Humanoid Robotics

Perception 2: 3D World Representations

Maren Bennewitz

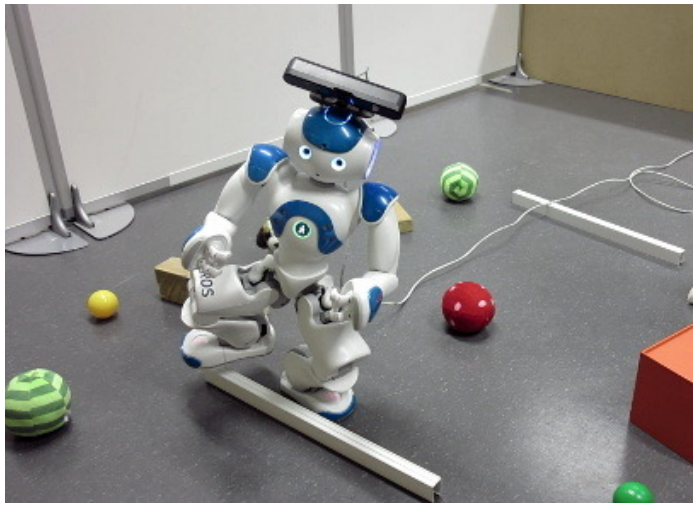
UNIVERSITÄT BONN

Goal of This Chapter

- Overview of different world representations with their strengths and weaknesses
- Understand how they can be generated from sensor data
- Understand which representation is useful for which robotic application

Robots in 3D Environments

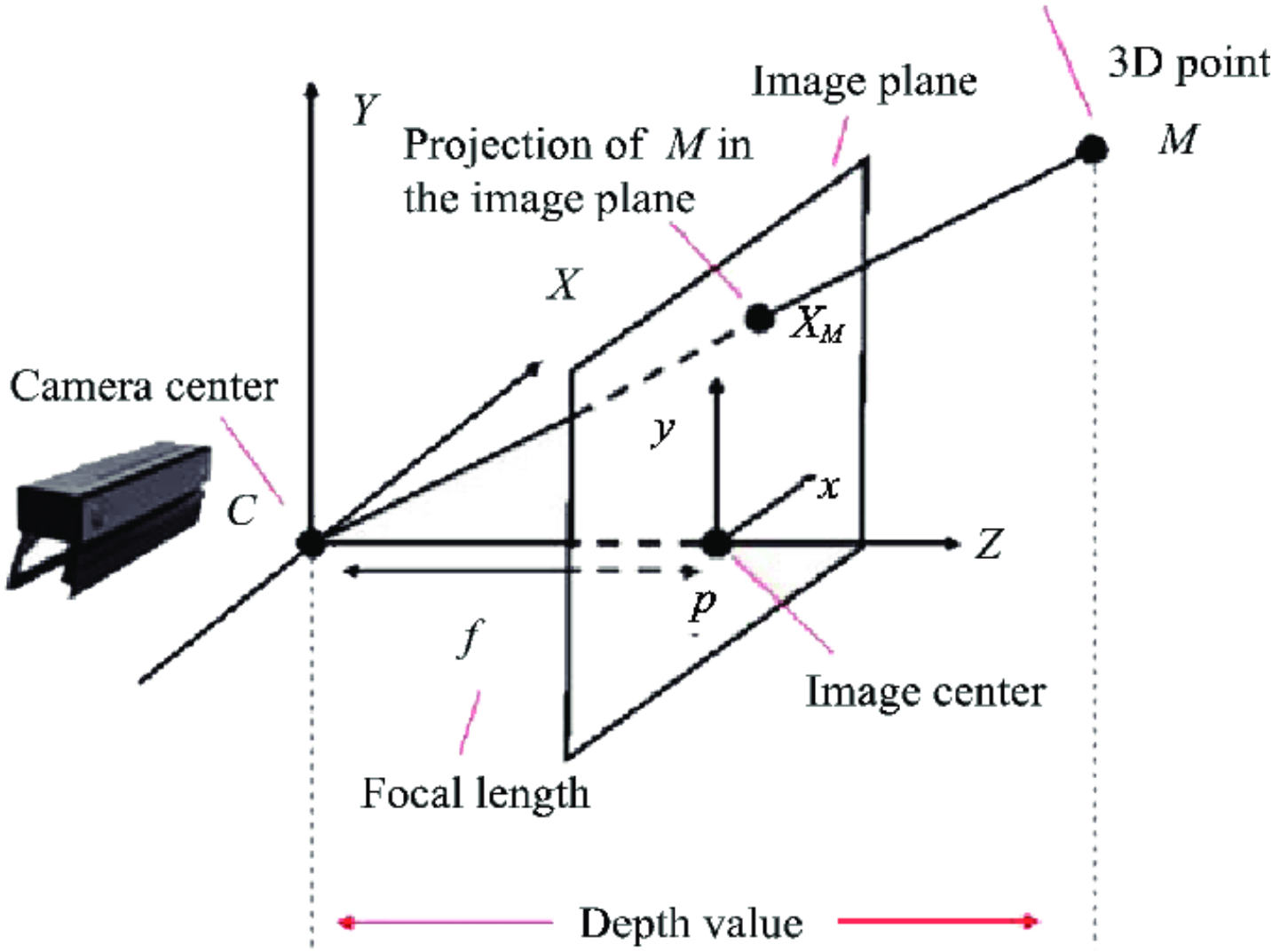
source: Honda



Motivation

- Robots live in the 3D world
- Collision avoidance, motion planning, grasping, etc. require adequate 3D models
- Given: RGB-D or 3D point cloud data acquired with the robot’s sensors
- Question: How to represent the 3D structure of the environment?

Basics: Depth Images and Point Clouds



[H. Xu, J. Xu, and W. Xu, Survey of 3D modeling using depth cameras, 2019]

Basics: From Depth Images to Point Clouds

- Depth image point X_m with coordinates (u, v) represents the distance z from the camera center C to the point M along the optical axis
- $P = (c_x, c_y)$: projection of the camera center C onto the image plane
- $f_x, f_y \rightarrow$ focal length along x and y axes (similar if pixels are scaled identically along both axes)
- $z = D(u, v)$, $x = z \cdot \frac{u - c_x}{f_x}, y = z \cdot \frac{v - c_y}{f_y}$, $M = [x, y, z]$ 3D point

Basics: Aligning Depth and RGB Images

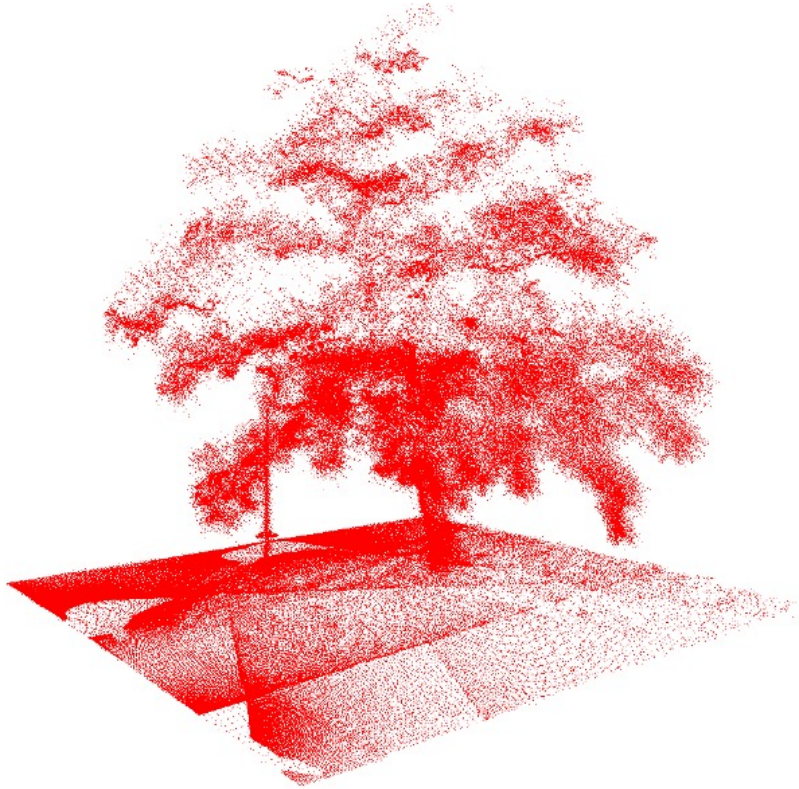
- Example: RGB 1920×1080, depth 640×480, unsynced sensors
- Undistort both images with respective parameters
- Convert depth (640×480) to 3D points
- Project into 1920×1080 RGB image plane
- Interpolate depth to RGB image using nearest values

Popular Representations of the 3D World

- Point clouds
- Voxel grids
- Height maps
- Distance fields
- Meshes
- Semantic-aware volumetric maps
- …

Point Clouds

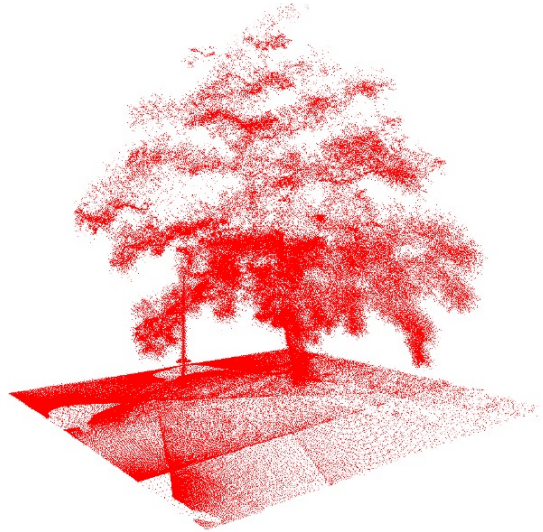
- Set of 3D data points in world frame



Point Clouds

Pros

- No discretization of data
- Mapped area not limited



Cons

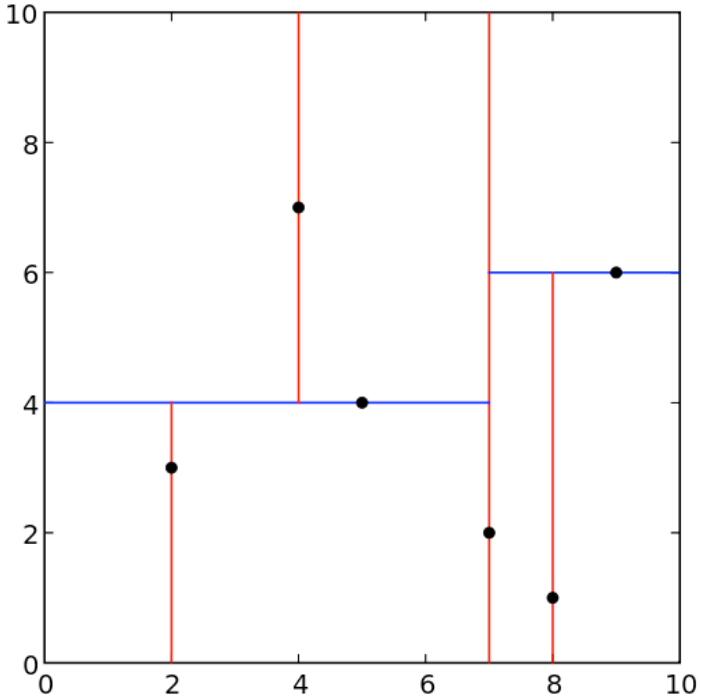
- Unbounded memory usage
- No constant time access for location queries (in the naïve implementation)
- No distinction between free or unknown space

Point Clouds and Efficient Location Queries

- Naïve implementation (list, array) has linear complexity for location queries
- More effective solutions through kd-trees
- **kd-trees** operate in k-dimensions
- Space-partitioning data structure for organizing k-dimensional points
- Search/insert/delete in **logarithmic** time on average

Example: kd-Tree (2-dimensional)

