TÉCNICO LISBOA

Robótica

Laboratório nº2

Basic navigation strategies for mobile robots motion

2º Semestre – 2015/2016

Realizado por:

73177 – Ana Catarina Rosa Gonçalves

75268 – Rúben Miguel Oliveira Tadeia

75912 – David Luís Dias Fernandes

Data de Entrega: 06/06/2016 – 10/06/2016

Índice

1 Introdução

Com este projeto pretende-se realizar a movimentação de um robô móvel, num espaço pré-definido (quinto piso da Torre) de forma estruturada e concisa.

Foi necessário determinar e simular a cinemática diferencial do robot móvel, criar um mapa para que fosse possível a geração caminhos e navegação num ambiente de baixa complexidade. Por fim, foi também implementado o controlo por seguimento de trajectória, evitando colisões com obstáculos inesperados.

A movimentação ocorrer entre dois pontos indicados pelo utilizador, em que a trajectória utilizada pelo robot vai ser calculada pelo programa implementado em Matlab e sem intervenção exterior. Durante o processo, o robot deverá ser capaz de evitar obstáculos, tanto os conhecidos previamente como os que o robot será capaz de identificar, podendo estes, também, ser estáticos (paredes) ou dinâmicos (eventuais pessoas). O controlo do robot está assente num modelo cinemático, sendo os efeitos dinâmicos desprezados.

Neste Relatório será realizada a identificação das metodologias envolvidas no projeto, bem como a análise de todos os processos inerentes às mesmas, código que se encontra em anexo.

Figura 1 - Robô P3-DX utilizado ao longo do Laboratório 2

2 Atribuição de referenciais e modelo do robot

Relativamente ao sistema de coordenadas utilizado para representar a posição e orientação do robot, este encontra-se representado na figura 2.

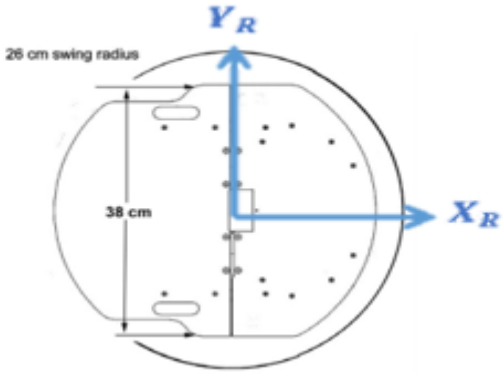

Figura 2- Representação do sistema de coordenadas utilizado para o robot

O sistema de coordenadas utilizado no referencial world encontra-se representado na figura 3:

