上海科技大学
ShanghaiTech University

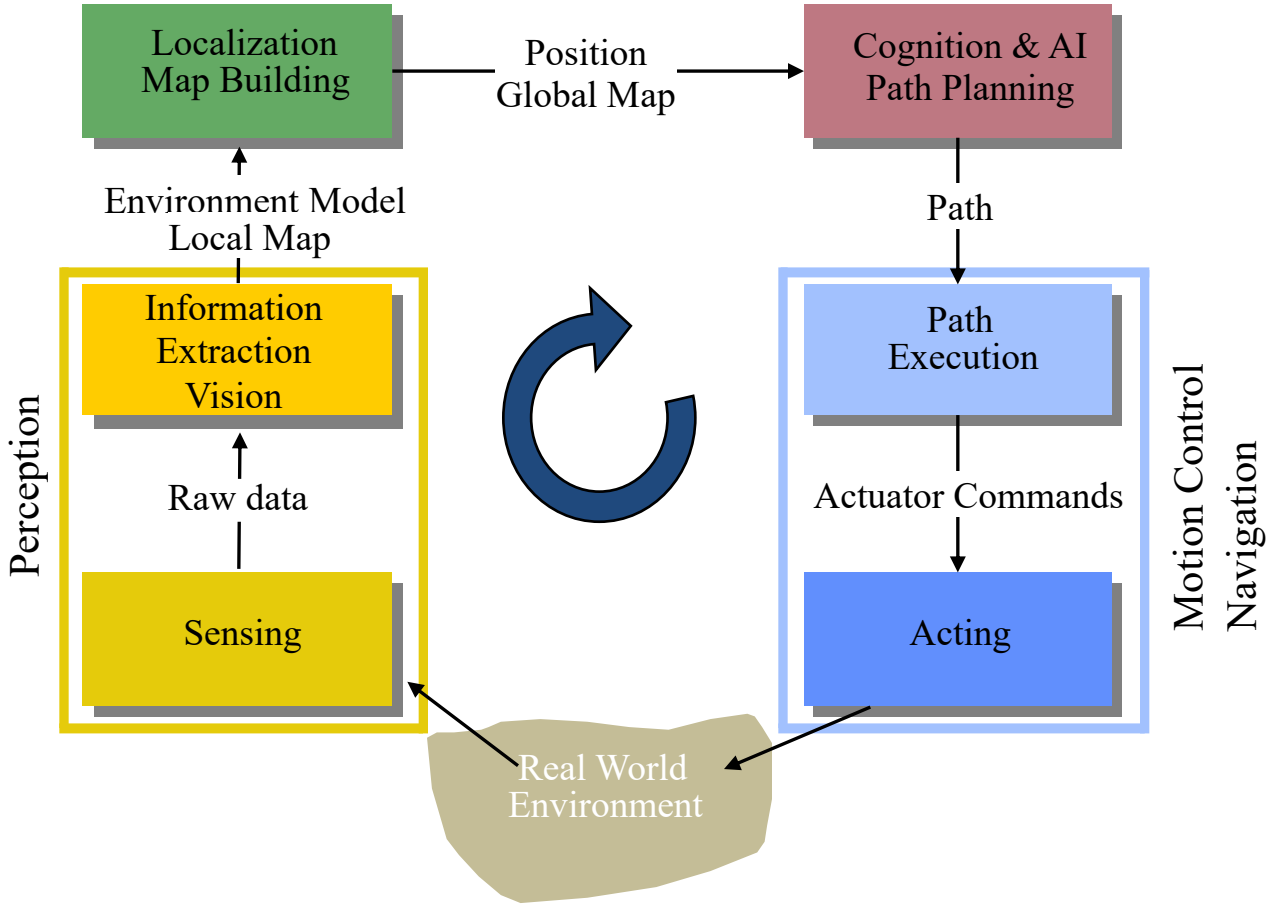
CS283: **Robotics Spring 2023: Maps**

Sören Schwertfeger / **师泽仁**

ShanghaiTech University

MAP REPRESENTATION

General Control Scheme for Mobile Robot Systems

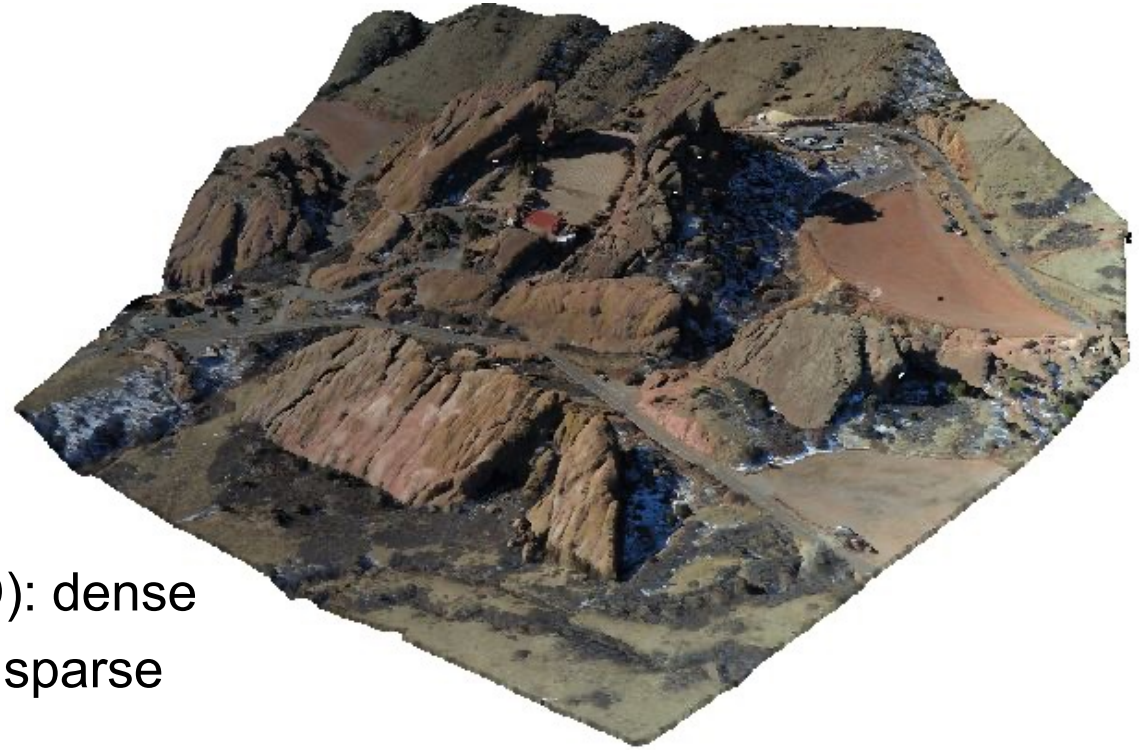


With material from Roland Siegwart and Davide Scaramuzza, ETH Zurich

Map Representation: what is "saved" in the map

- Points (surface of objects, buildings): 2D or 3D
  - What: x,y or x,y,z coordinates;
    Optional: intensity; maybe RGB; maybe descriptor; temperature; …
  - From range sensors (laser, ultrasound, stereo, RGB-D): dense
  - From cameras (structure from motion; feature points): sparse
  - Variant: kd-tree
- Grid-map: 2D or 3D
  - Option: probabilistic grid map
  - Option: elevation map
  - Option: cost map
  - Option: Truncated Signed Distance Field
  - Option: Normal Distributions Transform (NDT)
  - Variant: Quad-tree; Oct-tree
- Higher-level Abstractions
  - Lines; Planes; Mesh
  - Curved: splines; Superquadrics
- Semantic Map
  - Assign semantic meaning to entities of a map representation from above
  - E.g. wall, ceiling, door, furniture, car, human, tree, …
- Topologic Map
  - High-level abstraction: places and connections between them
- Hierarchical Map
  - Combine Maps of different scales. E.g.:
  - Campus, building, floor
- Pose-Graph Based Map
  - Save (raw) sensor data in graph, annotated with the poses; generate maps on the fly
- Dynamic Map
  - Capture changing environment
- Hybrid Map
  - Combination of the above

Point based map

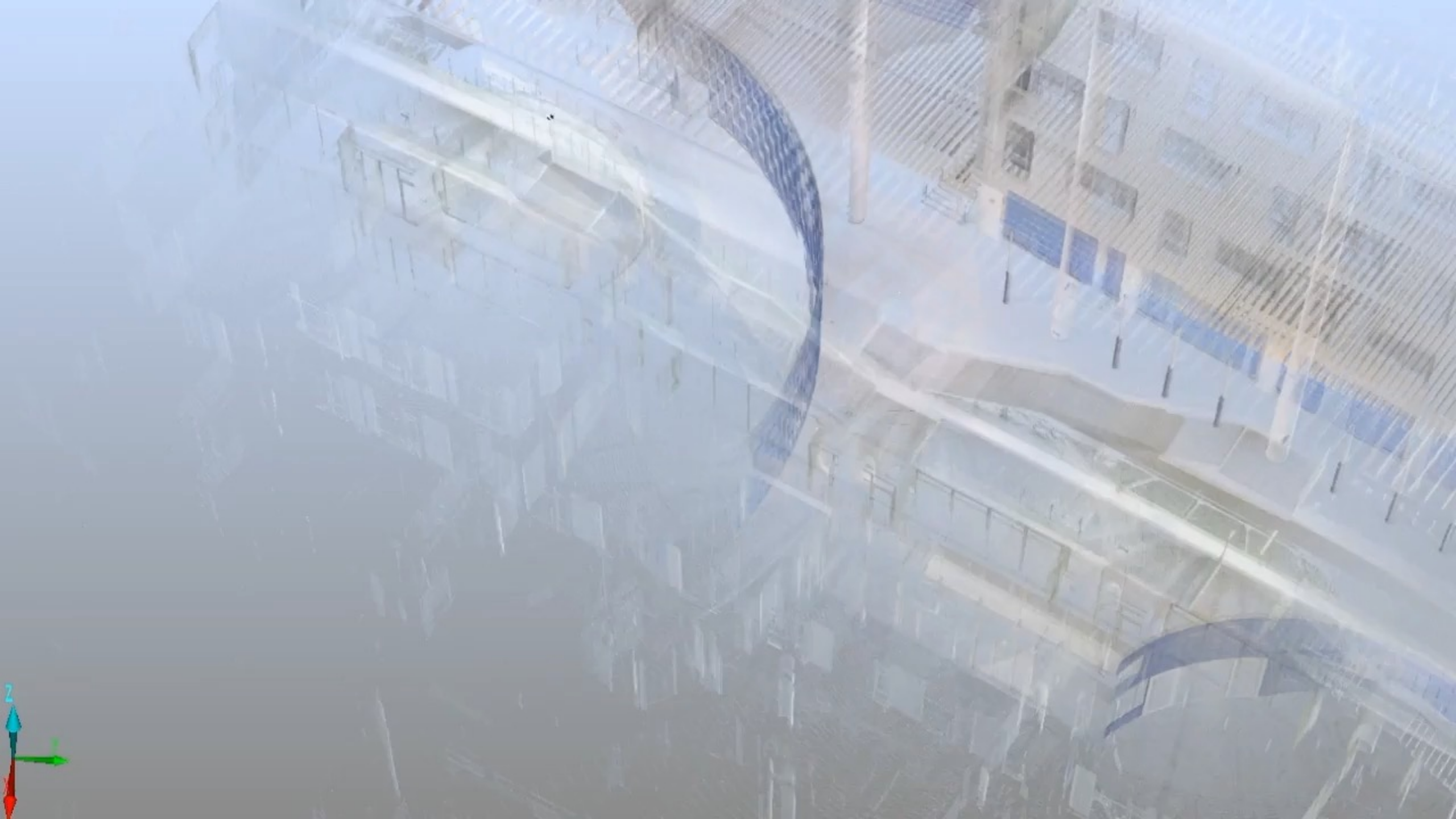


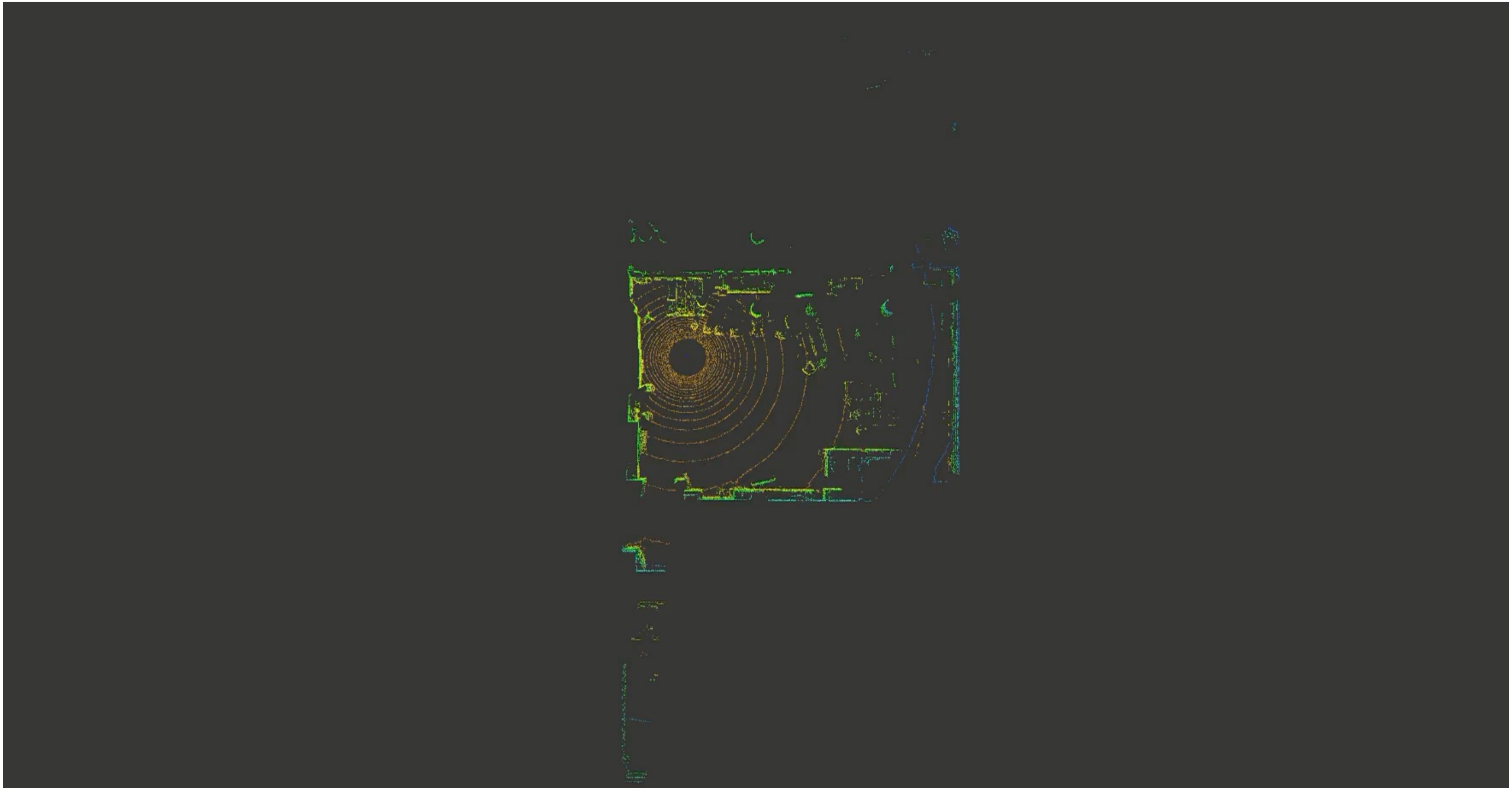
- Point Cloud
  - What: x,y or x,y,z coordinates;
    Optional: intensity; maybe RGB; maybe descriptor; temperature; …
  - From range sensors (laser, ultrasound, stereo, RGB-D): dense
  - From cameras (structure from motion; feature points): sparse
  - Variant: kd-tree
  - Structured point cloud: points are ordered in a 2D grid -> could calculate depth image from it
  - PCL: point cloud library: Used in ROS; https://pointclouds.org/
  - sensor_msgs/PointCloud sensors_msgs/PointCloud2
  - Excellent viewer: CloudCompare https://www.danielgm.net/cc/
  - File formats: PLY (bin & ASCII); XZY (ASCII); PCD (from pcl); LAS (terrestrial scanning); …

