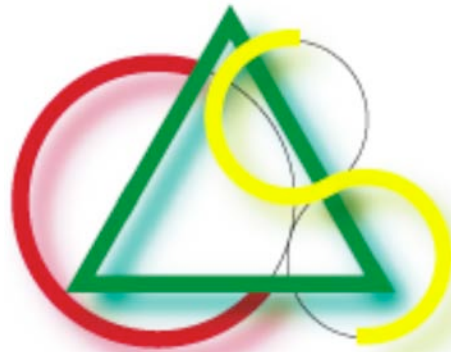


# Robust SLAM

John Folkesson & Henrik I. Christensen  
Centre for Autonomous Systems  
Kungl Tekniska Högskolan  
{johnf,hic}@nada.kth.se



Centre for Autonomous Systems

SLAM-2004: Robust SLAM

# Why is SLAM difficult?



- The number of map hypotheses is very large
- Often the signal to noise ratio for feature is  $\approx 1$
- Robust discriminative features are not common
- The “process” is often approximated

# Problems?



1. Flexible inclusion/exclusion of measurements?
2. Handling of linearization?
3. Dealing with topological constraints?
  - Loop closing etc. See also presentation by Leonard, Tardos & Nebot

# Data handling



- Easy inclusion and/or exclusion of data at any time in the process.
- How to avoid too early a commitment to a particular map hypothesis.
- Design of a representation that allow any-time inclusion/exclusion of data?



# Linearizations?

- Linearization might cause divergence in the data.
- Reported by several, e.g. [Julier & Uhlmann \(2001\)](#); [Castellanos \*et al.\* \(2004\)](#)
- Consistent handling of non-linearities
  - Start by exact handling of non-linearities
  - As data matures a linearization is permitted
  - Identification of major non-linearities to include them

# Topological constraints?



- Consistent inclusion of topological constraints
- Two step strategy:
  1. Close approximation of system in a trivial way
  2. Fine tune full model by adding “smaller” corrections



# Problem statement

- $\{x_i\}$  the robot path (set of poses), ( $i \in \{1 \dots N_p\}$ )
- $\{z_j\}$  feature coordinates ( $j \in \{1..N_m\}$ )
- $\{d_i\}$  dead reckoning measurements, between feature measurements
- $\{f_k\}$  feature measurements, ( $k \in \{1..N_f\}$ )
- $\Lambda$  the  $f \leftrightarrow z$  association

$$P(x, z, d, f, \Lambda) = P(d, f|x, z, \Lambda)P(x, z, \Lambda)$$



# Probabilistic model

$$P(x, z, d, f, \Lambda) \propto P(d, f|x, z, \Lambda)P(x, z, \Lambda)$$

$$P(d, f|x, z, \Lambda) \propto P(d|x)P(f|x, z, \Lambda)$$

$$P(x, z, \Lambda) \propto P(\lambda) = P(N_f) \propto e^{-\lambda N_f}$$

$$P(x, z, d, f, \Lambda) \propto P(d|x)P(f|x, z, \Lambda)e^{-\lambda N_f}$$



# An energy model

- Definition of energy/entropy of the model:

$$E(x, z, d, f, \Lambda) = -\log(P(d|x)) - \log(P(f|x, z, \Lambda)) + \lambda N_f$$

- Or  $E(x, z, d, f, \Lambda) = E_d + E_f + E_\Lambda$
- Or: . . . .

# An energy model



$$E(x, z, d, f, \Lambda) = E_d(x) + E_f(x, z) + E_\Lambda(n_j) \quad (1)$$

$$E_d = - \sum_{i=1}^{N_p} \log(P(d_i|x_{i-1}, x_i)) = \frac{1}{2} \sum_{i=1}^{N_p} \xi_i^T k_i \xi_i \quad (2)$$

$$E_f = - \log(P(f|x, z, \Lambda)) = \frac{1}{2} \sum_{k=1}^{N_m} \eta_k^T k_k \eta_k \quad (3)$$