Binary space partitioning



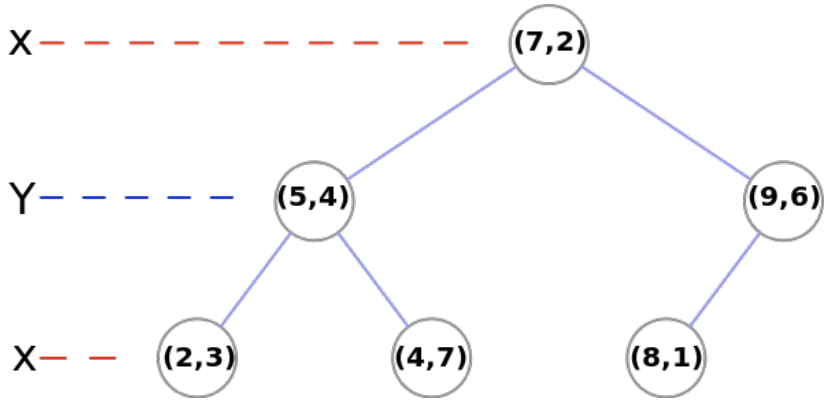
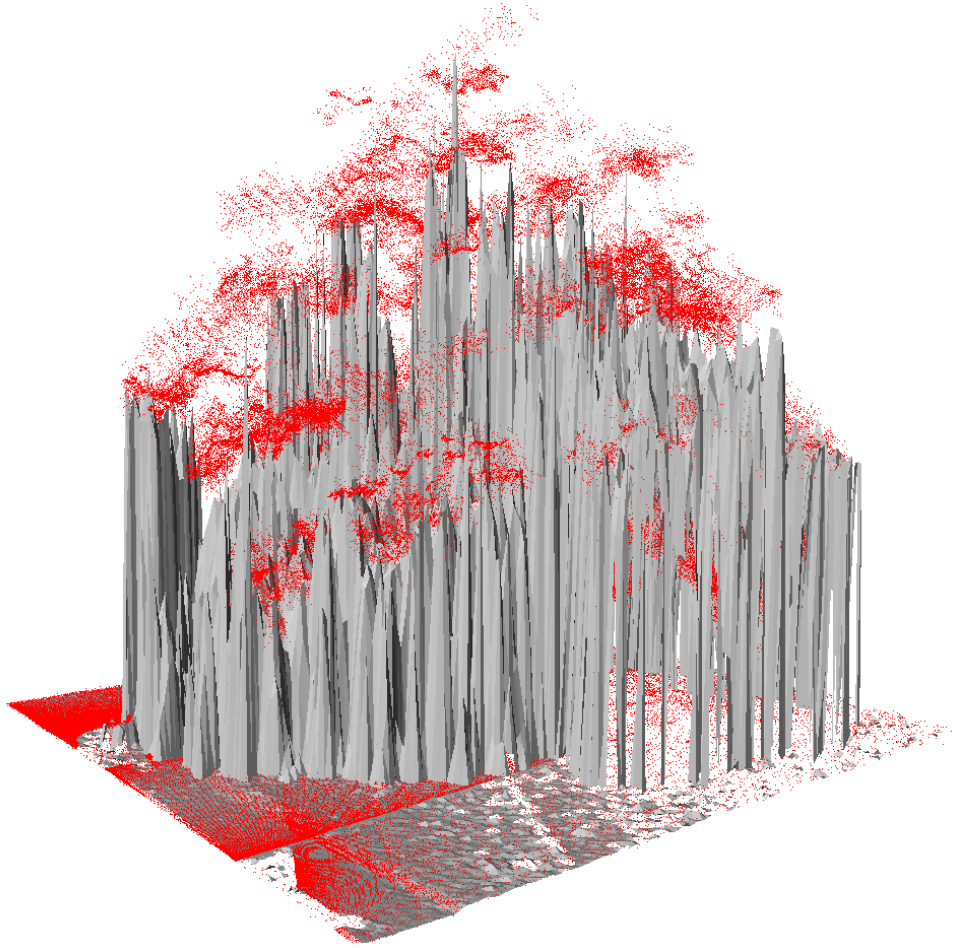


Image courtesy: Wikipedia

2.5D Maps: Height Maps

Average over all points that fall into a 2D cell and consider this as the height value



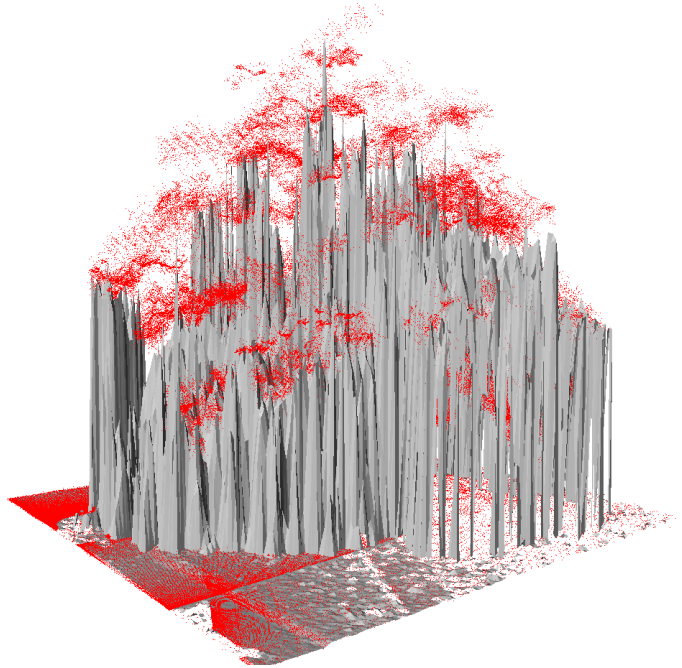
2.5D Maps: Height Maps

Pros

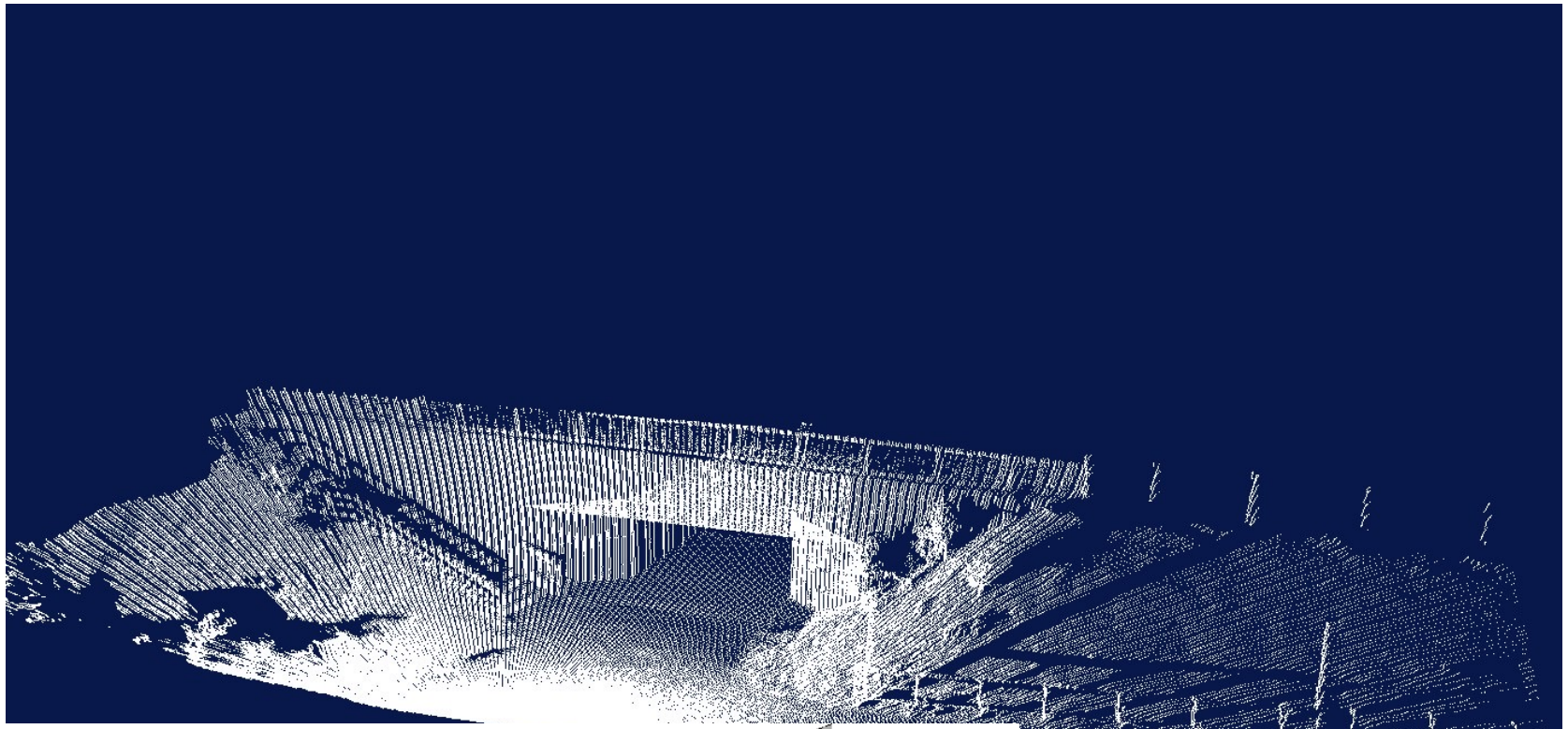
- Memory efficient (2D)
- Constant time access

