Planification déterministe

Modèles d'environnement, planification de trajectoires

Francis Colas

5 décembre 2019

01

Introduction

Introduction

Planification de mouvement

- trouver une trajectoire ou un chemin
- dans un environnement donné
- faisable par un robot donné
- en évitant les obstacles

Introduction

Planification de mouvement

- trouver une trajectoire ou un chemin
- dans un environnement donné
- faisable par un robot donné
- en évitant les obstacles

Cartes et espace de configuration

- carte :
  - espace de travail $\mathcal{W}$
  - obstacle, mais indépendants du robot
- espace de configuration $\mathcal{C}$
  - construit pour un robot donné (mobile ou articulé)
  - $\mathcal{E}$ espace libre de collision

Introduction

Problématique

- à partir de l'espace libre de collision
- comment planifier un chemin ?

Introduction

Problématique

- à partir de l'espace libre de collision
- comment planifier un chemin ?

Objectifs de la séance

- rappels sur la recherche dans un graphe
- planification déterministe
- A*

02

Recherche dans un graphe

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche

- recherche un nœud
- deux méthodes :
  - recherche en profondeur d'abord (*Depth-First Search*)
  - recherche en largeur d'abord (*Breadth-First Search*)

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche

- recherche un nœud
- deux méthodes :
  - recherche en profondeur d'abord (*Depth-First Search*) $\rightarrow$ pile
  - recherche en largeur d'abord (*Breadth-First Search*)

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche

- recherche un nœud
- deux méthodes :
  - recherche en profondeur d'abord (*Depth-First Search*) $\rightarrow$ pile
  - recherche en largeur d'abord (*Breadth-First Search*) $\rightarrow$ file

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche

- recherche un nœud
- deux méthodes :
  - recherche en profondeur d'abord (*Depth-First Search*) $\rightarrow$ pile
  - recherche en largeur d'abord (*Breadth-First Search*) $\rightarrow$ file

Recherche de chemin

- recherche des nœuds départ et arrivée
- remontée jusqu'à un parent commun
- un seul chemin

Recherche dans un arbre

Arbre

- graphe sans cycle
- un nœud racine

Recherche

- recherche un nœud
- deux méthodes :
  - recherche en profondeur d'abord (*Depth-First Search*) $\rightarrow$ pile
  - recherche en largeur d'abord (*Breadth-First Search*) $\rightarrow$ file

Recherche de chemin

- recherche des nœuds départ et arrivée
- remontée jusqu'à un parent commun
- un seul chemin

Recherche dans un graphe

Graphe

- pas de racine
- possibilité de cycles
- plusieurs chemins

Recherche dans un graphe

Graphe

- pas de racine
- possibilité de cycles
- plusieurs chemins

Recherche de nœud

- comme un arbre (choix d'une racine)
- sans répétition (marquage)

Recherche dans un graphe

Graphe

- pas de racine
- possibilité de cycles
- plusieurs chemins

Recherche de nœud

- comme un arbre (choix d'une racine)
- sans répétition (marquage)

Recherche de chemin

- nœud initial : départ ou arrivée
- largeur d'abord

Dijkstra

Graphe avec cout

- cout associé à un arc
- nombre d'arc $\neq$ cout

Dijkstra

Graphe avec cout

- cout associé à un arc
- nombre d'arc $\neq$ cout

Algorithme de Dijkstra

- modification de *Breadth-First Search* $\rightarrow$ *Best-First Search*
- inclusion du cout dans la recherche
- file $\rightarrow$ file de priorité
- priorité : cout cumulé jusqu'au départ (g)
- mise-à-jour possible des priorités
- mémorisation du meilleur prédécesseur

Dijkstra

Graphe avec cout

- cout associé à un arc
- nombre d'arc $\neq$ cout

Algorithme de Dijkstra

- modification de *Breadth-First Search* $\rightarrow$ *Best-First Search*
- inclusion du cout dans la recherche
- file $\rightarrow$ file de priorité
- priorité : cout cumulé jusqu'au départ (g)
- mise-à-jour possible des priorités
- mémorisation du meilleur prédécesseur

Dijkstra

Graphe avec cout

- cout associé à un arc
- nombre d'arc $\neq$ cout

Algorithme de Dijkstra

- modification de *Breadth-First Search* $\rightarrow$ *Best-First Search*
- inclusion du cout dans la recherche
- file $\rightarrow$ file de priorité
- priorité : cout cumulé jusqu'au départ (g)
- mise-à-jour possible des priorités
- mémorisation du meilleur prédécesseur

Problème

Espace de configuration

- espace continu
- espace libre de collision

Algorithme de recherche

- entrée : arbre ou graphe
- sortie : chemin le plus court

Problème

- définition d'un graphe ou un arbre ?

Problème

Problème

- définition d'un graphe ou un arbre ?

Approches

- décomposition de l'espace
- résolution géométrique
- échantillonnage (semaine prochaine)

03

Décomposition de l’espace

Décomposition de l'espace

Problème

- définir un graphe
- à partir de l'espace libre

Approche

- diviser l'espace en zones $\longrightarrow$ nœuds
- relier les zones $\longrightarrow$ arcs

Techniques

- pavage régulier
- cellules de Voronoi
- bandes
- ...

Pavage régulier

Définition des zones

- discrétisation régulière sur chaque dimension
- grille (produit cartésien des intervales)
- zones (nœuds) : cellules de la grille

Pavage régulier

Définition des zones

- discrétisation régulière sur chaque dimension
- grille (produit cartésien des intervales)
- zones (nœuds) : cellules libres de la grille

Pavage régulier

Définition des zones

- discrétisation régulière sur chaque dimension
- grille (produit cartésien des intervales)
- zones (nœuds) : cellules libres de la grille

Définition des liens

- contigüité entre les zones
- partage d'un côté ou d'un sommet (ou d'une arête)

Pavage régulier

Définition des zones

- discrétisation régulière sur chaque dimension
- grille (produit cartésien des intervales)
- zones (nœuds) : cellules libres de la grille

Définition des liens

- contigüité entre les zones
- partage d'un côté ou d'un sommet (ou d'une arête)

Poids

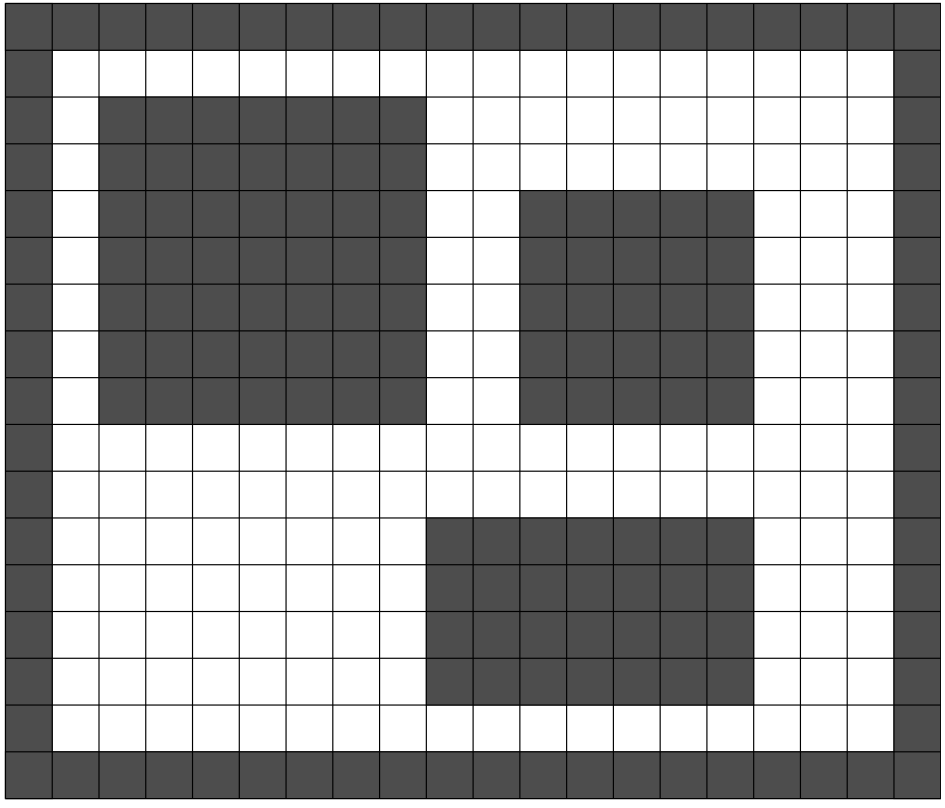
- distance
- cout dépendant de l'application
- parfois sur les sommets

Exemple de planification avec pavage régulier

Espace libre

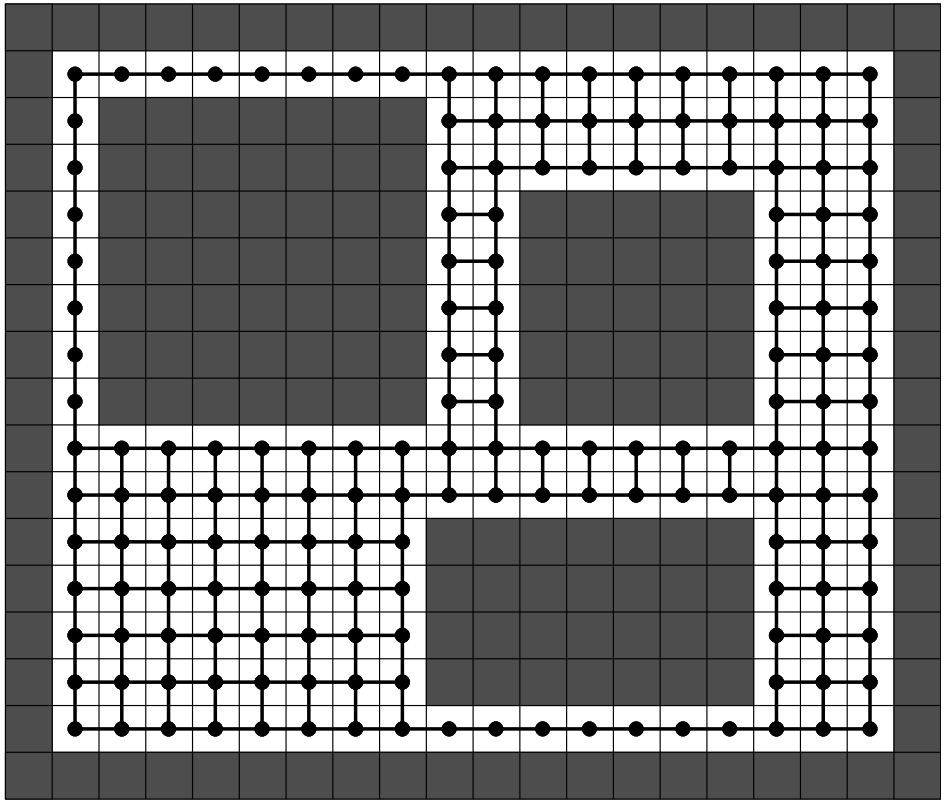
Exemple de planification avec pavage régulier

Décomposition avec une grille



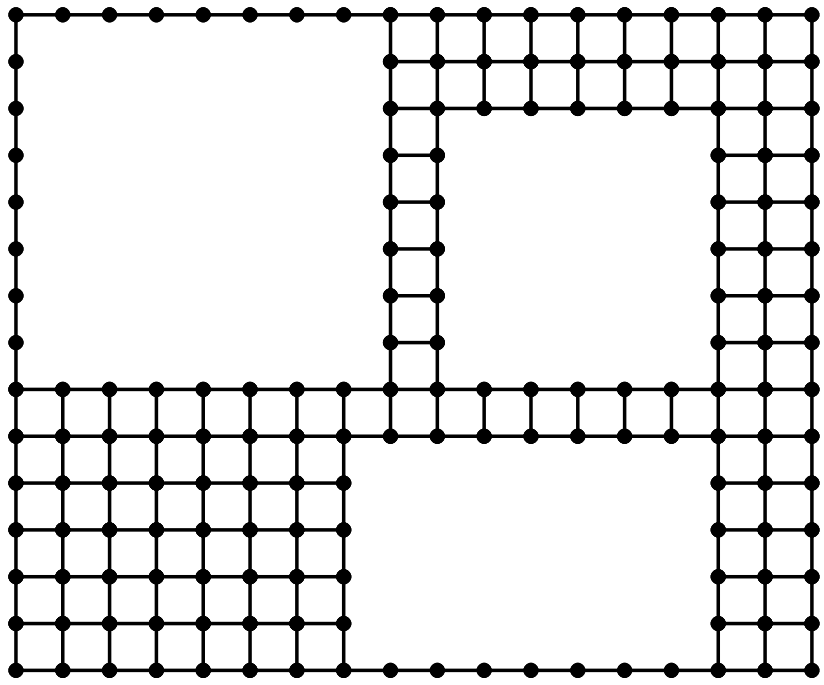
Exemple de planification avec pavage régulier

Graphe de cellules



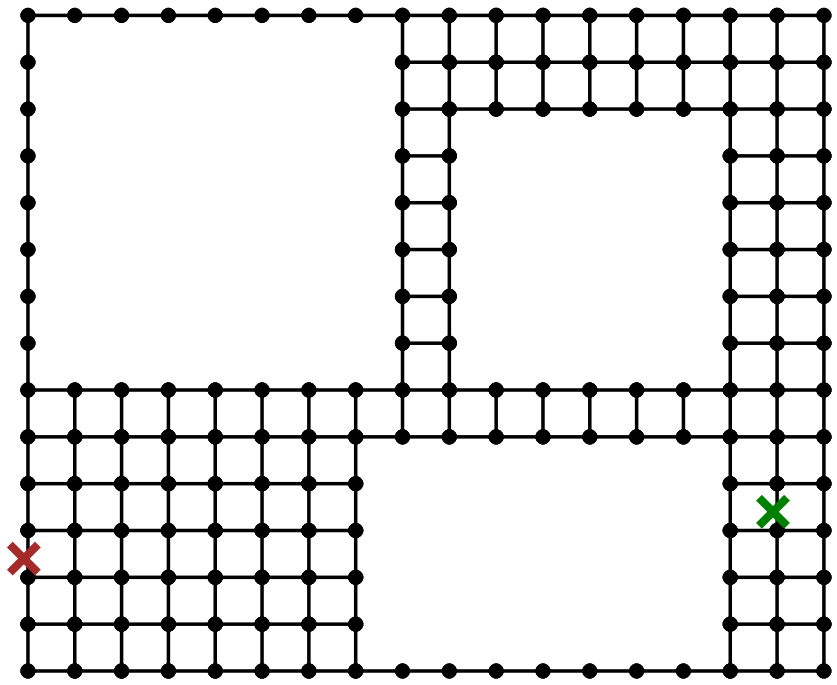
Exemple de planification avec pavage régulier

Graphe de cellules



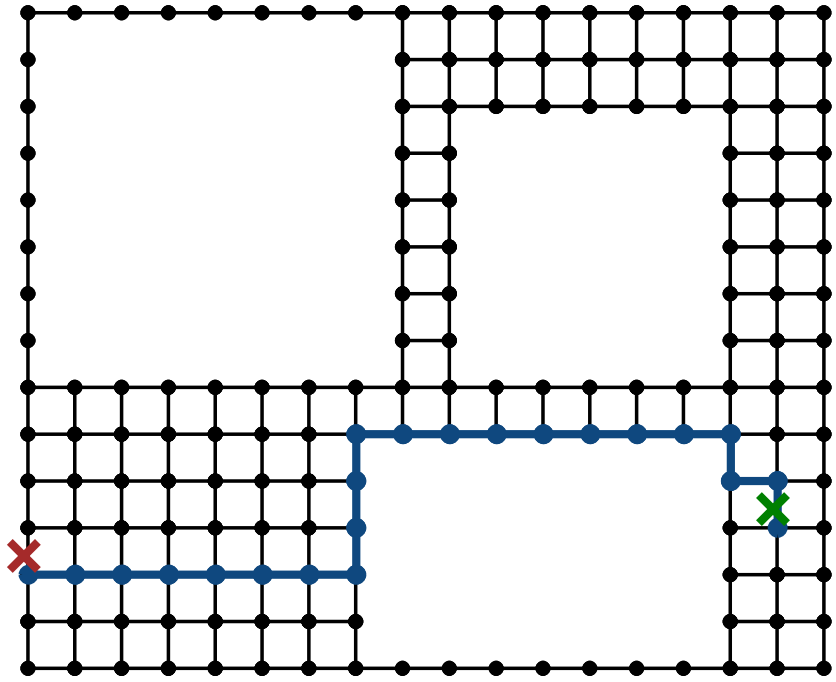
Exemple de planification avec pavage régulier

Position actuelle et but



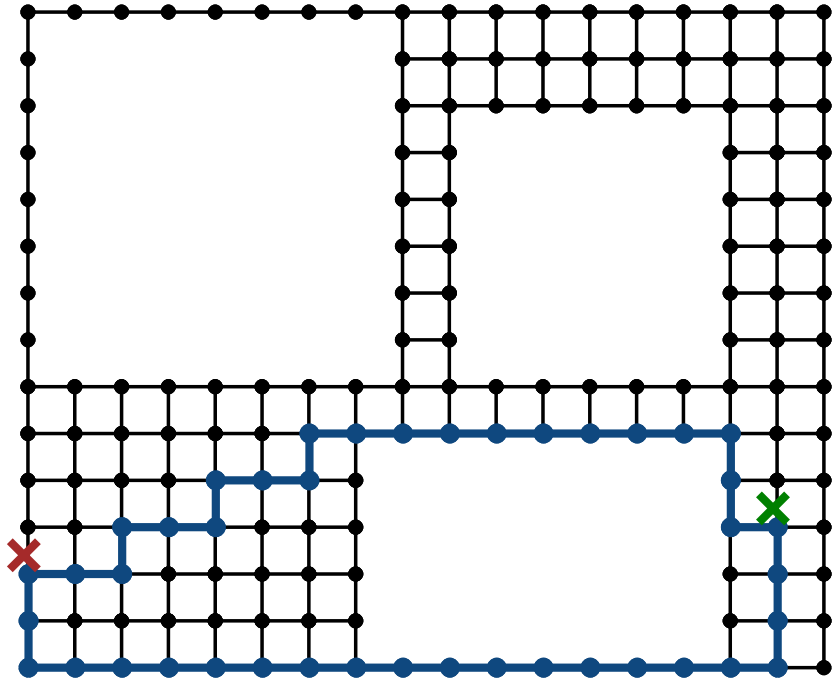
Exemple de planification avec pavage régulier

Planification



Exemple de planification avec pavage régulier

Planification



Pas d'unicité du chemin le plus court !

Exemple de planification avec pavage régulier

Résultat

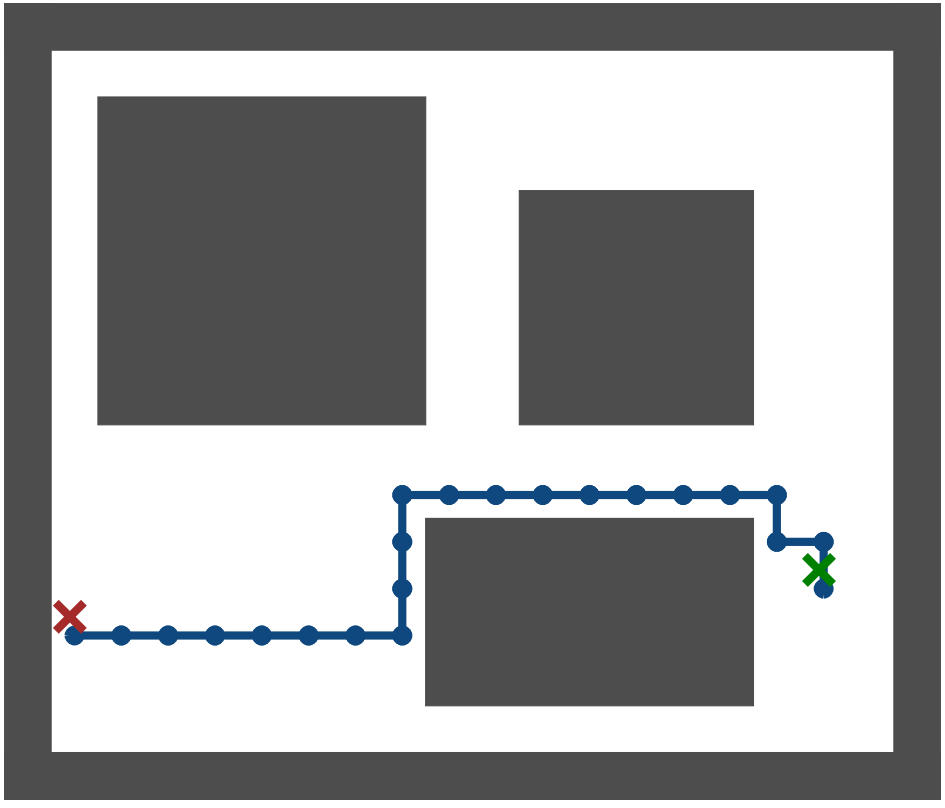


Diagramme de Voronoi

Diagramme de Voronoi

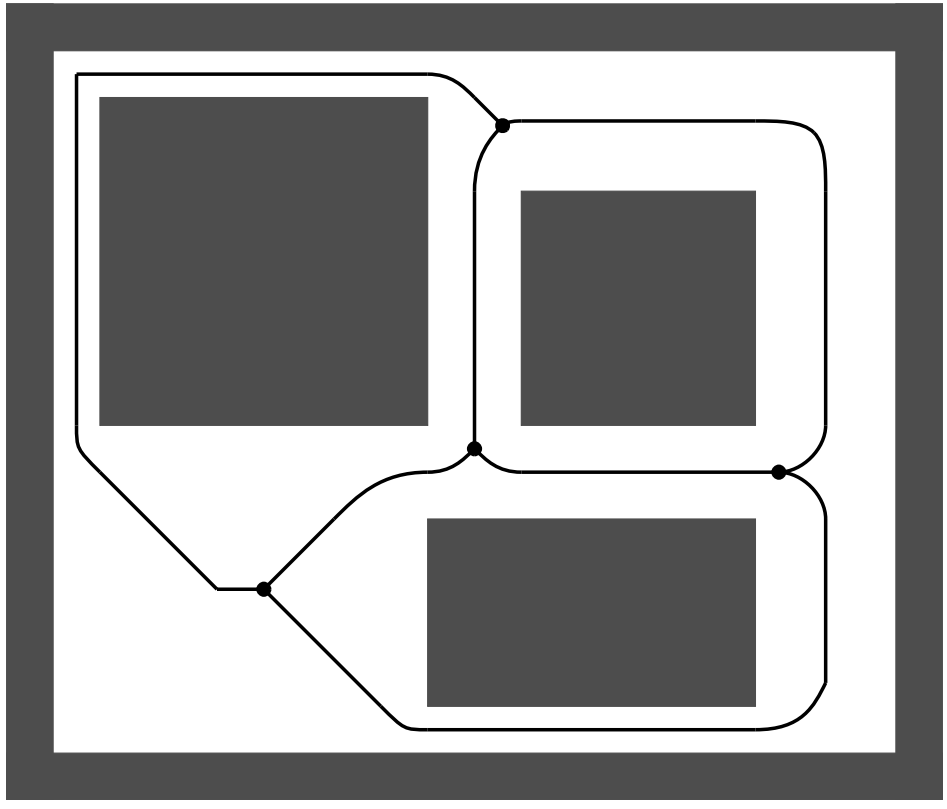
- graphe des points les plus éloignés aux obstacles
- dual de la triangulation de Delaunay
  - centres des cercles circonscrits
  - reliés par contigüité
- extension à des obstacles non-ponctuels
- frontière des cellules de Voronoi

Exemple de planification avec diagramme de Voronoi

Espace libre

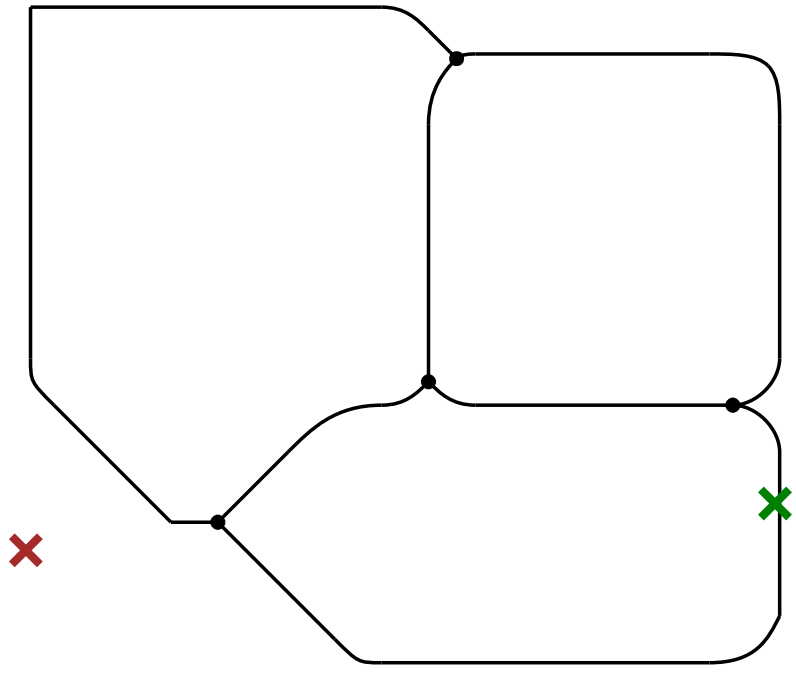
Exemple de planification avec diagramme de Voronoi

Diagramme de Voronoi



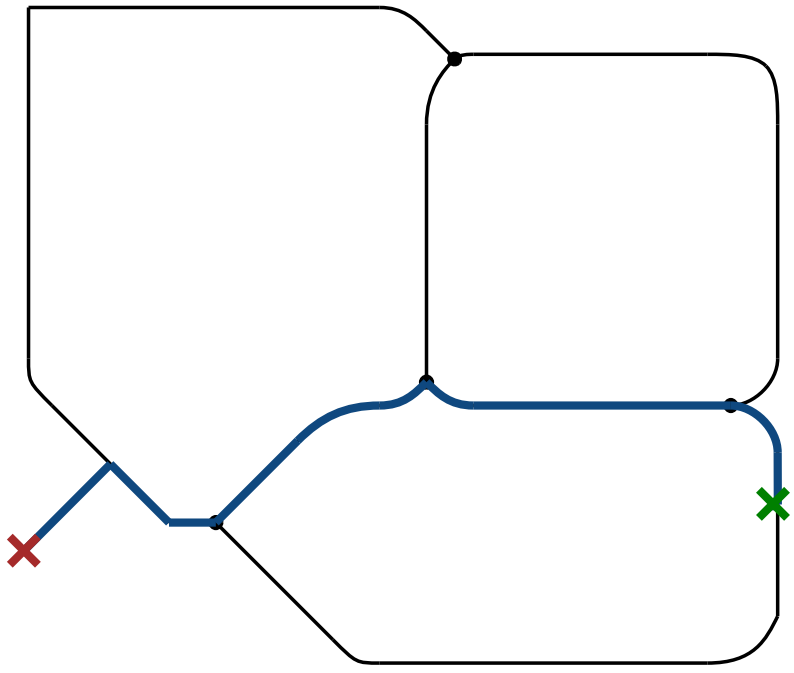
Exemple de planification avec diagramme de Voronoi

Diagramme de Voronoi



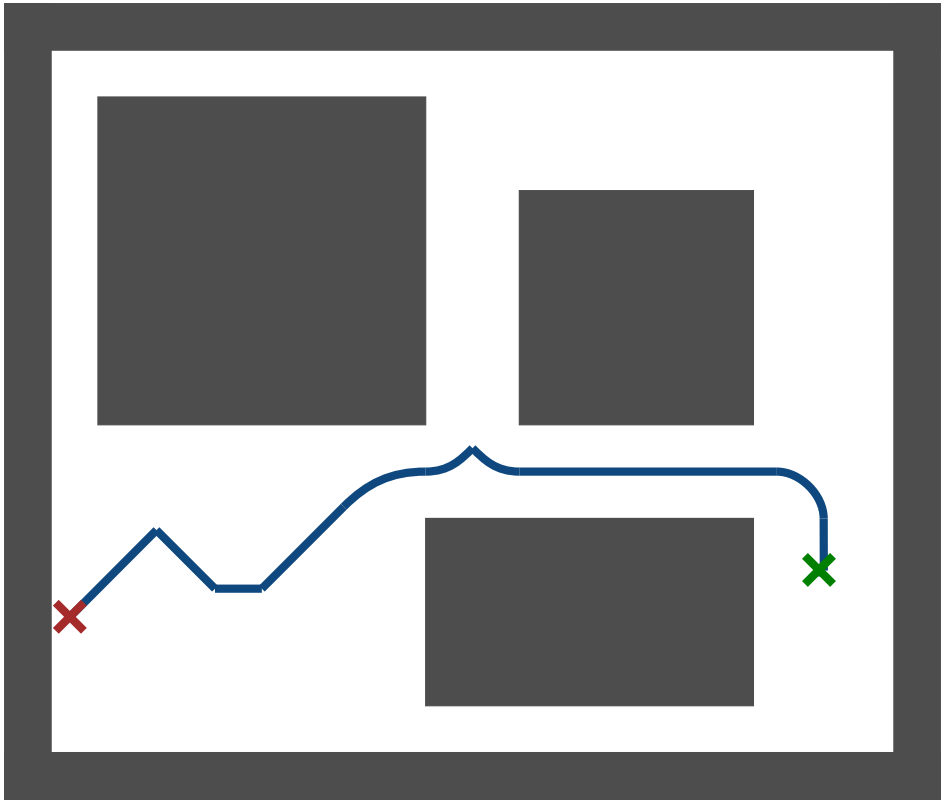
Exemple de planification avec diagramme de Voronoi

Planification dans le graphe



Exemple de planification avec diagramme de Voronoi

Résultat



04

A*

Dijkstra

Dijkstra

- recherche de chemin
- dans un graphe avec des poids
- extension du parcours en largeur

Espace libre

- souvent poids uniformes
- expansion concentrique à partir du départ
- beaucoup de nœuds inutiles

A*

A*

- minimiser le nombre de nœuds consultés
- en orientant la recherche
- ajout d'une heuristique à la priorité ($f = g + h$)

Heuristique

- fonction approchée aidant à résoudre un problème
- en pratique estimation du chemin restant
- condition d'admissibilité :
  - sous-estimation du cout réel
- exemple :
  - distance euclidienne

05

Résolution géométrique

Résolution géométrique

Recherche du plus court chemin

- ligne droite
- longer les obstacles

Résolution exacte

- obstacles polyédraux
- test de visibilité
- développée pour Shakey

Planification avec graphe de visibilité

Graphe de visibilité

- nœuds : sommets des obstacles ;
- arêtes : ssi ligne de vue entre nœuds ;
- position courante et but comme nœuds ;
- recherche dans le graphe.

Résultat

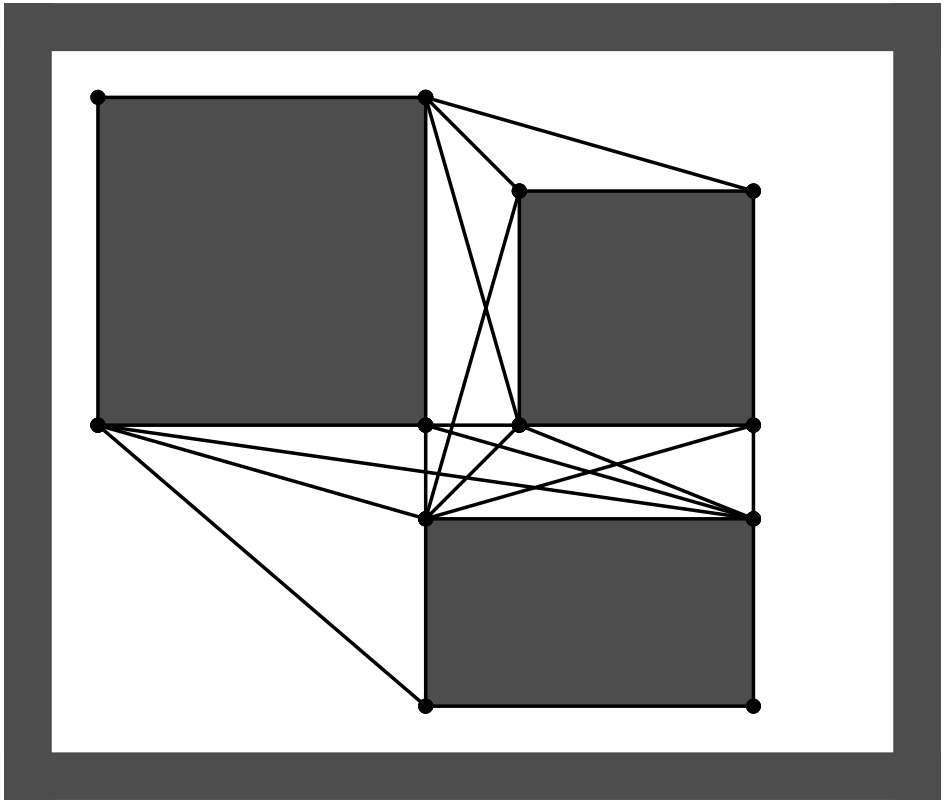
- chemin ;
- optimal en distance ;
- longe les obstacles au plus près ;
- nécessite des obstacles polygonaux/polyédraux.

