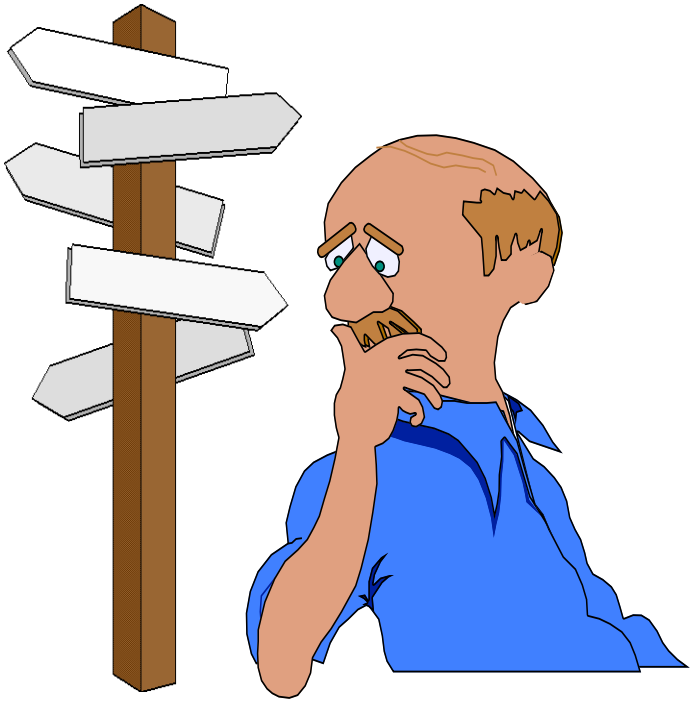
Motion Planning

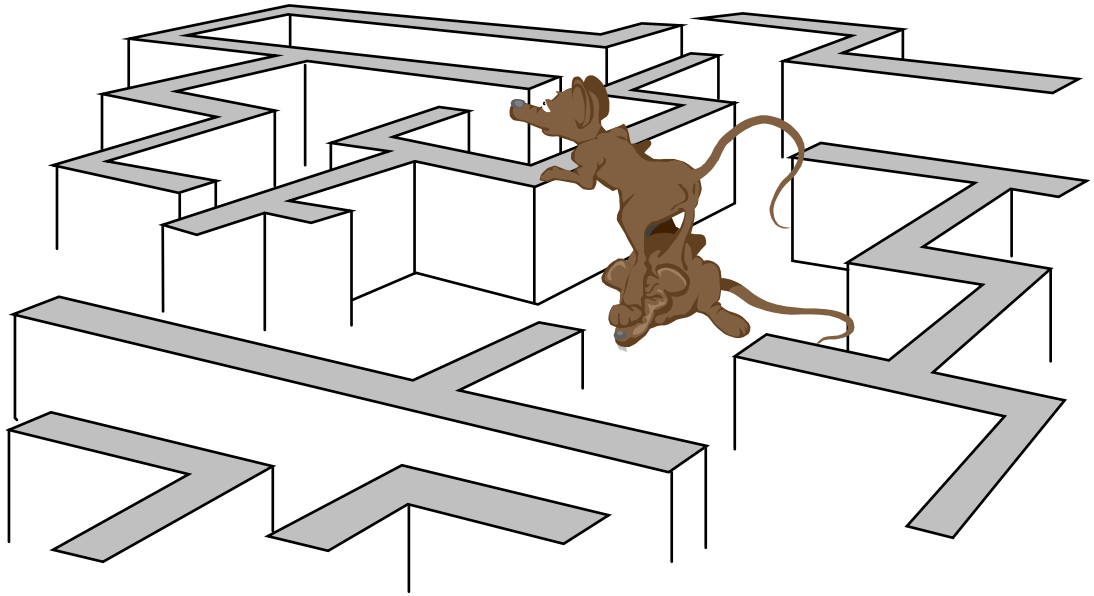
-M. A. El-dosuky

Robotics

What is Motion Planning?

Determining where to go without hitting obstacles





Robotics

The World consists of...

- Obstacles
  - Already occupied spaces of the world
  - In other words, robots can't go there
- Free Space
  - Unoccupied space within the world
  - Robots "might" be able to go here
  - To determine where a robot can go, we need to discuss what a *Configuration Space* is

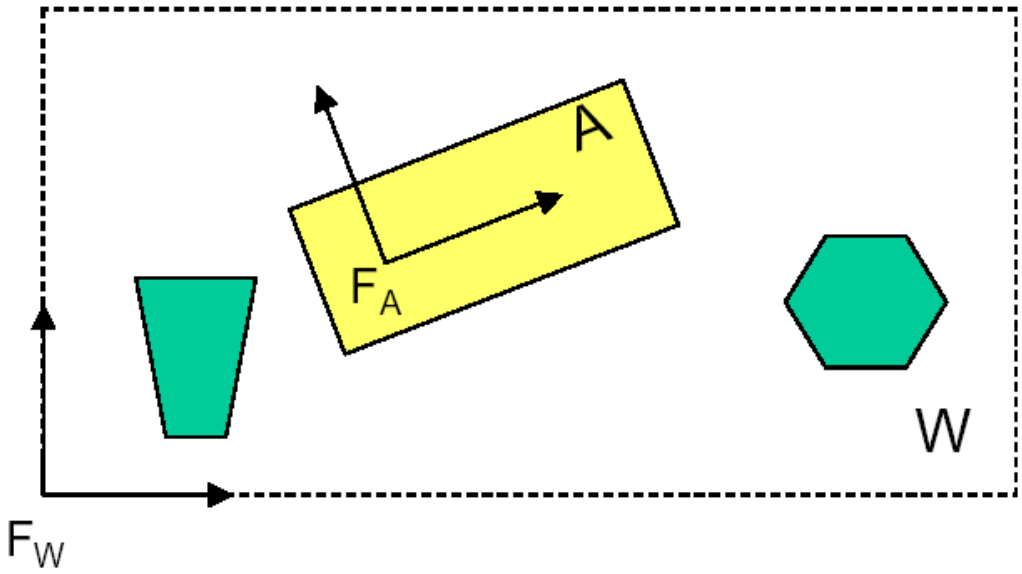
Configuration Space

Notation:

A: single rigid object –(the robot)