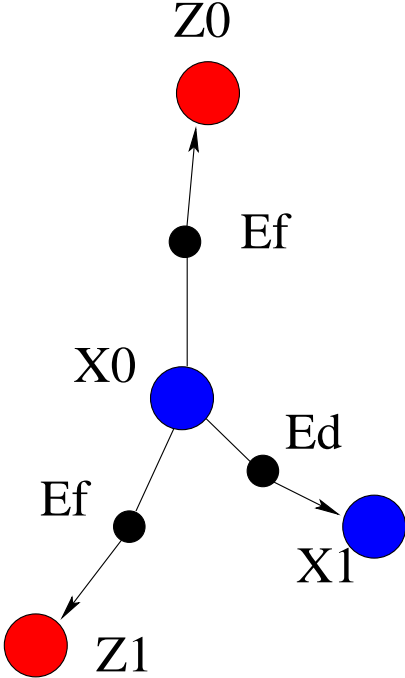
$$E_\Lambda = - \sum_{j=1}^{N_f} \lambda(n_j - 1) \quad (4)$$

$$\xi_i = T(x_i|x_{i-1}) - d_i \quad \eta_k = h(T(z_j|x_i)) - f_k$$

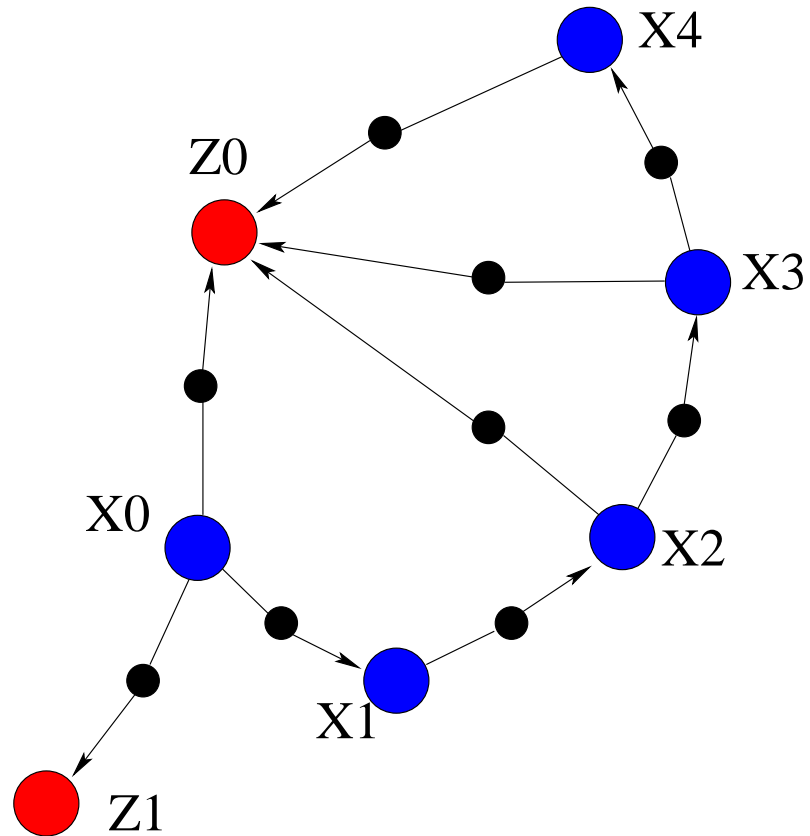
# Organizing a model

- Graph representation
- Two types of nodes:
  1. State nodes
    - Poses ( $x_i$ )
    - Features ( $z_j$ )
  2. Energy Nodes (Computation of Eqn (1))
    - Connected to the state nodes needed for computation
    - Movement ( $d_i, k_i$ )

