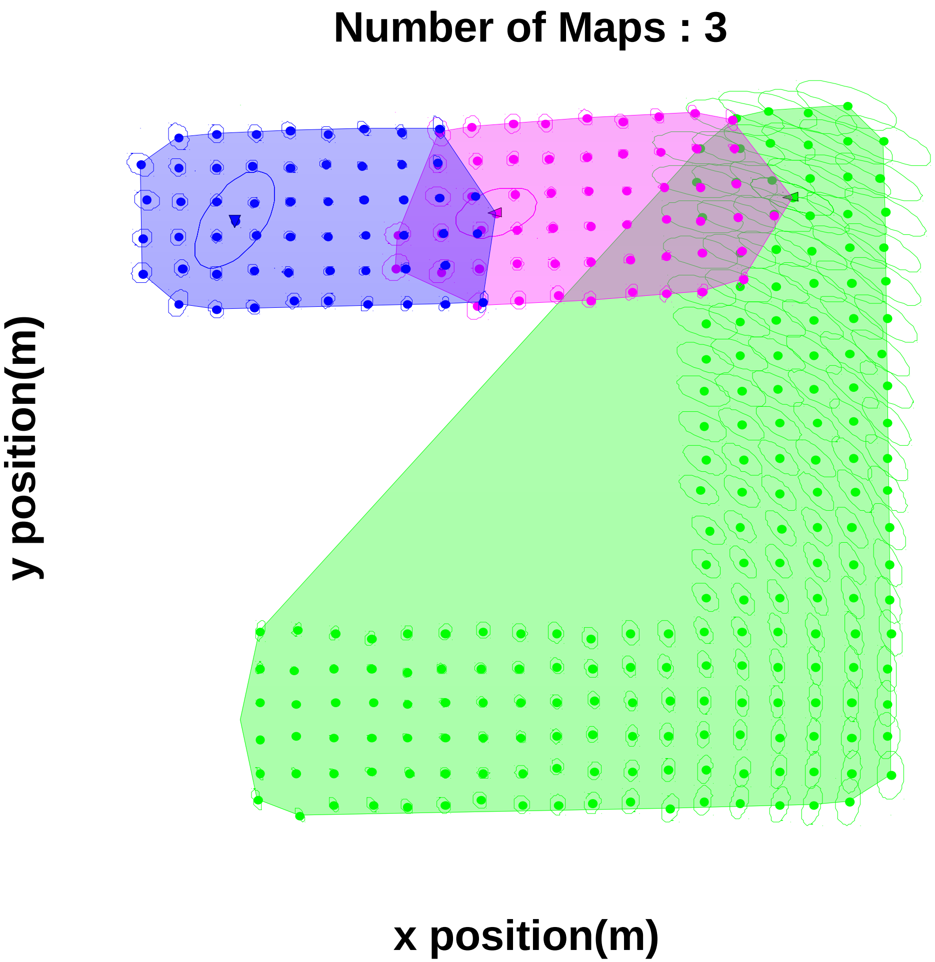
Divide & Conquer SLAM



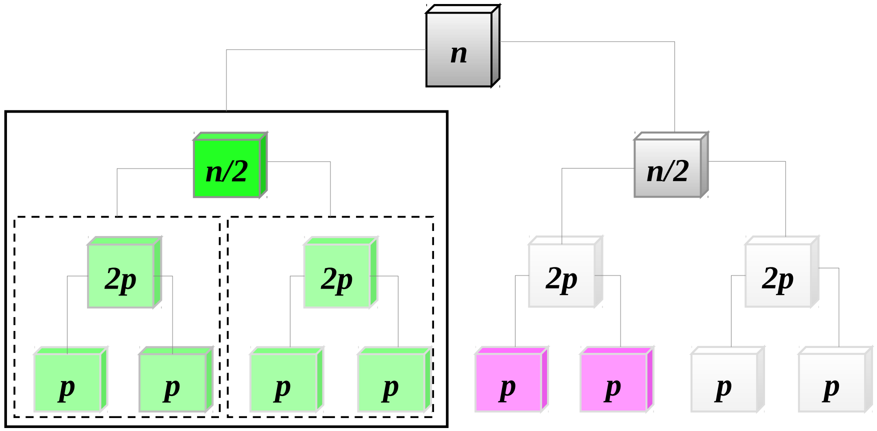


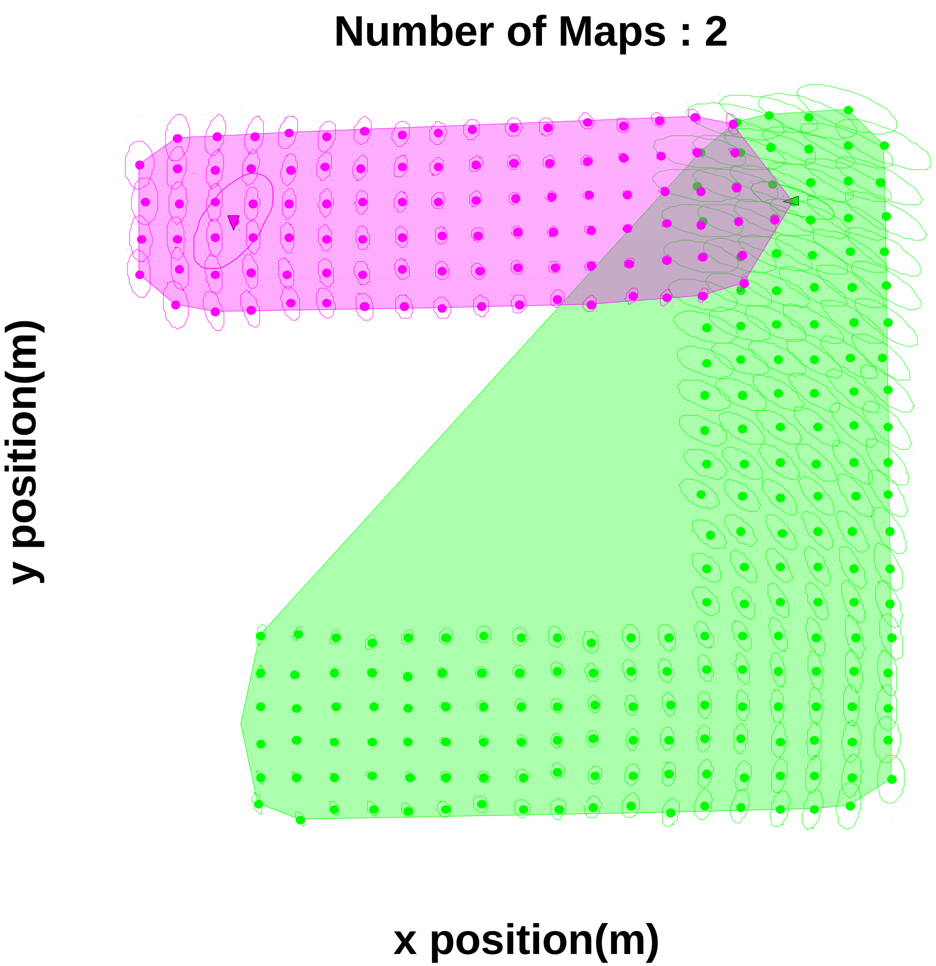
Divide & Conquer SLAM



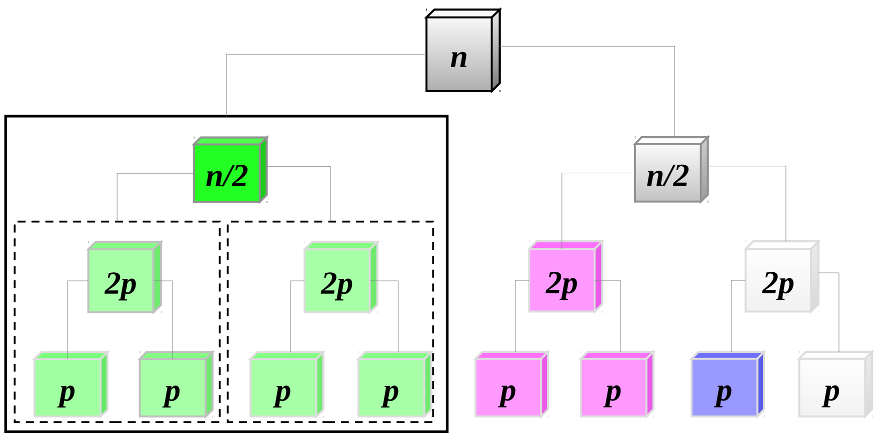


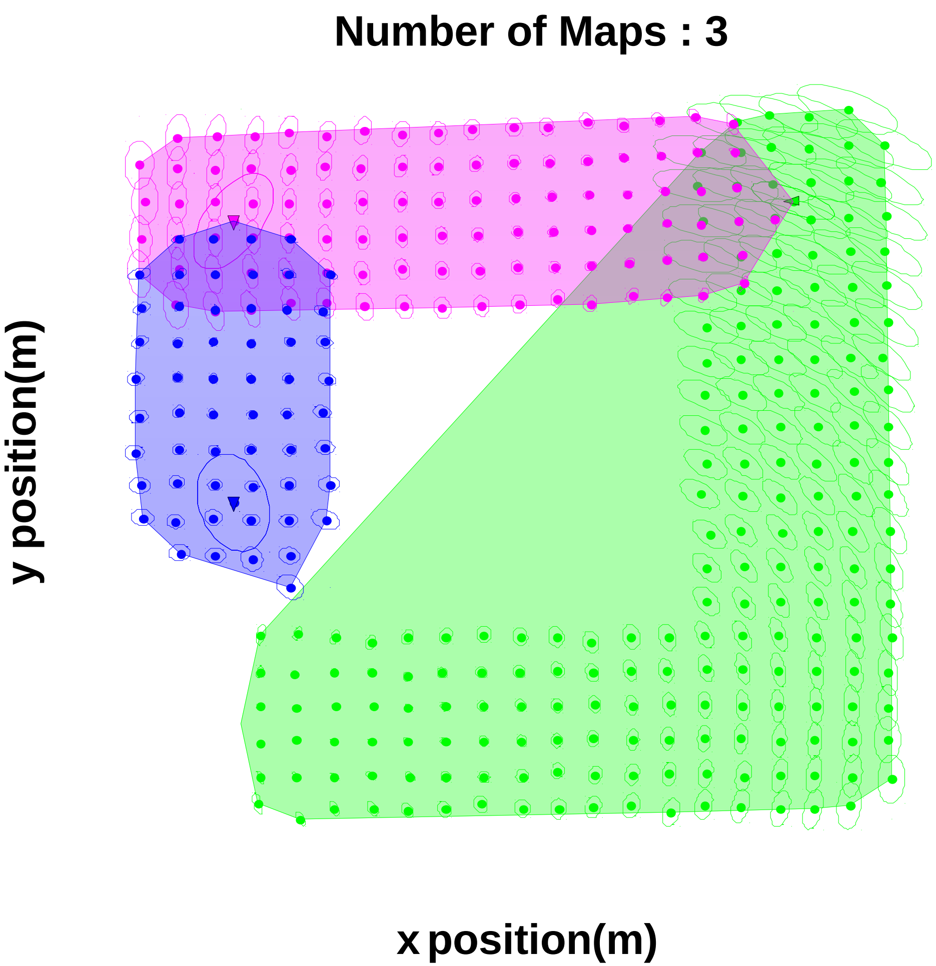
Divide & Conquer SLAM



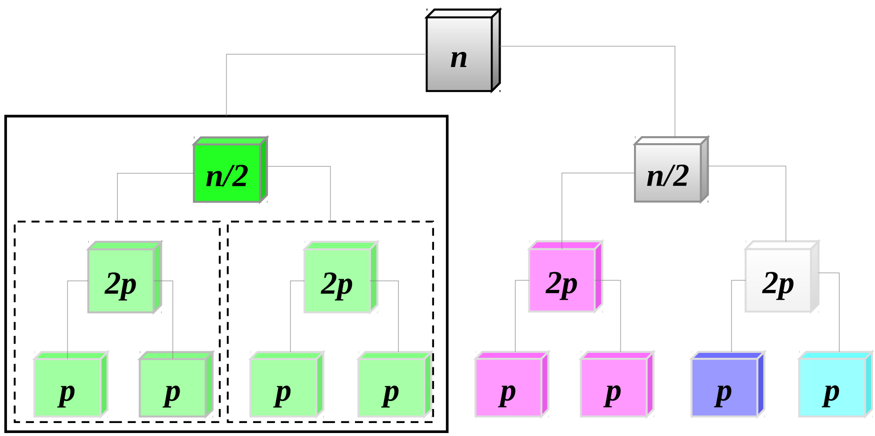


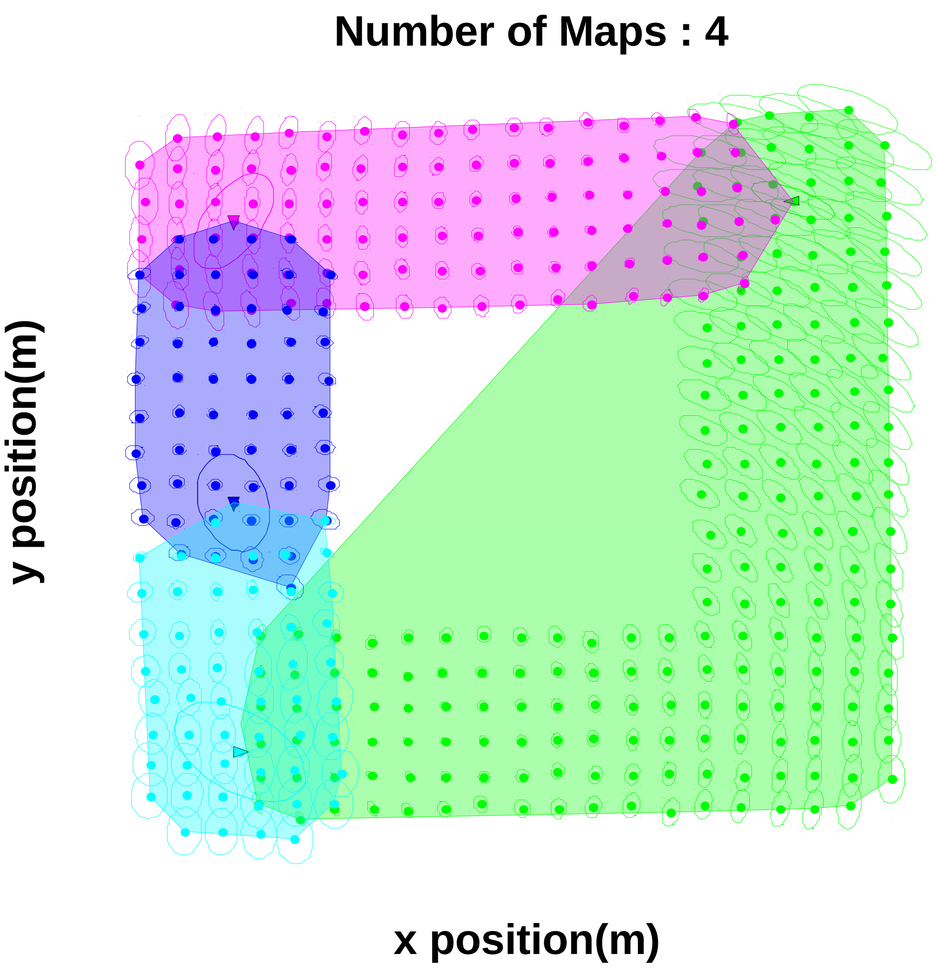
Divide & Conquer SLAM



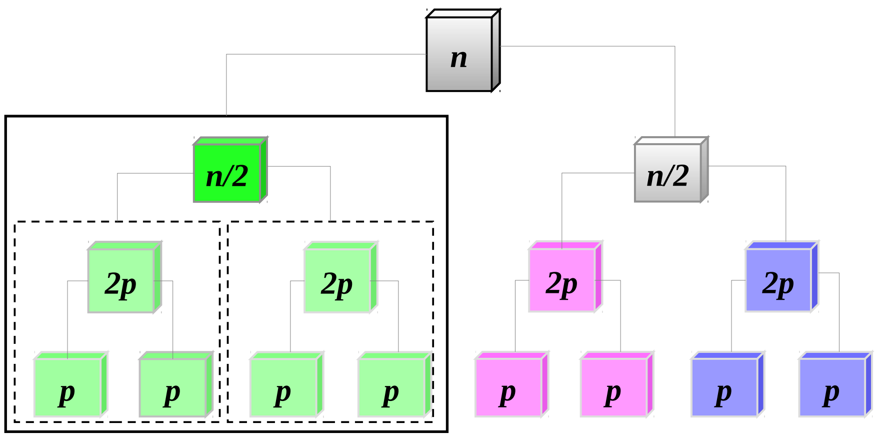


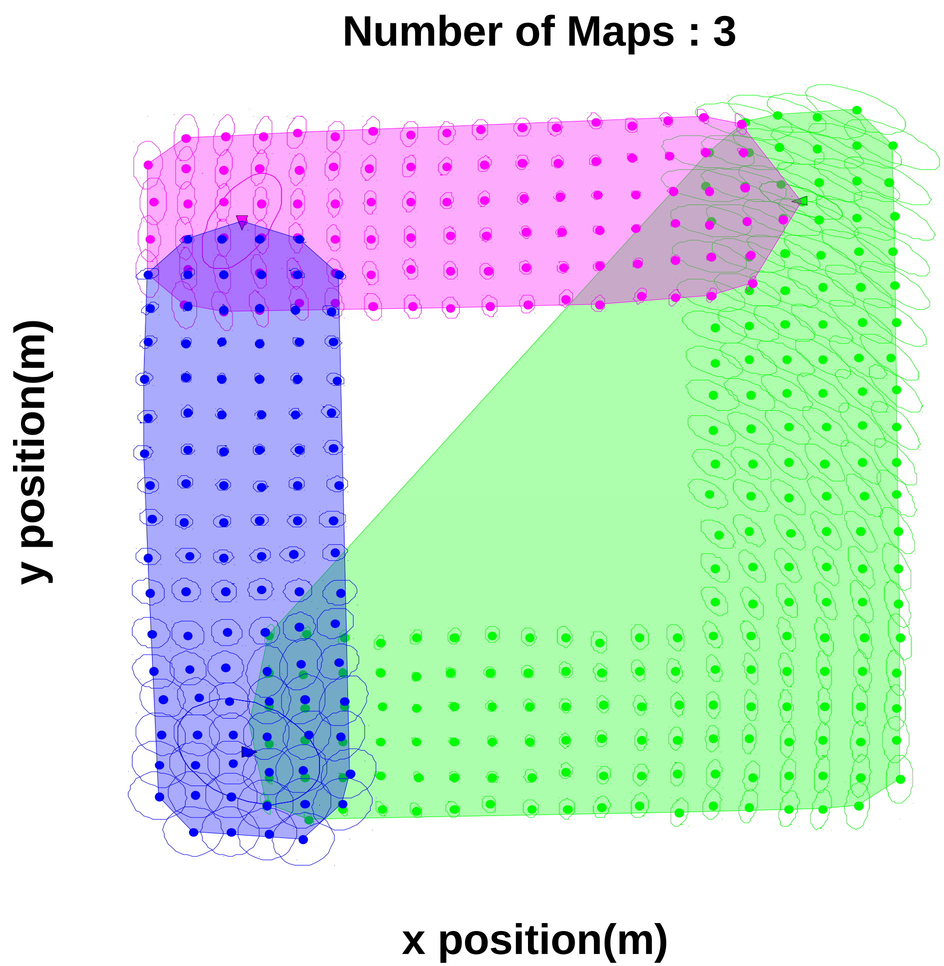
Divide & Conquer SLAM



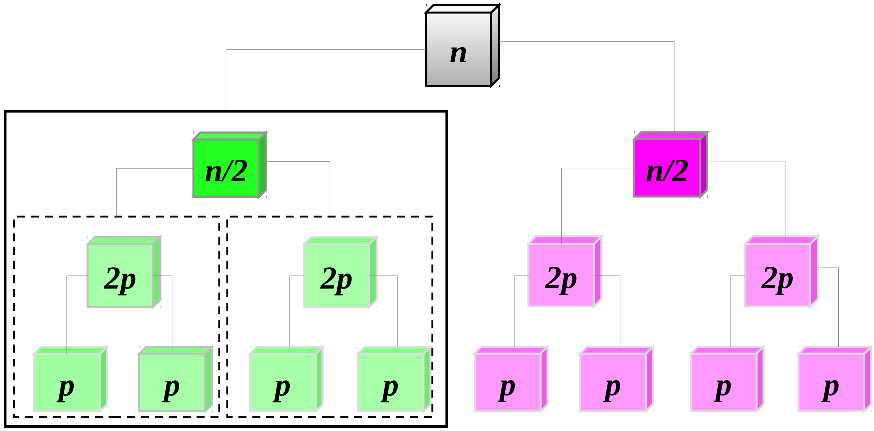


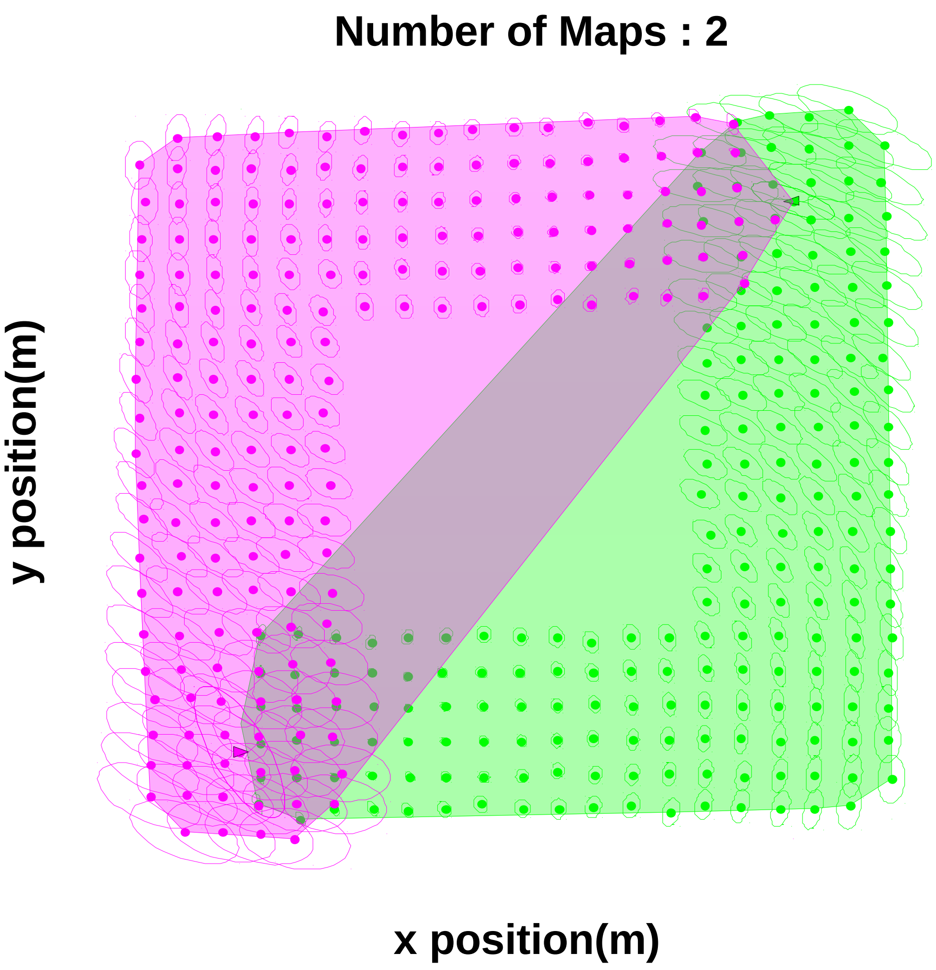
Divide & Conquer SLAM



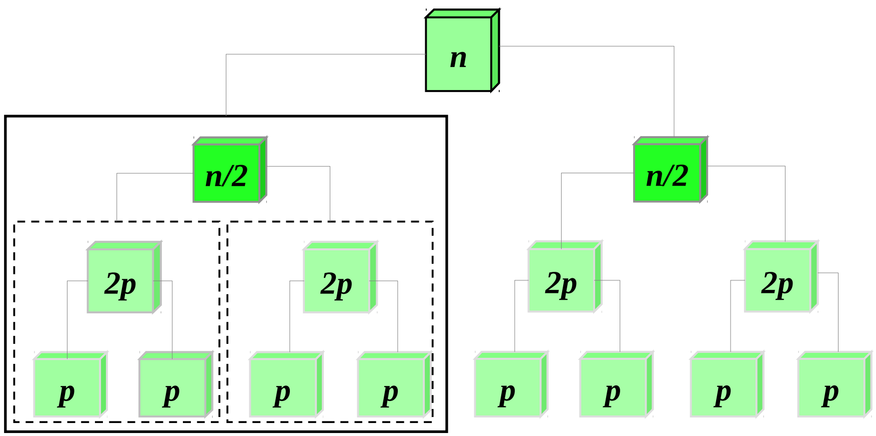


Divide & Conquer SLAM

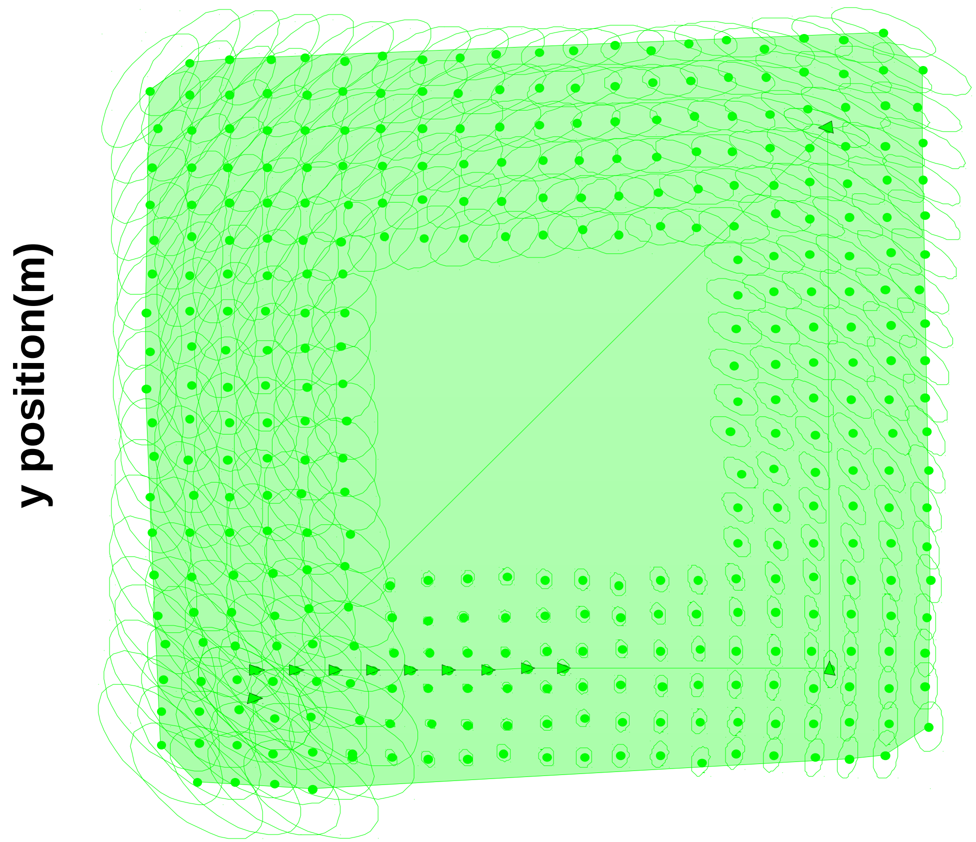




Divide & Conquer SLAM