W: Euclidean space where A moves; $W = R^2$ *or* R^3

B1,…Bm: fixed rigid obstacles distributed in ***W***



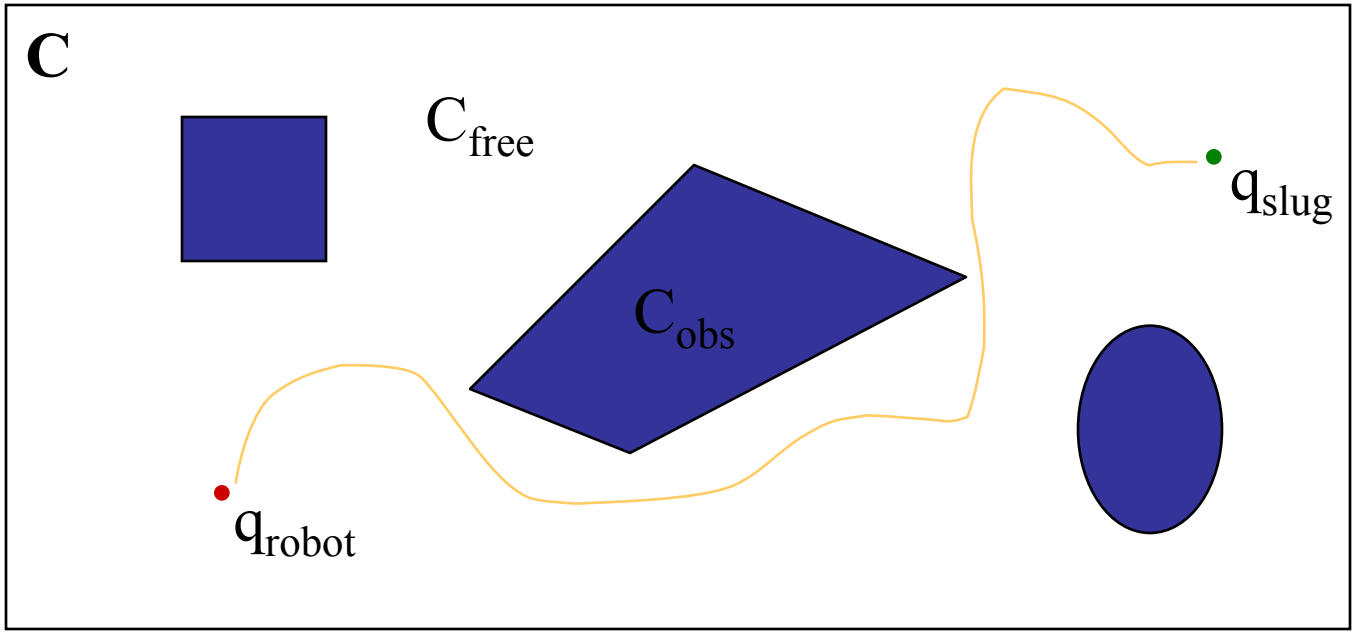
- F_W – world frame (fixed frame)
- F_A – robot frame (moving frame rigidly associated with the robot)

Configuration q of A is a specification of the physical state (position and orientation) of A w.r.t. a fixed environmental frame F_W.

Configuration Space

Configuration Space of ***A*** is the space (C)of all possible configurations of ***A***.

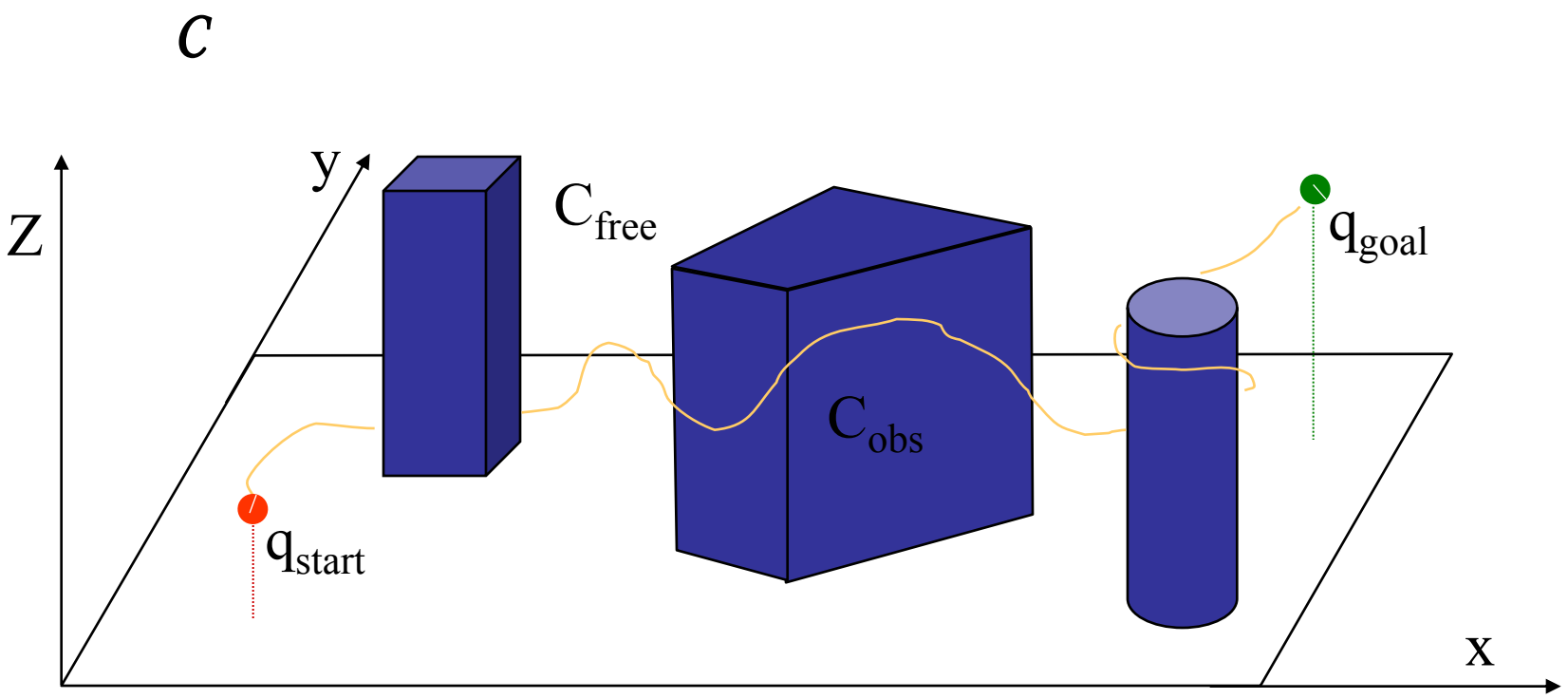
Point robot (free-flying, no constraints)



For a point robot moving in 2-D plane, C-space is R^2

Robotics

Configuration Space



For a point robot moving in 3-D, the C-space is R^3

Robotics

Free Space

From
Robot Motion Planning
J.C. Latombe

C-OBSTACLE REGION

$B_1, B_2, \ldots, B_m$ ——— obstacles

CB_i ——— C-obstacle

$\bigcup_{i=1}^{m} CB_i$ ——— C-obstacle region

FREE SPACE

$$C_{free} = C \setminus \bigcup_{i=1}^{m} CB_i = \{q \in C : A(q) \cap \bigcup_{i=1}^{m} CB_i = \phi\}$$

Free configuration q iff $q \in C_{free}$

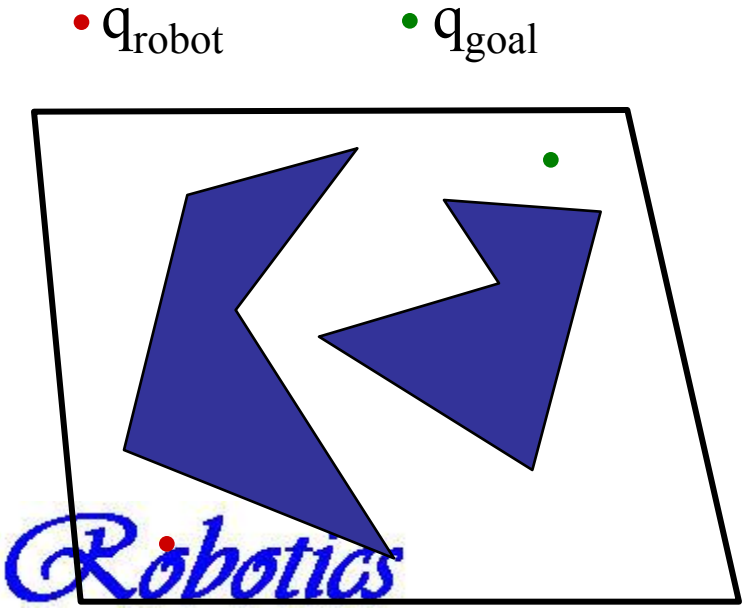
Robotics

Motion Planning Methods

The motion planning problem consists of the following:

Input

- geometric descriptions of a robot and its environment (obstacles)
- initial and goal configurations



Output

- a path from start to finish (or the recognition that none exists)

Applications

Robot-assisted surgery

Automated assembly plans

Drug-docking and analysis

Moving pianos around...