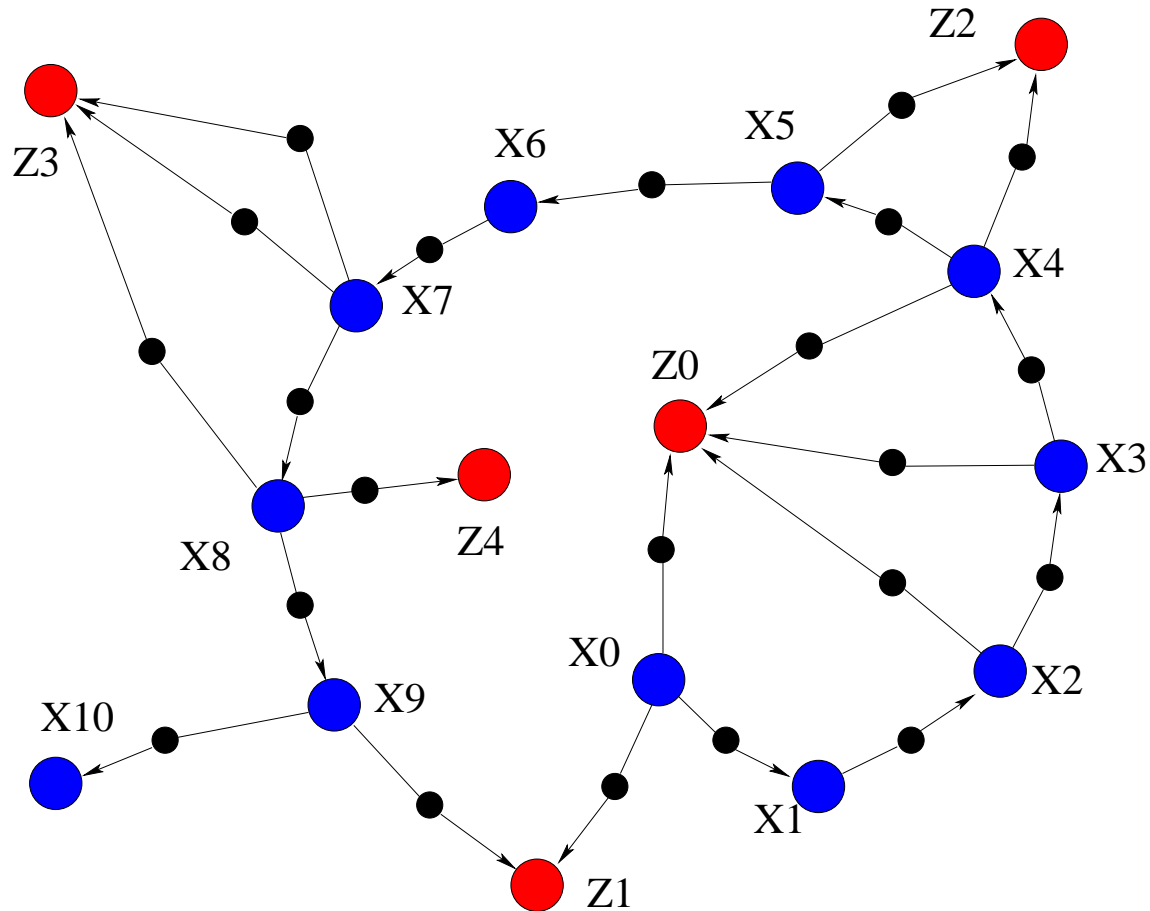
# Starting a model



# Entering more data



# A graphical model example



# Map updating



- Optimal solution to Eq. (1): ( $\operatorname{argmin}E$ ) is not realistic.
- Relaxation techniques allow iterative updating
- In a time step:
  1. Add a new state node (pose)
  2. Any new features/measurements?
  3. Update the rest of the map – minimize energy



# Map Update–Energy

- Eqn (1) Taylor expanded for a node A:

$$E_A = \sum_{i \in \text{edge}(A)} [E_i(\bar{x}_A, \bar{x}_i) + \nabla E_i(\bar{x}_A, \bar{x}_i) \begin{pmatrix} \Delta x_A \\ \Delta x_i \end{pmatrix} + \frac{1}{2} \begin{pmatrix} \Delta x_A & \Delta x_i \end{pmatrix} \nabla^T \nabla E_i(\bar{x}_A, \bar{x}_i) \begin{pmatrix} \Delta x_A \\ \Delta x_i \end{pmatrix}]$$

Notation:

$$\mathcal{G} = \nabla E_A = \begin{pmatrix} \mathcal{G}_A \\ \mathcal{G}_i \end{pmatrix}$$
$$\mathcal{H} = \nabla^T \nabla E_A = \begin{pmatrix} \mathcal{H}_{AA} & \mathcal{H}_{Ai} \\ \mathcal{H}_{Ai} & \mathcal{H}_{ii} \end{pmatrix}$$



