Institut Supérieur des Etudes Technologiques de Bizerte
Département GE (A.U : 2022– 2023)

# MASTER PROFESSIONNEL RAIA

# TRAVAUX PRATIQUES

## Atelier Robots Mobiles

## *TPN°4 : Planification de trajectoire probabiliste d'un robot mobile (Probabilistic Roadmaps Path planning : PRM)*

***Date :*** ---------- ***Classe :*** ---------- ***Durée : 3h***

| | *Nom & Prénom :* | *AB/PR* | *Mot/Part* | *TP N°* | *Total* |
|---|---|---|---|---|---|
| *1* | ---------- | | /10 | /10 | /20 |
| *2* | ---------- | | /10 | /10 | /20 |
| *3* | ---------- | | /10 | /10 | /20 |

***Objectifs du TP :***

- ✓ ----------
- ✓ ----------
- ✓ ----------

***Conditions de réalisation et moyens :***

- ✓ ----------
- ✓ ----------
- ✓ ----------

***Objectifs :***

- Comprendre le fonctionnement du planificateur de chemin probabiliste (PRM)
- Générer une carte en utilisant la fonction binaryOccupancyMap() de Matlab
- Générer des points aléatoires dans la carte et les connecter pour construire le PRM
- Planifier un chemin sûr et efficace en utilisant un algorithme de recherche de chemin
- Tracer le chemin optimal sur la carte.

## I- Introduction :

La planification de trajectoire est un problème important en robotique qui consiste à trouver un chemin sûr et efficace pour déplacer un robot d'un point de départ à un point d'arrivée en évitant les obstacles. Le planificateur de trajectoire probabiliste (PRM) est une méthode de planification de chemin basée sur l'échantillonnage aléatoire pour construire un graphe de connectivité qui représente l'environnement de travail. Dans ce TP, nous allons implémenter un planificateur de trajectoire PRM en Matlab pour trouver la trajectoire optimale.

## II- Mise en œuvre d'un planificateur de trajectoire probabiliste (PRM)

### a- Activité N°1 :

1- Ouvrir MATLAB et créer un nouveau script puis écrivez les codes ci-dessous et enregistrez-les sous un nom de fichier de votre choix, par exemple 'TP_PRM.m'.

### 2- Définition de la carte d'occupation

```
run ('MapPersonnalise.m')% Charger l'environnement de travail
resolution = 0.5;   %taille de l'environnement
map = robotics.BinaryOccupancyGrid(map, resolution);
inflatedmap = copy(map);
robotRadius = 0.1;
inflate(inflatedmap,robotRadius);
subplot(121), show(map);
subplot(122), show(inflatedmap);
```

3- Exécuter le code et observer les deux cartes affichées. La première carte représente la carte d'occupation initiale, tandis que la seconde représente la carte après avoir effectué une dilatation autour des obstacles. Discuter des différences entre les deux cartes.

### 4- Définir les positions de départ et d'arrivée

```
start = [35.0 5.0];
goal = [5.0 38.0];
```

### 5- Définir les paramètres de l'algorithme

```
prm = robotics.PRM;% Create a probabilistic roadmap
```

```
prm.Map = inflatedmap;
prm.NumNodes = 200;
prm.ConnectionDistance = 5;
show(prm)
```

6- Modifier les positions de départ et d'arrivée du robot en changeant les valeurs des variables "start" et "goal". Exécuter le code et observer le résultat.

7- Modifier les paramètres de l'algorithme PRM en modifiant le nombre de nœuds et la distance de connexion. Exécuter le code et observer le résultat.

