Instituto Tecnológico de Aeronáutica – ITA
Planejamento e Controle para Robótica Móvel – CM-202
Laboratório 1 – Campos Potenciais

Professor: Marcos Ricardo Omena de Albuquerque Maximo

10 de agosto de 2021

Observação: por questões de compatibilidade, este laboratório deve ser feito utilizando MATLAB 2021a.

1 Introdução

Neste laboratório, implementa-se planejamento de caminhos usando a técnica de campos potenciais. Seja $\mathbf{q} = [x\ y]^T$ a posição de um robô móvel, define-se um campo potencial $U(\mathbf{q}) : \mathbb{R}^2 \to \mathbb{R}$ para guiar o movimento do robô. Para isso, o robô move-se de acordo com o vetor

$$\mathbf{F}(\mathbf{q}) = -\nabla U(\mathbf{q}). \tag{1}$$

No caso da implementação realizada neste laboratório, considera-se que o vetor $\mathbf{F}(\mathbf{q})$ define diretamente a velocidade do robô:

$$\mathbf{v} = \mathbf{F}(\mathbf{q}) = -\nabla U(\mathbf{q}). \tag{2}$$

O campo potencial total pode ser dividido entre parcelas atrativas e repulsivas de acordo com

$$U(\mathbf{q}) = U_{att}(\mathbf{q}) + \sum_{i=1}^{N_{obs}} U_{rep,i}(\mathbf{q}), \tag{3}$$

em que $U_{att}(\mathbf{q})$ é um campo que atrai para o objetivo e $U_{rep,i}(\mathbf{q})$ é o campo repulsivo associado ao i-ésimo obstáculo. Ademais, N_{obs} indica o número de obstáculos presentes. Aplicando-se o operador gradiente em (3), obtém-se

$$\nabla U(\mathbf{q}) = \nabla U_{att}(\mathbf{q}) + \sum_{i=1}^{N_{obs}} \nabla U_{rep,i}(\mathbf{q}). \tag{4}$$

Como potencial atrativo, combina-se um potencial cônico com um parabólico de acordo com

$$U_{att}(\mathbf{q}) = \begin{cases} \frac{1}{2} k_{att} d^2(\mathbf{q}),\ d(\mathbf{q}) \leq d_{0,att}, \\ d_{0,att} k_{att} d(\mathbf{q}) - \frac{1}{2} k_{att} d_{0,att}^2,\ d(\mathbf{q}) > d_{0,att}, \end{cases} \tag{5}$$

em que k_{att} é uma constante, $d_{0,att}$ define a distância em que o potencial muda de parabólico para cônico e $d(\mathbf{q}) = \|\mathbf{q} - \mathbf{q}_{goal}\|$ é a distância do robô até a posição do objetivo $\mathbf{q}_{goal}$. Com isso, o gradiente do potencial atrativo fica

$$\nabla U_{att}(\mathbf{q}) = \begin{cases} k_{att}(\mathbf{q} - \mathbf{q}_{goal}), \ d(\mathbf{q}) \leq d_{0,att}, \\ d_{0,att} k_{att} \dfrac{\mathbf{q} - \mathbf{q}_{goal}}{d(\mathbf{q})}, \ d(\mathbf{q}) > d_{0,att}. \end{cases} \tag{6}$$

O potencial repulsivo para um obstáculo é dado por

$$U_{rep}(\mathbf{q}) = \begin{cases} \dfrac{1}{2} k_{rep} \left(\dfrac{1}{d(\mathbf{q})} - \dfrac{1}{d_{0,rep}} \right)^2, \ d(\mathbf{q}) \leq d_{0,rep}, \\ 0, \ d(\mathbf{q}) > d_{0,rep}, \end{cases} \tag{7}$$

em que, neste caso, $d(\mathbf{q}) = \|\mathbf{q} - \mathbf{q}_{obs}\|$ representa a distância do robô até a posição do obstáculo $\mathbf{q}_{obs}$. Portanto, o gradiente do potencial repulsivo é

$$\nabla U_{rep}(\mathbf{q}) = \begin{cases} k_{rep} \left(\dfrac{1}{d_{0,rep}} - \dfrac{1}{d(\mathbf{q})} \right) \dfrac{1}{d^2(\mathbf{q})} \dfrac{\mathbf{q} - \mathbf{q}_{obs}}{d(\mathbf{q})}, \ d(\mathbf{q}) \leq d_{0,rep}, \\ 0, \ d(\mathbf{q}) > d_{0,rep}. \end{cases} \tag{8}$$

2 Tarefas

Para facilitar a implementação do laboratório, foi entregue um código base com os seguintes arquivos:

- `AttractivePotentialField.m`*: contém classe que representa um potencial atrativo combinado.
- `RepulsivePotentialField.m`*: contém classe que representa um potencial repulsivo.
- `TotalPotentialField.m`: combina campos potenciais atrativos e repulsivos para calcular um campo potencial total.
- `PotentialFieldSimulator.m`: contém classe que simula um robô que se desloca sob influência de um campo potencial.
- `runSimulation.m`: executa uma simulação de campo potencial pré-definida.
- `PotentialFieldDrawer.m`: contém classe com métodos auxiliares para traçar gráficos relativos a campos potenciais.
- `drawArrow.m`: função auxiliar para traçar setas.

Na lista acima, os arquivos marcados com asterisco serão editados pelo aluno, enquanto os demais já estão completamente implementados e não precisam ser editados. Recomenda-se que o aluno inicialmente analise o código entregue para se ambientar. Ademais, recomenda-se ao aluno que leia a documentação de cada método para entender suas entradas e saídas.