Motion Planning Methods

(1) Roadmap approaches

(2) Cell decomposition

(3) Potential Fields

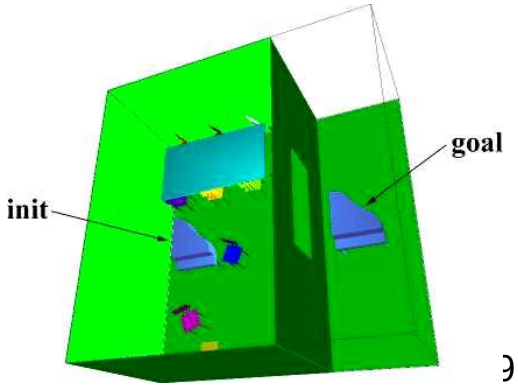
(4) Bug algorithms

Goal reduce the N-dimensional configuration space to a set of one-D paths to search.

Goal account for all of the free space

Goal Create local control strategies that will be more flexible than those above

Limited knowledge path planning



Robotics

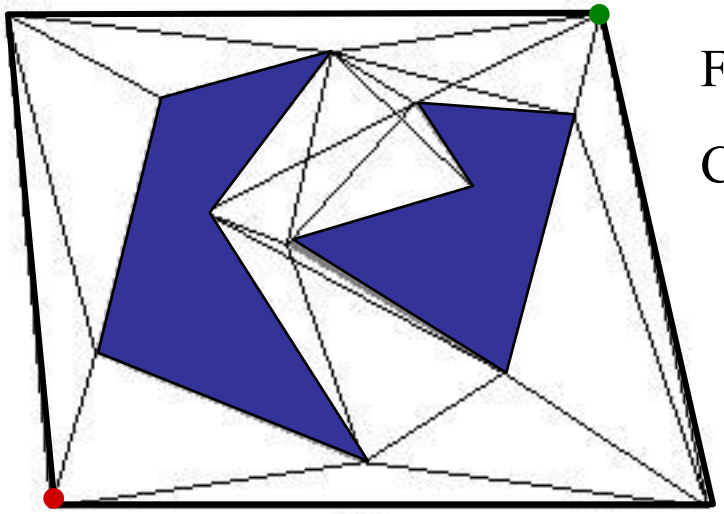
Roadmap: Visibility Graphs

Visibility graphs: In a polygonal (or polyhedral) configuration space, construct all of the line segments that connect vertices to one another (and that do not intersect the obstacles themselves).

Formed by connecting all "visible" vertices, the start point and the end point, to each other.

For two points to be "visible" no obstacle can exist between them

Paths exist on the perimeter of obstacles



From C_{free}, a graph is defined

Converts the problem into graph search.

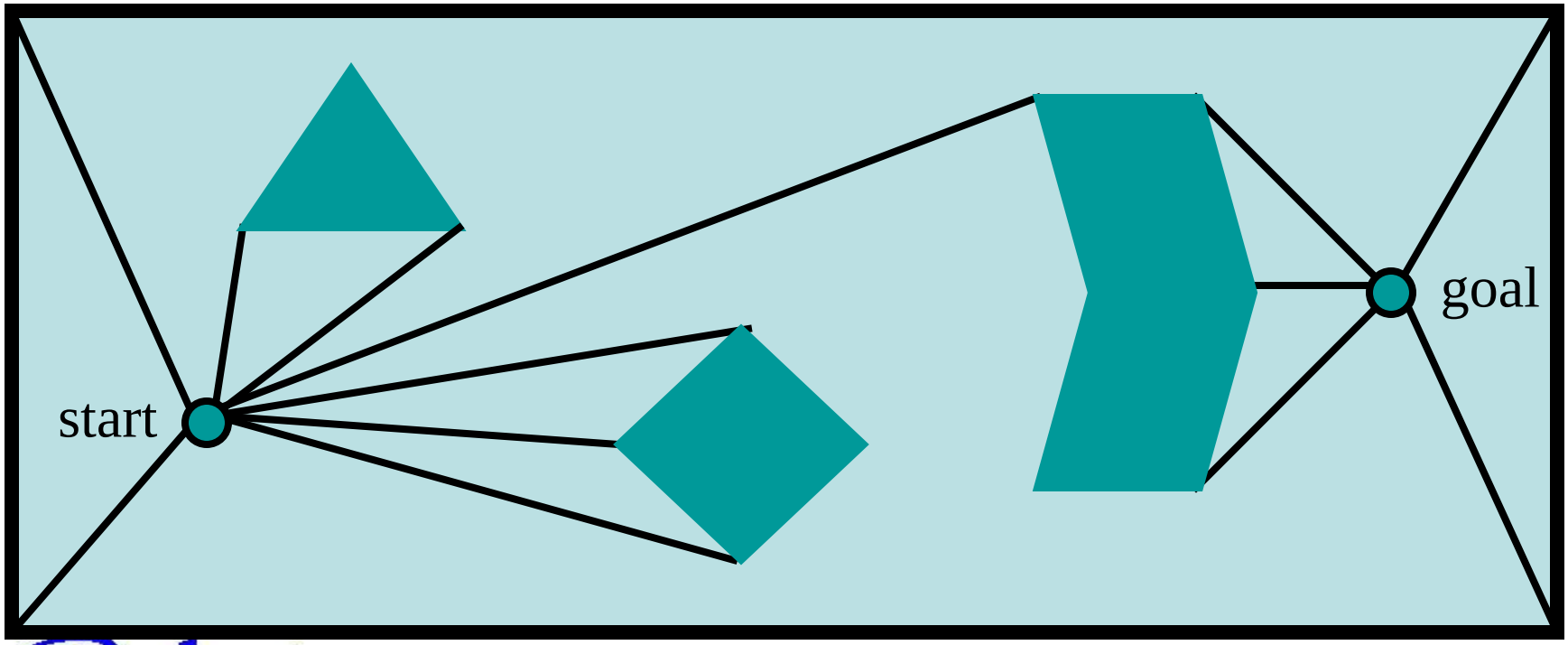
Dijkstra's algorithm O(N^2)

