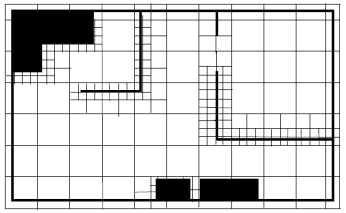
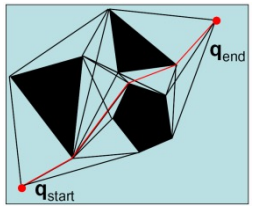
Lecture 21

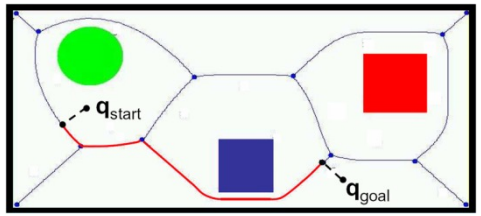
Probabilistic Path Planning: RRTs

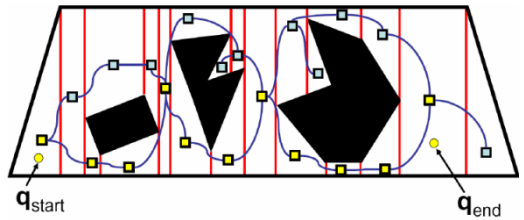
CS 3630

Recap









Last time we discussed multiple approaches for path planning

- occupancy grids
- roadmap algorithms (visibility graphs, voronoi diagrams)

All of them require exhaustively reproducing the entire state in some searchable representation

How do we evaluate plan algorithm quality?

- A **complete** algorithm finds a path if one exists, and otherwise indicates that there is no path.
  - Returns an answer in finite time.

- An **optimal** algorithm finds the shortest path to the goal

The challenge of classic approaches to path planning:

- Memory and compute limitations:
  - Run time increases exponentially with the dimensionality of the configuration space.
    - Our examples were in 2D, where a 10x10 occupancy grid is only 100 cells
    - In d-dimentional space, there are 10^d grid cells

The Rise of Monte Carlo Techniques

- **KEY IDEA:** Rather than exhaustively explore ALL possibilities, randomly explore a smaller subset of possibilities while keeping track of progress

- What's the catch?
  - Typically, we must sacrifice both completeness and optimality.
  - Classic tradeoff between solution quality and runtime performance.

- **Sampling Based Planning:**
  - Search for collision-free path only by sampling points.

Probabilistic Completeness

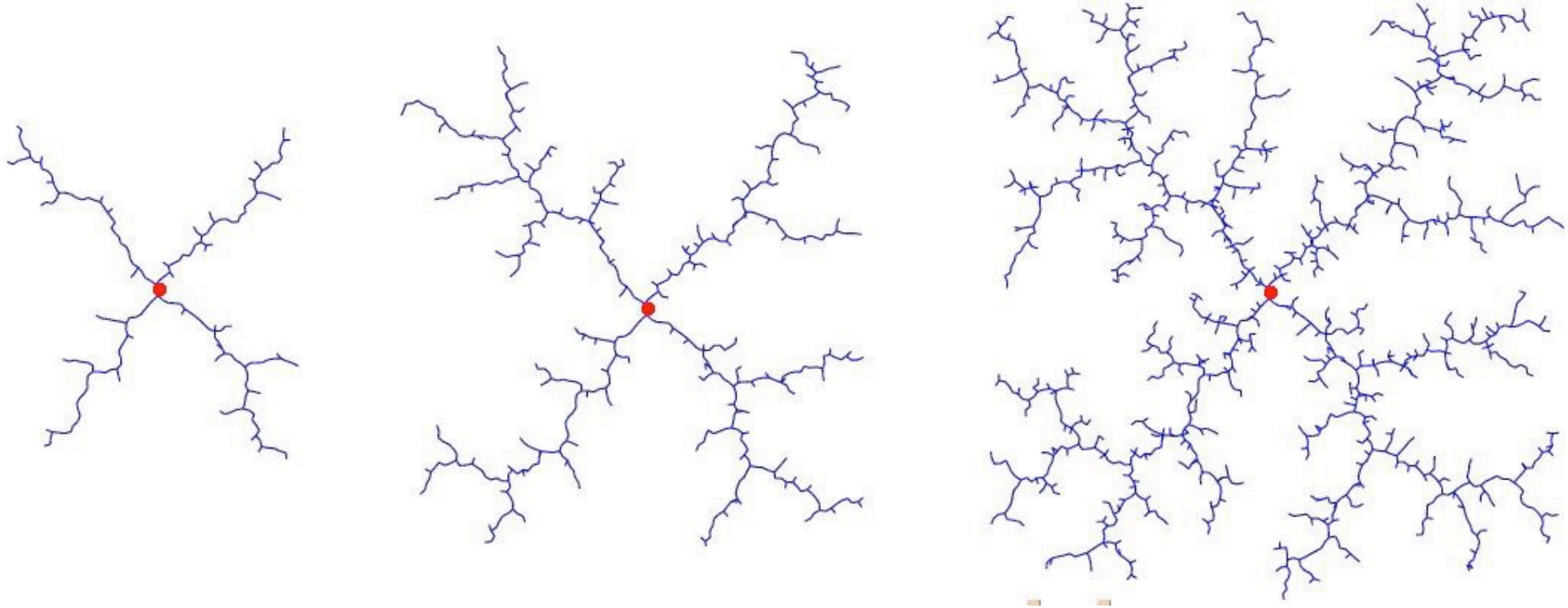
- A **probabilistically complete** algorithm will find a path with high probability, given that one exists.
  - Often does not report that a path doesn’t exist, just that one hasn’t been found within allotted time.

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.

Rapidly-Exploring Random Tree (RRT)

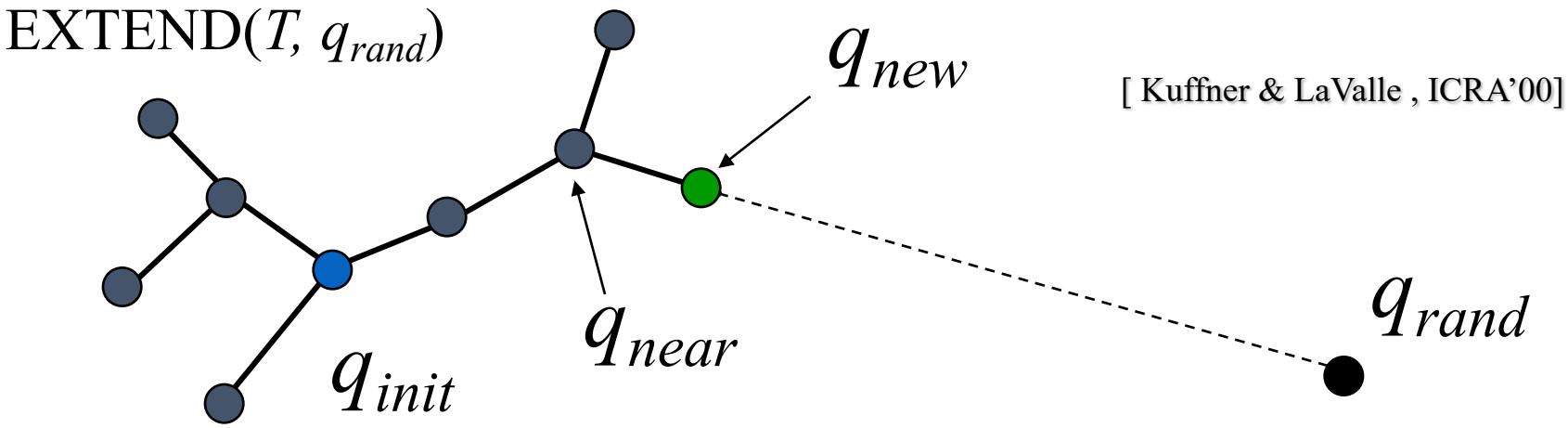


The Basic Idea: Iteratively expand the tree

- 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}

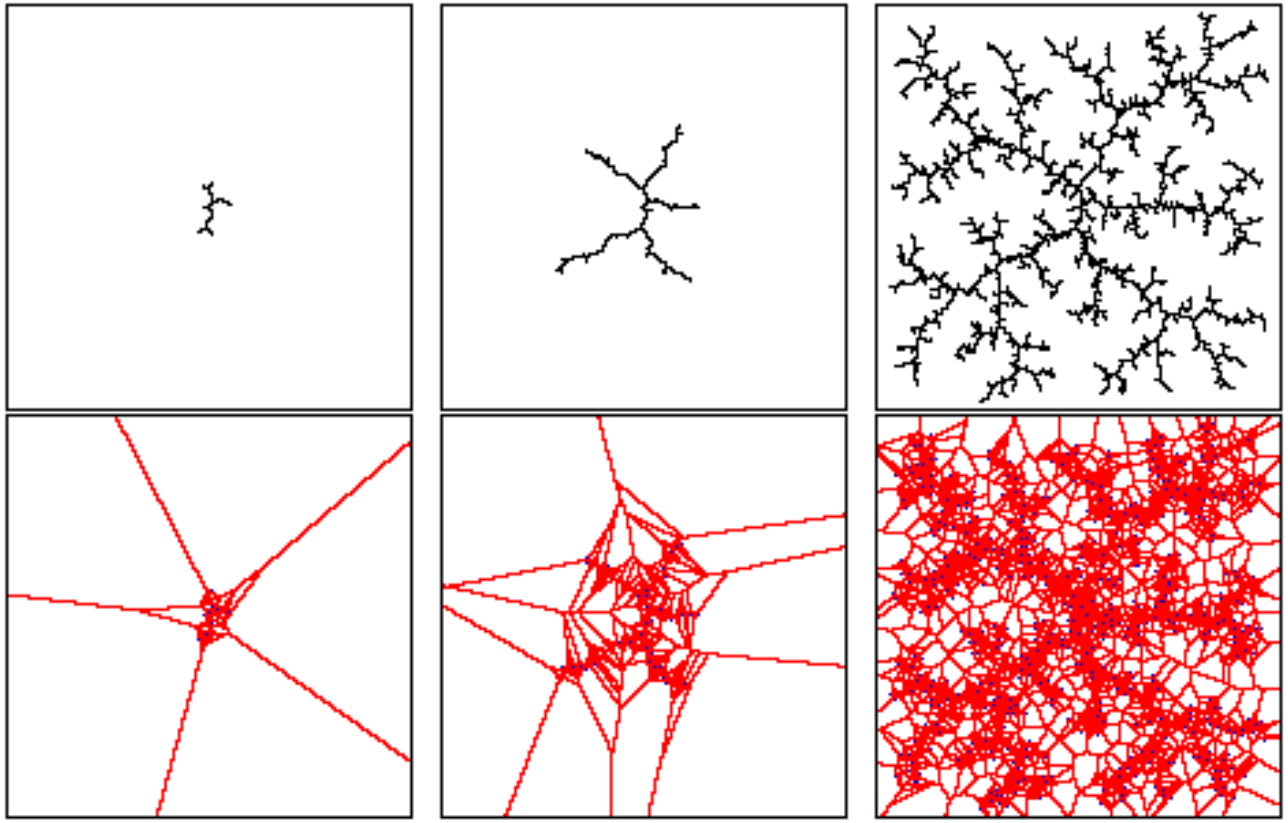
Path Planning with RRTs

```
BUILD_RRT (q_init)  {
   T.init(q_init);
   for k =  1 to K do
      q_rand = RANDOM_CONFIG();
      EXTEND(T, q_rand)
}
```



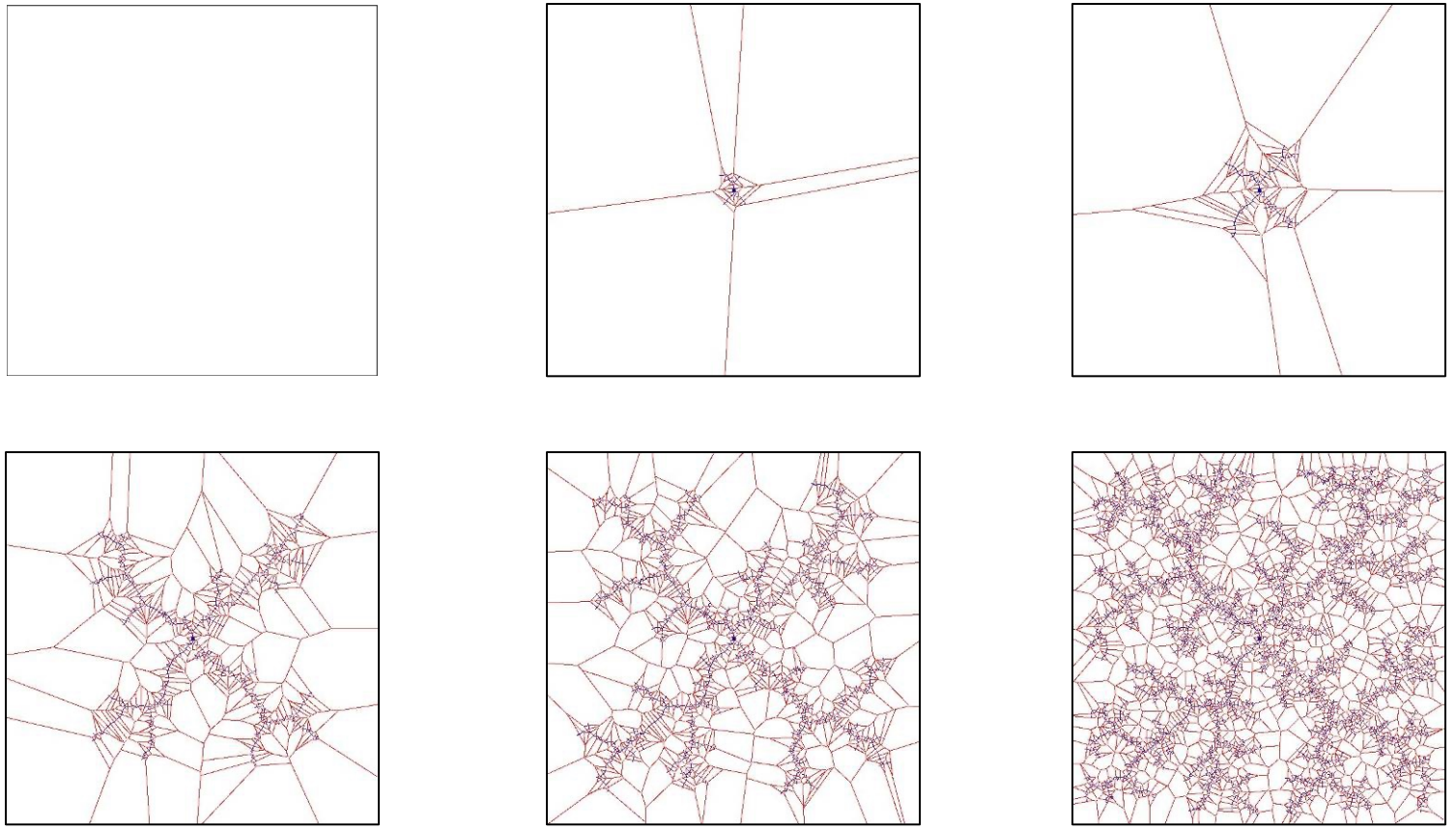
[Kuffner & LaValle , ICRA'00]

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.

RRTs and
Bias toward large Voronoi regions



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

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… most of the time 5-10% works well

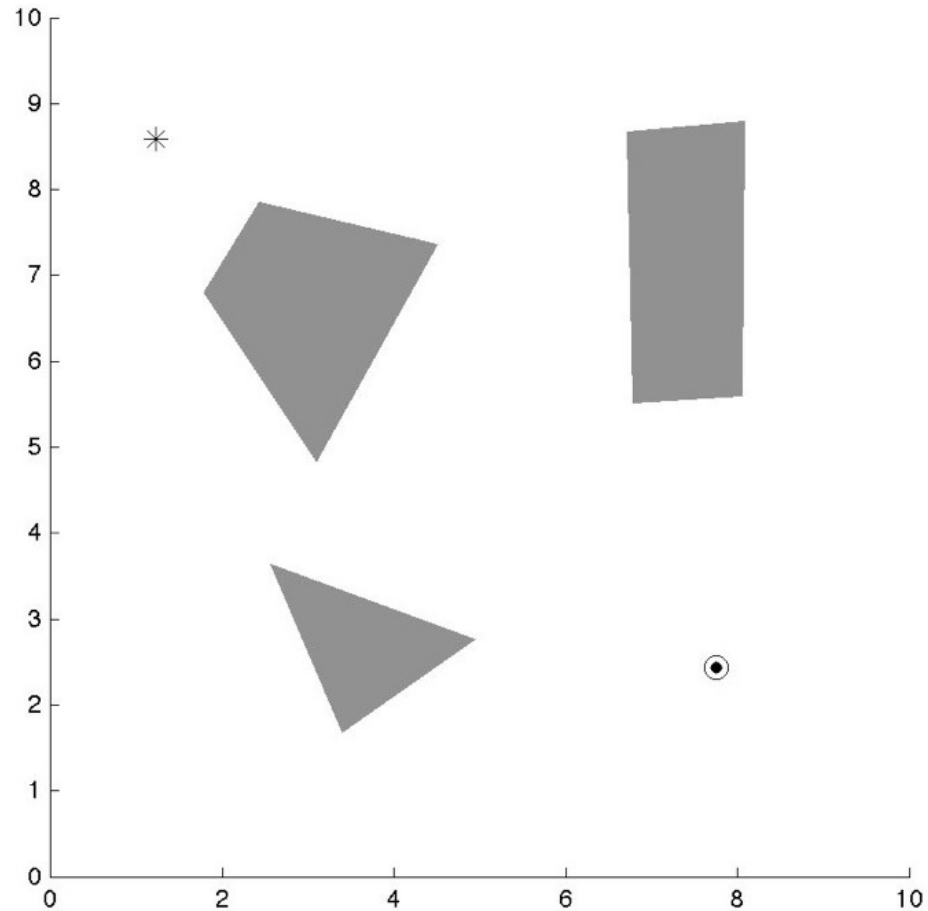
RRT

- 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 path between two points is in the free space

