A Balanced Distributed Graph-based Framework for Multi-Robot Mapping

Dario Lodi Rizzini, Stefano Caselli

Dipartimento di Ingegneria dell'Informazione
Università degli Studi di Parma

Graphbot 2010, October 22 2010
IROS Workshop on Probabilistic Graphical Models in Robotics

Summary

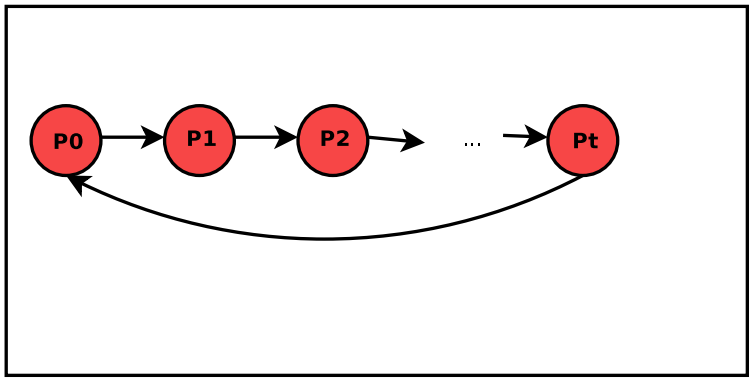
Motivation

- Scenario: robot team exploring the same environment
  - separated maps/single distributed map
  - issues: centralized vs shared map, inter-robot data association, etc.
- Graphical formulation of mapping problem
  - exact sparseness of information matrix
  - suitable for submap decomposition
  - management of computational load

Related Works

- Constrained Local Submap Filter (CLSF) [Williams *et al*, 2002]
- Sparse Extended Kalman Filter (SEIF) [Thrun *et al*, 2003]
- Rao-Blackwellized Particle Filters [Howard, 2006]
- Graph-based Multi-robot SLAM
  - Multifrontal QR [Dellaert *et al*, 2005]
  - Conjugate Gradient [Nerurkar *et al*, 2009]
  - DDF-SAM [Cunninghan *et al*, 2010]

Graphical Formulation of SLAM



- Pose-Graph
  - nodes: poses $p_{0:t}$
  - edges: constraints $\langle i, j \rangle$, observation of p_j from p_i
- Constraints described by $(\delta_{ij}, \Omega_{ij})$

- Solution is the configuration $\mathbf{p}^*$ that minimizes error (maximizes likelihood)

$$\mathbf{p}^* = \underset{\mathbf{p}}{\operatorname{argmin}} \sum_{\langle j,i \rangle \in \mathcal{C}} e_{ij}^T(\mathbf{p})\, \Omega_{ij}\, e_{ij}(\mathbf{p})$$

$$e_{ij}(\mathbf{p}) = f_{ij}(\mathbf{p}) - \delta_{ij}$$

$f_{ij}(\mathbf{p})$: frame reference change

- *Maximum Likelihood* (ML) reformulates mapping as a nonlinear optimization problem

Gauss-Seidel Relaxation

- Several methods to find minimum $\mathbf{p}^*$ are *fixed-point iterative methods*
  - Linearization of each constraint $\langle i, j \rangle \in \mathcal{C}$ given an initial estimation $\hat{\mathbf{p}}^k$:

$$e_{ij}(\mathbf{x}) \approx f_{ij}(\hat{\mathbf{p}}) - \delta_{ij} + J_{ij}^i(x_i - \hat{p}_i) + J_{ij}^j(x_j - \hat{p}_j)$$

  - Refined estimation $\hat{\mathbf{p}}^{k+1}$ by solving the linearized problem
- *Gauss-Seidel Relaxation* relaxes constraints and solves each pose p_i independently (*no back-substitution*)

$$p_i^{(k+1)} = A_{ii}^{-1} \left(b_i - \sum_{j<i} A_{ij} p_j^{(k+1)} - \sum_{j>i} A_{ij} p_j^{(k)} \right)$$

- Other numeric methods: gradient descent, matrix factorization, sparsification, etc.