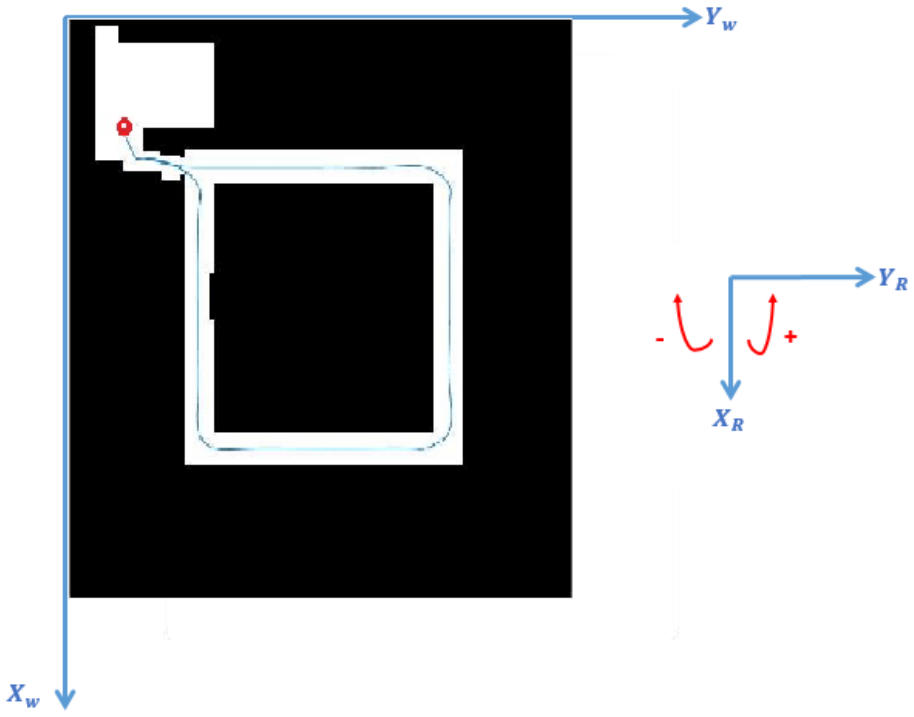

Figura 3 - Representação do sistema de coordenadas utilizado para o mapa (referencial world) & Relação entre o referencial world e as coordenadas do robô, está identificado a vermelho a posição inicial

A posição do robot relativamente ao referencial world é dada pela localização da origem. Por outro lado, a orientação do robot é dada pelo ângulo entre os eixos xx relativos a cada referencial, que quando se encontram alinhados a orientação é 0º e para outras orientações o sinal está indicado na figura 3.

O robô utiliza um mapa para se orientar que considera valores em pixéis, o valor para a posição inicial é (x; y) = (130; 70).

3 Arquitectura do Sistema

A arquitetura do sistema baseia-se no seguinte esquema:

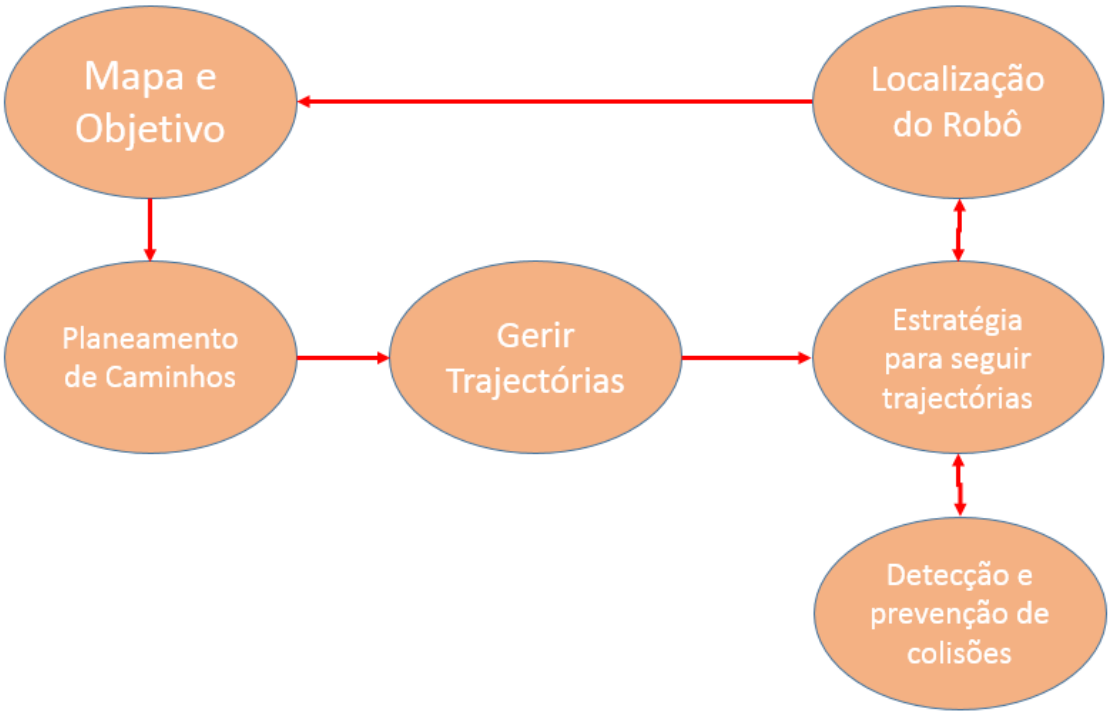

Figura 4 - Arquitetura do Sistema

a) Mapa e Objectivos

O mapa que se encontra na figura 3 é conhecido e dado como input, tem dimensões de 640x640 pixéis, em que a zona a branco (valor 1 na matriz) corresponde à zona desocupada e onde o robot se pode movimentar e a zona a preto (valor 0 na matriz) encontra-se interdita à circulação do robot. Sabendo a relação entre as dimensões reais e as dimensões consideradas na imagem, pode facilmente fazer-se a conversão e aplicá-la ao processo. A relação considerada é de 2850 cm correspondendo a 640 pixéis. O objetivo deste projeto é então fornecer uma dada posição inicial e uma posição final relativas ao mapa do ambiente e conseguir que o robot percorra uma trajectória eficiente e se desvie de possíveis obstáculos até chegar ao objetivo. Partimos de uma situação de simulação e de seguida avançámos para o ambiente real.

b) Planeamento de Caminhos

Sabendo os pontos pertencentes ao caminho que o robot terá que percorrer, podemos achar a trajetória a partir de uma interpolação que pode ser aplicada através de funções do Matlab. Estes pontos, estão numa variável chamada “path”, a mesma pode ser verificada no **Anexo 1**. Com estes pontos, é feita uma interpolação cúbica que

cria uma trajectória pelo planeamento de caminhos. A interpolação é efetuada separadamente para as coordenadas em x e em y.

Geração de trajectórias

Ainda neste tópico, foi feita a divisão do mapa em células, utilizando o quadtree algoritmo daí foi obtido um grafo em cada vértice era um bloco.

A partir daqui, atribuindo um ponto inicial e um final e utilizando um algoritmo de pesquisa, conseguíamos obter o melhor caminho.

c) Estratégia utilizada para seguir as trajectórias

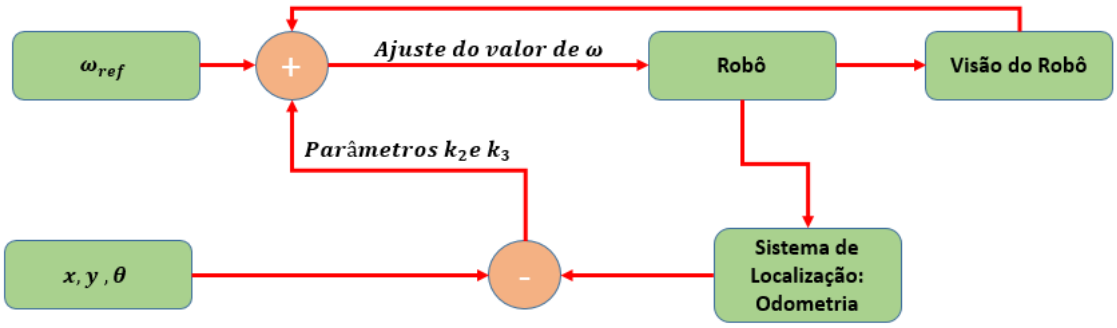

Figura 5 - Estratégia utilizada para seguir as trajectórias

Inicialmente, com a localização do robot, conseguimos obter a trajectória pretendida. Daí obtemos uma a velocidade angular de referência, assim como os valores de x, y e θ.

De seguida, é calculada uma diferença entre os valores de referência e os valores de x, y e θ da localização atual para que se verifique que o robô está a seguir a trajectória certa. Calculam-se então dois parâmetros $\boldsymbol{k_2}$ e $\boldsymbol{k_3}$, para se conseguir fazer um ajuste da velocidade angular.