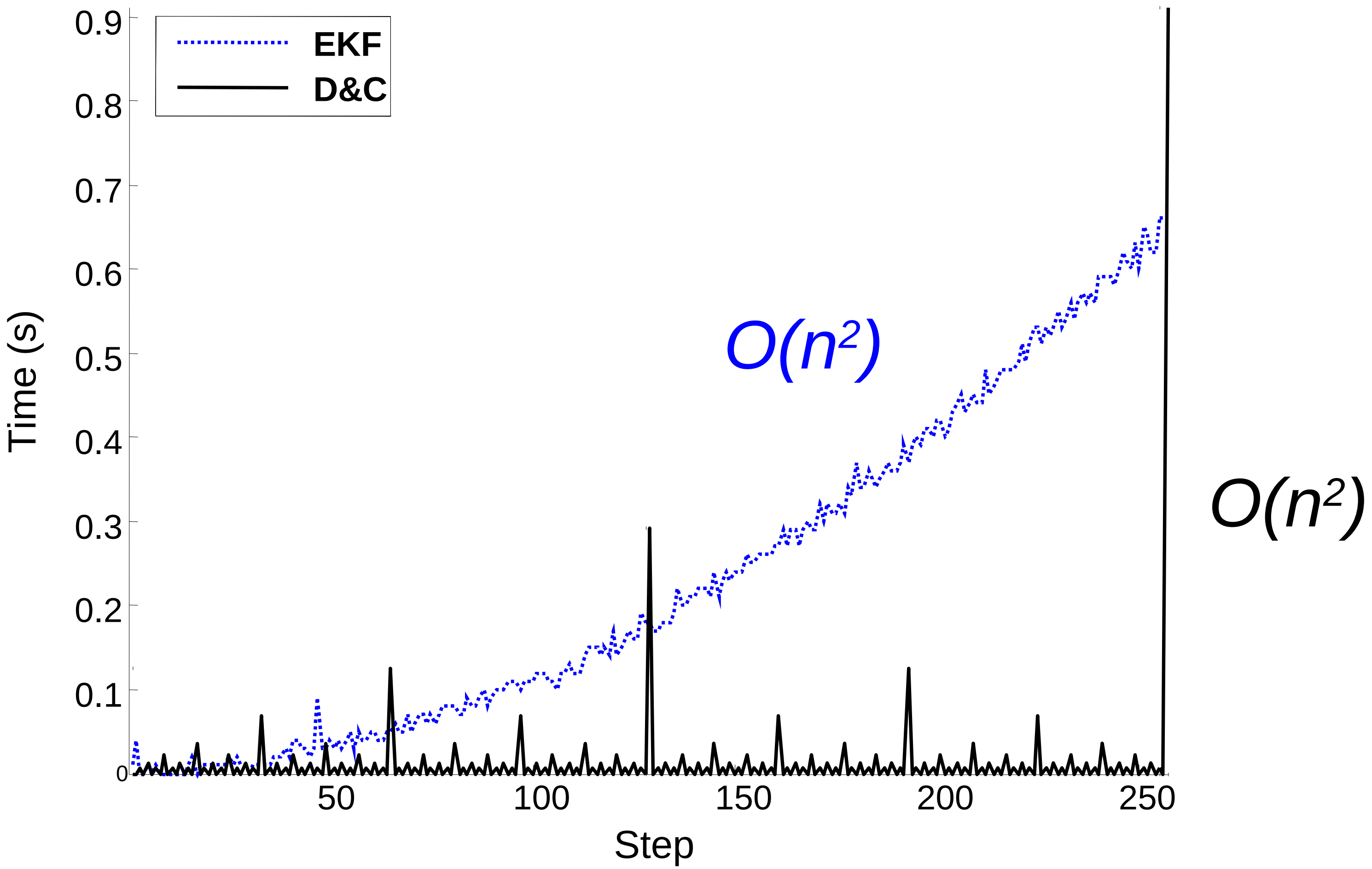


Number of Maps : 1

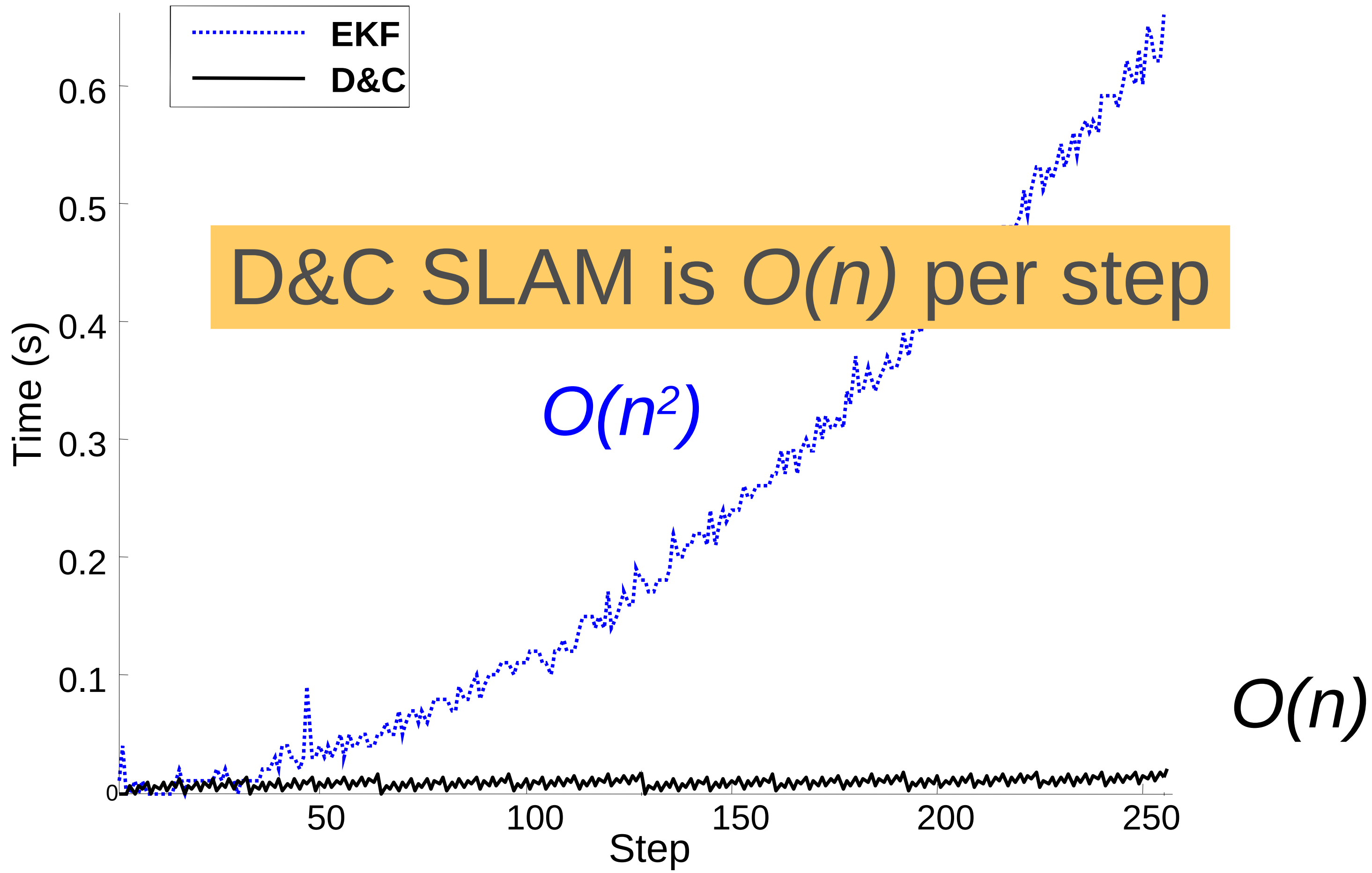


L.M. Paz, P. Jensfelt, J.D. Tardós and J. Neira. **EKF SLAM updates in O(n) with Divide and Conquer SLAM** 2007 IEEE Int. Conf. Robotics and Automation, April 10-14, Rome, Italy

Computational cost per step



Amortized cost per step

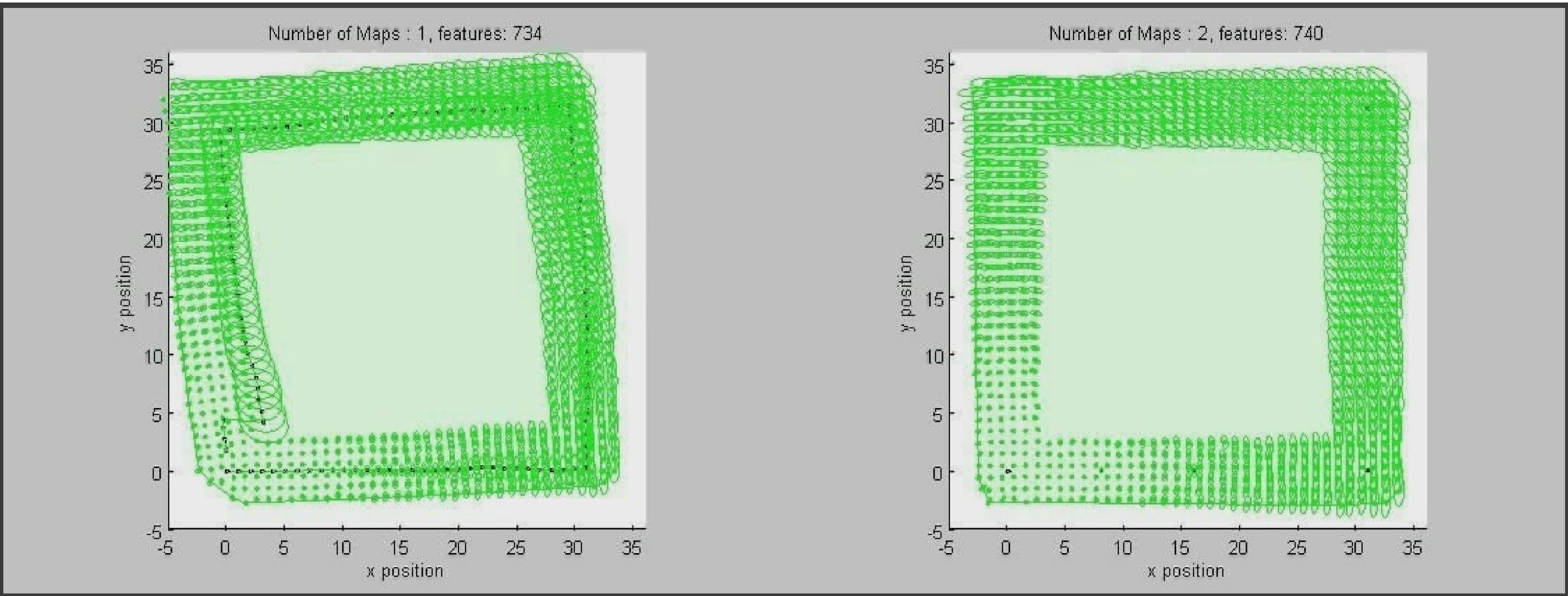


D&C SLAM is *O(n)* per step

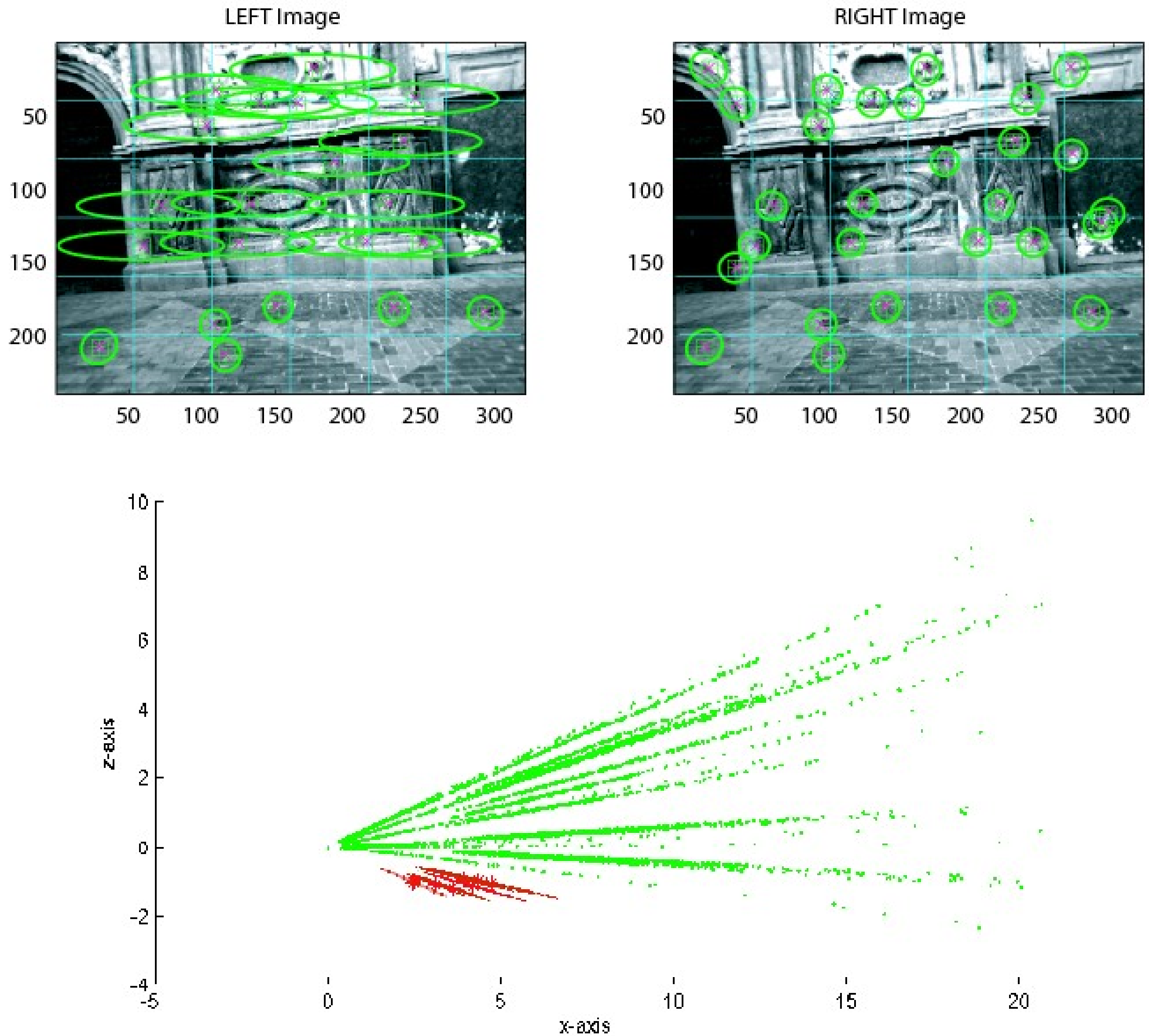
The full map can be recovered at any time in a single *O(n²)* step

- But no part of the algorithm or process requires it

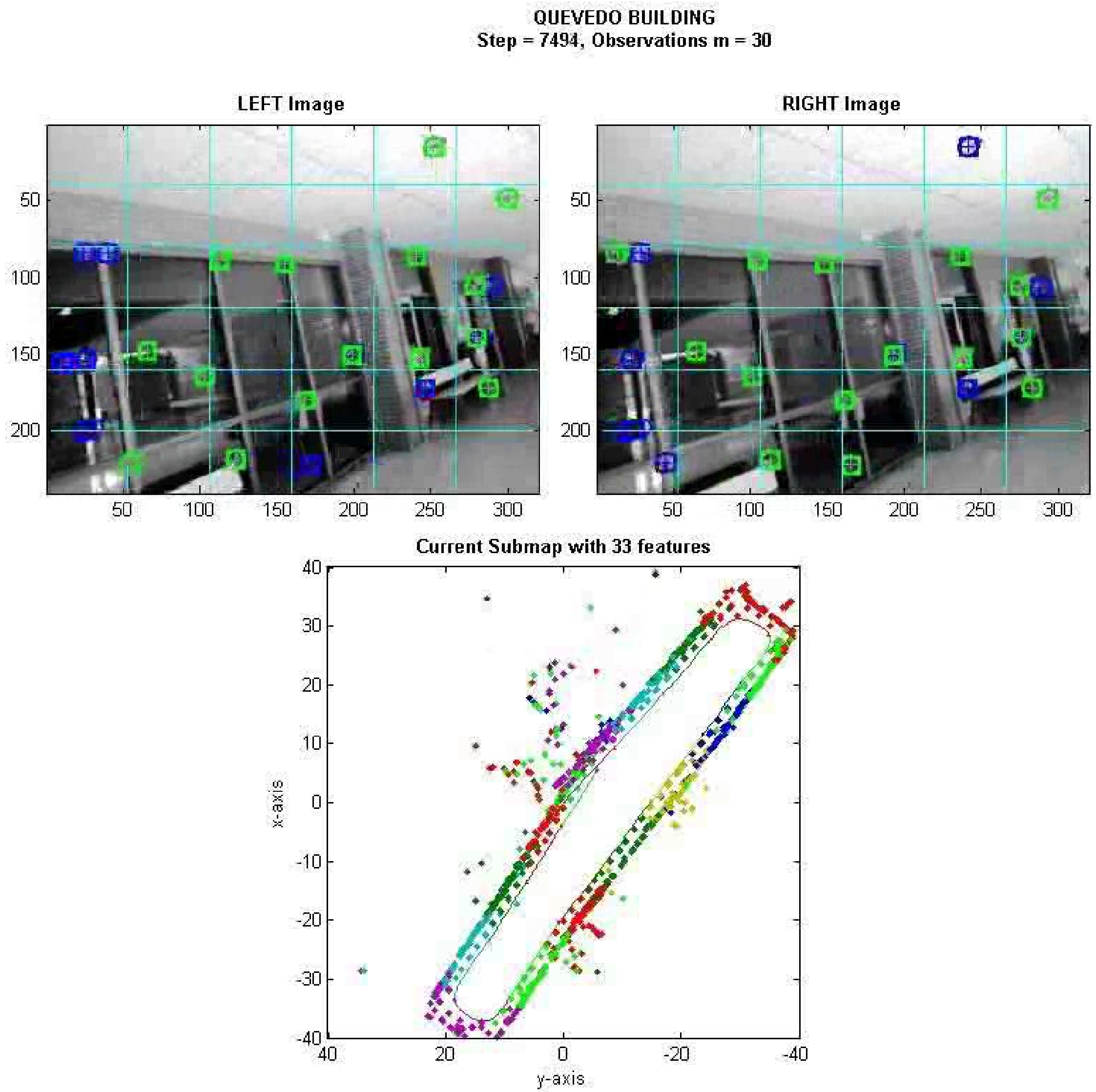
Consistency: LOOP



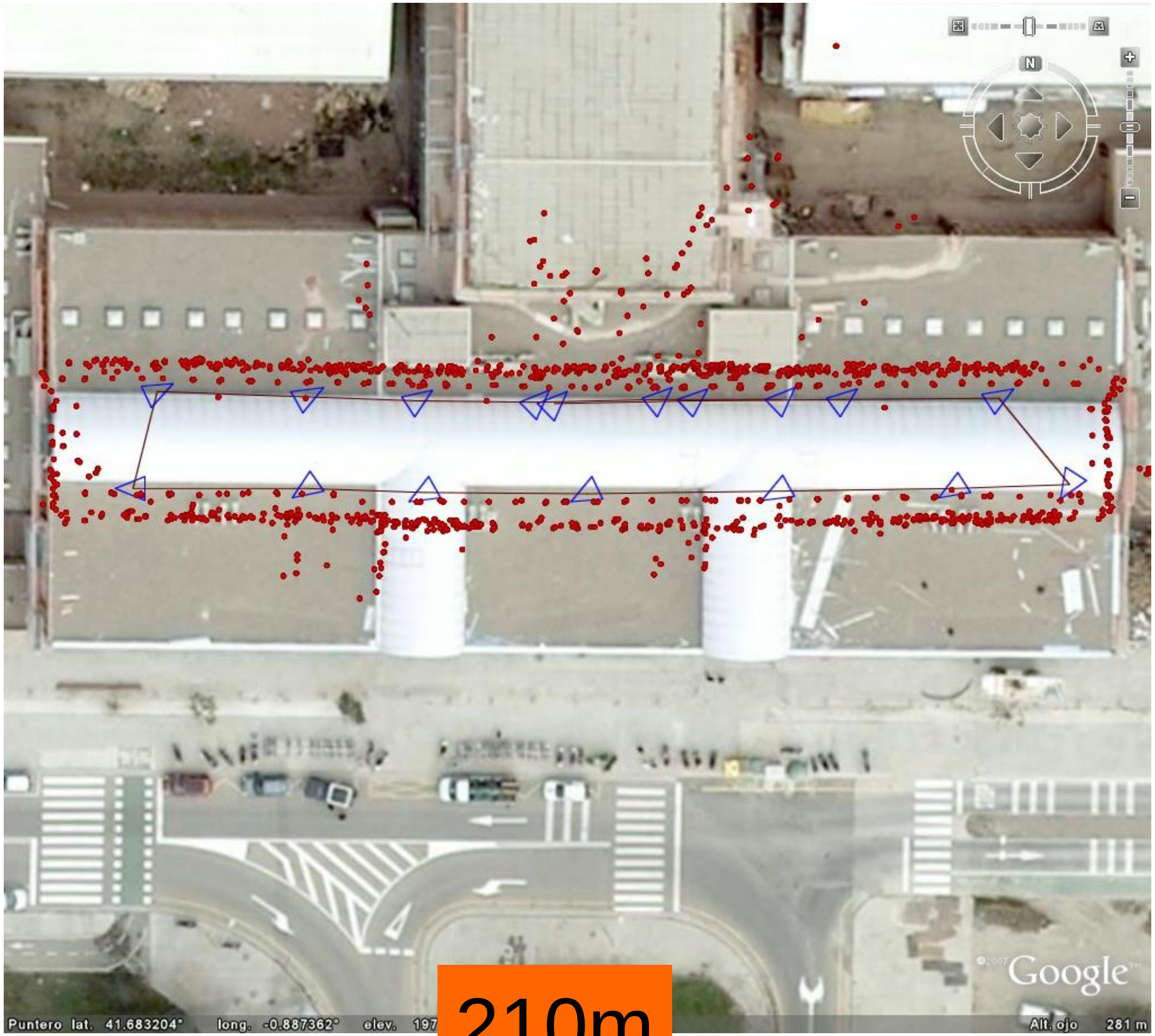
Close and Distant Points (Inverse Depth)



