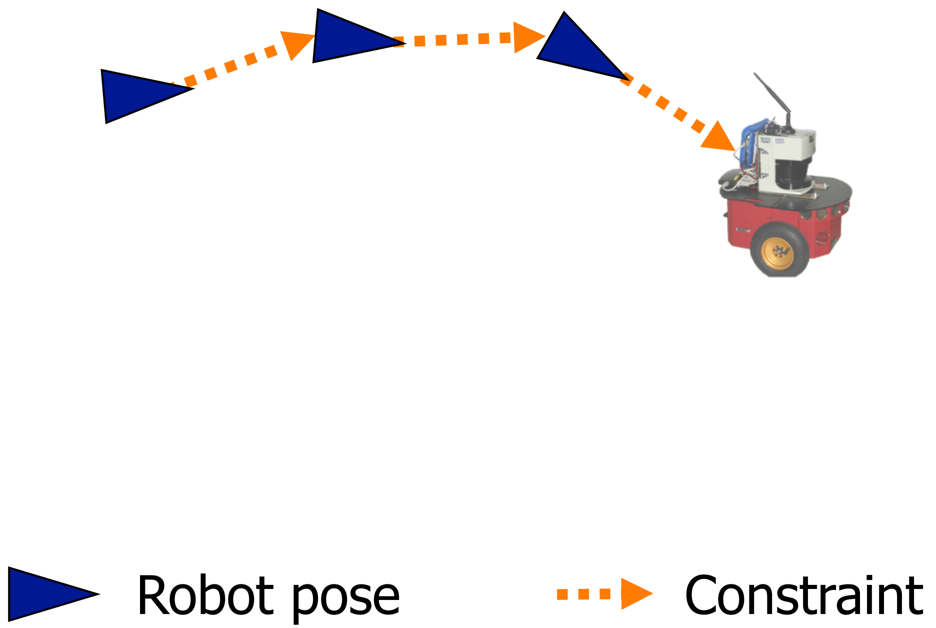
Robot Mapping

Hierarchical Pose-Graphs for Online Mapping

Cyrill Stachniss

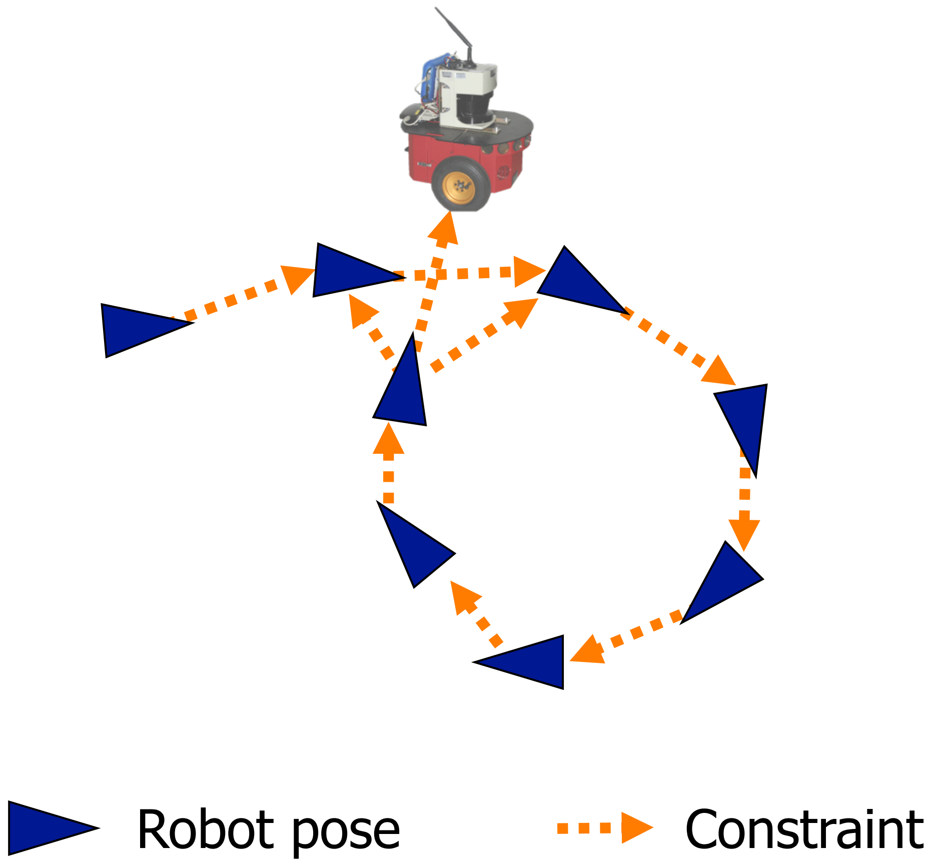
Graph-Based SLAM (Chap. 15)

- Constraints connect the poses of the robot while it is moving
- Constraints are inherently uncertain



Graph-Based SLAM (Chap. 15)

- Observing previously seen areas generates constraints between non-successive poses

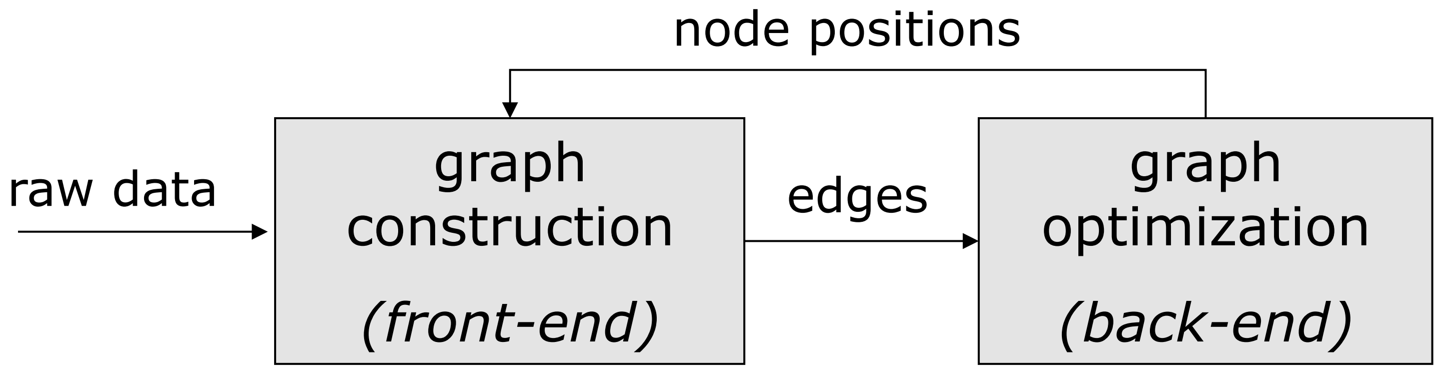


Graph-Based SLAM (Chap. 15)

