OA3602

*Search Theory and Detection*

Coverage Planning

**Objectives:**
- Complete coverage patterns
- Track spacing calculation
- Decomposition of environments
- Coverage algorithms
  - Wavefront algorithm
Coverage Planning

**Coverage planning** is the study of how to algorithmically generate covering paths

- In general, computing constrained paths is a difficult problem!
  - Keep in mind assumptions

- However, analytic models provide **insights** and **performance bounds**!
Complete coverage patterns

Complete patterns have theoretical and practical application

A variety of **simple** approaches for covering areas* exist, e.g.,

- **sweeping** (a.k.a. "lawnmower," "ladder," "back-and-forth")
- **spiral in/out**

* – Note that continuous models usually require discretization for implementations (e.g., waypoints, state machines)
Other coverage patterns

Coverage patterns used in the field: Search and rescue

Sector search for circular region

Contour search of mountain slopes

Ref: Australian National SAR Manual, Ch 5
Calculating track spacing

*Applying exhaustive coverage patterns*

• For ideal complete coverage:

\[ A_S = \nu \omega t^* \quad \rightarrow \quad t^* = \frac{A_S}{\nu \omega} \]

— Given 3 parameters, can calculate 4\(^{th}\)

**Example:**

Given a 20 minute (hard limit) to cover a given sector of size 500 sq. nm, what is the sweep width necessary for a P3 (flight speed of 300 knots) to cover the area?
Field experimentation with UAVs

Track spacing is a function of aerial asset *altitude* and camera field of view (*zoom*) settings

- Computed real-time by onboard *autonomy*
Coverage planning for complex environments

*Partitioning of the environment*

**Complex environments** have:
- obstacles
- nonconvex shapes
- nonlinear boundaries

**Cellular decomposition**: Break up complex $A_S$ into simple $A_i$'s, $i = 1, \ldots, C_S$ and use simple patterns in each cell

- **exact**: $\bigcup_{i=1}^{C_S} A_i = A_S$
- **approximate**: $\bigcup_{i=1}^{C_S} A_i \supseteq A_S$
Exact cellular decomposition

Various approaches

Trapezoidal

- draw vertical lines from all obstacle/boundary vertices

Boustrophedon (“way of the ox”)

- extension of trapezoidal method
- appropriately combines adjacent cells

- Trade off between resolution (size) and complexity (number)
- Still need to piece all cells together
  - Use graph/network structure → Traveling Salesman
Approximate cellular decomposition

- Consider **uniformly sized** cells (e.g., grid, hexagonal)
- Allow excess, but no overlap
- Each cell is numbered
  - convention: top-down, left-right

- can still use sweeping/spiral type patterns
- but can’t easily incorporate “search preference”
  - right now, there is no preference but there will be!

⇒ use the **wavefront algorithm for coverage**
Wavefront algorithm for coverage

Ref: Zelinsky et al., 1993

Algorithm Pseudo-code:

1. Discretize with uniform cell size
2. Give all cells outside of area value of -1
3. Give goal cell value of 0
4. Starting from goal cell, for each cell:
   1. Determine grid distance from goal, $d_k$, to cell $k$
   2. Give value of $d_k$ to cell $k$
   3. Continue until all cells given a value
5. Initialize trajectory $S = \{s_{start}\}$ and counter $k = 0$
6. Start pseudo-gradient ascent (similar to Tabu Search):
   1. Find cell with highest value in unvisited neighboring cells
      - If multiple cells with same value, employ tie-break rule
   2. Add selected cell to trajectory, $S \leftarrow \{S, s_k\}$
   3. If no unvisited cells in neighborhood, go to previous cell
7. Repeat until goal cell reached

Tie-break rule:
West first, clockwise
Wavefront algorithm in action
Wavefront algorithm in action

[1] Discretize the environment uniformly
Wavefront algorithm in action

[2] Give all areas outside $A_S$ a value of -1
Wavefront algorithm in action

[3] Give goal cell a value of 0
Wavefront algorithm in action

[4] Give all cells value representing distance to goal
Wavefront algorithm in action

[5] Use gradient ascent to determine path (employing tie-break rule if necessary)

Tie-break rule: West first, clockwise
From coverage path to coverage ratio

Any coverage path can be evaluated to compute coverage ratio

Coverage path, \( S = \{s_{\text{start}}, \ldots, s_{\text{goal}}\} \), yields coverage fraction

- computed either online or offline

\[ |S| = n^* = 42, \quad C_S = 39 \]
Discussion: Wavefront for coverage

*Simple algorithm for automated coverage path*

**Comments:**

- analytic coverage computation not straightforward
- guaranteed to be complete coverage path
- **BUT** generated path is not necessarily shortest length
  - requires solving Traveling Salesman Problem (TSP)
- **many(!) optimization extensions**
  - choice of start/goal cell affects path length
  - minimize number of revisits
  - minimize number of sharp turns
  - maximize distance from obstacles (collision avoidance)
- Why not just use simple patterns (e.g., sweep)?
  - Prior and real-time information easily incorporated with only slight modification to wavefront algorithm
Ideal, complete coverage, almost!

*Suboptimal (in time) coverage*

Time spent conducting both coverage and transit tasks:

\[ n^* = \left( \text{# steps in coverage} \right) + \left( \text{# steps in transit} \right) = n^{\text{cover}} + n^{\text{transit}} \]
Parting shots

• Today we learned...
  – Coverage planning
  – Cellular decomposition
  – Wavefront algorithm for coverage

• Next time...
  – Random coverage models

Don’t forget to turn in your mud cards!