Sampling-Based Methods for Path Planning: Part II

With so many slides and ideas from so many people:

Howie Choset, Nancy Amato, David Hsu, Sonia Chernova, Steve LaValle, James Kuffner, Greg Hager

Mobile Robots vs Robot Manipulators

So far, we've seen two kinds of robots in this class:

Mobile robots:

- Treat the robot as a single rigid object that moves in the plane.
- Can describe position and orientation using $SE(2)$
  - ***If we know the values for $(x, y, \theta) \in SE(2)$, we know everything there is to know about the robot.***

Robot arms:

- A series of links connected by single-dof joints
- Describe the robot using a vector of joint variables
- Describe the position and orientation of the tool (end effector) using $SE(2)$
- The mapping between the joint variables and tool position/orientation can be tricky, but the key idea is this:
  - ***If we know the values for the joint variables, we know everything there is to know about the manipulator.***

For both robots, given the values of some set of variables, we know everything there is to know about the robot (w.r.t. its geometry).

Configuration Space

C-space formalism:
Lozano-Perez 1979

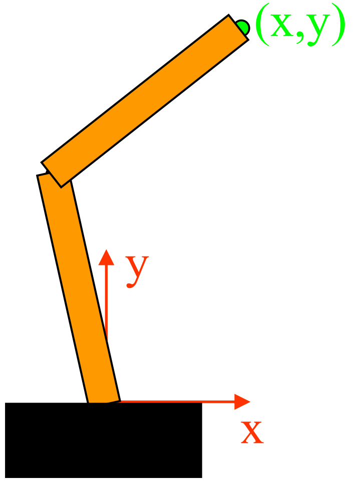
A key concept for motion planning is a **configuration**:

- *A **configuration** of a system is a complete specification of the position of every point in the system*

- The space of all configurations is the ***configuration space*** or ***C-space*****.**

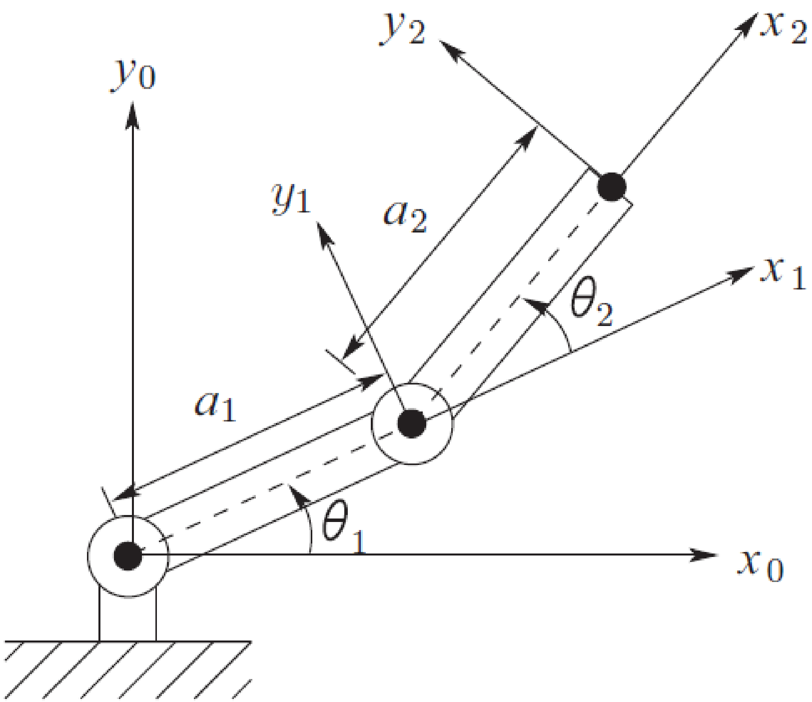
Forward Kinematics

Compute position of some point on the robot, given the values of joint variables



Find (x,y) in terms

of joint angles



$$x = a_1 cos\,\theta_1 + a_2 cos\,(\theta_1 + \theta_2)$$
$$y = a_1 sin\,\theta_1 + a_2 sin\,(\theta_1 + \theta_2)$$

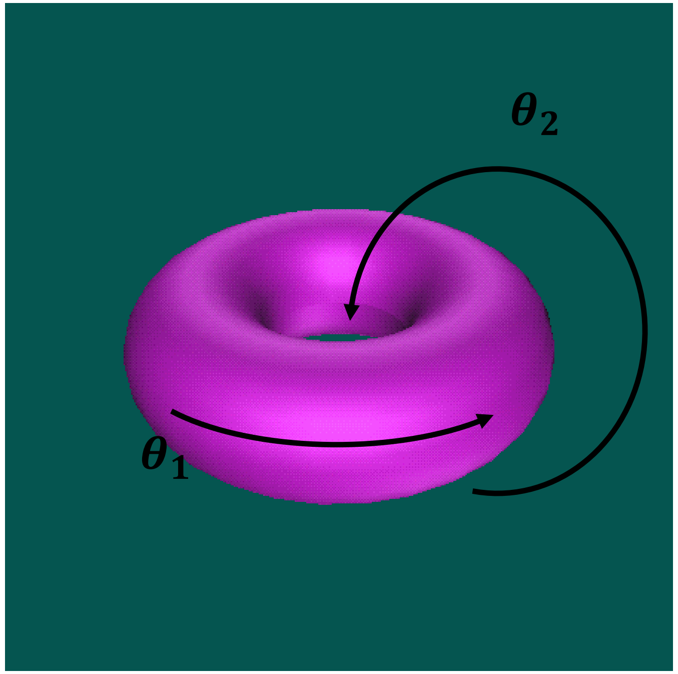
Configuration Space: two-link arm

For the End Effector:

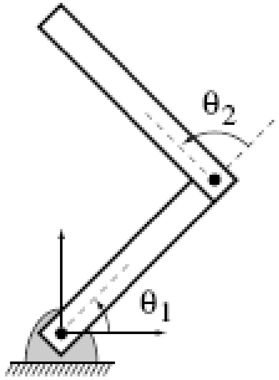
$$x = a_1 \cos\theta_1 + a_2 \cos(\theta_1 + \theta_2)$$
$$y = a_1 \sin\theta_1 + a_2 \sin(\theta_1 + \theta_2)$$

For any point (x, y) on the two-link robot, we can derive a similar set of equations:

- The angles θ_1, θ_2 completely define the configuration of this robot!
- The Configuration Space for this robot is a torus!
- θ_1 is the angle *around the donut*
- θ_2 is the angle *through the hole*

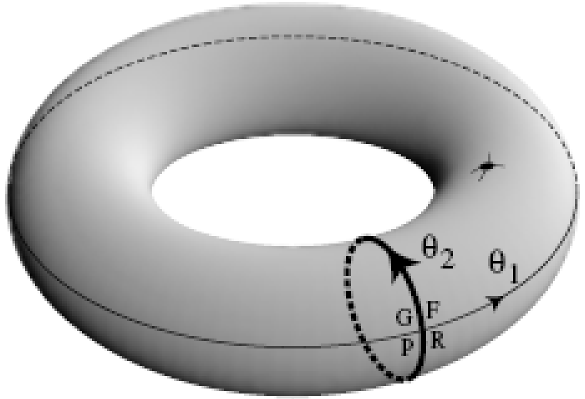


Representing the configuration space

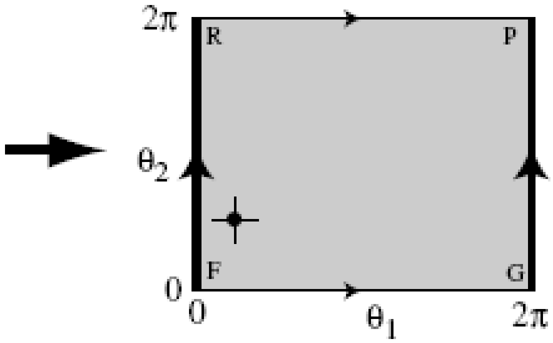


(a)

Represent the torus as a
'square' with 'edges identified'



(b)



(c)

Obstacles in C-Space

- Let q denote a point in a configuration space $\mathcal{Q}$
- The path planning problem is to find a mapping $\gamma: [0,1] \rightarrow \mathcal{Q}$ s.t. no configuration along the path intersects an obstacle.

- Denote the i-th workspace obstacle by $\mathcal{O}_i$, *and by* $R(q)$ *the volume occupied by the robot at configuration* q.
- A *configuration space obstacle* $\mathcal{QO}_i$ is the set of configurations q at which the robot intersects $\mathcal{O}_i$

$$\mathcal{QO}_i = \{ q \in \mathcal{Q} \mid R(q) \cap \mathcal{O}_i \neq \emptyset \}$$

- The *free configuration space* (or just *free space*) $\mathcal{Q}_{free}$ is

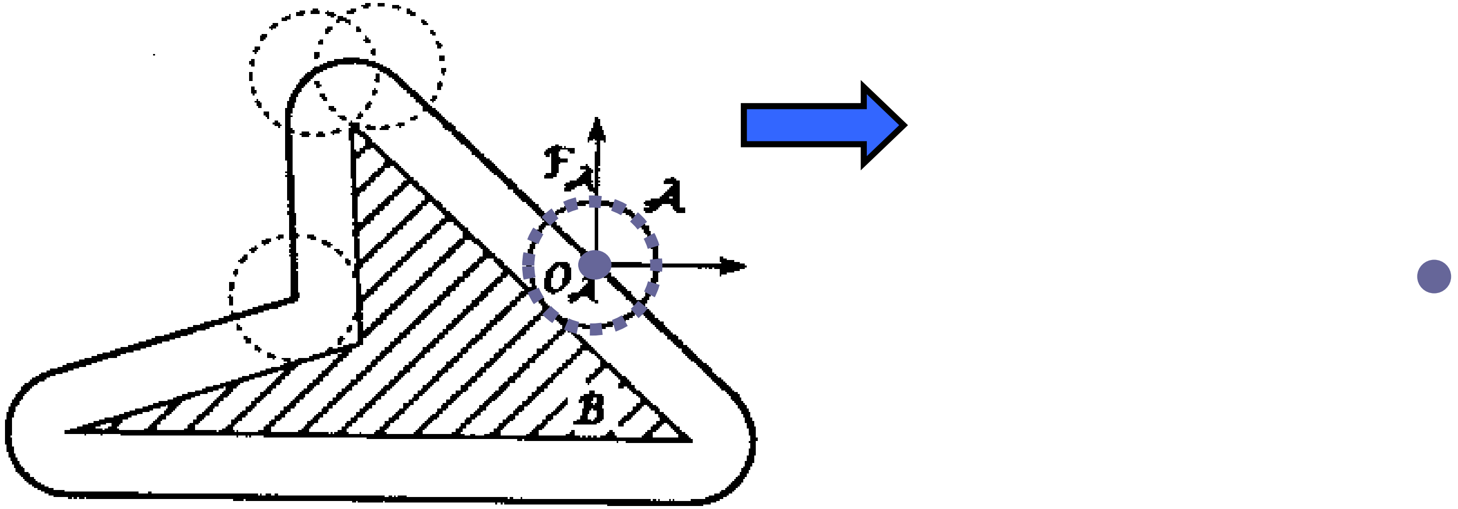
$$\mathcal{Q}_{free} = \mathcal{Q} - \cup_i \mathcal{QO}_i$$

- The free space is generally an open set.
- A *free path* is a mapping $\gamma: [0,1] \rightarrow \mathcal{Q}_{free}$.
- A *semi-free path* is a mapping $\gamma: [0,1] \rightarrow cl(\mathcal{Q}_{free})$.

Disc in 2-D workspace

workspace

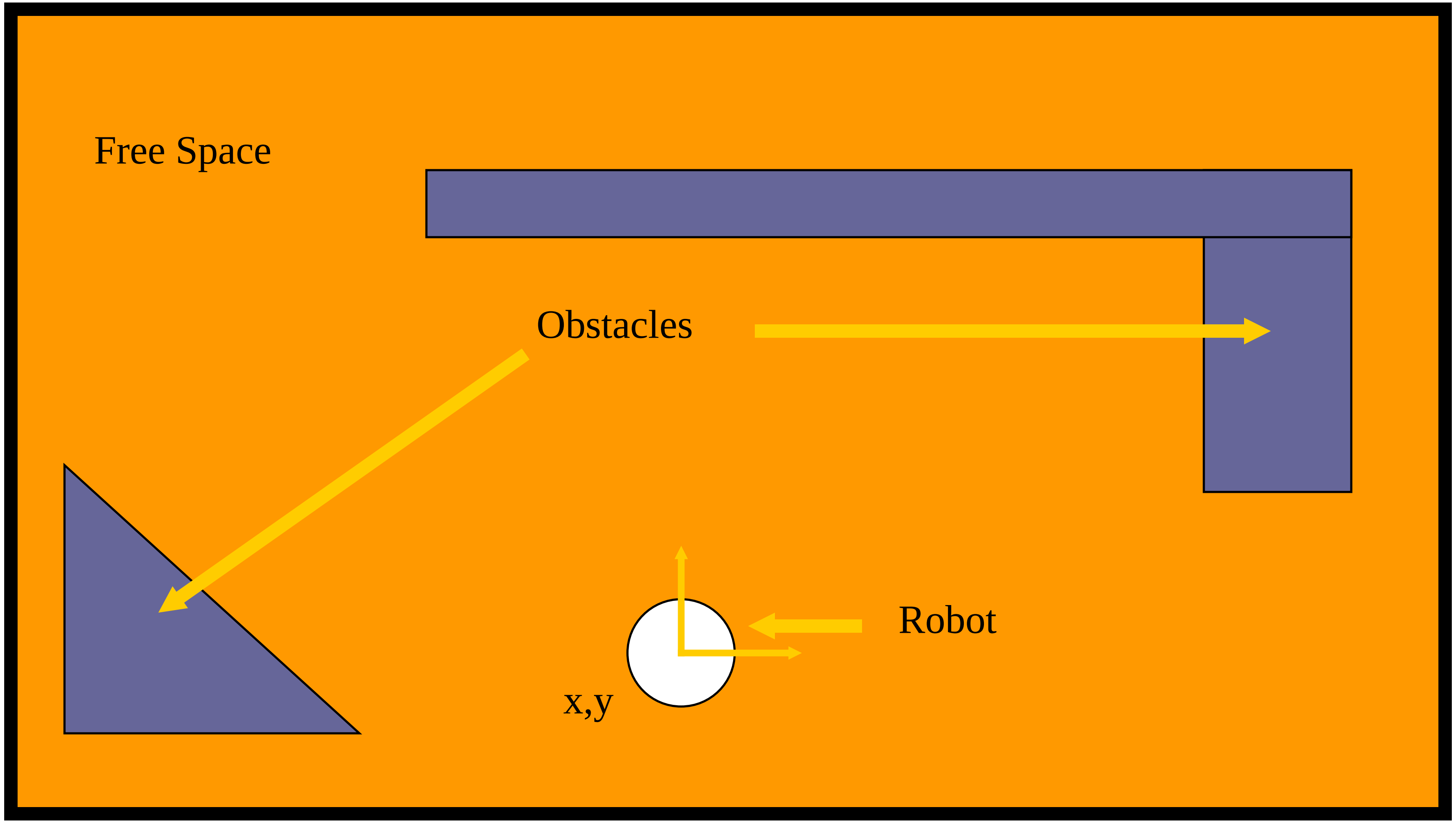
configuration
space



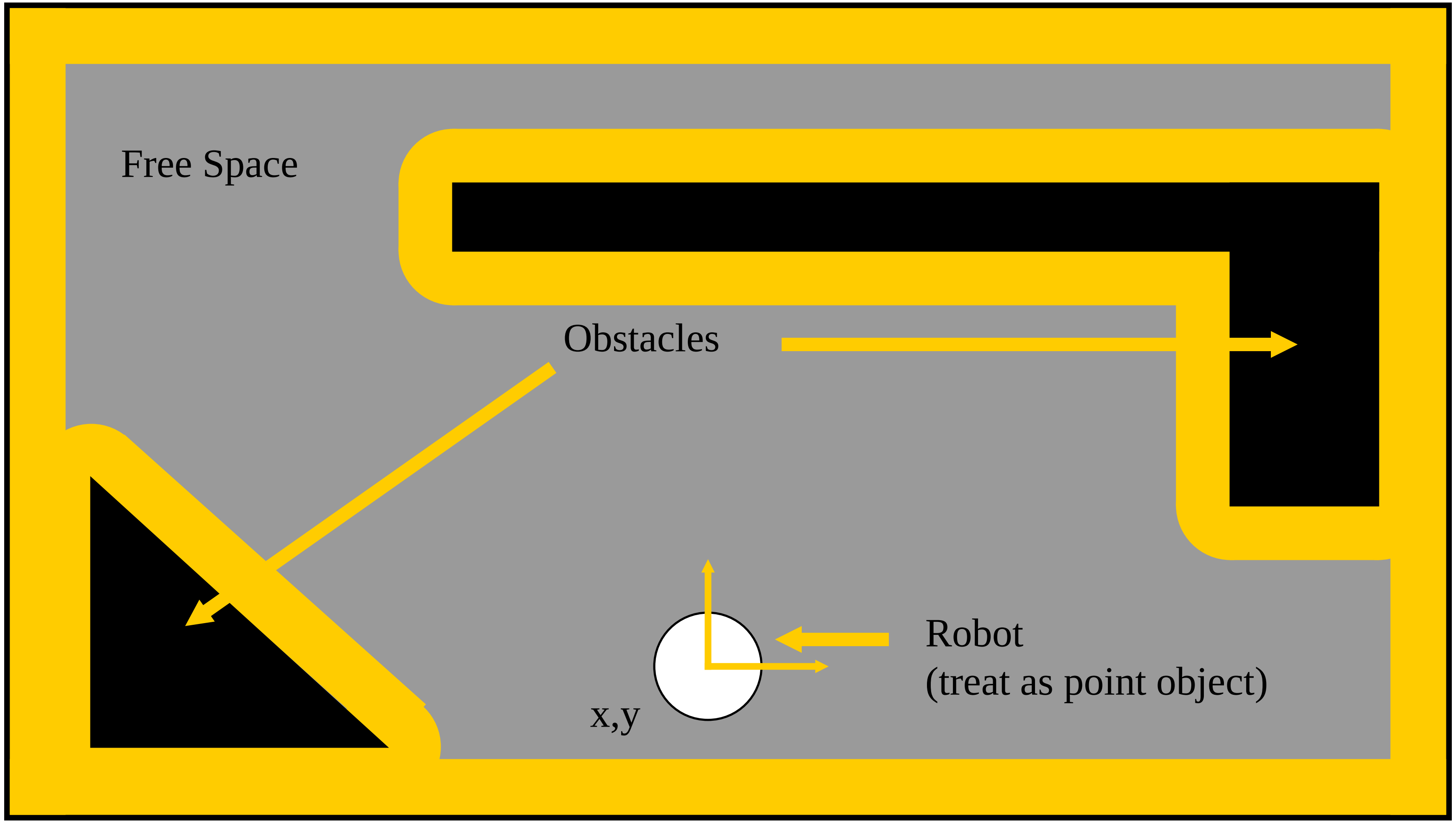
For our application

- Our mobile robot is close enough to being circular that it is fine to model it as a circle with a fixed radius.

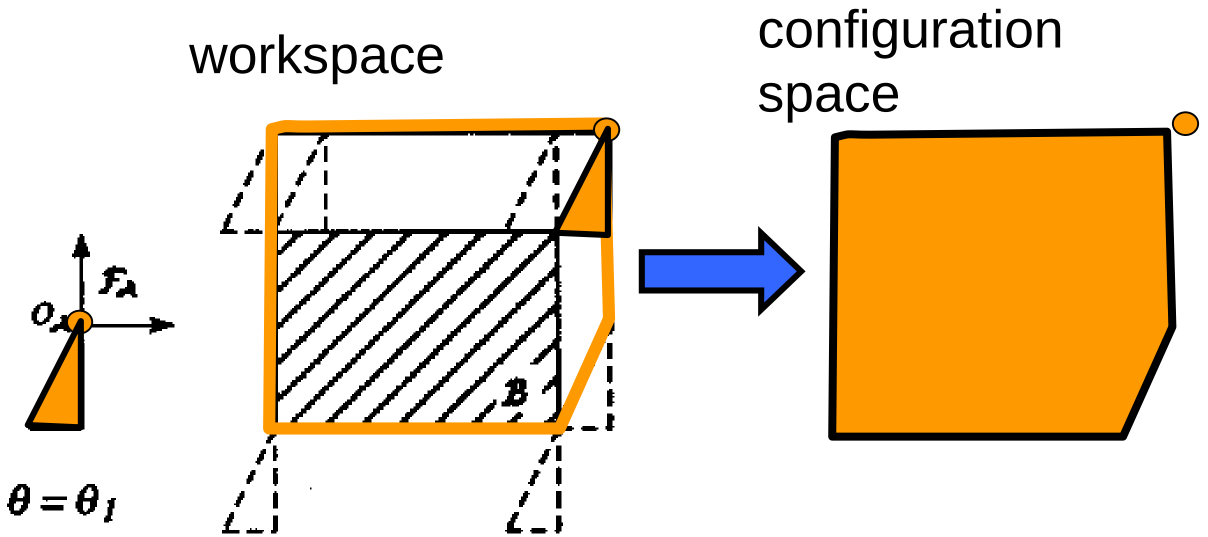
Example of a World (and Robot)



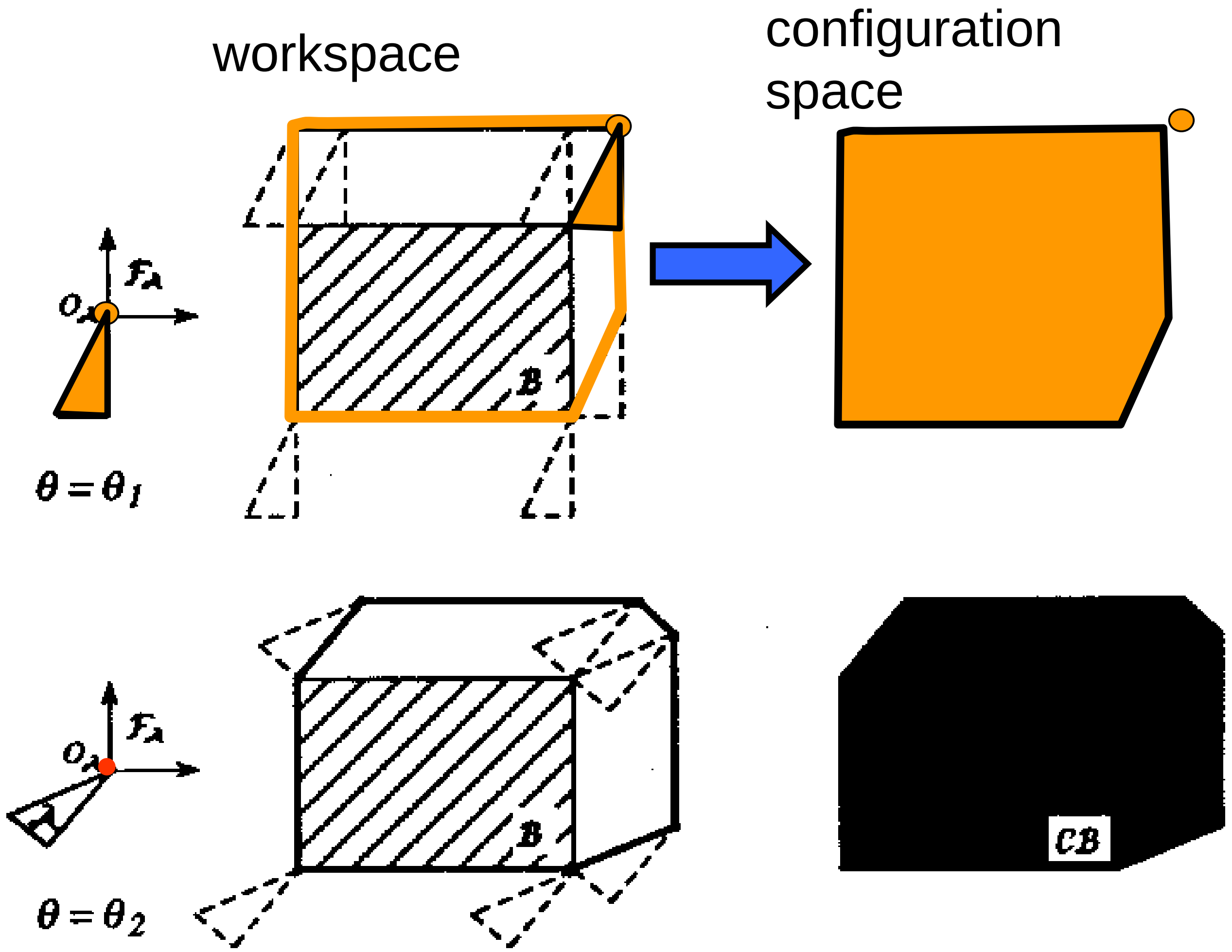
Configuration Space: Accommodate Robot Size