## Map Update–Energy (II)

- Optimizing a node A:

$$(x_A - \bar{x}_A) = -\mathcal{H}_{AA}^{-1} \mathcal{G}_A \quad (5)$$

- With an energy change of

$$\Delta E_A = \frac{1}{2} \mathcal{G}_A^T \mathcal{H}_{AA}^{-1} \mathcal{G}_A \quad (6)$$

# Map Update–Energy (III)



- $\Delta E_A$  decides on update strategy:
  - Steepest decent – close to saddle point
  - Use eqn. (5) direct
  - Chained update
    - Locate a node that is “good”
    - Update from that node



# Feature Matching

- $\lambda$  decides on the value of new measurements
  1. Add a new feature
  2. Compute energy change
  3. If change too large, remove association
- Similar to EM ([Thrun et al. , 1998](#))
- Matching/graph updating anytime

# Example

- ATRV Vehicle with Trimble DGPS 212
- Sick LMS 291 Laser scanner
- CrossBow DFOG INS package
- State  $(\theta_n, x_n, y_n)^T$
- Lines w. end-points  $(x, y)$
- See [Folkesson & Christensen \(2003\)](#) for details
- Operating around a house.
- 7500 Pose Nodes, 11975 line measurements
- Update time 30 ms (550 MHz Pentium)

# SLAM example



# Status?



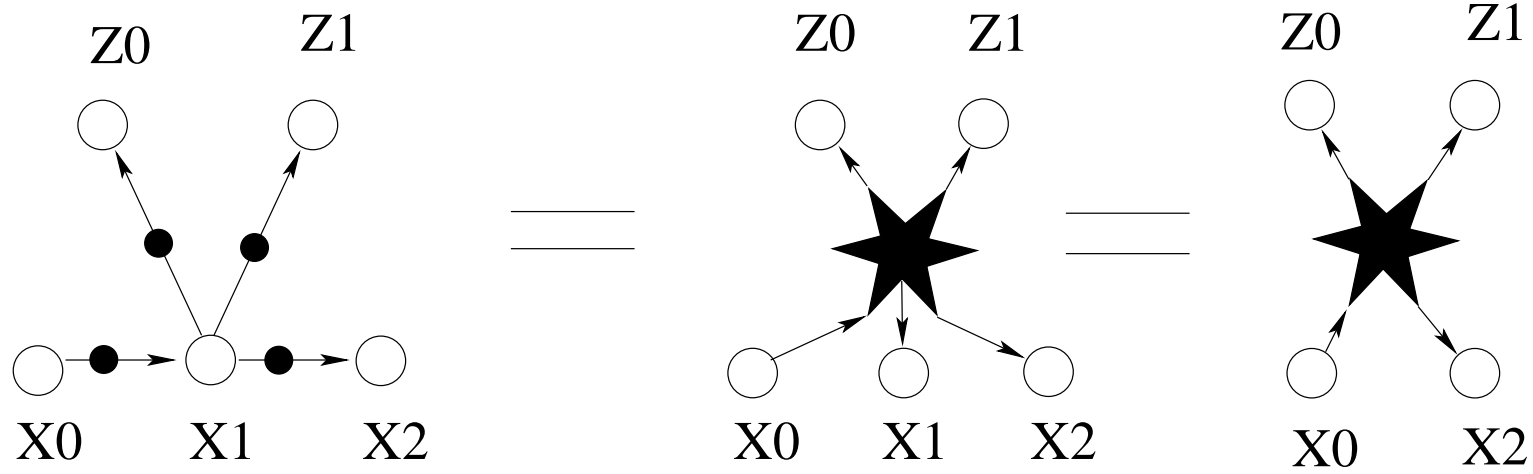
- Flexible association of data, anytime
- A framework to avoid linearization effects
- Loop closing:
  1. Recognition of loop closure (place recognition)
  2. Updating map to include the topological constraint
- Doing 2) often has limited effect
- A proposal . . .