RGBD-Inertial Trajectory Estimation and Mapping for Ground Robots

Zeyong Shan, Ruijian Li and Sören Schwertfeger

Source code: https://github.com/STAR-Center/VINS-RGBD

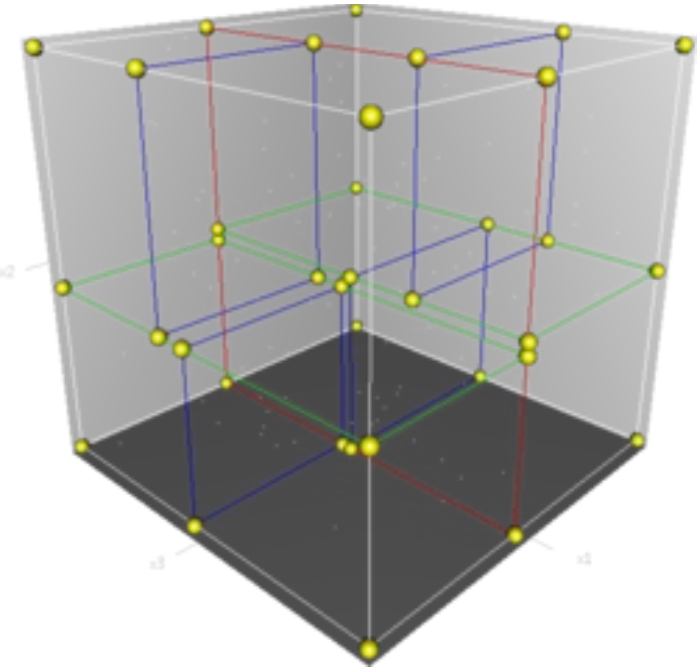

Z
Y

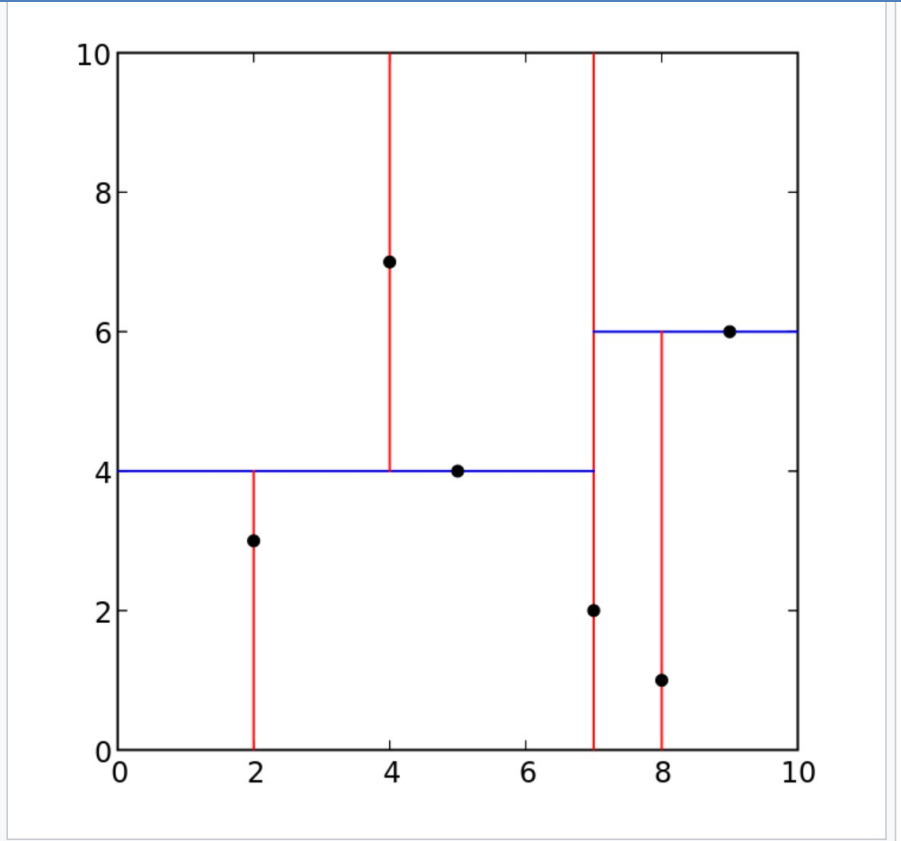


k-d tree

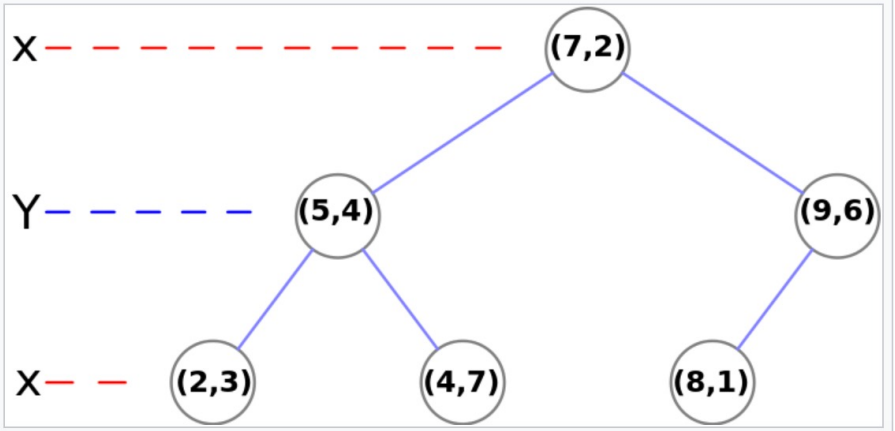
- k-dimensional binary search tree
- Robotics: typically 3D or 2D
- Every level of tree:
  - For a different axis (e.g. x,y,z,x,y,z,x,y,z) (=> split space with planes)
  - Put points in left or right side based on median point (w.r.t. its value of on the current axis) =>
  - Balanced tree
- Fast neighbor search -> ICP!

Algorithm	Average	Worst case
Space	$O(n)$	$O(n)$
Search	$O(\log n)$	$O(n)$
Insert	$O(\log n)$	$O(n)$
Delete	$O(\log n)$	$O(n)$



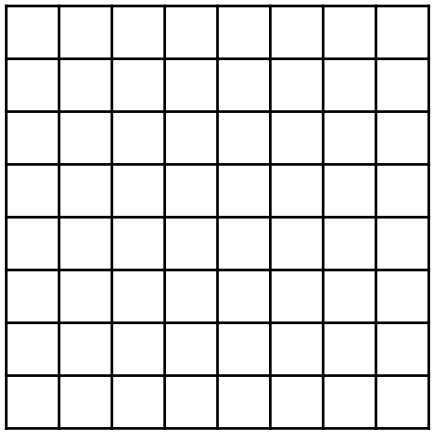


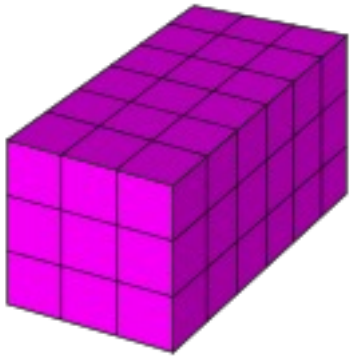
k-d tree decomposition for the point set `(2,3), (5,4), (9,6), (4,7), (8,1), (7,2)`.



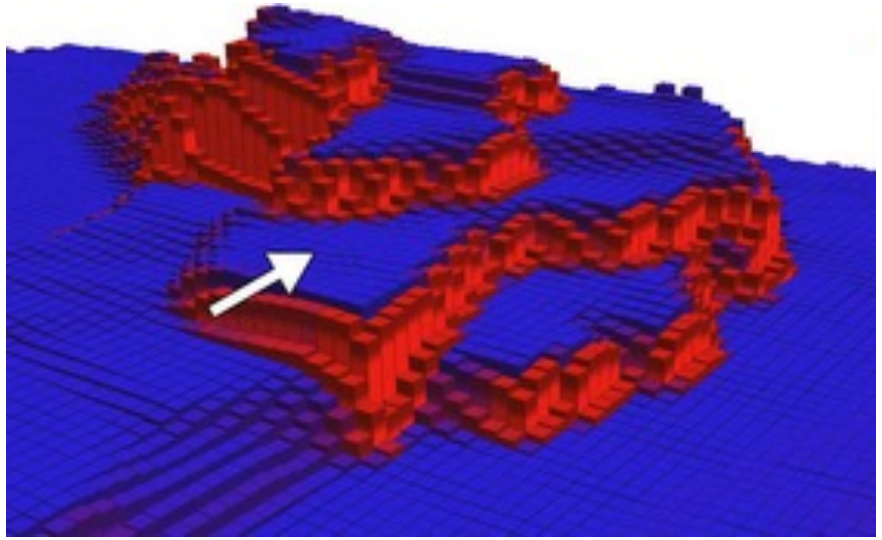
The resulting *k*-d tree.

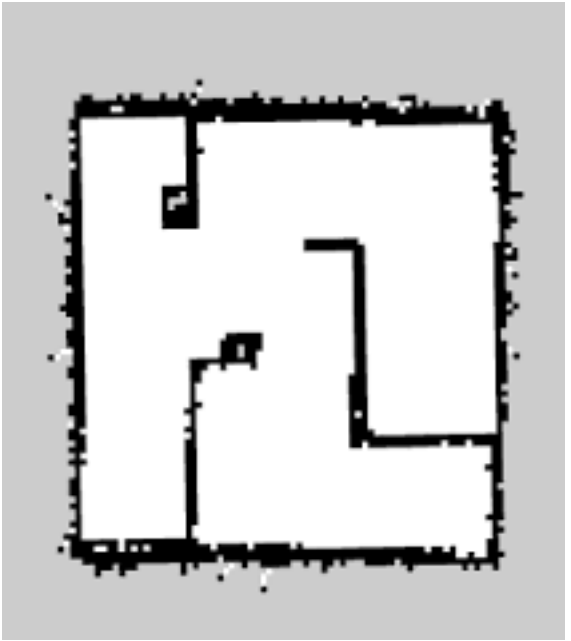
Grid Maps





- Grid-map: 2D or 3D cartesian grid
- What to save in the cell:
  - Binary: Free; Occupied;
  - Colored: +Unknown; +Searched; +Path; …
  - Probability of being occupied 0…1.0
  - Height above ground: Elevation Map
  - Cost map: used for planning (covered later)





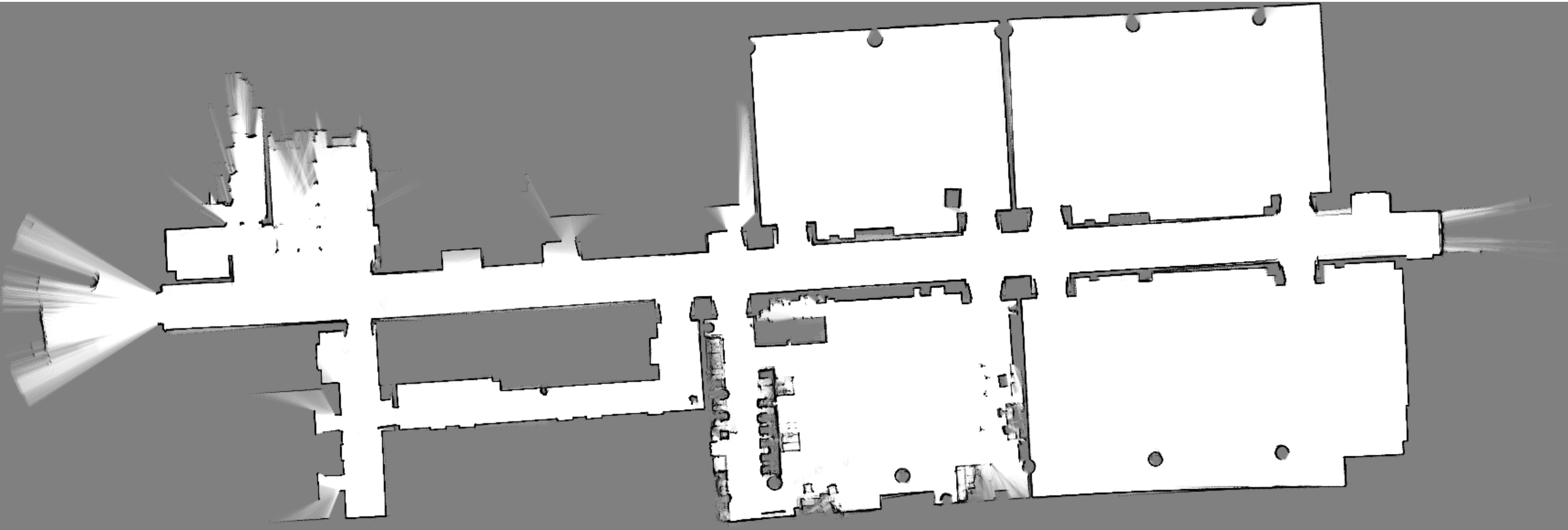
https://emanual.robotis.com/docs/en/platform/turtlebot3/slam/

Grid Maps

- Resolution: e.g. 1 pixel == 5cm
- Size: e.g. 2000 pixel x 2000 pixel
- Scale: e.g. 1:100.000 (2DoF of 3 parameters resolution, size, scale)
- Frame
  - A map has a frame. Where? In the center of the image? Top left corner?
- Time stamp
- Formats:
  - Image: png, pgm (grey scale), tiff, jpeg (lossy!)
  - GeoTIFF: georeferenced TIFF (inlined tfw World File): map projection, coordinate systems, ellipsoids, datums (see GPS slide)
  - 3D: bt (octomap)
- QGIS: open source Geographic Information System for raster (image) and vector data http://qgis.org/en/site

Probabilistic grid map

- 1: occupied; 0: free; 0.5: unknown
- Need error model of sensor (and of localization) to properly update cells with a scan
- Can remove dynamic (moving) objects (by observing the free space multiple times)



Normal Distributions Transform (NDT)

- Sparse Gaussian mixture model
- Each cell has NDT Computed from covariance matrix of points inside the cell
- Useful for scan matching/ registration

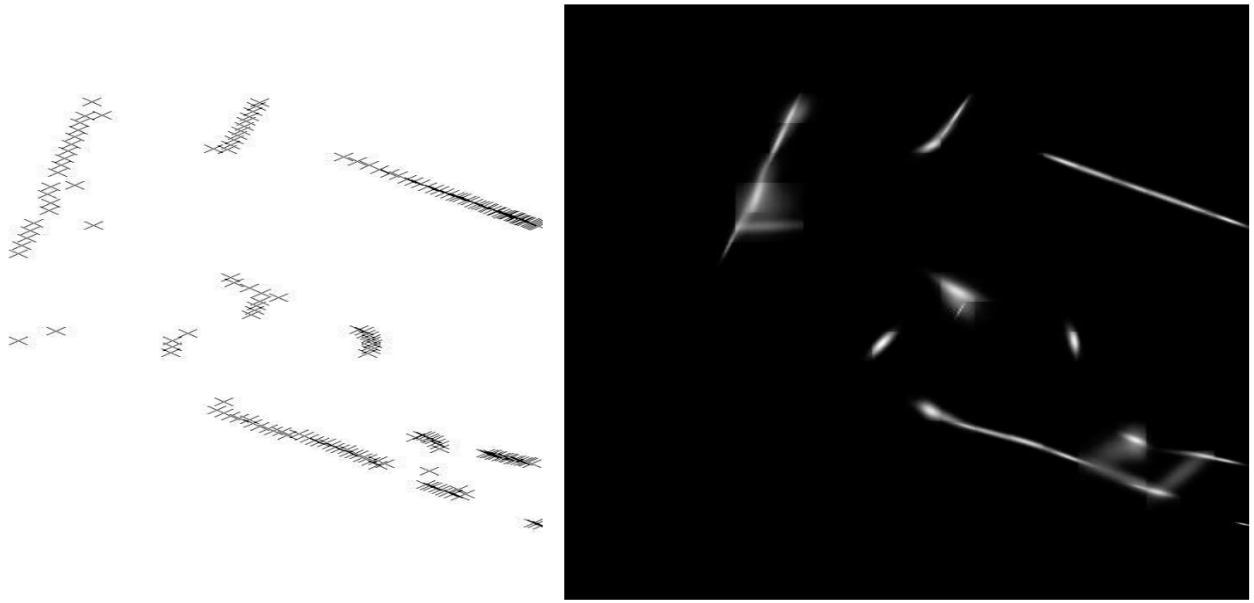
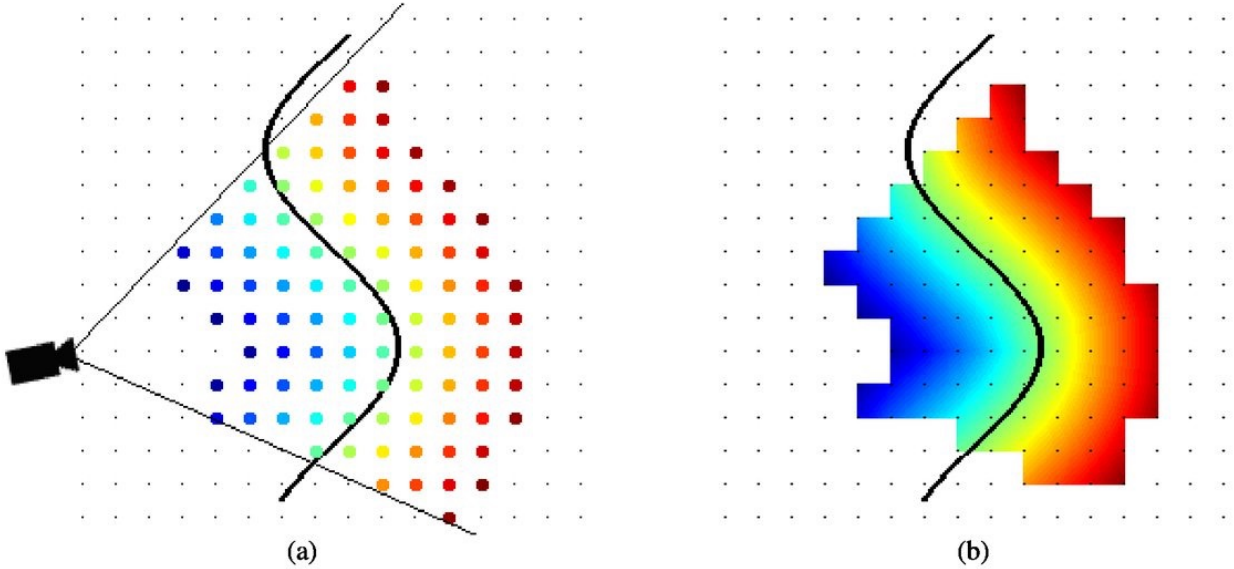


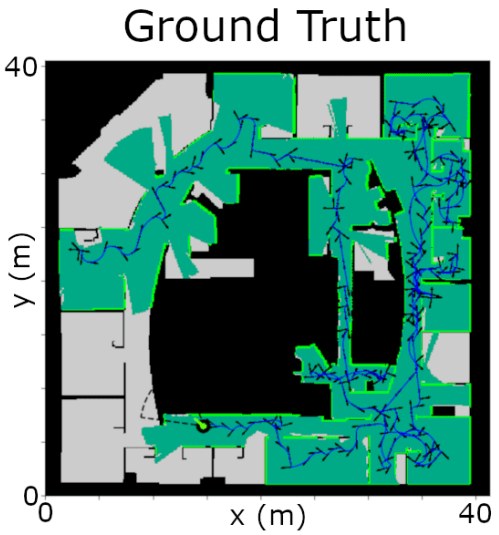
Fig. 1. An example of the NDT: The original laser scan and the resulting probability density.