2.1 Implementação dos Campos Potenciais Atrativo e Repulsivo

Na classe `AttractivePotentialField`, implemente:

- `computePotential()`: calcula o potencial $U_{att}(\mathbf{q})$ do campo potencial atrativo combinado, segundo (5).
- `computeGradient()`: calcula o gradiente $\nabla U_{att}(\mathbf{q})$ do campo potencial atrativo combinado, segundo (6).

Analogamente, para a classe `RepulsivePotentialField`, implemente:

- `computePotential()`: calcula o potencial $U_{rep}(\mathbf{q})$ do campo potencial repulsivo, segundo (7). Considera a existência de um único obstáculo.
- `computeGradient()`: calcula o gradiente $\nabla U_{rep}(\mathbf{q})$ do campo potencial repulsivo, segundo (8). Considera a existência de um único obstáculo.

2.2 Simulação e Análise de Campos Potenciais

Para verificar a sua implementação de campos potenciais, use a função `runSimulation.m`. Esta função admite como argumento um caractere que representa qual simulação será executada:

'a'. Robô começa em $\mathbf{q}_0 = [10\ 10]^T$, objetivo em $\mathbf{q}_{goal} = [5\ 5]^T$ e nenhum obstáculo.

'b'. Robô começa em $\mathbf{q}_0 = [10\ 10]^T$, objetivo em $\mathbf{q}_{goal} = [0\ 0]^T$ e conjunto de obstáculos pontuais dado por $\mathcal{O} = \left\{[8{,}1\ 8{,}5]^T, [8{,}4\ 6{,}5]^T, [4{,}5\ 5]^T, [5\ 1{,}1]^T\right\}$.

'c'. Robô começa em $\mathbf{q}_0 = [5\ 10]^T$, objetivo em $\mathbf{q}_{goal} = [5\ 0]^T$ e conjunto de obstáculos pontuais dado por $\mathcal{O} = \left\{[4{,}6\ 5]^T, [5{,}4\ 5]^T\right\}$.

Além disso, os campos potenciais são configurados com os parâmetros descritos na Tabela 1. Por fim, executa-se a simulação por 10 s com passo $\Delta t = 0{,}1$ s.

Parâmetro	**Valor**
k_{att}	1
$d_{0,att}$	3
k_{rep}	1
$d_{0,rep}$	3

Tabela 1: Parâmetros usados para configurar os campos potenciais.

Perceba que, em código, o conjunto de obstáculos $\mathcal{O} = \left\{[x_1\ y_1]^T, [x_2\ y_2]^T, ..., [x_{N_{obs}}\ y_{N_{obs}}]^T\right\}$ é representado pela tabela

$$\mathbf{O} = \begin{bmatrix} x_1 & x_2 & x_3 & \cdots & x_{N_{obs}} \\ y_1 & y_2 & y_3 & \cdots & y_{N_{obs}} \end{bmatrix}. \tag{9}$$

Considere que as unidades da simulação são arbitrárias e que não há um limite de velocidade para o robô. Ademais, assume-se que o robô é capaz de alterar sua velocidade instantaneamente (como se sua inércia fosse desprezível).

Com base nos resultados obtidos nas simulações:

(a) Analise os resultados das simulações e descreva qualitativamente o que observa.

(b) Com base no que foi discutido em sala, explique o que acontece na simulação 'c'.

Além disso, inclua os gráficos gerados no seu relatório. Observe que os gráficos são salvos em arquivos automaticamente na função `runSimulation()`.

3 Instruções

- A entrega da solução desse laboratório consiste de arquivos de código (MATLAB e Simulink) e de um relatório (em `.pdf`), que devem ser submetidos no Google Classroom como um único arquivo `.zip`.
- Compactar todos os arquivos a serem submetidos em um único `.zip` (use obrigatoriamente `.zip`, e **não** outra tecnologia de compactação de arquivos) e anexe esse `.zip` no Google Classroom.
- Use o padrão de nome `<login_ga>_labX.zip`, em que `<login_ga>` é seu login `@ga.ita.br` no Classroom e X é o número do laboratório em questão. Por exemplo, se o login do aluno for `marcos.maximo` e estiver entregando o laboratório 1, o nome do arquivo deve ser `marcos.maximo_lab1.zip`.
- O relatório deve ser sucinto, preocupe-se apenas em incluir discussões e entregáveis solicitados no roteiro. Pede-se apenas um cuidado mínimo na elaboração do relatório: responder adequadamente as perguntas, incluir figuras diretamente no relatório (ao invés de deixar como arquivos separados), usar figuras de boa qualidade, colocar nomes nos eixos dos gráficos, colocar legenda para diferenciar curvas num mesmo gráfico etc.
- **Não** é permitido o uso de funções ou comandos prontos do MATLAB que realizem toda a funcionalidade atribuída a uma certa função cuja implementação foi solicitada. Entretanto, o uso destas funções para verificação das implementações realizadas é encorajado. Em caso de dúvida, consulte o professor.
- A criação de *scripts* e funções auxiliares no MATLAB para execução de experimentos e geração de gráficos é fortemente recomendada para facilitar seu trabalho. Porém, não há necessidade de entregar códigos auxiliares.
- **Não** há necessidade de copiar e colar o código no seu relatório, dado que você também submeterá os arquivos de código. Também **não** há necessidade de explicar sua implementação no relatório, a **não** ser que o roteiro tenha solicitado explicitamente. Porém, organizar e comentar o código é muito salutar, pois ele será o foco da correção.

4 Dicas

- Para calcular a norma do vetor `v`, use `norm(v)`.
- Este código utiliza Programação Orientada a Objetos (POO) em MATLAB. Embora POO seja uma técnica de programação complexa, o uso neste laboratório é bem básico. Caso não saiba como usar POO em MATLAB, recomenda-se estudar o *tutorial* entregue como Material Complementar.