Motion planning

Florent Lamiraux

CNRS-LAAS, Toulouse, France

Motion planning

Context

Industrial robots

aerial robots

autonomous vehicles

Mobile autonomous system

- moving in an environment cluttered with obstacles
- subject to kinematic or dynamic constraints

Motion planning : automatically computing a feasible trajectory between two configurations.

Context

Industrial robots

aerial robots

autonomous vehicles

Mobile autonomous system

- moving in an environment cluttered with obstacles
- subject to kinematic or dynamic constraints

Motion planning : automatically computing a feasible trajectory between two configurations.

Context

Industrial robots

aerial robots

autonomous vehicles

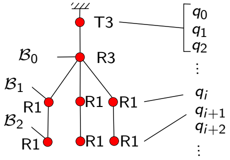
Mobile autonomous system

- moving in an environment cluttered with obstacles
- subject to kinematic or dynamic constraints

Motion planning : automatically computing a feasible trajectory between two configurations.

Robot

Set of rigid bodies $\mathcal{B}_0, \cdots \mathcal{B}_m$, linked to one another by *joints*.



Joint : parameterized rigid-body transformation between two frames (in SE(3)).

Rigid body transformation

Definitions

- SO(3) : group of 3 by 3 rotation matrices.

$$R \in SO(3) \Leftrightarrow R^T R = I_3 \text{ and } \det(R) = 1$$

- SE(3) : group of rigid body tranformations

$$T \in SE(3) \Leftrightarrow \begin{array}{l} \exists t \in \mathbb{R}^3, \exists R \in SO(3) \\ \forall x \in \mathbb{R}^3 \;\; T(x) = Rx + t \end{array}$$

We denote $T = T_{(R,t)}$.

Joint

A joint is represented by a mapping from a sub-manifold of $\mathbb{R}^p$ in SE(3), where $p \geq 1$ is an integer.

Examples :

- Translation T1 :

$$\begin{array}{ccc} \mathbb{R} & \rightarrow & SE(3) \\ t & \rightarrow & T_{(I_3,(t\ 0\ 0))} \end{array} \qquad \text{translation along x}$$

Joint

A joint is represented by a mapping from a sub-manifold of $\mathbb{R}^p$ in SE(3), where $p \geq 1$ is an integer.
Examples :

- Translation T3 :

$$\begin{array}{ccc} \mathbb{R}^3 & \rightarrow & SE(3) \\ t & \rightarrow & T_{(I_3,t)} \end{array} \qquad \text{translation}$$

Joint

A joint is represented by a mapping from a sub-manifold of $\mathbb{R}^p$ in SE(3), where $p \geq 1$ is an integer.
Examples :

- Rotation R1 :

$$\begin{array}{ccc} \mathbb{R} & \rightarrow & SE(3) \\ t & \rightarrow & T_{(R,0)} \end{array} \qquad R = \left(\begin{array}{ccc} \cos t & -\sin t & 0 \\ \sin t & \cos t & 0 \\ 0 & 0 & 1 \end{array}\right)$$

Joint

A joint is represented by a mapping from a sub-manifold of $\mathbb{R}^p$ in SE(3), where $p \geq 1$ is an integer.
Examples :

- Rotation R3 :

$$\begin{array}{ccc} \mathbb{R}^4 & \rightarrow & SE(3) \\ t & \rightarrow & T_{(R,0)} \end{array}$$

$$\|t\| = 1$$

$$R = \begin{pmatrix} 1-2(t_2^2+t_3^2) & 2t_2t_1 - 2t_3t_0 & 2t_3t_1 + 2t_2t_0 \\ 2t_2t_1 + 2t_3t_0 & 1-2(t_1^2+t_3^2) & 2t_3t_2 - 2t_1t_0 \\ 2t_3t_1 - 2t_2t_0 & 2t_3t_2 + 2t_1t_0 & 1-2(t_1^2+t_2^2) \end{pmatrix}$$

$t_0 + t_1 i + t_2 j + t_3 k$ is a quaternion.

Quaternions

Non-commutative field isomorphic to $\mathbb{R}^4$, spanned by three elements i, j, k that satisfy the following relations :

$$i^2 = j^2 = k^2 = ijk = -1$$

from which we immediately deduce

$$ij = k, \;\; jk = i, \;\; ki = j$$

Hamilton (1843)

Quaternions

Non-commutative field isomorphic to $\mathbb{R}^4$, spanned by three elements i, j, k that satisfy the following relations :

$$i^2 = j^2 = k^2 = ijk = -1$$

from which we immediately deduce

$$ij = k, \; jk = i, \; ki = j$$

Hamilton (1843)

Unit Quaternions and rotations

Let $q = q_0 + q_1 i + q_2 j + q_3 k$ be a unit quaternion :

$$q_0^2 + q_3^2 + q_2^2 + q_3^2 = 1$$

$\forall x = (x_0, x_1, x_2) \in \mathbb{R}^3$, let $u = x_0 i + x_1 j + x_2 k$

$$q \,.\, u \,.\, q^* = y_0 i + y_1 j + y_2 k$$

where $q^* = q_0 - q_1 i - q_2 j - q_3 k$ is the conjugate of q.
$y = (y_0, y_1, y_2)$ is the image of x by the rotation of matrix

$$\begin{pmatrix} 1 - 2(q_2^2 + q_3^2) & 2q_2q_1 - 2q_3q_0 & 2q_3q_1 + 2q_2q_0 \\ 2q_2q_1 + 2q_3q_0 & 1 - 2(q_1^2 + q_3^2) & 2q_3q_2 - 2q_1q_0 \\ 2q_3q_1 - 2q_2q_0 & 2q_3q_2 + 2q_1q_0 & 1 - 2(q_1^2 + q_2^2) \end{pmatrix}$$

Unit Quaternions and rotations

Let $q = q_0 + q_1 i + q_2 j + q_3 k$ be a unit quaternion :

$$q_0^2 + q_3^2 + q_2^2 + q_3^2 = 1$$

$\forall x = (x_0, x_1, x_2) \in \mathbb{R}^3$, let $u = x_0 i + x_1 j + x_2 k$

$$q \,.\, u \,.\, q^* = y_0 i + y_1 j + y_2 k$$

where $q^* = q_0 - q_1 i - q_2 j - q_3 k$ is the conjugate of q.
$y = (y_0, y_1, y_2)$ is the image of x by the rotation of matrix

$$\begin{pmatrix} 1 - 2(q_2^2 + q_3^2) & 2q_2 q_1 - 2q_3 q_0 & 2q_3 q_1 + 2q_2 q_0 \\ 2q_2 q_1 + 2q_3 q_0 & 1 - 2(q_1^2 + q_3^2) & 2q_3 q_2 - 2q_1 q_0 \\ 2q_3 q_1 - 2q_2 q_0 & 2q_3 q_2 + 2q_1 q_0 & 1 - 2(q_1^2 + q_2^2) \end{pmatrix}$$

Unit Quaternions and rotations

Let $q = q_0 + q_1 i + q_2 j + q_3 k$ be a unit quaternion :

$$q_0^2 + q_3^2 + q_2^2 + q_3^2 = 1$$

$\forall x = (x_0, x_1, x_2) \in \mathbb{R}^3$, let $u = x_0 i + x_1 j + x_2 k$

$$q \,.\, u \,.\, q^* = y_0 i + y_1 j + y_2 k$$

where $q^* = q_0 - q_1 i - q_2 j - q_3 k$ is the conjugate of q.
$y = (y_0, y_1, y_2)$ is the image of x by the rotation of matrix

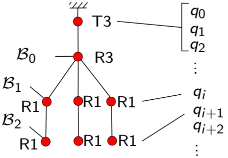
$$\begin{pmatrix} 1 - 2(q_2^2 + q_3^2) & 2q_2q_1 - 2q_3q_0 & 2q_3q_1 + 2q_2q_0 \\ 2q_2q_1 + 2q_3q_0 & 1 - 2(q_1^2 + q_3^2) & 2q_3q_2 - 2q_1q_0 \\ 2q_3q_1 - 2q_2q_0 & 2q_3q_2 + 2q_1q_0 & 1 - 2(q_1^2 + q_2^2) \end{pmatrix}$$

Unit Quaternions and rotations

- Notice that q and $-q$ represent the same rotation
- $SO(3)$ is isomorphic to $Sp(1)/\{\pm 1\}$, the half-sphere of $\mathbb{R}^4$.