# Reducing the graph

- How can state nodes be removed from the graph?
- In a linear system one could generate a closed form solution
- In a non-linear case an approximation can be used.
- Consider elimination of a node  $A$  given neighboring nodes  $B$ .
- Consider the energy from state  $A$  expanded.

# The basic idea



# The Energy from A



$$E_A = \sum_{j \in \text{edge}(A)} [E_i(\bar{x}_A, \bar{x}_j) + \mathcal{G}_j(x_j - \bar{x}_j) + \frac{1}{2} \begin{pmatrix} x_A - \bar{x}_A & x_j - \bar{x}_j \end{pmatrix} \mathcal{H}_j(\bar{x}_A, \bar{x}_j) \begin{pmatrix} x_A - \bar{x}_A \\ x_j - \bar{x}_j \end{pmatrix}] \quad (7)$$

Transforming to B:

$$(x_A - \bar{x}_A) = -\mathcal{H}_{AA}^{-1} \mathcal{H}_{AB} (x_B - \bar{x}_B)$$

# Elimination of A

- Substitution into eqn (7) eliminates A

$$\begin{aligned}
 E^* &= \sum_{j \in \text{edge}(A)} [E_j(\bar{x}_A, \bar{x}_j) + \mathcal{G}_j(x_j - \bar{x}_j) \\
 &+ \frac{1}{2} (x_j - \bar{x}_j)^T \mathcal{H}_{jj} (x_j - \bar{x}_j) \\
 &- \sum_{i \in \text{edge}(A)} (x_i - \bar{x}_i)^T \mathcal{H}_{iA} \mathcal{H}_{AA}^{-1} \mathcal{H}_{Aj} (x_j - \bar{x}_j)]
 \end{aligned}$$

- New term connects i and j (new connection)

# Star nodes



- The fused nodes are termed *Star* nodes.
- In principle the entire graph could be fused into a single node, as done in EKF.
- In practice star nodes are used to fuse local “maps”
- Typically upto 128 state nodes are considered for “fusion” .
- The star nodes are considered retroactively after say 50 poses.

# Making star nodes invariant



- Star nodes updates energy using the Hessian  $\mathcal{H}$
- If measurements have symmetries (such as lines)  $\mathcal{H}$  will have zero eigenvalues.
- Project state nodes to “natural coordinates” ( $q$ ) in a lower dimensional space without symmetries.

$$q_i = PT(x_i|x_0)$$

# Mapping to natural coordinates



- In general:

$$\mathcal{H}_{xx} = J^T \mathcal{H}_{qq} J + \frac{\partial^2 q}{\partial x \partial x} \mathcal{G}_q \approx J^T \mathcal{H}_{qq} J$$

- Thus:

$$\mathcal{H}_{qq} = \tilde{J} \mathcal{H}_{xx} \tilde{J}^T$$

- Using SVD it is possible to find eigenvalues  $b_j$  and eigenvectors  $V_j$  of the Hessian