RRT in Action…

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

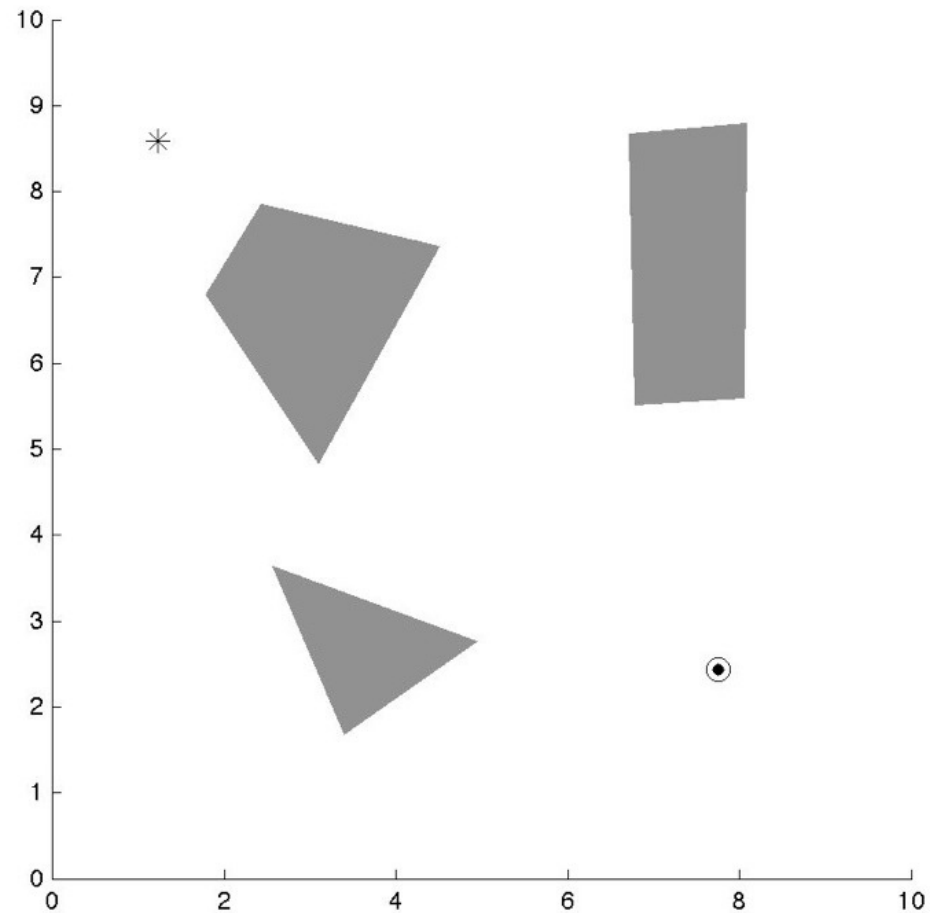
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

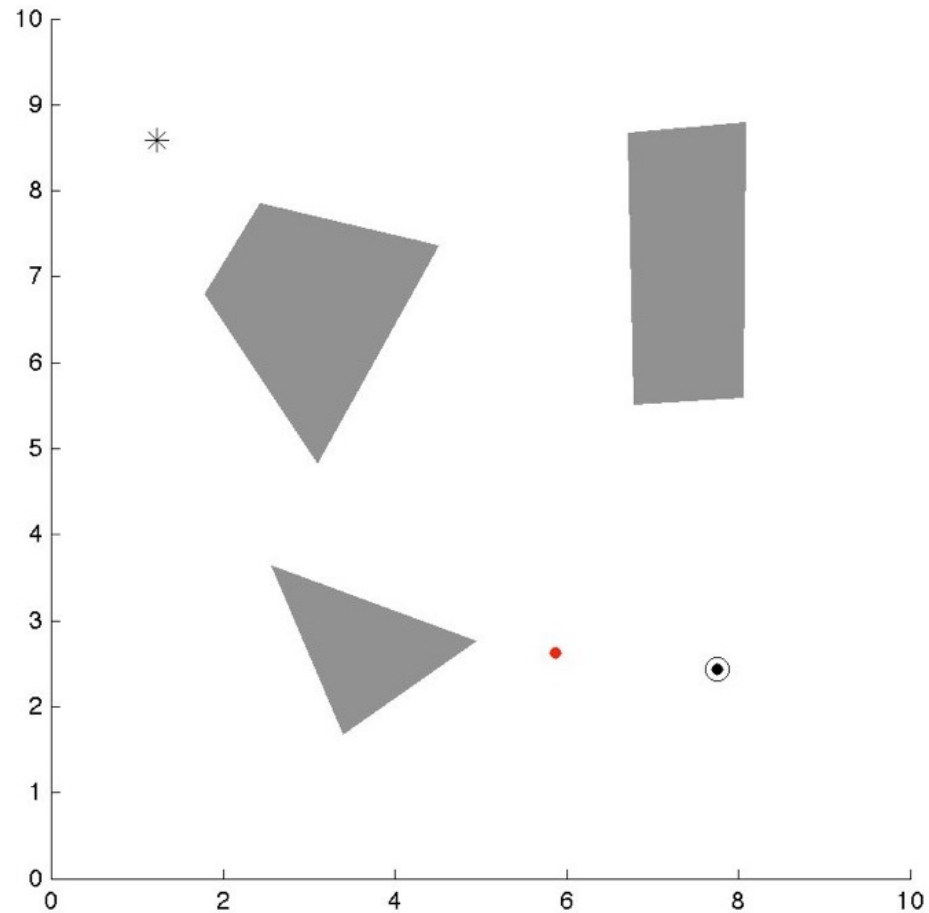
$V \leftarrow V \cup \{x_{new}\}$

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



RRT

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

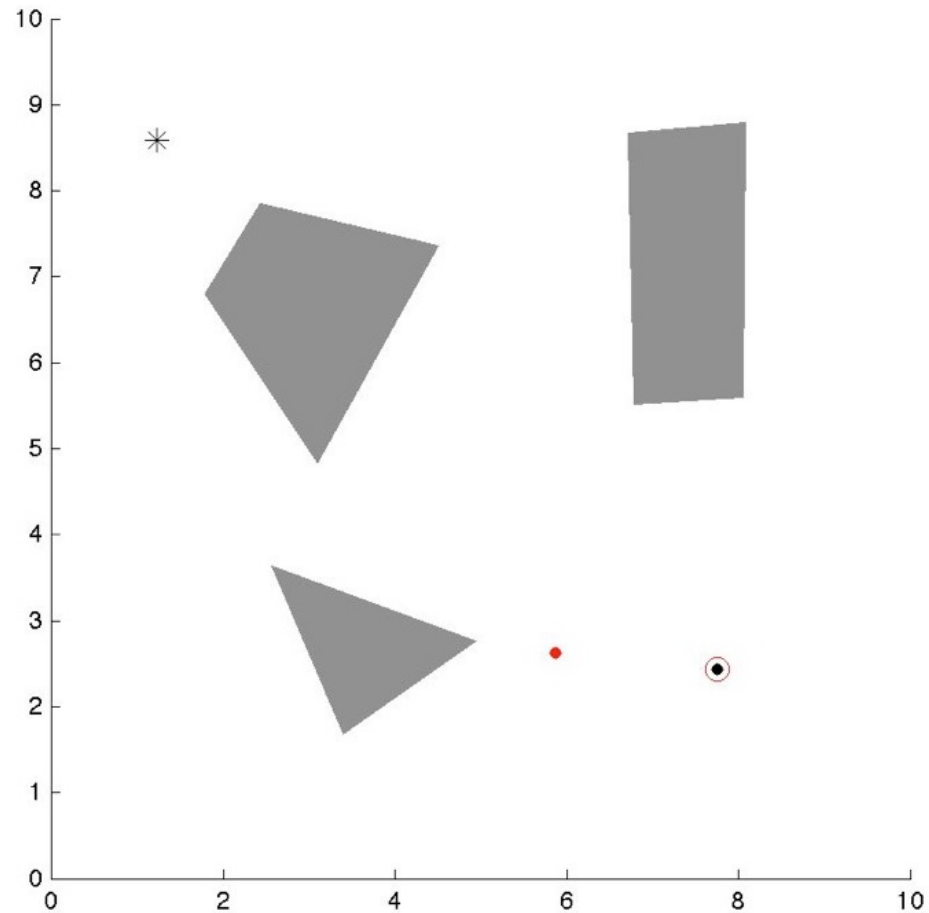
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

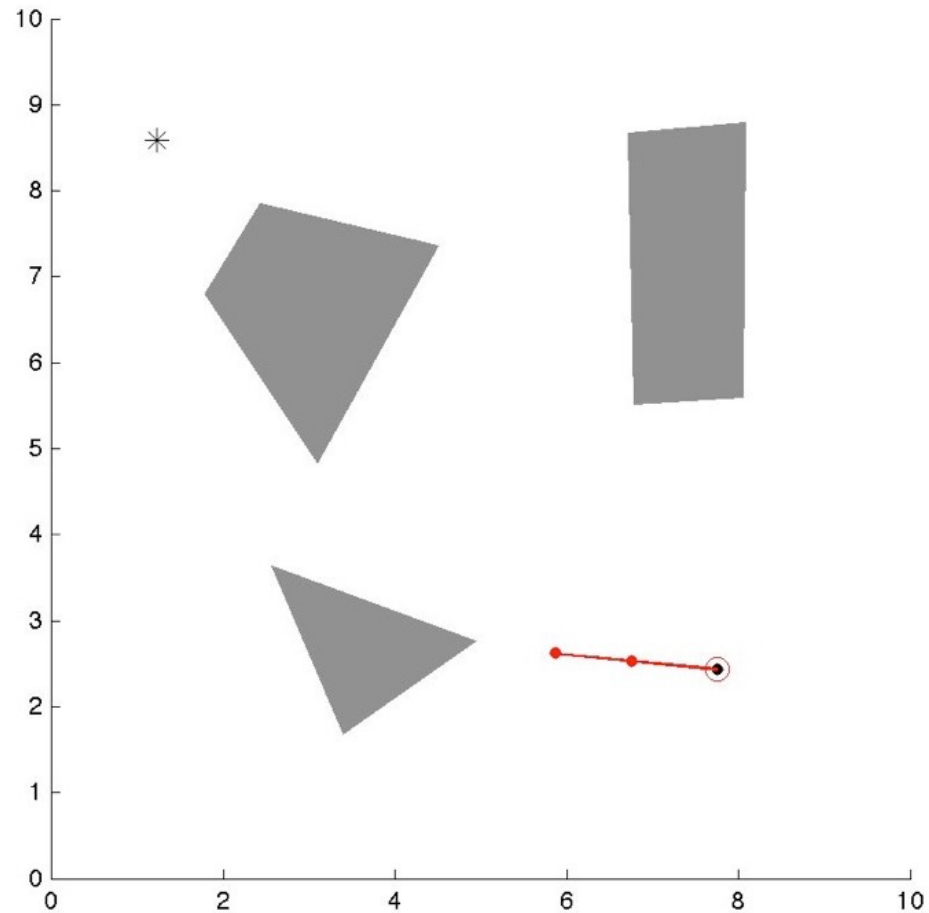
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

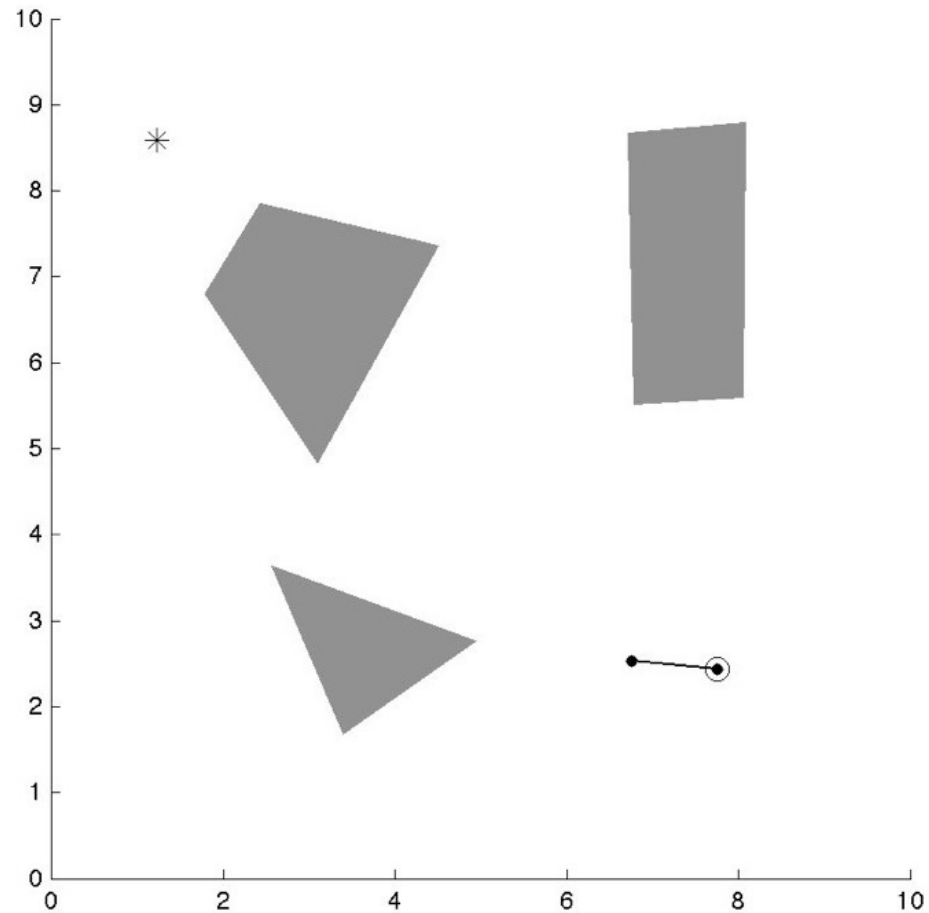
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

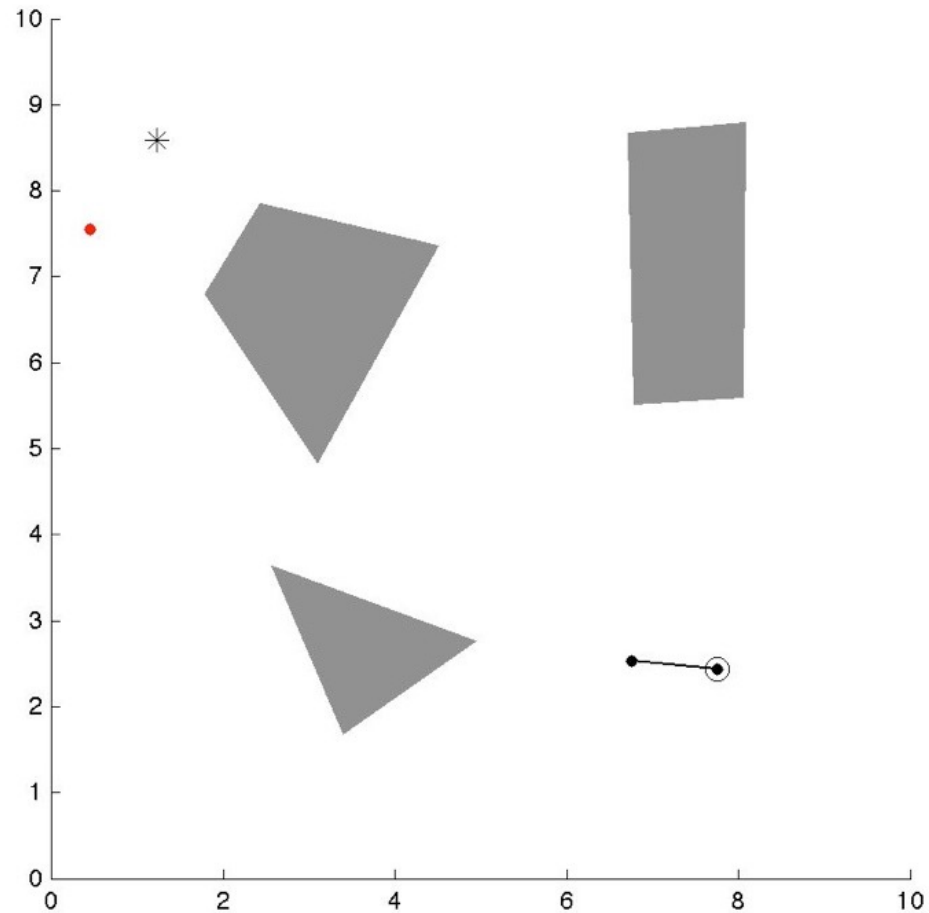
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

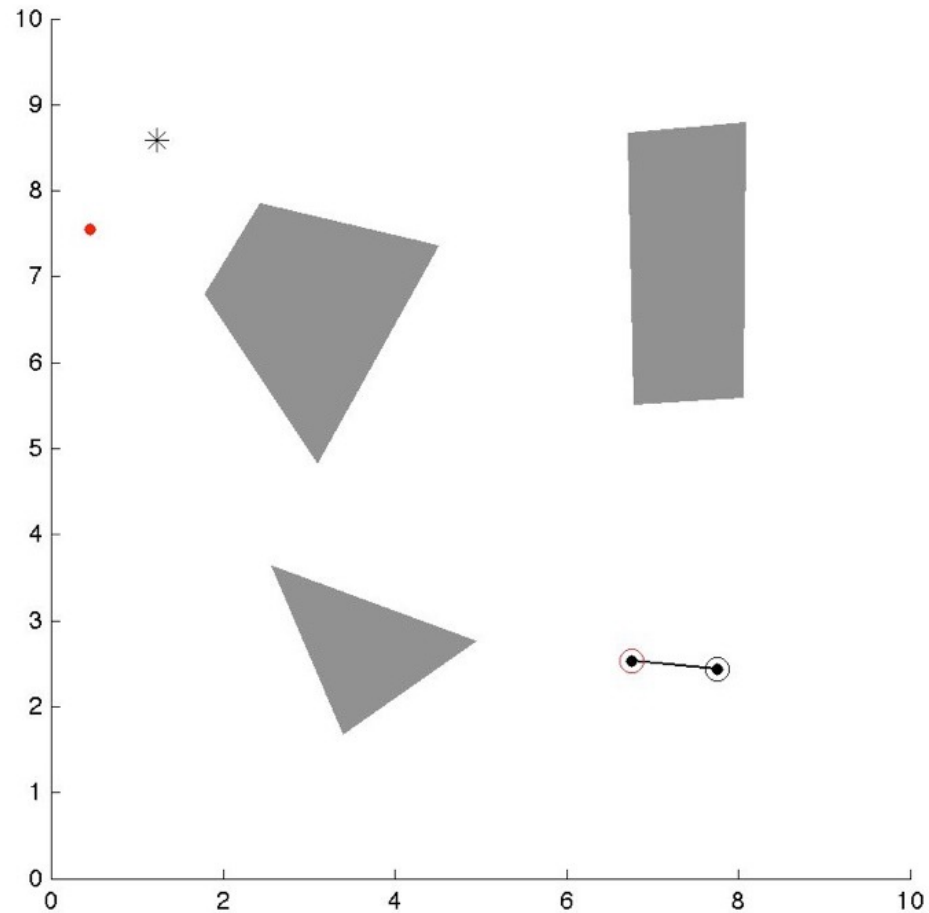
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