Configuration of a robot

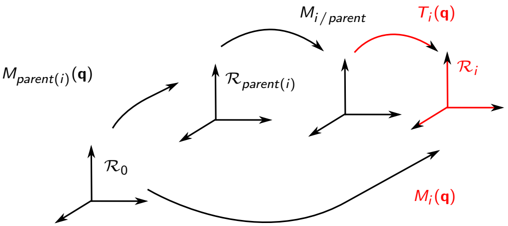
The configuration **q** of a robot is represented by the concatenation of the parameters of each joint.



Forward kinematics

Computation of the position of each joint in the global frame

$$M_i(\mathbf{q}) = M_{parent(i)}(\mathbf{q})\ M_{i/parent}\ \ T_i(\mathbf{q})$$



Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$\begin{aligned} \mathcal{C}_{obst} = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\ & \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\ & \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\} \end{aligned}$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$\begin{aligned}\mathcal{C}_{obst} = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\ & \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\ & \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\}\end{aligned}$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$\begin{aligned}\mathcal{C}_{obst} = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\ & \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\ & \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\}\end{aligned}$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$\begin{aligned}\mathcal{C}_{obst} = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\ & \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\ & \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\}\end{aligned}$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$
\begin{aligned}
\mathcal{C}_{obst} \quad = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\
& \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\
& \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\}
\end{aligned}
$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Definitions

- Workspace : $\mathcal{W} = \mathbb{R}^2$ or $\mathbb{R}^3$: space in which the robot evolves
- Obstacle in workspace : compact subset of $\mathcal{W}$, denoted by $\mathcal{O}$.
- Configuration space : $\mathcal{C}$.
- Position in configuration $\mathbf{q}$ of a point $M \in \mathcal{B}_i$: $\mathbf{x}_i(M, \mathbf{q})$.
- Obstacle in the configuration space :

$$
\begin{aligned}
\mathcal{C}_{obst} \quad = \{\mathbf{q} \in \mathcal{C}, \quad & \exists i \in \{1, \cdots, m\},\ \exists M \in \mathcal{B}_i,\ \mathbf{x}_i(M, \mathbf{q}) \in \mathcal{O} \text{ or} \\
& \exists i, j \in \{1, \cdots, m\},\ \exists M_i \in \mathcal{B}_i,\ \exists M_j \in \mathcal{B}_j, \\
& \mathbf{x}_i(M_i, \mathbf{q}) = x_j(M_j, \mathbf{q})\}
\end{aligned}
$$

- Free configuration space : $\mathcal{C}_{free} = \mathcal{C} \setminus \mathcal{C}_{obst}$.

Motion

- Configuration space :
  - differential manifold
- Motion :
  - continuous function from $[0,1]$ to $\mathcal{C}$.
- Collision-free motion :
  - continuous function from $[0,1]$ to $\mathcal{C}_{free}$.

Motion

- Configuration space :
  - differential manifold
- Motion :
  - continuous function from $[0, 1]$ to $\mathcal{C}$.
- Collision-free motion :
  - continuous function from $[0, 1]$ to $\mathcal{C}_{free}$.

Motion

- Configuration space :
  - differential manifold
- Motion :
  - continuous function from $[0, 1]$ to $\mathcal{C}$.
- Collision-free motion :
  - continuous function from $[0, 1]$ to $\mathcal{C}_{free}$.

Random methods

- In the early 1990's, random methods started being developed
- Principle
  - shoot random configurations
  - test whether they are in collision
  - build a graph (roadmap) the nodes of which are free configurations
  - and the edges of which are collision-free linear interpolations

Random methods

- In the early 1990's, random methods started being developed
- Principle
  - shoot random configurations
  - test whether they are in collision
  - build a graph (roadmap) the nodes of which are free configurations
  - and the edges of which are collision-free linear interpolations

Random methods

- In the early 1990's, random methods started being developed
- Principle
  - shoot random configurations
  - test whether they are in collision
  - build a graph (roadmap) the nodes of which are free configurations
  - and the edges of which are collision-free linear interpolations

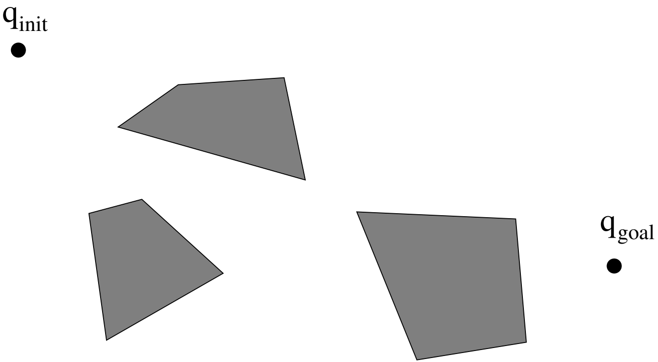
Random methods

- In the early 1990's, random methods started being developed
- Principle
  - shoot random configurations
  - test whether they are in collision
  - build a graph (roadmap) the nodes of which are free configurations
  - and the edges of which are collision-free linear interpolations

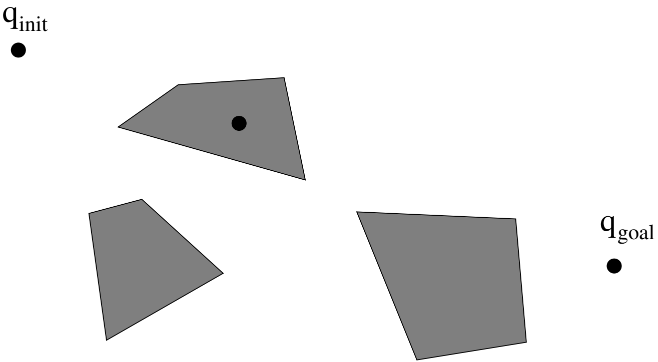
Random methods

- In the early 1990's, random methods started being developed
- Principle
  - shoot random configurations
  - test whether they are in collision
  - build a graph (roadmap) the nodes of which are free configurations
  - and the edges of which are collision-free linear interpolations