Multi-Robot ML-Mapping

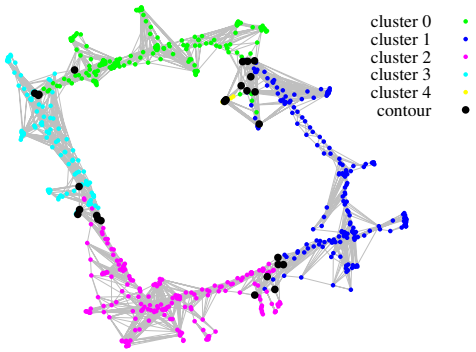
Formulation

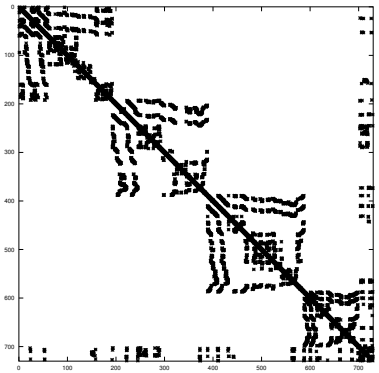
- Several robots (*mapping units*) explore the environment building a single map
- Each robot stores a submap which is a subset of the whole map

Issues

- Global constraint network consists of poses on different mapping units
- Communication and synchronization of common pose values shared by mapping units
- Balanced size of submaps

Parallel Gauss-Seidel Relaxation

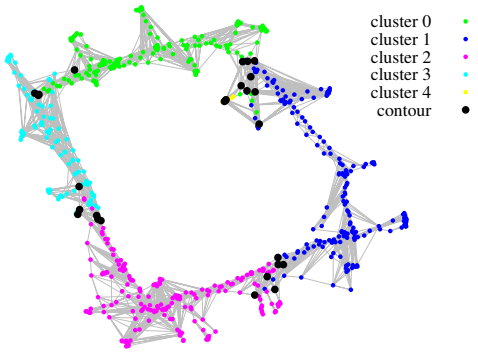


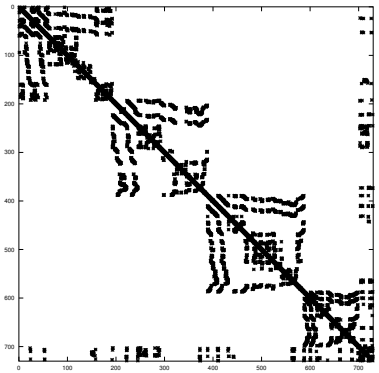


Lodi Rizzini & Caselli, IROS 2009

- Gauss-Seidel relaxation to solve linearized system $A\,\mathbf{p} = b$
- Sorted matrix A in *block-bordered-diagonal form* (BBD) achieved through node reordering
  - nodes of a cluster have consecutive indices
  - contour nodes at the end
- Each block can be solved *independently*; countour nodes are solved in the end
- Parallel relaxation can be adapted to multi-robot mapping

Parallel Gauss-Seidel Relaxation





Lodi Rizzini & Caselli, IROS 2009

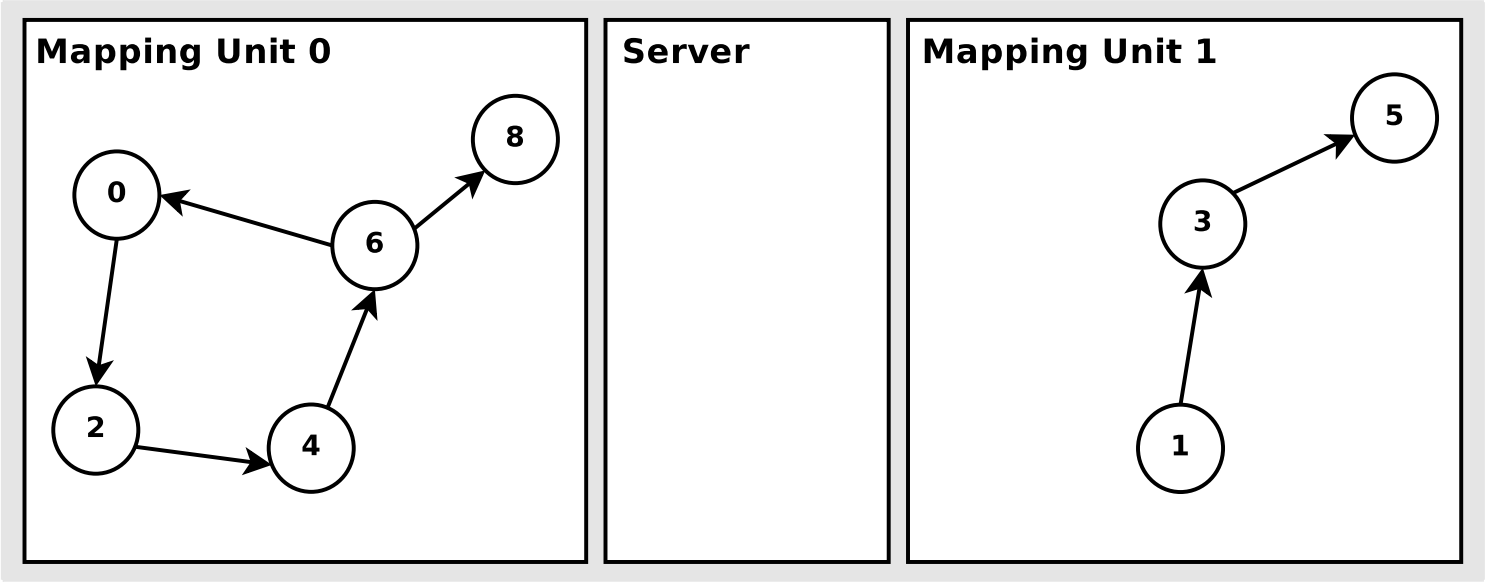
- Gauss-Seidel relaxation to solve linearized system $A\,\mathbf{p} = b$
- Sorted matrix A in *block-bordered-diagonal form* (BBD) achieved through node reordering
  - nodes of a cluster have consecutive indices
  - contour nodes at the end
- Each block can be solved *independently*; countour nodes are solved in the end

