COMP417
Introduction to Robotics and Intelligent Systems
Global Localization: MDL
Global Localization

• Recall, incremental localization (state estimation) refines a starting guess.
• Global localization start *ab initio* (i.e. without a good prior).
• Analogy with function minimization.
• Localization problems (as usual) presupposes a map and find a position.
• This algorithm is about action selection: probabilistic filtering can be used in parallel.
Mobile Robot Pose Estimation Revisited

- Recall, to answer the question: where am I?
  
  directly analogous to global vs local function optimization

  A. **Globally**, assuming I could be “anywhere”.
  
  B. **Locally**, assuming I know I am in a specified region but want an accurate position fix.

Almost all robotics deals with the local problem!
Overview

• Introduction
  – Minimum Distance Localization (MDL)
    • Localization with efficient travel (LET).

• Part 1
  – Complexity result, Deterministic Algorithm

• Part 2: Randomized Algorithms
  – Common Overlay Localization (COL) Algorithm
    • Useful Region Localization (URL) Algorithm

• Experimental Evaluation
Problem Statement

• Localize a robot with no prior position estimate.
  – a) How difficult is this problem, formally?
  – b) What is an efficient algorithm?

• "Global Localization" [Book Sec 8.4]
  – Also known more informally as the kidnapped robot problem.
How hard the the problem?

- In general, observations from a single position may be ambiguous.
- To uniquely localize, most move around and take more observations.
- Complexity measure: what is the minimum amount of travel needed to localize, in the worst case?
Formalism & Assumptions

• To consider the *feasibility* of the global problem, we start with an idealized model:
  1. 2D world
  2. Point robot
  3. Assume the environment is a polygon P without obstacles
  4. Assume our robot has a perfect map
  5. Assume our robot has a perfect range sensor with infinite maximum range.
  6. Perfect compass (known orientation)

• The region seen by the robot at any time is a **visibility polygon**
  – The key observation is the set of vertices visible at any time.

• We can divide the environment into regions within which we seem the same set of vertices: **visibility cell decomposition**.
Example

- Intuition: you wake up in a hotel room with memory of getting there, what room is it?
  - Check what end of the hall it is.
  - Check what floor it is.

- Robot example:
  - System gets rebooted.
  - Range data from current location is ambiguous.
  - Where are we?
  - More around to reduce pose uncertainty.

  • Saw some examples of this already in this session.
What if no approximate pose estimate is available? (i.e. a uniform prior)

- Find the best match of our observations over the entire known map.

- Problem: Observations from a single viewpoint may be ambiguous: two offices may look alike.
  
  This is a risk even with perfect noise-free maps & sensors.

- We may need to combine multiple observations to determine our position uniquely.

Positions A and B are indistinguishable without moving into the “hallway” above.

Common visibility polygon.
Minimum Distance Localization

- Mobile robot placed at an unknown location inside a polygonal environment $P$.
- Robot has a map of $P$ and can sense its environment.

Objective:

- Minimize distance traveled while uniquely determining initial location.
Problem statement

• Determine where we are.
• But the problem is not really what are my precise coordinates, but
  Which of several different hypothetical locations am in in?
• If we can resolve ambiguity, the precise position is trivial given our (idealized) assumptions.
• Localization in thus case is thus combinatorial, not metric.
Assumptions

- Robot’s equipped with a finite-distance range sensor.
  - Note: prior theoretical results assumed unbounded range.
  - In reality, with a noisy (i.e. real) sensor, the problem can only become more difficult.
- Sensed data consists of an ordering of vertices and edges seen by the robot (visibility polygon).
- Robot is assumed to be a point robot moving in a static 2D, obstacle-free environment.
  - Obstacles have not effect on the result, but make it more tedious.
- Robot is able to determine its orientation.
Basic Approach

• Determine set of *hypothetical locations* that match robot’s initial observations.
• Compute next sensing location from where distinguishing landmarks may be seen.
• Travel to this location and make observations that disambiguate the hypotheses.
• Eliminate incorrect hypotheses.
Theoretical Results

• **MDL** expressed as optimal localizing decision tree is NP-hard.
  • (Even with unbounded range.)

• **Precomputation:**
  – *Hypothesis Generation:* Computes the set of hypothetical locations matching the robot’s initial observations.
Competitive Ratio

• Ratio of solution quality to the optimal.

• Normally used for hard problems.
Results

– *Hypothesis Elimination*: Rules out incorrect hypotheses by directing the robot to travel to the nearest distinguishing visibility cell.

– Competitive ratio: $(k-1)d$, $k$ is number of hypotheses, $d$ is length of an optimal verification tour.

– Complexity: $O(n^6)$, $n$ is number of vertices in $P$. 
Hardness of MDL

- Even for a point robot in a polyhedral environment, and with perfect range sensor the global localization problem is NP-hard.
  - There is a reduction to abstract decision tree, a previously-established NP-hard problem.