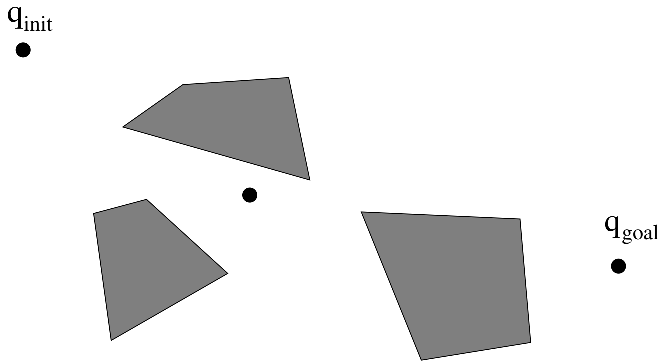
Probabilistic roadmap (PRM) 1994



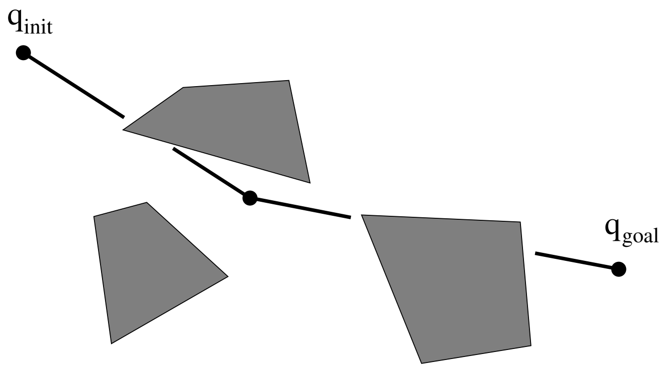
Probabilistic roadmap (PRM) 1994



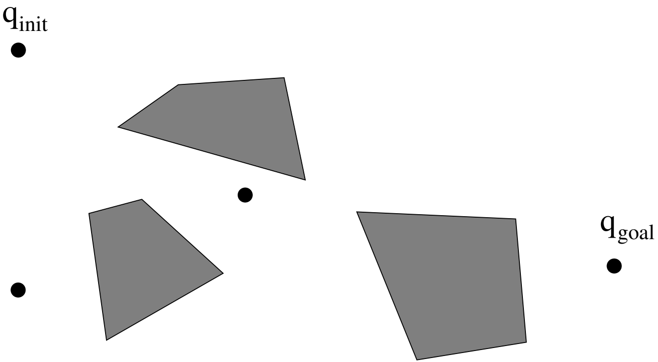
Probabilistic roadmap (PRM) 1994



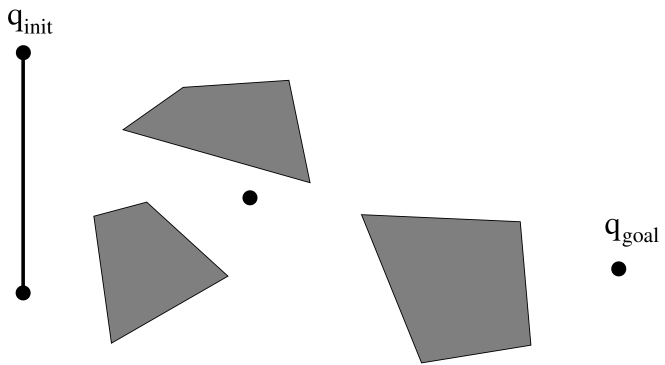
Probabilistic roadmap (PRM) 1994



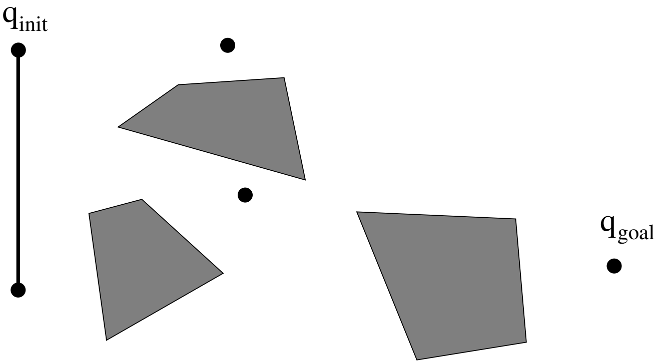
Probabilistic roadmap (PRM) 1994



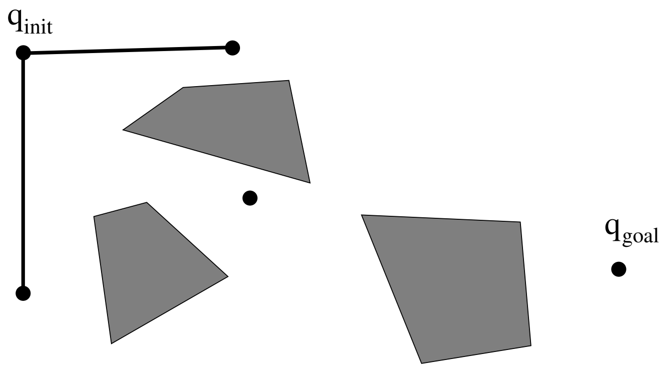
Probabilistic roadmap (PRM) 1994



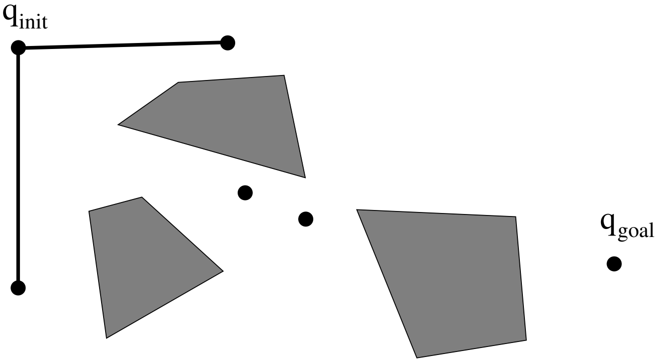
Probabilistic roadmap (PRM) 1994



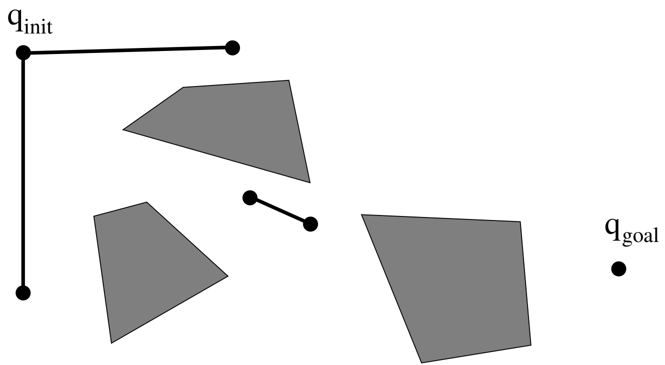
Probabilistic roadmap (PRM) 1994



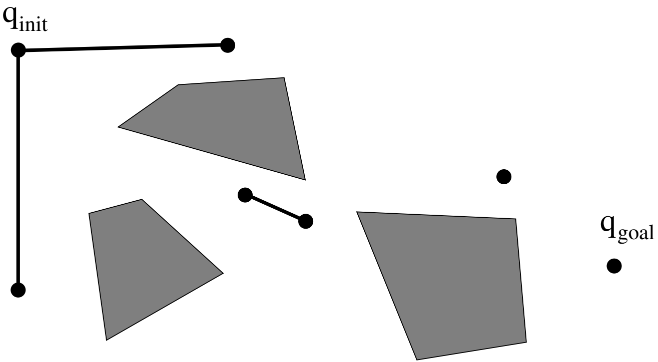
Probabilistic roadmap (PRM) 1994



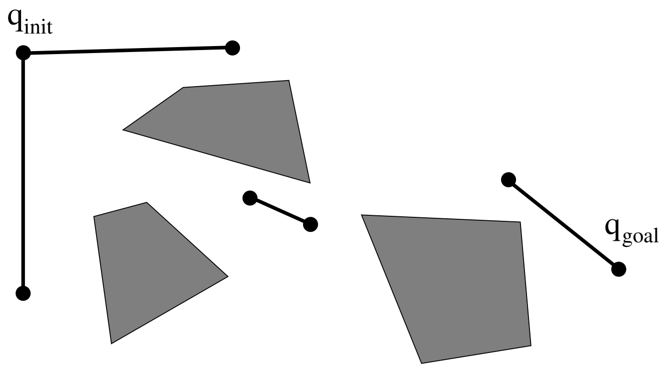
Probabilistic roadmap (PRM) 1994



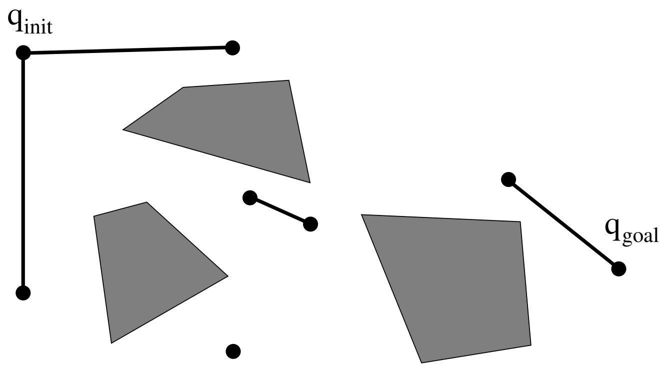
