

Universidade do Minho

Escola de Engenharia

Departamento de Informática

Planeamento de Trajetórias

Sistemas Autónomos

Perfil Sistemas Inteligentes @ MEI/MiEI 1º/4º – 2º semestre

Cesar Analide, Bruno Fernandes

- *How can a robot decide what motions to perform in order to achieve tasks in the **physical world**?*

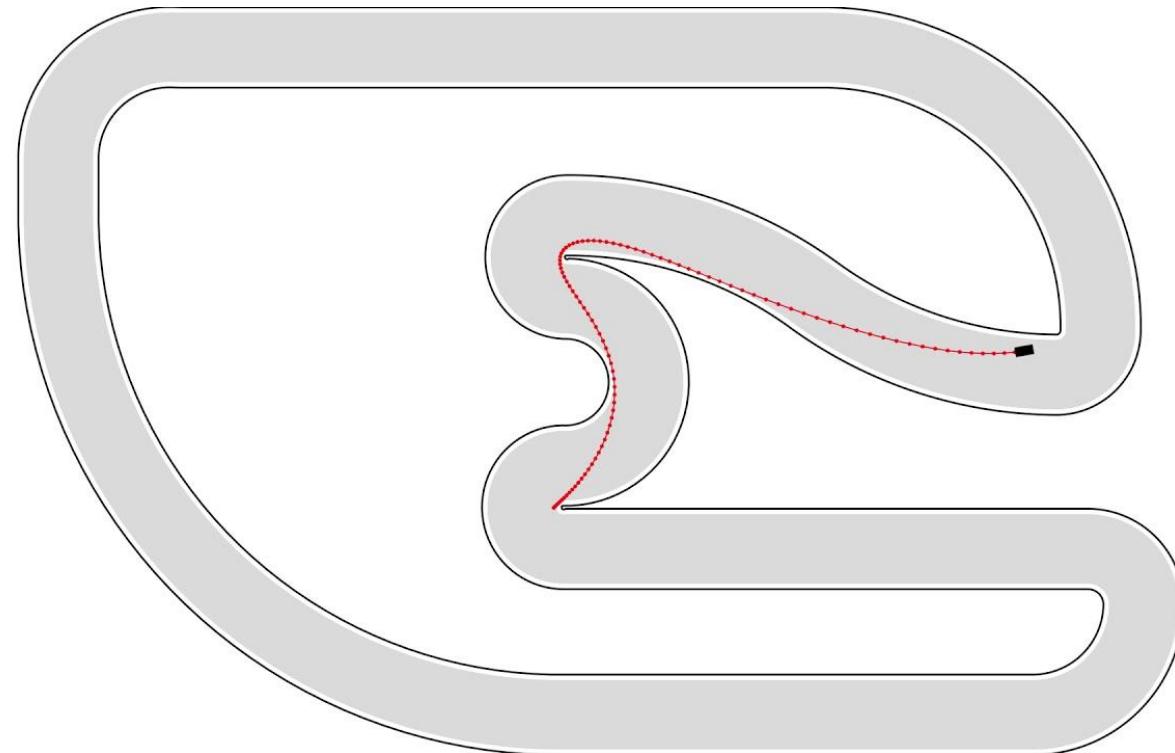
Jean-Paul Laumond, "Robot Motion Planning and Control", Springer, 1998

- Como definir os movimentos que um robô deve executar por forma a realizar tarefas no mundo físico?



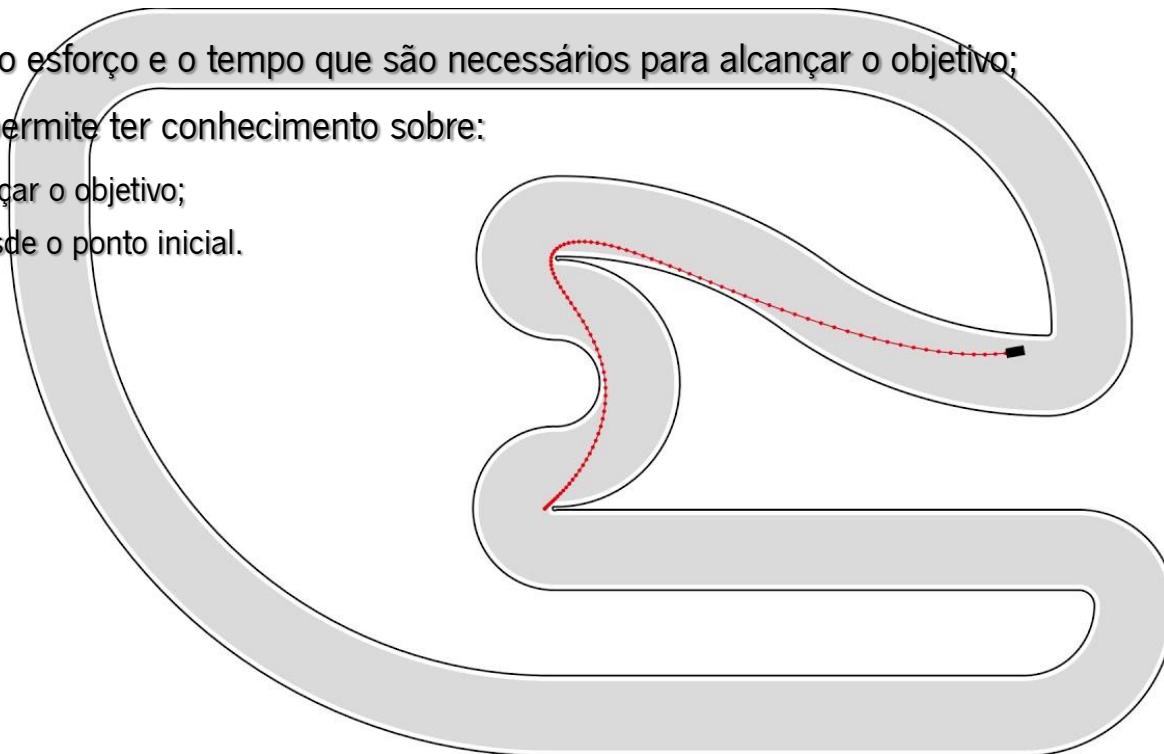
O que é Planeamento?

- **Planeamento** é a preparação de um conjunto de passos de execução ou de ações que permitam atingir um determinado objetivo.



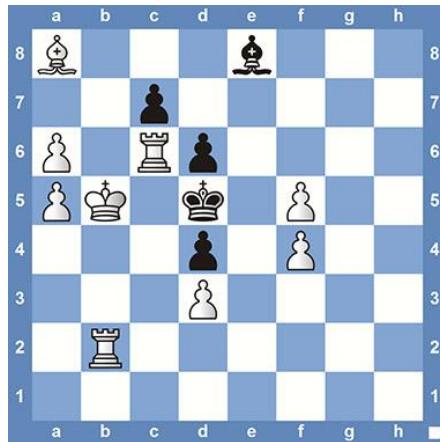
Por quê fazer Planeamento?

- Planeamento é a preparação de um conjunto de passos de execução ou de ações que permitam atingir um determinado objetivo;
- Um plano permite reduzir o esforço e o tempo que são necessários para alcançar o objetivo;
- A execução de um plano permite ter conhecimento sobre:
 - quanto falta para alcançar o objetivo;
 - quanto se executou desde o ponto inicial.



Quando fazer Planeamento?

- O planeamento **é importante** quando os passos necessários à resolução do problema não podem ser desfeitos ou ignorados (exemplo: jogo de xadrez);
- Quando a execução de ações é reversível (exemplo: puzzles ou demonstração de teoremas), o planeamento **perde** alguma **importância**.



- Em **Teoria do Controlo**, o **Planeamento** está (mais) diretamente relacionado com o **Controlo** físico do sistema, em termos:
 - da dinâmica dos corpos;
 - das forças a empregar;
 - da estabilidade dos corpos;
 - do cálculo das condições ótimas;
 - etc.
- O **Planeamento** é um estudo realizado em espaços contínuos (não-finitos, não-discretos).

O que é Planeamento?



The Cubli

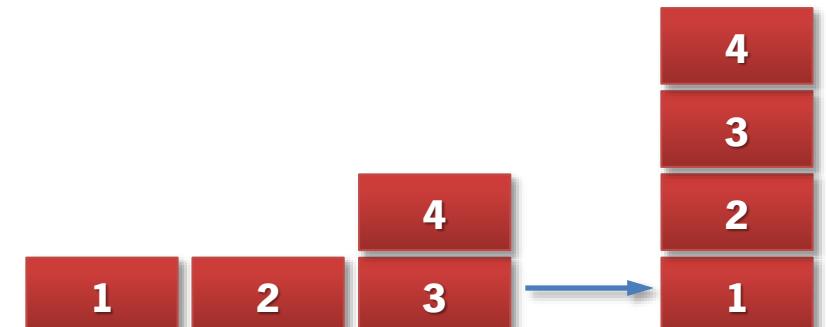
Building a cube that can jump up and balance



ETH zürich

O que é Planeamento?

- Em **Inteligência Artificial**, o **Planeamento** está relacionado com a procura de uma sequência de operações (lógicas) ou **ações que transformam um** determinado **estado inicial num** desejado **estado final**;
- O problema canónico de planeamento em IA é o **Blocks World**, que consiste em determinar os passos a executar para deslocar um conjunto de blocos numa mesa, de um estado inicial até um estado final;

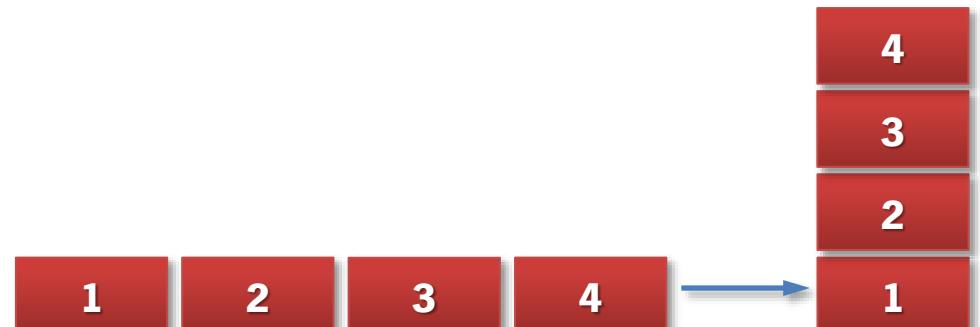


<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwoptcgi>

<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwcgi>

Problema: *Blocks World*

- O problema é composto por um conjunto de blocos (caixas), inicialmente dispostos numa superfície (mesa);
- O objetivo é construir uma torre (ou mais) de blocos;
- Consideram-se as restrições:
 - só é possível mover um bloco de cada vez;
 - só é possível mover um bloco que esteja livre
(sem nenhum outro bloco em cima);
- Não existem restrições físicas!

