3D Collision Avoidance for Navigation in Unstructured Environments

Armin Hornung
Humanoid Robots Lab, University of Freiburg
May 5, 2011
Motivation

- Personal robots are to operate in arbitrary complex environments:
  - Unstructured, cluttered
  - Dynamic changes
- 3D environment model needed for motion planning & obstacle avoidance
- Efficient storage and map updates desirable
Octrees as 3D Environment Representation

- Tree-based, recursive subdivision of space into octants
- Volumetric representation, allocated as needed
- Multi-resolution
- Probabilistic model of *occupancy*
- Available in the OctoMap framework at [http://octomap.sf.net](http://octomap.sf.net)
OctoMap Improvements: Speed

Scan insertion more than 2x faster compared to v1.0!
Further OctoMap Improvements

- Flexible and fast access with iterators for tree traversal
- ROS interface `octomap_ros`
  - Conversions from most common ROS datatypes
- Scan integration for incremental 3D mapping in `octomap_server`
- OctoMap 1.0 released with `octomap_mapping` stack
  - Latest improvements in an experimental branch, will be merged into trunk for next 1.2 release
Arm Navigation with **collider**

- Collision map for tabletop manipulation
- Sensor fusion between laser and stereo
- Uses **octomap** as probabilistic map containing free, occupied, and unknown space
- Stores timestamps in octree nodes for clearing outdated voxels
- Constantly updates collision map with 3D raycasting in octomap
Video Collider
Collider: Results

- Properly handles dense stereo data with missing disparity and occlusions (self and shadow filtered)
- Sensor data integration fast enough to react to changes in the environment (~10 Hz from narrow stereo, 2.5 cm res.)
- Enables dynamic updates of the manipulation collision map and an a priori unbounded workspace
- Will be released in `arm_navigation_experimental` (e-turtle)
Navigation in Unstructured Environments

- Navigation stack uses 3D data, but plans for 3DoF base footprint in downprojected 2D

→ Extend navigation stack to use 3D world model and collision checks with full robot model

- Uses: Mobile manipulation, navigation with untucked arms, cluttered environments

[B. Steder] [Marder-Eppstein et al.]
The 3d_navigation Stack

- New global and local planners as drop-in replacements for the navigation stack
- `octomap_server` for incremental 3D mapping and as replacement for `voxel_grid`
- Collision check in `planning_environment` (ODE)
- Textured narrow stereo as sensor
  - Too much trouble with Kinect when moving
  - Possible to integrate laser(s) as well in the future
- Simulation in `3d_nav_gazebo`
Global Planner: **sbpl_lattice_planner_3d**

- Based on **sbpl_lattice_planner**
- Search-based planning with motion primitives
- Exploits holonomic capabilities of the base
- Fast collision check of robot footprint in downprojected 2D map
- 3D collision check of robot's current kinematic configuration in OctoMap only when 2D check fails
- ARA* enables a feasible solution in a reasonable time
SBPL – Search-based Planning Library

- Applies motion primitives at each search expansion to construct lattice graph in \((x,y,\theta)\)
- Collision check determines costs of expanded nodes
- Graph search on lattice graph yields solution as a sequence of collision-free poses \((x,y,\theta)\)

[M. Likhachev]
Local Planner: pose_follower_3d

- Based on pose_follower
- Necessary to use originally planned sideways and backwards motion
- Single trajectory rollout to next pose on the path
  - Fast collision check of footprint in 2D
  - 3D check against constantly updated map
- Head monitoring ensures dynamic update of collision map in movement direction
Video 3D Navigation
Results

- Optimal results for short paths or in free areas
- Reasonable planning results in 15 s planning horizon, room-sized cluttered environment (*Green Room*)
- PR2 reliably docks and undocks tables with untucked arms
- Local planner avoids obstacles during plan execution
  - 2-4 Hz map update, 0.4s base controller timeout
Conclusions

- Improvements to OctoMap
- Arm navigation in manipulation to be more reactive with dynamically updated collision map in `collider`
- `3d_navigation` augments the navigation stack for navigation in unstructured 3D environments without requiring tucked arms
Thank you!