What Can Be Learned by Following Walls

Anna Yershova

Department of Computer Science
University of Illinois at Urbana-Champaign

October 18, 2006

Joint work with Benjamin Tovar, Robert Ghrist, Steven LaValle

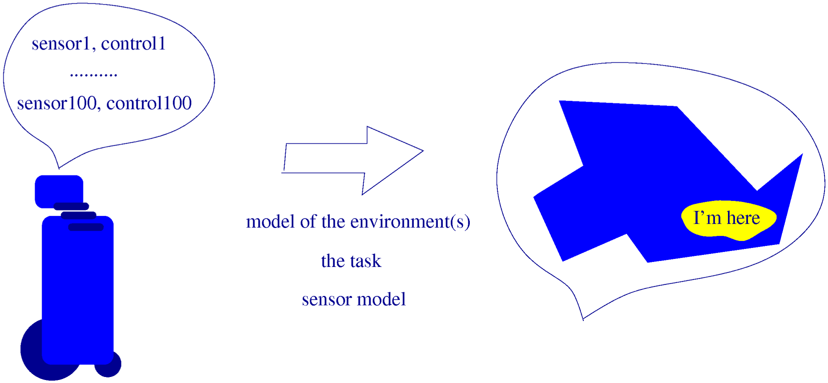
Information Spaces

Setting:

- A robot wanders around, gathers information
- Information: histories of controls and sensor observations
- The amount of histories grows in time
- The resulting space is called *information space*
- The solution to a robotics tasks lies in the information space

Our long-term goal is to study information spaces to solve robotics tasks

Information Spaces



Planning in Information Space

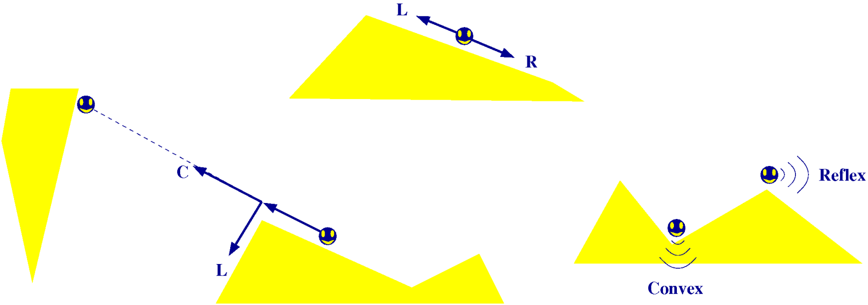
- Many approaches are aimed at estimating the state before planning
- This is sometimes impossible
- This is often not needed to solve the problem
- We are interested in planning in information spaces directly [Erdmann, Mason, 88; Donald, Jennings, 91; Rimon-Canny, 94]

Open Problems:

- Understand the structure of information spaces [LaValle05]
- Design simple information spaces
- Study simple robots with simple sensors [Tovar,et.al.04,05]
- Understand the connection between difficulty of the tasks and sensors
- Compare the powers of sensors [OKane,et.al.06]

A simple robot

- A point robot
- It can follow walls
- It can detect reflex and convex vertices
- It can follow the continuation of edges from the convex vertices

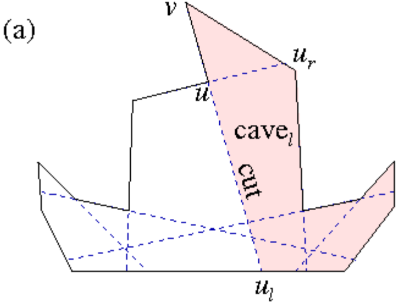


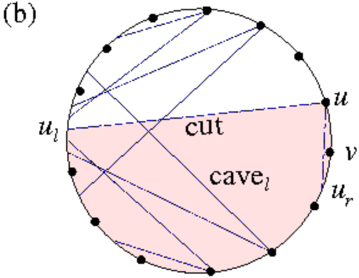
High Uncertainty

- No initial information is given to the robot
- No model of the environment
- No distance or angular measurements
- No odometry
- No GPS or compass
- Simple controls and contact sensor only

What can the robot learn?

Draw a nice progression to this cut diagram: case no pebble and case with pebble





The Algorithm for Learning the Cut Diagram

dropPebble; i = 0
do **goLeft**
    if $y_c =$ **reflex** add v_i to V^e_{rD}
    add v_i to V^e_D and i++
until $y_p =$ **seePebble**
for each $v \in V^e_{rD}$
    for each $\mathbf{u} \in \{$**goLeft**, **goRight**$\} \subset U_c$
        apply $\mathbf{u}$ $index\_in\_V^e_D(v)$ times
        if u = goLeft
            then goLeftOf at v reaching v_c
            else goRightOf at v reaching v_c
        apply $\mathbf{u}$ and count $index\_in\_V(v_c)$
        until $y_p =$ **seePebble**
        initialize the cut $[v, v_c]$ in C_D

Learning the Cut Diagram

Proposition

The robot with a pebble is able to construct a cut diagram of the environment.

Proposition

The robot with no pebble is not able to construct a cut diagram of the environment.

Cut Diagram Stores Geometry and Visibility Information

Once the cut diagram is completed, the robot obtains information encoded by it.

- Information about geometry of the polygon
- Information about the visibility from the current position in the polygon

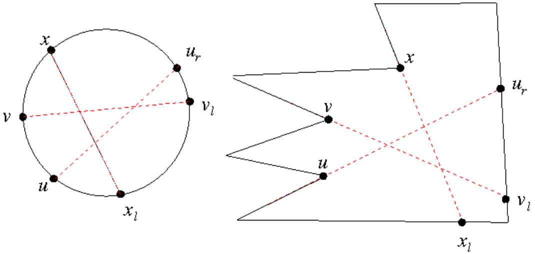
We studied the properties which relate the cut diagram to the information about the polygon.