O ajuste entre os valores relativos ao robot e os valores de referência são realizados com base no controlo de feedback linear na seguinte expressão:

$$\boldsymbol{u_1 = -k_1 * e_1}$$

$$\boldsymbol{u_2 = -k_2 * sgn(e_2) - k_3 * e_3}$$

Quanto ao erro $\boldsymbol{e_2}$, este foi substituído pelo seu integral num intervalo finito, pois é mais problemático o facto de o robô se afastar da trajectória por um longo período do que só pontualmente. Dado que a velocidade linear utilizada é constante, não é necessária a atribuição de um valor à constante $\boldsymbol{k_1}$.

$$\boldsymbol{k_2 = b * |v_{ref}|}$$

$$\boldsymbol{k_3 = 2 * \xi * \sqrt{\omega_{ref}{}^2 + b * v_{ref}}}$$

As constantes ξ e b foram ajustadas de acordo com testes experimentais com o robô. O robô, durante o seguimento da trajectória, vai tentar sempre convergir para os valores de referência, evitando os obstáculos.

Efetuamos o ajuste para $\boldsymbol{k_2}$, $\boldsymbol{k_3}$ e os valores dos sonares, mas não tendo sido possível juntar os 3 para que não fossem criados erros, decidimos eliminar os sonares. O ajuste que foi considerado mais importante foi o da odometria, pois é fundamental que o robot não colida com outros objetos e que este saiba o local por onde tem e pode efetuar os seus movimentos.

d) Localização do Robô

Relativamente à localização do robot no mundo onde este se movimenta, esta é puramente baseada na odometria lida pelo robot. Nomeadamente, a leitura da sua posição e ângulo relativo à posição inicial fornecida. Estes valores são obtidos através da movimentação das suas rodas e o robot faz cálculos utilizando as expressões a baixo, o fim o robô conseguiria saber em que posição do espaço se encontraria relativamente ao mapa inicialmente dado.

$$\dot{\mathbf{x}} = \mathbf{V} * \cos\boldsymbol{\theta}$$
$$\dot{\mathbf{y}} = \mathbf{V} * \sin\boldsymbol{\theta}$$
$$\dot{\boldsymbol{\theta}} = \boldsymbol{\omega}$$

Assim, após a obtenção dos valores da odometria do robô, poderiam ter sido aplicados corretamente os sonares. Estes analisariam a odometria do robô e verificariam se a distância às paredes seria suficiente para que o robô não estivesse na posição central do corredor. Infelizmente os sonares não ficaram a funcionar corretamente.

e) Detecção e prevenção de colisões

O robot está equipado com um conjunto de 8 sonares, os quais fornecem dados sobre a visão que este obtém sobre o ambiente e que não se encontra no mapa. Os valores que são lidos são em milímetros e referem a distância a que se deteta um objeto do robot.

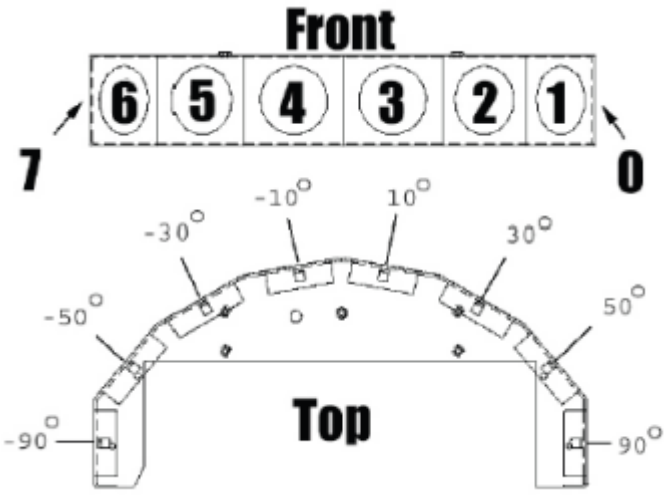

Figura 6 - Representação dos sonares no robot Pioneer P3-DX

Como é possível de ver num dos vídeos em que utilizámos sonares, utilizámos apenas os que correspondem aos ângulos 90º e -90º. O seu objetivo seria evitar a colusão com objetos, levando o robô a afastar-se da sua trajectória inicial para a direção contrária ajustando a velocidade angular.