- Parallel relaxation can be adapted to multi-robot mapping

Issues for Distributed Multi-Robot Relaxation

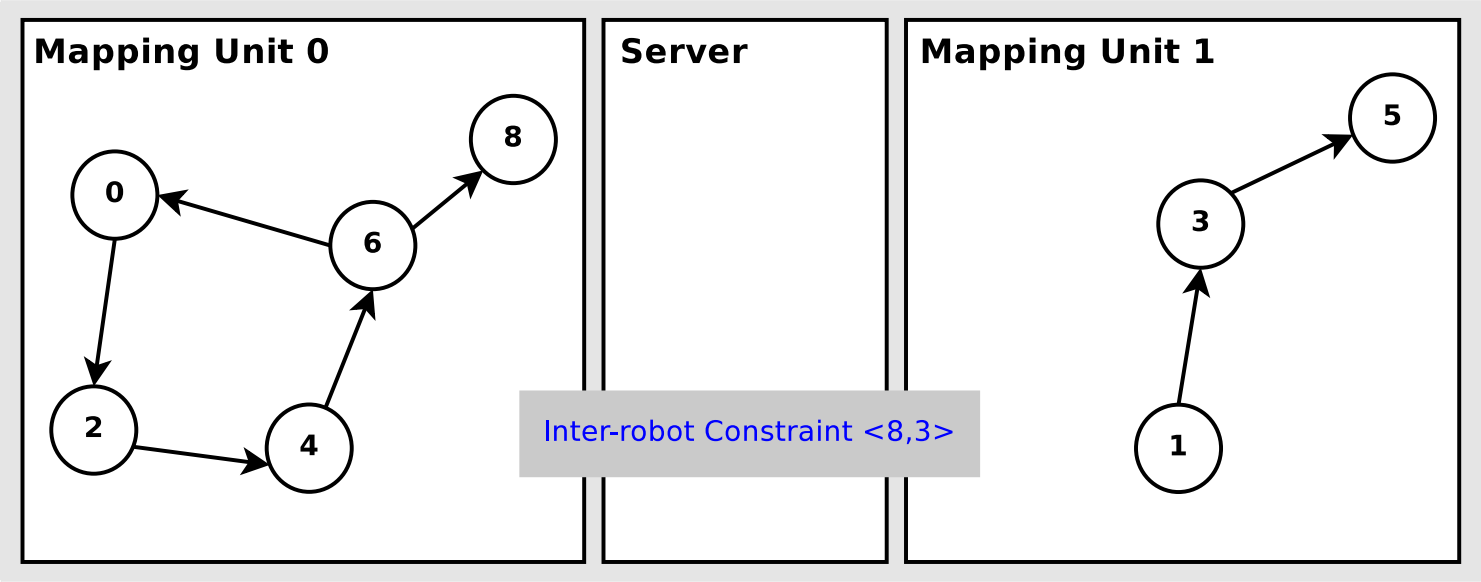
- Shared representation among distributed robots
  - shared pose update using data stored on different units
- Contour nodes representing intersection between different submap
  - stored on a server
  - updated with the contribution sent from mapping units
- Balanced submap sizes
  - each submap is built by a robot and not obtained through node reordering
  - the resulting submap sizes and computational load between mapping units may not be balanced

