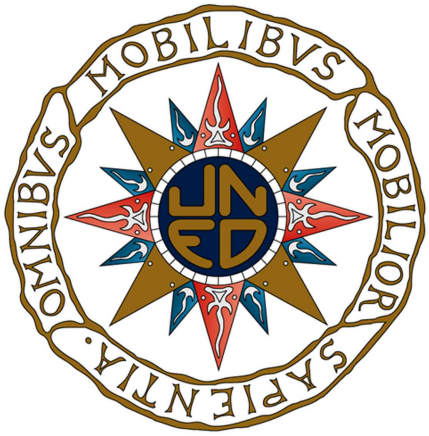

ROBOTS AUTÓNOMOS

Ricardo de Figueiredo Minelli

ROBOTS AUTÓNOMOS

Ricardo de Figueiredo Minelli

Construcción de mapas

Tutorizada por

Jose Antonio Lopez Orozco

Índice general

1 Puntos y aspectos más destacables de los programas desarrollados

1.1 robot.m

Fuerón desarrolados dos ficheros de codigos en el entorno MATLAB, robot.m y ultrasonidos.m, los cuales tienen la finalidad de implementar el algoritmo de construcción de mapas, en concreto ha sido un mapa de rejilla, que se actualiza la información sensorial obtenida mediante la teoría bayesiana.

El primer fichero, robot.m se encarga del movimiento del robot, este movimiento se carga através del fichero Encoder.mat. En este caso tenemos un robot de ruedas diferenciales que realiza un movimiento, los encoders acoplados a la rueda derecha e izquierda miden NR y NL pulsos respectivamente. Se puede observar que el robot gira entorno a un punto virtual (denominado Centro Instantáneo de Rotación, CIR) un ángulo y se desplazan las ruedas DR y DL. Si los encoders miden Nc cuentas en una rotación completa de la ruedas se puede calcular la distancia y el angulo al que se mueve el robot, para eso hacemos uso de las ecuaciones:

$$D_I = \frac{2\pi R_L}{N_C} N_L$$

$$D_D = \frac{2\pi R_R}{N_C} N_R$$

$$\Delta\theta = \frac{DR_i - DL_i}{L}$$

$$D_T = \frac{D_D + D_I}{2}$$

$$A°(acum)_{n+1} = A\theta° + A_i°$$

El desplazamiento angular $\Delta\theta$ se mide por la diferencia de orientación de un sólido en un instante dado respecto a su orientación inicial y haciendo uso de la razones trigonometricas de un triángulo rectángulo podemos hallar las posiciones X e Y en el mapa cujos datos han sidos cargados

previamente del fichero Mapa.mat.

$$X_{n+1} = X_n + D_i \cos A\theta°(acum)_n$$

$$Y_{n+1} = Y_n + D_i \operatorname{sen} A\theta°(acum)_n$$

Una vez cargado estos datos y ya con la lectura de los encodres que nos proporciona el movimiento del robot en un plano 2D nos disponemos a calcular el barrido de nuestro sensor ultrasonido que se encuentra al centro del robot con el intuito de detectar el terreno y proporcionarnos información sobre el entorno al que se mueve.

```
Encoder=load('Encoder.mat');      % Cargamos los enconders
MapReal=load('Mapa.mat');         % Cargamos el Mapa l gico
EI=Encoder.Enc(:,1);              % Encoders rueda izquierda
ED=Encoder.Enc(:,2);              % Encoders rueda derecha
ancho=10;
largo=20;
DI=2*pi*5.*EI/100;                % rueda izquierda
DD=2*pi*5.*ED/100;                % rueda derecha
DT=(DI+DD)/2;                     % centro del eje
ang=(DD-DI)/10;                   % angulo del movimiento
```

Tenemos la posición incial del robot y su orientación, a partir de esta posición podemos calcular la detección del ultrasonido y para eso llamamos a la función ultrasonidos enviando la posición X e Y a la que haremos la medición, el angulo acumulado, lo cual, se tiene en cuenta donde estaba antes el robot, el Mapa lo cúal haremos el barrido, también se especifican cuantos grados deberá medir el sensor y si el mapa detectado debe de ser ocultado o no.

```
posX=200;
posY=100;
Mapa=0.5*ones(size(MapReal.M));
for i=2:40
    angAcum(i)=ang(i-1)+angAcum(i-1);     % angulo acumulado
    posX(i)=DT(i-1)*cos(angAcum(i-1))+posX(i-1);% trigonometria
    posY(i)=DT(i-1)*sin(angAcum(i-1))+posY(i-1);% trigonometria
    figure(1);hold;
    image(300*Mapa);
    ax = gca;
    ax.YDir = 'normal';
    hold on;
    plot(posX,posY,'o-','MarkerFaceColor','red','
        MarkerEdgeColor','red','MarkerIndices',i)
    pause(.1);
end
```