- Use a **graph** to represent the problem
- Every **node** in the graph corresponds to a pose of the robot during mapping
- Every **edge** between two nodes corresponds to a spatial constraint between them
- **Graph-Based SLAM:** Build the graph and find a node configuration that minimize the error introduced by the constraints

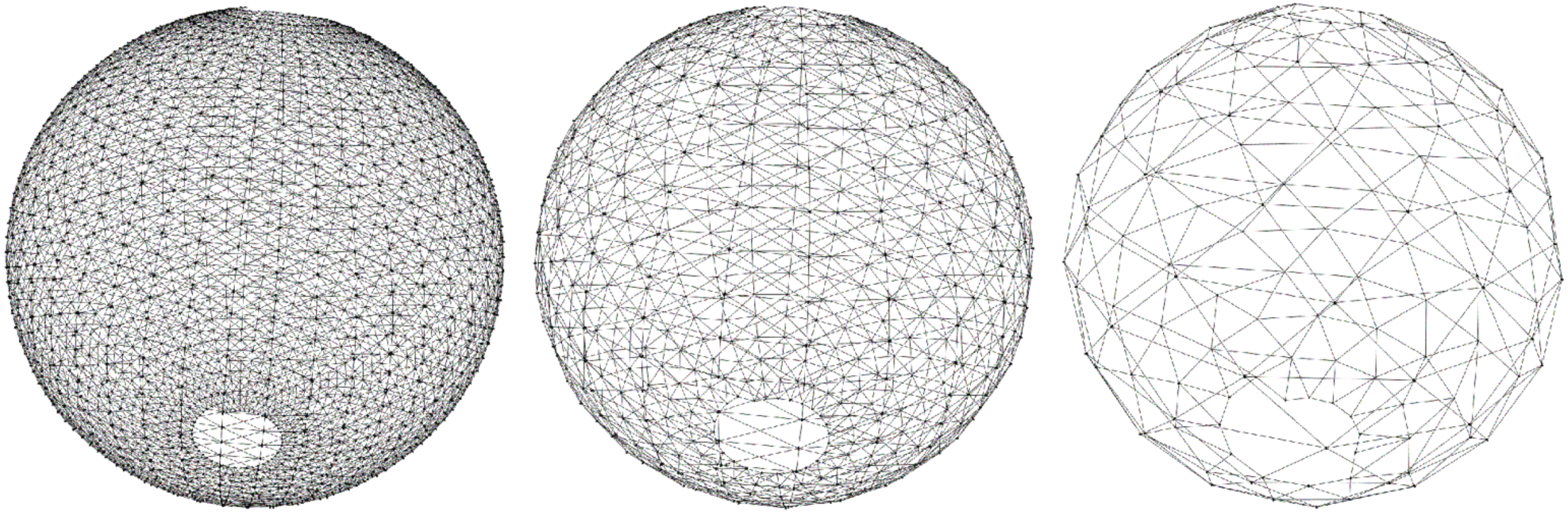
Front-End and Back-End

- Front-end extracts constraints from the sensor data (data association!)
- Back-end optimizes the pose-graph to reduce the error introduced by the constraints



➡ Intermediate solutions are needed to make good data associations

Hierarchical Pose-Graph

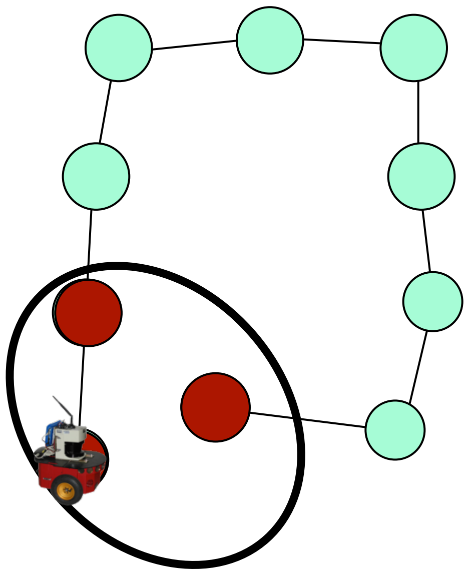


bottom layer (input data) | first layer | second layer | top layer

“There is no need to optimize the whole graph when a new observation is obtained”

Motivation

- SLAM front-end seeks for loop-closures
- Requires to compare observations to all previously obtained ones
- In practice, limit search to areas in which the robot is likely to be
- This requires to know **in which parts of the graph to search for data associations**

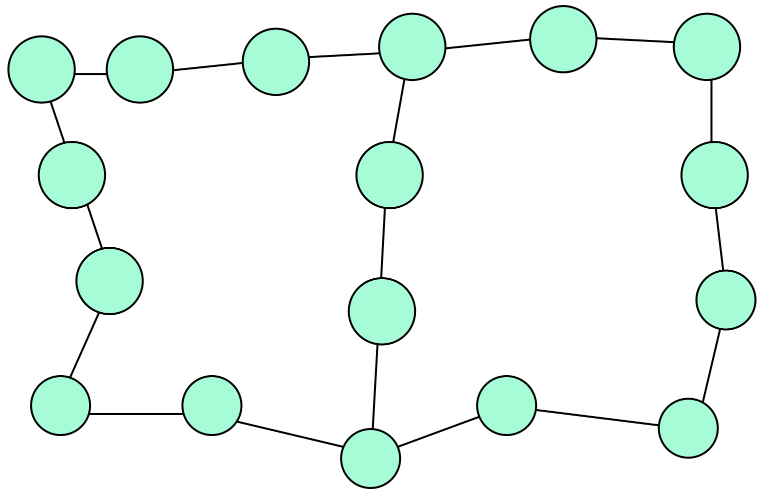


Hierarchical Approach

- **Insight:** to find loop closing points, one does not need the perfect global map
- **Idea:** correct only the core structure of the scene, not the overall graph
- The hierarchical pose-graph is a sparse approximation of the original problem
- It exploits the facts that in SLAM
  - Robot moved through the scene and it not "teleported" to locations
  - Sensors have a limited range