6Dof Stereo SLAM, indoors

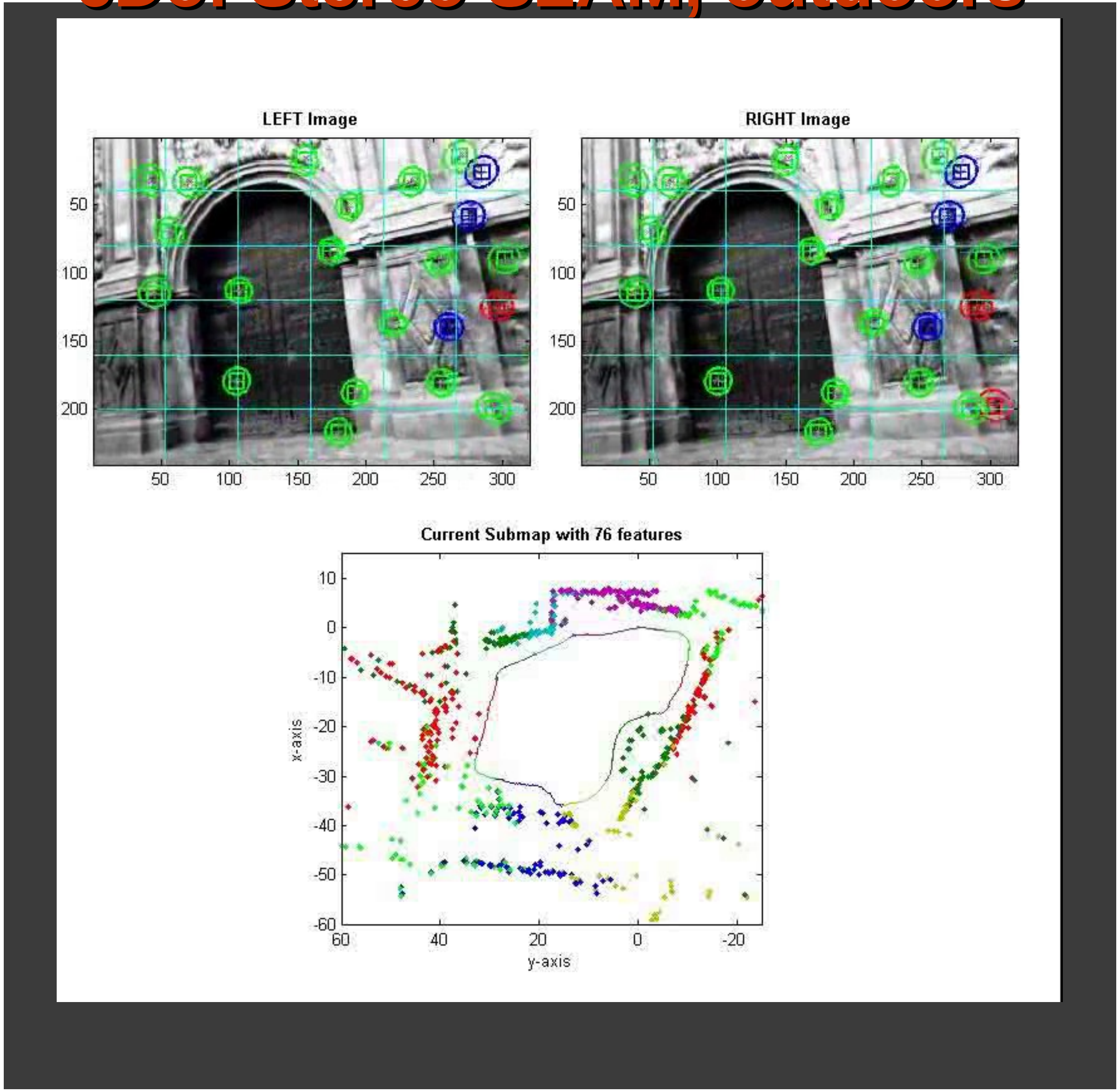


6Dof Stereo SLAM, indoors



210m

6Dof Stereo SLAM, outdoors



L. Paz, P. Piníes, J. Neira and J.D. Tardós, **Large Scale 6DOF SLAM with Stereo-in-Hand.** IEEE Transactions on Robotics, 25(4): 946-957, Oct 2008.

6Dof Stereo SLAM, outdoors

150m

Outline

1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

Conditionally Independent Maps

- **Independence**
  - **x** e **y** are independent if:

$$p(\mathbf{x},\mathbf{y}) = p(\mathbf{x})p(\mathbf{y})$$
$$p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x})$$

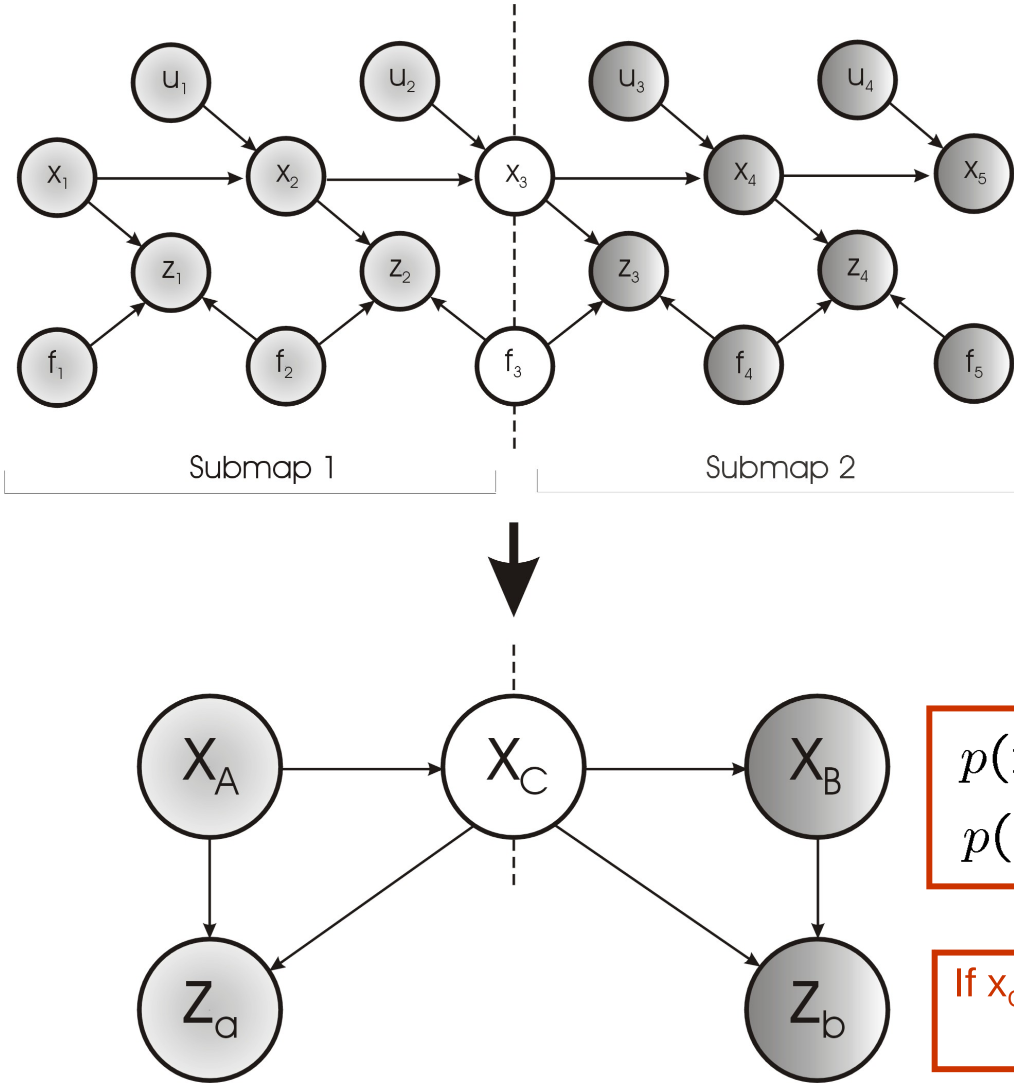
  - Variable **y** does not provide information about **x**
  - **The maps cannot share information**

- **Conditional Independence**
  - **x** e **y** are independent, given **z** if:

$$p(\mathbf{x},\mathbf{y}|\mathbf{z}) = p(\mathbf{x}|\mathbf{z})p(\mathbf{y}|\mathbf{z})$$
$$p(\mathbf{x}|\mathbf{y},\mathbf{z}) = p(\mathbf{x}|\mathbf{z})$$

  - If **z** is known, **y** does not provide additional information about **x**
  - **The maps can share information (z)**

Conditionally Independent Maps



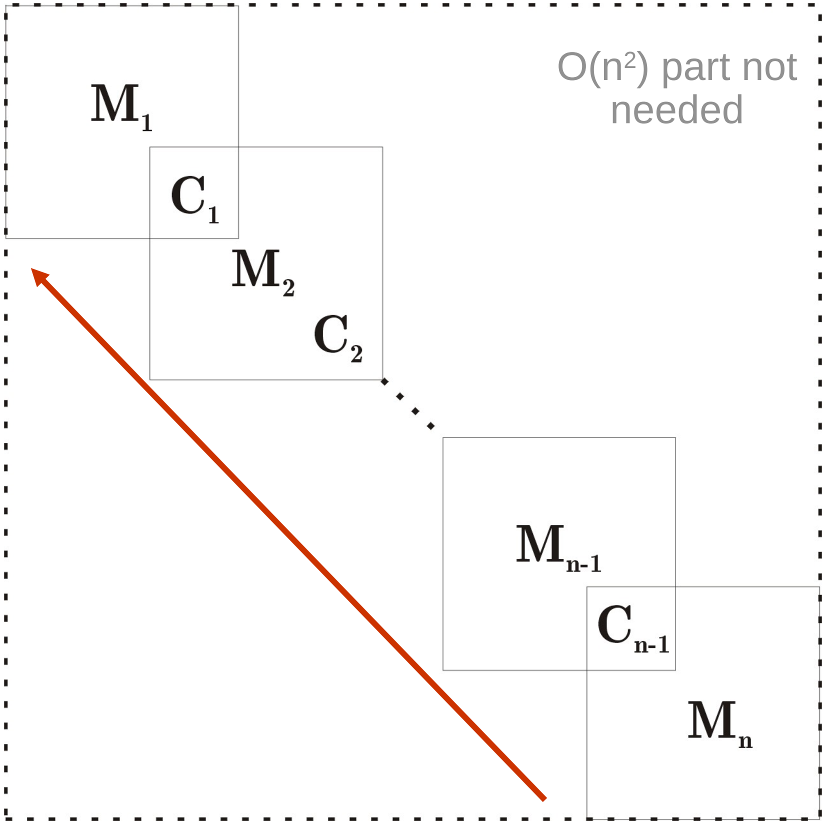
- Both maps share $\mathbf{x}_3$ and $\mathbf{f}_3$
- They are not independent
- But they are **Conditionally Independent**, given $\mathbf{x}_3$ and $\mathbf{f}_3$

$$p(\mathbf{x}_A|\mathbf{x}_B, \mathbf{x}_C, \mathbf{z}_a, \mathbf{z}_b) = p(\mathbf{x}_A|\mathbf{x}_C, \mathbf{z}_a)$$
$$p(\mathbf{x}_B|\mathbf{x}_A, \mathbf{x}_C, \mathbf{z}_a, \mathbf{z}_b) = p(\mathbf{x}_B|\mathbf{x}_C, \mathbf{z}_b)$$

If x_C is known, x_A and z_a do not provide additional information about x_B and z_b

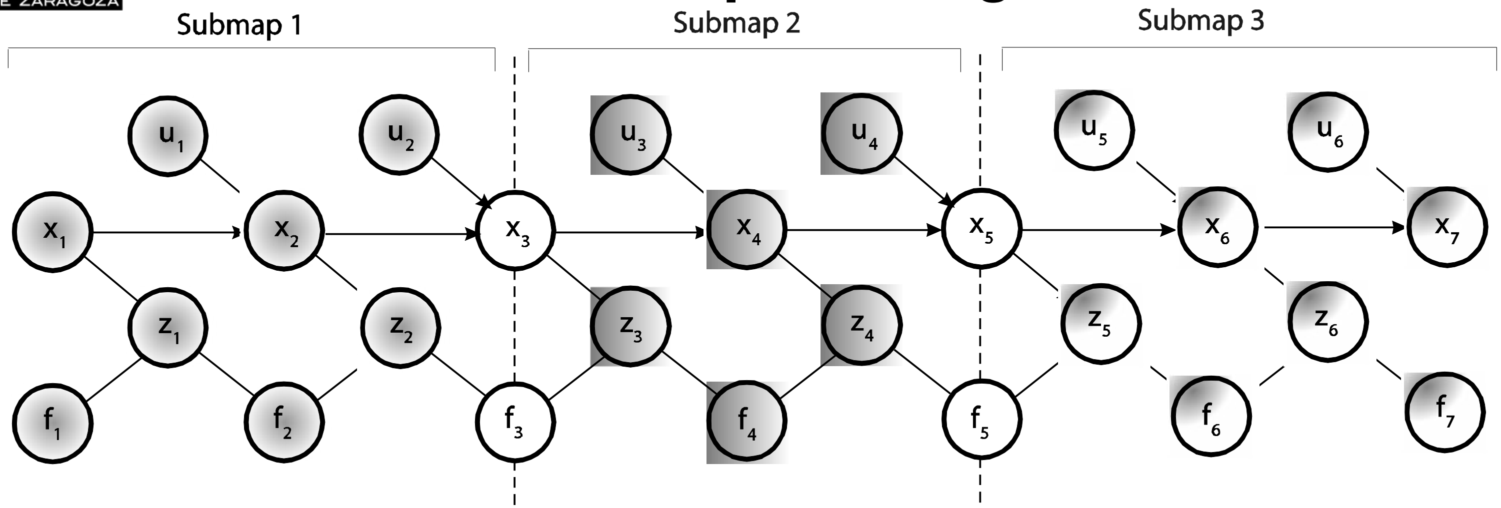
Only the block-diagonal Covariance is computed

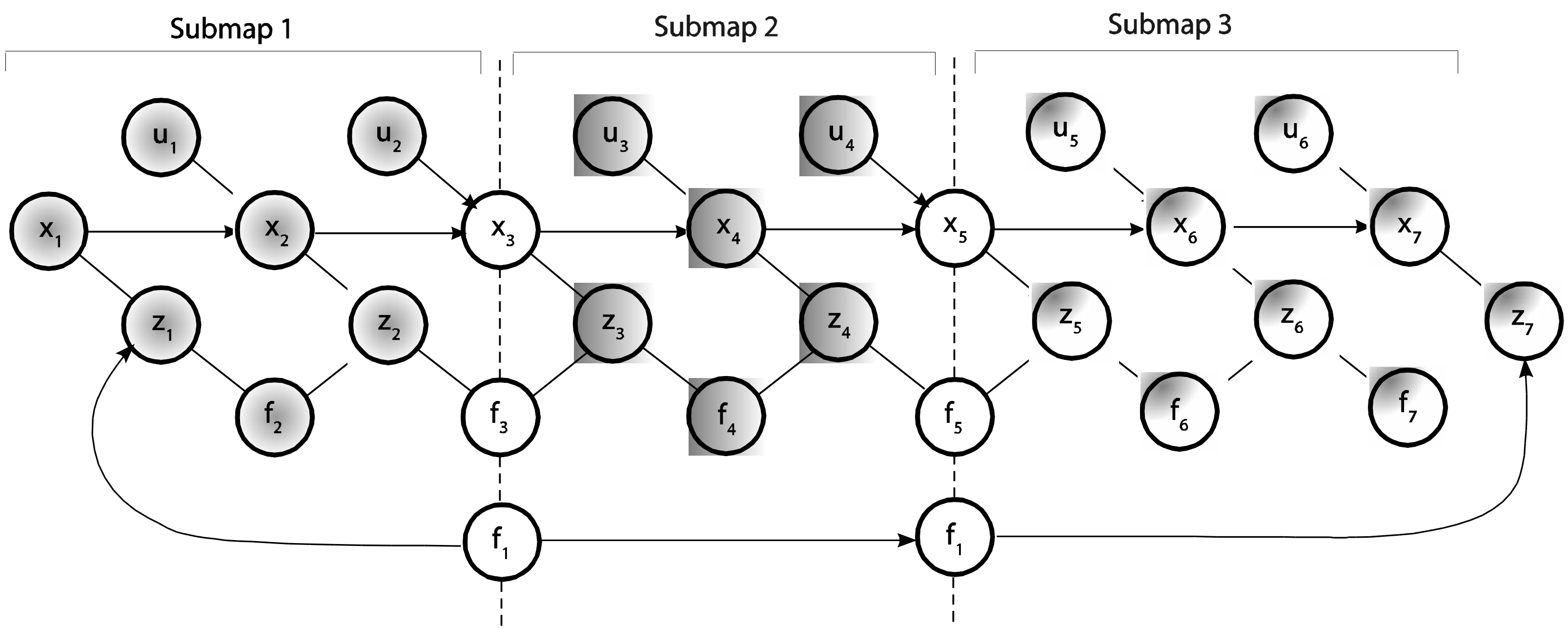
Local Maps in O(1)



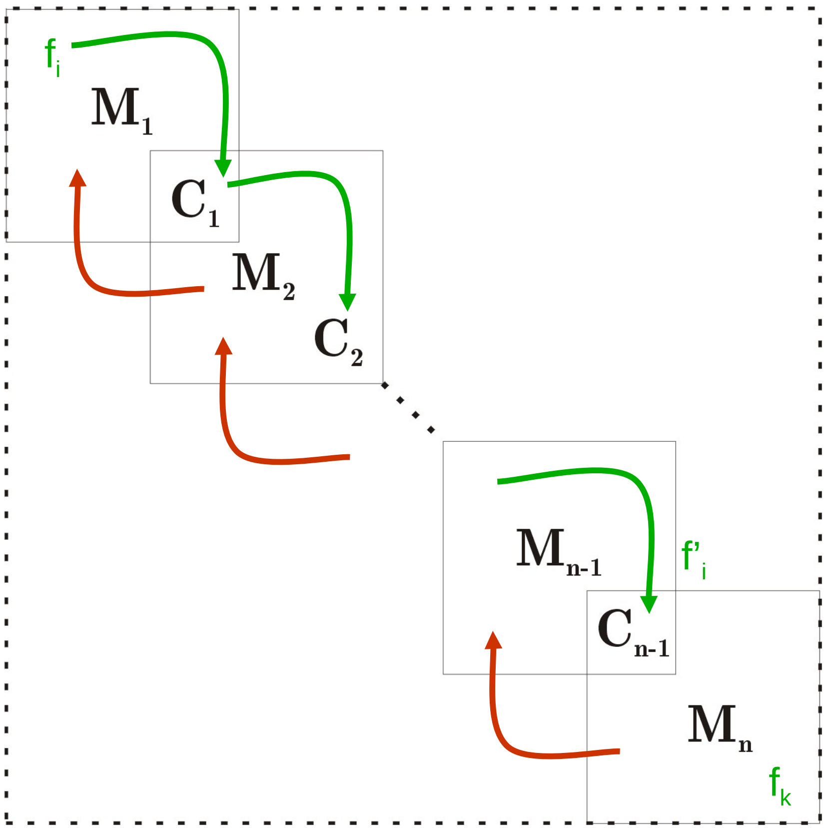
Back-Propagation:
Optimal Global Map in O(n)

Loop Closing





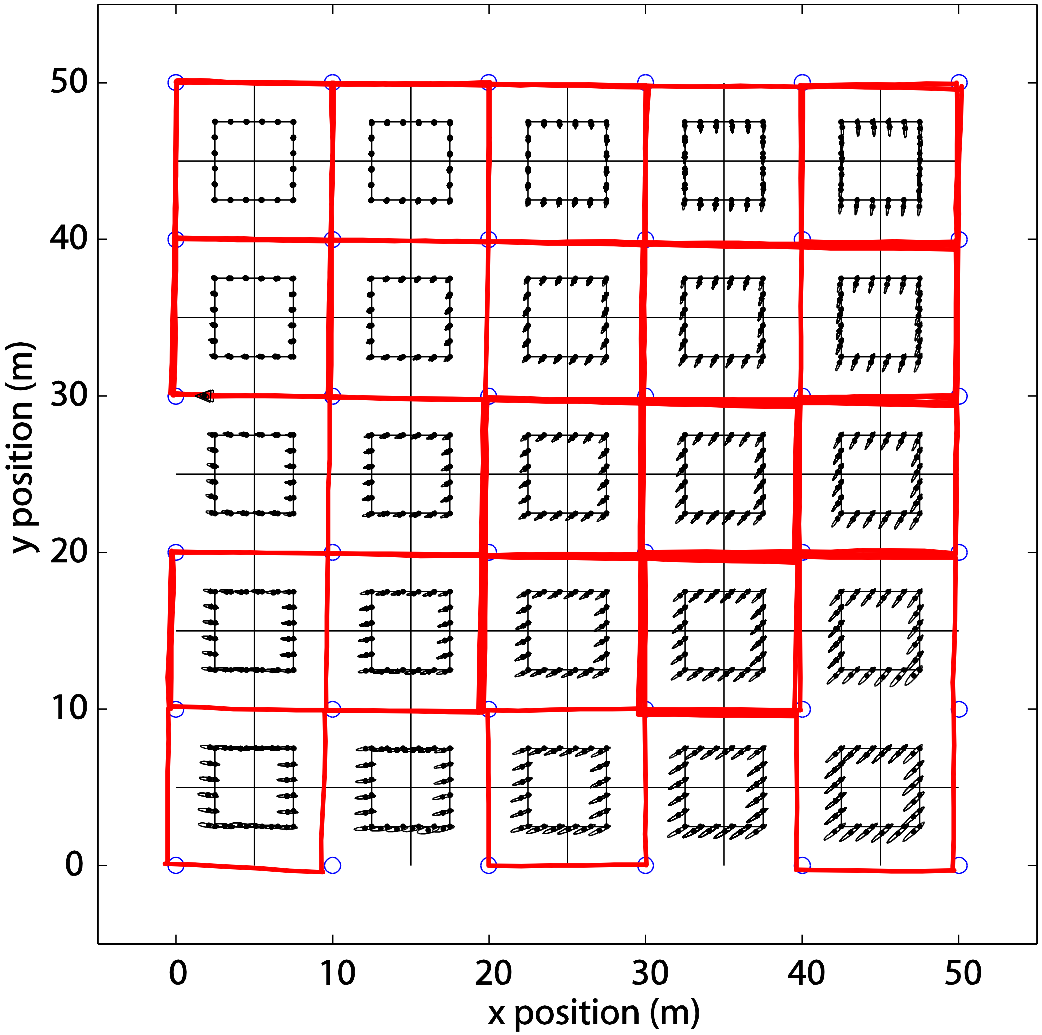
Loop Closing

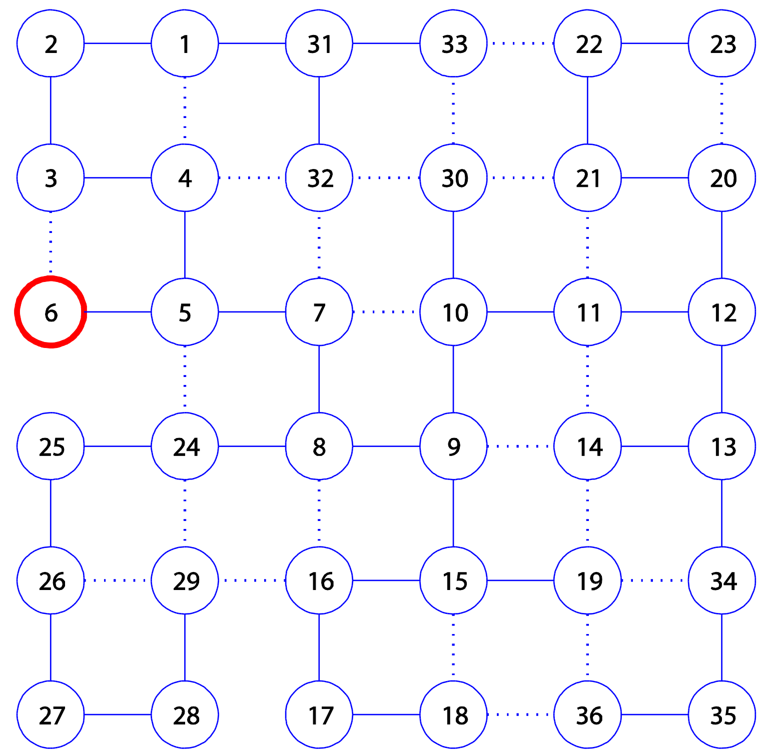


Optimal Global Map in O(n)

