

Planification de chemin par la méthode A*

ENSTA Paris - Cours ROB312

David FILLIAT

4 décembre 2019

1 Introduction

Dans ce TP, nous allons utiliser la méthode A* pour la recherche de plus court chemin. Pour cela, nous utiliserons le code MATLAB disponible sur la page web du cours, originellement écrit par Anthony Chrabieh¹ et modifié pour les besoins du TP.

2 Fonction fournie

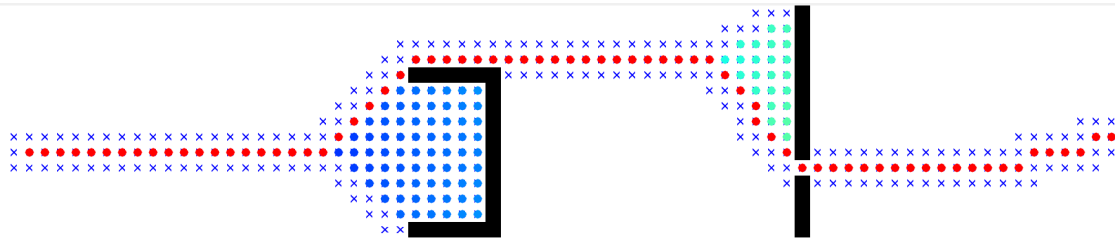


FIGURE 1 – Exemple de résultats de l’algorithme A*.

Le script `Astar` permet de définir une carte, un point de départ, un point d’arrivée et de calculer le chemin le plus court entre les deux (Figure 1).

Question 1 : Expliquez en détail le fonctionnement de la méthode implémentée par le script dans votre compte-rendu.

3 Role de l’heuristique

Dans cette partie, vous utiliserez le script `Astar` avec ses paramètres par défaut pour répondre aux questions suivantes :

Question 2 : Avec le paramètre `weight=0` quel est l’algorithme qui est exécuté ?

Question 3 : Modifiez le paramètre `weight`. Que se passe-t-il pour la valeur 1 ? pour la valeur 3 ? pour la valeur 1.4 ? Quelle est la valeur la plus adaptée ?

1. <https://fr.mathworks.com/matlabcentral/profile/authors/5287954-anthony-chrabieh>

4 Role des poids associés aux noeuds

Dans la configuration par défaut, aucun poids n'est associé aux noeud eux-même (variable w). Tous les noeuds sont donc traités de la même manière, et le chemin optimal va passer près des obstacles et dans des passages étroits.

Question 4 : Modifiez le poids des noeuds afin que les chemins ne passent pas trop près des obstacles. Vous pourrez utiliser la fonction matlab `bwdist (MAP)` qui permet de réaliser une transformée en distance, c'est à dire de calculer, pour chaque position libre de la carte, la distance à l'obstacle le plus proche. Illustrez les trajectoires obtenues sur l'environnement proposé par défaut dans le script.

5 Influence de l'environnement

Question 5 : Utilisez la fonction de génération d'obstacles aléatoire qui est commentée en début de script. Comparez les performances de l'algorithme avec `weight=0` et avec `weight=1.4` en faisant varier le nombre d'obstacle. Que constatez vous ? Dans quels cas l'algorithme A* apporte-t-il le plus de gain ?