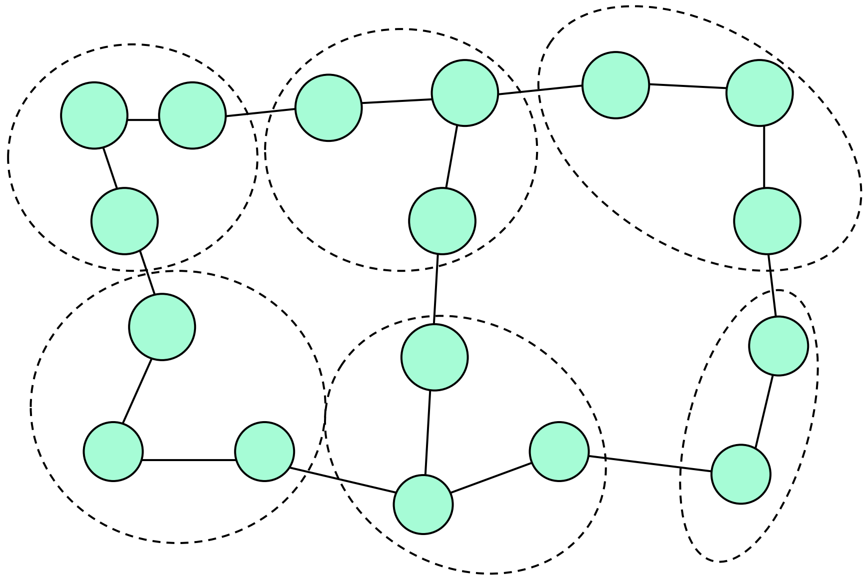
Key Idea of the Hierarchy

- Input is the dense graph



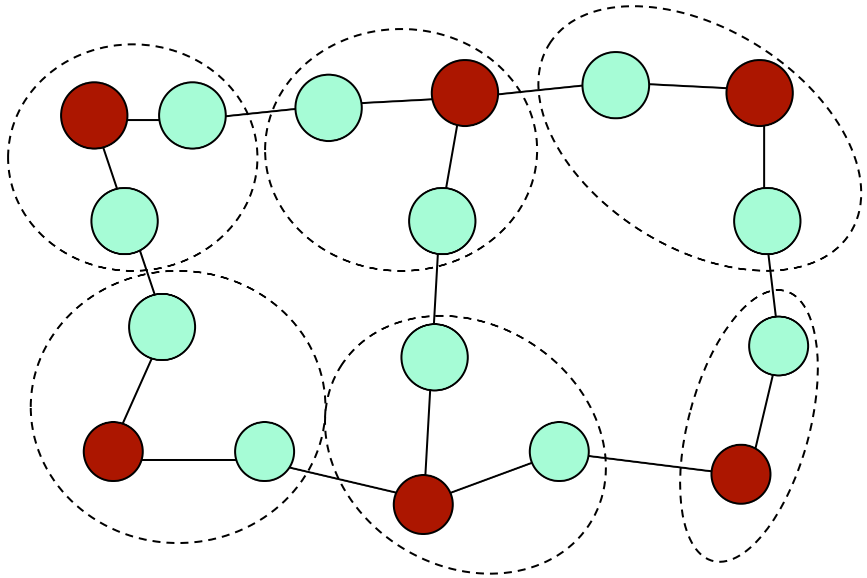
Key Idea of the Hierarchy

- Input is the dense graph
- Group the nodes of the graph based on their local connectivity



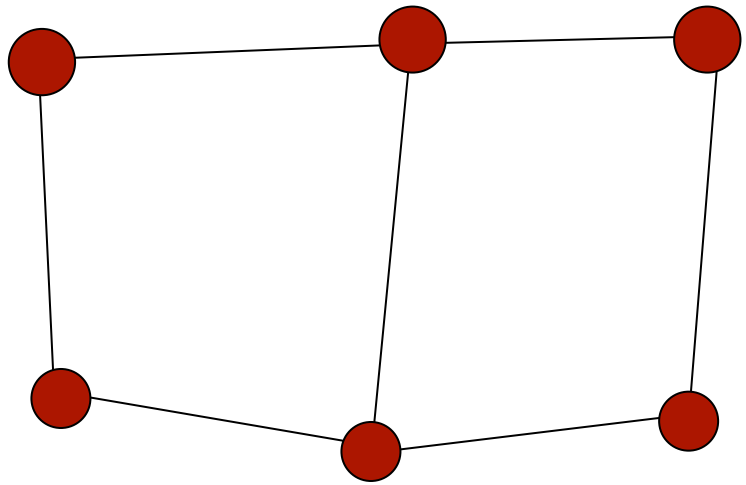
Key Idea of the Hierarchy

- Input is the dense graph
- Group the nodes of the graph based on their local connectivity
- For each group, select one node as a "representative"



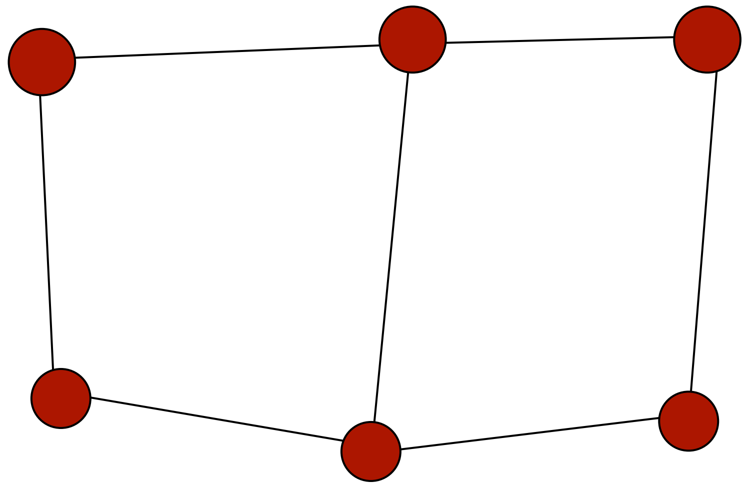
Key Idea of the Hierarchy

- The representatives are the nodes in a new sparsified graph (upper level)



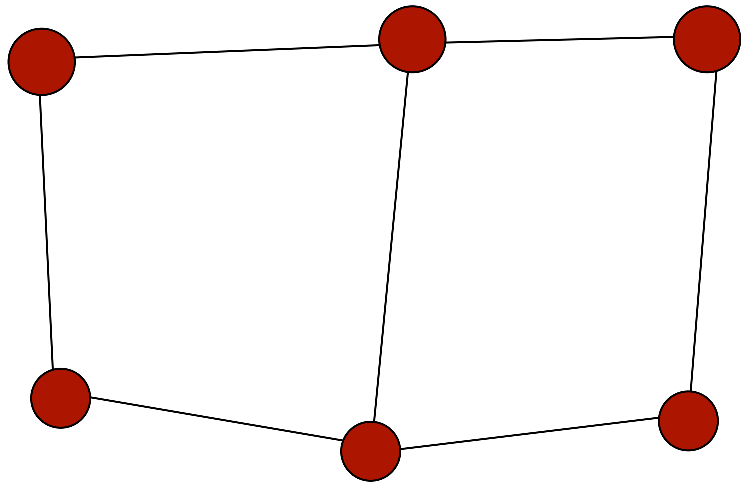
Key Idea of the Hierarchy

- The representatives are the nodes in a new sparsified graph (upper level)
- Edges of the sparse graph are determined by the connectivity of the groups of nodes
- The parameters of the sparse edges are estimated via local optimization