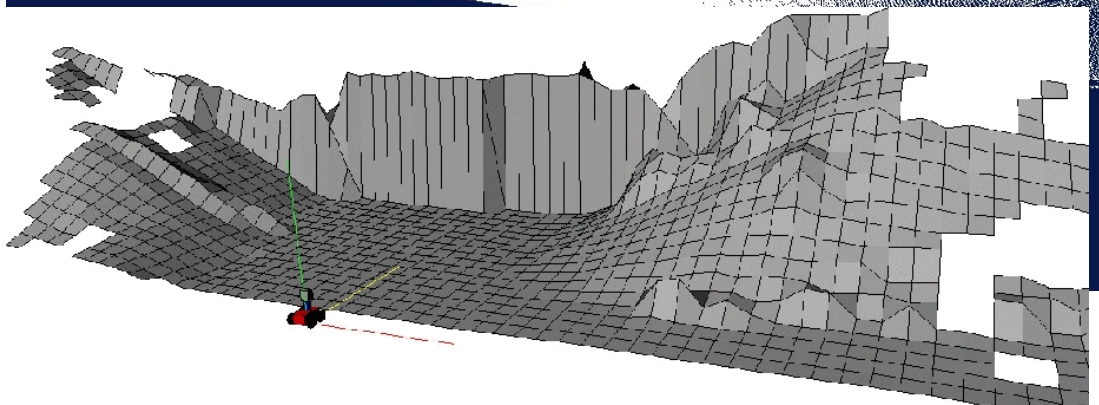
Cons

- No vertical objects
- Only one level is represented

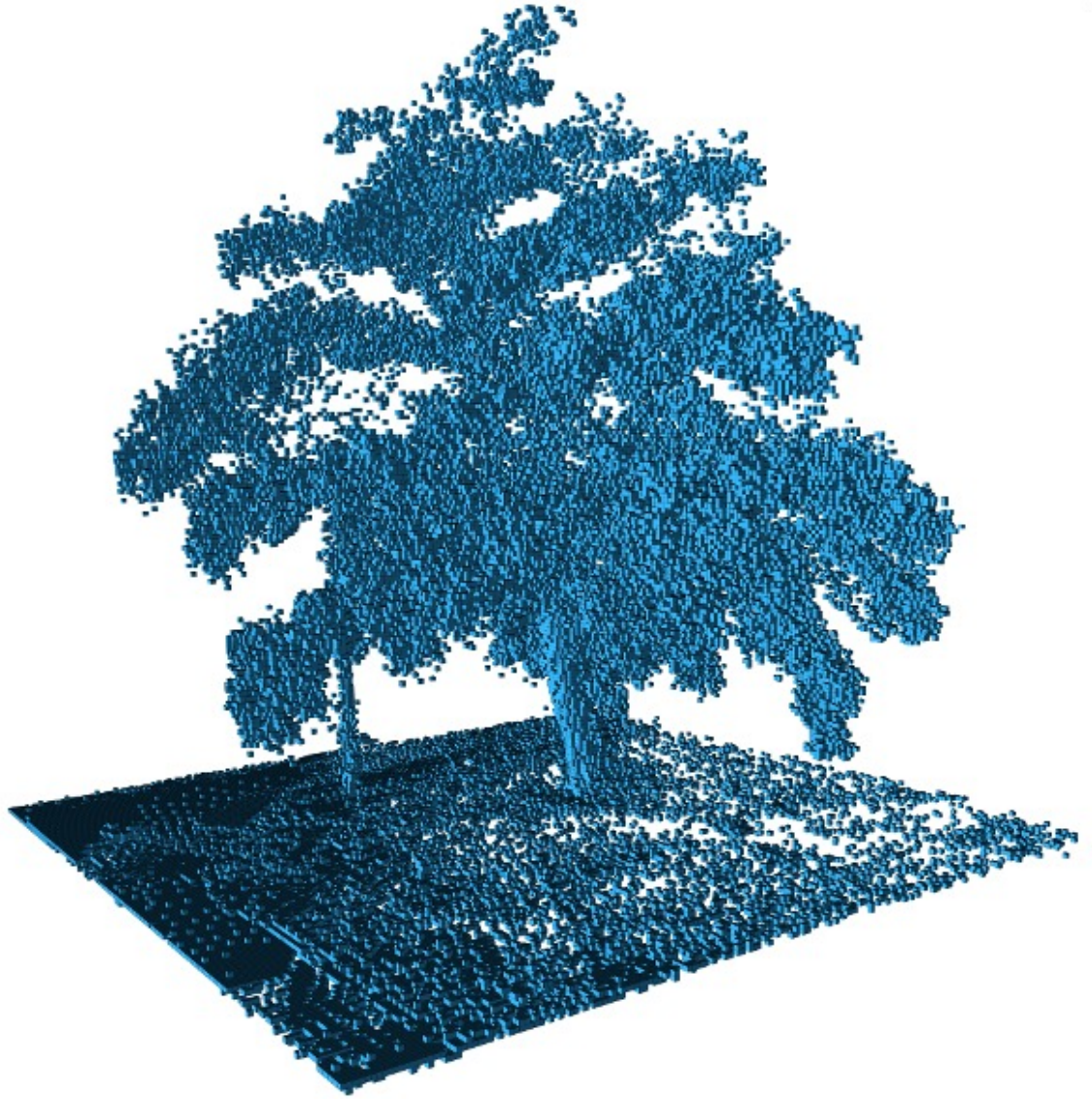


Problem of Height Maps: Just 1 Height Value





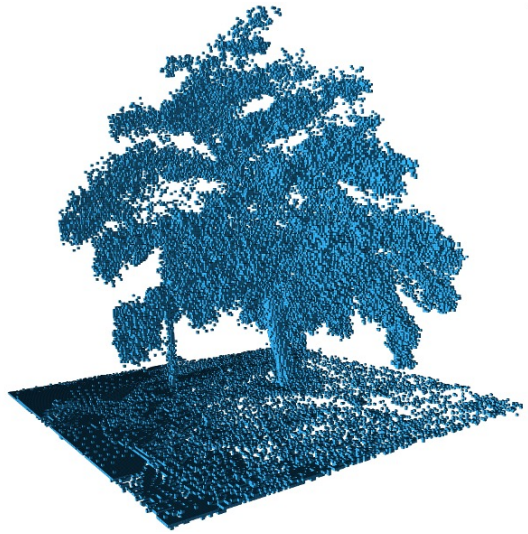
3D Voxel Grids



3D Voxel Grids

Pros

- Volumetric representation
- Constant access time
- Probabilistic update possible (Bayes' filter, Cognitive Robotics)

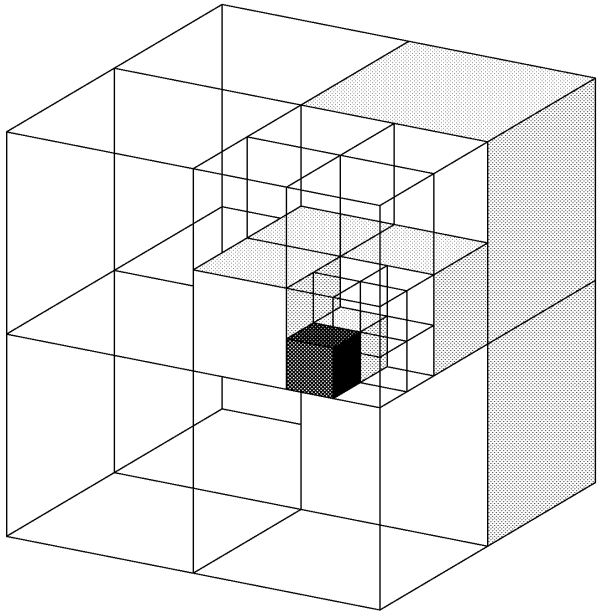


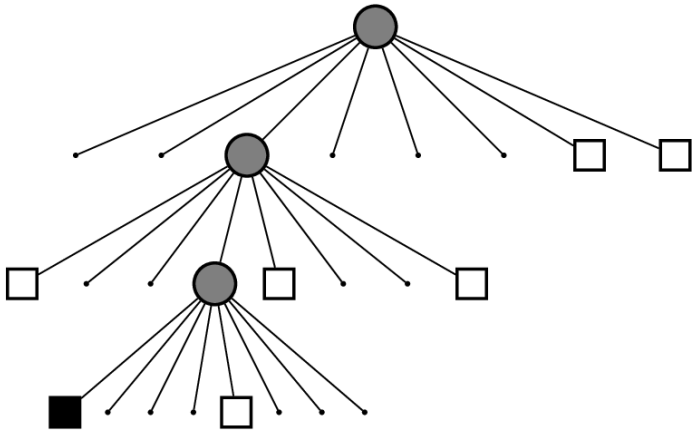
Cons

- Memory requirement: Complete grid allocated in memory
- Extent of the map has to be known/guessed
- Discretization errors

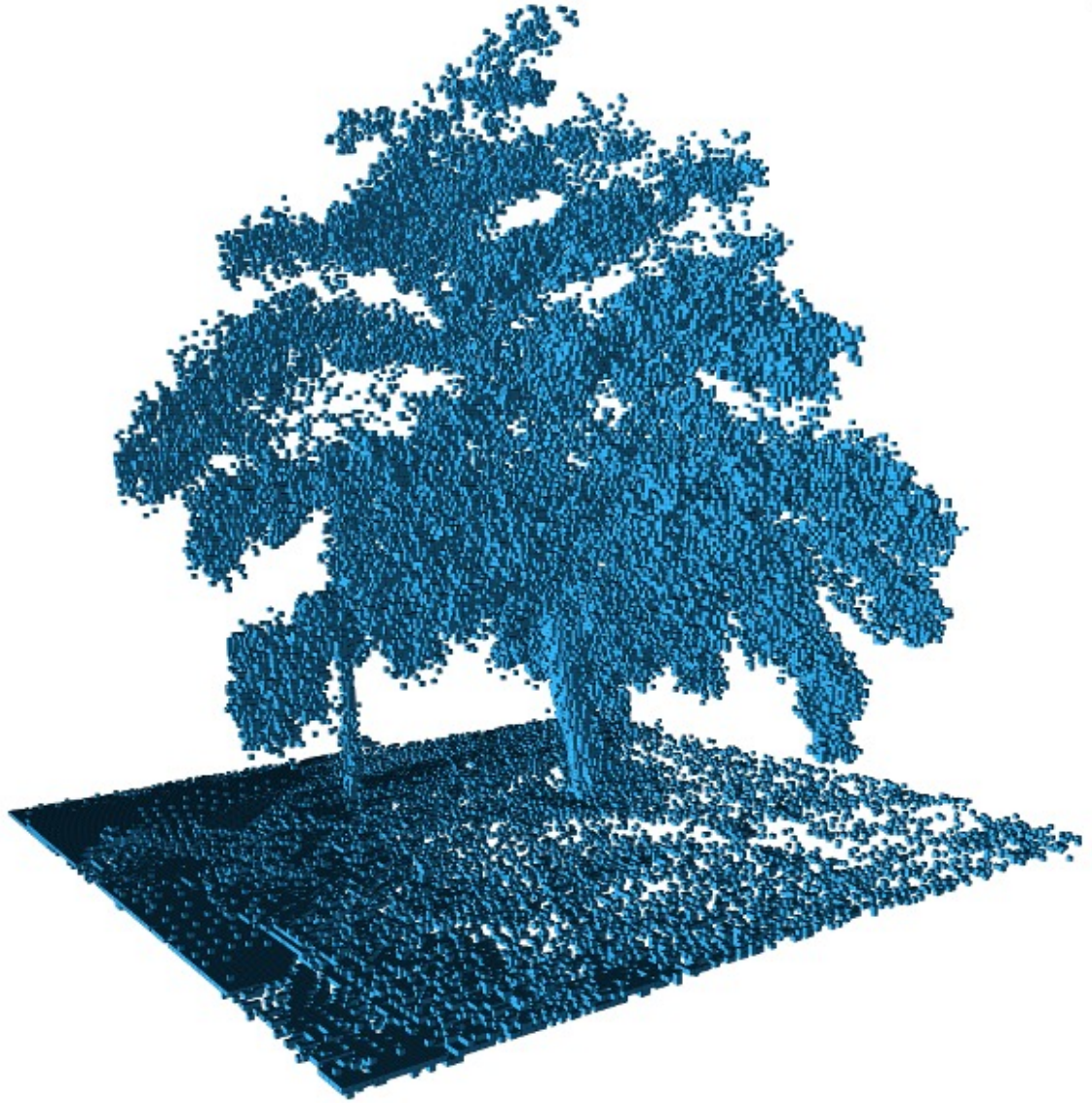
Octree-Based Representation

- Tree-based data structure
- Recursive subdivision of the space into octants
- Volumes allocated as needed
- “Smart” 3D grid

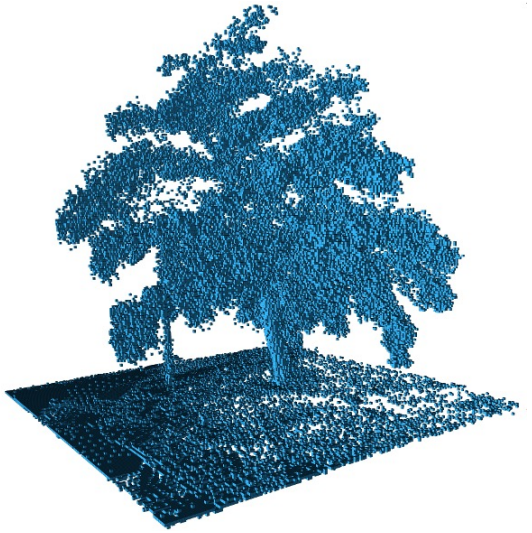




Octrees



Octrees



Pros

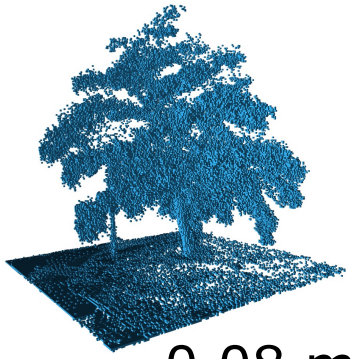
- Full 3D model
- Inherently multi-resolution
- Memory-efficient,
  volumes only allocated as needed
- Probabilistic update possible (Bayes' filter, see Cognitive Robotics/Intro in MoRo lecture)

Cons

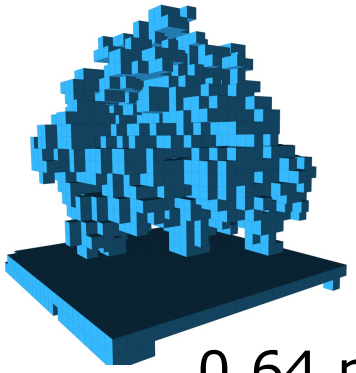
- Efficient implementation more tricky
  (memory allocation, update, map files, …)

Multi-Resolution Queries

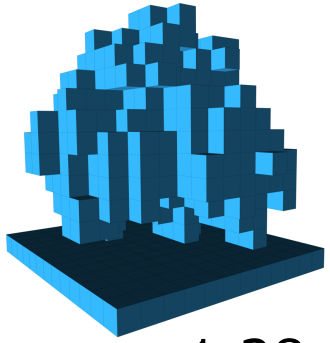
$$P(m_i) = \max_{j=1\ldots8} P(m_{i_j}) \text{ with } m_{i_j} \in \text{children}(m_i)$$