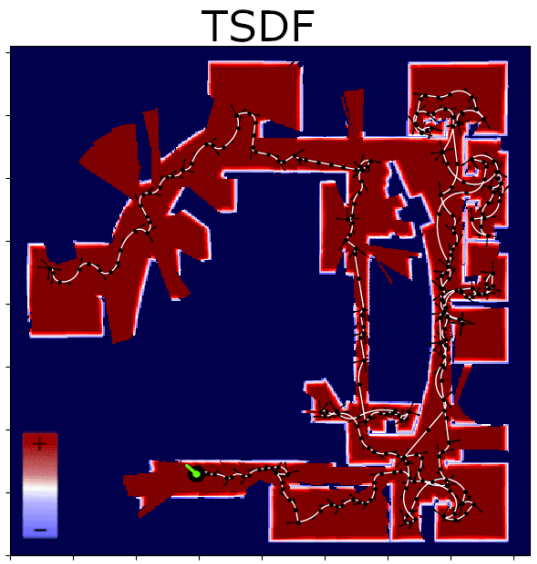
Truncated Signed Distance Field

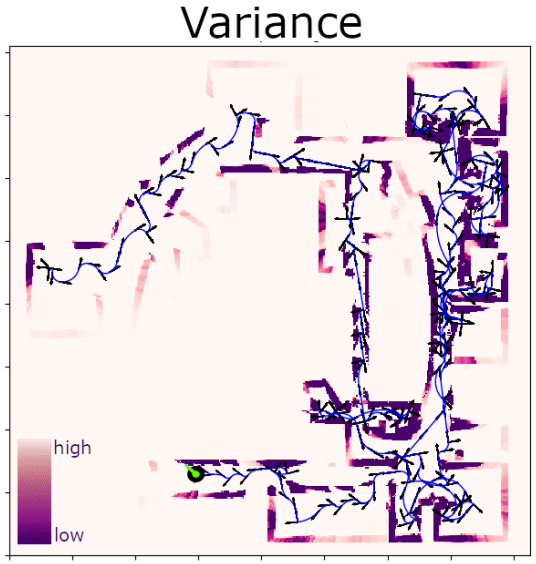
https://www.dcist.org/2020/03/20/active-exploration-in-signed-distance-fields/
https://www.spiedigitallibrary.org/conference-proceedings-of-spie/9091/909117/Real-time-reconstruction-of-depth-sequences-using-signed-distance-functions/10.1117/12.2054158.short?SSO=1&tab=ArticleLinkFigureTable

- Distance to surface:
- positive in free space; negative behind
- Up to a certain distance (truncated)
- Useful for collision checking; planning; meshing
- Can be modeled using Gaussian random variable in each cell

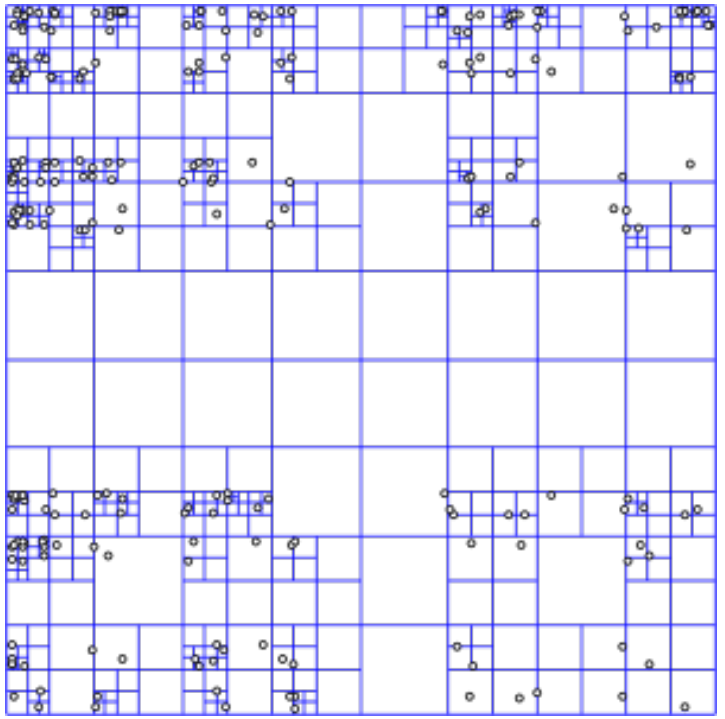








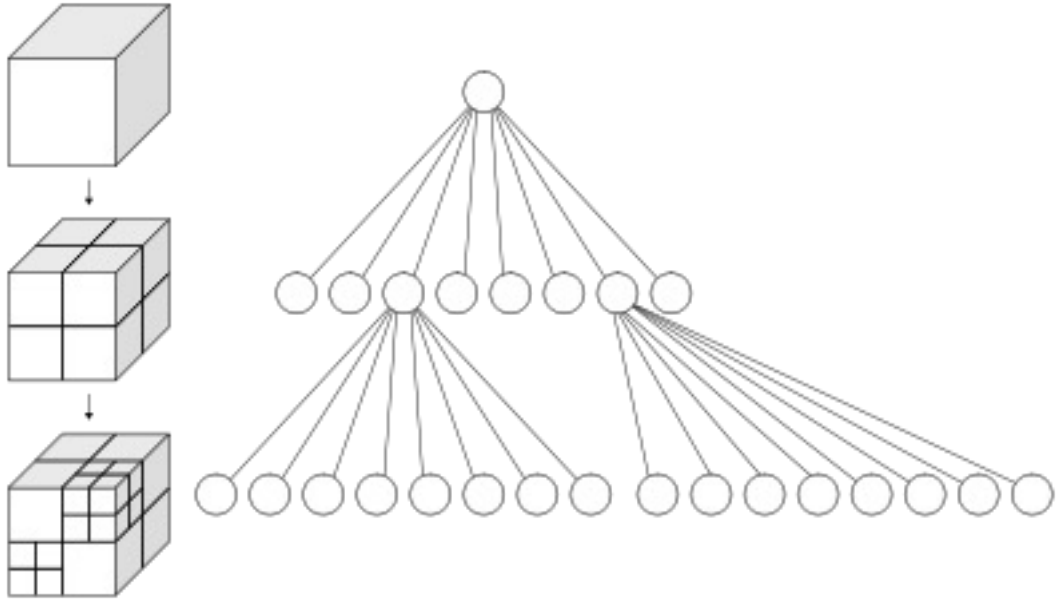
Adaptive Cell Decomposition

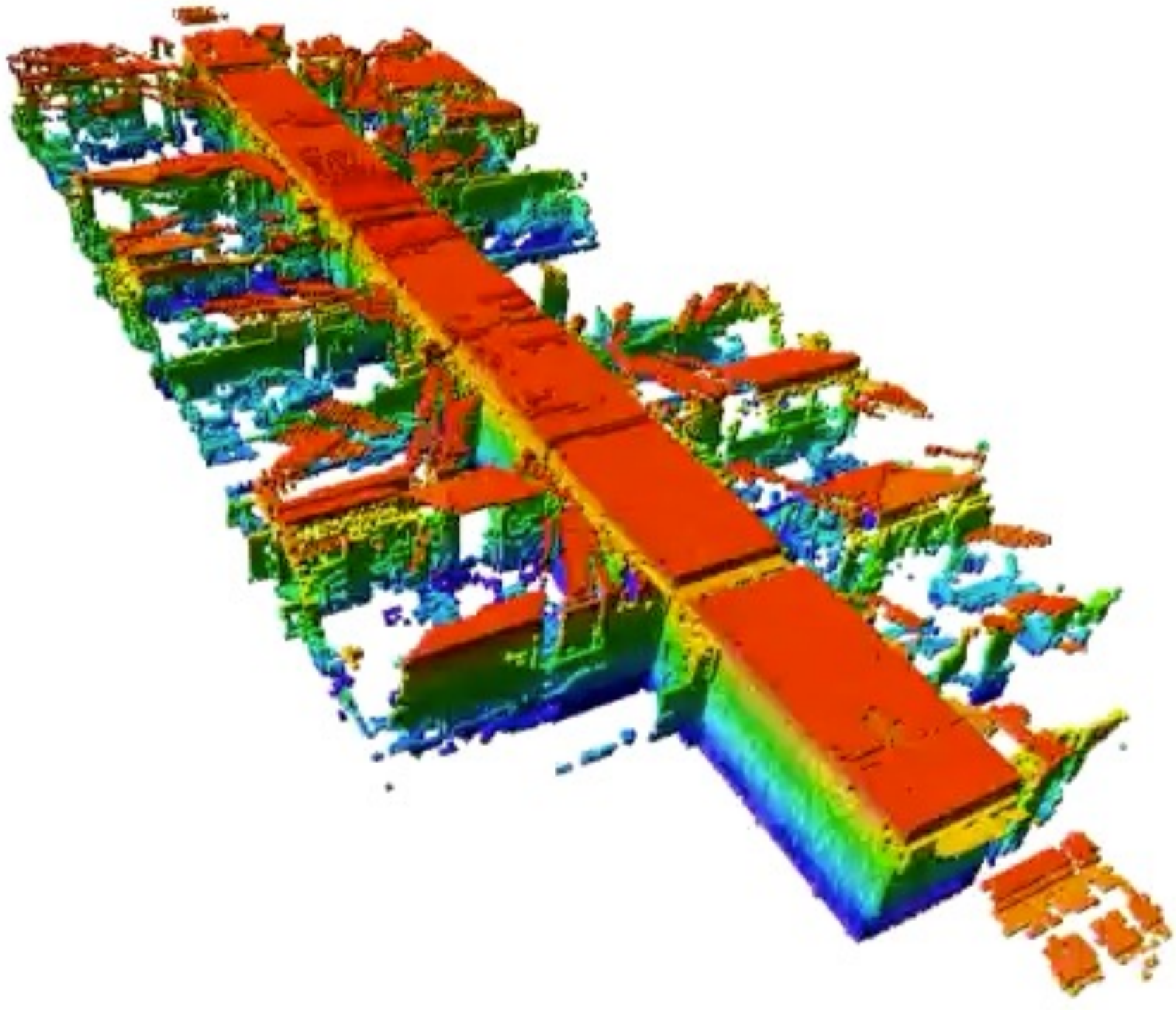


- Quad-tree
  - 2D map/ grid is recursively divided into 4 smaller cells
  - Only cells with different values/ points get divided further =>
  - Compact representation of big space (if many cells stay merged)

- Oct-tree
  - 3D grid divided into 8 smaller cells
  - Very compact! (There is lots of free space!)

- OctoMap: probabilistic oct-tree!
  - http://octomap.github.io
  - Good support in ROS (e.g. MoveIt!)





Multi-level surface maps

- 2D grid map
- Each cell can store multiple levels (saves mean and variance of height plus depth for each level)
- Useful for terrain classification and planning

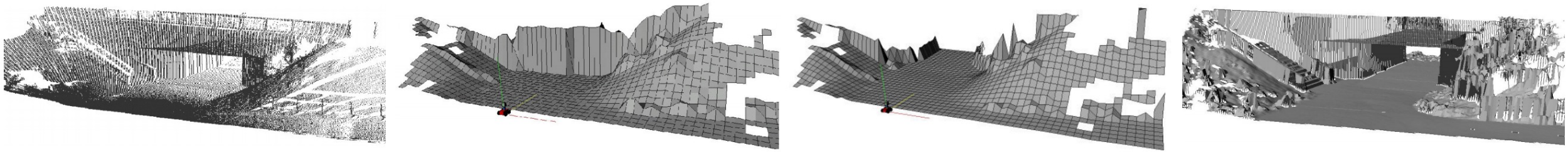


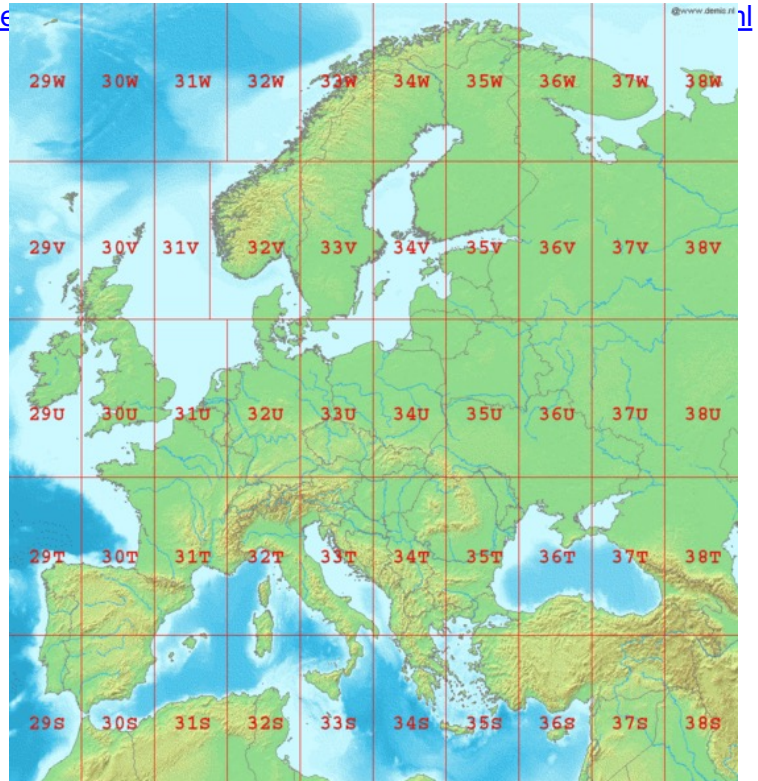
Fig. 1. Scan (point set) of a bridge (a), standard elevation map computed from this data set (b), extended elevation map which correctly represents the underpass under the bridge (c), and multi level surface map that correctly represents the height of the vertical objects (d)

Big Maps/ GPS

- World is NOT flat! =>
- Have to project 2D cartesian map onto a sphere!
- Different Projections, e.g. Mercator (preserves local directions and shapes, but not sizes)
- GPS polar coordinates: longitude & latitude of earth
- Universal Transverse Mercator (UTM): specify 60 planes on the surface of earth; specify location as Cartesian x,y,z coordinates on those planes
- World Geodetic System WGS 84 datum: used by GPS: model lang and lat to ellipsoid (Earth is not a perfect sphere!) from center of gravity …
- China: GCJ-02: WGS 84 plus random offsets (about 300-500m) (for national safety) – makes robotics difficult :/

