Introduction to Mobile Robotics

Graph-Based SLAM

Wolfram Burgard
Particle Filter: Campus Map

- 30 particles
- 250x250m²
- 1.75 km (odometry)
- 20cm resolution during scan matching
- 30cm resolution in final map
Graph-Based SLAM

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

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

Robot pose
Constraint (odometry)
Constraint (loop closure)
Example: Odometry Map

Hanover2 dataset (Courtesy of Oliver Wulf)
Example: Loop Closures

Hanover2 dataset (Courtesy of Oliver Wulf)
How to correct the trajectory?

Imagine this to be a system of masses and springs!
Idea of Graph-Based SLAM

- 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
Graph-Based SLAM in a Nutshell

- Every node in the graph corresponds to a robot position and a laser measurement.
- An edge between two nodes represents a spatial constraint between the nodes.

KUKA Halle 22, courtesy of P. Pfaff
Graph-Based SLAM in a Nutshell

- Every node in the graph corresponds to a robot position and a laser measurement
- An edge between two nodes represents a spatial constraint between the nodes
Graph-Based SLAM in a Nutshell

- Once we have the graph, we determine the most likely map by correcting the nodes
Graph-Based SLAM in a Nutshell

- Once we have the graph, we determine the most likely map by correcting the nodes
  ... like this
Graph-Based SLAM in a Nutshell

- Once we have the graph, we determine the most likely map by correcting the nodes
  ... like this
- Then, we can render a map based on the known poses
The Overall SLAM System

- Interplay of front-end and back-end
- A consistent map helps to determine new constraints by reducing the search space
- This lecture focuses only on the optimization
Least Squares in General

- Approach for computing a solution for an **overdetermined system**
- “More equations than unknowns”
- Minimizes the **sum of the squared errors** in the equations
- Standard approach to a large set of problems
Problem

- Given a system described by a set of $n$ observation functions $\{f_i(x)\}_{i=1:n}$
- Let
  - $x$ be the state vector
  - $z_i$ be a measurement of the state $x$
  - $\hat{z}_i = f_i(x)$ be a function which maps $x$ to a predicted measurement $\hat{z}_i$
- Given $n$ noisy measurements $z_{1:n}$ about the state $x$

Goal: Estimate the state $x$ which bests explains the measurements $z_{1:n}$
Graphical Explanation

\[ f_1(x) = \hat{z}_1 \quad \text{z}_1 \]
\[ f_2(x) = \hat{z}_2 \quad \text{z}_2 \]
\[ \ldots \]
\[ f_n(x) = \hat{z}_n \quad \text{z}_n \]

- state (unknown)
- predicted measurements
- real measurements
Error Function

- Error $e_i$ is typically the **difference** between the **predicted and actual** measurement.

  $$e_i(x) = z_i - f_i(x)$$

- We assume that the error has **zero mean** and is **normally distributed**.

- Gaussian error with information matrix $\Omega_i$.

- The squared error of a measurement depends only on the state and is a scalar.

  $$e_i(x) = e_i(x)^T \Omega_i e_i(x)$$
Least Squares for SLAM

- Overdetermined system for estimating the robot’s poses given observations
- “More observations than states”
- Minimizes the **sum of the squared errors**

Today: Application to SLAM
The Graph

- It consists of $n$ nodes $x = x_{1:n}$
- Each $x_i$ is a 2D or 3D transformation (the pose of the robot at time $t_i$)
- A constraint/edge exists between the nodes $x_i$ and $x_j$ if...
Create an Edge If... (1)

- ...the robot moves from $x_i$ to $x_{i+1}$
- Edge corresponds to odometry

The edge represents the **odometry** measurement
Create an Edge If... (2)

- ...the robot observes the same part of the environment from $x_i$ and from $x_j$
- Construct a **virtual measurement** about the position of $x_j$ seen from $x_i$
Create an Edge If... (2)

- ...the robot observes the same part of the environment from $x_i$ and from $x_j$
- Construct a **virtual measurement** about the position of $x_j$ seen from $x_i$

Edge represents the position of $x_j$ seen from $x_i$ based on the **observation**
Pose Graph

observation of $x_j$ from $x_i$

nodes according to the graph

$\langle z_{ij}, \Omega_{ij} \rangle$

edge

e$e_{ij}(x_i, x_j)$

error
Pose Graph

Goal: \[ x^* = \arg\min_x \sum_{ij} e_{ij}^T \Omega_{ij} e_{ij} \]
Gauss-Newton: The Overall Error Minimization Procedure

- Define the error function
- Linearize the error function
- Compute its derivative
- Set the derivative to zero
- Solve the linear system
- Iterate this procedure until convergence
Sparse Pose Adjustment
Example: CS Campus Freiburg
There are Variants for 3D

- Highly connected graph
- Poor initial guess
- LU & variants fail
- 2200 nodes
- 8600 constraints
Hanover2: 3D SLAM Map
Campus : Scan Matching Map
Campus : Graph Optimization
Campus : SLAM Map
Freiburg Campus Octomap
Example: Stanford Garage
3D Map of the Stanford Parking Garage

approx. 260MB
Application: Navigation with the Autonomous Car Junior

- Task: reach a parking spot on the upper level of the garage.
Autonomous Parking
Graph-SLAM with more Sensors

Graph SLAM is flexible regarding additional information (GPS, IMU, road network matches, ...)

Conclusions

- The back-end part of the SLAM problem can be effectively solved with Gauss-Newton error minimization.
- Error functions compute the mismatch between the state and the observations.
- Currently one of the state-of-the-art solutions for SLAM.