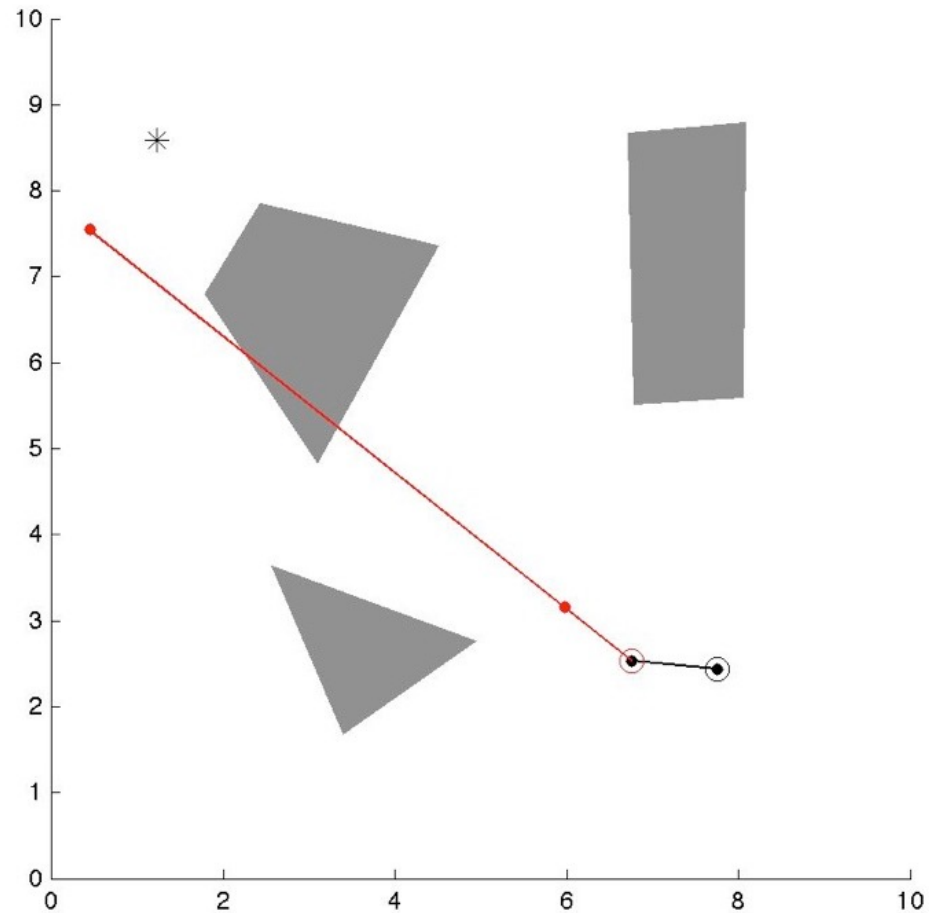
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

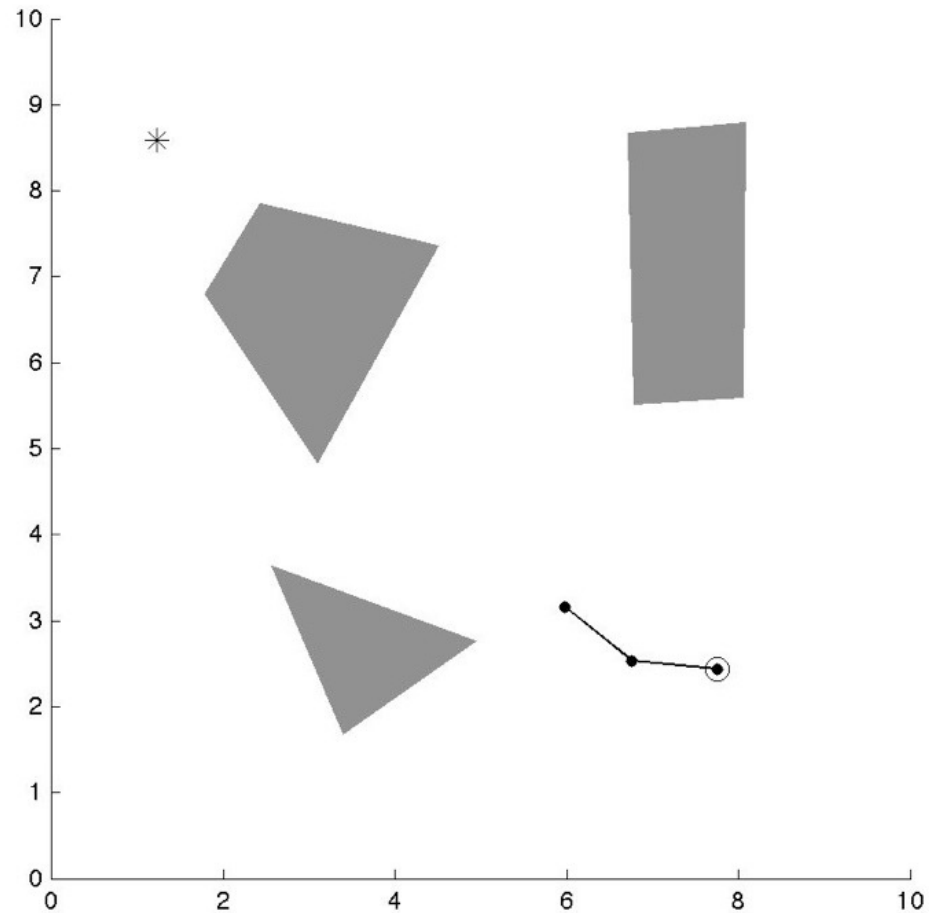
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

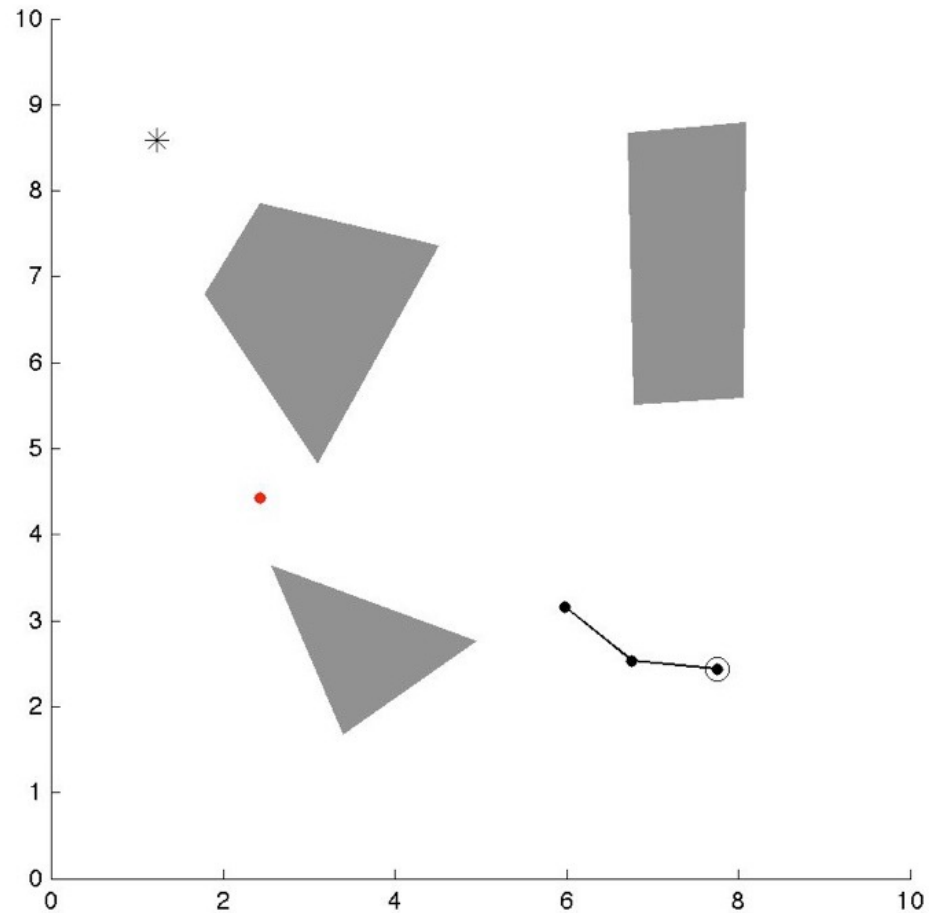
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

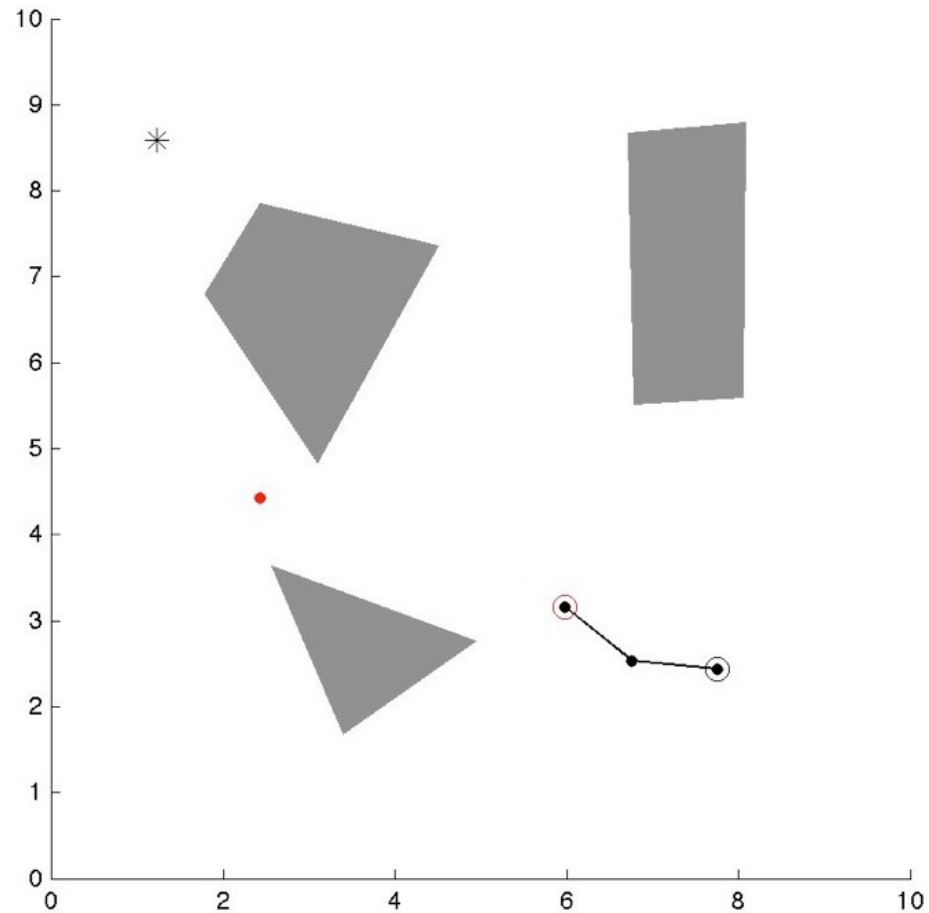
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

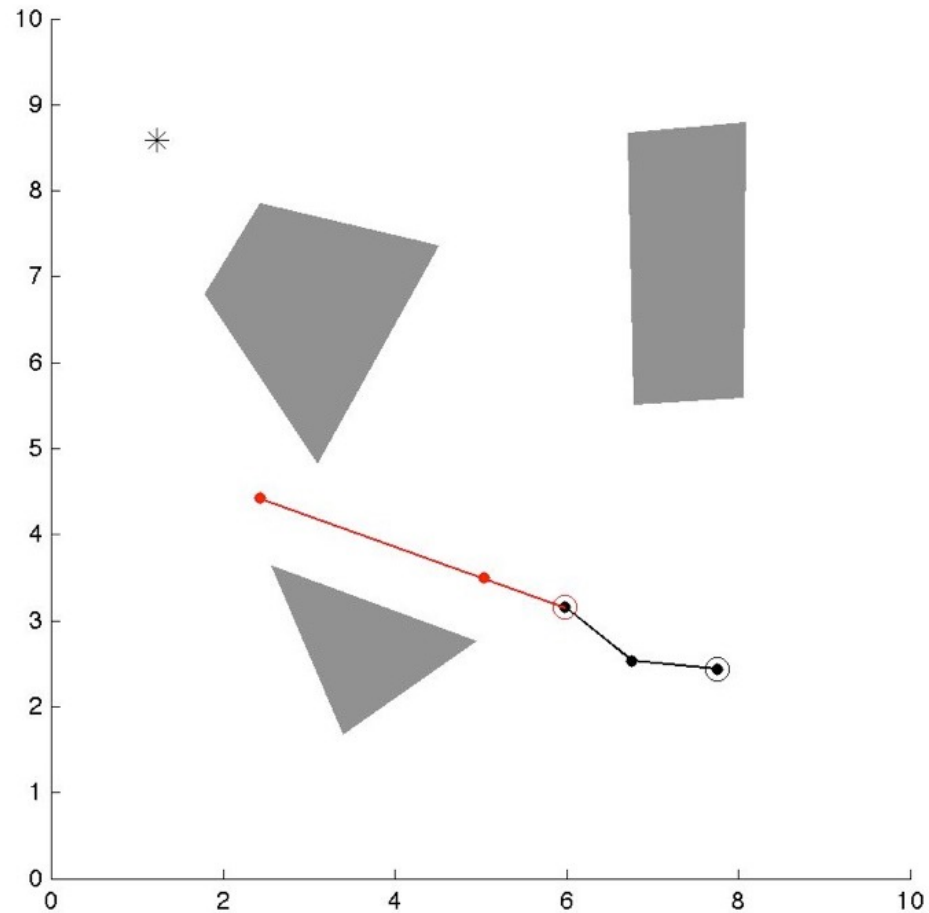
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

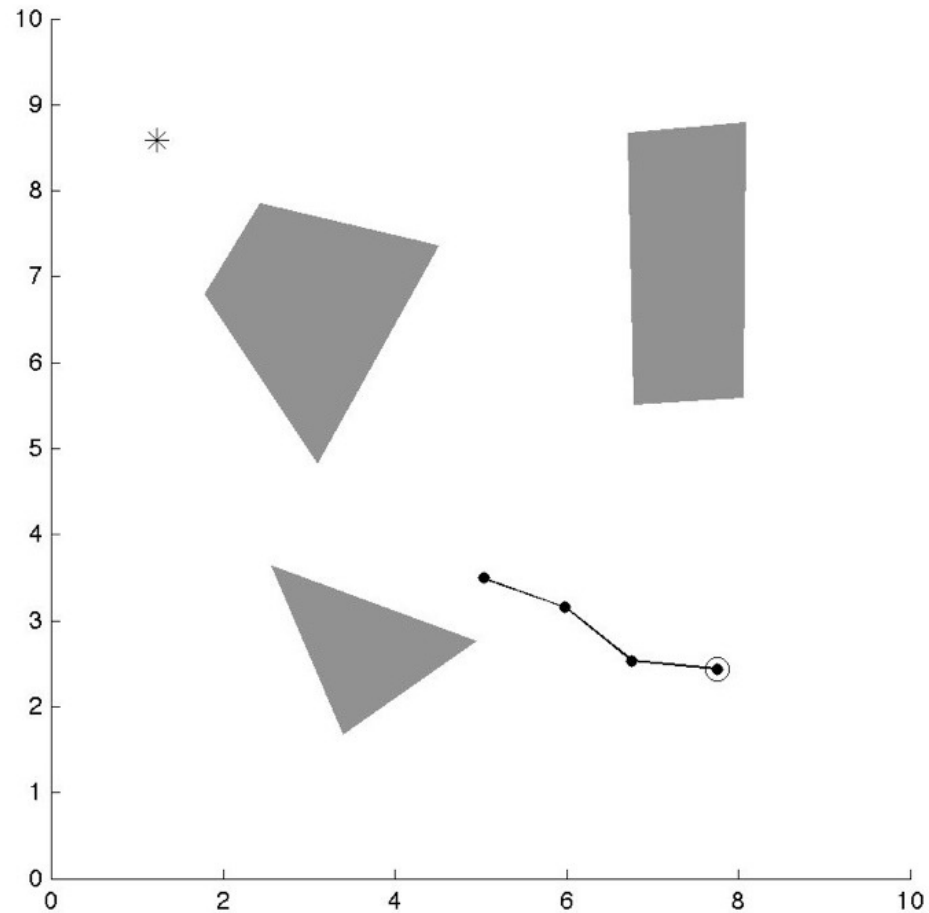
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