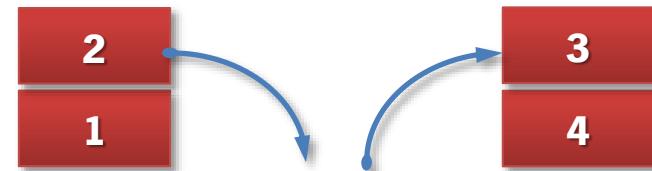


<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwoptcgi>

<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwcgi>

Problema: *Blocks World*

- O problema é composto por um conjunto de blocos (caixas), inicialmente dispostos numa superfície (mesa);
- O objetivo é construir uma torre (ou mais) de blocos;
- Executam-se as ações:
 - colocar um bloco em cima de outro;
 - colocar um bloco na mesa.
- Estas ações não consideram restrições físicas dos blocos ou do ambiente.

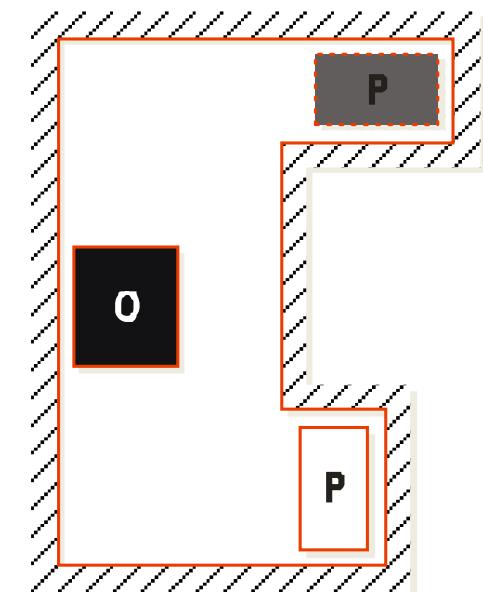


<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwoptcgi>

<http://users.rsise.anu.edu.au/~jks/cgi-bin/bwstates/bwcgi>

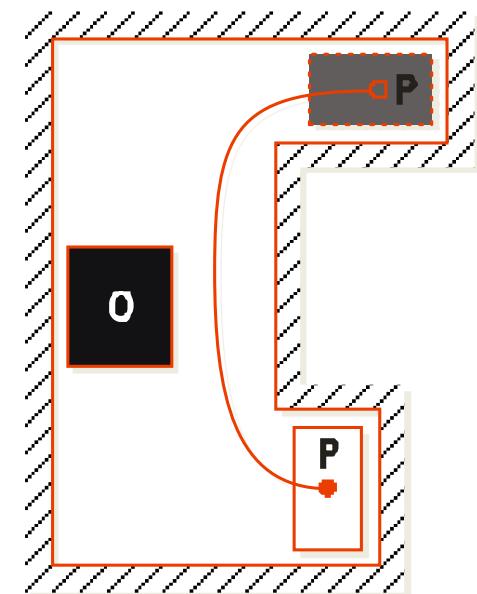
O que é Planeamento?

- Em **Sistemas Autónomos/Robótica**, o planeamento é entendido como sinónimo de **Planeamento de Trajetórias**;
- O exemplo canónico no Planeamento de Trajetórias, é o **Piano Mover**:
 - Como deslocar um piano dentro de um apartamento, desde um ponto inicial até um ponto final, através de corredores com obstáculos?



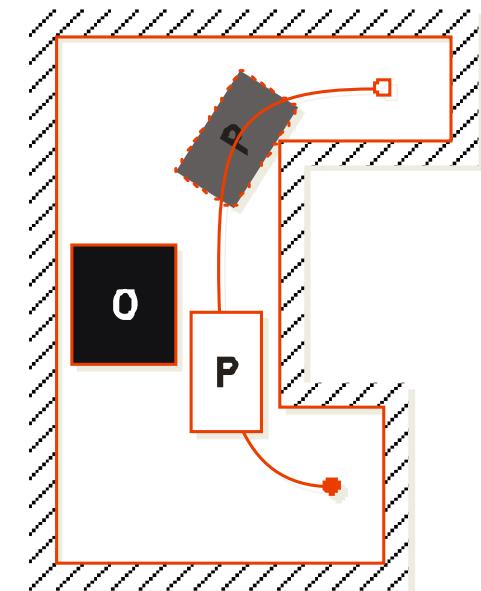
Problema: *Piano Mover*

- A resolução do problema do ***Piano Mover*** passa por considerar duas situações importantes:
 - Como **planejar uma boa trajetória** (por exemplo, a mais curta), que transporte o robô desde a origem até ao destino;



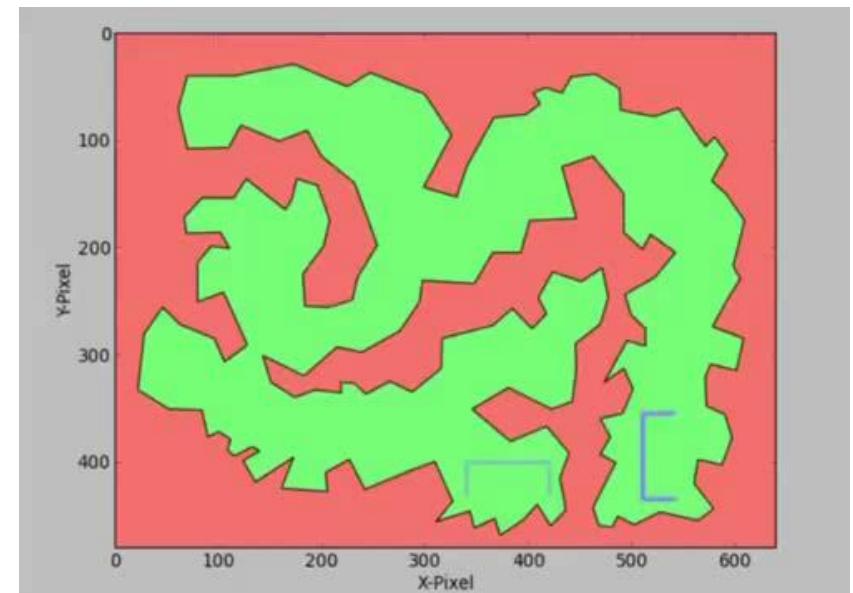
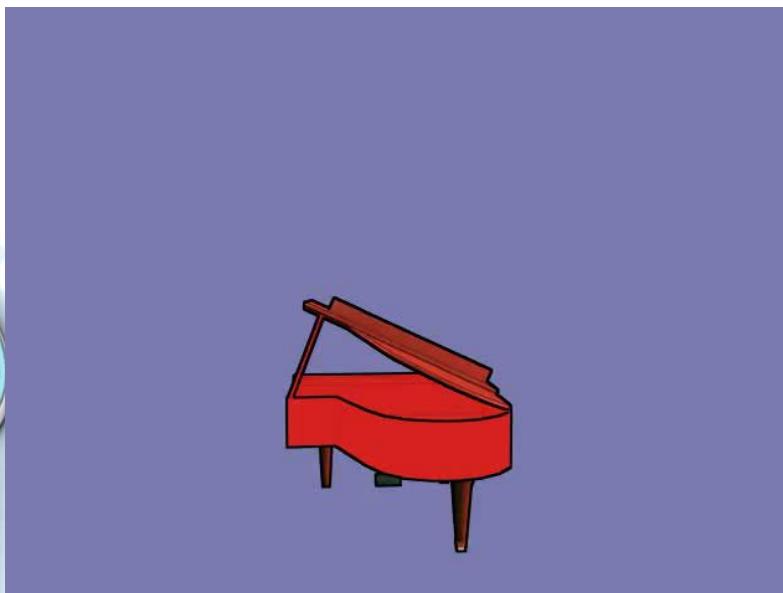
Problema: *Piano Mover*

- A resolução do problema do ***Piano Mover*** passa por considerar duas situações importantes:
 - Como planear uma boa trajetória (por exemplo, a mais curta), que transporte o robô desde a origem até ao destino;
 - Como **controlar o robô** no seu caminho até ao destino, durante o trajeto planeado.

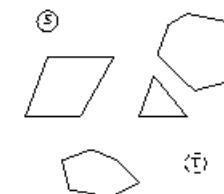


Problema: *Piano Mover*

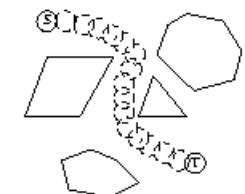
- A resolução do problema do ***Piano Mover*** passa por considerar duas situações importantes:
 - Como planear uma boa trajetória (por exemplo, a mais curta), que transporte o robô desde a origem até ao destino;
 - Como controlar o robô no seu caminho até ao destino, durante o trajeto planeado.



- Existe uma grande complexidade em problemas de Planeamento de Trajetórias, considerando diversos fatores:
 - É possível simplificar a forma do robô?
 - O robô é um ponto?
 - Qual a liberdade de movimentos do robô?
 - Os movimentos são limitados (translação, rotação)?
 - Os obstáculos são conhecidos antecipadamente?