**8- Calcul du chemin optimal**

```
path = findpath(prm, start, goal);
```

**9- Affichage de la carte et du chemin optimal**

```
figure;
show(inflatedmap);
hold on;
if ~isempty(path)
    show(prm)
    plot(path(:,1), path(:,2), 'r', 'LineWidth', 2,
'DisplayName', 'Chemin optimal');
    legend('show');
else
    disp("Aucun chemin trouvé entre la position de départ et la
position d'arrivée.");
end
plot(start(1), start(2), 'bo', 'MarkerSize', 10, 'LineWidth', 2,
'DisplayName', 'Position de départ');
plot(goal(1), goal(2), 'go', 'MarkerSize', 10, 'LineWidth', 2,
'DisplayName', 'Position d''arrivée');
title('Carte et chemin optimal');
xlabel('X [m]');
ylabel('Y [m]');
```

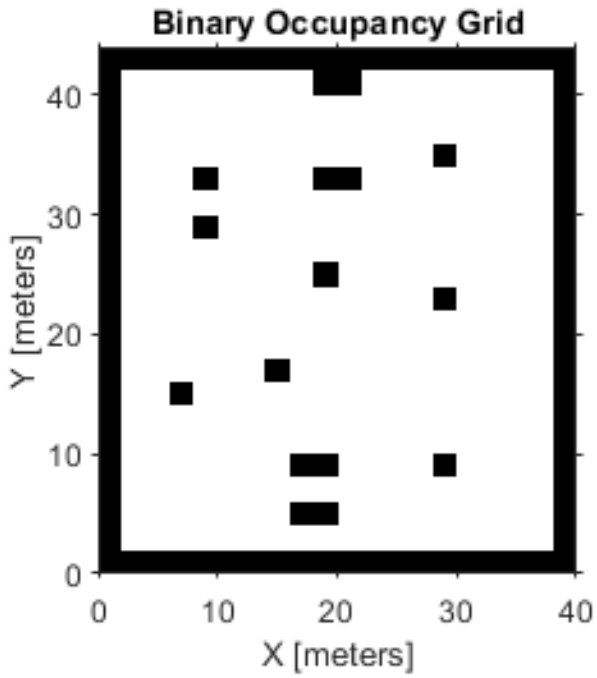
10- Ajouter une nouvelle forme d'obstacle à la carte en utilisant la fonction "setOccupancy". Réexécuter le code et observer comment l'algorithme PRM s'adapte aux nouveaux obstacles.

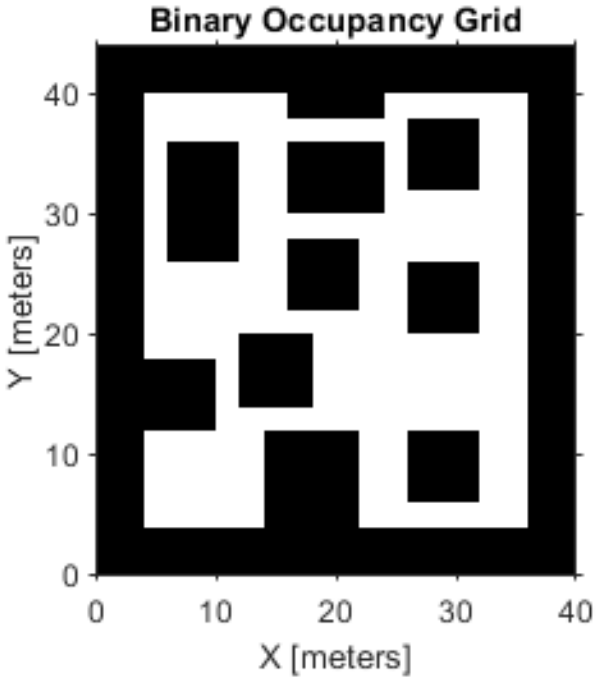
11- Ajouter une contrainte de mouvement supplémentaire pour le robot, par exemple une zone à éviter. Modifier le code pour prendre en compte cette contrainte et exécuter le code.