N = the number of vertices in *C*-space

Robotics

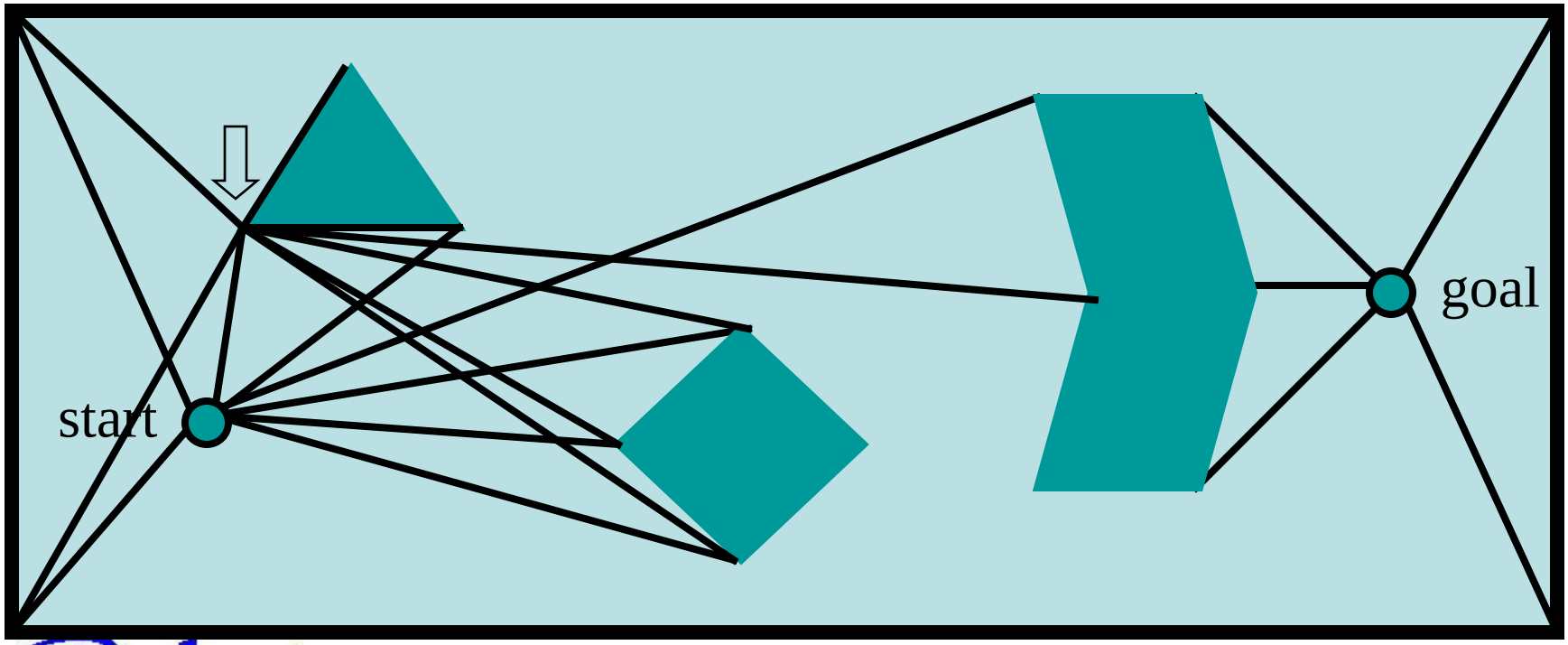
The Visibility Graph in Action (Part 1)

- First, draw lines of sight from the start and goal to all "visible" vertices and corners of the world.



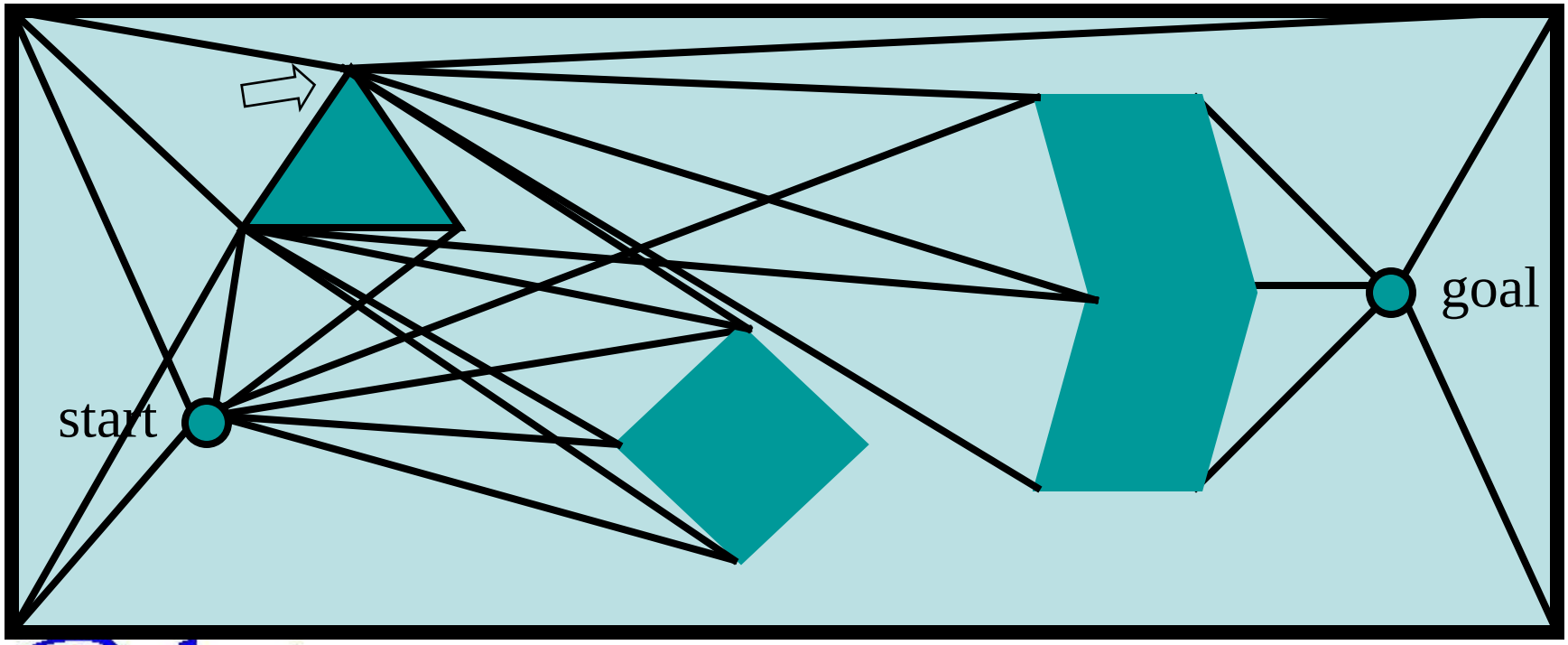
The Visibility Graph in Action (Part 2)

- Second, draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.



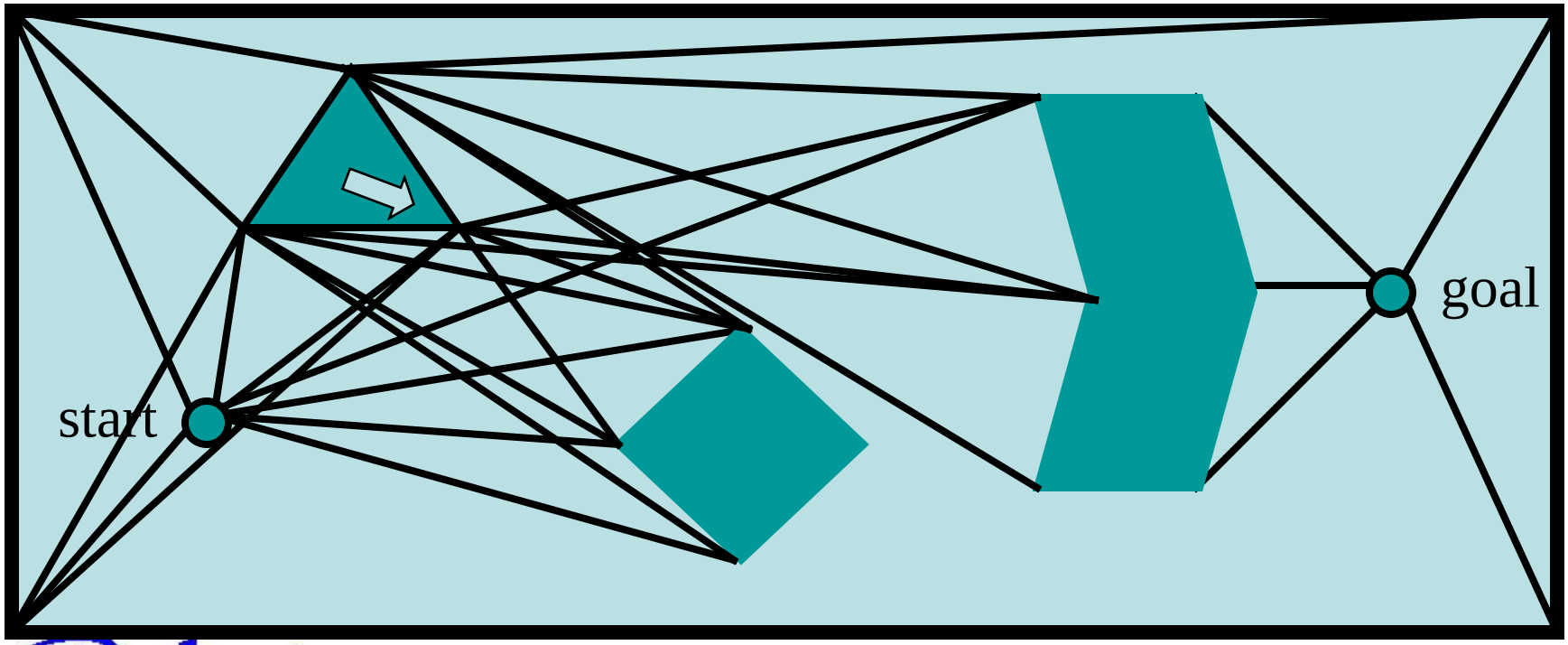
The Visibility Graph in Action (Part 3)

