Motion Planning with Uncertainty

Lakulish Antani

October 15, 2008

Outline

Motivation

- Robots need to deal with the real world when planning
- Nature may act in unexpected ways
- Sensors and/or actuators may be faulty/inaccurate
- Perfect knowledge of state may be unavailable

Problem Formulation

Given:

- State space X
- Action space U
- State transition function $f : X \times U \rightarrow X$
- Initial state x_I
- Goal set X_G

Compute a plan $\pi : X \rightarrow X$

Feasible Planning

- Consider a graph with vertices labeled with states and edges labeled with actions
- Planning reduces to searching for a path from x_I to some $x_G \in X_G$
- Well-studied graph algorithms used for planning, such as A*

Optimal Planning

- We are given a cost function $l(x, u)$
- We need to find a path from x_I to X_G with lowest cost
- One solution is the **value iteration** algorithm

Value Iteration

- $G_k(x)$ is the lowest cost to reach the goal from x in k steps
- $G_0(x)$ is 0 for goal states, inf otherwise
- Use the recurrence

$$G_k(x) = \min_u \{l(x,u) + G_{k-1}(f(x,u))\}$$

- Plan can be constructed using the arg min form of the recurrence

Modeling Nature

- Treat nature as another agent
- Nature chooses action $\theta \in \Theta$ after robot does
- Robot doesnt know the nature action, only $\Pr(\theta)$
- State transition function of the form $f(x, u, \theta)$

Modeling Nature

- Edges in the state space graph are labelled with (u, θ)
- From state x after action u the next state is not known
- We follow out-edge labelled with (u, θ) with probability $\Pr(\theta)$

Modeling Nature

- New cost function of the form $l(x, u, \theta)$
- Optimal next action given by

$$u^* = \arg\min_u \{E_\theta[l(u, \theta)]\}$$

Modeling Nature

- We can extend value iteration to account for nature
- The new recurrence is

$$G_k(x) = \min_u \{ E_\theta [l(x, u, \theta) + G_{k-1}(f(x, u, \theta))] \}$$

Modeling Sensors

- What if we can't determine θ?
- Suppose we use a sensor to try to determine state
- Use a sensor mapping, $h : X \rightarrow Y$
- This is a deterministic model ($y = h(x)$)

Modeling Sensors

- Model uncertainty using nature sensing actions $\psi \in \Psi$
- Now $y = h(x, \psi)$
- Assume we know $\Pr(\psi \mid x)$
- y plays the role of θ from now on

Information Spaces

- Robot has no knowledge of its state
- The only information it has is the history of actions and sensor observations
- Decisions must be made based on available information
- Planning occurs in information space

Information Spaces

- In this case, the history information space I_{hist}
- Each state η_k has 3 components:
  - Initial condition η_0
  - Action history $\tilde{u}_{k-1}$
  - Sensor history $\tilde{y}_k$

Information Spaces

Three kinds of initial condition:

- **Deterministic**: η_0 is some state x
- **Nondeterministic**: η_0 is some subset of X
- **Probabilistic**: η_0 is some distribution $\Pr(x)$

Derived Information Spaces

- I_{hist} is too large!
- We derive a simpler I-space from I_{hist}
- Use an information mapping $\kappa : I_{hist} \longrightarrow I_{der}$
- Plans made using I_{der} should still work!

Probabilistic Information Space

For any history state $\eta_k = (\eta_0, \tilde{u}_{k-1}, \tilde{y}_k)$, compute the distribution $\Pr(x \mid \eta_k)$ over possible states the robot can be in.
This is the probabilistic information space I_{prob}.

Planning in Information Spaces

We are now given:

- Information state $\eta \in I_{prob}$
- Actions $u \in U$
- Nature action space $\theta \in \Theta \subseteq Y$
- Initial state η_0
- Goal state η_G

We can use existing planning algorithms in this space.

Planning in Information Spaces

- States are now probability distributions
- We need to define a new cost function
- We can use

$$L(\eta, u, y) = \sum_{x} \sum_{y} \Pr(x) \Pr(y \mid x) l(x, u, y)$$

Value Iteration

- We can use value iteration to find an optimal plan in I_{prob}
- However, the state space is continuous
- This leads to complications when deciding the range of θ to consider
- Another issue is choice of distributions

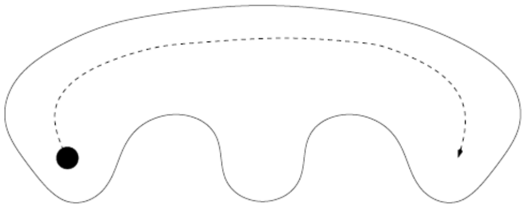
Simplifications

- One way out is to use a nondeterministic model instead of probabilistic
- Basically, we perform worst-case analysis
- Consider only the costliest choices at each step, and assume nature will do its worst
- Murphy's law might not always be a good model

Problem Statement

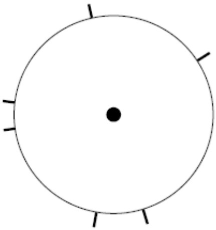
- Given some region R with a pursuer p and evader e
- Find a path for the pursuer to follow such that the evader will be seen
- If no such path exists, report that this is the case

The Environment



- 2-dimensional, with piecewise smooth boundary
- Simply connected

Gap Sensing

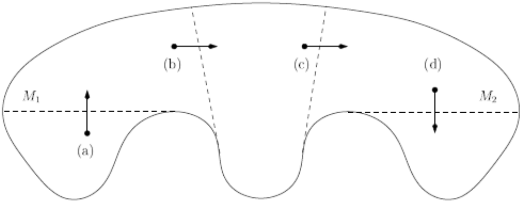


- Pursuer has omnidirectional view
- Can determine distance to closest wall along any direction
- Sensor reports directions of gaps (depth discontinuities)

Critical Lines

- Sensor events occur when gaps appear/disappear or merge/split
- These occur along specific lines in R

Critical Lines



- Appear lines arise due to inflection points in the boundary
- Merge lines correspond to bitangent lines

Navigation

- Following walls and merge lines is sufficient to solve the problem
- Walls and merge lines divide R into cells
- Pursuer moves from vertex to vertex in the cell decomposition graph G_n
- Each vertex has upto 4 neighbours, so 4 motion primitives

The State Space

- At any vertex u in G_n, gaps can be labeled as contaminated or cleared at any time
- u combined with the labeling gives the current state
- Movement causes state transitions
- This leads to a state graph G_s

Planning

- The goal is to reach a state where all gaps are labelled cleared
- We can use any standard algorithm to search G_s
- If no path exists which leads to a goal state, we report that R cannot be cleared

Variants

- Simpler environments
- Multiple evaders
- Limited field of view for the pursuer
- Unknown environments

Possible Extensions

- 3-dimensional environments
- Multiple pursuer coordination

References

LaValle, S. M.
Planning Algorithms.
Cambridge University Press, 2006.

S. Sachs, S. R., and LaValle, S. M.
Visibility-based pursuit-evasion in an unknown planar environment.
In *International Journal of Robotics Research* (2004), pp. 23(1):3–26.