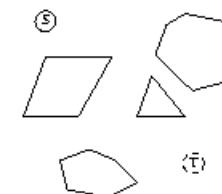
INPUT



OUTPUT

ROBOCODE

- Existem muitas questões de complexidade em problemas de Planeamento de Trajetórias, considerando diversos fatores:
 - É possível simplificar a forma do robô? **NÃO**
 - O robô é um ponto? **NÃO**
 - Qual a liberdade de movimentos do robô? **?**
 - Os movimentos são limitados (translação, rotação)?
 - Os obstáculos são conhecidos antecipadamente? **SIM ou NÃO?**



INPUT



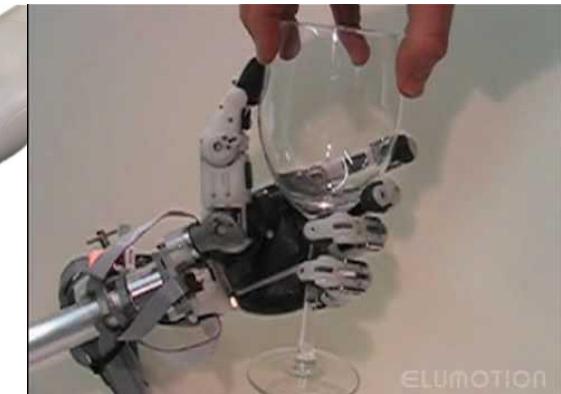
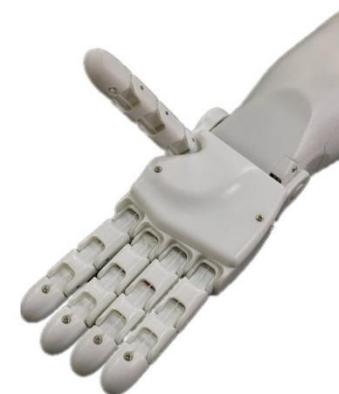
OUTPUT

- A **Configuração** de um robô é definida por um determinado conjunto de variáveis:

- em robôs móveis, as variáveis são, tipicamente, a posição e a orientação;



- em robôs articulados, as variáveis são definidas pela posição das diversas juntas/articulações que o compõem (as juntas definem os graus de liberdade do robô);



- **Graus de liberdade (Degrees of freedom - DOF):**
 - Para um robô fixo no ambiente, o número de **DOF** é dado pela quantidade de juntas;



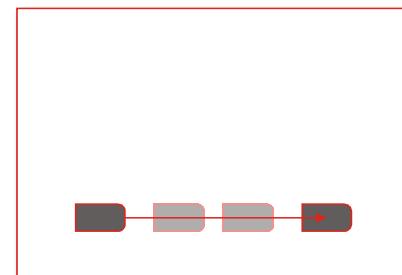
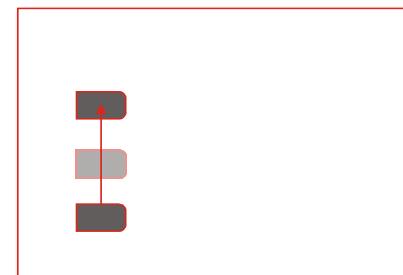
- **Graus de liberdade (Degrees of freedom - DOF):**
 - Para um robô fixo no ambiente, o número de **DOF** é dado pela quantidade de juntas;



Definições

- **Graus de liberdade (Degrees of freedom - DOF):**

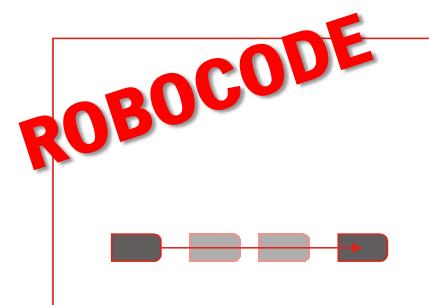
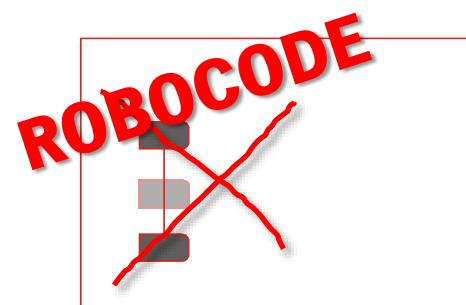
- Para um robô fixo no ambiente, o número de **DOF** é dado pela quantidade de juntas;
- Para um robô móvel, o número de **DOF** depende da sua capacidade de locomoção:



Definições

- **Graus de liberdade (Degrees of freedom - DOF):**

- Para um robô fixo no ambiente, o número de **DOF** é dado pela quantidade de juntas;
- Para um robô móvel, o número de **DOF** depende da sua capacidade de locomoção:

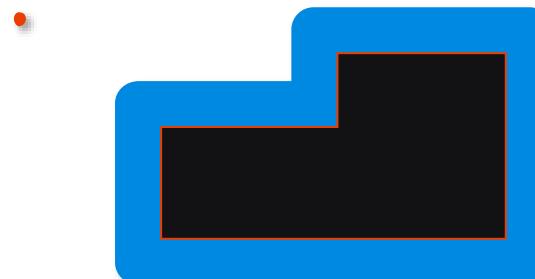


- 
- O **Espaço de Configurações** (*Configuration Space*) é o conjunto de todas as configurações admissíveis (possíveis) de um robô;
 - O **Espaço de Configurações** define, também, todas as possibilidades de movimentos contínuos de um robô;
 - A **Trajetória** de um robô é um caminho no **Espaço de Configurações**;
 - Para robôs móveis (rígidos, sem articulações), existe uma transformação aplicável ao robô e aos objetos que convertem o robô em 1 ponto:
 - *C-Space Transform*.

- Esta transformação permite planear trajetórias para pontos, em vez de para polígonos (ou poliedros).
- Até onde posso deslocar **um robô**, na vizinhança de **um objeto**:



- Até onde posso deslocar **um ponto**, na vizinhança de **um objeto expandido**:



- A **Trajetória** de um robô é um caminho no **Espaço de Configurações**:

- Esse caminho deve existir no subespaço de configurações no qual **não existe** qualquer possibilidade de o robô chocar com os obstáculos;
- Este subespaço de configurações designa-se **Espaço Livre**;
- O **Planeamento de Trajetórias** deve encontrar o melhor caminho neste **Espaço Livre** de configurações.

Como fazer Planeamento?

- Construção de Rotas;
- Desvio de Obstáculos;
- Arquiteturas de Navegação.

Construção de Rotas

- 
- A Construção de Rotas é importante para que um robô (sistema autónomo) possa mover-se ou realizar uma determinada tarefa **da forma mais eficaz e no menor período de tempo**, com o objetivo de o tornar mais produtivo e fiável;
 - Existem vários métodos para Construção de Rotas:
 - *Road Map*;
 - Decomposição Celular;
 - Navegação por Marcos de Influência.

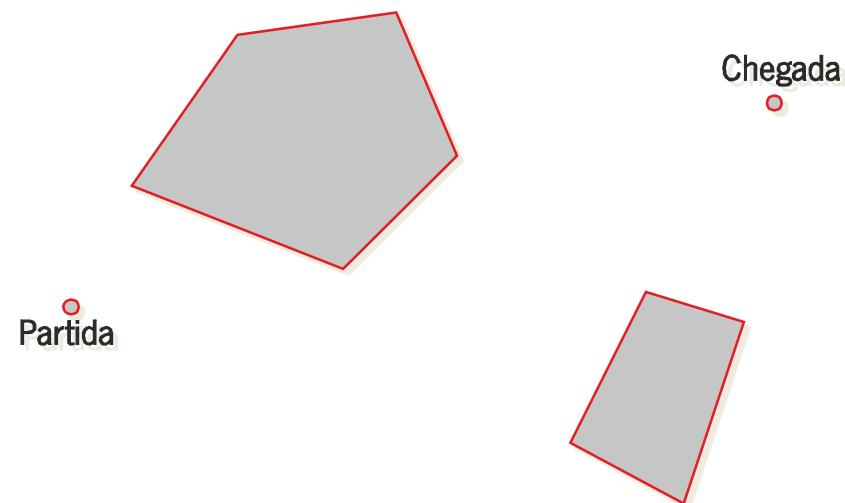
Construção de Rotas: *Road Map*

- Captura o **espaço livre** em torno do robô através de um traçado de linhas e curvas (rede);
- Esta rede é formada por segmentos, de modo a determinar a melhor rota para mover o robô desde a partida até à chegada;
- São dois os métodos mais comuns:
 - Grafos de Visibilidade;
 - Diagramas de *Voronoi*.



Construção de Rotas: *Road Map* Grafos de Visibilidade

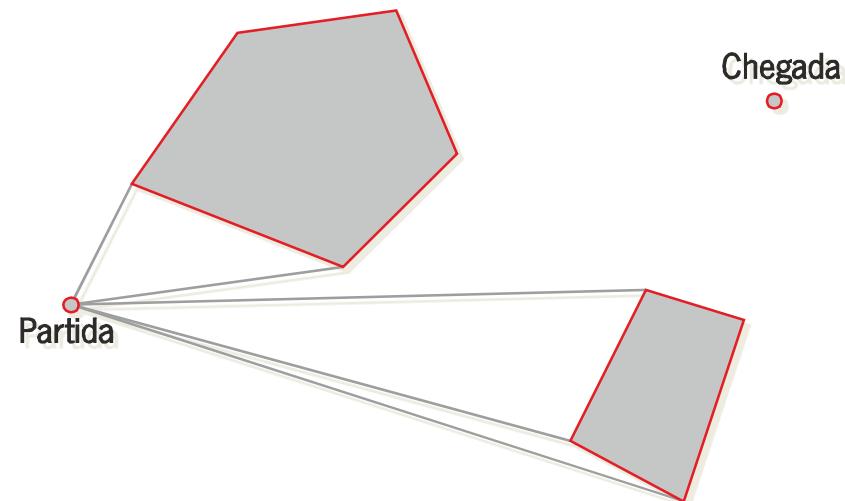
- Partindo de uma situação caracterizada por um ponto de partida, um ponto de chegada e uma descrição dos obstáculos;





Construção de Rotas: *Road Map* Grafos de Visibilidade

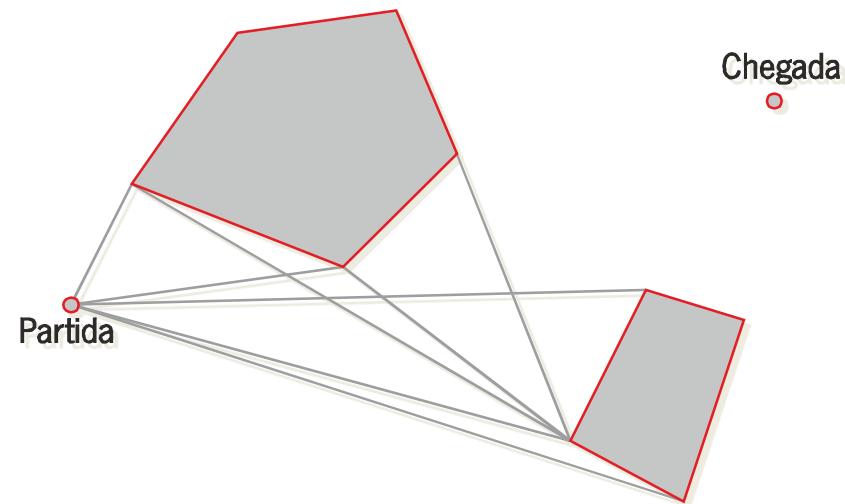
- ...
- Este método começa por determinar segmentos de reta entre a partida, a chegada e todos os **vértices visíveis** de todos os obstáculos;