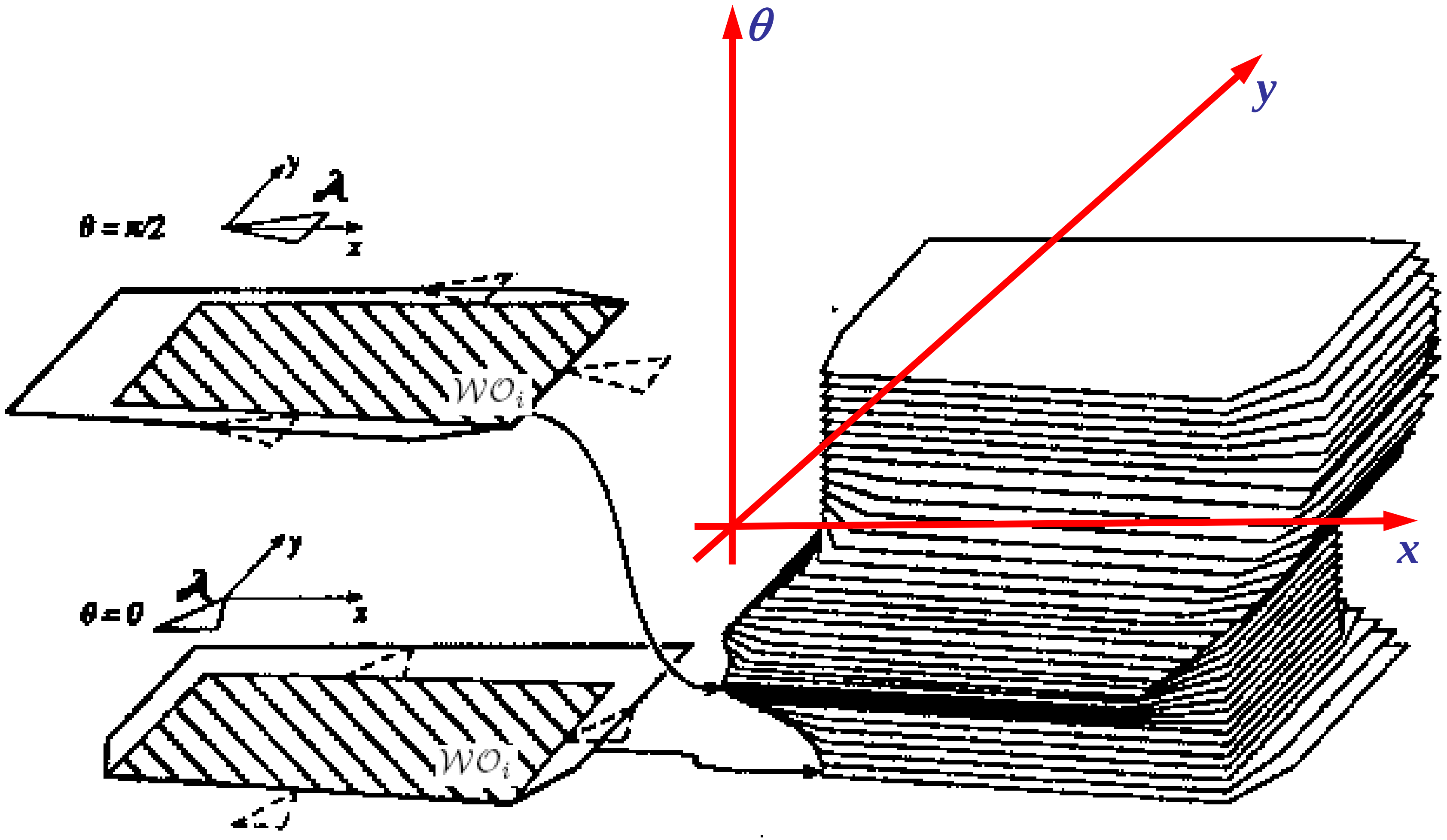
Polygonal robot translating in 2-D workspace



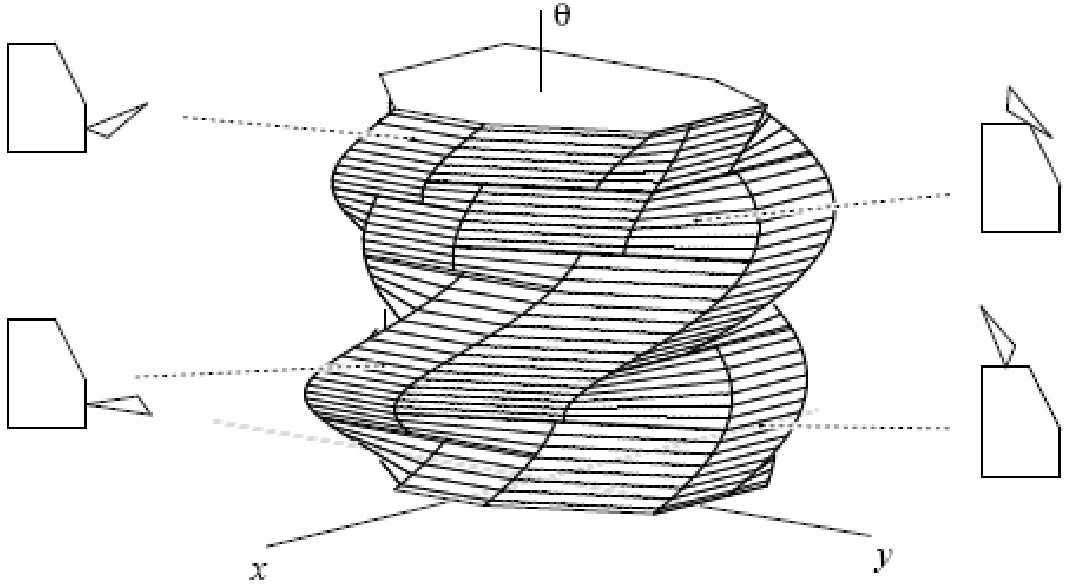
Polygonal robot translating in 2-D workspace



Polygonal robot translating & rotating in 2-D workspace

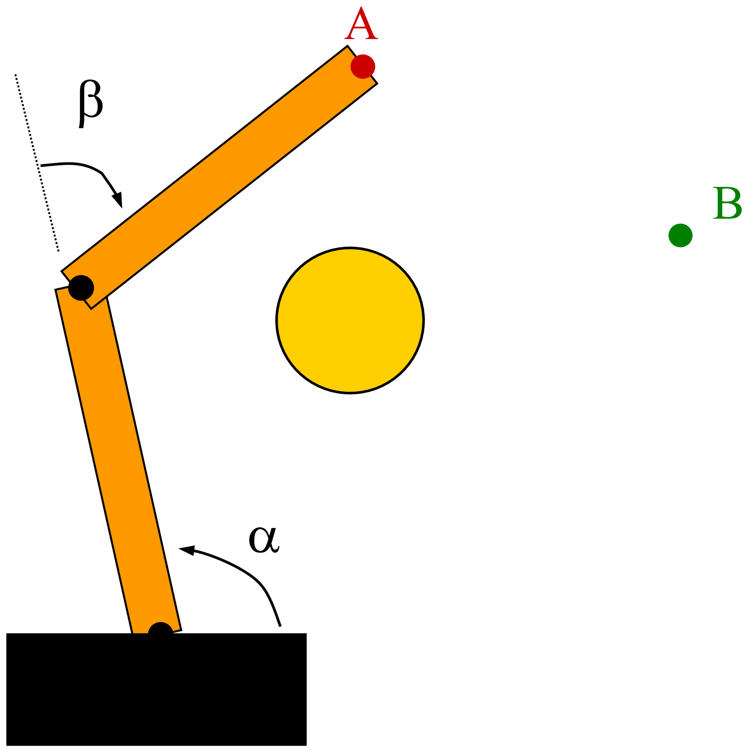


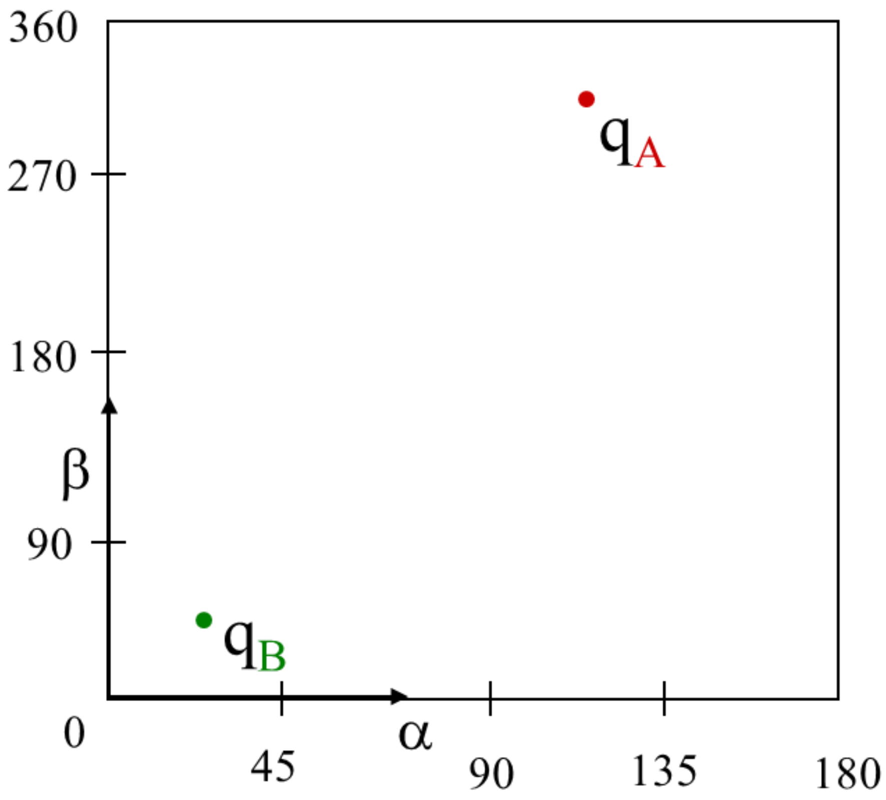
SE(2)



Configuration Space

Where do we put ?

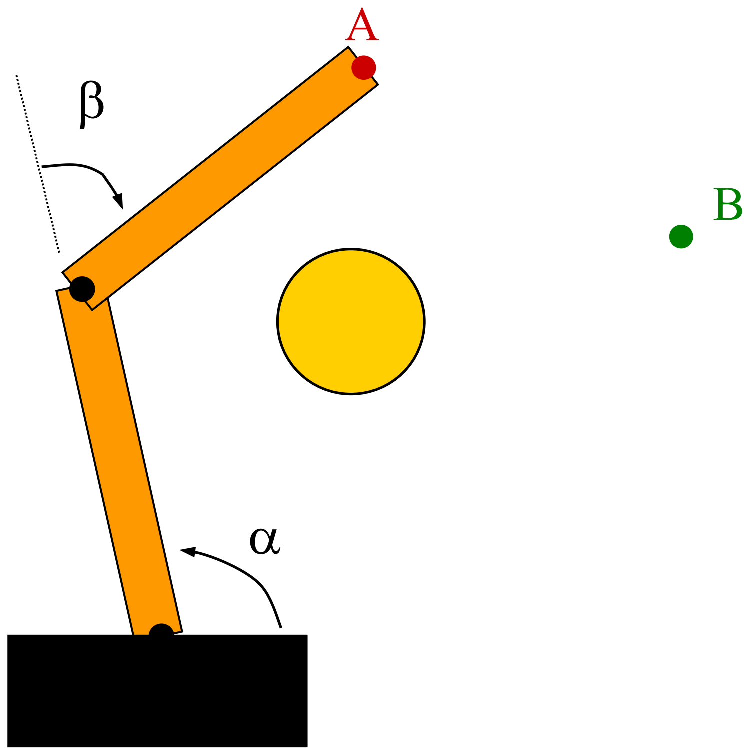


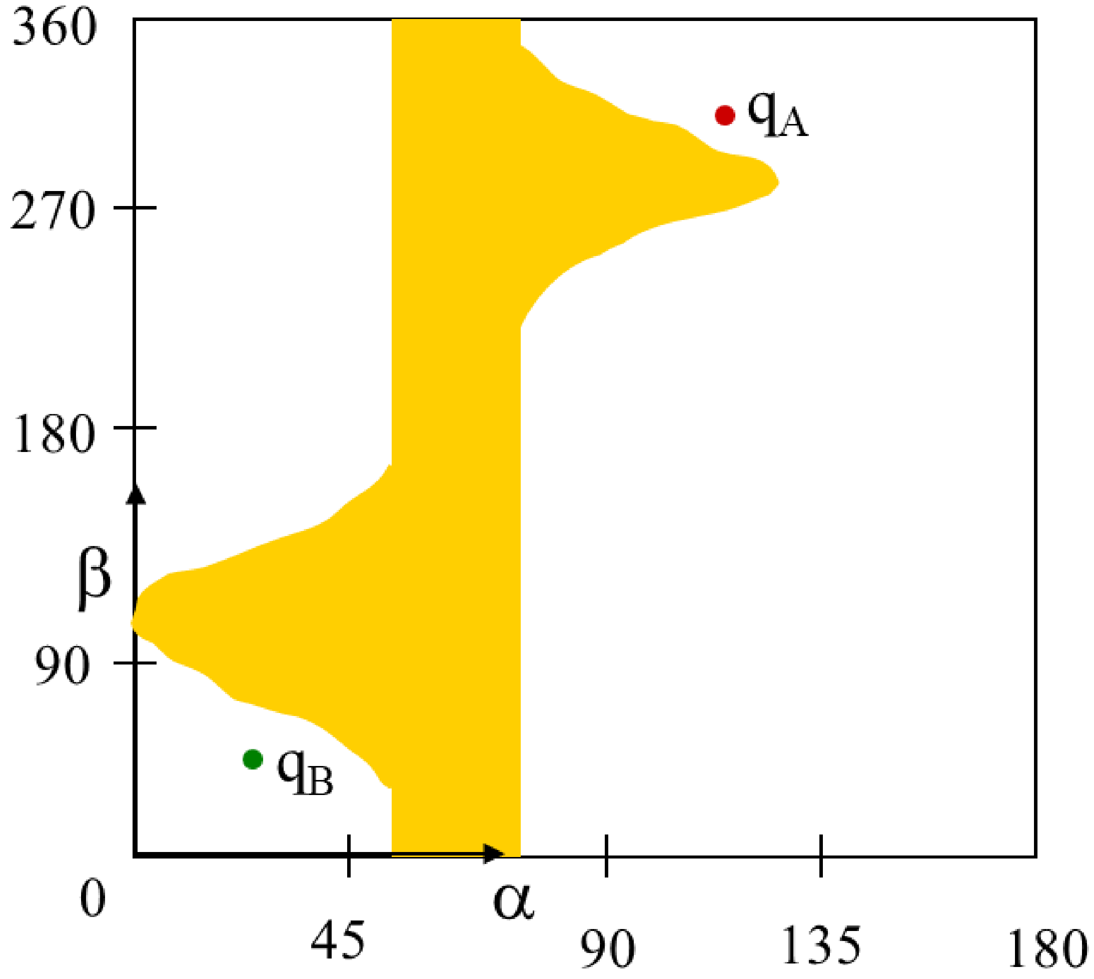


Torus

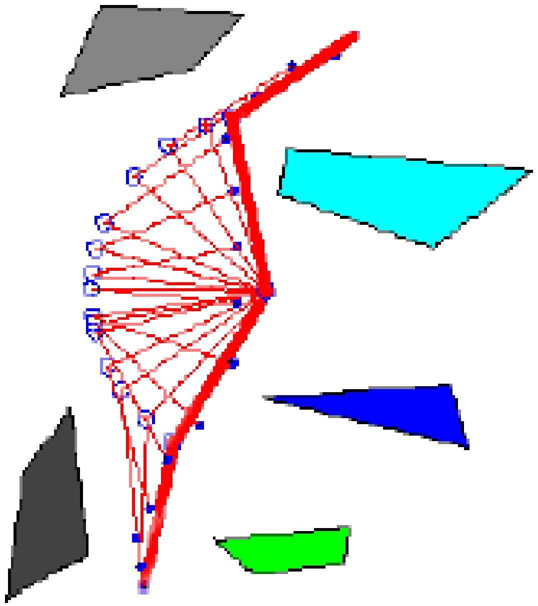
(wraps horizontally and vertically)

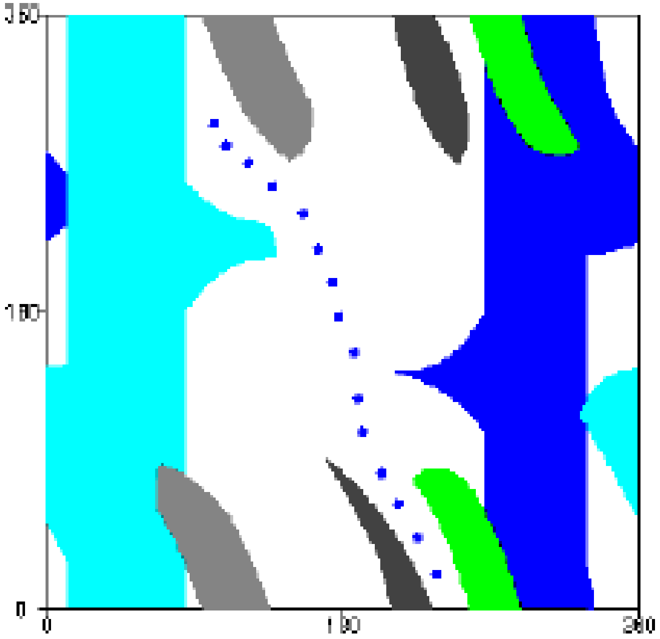
Configuration Space





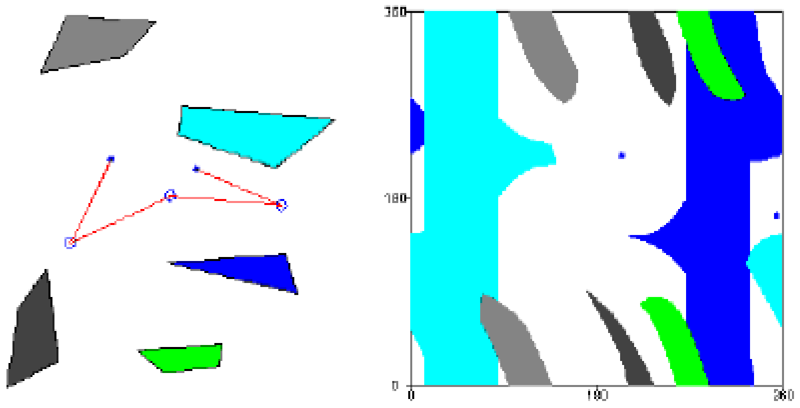
Two Link Path

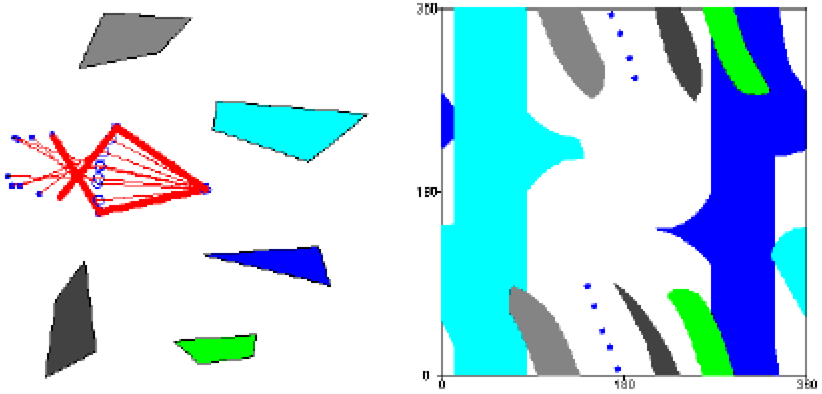




Thanks to Ken Goldberg

Two Link Path





How can we automatically plan these paths?

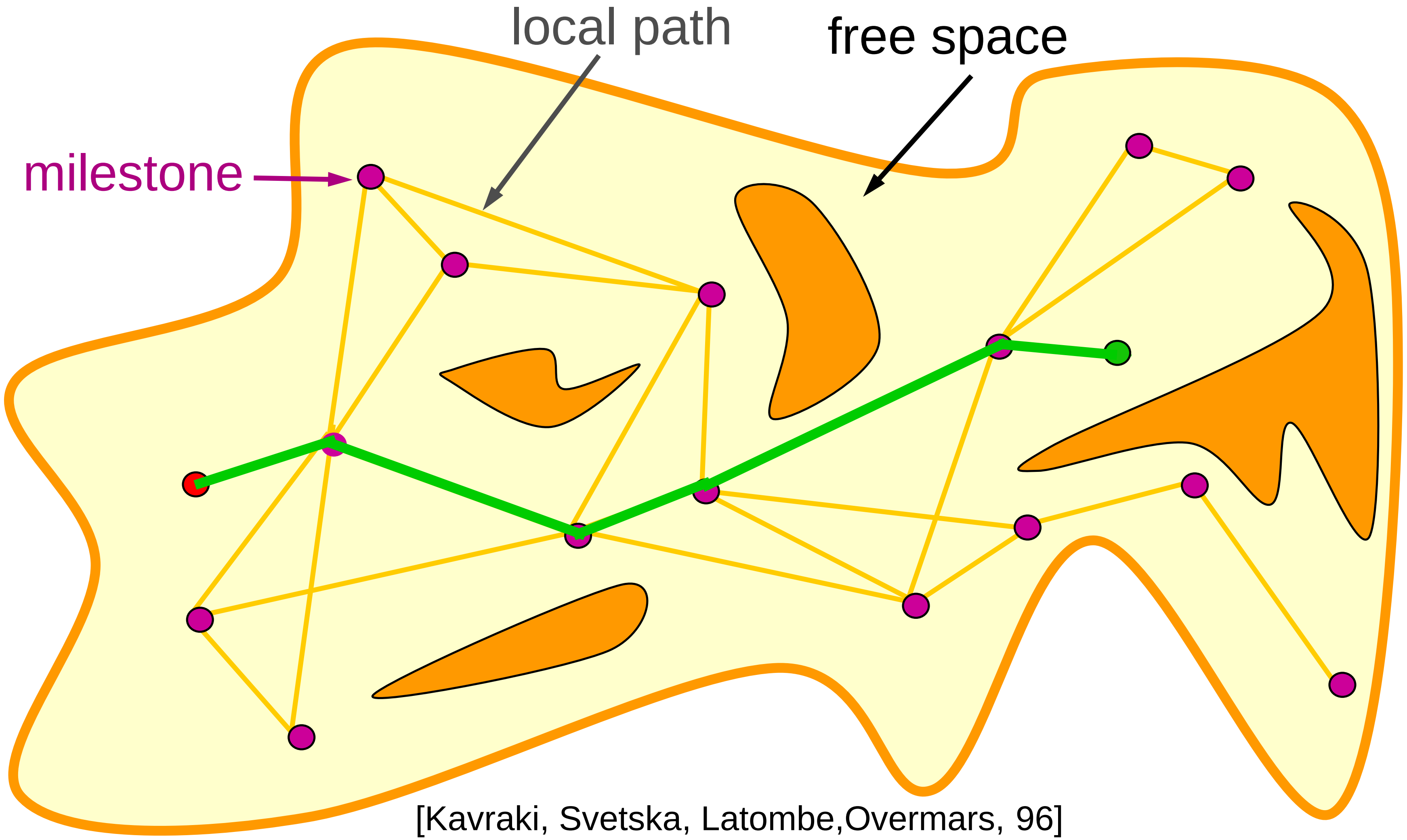
In fact, we already know how to do this:

Sampling-Based Planning!

When we first saw PRM, it was for planning paths of simple mobile robots moving in the plane.

With the concept of C-Space, We can easily generalize the method.

The only changes needed are in the local path planner and collision-checking routines.



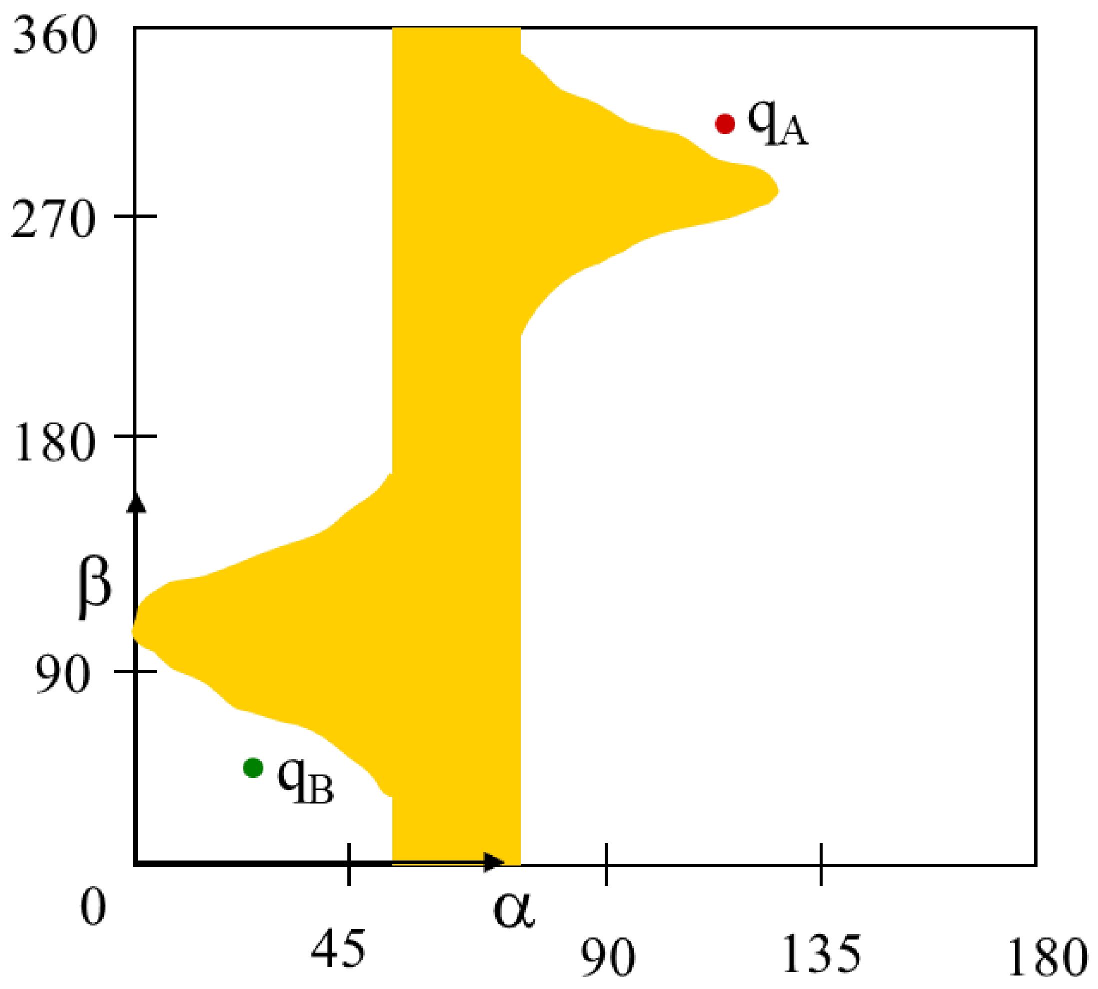
[Kavraki, Svetska, Latombe,Overmars, 96]

Why is Path Planning Difficult?