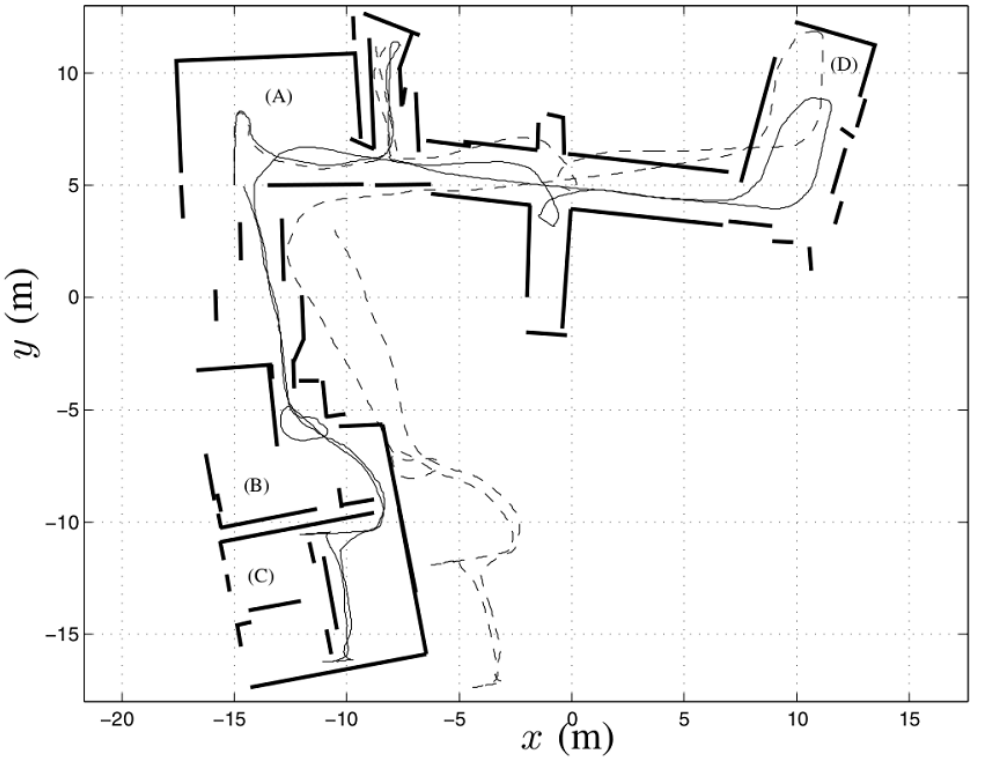
  地形图非线性保密处理算法
- Baidu: BD-09 further offsets (so competitors don't copy)

https://www.livescie

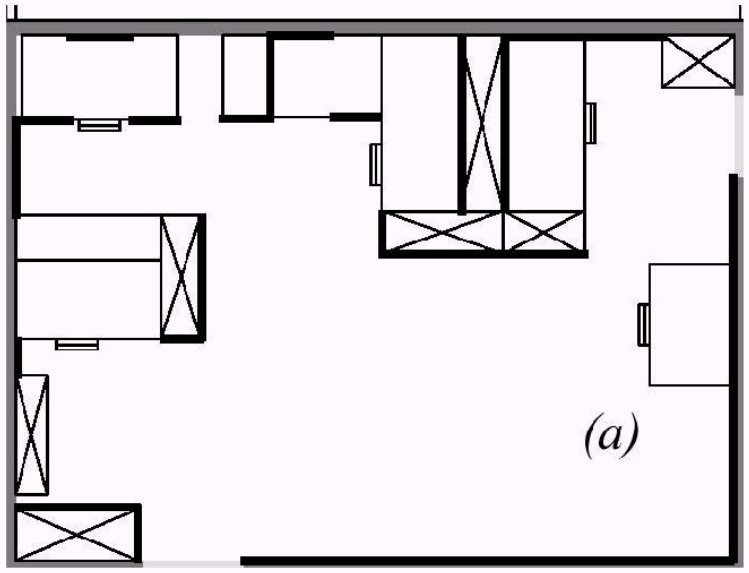


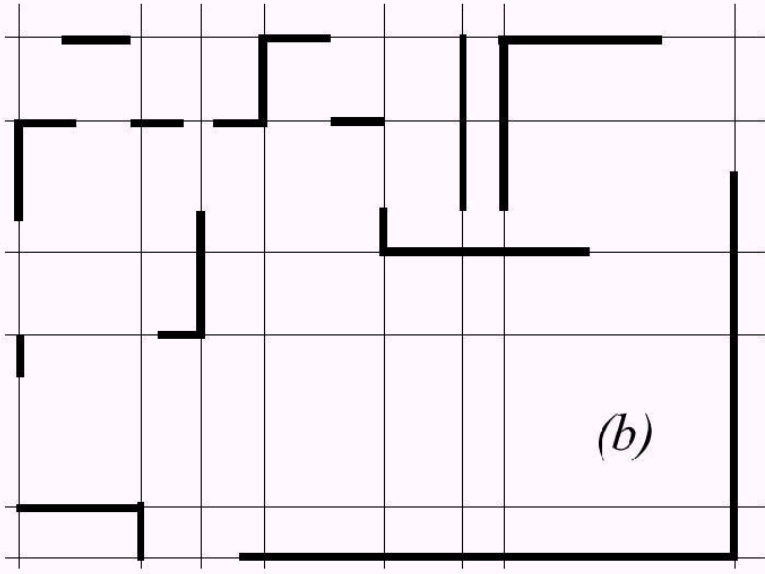
Line Map

- Abstract from points =>
- Extract features (e.g. lines, planes)
- E.g. using RANSAC, Hough Transform
- E.g. region growing, e.g. via normals



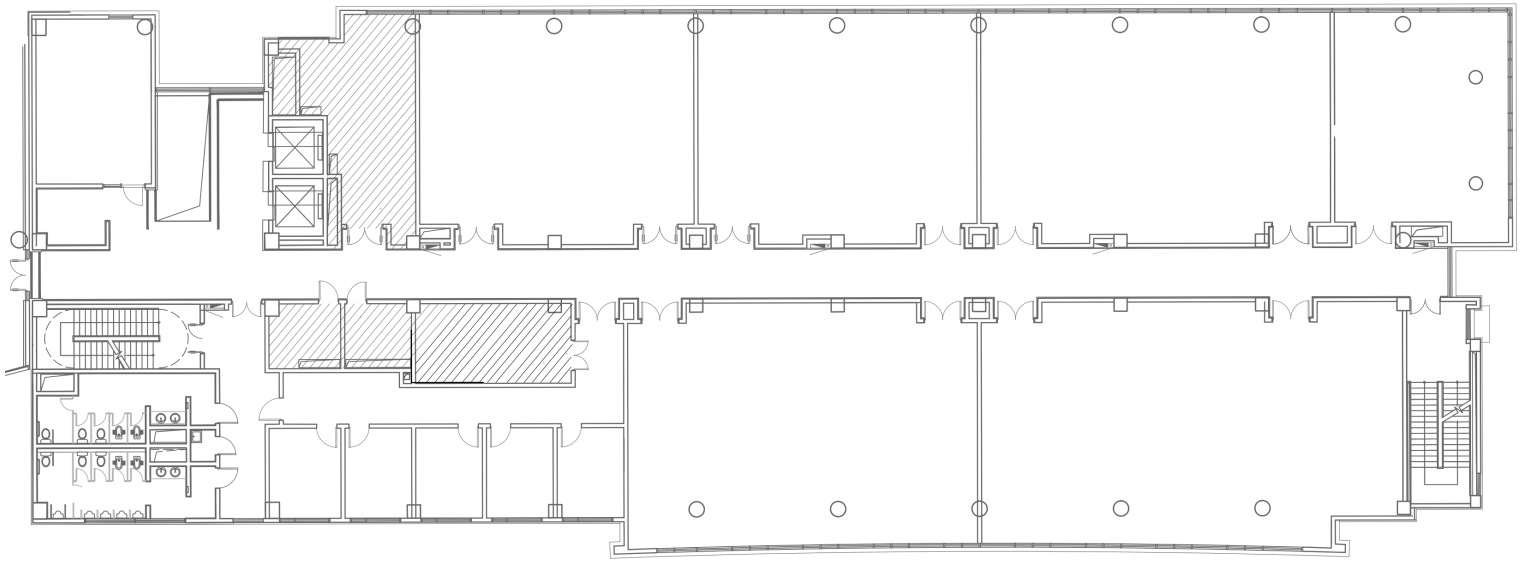
- Finite lines (a)
- Infinite lines (b)



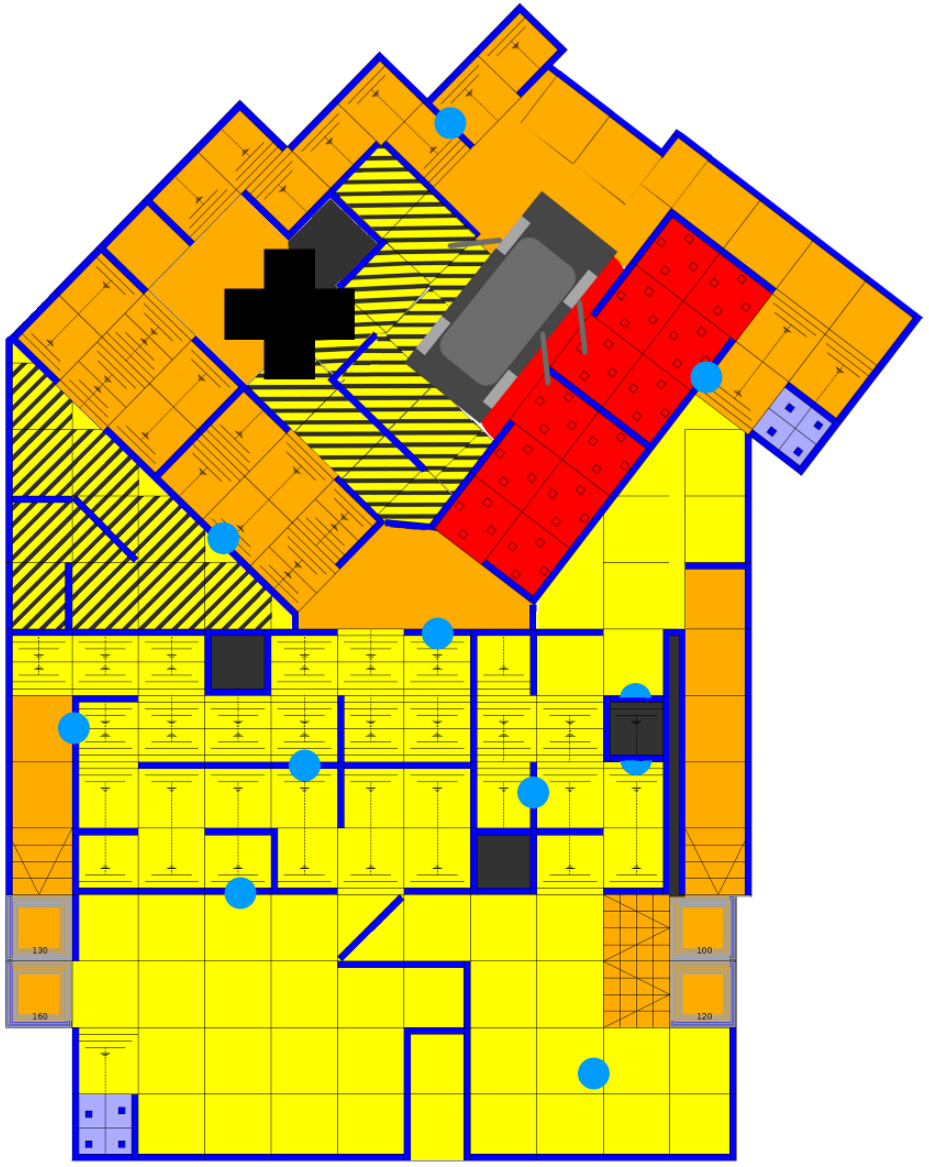


- Very compact
- Can do scan-matching (e.g. against laser scan)

Ground Truth Maps

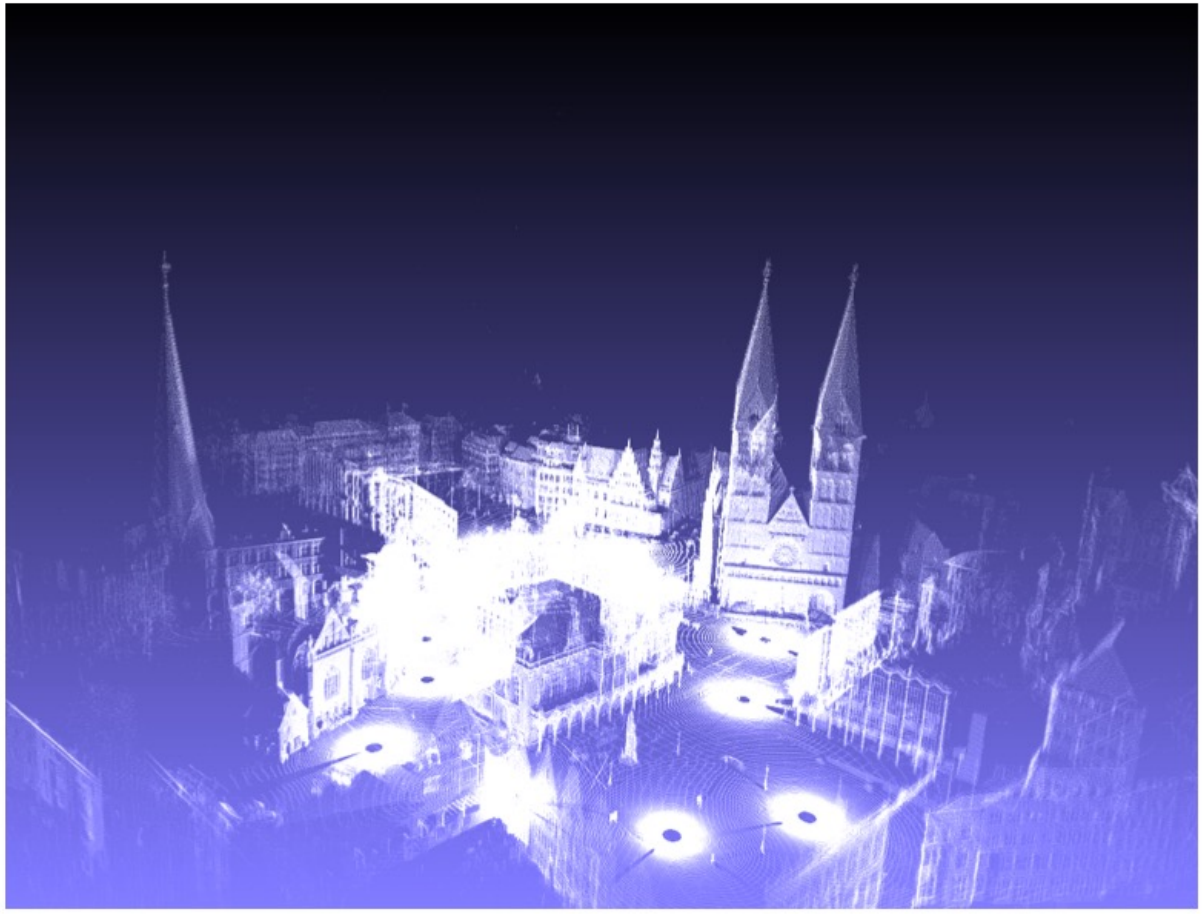


CAD drawing (STAR Center)
Vector format (lines, circles, …)