Intersection in the Cut Diagram

Proposition

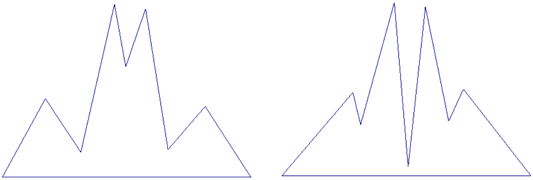
Two cuts intersect in the polygon if and only if they intersect in the cut diagram.

Note: The cut diagram does not always preserve three-intersections of the cuts.



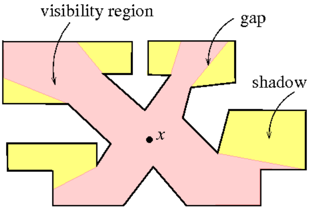
Uncertainty Represented by a Cut Diagram

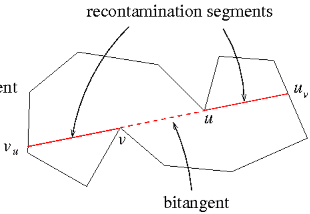
Different polygons with the same cut diagram:



Visibility Concepts

- Visibility region $V(x)$ from the point x
- Shadow regions
- Bitangent
- Recontamination segments

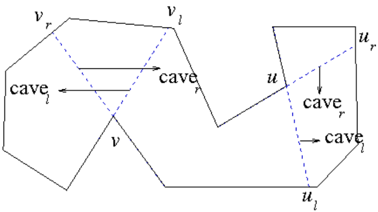




Where Are Bitangents?

Proposition

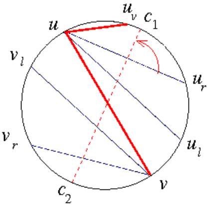
If a bitangent exists between two reflex vertices, then each of the reflex vertices is inside exactly one of the caves of the other vertex.

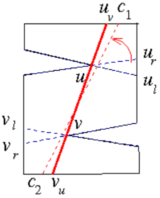


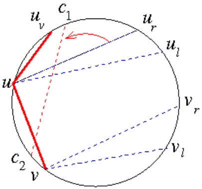
Where Are Recontamination Segments?

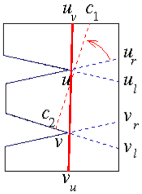
Proposition

Let a cut $\overline{c_1 c_2}$ intersect the chord $\overline{uv}$ in the cut diagram, such that $c_1 \in \widehat{uu_r}$. If a bitangent exists between u, and v, then the bitangent complement $\overline{uu_v}$ intersects the arc $\widehat{uc_1}$.









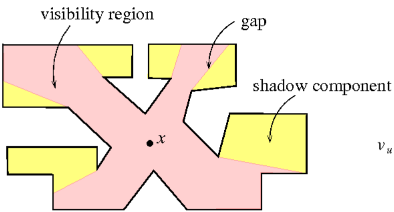
An Open Problem?

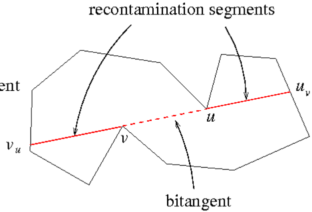
Conjecture

If a bitangent exists between u, and v, then, for any $\epsilon > 0$, there exists a polygon E, in which an endpoint u_v of the bitangent complement $\overline{uu_v}$ is arbitrarily close to the endpoint c_1^k of its approximation $\overline{uc_1^k}$, with the distance between them $|\overline{u_v c_1^k}| < \epsilon$.

Task 2: Pursuit-Evasion

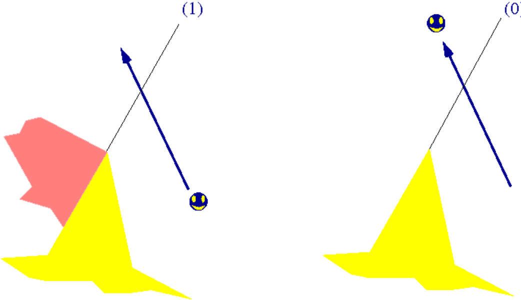
- A robot is the pursuer
- Arbitrary fast evaders in an unknown position in the environment
- The goal is to execute strategy which will guarantee the evaders capture
- If the evader comes into point of view of the robot they are considered to be caught





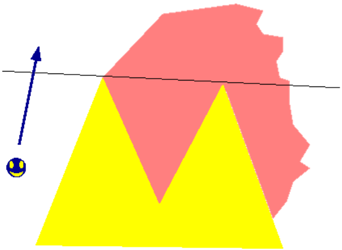
Critical Events

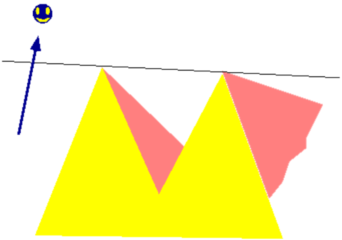
- Disappearance or appearance of a shadow component



Critical Events

- Merge or split of shadow components





Pursuit-Evasion

- We keep track of what is behind each of the corners.
- Each cut represents a part of a shadow region.
- Each cut has a label (1 = contaminated) or (0 = clear).
- The label can only change if one of the critical events happen

The Pursuit-Evasion Algorithm

- Breadth first search on the cut diagram graph
- Keep track of the corresponding labels as the critical events happen

movies