0.08 m



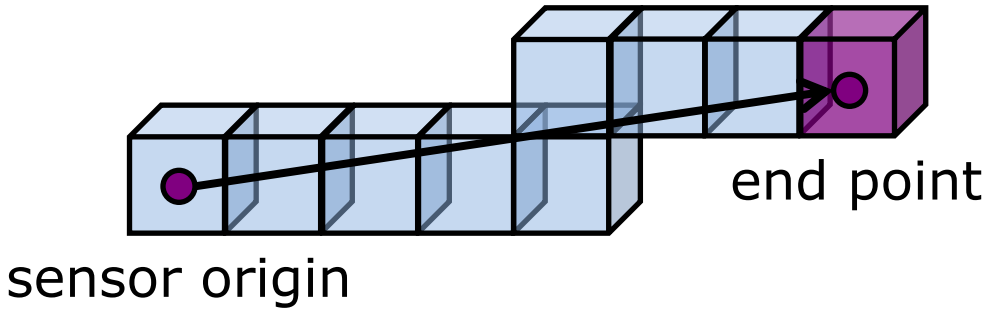
0.64 m



1.28 m

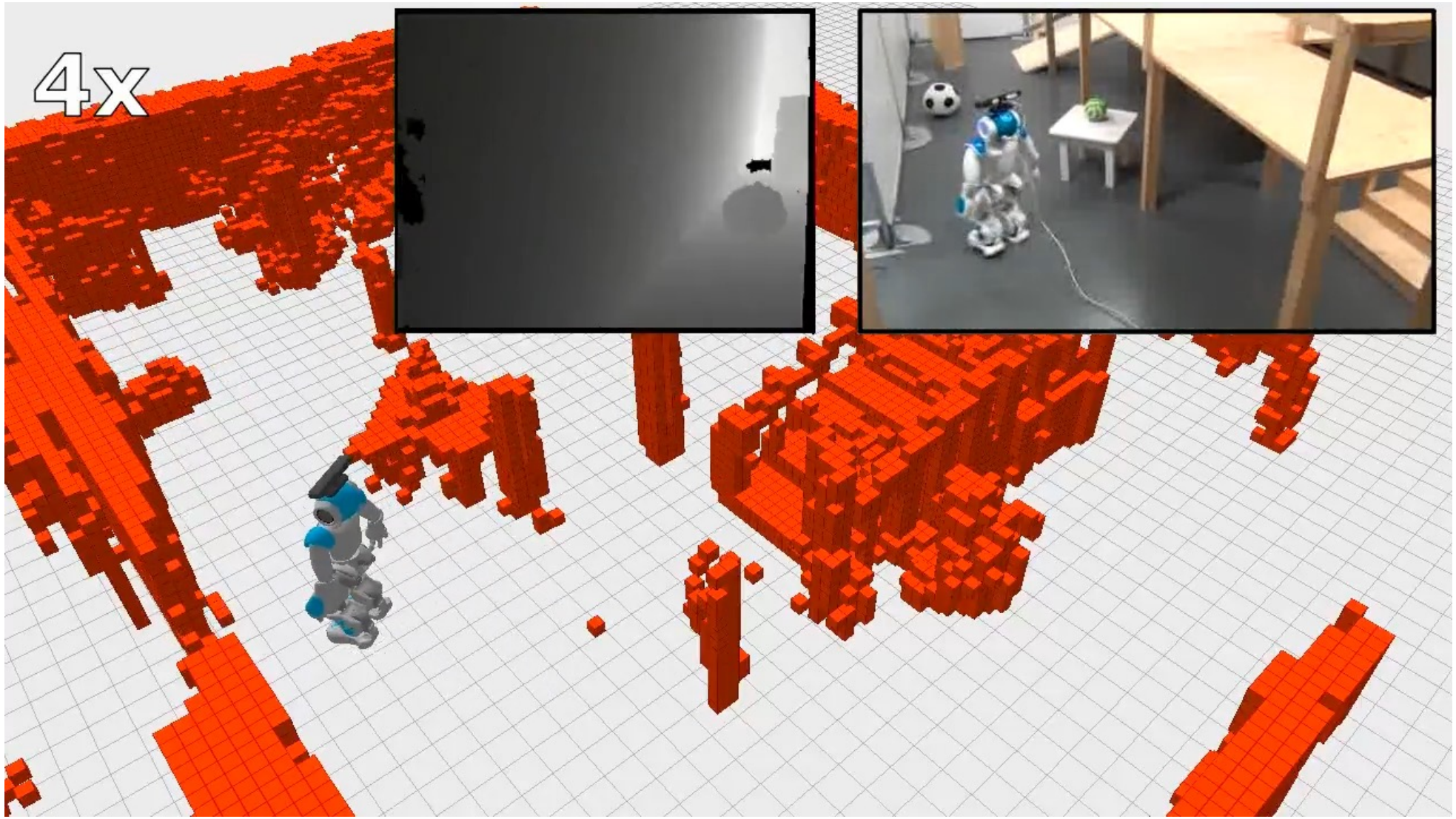
Map Updates via Ray Casting

- Ray casting from sensor origin to end point in the map along the beam
- Mark last voxel as occupied, all other voxels on ray as free
- Measurements are integrated probabilistically given the robot's pose (recursive binary Bayes' filter)



[Lecture on Cognitive Robotics/Intro into MoRo]

Online Mapping With Octomap



[D. Maier, A. Hornung and M. Bennewitz, Humanoids 2012]

Signed Distance Function

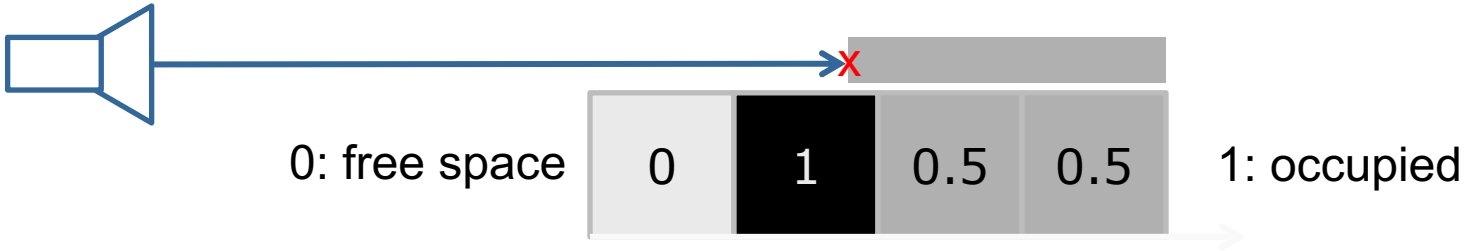
Signed Distance Function (SDF)

Key idea:

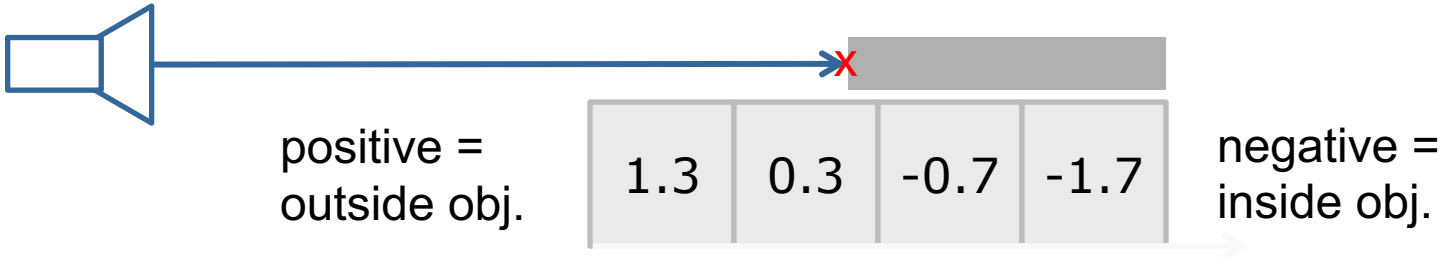
- Instead of representing occupancy values, represent the **distance of each voxel** to the **nearest measured surface**
- Surface can be extracted afterwards at **sub-voxel** accuracy

Signed Distance Function (SDF)

- Grid maps: explicit representation

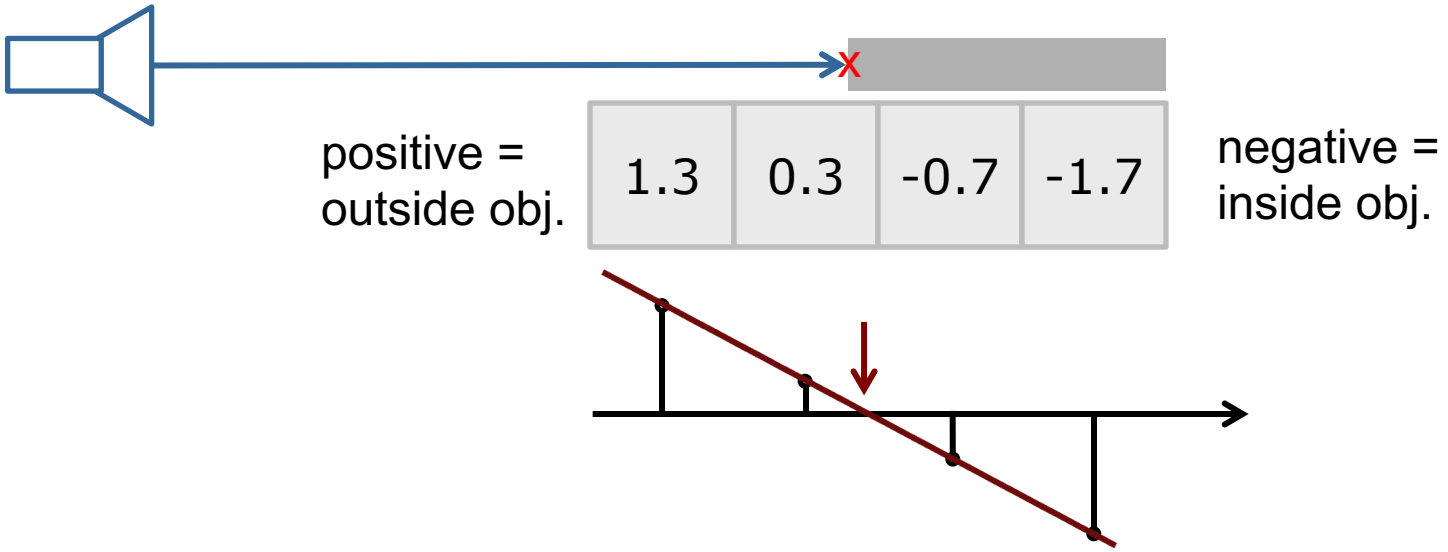


- SDF: implicit representation



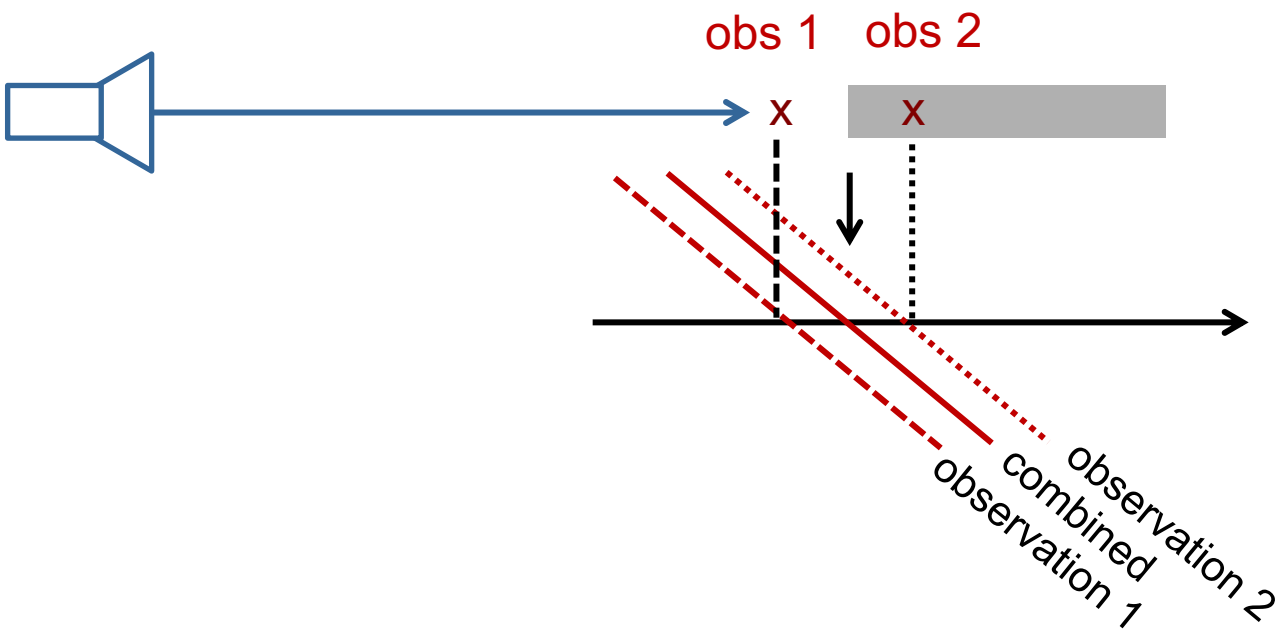
SDF Approach

- Compute the signed distance values
- Extract the surface using interpolation
- Surface is located at the zero-crossing



