Robot Mapping

TORO – Gradient Descent for SLAM

Gian Diego Tipaldi, Luciano Spinello, Wolfram Burgard
Stochastic Gradient Descent

- Minimize the error individually for each constraint (decomposition of the problem into sub-problems)
- Solve one step of each sub-problem
- Solutions might be contradictory
- The magnitude of the correction decreases with each iteration
- Learning rate to achieve convergence

[First used in the SLAM community by Olson et al., ’06]
Stochastic Gradient Descent

- Minimize the error individually for each constraint (decomposition of the problem into sub-problems)
- Solve one step of each sub-problem
- Solutions might be contradictory
- The magnitude of the correction decreases with each iteration
- Learning rate to achieve convergence

[First used in the SLAM community by Olson et al., ’06]
Stochastic Gradient Descent

- Minimize the error individually for each constraint (decomposition of the problem into sub-problems)
- Solve one step of each sub-problem
- Solutions might be contradictory
- The magnitude of the correction decreases with each iteration
- Learning rate to achieve convergence

[First used in the SLAM community by Olson et al., ’06]
Preconditioned SGD

- Minimize the error individually for each constraint
- Solve one step of each sub-problem
- A solution is found when an equilibrium is reached
- Update rule for a single constraint:

$$x^{t+1} = x^t + \lambda H^{-1} J^T \Omega_{ij} r_{ij}$$
Node Parameterization

- How to represent the nodes in the graph?
- Impacts which parts need to be updated for a single constraint update
- Transform the problem into a different space so that:
  - the structure of the problem is exploited
  - the calculations become fast and easy

\[ x = g(p) \leftrightarrow p = g^{-1}(x) \]
\[ x^* = \arg\min_x \sum_{i,j} e'_{ij}(x)^T \Omega_{ij} e'_{ij}(x) \]
Parameterization of Olson

- Incremental parameterization:

\[ x_i = p_i - p_{i-1} \]

- **Problem:** for optimizing a constraint between the nodes \( i \) and \( k \), one needs to updates the nodes \( i, \ldots, k \) ignoring the topology of the environment
Alternative Parameterization

- Exploit the topology of the space to compute the parameterization
- Idea: “Loops should be one sub-problem”
- Such a parameterization can be extracted from the graph topology itself
Tree Parameterization

- How should such a problem decomposition look like?
Tree Parameterization

- Use a spanning tree!
Tree Parameterization

- Construct a spanning tree from the graph
- Mapping between poses and parameters
  \[ X_i = P_{\text{parent}(i)}^{-1} P_i \]
- Error of a constraint in the new parameterization

Only variables along the path of a constraint are involved in the update

\[ E_{ij} = \Delta_{ij}^{-1} \text{UpChain}^{-1} \text{DownChain} \]
Stochastic Gradient Descent With The Tree Parameterization

- The tree parameterization leads to several smaller problems which are either:
  - constraints on the tree ("open loop")
  - constraints not in the tree ("a loop closure")
- Each SGD equation independently solves one sub-problem at a time
- The solutions are integrated via the learning rate
Computation of the Update Step

- 3D rotations are non-linear
- Update according to the SGD equation may lead to poor convergence

SGD update:

$$\Delta x = \lambda H^{-1} J_i^T \Omega_i r_i$$

- Idea: distribute a fraction of the residual along the parameters so that the error of that constraint is reduced
Computation of the Update Step

Alternative update in the “spirit” of the SGD: Smoothly deform the path along the constraints so that the error is reduced.

Distribute the rotational error
Distribute the translational error
Rotational Error

- In 3D, the rotational error cannot be simply added to the parameters because the rotations are not commutative.
- Find a set of **incremental** rotations so that the following equality holds:

\[ R_1 R_2 \cdots R_n B = R'_1 R'_2 \cdots R'_n \]
Rotational Residual

- Let the first node be the reference frame
- We want a correcting rotation around a single axis
- Let $A_i$ be the orientation of the i-th node in the global reference frame

\[ A'_n = A_n B \]
Rotational Residual

- Written as a rotation in global frame
  \[ A'_n = A_n B = QA_n \]
- with a decomposition of the rotational residual into a chain of incremental rotations obtained by spherical linear interpolation (slerp)
  \[ Q = Q_1Q_2 \cdots Q_n \]
  \[ Q_k = \text{slerp}(Q, u_{k-1})^T \text{slerp}(Q, u_k) \quad u \in [0 \ldots \lambda] \]
- Slerp designed for 3d animations: constant speed motion along a circle
What is the SLERP?

- Spherical LinEar inteRPolation
- Introduced by Ken Shoemake for interpolations in 3D animations
- Constant speed motion along a circle arc with unit radius
- Properties:

\[ R' := \text{slerp}(R, u) \]
\[ \text{axisOf}(R') = \text{axisOf}(R) \]
\[ \text{angleOf}(R') = u \text{ angleOf}(R) \]
Rotational Residual

- Given the $Q_k$, we obtain
  \[ A'_k = Q_1 \ldots Q_k A_k = Q_{1:k} A_k \]

- as well as
  \[ R'_k = A'_{k-1}^T A'_k \]

- and can then solve:
  \[
  \begin{align*}
  R'_1 &= Q_1 R_1 \\
  R'_2 &= (Q_1 R_1)^T Q_{1:2} R_{1:2} = R_1^T Q_1^T Q_1 Q_2 R_1 R_2 \\
  & \vdots \\
  R'_k &= [(R_{1:k-1})^T Q_k R_{1:k-1}] R_k
  \end{align*}
  \]
Rotational Residual

- Resulting update rule

\[ R'_k = (R_{1:k-1})^T Q_k R_{1:k} \]

- It can be shown that the change in each rotational residual is bounded by

\[ \Delta r'_{k,k-1} \leq \text{angleOf}(Q_k) \]

- This bounds a potentially introduced error at node k when correcting a chain of poses including k
How to Determine $u_k$?

