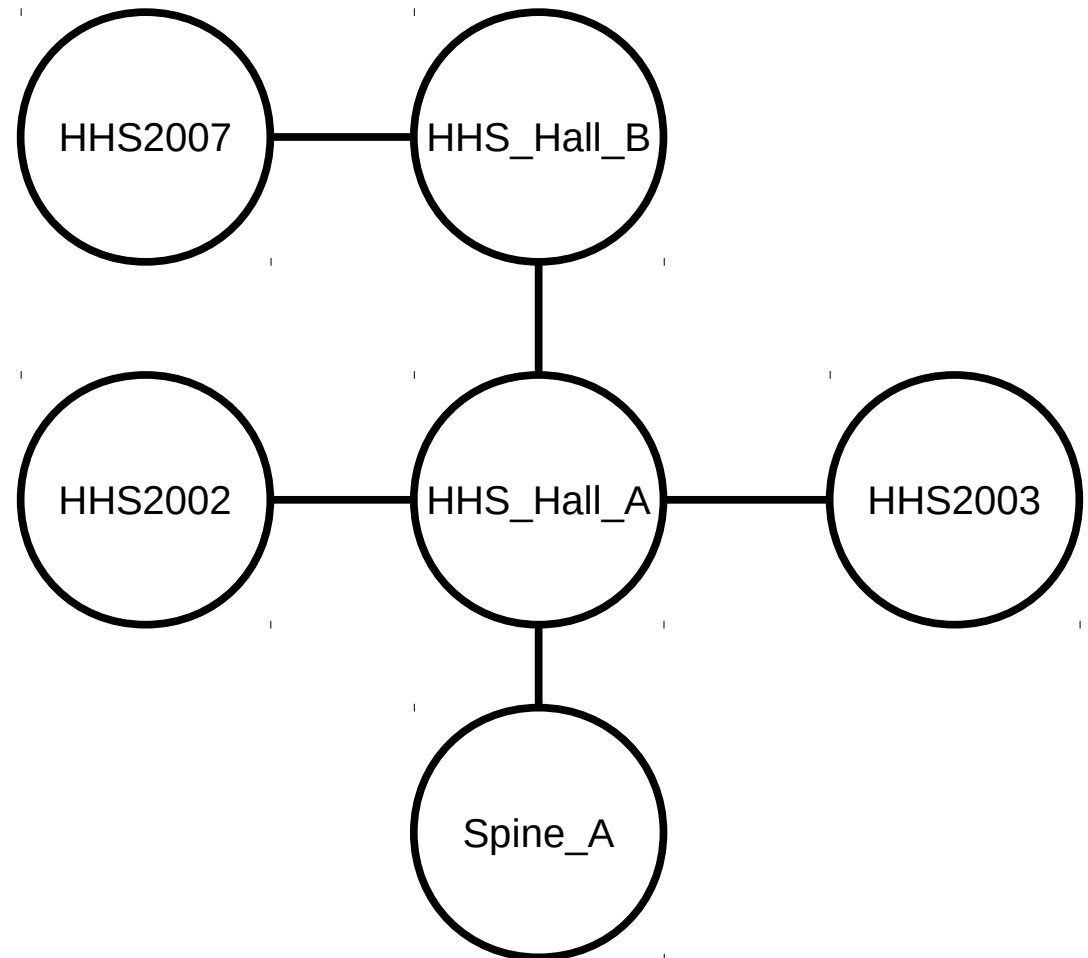


CS354



Representing Maps: Topological

- Represent relative locations using a graph structure.
 - + Good for high level navigation
 - Difficult to build autonomously
 - Not good for low-level localization and navigation



Representing Maps: Geometric Landmark Based

- Store the geometric location of recognizable landmarks.
 - Maybe artificial beacons or markers.
 - Maybe distinctive environmental features.
- + Memory-efficient
- + Allows precise localization
 - Landmark mis-identification can cause problems
 - May not be ideal for identification: only landmark positions are stored, not necessarily the positions of all obstacles

Representing Maps: Occupancy Grid

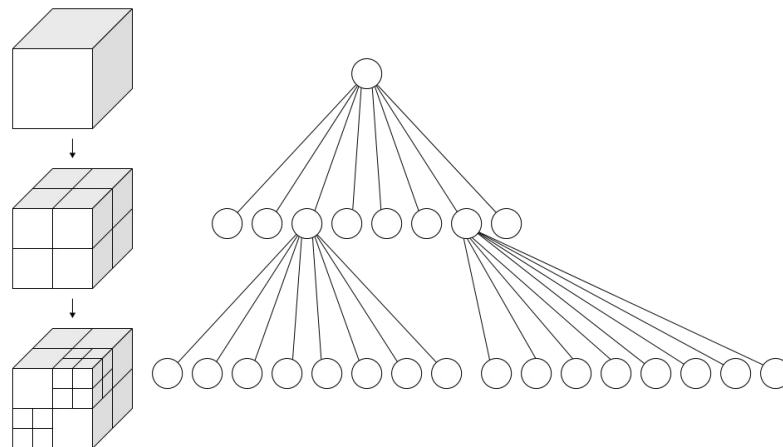
- Divide the environment into grid cells, maintain an “occupied” probability for each cell.
 - Memory intensive (particularly in 3D)
 - + Good for navigation
 - + Good for localization
 - + Relatively simple to create autonomously

Quadtrees/Octrees

- Large occupancy grids can be expensive to store:
 - 100m x 100m map, 1cm resolution
 - 100,000,000 cells
- Quadtree is a more space-efficient alternative...

