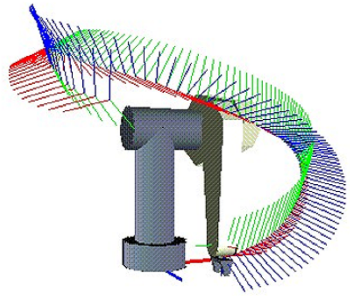
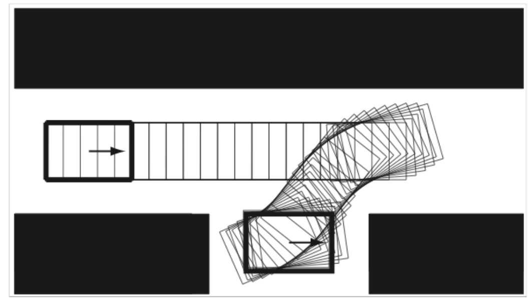




Motion planning

Chapter 10

Introduction to Robotics:

Mechanics, Planning, and Control

Frank Park and Kevin Lynch

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

Some definitions

- q is the configuration (e.g., $q = (\theta_1, \ldots, \theta_n)$)
- C is the configuration space (C-space)
- C_{free} is the collision-free C-space
- $x = (q, v)$ is the state
- $q(x)$ returns the configuration associated with x
- u is a control
- U is the set of possible controls

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

The problem definition

Equations of motion:

$$\dot{x} = f(x, u)$$

or $$x(T) = x(0) + \int_0^T f(x(t), u(t))dt \qquad (10.2)$$

Given an initial state $x(0) = x_{start}$ *and a desired final state* x_{goal}*, find a time* T *and a set of controls* $u : [0, T] \to \mathcal{U}$ *such that the motion* (10.2) *satisfies* $x(T) = x_{goal}$ *and* $q(x(t)) \in \mathcal{C}_{free}$ *for all* $t \in [0, T]$.

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

Types of motion planning problems

- Path planning (piano mover's) vs. motion planning
- Fully actuated vs. underactuated (e.g., a car)
- Online vs. offline
- Optimal vs. satisficing
- Exact vs. approximate
- With vs. without obstacles

Properties of motion planners

- Multiple queries vs. single queries
- "Anytime" planning
- Completeness
  - complete
  - resolution complete
  - probabilistically complete
- Computational complexity

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

Motion planning methods

- Complete methods
- Grid methods
- Sampling methods
- Virtual potential fields
- Nonlinear optimization

A summary of (many) path planning methods

robot + obstacles →

discretized graph representation of C_{free} →

search of that graph for a path from q_{start} to q_{goal}

C-space obstacles: 2R arm

Robot's configuration is treated as a point in C-space.
Q: How many connected components does C_{free} have?