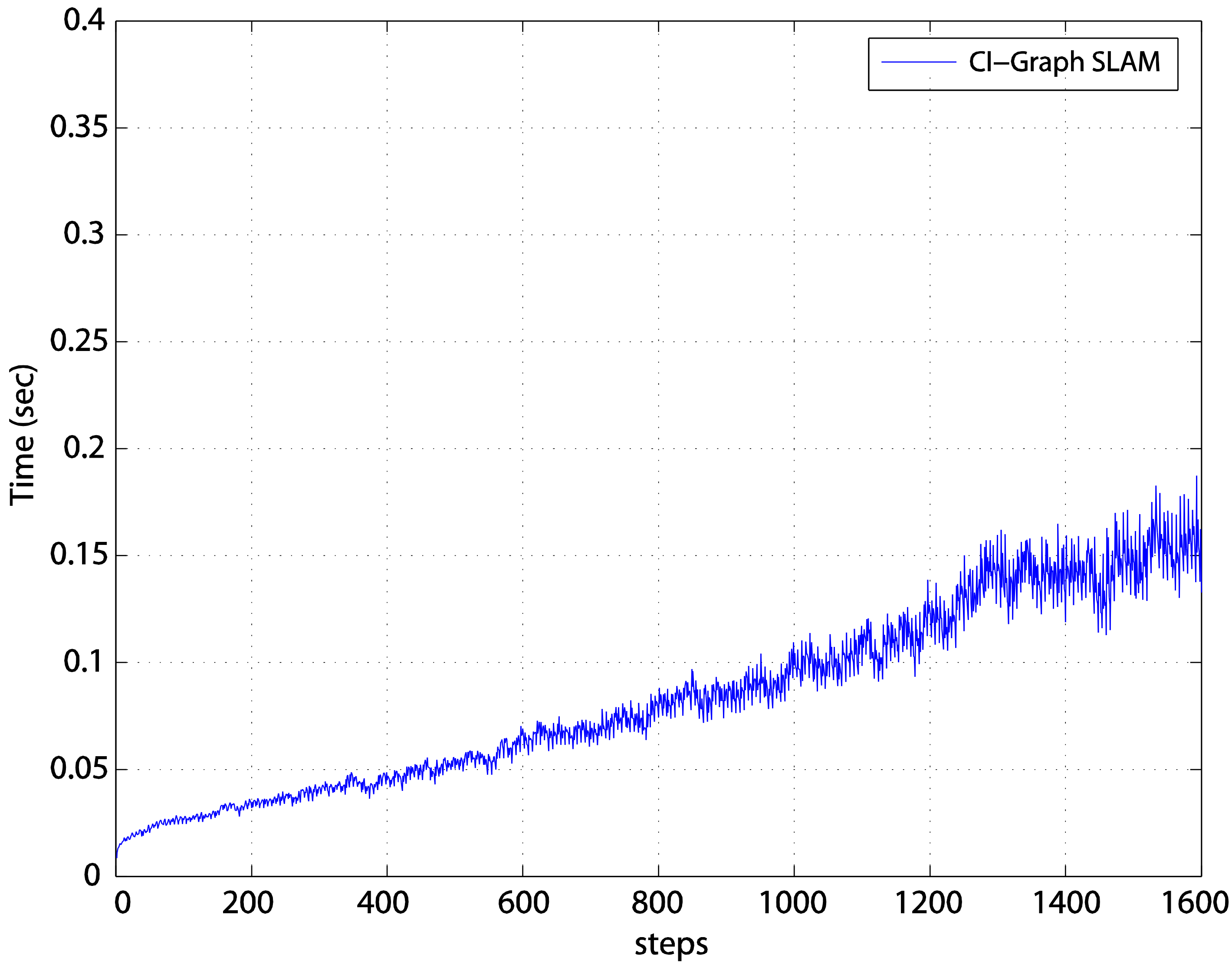
- Detect loop closing ($f_i = f_k$)
- Copy f_i to the common part of every map
- Impose $f'_i = f_k$ in the last map
- Back-propagate the correction

Graph of maps + Spanning tree





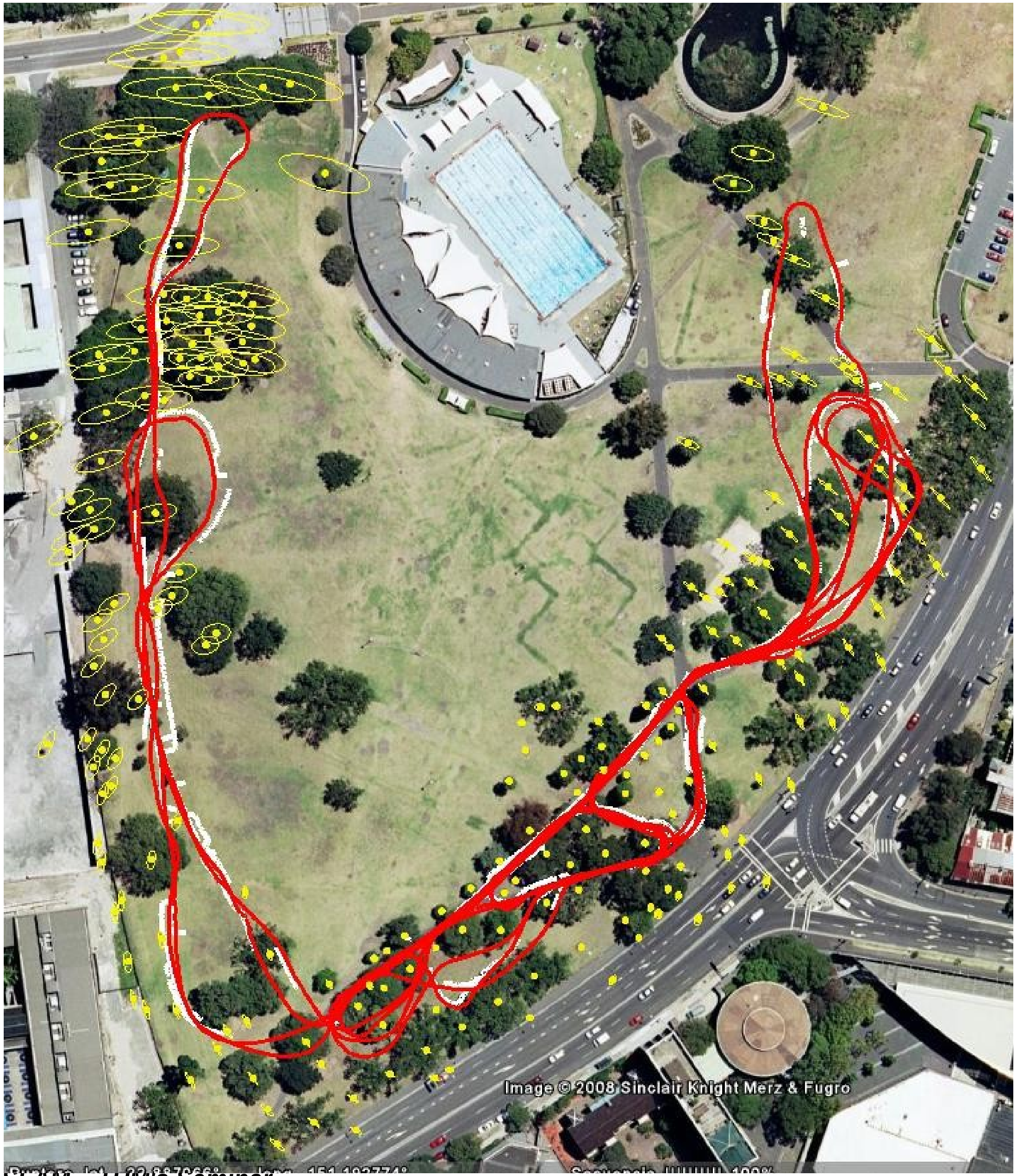
CI-Graph: close to O(n)



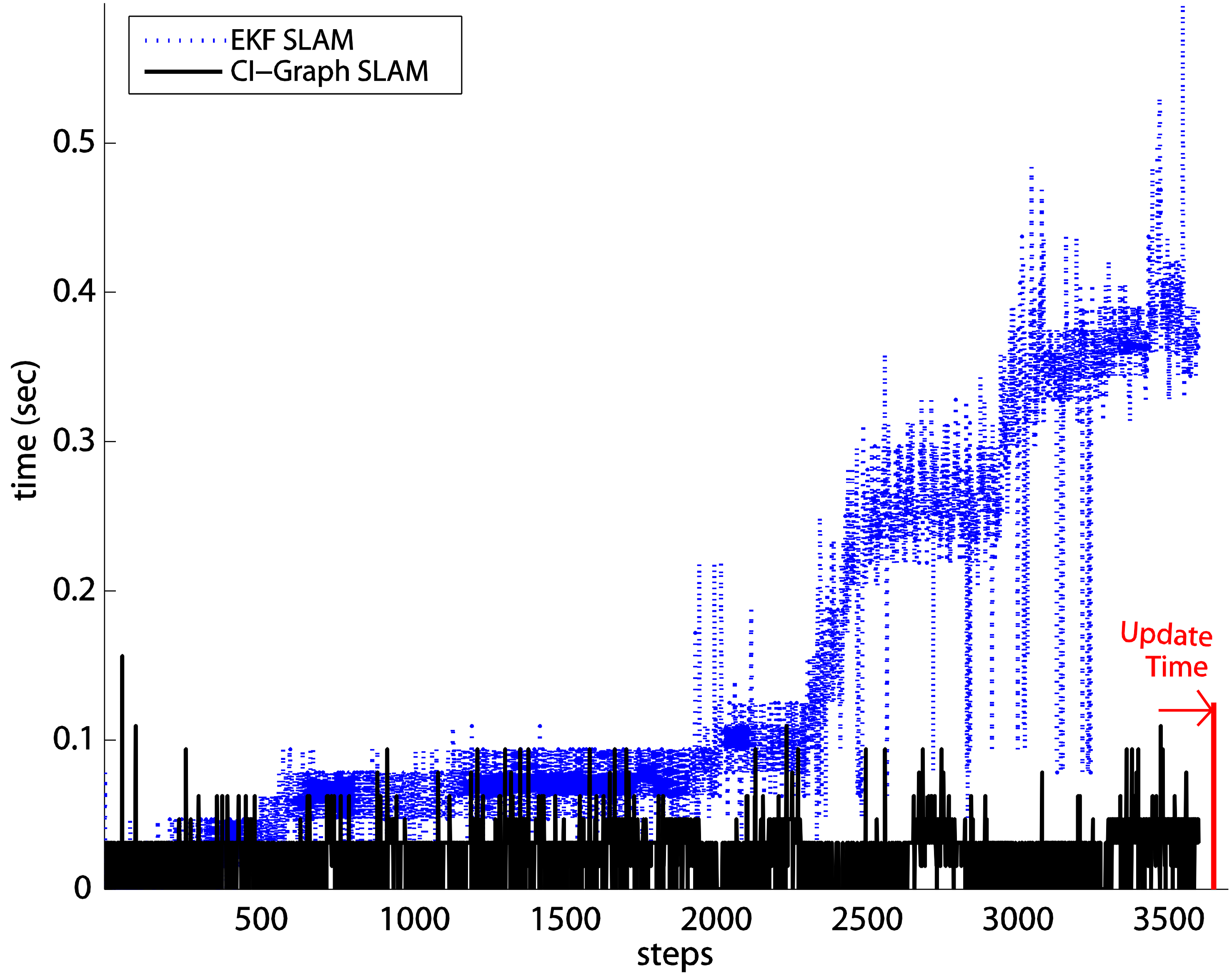
Victoria Park: Graph and Spanning Tree

P. Piniés, L. Paz and J.D. Tardós, CI-Graph: an efficient approach for Large Scale SLAM..
IEEE ICRA 2009 Kobe, Japan

CI-GRAPH: Victoria Park Results



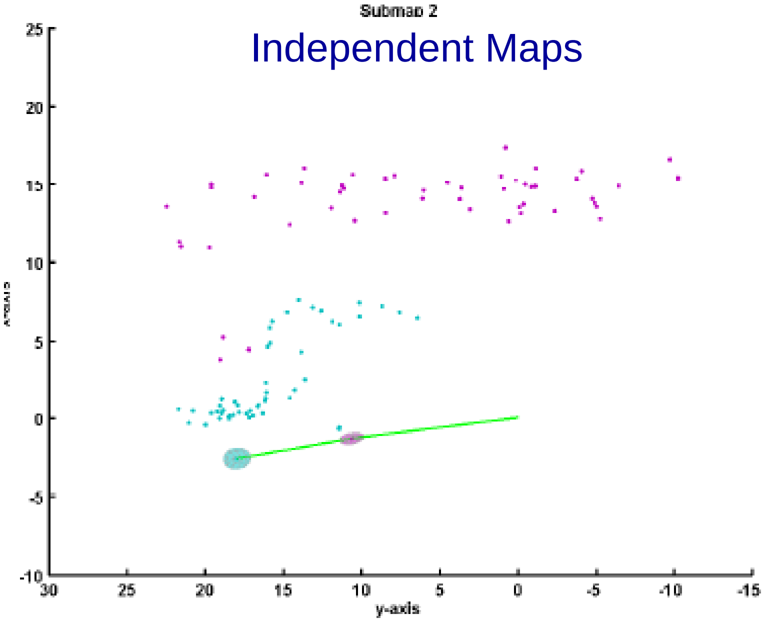
CI-GRAPH: Victoria Park Results

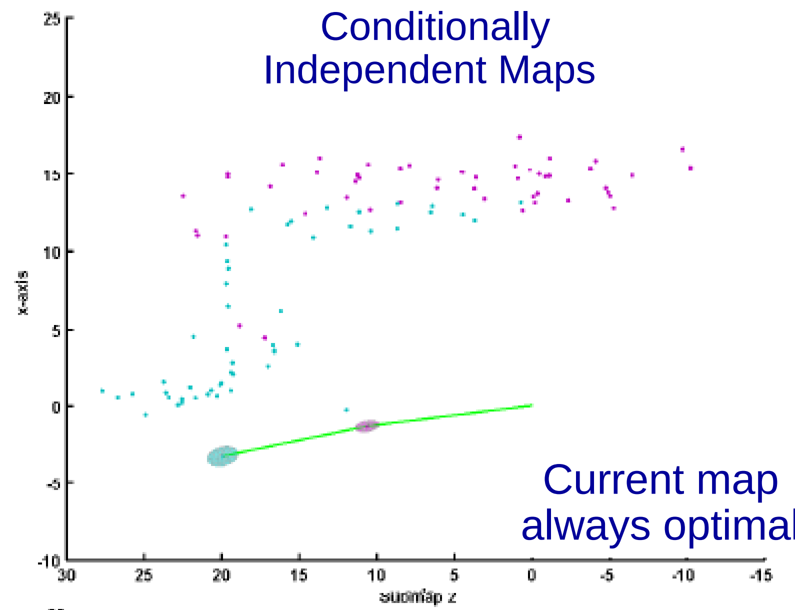


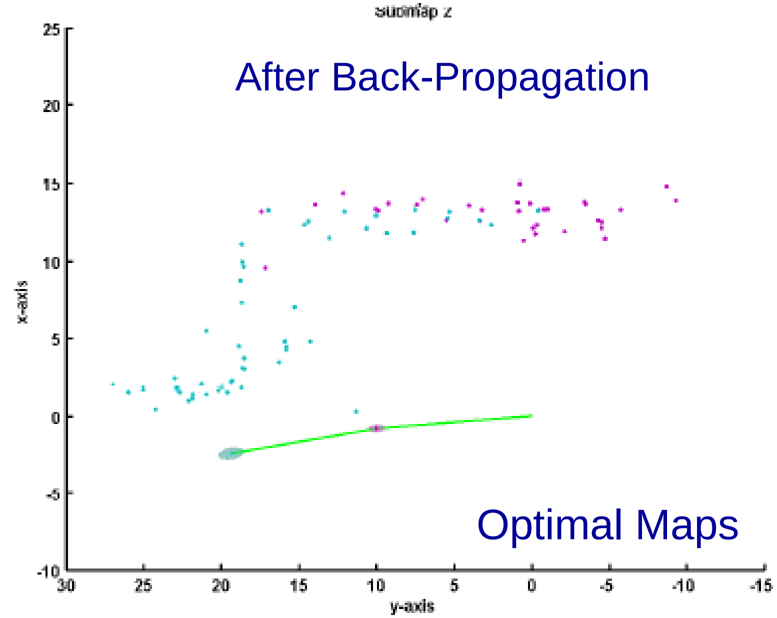
Results: Public Square (150m loop)

P. Piniés, J. D. Tardós. Large Scale SLAM Building Conditionally Independent Local Maps: Application to Monocular Vision. IEEE Transactions on Robotics 24(5): 1094 - 1106, Oct. 2008

Independence .vs. Conditional Independence

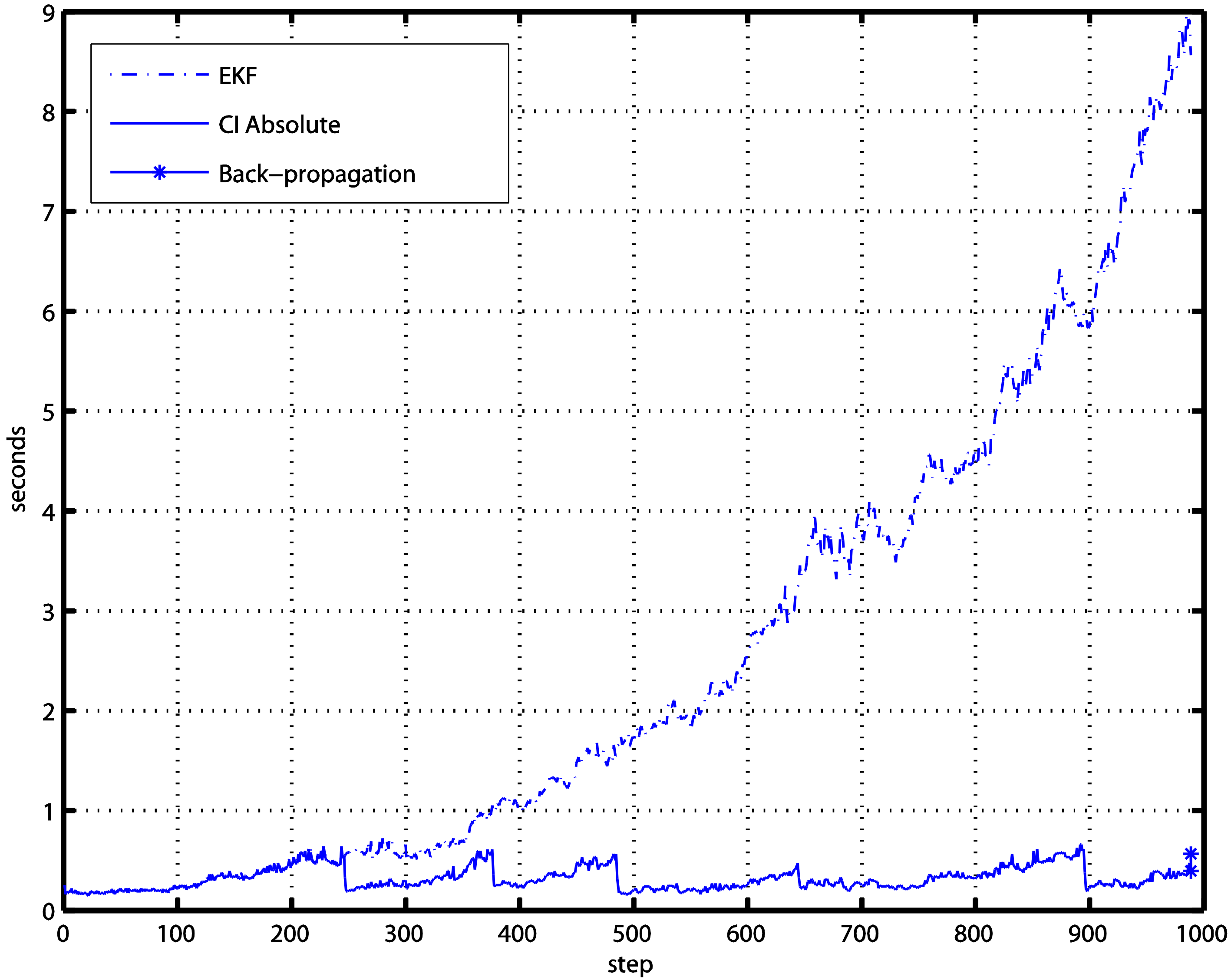






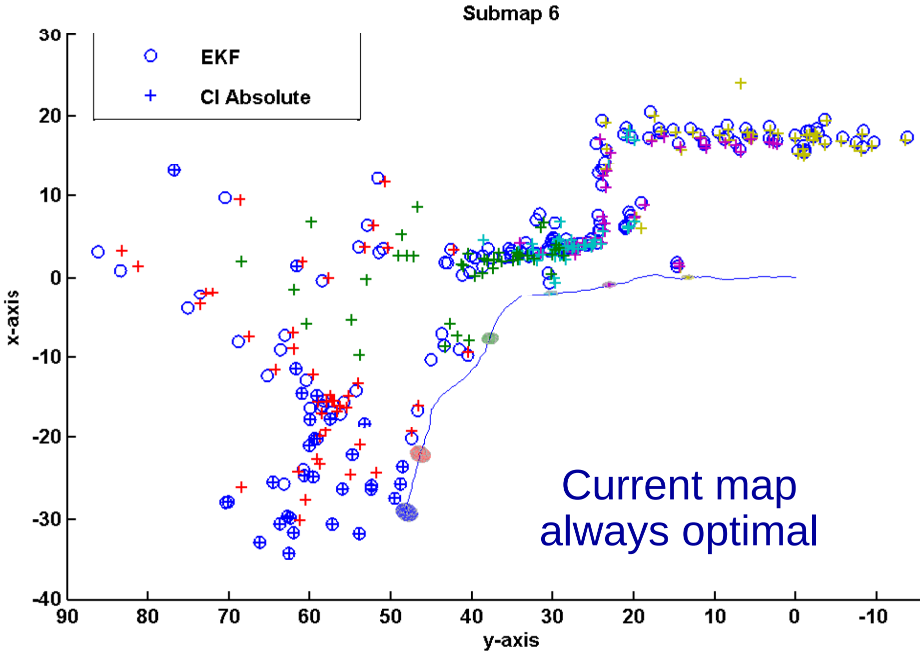
- Features and vehicle states cannot be shared between maps
- Sub-optimal maps
- Different map scales

CI-SLAM: Complexity

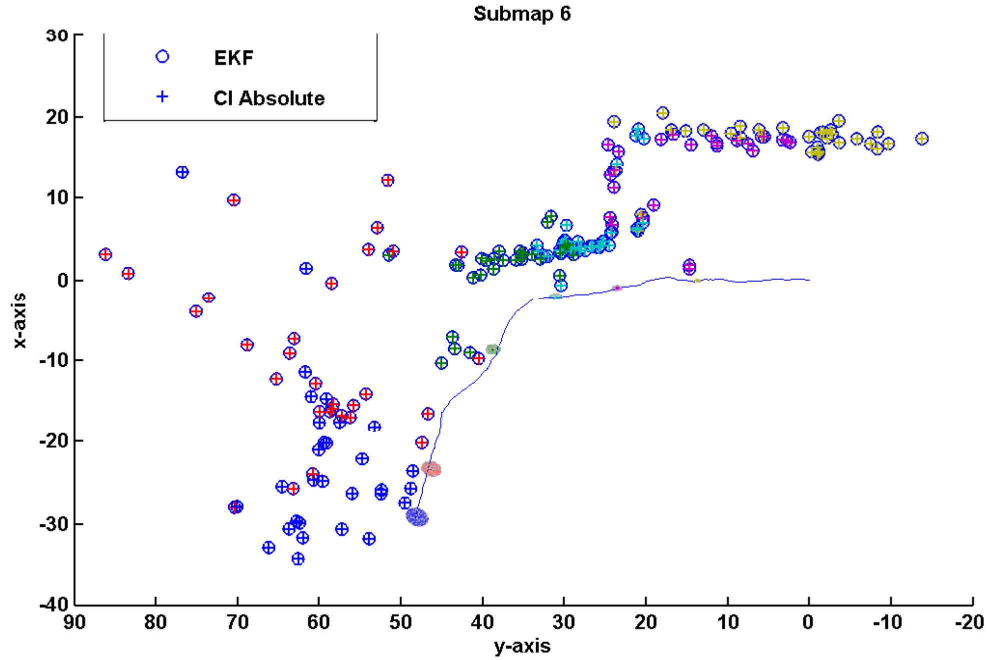


Same results than EKF-SLAM in O(1) .. O(n) !!