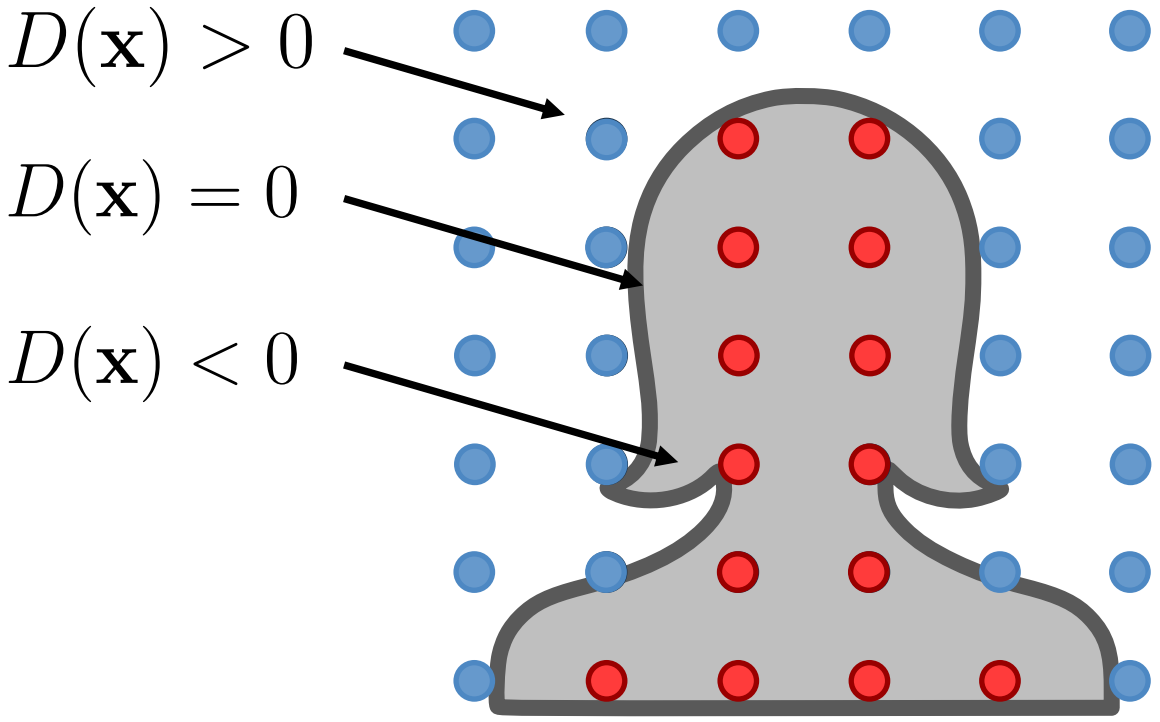
Properties

- Noise cancels out over multiple measurements



- Zero-crossing can be extracted at sub-voxel accuracy

Voxel Grid to Store SDF in 3D

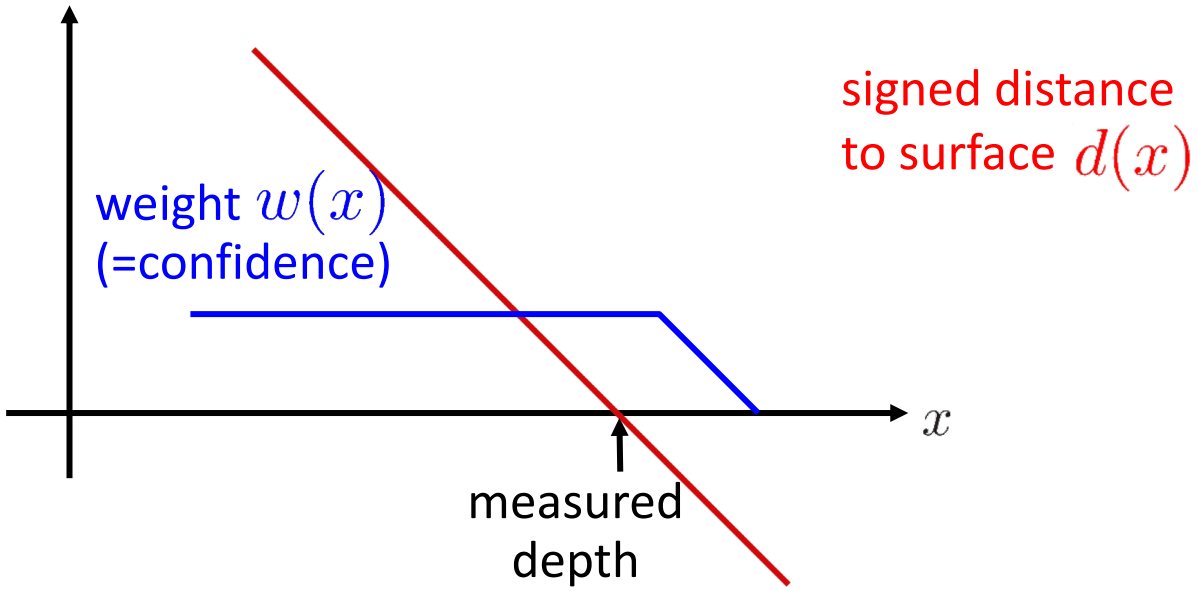


- Positive signed distance (=outside)
- Negative signed distance (=inside)

in general, there are several measurements for the voxels

Weighting Function for Multiple Measurements

- For each **voxel along the beam**, weigh the observation according to its **confidence**

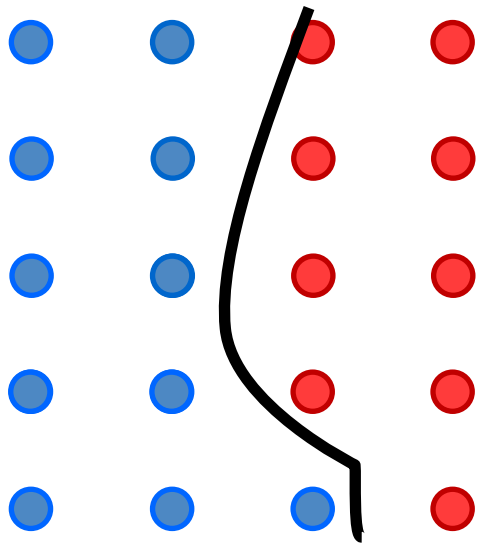


- Small weights ensure that values can be updated when new observations are available

Updating SDF

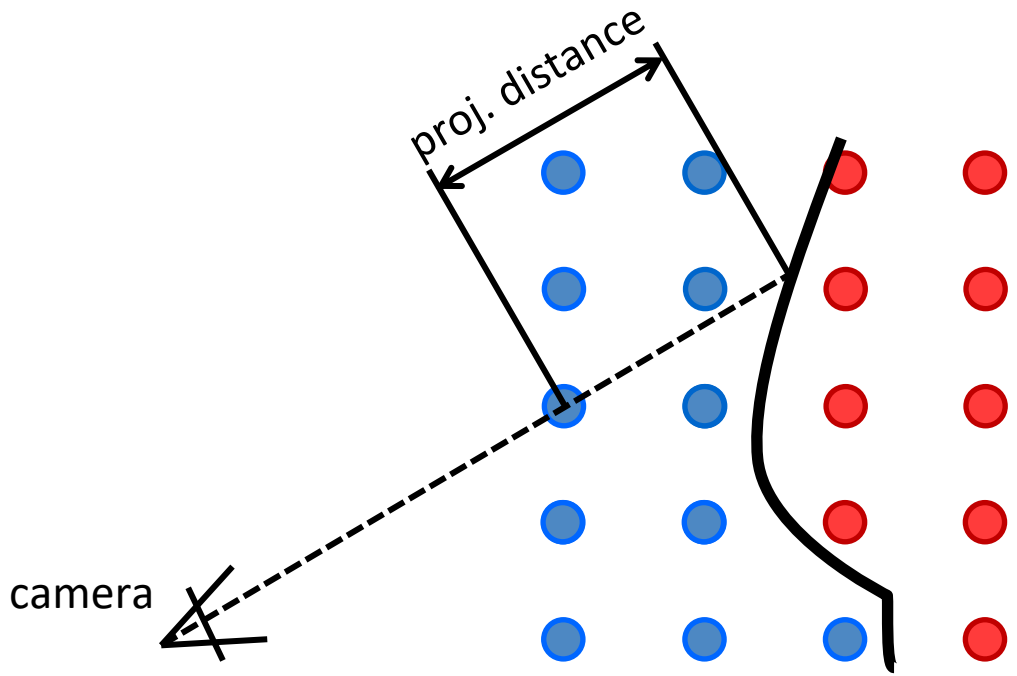
For each voxel along the measurement beam, use

- Distance to the next surface D
- Weight W



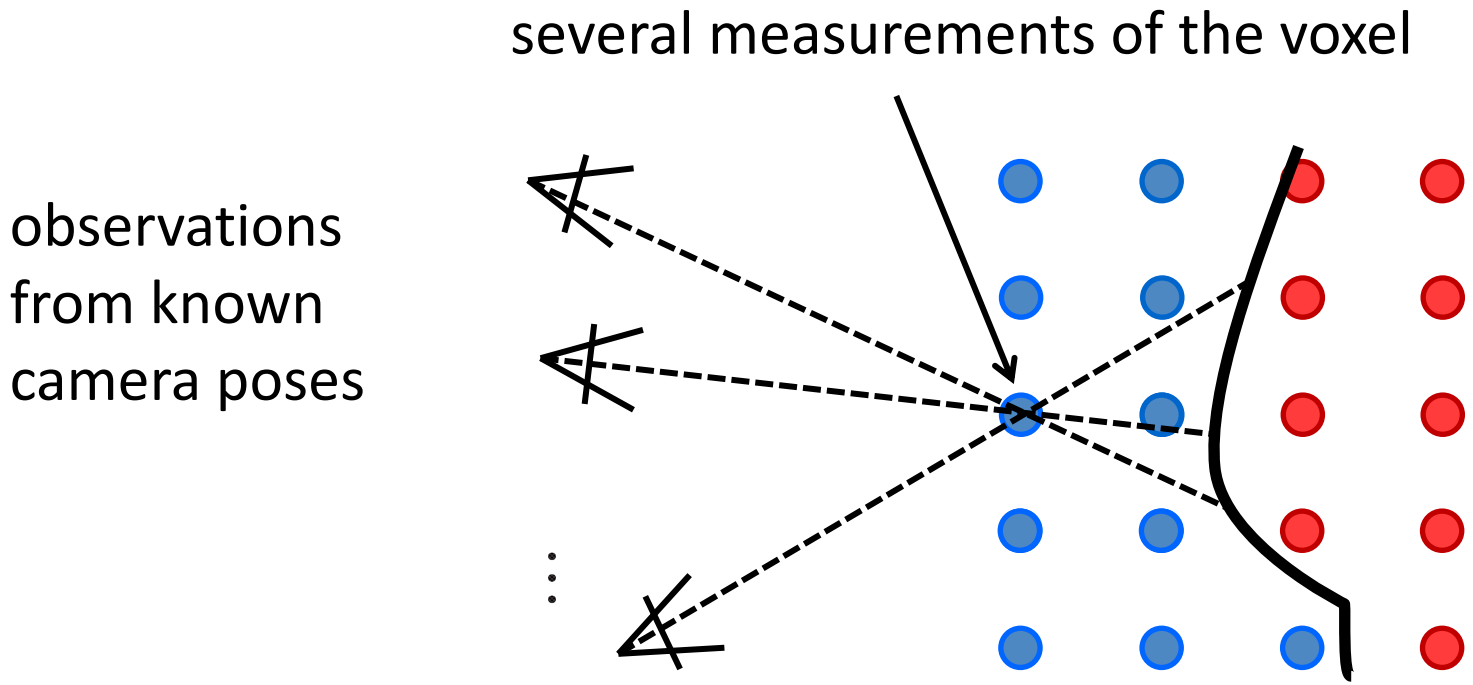
Truncated SDF

- Compute the **distance** of the voxels to the observed surface **along the beam**
- For efficiency: Update only a **small region around the endpoint** (truncation)



Weighted Update

For each voxel, calculate the weighted average over all its measurements



Weighted Average

- For each voxel, store two values
  - Weighted sum of signed distances D
  - Sum of all weights W
- When new data arrive, update the values of each voxel according to