- Can we simply use suitable heuristics to get good performance?
Approximating MDL

• A simple greedy algorithm:
  – visit the closest place that tells me something about where I am.
  – Next, visit the next closest place that tells me something.

This seems intuitive and has actually been used in practice.

This approach can be shown to be exponentially worse than optimal.
Approximating MDL accurately

• Polynomial time strategies with provably good performance can be constructed [Dudek, Romanik, Whitesides].
  – For $h$ alternative hypothesis about where we are, we can be sure the path length is never more than $(h-1)\text{optimal}$.
  – Cost of computing the solution is polynomial.
While these results are satisfying, they have 2 key shortcomings.

- The computational cost of computing a solution is high.
- The observation that need to be made may not be practical.
  - It may be necessary to visit very small cells that cannot be easily reached given realistic odometry errors.
Intractability of the Global Problem

Resolving ambiguity?

- We compute a decision tree that will allow us to optimally decide what to do if we find ourselves in an ambiguous location.
  - The height of this decision tree gives the length of the longest information-gathering path, and we seek to minimize this.

- We will refer to this problem of determining pose using a minimum length path through a known map as the global localization problem.
Limited Range
Another Approach: Randomized Algorithms

- Hypothesis Generation:
  - Environment $P$
  - range data $W$, bounded distance $d$
  - find all locations in map that match (i.e. "congruent" to $W$ [congruence, but with missing data]).
  - Later, find good, but maybe not optimal path.
Approach

- **Overlay Intersection (OI):** We consider only the connected component of the intersection region of the *overlay arrangement* that contains the origin since it is the area known to exist in all hypotheses.
Approach

Overlay Intersection (*OI*): We consider only the connected component of the intersection region of the *overlay arrangement* that contains the origin since it is the area known to exist in all hypotheses.
Approach

**Overlay Intersection (OI):** We consider only the connected component of the intersection region of the *overlay arrangement* that contains the origin since it is the area known to exist in all hypotheses.
Approach

**Hypothesis Elimination:** Weighted decision strategy.
- A set $R$ of random sample points is picked from $OI$.
- For each $r$ in $R$, $Value[r] = info/distance_{OI}$ is computed

*info* is the expected number of **hypotheses that could be eliminated** at $r$.

*distance*$_{OI}$ is the shortest path **constrained to lie in** $OI$, from robot’s **initial** location at overlay origin to $r$. 
Common Overlay Localization (COL) Algorithm

1. Sense initial visibility data $\mathcal{W}$.
2. Generate set of all hypotheses.
3. Compute overlay intersection $\mathcal{O}I$.
4. Pick set $R$ of random points in $\mathcal{O}I$.
5. For each $r$ in $R$, compute Value $[r]$.
6. (…)
7. Move robot to point $r^*$ in $\mathcal{O}I$.
8. Eliminate hypotheses inconsistent with robot’s observations at $r^*$.
9. Repeat (3-8) until only 1 hypothesis remains.
Useful Region Localization (URL) Algorithm

- Internal Edge: Edge that separates the inside of $OI$ from other parts of the overlay arrangement.

- Weak Visibility Polygon: Given a polygon $P$ the weak visibility polygon of an edge $e$ in $P$ is the set of all points $y$ in $P$ that are visible from some point on $e$. 
Useful Region Localization Algorithm

• Computing the useful region $U$ of $OI$:
  – Let the set of internal edges of $OI$ be $B$. For each edge in $B$ compute its weak visibility polygon within $OI$. The union of all such weak visibility polygons gives us $U$. 
Complexity Analysis

• **COL**: $O(n^3 \log n + Xn^2)$

• **URL**: $O(n^4 \log n + Yn^3)$
  
  • URL algorithm has computable and finite time bound.
Empirical analysis

- Random simulated office environments: 120 trials

- Observation one: it works, always, of course. How good are the results? How do they compare to the "optimal" (non-random, expensive-to-compute) path?
Empirical analysis: example
Results

Short version: a few hundred samples are enough.
Results

Performance of weighted decision strategy of COL algorithm vs. greedy strategy (less informed) and alternative heuristic.
Effect of Range Limits

Average Path Length for Wtd and Hybrid with Decreasing Visibility for 40 Rand. Pts

- Weighted
- Hybrid

Weighted versus hybrid strategy as visibility decreases (40 random points, 120 trials.)
Effect of Range Limits

Weighted versus hybrid strategy as visibility decreases

(40 random points, 120 trials.)