- The hard part for path planning is explicitly constructing a representation of the configuration space obstacle region (or the free configuration space).
- For the example here, we used a grid, and merely evaluated each grid point to see if it was collision free.
- The works for simple 2D cases, but if we discretize each axis into N intervals, the number of grid cells becomes N^d for a d-dimensional configuration space:
  - ➢ ***This approach does not scale!***

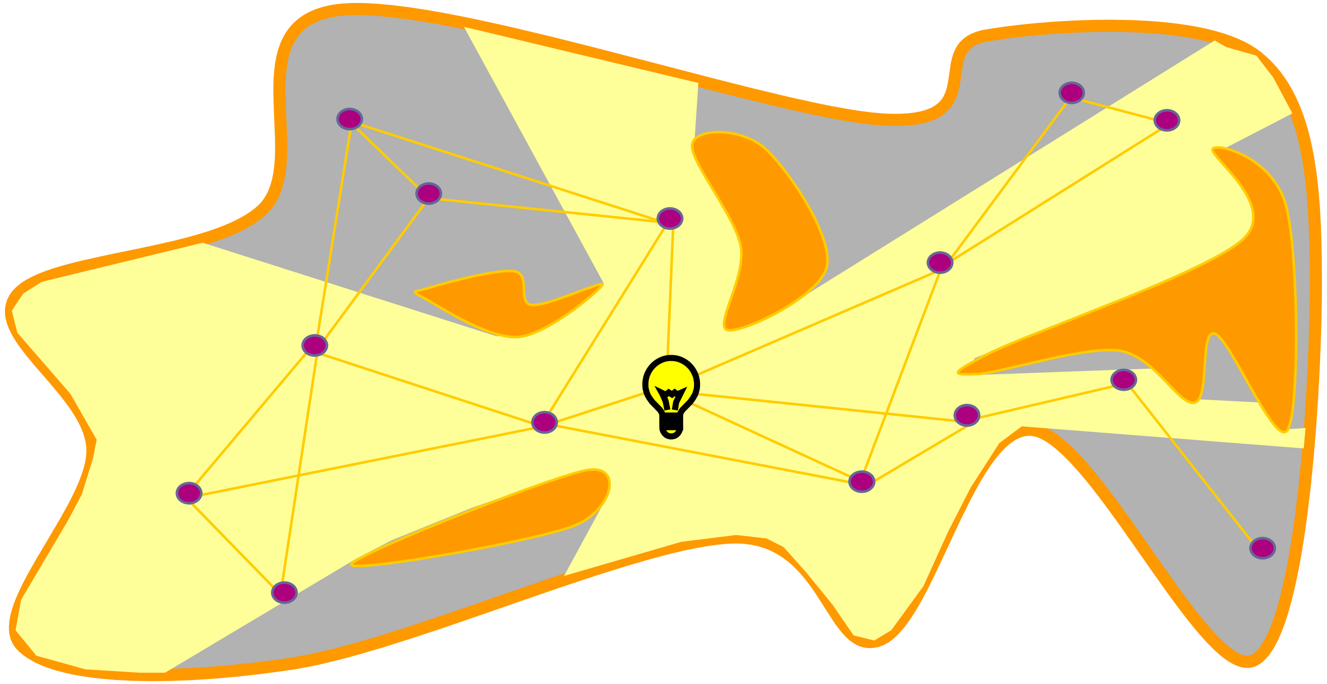
➢ With sampling-based planning, we need to answer the question:
  - ➢ *Does the straight-line path between two samples cause a collision?*

  This is not such a difficult query – fast collision checking algorithms exist.



Why does it work? Intuition

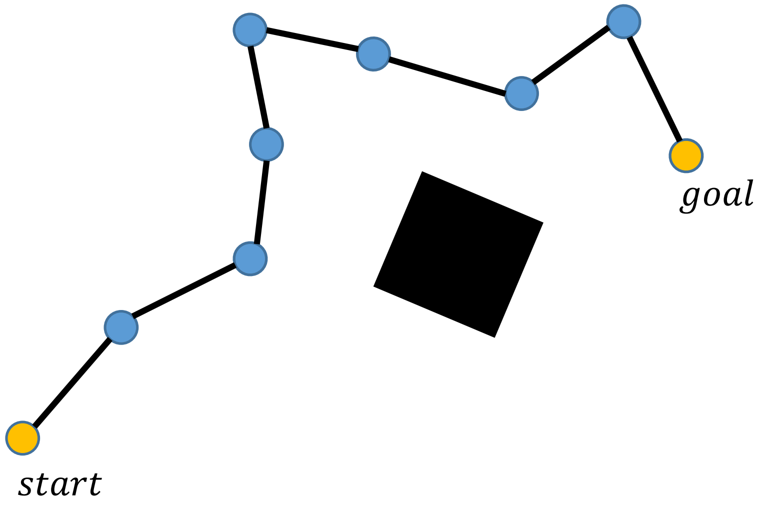
- A small number of milestones **almost** "cover" the **entire** configuration space.



- Rigorous definitions and exist (of course!)

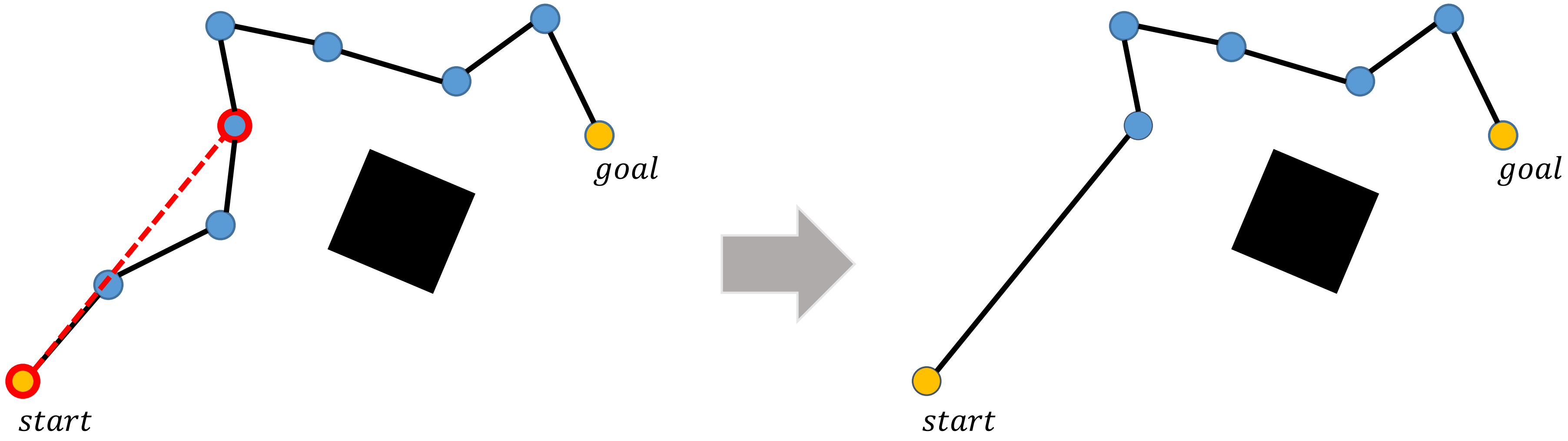
Optimizing the path

- Milestone-based paths are far from optimal and require additional refinement before they are usable
- A typical solution can look like this:



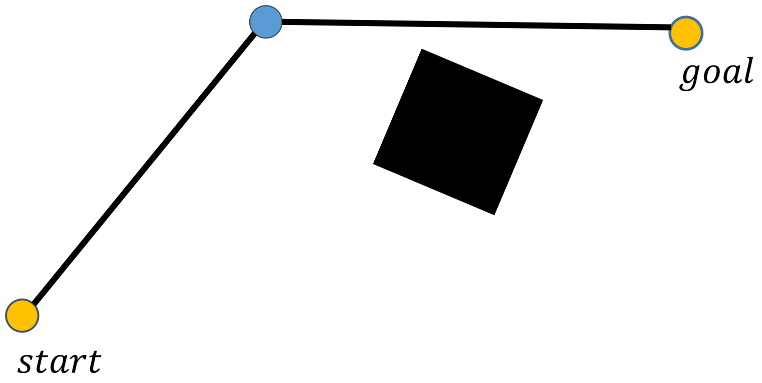
Optimizing the path

- A simple way to improve the path, is to repeatedly pick two nodes at random, and check whether they can be connected by a straight line without collision. If so, use the line to shorten the path.



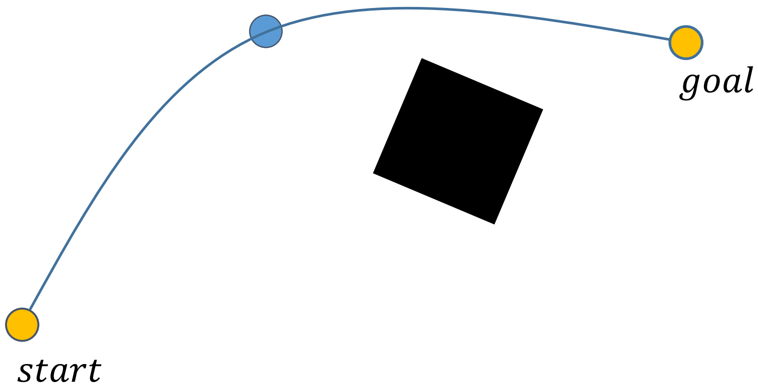
Optimizing the path

- Repeat for N iterations, or until no further improvements are being made
- The result is not an optimal path, but shorter and more efficient than the original

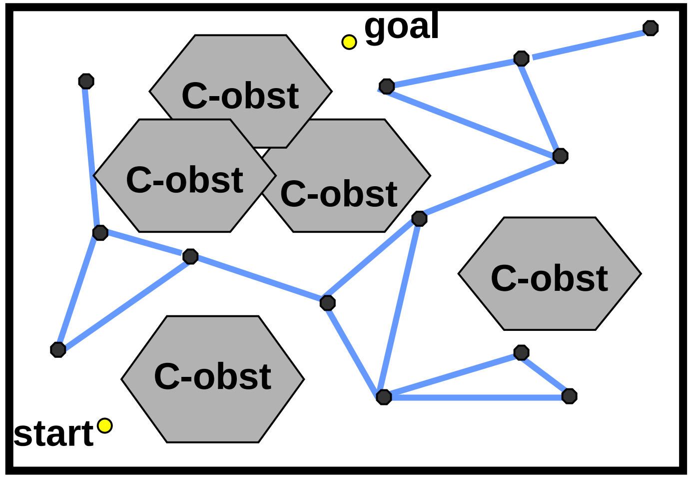


Smoothing the path

- Optionally, the shortened path can then be smoothed to allow for continuous robot motion



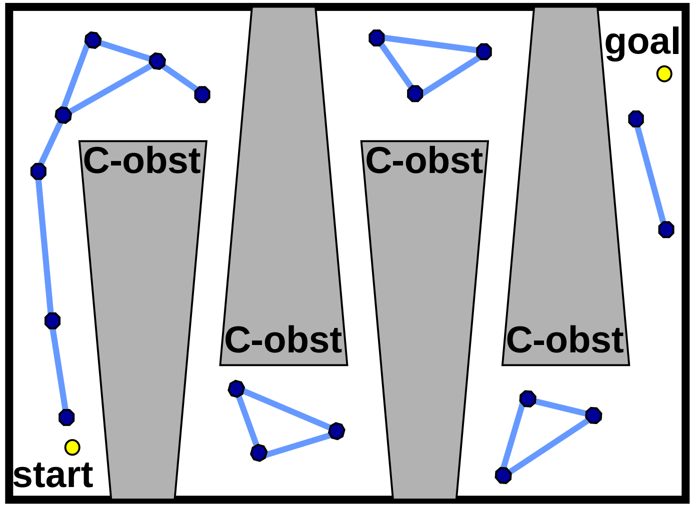
Good news, but bad news too



Sample-based: The Good News

1. *probabilistically complete*
2. Do not construct the C-space
3. apply easily to high-dimensional C-space
4. support fast queries w/ enough preprocessing

Many success stories where PRMs solve previously unsolved problems



Sample-Based: The Bad News

1. don't work as well for some problems:
– unlikely to sample nodes in *narrow passages*
– hard to sample/connect nodes on constraint surfaces
2. No optimality or completeness

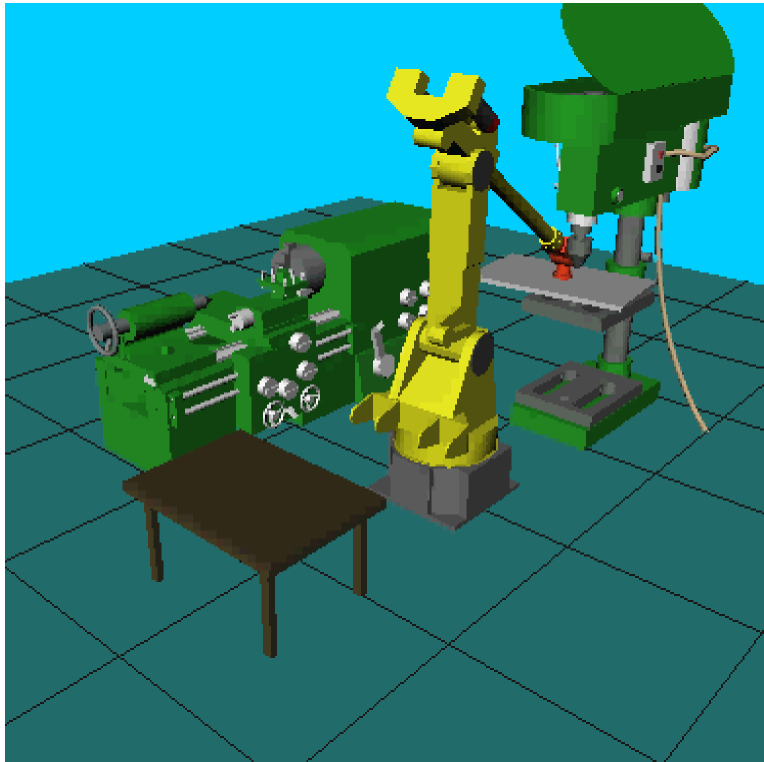
PRM variants

- There are (very) many…

- Lazy PRM:
  - Create a dense PRM without ANY collision checking
  - When you have q_{init} and q_{goal}:
    - Find $q_{init} \rightarrow s_1 \rightarrow s_2 \rightarrow q_{goal}$
    - Check only the edges in the returned path for collisions, remove any edges with collisions.

Assumptions

- Static obstacles
- Many queries to be processed in the same environment
- Examples
  - Navigation in static virtual environments
  - Robot manipulator arm in a workcell
- Advantages:
  - Amortize the cost of planning over many problems
  - Probabilistically complete



General Types of approaches that use sampling

Sampling-based methods typically fall into two categories:

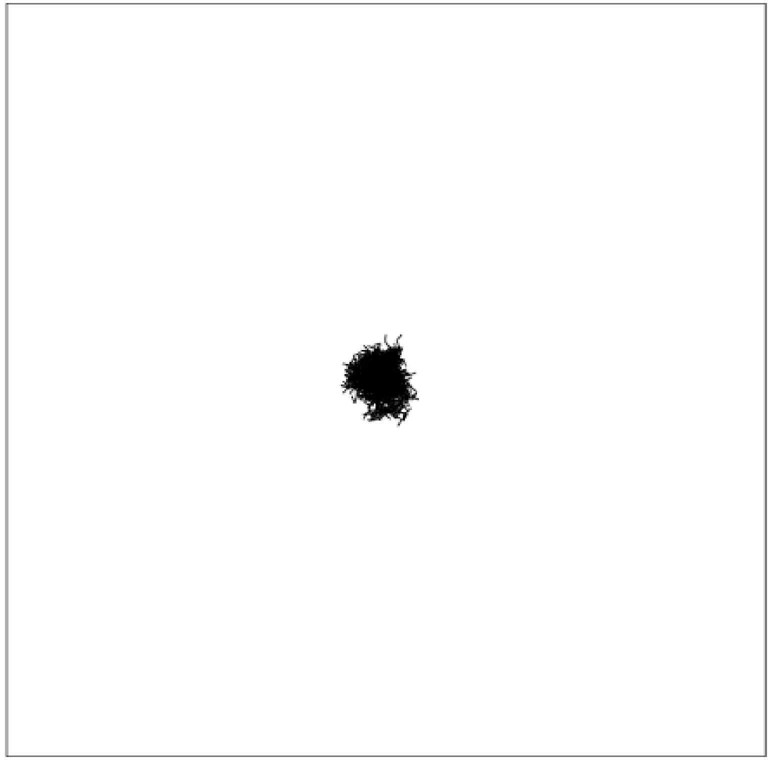
	Multi-query	**Single-query**
Phases	1. Roadmap construction 2. Searching	Roadmap construction and searching online
Typical algorithm	Probabilistic Roadmap (PRM)	Rapidly Exploring Random Tree (RRT)
Pros	Fast searching	No preprocessing
Cons	Inability to deal with environment changes	No memory

Rapidly-Exploring Random Tree (RRT)

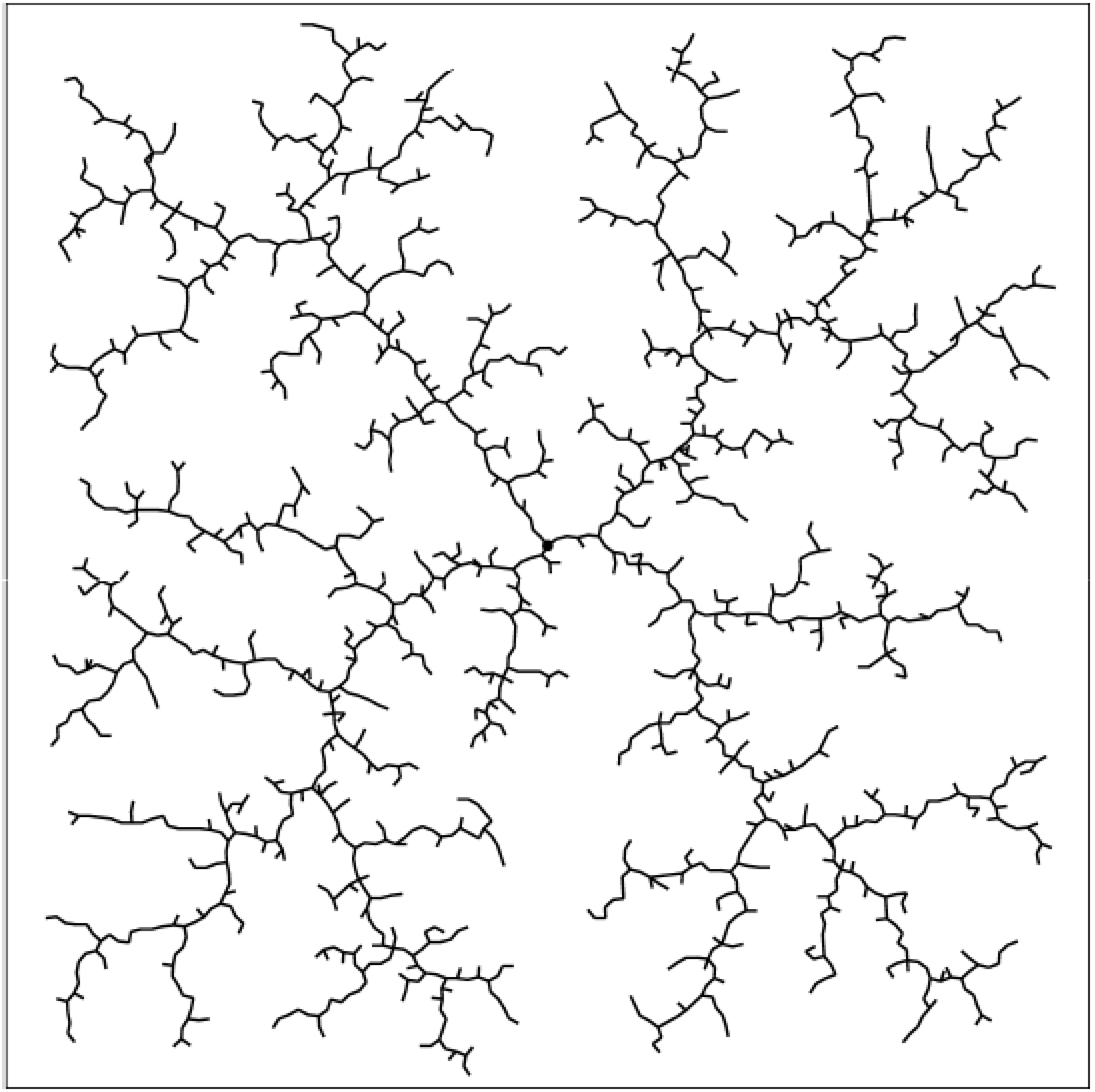
- Searches for a path from the initial configuration to the goal configuration by expanding a search tree
- For each step,
  - The algorithm samples a target configuration and expands the tree towards it.
  - The sample can either be a random configuration or the goal configuration itself, depends on the probability value defined by the user.