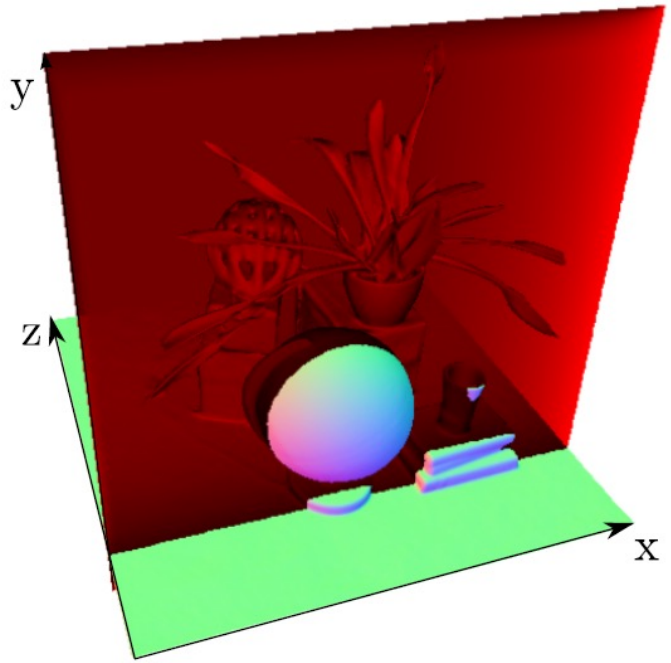
$$D \leftarrow \frac{WD + w_t d_t}{W + w_t}$$

$$W \leftarrow W + w_t$$

incremental computation of the weighted mean

SDF Example

Cross section through 3D signed distance function



Surface with cross-section

brightness encodes distance

SDF

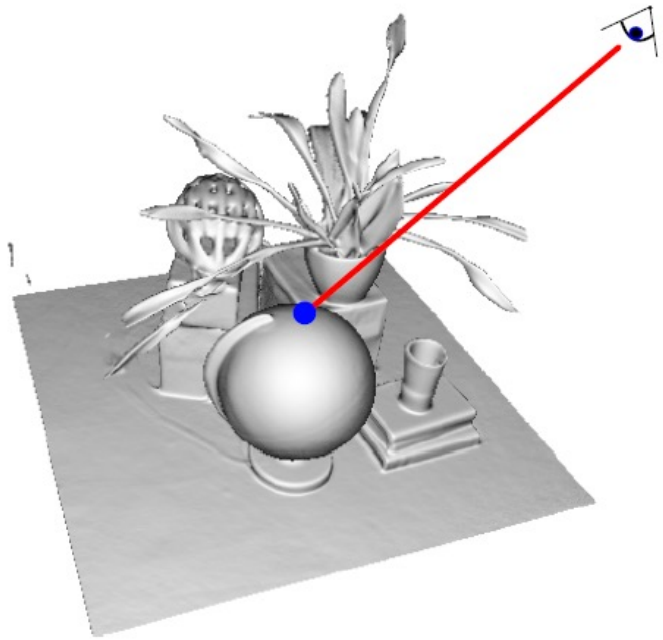
Surface Rendering

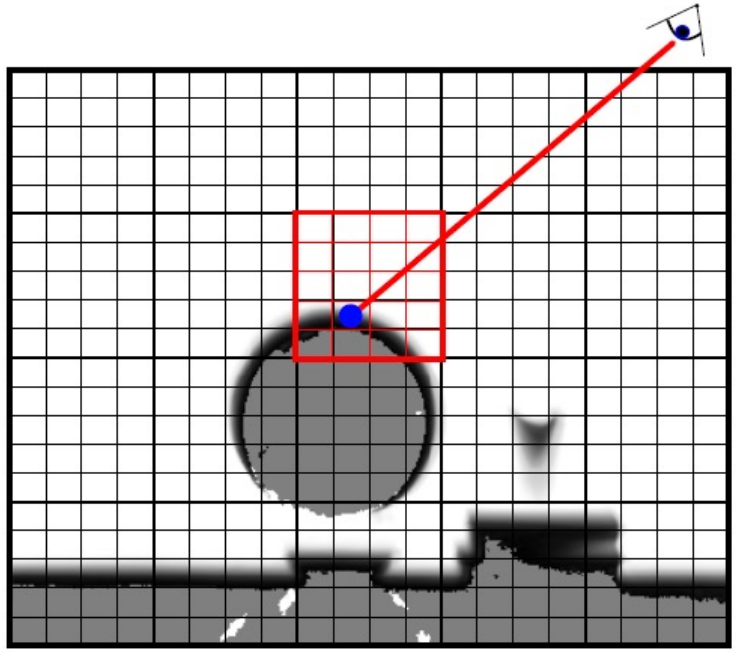
- **Ray casting**: For each camera pixel, shoot a ray and search for the zero crossing (viewpoint dependent)

- **Polygonization**: Marching cubes algorithm to generate a triangle mesh (more complex, yields explicit surface structure)

Ray Casting

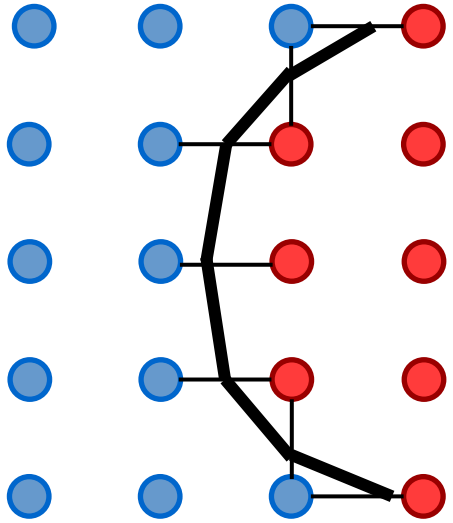
- For each pixel, shoot a ray and search for the first zero crossing in the SDF
- Value in the SDF used to skip along the ray when far from surface



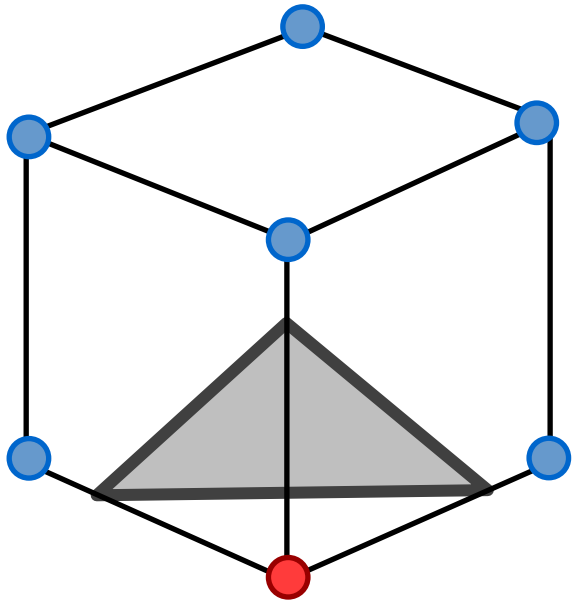


Mesh Extraction Using Marching Cubes

- Process the whole grid
- Find zero-crossings in the signed distance function by interpolation



2D



3D

Signed Distance Functions

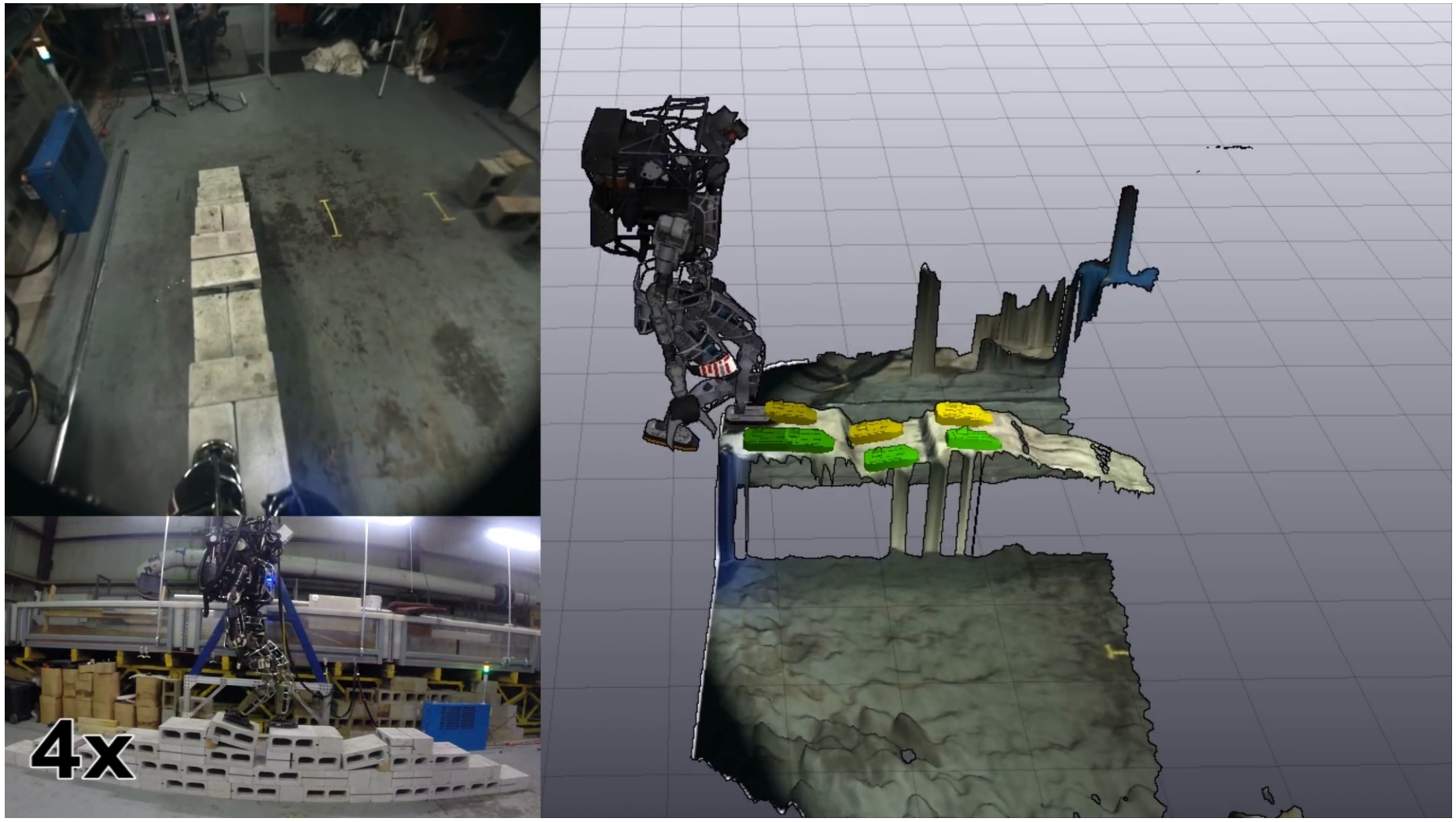
Pros

- Full 3D model
- Sub-voxel accuracy
- Supports fast GPU implementations

Cons

- Space-consuming voxel grid
- Polygonization: slow

Application: Estimation of Traversable Terrain using TSDF



[Fallon, Deits, Whelan, Antone, McDonald, and Tedrake, Humanoids 2015]

Multi-Layer Mapping

Multi-Layer Mapping

- All 3D world representations studied so far have geometric features
  - Occupancy probability
  - Weight and signed distance field
- However, scene understanding needs further features
  - Semantics, i.e., class labels
  - Instance ID
  - Uncertainty/Confidence

Semantic-Aware Volumetric Mapping

Key idea:

- Combine semantic scene understanding with geometric information to produce semantic-aware volumetric maps
- Semantics about objects/environment enable better understanding and long-term interaction

2D to 3D Projection of Semantics

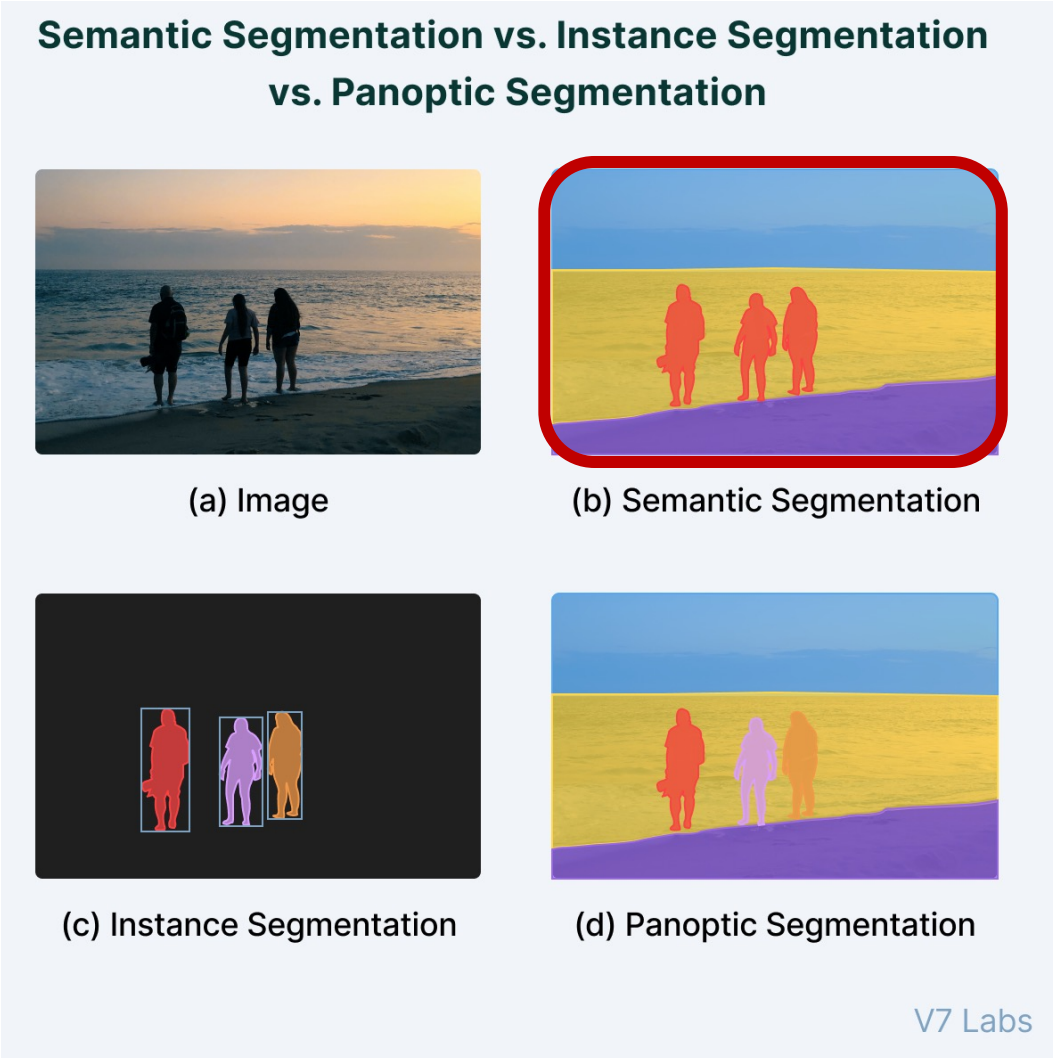
- Typically, metric-semantic maps have 2 separate layers:
  - Metric: Occupancy, TSDF
  - Semantics: Class label, instance label