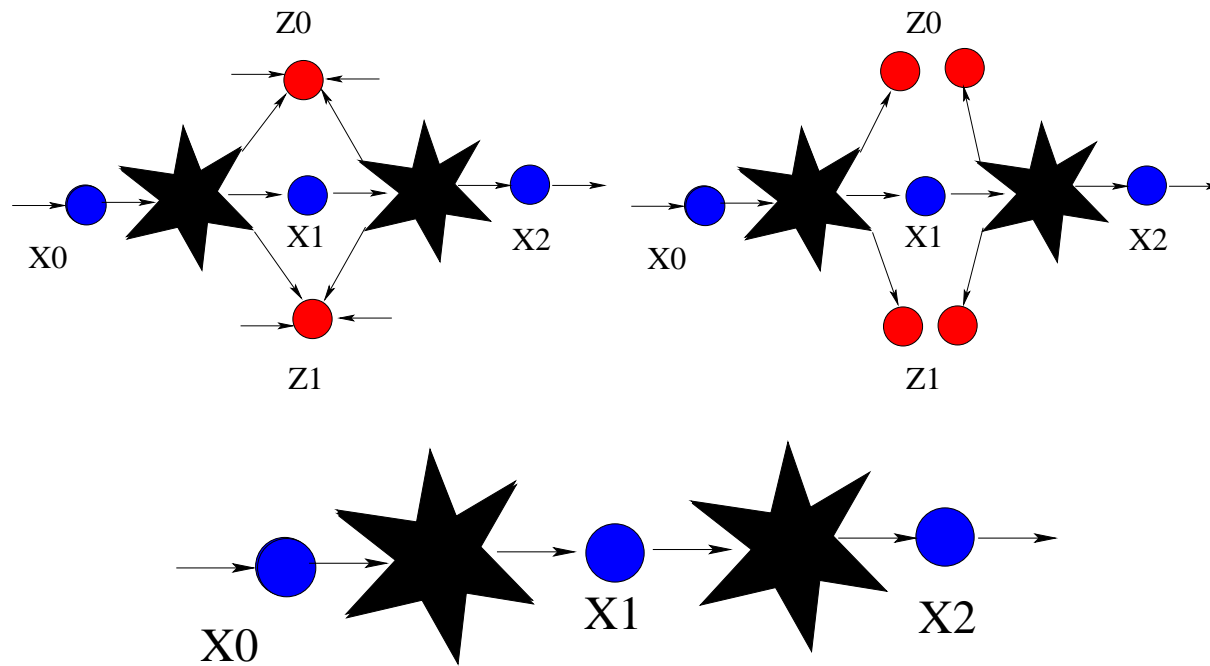
# Energy & Properties

- Using the eigenvectors the energy at  $\bar{q}$  is now:

$$E^* = E^*(\bar{q}) + \frac{1}{2} \sum_j b_j (V_j \Delta q)^2$$

- Recentered nodes are linear in energy
- A state node connected to only one star node can be eliminated ( $\Delta q = 0$ )
- Star nodes like local maps in Atlas ([Bosse et al. , 2003](#))

# Optimizing global calculations



# Closing loops

- With  $\Delta q = 0$  the state nodes between star nodes is ignored.
- Define a cost function using Lagrange multipliers

$$C(\Lambda, \Delta q) = \Lambda \left( \sum_i \Delta x - d_c \right) + \frac{1}{2} \sum_i \sum_j b_{ji} (V_{ji} \Delta q)^2$$

- Where  $i$  is the star node index,  $\Delta x_i$  is the pose difference between the poses.  $d_c$  is the pose constraint.
- I.e.

$$\Delta x_i = \begin{pmatrix} \mathcal{R}_i & 0 \\ 0 & 1 \end{pmatrix} (\Delta q_i + \bar{q}_i)$$

# Solving for closed loops

- Linearisation gives:

