Chapter 8: Maps and related tasks

Harm Dermois Joris Stork

January 23, 2011

Robots and maps

Sensorial maps

Image based mapping

Spacial occupancy representations

Geometric maps

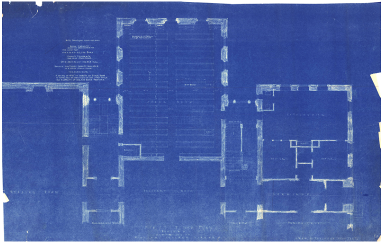
Topological Maps

Multiple Robots

Maps for robots

Human maps:

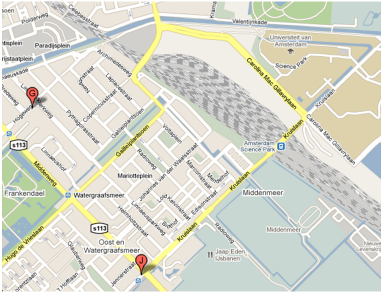
- Often unavailable;
- Often incomplete:
  - human relevant data;
  - robot relevant data;



Maps for robots

Human maps:

- Often unavailable;
- Often incomplete:
  - human relevant data;
  - robot relevant data;
- Wrong level(s) of abstraction: human oriented;



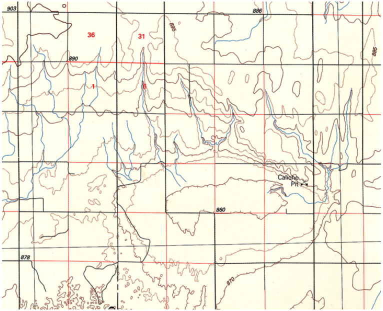
Maps by robots

- Building a robot friendly map is very difficult and tedious.
- Robots are good candidates to build maps with and for their own sensory suite;

Conclusion: Design robots to autonomously construct, update and validate maps destined for robot use.

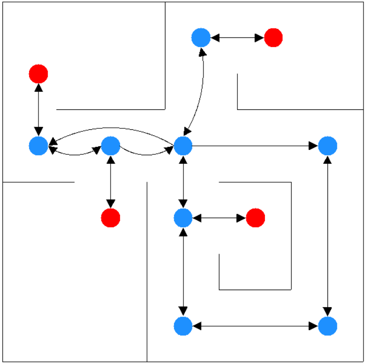
Map paradigms: metric vs. topological

- Metric
  - Sensorial
  - Geometric



Map paradigms: metric vs. topological

- Metric
  - Sensorial
  - Geometric
- Topological
  - Local relational
  - Topological
  - Semantic



Direction of map hierarchy

- Giralt et al.: metric to topological.
- Kuipers and Levitt : topological to metric. Low level topological landmarks as starting point.

Types of data

- (Derived) spacial occupancy;
- (Direct) sensor measurements in relation to position. e.g. olfaction.

Sensorial maps

- Represent sensor measurements against odometry;
- Collection of measurements: $[I_i(x_i, y_i, \theta_i)]$

Image based mapping

Harm Dermois, Joris Stork

The challenge:

- how to sample the set of possible measurements, $\{I_i\}$;
- how to turn the samples into a continuous I.

Li: street panoramas

Li et al.: robots builds graph representing street network:

- edges = streets;
- nodes = intersections;

Robot collects by:

- moving in a closed loop, always turning left;
- recording panoramas of left and right sides of streets;
- concluding a loop by identifying previously recorded street side;

Bourque: robot sightseeing

Bourque et al.: robots builds graph nodes corresponding to panorama shots, in a less constrained environment, by:

- choosing sample (panorama) points based on models of human attention;
- using "alpha backtracking" to make trade-off between distance to next sample point and optimality of next sample point;

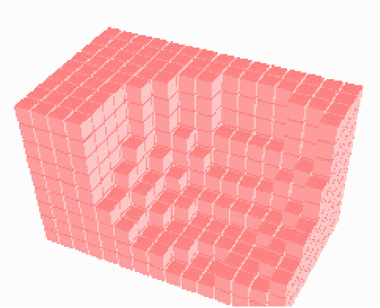
Spacial occupancy grid

Pioneered by Elfes and Moravec

- Grid of pixels

Spacial occupancy grid

- Grid of pixels
- Volume of voxels



Spacial occupancy: data represented

- Fill pixels / voxels with degree of occupancy data.
- More refined: fill pixels / voxels with probability of occupancy data.