- The $u_k$ describe the distribution of the error
  \[ Q_k = \text{slerp}(Q, u_{k-1})^T \text{slerp}(Q, u_k) \quad u \in [0 \ldots \lambda] \]

- Consider the uncertainty of the constraints

\[
\begin{align*}
    u_k &= \min \left(1, \lambda |\mathcal{P}_{ij}| \right) \left[ \sum_{m \in \mathcal{P}_{ij} \land m \leq k} d_m^{-1} \right]^{-1} \\
    d_m &= \sum_{\langle l, m \rangle} \min \left[ \text{eigen}(\Omega_{lm}) \right] \\
    \text{all constraints connecting m}
\end{align*}
\]

- This assumes roughly spherical covariances!
Distributing the Translational Error

- That is trivial
- Just scale the x, y, z movements
Summary of the Algorithm

- Decompose the problem according to the tree parameterization

- Loop:
  - Select a constraint
    - Randomly or sample inverse proportional to the number of nodes involved in the update
  - Compute the nodes involved in update
    - Nodes according to the parameterization tree
  - Reduce the error for this sub-problem
    - Reduce the rotational error (slerp)
    - Reduce the translational error
Complexity

- In each iteration, the approach handles all constraints
- Each constraint optimization requires to update a set of nodes (on average: the average path length according to the tree)

\[ \mathcal{O}(Ml) \]

#constraints \quad \text{avg. path length (parameterization tree)}
Cost of a Constraint Update

\[ \approx \mathcal{O}(M \log(N)) \]
Node Reduction

- Complexity grows with the length of the trajectory
- Combine constraints between nodes if the robot is well-localized
  \[ \Omega_{ij} = \Omega_{ij}^{(1)} + \Omega_{ij}^{(2)} \]
  \[ z_{ij} = \Omega_{ij}^{-1} \left( \Omega_{ij}^{(1)} z_{ij}^{(1)} + \Omega_{ij}^{(2)} z_{ij}^{(2)} \right) \]
- Similar to adding rigid constraints
- Then, complexity depends on the size of the environment (not trajectory)
Simulated Experiment

- Highly connected graph
- Poor initial guess
- 2200 nodes
- 8600 constraints
Spheres with Different Noise
Mapping the EPFL Campus

- 10km long trajectory with 3D laser scans
Mapping the EPFL Campus
TORO vs. Olson’s Approach

Olson’s approach

1 iteration

10 iterations

50 iterations

100 iterations

300 iterations

TORO
TORO vs. Olson’s Approach

![Graphs showing comparison between TORO and Olson’s approach]

- Olson’s approach
- Tree approach + node reduction
- Tree approach

![Comparative plots with error per constraint against iteration]

- Olson’s approach (big noise)
- Tree approach (big noise)
- Olson’s approach (small noise)
- Tree approach (small noise)
Time Comparison

![Bar chart showing execution time comparison across different methods and constraint counts.](chart)
Robust to the Initial Guess

- Random initial guess
- Intel dataset as the basis for 16 floors distributed over 4 towers

initial configuration  intermediate result  final result (50 iterations)
Drawbacks of TORO

- The slerp-based update rule optimizes rotations and translations separately
- It assumes roughly *spherical covariance* ellipses
- Slow convergence speed close to minimum
- No covariance estimates
Conclusions

- TORO - Efficient maximum likelihood estimate for 2D and 3D pose graphs
- Robust to bad initial configurations
- Efficient technique for ML map estimation (or to initialize GN/LM)
- Works in 2D and 3D
- Scales up to millions of constraints
- Available at OpenSLAM.org
  http://www.openslam.org/toro.html
Literature

SLAM with Stochastic Gradient Descent

- Olson, Leonard, Teller: “Fast Iterative Optimization of Pose Graphs with Poor Initial Estimates”
- Grisetti, Stachniss, Burgard: “Non-linear Constraint Network Optimization for Efficient Map Learning”
Slide Information

- These slides have been created by Cyrill Stachniss as part of the robot mapping course taught in 2012/13 and 2013/14. I created this set of slides partially extending existing material of Giorgio Grisetti, Wolfram Burgard, and myself.

- I tried to acknowledge all people that contributed image or video material. In case I missed something, please let me know. If you adapt this course material, please make sure you keep the acknowledgements.

- Feel free to use and change the slides. If you use them, I would appreciate an acknowledgement as well. To satisfy my own curiosity, I appreciate a short email notice in case you use the material in your course.

- My video recordings are available through YouTube: http://www.youtube.com/playlist?list=PLgnQpQtFTOGQrZ4O5Qzb1Hgl3b1JHimN_&feature=g-list