Autonomous Systems

Mtrx 4700 : Experimental Robotics

Obstacle Avoidance & Path Planning

Dr. Robert Fitch
Dr. Stefan B. Williams

[with slides by Bob Wang, Latombe (Stanford), Moore (CMU), Thrun (Stanford) …]

Autonomous Systems

DARPA Grand Challenge

Autonomous Systems

Today' s Goals

- To familiarize you with the basic techniques of planning.
- To get you excited!
- To let you experience the difficulties of planning in the real world.

Autonomous Systems

Stanford Planning Course (Undergraduate)

`robotics.stanford.edu/~latombe/cs326/2004/index.htm`

1	Overview
2	Path planning for point robot
3	Configuration space of a robot
4	Collision detection 1/2: Hierarchical methods
5	Collision detection 2/2: Feature-tracking methods
6	Probabilistic roadmaps 1/3: Basic techniques
7	Probabilistic roadmaps 2/3: Sampling strategies
8	Probabilistic roadmaps 3/3: Sampling strategies
9	Criticality-based motion planning: Assembly planning and target finding
10	Coordination of multiple robots
11	Kinodynamic planning
12	Humanoid and legged robots
13	Modular reconfigurable robots
14	Mapping and inspecting environments
15	Navigation in virtual environments
16	Target tracking and virtual camera
17	Motion of crowds and flocks
18	Motion of bio-molecules
19	Radiosurgical planning

CMU Planning course (Post-graduate)

http://www-2.cs.cmu.edu/~mmv/planning/schedule.html

1. Planning, Execution, and Learning
2. Linear Planning and Non-Linear planning
3. State-Space Planning
4. Partial Order Planning
5. Comparison: State-Space and Plan-Space Planning
6. Hierarchical Task Net Planning
7. GraphPlan, StatPlan
8. Heuristic Planning
9. BDD-based Planning
10. Conditional Planning
11. Markov Decision Processes
12. Reinforcement Learning
13. Decision-Theoretic Planning
14. Transformational Planning
15. POMDP
16. Explanation-based Learning in Planning
17. Analogical reasoning in Planning
18. Learning Domain-Specific Planners
19. Robot Path Planning
20. Execution
21. Execution and Learning
22. Multi-robot Systems

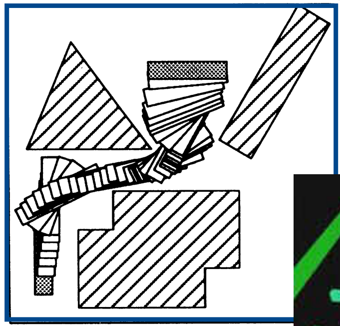
Path Planning and Obstacle Avoidance

- Path Planning
  - Given a map and a goal location, path planning involves identify a trajectory that will cause the robot to reach the goal location when executed.
- Obstacle Avoidance (Reacting)
  - Given real-time sensor readings, obstacle avoidance means modulating the trajectory of the robot in order to avoid collisions.
- Integrated Planning & Execution
  - The planner incorporates every new piece of information in real time, instantly producing a new plan that in fact reacts to the new information appropriately.

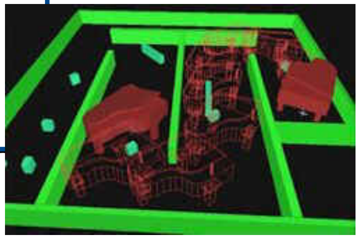
Goals of Planning

- Task-level programming
  - go to A without colliding with obstacles
  - assemble product P
  - build map of environment E
  - find object O
- Compute motion strategies, e.g.
  - geometric paths (*path planning*)
  - time-parameterized trajectories (*motion planning*)
  - sequence of sensor-based motion commands