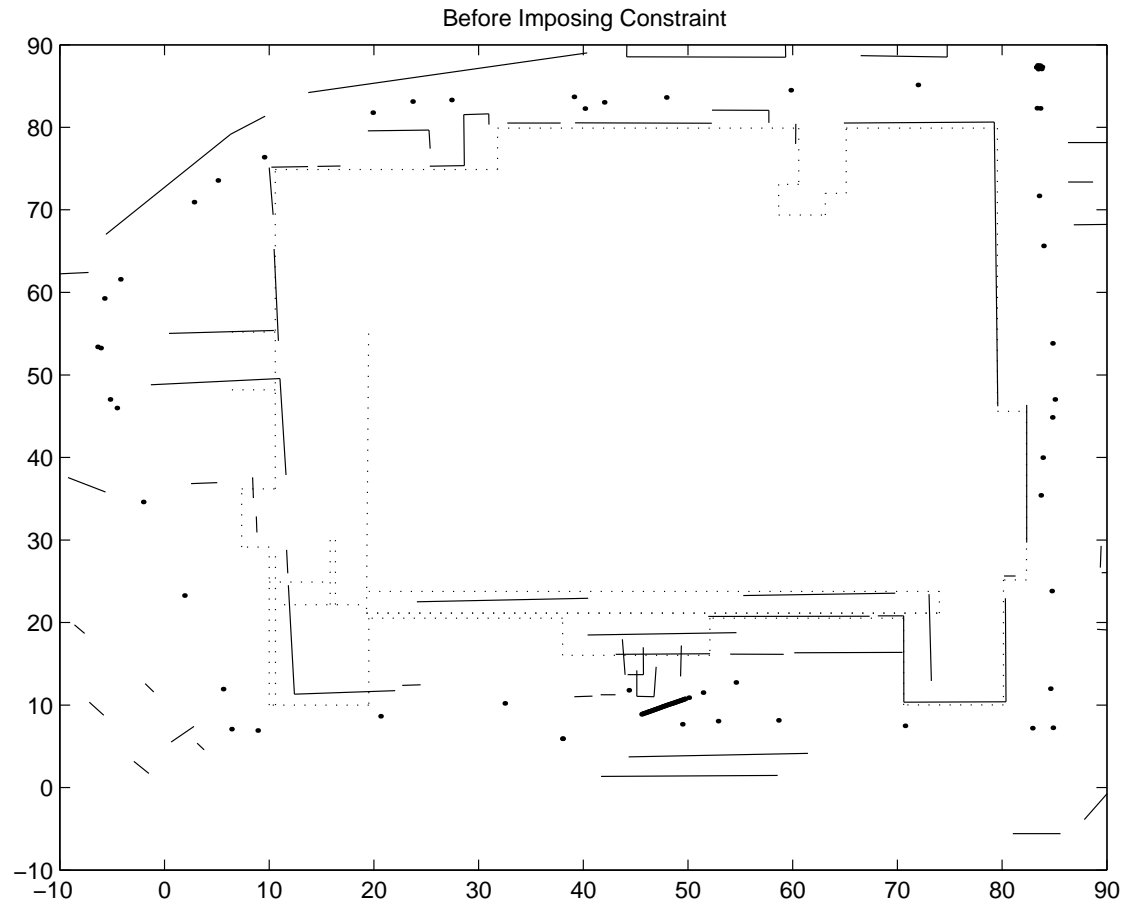
$$\Delta q_i = - \sum_j \frac{V_{ji}^T V_{ji}}{b_j} \begin{pmatrix} \mathcal{R}_i & 0 \\ 0 & 1 \end{pmatrix} \Lambda^T$$

- Where:

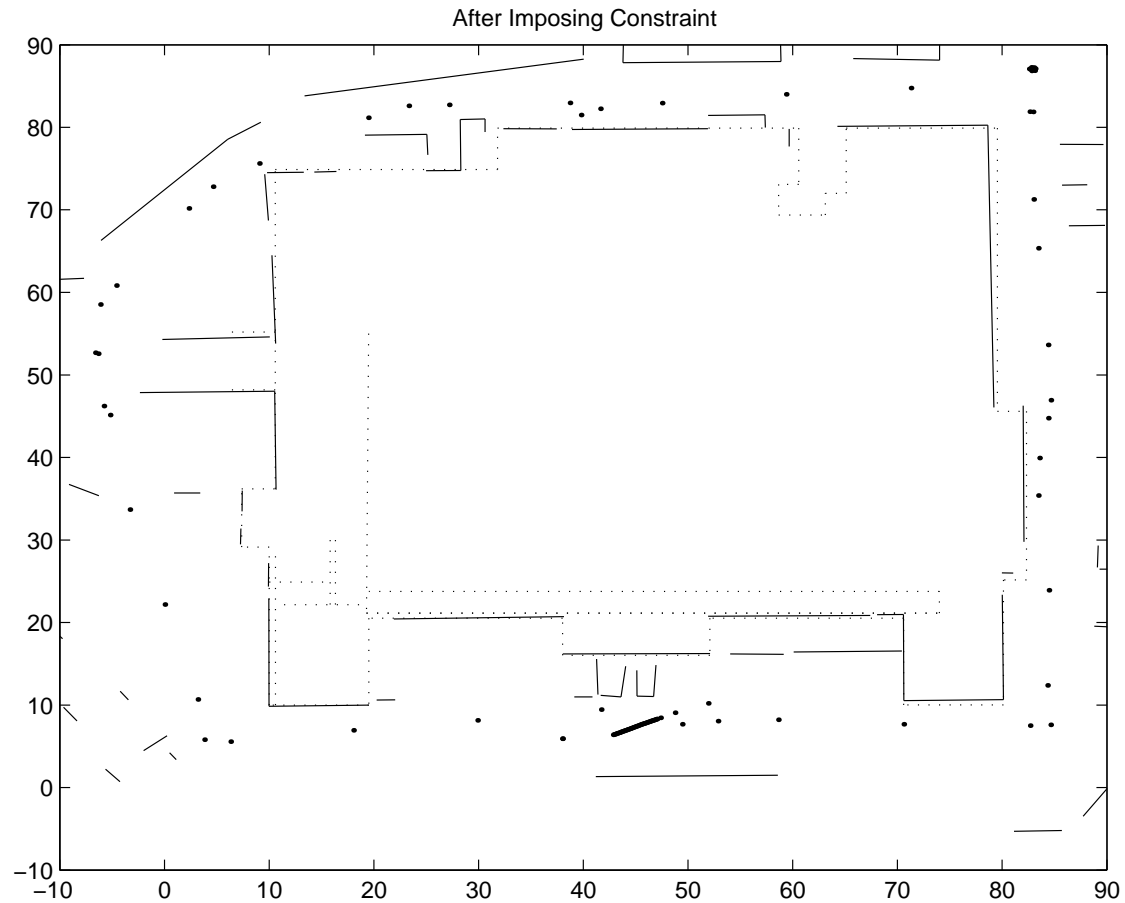
$$\Lambda^T = S^{-1} \left\{ \left[ \sum_i \begin{pmatrix} \mathcal{R}_i & 0 \\ 0 & 1 \end{pmatrix} \bar{q}_i \right] - d_c \right\}$$