4 Resultados Experimentais

Ao longo de vários testes, pré-avaliação, o nosso robô conseguiu realizar o percurso assinalado a azul (mapa) única e exclusivamente utilizando odometria, sendo necessário realizar uns ajustes por erros de posição, momento a partir do qual consideramos utilizar os sonares. Com os sonares obtivemos o mesmo resultado (mesma posição de paragem, ver Vídeo).

Durante a realização da avaliação este foi o resultado experimental que foi obtido:

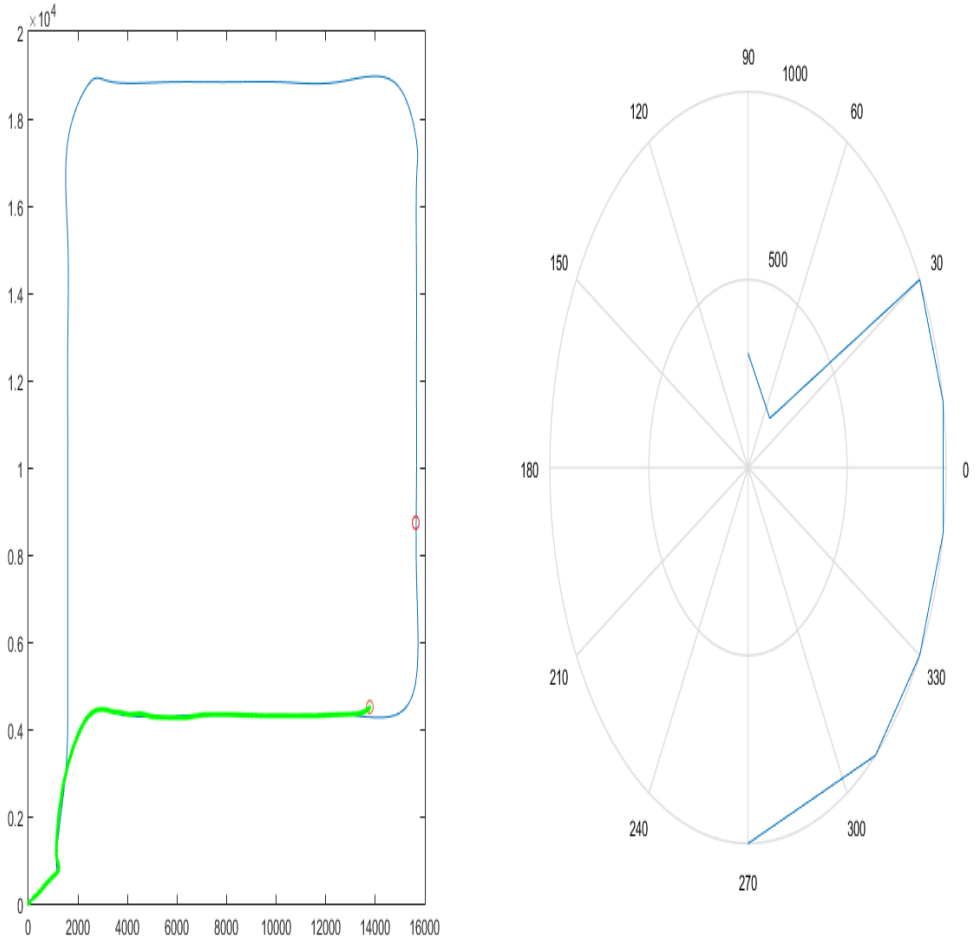

Figura 7 - Resultados experimentais durante avaliação

A verde vemos a posição do robô e a azul o mapa. O mapa é composto por pontos dados por nós. Inicialmente dentro da sala atribuímos ao robô certos pontos, e através de interpolação ele cria a trajectória a realizar, mas como escolhemos poucos pontos este pode ter várias trajetórias, no entanto parecidas, dentro da sala. Ao longo dos 3 testes, o nosso robô mostrou ser conciso no algoritmo, pois terminou sempre no mesmo local. Um dos erros que podemos ver na figura 7 do lado esquerdo, é que o robô tenta virar antes de chegar à curva, pensámos inicialmente que seria um problema do controlador, mas em conversa com o professor e após a avaliação, vimos que seria um problema de código. Do lado direto temos uma imagem que reflete a leitura dos valores dos sonares (não utilizados durante a avaliação).