Key Idea of the Hierarchy

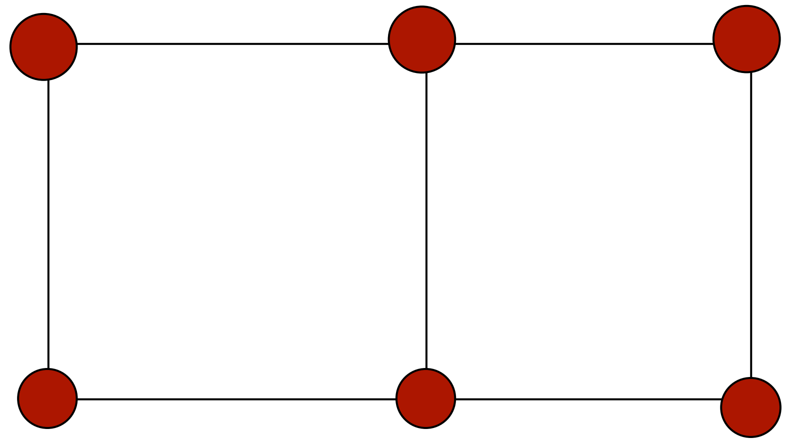
- The representatives are the nodes in a new sparsified graph (upper level)
- Edges of the sparse graph are determined by the connectivity of the groups of nodes
- The parameters of the sparse edges are estimated via local optimization



Process is repeated recursively

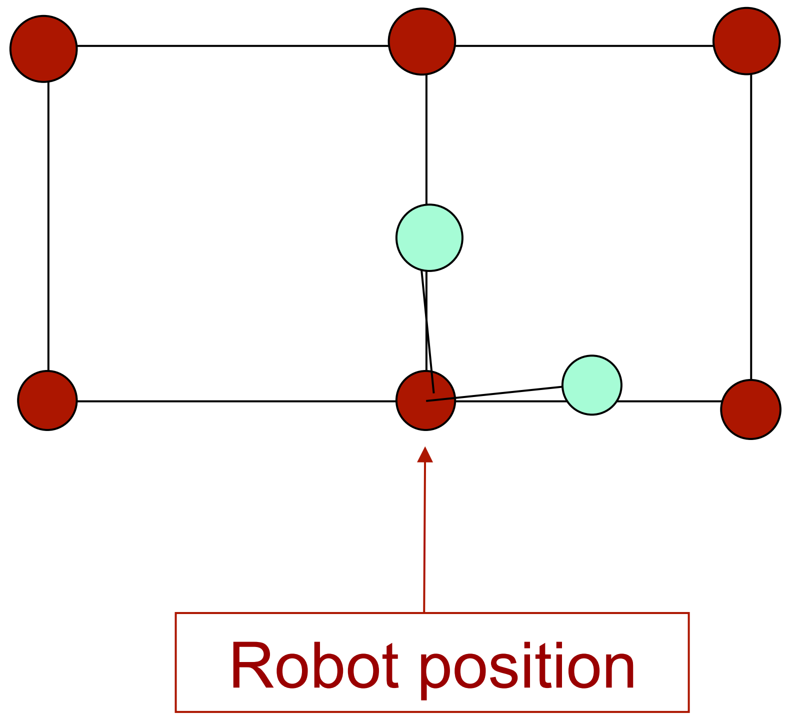
Key Idea of the Hierarchy

- Only the upper level of the hierarchy is optimized completely



Key Idea of the Hierarchy

- Only the upper level of the hierarchy is optimized completely
- The changes are propagated to the bottom levels only close to the current robot position
- Only this part of the graph is relevant for finding constraints

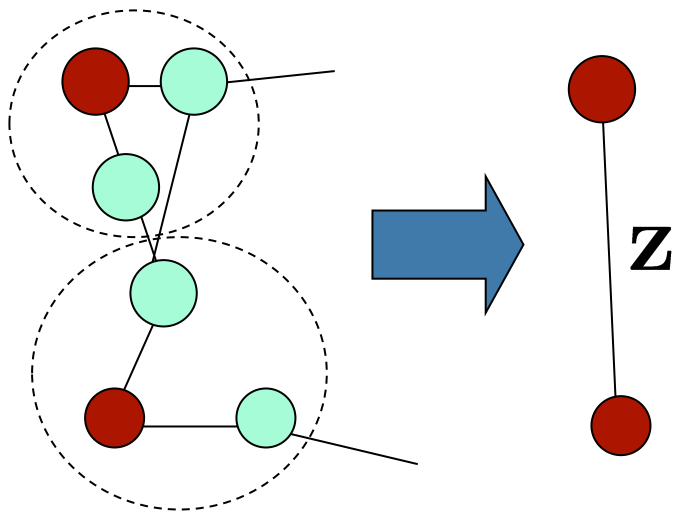


Construction of the Hierarchy

- When and how to generate a new group?
  - A simply, distance-based heuristic on the graph
  - The first node of a new group is the representative
- When to propagate information downwards?
  - Only when there are inconsistencies
- How to construct an edge in the sparsified graph?
  - Next slides
- How to propagate information downwards?
  - Next slides

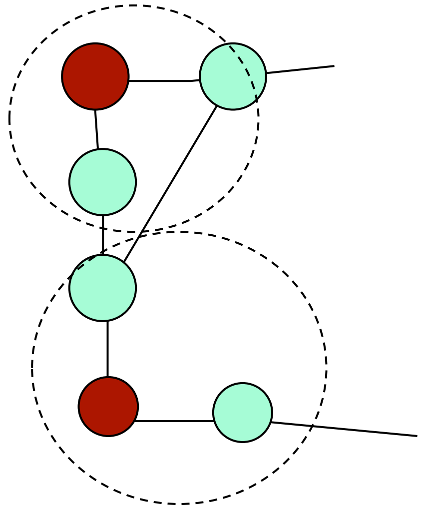
Determining Edge Parameters

- Given two connected groups
- How to compute a virtual observation $\mathbf{z}$ and the information matrix Ω for the new edge?