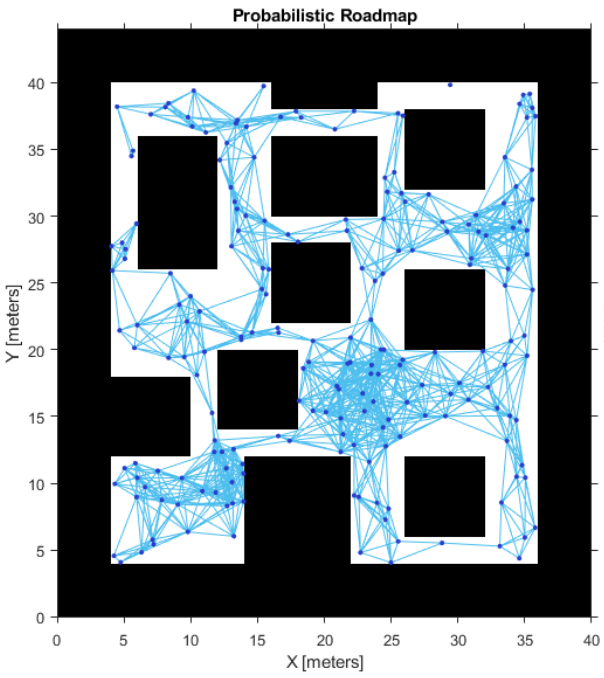
12- Proposer d'autres modifications ou extensions possibles pour améliorer l'algorithme de planification de chemin. Discuter des avantages et des inconvénients de ces modifications.

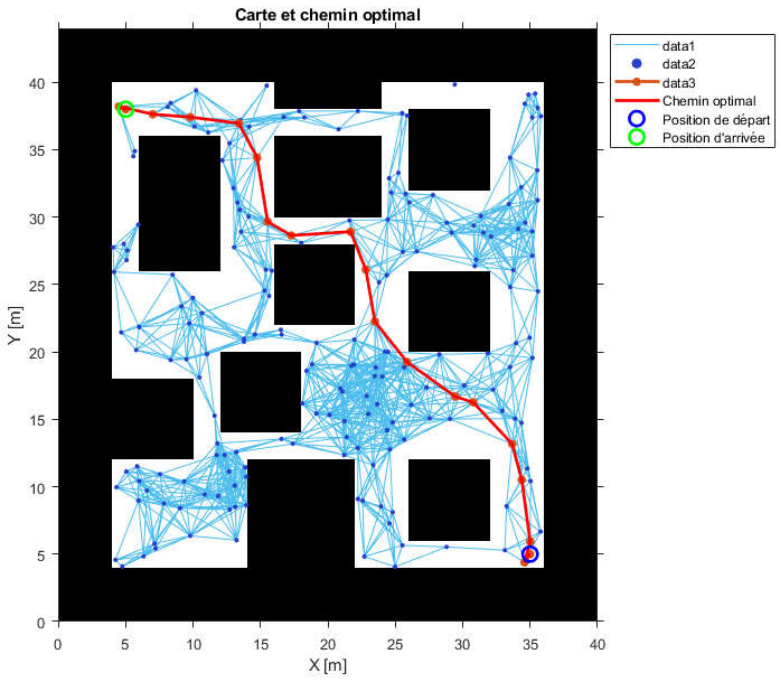
13- Enregistrer le script modifié sous un nouveau nom et l'utiliser comme base pour d'autres projets de planification de chemin en utilisant l'algorithme PRM.

Les figures suivantes représentent les résultats d'exécution du script Matlab 'TP_PRM.m'