Construção de Rotas: *Road Map* Grafos de Visibilidade

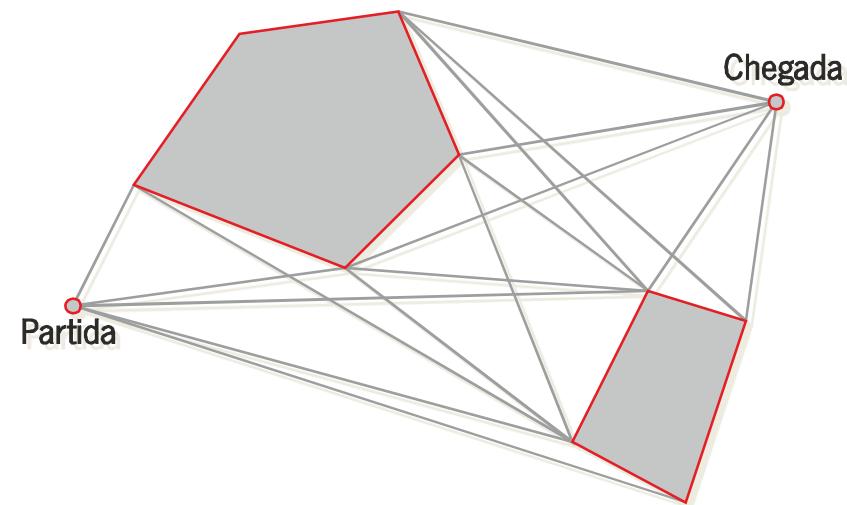
- ...
- Este método começa por determinar segmentos de reta entre a partida, a chegada e todos os **vértices visíveis** de todos os obstáculos;





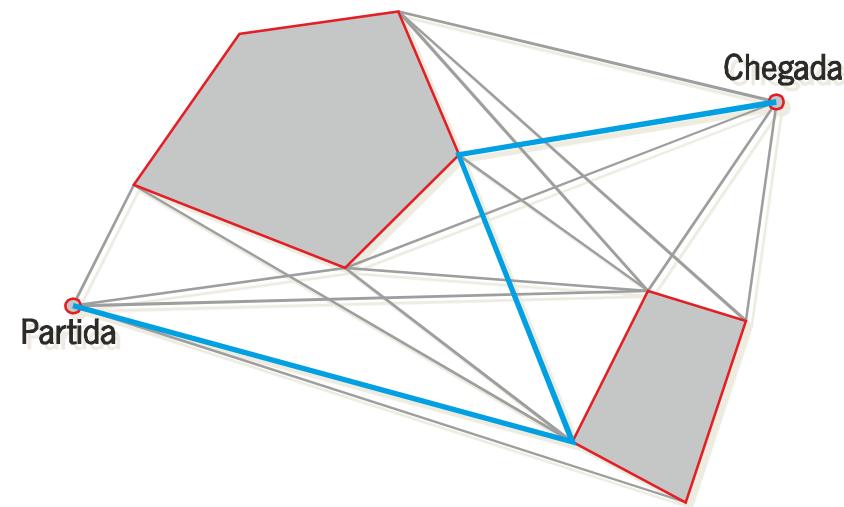
Construção de Rotas: *Road Map* Grafos de Visibilidade

- ...
- Este método começa por determinar segmentos de reta entre a partida, a chegada e todos os **vértices visíveis** de todos os obstáculos;



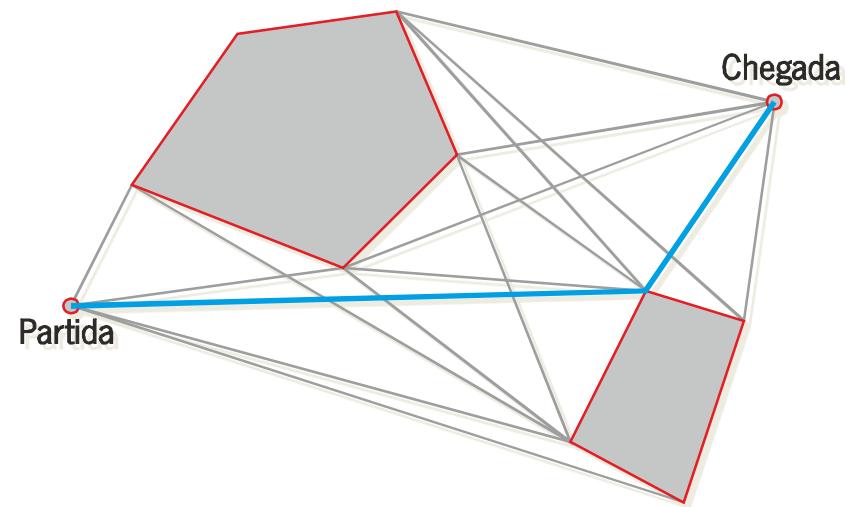
- ...
- ...
- De seguida, utiliza estes segmentos para calcular a melhor rota entre a partida e a chegada;

Construção de Rotas: *Road Map* Grafos de Visibilidade



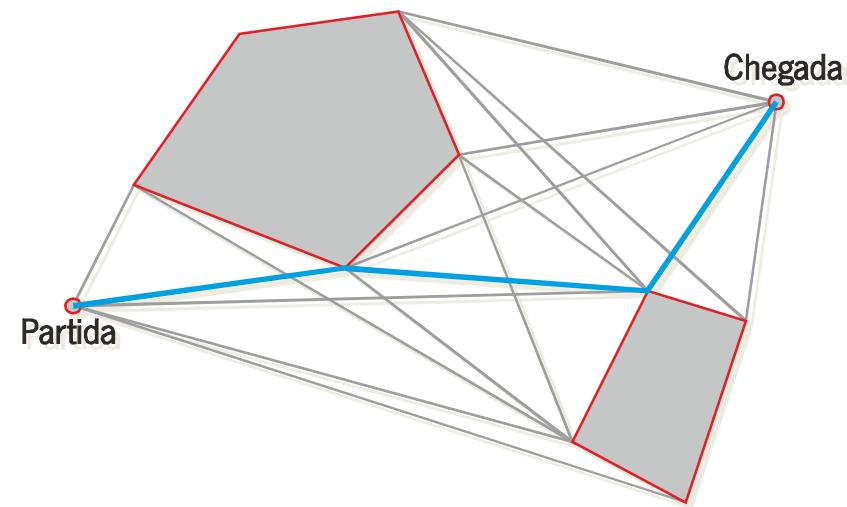
- ...
- ...
- De seguida, utiliza estes segmentos para calcular a melhor rota entre a partida e a chegada;

Construção de Rotas: *Road Map* Grafos de Visibilidade



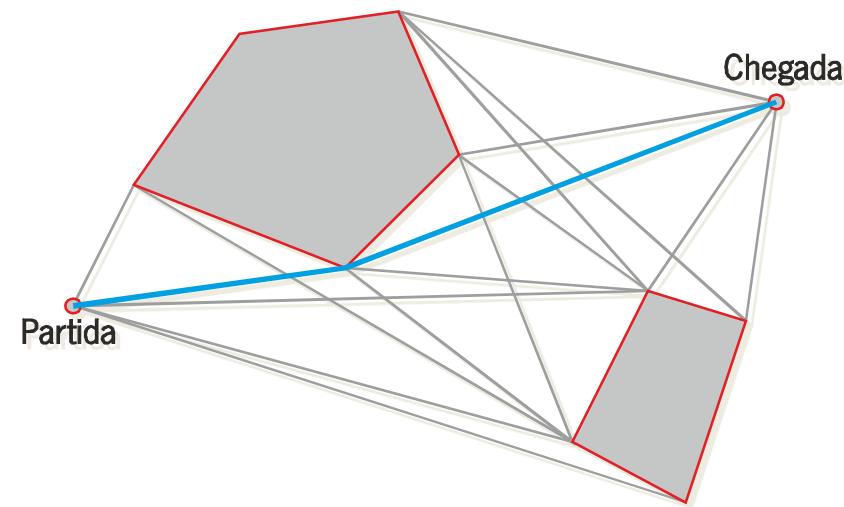
- ...
- ...
- De seguida, utiliza estes segmentos para calcular a melhor rota entre a partida e a chegada;

Construção de Rotas: *Road Map* Grafos de Visibilidade



- ...
- ...
- De seguida, utiliza estes segmentos para calcular a melhor rota entre a partida e a chegada;

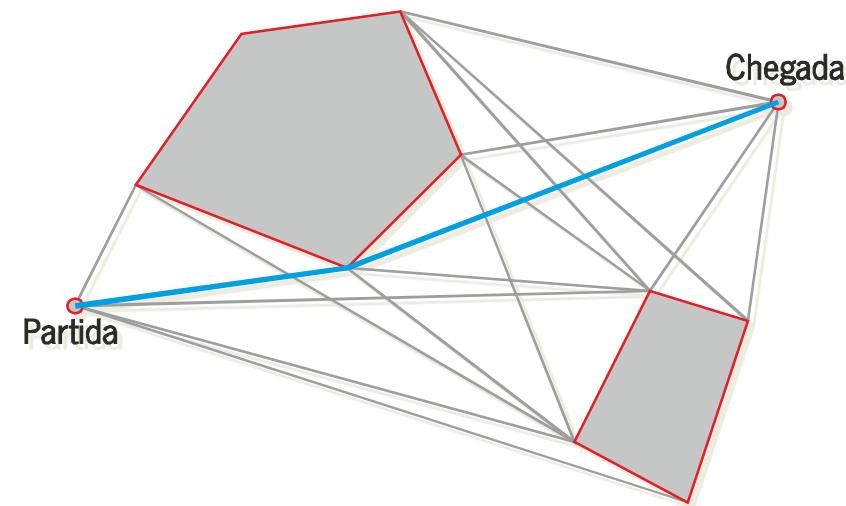
Construção de Rotas: *Road Map* Grafos de Visibilidade