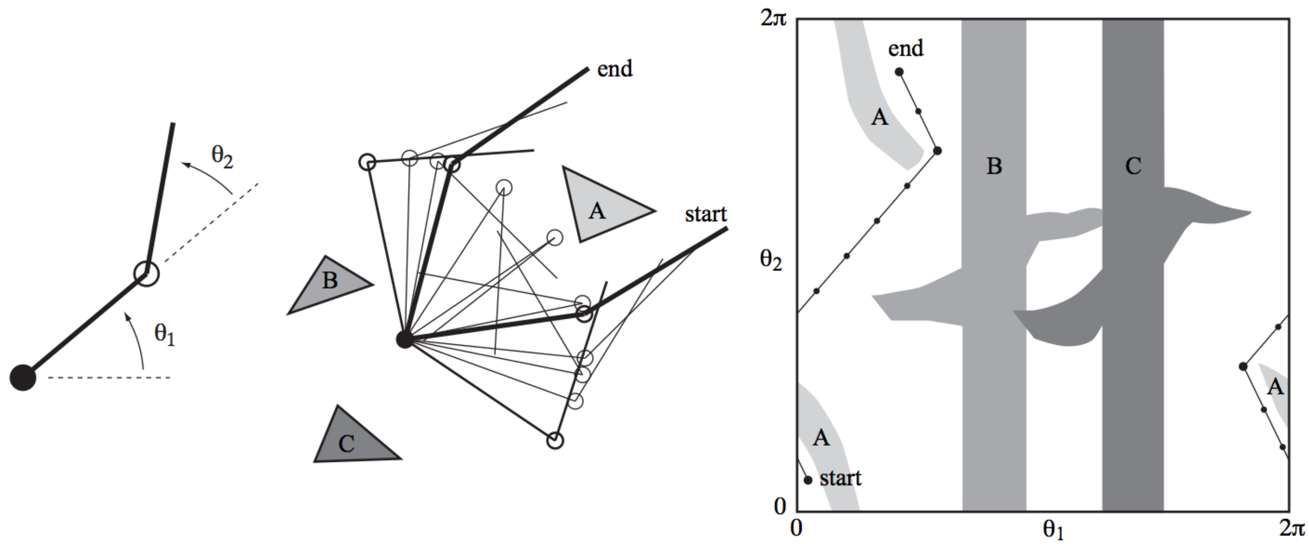


Figure 10.2: (Left) The joint angles of a 2R robot arm. (Middle) The arm navigating among obstacles. (Right) The