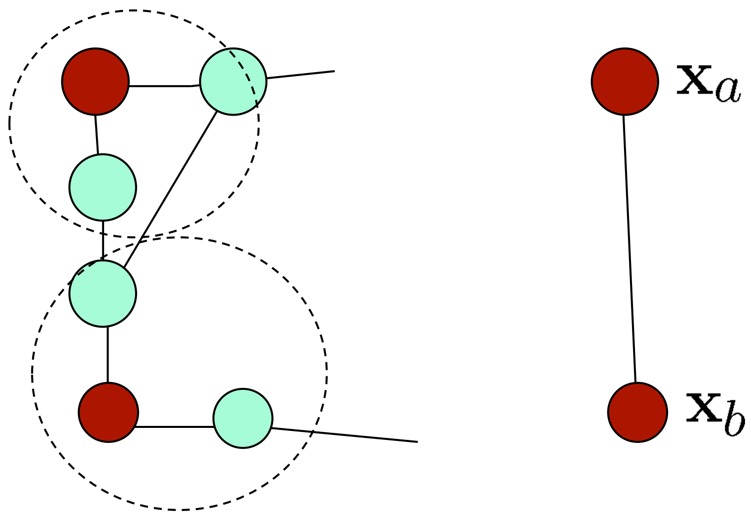
Determining Edge Parameters

- Optimize the two sub-groups independently from the rest



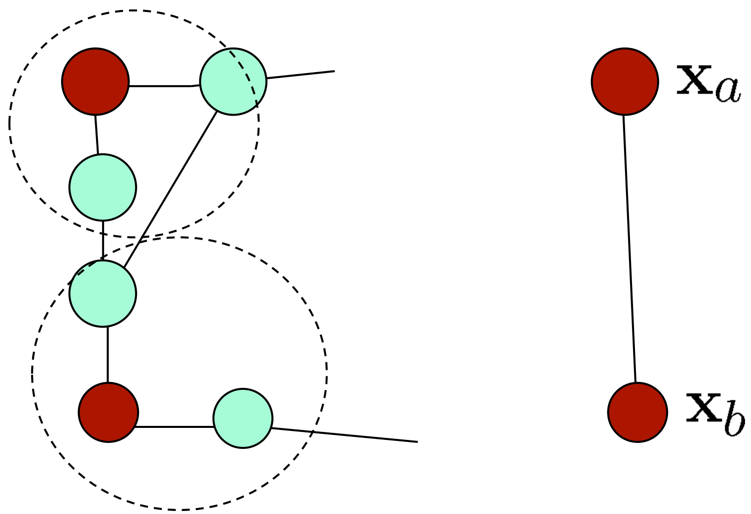
Determining Edge Parameters

- Optimize the two sub-groups independently from the rest
- The observation is the relative transformation between the two representatives



Determining Edge Parameters

- Optimize the two sub-groups independently from the rest
- The observation is the relative transformation between the two representatives
- The information matrix is computed from the diagonal block of the matrix ***H***