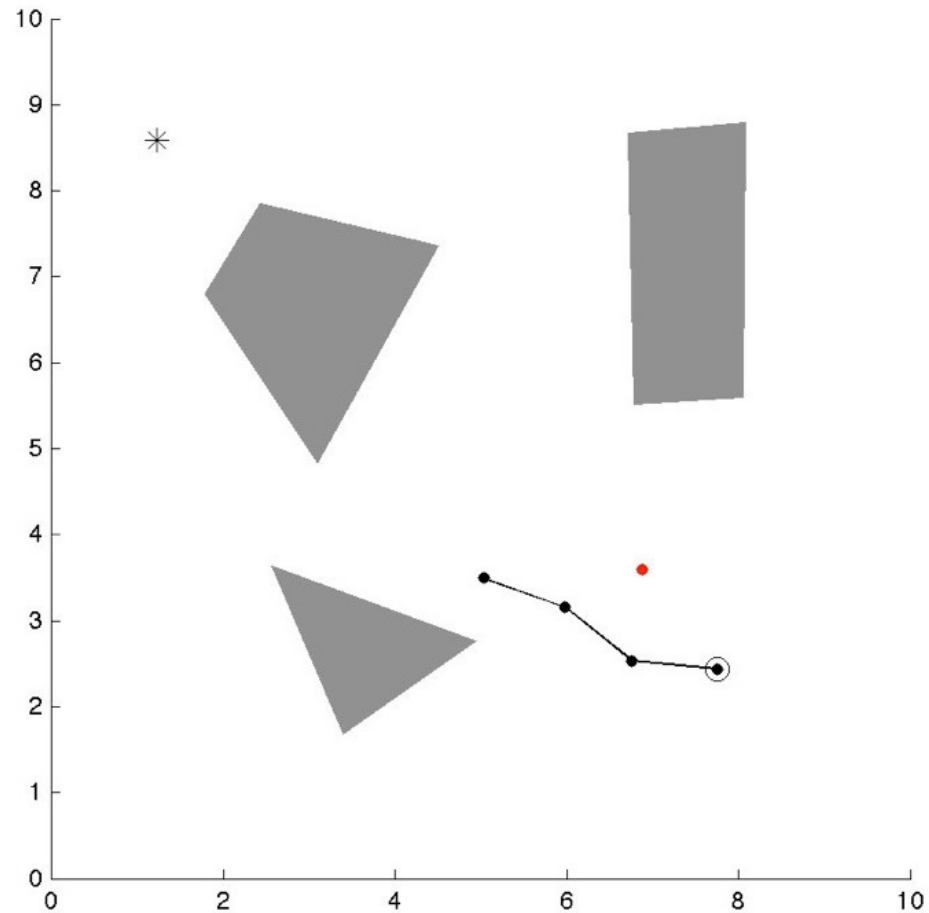
$V \leftarrow V \cup \{x_{new}\}$

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



RRT

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

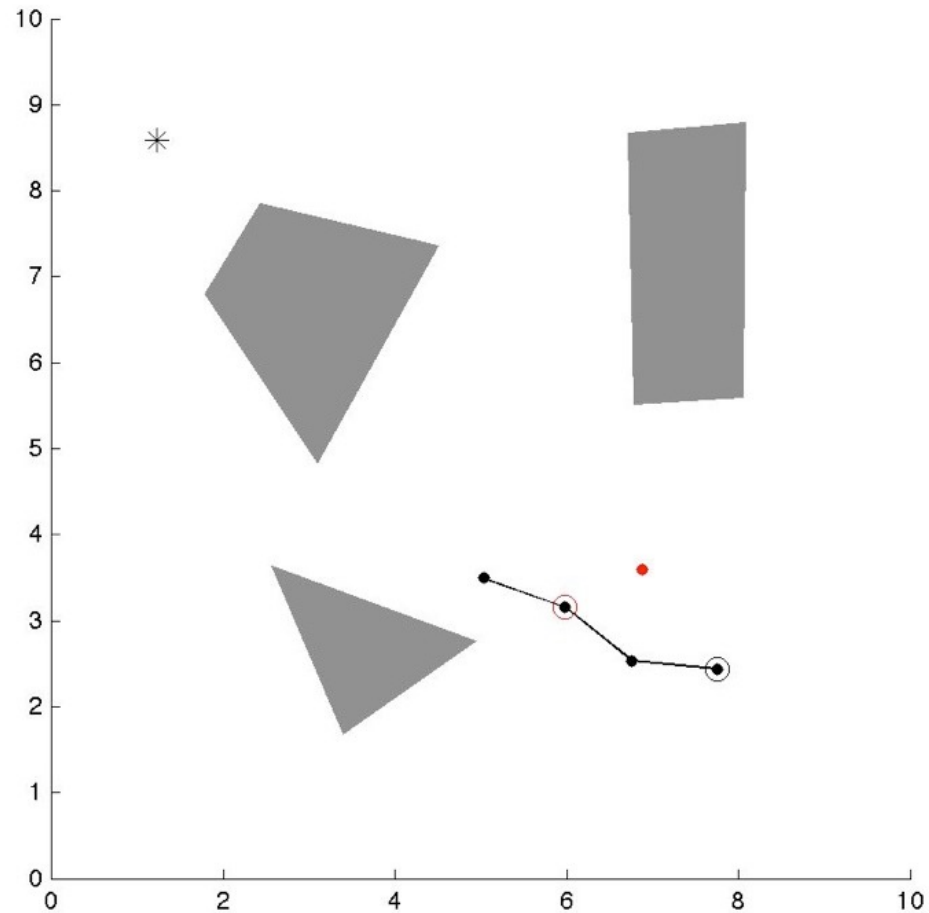
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

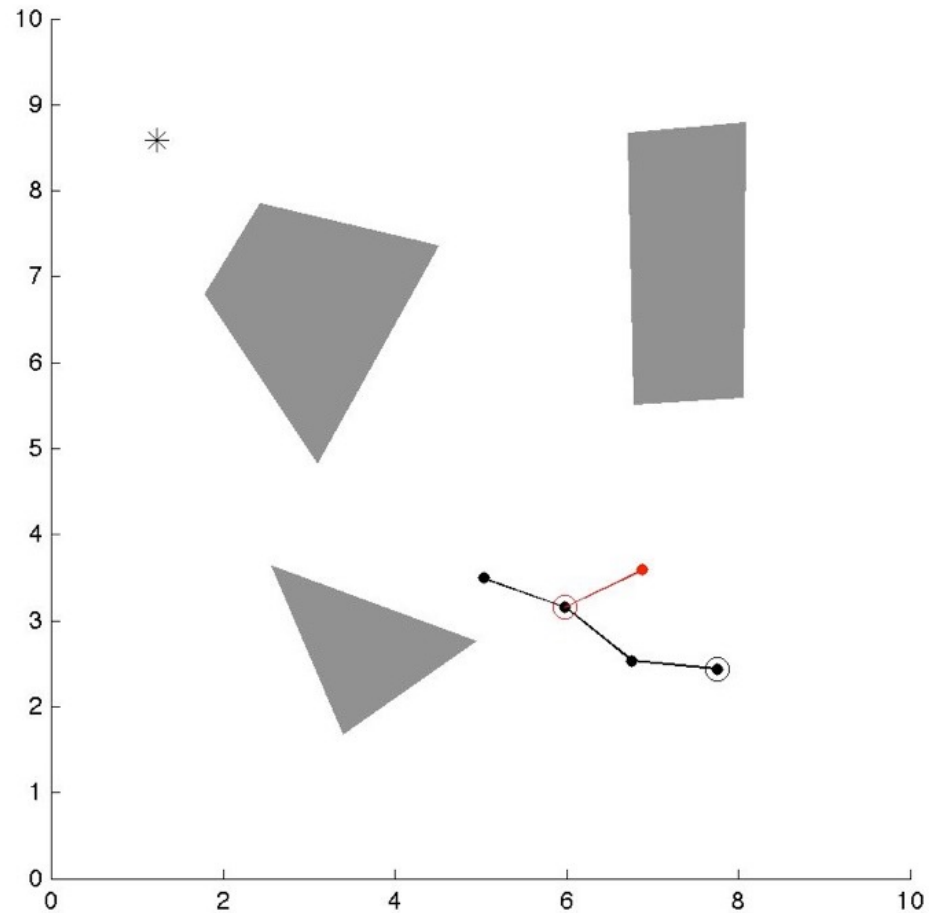
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

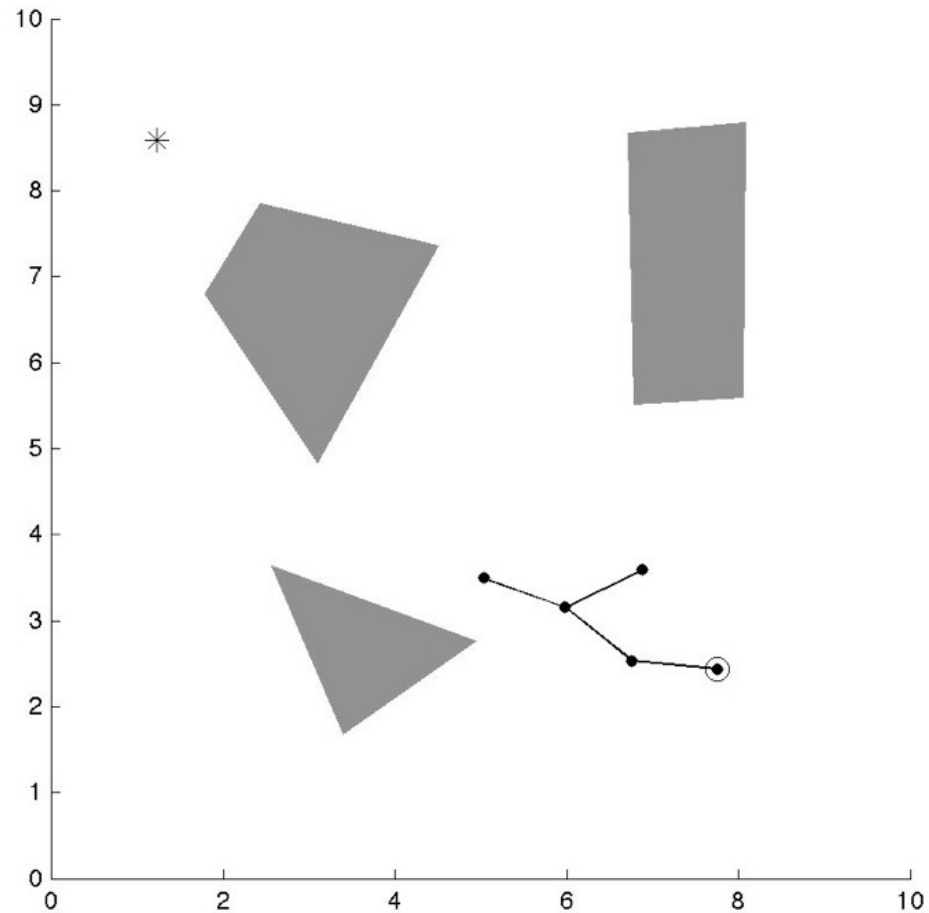
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

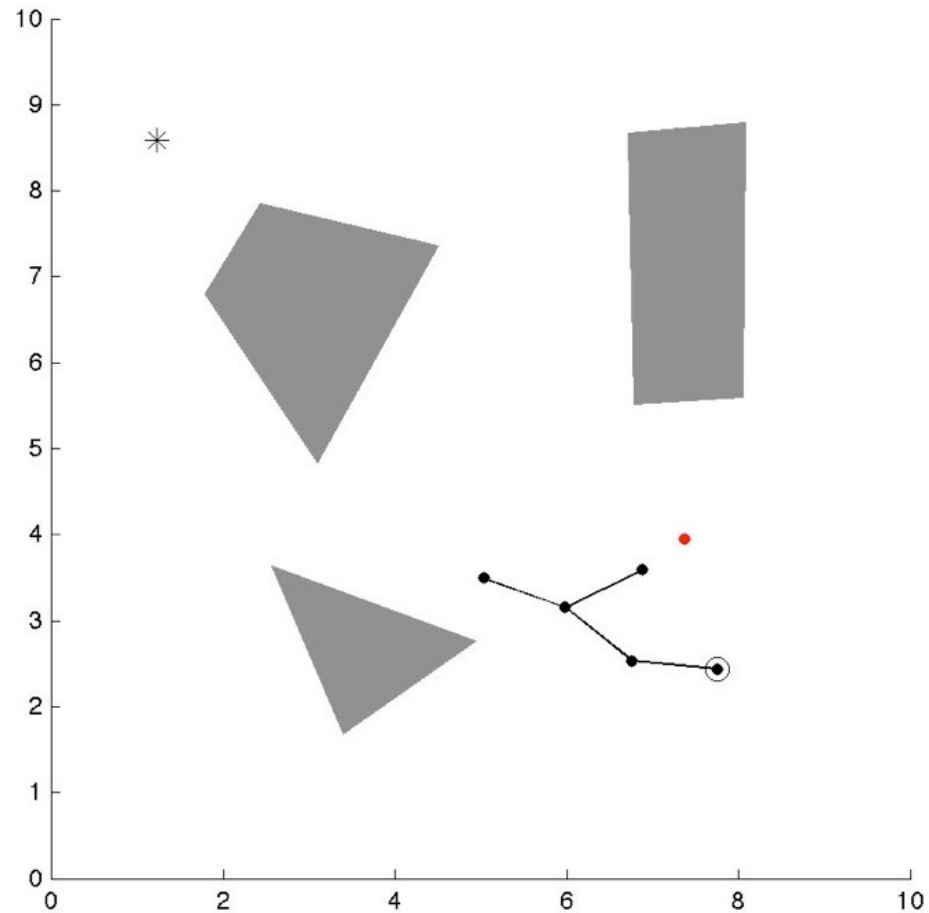
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

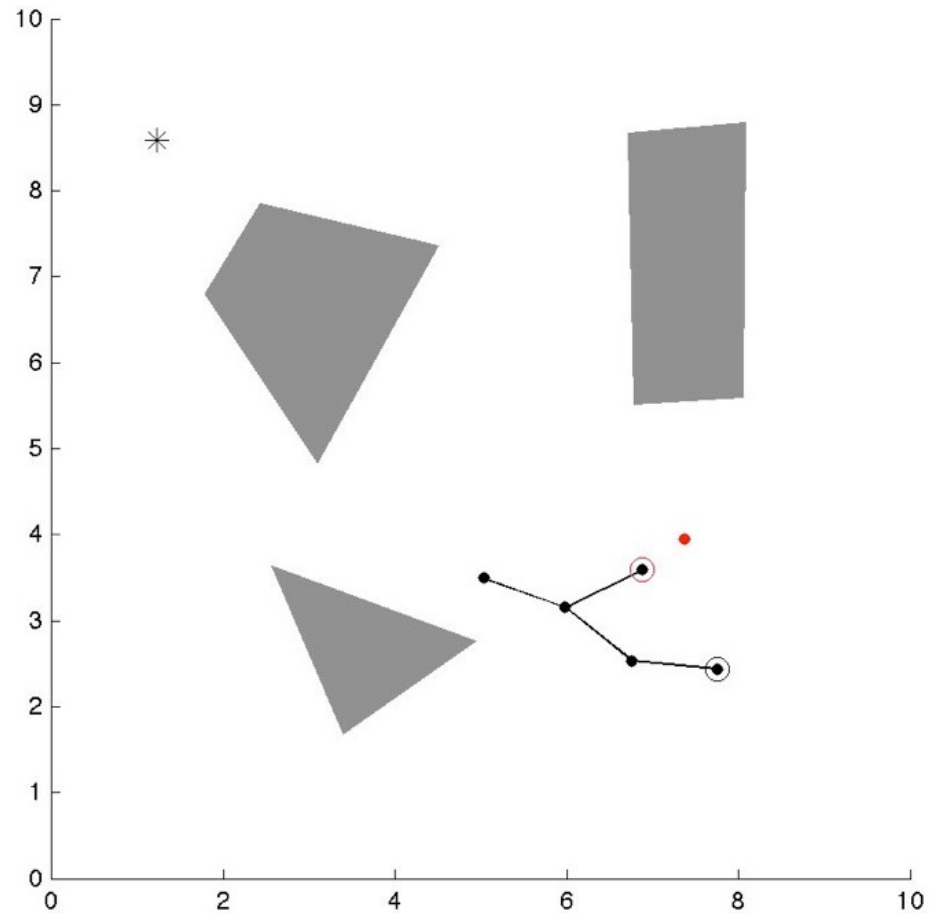
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

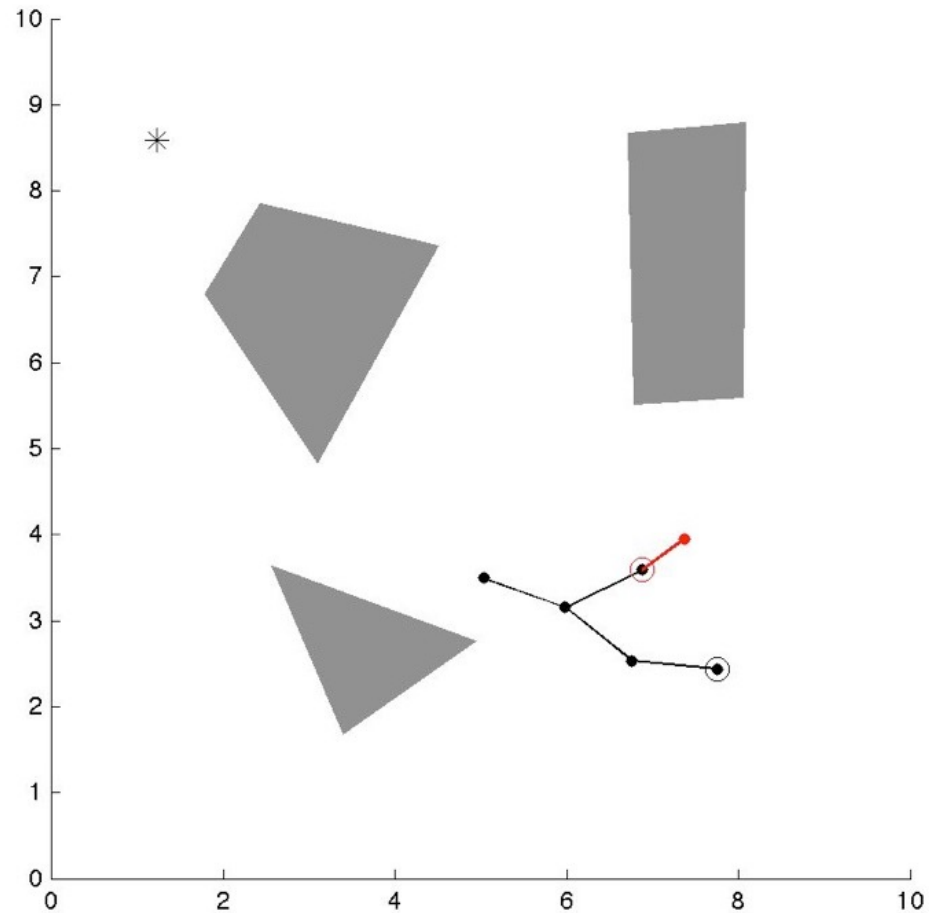
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

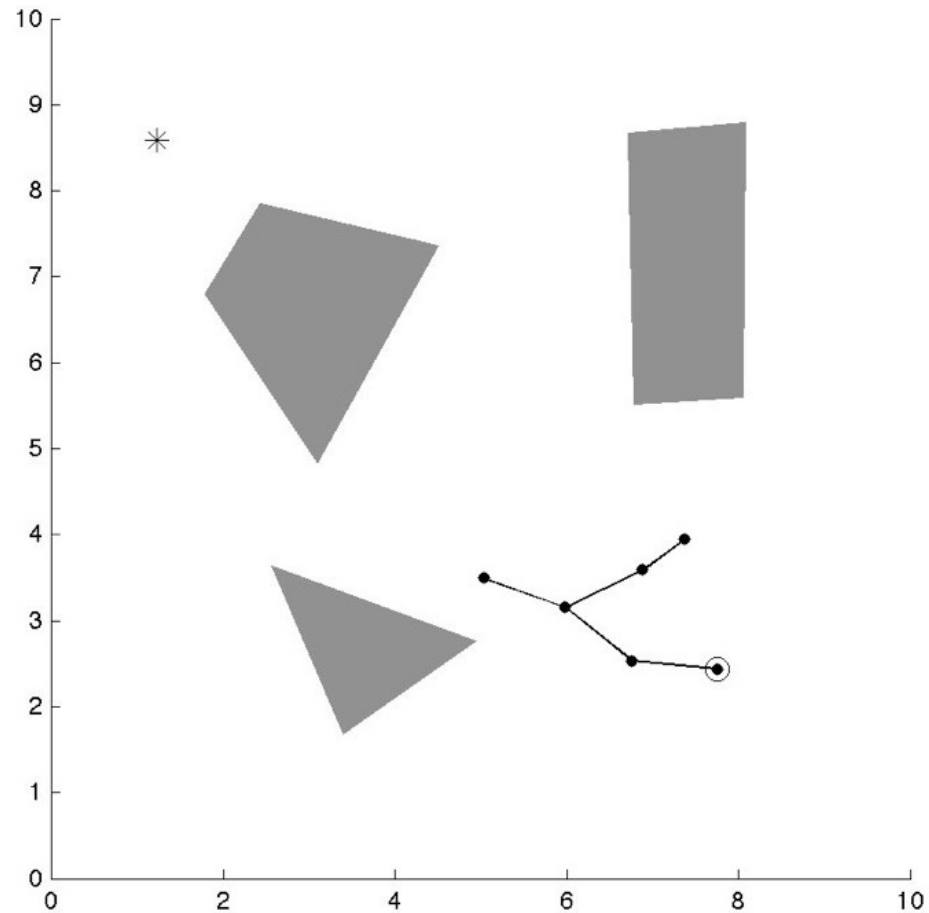
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

