CiberRato Assignment 1

Rodrigo Santos and Rui Santos

Universidade de Aveiro, 3810-193 Aveiro

rodrigo.l.silva.santos@ua.pt, ruipedro99@ua.pt

Abstract. Para o desenvolvimento do Assigment 1 foram aplicadas diferentes estratégias para diferentes challenges. Para o challenge 1, o objetivo era ser rápido, e percorrer o percurso fechado o maior número de vezes, logo optou-se sempre que possível fornecer a máxima potência aos motores e descrever as mudanças de direção em andamento. No challenge 2, sendo o objetivo o de mapear o mapa na sua totalidade e tendo em conta que o robô tinha acesso a um GPS e uma bússola, a sua movimentação foi baseada nestas duas últimas ferramentas, para pesquisa do mapa foi aplicado um algoritmo de pesquisa A*. Para desenvolver o challenge 3 e ir de encontro ao objetivo de procurar os dois *targets* no mapa e calcular o melhor caminho fechado que contenha o ponto inicial e estes dois pontos desconhecidos, adaptou-se a estratégia do challenge 2 e foi apenas adicionada lógica de modo a armazenar as posições dos pontos desconhecidos e calcular quando possível o melhor caminho desejado. Desenvolvendo e concluindo assim os 3 challenges com sucesso.

Keywords: CiberRato, Challenge, Labirinto, Estratégia, Sensores, Bússola, GPS, Robô.

1 Introdução

Neste relatório serão abordadas as diferentes estratégias aplicadas para a concretização de cada Challenge do projeto Assignment 1, da cadeira de Robótica Móvel Inteligente, do 5º ano do Mestrado Integrado em Engenharia de Computadores e Telemática, da Universidade de Aveiro.

Tendo em conta os diferentes objetivos dos 3 Challenges foram aplicadas diferentes estratégias.

No primeiro Challenge, o objetivo era percorrer o labirinto o mais rapidamente possível. Optou-se assim, por uma estratégia que consistia em aplicar sempre que possível a máxima potência aos motores do robô, e fazer as mudanças de direção em movimento.

No segundo Challenge, o objetivo era percorrer todo o labirinto, mapeando-o, e escrever esse mapeamento num ficheiro. Para satisfazer este objetivo, foi necessário implementar um algoritmo de pesquisa para o cálculo de caminhos para posições que ainda não tivessem sido exploradas.

Por fim, o terceiro Challenge tinha como objetivo encontrar o melhor caminho fechado, entre 3 pontos, o ponto de início do labirinto, e dois pontos cuja localização

é inicialmente desconhecida no mapa. Neste challenge optou-se por fazer um mecanismo de pesquisa semelhante ao Challenge 2, no qual após descobertos os dois pontos desconhecidos, é calculado se possível o melhor caminho fechado que os atravesse contendo também o ponto de início.

2 Challenge 1

Para o Challenge 1 o objetivo era percorrer o labirinto o mais rapidamente possível, fazendo assim o maior número de voltas possível, sem colidir com as paredes. Neste challenge todo o movimento tem de ser controlado apenas pelos sensores que o robô possui, não lhe sendo disponibilizadas informações relativas à sua orientação e localização.

Com isto em mente, a estratégia aplicada a este challenge baseia-se em aplicar, sempre que possível, a potência máxima aos motores do robô e fazer quaisquer mudanças de direção necessárias em movimento, perdendo assim o mínimo de tempo possível quando o robô tem de efetuar um movimento curvilíneo em torno de uma parede.

Uma vez que os motores do robô possuem ruído, a potência aplicada pode não ser exatamente a potência desejada, logo o robô pode não descrever o movimento pretendido. Por isso e para evitar colisões, em cada ciclo é feita a verificação da distância às paredes que rodeiam o robô, através do sensor frontal, sensor esquerdo e sensor direito. Visto que não se pretende andar para trás no percurso, o sensor traseiro não é utilizado. Caso o valor da distância a uma parede diminua, significa que o robô poderá colidir se continuar com o movimento que apresenta, nestas situações o movimento é ajustado para que o robô se afaste da parede, tentando assim mantê-lo centrado.

Sabendo que o labirinto do challenge 1 é um percurso fechado, as mudanças de direção são iniciadas quando o valor do sensor do lado interior da curva é menor do que um threshold, significando que o robô terá de descrever uma curva, na direção da parede em falta.