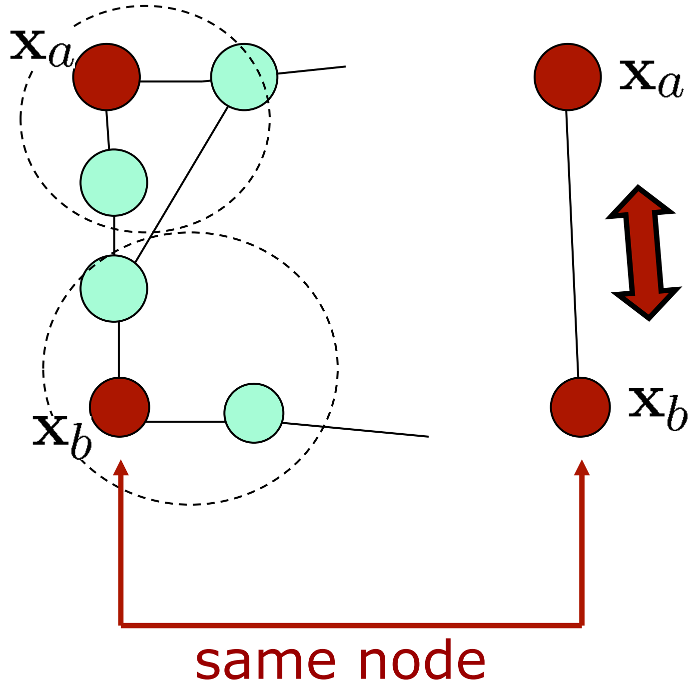


Inverse of the *[b,b]* block of ***H***$^{-1}$

$$\Omega_{ab} = (\mathbf{H}^{-1}_{[b,b]})^{-1}$$

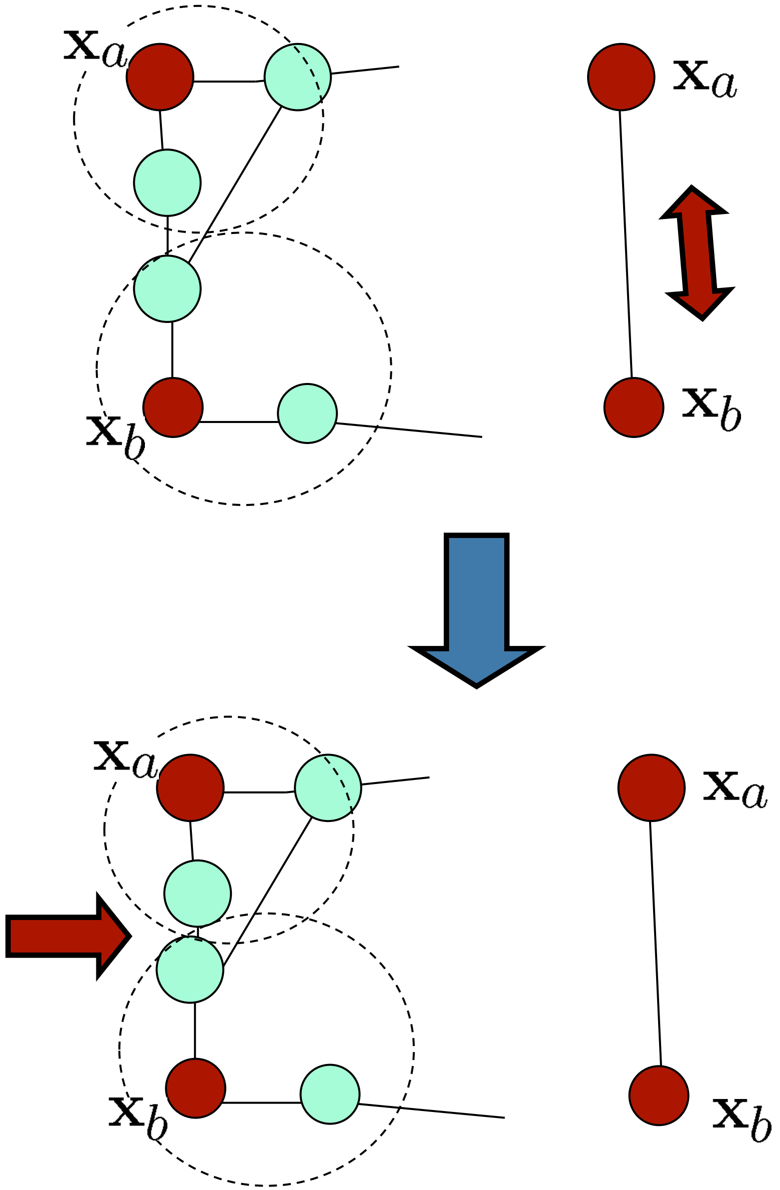
Propagating Information Downwards

- All representatives are nodes from the lower (bottom) level



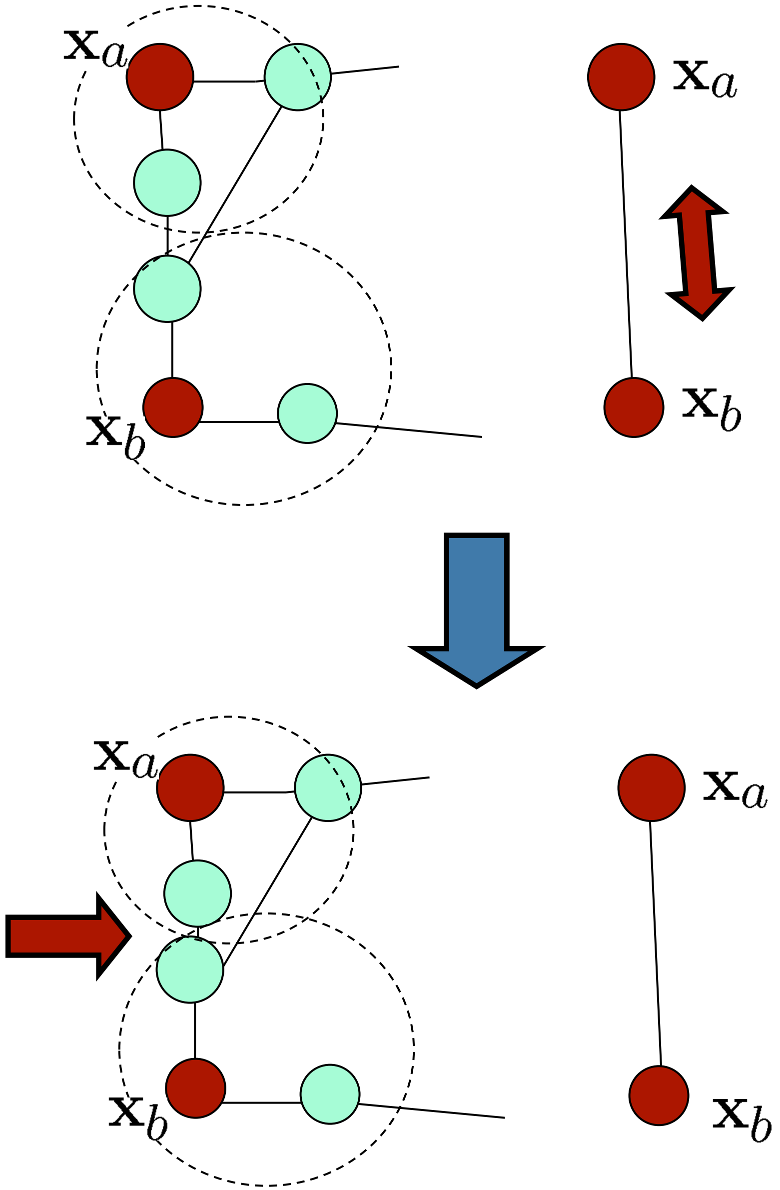
Propagating Information Downwards

- All representatives are nodes from the lower (bottom) level