Rigid Objects

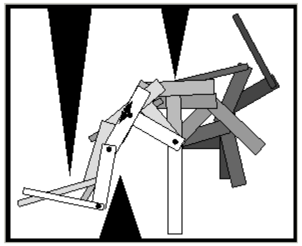


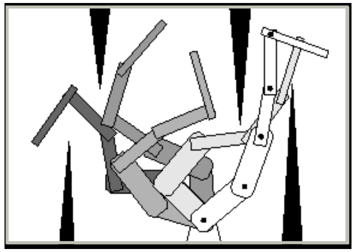
→ Ladder problem



Piano-mover problem ←

Articulated Objects

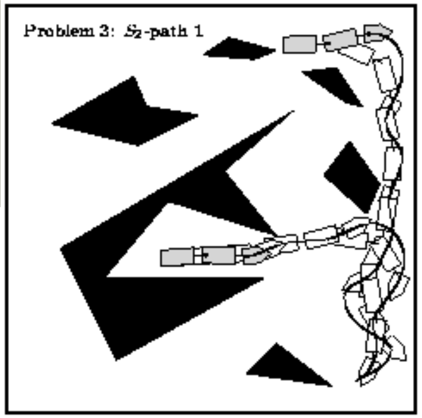




CMU Urban Search and Rescue

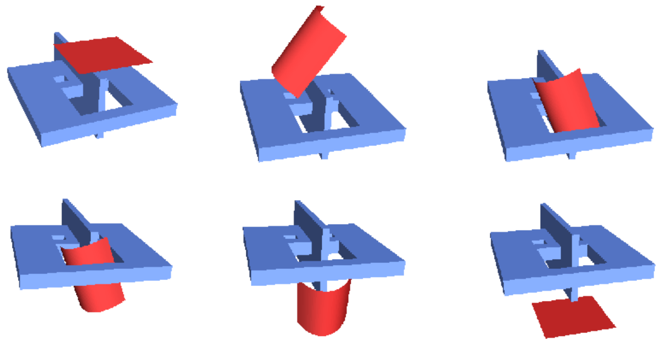
Nonholonomic Robots

CMU Navlab



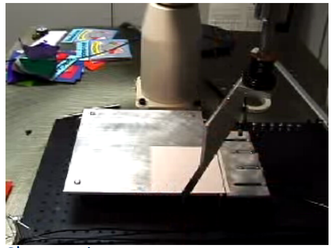
DARPA Urban Challenge

Deformable Objects



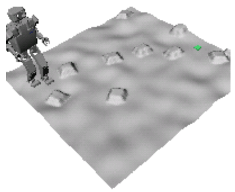
[Kavraki] (Rice)

Origami Mathematics



Show movie http://www.cs.dartmouth.edu/~robotics/hat.mov

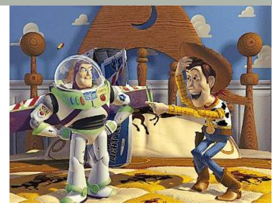
Humanoid: Footstep planning



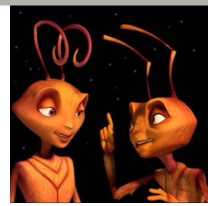
James Kuffner et al.

Digital Actors

A Bug's Life (Pixar/Disney)



Toy Story (Pixar/Disney)



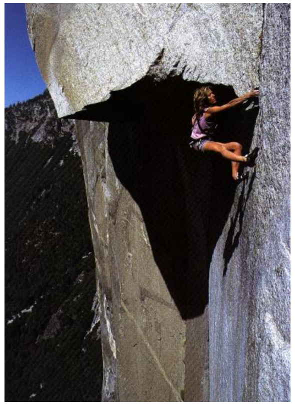
Antz (Dreamworks)

Tomb Raider 3 (Eidos Interactive)

The Legend of Zelda (Nintendo)

Final Fantasy VIII (SquareOne)