Inter-robot Associations



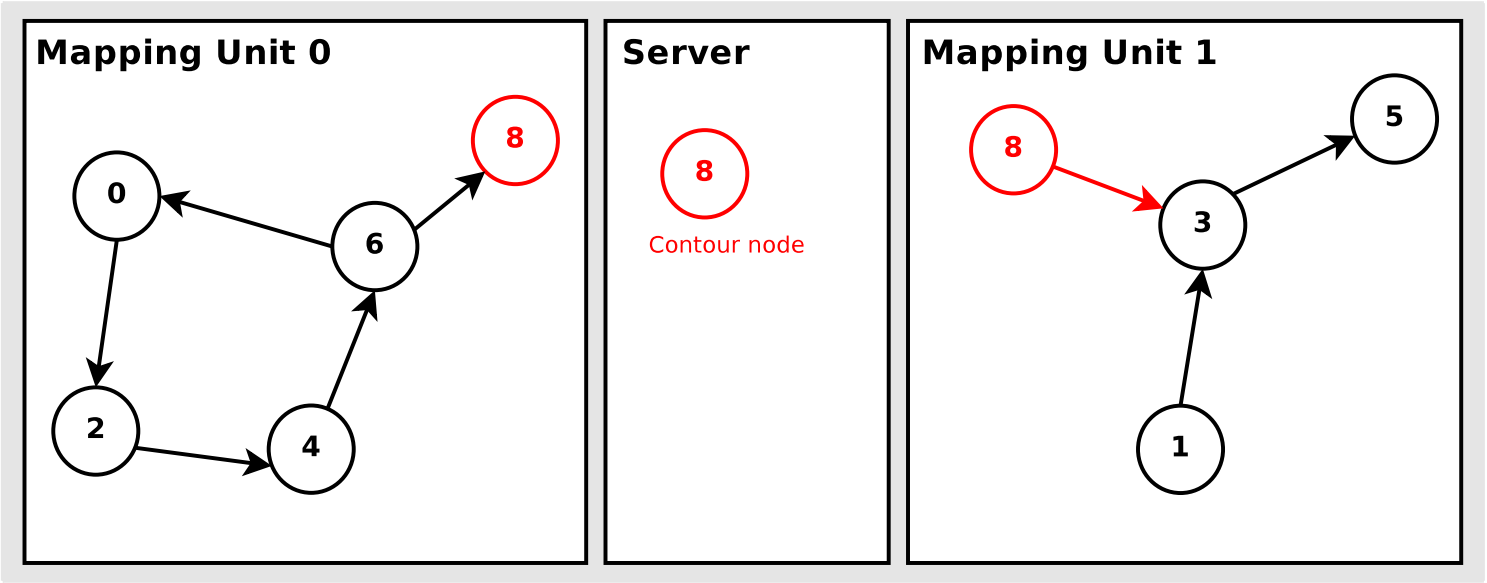
- Each mapping unit builds a local submap in the form of a pose graph
- When an inter-robot constraint $\langle i, j \rangle$ is observed from pose i, pose i is marked as *contour node*
- An *alias* of pose i is added to the submap containing j and to the server

Inter-robot Associations



- Each mapping unit builds a local submap in the form of a pose graph
- When an inter-robot constraint $\langle i, j \rangle$ is observed from pose i, pose i is marked as *contour node*
- An *alias* of pose i is added to the submap containing j and to the server