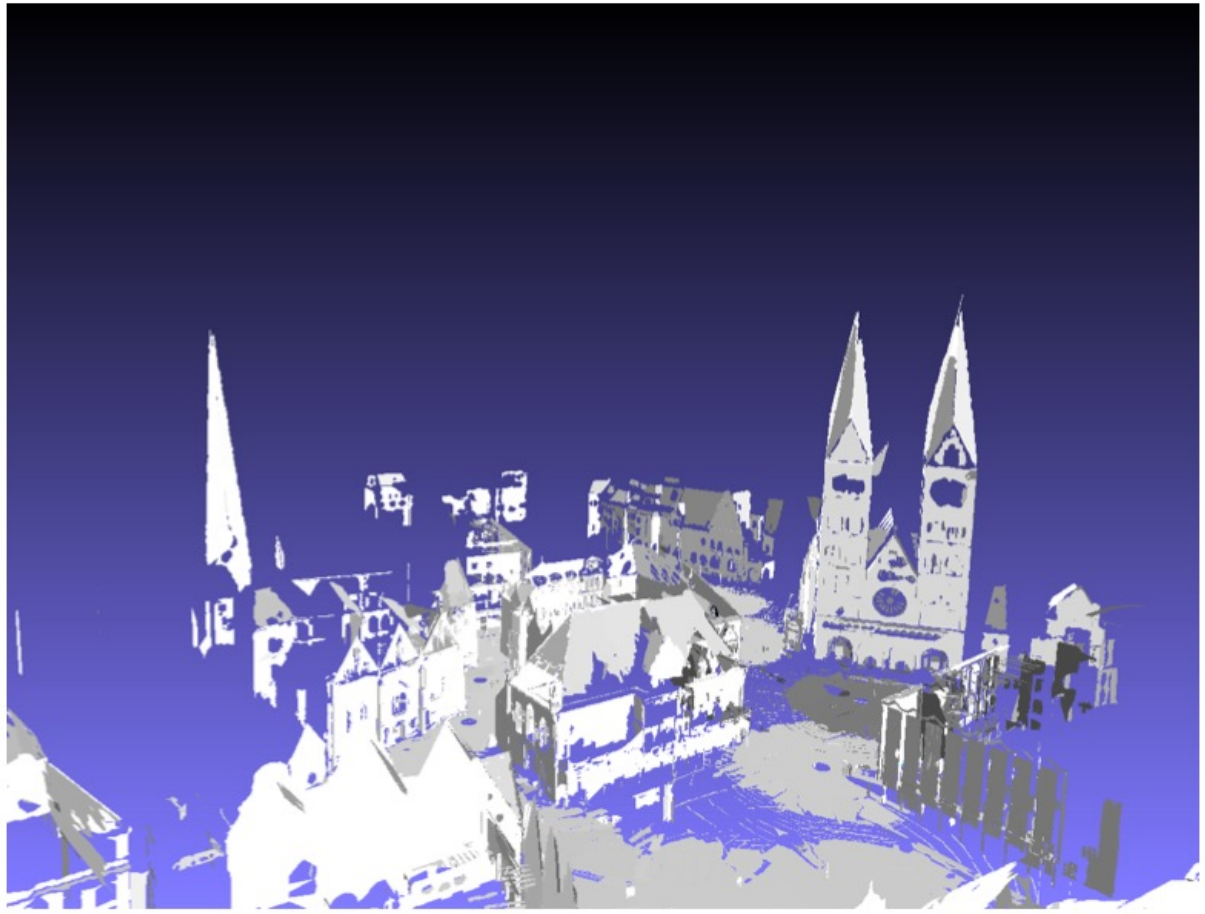


Grid Map (from vector map)

Bremen: 3D Point Cloud & 3D Plane Map

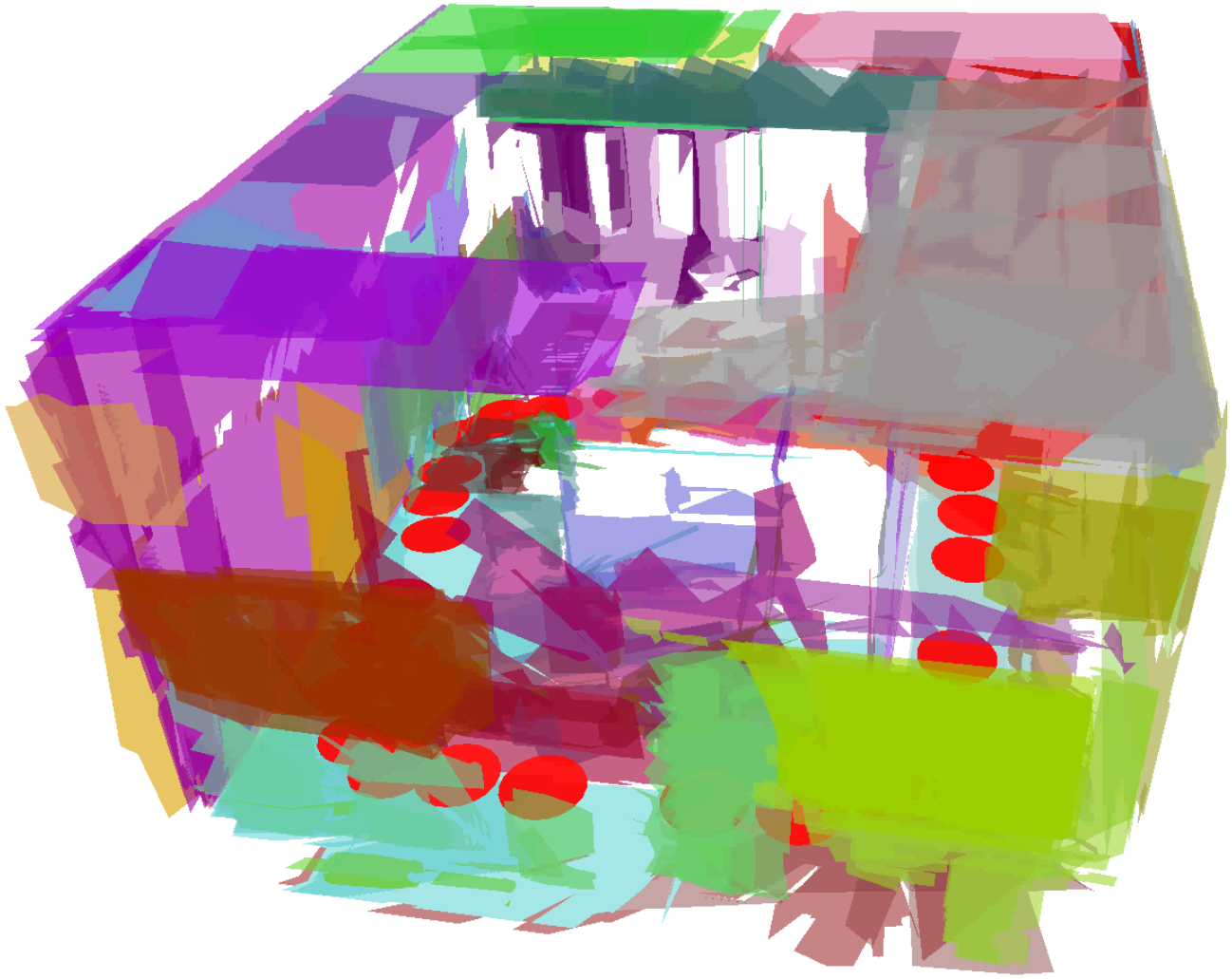


(a) 3D Point Cloud



(b) 3D Plane Map

Plane map: 29 poses; each with several planes



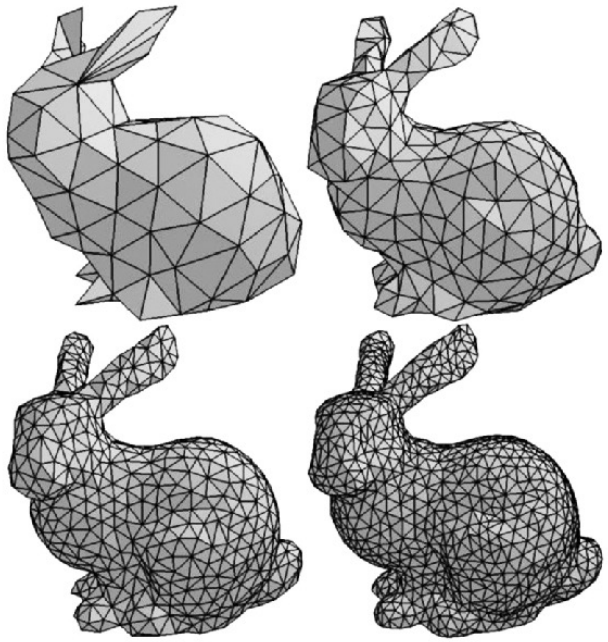
Vector Maps

- E.g. Open Street Maps
- Represented as OSM files (xml) or PBF (binary)

- Nodes in WGS 84 (vertices)
  - Only entity with position
  - Just for ways or
  - Object (e.g. sign)

- Ways:
  - Open polygon (street)
  - Closed polygons
  - Areas
  - With tags (e.g. name, type, …)

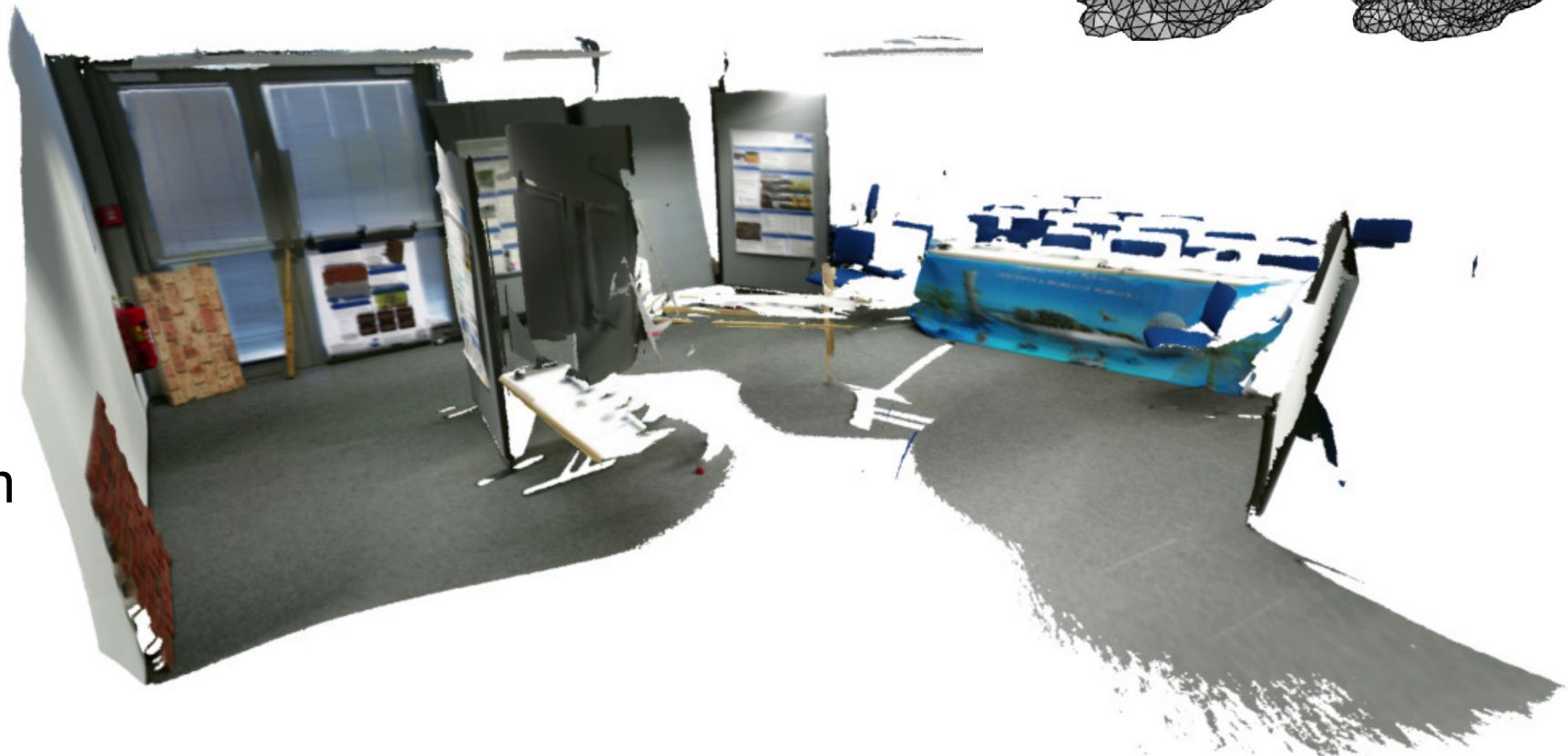
Mesh



- Often build via Signed Distance Field

- Close relation to 3D reconstruction from Computer Vision

- Often with texture (RGB information from camera)

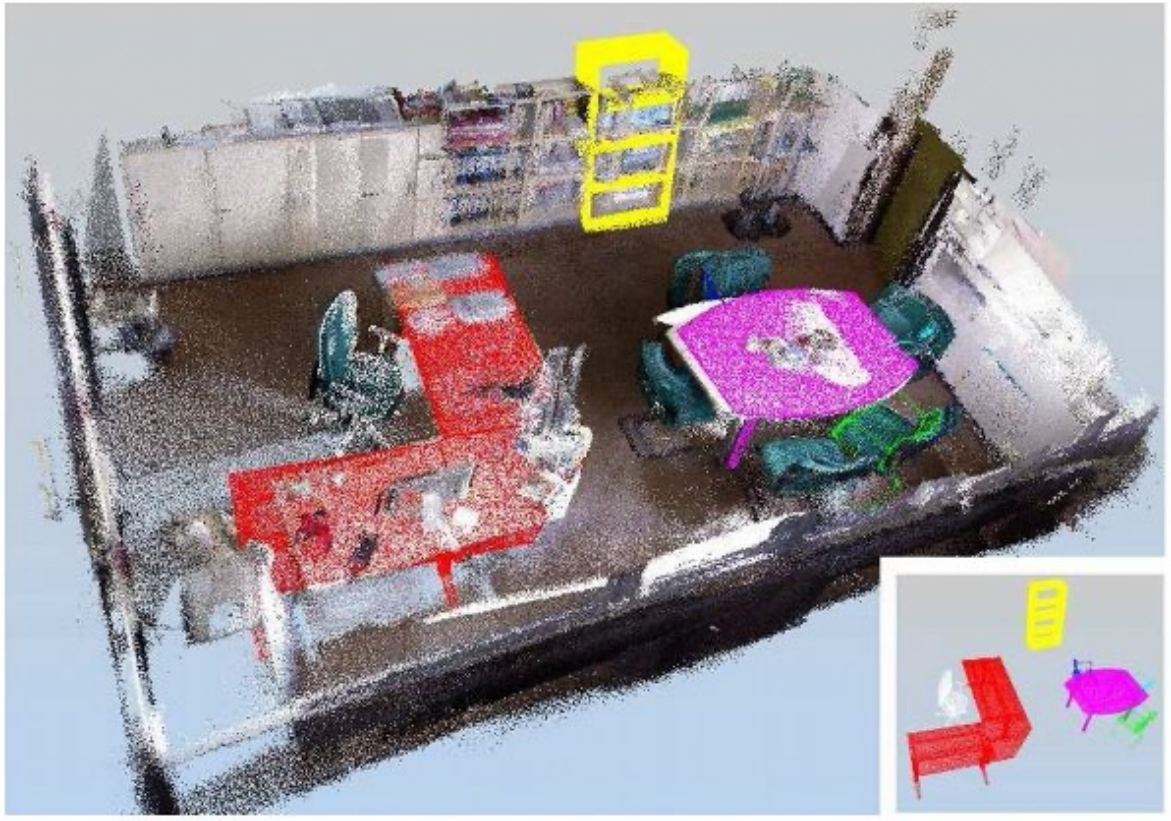


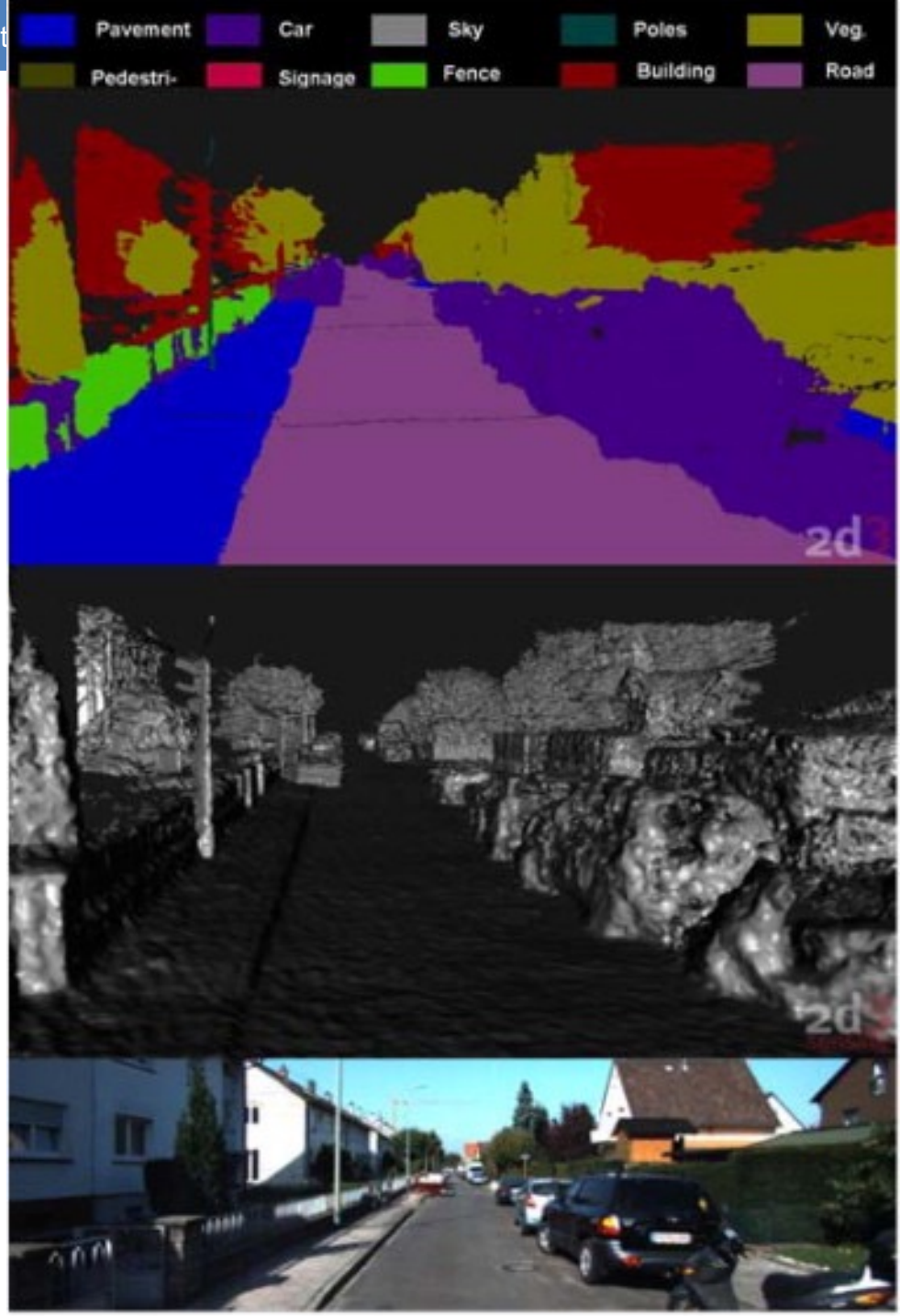
Stereye: Mesh Simplification

- Plane recognition via plane growing (add points with similar normals)
- Plane contour via alpha shape algorithm
- Planar intersections to make the model tight
- Mesh via Ear Clipping algorithm