Spacial occupancy: probabilistic approach

- Example of a laser sensor: probability of an actual distance z for a given laser reading r computed using Bayes' theorem:

$$P(z|r) = \frac{P(z)P(r|z)}{P(r)}$$

Spacial occupancy: probabilistic approach

- Example of a laser sensor: probability of an actual distance z for a given laser reading r computed using Bayes' theorem:

$$P(z|r) = \frac{P(z)P(r|z)}{P(r)}$$

- Generalised:

$$P(W|R) = \frac{P(W)P(R|W)}{P(R)}$$

with $P(R_i) = \sum_j P(R_i|W_j)P(W_j)$

Spacial occupancy: probabilistic approach

The result: *maximum a posteriori* (MAP). World model that most reasonably estimates environment according to Bayesian approach.

Spacial occupancy: probabilistic approach

The result: *maximum a posteriori* (MAP). World model that most reasonably estimates environment according to Bayesian approach.
Considerations:

- very general: no assumed model, deals with multiple sensors;
- requires accurate probabilistic model of the sensors;
- requires a lot of memory for the occupancy map;
- measurement locations/times discarded: geometric accuracy reduced;
- important to avoid accumulated positional errors - e.g. by iteratively recomputing position;
- needs an exploration policy: e.g. random or towards "unkown' 'areas.

Spacial occupancy: Markov models

Markov localisation: estimating robot's location based on sensor data by maintaining probability density grid for the robot's environment, with each cell representing a possible robot pose.

Geometric maps

Accurate, with two assumptions:

- sensor data is suitable
- environment is suitable

Geometric maps: exploration

Challenge is exploration. Includes searching for:

- a goal position;
- route with specific properties;
- "covering" a space;
- occupancy.

Geometric maps: reach goal

Papadimitriou and Yannakakis's bug-like algorithm for reaching known goal from known origin in unkown environment with obstacles:

- move "towards" line connecting origin and goal;
- if not possible, move in arbitrary direction;

Useful in certain simple types of environment, notably where obstacles are:

- rectilinear;
- nonintersecting;
- aligned with world coordinates.

In more general environments no bound is possible.

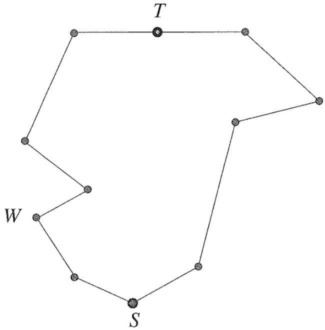
Geometric maps: geometric representations

Chosen geometric representation influences applicable algorithms. Important representation is that of "street polygons".

Geometric maps: Street polygons

Polygon such that:

- there is a start vertex S and end vertex T
- vertices and lines categorised as "left" or "right" with respect to line segment from S to T;
- every vertex on either side is visible to some vertex on the other



Geometric maps: Street polygons

Topological Maps

Topological maps: describe the environment as a graph that connectecs specific locations in the world and represents them as vertices.

1. Because metric representations cost too much memory to maintain in the long run.
2. Easy to understand for humans.
3. The nodes on the graphs are landmarks or features of the environment. The edges are paths between the different nodes.
4. Landmarks can be artificial or natural.
5. Landmarks can look the same so you need to make sure you dont use two or more nodes to represent the same landmark.

Marker based exploration

1. No prior information about the environment available.
2. Can be extended by enumerating the edges incident to the node entered. Edge you traveled along is 0 and enumerate clockwise. This enumeration is local cause it depends on the edge the robot moved over.
3. Landmarks aren't distinguishable from eachother.
4. The robot needs to have something to mark where it has already been.(spray paint, bread crumbs)
5. Use unique marks which it can pick up, drop and recognize.

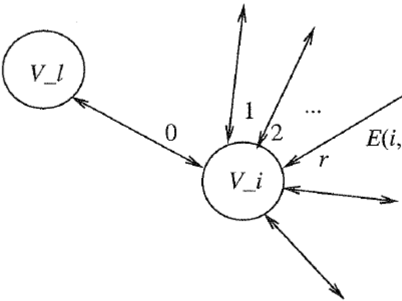
Marker based exploration algorithm

1. Builds up the known graph by traveling along the incident edges.
2. v_i is the node where the robot is currently at. v_j is the node where the robot is moving to. $E_{i,j}$ is the edge between the 2 nodes.
3. Transition function need to follow these properties. If $(v_i, E_{i,j}, r) = v_j$ and $(v_j, E_{i,j}, s) = v_k$, then $v_j, E_{i,j}, -s) = v_i$
4. Moves are invertible and can be retraced.
5. $t \neq -s$ then $v_j, E_{i,j}, -s) = v_i$ and $(v_j, E_{i,k}, s) = v_j$ are not valid. To avoid redundant and degenerate paths.