same motion in C-space.

C-space obstacles: circular mobile robot

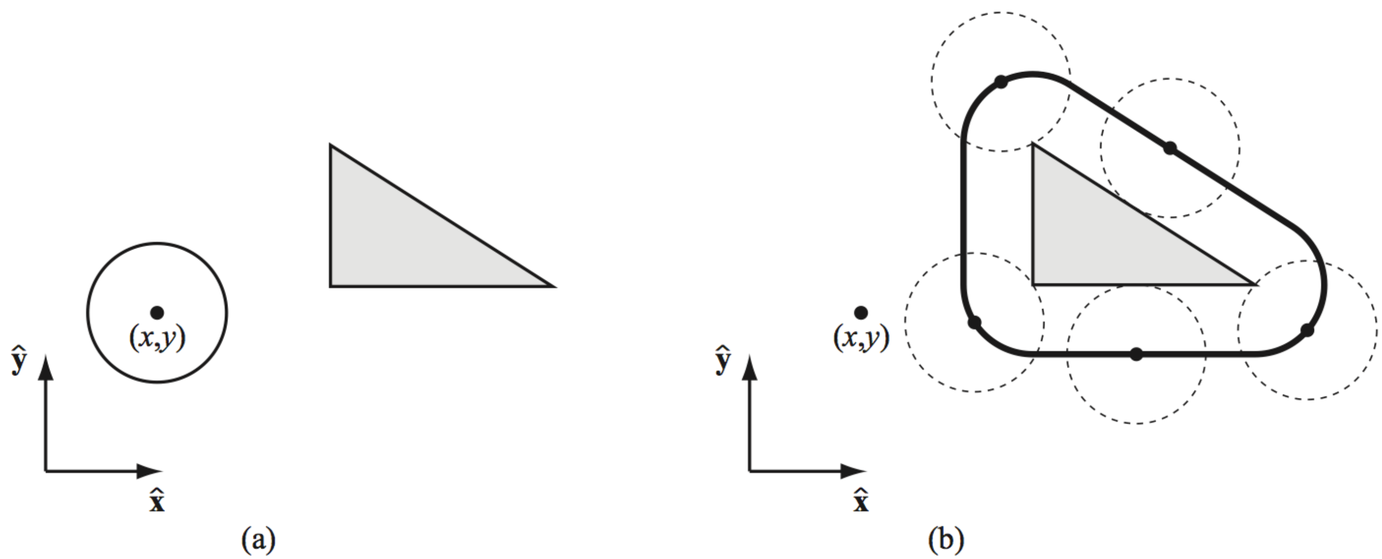
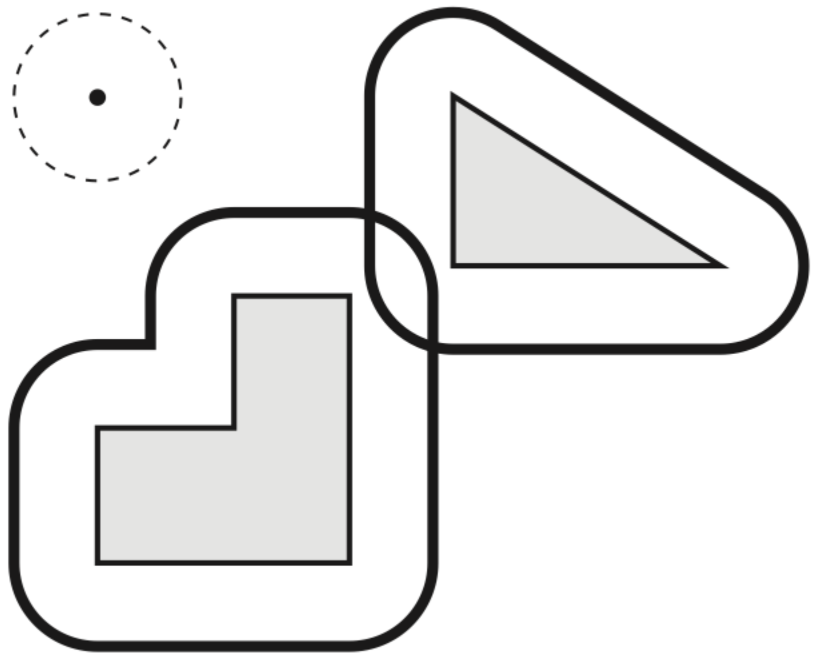