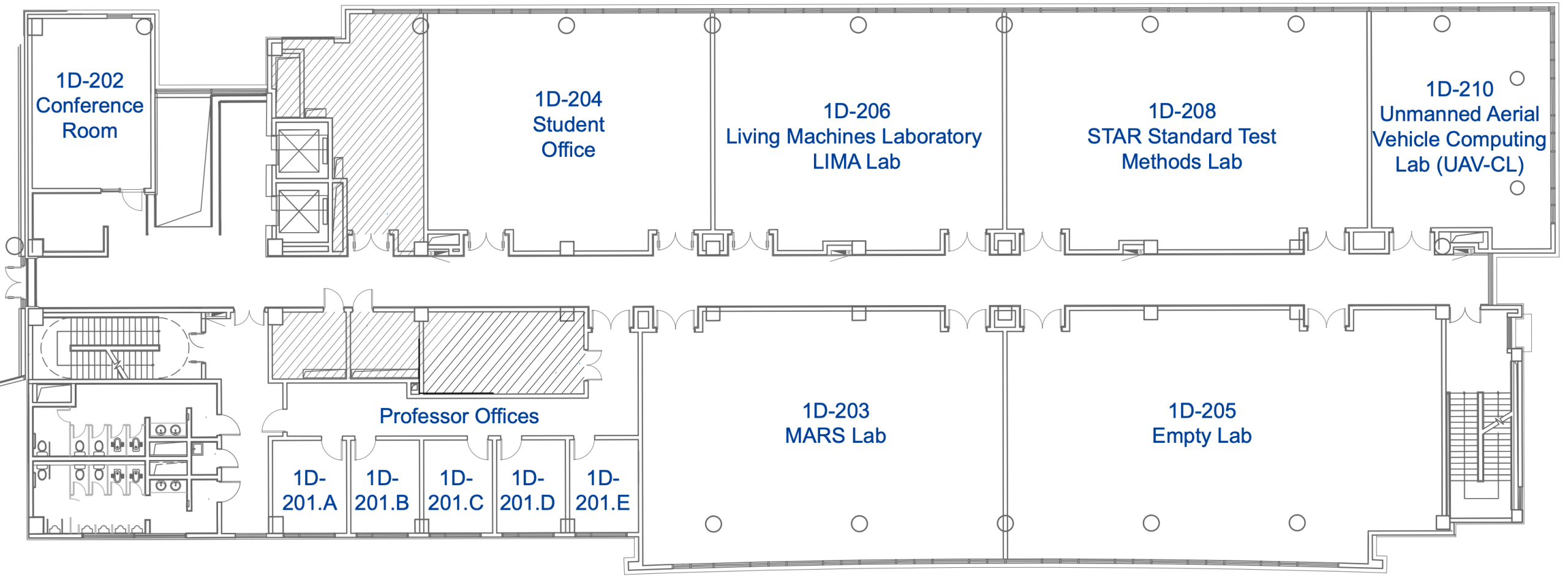
Semantic Map

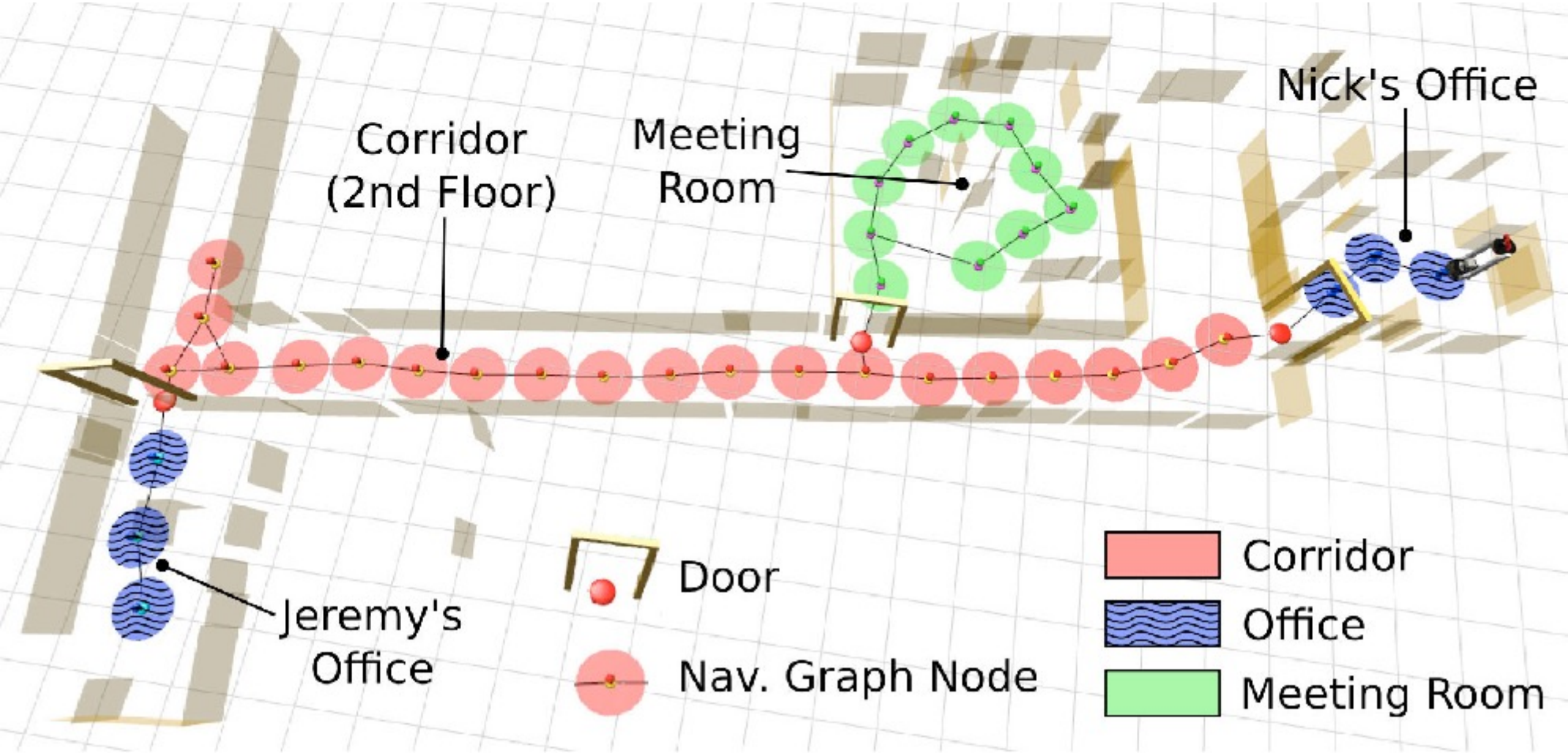
- Semantic Segmentation
  - In room (e.g. detect furniture)
  - Outdoors





Semantic Annotation: Room Names




Corridor
(2nd Floor)
Meeting
Room
Nick's Office
Jeremy's
Office
Door
Nav. Graph Node
Corridor
Office
Meeting Room

Semantic Information

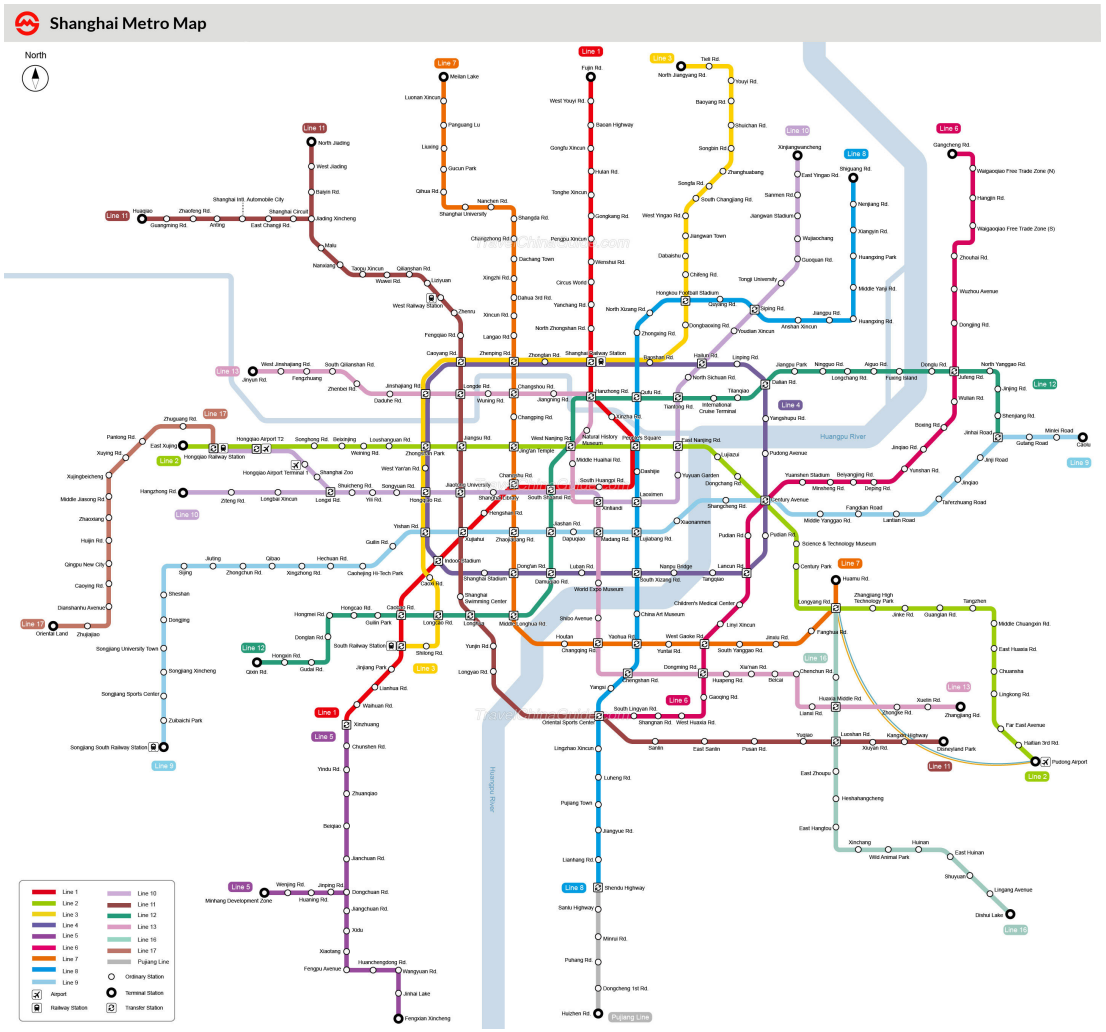
- Assign labels to data
- Segmentation: automatically group data (e.g. points) according to their semantic class

- Even save just very high level data; e.g. room at (x,y); Eiffel tower; …

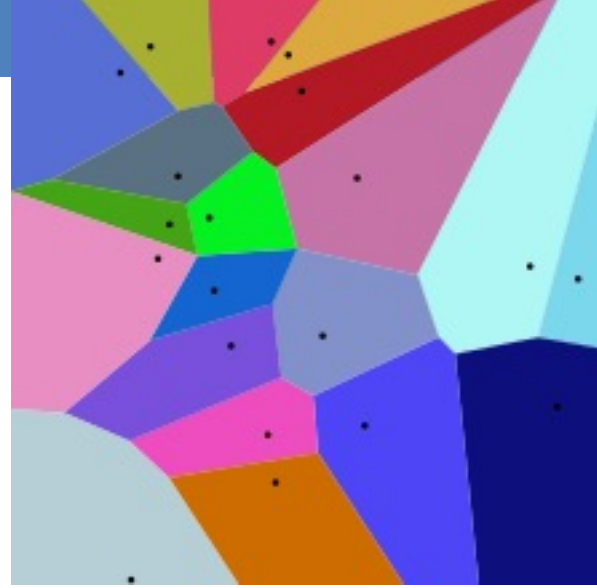
- Applications:
  - Human Robot Interaction (“go to kitchen”)
  - Scene understanding
  - Navigation (detect road; detect door)
  - Localization
  - …

Topologic Map

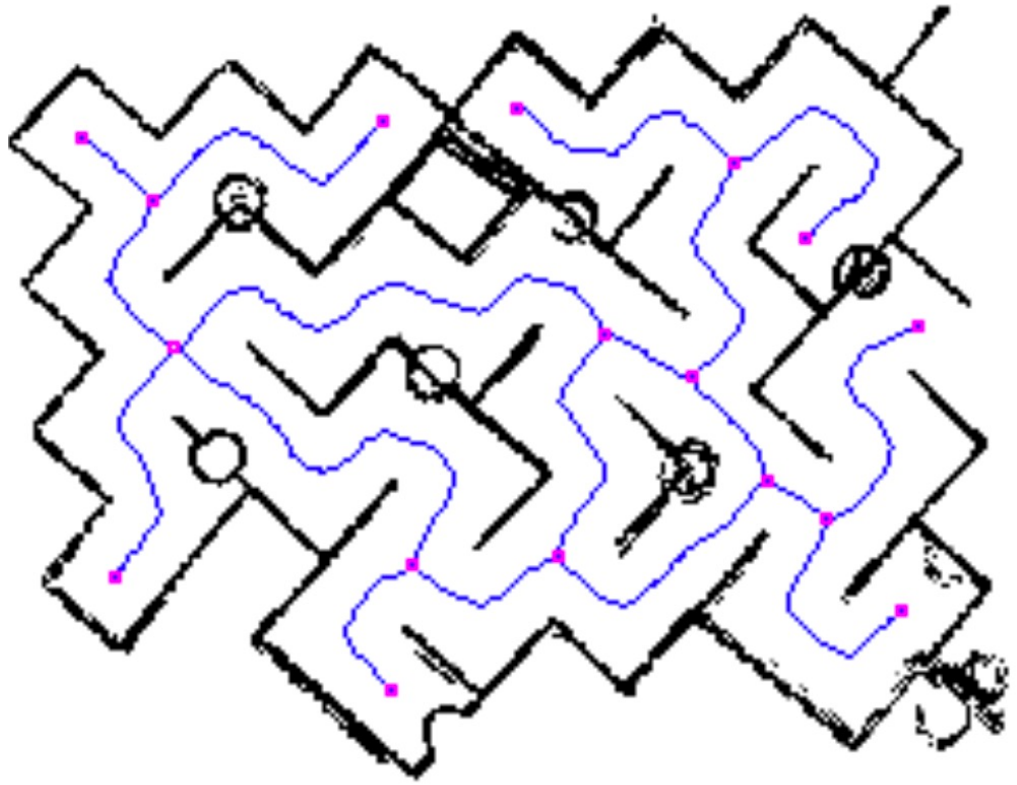
- A (undirected) graph
- Places (vertices) and their connections (edges)
- E.g. subway map of Shanghai: stations (vertices) and lines (edges)
- Do not have coordinates
- Topometric map: vertices and/ or edges are attributed with coordinates
- Very abstract – e.g. no obstacles anymore



Voronoi Diagram (-> Topology Graph)