Construção de Rotas: *Road Map* Grafos de Visibilidade

- Este método encontra o caminho mais curto;
- O caminho é calculado através de tangentes aos obstáculos;

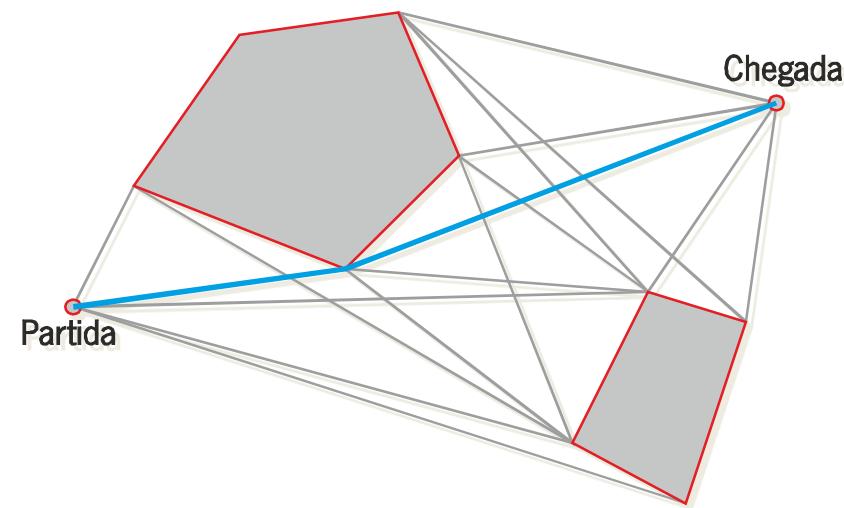
MAS...





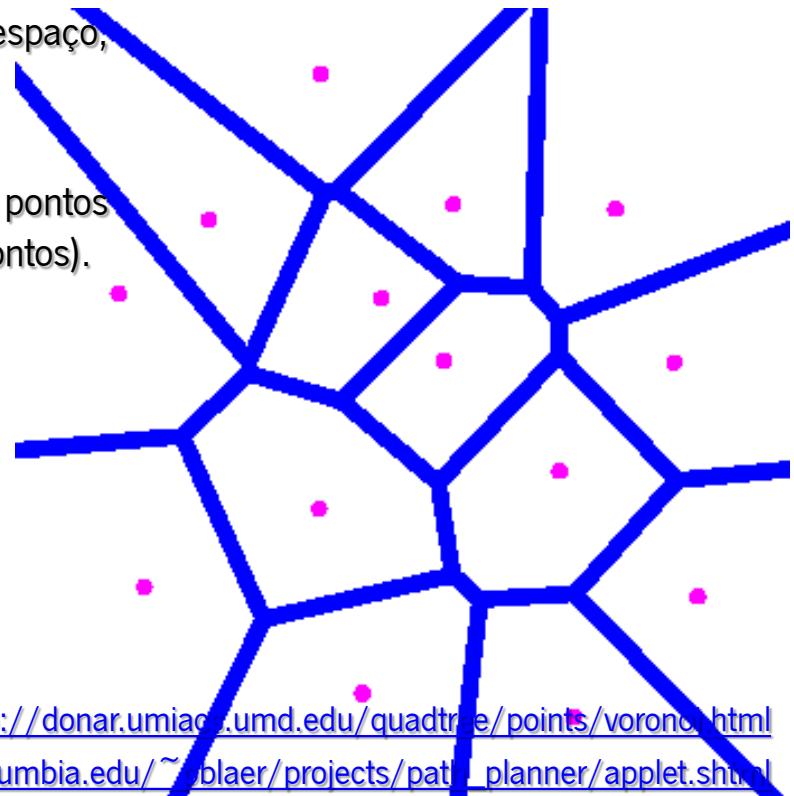
Construção de Rotas: *Road Map* Grafos de Visibilidade

- Este método encontra o caminho mais curto;
- O caminho é calculado através de tangentes aos obstáculos;
- Um pequeno erro de controlo ou de modelação dos obstáculos é um problema (grave?!);
- Este método não considera as dimensões do robô, o que significa que a rota calculada não evita colisões com os obstáculos.



Construção de Rotas: *Road Map* Diagramas de *Voronoi*

- Um Diagrama de *Voronoi* consiste numa decomposição do espaço, calculada através das distâncias entre si de um conjunto pré-definido de obstáculos (ou de pontos);
- Um Diagrama de *Voronoi* descreve as áreas formadas pelos pontos que estão mais próximos de um dado conjunto de locais (pontos).



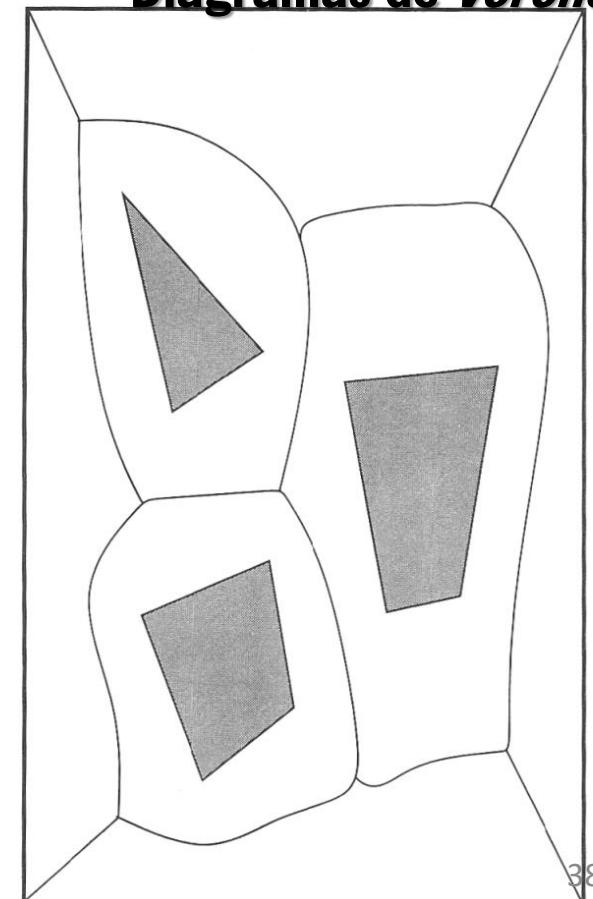
<http://donar.umiacs.umd.edu/quadrilateral/points/voronoi.html>

http://www.cs.columbia.edu/~blaer/projects/path_planner/applet.shtml



- Método semelhante ao método baseado no cálculo de Grafos de Visibilidade;
- Tende a maximizar a distância entre os obstáculos presentes no espaço;
- Por um lado, a rota traçada para o caminho a percorrer pelo robô é mais longa;
- Por outro lado, reduz a possibilidade de colisão com os obstáculos, uma vez que maximiza a distância entre eles.

Construção de Rotas: *Road Map* Diagramas de *Voronoi*



Construção de Rotas

- 
- A Construção de Rotas é importante para que um robô (sistema autónomo) possa mover-se ou realizar uma determinada tarefa **da forma mais eficaz e no menor período de tempo**, com o objetivo de o tornar **mais produtivo e fiável**;
 - Existem vários métodos para Construção de Rotas:
 - *Road Map*;
 - Decomposição Celular;
 - Navegação por Marcos de Influência.

Construção de Rotas: Decomposição celular

- A ideia chave subjacente a este método de construção de rotas é a de que é possível determinar áreas geométricas, **células**, de forma que umas representam **áreas livres** e outras representam **áreas ocupadas** (obstáculos);
- O espaço contínuo é dividido num número finito de células, por forma a possibilitar uma pesquisa discreta de soluções;
- A construção da rota é determinada pelo cálculo da ligação entre células adjacentes;

Construção de Rotas: Decomposição celular

- Decomposição Celular:

- Dividir o espaço em regiões simples;
- Construir um grafo de adjacências;
- Procurar um caminho entre as células de partida e de chegada;
- Construir a rota pela ligação entre as células do caminho encontrado.

- Existem dois tipos de métodos para decompor um espaço num conjunto de células:

- Decomposição Celular Exata;
- Decomposição Celular Aproximada.



- Dividir o espaço em regiões simples;
- Construir um grafo de adjacências;

Construção de Rotas: Decomposição celular Exata





- Dividir o espaço em regiões simples;
- Construir um grafo de adjacências;
- Procurar um caminho entre as células de partida e de chegada;

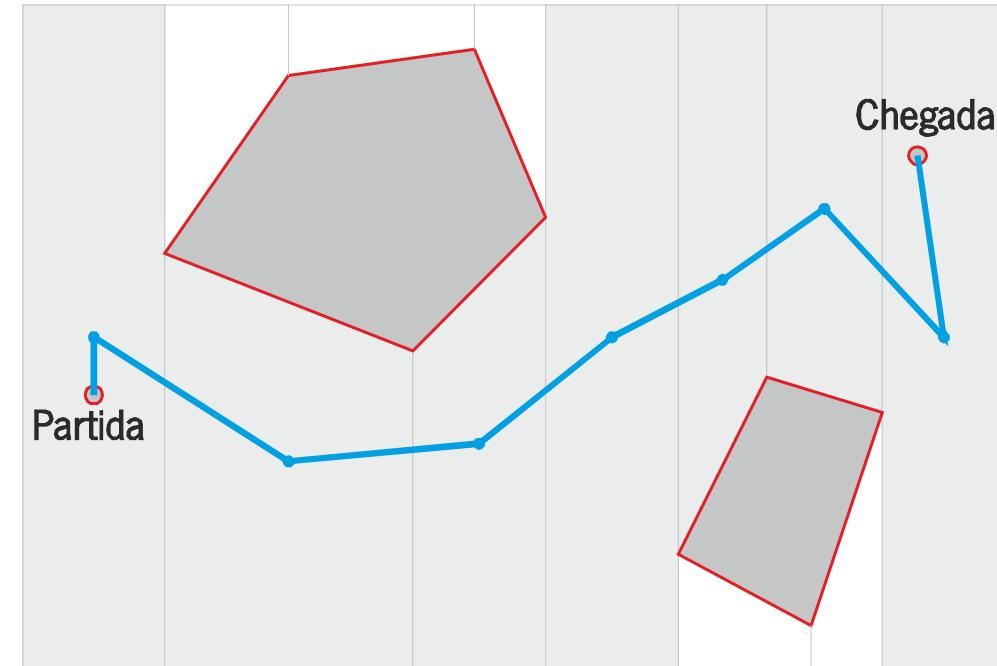
Construção de Rotas: Decomposição celular Exata