- Information is propagated downwards by transforming the group at the lower level using a rigid body transformation



Propagating Information Downwards

- All representatives are nodes from the lower (bottom) level
- Information is propagated downwards by transforming the group at the lower level using a rigid body transformation
- Only if the lower level becomes inconsistent, optimize at the lower level



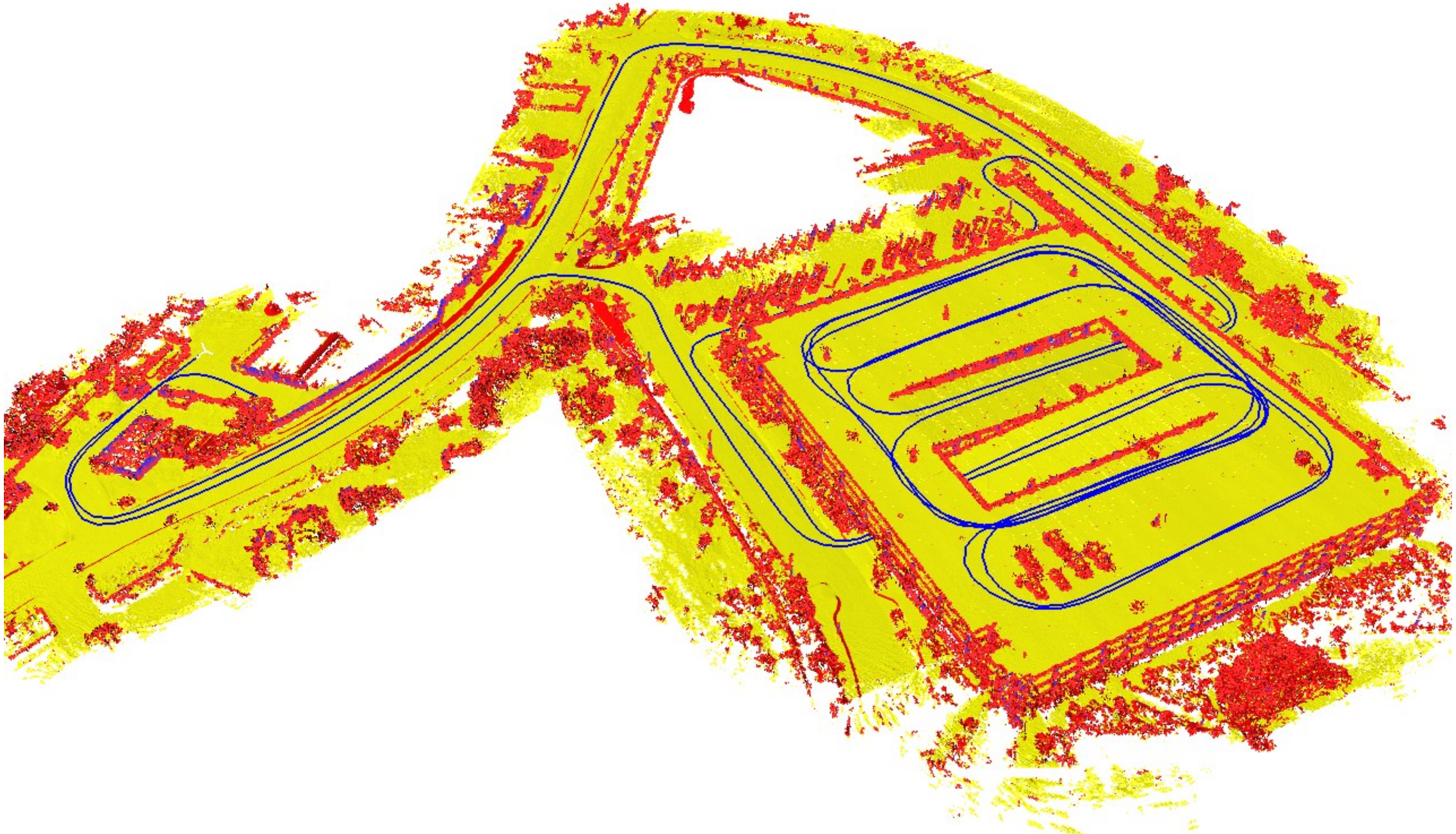
For the Best Possible Map...