Probabilistic roadmap (PRM) 1994



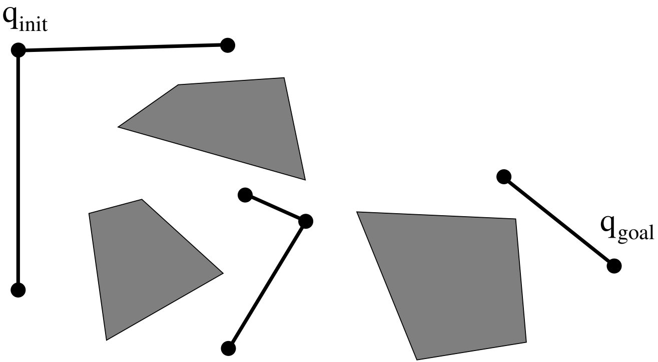
Probabilistic roadmap (PRM) 1994



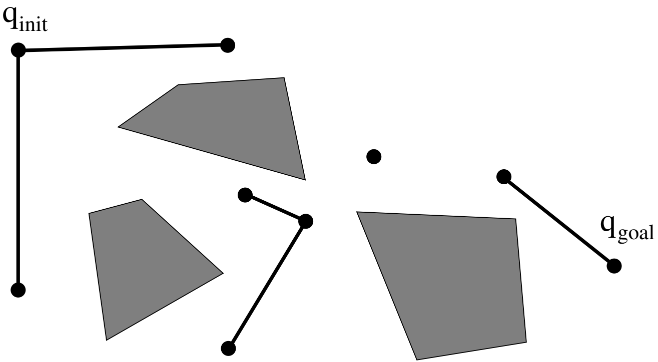
Probabilistic roadmap (PRM) 1994



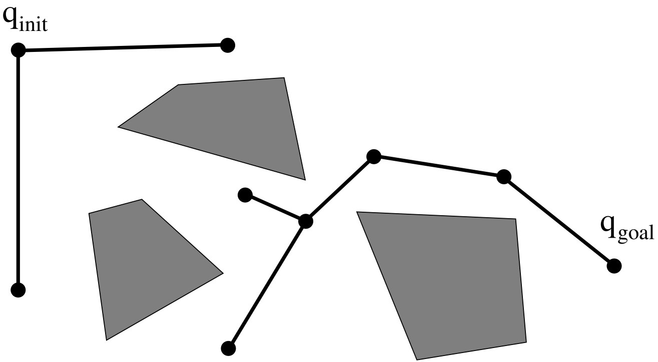
Probabilistic roadmap (PRM) 1994



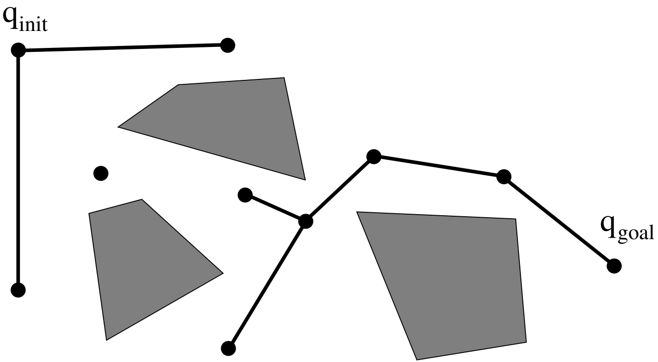
Probabilistic roadmap (PRM) 1994



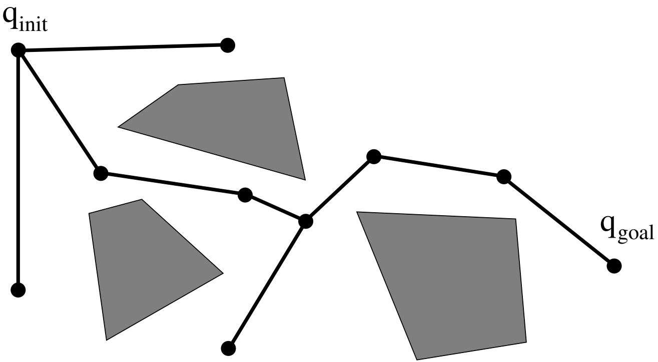
Probabilistic roadmap (PRM) 1994



Probabilistic roadmap (PRM) 1994



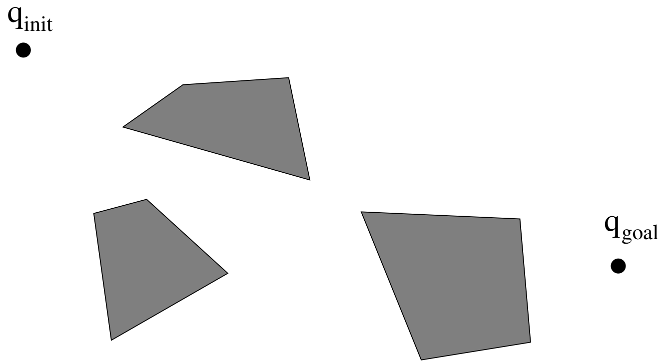
Probabilistic roadmap (PRM) 1994



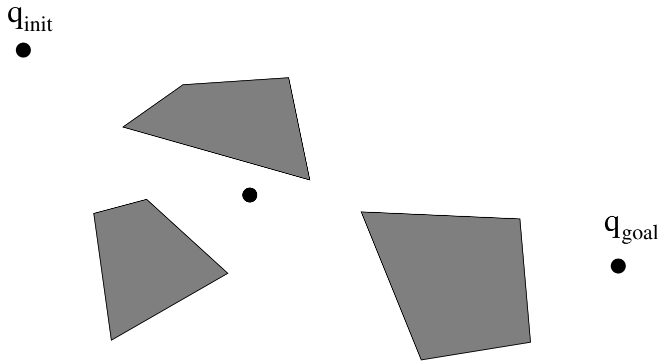
Probabilistic roadmap (PRM)

- A lot of useless nodes are created,
  - this increases the cost to connect new nodes to the existing roadmap
- Improvement : visibility-based PRM
  - Only *interesting* nodes are kept.