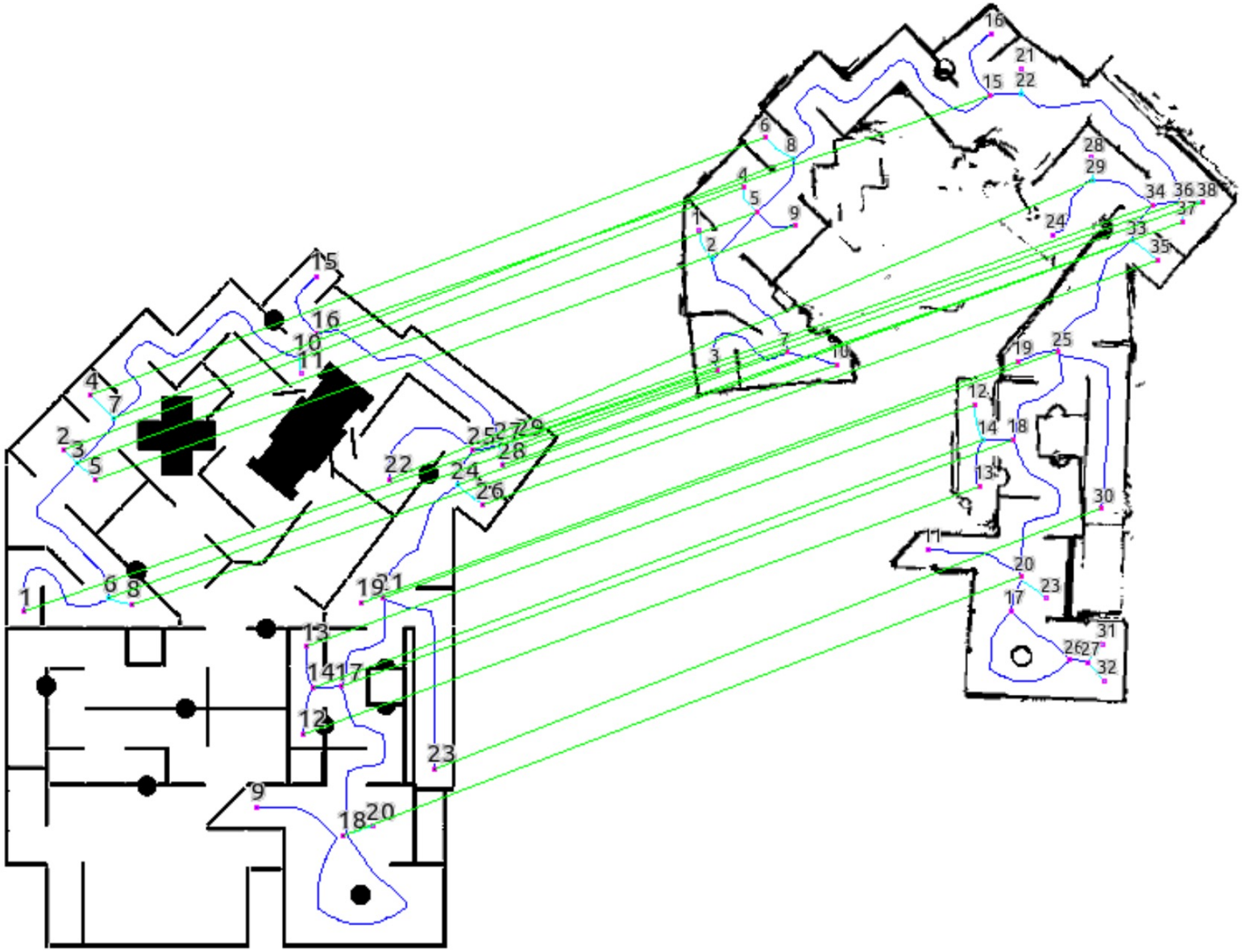


- Voronoi Diagram (VD): partition space such that edge is always equidistant to 2 closest obstacles.
- Topology Graph: vertices at junctions and dead ends
- Applications:
  - Very fast path planning
  - Human robot interaction (follow corridor, then go left)
  - Map matching

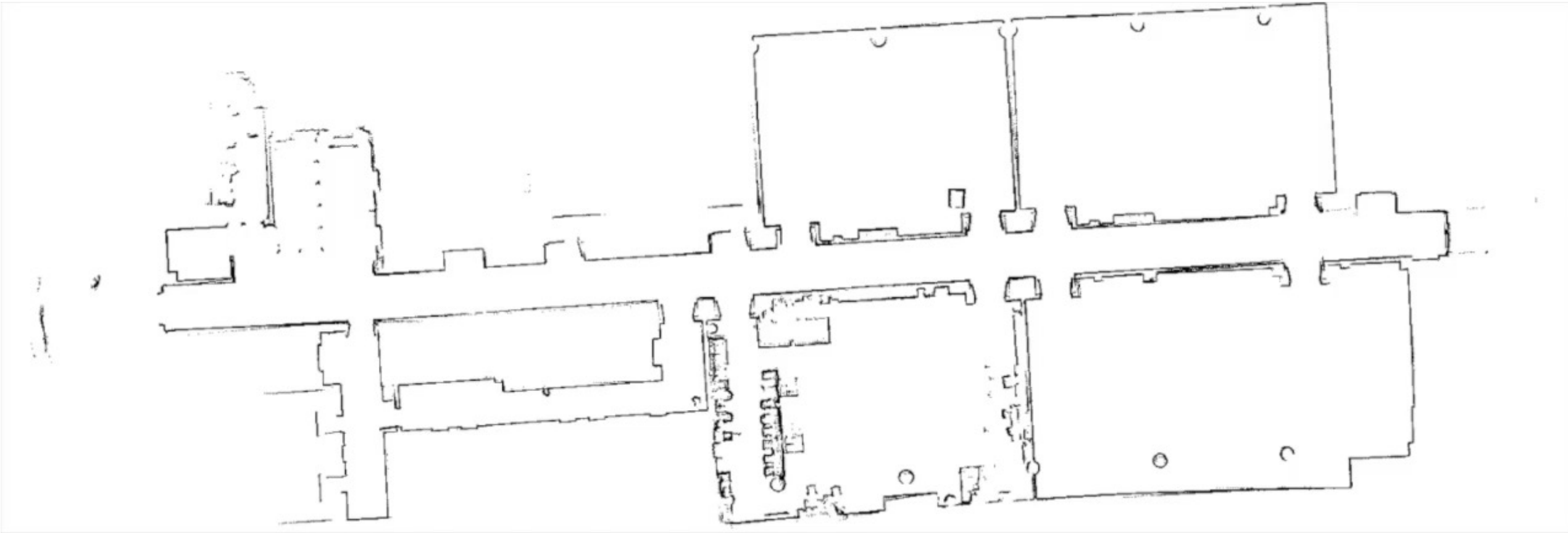


Map Matching

- Of 2D grid maps based on Topology Graph
- Left: ground truth map
- Right: Robot generated map
- RoboCup Rescue environment!

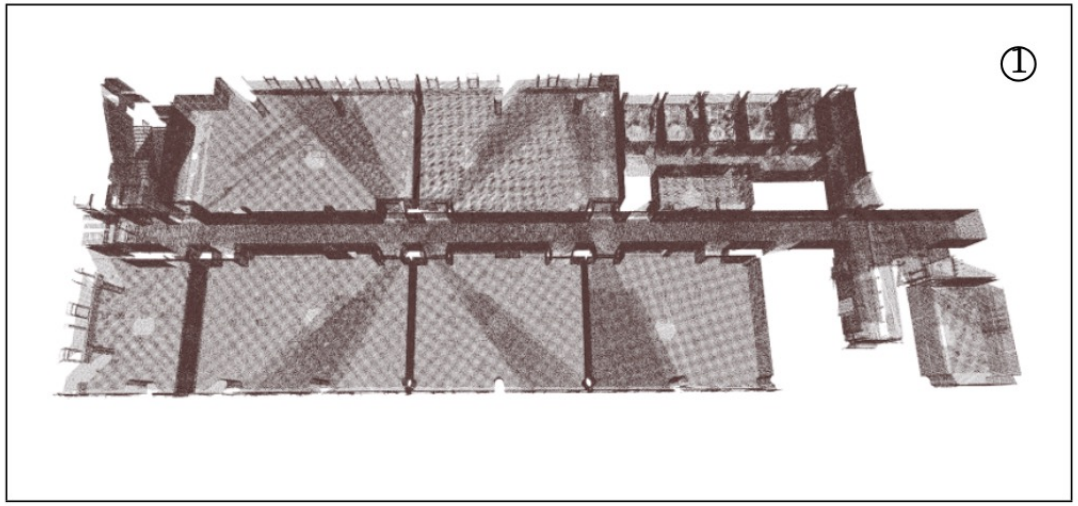


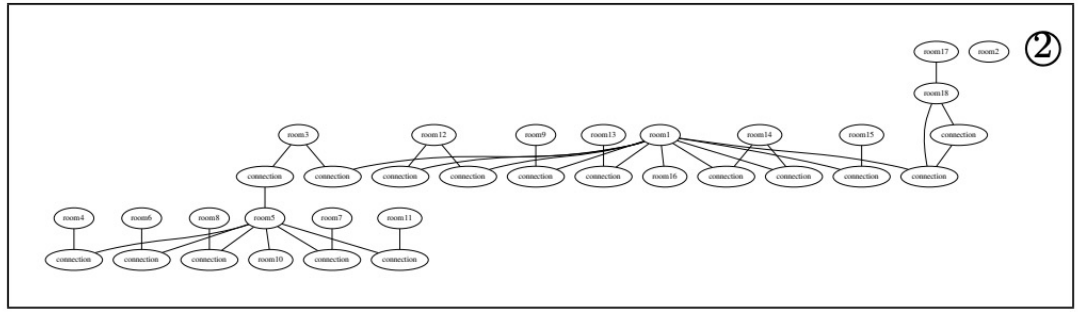
Area Graph: from 2D Grid Map to Topology Graph

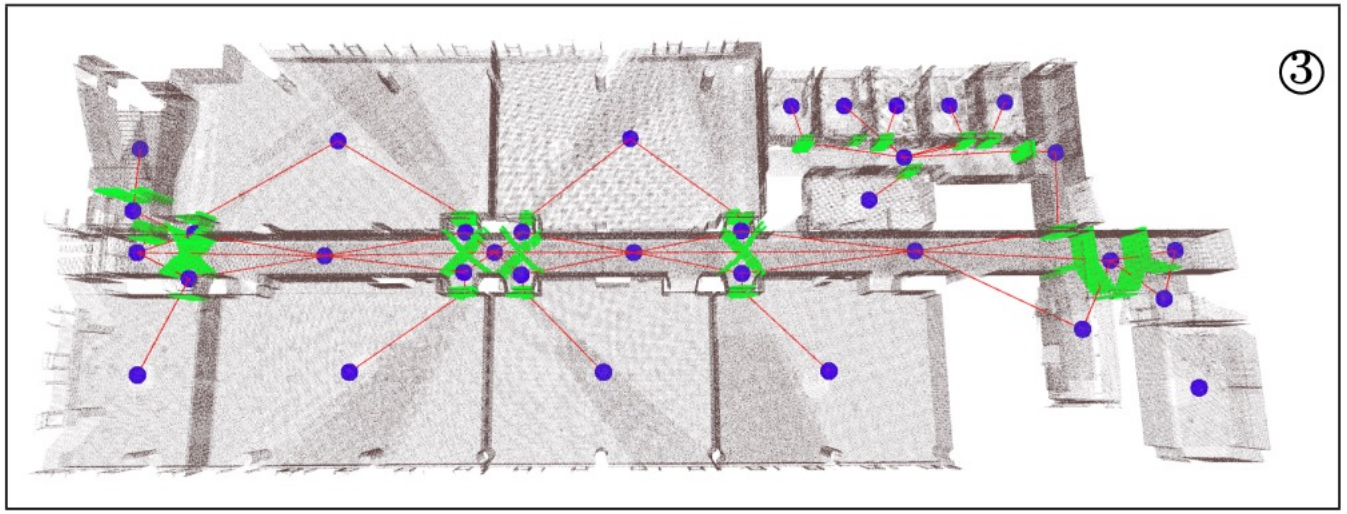


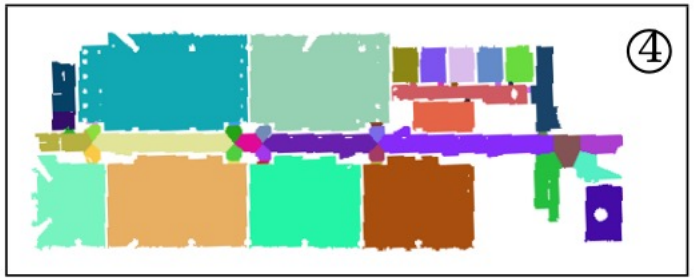
Topological Map in different Dimensions

- (2): 0D; (3): 1D; (4): 2D; (5): 3D









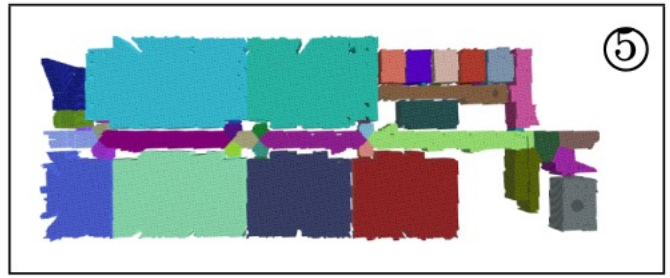
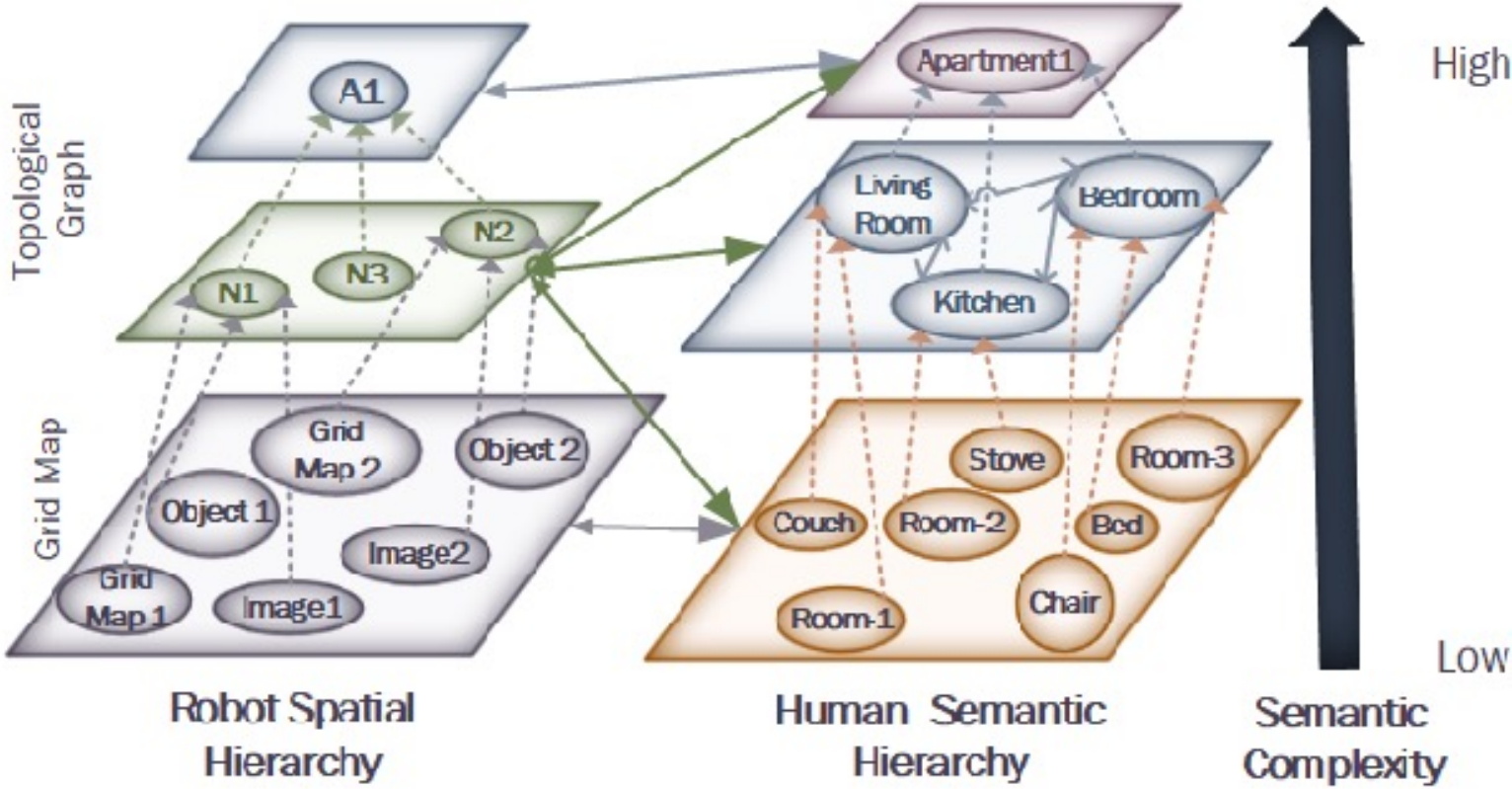