Naïve random tree

- Pick a vertex at random
- Move in a random direction to generate a new vertex
- Repeat…



Rapidly-Exploring Random Tree



The Basic Idea: Iteratively expand the tree

- Denote by T_k the tree at iteration k

- Randomly choose a configuration q_{rand}

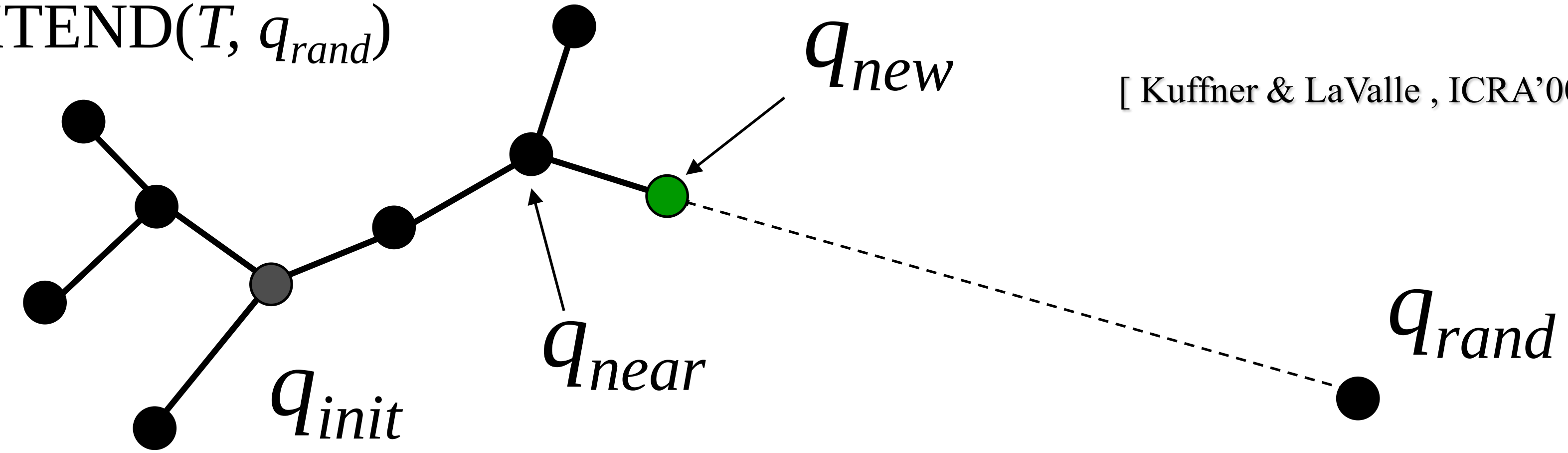
- Choose $q_{near} = \arg\min_{q \in T_k} d(q, q_{rand})$
  - ➢ q_{near} is the nearest existing node in the tree to q_{rand}

- Create a new node, q_{new} by taking a small step from q_{near} toward q_{rand}

Path Planning with RRTs

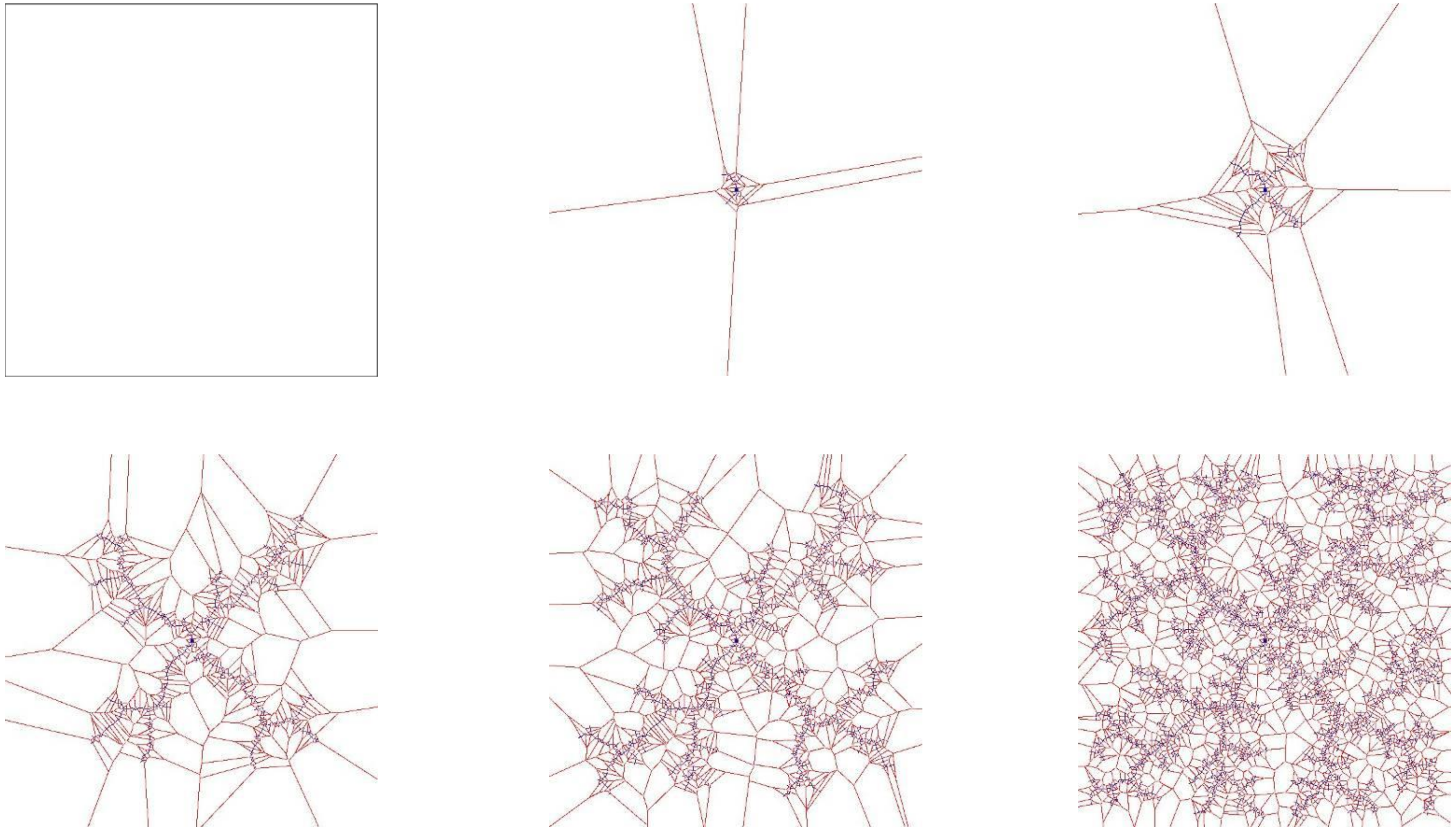
BUILD_RRT (q_{init}) {
  $T.init(q_{init})$;
  for k = 1 to K do
    q_{rand} = RANDOM_CONFIG();
    EXTEND(T, q_{rand})
}

EXTEND(T, q_{rand})



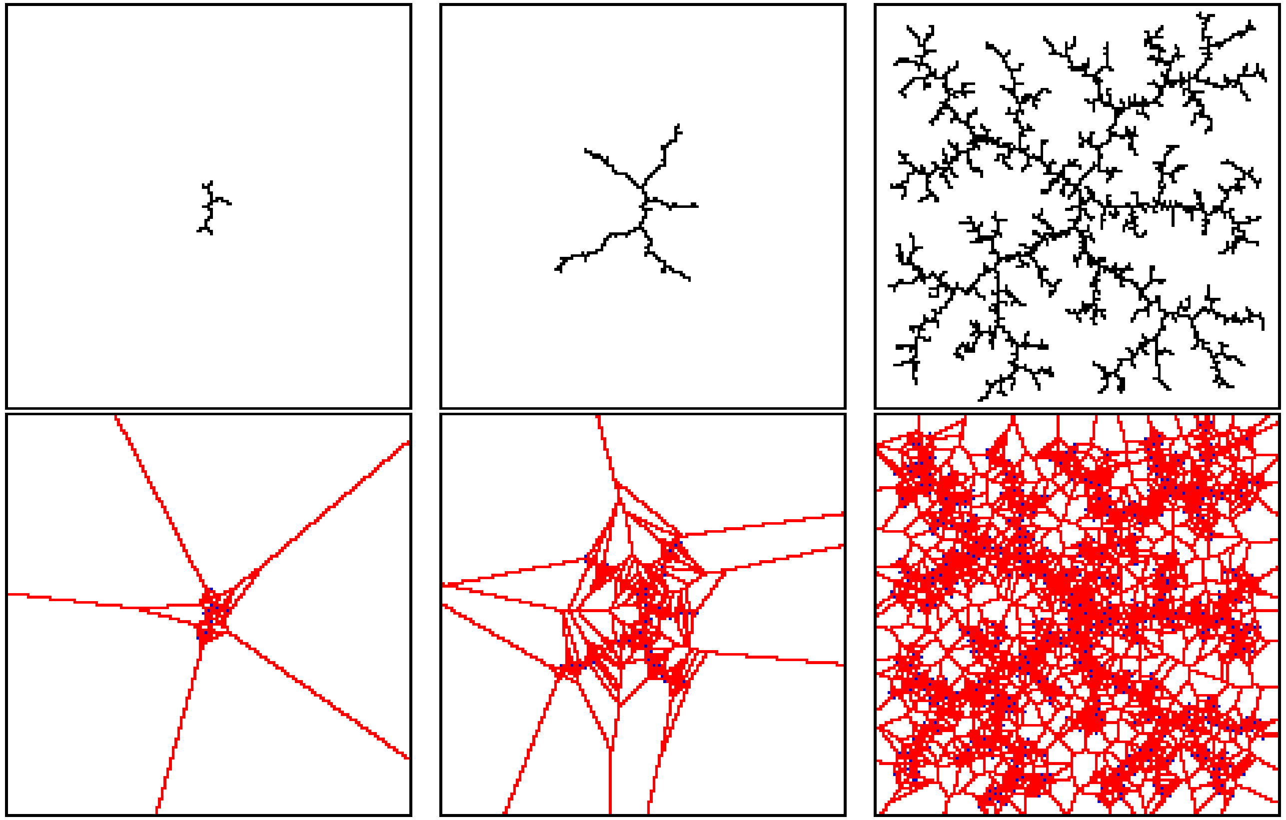
[Kuffner & LaValle , ICRA'00]

RRTs and
Bias toward large Voronoi regions



http://msl.cs.uiuc.edu/rrt/gallery.html

Why are RRT's rapidly exploring?



The probability of a node being selected for expansion (i.e. being a nearest neighbor to a new randomly picked point) is proportional to the area of its Voronoi region.

Biases

- Bias toward larger spaces
- Bias toward goal
  - When generating a random sample, with some probability pick the goal instead of a random node when expanding
  - This introduces another parameter
  - James' experience is that 5-10% is the right choice
  - If you do this 100%, then this is a RPP

RRT in Action…

RRT

Requires the following functions:

RRT

Requires the following functions:

p = RandomSample()

Uniform random sampling of free configuration space

RRT

Requires the following functions:

p = RandomSample()

Uniform random sampling of free configuration space

v = Nearest(p)

Given point in Cspace, find vertex on tree that is closest to that point

RRT

Requires the following functions:

p = RandomSample()

Uniform random sampling of free configuration space

v = Nearest(p)

Given point in Cspace, find vertex on tree that is closest to that point

p’ = Steer(p, goal)

For a point p and a goal point, find p’ that is closer to the goal than p

RRT

Requires the following functions:

p = RandomSample()

Uniform random sampling of free configuration space

v = Nearest(p)

Given point in Cspace, find vertex on tree that is closest to that point

p’ = Steer(p, goal)

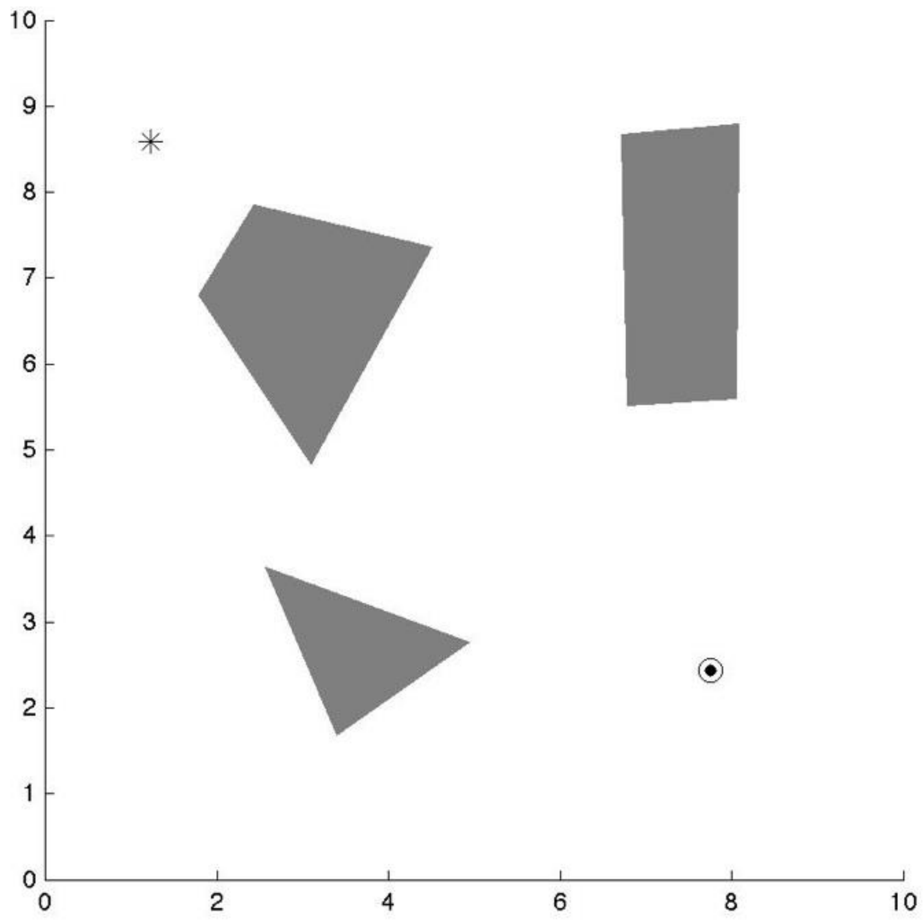
For a point p and a goal point, find p’ that is closer to the goal than p

ObstacleFree(p)

Check if a given Cspace point is in the free space

RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$
for $i = 1$ to N
  $G \leftarrow (V, E)$
  $x_{rand} \leftarrow RandomSample()$
  $x_{nearest} \leftarrow Nearest(G, x_{rand})$
  $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
  if $ObstacleFree(x_{nearest}, x_{new})$
    $V \leftarrow V \cup \{x_{new}\}$
    $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

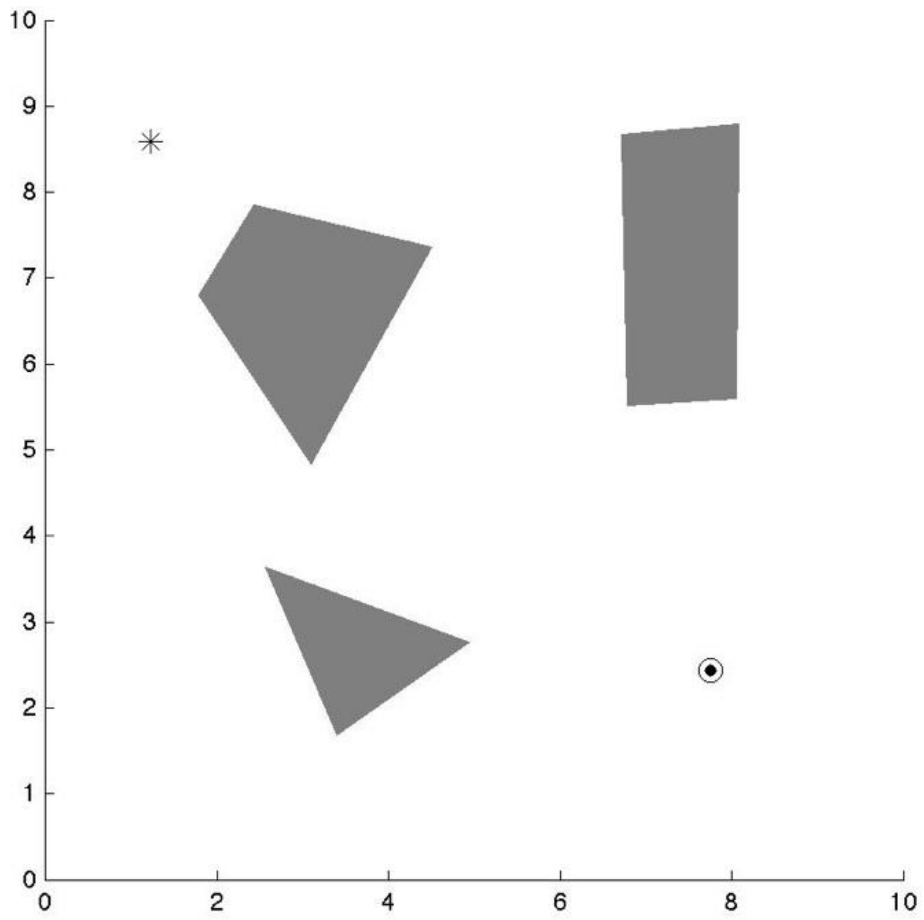
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

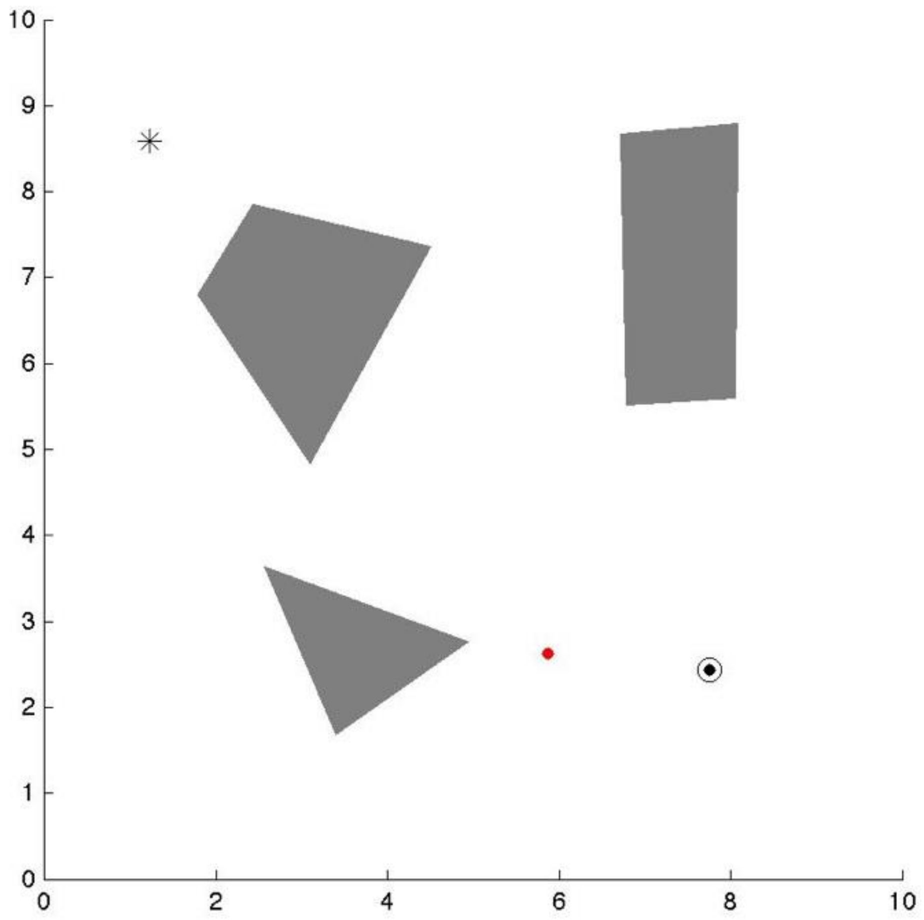
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

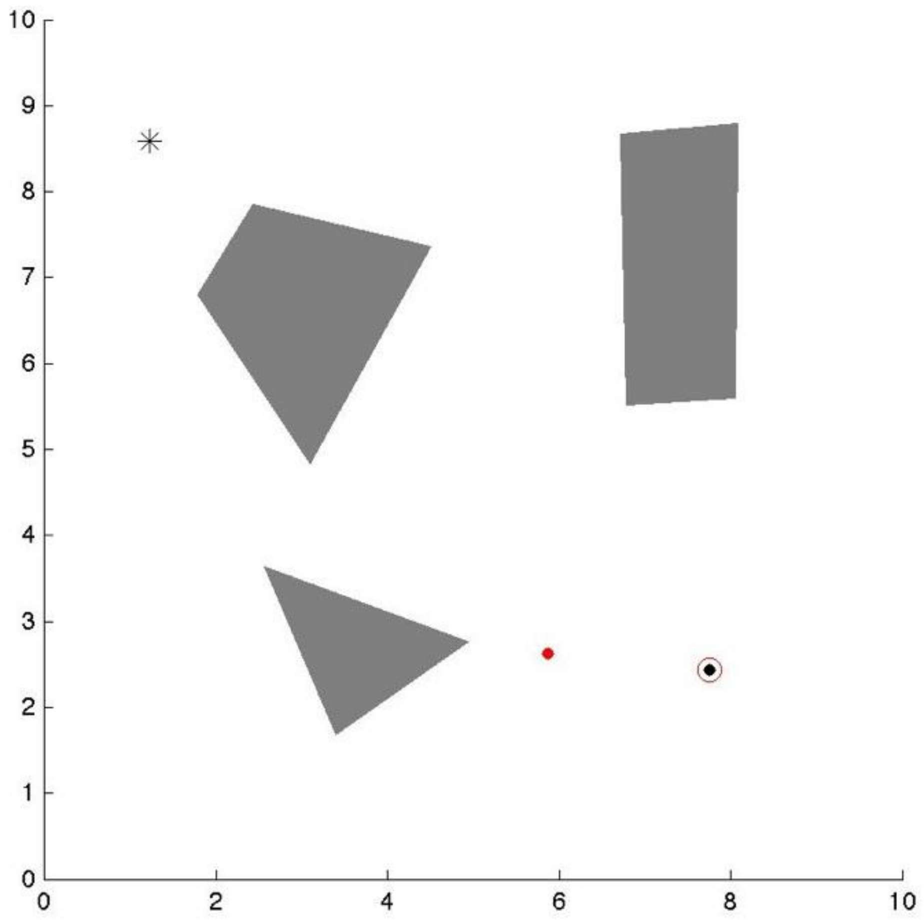
$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$
for $i = 1$ to N
  $G \leftarrow (V, E)$
  $x_{rand} \leftarrow RandomSample()$
  $x_{nearest} \leftarrow Nearest(G, x_{rand})$
  $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
  if $ObstacleFree(x_{nearest}, x_{new})$
    $V \leftarrow V \cup \{x_{new}\}$
    $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

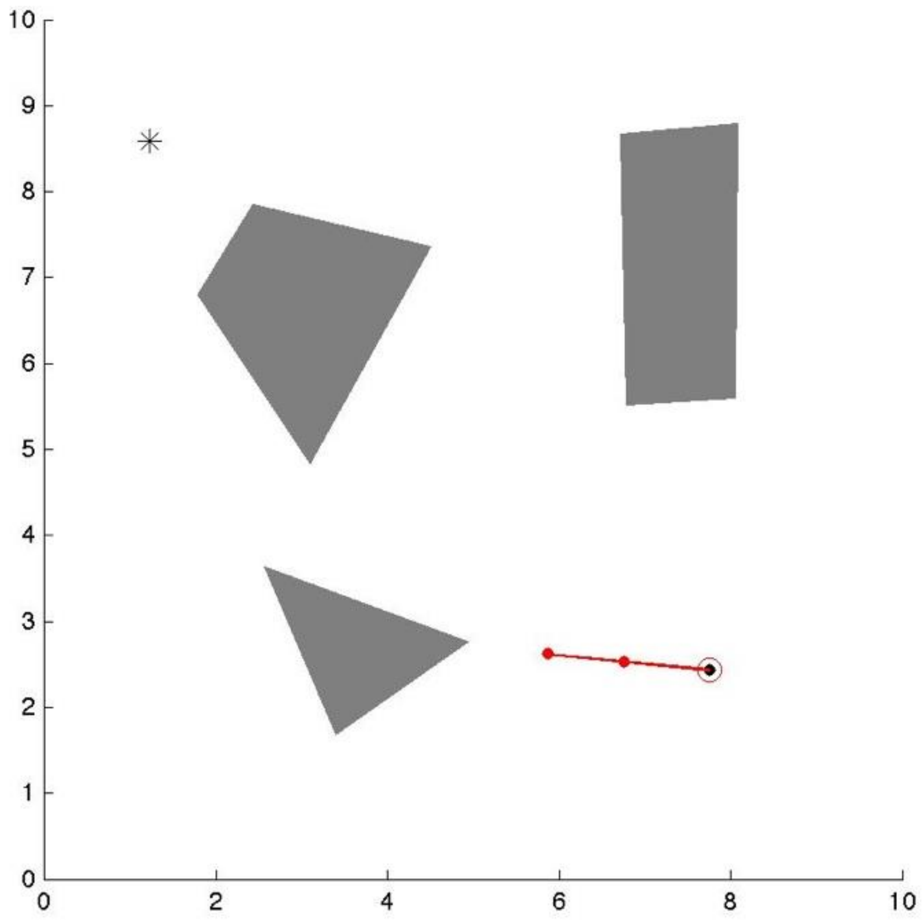
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

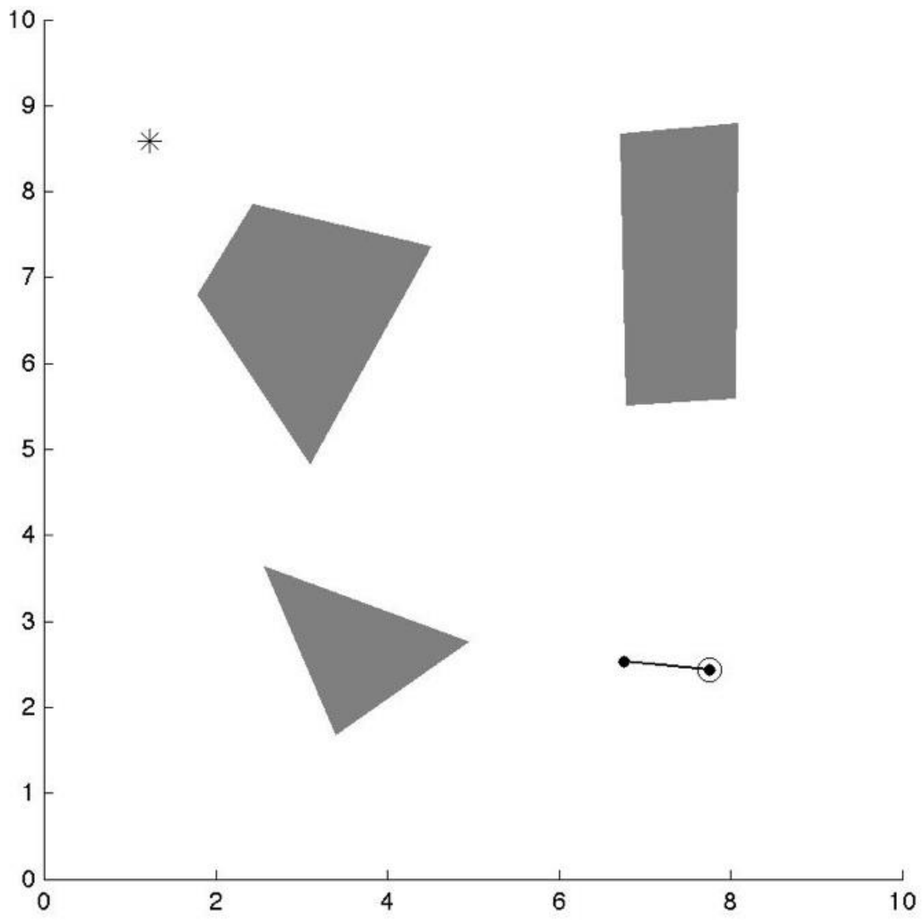
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