5 Conclusões

O objetivo final do projeto, colocar o robô a dar uma volta ao 5º piso da Torre e voltar ao ponto de partida não foi cumprido. Embora tenhamos tido um bom funcionamento das simulações com o robô real. O robô segue os objetivos propostos para o projeto, efetuando a calibração dos controlos, um planeamento de caminhos, geração de trajectória, seguimento da trajectória definida e deteção e prevenção de colisões com obstáculos de forma eficiente e com êxito.

Contudo, há que ter em conta certos parâmetros que podem dar origem a erros durante a execução de todo o projeto. Dentro deste âmbito, temos erros nas medições realizadas às posições e os erros na movimentação do robô, nomeadamente um descaimento constante para a esquerda. Outro fator que contribuiu para a incerteza durante a realização do trabalho foi a escolha dos parâmetros ξ e b que tiveram que ser obtidos experimentalmente. Apesar de se ajustarem corretamente ao procedimento geral e terem sido os que se revelaram mais próximos do desejado, podem não ser os valores ótimos para o caso.

Por fim, também há que considerar os erros associados ao equipamento utilizado, estes influenciam o seguimento da trajectória e/ou a própria localização e orientação do robô móvel, nomeadamente rodas do robô, direção desalinhada, entre outros. Neste âmbito, há que referir especialmente os erros relativos aos sonares e dado que foram usados apenas 2, o de 90º e de -90º, estes por vezes comprometem correções a objetos que apareçam.

Outro dos aspetos a referir é o facto de termos 2 vídeos utilizando e não utilizando sonares, o mesmo se deve ao facto de termos desenvolvido 2 versões para o projeto que culminam em resultados semelhantes, como se pode ver pelo vídeo. No entanto, no dia da apresentação, mostramos a versão funcional apenas com odometria.

6 Anexos

a) Anexo 1 Código da função lap_teste

Aqui encontra-se o código da função lap_teste que contém, não só a variável "path", que é uma matriz com os pontos que que constituem o mapa na visão do robô, mas é também a função global de todo o código.

```
global rob;
mapa=imread('mapa.bmp');

for i=1:640
    for j=1:640

        if ((i < 145) && (j >= 190)) || ((j>=500)) || (i > 493) || ((i
> 187) && (j <= 147)) || ((j>=184 && j < 320) && (i>200 && i <320 ) )
            mapa(i,j)=0;
        end
    end
end

for i=1:640
    for j=1:640
     if((i>=145) && (i <= 493)) && ((j == 500) || (j == 501) || (j ==
502))
         mapa(i,j) = 1;
     end
    end
end



path = [130 70;
    155 85;
    155 95;
    162.5 132.5;
    185 165;
    215 165;
    265 165;
    315 165;
    365 165;
    415 165;
    455 165;
    475 185;
    475 235;
    475 285;
    475 335;
    475 385;
    475 435;
    475 455;
    455 485;
    405 485;
    355 485;
    305 485;
```

```
    255 485;
    205 485;
    185 485;
    165 455;
    165 405;
    165 355;
    165 305;
    165 255;
    165 205;
    165 166;
    162.5 132.5;
    155 95;
    155 85;
    130 70;];

figure(5);
imshow(mapa); hold on;
plot(path(:,2),path(:,1),'.-');
pause(0.001);hold off;

[y,x,ppx,ppy]=interpolation(path,2,10000);
imshow(mapa);hold on;
plot(y,x);
hold off;

ratio = 15800/348;
p0 = path(1,:);
for k=1:length(path(:,1));
    path_real(k,:)=ratio*(path(k,:) - path(1,:));
end


[y,x,ppx,ppy]=interpolation(path_real,2,10000);

via_1 = transpose([x;y]);
pathfollow_virt_ang(via_1,10,1,1,0);
```