Free-Climbing Robot



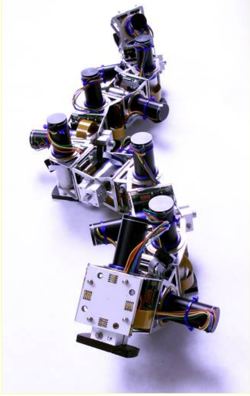
JPL's LEMUR robot

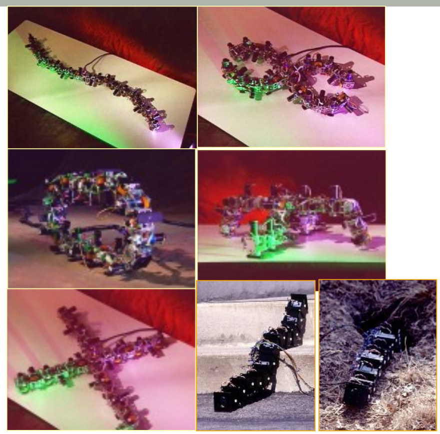
Free-Climbing Robot

http://ai.stanford.edu/~latombe/projects/fulltry_mpeg.mpg

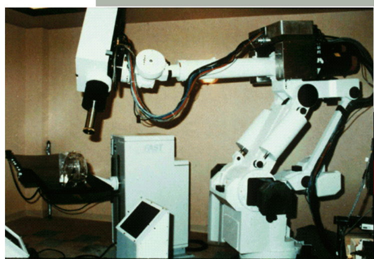
Modular Reconfigurable Robots

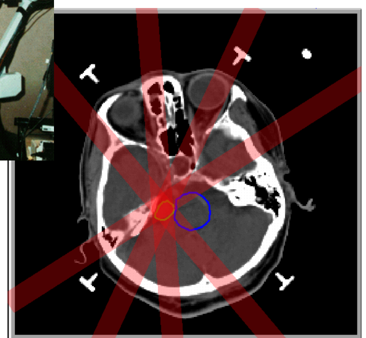
Casal and Yim, 1999





Radiosurgical Planning

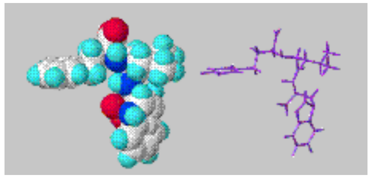




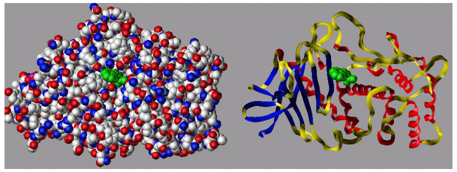
Cross-firing at a tumor while sparing healthy critical tissue

Autonomous Systems

Motion of Bio-Molecules



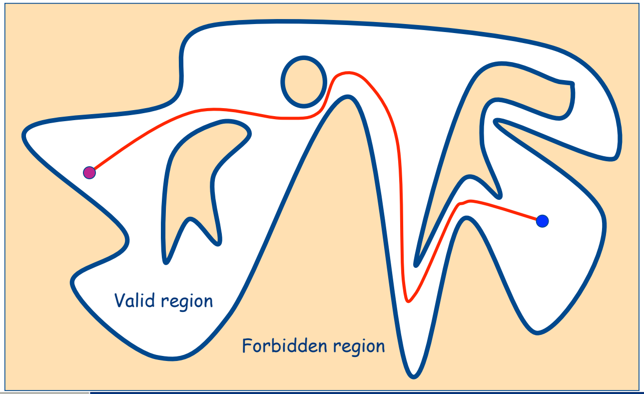
- Protein folding
- Ligand binding



Autonomous Systems

Fundamental Question

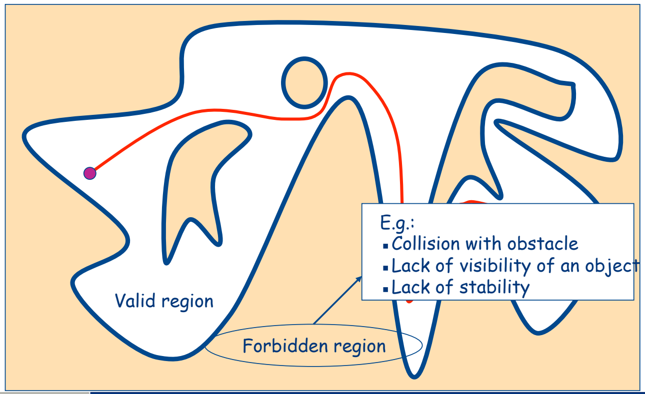
Are two given points connected by a path?



Autonomous Systems

Fundamental Question

Are two given points connected by a path?



Autonomous Systems

Basic Problem

1. Statement:
   Compute a **collision-free path** for a rigid or articulated object (the robot) among static obstacles
2. Inputs:
   1. Geometry of robot and obstacles
   2. Kinematics of robot (degrees of freedom)
   3. Initial and goal robot configurations (placements)
3. Output:
   - Continuous sequence of collision-free robot configurations connecting the initial and goal configurations
   - Report failure if no such path exists.