Marker based exploration algorithm(2)

Operations for the robot

1. r stands for move along the given edge.
2. Each marker can be in 3 different states[pickup,putdown,null]
3. Marker based perception: At each vertex the robot can see two things [present, not-present]

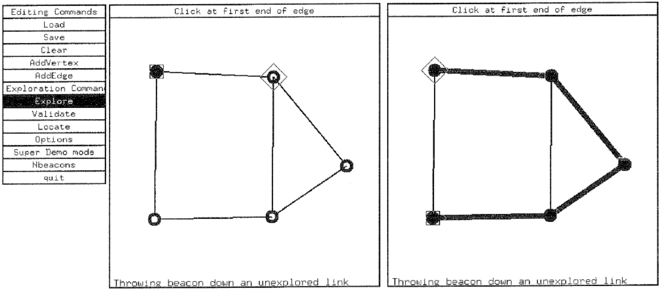


Marker based exploration algorithm (3)

Operations for the robot

1. Robot can determine the relative posotions of the edges by enumerating the edges like said before.
2. Entering the same vertex from a different edge gives 2 different ordering. The robot needs to make a global ordering.
3. Subgraph S for explored edges and U for unexplored are incident to unknow nodes.
4. First validate all explored nodes. Make sure there aren't any doubles by looking for markers.
5. If there is no marker found at a certain node v add it the subgraph S and add the edge which was taken aswell.
6. Enumerate all edges incident to the new node and add them to U.
7. Do this till subgraph U is empty.

Example



Editing Commands
Load
Save
Clear
AddVertex
AddEdge
Exploration Comman
Explore
Validate
Locate
Options
Super Demo mode
Nbeacons
quit
Click at first end of edge
Throwing beacon down an unexplored link
Click at first end of edge
Throwing beacon down an unexplored link

Multiple Robots

Why would you use multiple Robots

1. **Improved Robustness**: A multirobot can ,in principle, keep functioning even if one indiviudual robots dail completely.
2. **Improved effiency**: It is possible for a group of robots to accomplish a search or exploration task faster than an equivalent single robot.
3. **Alternative Algorithms**: For some tasks, the availability of multiple robots allows feadible or guaranteed algorithms to be implemented when no such algorithm is available for a single robot system.

Multi Robots in Practice

Problems

1. **Where are the other robots?**: Rendezvous with other robots
2. **Partitioning**: Finding a good way to distribute the work amongst the robots.
3. **Multi-robot planning**: Prevent the trajectories of the robots to collide.
4. **Merging the data from the indivual team**:

Rendezvous

Rendezvous: is having two or more robots meet at an appointed place and time.

1. Rendezvous is needed for robots that can only communicate in close proximities, but may also be needed to exchange objects between robots.
2. When Multiple robots try to complete a task collaborotavely without prior knowledge. They need to to exchange information while they are still working at the task at hand.
3. If they dont meet they cannot benefit from what others have already learned.

Too many Rendezvous

Problems: Robots mustn't devote too much energy to rendezvous

1. The extent to which the two robots agree on their perceptions of the environment. What is the difference
2. The degree of synchornization of the robots can attain expressed as the likelihood that an appointed rendezvous at a common location will fail owning to a failure to arrive at the same time
3. The extent Of the commonality between the region of space the robots have explored.

There are many different rendezvous algorithms

1. **plan based**:
2. **stochastic algorithms**: One stationary and one seeks or Randomly visit rendezvous points which are points or interest.

Map fusion

Map fusion: is needed when the problem doesn't involve foraging.

1. Is needed to make the collaborative effort worthwhile.
2. Complexity of the map-merging depends on the , Odemetry error, the fidelity of the sensing used. and the richness of the evironment.
3. Fusing maps using cross correlation depends on teh fact that the individual maps overlap "sufficiently'.
4. Done by rotation and translating of the given maps.

Exploration with Multiple robots

1. Robots are only allowed to commnicate when they are in the same node.
2. Split all the work between all the robots and have them explore their own part of the graph.
3. Plan rendezvous, to harmonize the information they got till then and make a single consistent representation of the environment they are in.
4. Redevide the work and repeat this till everything is known.

links

http://www.csupomona.edu/~ftang/courses/CS499/notes/navigation3.pdf