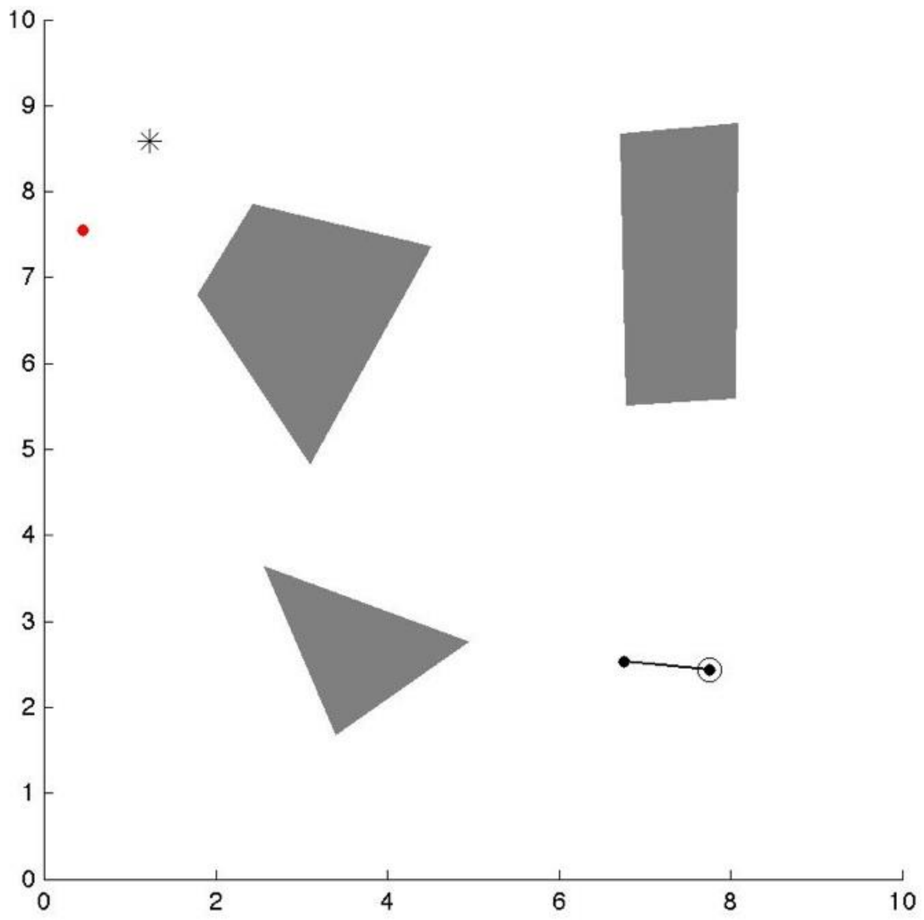
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

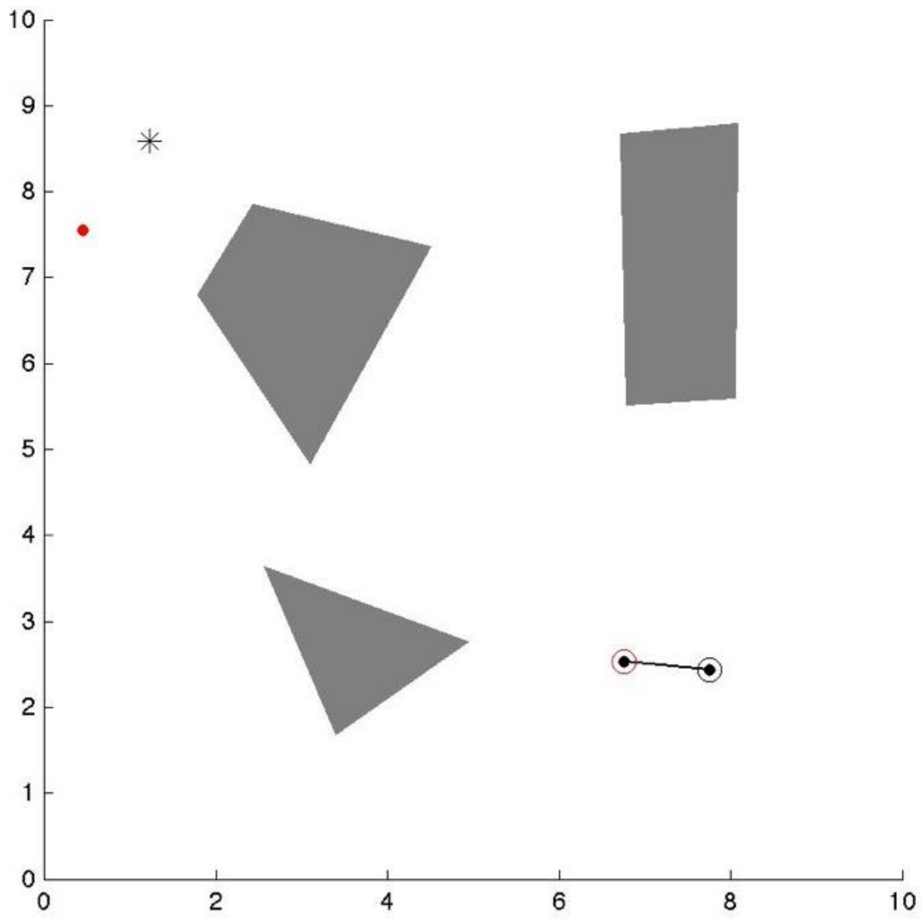
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

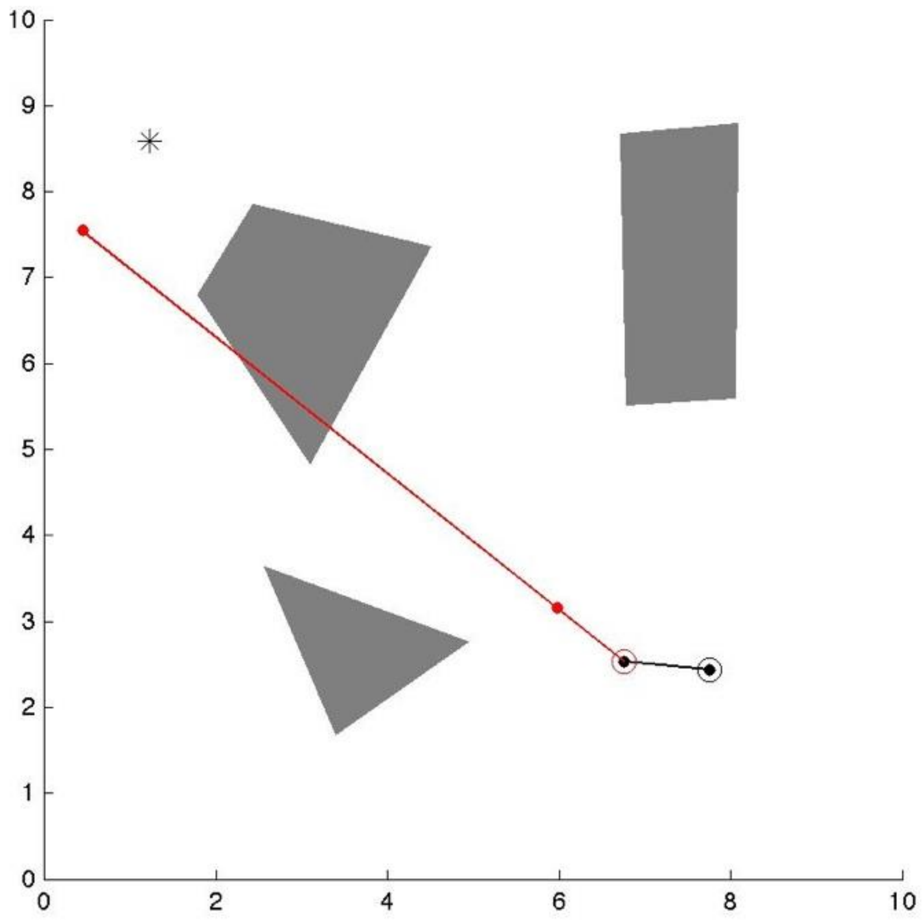
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad \boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

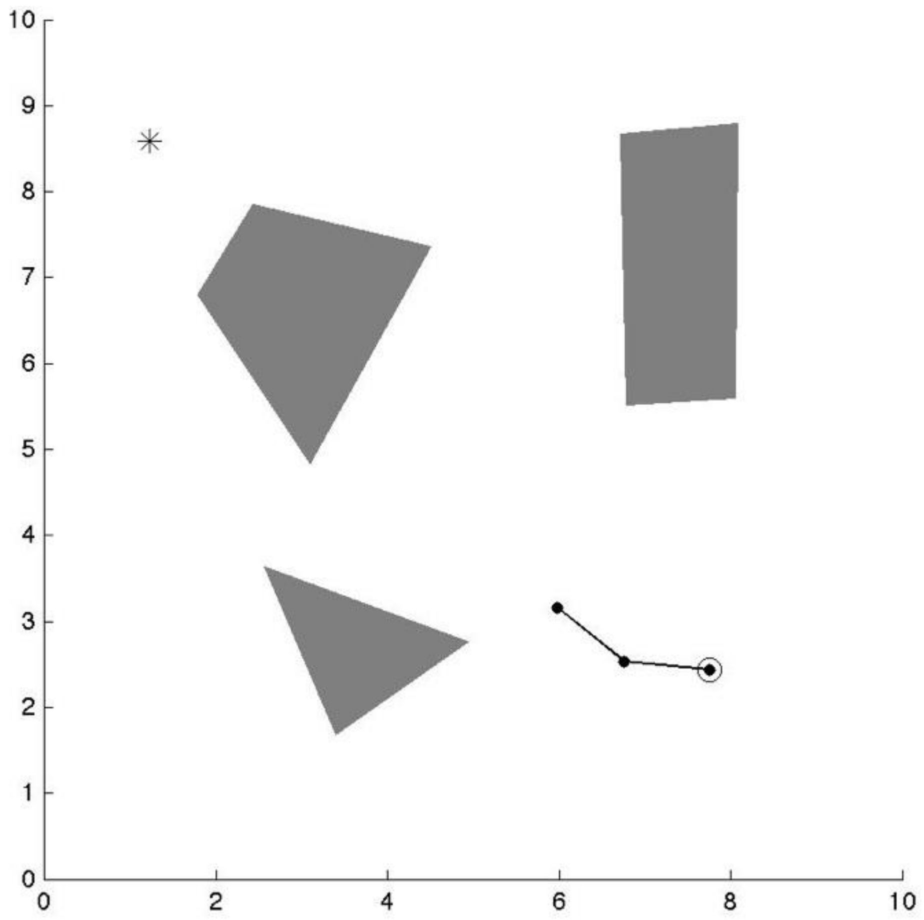
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

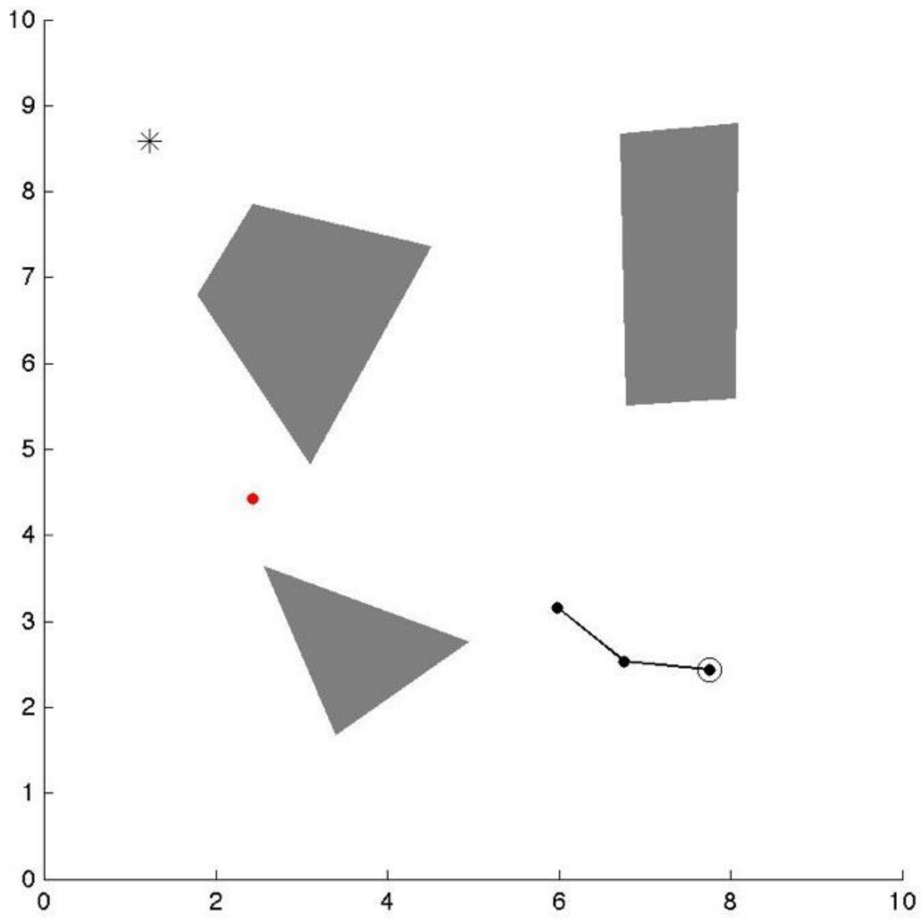
$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$
for $i = 1$ to N
  $G \leftarrow (V, E)$
  $x_{rand} \leftarrow RandomSample()$
  $x_{nearest} \leftarrow Nearest(G, x_{rand})$
  $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
  if $ObstacleFree(x_{nearest}, x_{new})$
    $V \leftarrow V \cup \{x_{new}\}$
    $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

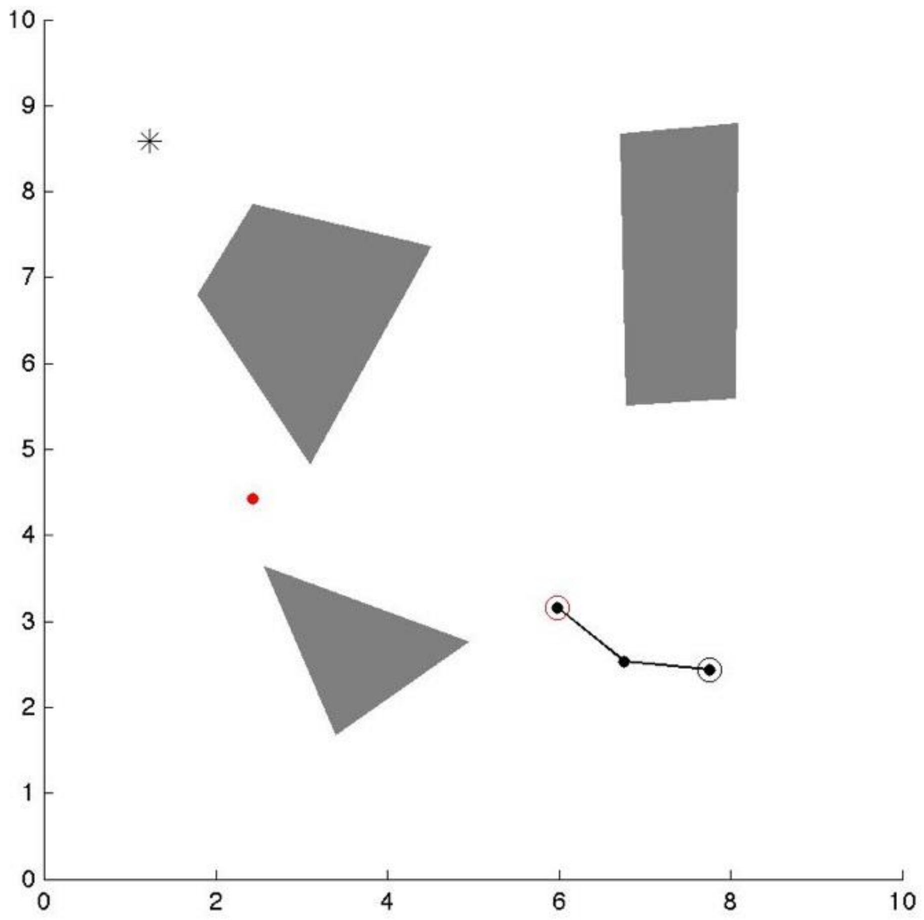
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

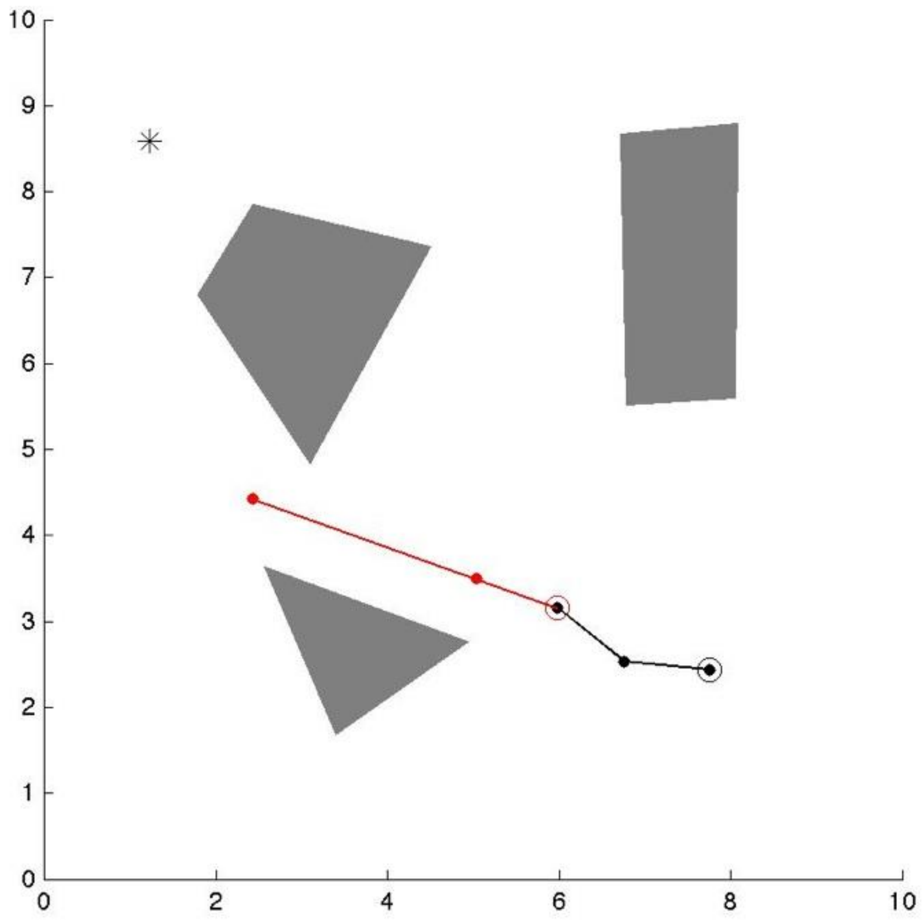
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

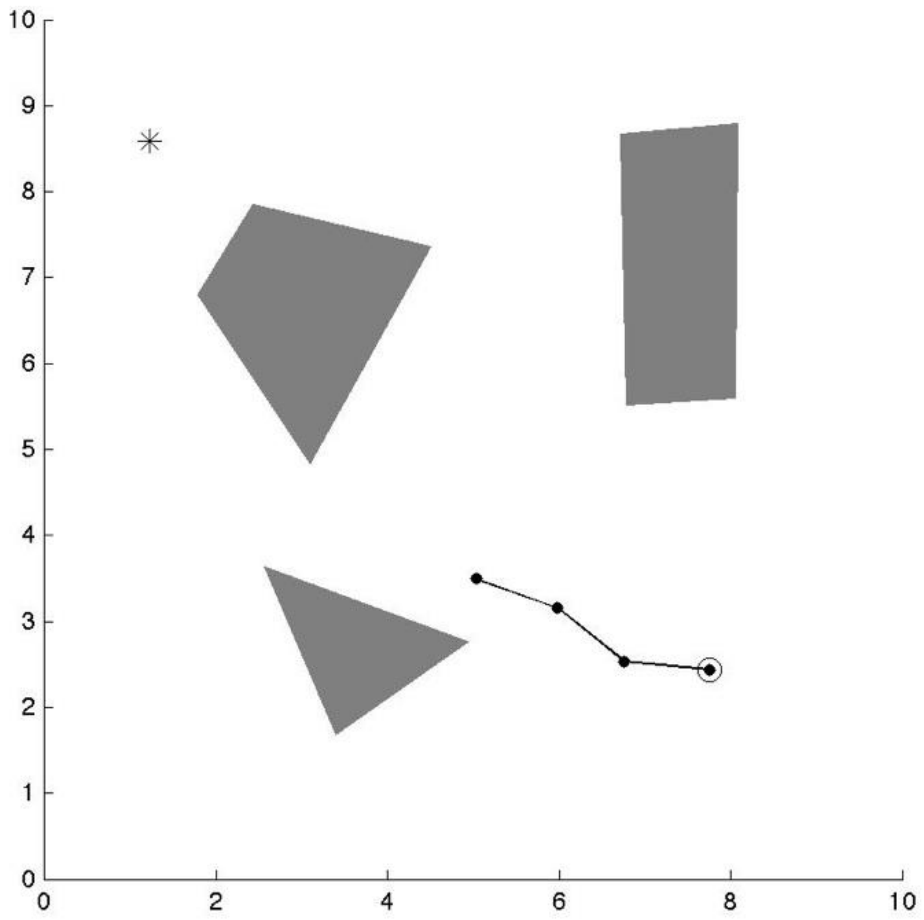
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

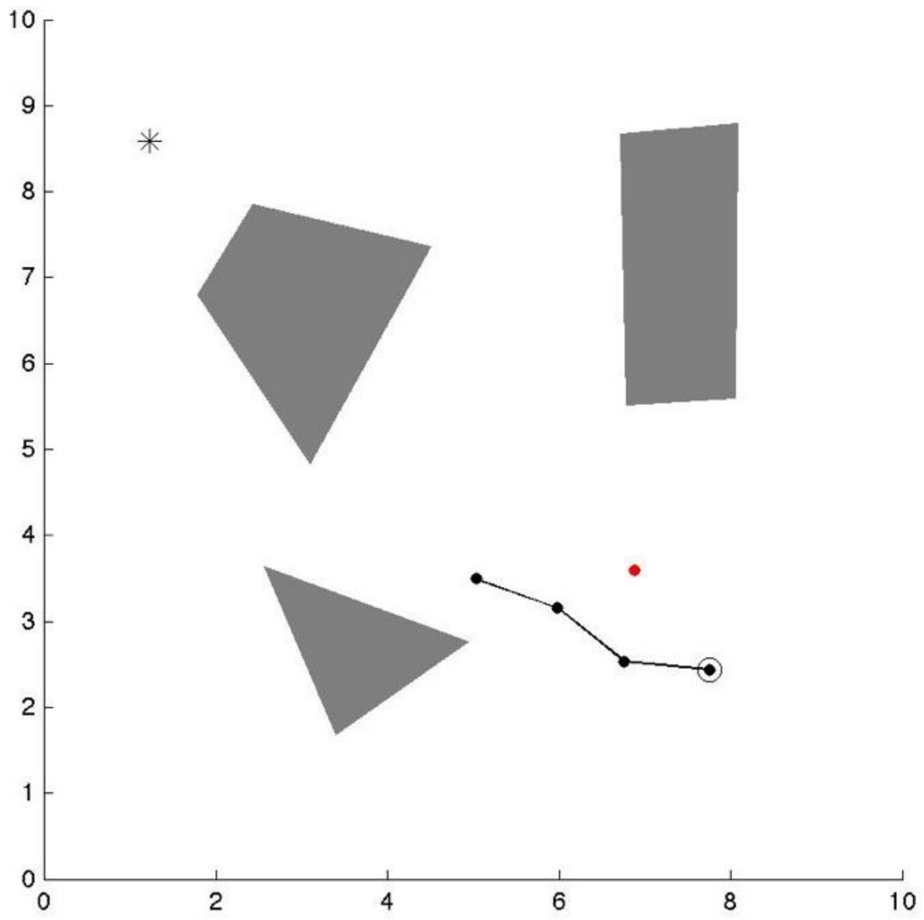
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

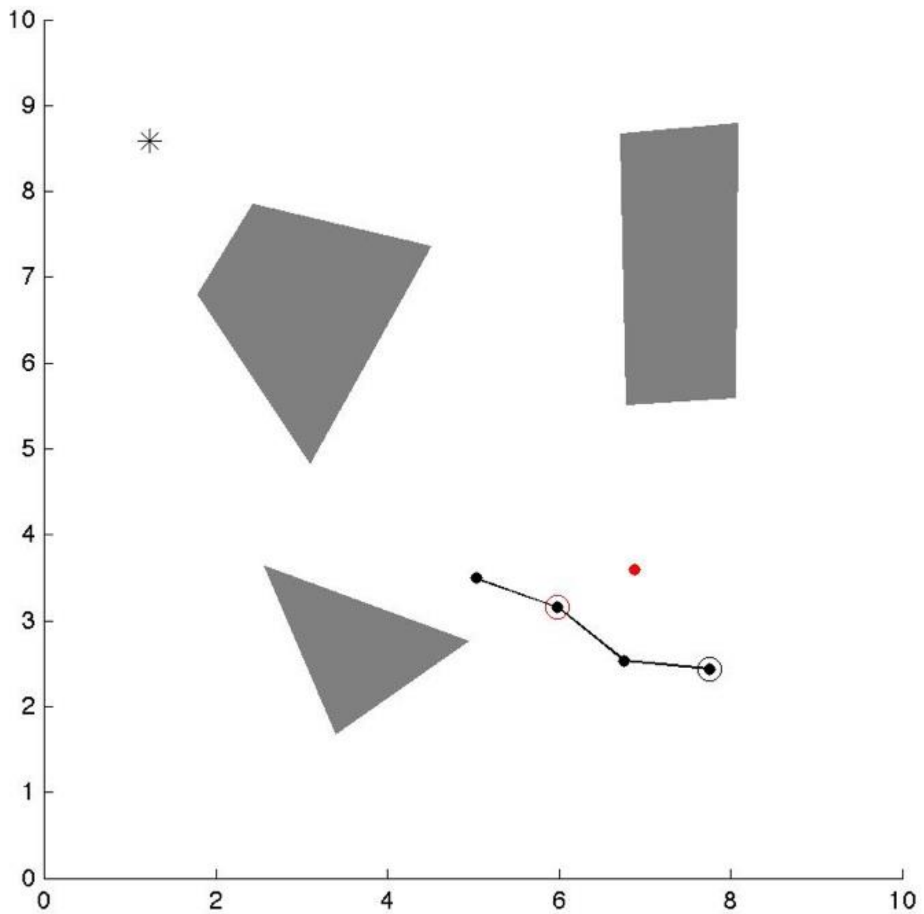
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