Inter-robot Associations



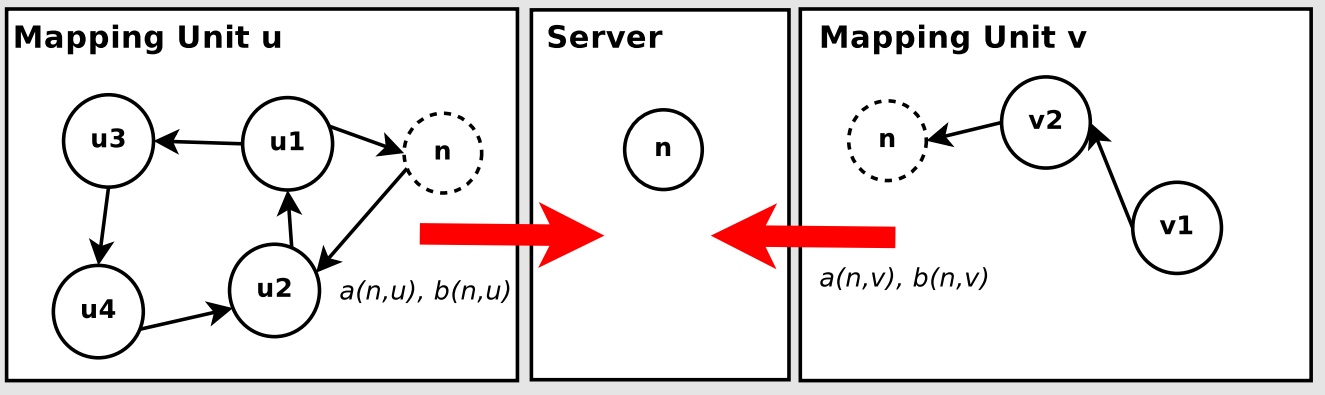
- Each mapping unit builds a local submap in the form of a pose graph
- When an inter-robot constraint $\langle i, j \rangle$ is observed from pose i, pose i is marked as *contour node*
- An *alias* of pose i is added to the submap containing j and to the server

Asynchronous Gauss-Seidel Relaxation

- Relaxation is independently performed on each submap for non-contour poses
- For each contour node alias ${}^{u}p_n$ computes residuals ${}^{u}a_n$ and ${}^{u}b_n$ for mapping unit u

$$
\begin{aligned}
{}^{u}a_n &= \sum_{\langle i,n\rangle \in \hat{\mathcal{C}}_u} {J_{in}^{n}}^{T}\, \Omega_{in}\, J_{in}^{n} + \sum_{\langle n,j\rangle \in \hat{\mathcal{C}}} {J_{nj}^{n}}^{T}\, \Omega_{nj}\, J_{nj}^{n} \\
{}^{u}b_n &= \sum_{\langle i,n\rangle \in \hat{\mathcal{C}}_u} {J_{in}^{n}}^{T}\, \Omega_{in}\, (r_{in} - J_{in}^{i}\, \Delta p_i) + \sum_{\langle n,j\rangle \in \hat{\mathcal{C}}} {J_{nj}^{n}}^{T}\, \Omega_{nj}\, (r_{nj} - J_{nj}^{j}\, \Delta p_j)
\end{aligned}
$$

- Server sums the contributions ${}^{u}a_n$ and ${}^{u}b_n$ to compute the new value of pose



Balancing Submap Size

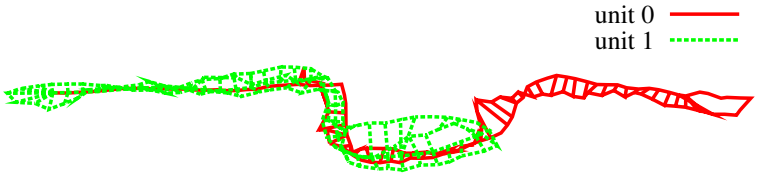
- Each mapping unit periodically sends the size of the stored submap to the server and receives the average submap size
- When current submap size exceeds the average size, the excess nodes are converted to contour nodes and transferred to the server
- *Layering algorithm* used to select the candidate nodes for transfer
  - breadth-first expansion from contour nodes

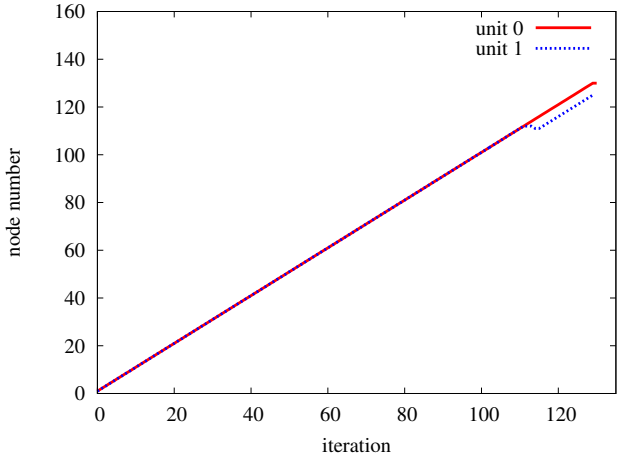
Results

- Real environment data set
  - hallway of the Dep. of Information Engineering (about 28×8 m)
  - data acquired and segmented to emulate multi-robot setup
- Simulated environment
  - office-like environment
  - exploration with 4 robots
  - odometry uncertanty with normal noise $\sigma_{pos} = 0.02$ m, $\sigma_{ang} = 2^\circ / m$