- Make sure to run the optimization on the lowest level in the end
- For offline processing with all constraints, the hierarchy helps convergence faster in case of large errors
- In this case, one pass up the tree (to construct the edges) followed by one pass down the tree is sufficient

Stanford Garage

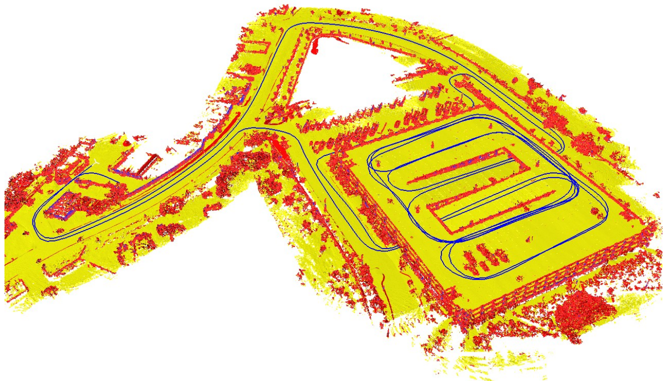
- Parking garage at Stanford University
- Nested loops, trajectory of ~7,000m

Stanford Garage Result

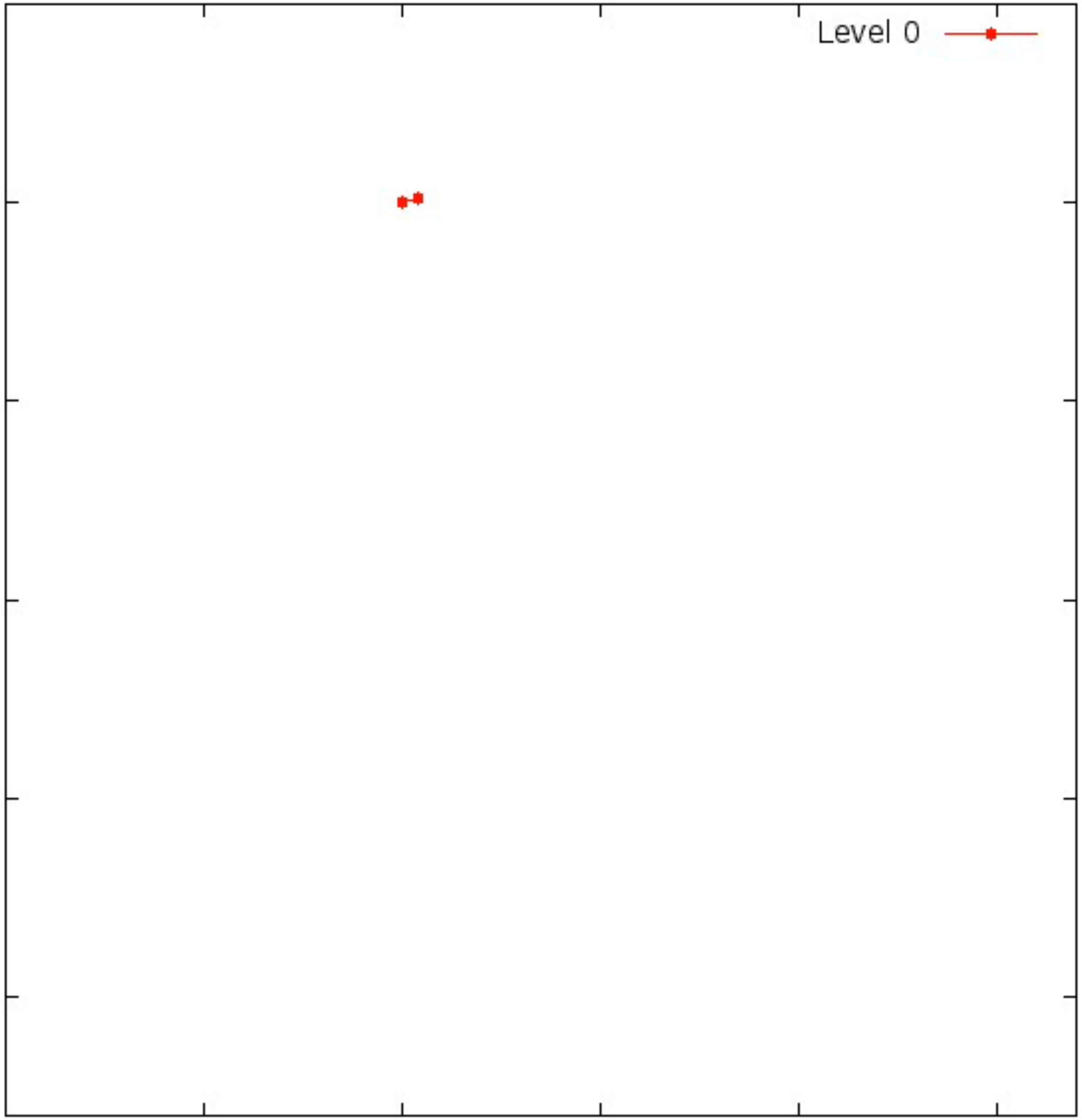


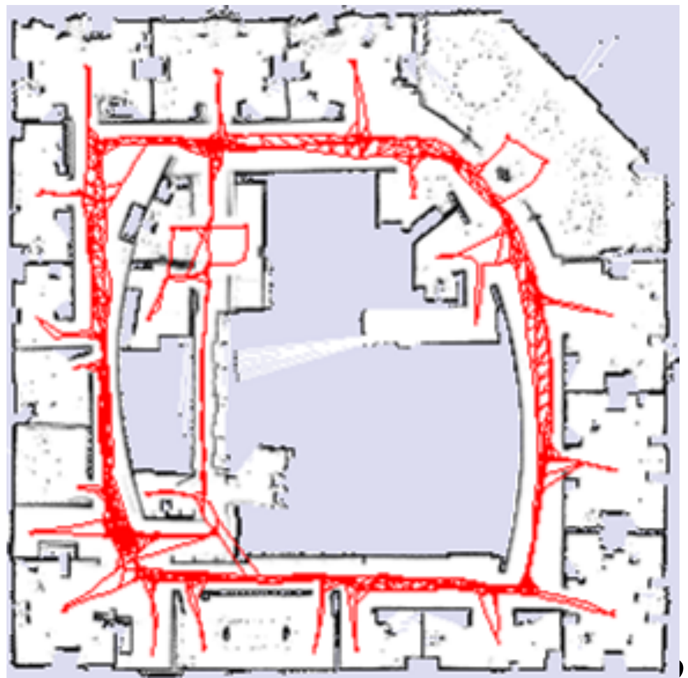
- Parking garage at Stanford University
- Nested loops, trajectory of ~7,000m

Stanford Garage Video



Intel Research Lab Video





Consistency

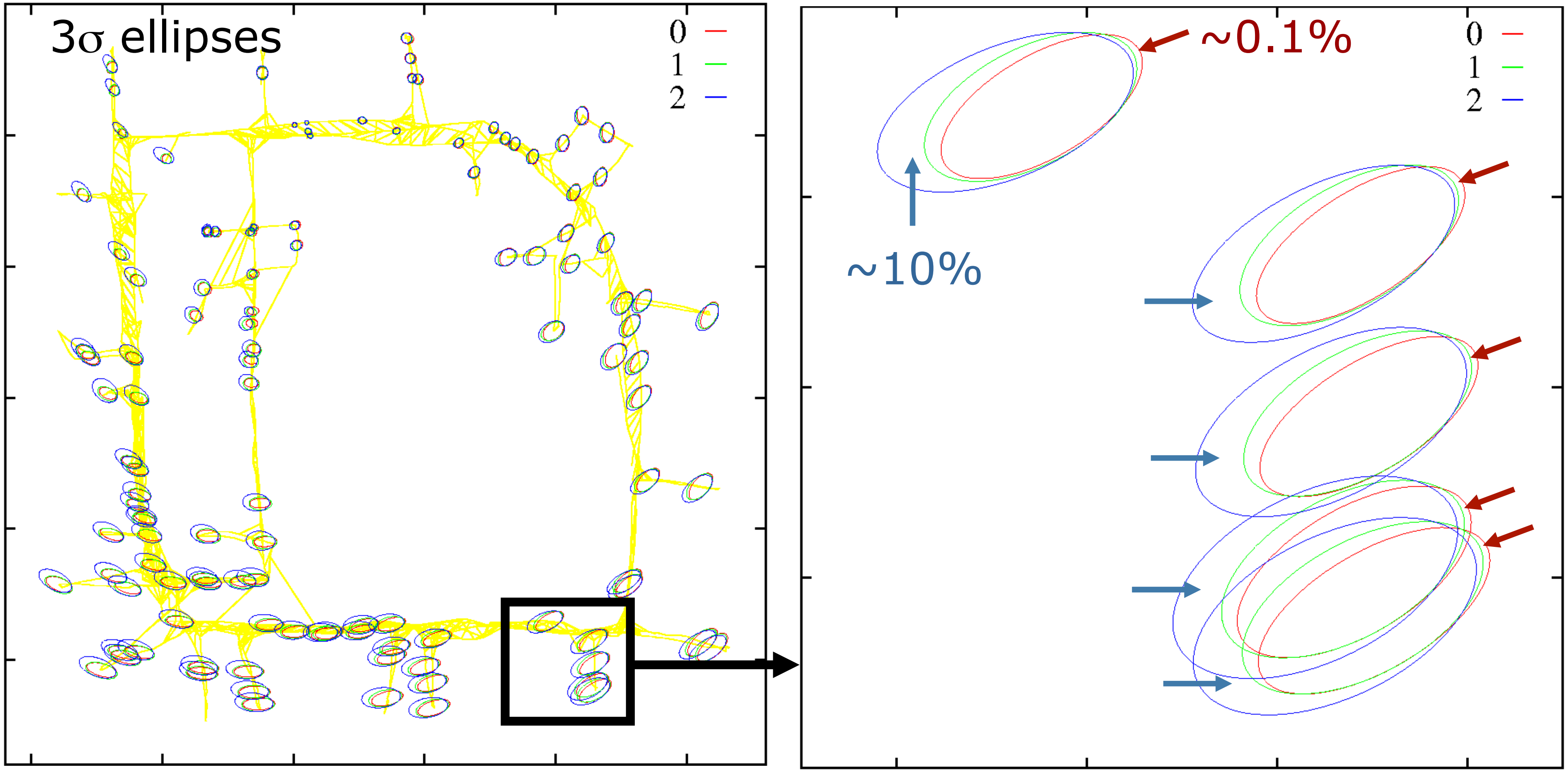
- Evaluation how well does the top level in the hierarchy represent the original input
- Probability mass of the marginal distribution in the highest level vs. the one of the true estimate (original problem, lowest level)

	Prob. mass not covered	Prob. mass outside
Intel	0.10%	10.18%
W-10000	2.53%	24.05%
Stanford	0.01%	7.88%
Sphere	2.75%	10.21%

low risk of becoming overly confident

one does not ignore too much information

Consistency



- Red: overly confident (~0.1% prob. mass)
- Blue: under confident (~10% prob. mass)

Conclusions

- Hierarchical pose-graph to estimate the structure to support efficient data association
- Designed for online mapping (interplay between optimization and data association)
- Higher level represent simplified problem

Literature

Hierarchical Pose-Graph Optimization

- Grisetti, Kümmerle, Stachniss, Frese, and Hertzberg: "Hierarchical Optimization on Manifolds for Online 2D and 3D Mapping"
- Open-source implementation hosted at http://openslam.org/hog-man.html