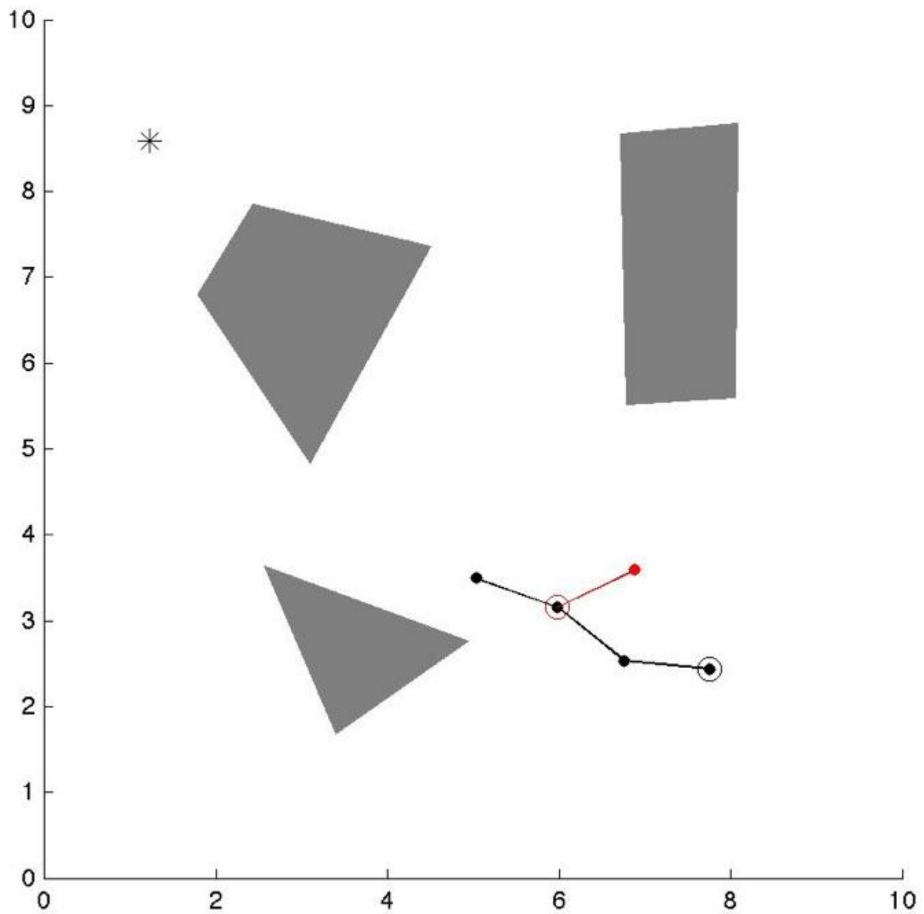
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad \boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

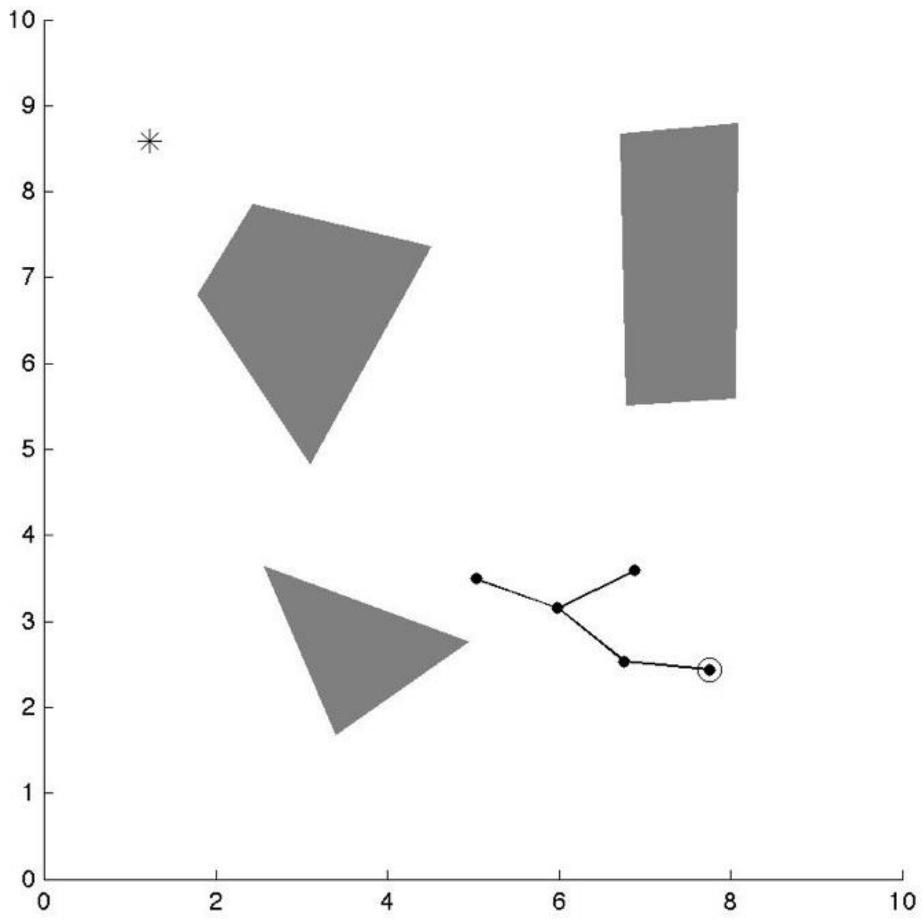
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

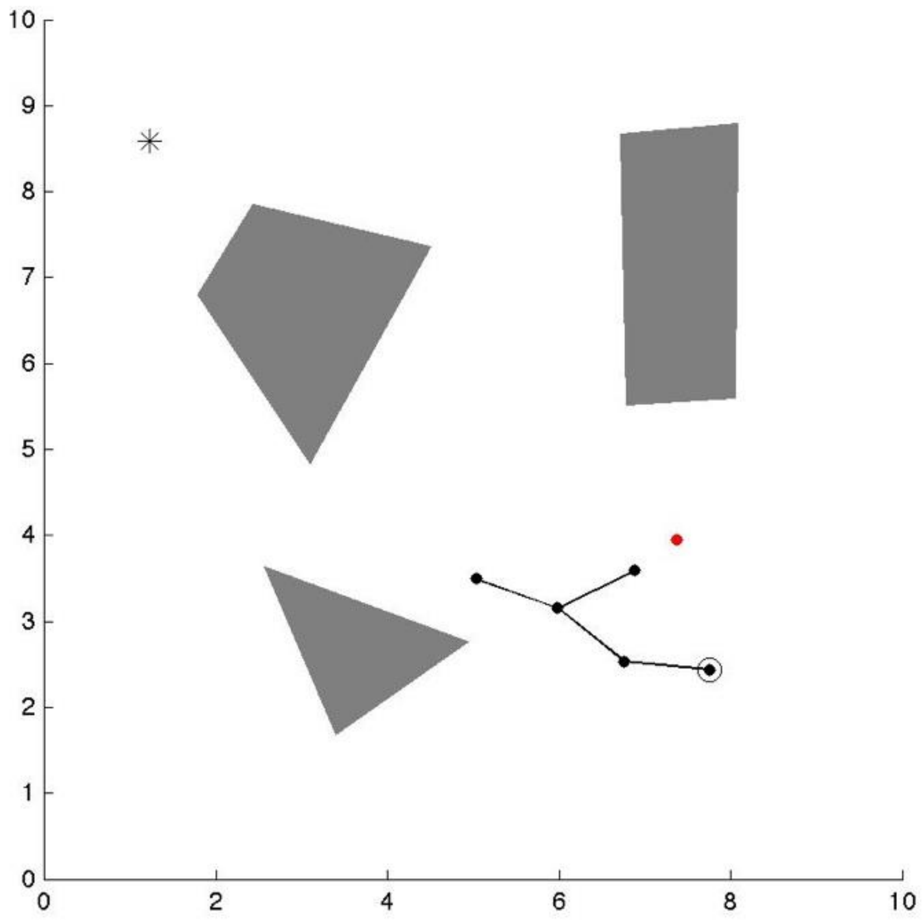
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

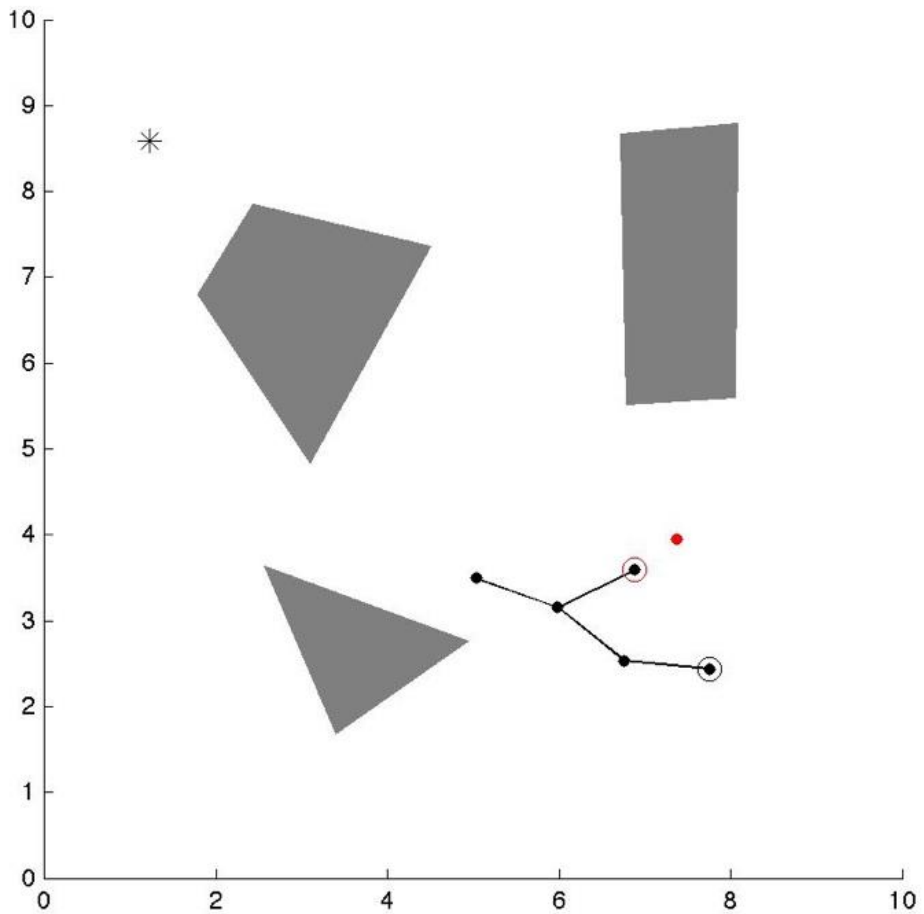
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

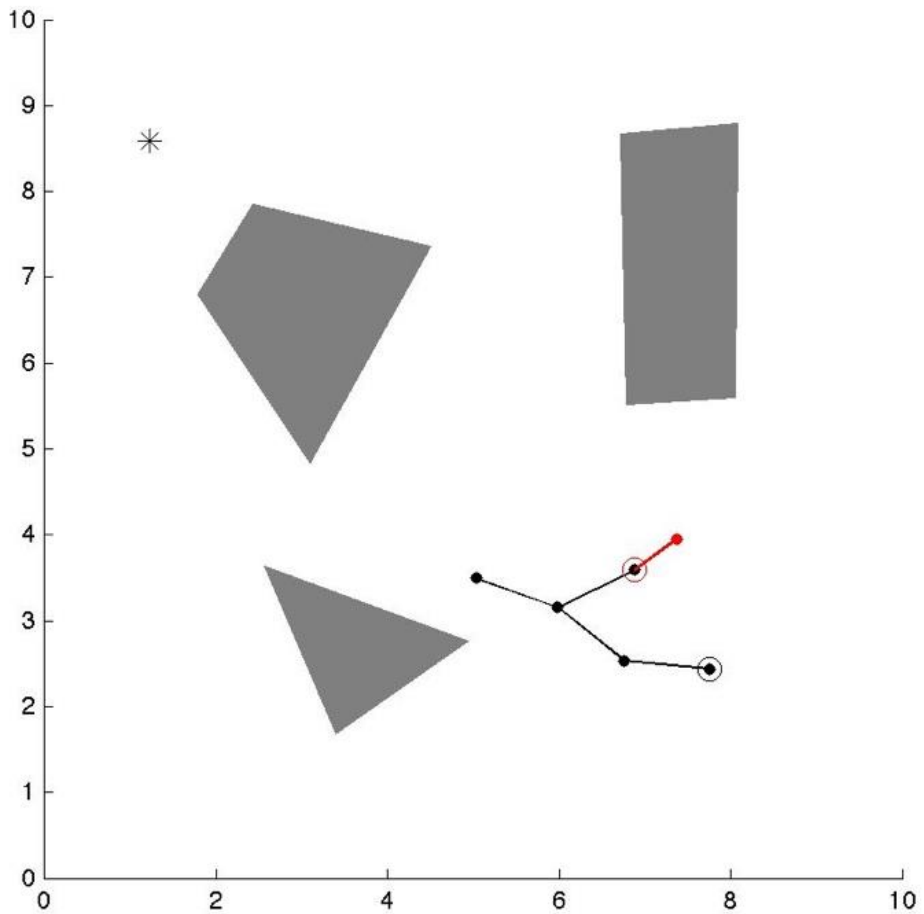
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad \boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

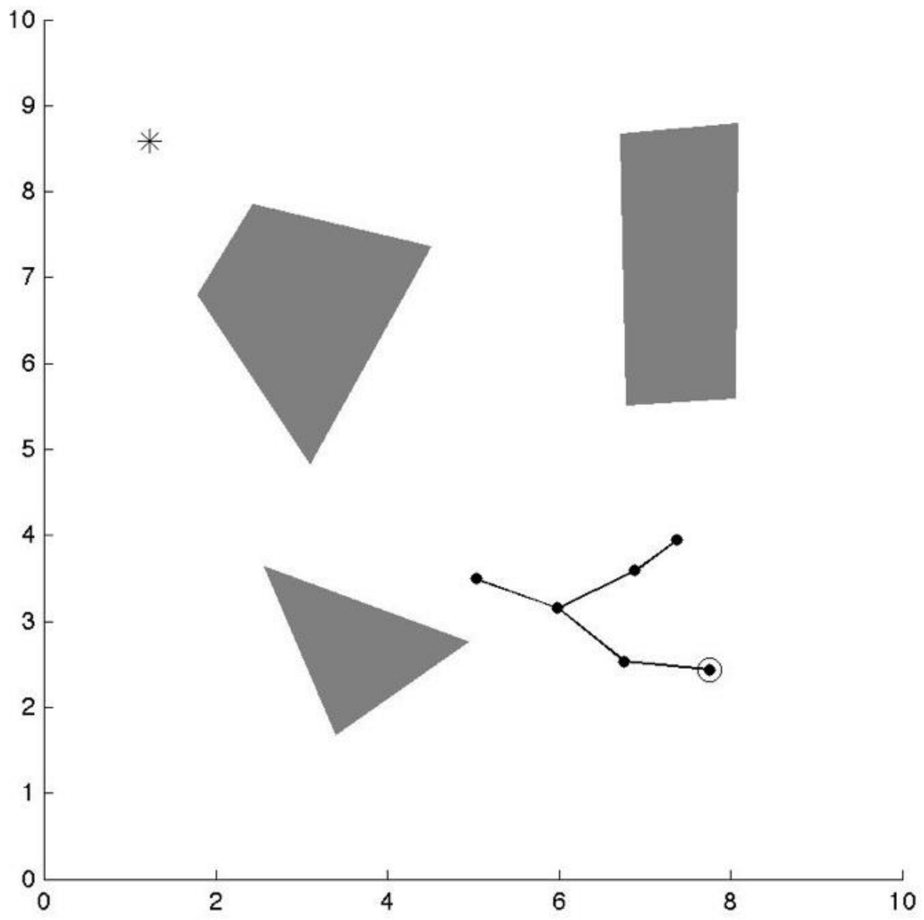
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

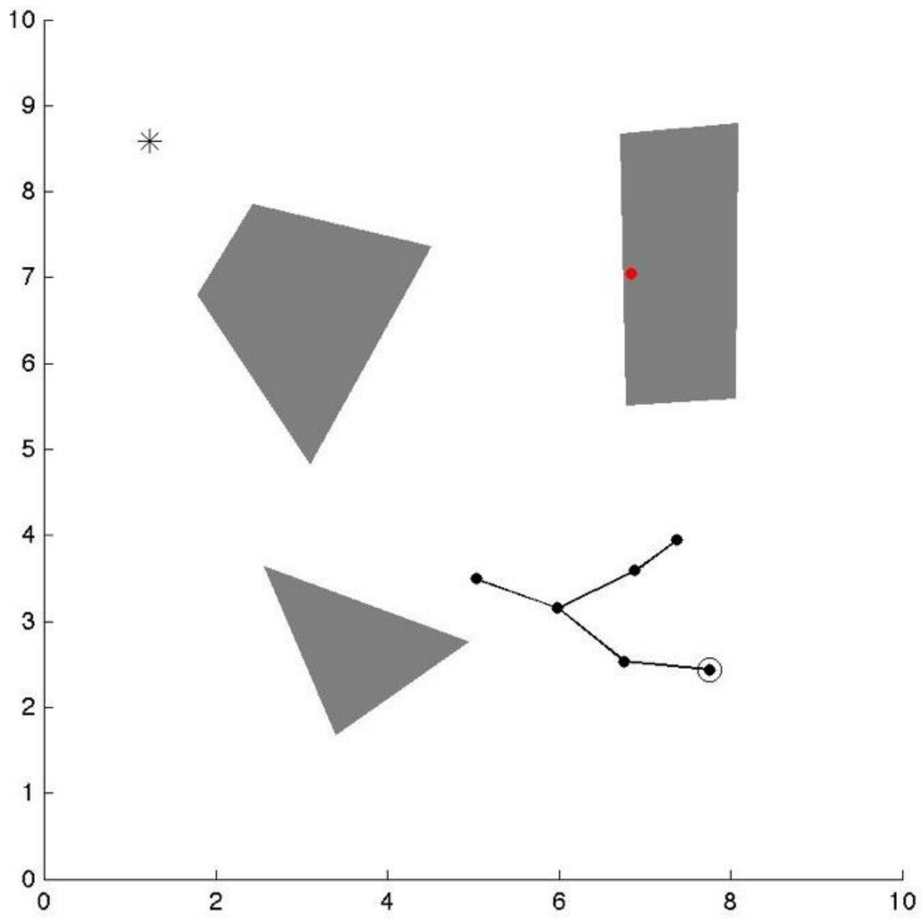
$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



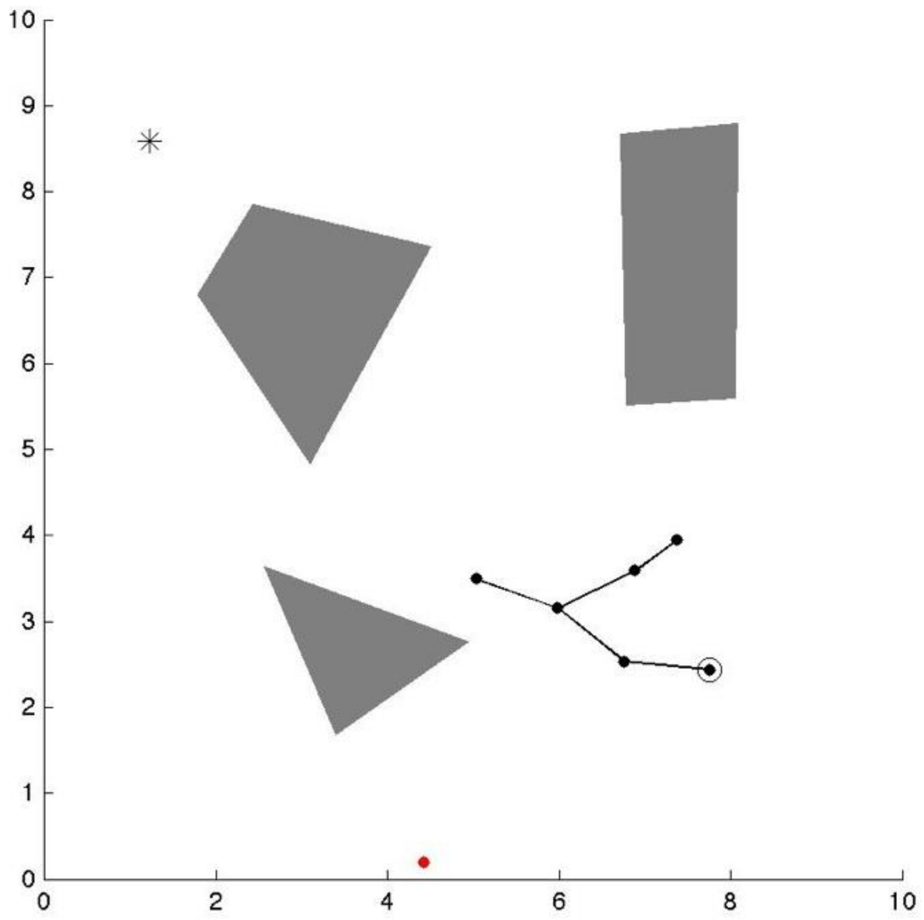
RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$
for $i = 1$ to N
  $G \leftarrow (V, E)$
  $x_{rand} \leftarrow RandomSample()$
  $x_{nearest} \leftarrow Nearest(G, x_{rand})$
  $x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$
  if $ObstacleFree(x_{nearest}, x_{new})$
    $V \leftarrow V \cup \{x_{new}\}$
    $E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



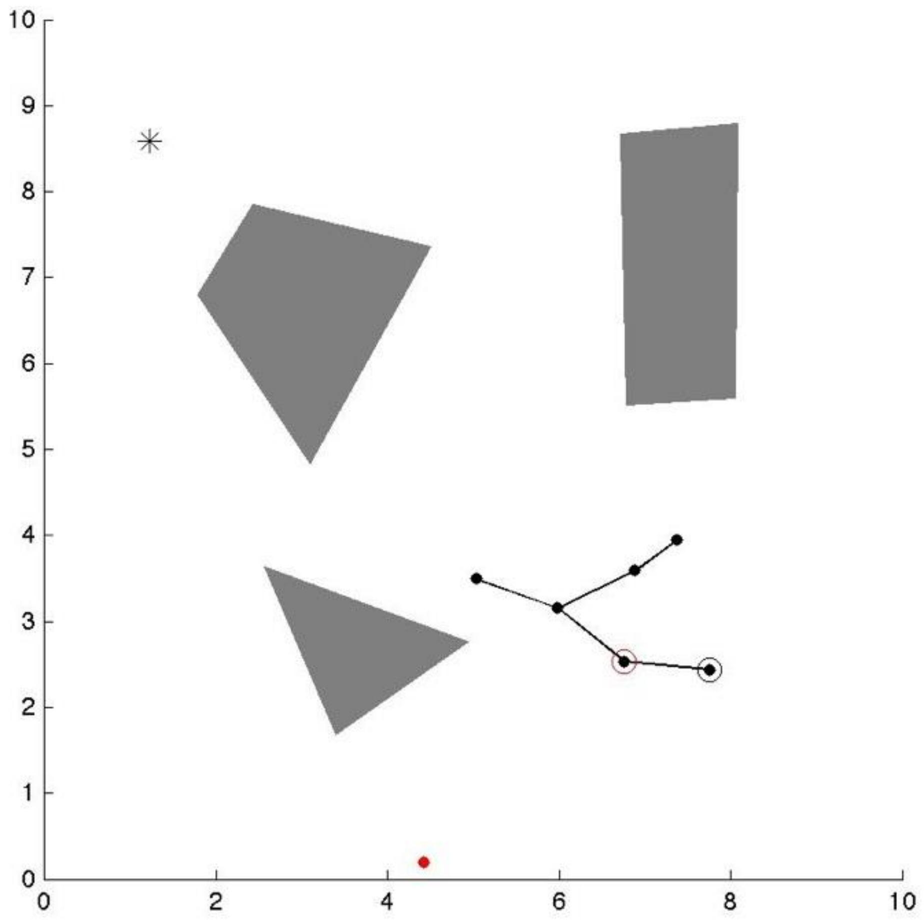
RRT

$$
\begin{array}{l}
V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset \\
\textbf{for } i = 1 \text{ to } N \\
\quad G \leftarrow (V, E) \\
\quad x_{rand} \leftarrow RandomSample() \\
\quad x_{nearest} \leftarrow Nearest(G, x_{rand}) \\
\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand}) \\
\quad \textbf{if } ObstacleFree(x_{nearest}, x_{new}) \\
\quad \quad V \leftarrow V \cup \{x_{new}\} \\
\quad \quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}
\end{array}
$$