Ave. Path Length for COL and URL with Decreasing Visibility for 250 Random Points

<table>
<thead>
<tr>
<th>Visibility Range</th>
<th>Average Path Length (URL)</th>
<th>Average Path Length (COL)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>100</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>200</td>
<td>200</td>
<td>200</td>
</tr>
<tr>
<td>300</td>
<td>300</td>
<td>300</td>
</tr>
<tr>
<td>400</td>
<td>400</td>
<td>400</td>
</tr>
<tr>
<td>500</td>
<td>500</td>
<td>500</td>
</tr>
<tr>
<td>600</td>
<td>600</td>
<td>600</td>
</tr>
<tr>
<td>700</td>
<td>700</td>
<td>700</td>
</tr>
<tr>
<td>800</td>
<td>800</td>
<td>800</td>
</tr>
<tr>
<td>900</td>
<td>900</td>
<td>900</td>
</tr>
<tr>
<td>1000</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>1100</td>
<td>1100</td>
<td>1100</td>
</tr>
<tr>
<td>1200</td>
<td>1200</td>
<td>1200</td>
</tr>
<tr>
<td>1300</td>
<td>1300</td>
<td>1300</td>
</tr>
<tr>
<td>1400</td>
<td>1400</td>
<td>1400</td>
</tr>
<tr>
<td>1500</td>
<td>1500</td>
<td>1500</td>
</tr>
<tr>
<td>1600</td>
<td>1600</td>
<td>1600</td>
</tr>
<tr>
<td>1700</td>
<td>1700</td>
<td>1700</td>
</tr>
<tr>
<td>1800</td>
<td>1800</td>
<td>1800</td>
</tr>
<tr>
<td>1900</td>
<td>1900</td>
<td>1900</td>
</tr>
<tr>
<td>2100</td>
<td>2100</td>
<td>2100</td>
</tr>
<tr>
<td>2200</td>
<td>2200</td>
<td>2200</td>
</tr>
<tr>
<td>2300</td>
<td>2300</td>
<td>2300</td>
</tr>
<tr>
<td>2400</td>
<td>2400</td>
<td>2400</td>
</tr>
<tr>
<td>2500</td>
<td>2500</td>
<td>2500</td>
</tr>
<tr>
<td>2600</td>
<td>2600</td>
<td>2600</td>
</tr>
<tr>
<td>2700</td>
<td>2700</td>
<td>2700</td>
</tr>
<tr>
<td>2800</td>
<td>2800</td>
<td>2800</td>
</tr>
<tr>
<td>2900</td>
<td>2900</td>
<td>2900</td>
</tr>
<tr>
<td>3000</td>
<td>3000</td>
<td>3000</td>
</tr>
<tr>
<td>3100</td>
<td>3100</td>
<td>3100</td>
</tr>
<tr>
<td>3200</td>
<td>3200</td>
<td>3200</td>
</tr>
<tr>
<td>3300</td>
<td>3300</td>
<td>3300</td>
</tr>
<tr>
<td>3400</td>
<td>3400</td>
<td>3400</td>
</tr>
<tr>
<td>3500</td>
<td>3500</td>
<td>3500</td>
</tr>
<tr>
<td>3600</td>
<td>3600</td>
<td>3600</td>
</tr>
<tr>
<td>3700</td>
<td>3700</td>
<td>3700</td>
</tr>
<tr>
<td>3800</td>
<td>3800</td>
<td>3800</td>
</tr>
<tr>
<td>3900</td>
<td>3900</td>
<td>3900</td>
</tr>
<tr>
<td>4000</td>
<td>4000</td>
<td>4000</td>
</tr>
</tbody>
</table>
Experimental Results

URL algorithm: More dense sampling of useful random points produces rapid reduction in path length.
Effect of denser sampling (i.e. more computation) is more significant for sensors with limited performance. Computation can compensate for bad sensing.
Experimental Results
Summary

- Two randomized approximation algorithms that efficiently solve MDL.
- Random sampling leads to more practical localization trajectory than prior work.
- Performance of algorithms improves with number of random points used.
- Limited number of samples is usually sufficient to obtain near-optimal length localization trajectory.
- Open: how to best incorporate noisy
Summary

• Weighted decision strategy outperforms alternative greedy strategies.
  – Intrinsic robustness of randomized method (relative to deterministic approach) can be further augmented heuristically.
  – Effect of bounded range ameliorated by more computation.

• **Open questions:**
  • How to incorporate probabilistic data.
  • Tight average-case analysis.
  • 3D setting
Details of Approach

• info at a point $r$ is calculated as follows:
  – Assume all hypotheses are equally likely.
  – Two hypotheses $h_i$ and $h_j$ are equivalent if $Vis(h_i, r)$ is congruent to $Vis(h_j, r)$.
  – If there exist $b$ equivalence classes of hypotheses at $r$ of sizes $s_1, s_2, \ldots, s_b$ where the total number of hypotheses is $k = s_1 + s_2 + \ldots + s_b$, then: