Robotique autonome

Planification

Francis Colas

Introduction

Mouvement

- généré par des moteurs :
  - électriques (AC/DC), pneumatiques ou hydrauliques,
  - moteurs pas-à-pas : positions fixes,
  - servo-moteurs : moteur + capteur intégré pour contrôle bas-niveau en position ;
- commandé pour réaliser une trajectoire définie.

Introduction

Mouvement

- généré par des moteurs :
  - électriques (AC/DC), pneumatiques ou hydrauliques,
  - moteurs pas-à-pas : positions fixes,
  - servo-moteurs : moteur + capteur intégré pour contrôle bas-niveau en position ;
- commandé pour réaliser une trajectoire définie.

Planification de mouvement

- calculer une trajectoire,
- pour atteindre un but,
- en respectant des contraintes (évitement des collisions, limites de vitesse, . . .).

Introduction

Mouvement

- généré par des moteurs :
  - électriques (AC/DC), pneumatiques ou hydrauliques,
  - moteurs pas-à-pas : positions fixes,
  - servo-moteurs : moteur + capteur intégré pour contrôle bas-niveau en position ;
- commandé pour réaliser une trajectoire définie.

Planification de mouvement

- calculer une trajectoire,
- pour atteindre un but,
- en respectant des contraintes (évitement des collisions, limites de vitesse, . . .).

Objectifs de la séance

- rappel sur l'espace de configuration ;
- algorithmes de planification.

1

Espace de configuration

Espace de configuration

Espace de travail

- espace dans lequel évolue le robot :
  - robot à roues standard : 2D ;
  - robot normal : 3D.

Espace de configuration

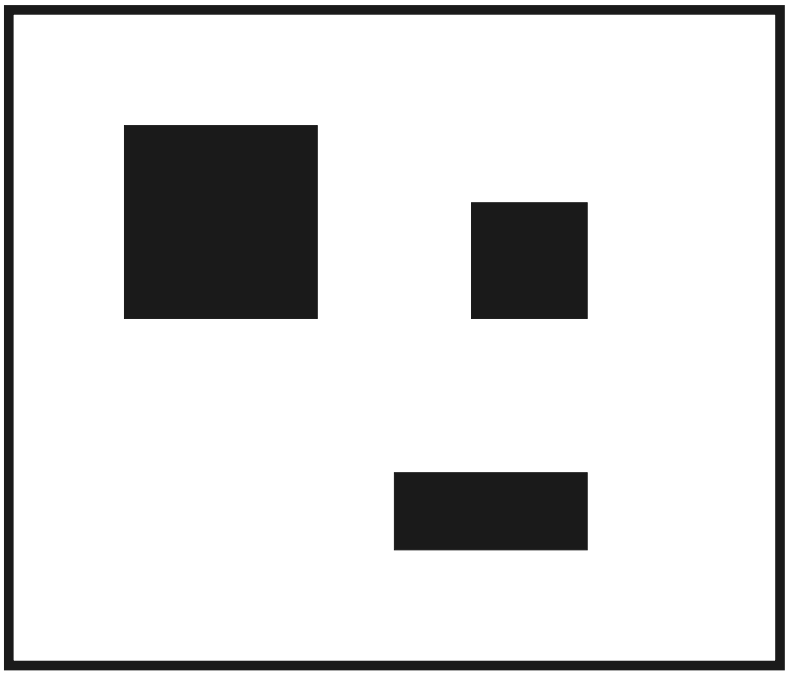
- ensemble des configurations réalisables ;
- prend en compte les contraintes physiques du robot.

Espace libre

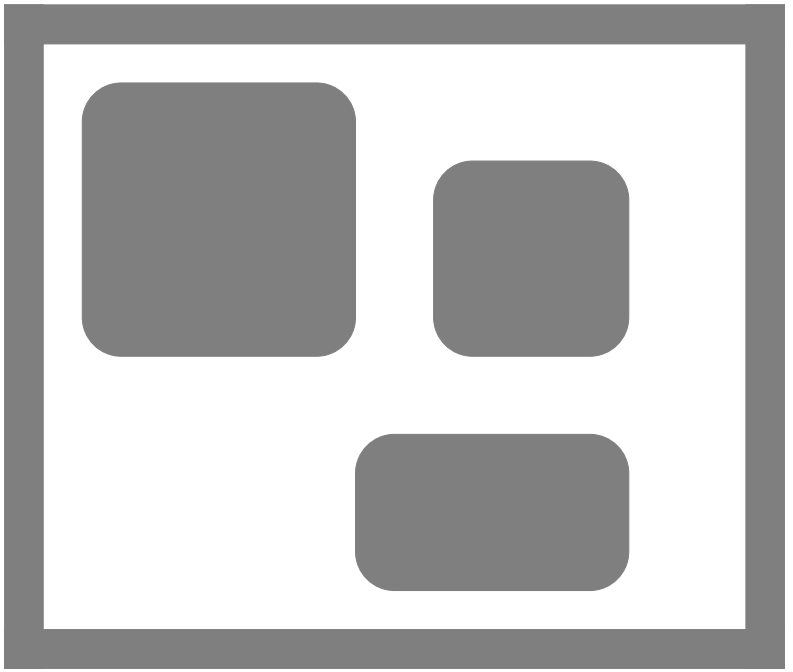
- ensemble des configurations réalisables sans collision ;
- prend en compte les collisions avec les obstacles.

Exemple d'un robot mobile circulaire

Espace de travail

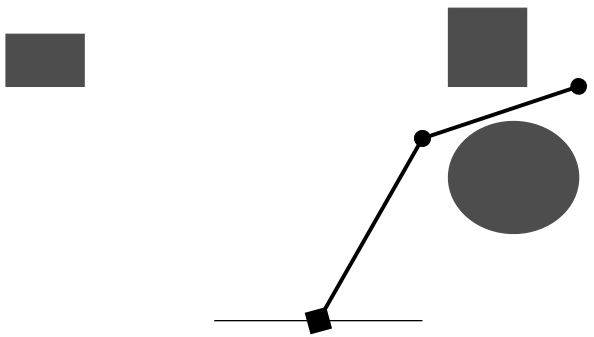


Espace libre

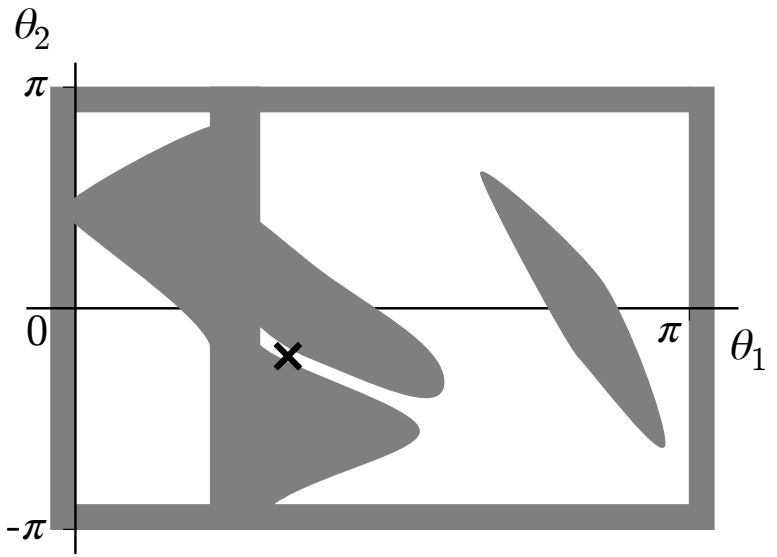


Exemple d'un bras robotique

Espace de travail



Espace libre



2

Algorithmes de planification

Algorithmes

Approches

- décomposition de l'espace :
  - grille,
  - pavage ;
- échantillonnage ;
- champs de potentiel ;
- résolution géométrique ;
- optimisation d'un chemin existant.

Planification dans une grille

Planification dans une grille

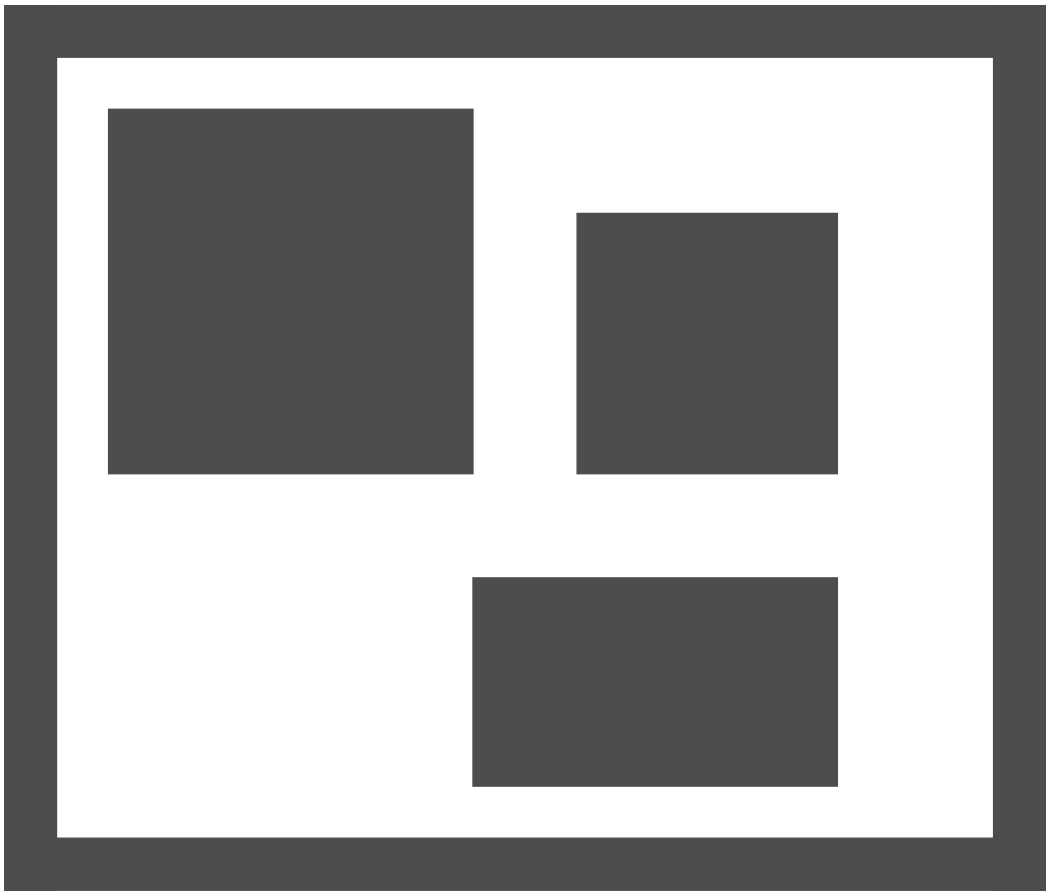
- adapté aux grilles d'occupation ;
- graphe de voisinage ;
- recherche dans un graphe (Dijkstra, A^*).

Résultat

- chemin ;
- orientation discrétisée ;
- pas nécessairement optimal en distance ;
- complexité importante en moyenne ou grande dimension.