Conditionally Independent Maps

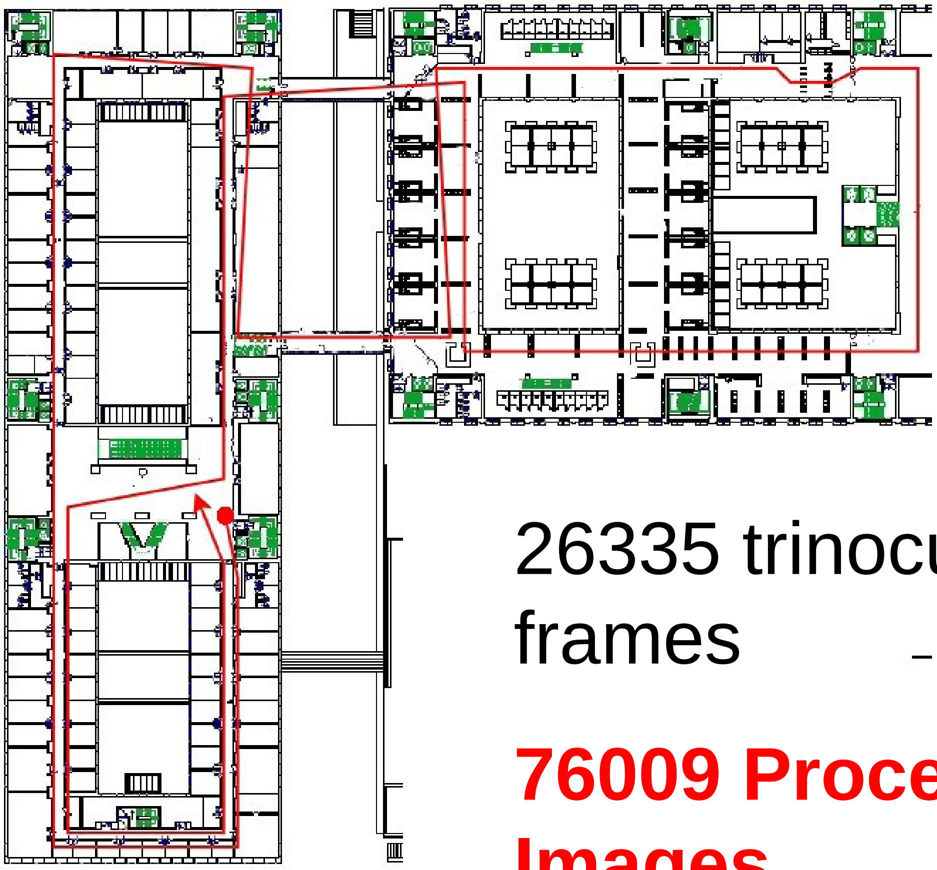


After Back-Propagation



RAWSEEDS dataset: Bicocca_2009-02-25b

www.rawseeds.org



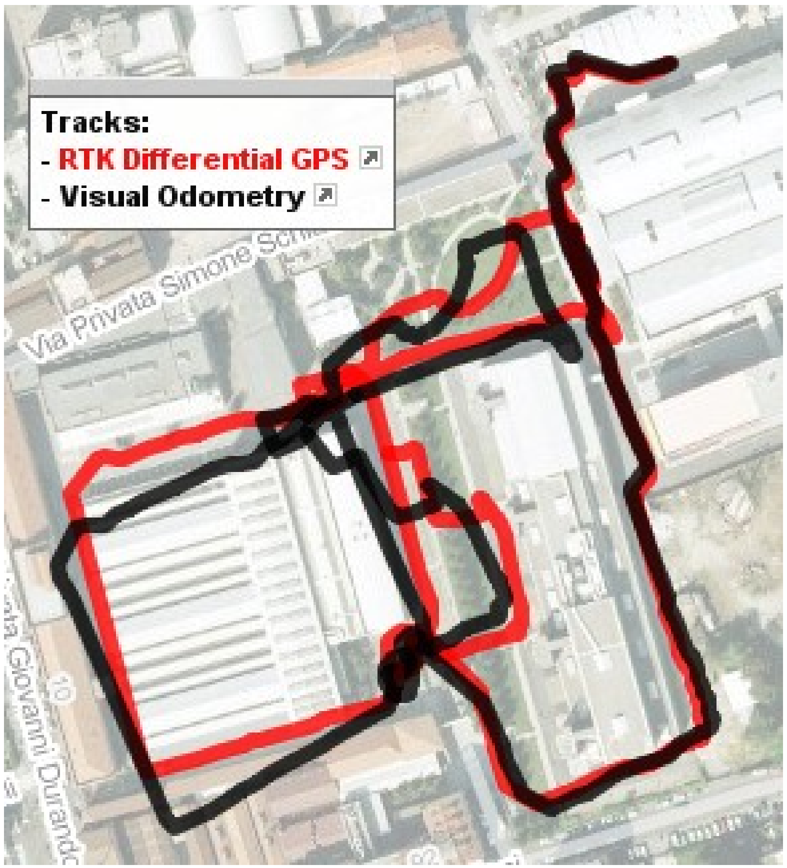
26335 trinocular frames

76009 Processed Images

700 m of Trajectory

RAWSEEDS dataset: Bovisa

www.rawseeds.org



34173 trinocular frames

102519 Processed Images

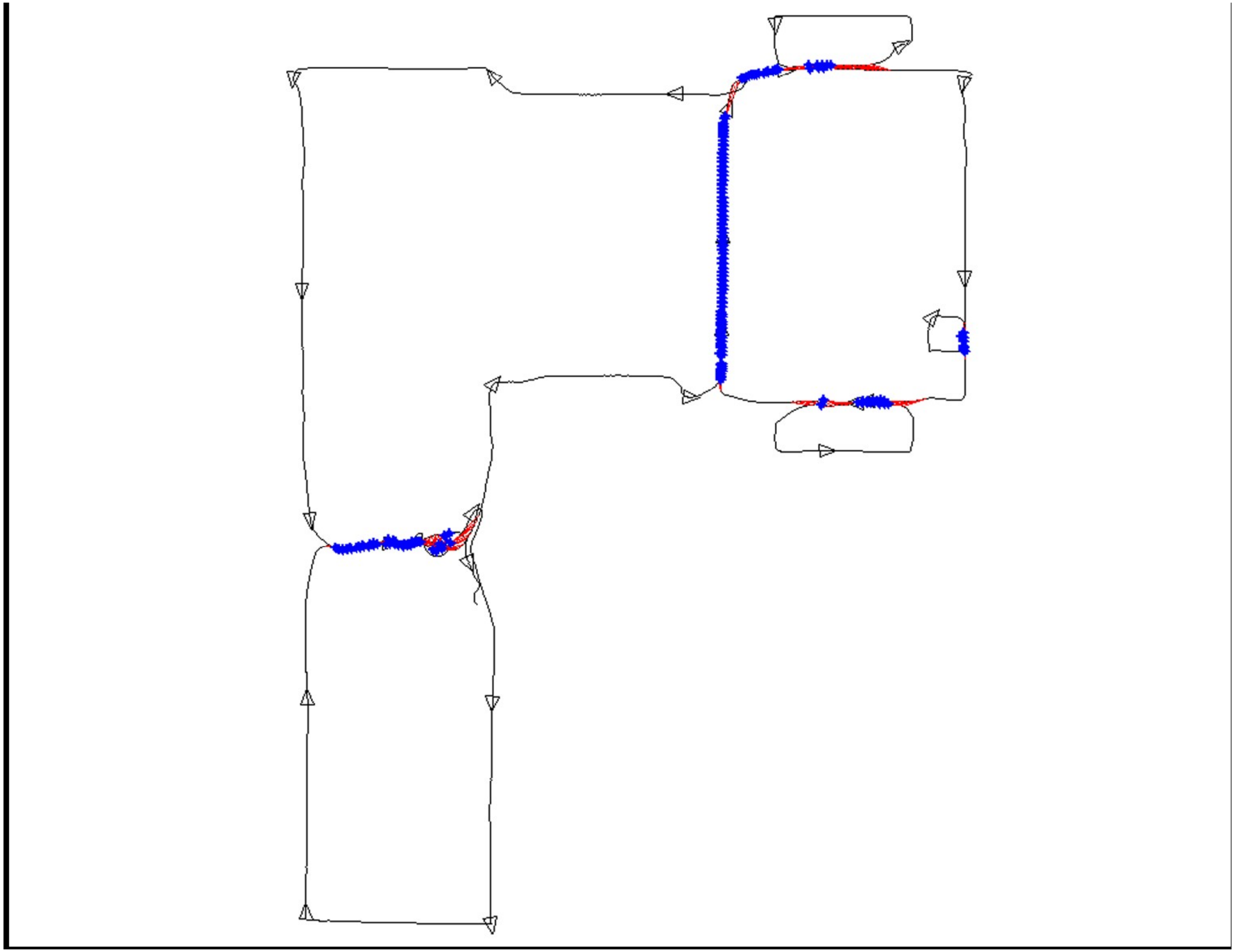
1.365 km of Trajectory

Appareance-Based Loop Detection

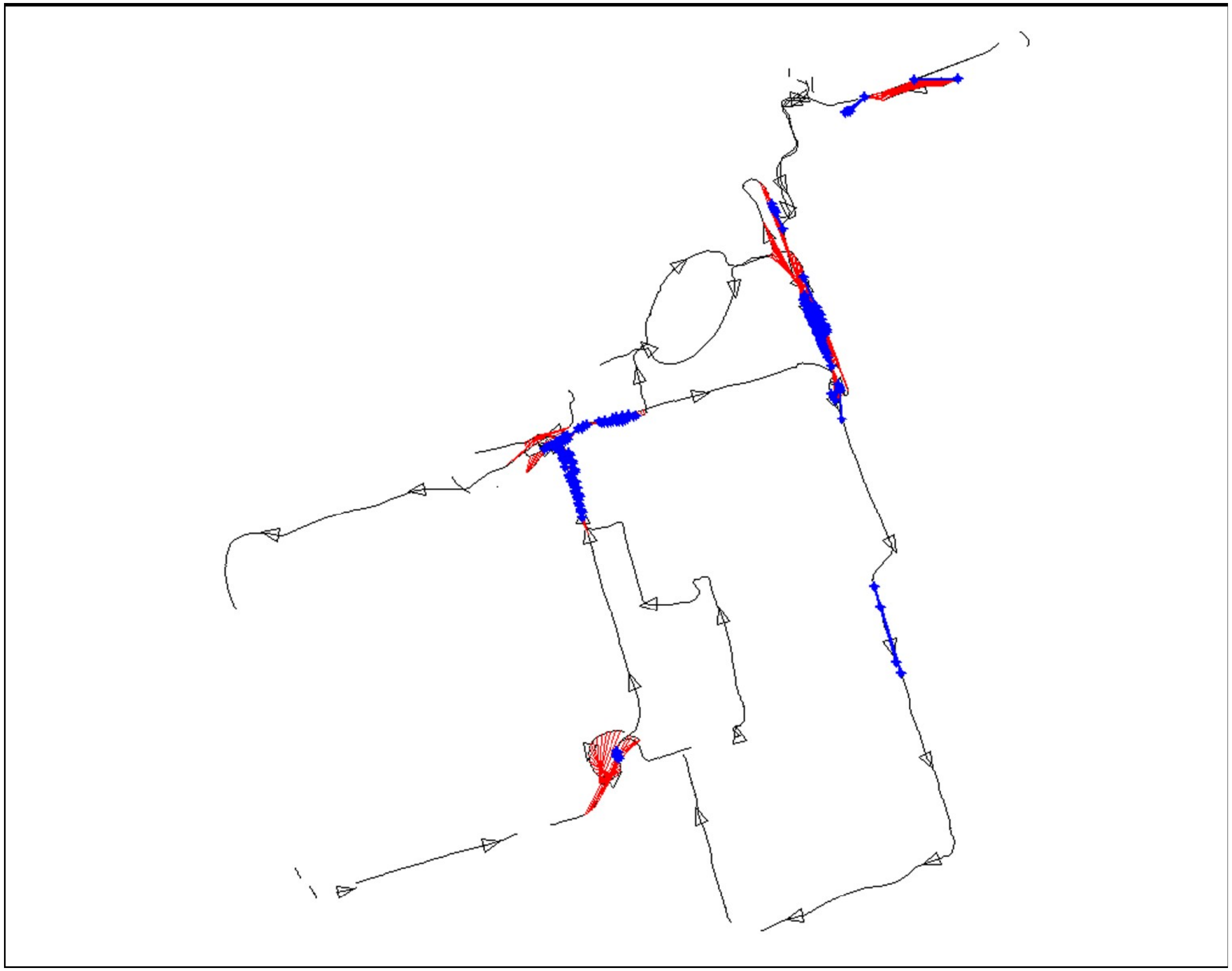
- Bag of words representation
  - SURF features clustered into visual words
  - Hierarchical vocabulary tree (Nister 2006)
  - Vocabulary trained off-line with a RAWSEEDS mixed dataset

- On-line loop detection
  - Learn one image per second
  - Match with previous learned images
  - Find the loop closing transformation solving the multi-view geometry

Appareance-Based Loop Detection



Appareance-Based Loop Detection

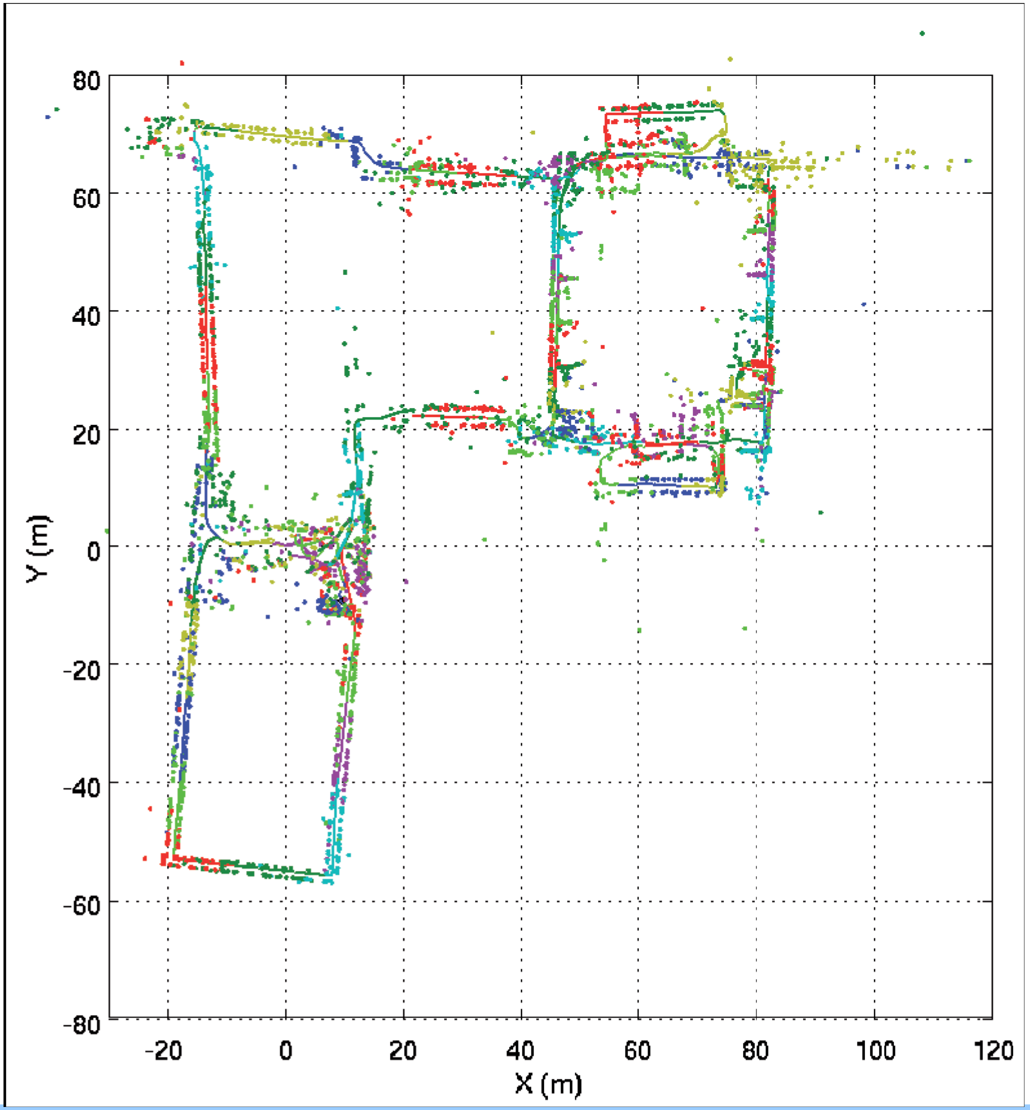


Example of Loop Succesfully Detected

Missed Loop Closure

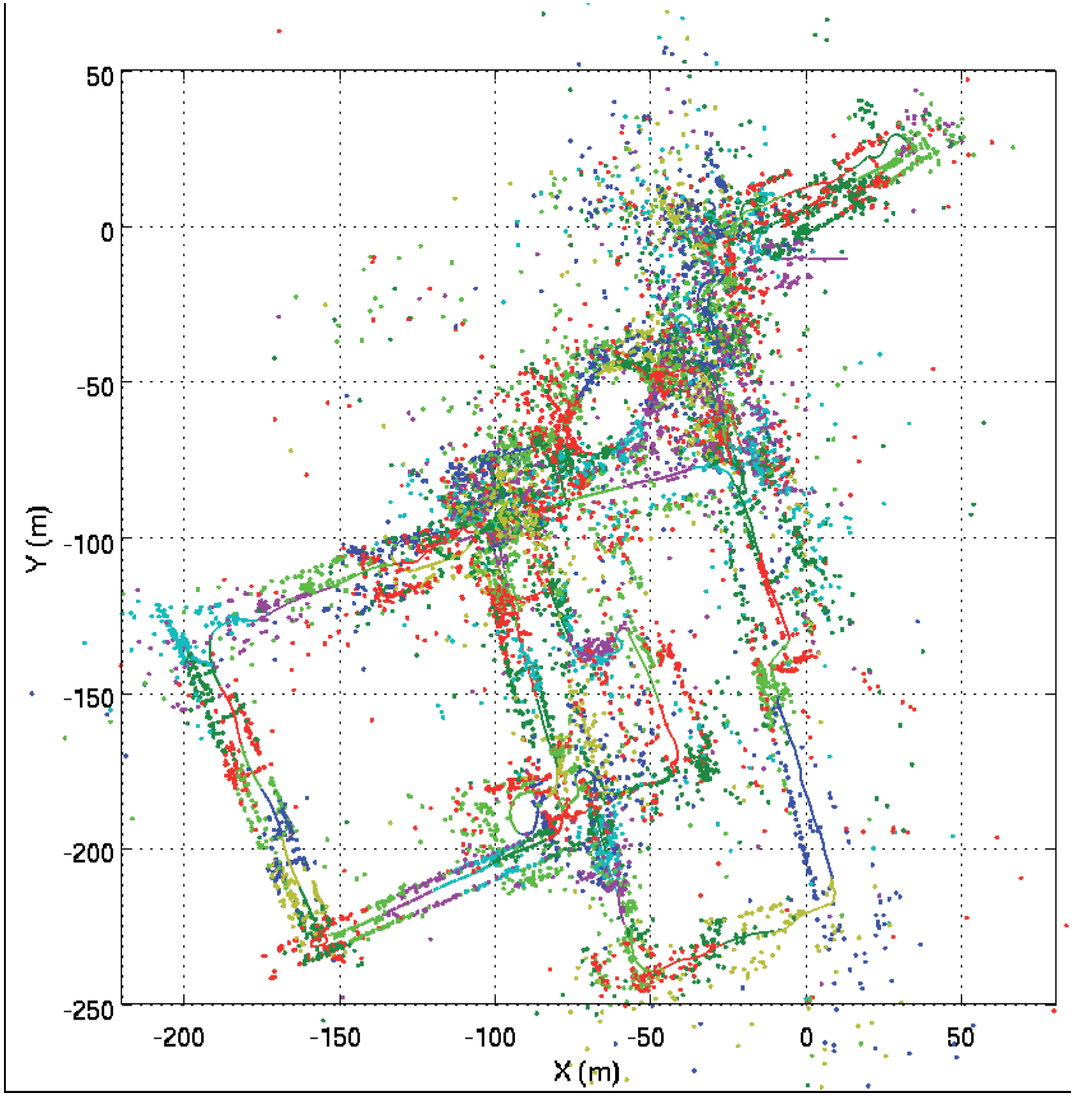
Example of Loop Succesfully Detected

CI-Graph Results: 700 mts of Trajectory



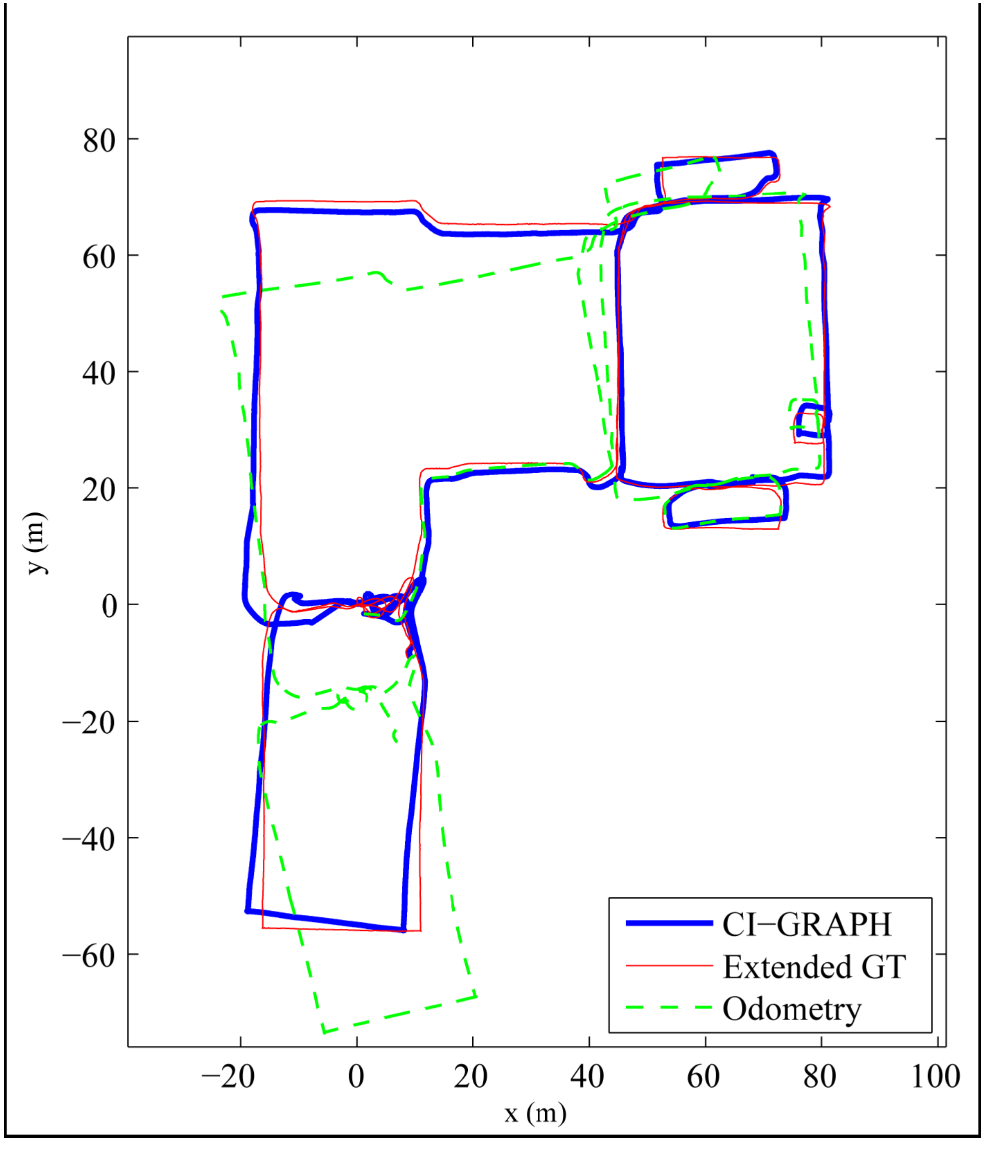
P. Piniés, L. M. Paz, D. Gálvez, J. D. Tardós: **CI-Graph SLAM for 3D reconstruction of large and complex environments using a multi-camera system**. Journal of Field Robotics, Oct 2010