- Multi-view integration of semantic labels
  - Majority voting: Label with highest votes for a voxel wins
  - Recursive Bayesian update: Requires probabilistic Bayesian neural networks for semantic prediction

Deep Learning-Based Scene Understanding

Semantic Segmentation

- **Pixel-level class labeling**
- No explicit object detection
- Different instances of same class grouped together
- No foreground/background differentiation
- Application: Geo-sensing, autonomous driving

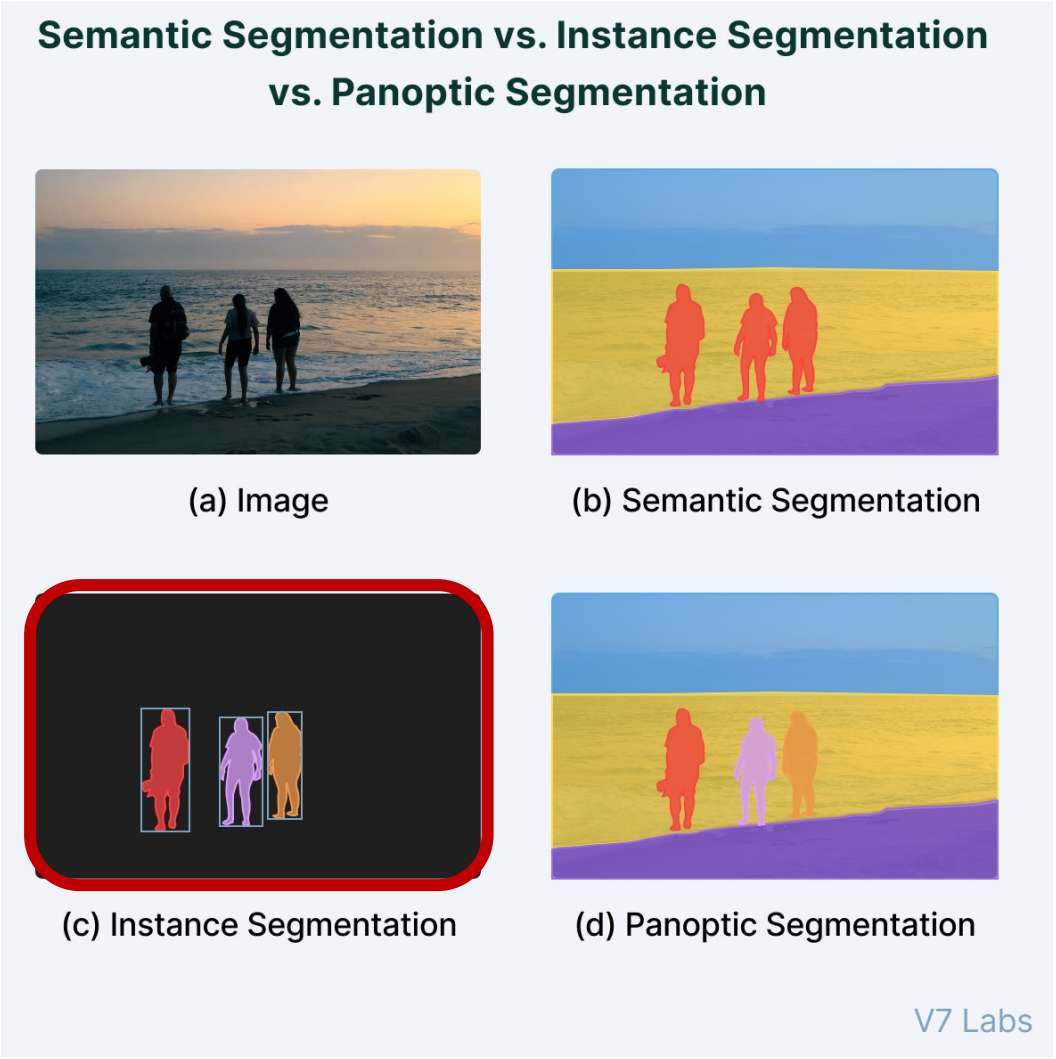


https://www.v7labs.com/blog/panoptic-segmentation-guide

Deep Learning-Based Scene Understanding

Instance Segmentation

- Hybrid of **object detection** and **semantic segmentation**
- **Unique instance ID** for two different objects of same class
- Mostly used for countable foreground objects, e.g., humans, objects
- Application: Grasping, object tracking

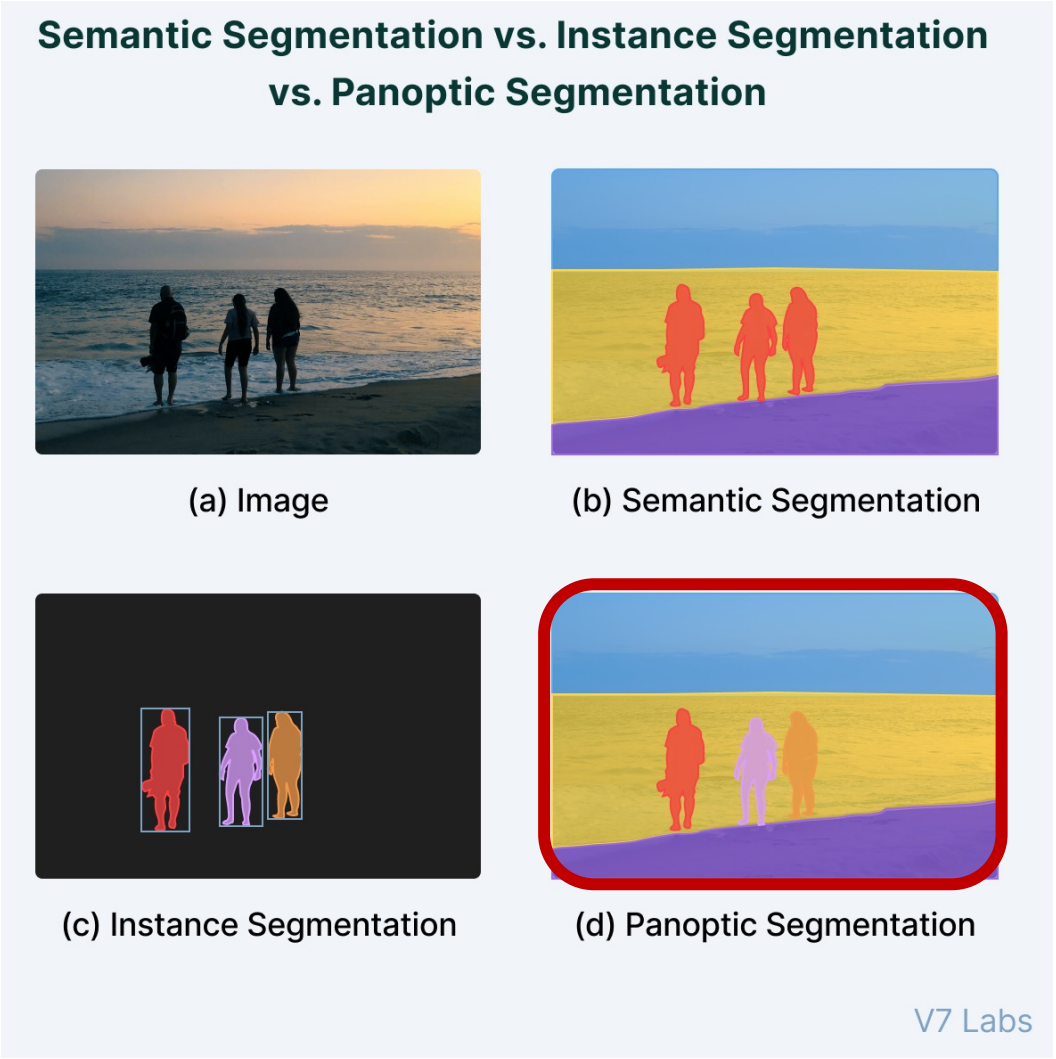


https://www.v7labs.com/blog/panoptic-segmentation-guide

Deep Learning-Based Scene Understanding

Panoptic Segmentation

- Combination of **semantic and instance segmentation**
- **Foreground** (things) with instance IDs
- **Background** (stuff) without instance IDs
- Richer semantic information
- Application: Long-term object interaction



https://www.v7labs.com/blog/panoptic-segmentation-guide

Majority Voting for Semantic Labels

- Let *K* be the number of classes for each semantic voxel v
- Each voxel v maintains a **vote vector**

$$\boldsymbol{c} = [c_1, c_2, \dots, c_k, \dots, c_{K-1}, c_K]$$

- At time t, let measurement assign **class *k*** to the voxel v

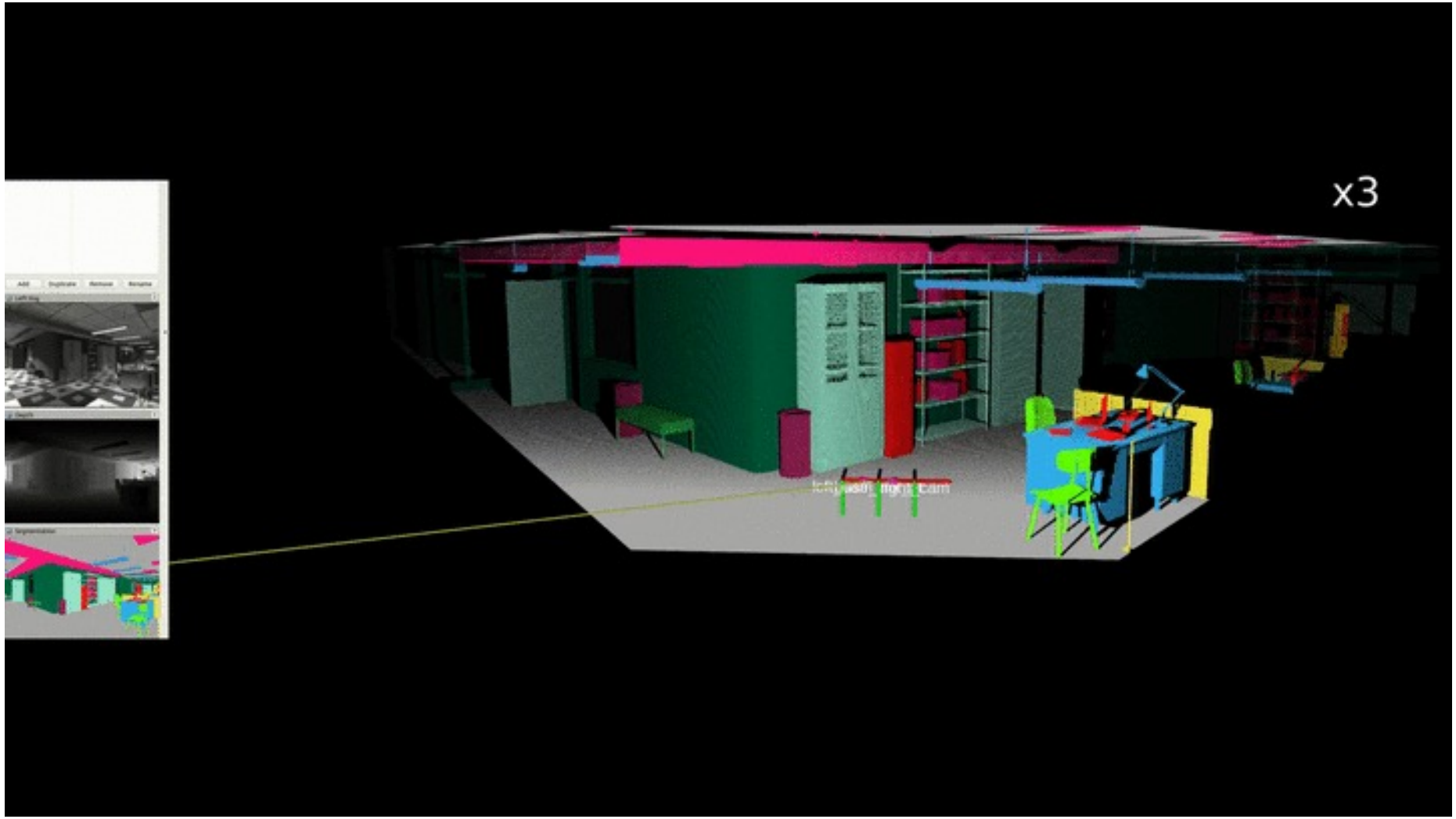
$$\boldsymbol{c_t} = \boldsymbol{c_{t-1}} + [0_1, 0_2, \dots, 1_k, \dots, 0_{K-1}, 0_K]$$