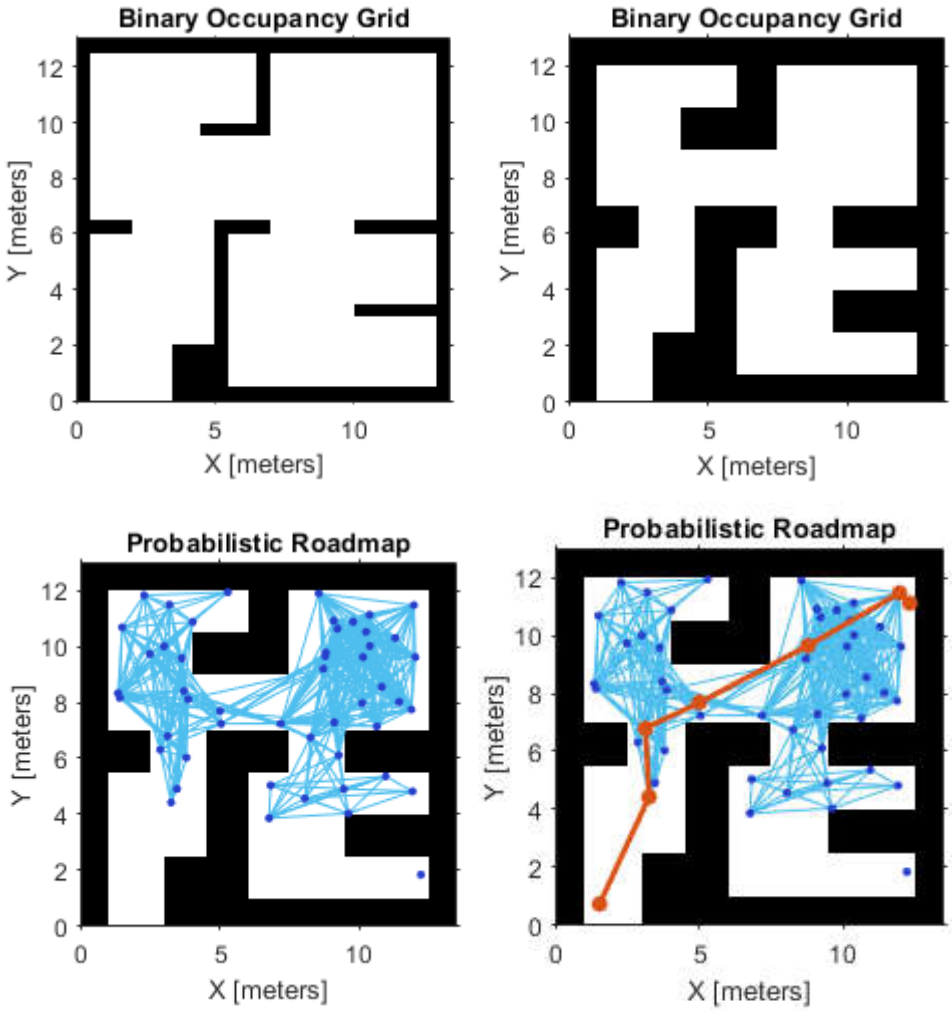



### b- Activité N°2 :

Créer un nouveau script puis écrivez le code ci-dessous et enregistrez-le sous un nom de fichier de votre choix, par exemple 'TP1_PRM.m'.

```
%1- Charger puis dilater l'environnement de travail
% Dans « exampleMaps.mat » il existe trois environnements de
% travail prédéfinit (complexMap, simpleMap, emptyMap)
load exampleMaps.mat
map = robotics.BinaryOccupancyGrid(simpleMap,2);
inflatedmap = copy(map);
robotRadius = 0.2;
inflate(inflatedmap,robotRadius);
subplot(121), show(map);
subplot(122), show(inflatedmap);

%2- Générer des trajectoires avec les paramètres initiaux
prm = robotics.PRM;
prm.NumNodes = 50;
prm.ConnectionDistance = 5;
prm.Map = inflatedmap;
show(prm)

%3- Définition des positions de départ et d'arrivée
disp('Cliquer pour l'emplacement de départ');
startLocation = ginput(1);
disp('Cliquer pour l'emplacement d'arrivée');
endLocation = ginput(1);

%4- Effectuer le calcul du chemin optimal et afficher à la fois
%la carte et le chemin obtenu.

path = findpath(prm,startLocation,endLocation)

while isempty(path)
prm.NumNodes * prm.NumNodes + 20;
update(prm);
path = findpath(prm,startLocation,endLocation);
 show(prm)
 pause(1);
end;
show(prm) ;
```

Les figures suivantes représentent les résultats d'exécution du script Matlab 'TP1_PRM.m'




Quelques références sur le PRM :

[1] Steven M. LaValle, "Planning Algorithms", Cambridge University Press, 2006.

[2] Lydia E. Kavraki, Petr Svestka, Jean-Claude Latombe, and Mark H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces", IEEE Transactions on Robotics and Automation, vol. 12, no. 4, August 1996.

[3] Nancy M. Amato and Wesley H. Huang, "On the Complexity of Probabilistic Roadmap Methods", International Journal of Computational Geometry and Applications, vol. 9, no. 3, 1999.

[4] Mark Moll and Jean-Claude Latombe, "Path planning for mobile robots: A review", IEEE Transactions on Robotics and Automation, vol. 21, no. 5, October 2005.