On Use of the Spatio-Temporal Voxel Layer:
A Fresh(er) look at 3D Perception for the Planar World

Steve Macenski, Simbe Robotics
OVERVIEW

- Background and Motivating Problems
- Spatio-Temporal Voxel Layer
- Use, Configuration, and Examples
- In the Wild and Roadmap
Background - Navigation and Costmaps

Inputs:
- Desired Goal Pose
- Current Position (AMCL, SLAM, etc)
- Odometry & Sensor Sources
- Map

Move Base:
- Global Costmap
- Local Costmap
- Behaviors
- Global Planner
- Local Planner

Outputs:
- Robot Velocity Command
Background - Navigation and Costmaps
### The Voxel Layer

- Costly Raycasting - not pragmatic for many long distance sensors
- Limited to 16 binned heights - limited representations for tall or large robots
- Dynamic obstacles can leave trails - requires suboptimal work arounds
- Not all modern sensors are dense integral images - use cases have expanded
- No temporal clearing - maintaining data from potentially weeks ago
- High CPU load for multiple sensors - untenable for full and/or redundant coverage
Background - A(nother) Motivating Example

Chasing WALL•E. Followed him for 3 aisles. @simberobotics

Tally made some new friends in this @SchnuckMarkets store over the weekend. Thank you, @StandefordL for sharing and introducing the next generation to our friendly robot!
Spatio-Temporal Voxel Layer (STVL)

General Purpose Voxel Grid and Costmap2D Layer

Use: depth cameras, VLP-16, RADAR, and more

Fast access to voxels and manipulation with OpenVDB

Temporal clearing and configurable acceleration models

No assumptions on a static environment or map size

No maximum number of voxel height constraint

Used with 10+ depth cameras at once
STVL - OpenVDB Basics

DreamWorks Animation

Used in *How to Train Your Dragon* and 70+ others

Low memory overhead - highly optimized octree

O(1) voxel query/access

Contains structure and tools to manipulate voxel grids

Fast, elegant tools adequate for soft real-time robotics

... and so much more

Traditional frustum modelled as bounding planes

Given: FOV and min / max reading distance
Find: relative pose of a point to each plane
→ If on correct side of all 6, interior to frustum

Acceleration modelled as linear / exponential decay

The “Aisle Problem”:
Adjacent aisles’ voxels may be accelerated but not viewable
Proposition: This is not undesirable in dynamic environments, the state of that space unknowable to an agent.
→ Will likely be cleared anyhow from global decay before returning.
Configuration and Use

Key Parameters

Layer Parameters:

- Voxel Decay: Default: 0 s
- Decay Model: None, Linear, Exponential
- Observation Persistence: Default: 0 s
- Publish Voxel Map: Default: False
- Mapping Mode: Default: False

Observation Parameters:

- Voxel Filter: Default: False
- Min / Max Z: Default: 0 m
- Vertical / Horizontal FOV: Default: 0 rad
- Decay Acceleration: Default: 0 1/s2

Several Preconfigured Profiles available in documentation
Configuration and Use

Other Notes

Velocity scaling:
  The faster you move, the more data you want to store
  No hard boundary like existing layer implementations

Local Costmap:
  Recommend faster decay 1-15 seconds with good coverage
  Covers smaller area and runs faster

Global Costmap:
  Recommend slower decay 5-45 seconds with good coverage
  Covers larger area and runs slower

Want Only The Most Recent Measurement?
  Non-Persistent Voxel Layer may be for you

https://github.com/SteveMacenski/nonpersistent_voxel_layer
Examples

Temporal clearing band gaps - raycasting ineffective

Frustum clearing with linear acceleration model
Brief Incursion into Mapping...

**Hijacking the costmap layer**

Naively recorded voxels as seen from sensors

Trivial to 3D map a 60,000 sq.ft. environment

Less than 7 MB on disk

Convenience methods provided to transfer OpenVDB serialization types to ROS-y types

No assumptions made about a static environment, Frustum clearing removed dynamic obstacles
Not intended use-case
What’s Next? (Call for Action)

**Navigation**
Support 3D LIDAR frustum acceleration models

Split and Merge OpenVDB trees for parallelizable sensor processing

Already iterating through local grid, let’s use it:
  - Improved spatial reasoning using CCA
  - Integrated 3D blob dynamic obstacle tracking / response

**Mapping**
Standalone node + Binary Bayes Filter = Octomap-like 3D mapping

Using as the engine for an integrated 3D SLAM solution
In the Wild

10,000+ hours in collision free operations
In dynamic retail environments (with kids hugging it)
Processing 9 or more sensors

Personal Robotics on a budget :)

Repository, Documentation, and Issue Tracker:
(Current)
https://github.com/SteveMacenski/spatio_temporal_voxel_layer
(Soon to be)
https://github.com/SimbeRobotics/spatio_temporal_voxel_layer

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

We’re Hiring!
https://jobs.lever.co/simberobotics.com

Thanks to Other Active Contributor David Tsai

Steve Macenski
Senior Robotics Engineer
steve@simberobotics.com
stevenmacenski@gmail.com