Fig. 1: Input 3D point cloud of 4 floors of a building. Below: results of our algorithm in different dimensions. From top to bottom, left to right are: 0D, 1D, 2D, 3D

Hierarchical Maps

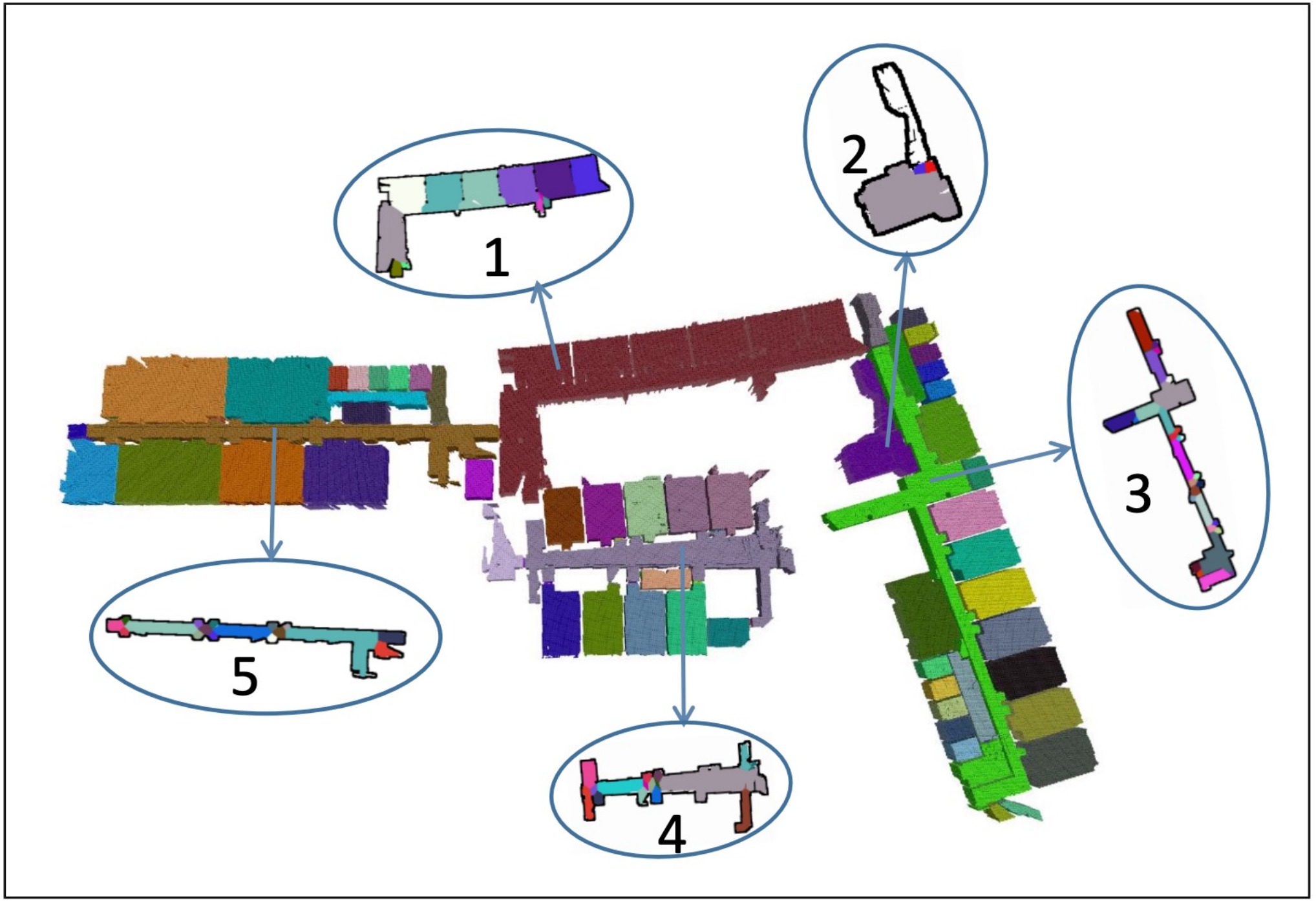
- Higher abstracted maps that contain lower ones with more details



- E.g. Grid map & Topological

- Useful for very fast planning; Human Robot Interaction; …

- Another form: image pyramid in GIS (different scale images)


1
2
3
4
5

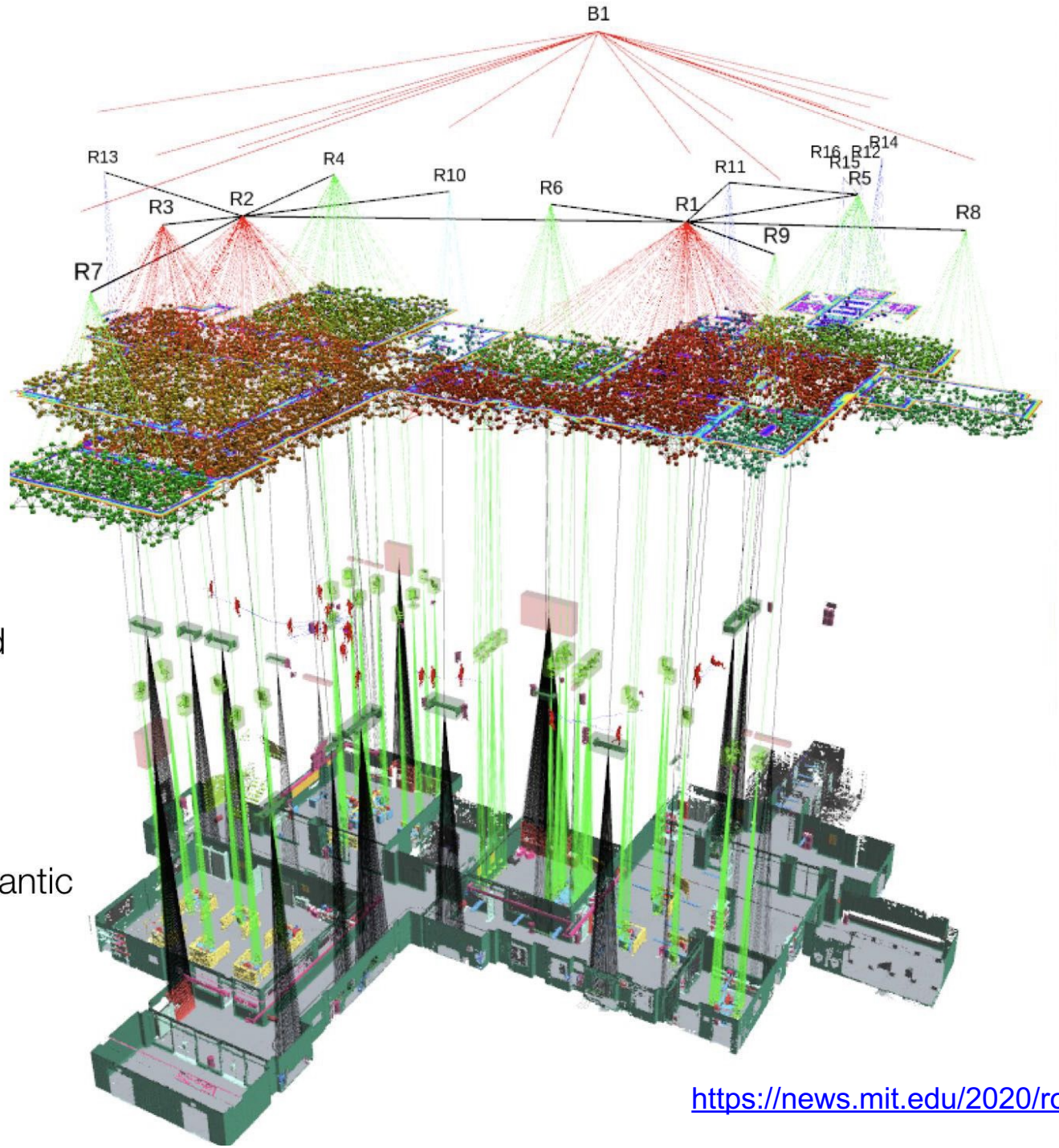
Layer 5:
Buildings

Layer 4:
Rooms

Layer 3:
Places and Structures

Layer 2:
Objects and Agents

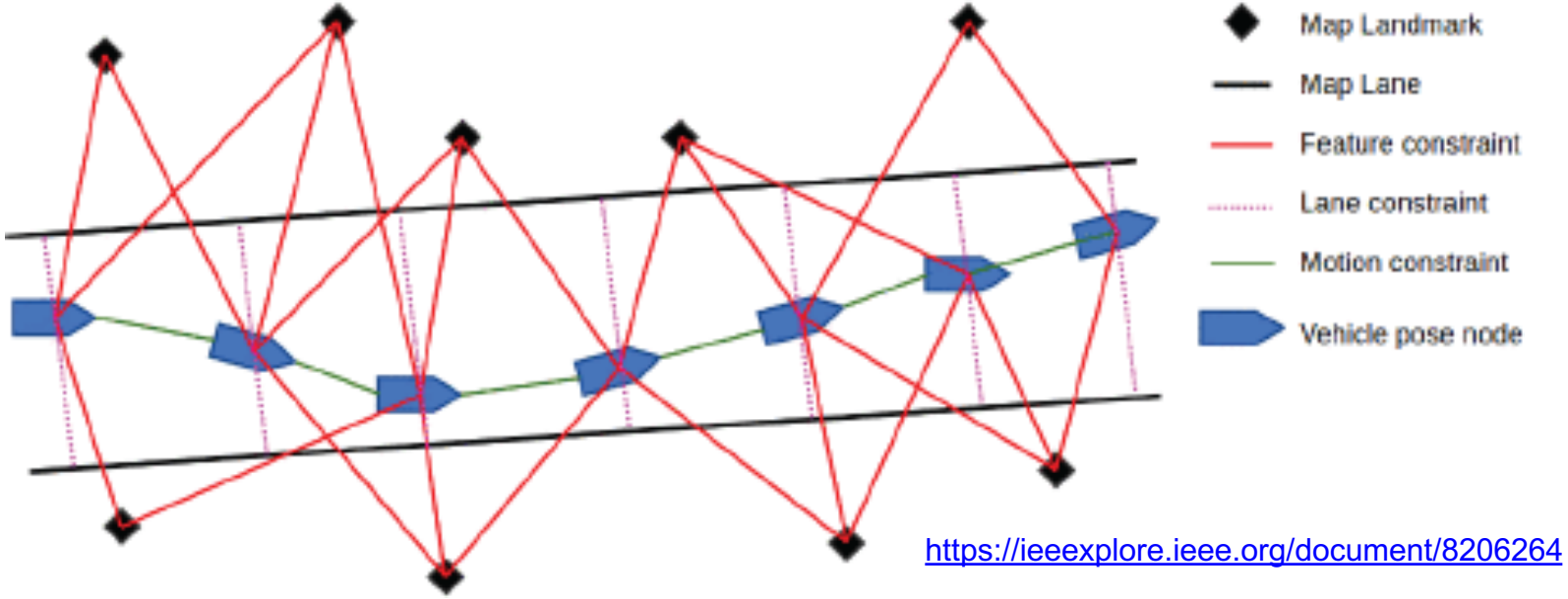
Layer 1:
Metric-Semantic Mesh



https://news.mit.edu/2020/robots-spatial-perception-0715

Pose Graph

- Graph structure
- Nodes are:
  - Robot
  - Landmarks/ observations



https://ieeexplore.ieee.org/document/8206264

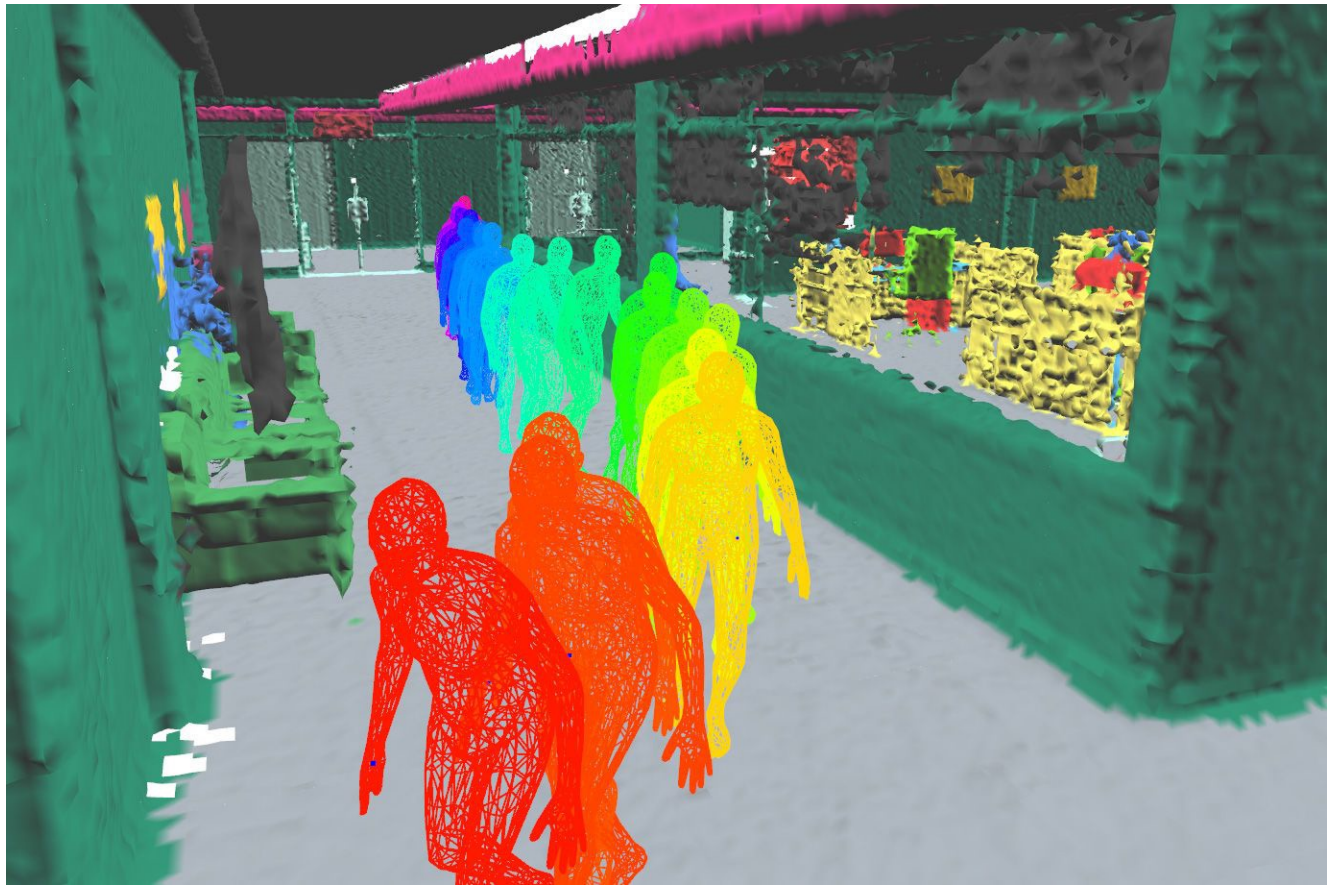
- Used for Simultaneous Localization and Mapping (more details later in course)

- Typically saves (raw) sensor data in robot nodes =>
- For most applications: needs to be rendered before using it:
  put all sensor data in common frame in a point cloud or grid map or plane map
  or ...

Dynamic Map

- All map representations above assumed a static environment: nothing moves
- Dynamic Map: capture moving objects (e.g. cars, humans)

- E.g: 3D Dynamic Scene Graphs:
- enables a robot to quickly generate a 3D map of its surroundings that also includes objects and their semantic labels
- Some can be dynamic (can move)



https://news.mit.edu/2020/robots-spatial-perception-0715

Other Models of the Environment/ Map Representations

- Many different possibilities:
- A set of images
- All kinds of sensor data (e.g. smoke map; noise map; radiation map)
- Heat map (infrared readings)
- …

Mapping

- Process of building a map
- Basic principle:
  1. Initialize the map with unknown or free
  2. Take a sensor scan
  3. Maybe pre-process it (e.g. plane detection)
  4. Localize the robot w.r.t. the map frame (maybe difficult!)
  5. Transform the (processed) sensor scan to the global frame
  6. “Merge” the new data with the old map data, e.g.:
     - Add scanned points to map point cloud
     - Update cells in a probabilistic occupancy grid
  7. Sometimes: Also do ray-casting to mark all cells from sensor to obstacle as free
  8. Repeat for every new sensor scan

- Localization step may need the map (e.g. matching the scan against the map) => both should be done at the same time =>
- Simultaneous Localization and Mapping : SLAM