Figure 10.3: (a) A circular mobile robot (white) and a workspace obstacle (grey). The configuration of the robot is represented by (x, y), the center of the robot. (b) In the C-space, the obstacle is "grown" by the radius of the robot and the robot is treated as a point. Any (x, y) configuration outside the dark boundary is collision-free.

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

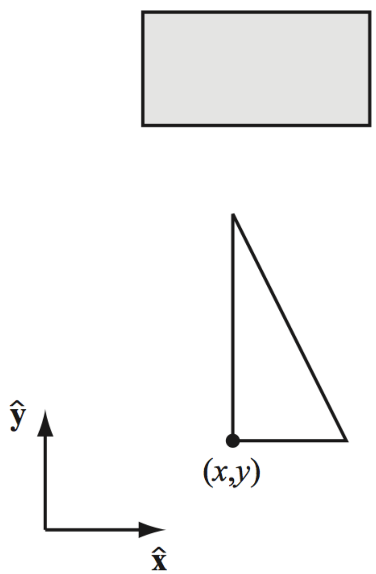
C-space obstacles: circular mobile robot

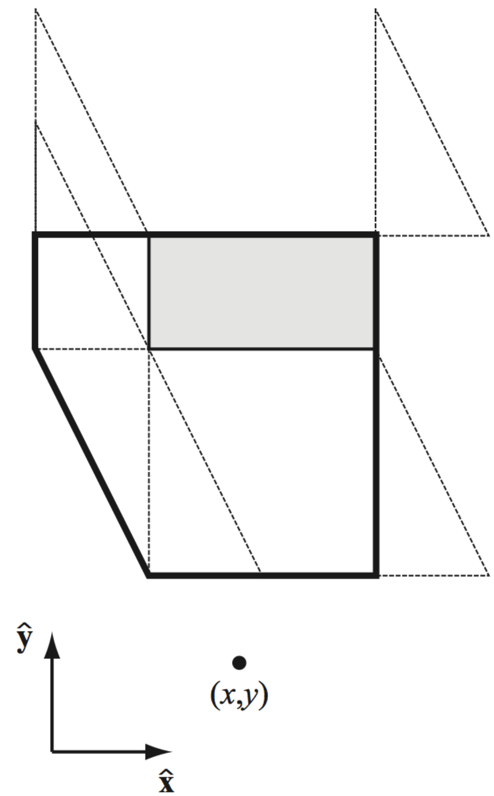


SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

C-space obstacles: polygonal mobile robot

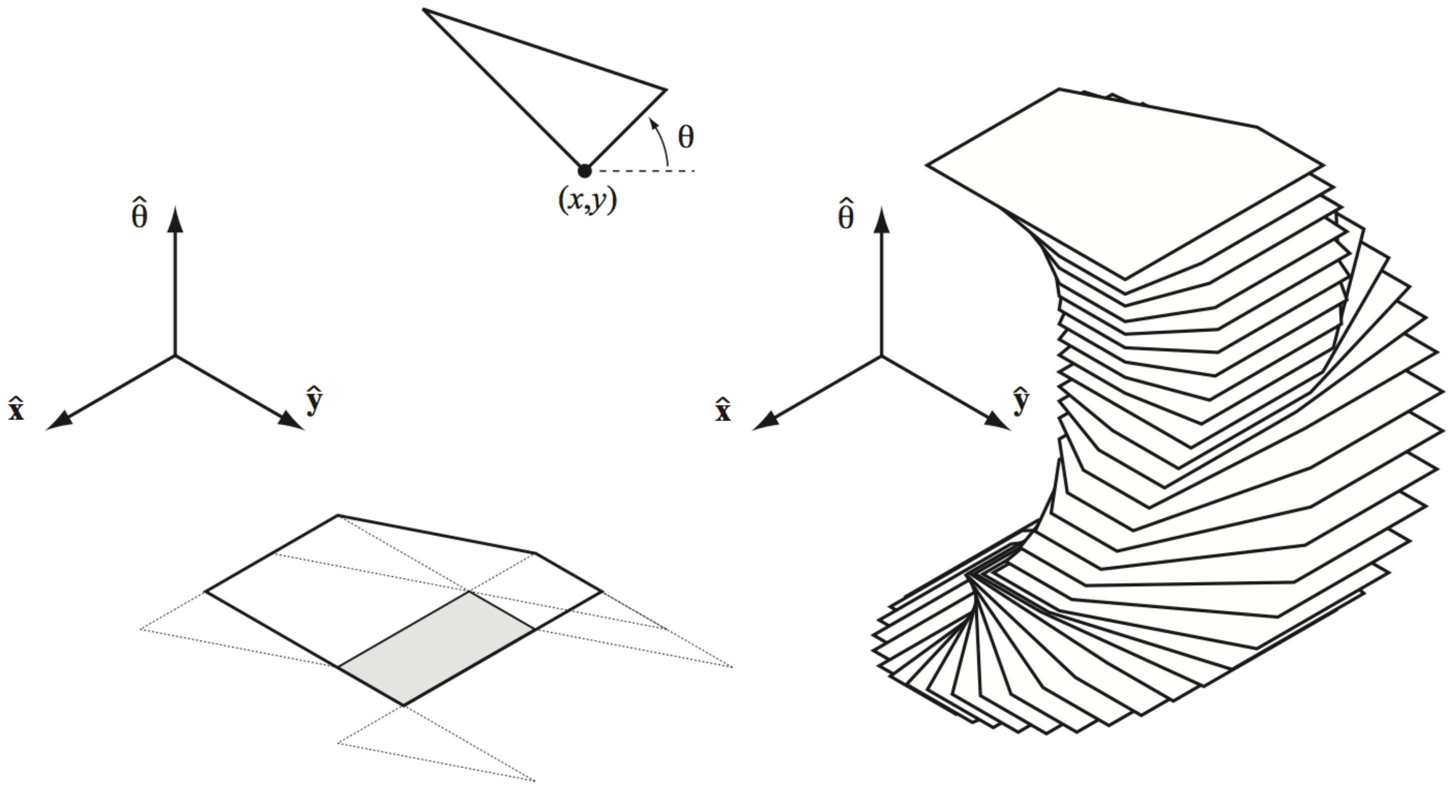




SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

C-space obstacles: rotating/translating polygon



SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

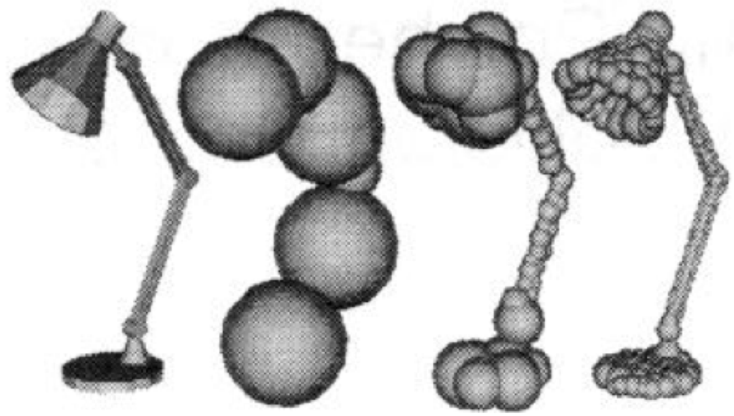
Distance to C-obstacle $\mathcal{B}$

$$
\begin{array}{ll}
d(q,\mathcal{B}) > 0 & \text{no contact with the obstacle} \\
d(q,\mathcal{B}) = 0 & \text{contact} \\
d(q,\mathcal{B}) < 0 & \text{penetration.}
\end{array}
$$

A *distance-measurement algorithm* returns *d*.

A *collision-detection algorithm* returns true if *q* is in collision, false otherwise.