Pintamos la trayectória del robot y el propio robot a través de la función plot con los parametros:

```
plot(posX,posY,'o-','MarkerFaceColor','red','MarkerEdgeColor','
    red','MarkerIndices',i)
```

1.2 ultrasonidos.m

En el programa ultrasonidos primeramente hacemos un redondeo de las posiciones X e Y para poder consultarlas y modificarlas en nuestro mapa (matriz).

```
vec_X = round(posX);    %posicion X del robot
vec_Y = round(posY);    %posicion X del robot
```

El sensor de ultrasonidos tiene un angulo de detección de 30 grados, al angulo al que se direcciona el robot que nos envia el programa robot.m le restamos 15 grados y hacemos un barrido de 30 o 360 grados según se indique en la variable grados (el angulo en el programa esta expresado en radianes). También definimos la variable epsilon para cuando detecte un obstáculo canculamos 10cm mas.

```
epsilon = 10;
i = -0.261799;    %Aprox -15 grados
angU = angU + i;     % restamos al angulo 15
```

La variables ocultarMapa nos indica si debemos mostrar el mapa detectado o no.

```
if(ocultarMapa)
    Mapa=0.5*ones(size(MapReal.M));
end
```

Recogemos cada centimetro de los 80 que alcanza nuestro sensor de grado en grado y asignamos un valor si encontramos con un 0 o con un 1 en nuestro mapa binario de obstaculos, luego hacemos el calculo de la probabilidad de Bayes y lo ponemos en nuestro mapa resultado la probabilidad de detección del obstaculo. Si detectamos un obstáculo entramos en un bucle while, lo cual nos vá a calcular 10 iteraciones mas de la probabilidad de Bayes.

```
for k=1:grados
obstaculo = false;
rayoU = 1;
m = 0;
for k=1:80
    vec_UX(k) = round(cos(angU)*rayoU);
    vec_UY(k) = round(sin(angU)*rayoU);
    X = vec_X+vec_UX(k);
    Y = vec_Y+vec_UY(k);
    if(MapReal.M(X,Y)== 1 & ~obstaculo)
        prob = 0.5+0.5/3;
        proBayes = (prob*Mapa(X,Y))/((prob*Mapa(X,Y))+(1-
            prob)*(1-Mapa(X,Y)));
        while(m <= epsilon)
            vec_UX(k+m) = round(cos(angU)*rayoU);
            vec_UY(k+m) = round(sin(angU)*rayoU);
            X = vec_X+vec_UX(k+m);
            Y = vec_Y+vec_UY(k+m);
            Mapa(X,Y)= proBayes;
            m = m + 1;
            rayoU = rayoU + 1;
        end
        obstaculo = true;
    elseif(MapReal.M(X,Y)== 0 & ~obstaculo)
        prob = 0.1;
        proBayes = (prob*Mapa(X,Y))/((prob*Mapa(X,Y))+(1-
            prob)*(1-Mapa(X,Y)));
        Mapa(X,Y)= proBayes;
    end
    rayoU = rayoU + 1;
end
angU = angU + 0.0174533; %Aprox 1 grado
end
```

Se considera que existe un objeto cuando su creencia supera el valor de 0.8, en la primera iteración obtenemos el valor aproximado de 0,66, en la segunda iteración obtenemos el valro aproximado de 0,8 considerando así que existe un obstáculo en esta posición.

2 | Ejecución del codigo

Al ejecutar el programa principal "robot.m" en el entorno Matlab, este nos dará un menu con algunas opciones para la ejecucon del programa:

Figura 2.1: Menu de opciones de ejecución

Entre las opciones podemos elegir:

- Solamente el Robot

  Trayectoria del robot en un mapa vacio.