CI-Graph Results: 1.365 km of Trajectory

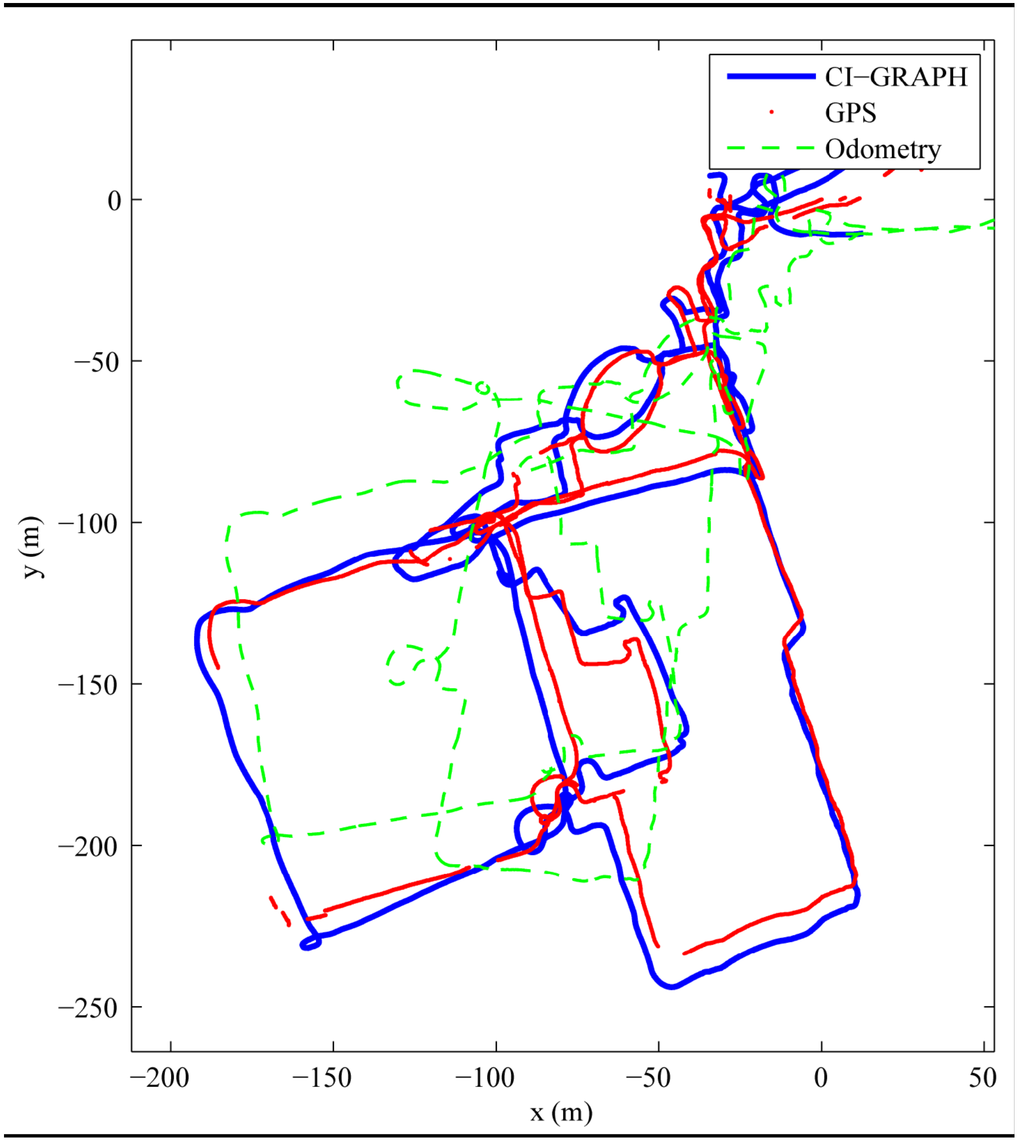


P. Piniés, L. M. Paz, D. Gálvez, J. D. Tardós: **CI-Graph SLAM for 3D reconstruction of large and complex environments using a multi-camera system**. Journal of Field Robotics, Oct 2010

CI-Graph Results: 700 mts of Trajectory



CI-Graph Results: 1.365 km of Trajectory



SLAM in Large Environments

Ord	Method	Publication	Memory	T local	T Loop	Aproxim.	Coord.
1	EKF	Smith 1986	$O(n^2)$	$O(n^2)$	$O(n^2)$		Abs
2	SEIF	Thrun 2004	O(n)	O(1) amort	$O(n^2)$	Sparsif.	Abs
3	ESEIF	Walter 2007	O(n)	O(1)	$O(n^2)$	Kidnap	Abs
7	ESDF	Eustice 2006	O(p)	O(1)	$O(p^2)$		Abs
8	Postpon.	Knight 2001	$O(n^2)$	O(n)	$O(n^2)$		Abs
9	CEKF	Guivant 2001	$O(n^2)$	O(n)	$O(n^2)$		Abs
13	CLSF	Williams 2002	$O(n^2)$	O(1)	$O(n^2)$		Local
14	Map Joining	Tardós 2002	$O(n^2)$	O(1)	$O(n^2)$		Local
4	D&C SLAM	Paz 2007	$O(n^2)$	O(1)	O(n) amort.		Local
5	CTS	Newman 2003	O(n)	O(1)	--	FD, Loop	Local
6	Atlas	Bosse 2004	O(n)	O(1)	--	FD, Loop	Local
10	H-SLAM	Estrada 2005	O(n)	O(1)	O(n)	FD	Local
11	TJTF	Paskin 2003	O(n)	O(1)	O(n)	Sparsif.	Abs
12	Treemap	Frese 2006	O(n)	O(1)	O(log n); O(n)		Abs
17	Graph SLAM	Thrun 2006	$O(p^2)$	--	$O(p^3)$		Abs
	SAM	Dellaert 2006	O(n+p)	--	O(n+p) ??		Abs
	FastSLAM	Montemerlo 20(	O(Kn)	O(K)	O(K)		Abs
	CI-SLAM	Piniés 2007	O(n)	O(1)	O(n)		Local
	CI-GRAPH	Piniés 2009	O(n)	O(1)	O(n) ??		Local

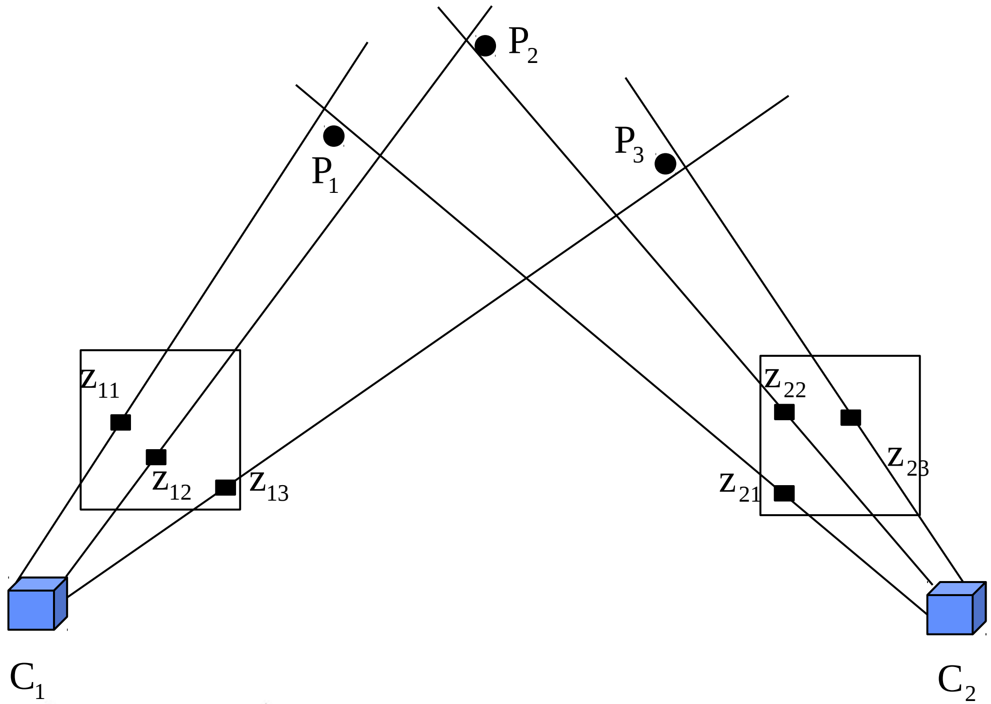
Linearization errors are reduced

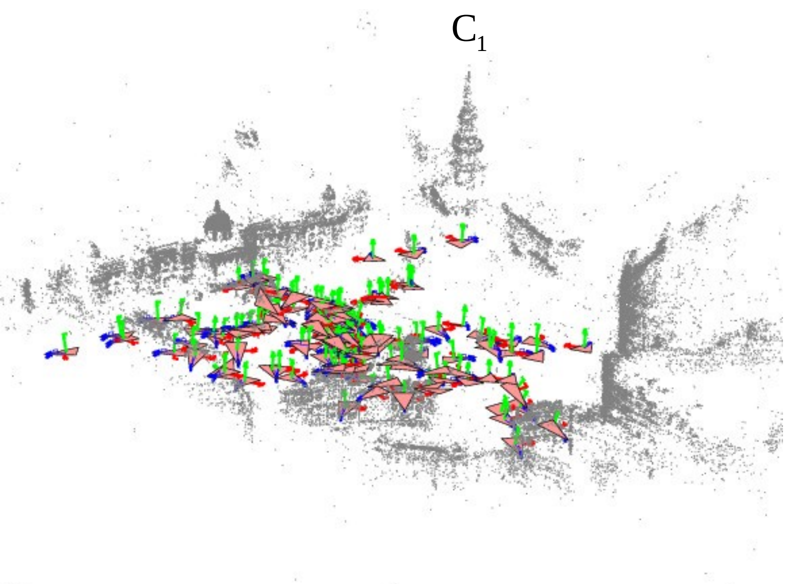
Can use Joint Compatibility for D.A.

Outline

1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

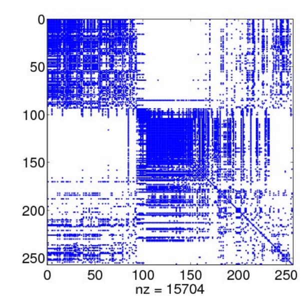
Context



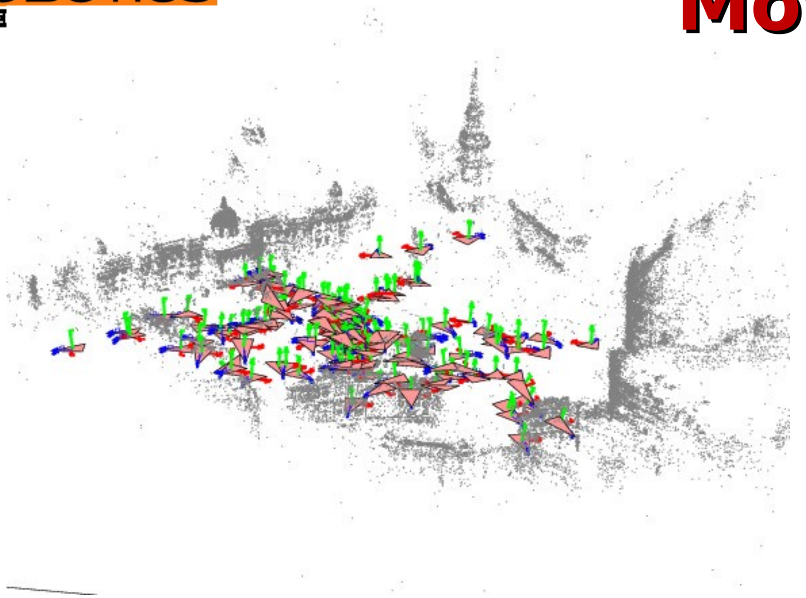


Solve at each iteration
A x = b

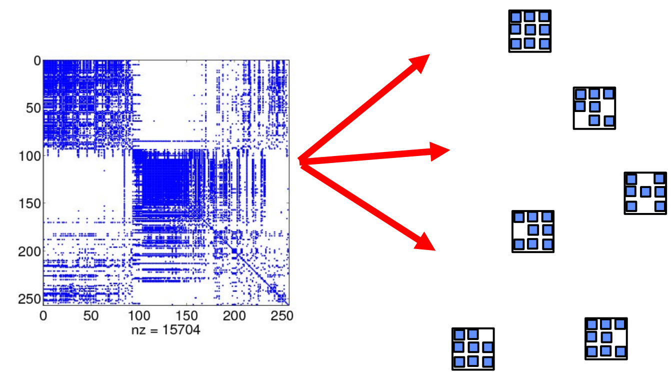
A =



Motivation



$A =$



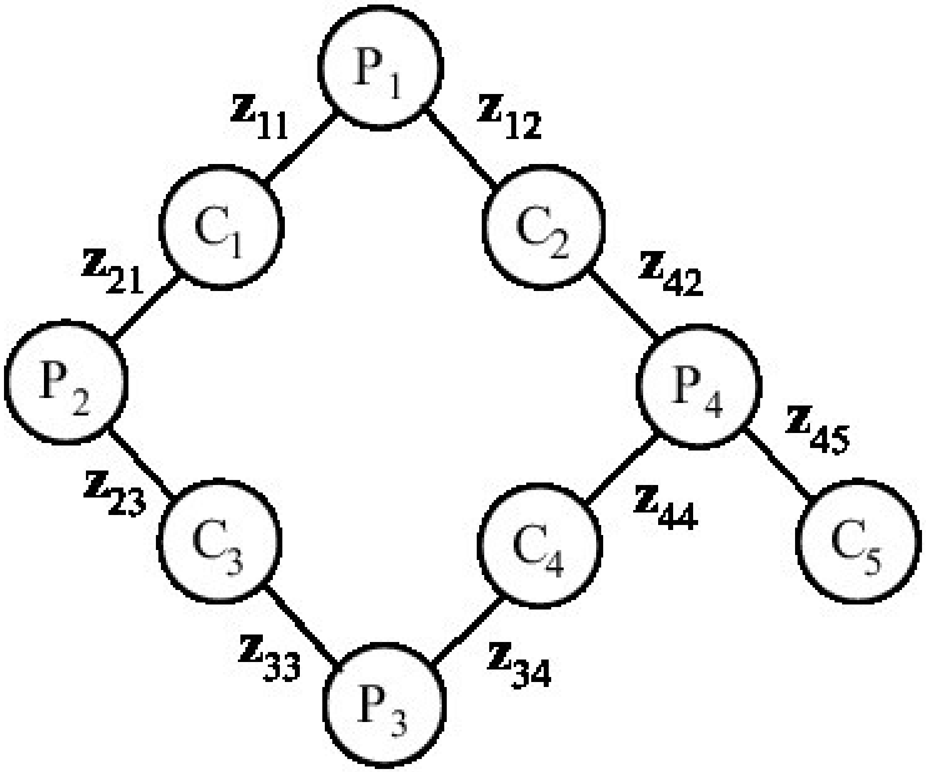
Goal: Develop a distributed algorithm

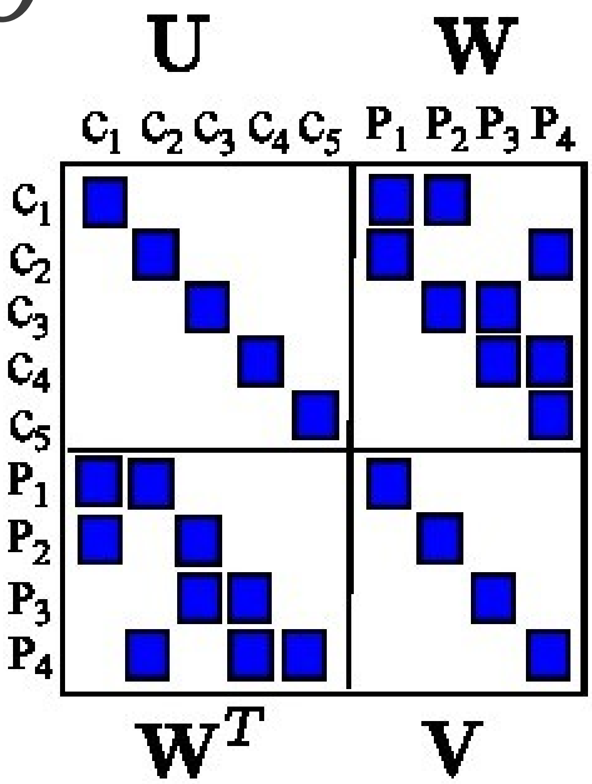
Examples:

- distributed BA using a team of robots
- solve big systems in a cluster of computers

Primary structure

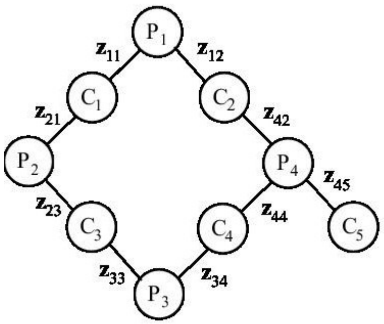
$$A\,x = b$$

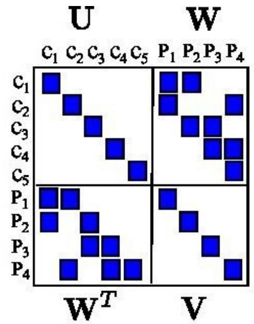




Benefits of the Primary structure

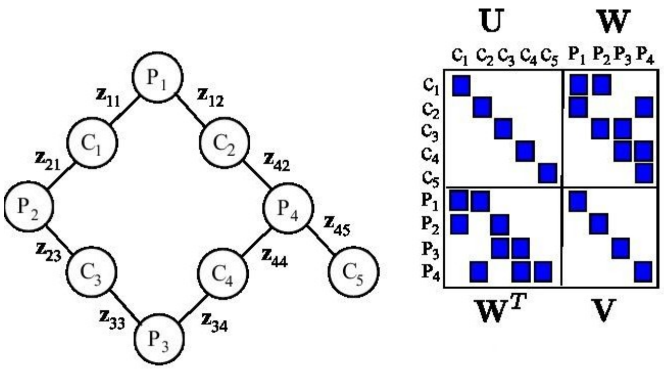
1. Get rid of the points (np >> nc):

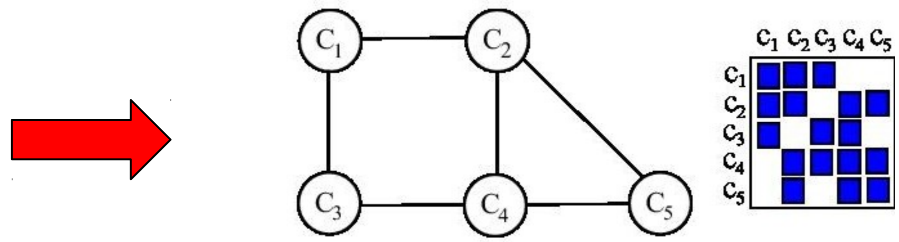




Benefits of the Primary structure

1. Get rid of the points (np >> nc): Variable elimination



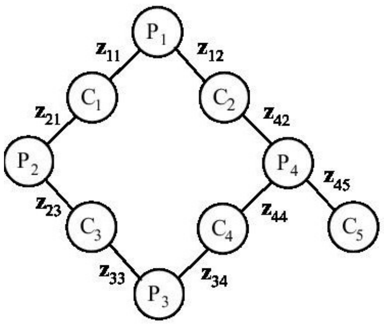


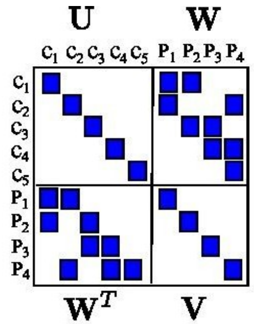
$$U^m = U - W\,V^{-1}\,W^T$$

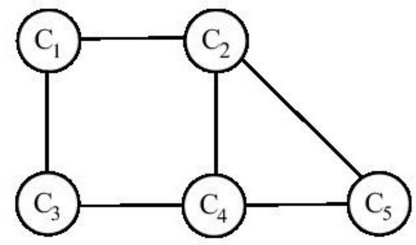
$$u^m = u - W\,V^{-1}\,v$$

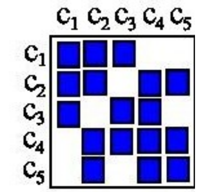
Benefits of the Primary structure

1. Get rid of the points (np >> nc): Schur complement









$$U^m = U - W V^{-1} W^T$$

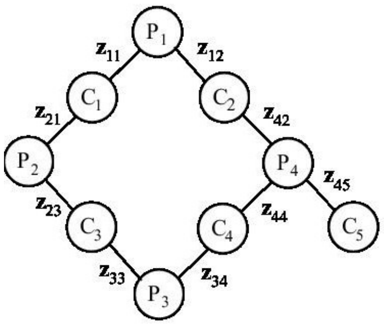
$$u^m = u - W V^{-1} v$$

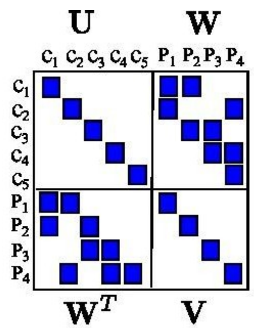
2. Solve for the reduced camera matrix

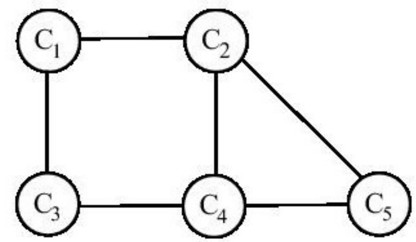
$$U^m x_C = u^m$$

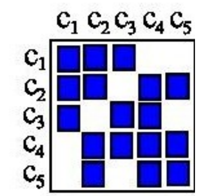
Benefits of the Primary structure

1. Get rid of the points (np >> nc): Schur complement









$$U^m = U - W V^{-1} W^T$$

$$u^m = u - W V^{-1} v$$

- **Solution linear in the number of points**
- **Reduced camera matrix decomposable**