3 Challenge 2

No segundo Challenge, o objetivo era percorrer todo o labirinto, mapeando-o, e escrever o resultado num ficheiro. Neste challenge, para além dos sensores laterais com ruído, o robô possui também um GPS e uma bússola. Uma vez que estes não apresentam ruído, é possível conhecer a posição e direção exatas do robô. Deste modo, e tendo em conta o objetivo de mapear todo o labirinto, a estratégia aplicada, segue o conceito de percorrer o mapa de 2 em 2 unidades de GPS, parar, fazer o reconhecimento das paredes e espaços livres à volta do robô e decidir o que fazer de seguida.

3.1 **Mapa**

Na criação do mapa, foi utilizado um array bidimensional com 55 unidades de comprimento e 27 unidades de largura inicializado a 0. Para a determinação destes valores, foi necessário considerar que o robô poderia surgir em cada uma das pontas do mapa 29x15. Na sua construção, foram utilizados os valores obtidos pelos sensores do robô que foram posicionados a 0, 90, -90 e a 180 graus. Quando estes valores fossem superiores a 1, seria atribuída à posição correspondente do mapa 55x27 o símbolo de uma parede que podia ser "|" ou "-" dependendo da sua posição relativamente ao robô. Se os valores fossem inferiores a 1, seria atribuído um espaço (" ") à posição correspondente, simbolizando uma posição livre inexplorada. Foram também atribuídos "X" a todas as posições em que o robô passou.

Para armazenar o mapeamento feito no array bidimensional criado, é necessário traduzir as coordenadas GPS, em coordenadas do array, para este efeito foi criada uma variável global que guarda o fator da escala a ser usada. Tendo por base as coordenadas iniciais e sabendo que o array começará a ser escrito na posição (27, 13), a escala pode ser calculada subtraindo a posição inicial do array bidimensional à posição GPS inicial. Deste modo, numa função à parte é calculada a posição de cada coordenada GPS a traduzir para coordenadas no array de armazenamento, subtraindo à coordenada GPS a escrever, o valor da escala.

3.2 **Movimento**

Durante o movimento do robô, tal como no challenge 1, os motores têm ruído podendo alterar a trajetória desejada do robô. Para contornar este problema, foi criada uma função que verifica a direção de movimento pretendida. Se os valores de direção da bússola não estiverem dentro dos limites de -1 e +1 graus do ângulo ideal para o tipo de movimento atual, é feita uma correção de acordo com o desvio que o robô tem em relação à direção do movimento, certificando assim que o robô andará sempre na direção desejada.

O deslocamento do robô tem em conta a sua inércia, portanto, foi necessário adicionar um mecanismo que o previne de exceder a posição destinada. Tendo isto em conta, no movimento simples de 2 unidades de deslocamento, o robô dá o valor máximo de 0.15 a ambos os motores até a distância ao destino ser inferior a 0.15. Posteriormente, os valores dados aos motores são bastante mais reduzidos (0.02). O robô sai do estado de deslocamento apenas quando percorreu uma distância mínima de 1.8 unidades e as casas decimais da coordenada GPS (x ou y, dependendo da direção do deslocamento) onde se encontra são iguais às casas decimais da posição inicial (quando o robô se encontra exatamente no centro da célula) com uma margem de 0.05.

A rotação do robô segue a mesma estratégia do movimento simples, diminuindo o valor enviado aos motores quando o ângulo se aproxima do pretendido.

3.3 **Algoritmo de pesquisa**

Após o robô fazer a atualização do mapa com base nos sensores, deve prioritizar o deslocamento na direção dos espaços (“ ”) que representam células livres inexploradas, mais especificamente nos espaços à sua frente para não perder tanto tempo com tantas rotações. No caso do robô não ter um espaço a 2 unidades de distância, deve aplicar o algoritmo de pesquisa A* para encontrar o caminho mais curto entre a posição atual e a posição do espaço mais próximo. Quando não existirem mais espaços no mapa significa que a sua elaboração está concluída e o programa pode terminar.