RRT

$$
\begin{aligned}
&V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset \\
&\textbf{for } i = 1 \text{ to } N \\
&\quad G \leftarrow (V, E) \\
&\quad x_{rand} \leftarrow RandomSample() \\
&\quad x_{nearest} \leftarrow Nearest(G, x_{rand}) \\
&\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand}) \\
&\quad \textbf{if } ObstacleFree(x_{nearest}, x_{new}) \\
&\qquad V \leftarrow V \cup \{x_{new}\} \\
&\qquad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}
\end{aligned}
$$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

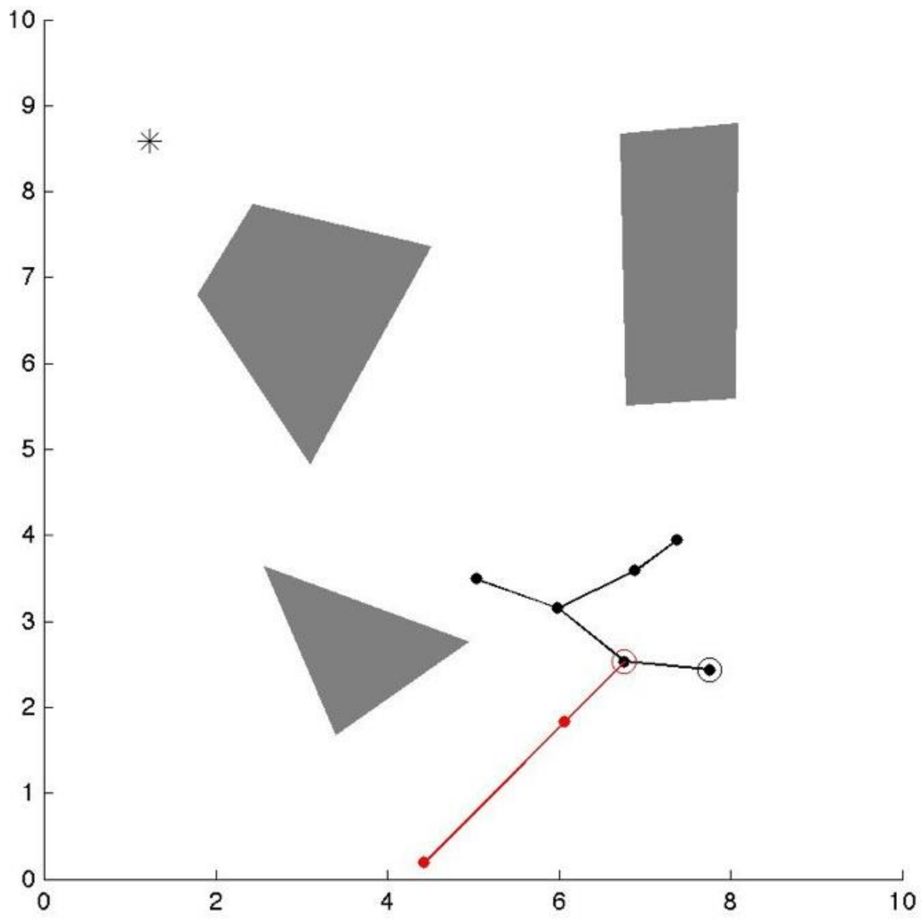
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad \boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

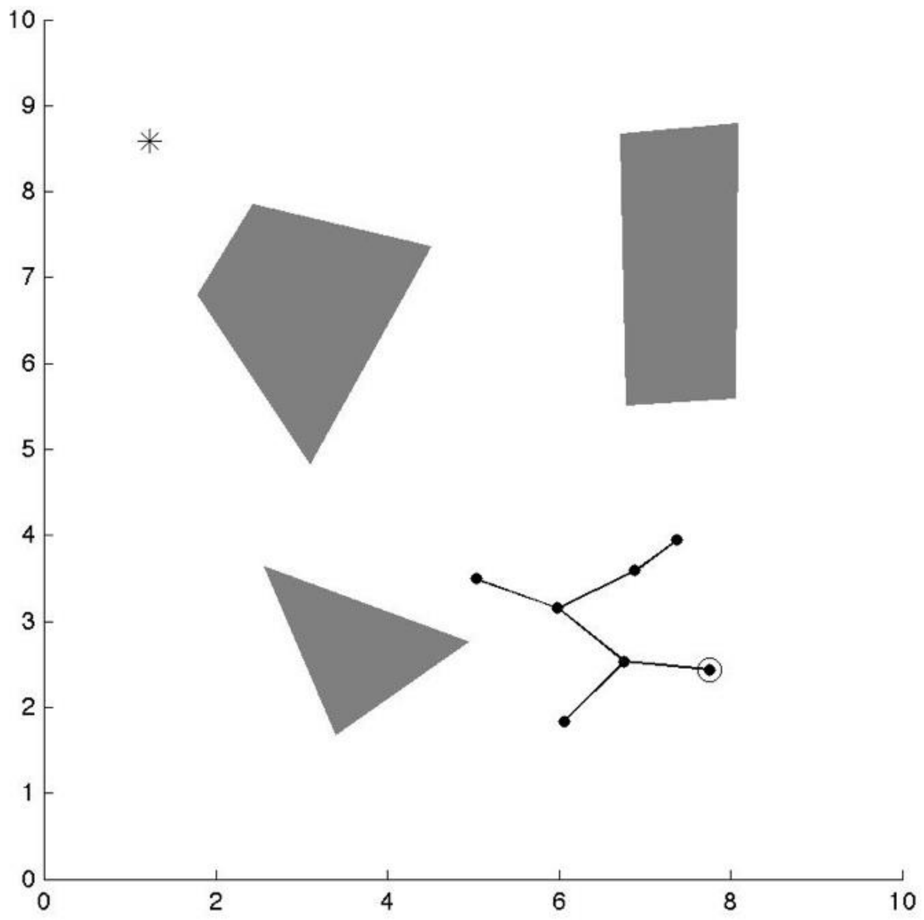
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

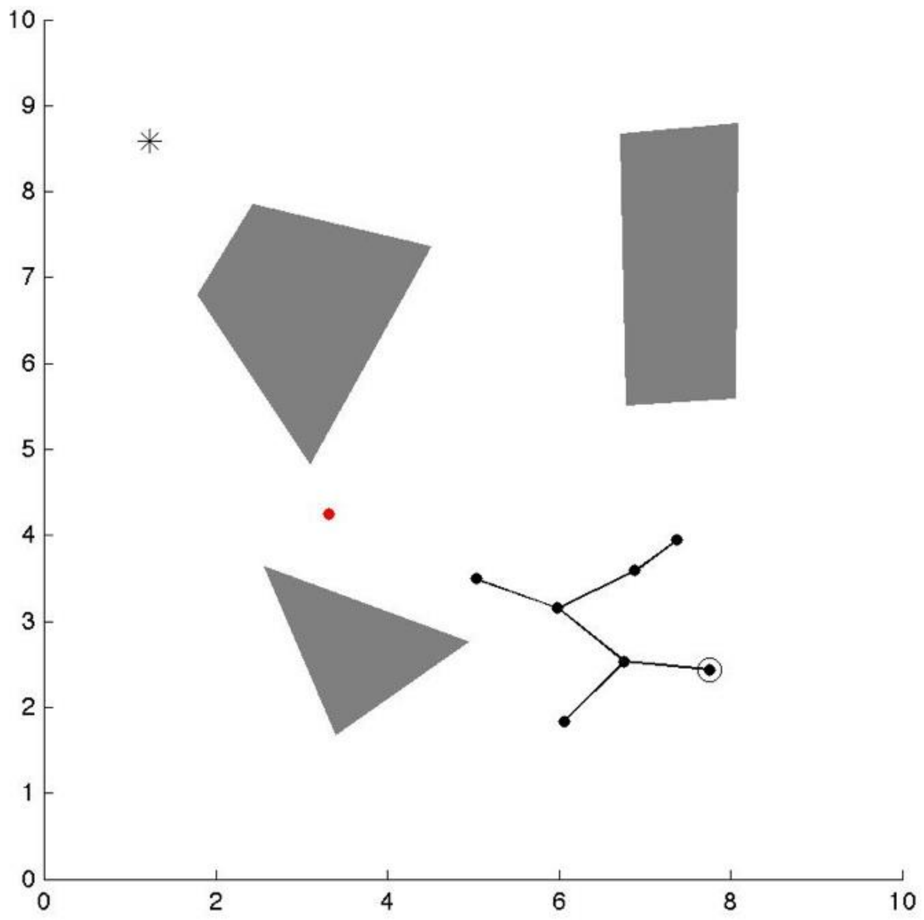
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$$

for $i = 1$ to N

$$G \leftarrow (V, E)$$

$$x_{rand} \leftarrow RandomSample()$$

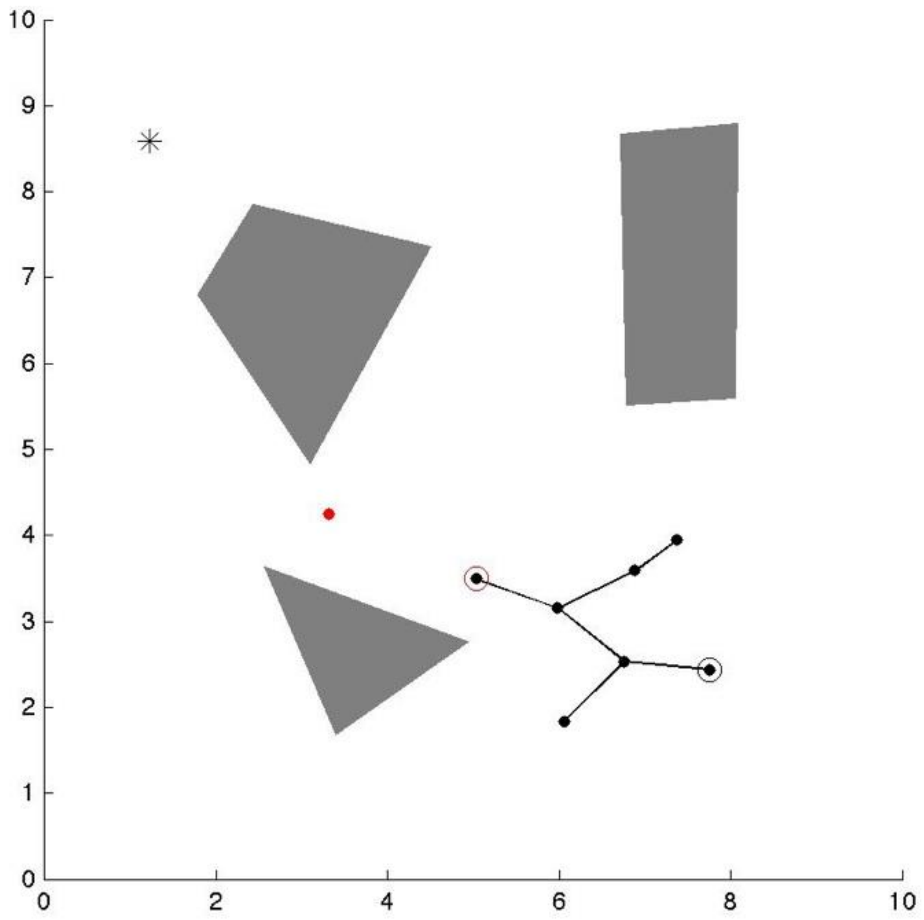
$$x_{nearest} \leftarrow Nearest(G, x_{rand})$$

$$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$$

if $ObstacleFree(x_{nearest}, x_{new})$

$$V \leftarrow V \cup \{x_{new}\}$$

$$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

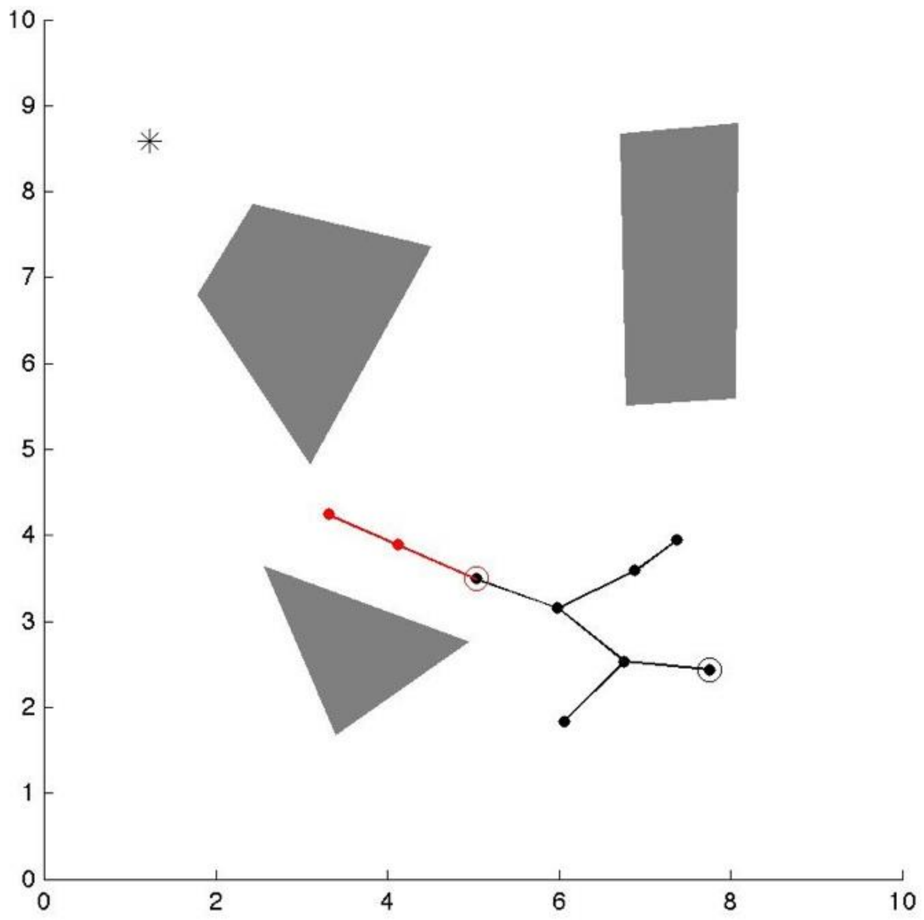
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad \boldsymbol{x_{new} \leftarrow Steer(x_{nearest}, x_{rand})}$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

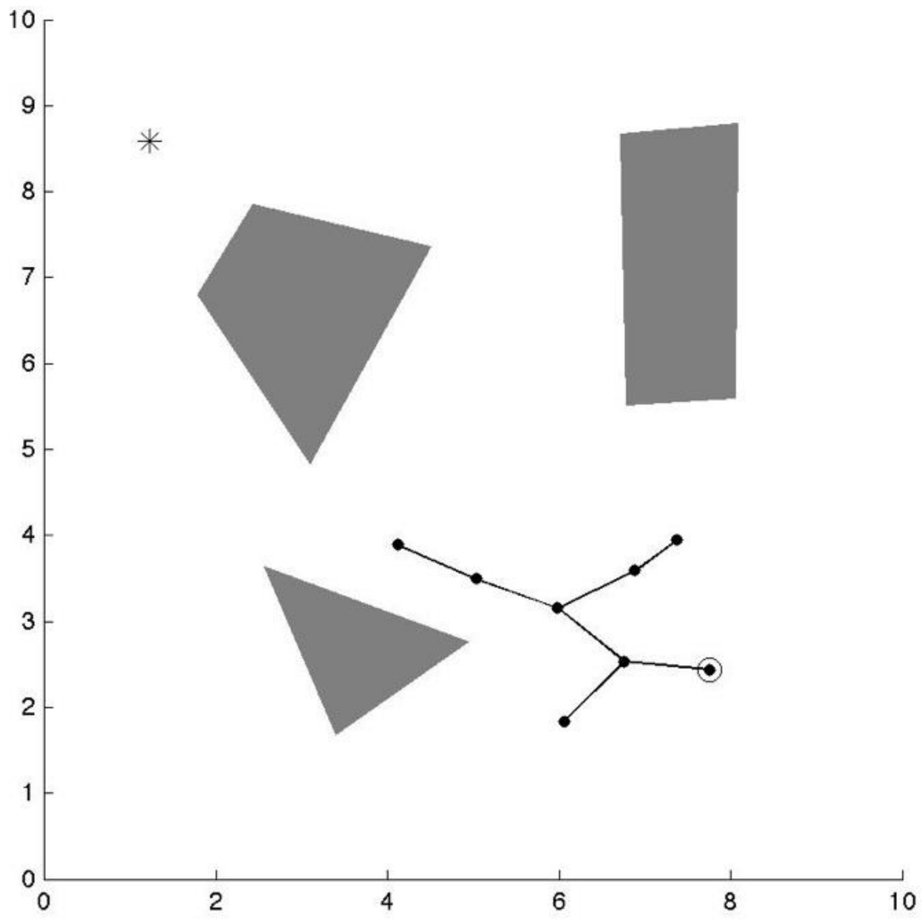
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

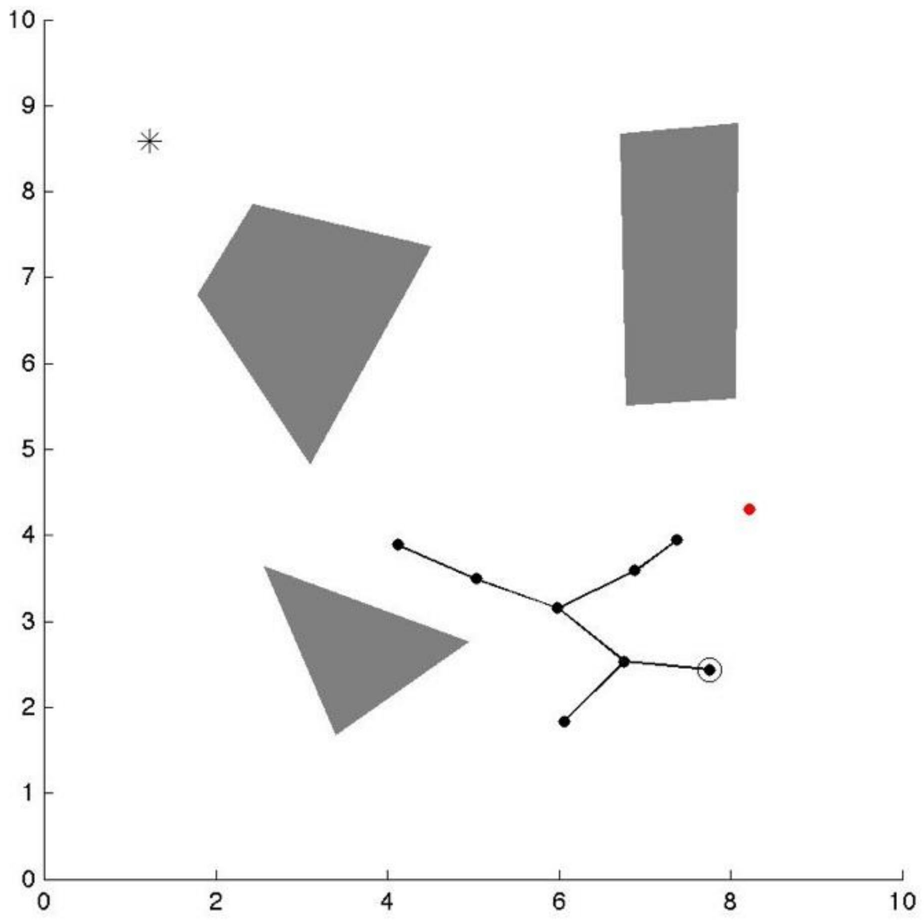
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

$x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

if $ObstacleFree(x_{nearest}, x_{new})$

$V \leftarrow V \cup \{x_{new}\}$

$E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

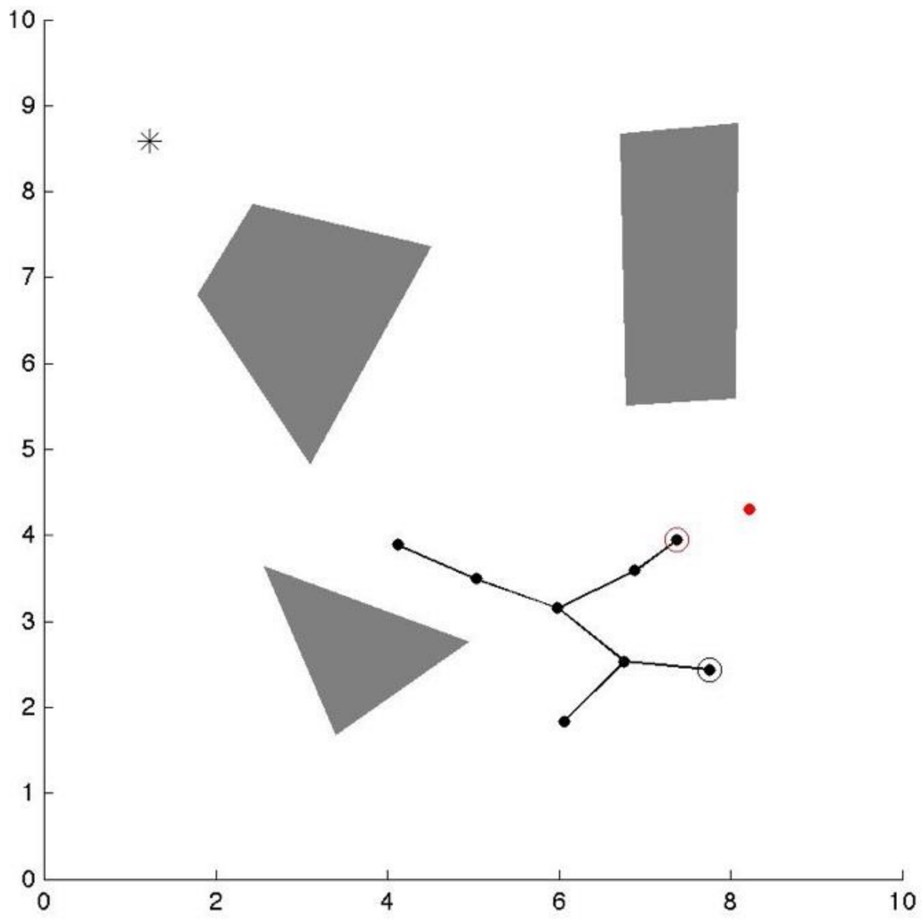
$\quad \boldsymbol{x_{nearest} \leftarrow Nearest(G, x_{rand})}$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

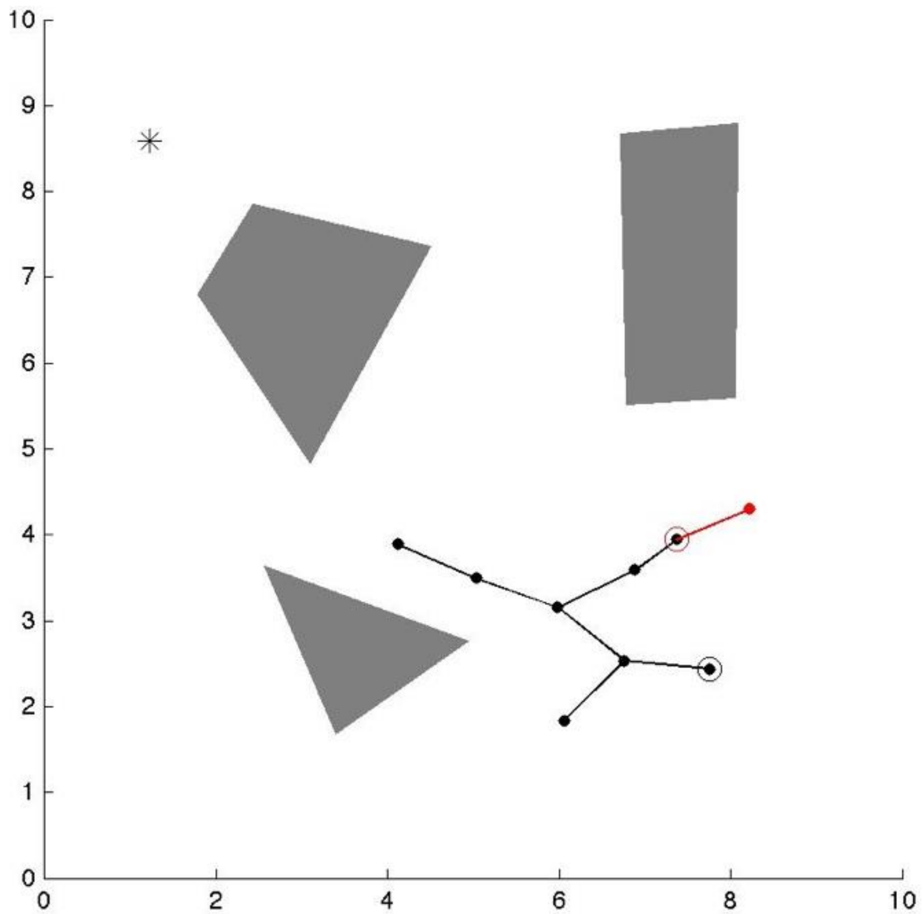
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