Exemple

Espace libre

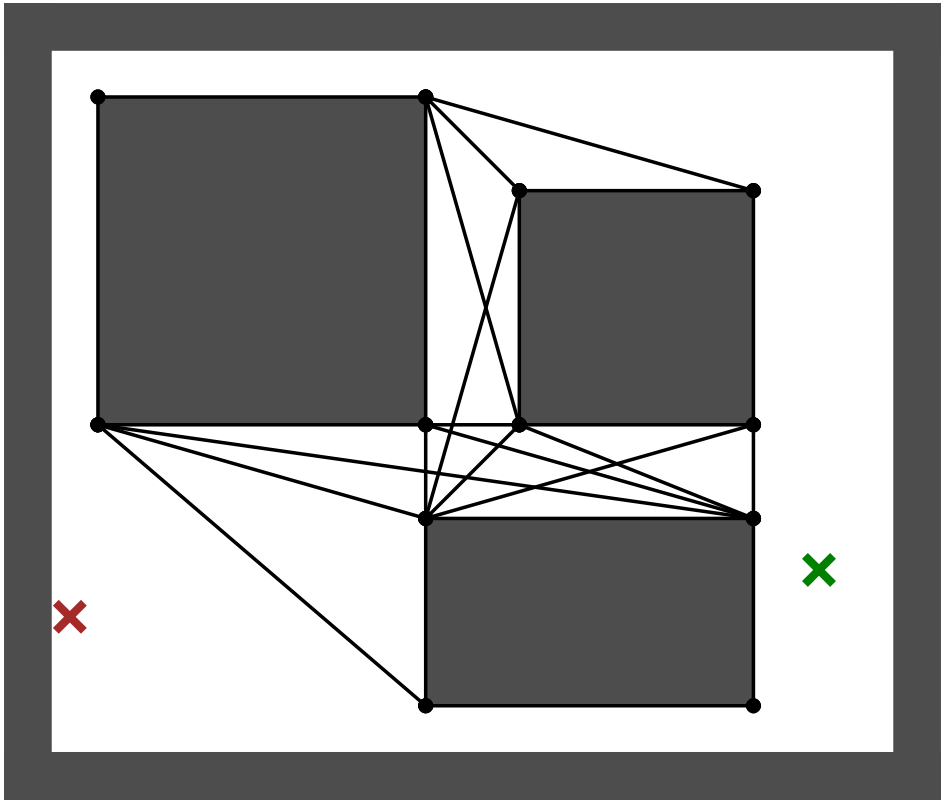
Exemple

Graphe de visibilité des obstacles



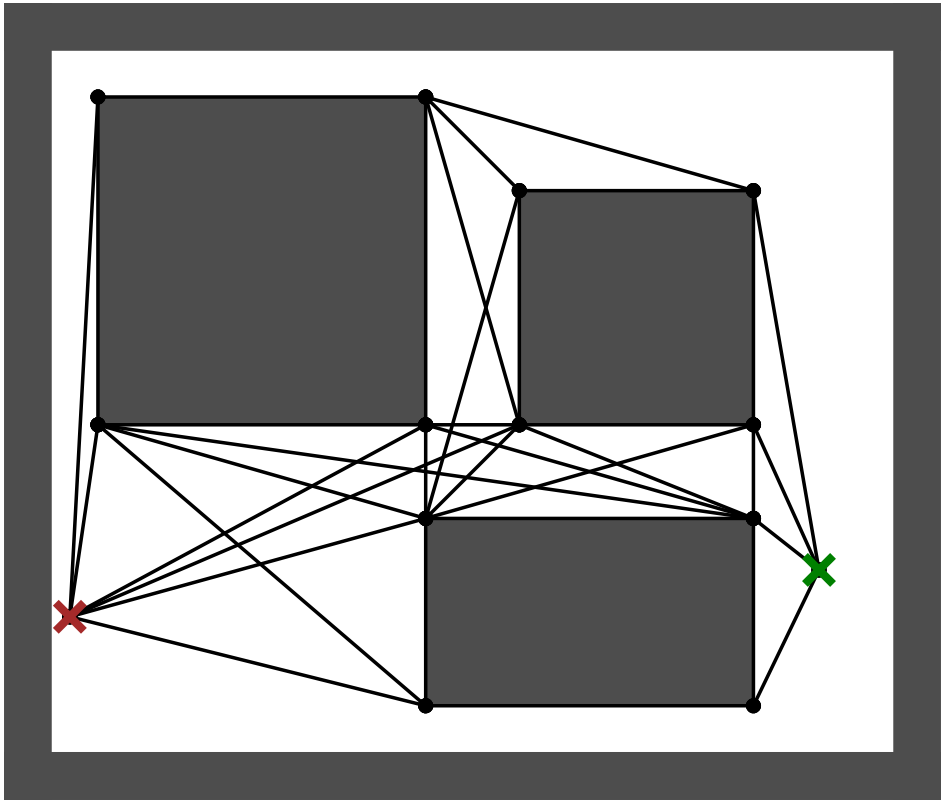
Exemple

Inclusion des extrémités



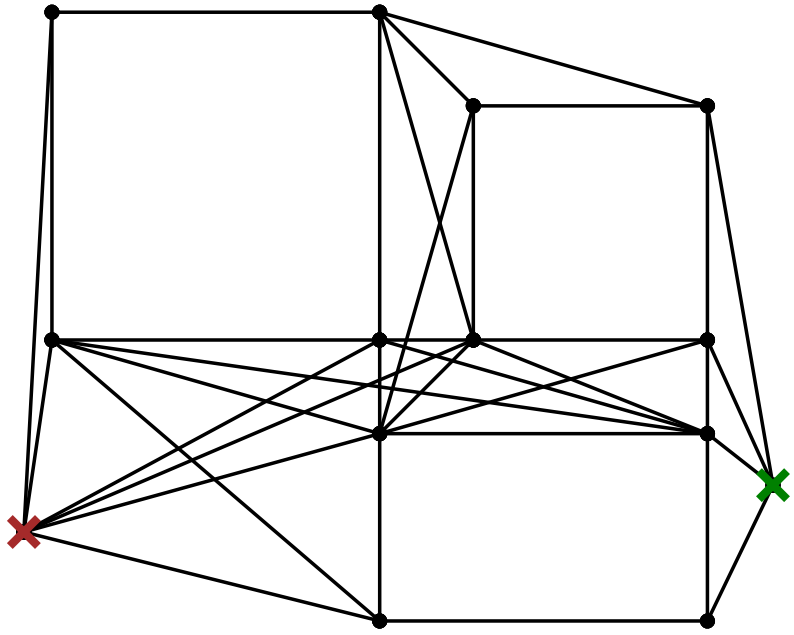
Exemple

Inclusion des extrémités



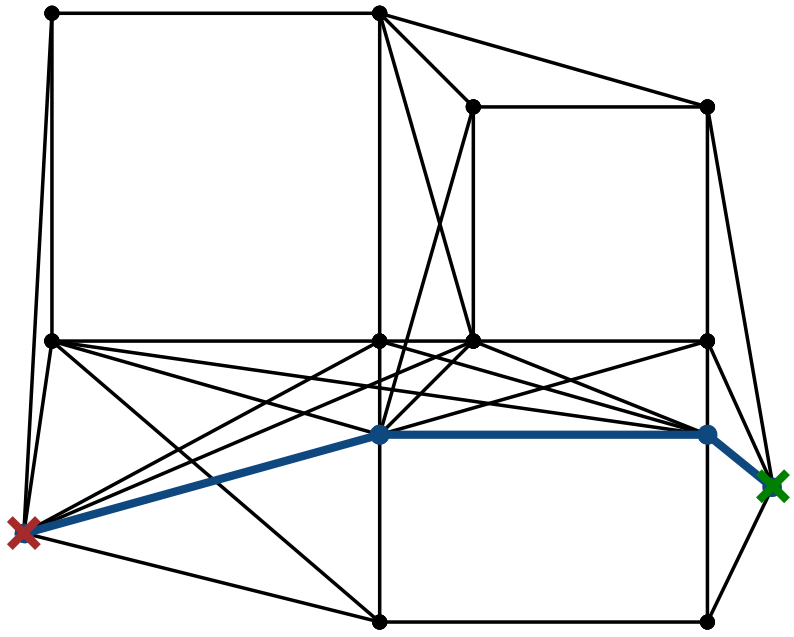
Exemple

Graphe complet



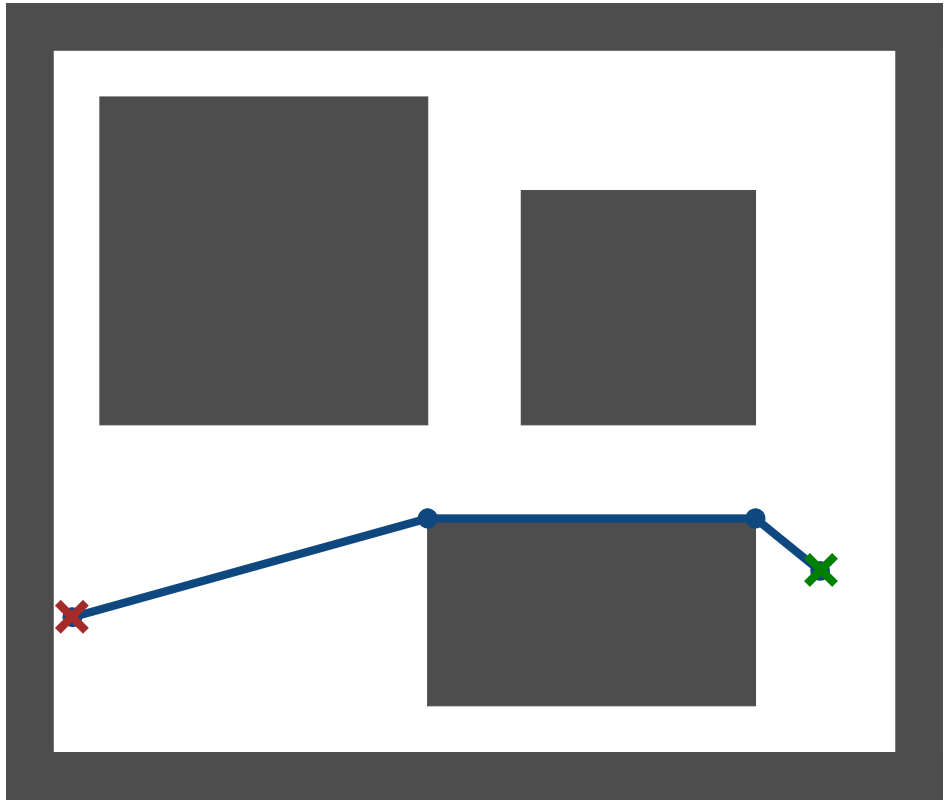
Exemple

Planification (A* ou autre)



Exemple

Résultat



06

Conclusion

Recherche de chemin

Recherche de chemin

- recherche dans un graphe
- en largeur, avec cout
- avec heuristique

Création du graphe

- décomposition de l'espace
- graphe de visibilité

Bibliographie

Algorithmes

- Dijkstra, *A note on two problems in connexion with graphs*, NM, 1959.
- Hart et al, *A Formal Basis for the Heuristic Determination of Minimum Cost Paths*, IEEE Trans. SSC, 1968.
- Lozano-Pérez et Wesley, *An algorithm for planning collision-free paths among polyhedral obstacles*, ACM, 1979.

Livres

- Latombe, *Robot Motion Planning*, Kluwer Academic Publishers 1991.
- Lavalle, *Planning Algorithms*, Cambridge University Press 2006.
- Siciliano et al., *Springer Handbook of Robotics*, Springer 2016.

Merci de votre attention

Des questions ?