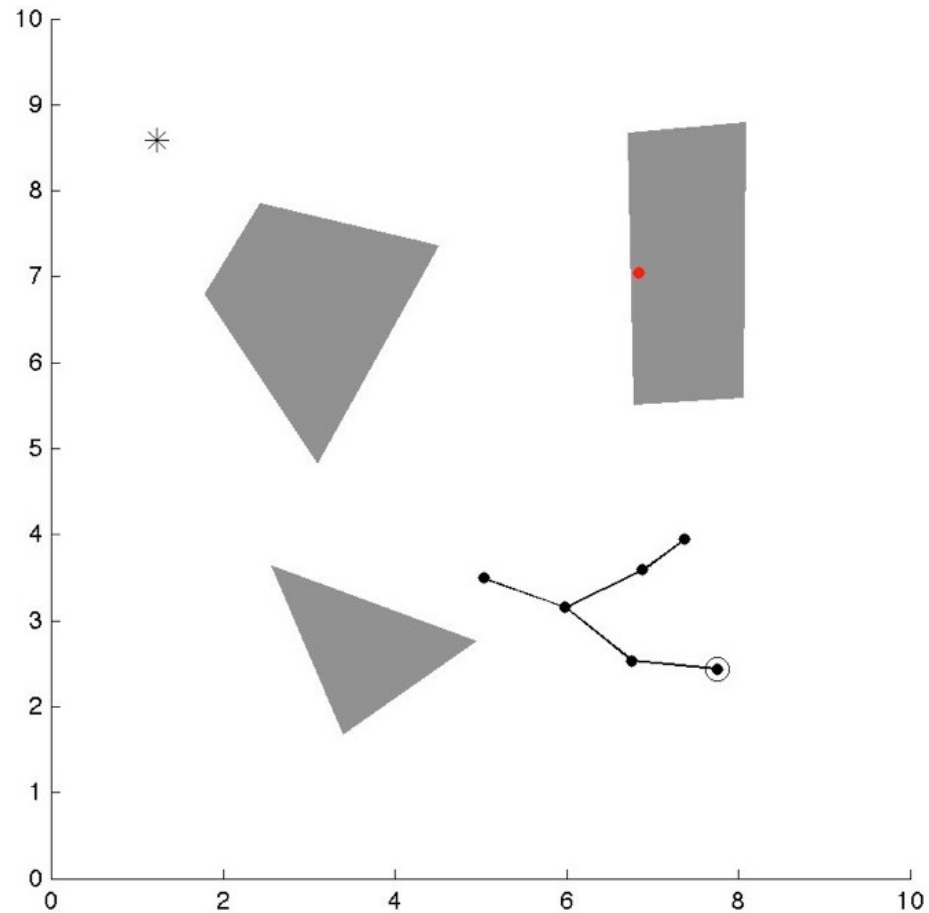
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

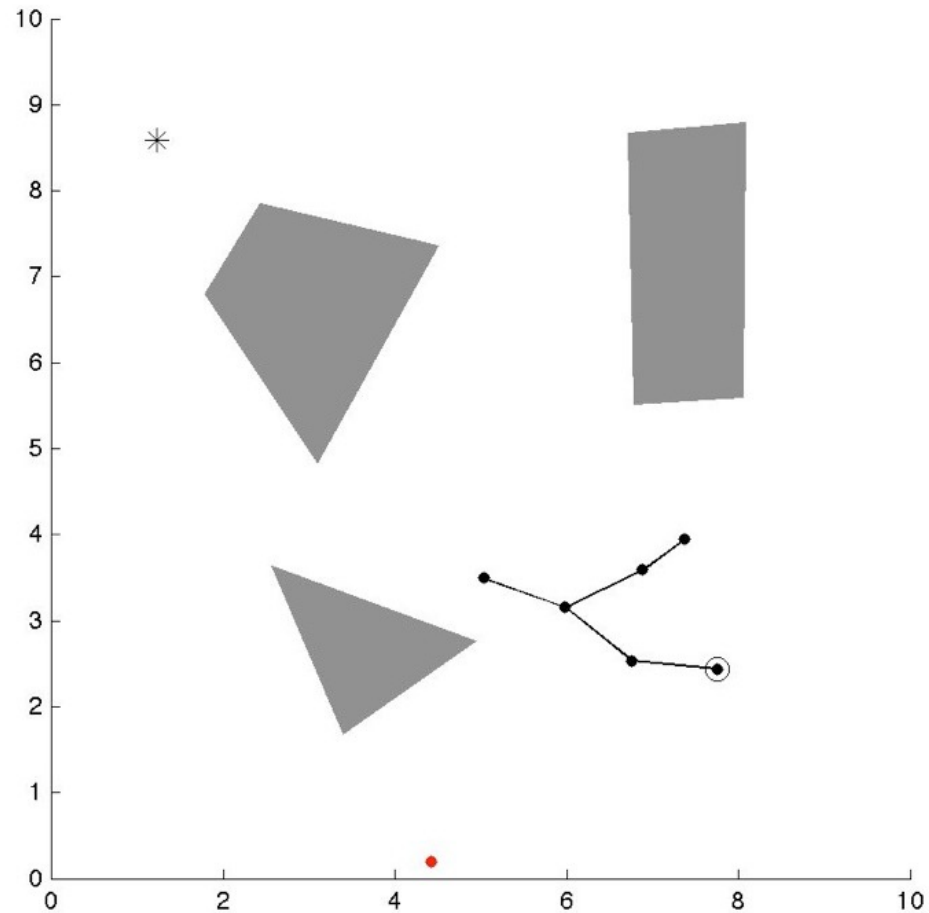
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

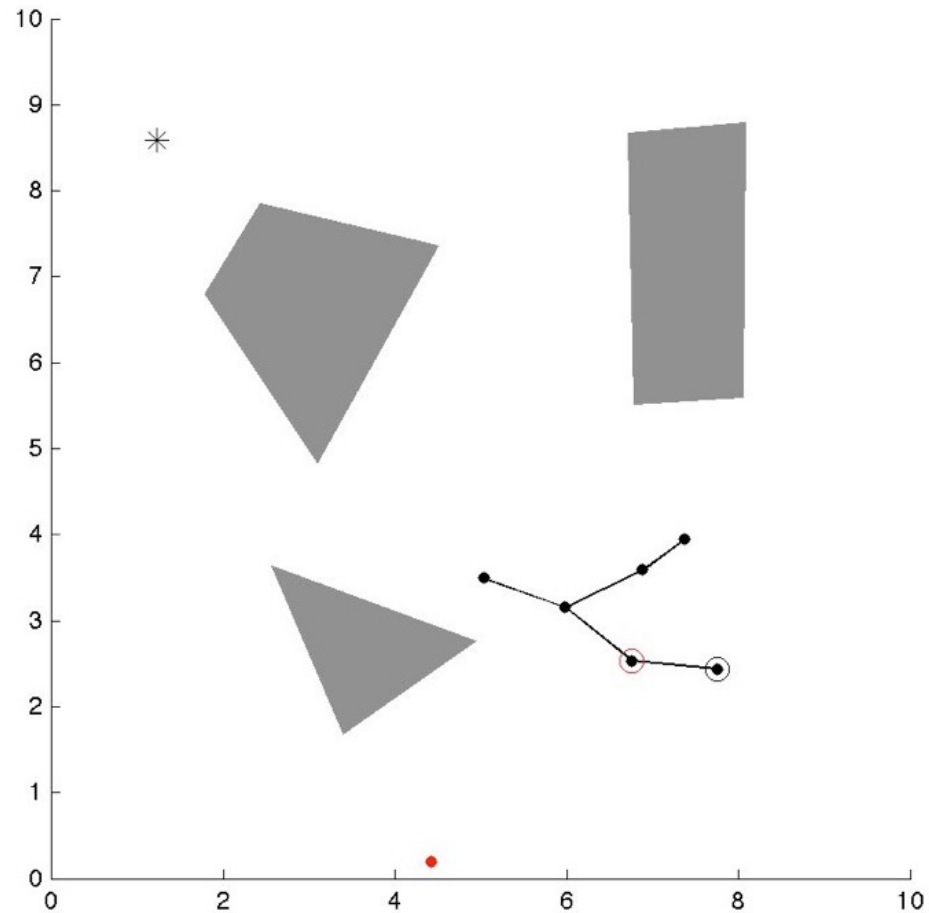
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

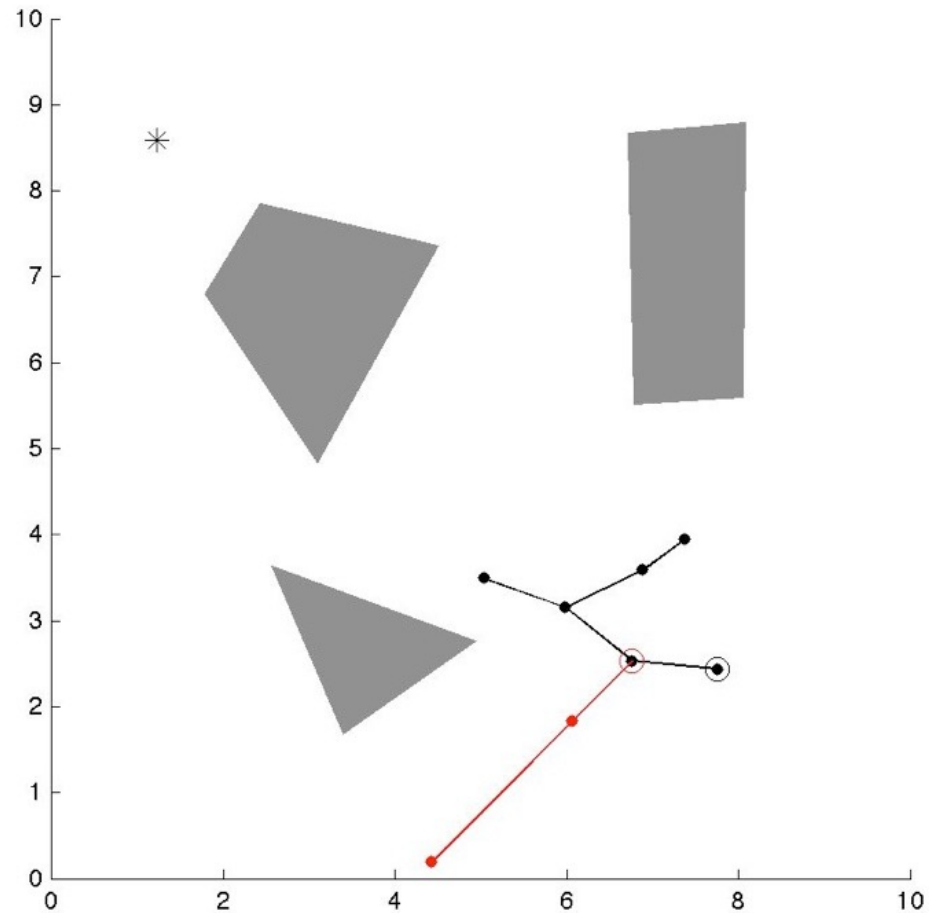
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

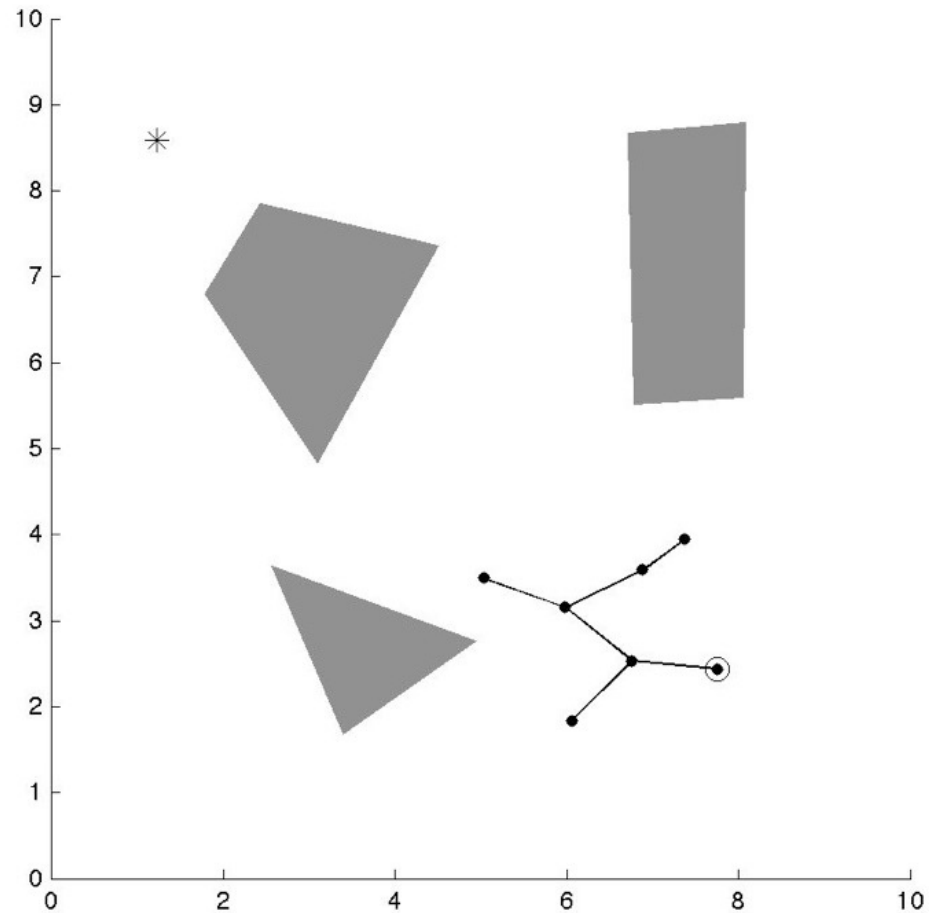
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

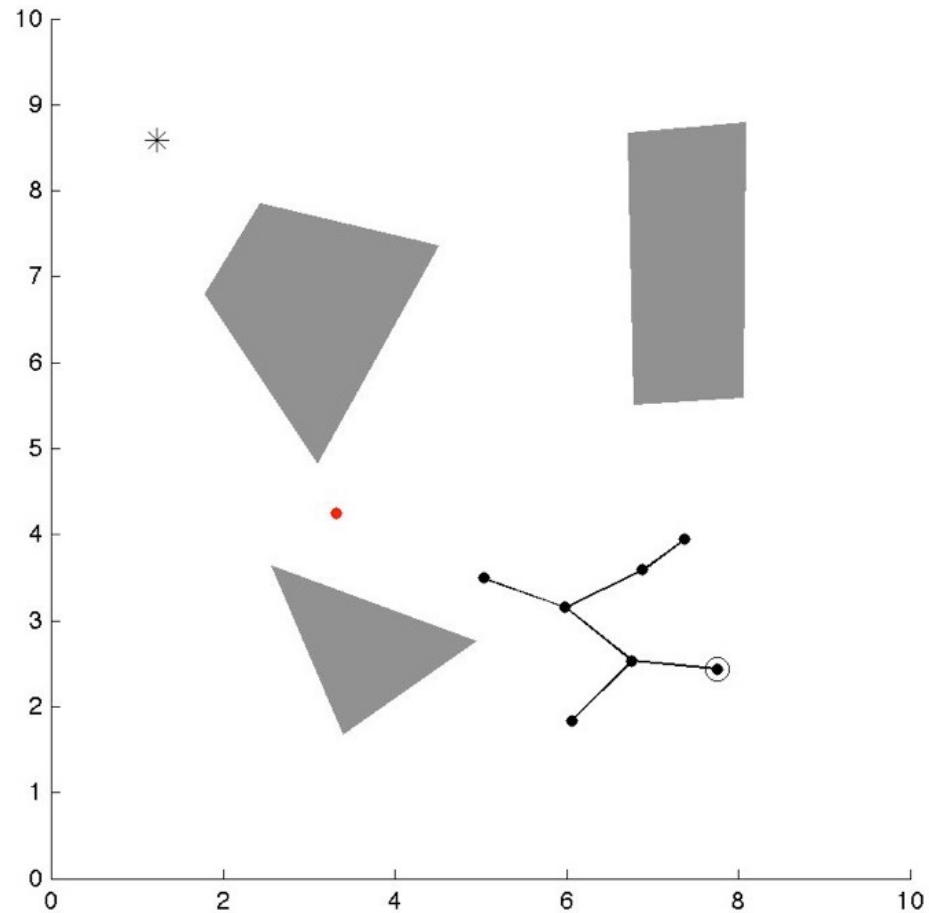
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

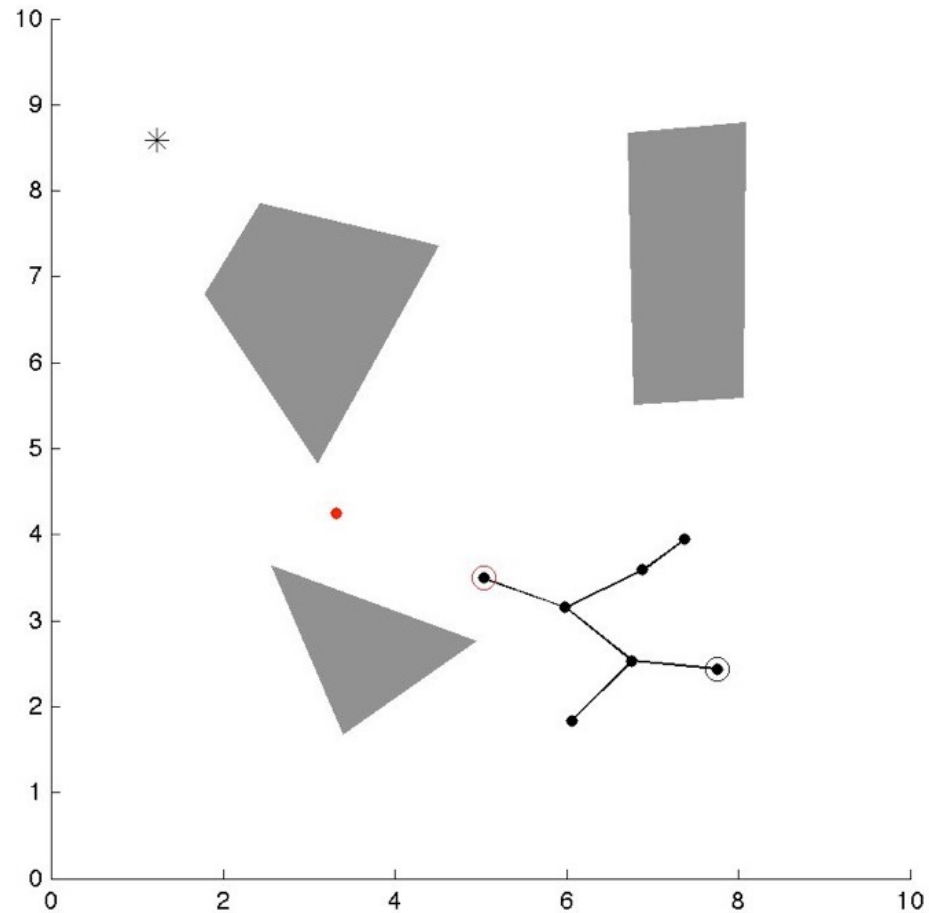
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