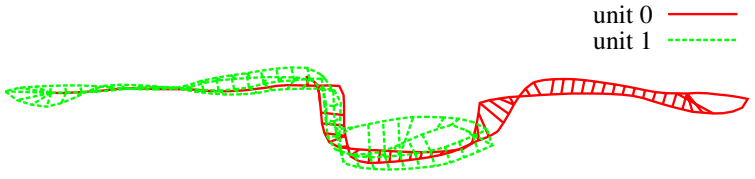
Experiment

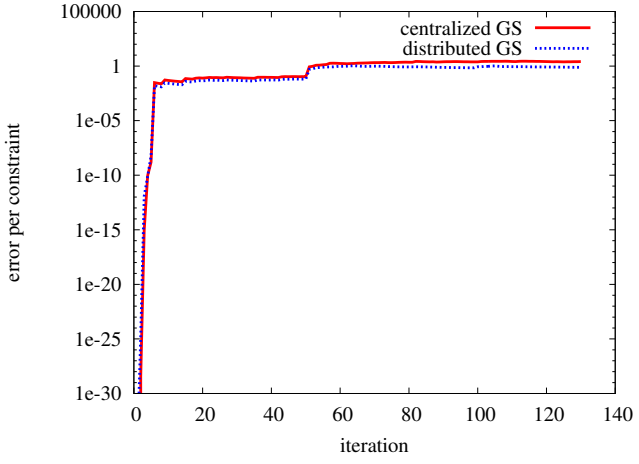
Odometry





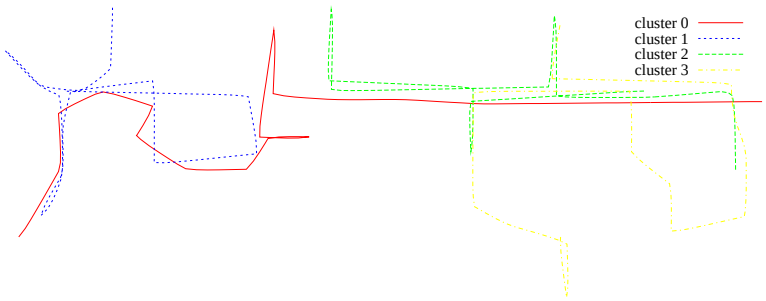
Solved



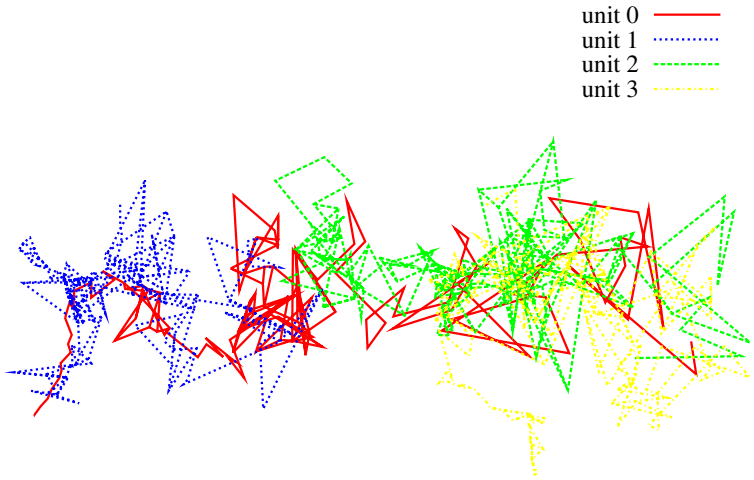


Simulation (1)