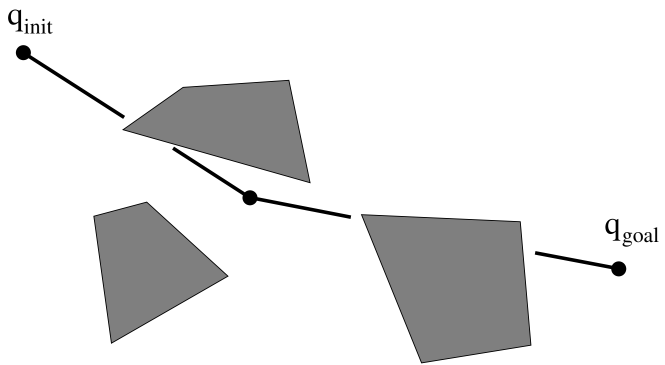
Visibility-based probabilistic roadmap (Visi-PRM) 1999



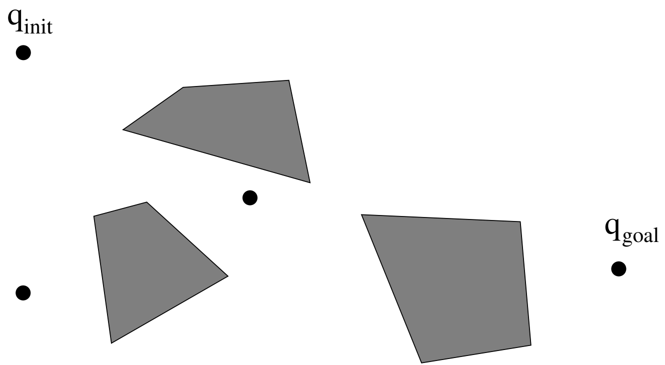
Visibility-based probabilistic roadmap (Visi-PRM) 1999



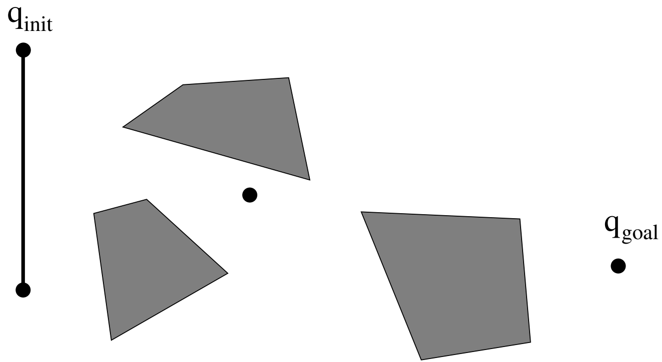
Visibility-based probabilistic roadmap (Visi-PRM) 1999



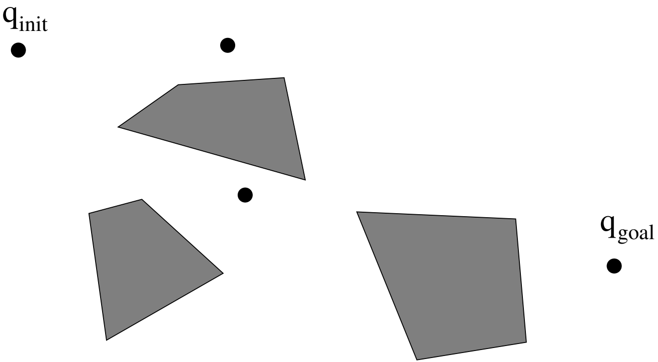
Visibility-based probabilistic roadmap (Visi-PRM) 1999



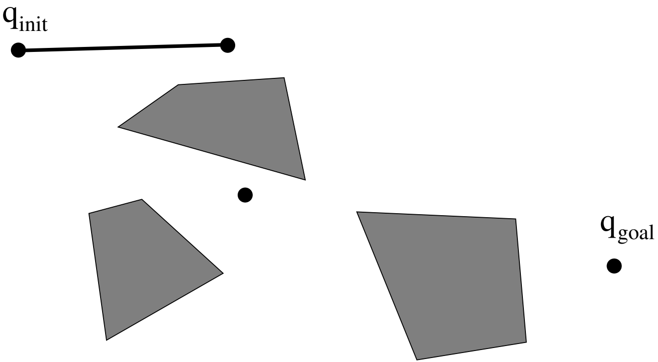
Visibility-based probabilistic roadmap (Visi-PRM) 1999



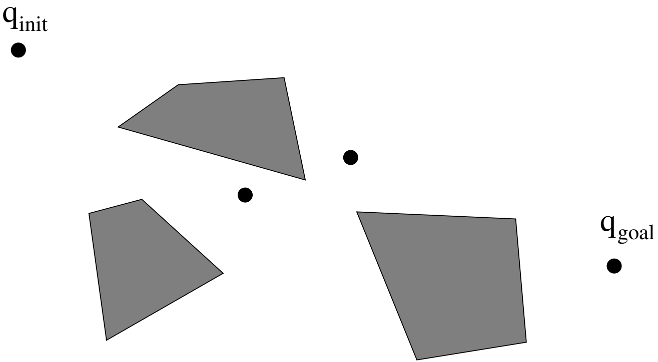
Visibility-based probabilistic roadmap (Visi-PRM) 1999



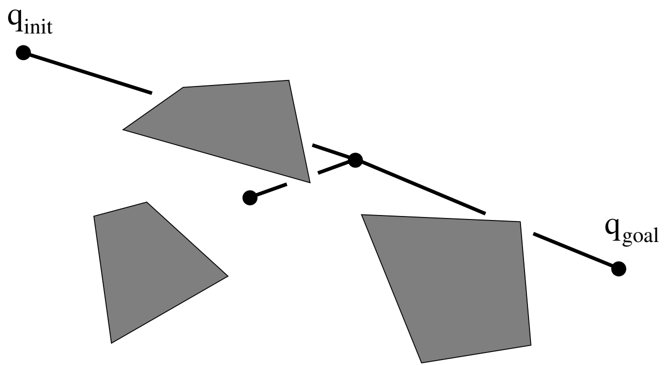
Visibility-based probabilistic roadmap (Visi-PRM) 1999



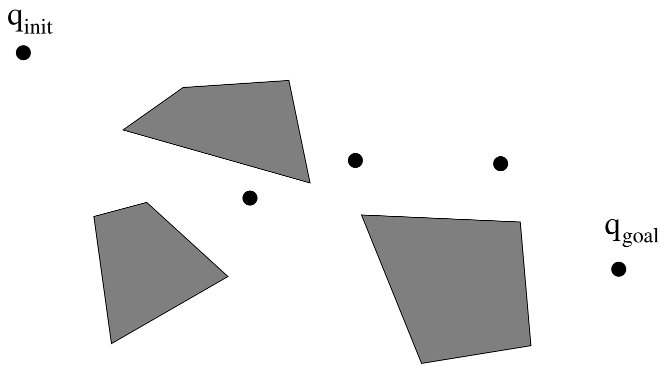
Visibility-based probabilistic roadmap (Visi-PRM) 1999