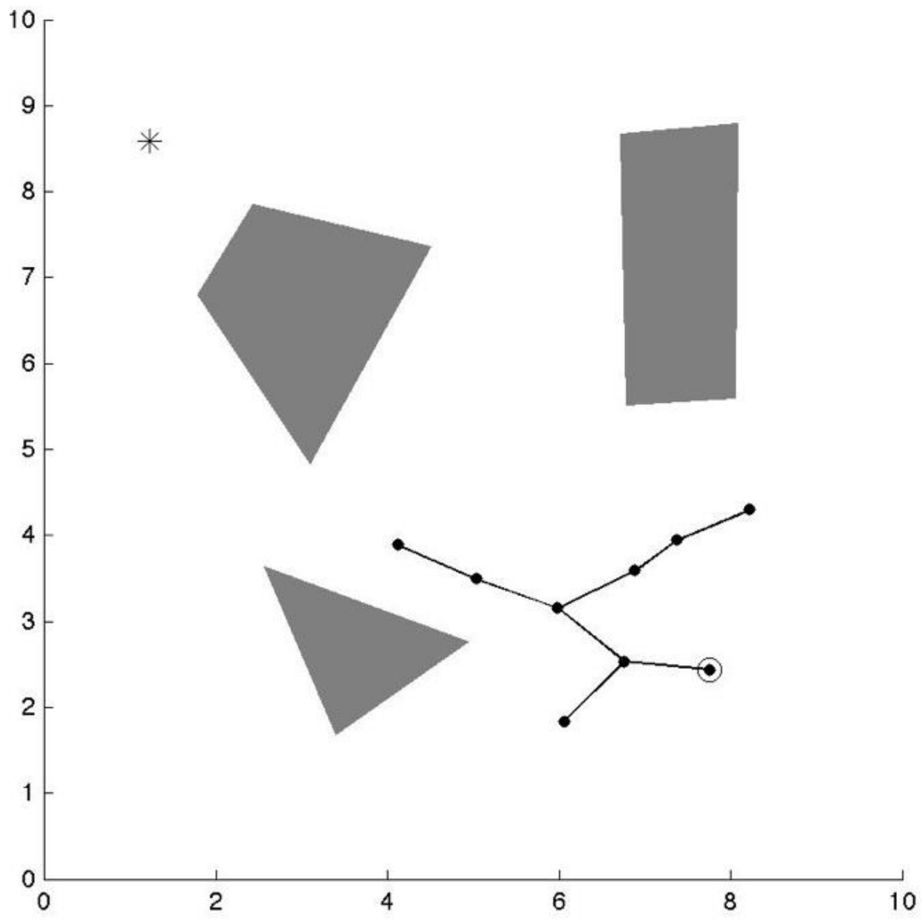
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad x_{rand} \leftarrow RandomSample()$

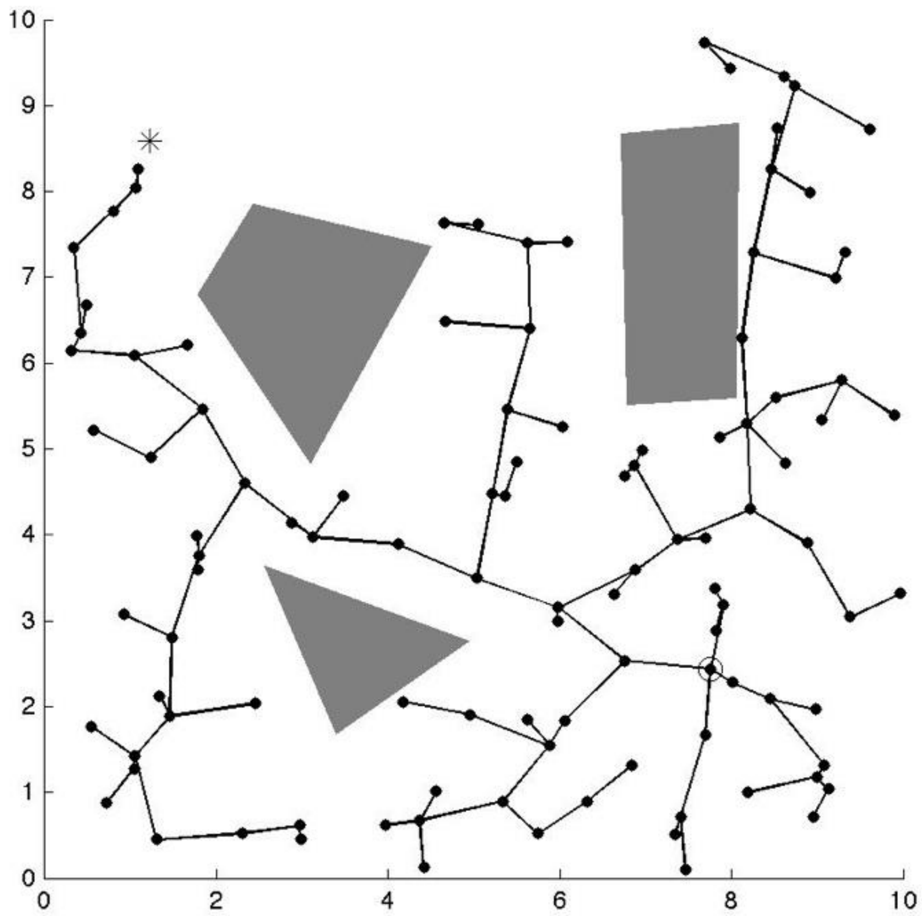
$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$



RRT - Bias to Goal

$V \leftarrow \{x_{init}\}; \quad E \leftarrow \emptyset$

for $i = 1$ to N

$\quad G \leftarrow (V, E)$

$\quad$ **with** probability p

$\quad\quad x_{rand} \leftarrow RandomSample()$

$\quad$ **otherwise**

$\quad\quad x_{rand} \leftarrow x_{goal}$

$\quad x_{nearest} \leftarrow Nearest(G, x_{rand})$

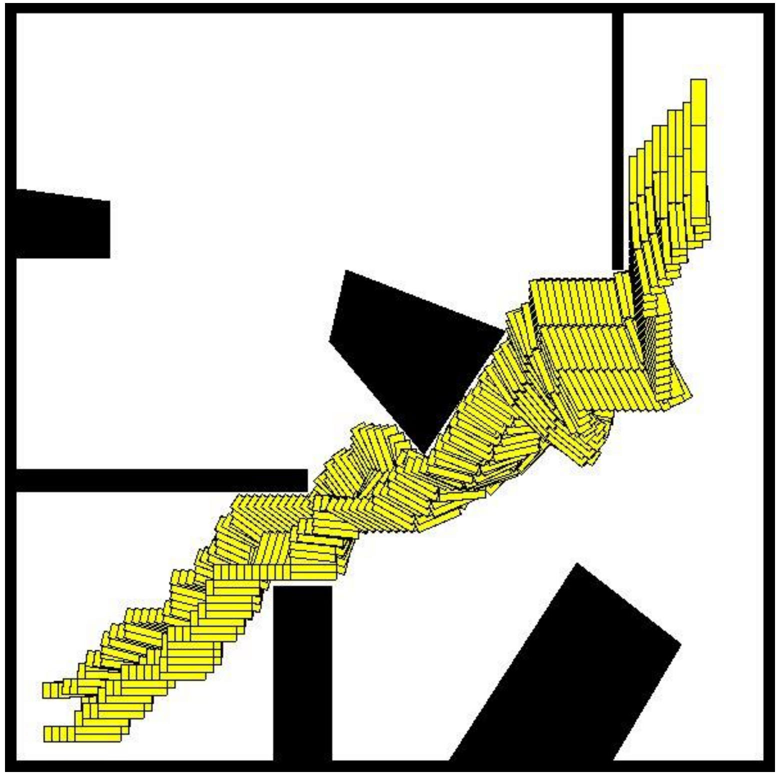
$\quad x_{new} \leftarrow Steer(x_{nearest}, x_{rand})$

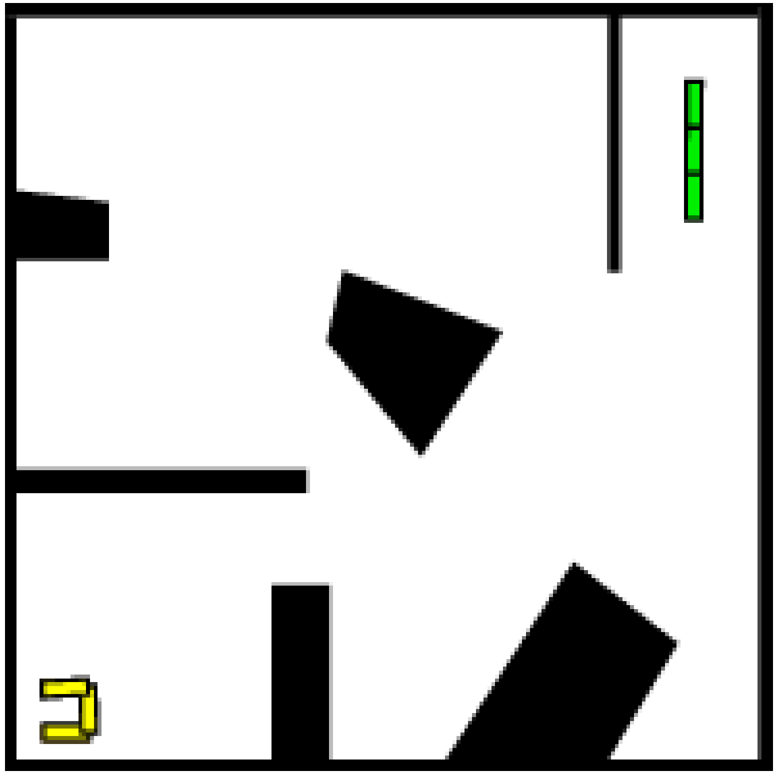
$\quad$ **if** $ObstacleFree(x_{nearest}, x_{new})$

$\quad\quad V \leftarrow V \cup \{x_{new}\}$

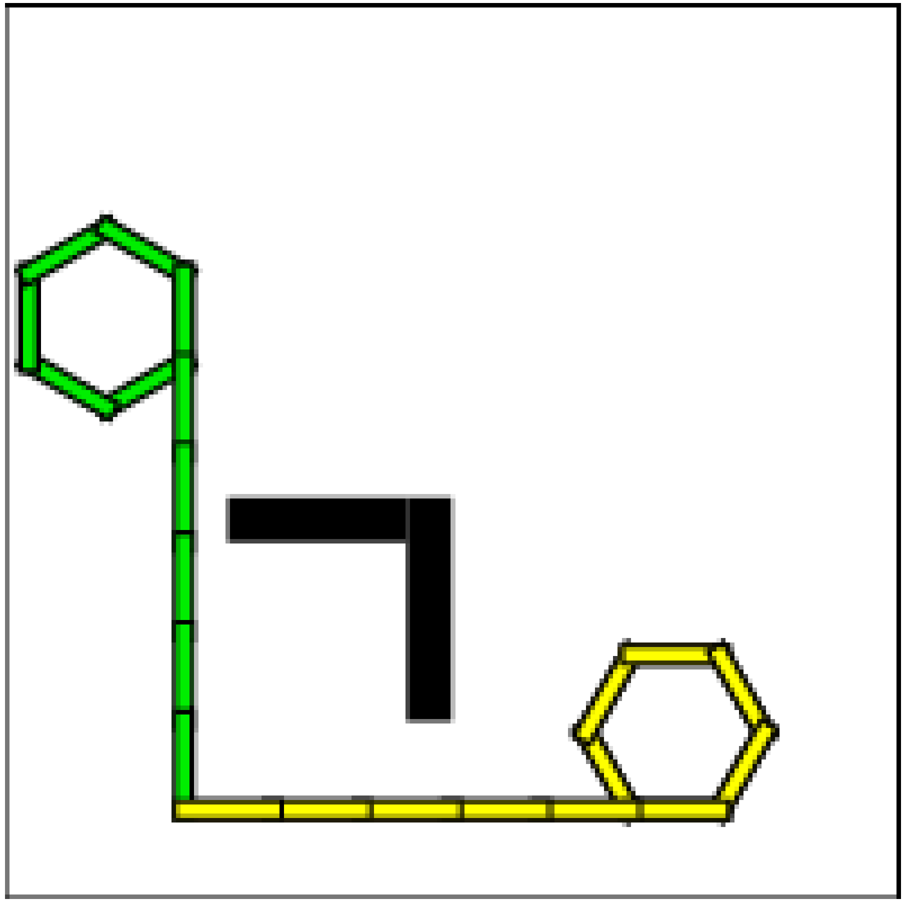
$\quad\quad E \leftarrow E \cup \{(x_{nearest}, x_{new})\}$

Articulated Robot

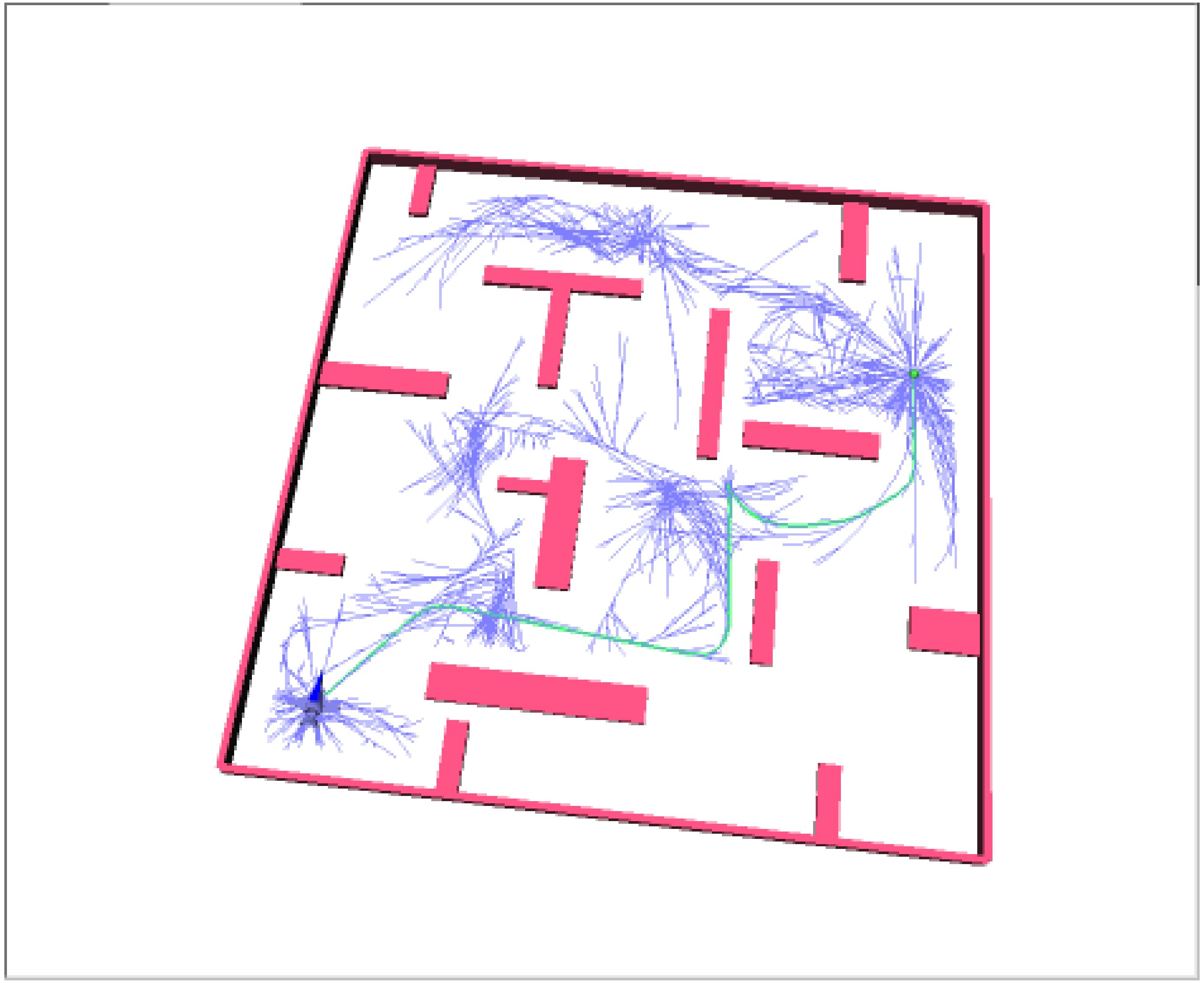




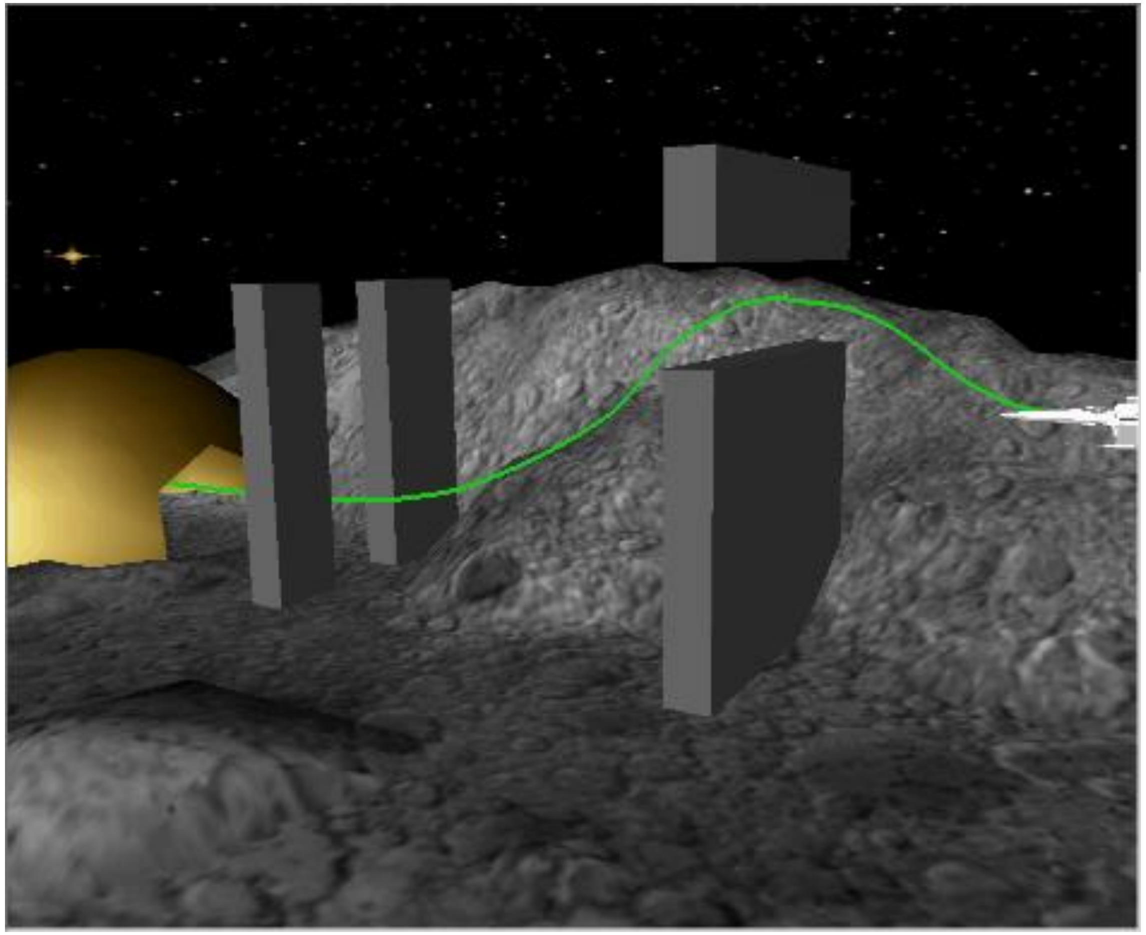
Highly Articulated Robot



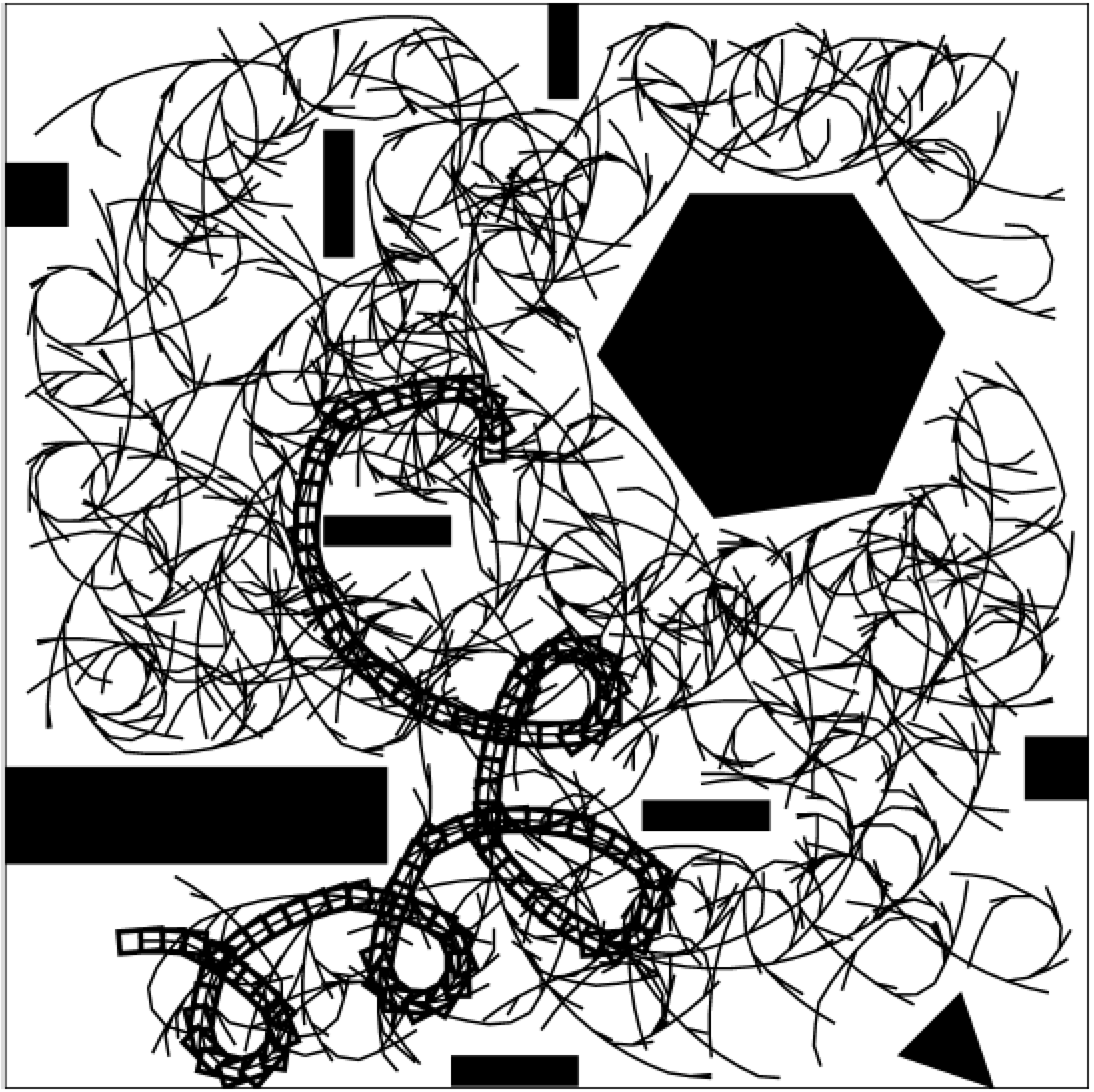
Hovercraft with 2 Thusters



Out of This World Demo



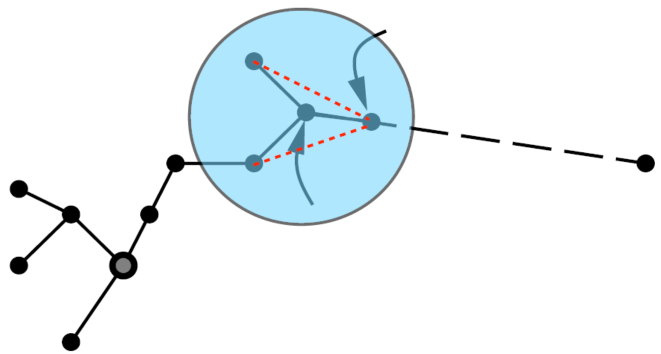
Left-turn only forward car



Rapidly-Exploring Random Tree (RRT)

- Advantages of RRT: very fast, works well for dynamic environments
- Disadvantages: Not optimal
  - in fact, it has been proven by Karaman & Frazzoli that the probability of RRT converging to an optimal solution is 0

Variants of RRT



- There are (very) many…
- Rapidly-exploring Random Graph (RRG):
  - Connect all vertices within neighboring region, forming a graph
- RRT*:
  - a variant of RRG that essentially “rewires" the tree as better paths are discovered.

Summary

- Both RRT and PRM are examples of **sampling based algorithms** that are **probabilistically complete**

- Definition: A path planner is ***probabilistically complete*** if, given a solvable problem, the probability that the planner solves the problem goes to 1 as time goes to infinity.

Links to Further Reading

- Steve LaValle's online book:
  "Planning Algorithms" *(chapters 5 & 14)*
  http://planning.cs.uiuc.edu/

- The RRT page:
  http://msl.cs.uiuc.edu/rrt/

- Motion Planning Benchmarks
  Parasol Group, Texas A&M
  http://parasol.tamu.edu/groups/amatogroup/benchmarks/mp/