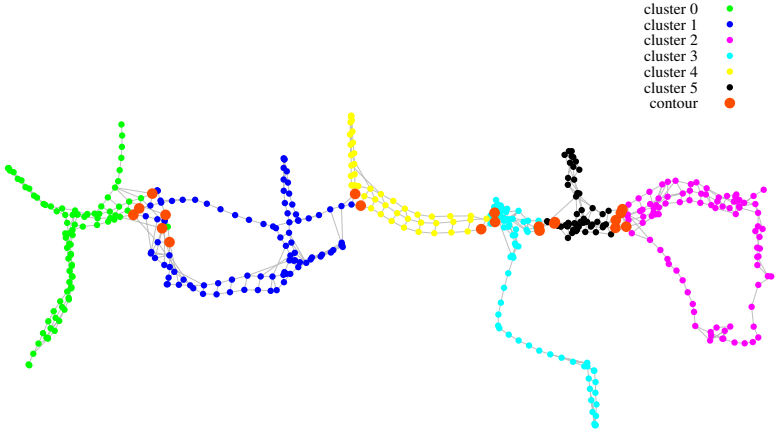
Groundtruth



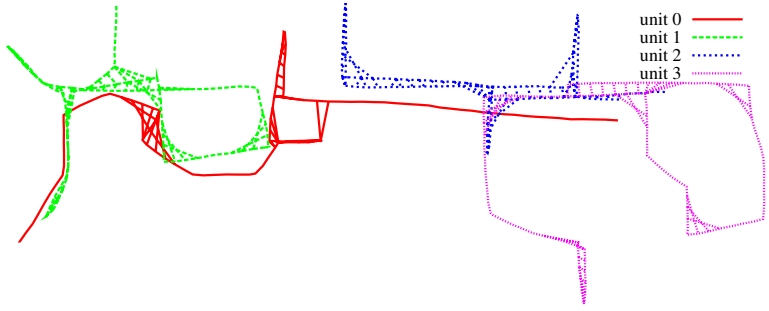
Odometry



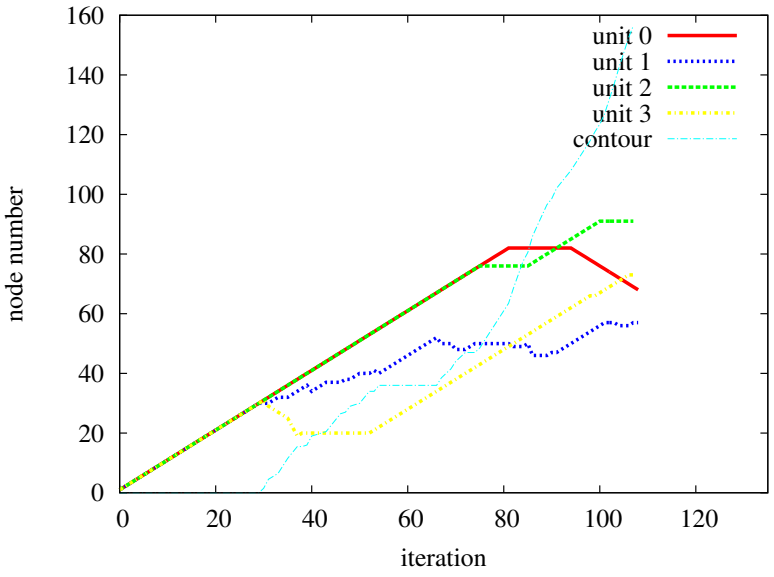
Direct Cluster ($n_{max} = 100, p = 0.6$)

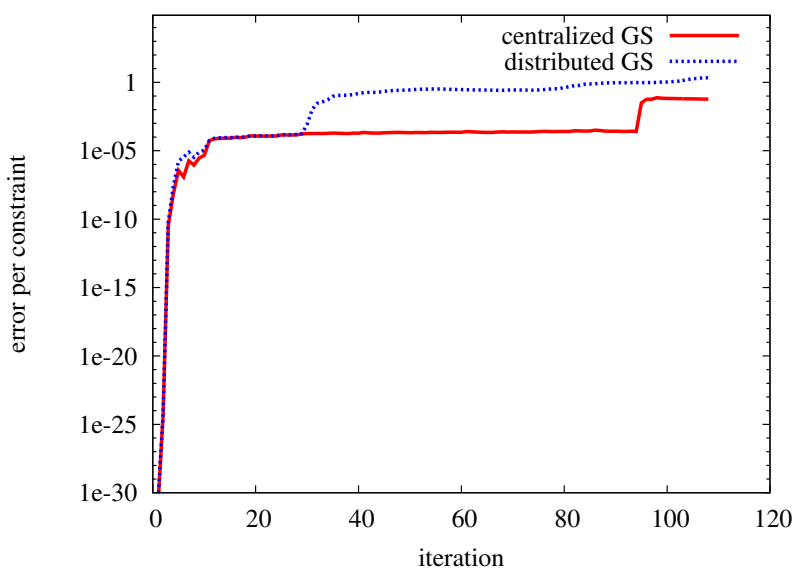


Solved



Simulation (2)