An approximation approach



$r_i(q)$ is the center of robot sphere i
R_i is the radius of robot sphere i
b_j is the center of obstacle sphere j
B_j is the radius of obstacle sphere j

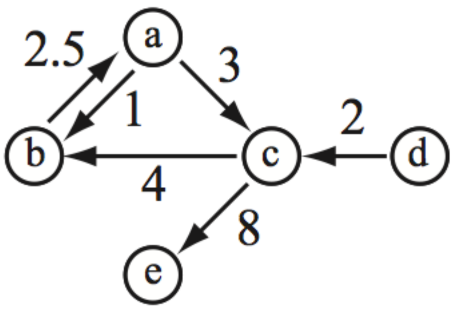
$$d(q, \mathcal{B}) = \min_{i,j} \|r_i(q) - b_j\| - R_i - B_j$$

Other primitives (cylinders, rectangular prisms, meshes or polygon soups) can also be used, with increasing accuracy and computational complexity.

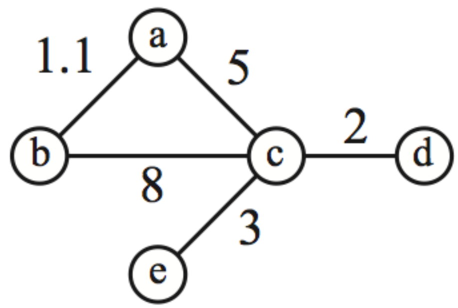
SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

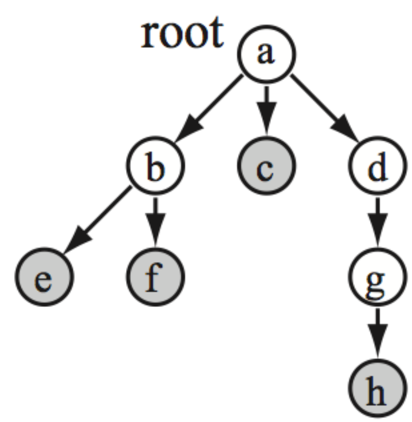
Graphs and trees



weighted directed
graph (digraph)



weighted
undirected graph



tree
(leaves are in gray)

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

Complete path planning using a roadmap

Reduce the complex high-dimensional C_{free} to a one-dimensional roadmap R (or a graph) with the following properties:

(i) *Reachability.* From every point $q \in \mathcal{C}_{\text{free}}$, a free path to a point $q' \in R$ can be found trivially (e.g., a straight-line path).

(ii) *Connectivity.* For each connected component of $\mathcal{C}_{\text{free}}$, there is one connected component of R.