Some extensions of basic problem

- Moving obstacles
- Multiple robots
- Movable objects
- Assembly planning
- Goal is to acquire information by sensing
  - Model building
  - Object finding/tracking
  - Inspection
- Nonholonomic constraints
- Dynamic constraints
- Stability constraints
- Optimal planning
- Uncertainty in model, control and sensing
- Exploiting task mechanics (sensorless motions, under-actuated systems)
- Physical models and deformable objects
- Integration of planning and control
- Integration with higher-level planning

Topics we will cover…

- Framework for Planning
- Configuration Space
- Key algorithms for basic planning
- Some planning applications

Framework of Planning

Continuous representation

↓

Discretization

↓

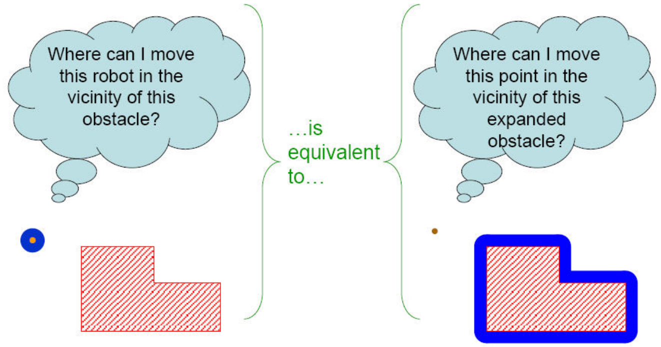
Graph searching
(blind, best-first, A*)

Map a robot to a point: *Configuration Space*

- Is the set of legal configurations of the robot. It also defines the topology of continuous motions.
- For a rigid-object robots (not joints) there exists a transformation to the robot and obstacles that turn the robot into a single point.
- The C-Space Transform.

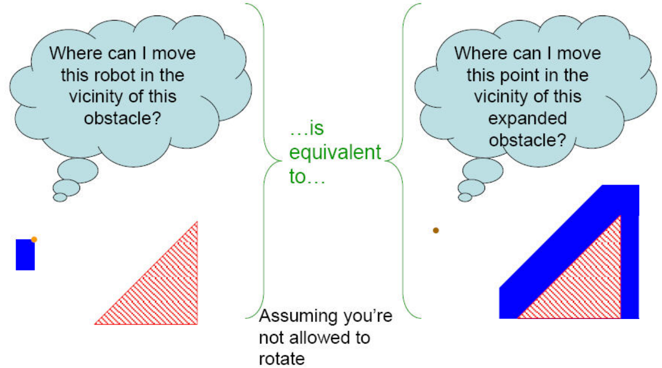
Configuration Space Example

- 2-D world, 2 DOFs



Configuration Space Example

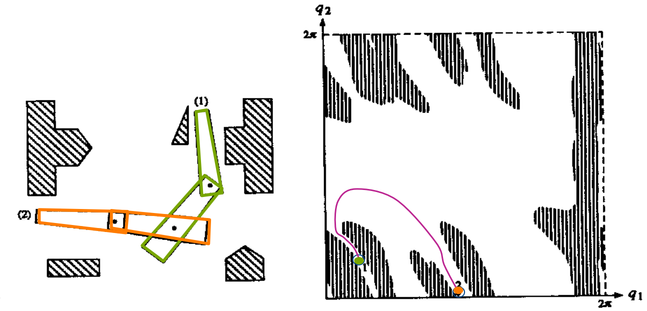
- 2-D world, 2 DOFs



Configuration Space Transform

- It is important for motion (path) planning.
- We have turned the problem from "Twist and turn this 2-D polygon past this other 2-D polygon" into "Find a path for this point in 3-D space past this weird 3-D obstacle".
- Now we can plan paths for points instead of polyhedrons/polygons.

Configuration Space Transform



Algorithms for motion planning

Autonomous Systems

Continuous representation

↓

Discretization

↓

Graph searching
(blind, best-first, A*)

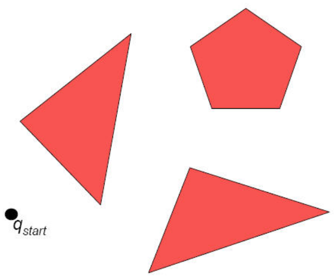
Algorithms for motion planning

Autonomous Systems

- **Road Map: Visibility Graph, Voronoi Diagrams**
  - Represent the connectivity of the free space by a network of 1-D curves
- **Cell Decomposition**
  - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells
- **Potential Methods**
  - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

Visibility Graph

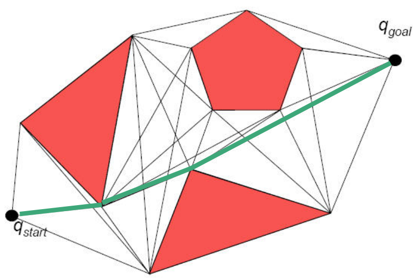
Autonomous Systems



- Now we have a CSPACE with polygonal obstacles
- If there were no blocks, shortest path would be a straight line. Else it must be a sequence of straight lines “shaving” corners of obstacles.