2. Solve for the reduced camera matrix

$$U^m x_C = u^m$$

3. Back-substitution to obtain points solution

$$V x_P = v - W^T x_C^{sol}$$

DBA

Algorithm proposed:

1. Use the decomposable structure of the reduced camera matrix to distribute the calculations

2. Take advantage of the natural Primary Structure to efficiently solve for the points

Decompose Camera Graph

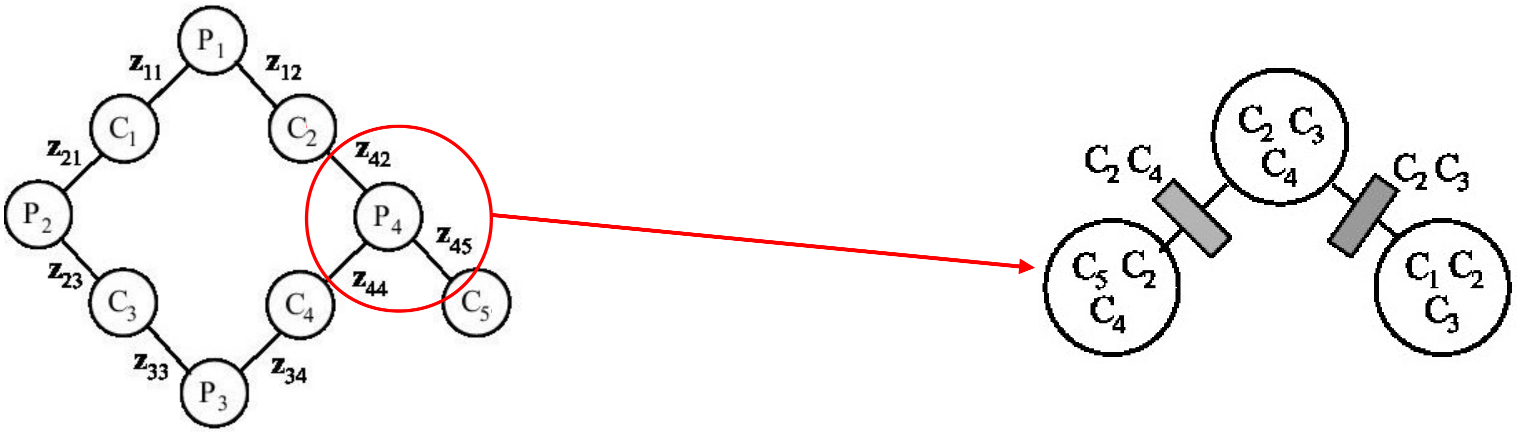
An ordered method to decompose the camera graph:
Junction Tree

All cliques of the original graph are at least in one of the tree nodes

M. A. Paskin and G. D. Lawrence, "Junction tree algorithms for solving sparse linear systems", Computer Science Division (EECS), University of California, Berkeley, California 94720, Tech. Rep. UCB/CSD-03-1271, September 2003.

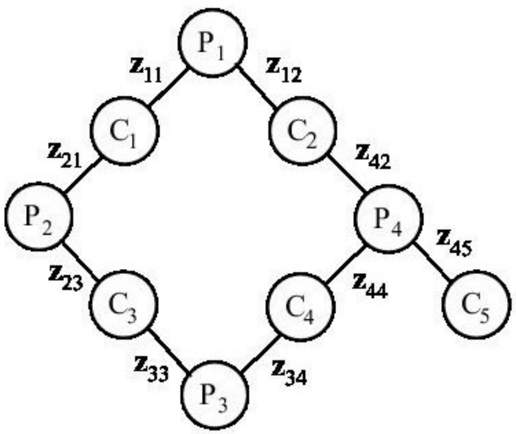
Locally maintain a Primary Structure

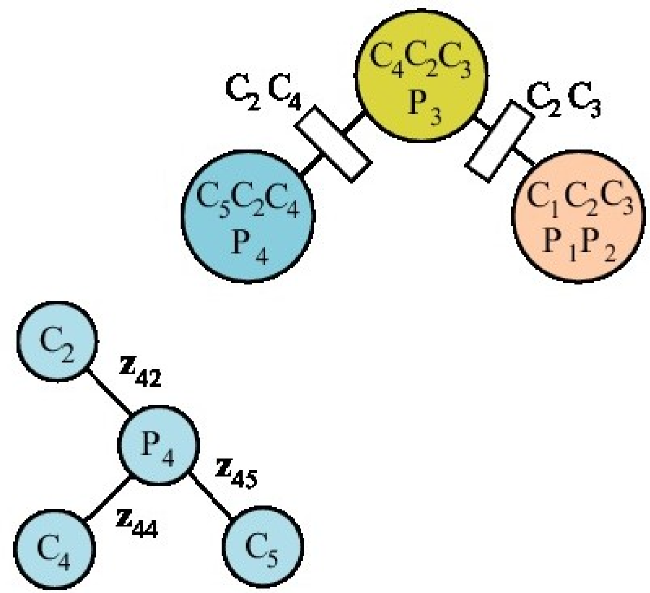
Distribute points and measurements to corresponding clique nodes



Locally maintain a Primary Structure

Each tree node maintains a Primary Structure: Solving for points is linear.





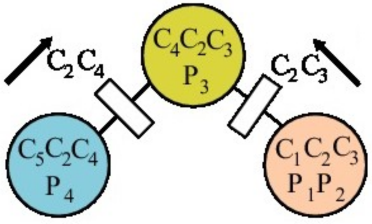
Solving the decomposed system

1. For each tree node obtain the reduced camera matrix (Shur points)

Solving the decomposed system

1. For each tree node obtain the reduced camera matrix (Shur points)

2. Collect common camera information from leaves to root (Schur cam)

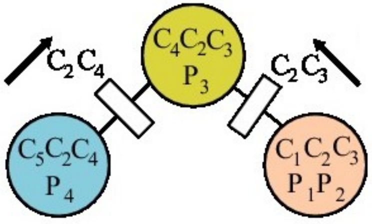


Solving the decomposed system

1. For each tree node obtain the reduced camera matrix (Shur points)

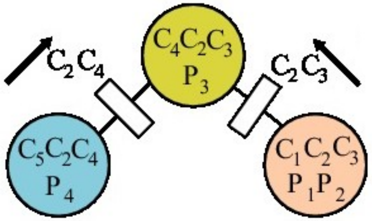
2. Collect common camera information from leaves to root (Schur cam)

3. Solve at root



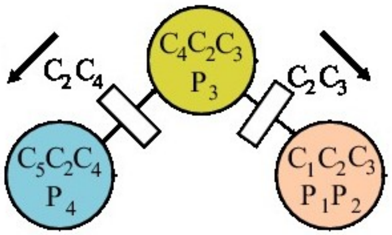
Solving the decomposed system

1. For each tree node obtain the reduced camera matrix (Shur points)



2. Collect common camera information from leaves to root (Schur cam)

3. Solve at root

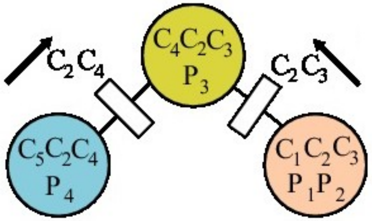


4. Distribute common camera information from root to leaves (Back cam)

Solving the decomposed system

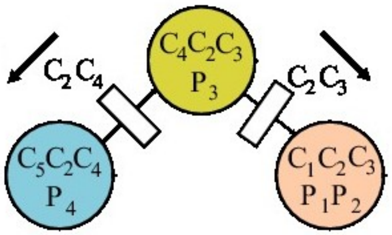
1. For each tree node obtain the reduced camera matrix (Shur points)

2. Collect common camera information from leaves to root (Schur cam)



3. Solve at root

4. Distribute common camera information from root to leaves (Back cam)

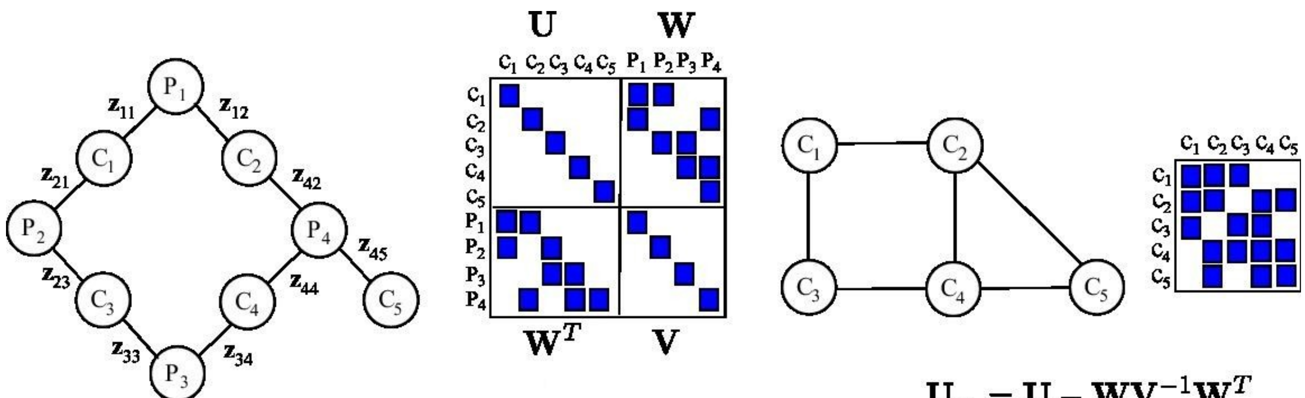


5. Solve for the points at each node (Back points)

Results

We compare two own implementations:

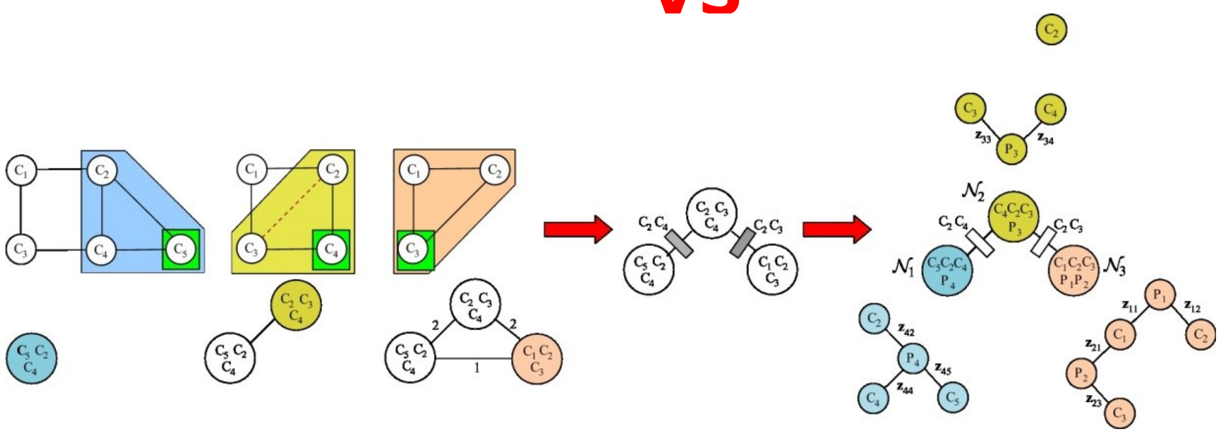
Standard BA



$$\mathbf{U}_m = \mathbf{U} - \mathbf{W}\mathbf{V}^{-1}\mathbf{W}^T$$

VS

DBA



Results

Implementation details: Pentium Core Quad 2.6 GHz, 4Gb RAM

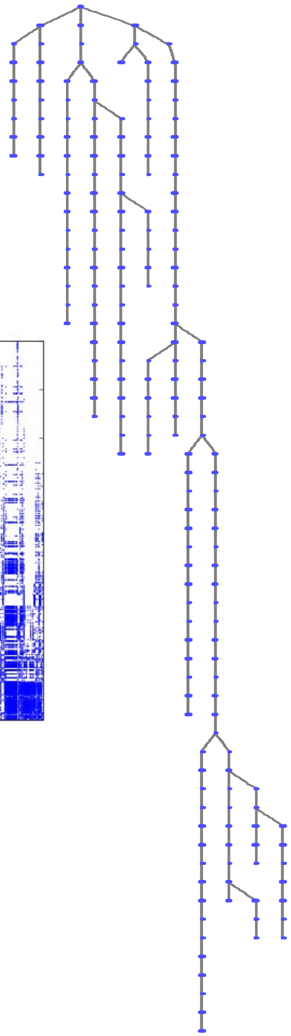
For both algorithms matrices are built using C mex functions

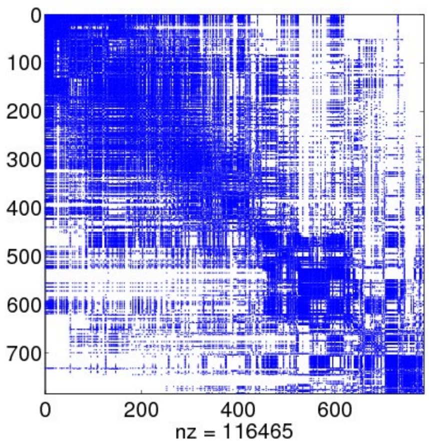
For standard BA: U, V and W are sparse and we use the MATLAB’s CHOLMOD built-in solver

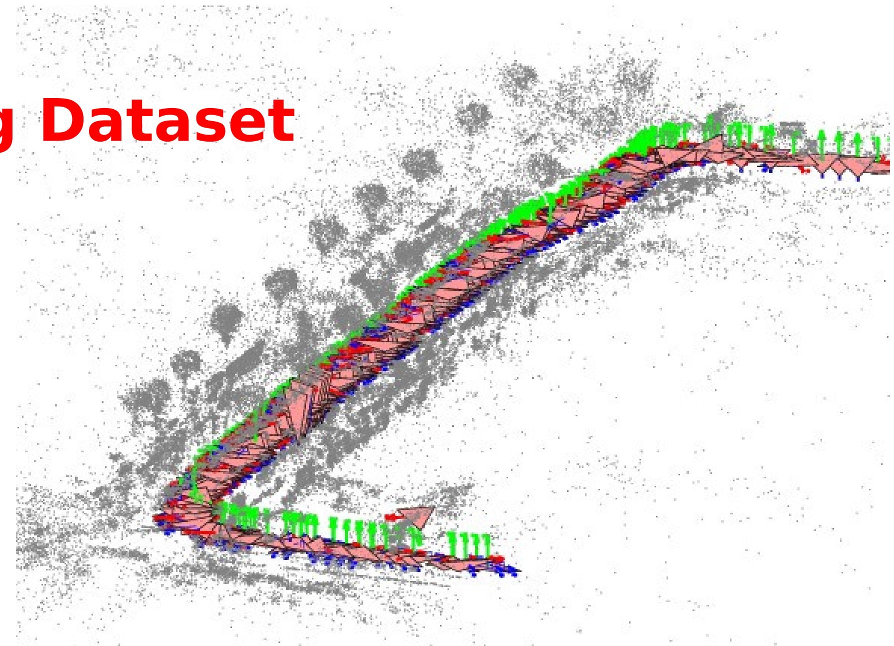
For DBA: Matrices in each tree node are usually small so we use a dense representation and MATLAB dense CHOLESKY solver.

Experiments

LadyBug Dataset





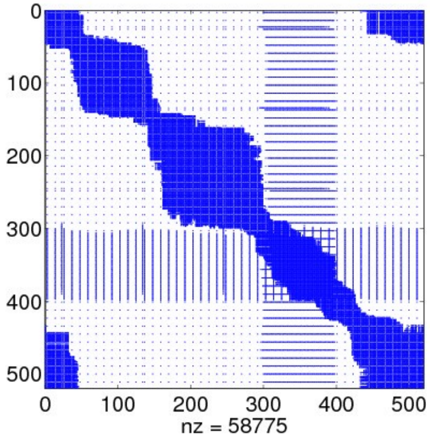


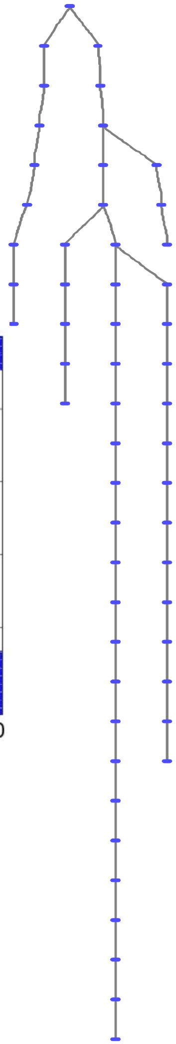
Experiment	nC	nP	nz	T improv %
LADYBUG	783	83581	372940	57,45

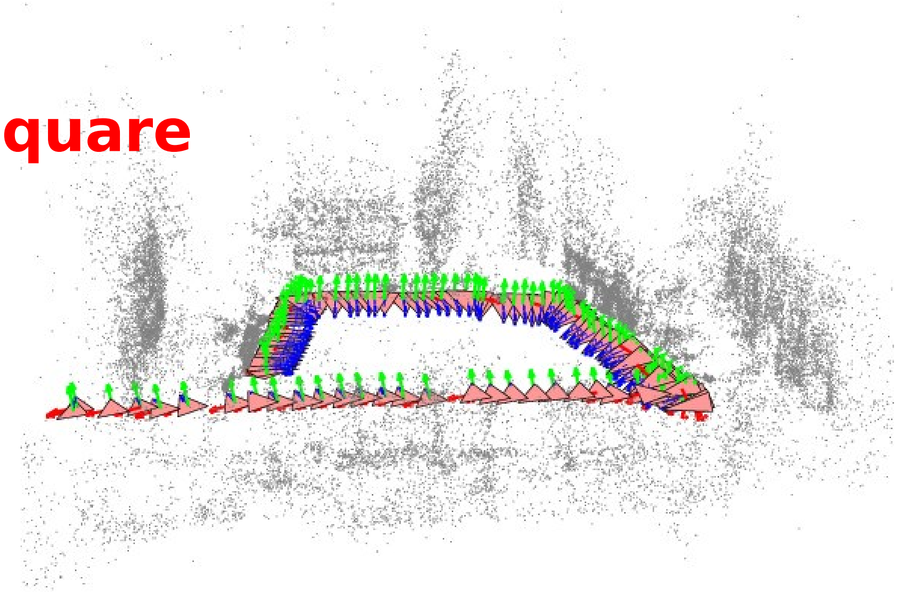
Bundle Adjustment in the Large
Sameer Agarwal, Noah Snavely, Steven M. Seitz and Richard Szeliski, European Conference on Computer Vision, 2010
http://grail.cs.washington.edu/projects/bal/

Experiments

Public Square





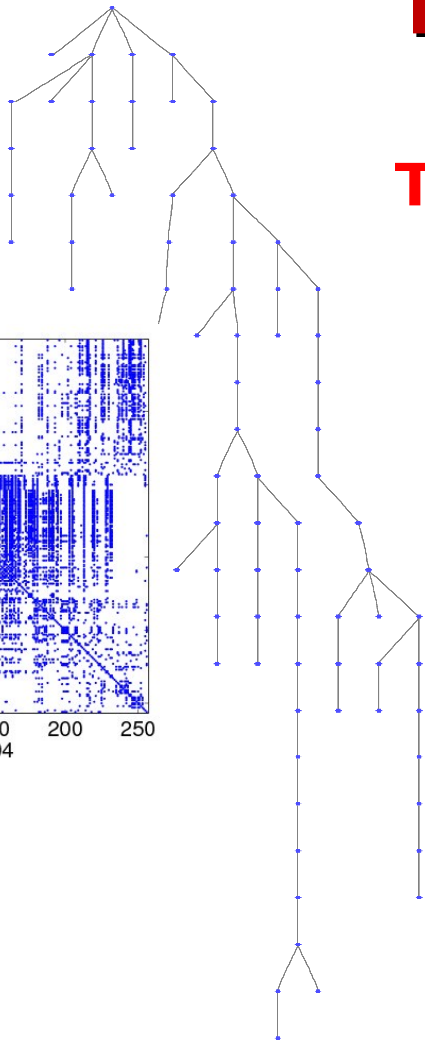


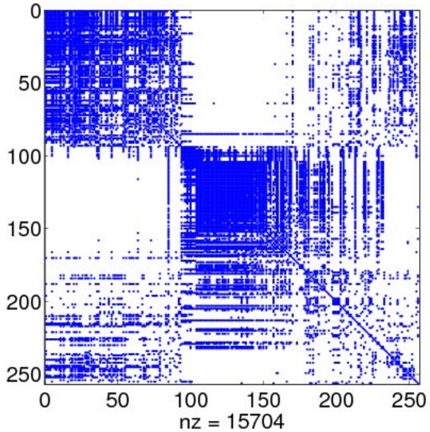
Experiment	nC	nP	nz	T improv %
PUBLIC SQUARE	519	32068	286162	46,52

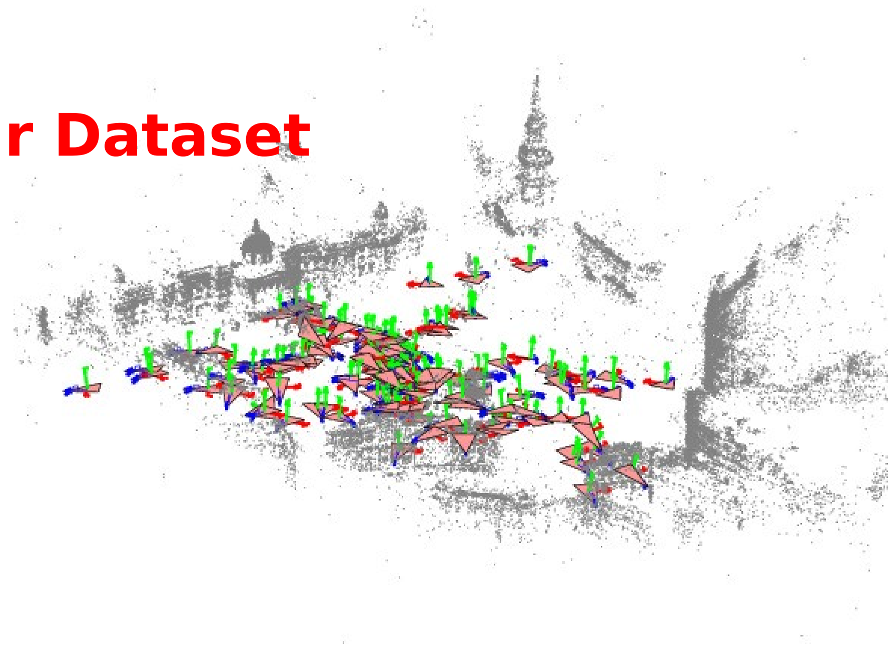
Bundle Adjustment in the Large
Sameer Agarwal, Noah Snavely, Steven M. Seitz and Richard Szeliski, European Conference on Computer Vision, 2010
http://grail.cs.washington.edu/projects/bal/

Experiments

Trafalgar Dataset



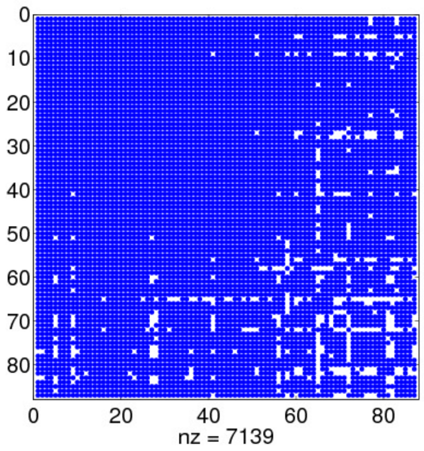




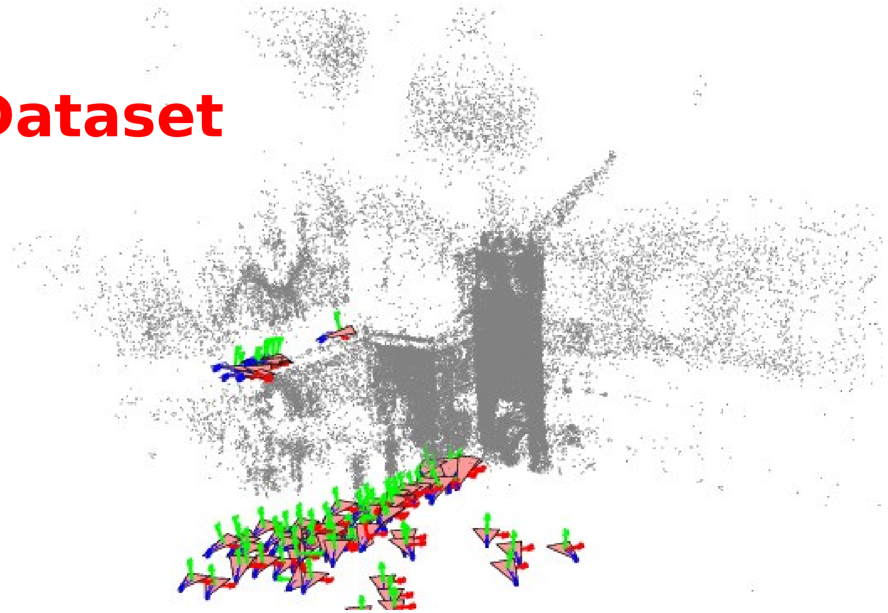
Experiment	nC	nP	nz	T improv %
TRAFALGAR	256	65127	225688	24,37

Bundle Adjustment in the Large
Sameer Agarwal, Noah Snavely, Steven M. Seitz and Richard Szeliski, European Conference on Computer Vision, 2010
http://grail.cs.washington.edu/projects/bal/

Experiments



Venice Dataset



Experiment	nC	nP	nz	T improv %
VENICE	87	110844	554826	9,86

Bundle Adjustment in the Large
Sameer Agarwal, Noah Snavely, Steven M. Seitz and Richard Szeliski, European Conference on Computer Vision, 2010
http://grail.cs.washington.edu/projects/bal/

Real experiments

Experiment	nC	nP	nz	T improv %
LADYBUG	783	83581	372940	57,45
PUBLIC SQUARE	519	32068	286162	46,52
TRAFALGAR	256	65127	225688	24,37
VENICE	87	110844	554826	9,86

?