SEOUL NATIONAL UNIVERSITY

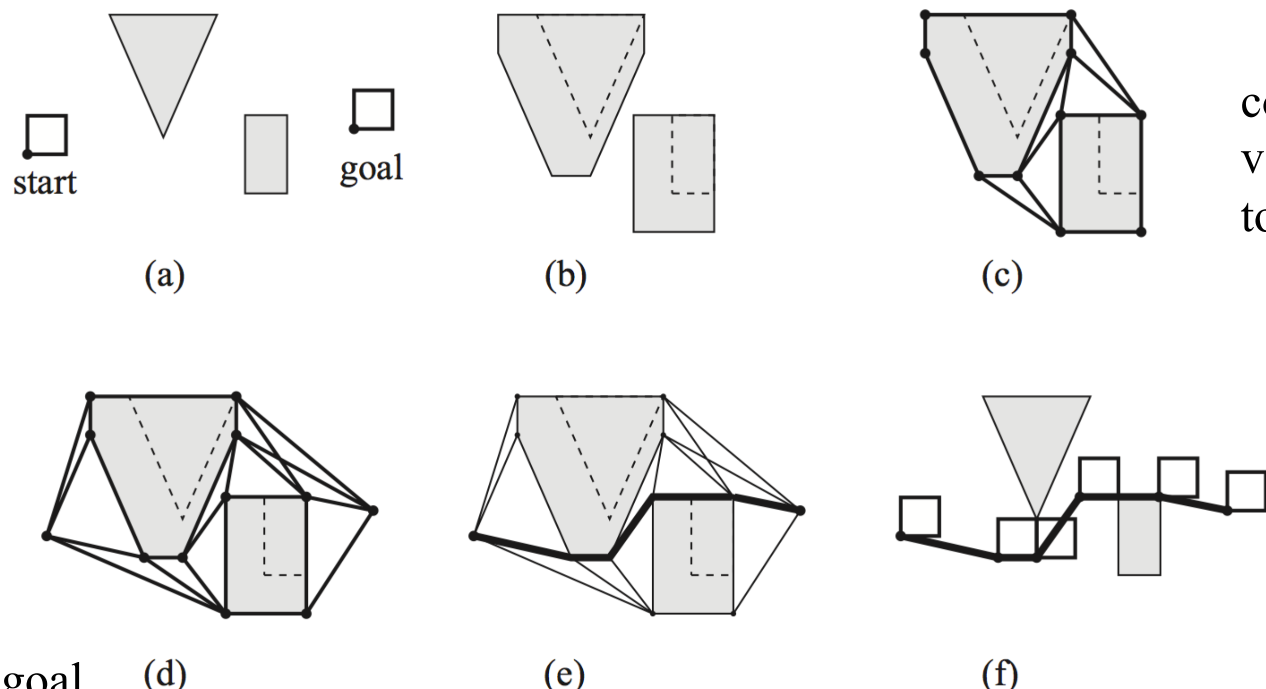
NORTHWESTERN UNIVERSITY

Constructing a roadmap

Very difficult to do in general (see John Canny's PhD thesis, 1988), but

- easy for some simple problems
- approximations are possible (e.g., probabilistic roadmap PRM)

Visibility graph for translating polygon

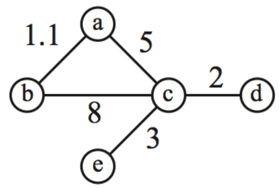


connect all vertices visible to each other

connect start and goal

This weighted graph roadmap allows direct search for a shortest path.

Finding a shortest path: A* search



- Keep a list `OPEN` of nodes reached so far in the search, sorted by the estimated cost of a path going through that node to the goal (the sum of the minimum cost known to reach the node, plus an underestimate to reach the goal).
- Pick the first `node` from `OPEN`. If it's at the goal, done.
- For each neighbor of `node` that has not already been searched from, check if the minimum cost known to reach it (and its parent node) should be replaced.
- Insert these nodes in `OPEN`.