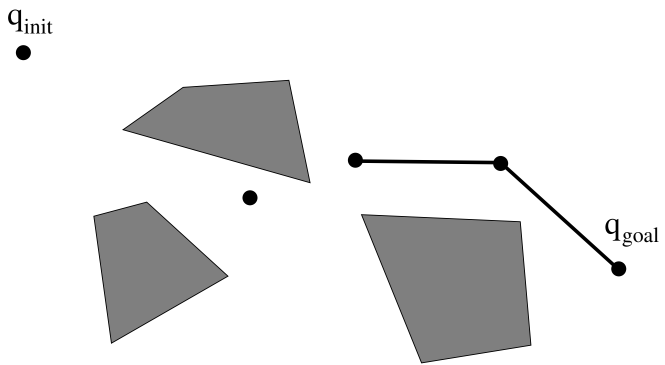
Visibility-based probabilistic roadmap (Visi-PRM) 1999



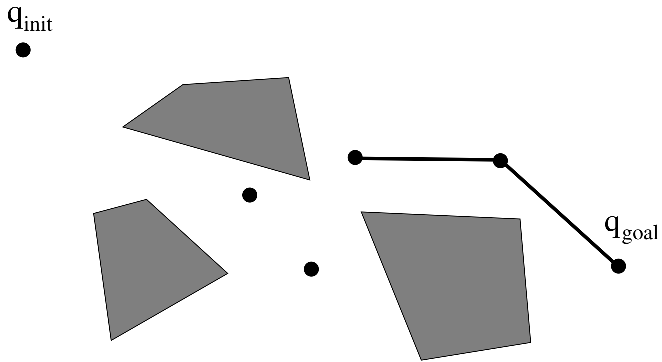
Visibility-based probabilistic roadmap (Visi-PRM) 1999



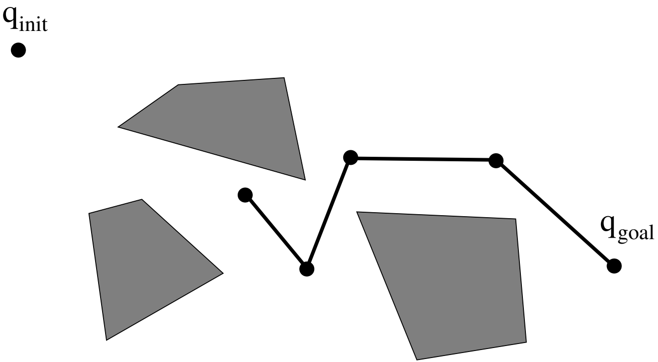
Visibility-based probabilistic roadmap (Visi-PRM) 1999



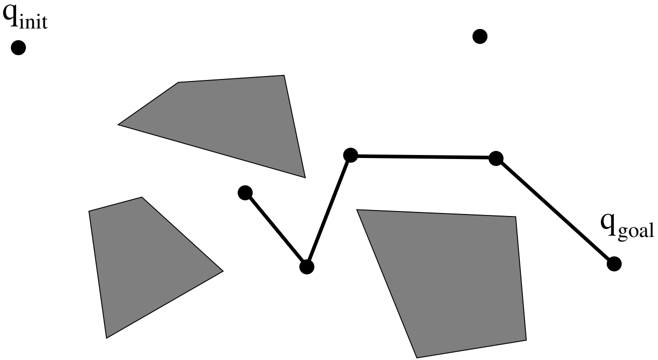
Visibility-based probabilistic roadmap (Visi-PRM) 1999



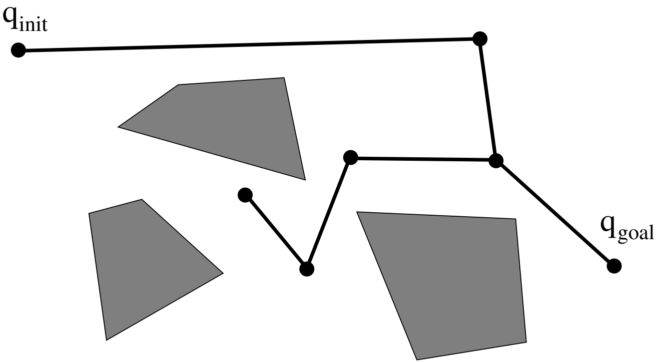
Visibility-based probabilistic roadmap (Visi-PRM) 1999



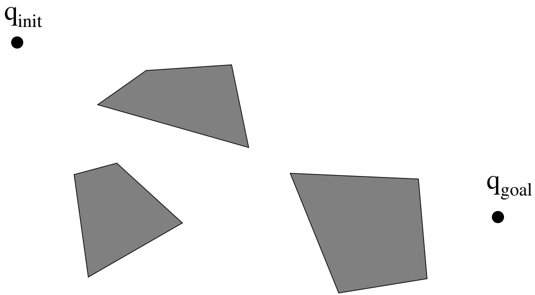
Visibility-based probabilistic roadmap (Visi-PRM) 1999



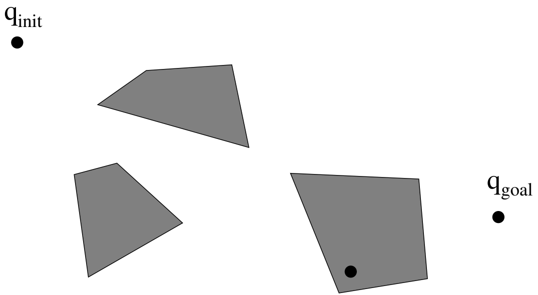
Visibility-based probabilistic roadmap (Visi-PRM) 1999



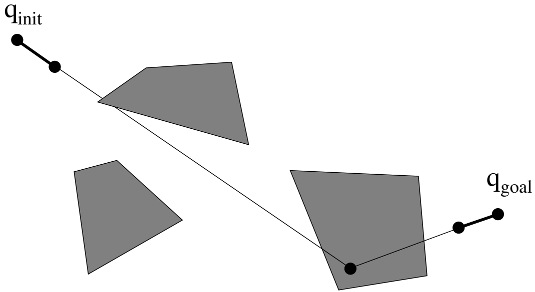
Rapidly exploring Random Tree (RRT) 2000



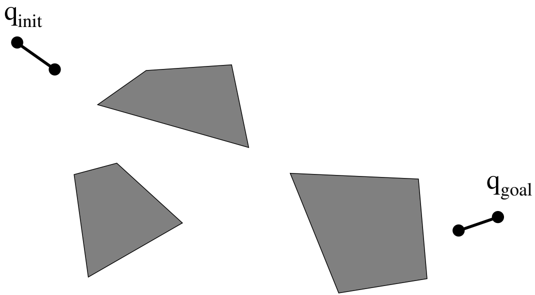
Rapidly exploring Random Tree (RRT) 2000



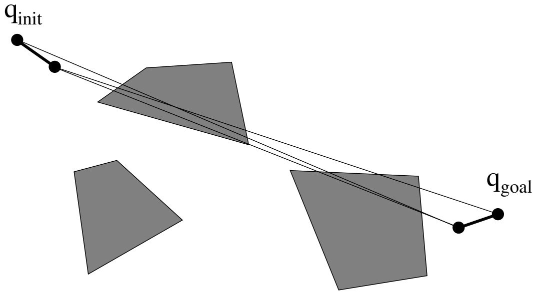
Rapidly exploring Random Tree (RRT) 2000



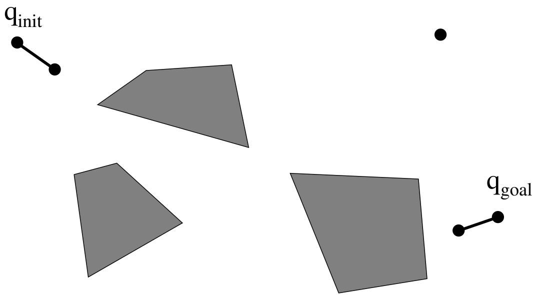
Rapidly exploring Random Tree (RRT) 2000