$$S = \sum_i \left\{ \begin{pmatrix} \mathcal{R}_i & 0 \\ 0 & 1 \end{pmatrix} \left[ \sum_j \frac{V_{ji}^T V_{ji}}{b_j} \right] \begin{pmatrix} \mathcal{R}_i & 0 \\ 0 & 1 \end{pmatrix} \right\}$$

# Map example - no closing



# Map example - closing



# Example Movie



Video

# Closing loops



- Loop closing has a complexity of  $O(N + 1)$  where  $N$  is the number of star nodes.
  - Once loop closing is achieved
  - Turn back on inter-star relation to fine tune
- Achieved in 1-2 seconds for large environments
- Constraints similar to “strong links” in [Lu & Milios \(1997\)](#).

# Summary



- A new efficient representation for SLAM
  - Easy integration of data anytime
  - “Chunk-ing” data into local maps (star nodes)
  - Addition of constraint for loop closing
- Computationally efficient (10-12 ms / pose)
- Losing closing without linearization problems
- Further details in [Folkesson & Christensen \(2004b,a\)](#)

# References



## \*References

- Bosse, M., Newman, P., & Leonard, J. 2003. An ATLAS framework for scalable mapping. *Pages 1899–1906 of: Proc. ICRA-03*, vol. 1.
- Castellanos, J. A., Neira, J., & Tardos, J. D. 2004. Limits to the Consistency of the EKF-based SLAM. *In: Ribeiro, M. Isabel, & Santos-Victor, J. (eds), Intelligent Autonomous Vehicles (IAV-2004)*. Lisboa, PT: IFAC/Elsevier, for IFAC/EURON.
- Folkesson, J., & Christensen, H. I. 2003 (sep). Outdoor exploration and SLAM using a compressed filter. *Pages 419–426 of: Proc. ICRA-03*.
- Folkesson, J., & Christensen, H. I. 2004a (April). Graphical SLAM - A self-correcting map. *In: ICRA-05*. IEEE, New Orleans.
- Folkesson, J., & Christensen, H. I. 2004b (July 5-7). Robust SLAM. *In: IFAC – 5th IAV-2004*.
- Julier, S. J., & Uhlmann, J. K. 2001 (May). A counter example to the theory of simultaneous localisation and map building. *Pages 4238–4243 of: Proc. ICRA-01*.
- Lu, F., & Milius, E. 1997. Globally Consistent Range Scan Alignment for Environmental Mapping. *Autonomous Robots*, **4**, 333–349.
- Thrun, S., Burgard, W., & Fox, D. 1998. A Probabilistic Approach to Concurrent Mapping and Localization for Mobile Robots. *Autonomous Robots*, **5**, 253–271.