Visibility Graph Algorithms

Autonomous Systems

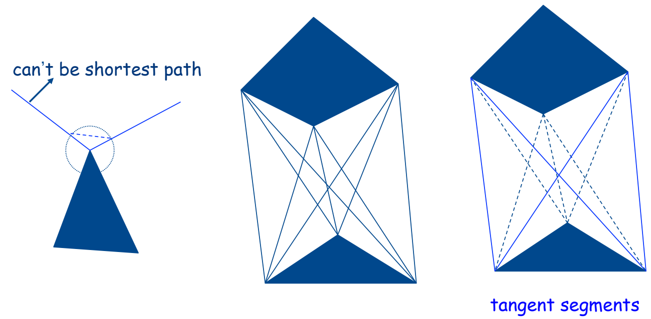


- 1. Find all non-blocked lines between polygon vertices, start and goal
- 2. Search the graph of these lines for the shortest paths

Visibility Graph Algorithms

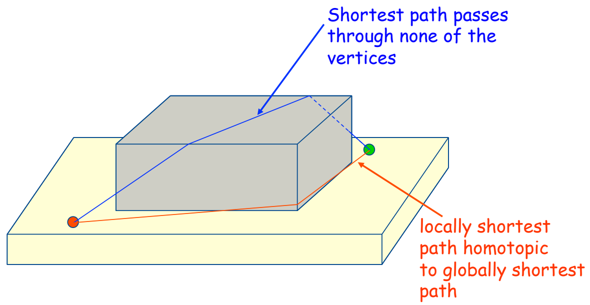
1. Install all obstacles vertices in VG, plus the start and goal positions
2. For every pair of nodes u, v in VG
3.   If segment(u,v) is an obstacle edge then
4.   insert (u,v) into VG
5.   else
6.   for every obstacle edge e
7.   if segment(u,v) intersects e
8.   then goto 2
9.   insert (u,v) into VG
10. Search VG using A*

Reduced Visibility Graph



→ Eliminate concave obstacle vertices

3-D space



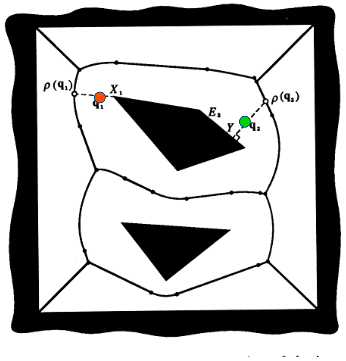
Computing the shortest collision-free path in a polyhedral space is NP-hard

Visibility Graph Algorithms

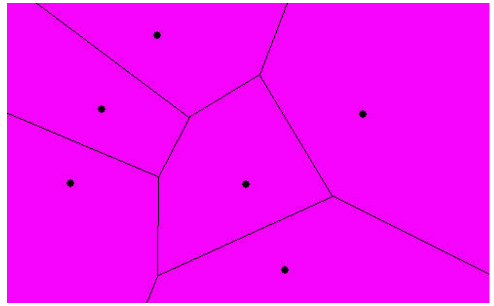
- Visibility Graph method finds the shortest path.
- But it does so by skirting along and close to obstacles.
- Any errors in control, or model of obstacle locations, bad thing happens

Voronoi Diagrams

- Idea: We may not care about optimality. Instead, we may want to get a non-stupid path that steers as far from the obstacles as it can.



Compute the Voronoi Diagram



http://www.cs.cornell.edu/Info/People/chew/Delaunay.html

Voronoi Diagram Methods for C-Space Motion Planning

1. Compute the Voronoi Diagram of C-Space.
2. Compute the shortest straightline path from start to any point on Voronoi Diagram.
3. Compute the shortest straight line path from goal to any point on Voronoi Diagram.
4. Compute shortest path from start to goal along Voronoi Diagram.