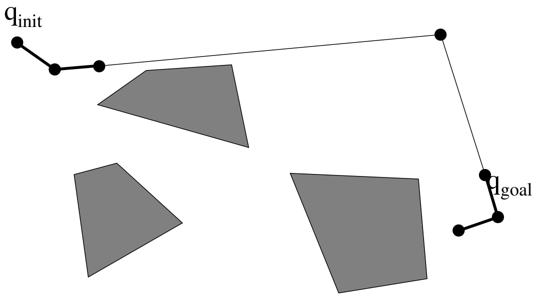
Rapidly exploring Random Tree (RRT) 2000



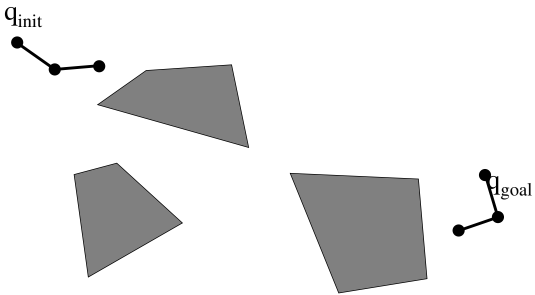
Rapidly exploring Random Tree (RRT) 2000



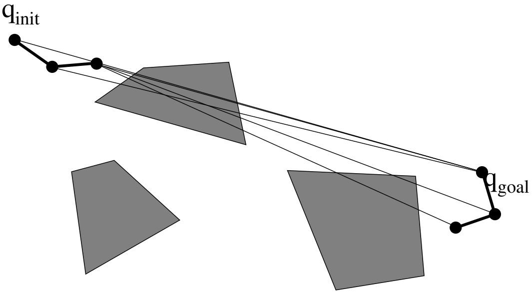
Rapidly exploring Random Tree (RRT) 2000



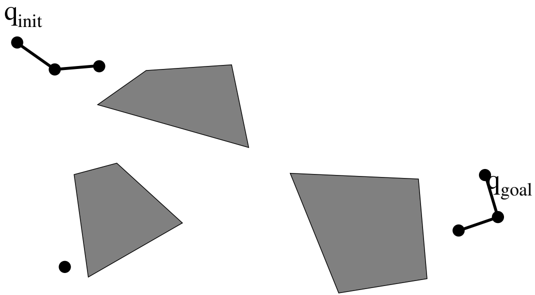
Rapidly exploring Random Tree (RRT) 2000



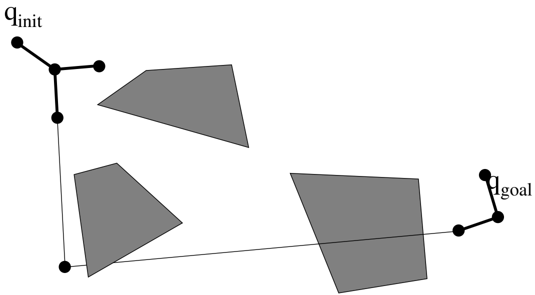
Rapidly exploring Random Tree (RRT) 2000



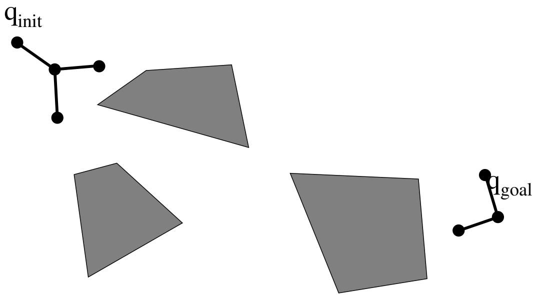
Rapidly exploring Random Tree (RRT) 2000



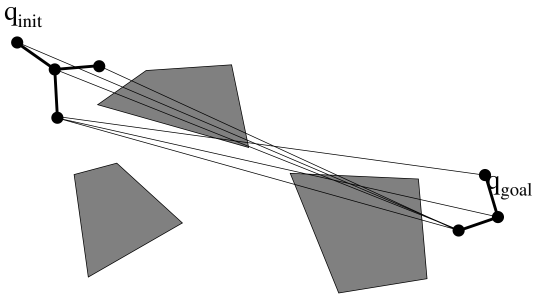
Rapidly exploring Random Tree (RRT) 2000



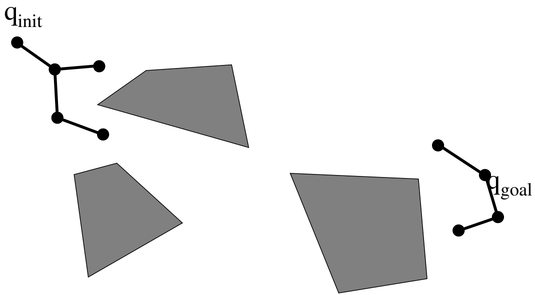
Rapidly exploring Random Tree (RRT) 2000



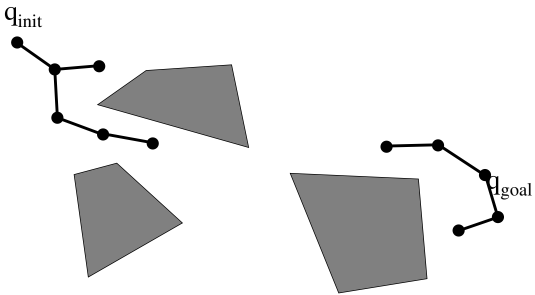
Rapidly exploring Random Tree (RRT) 2000



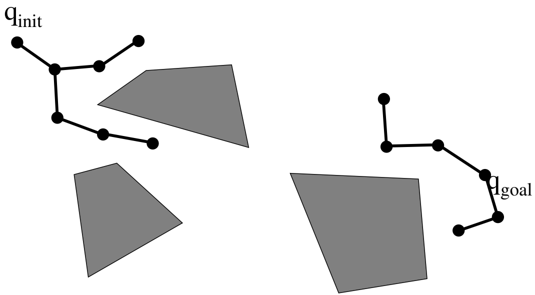
Rapidly exploring Random Tree (RRT) 2000



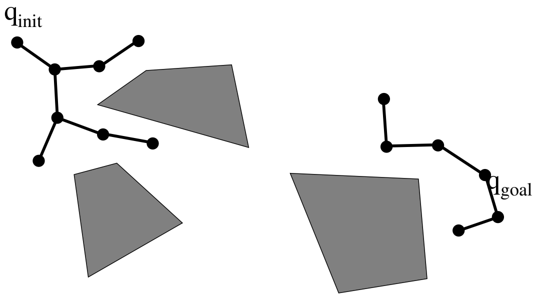
Rapidly exploring Random Tree (RRT) 2000



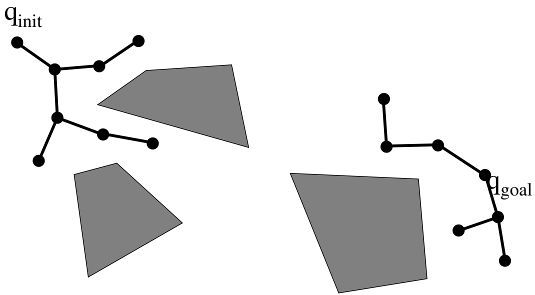
Rapidly exploring Random Tree (RRT) 2000