- Second, draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.



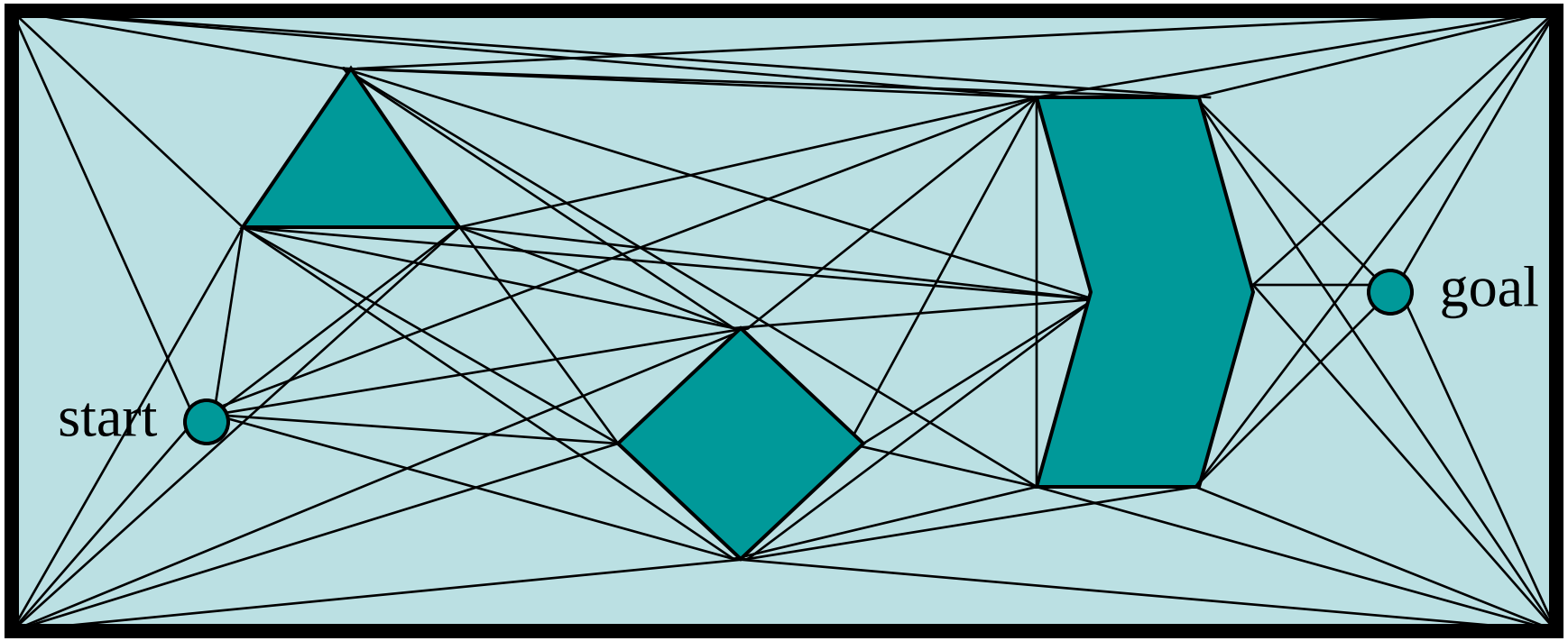
The Visibility Graph in Action (Part 4)

- Second, draw lines of sight from every vertex of every obstacle like before. Remember lines along edges are also lines of sight.



The Visibility Graph (Done)

- Repeat until you're done.

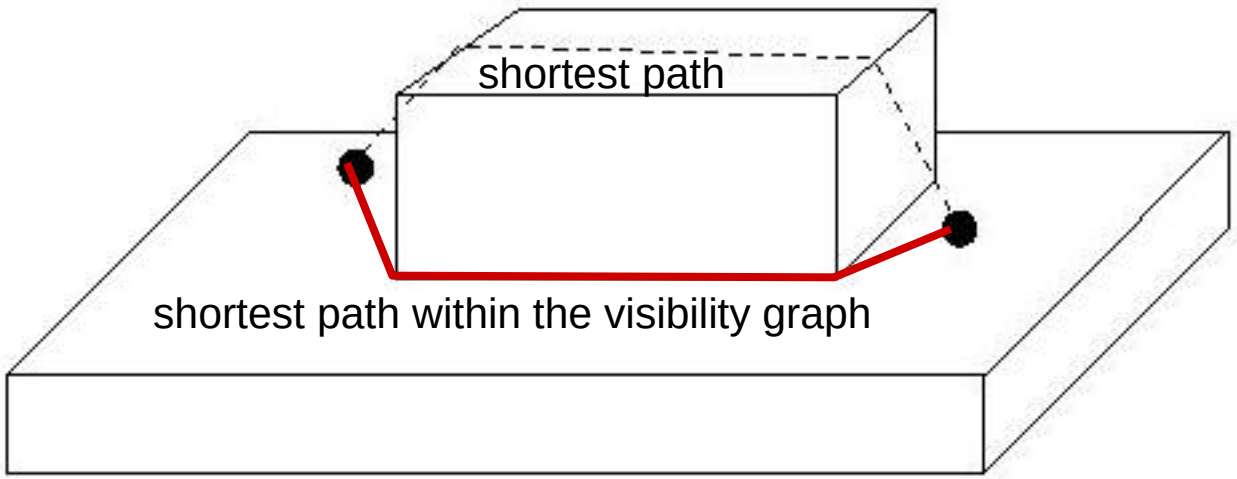


Since the map was in C-space, each line potentially represents part of a path from the start to the goal.

Robotics

Visibility graph drawbacks

Visibility graphs do not preserve their optimality in higher dimensions:



In addition, the paths they find are “semi-free,” i.e. in contact with obstacles.