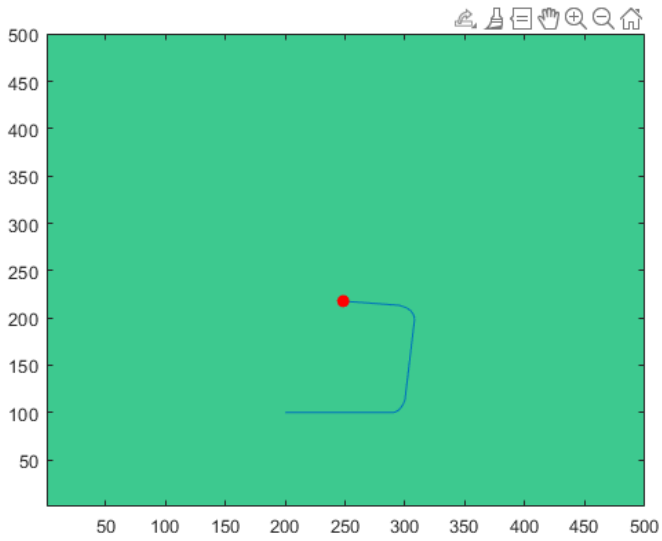

Figura 2.2: Solamente el Robot

- Robot Sensor Estático

  Trayectoria del robot con el sensor estático es un angulo de 30°, en esta opción no se pinta el mapa detectado, solamente la detección del sensor en la posición actual.

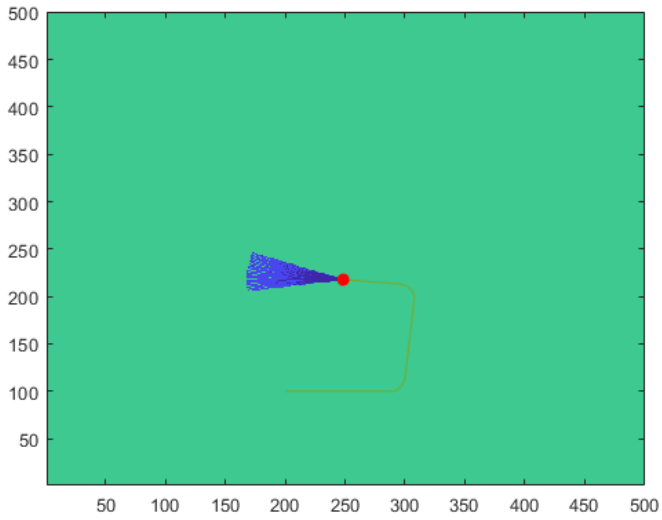

Figura 2.3: Robot Sensor Estático

- Robot y Mapa Sensor Estático 30°

  Trayectoria del robot con el sensor estático con un angulo de 30°, en esta opción se pinta el mapa detectado en las posiciones anteriores.

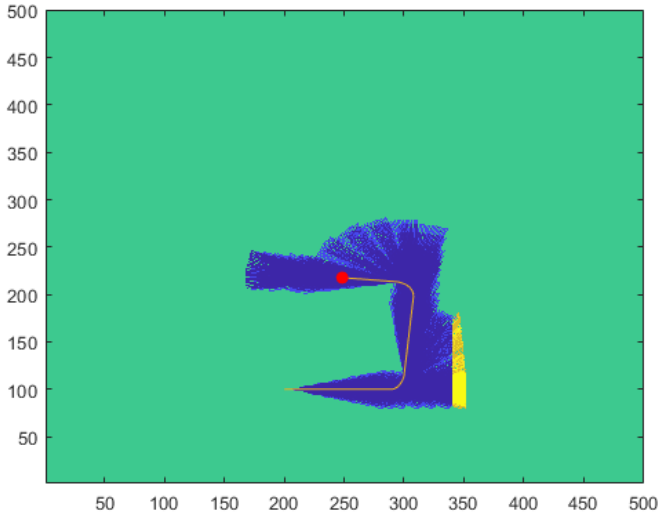

Figura 2.4: Robot y Mapa Sensor Estático 30°

- Robot Sensor 360°

  Trayectoria del robot con el sensor es un angulo de 360°, en esta opción no se pinta el mapa detectado, solamente la detección del sensor en la posición actual.

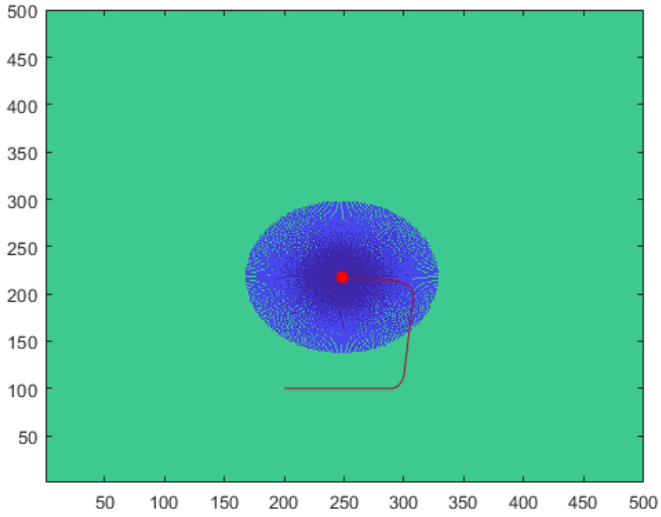

Figura 2.5: Robot y Mapa Sensor Estático 30°

- Robot y Mapa Sensor 360°

  Trayectoria del robot con el sensor estático con un angulo de 30°, en esta opción se pinta el mapa detectado en las posiciones anteriores.