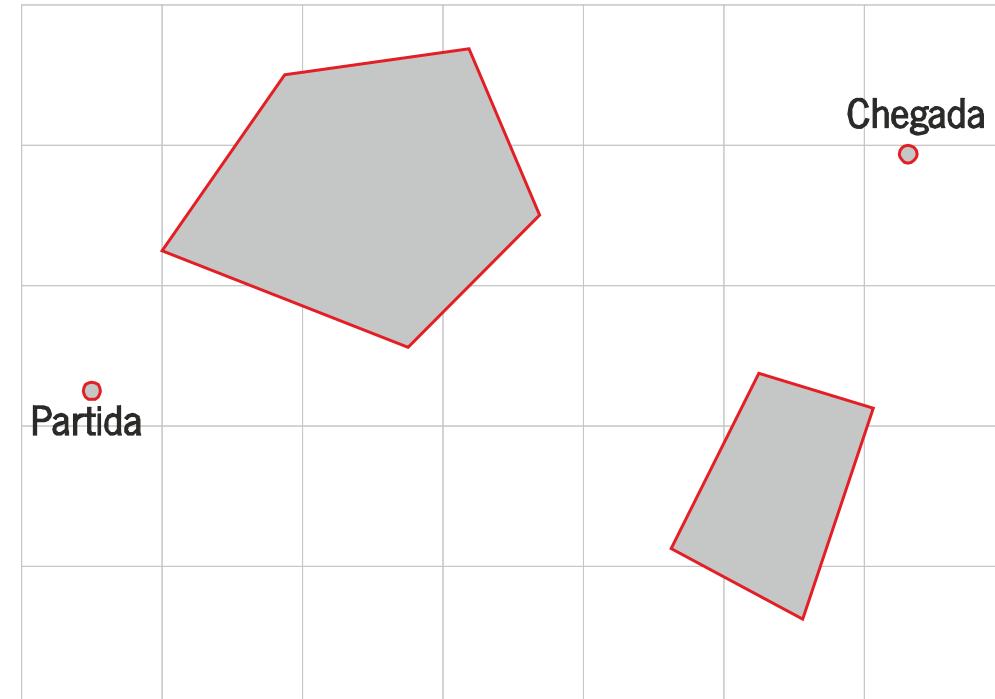
- Dividir o espaço em regiões simples;
- Construir um grafo de adjacências;
- Procurar um caminho entre as células de partida e de chegada;
- Construir a rota pela ligação entre as células do caminho encontrado.

Construção de Rotas: Decomposição celular Exata



- Definir uma grelha sobre o espaço de configuração;

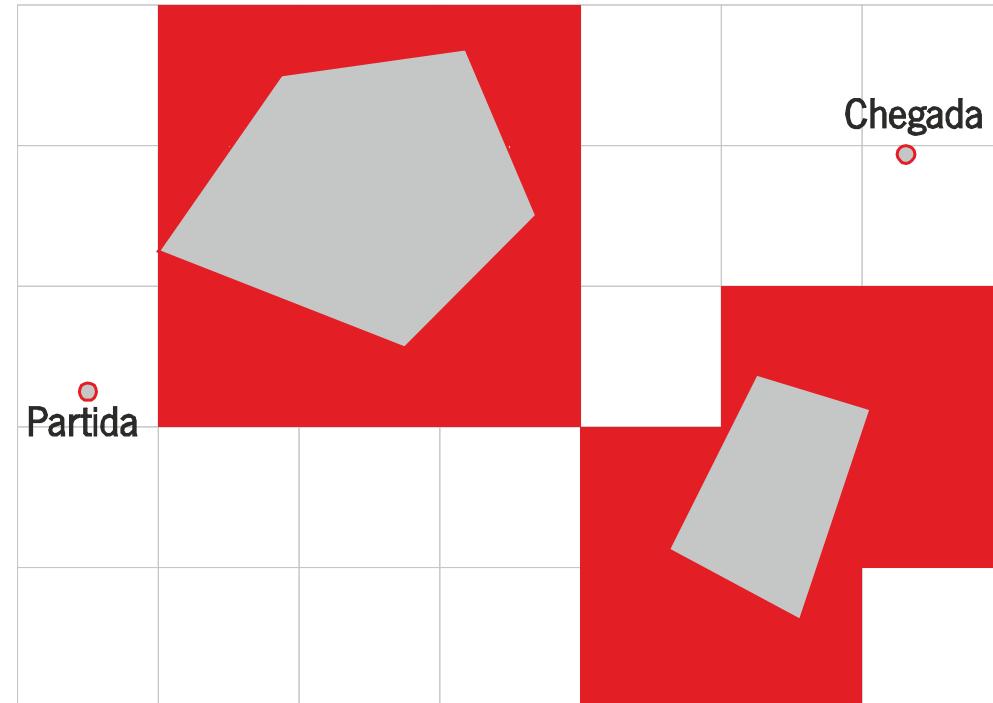
Construção de Rotas: Decomposição celular Aproximada





- Definir uma grelha sobre o espaço de configuração;
- Calcular o caminho mais curto entre a partida e a chegada através destas células, evitando qualquer das células ocupadas;

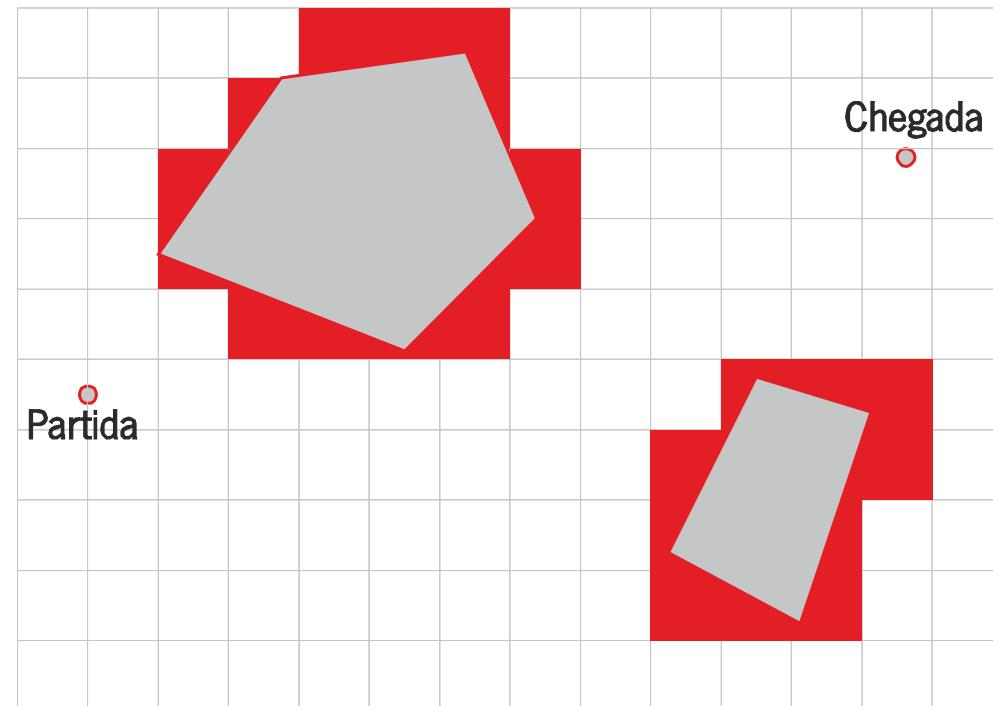
Construção de Rotas: Decomposição celular Aproximada





- Definir uma grelha sobre o espaço de configuração;
- Calcular o caminho mais curto entre a partida e a chegada através destas células, evitando qualquer das células ocupadas;
- Se não for possível calcular qualquer caminho, aumentar o detalhe da grelha e... continuar!

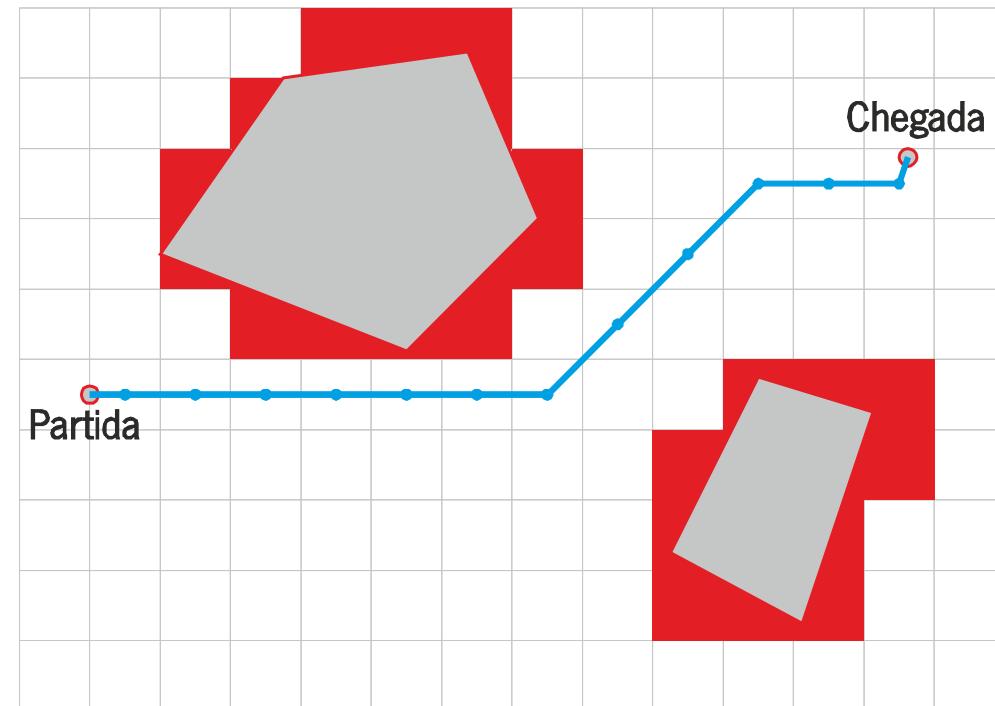
Construção de Rotas: Decomposição celular Aproximada





- Definir uma grelha sobre o espaço de configuração;
- Calcular o caminho mais curto entre a partida e a chegada através destas células, evitando qualquer das células ocupadas;
- Se não for possível calcular qualquer caminho, aumentar o detalhe da grelha e... continuar!

