On Use of the SLAM Toolbox:
A Fresh(er) look at Mapping and Localization for the Dynamic World

Steve Macenski, Samsung Research America
OVERVIEW

- Background and Motivating Problems
- SLAM Toolbox
- Use, Configuration, and Examples
- In the Wild and Roadmap
Background - What is SLAM? How does it look like to ROS (REP 105)?

**SLAM:** Simultaneous Localization and Mapping

Receive sensing from the environment
- Range Finders
- Odometry Sources (encoders, IMU, etc)
- Camera
- Radar

Create a model of the world for positioning
- Sometimes used navigate
- May be visualized as grids or feature maps

An REP-105 compliant implementation will provide the map -> odom transform, but often more.
Background - A 30,000 ft View*

GMapping (pf based)
- Presented in 2007 - year of the first iPhone
- Mapping large spaces is an “art”
- In my experience, diverges in >20,000 sqft

Karto (graph based)
- Excellent scan matcher
- Basis of several companies SLAM
- In my experience, an acceptable starting point

Cartographer (graph based)
- Localization, submap serialization, 3D support
- Discussions with Google haven’t inspired confidence
- In my experience, couldn’t reliably create usable maps
  - Thick hairy walls, missing simple loop closures, etc

* Look, I can’t talk about everything, we’ll be here all day!
4 Major Elements of Graph-based SLAM
- Sensor Matcher
- Pose-Graph
- Loop Closure Candidate Selector
- Graph Optimizer

Sensor matcher computes relative poses and confidences between candidate scans

Those are added as edges to a pose-graph

Loop closure candidates identified and correlated for new constraints

Optimizer tries to reduce error in the graph
Background - Quick Primer on Graph-based SLAM

4 Major Elements of Graph-based SLAM
- Sensor Matcher
- Pose-Graph
- Loop Closure Candidate Selector
- Graph Optimizer

Sensor matcher computes relative poses and confidences between candidate scans

Those are added as edges to a pose-graph

Loop closure candidates identified and correlated for new constraints

Optimizer tries to reduce error in the graph
SLAM Toolbox - An Overview

Map **Large and Dynamic Spaces**

- Reliably Map Spaces >>100,000 sqft Quickly
- Improved Ease of Mapping
- Rich Set of Toolbox Utilities
- Localization that Embraces Change
- Online and Offline Processing
- Continue Mapping from Prior Sessions Losslessly

Roughly 1 hour to map 45,000 sqft

Courtesy: SimbeRobotics.com
SLAM Toolbox - Mapping

Mapping Modes
- **Synchronous**: All measurement processing
  - May lag in large spaces if compute is limited
  - Offline: for highest quality
- **Asynchronous**: Best-effort processing
  - Online: navigating on limited compute

Ceres Plugin Optimizer

Serialization for Continued Mapping / Post-Processing

Map ~200,000 sqft Synchronously in Real-Time on Robot
- Much larger available in asynchronous & cloud processing

[Soon] Multi-robot distributed mapping
**SLAM Toolbox - Localization**

Takes Advantage of Data + Graph
- **Goal**: Localization as close to SLAM as possible
- Localize with map *and* current state in local horizon
  - New obstacles become features, not deviations!
- If no base map given, “local SLAM”

**Elastic Pose-Graph Deformation**
1. Fixed size buffer of new scans
2. Add new node/constraints to graph
3. Once expired from buffer
   - remove & free data / nodes
4. Pose-graph reverts to original state

Buffer size set by `scan_buffer_size` (10)
Continue Mapping = Refine or explore new spaces
Each mode can continue building a map
- Load graph, relocalize to a pose or keynode
- Match against local environment for a refined fit
- Find transform from current & serialized frames
- Continue refining map

Lifelong Mapping = Above + removal of extra nodes
* Cartographer’s Lifelong mapping is my continue mapping, it does not bound computation
[Experimental] A living, breathing map
- Online updates to map over many sessions
- Online removal of outdated/redundant data
- Currently computed by a (questionable) heuristic
I have pose-graphs and scans serialized, what can I do with it?

**Batteries Included:**
- Kinematic map merging
- Manual pose-graph manipulation
- Rotate maps without aliasing pixels
- Continue to build maps / localization

**Ideas:**
- Map rendering with different parameters
- Merge $N$ maps using graph structure / data
- Select & remove nodes/constraints
- Annotations
- Analyze odometric & sensor poses to model odometry drift
Configuration and Use

Select Parameters - See documentation for full list and descriptions

stack_size_to_use: Stack size, for serialization/deserialization only

tf_buffer_duration: TF buffer storage duration (offline speedups)

solver_plugin: Optimizer plugin to use

mode: Mapping or localization mode for optimizer settings

map_file_name: Base filepath of serialized map to load
map_start_{pose, at_dock}: Whether to start at a specific pose or dock

enable_interactive_mode: Enables interactive pose-graph manipulation
Examples

Sidewalk of San Francisco

Cars!
What’s Next?

The Work is Never Done

#76 - Multi-laser Robots

#68 - Multi-robot Cloud Processing Sample

Additional Tools for Processing Pose-Graph/Data
- Map merging method using graph structure
- Robot odometry drift estimation
- Selectively remove nodes/constraints

Promote “true” Lifelong Mapping Node to Production
In The Wild
Navigation2 Tiny Ticket Competition Winner - Shivang Patel

Write SLAM package from scratch

Use SLAM toolbox instead
Repository, Documentation, and Issue Tracker: https://github.com/SteveMacenski/slam_toolbox.git

ROS Wiki Page: http://wiki.ros.org/slam_toolbox

Thanks to Other Active Contributor: Ivona Jambrečić

Thanks to Circuit Launch & Andrew Etherington for their help in finishing this work with facilities & a LIDAR to test.