Na aplicação do algoritmo de pesquisa A*, o mapa é transformado numa matriz de zeros e uns em que os zeros representam as posições onde o robô se pode deslocar e os uns representam as posições onde não se pode deslocar. Na elaboração deste mapa de input ao algoritmo de pesquisa foi definido que os zeros correspondiam às células onde o robô já esteve (“X”) e espaços (“ ”) e os uns correspondiam aos restantes símbolos do mapa. Após receber o percurso do caminho mais curto, este é transformado para uma *stack* de movimentos e o robô entra no estado de “consumo da stack”. Enquanto a stack tiver movimentos, o robô irá consumi-los e percorrer o percurso até chegar ao seu destino.

Este procedimento descrito nesta seção é repetido até que o mapa esteja completamente explorado, ou seja, não haja mais espaços por explorar no mapa. Por fim, é impresso o mapa obtido no ficheiro “mapping.out”.

4 Challenge 3

No terceiro Challenge o objetivo era encontrar o melhor caminho fechado entre 3 pontos: o ponto de início do labirinto, e dois pontos cuja localização é inicialmente desconhecida no mapa. Tal como no Challenge 2, o robô possui sensores com ruído, GPS e bússola, ambos sem ruído.

No desenvolvimento deste Challenge, grande parte da estratégia adotada segue o raciocínio do Challenge 2, como o movimento, a exploração e armazenamento do mapa, a correção das oscilações de direção devido ao ruído dos motores, entre outros.

Foi adicionada lógica que permite ler o valor de *measures.ground* e armazenar as posições dos pontos 1 e 2 a encontrar, quando no decorrer da exploração do mapa o valor de *measures.ground* é 1 ou 2 respectivamente.

A partir do momento em que os dois pontos são encontrados, é calculado o melhor caminho de Ponto Inicial -> Ponto 1 -> Ponto 2 -> Ponto Inicial sempre que o robô pára para “pensar”, até o mapa ser totalmente explorado. O cálculo do melhor caminho é feito com recurso ao mesmo algoritmo de pesquisa A* utilizado na exploração e sempre que este é encontrado, os passos são escritos num ficheiro “pathC3.out” que é atualizado a cada ciclo de pensamento, até não haverem mais espaços no mapa.

5 Conclusion

Com o desenvolvimento deste trabalho, pudemos observar num ambiente de treino bem concebido o poder dos algoritmos de pesquisa tais como o A*, utilizado para os challenges 2 e 3. Aprendemos também diversas estratégias para contornar alguns problemas de inércia de objetos em movimento e ruído na transferência de potência para os motores de robôs.

Todos os challenges foram desenvolvidos com sucesso, com o challenge 1, obtendo um *score* médio de 3000 pontos percorrendo assim o labirinto 7 vezes em média, e os challenges 2 e 3 imprimindo para os ficheiros de output, os valores esperados.

Como trabalho futuro, começaríamos por melhorar a movimentação do robô no challenge 1, aplicando melhor as fórmulas de movimento que o robô descreve. Para o challenge 2 e 3, a seleção da próxima área a ser explorada teria em conta a distância devolvida pelo algoritmo de pesquisa (A*), preferindo assim o espaço cuja distância a percorrer fosse melhor. Esta abordagem ainda foi testada no presente projeto mas houveram problemas no retorno da lista de movimentos, provavelmente por uma ineficiência no algoritmo de pesquisa.

Por último, no Challenge 3 gostaríamos de implementar um método no qual o robô, após descobrir os dois *targets,* aplicasse o algoritmo de pesquisa entre os vários pontos, usando um mapa onde o espaço desconhecido é também considerado navegável. Deste modo, quando o robô percorresse o caminho retornado pelo algoritmo e descobrisse que uma célula que considerava navegável é afinal uma parede, abortava a consumo dos passos, aplicava outra vez o algoritmo de pesquisa considerando agora a parede que observou, e iria verificar se já se encontrava no caminho para o poder prosseguir. Se o robô não se encontrasse no caminho iria aplicar o algoritmo de pesquisa normal para chegar até à posição do ponto inicial. Este método iria ser aplicado sucessivamente até todos os caminhos retornados pelo novo algoritmo de pesquisa não conterem 0, ou seja, zonas desconhecidas. Esta estratégia foi estudada e houve uma tentativa de aplicação no Challenge 3 mas houveram problemas com o retorno da lista de movimentos por parte do algoritmo A*.

References

1. LNCS Homepage, http://www.springer.com/lncs, last accessed 2021/11/20.