```
function [dist_max,path_x,path_y] =
 planificadorBellmanFord(mapMatrix,Re,xini,yini,xfin,yfin)

% Este algoritmo determina la trayectoria minima entre un punto
 inicial y
% todos los restantes puntos del entorno
%
% el mapa tiene una resolucion de 0.5m lo que significa que la
 distancia
% entre estados es de 0.5m con los sig 4 movimientos:
% (i = i+1, j = j) (i = i-1, j = j) (i = i, j = j-1)   (i = i, j = j
+1)
% el costo de un estado a otro es 0.5m
%
% cuando se avanza con un movimiento diagonal(ang = 45) la distancia
 entre
% estados sera sqr(2*Re^2)->costo del movimiento. Esto es con los
% movimientos m1,m3,m4,m6
% se asume inicio en i=2,j=2

[large,wide] = size(mapMatrix);
weightsMatrix = mapMatrix;                     %Conjunto de todos los
 estados
n = large*wide;
costoMov1 = Re;                                %distancia horizontal
costoMov2 = sqrt(2*(Re^2));                    %distancia diagonal

for i = 1: 1:large                       %inicializando pesos con
 infinito
    for j = 1: 1: wide
        weightsMatrix(i,j) = inf;
    end
end
Parent_col = weightsMatrix;                 %matriz que almacena las
 coordenadas de las columnas de los padres
Parent_row = weightsMatrix;                 %matriz que almacena las
 coordenadas de las filas de los padres

source = [yini xini];
weightsMatrix(source(1),source(2)) = 0;           %coordenadas de
 inicio

for k = 0: 1: n-1
    for i = 2: 1: large -1
        for j = 2: 1: wide -1
            if( mapMatrix(i, j+1) ==1 && (weightsMatrix(i, j+1) >
 weightsMatrix(i,j) + costoMov1))
                weightsMatrix(i, j+1) = weightsMatrix(i,j) +
 costoMov1;
                Parent_col(i,j+1) = j;
                Parent_row(i,j+1) = i;
            end
            if( mapMatrix(i, j-1) ==1 && weightsMatrix(i, j-1) >
 weightsMatrix(i,j) + costoMov1)
                weightsMatrix(i, j-1) = weightsMatrix(i,j) +
 costoMov1;
                Parent_col(i, j-1) = j;
                Parent_row(i, j-1) = i;
            end
            if( mapMatrix(i+1,j) ==1 && weightsMatrix(i+1, j) >
 weightsMatrix(i,j) + costoMov1)
                weightsMatrix(i+1, j) = weightsMatrix(i,j) +
 costoMov1;
                Parent_col(i+1, j) = j;
                Parent_row(i+1, j) = i;
            end
            if( mapMatrix(i-1, j) ==1 && weightsMatrix(i-1, j) >
 weightsMatrix(i,j) + costoMov1)
                weightsMatrix(i-1, j) = weightsMatrix(i,j) +
 costoMov1;
                Parent_col(i-1, j) = j;
                Parent_row(i-1, j) = i;
            end
            if( mapMatrix(i+1, j+1) ==1 && weightsMatrix(i+1, j+1) >
 weightsMatrix(i,j) + costoMov2)
                weightsMatrix(i+1, j+1) = weightsMatrix(i,j) +
 costoMov2;
                Parent_col(i+1, j+1) = j;
                Parent_row(i+1, j+1) = i;
            end
            if( mapMatrix(i-1, j+1) ==1 && weightsMatrix(i-1, j+1) >
 weightsMatrix(i,j) + costoMov2)
                weightsMatrix(i-1, j+1) = weightsMatrix(i,j) +
 costoMov2;
                Parent_col(i-1, j+1) = j;
                Parent_row(i-1, j+1) = i;
            end
            if( mapMatrix(i+1, j-1) ==1 && weightsMatrix(i+1, j-1) >
 weightsMatrix(i,j) + costoMov2)
                weightsMatrix(i+1, j-1) = weightsMatrix(i,j) +
 costoMov2;
                Parent_col(i+1, j-1) = j;
                Parent_row(i+1, j-1) = i;
            end
            if( mapMatrix(i-1, j-1) ==1 && weightsMatrix(i-1, j-1) >
 weightsMatrix(i,j) + costoMov2)
                weightsMatrix(i-1, j-1) = weightsMatrix(i,j) +
 costoMov2;
                Parent_col(i-1, j-1) = j;
                Parent_row(i-1, j-1) = i;
            end
        end
    end
end
```

```
dist_max = weightsMatrix(yfin,xfin);
  %distancia del recorrido

path_x(1) = xfin;
path_y(1) = yfin;
C = 2;
i = yfin;
  %coordenadas fila pto final
j = xfin;
  %coordenadas col pto final
fin = 0;
while(fin == 0)                         %conformacion del camino
    if(i == inf)                        %no hay camino posible
        if (j == inf)
            fin = 1;
            path_x(:) = 0;
            path_y(:) = 0;
        end
    else
    path_x(C) = Parent_col(i,j);
    path_y(C) = Parent_row(i,j);
    auxi = i;
    auxj = j;
    i = Parent_row(auxi,auxj);
    j = Parent_col(auxi,auxj);
    C = C+1;
    if(i == yini)                      %si se llega al inicio del
 camino
        if(j == xini)
            fin = 1;
        end
    end
    end
end

% path_x(C) = Parent_col(i,j);
% path_y(C) = Parent_row(i,j);

% spy(mapMatrix);
% hold on;
% p = plot(path_x,path_y,'k');
% p.LineWidth = 1.5;
% grid on;
% xticks(0:1:large/Re);
% yticks(0:1:wide/Re);
% axis([0 large/Re 0 wide/Re]);

end
```

```
Not enough input arguments.

Error in planificadorBellmanFord (line 16)
[large,wide] = size(mapMatrix);
```

Published with MATLAB® R2017b