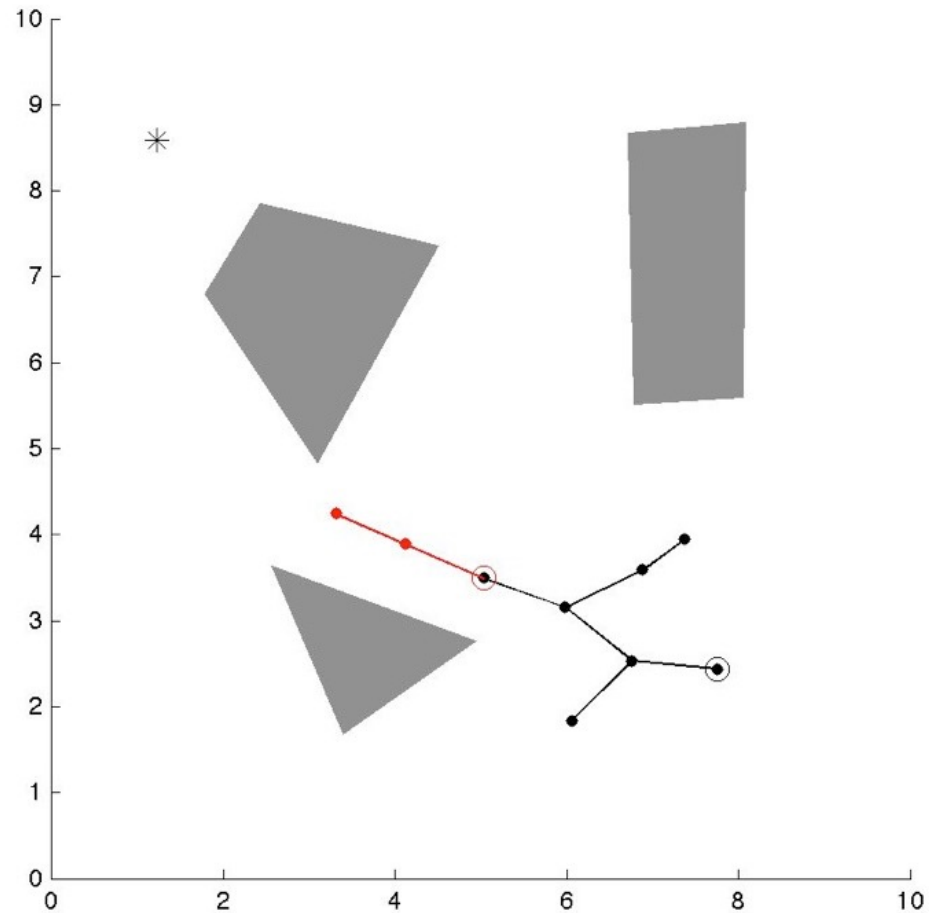
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

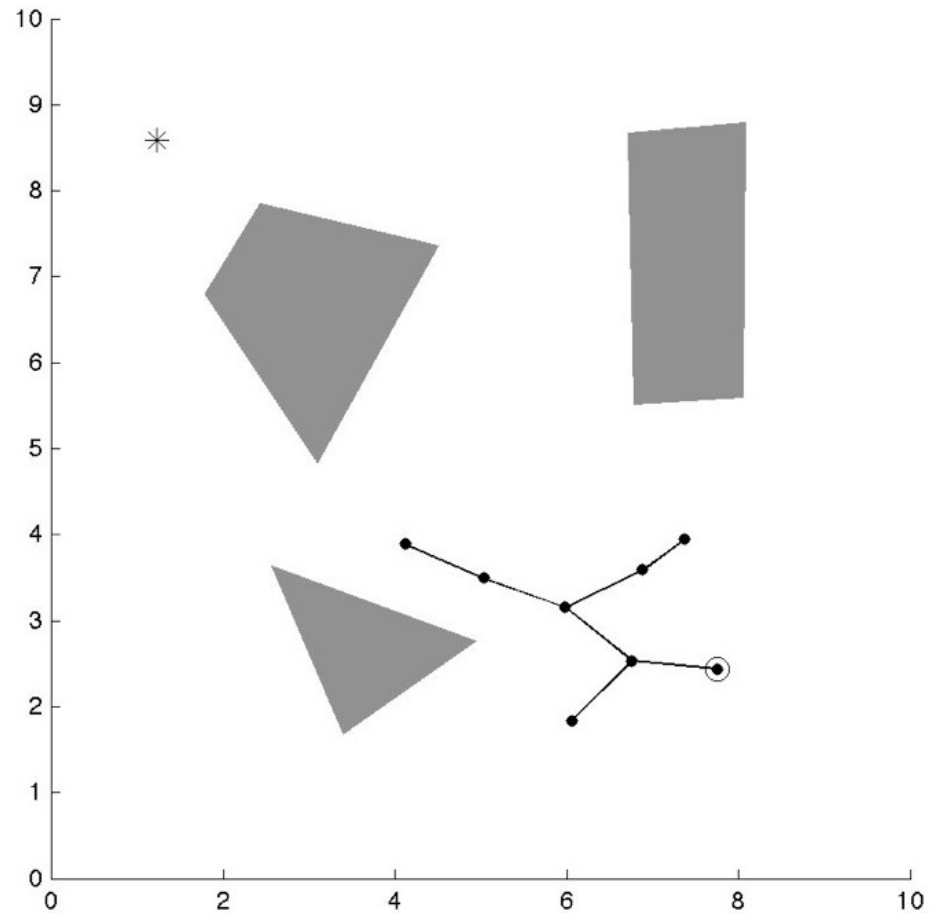
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

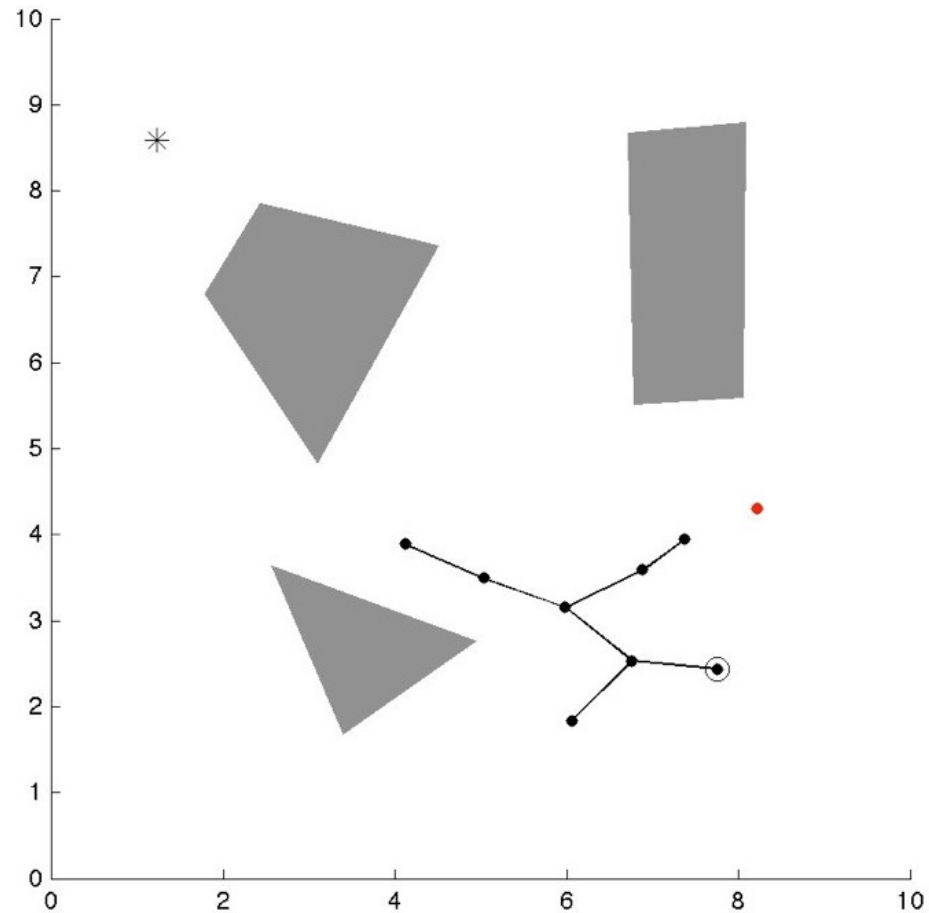
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

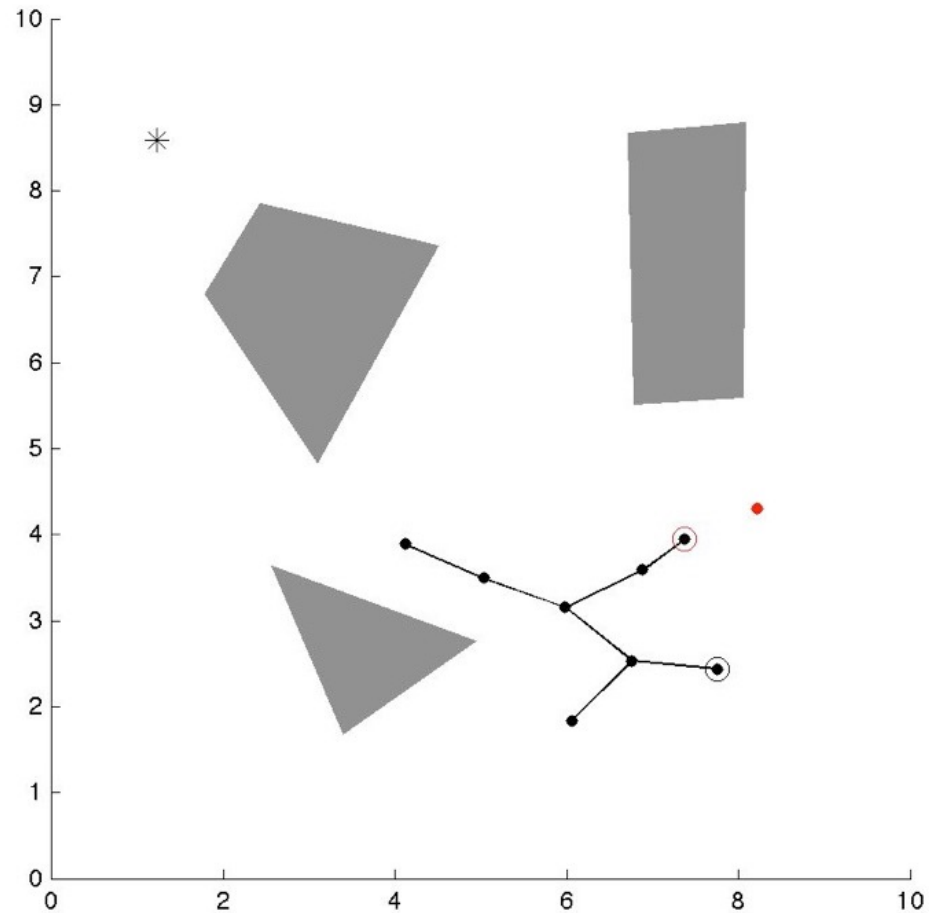
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

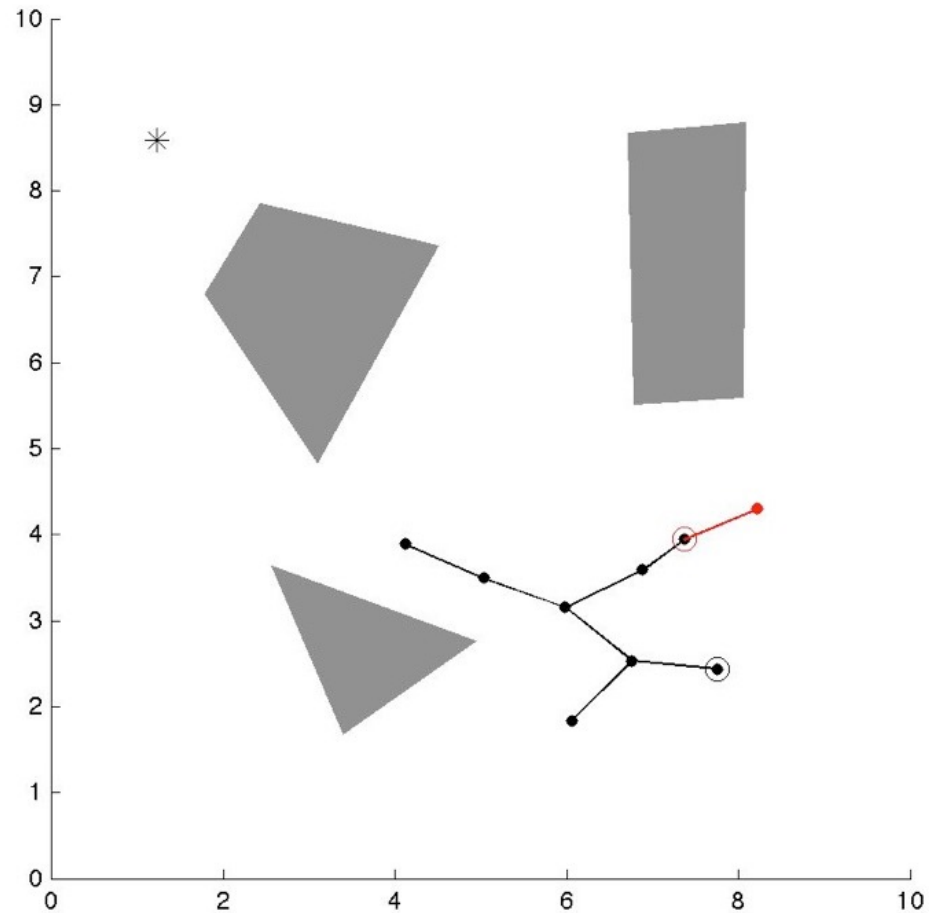
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

$G \leftarrow (V, E)$

$x_{rand} \leftarrow RandomSample()$

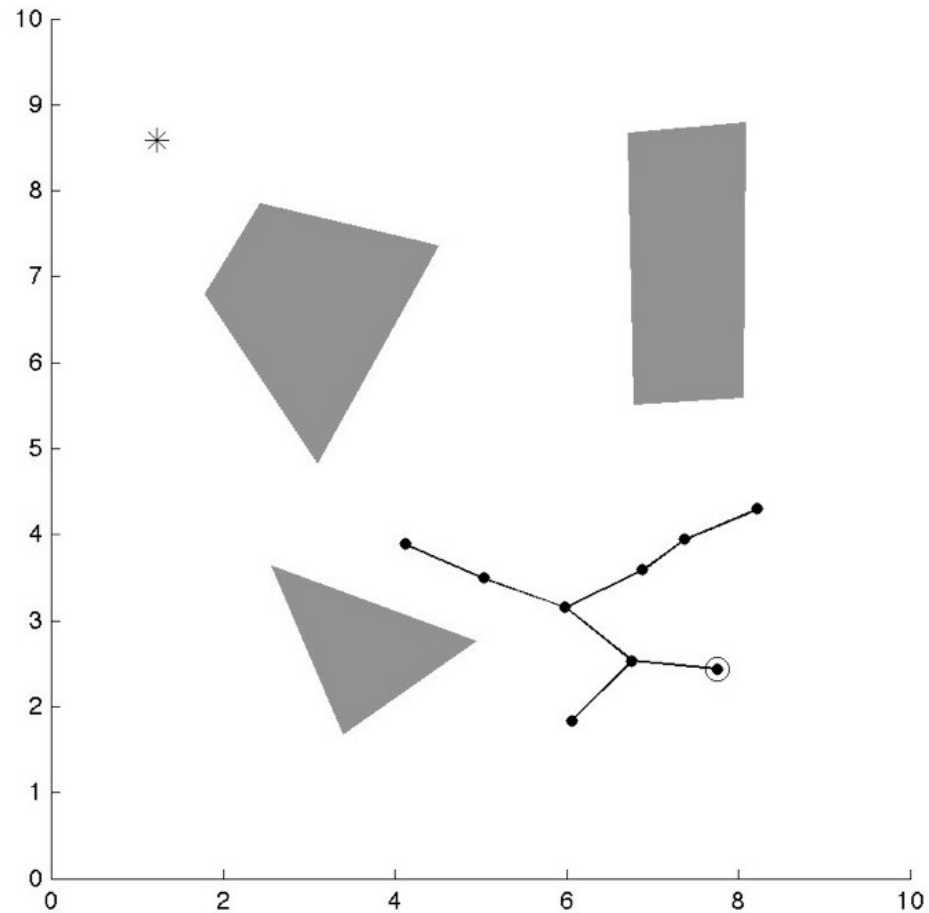
$x_{nearest} \leftarrow Nearest(G, x_{rand})$