No clearance

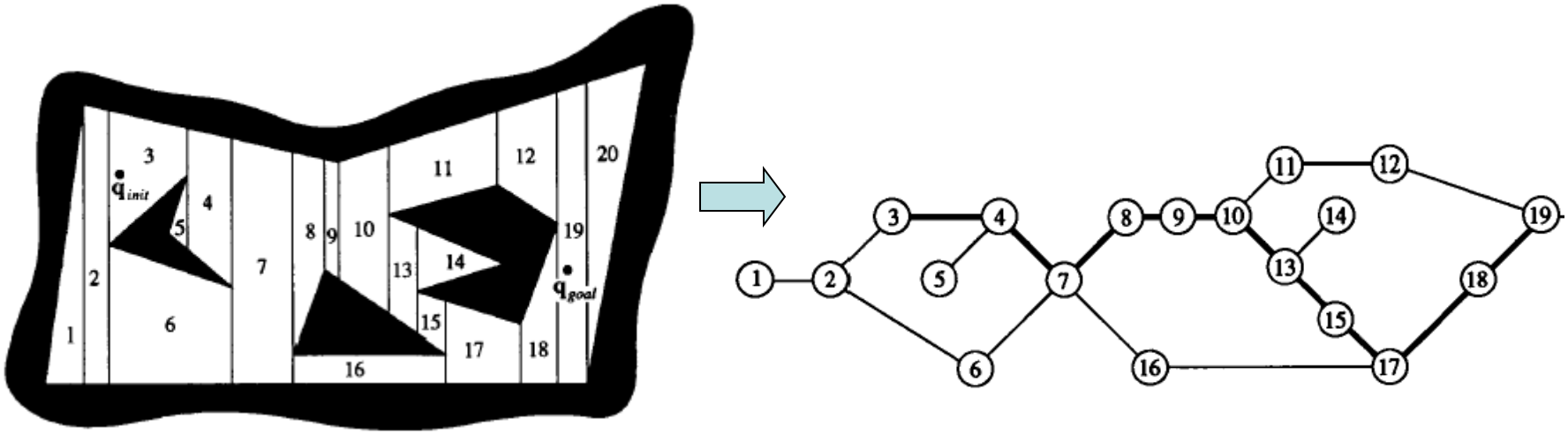
Robotics

Exact Cell Decomposition

Trapezoidal Decomposition:

Decomposition of the free space into trapezoidal & triangular cells

Connectivity graph representing the adjacency relation between the cells

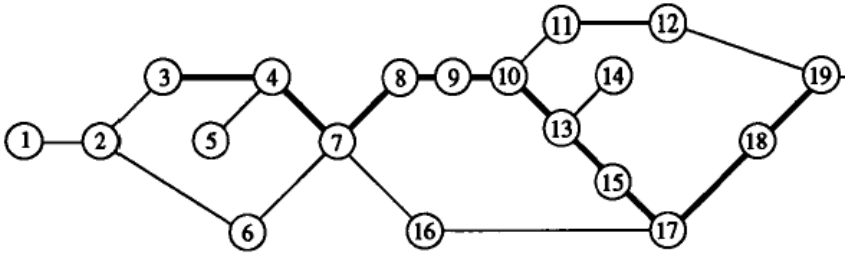


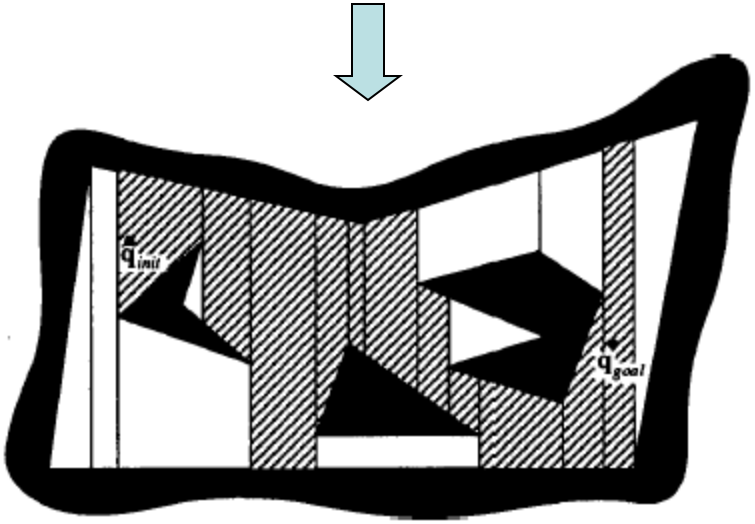
(Sweepline algorithm)

Robotics

Exact Cell Decomposition

Trapezoidal Decomposition:



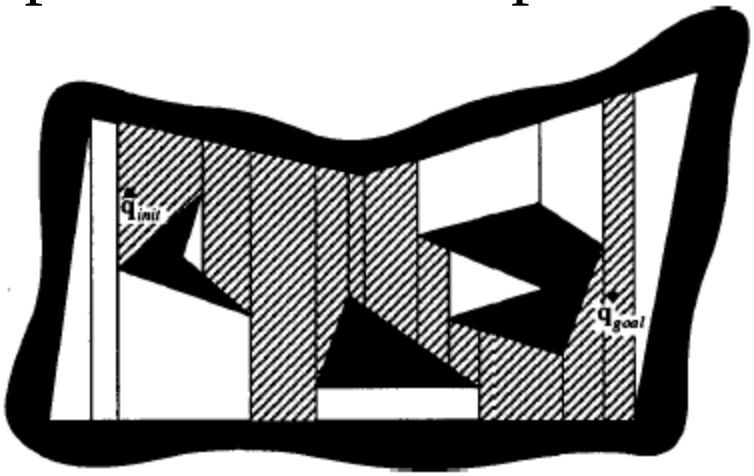


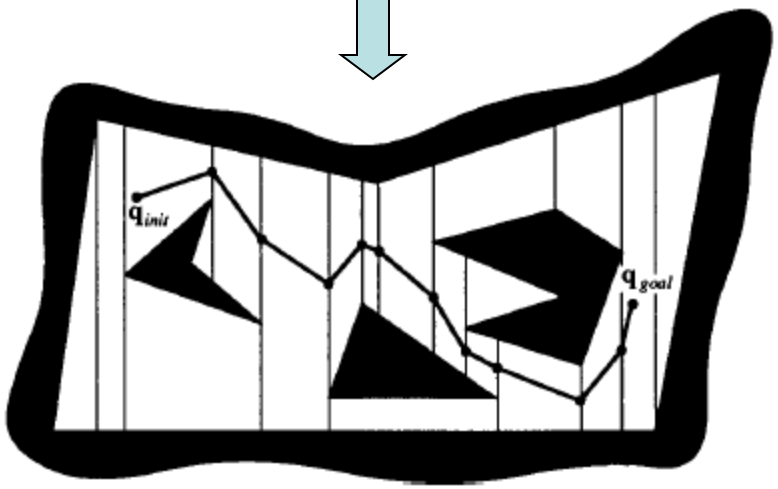
Search the graph for a path
(sequence of consecutive cells)

Robotics

Exact Cell Decomposition

Trapezoidal Decomposition:



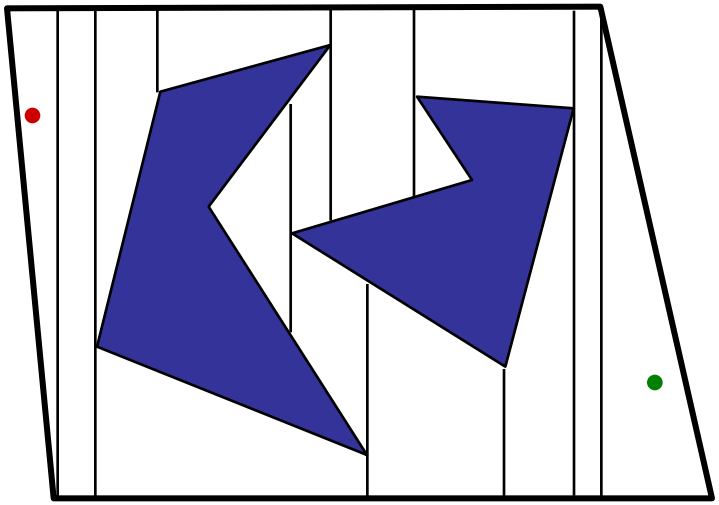


Transform the sequence of cells into a free path (e.g., connecting the mid-points of the intersection of two consecutive cells)