Voronoi Diagrams

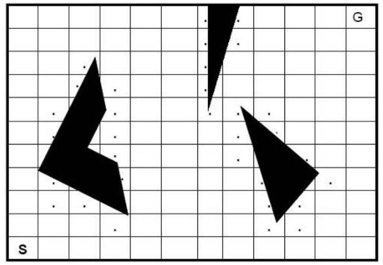
- It is very complex above 2-D.
- This “use Voronoi to keep clear of obstacles” is just a heuristic. And can be made to look stupid.

Algorithms for motion planning

- **Road Map: Visibility Graph, Voronoi Diagrams**
  - Represent the connectivity of the free space by a network of 1-D curves
- **Cell Decomposition**
  - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells
- **Potential Methods**
  - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

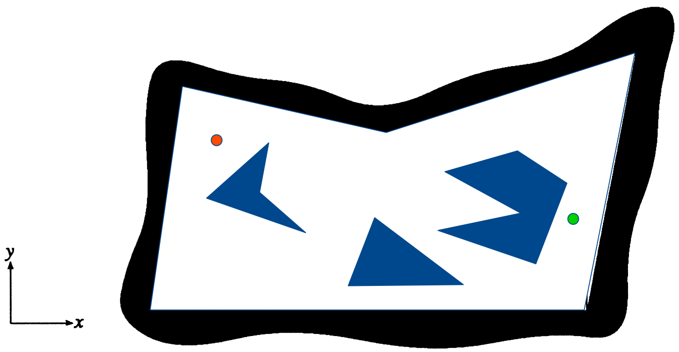
Cell Decomposition Methods

- Idea: Break Free Space into Convex Exact Polygons.

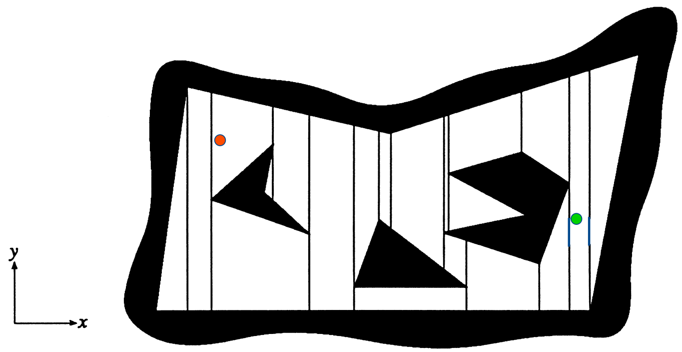


- 1. Lay Down a grid 2. Avoid any Cell which intersects an obstacle. 3. Plan shortest path through other cells.
- If no plan exists, double the resolution and try again!

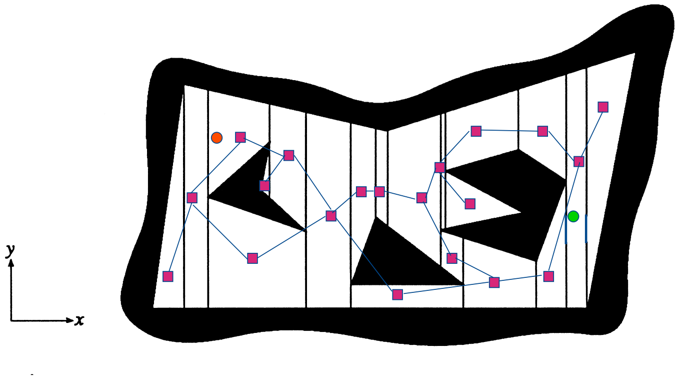
Trapezoidal Decomposition



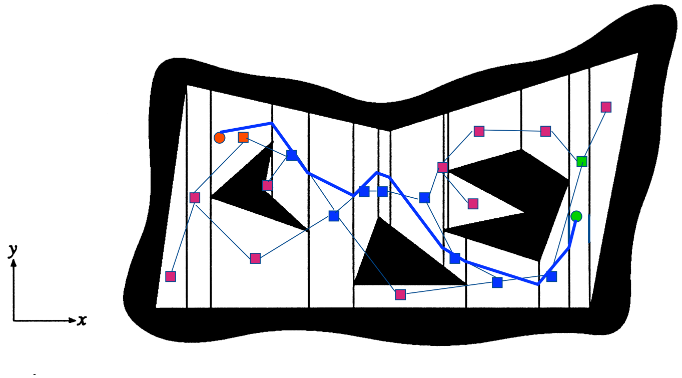
Trapezoidal Decomposition



Trapezoidal Decomposition



Trapezoidal Decomposition



Variable Resolution “Approximate & Decompose”

- Not so many complaints. This is actually used in practical systems.