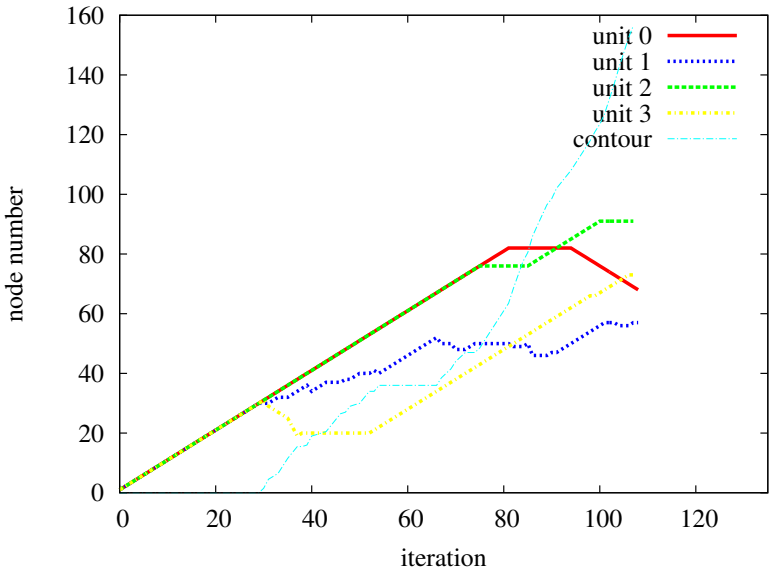


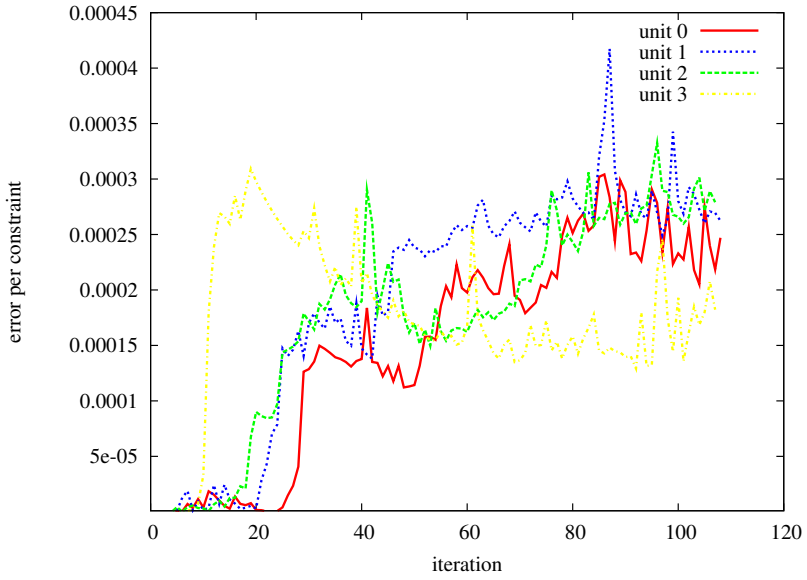


- Submap sizes[1] are quite balanced while contour node number increases
- Comparable error per constraint centralized/distributed Gauss-Seidel relaxation
  - large error increase when inter-robot association occurs
- Average error per constraint almost stable inside submaps

[1] Submap sizes include aliases of contour nodes

Simulation (2)





- Submap sizes[1] are quite balanced while contour node number increases
- Comparable error per constraint centralized/distributed Gauss-Seidel relaxation
  - large error increase when inter-robot association occurs
- Average error per constraint almost stable inside submaps

[1] Submap sizes include aliases of contour nodes

Conclusion

- Multi-robot Gauss-Seidel relaxation algorithm for ML Mapping
  - Distributed contour nodes to manage inter-robot constraints (with known inter-robot associations)
  - Asynchronous Gauss-Seidel iterations executed on different mapping units
  - Balanced size of submaps
  - Implementation suitable for incremental data association
- Experiments assess the correctness of the proposed approach

Future Works

- Apply a more efficient algorithm than Gauss-Seidel relaxation
  - Requirement: limited dependencies and communication (e.g. due to back-substitution)
- Improvement of balancing policy to bound contour node size
- Detection of inter-robot associations from observation