Construção de Rotas: Decomposição celular Aproximada



Construção de Rotas

- 
- A Construção de Rotas é importante para que um robô (sistema autónomo) possa mover-se ou realizar uma determinada tarefa **da forma mais eficaz e no menor período de tempo**, com o objetivo de o tornar **mais produtivo e fiável**;
 - Existem vários métodos para Construção de Rotas:
 - *Road Map*,
 - Decomposição Celular;
 - Navegação por Marcos de Influência.

Construção de Rotas: Navegação por marcos de influência

- Definem-se funções que:
 - assumem valores grandes (crescentes) à medida que o robô se aproxima de um objeto/obstáculo; (ou em torno da origem)
 - assumem valores pequenos (decrescentes) à medida que o robô se aproxima do objetivo/destino;
- para uma função objetivo que minimiza o custo do trajeto.

Como fazer Planeamento?

- Construção de Rotas:
 - *Road Map*:
 - Grafos de Visibilidade;
 - Diagramas de *Voronoi*;
 - Decomposição Celular:
 - Exata;
 - Aproximada;
 - Navegação por Marcos de Influência (ou campos gravitacionais);
- Desvio de Obstáculos;
- Arquiteturas de Navegação.

Desvio de obstáculos

- O Desvio de Obstáculos é a capacidade de um robô (sistema autónomo) mudar de trajetória durante o movimento, de acordo com a informação recolhida através dos sensores;
- Os algoritmos de desvio de obstáculos mais conhecidos são:
 - *Bug Algorithm*;
 - Técnica da Bolha (*Bubble Band*).

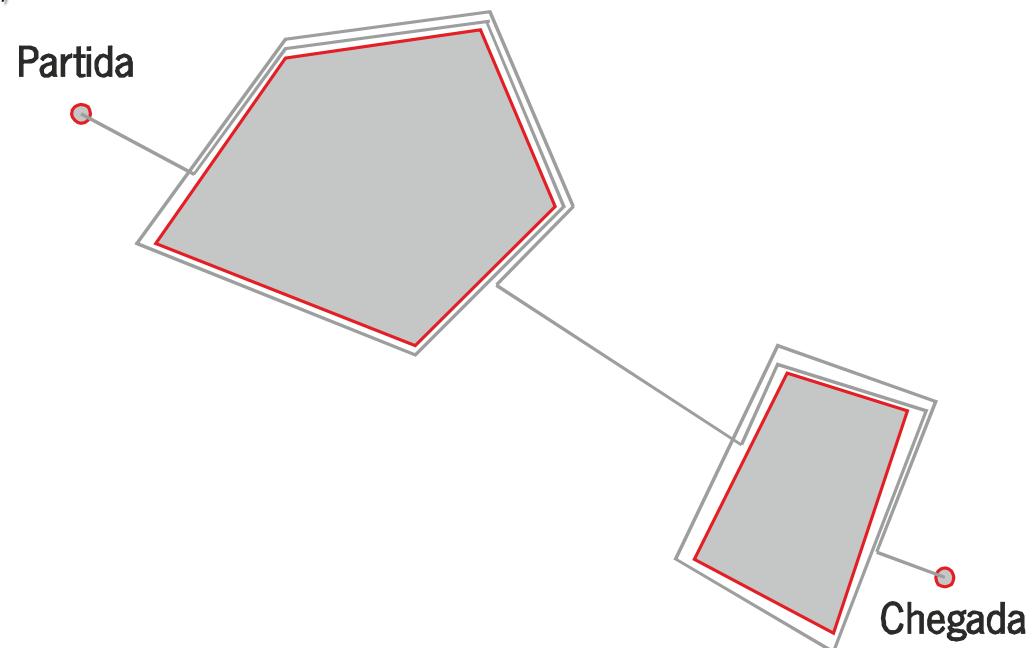
Desvio de obstáculos: ***Bug Algorithm***

- O *Bug Algorithm* implementa a estratégia mais simples para evitar obstáculos;
- A ideia básica é a de contornar cada obstáculo que se encontra na rota entre a partida e a chegada, por forma a “circum-navegar” todos os objetos no caminho do robô;



- O robô contorna, por completo, cada obstáculo;
- Seguidamente, circula o obstáculo por forma a chegar ao ponto do contorno que está mais próximo da chegada;
- Parte deste último ponto em direção ao ponto de chegada.

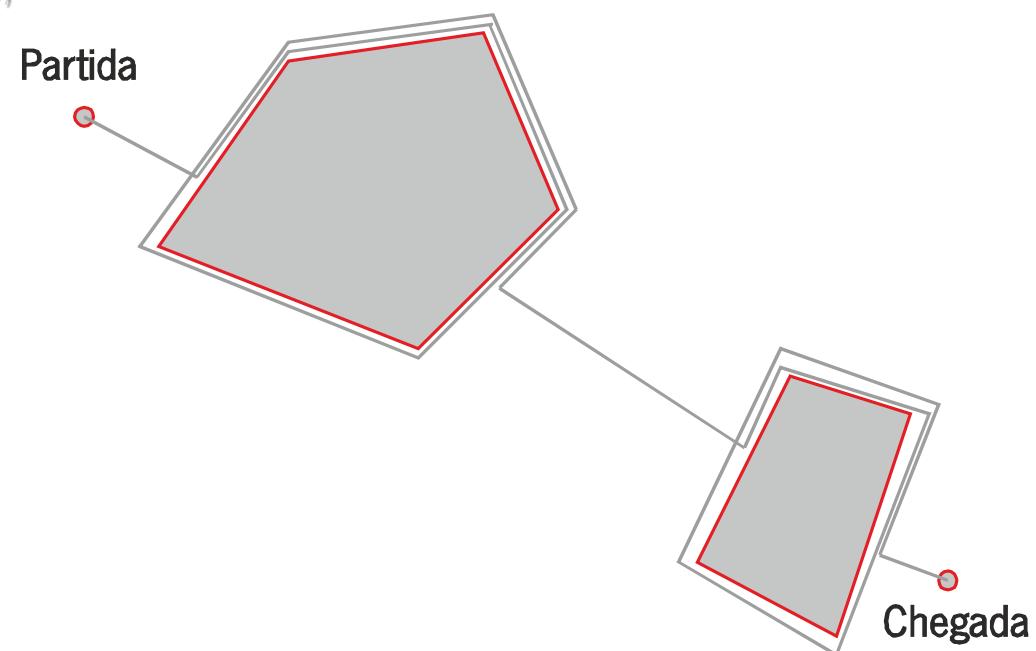
Desvio de obstáculos: *Bug Algorithm - v.1*





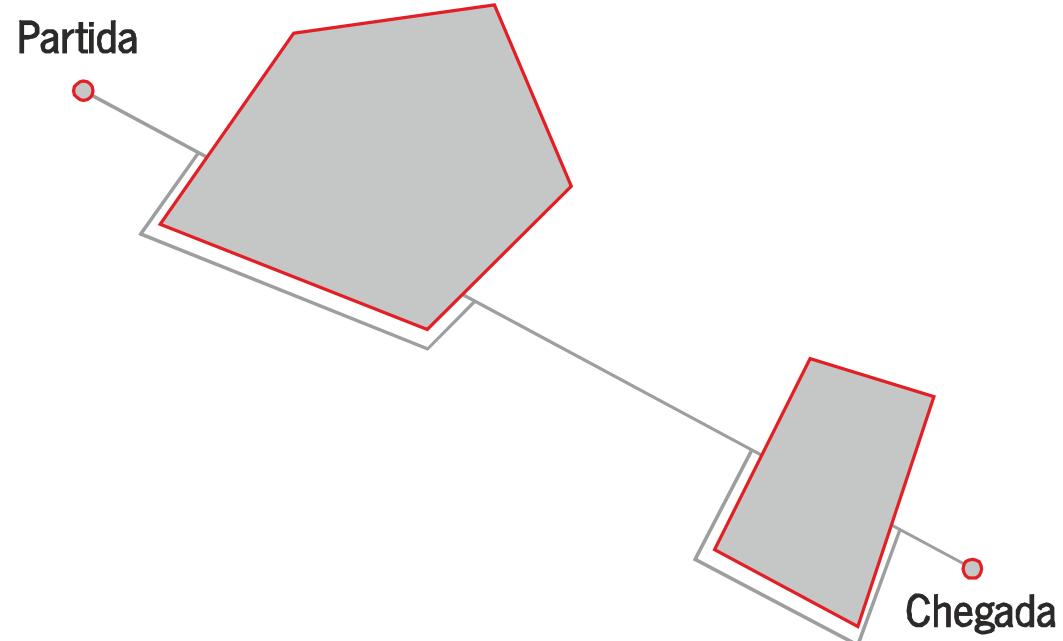
- O robô contorna, por completo, cada obstáculo;
- Seguidamente, circula o obstáculo por forma a chegar ao ponto do contorno que está mais próximo da chegada;
- Parte deste último ponto em direção ao ponto de chegada.
- Qual é o problema desta abordagem? ?

Desvio de obstáculos: *Bug Algorithm - v.1*



- O robô começa por contornar o obstáculo;
- Logo que encontra um ponto da rota inicialmente traçada, retoma o caminho projetado.