Why is the DBA implementation faster ?

- In fact is slower when solving for the reduced camera matrix (Sparse CHOLMOD is hard to beat)
- The costly operation in our standard BA implementation is:

$$U^{m} = U - W V^{-1} W^{T}$$

Conclusions and Future Work

- An easy to implement distributed algorithm for visual mapping applications (Schur-Back Subs.)

- Good Time performance (without considering communication). It is able to beat a simple standard BA implementation.

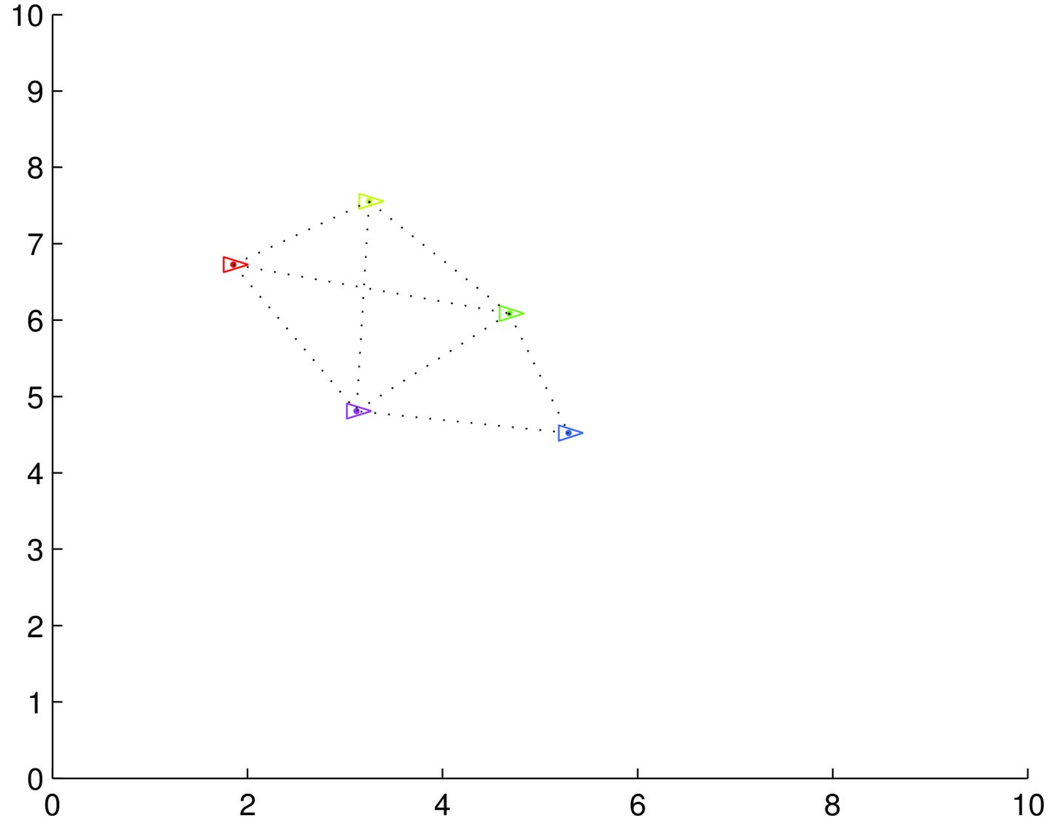
- Solution at each iteration is identical to a centralized system

- Develop methods to build balanced Junction Trees according to the available number of platforms

Outline

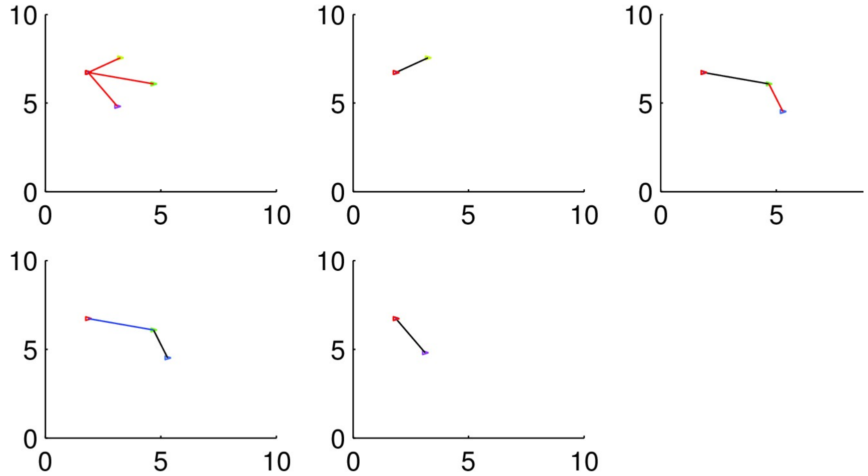
1. The SLAM scaling problem: Complexity and Consistency
2. D&C SLAM: Independent local maps
3. CI-Graph SLAM: Conditionally independent maps
4. DBA: Decomposable Bundle Adjustment
5. Multirobot SLAM

DBA + Pose Graph + Spanning Tree

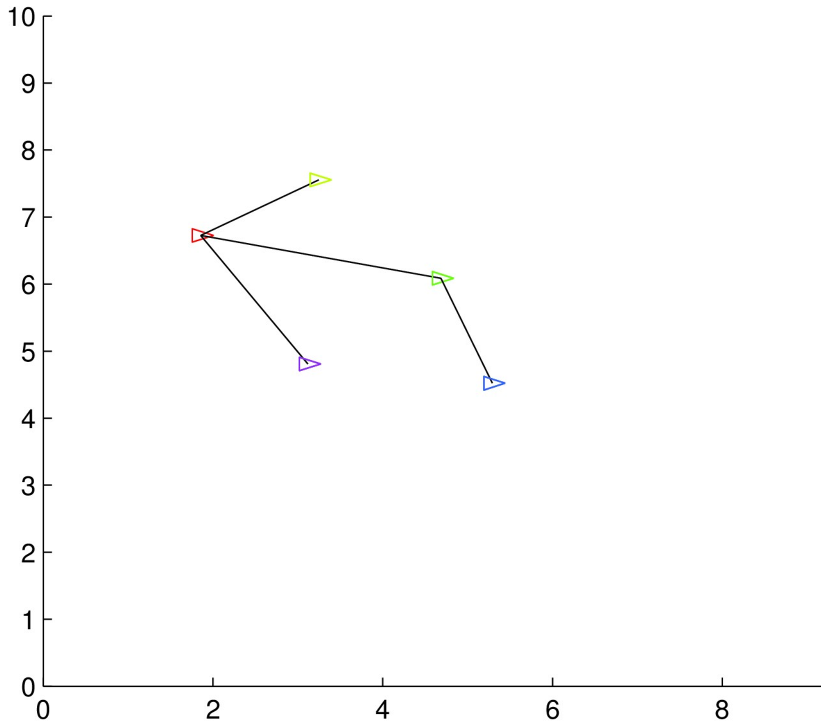


Communication Graph among 5 robots

DBA + Pose Graph + Spanning Tree

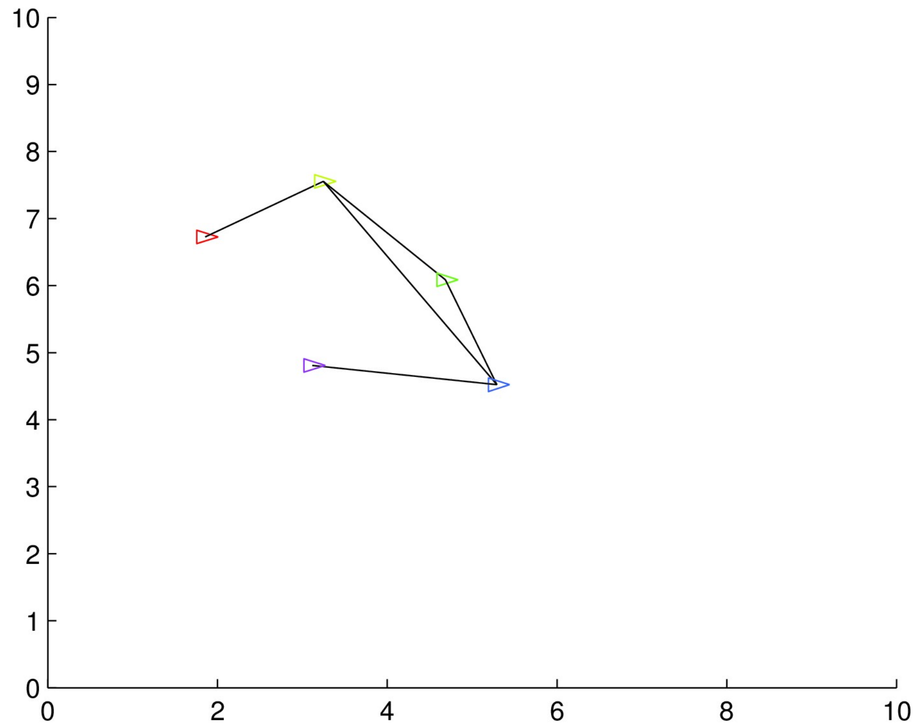


Distributed Spanning Tree



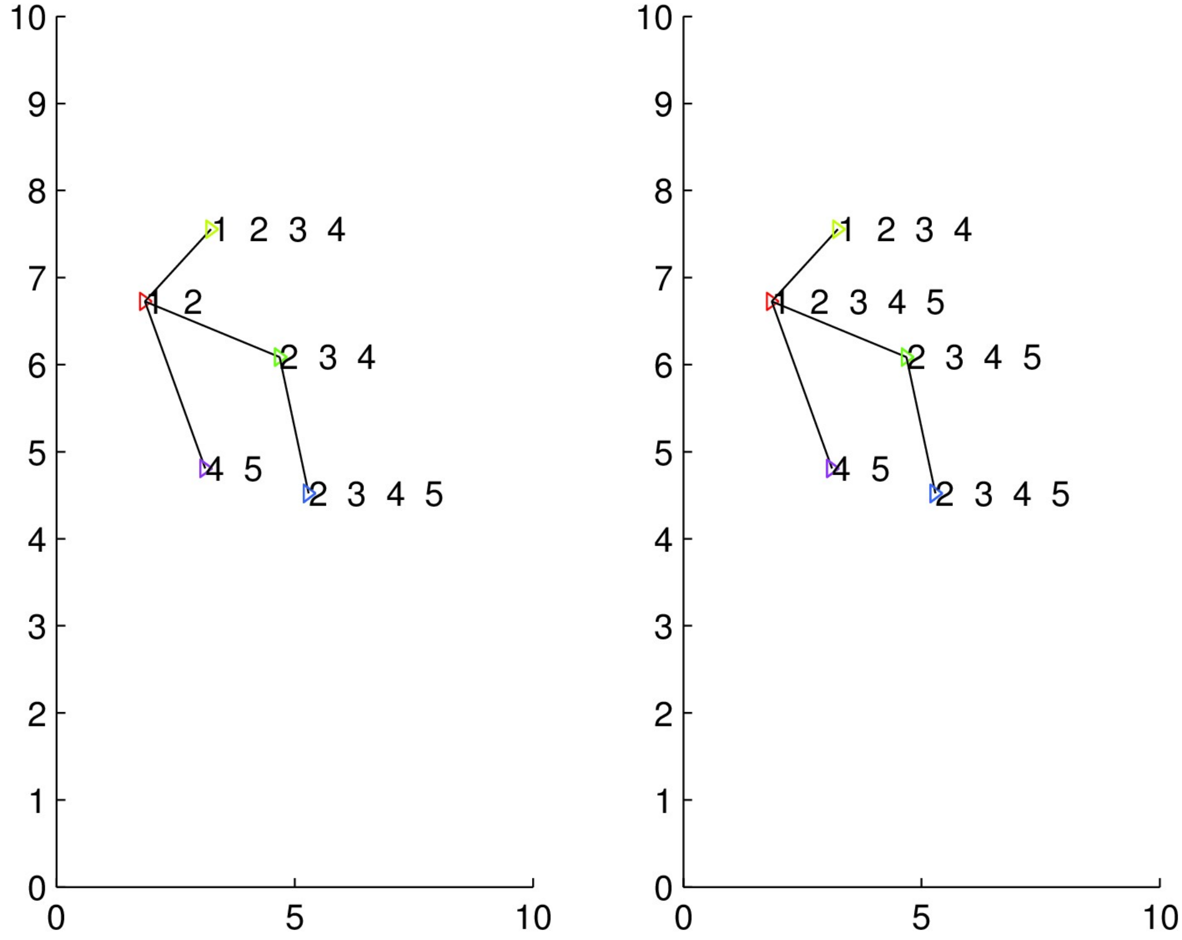
Centralized Spanning Tree

DBA + Pose Graph + Spanning Tree



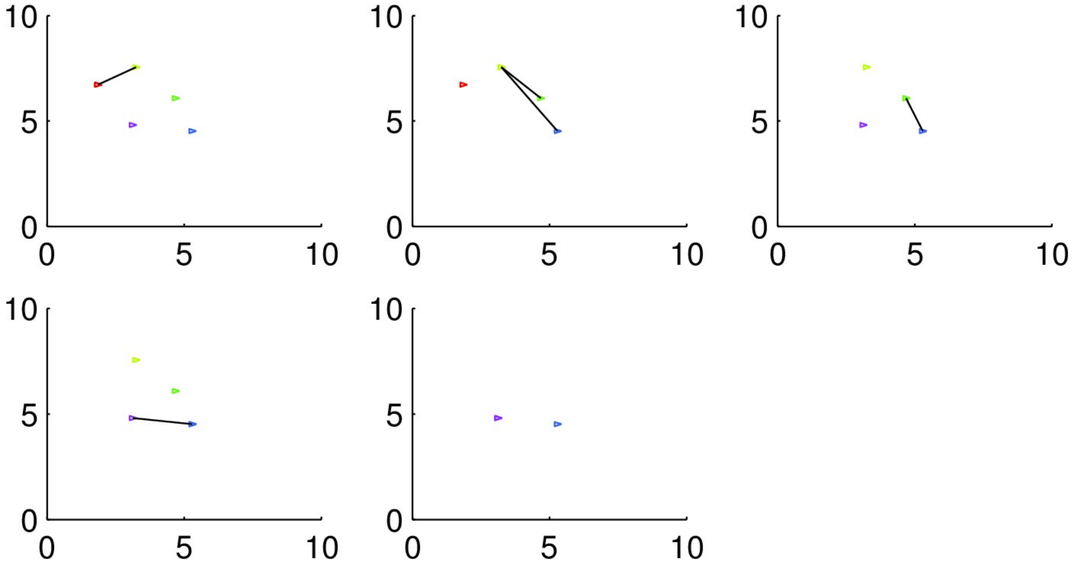
Pose Graph (Should not be the same to the connection Graph)

DBA + Pose Graph + Spanning Tree



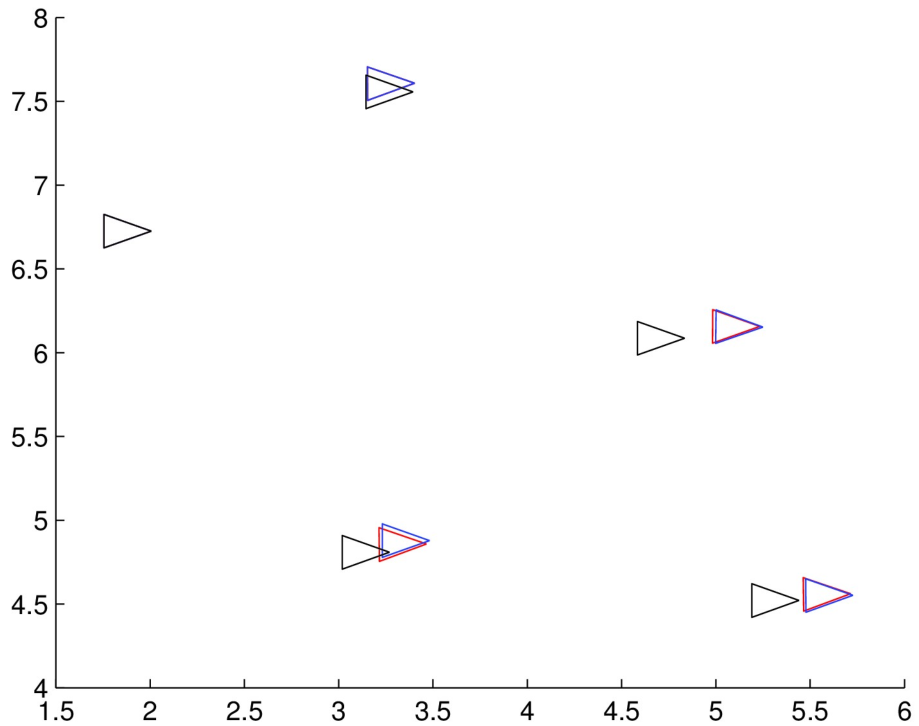
Junction Tree

DBA + Pose Graph + Spanning Tree



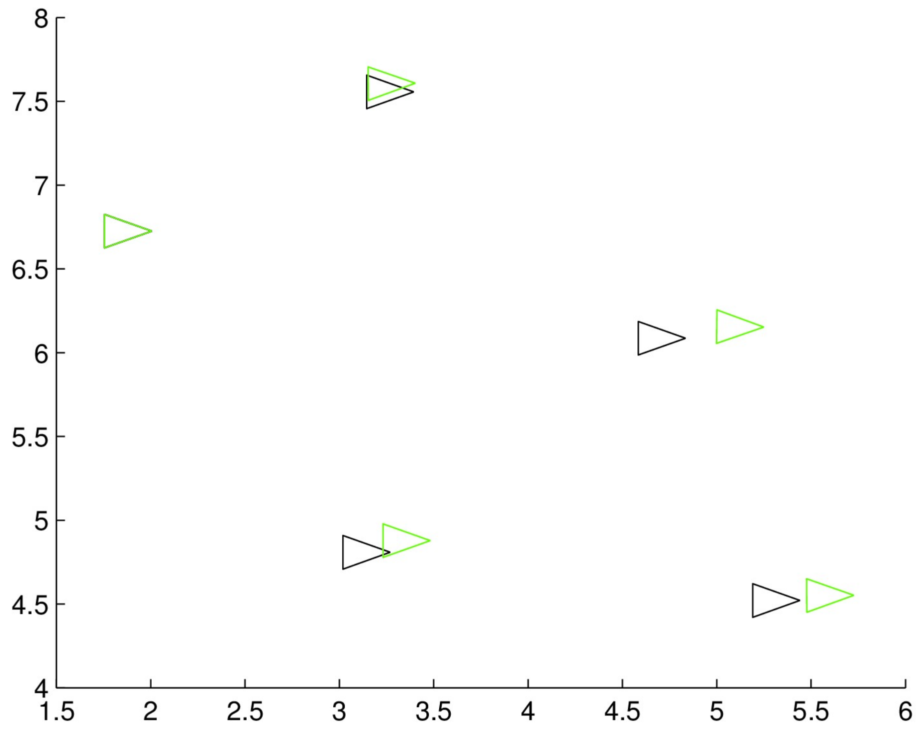
SubSystems

DBA + Pose Graph + Spanning Tree



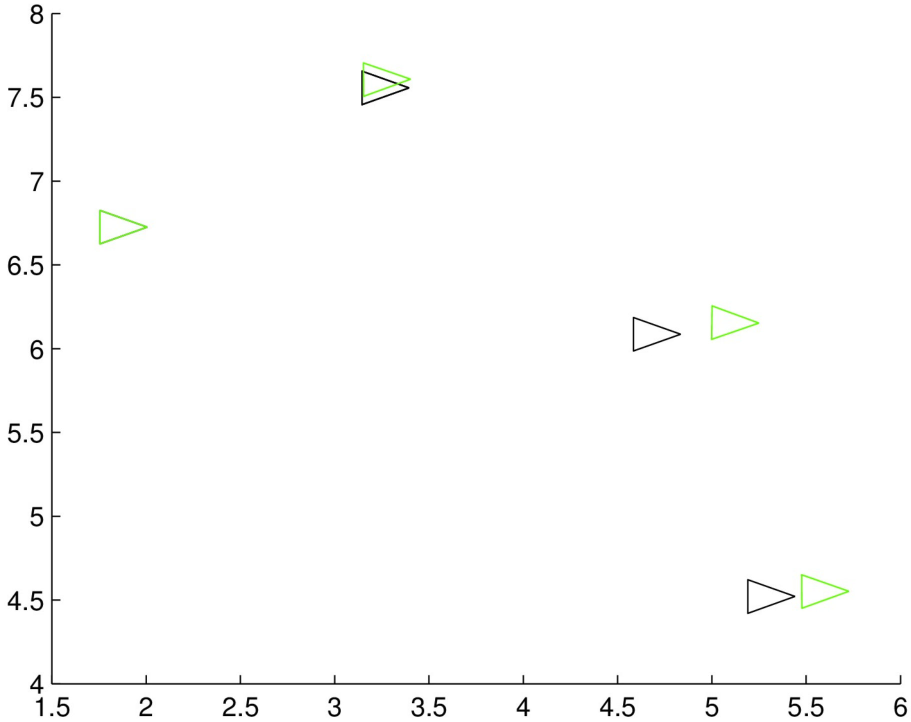
Global Solution

DBA + Pose Graph + Spanning Tree



Solution of SubSystem 1

DBA + Pose Graph + Spanning Tree



Solution of SubSystem 3

My Current interests

- Multi-robot SLAM research
- Dense 3D Reconstructions potentially for Real Time
- Object Categorization / Recognition
- Compression of object models

Thank you