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

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

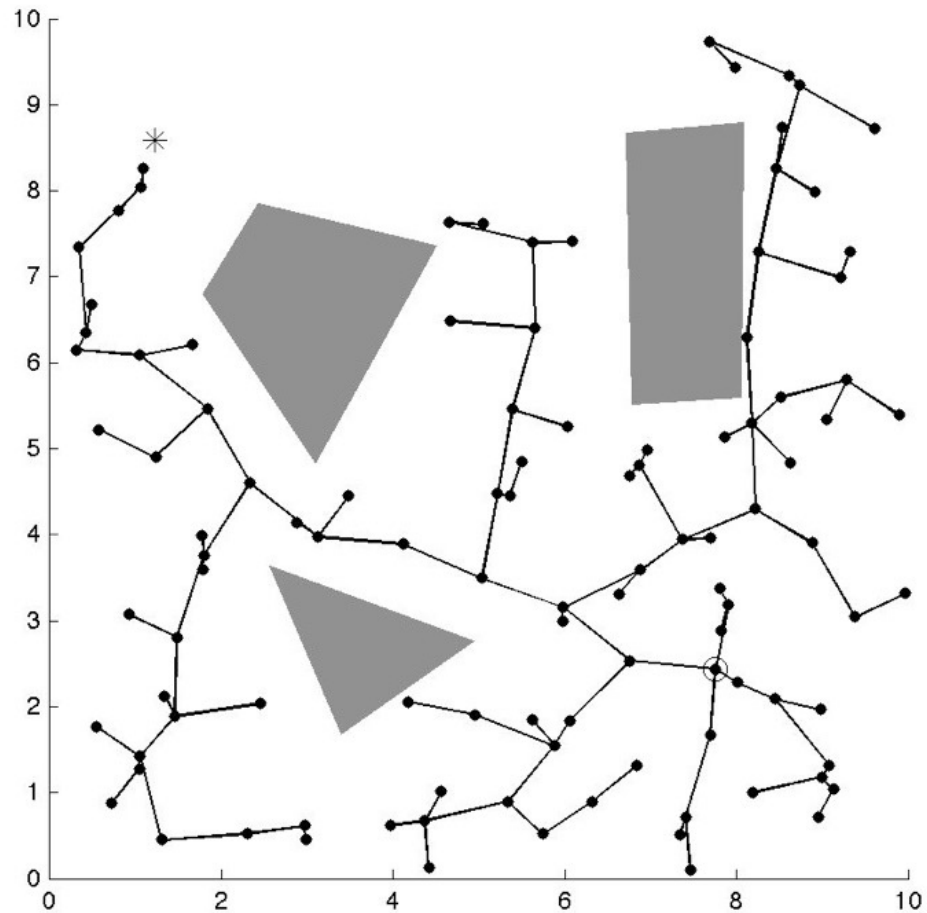
$V \leftarrow V \cup \{x_{new}\}$

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



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

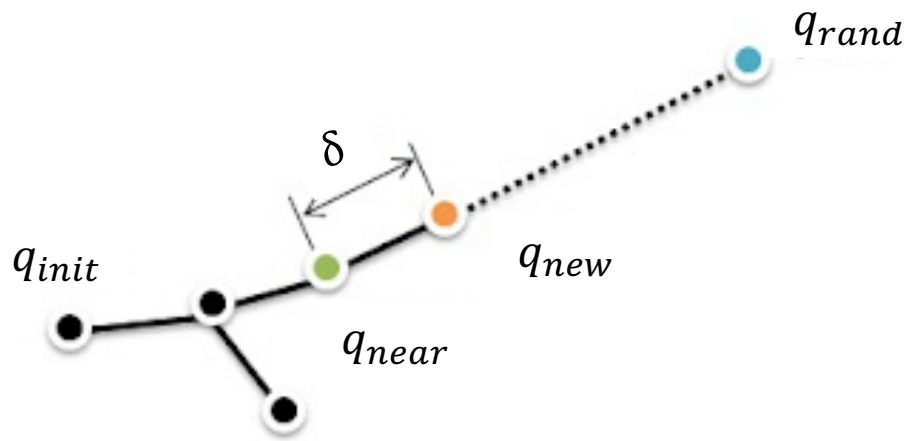


RRT - Bias to Goal

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

RRT - pseudocode

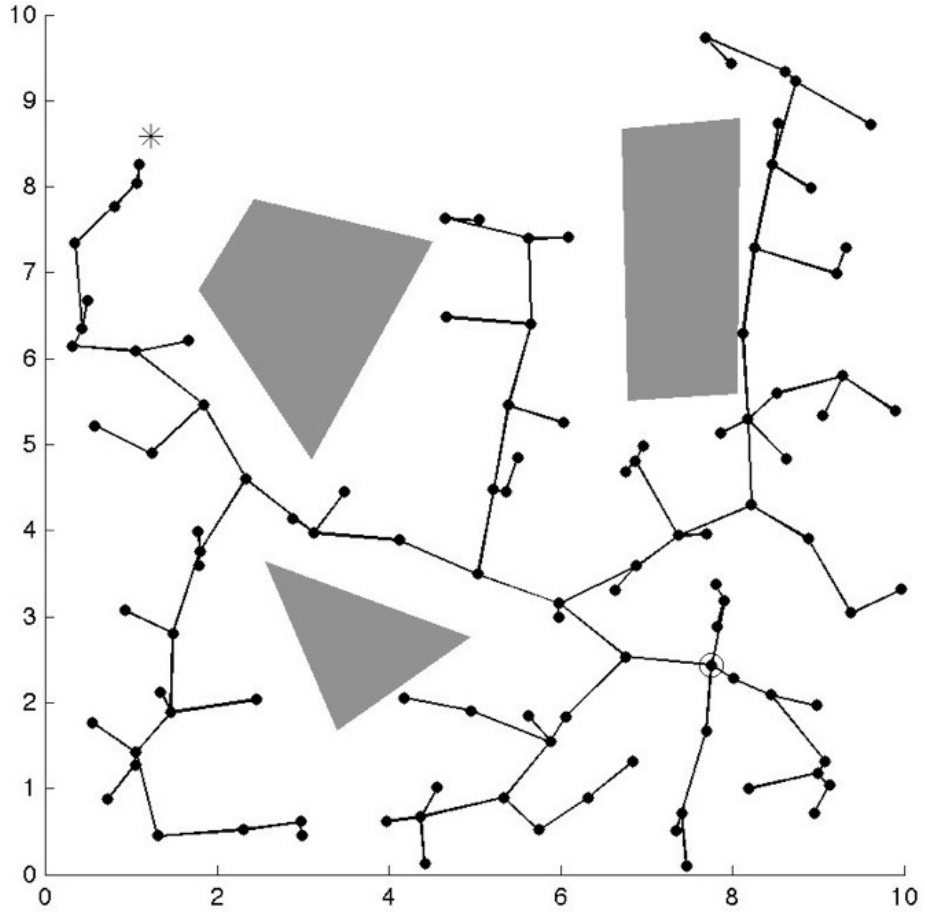
```
RRT_BUILD(q_init, q_goal)
        T.initialize(q_init)
        repeat N times:
                q_rand = RANDOM_CONFIGURATION
                RRT_EXPAND(T, q_rand)
        RETURN(T)

RRT_EXPAND(T, q_rand)
        q_near = NEAREST_NEIGHBOR(T, q_rand)
        if |q_near − q_rand| < δ then
                q_new = q_rand
        else
                q_new = q_near + δ (in the direction of q_rand)
        T.vertex_add(q_new)
        T.edge_add(q_near, q_new)
        if |q_goal − q_new| < ϵ then
                goal reached, return path
```



Finding the path

Once the RRT reaches the goal, you've found the solution.

- Backtrack along tree from *goal* to *start* to identify edges that connect the two
- [Optimize/smooth the path]
- Drive robot along the path



Rapidly-Exploring Random Tree (RRT)

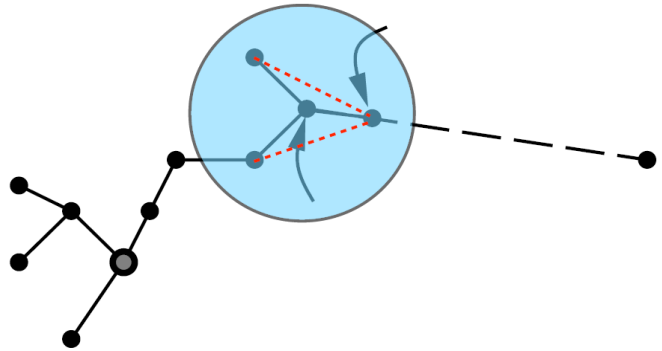
- Advantages: very fast, works well for dynamic environments

- Disadvantages: not optimal
  - in fact, it has been proven by Karaman & Frazzoli (2011) that the probability of RRT converging to an optimal solution is nearly 0

Paper with info: Sampling-based Algorithms for Optimal Motion Planning

Variants of RRT

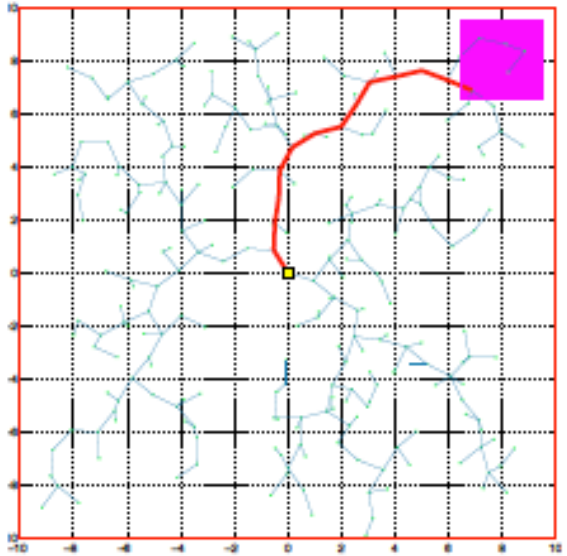
- There are (very) many…



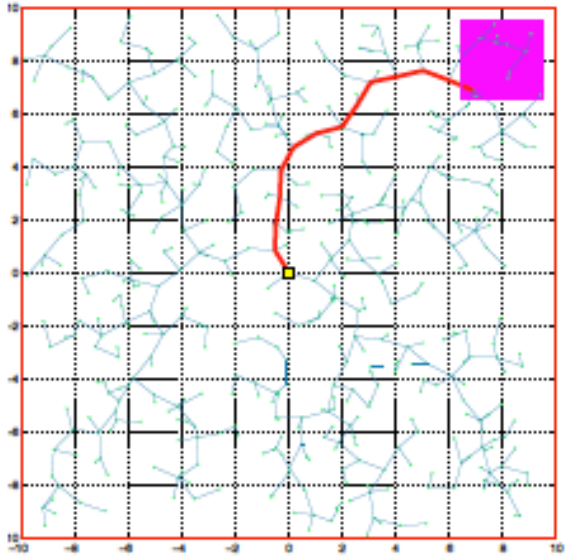
- Rapidly-exploring Random Graph (RRG):
  - Connect all vertices within neighboring region, forming a graph

- RRT*:
  - a variant of RRG that essentially “rewires" the tree as better paths are discovered.

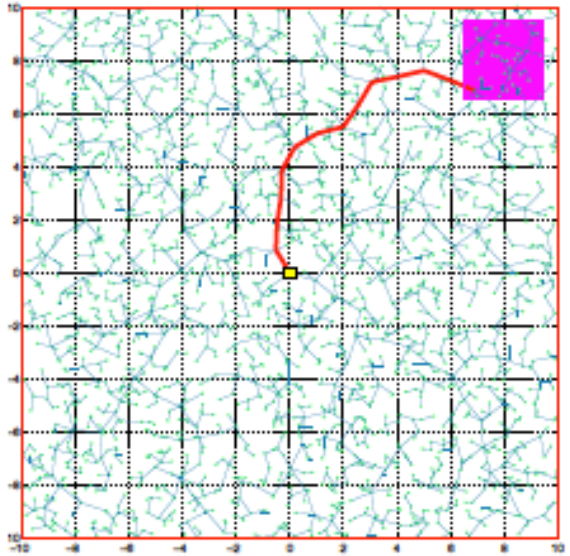
RRT



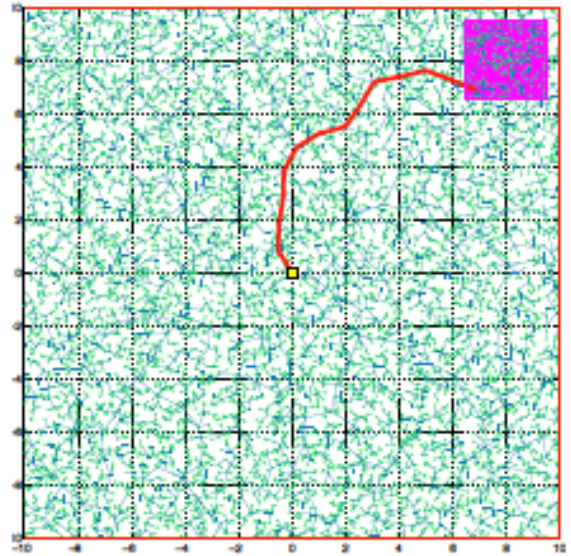
(a)



(b)

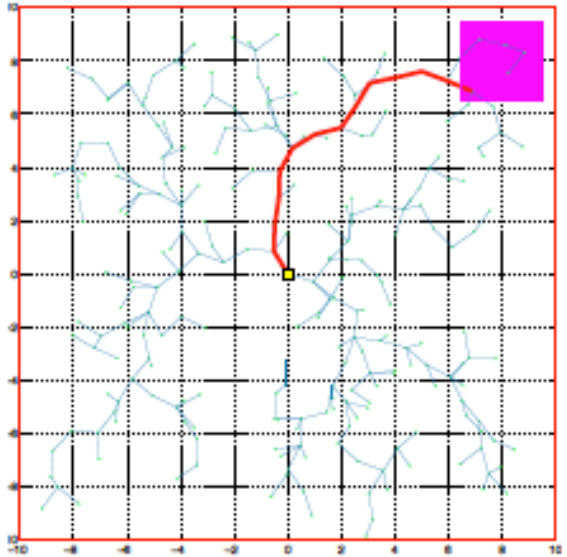


(c)

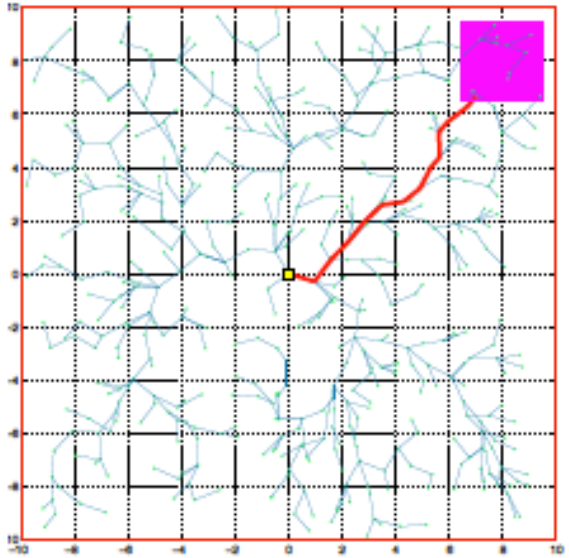


(d)

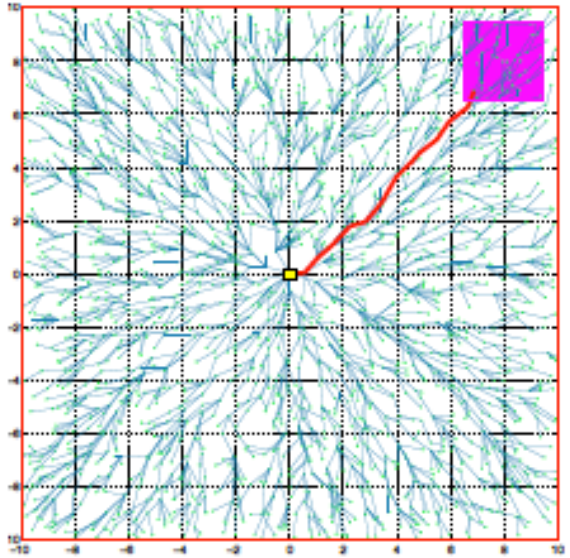
RRT*



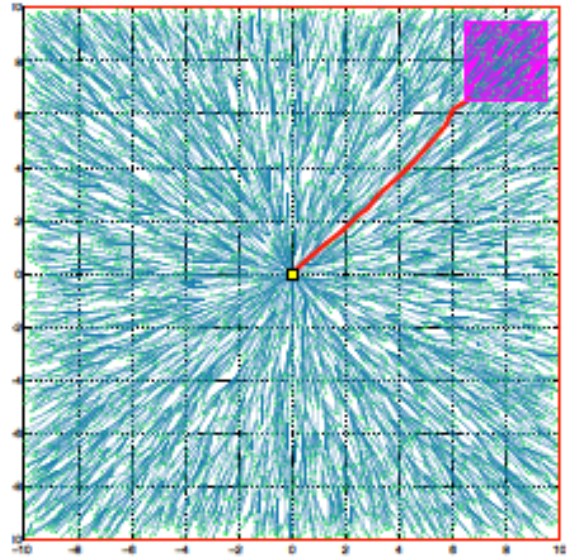
(e)



(f)

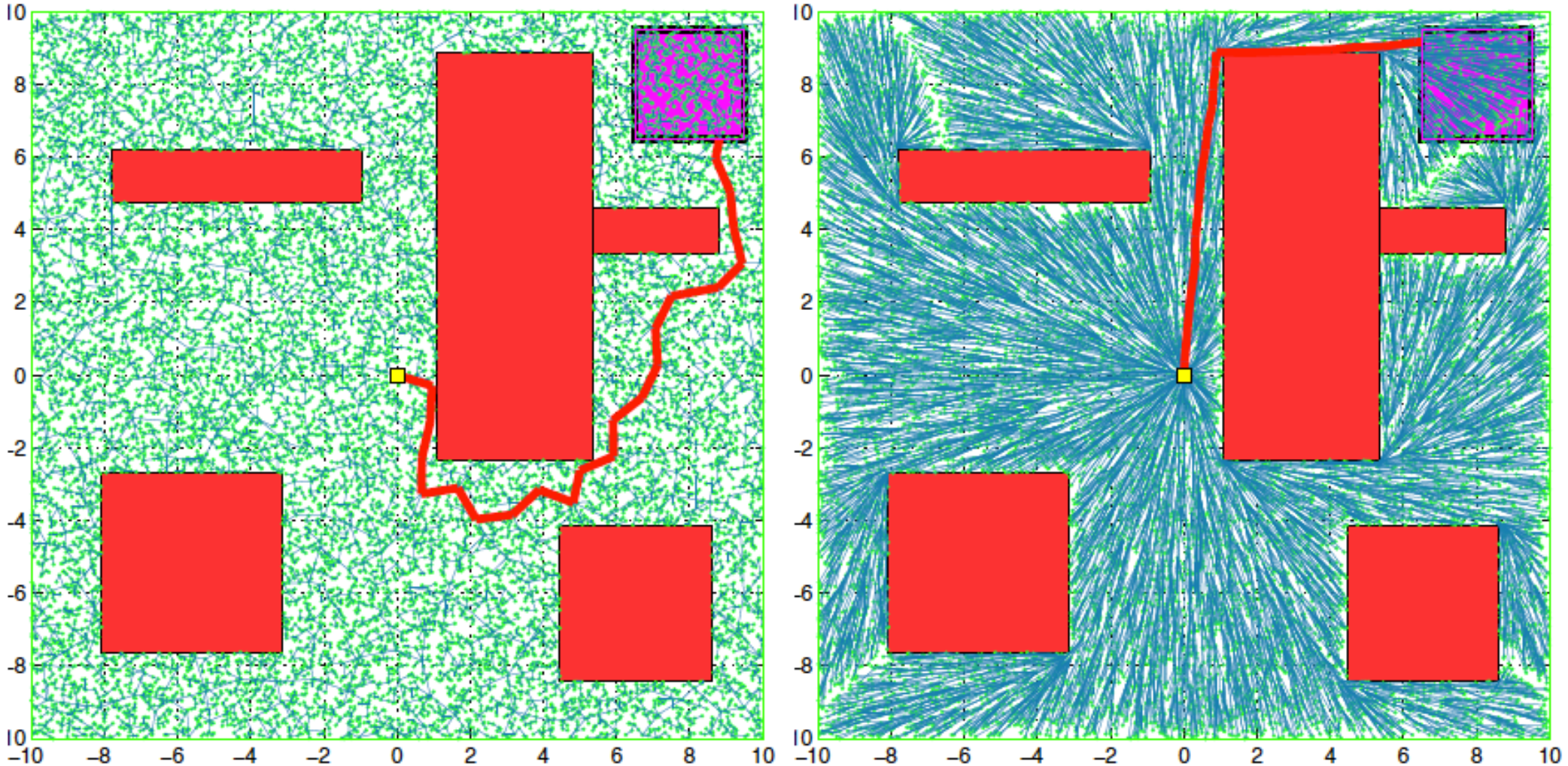


(g)