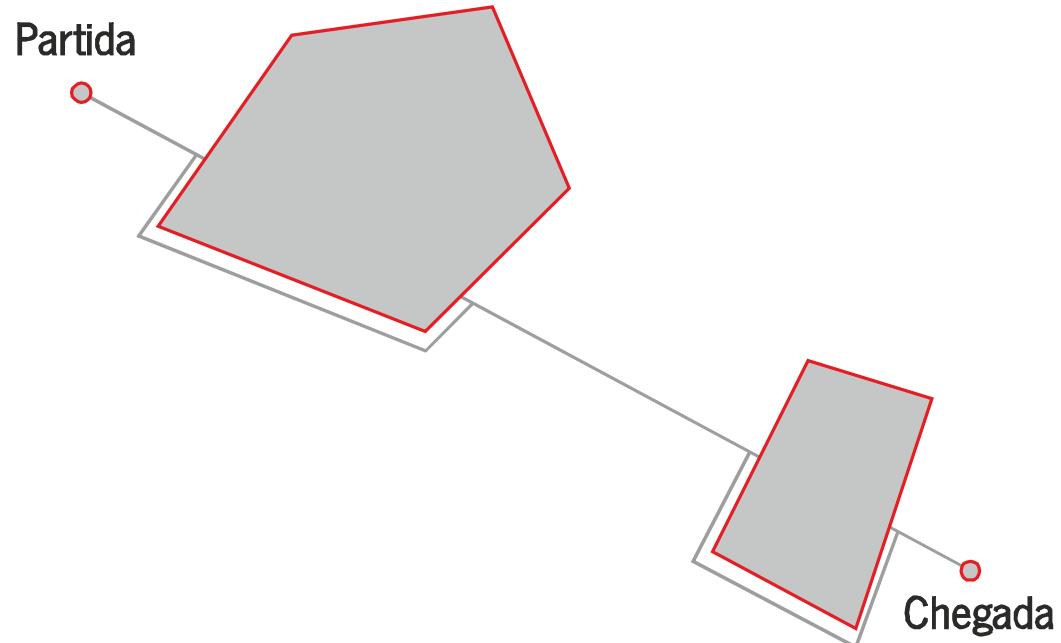
Desvio de obstáculos: *Bug Algorithm - v.2*





- O robô começa por contornar o obstáculo;
 - Logo que encontra um ponto da rota inicialmente traçada, retorna o caminho projetado.
-
- Qual é o problema desta abordagem? **?**

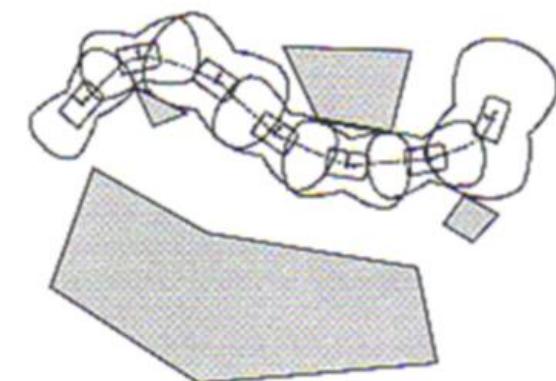
Desvio de obstáculos: *Bug Algorithm - v.2*





- Define-se uma “bolha” em torno do robô como sendo o máximo espaço livre que deve existir para uma dada configuração do robô, de modo a que não exista qualquer possibilidade de colisão quando se movimenta em qualquer direção;
- A rota é determinada garantindo que a “bolha” percorre a rota sem tocar em qualquer obstáculo;
- Utilizado quando um robô não pode ser considerado sem dimensão (ponto).

Desvio de obstáculos: *Bubble Band*



A typical bubble band (Courtesy of Raja Chatila)

Como fazer Planeamento?

- Construção de Rotas;
- Desvio de Obstáculos:
 - *Bug Algorithm* ;
 - Técnica da Bolha (*Bubble Band*).
- Arquiteturas de Navegação.

Arquiteturas de Navegação

- Tendo planeado a **trajetória** e selecionado o método para o **desvio de obstáculos**, como combinar estes dois sistemas num sistema autónomo, capaz de **navegar num ambiente** real (ou virtual)?
- Algumas arquiteturas de navegação são:
 - Planeamento *offline*;
 - Planeamento episódico;
 - Planeamento e execução integrados.

Arquiteturas de Navegação: Planeamento *offline*

- O **Planeamento da Navegação Offline** significa que não existe integração entre o planeamento da trajetória e a sua execução por parte do sistema autónomo;

-
-
-
-



Arquiteturas de Navegação: Planeamento *offline*

- O **Planeamento da Navegação Offline** significa que não existe integração entre o planeamento da trajetória e a sua execução por parte do sistema autónomo;
- Requer uma representação interna da trajetória e do mundo para a navegação desde a partida até à chegada;
- Não se pode deslocar para um ambiente desconhecido, sem necessidade de reprogramação do sistema de navegação;
- Utilizado na indústria;
- Utilizado em viagens espaciais!
(devido ao atraso nas comunicações, entre 3 e 22 minutos.)

ROBOCODE

O **Planeamento da Navegação Offline** significa que não existe integração entre o planeamento da trajetória e a sua execução por parte do sistema autónomo;

- Requer uma representação interna da trajetória e do mundo para a navegação desde a partida até à chegada;
- Não se pode deslocar para um ambiente desconhecido, sem necessidade de reprogramação do sistema de navegação;
- Utilizado na indústria;
- Utilizado em viagens espaciais!
(devido ao atraso nas comunicações, entre 3 e 22 minutos.)

Arquiteturas de Navegação: Planeamento *offline*

Arquiteturas de Navegação: Planeamento episódico

- O **Planeamento episódico** é um dos métodos de navegação de robôs mais comum;
- Contém um plano de navegação gerado *offline* e armazenado internamente, mas tem capacidade de recalcular a rota;
-
-

Arquiteturas de Navegação: Planeamento episódico

- 
- O **Planeamento episódico** é um dos métodos de navegação de robôs mais comum;
 - Contém um plano de navegação gerado *offline* e armazenado internamente, mas tem capacidade de recalcular a rota;
 - Existência suficiente de dados recolhidos do ambiente, para recalcular a rota;
 - Recacular frequentemente a rota pode ser um problema:
 - Consumo de recursos;
 - Comunicação entre sistemas;
 - Principalmente, quanto maior for o peso de uma arquitetura deliberativa para a resolução deste problema.

ROBOCODE

Planeamento episódico é um dos métodos de navegação de robôs mais comum;

- Contém um plano de navegação gerado *offline* e armazenado internamente, mas tem capacidade de recalcular a rota;
- Existência suficiente de dados recolhidos do ambiente para recalcular a rota;
- Recacular frequentemente a rota pode ser um problema:
 - Consumo de recursos;
 - Comunicação entre sistemas;
 - Principalmente, quanto maior for o peso de uma arquitetura deliberativa para a resolução deste problema.

Arquiteturas de Navegação: Planeamento episódico

Arquiteturas de Navegação: Planeamento e execução integrados

- Este método resulta de uma combinação dos dois anteriores;
- Combina rotas pré-determinadas com informação local do ambiente, por forma a que, quando detete obstáculos inesperados, em vez de recalcular uma rota, seleciona uma rota previamente calculada e otimizada;
-
-
-

Arquiteturas de Navegação: Planeamento e execução integrados

- 
- Este método resulta de uma combinação dos dois anteriores;
 - Combina rotas pré-determinadas com informação local do ambiente, por forma a que, quando detete obstáculos inesperados, em vez de recalcular uma rota, seleciona uma rota previamente calculada e otimizada;
 - Torna o processamento mais rápido e fiável;
 - Pode ser computacionalmente exigente;
 - Pouco útil em ambientes muito complexos.

Como fazer Planeamento?

- Construção de Rotas;
- Desvio de Obstáculos;
- Arquiteturas de Navegação:
 - Planeamento *offline* ;
 - Planeamento episódico;
 - Planeamento e execução integrados.

Universidade do Minho

Escola de Engenharia

Departamento de Informática

A white and teal 3D rendering of a robot's head and upper torso. The robot has a single visible eye and a small mouth. It wears a teal necktie and a teal circular badge on its chest.

RoboCode

Programação de Robôs

Sistemas Autónomos

Perfil Sistemas Inteligentes @ MEI/MiEI 1º/4º – 2º semestre

Cesar Analide, Bruno Fernandes

Programação RoboCode

- Que fazer?
- Que tanques construir?
- Que comportamentos desenvolver?
- Como?

Programação RoboCode

- Que fazer?
 - Programação de robôs;
 - robot, advanced robot, teamrobot, droid;
- Que tanques construir?
- Que comportamentos desenvolver?
- Como?

Programação RoboCode

- Que fazer?
- Que tanques construir?
 - Programação de tanques/robôs individuais;
 - Programação de equipas de robôs;
- Que comportamentos desenvolver?
- Como?

Programação RoboCode

- 
- Que fazer?
 - Que tanques construir?
 - Que comportamentos desenvolver?
 - Desafio 1: desenvolver um odômetro (medir a distância percorrida):
 - Medir a distância percorrida em cada episódio (*round*) de uma batalha (*battle*);
 - Calcular o acumulado das distâncias de todos os episódios;
 - Neste desafio, não há necessidade de combate;
 - Como?

Programação RoboCode

- 
- Que fazer?
 - Que tanques construir?
 - Que comportamentos desenvolver?
 - Desafio 1: desenvolver um odômetro (medir a distância percorrida);
 - Desafio 2: trajetória de circum-navegação de 3 obstáculos;
 - Obstáculos identificados como «rockquad»;
 - Início em 30x30;
 - Circular no sentido dos ponteiros do relógio;
 - Circular por fora da área demarcada pelos 3 obstáculos.
 - Como?

Programação RoboCode

- Que fazer?
- Que tanques construir?
- Que comportamentos desenvolver?
- Como?
 - Estratégias de controlo;
 - Arquiteturas de controlo;
 - Traçado de rotas/trajetórias.

Bibliografia

- Jean-Claude Latombe, “Robot Motion Planning”, Kluwer, 1991.
- Choset, Lynch, Hutchinson, Kantor, Burgard, Thrun, “Principles of Robot Motion”, 2005.
- Siegwart and Nourbakhsh, “Autonomous Mobile Robots”, The MIT Press, 2004.
- Ronald Arkin, “Behavior Based Robotics”, The MIT Press, 1998.
- Steven LaValle, “Planning Algorithms”, Cambridge University Press, 2006.

Universidade do Minho

Escola de Engenharia

Departamento de Informática

Sistemas Autónomos

Perfil Sistemas Inteligentes @ MEI/MiEI 1º/4º – 2º semestre

Cesar Analide, Bruno Fernandes