Finding a shortest path: A* search

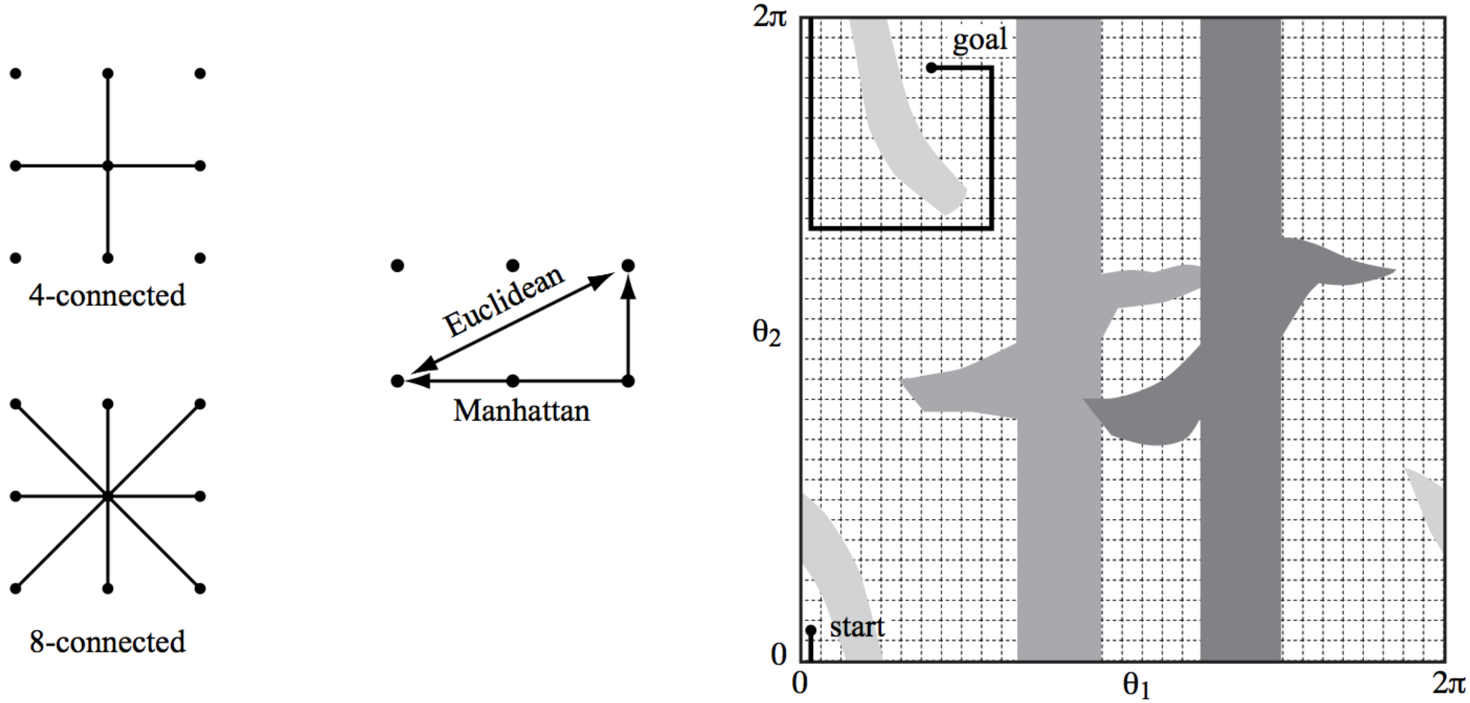
Algorithm 1 A^* search.

```
OPEN ← {1}
past_cost[1] ← 0, past_cost[node] ← infinity for node ∈ {2, ..., N}
while OPEN is not empty do
  current ← first node in OPEN, remove from OPEN
  add current to CLOSED
  if current is in the goal set then
    return SUCCESS and the path to current
  end if
  for each nbr of current not in CLOSED do
    tentative_past_cost ← past_cost[current] + cost[current,nbr]
    if tentative_past_cost < past_cost[nbr] then
      past_cost[nbr] ← tentative_past_cost
      parent[nbr] ← current
      put (or move) nbr in sorted list OPEN according to
          est_total_cost[nbr] ← past_cost[nbr] +
                heuristic_cost_to_go(nbr)
    end if
  end for
end while
return FAILURE
```

SEOUL NATIONAL UNIVERSITY

NORTHWESTERN UNIVERSITY

A* search on a grid



(a) (b) (c)

Need not explicitly construct the graph in advance.