(h)

RRT vs RRT*

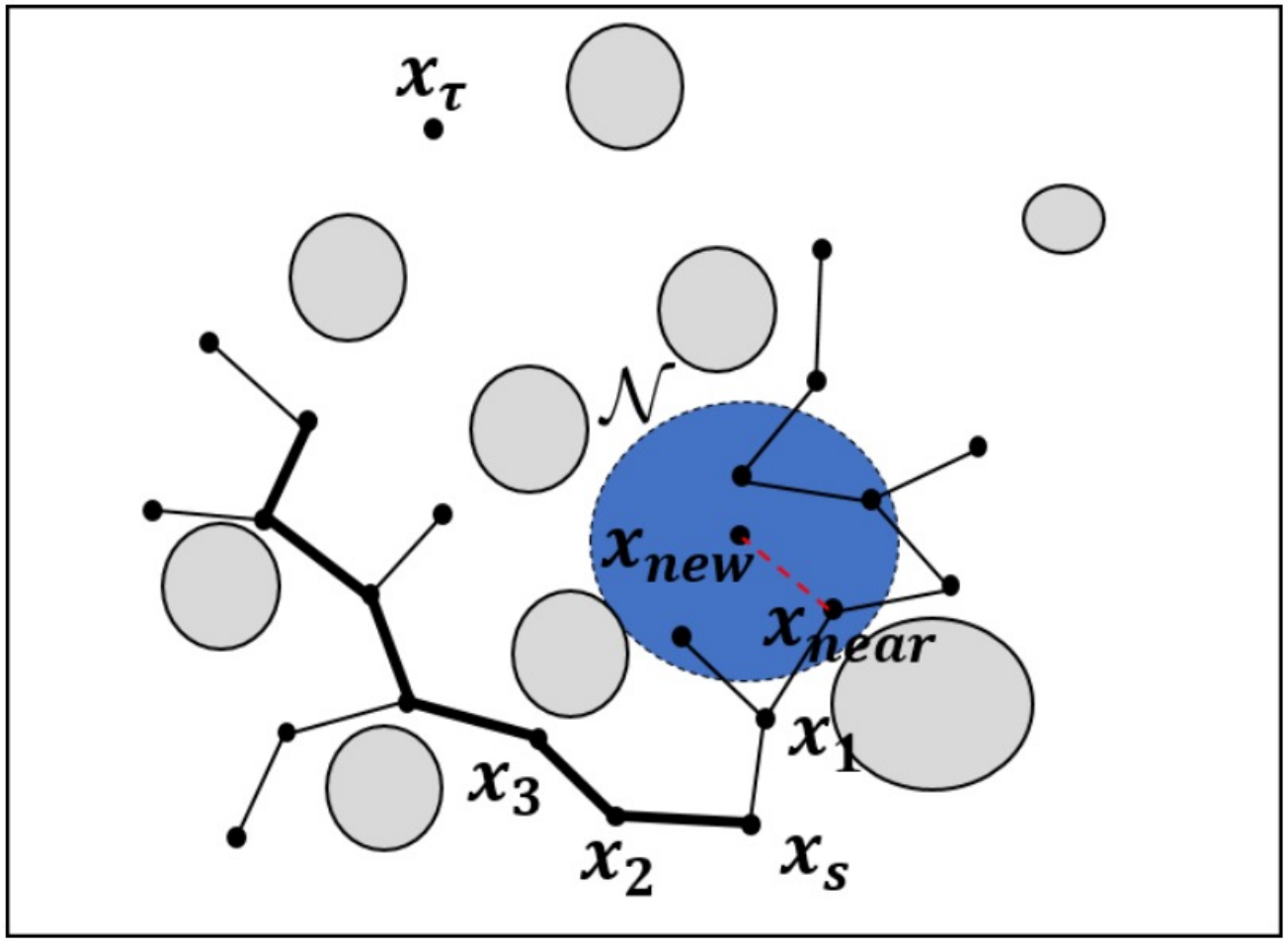


How exactly does RRT* reroute?

- **Key idea:** check neighborhood of new node
- Modification 1: Extend step
- Modification 2: Rewire Step

RRT*: Modification 1 – Extend step

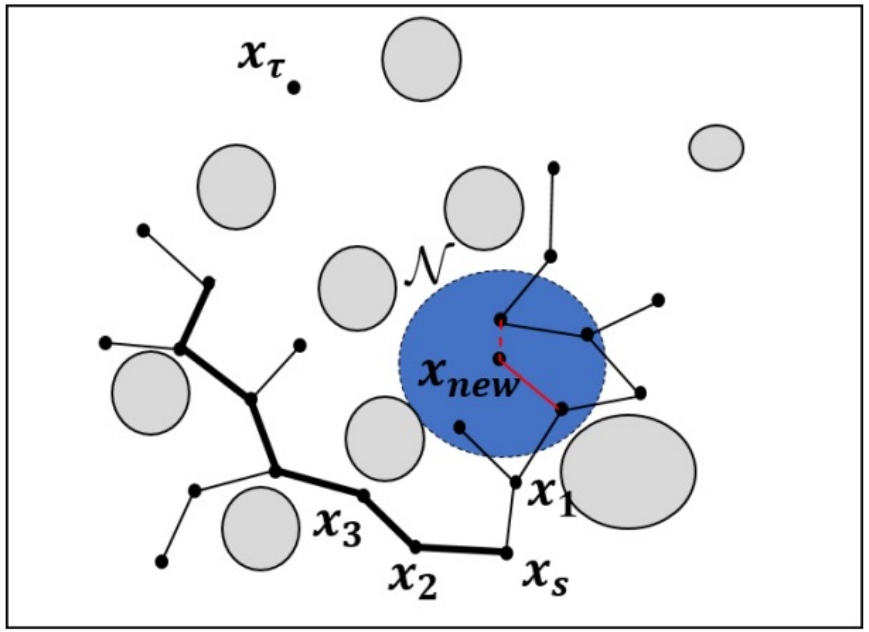
- Check neighborhood BEFORE adding points
- Connect new node to a parent that will result in **shorter path to start**
- Creates “fan” like structures

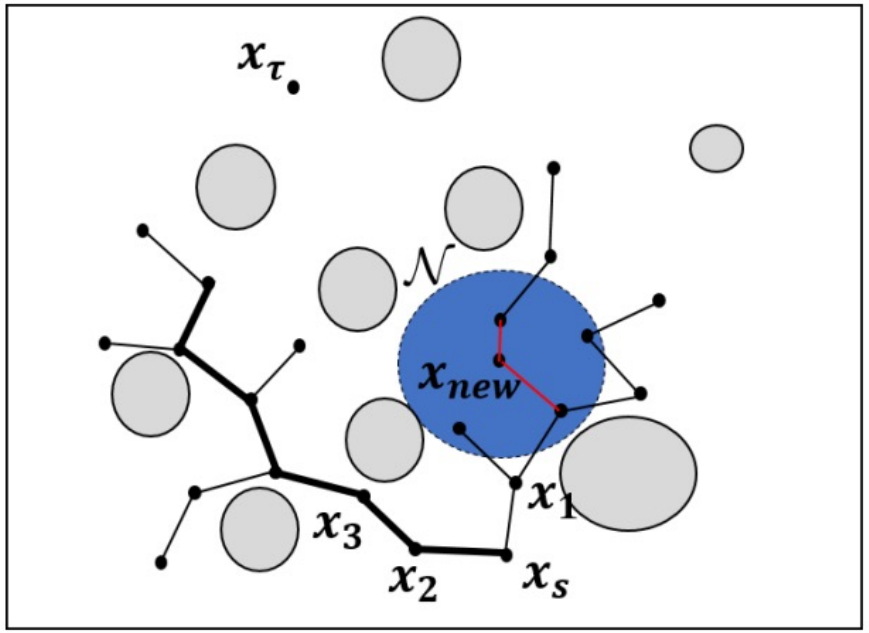


Credit: Nikolay Atanasov

RRT*: Modification 2 – Rewire Step

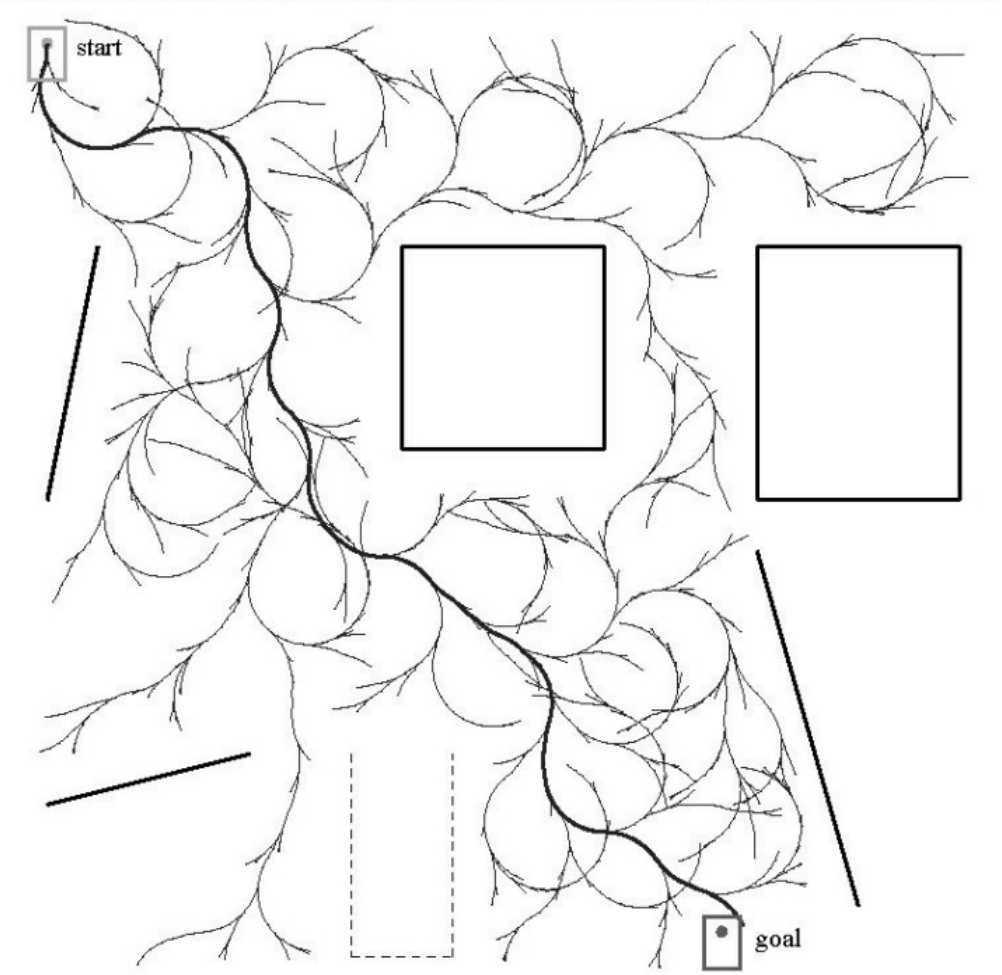
- Check neighborhood AFTER adding points
- If **any neighbor's** path to the start node will decrease, then reroute them through the new node
- Makes paths "smoother" and shorter



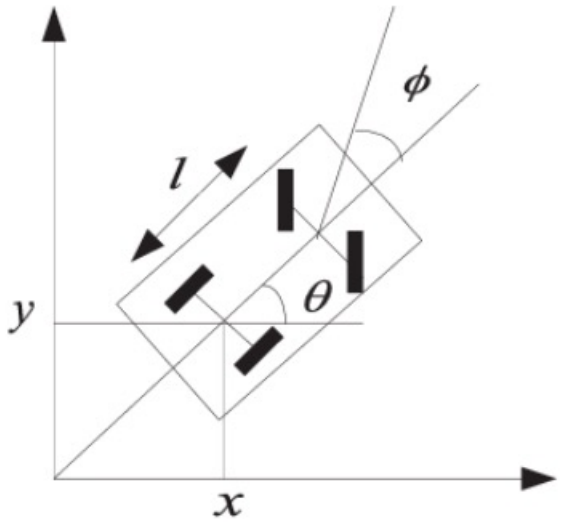


Credit: Nikolay Atanasov

RRTs with Kinematic constraints



Example: holonomic constraints



$$\dot{x} = v\cos\theta$$
$$\dot{y} = v\sin\theta$$
$$\dot{\theta} = \frac{v}{l}\tan\phi$$

RRTs with Kinematic constraints

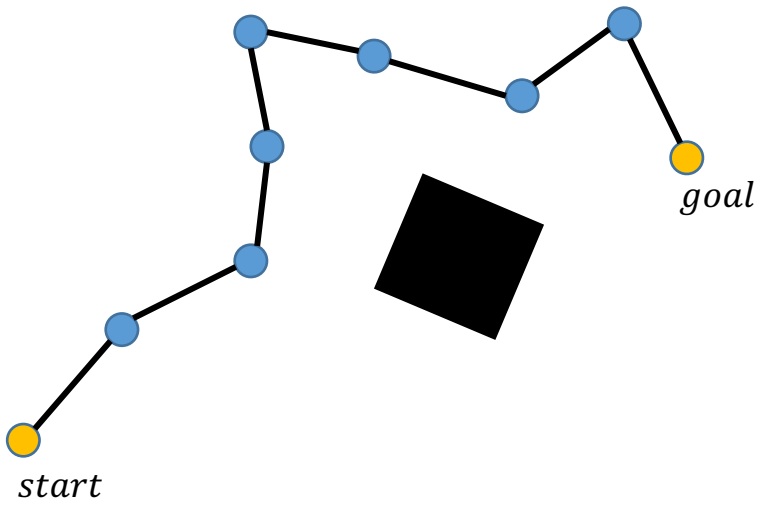
```
RRT_BUILD(q_init, q_goal)
        T.initialize(q_init)
        repeat N times:
                q_rand = RANDOM_CONFIGURATION
                RRT_EXPAND(T, q_rand)
        RETURN(T)
RRT_EXPAND(T, q_rand)
        q_near = NEAREST_NEIGHBOR(T, q_rand)
        if |q_near − q_rand| < δ then
                q_new = q_rand
        else
                q_new = q_near + δ
        T.vertex_add(q_new)
        T.edge_add(q_near, q_new)
        if |q_goal − q_new| < ϵ then
                goal reached, return path
```

```
RRT_BUILD(q_init, q_goal)
        T.initialize(q_init)
        repeat N times:
                q_rand = RANDOM_CONFIGURATION
                RRT_EXPAND(T, q_rand)
        RETURN(T)

RRT_EXPAND(T, q_rand)
        q_near = NEAREST_NEIGHBOR(T, q_rand)
        u = select_input(q_rand, q_near)
        q_new = new state(q_near, u, Δt)
        T.vertex_add(q_new)
        T.edge_add(q_near, q_new)
        if |q_goal − q_new| < ϵ then
                goal reached, return path
```

Restrict allowable paths!

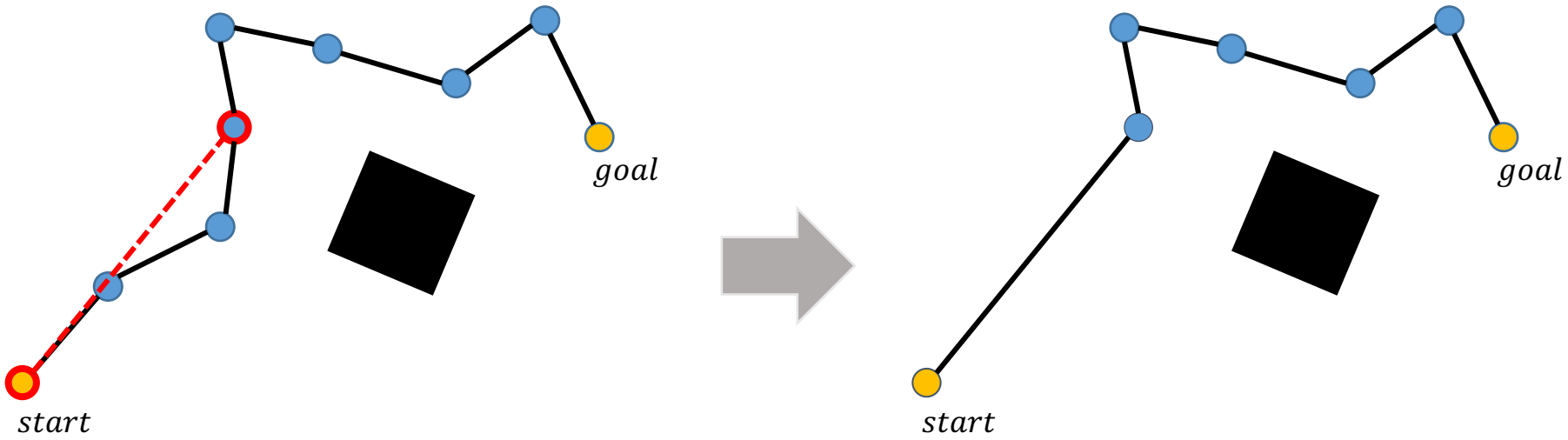
Optimizing the path

- Paths generated by sampling-based planning are far from optimal and require additional refinement before they are usable
- A typical solution can look like this:



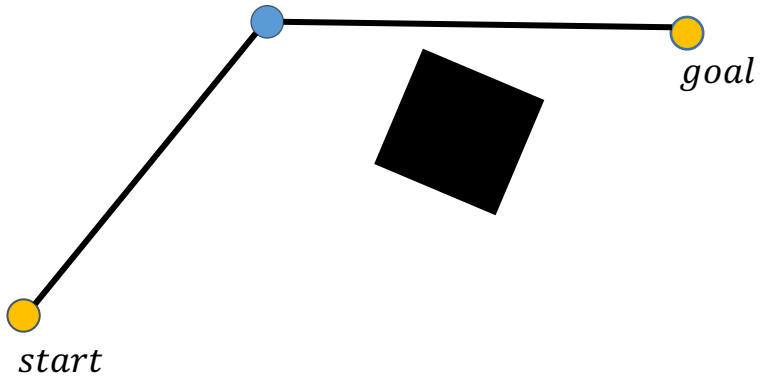
Optimizing the path

- 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