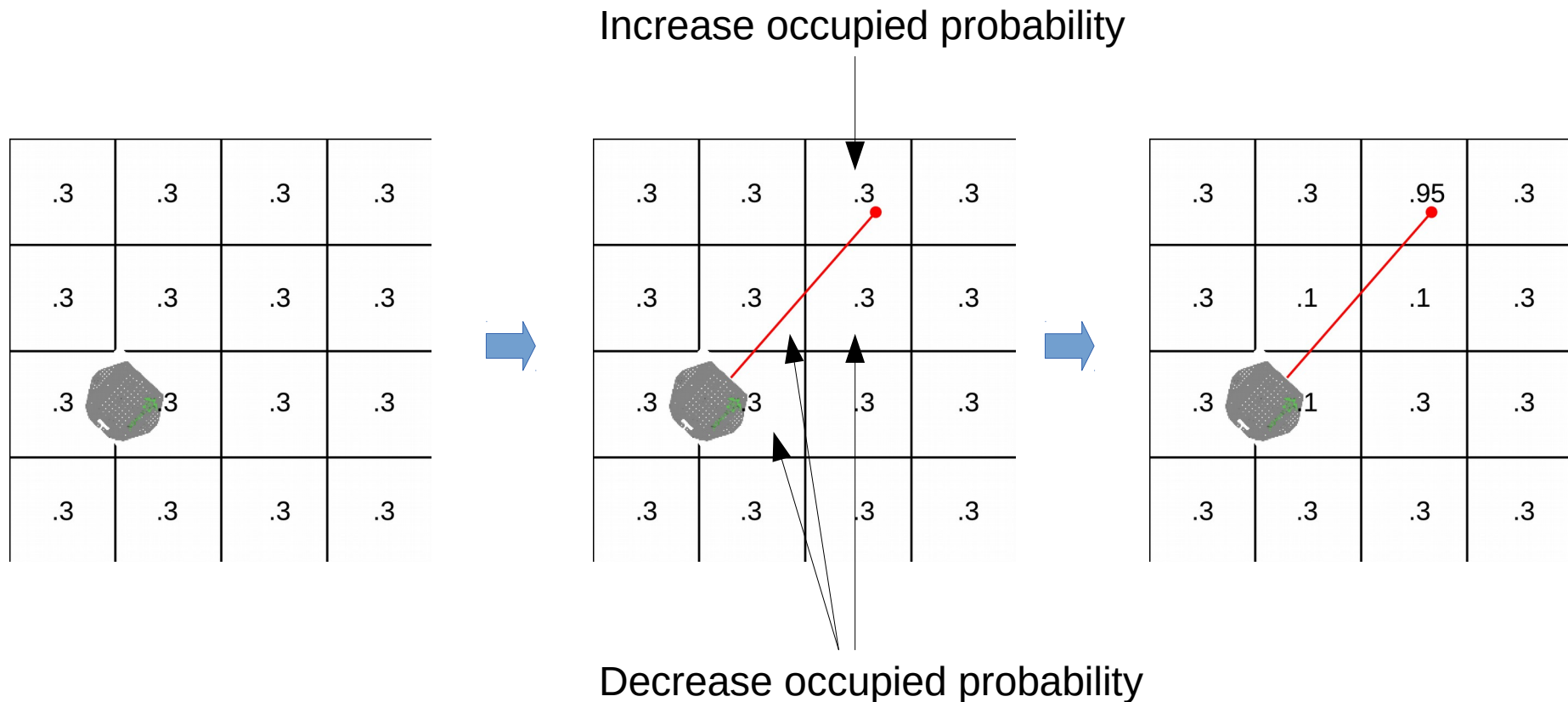
Quadtrees/Octrees

- Large occupancy grids can be expensive to store:
 - 100m x 100m map, 1cm resolution
 - 100,000,000 cells
- Quadtree is a more space-efficient alternative...
- Octree is the 3d generalization:



Mapping w/ Occupancy Grids

- Relatively easy if we know the robot pose:



H.P. Moravec. Sensor fusion in certainty grids for mobile robots. AIMA Magazine, pages 61–74, Summer 1988.

SLAM – Simultaneous Localization and Mapping

- Recall the localization problem:

$$P(\mathbf{x}_t | \mathbf{z}_{0:t}, \mathbf{u}_{0:t})$$

- The SLAM problem is reassuringly familiar:

$$P(\mathbf{x}_t, \Theta | \mathbf{z}_{0:t}, \mathbf{u}_{0:t})$$

- Where Θ represents the map.
- Before we wanted a probability distribution over all possible robot poses.
- Now we want a joint probability distribution over all possible robot poses and all possible maps.

“Distribution over possible maps” is not as manageable as “distribution over poses”

SLAM “Solution”

- Prediction:

$$Bel^-(\mathbf{x}_t, \Theta) = \int P(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{u}_t) Bel(\mathbf{x}_{t-1}, \Theta) d\mathbf{x}_{t-1}$$

- Correction:

$$Bel(\mathbf{x}_t, \Theta) = \eta P(\mathbf{z}_k | \mathbf{x}_t, \Theta) Bel^-(\mathbf{x}_t, \Theta)$$

(Extended) Kalman Filter SLAM

- Most appropriate for landmark-based map.
- Let's look at the **example from the textbook**.
- Problems:
 - Not clear how to use this for occupancy grids
 - The covariance matrix gets big as the number of landmarks grows

Particle Filter SLAM: Problems

- Covering the space of possible poses and maps with particles is not practical:
 - “Pose particle”: 3-6 dimensions
 - “Map particle” for a (tiny) 10x10 grid: 100 dimensions
 - Joint map x pose particle: 300-600 dimensions

Rao-Blackwellized Particle Filter

- Solution/Approximation:
 - Each pose particle has an associated map.
 - Each map is updated under the assumption that its particle represents the correct pose.
 - The map may be landmark-based or occupancy grid-based.
- The ROS gmapping package uses this approach
<http://www.openslam.org/gmapping.html>