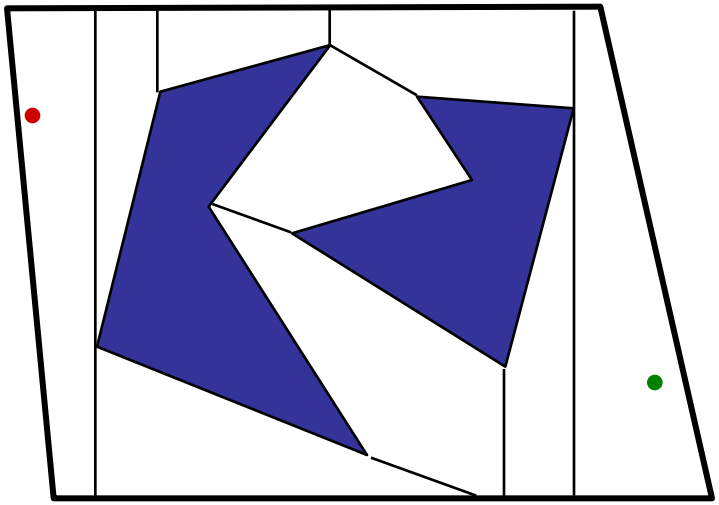
Kobotics

Optimality

Trapezoidal Decomposition:



15 cells



9 cells

Trapezoidal decomposition is exact and complete, but not optimal

Obtaining the *minimum* number of convex cells is NP-complete.

there may be more details in the world than the task needs to worry about...

Potential Field Method

Potential Field (Working Principle)

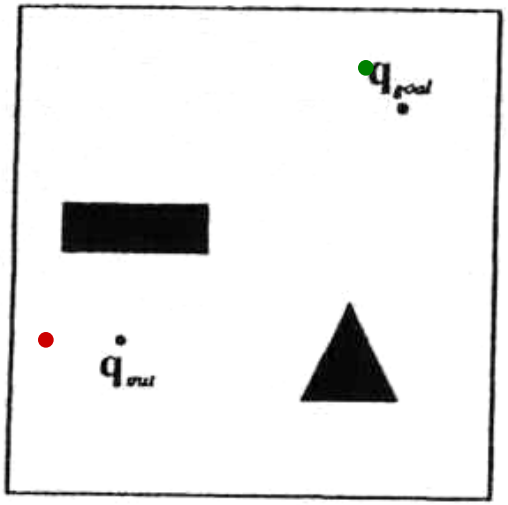
– The goal location generates an **attractive potential** – pulling the robot towards the goal

– The obstacles generate a **repulsive potential** – pushing the robot far away from the obstacles

– The **negative gradient of the total potential** is treated as an artificial force applied to the robot