- But

1. Not exact (no notion of “best” path)
2. Not complete: doesn’t know if problem actually unsolvable?
3. Still hopeless above a small number of dimensions?

Algorithms for motion planning

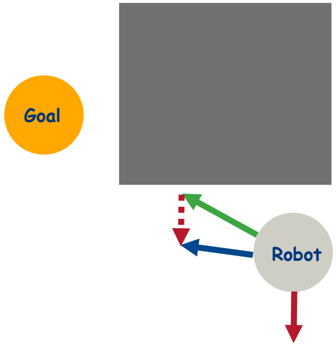
- **Road Map: Visibility Graph, Voronoi Diagrams**
  - Represent the connectivity of the free space by a network of 1-D curves
- **Cell Decomposition**
  - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells
- **Potential Methods**
  - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

Potential Field

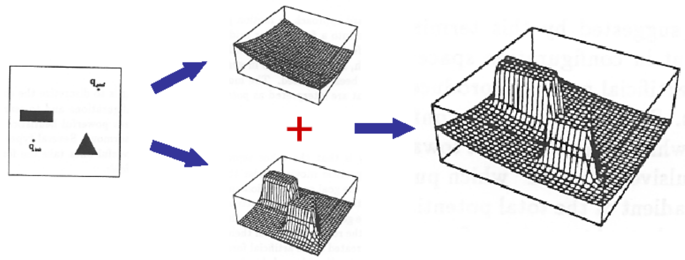
- Approach initially proposed for real-time collision avoidance [Khatib, 86]. Hundreds of papers published on it.

$$F_{Goal} = -k_p(x - x_{Goal})$$

$$F_{Obstacle} = \begin{cases} \eta\left(\frac{1}{\rho} - \frac{1}{\rho_0}\right)\frac{1}{\rho^2}\frac{\partial \rho}{\partial x} & \text{if } \rho \le \rho_0, \\ 0 & \text{if } \rho > \rho_0 \end{cases}$$

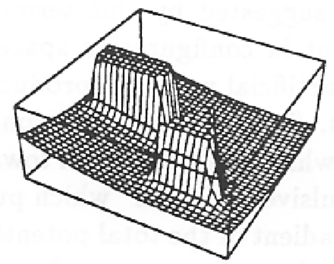


Attractive and Repulsive Fields/Forces



Local Minimum Issue

- Perform best-first search (possibility of combining with approximate cell decomposition)
- Alternate descents and random walks
- Use local-minimum-free potential (navigation function)



Algorithms for motion planning

- **Road Map: Visibility Graph, Voronoi Diagrams**
  - Represent the connectivity of the free space by a network of 1-D curves
- **Cell Decomposition**
  - Decompose the free space into simple cells and represent the connectivity of the free space by the adjacency graph of these cells
- **Potential Methods**
  - Define a function over the free space that has a global minimum at the goal configuration and follow its steepest descent

Completeness of Planner

1. A motion planner is complete if it finds a collision-free path whenever one exists and return failure otherwise.

2. Visibility graph, Voronoi diagram, exact cell decomposition, navigation function provide complete planners

3. Weaker notions of completeness, e.g.:
   - resolution completeness (PF with best-first search)
   - probabilistic completeness (PF with random walks)
   - A resolution complete planner discretizes the space and returns a path whenever one exists in this representation.
   - A probabilistically complete planner returns a path with high probability if a path exists. It may not terminate if no path exists.

What you should know

- To familiarize you with basic techniques of planning.
  - How to define configuration space
  - The basic idea behind
    - Visibility Graph methods
    - Voronoi methods
    - Cell Decomposition methods
    - Potential Field methods
- real-world planning problems

Further Reading

- J.C. Latombe. *Robot Motion Planning*. Kluwer Academic Publishers, 1991. http://robotics.stanford.edu/~latombe/
- S. Russell & Norvig. *Artificial Intelligence – A Modern Approach*. Prentice Hall, 2003.
- S. LaValle. *Planning Algorithms*. Cambridge University Press, 2006. http://planning.cs.uiuc.edu/
- H. Choset, K. Lynch, S. Hutchinson, G. Kantor, W. Burgard, L. Kavraki, S. Thrun. *Principles of Robot Motion*. MIT Press, 2005.