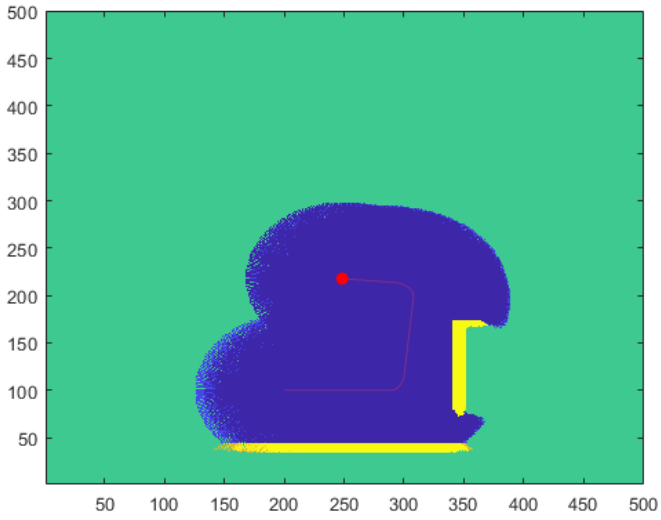

Figura 2.6: Robot y Mapa Sensor 360°

- Mapa Completo

  Mapa completo con todos los obstáculos.

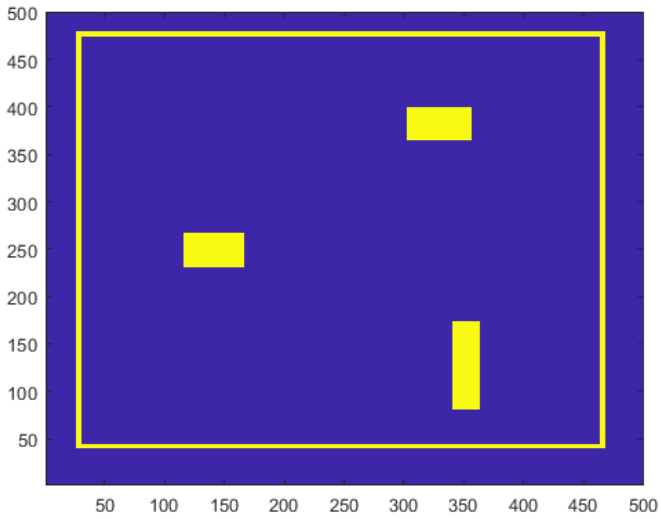

Figura 2.7: Mapa Completo

- Salir

  Para salir del programa clique aqui o cierre la ventana.

3 Representación del mapa

Los colores del mapa nos indican que:

- **Verde** - Zona desconocida.
- **Azul** - Las celdas de color azul se pueden considerar como vacías.
- **Amarillo** - Las celdas de color amarillo con probabilidad alta de ocupación.
- **Rojo** - Robot.

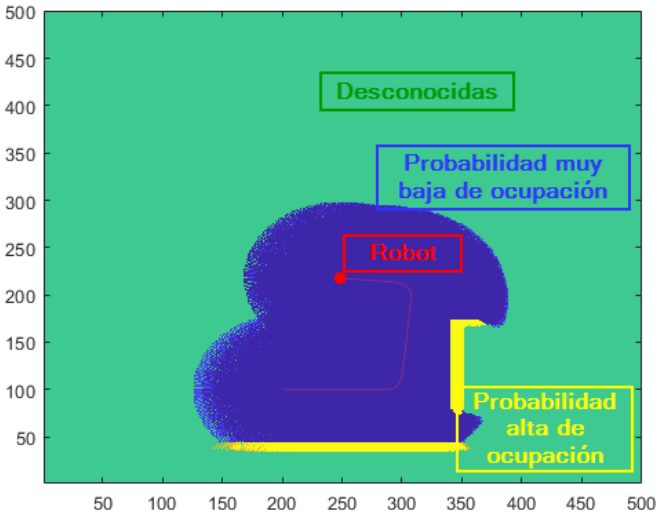

Figura 3.1: Definición de colores del mapa

4 Documentos anexados

- Programas robot.m y ultrasonidos.m
- Hola de cálculo excel (Robot.ods) con los cálculos de los encoders para la comprobación.