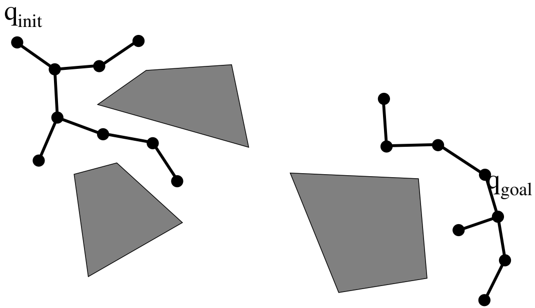
Rapidly exploring Random Tree (RRT) 2000



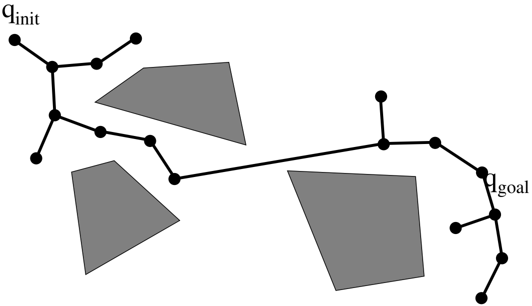
Rapidly exploring Random Tree (RRT) 2000



Rapidly exploring Random Tree (RRT) 2000



Rapidly exploring Random Tree (RRT) 2000



Random methods

- Pros :
  - no explicit computation of the free configuration space,
  - easy to implement,
  - robust.
- Cons :
  - no completeness property, only probabilistic completeness,
  - difficult to find narrow passages.
- Requested operators :
  - Collision tests
    - for configurations (static),
    - for paths (dynamic)

Random methods

- Pros :
  - no explicit computation of the free configuration space,
  - easy to implement,
  - robust.
- Cons :
  - no completeness property, only probabilistic completeness,
  - difficult to find narrow passages.
- Requested operators :
  - Collision tests
    - for configurations (static),
    - for paths (dynamic)

Random methods

- Pros :
  - no explicit computation of the free configuration space,
  - easy to implement,
  - robust.
- Cons :
  - no completeness property, only probabilistic completeness,
  - difficult to find narrow passages.
- Requested operators :
  - Collision tests
    - for configurations (static),
    - for paths (dynamic)

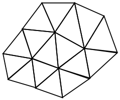
Collision tests

- for configurations
  - problem : given
    - two rigid sets of triangles,
    - the relative position of one set with respect to the other set,

    determine whether the intersection between the sets is empty, or compute the distance between the sets.

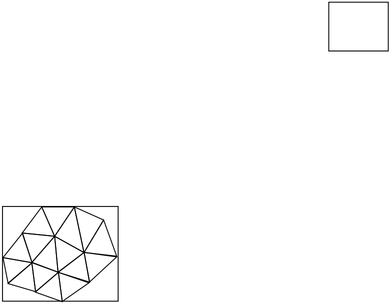
Hierarchy of bounding volumes

- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles



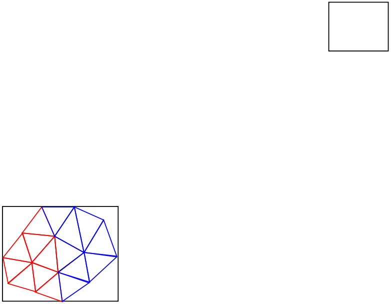
Hierarchy of bounding volumes

- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles



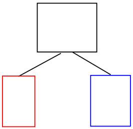
Hierarchy of bounding volumes

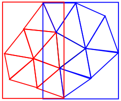
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles



Hierarchy of bounding volumes

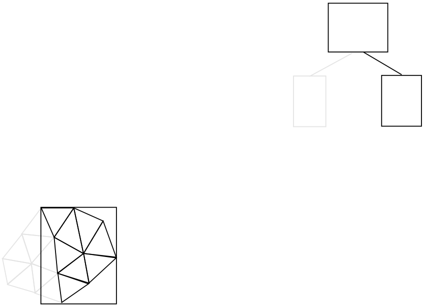
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles





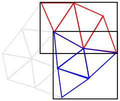
Hierarchy of bounding volumes

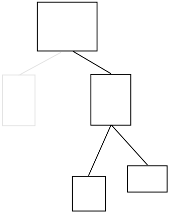
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles



Hierarchy of bounding volumes

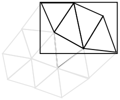
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles

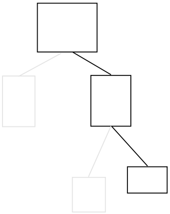




Hierarchy of bounding volumes

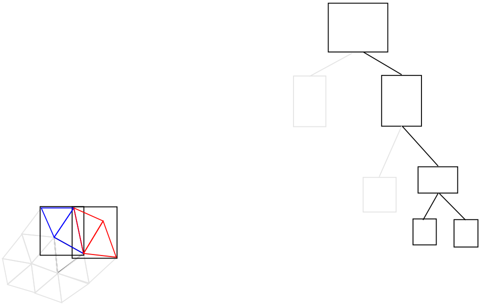
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles





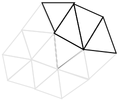
Hierarchy of bounding volumes

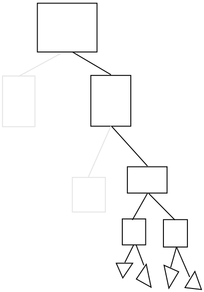
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles



Hierarchy of bounding volumes

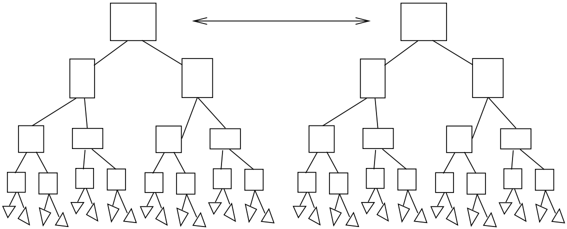
- Binary trees of bounding volumes such that
  - each node contains two children,
  - leaves are the triangles





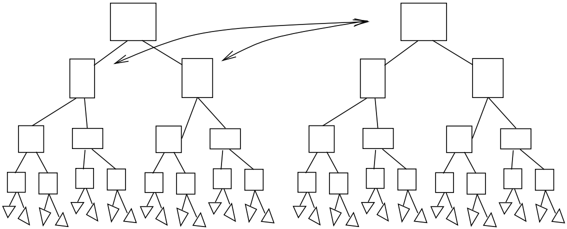
Collision tests for configurations

- Algorithm
  - test root nodes of each tree against one another
  - if two nodes are in collision, test one with the children of the other node



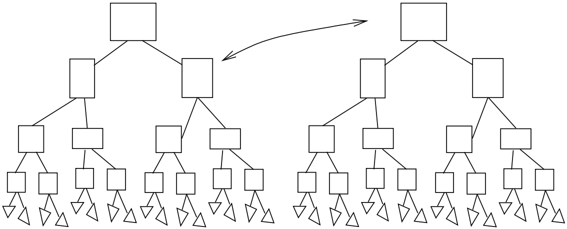
Collision tests for configurations

- Algorithm
  - test root nodes of each tree against one another
  - if two nodes are in collision, test one with the children of the other node



Collision tests for configurations

- Algorithm
  - test root nodes of each tree against one another
  - if two nodes are in collision, test one with the children of the other node



Collision tests for configurations

- Algorithm
  - test root nodes of each tree against one another
  - if two nodes are in collision, test one with the children of the other node