-- Let the sum of the forces control the robot

C-obstacles



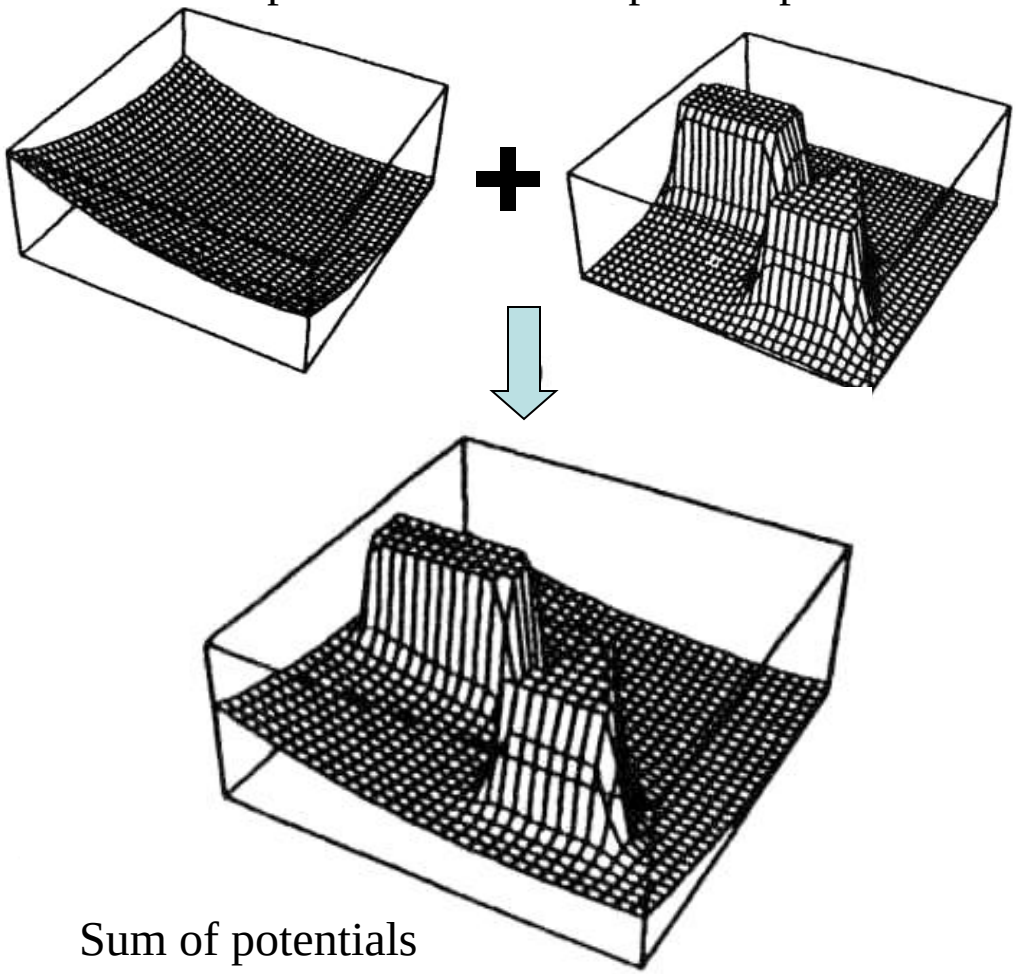
Robotics

Potential Field Method

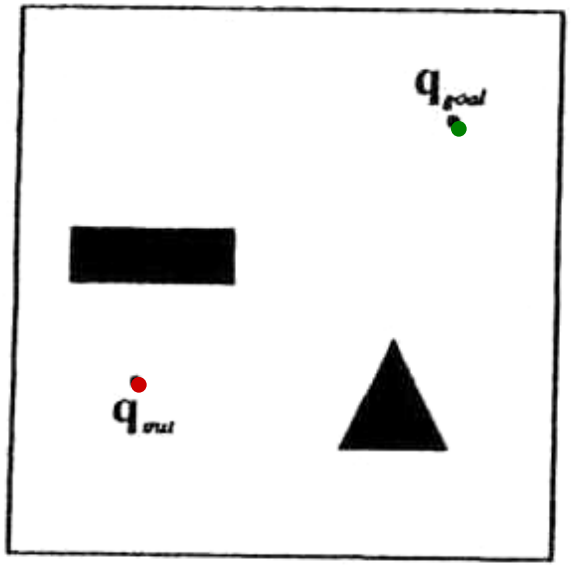
- Sum of Potential

Attractive potential

Repulsive potential



C-obstacle



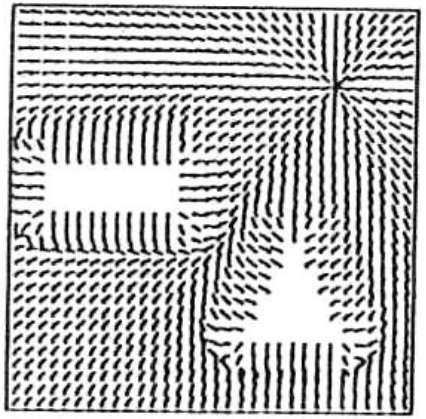
Sum of potentials

Robotics

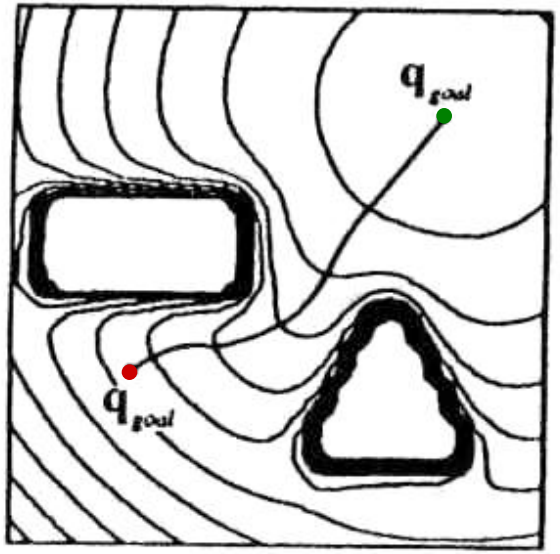
Potential Field Method

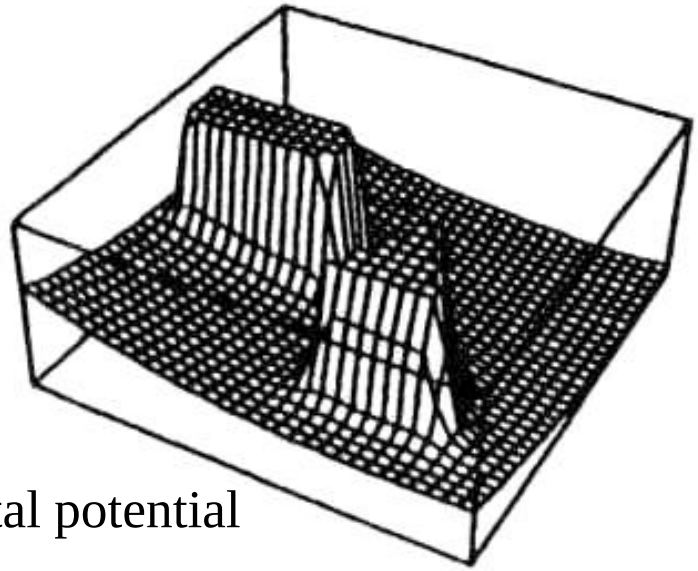
- After get total potential, generate force field (negative gradient)
- Let the sum of the forces control the robot

Negative gradient



Equipotential contours





Total potential

To a large extent, this is computable from sensor readings

Robotics

Potential Field Method

Pros:

- Spatial paths are not preplanned and can be generated in real time
- Planning and control are merged into one function
- Smooth paths are generated
- Planning can be coupled directly to a control algorithm

Cons:

- Trapped in local minima in the potential field
- Because of this limitation, commonly used for local path planning
- Use random walk, backtracking, etc to escape the local minima

Motion Planning Methods

Roadmap approaches

- Visibility Graph

Cell decomposition

- Trapezoidal decomposition

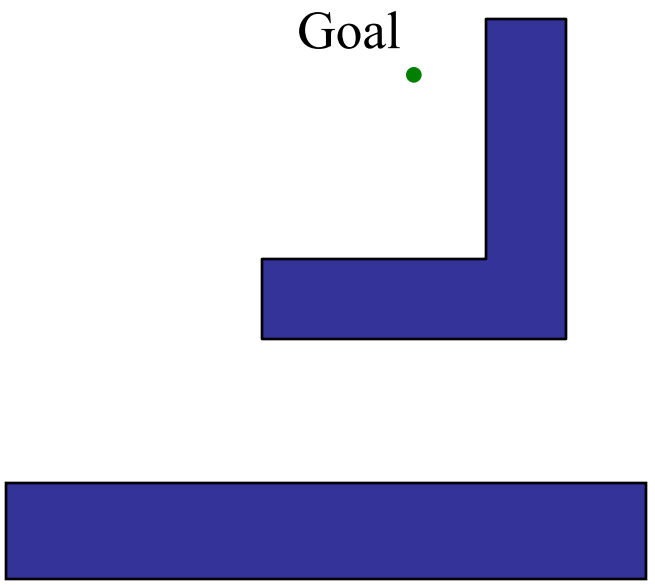
Potential Fields

Full-knowledge motion planning

Bug algorithm — Limited-knowledge path planning

Bug Algorithms

Path planning with limited knowledge
- Insect-inspired "bug" algorithms



- known direction to goal
- only local sensing (walls/obstacles encoders)
- "reasonable" world

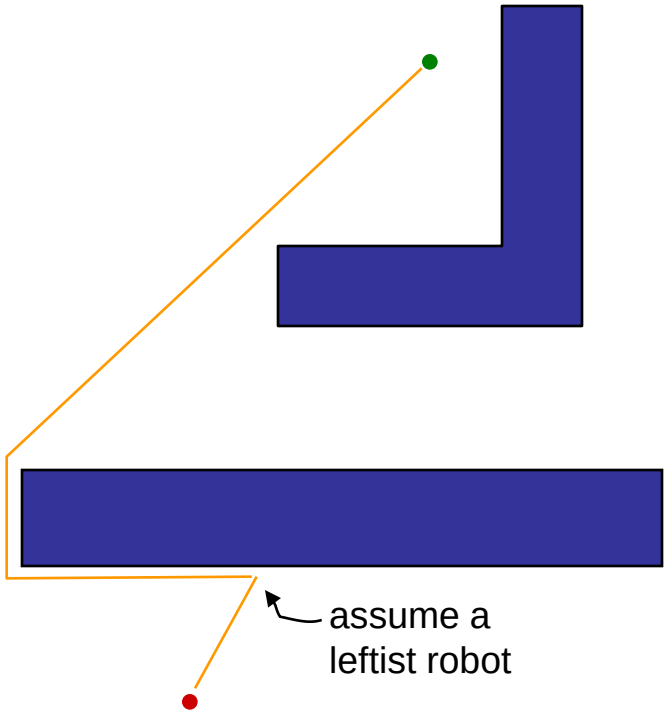
1) finite obstacles in any finite range

2) a line will intersect an obstacle finite times

Robotics

Beginner Strategy

Insect-inspired "bug" algorithms



Switching between two simple behaviors:

1. Moving directly towards the goal
2. Circumnavigating an obstacle

"Bug" algorithm

1) head toward goal

2) follow obstacles until you can head toward the goal again

3) continue

Robotics

Summary

Configuration Space

Motion Planning Methods

- Roadmap approaches
- Cell decomposition
- Potential Fields
- Bug Algorithms

References

• G. Dudek, M. Jenkin, *Computational Principles of Mobile Robots*, MIT Press, 2000 (Chapter 5)

• J.C. Latombe, *Robot Motion Planning*, Kluwer Academic Publishers, 1991.

– Path Planning with A* algorithm

• S. Kambhampati, L. Davis, "Multiresolution Path Planning for Mobile Robots", IEEE Journal of Robotrics and Automation,Vol. RA-2, No.3, 1986, pp.135-145.

– Potential Field

• O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", Int. Journal of Robotics Research, 5(1), pp.90-98, 1986.

• P. Khosla, R. Volpe, "Superquadratic Artificial Potentials for Obstacle Avoidance and Approach" Proc. Of ICRA, 1988, pp.1178-1784.

• B. Krogh, "A Generalized Potential Field Approach to Obstacle Avoidance Control" SME Paper MS84-484.

Thank you!

Robotics