- Conversion from vote vector to **hard label** for voxel v

$$l_t^v = argmax(\boldsymbol{c_t})$$

- When two classes have equal number of votes, either pick randomly or use uncertainty

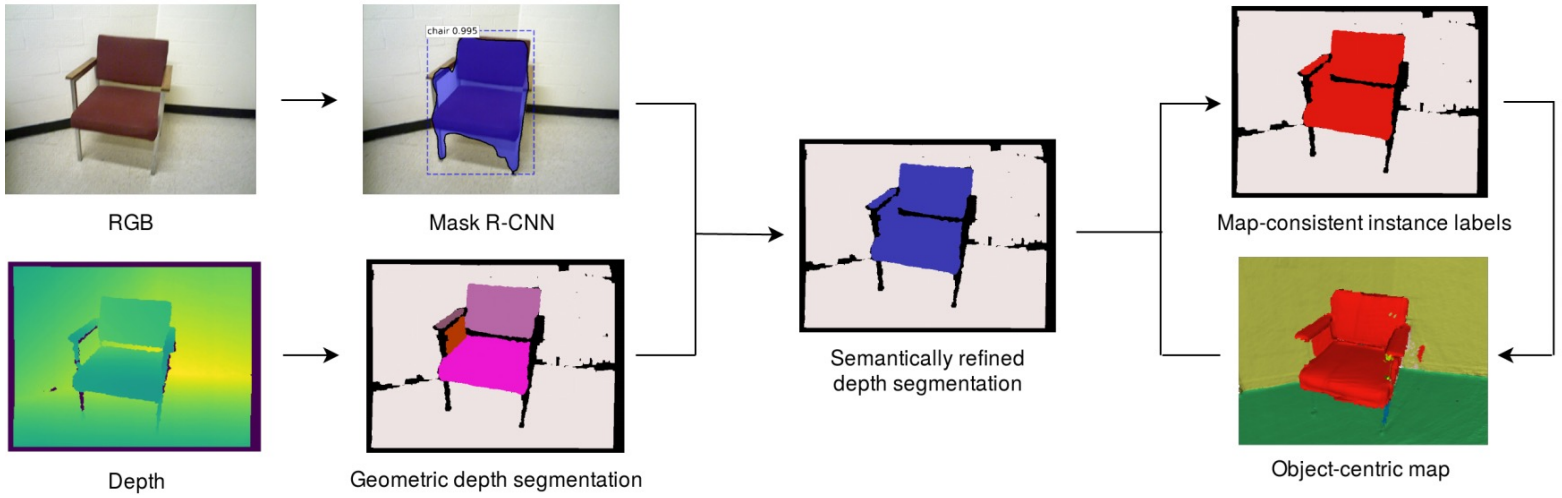
Semantic Mapping



[A. Rosinol, M. Abate, Y. Chang, and L. Carlone, ICRA 2020]

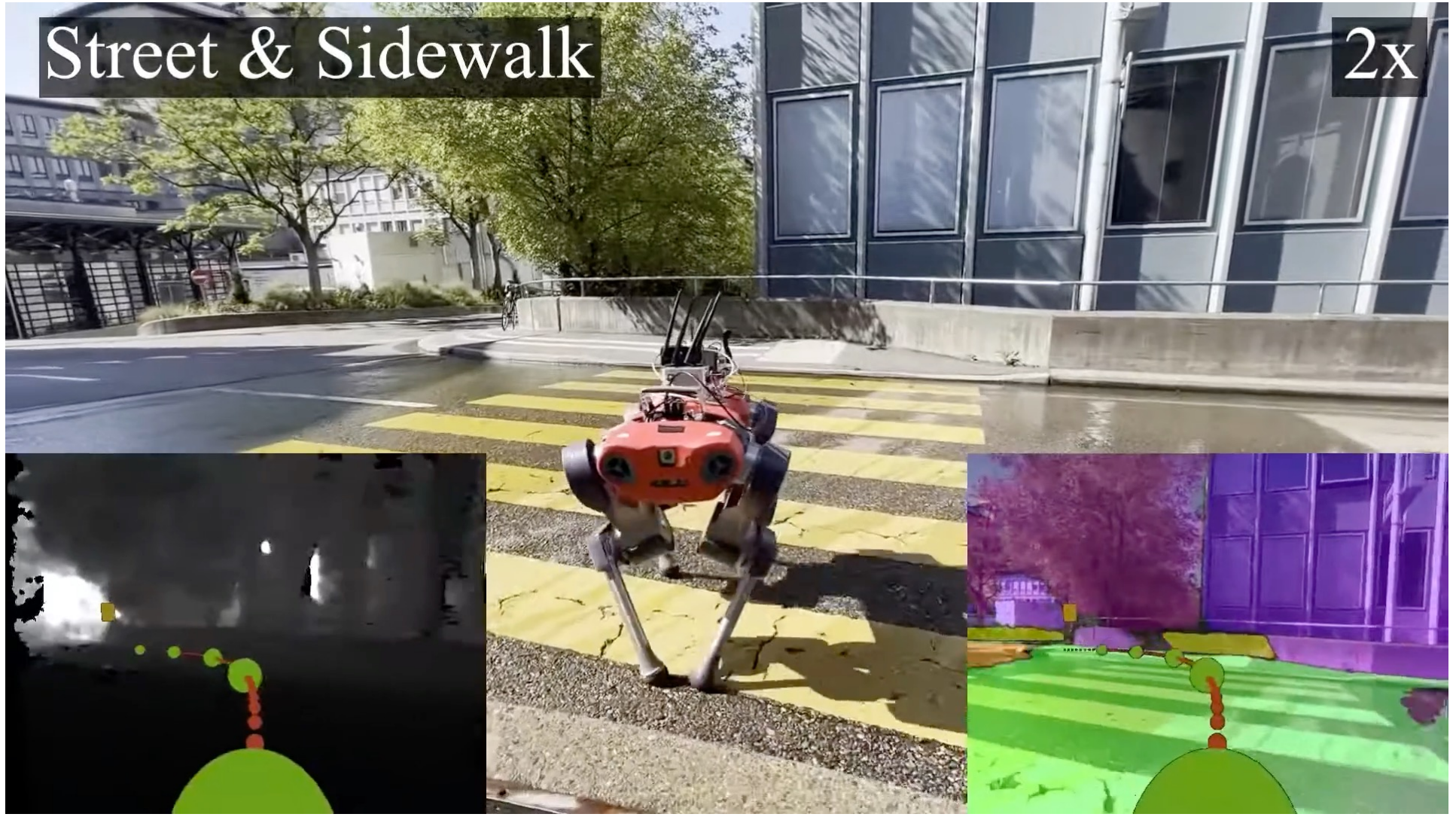
Semantic Instance Volumetric Mapping

- Convolutional networks, e.g., Mask RCNN or Panoptic DeepLab enable to label objects in RGB images
- Fuse per frame **semantic information from RGB images** with **geometric information from depth data**



[Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery, Grinvald et al., RA-L 2019]

Semantic Maps for Navigation



[P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, ICRA 2024]

Semantic-Aware Volumetric Mapping

Pros

- Semantically rich maps
- Enables semantic-based planning and interaction
- Closer to how humans abstract environments

Cons

- Requires large neural network models
- More complex and expensive

[Grinvald et al., RA-L 2019]

Scene Graph Representation

Hierarchical Perception Layers

Raw Geometry (Point Clouds)

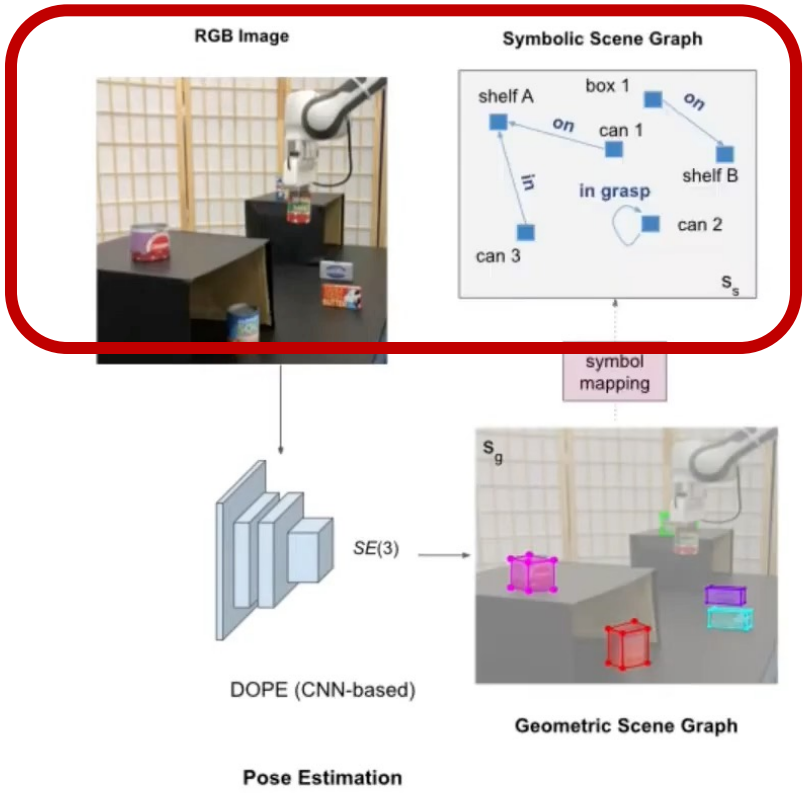
Metric Representation
(TSDF, Octomap, Heightmaps)

Semantic-Metric Representation
(Semantic Maps)

Hierarchical Relationships
(Scene Graphs)

Scene Graphs

Hierarchical Planning



Task Goal

On(box 2, shelf B)

On(can 2, can 1)

Symbol mapping to Symbolic Scene Graph

[Zhu, Tremblay, Birchfield, and Zhu, ICRA 2021]

Scene Graphs

- Nodes represent objects and agents
- Edges encode spatial and semantic relations
- Structured 3D environment representation
- Enables high-level environment queries and task planning
- Incrementally updated with perception data

Summary

- The best model depends on the application
- **Voxel** representations allow for a full 3D representation
- **Octrees** are a compact, inherently multi-resolution, probabilistic 3D representation
- **Signed distance functions** also use 3D grids but allow for a sub-voxel accuracy representation of the surface
- **Surface models** support traversability & graspability analysis
- **Semantic-aware volumetric maps** for long-term interaction
- **Scene graphs** are abstract models and support task planning

Literature 3D World Models (1)

- *OctoMap: An Efficient Probabilistic 3D Mapping Framework Based on Octrees*,
  A. Hornung,. K.M. Wurm, M. Bennewitz, C. Stachniss, and W. Burgard, Autonomous Robots, 2013
- *World Modeling*
  W. Burgard, M. Herbert, and M. Bennewitz, Handbook of Robotics (2nd edition), Chapter 45, Springer, 2016.
- *Real-Time Camera Tracking and 3D Reconstruction Using Signed Distance Functions*,
  E. Bylow, J. Sturm, C. Kerl, F. Kahl, and D. Cremers, Robotics: Science and Systems (RSS), 2013
- *Real-time navigation in 3D environments based on depth camera data*
  D. Maier, A. Hornung and M. Bennewitz, Humanoids 2012
- *Continuous Humanoid Locomotion over Uneven Terrain using Stereo Fusion*,
  M. F. Fallon, P. Marion, R. Deits, T. Whelan, M. Antone, J. McDonald, and R. Tedrake, Humanoids 2015

Literature 3D World Models (2)

- *Survey of 3D modeling using depth cameras*
  H. Xu, J. Xu, and W. Xu, 2019, Virtual Reality & Intelligent Hardware
- *Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery*,
  M. Grinvald, F. Furrer, T. Novkovic, J.J. Chung, C. Cadena, R. Siegwart, & J. Nieto,
  IEEE Robotics and Automation Letters, 2019
- *Kimera: An Open-Source Library for Real-Time Metric-Semantic Localization and Mapping*
  A. Rosinol, M. Abate, Y. Chang, and L. Carlone, ICRA 2020
- *ViPlanner: Visual Semantic Imperative Learning for Local Navigation*
  P. Roth, J. Nubert, F. Yang, M. Mittal, and M. Hutter, ICRA 2024
- *Hierarchical planning for long-horizon manipulation with geometric and symbolic scene graph*,
  Y. Zhu, J. Tremblay, S. Birchfield, and Y. Zhu, ICRA 2021