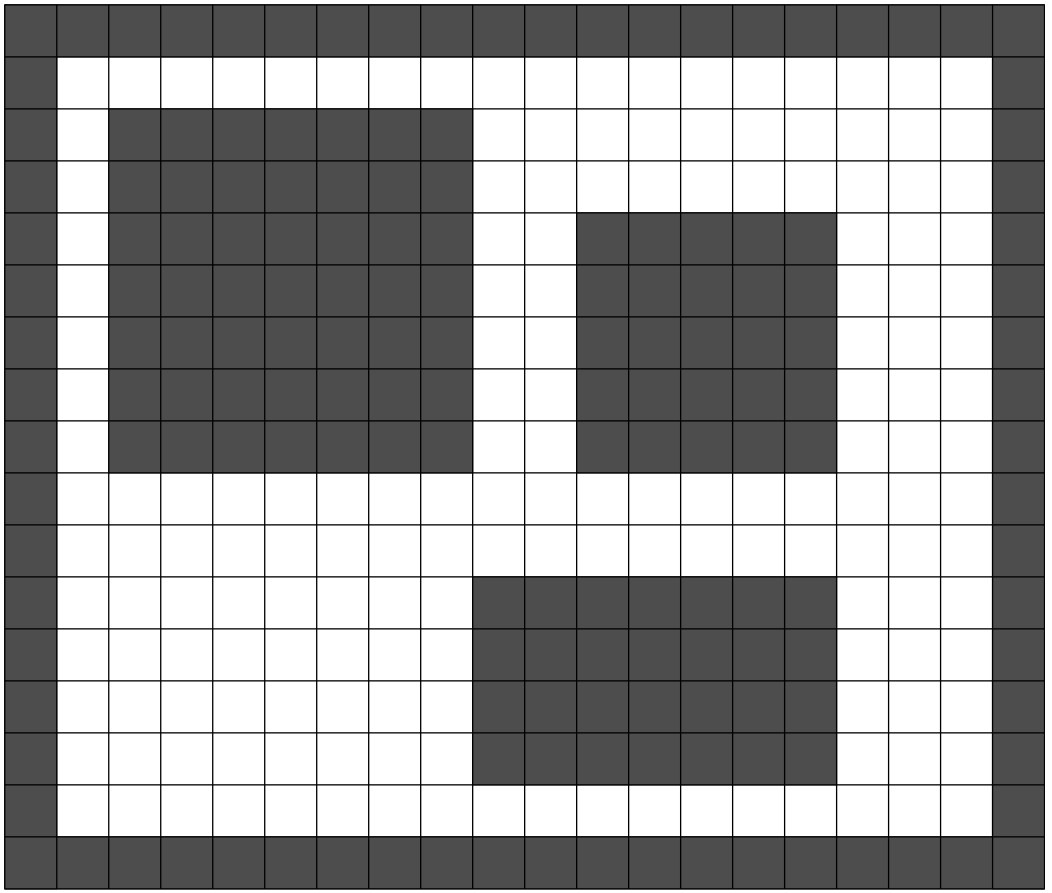
Exemple

Espace libre



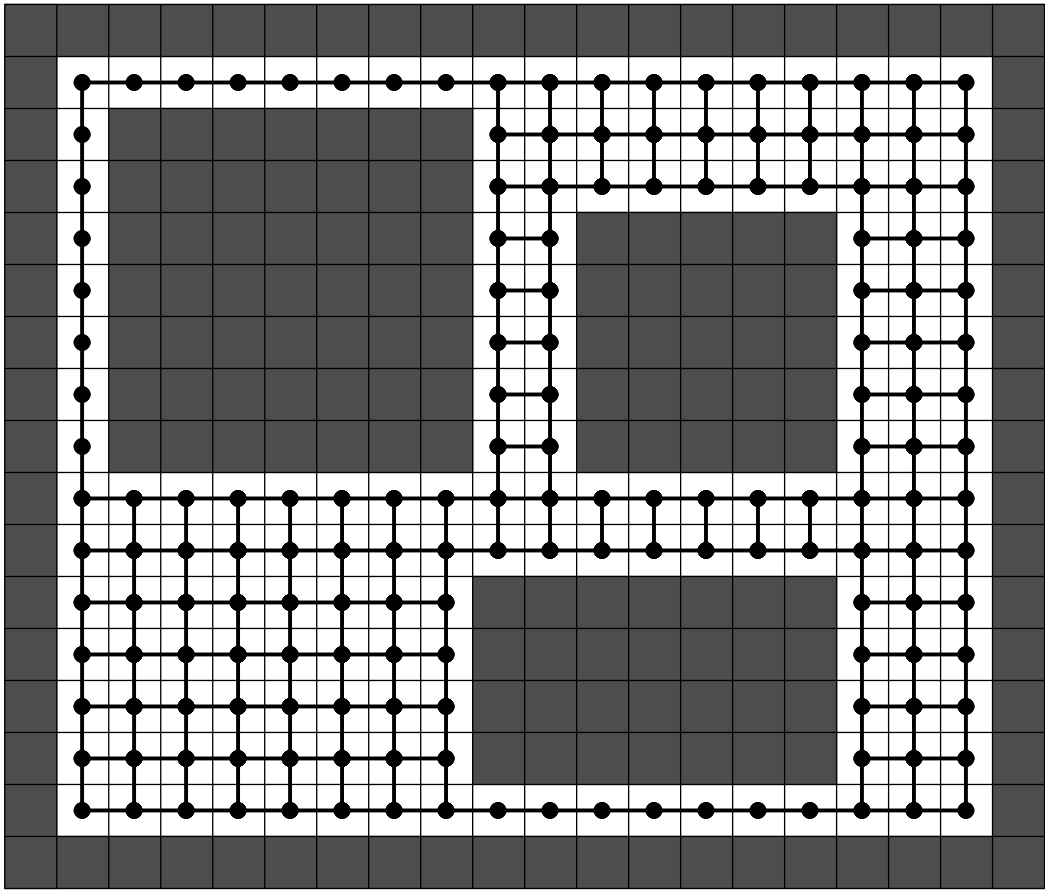
Exemple

Décomposition avec une grille



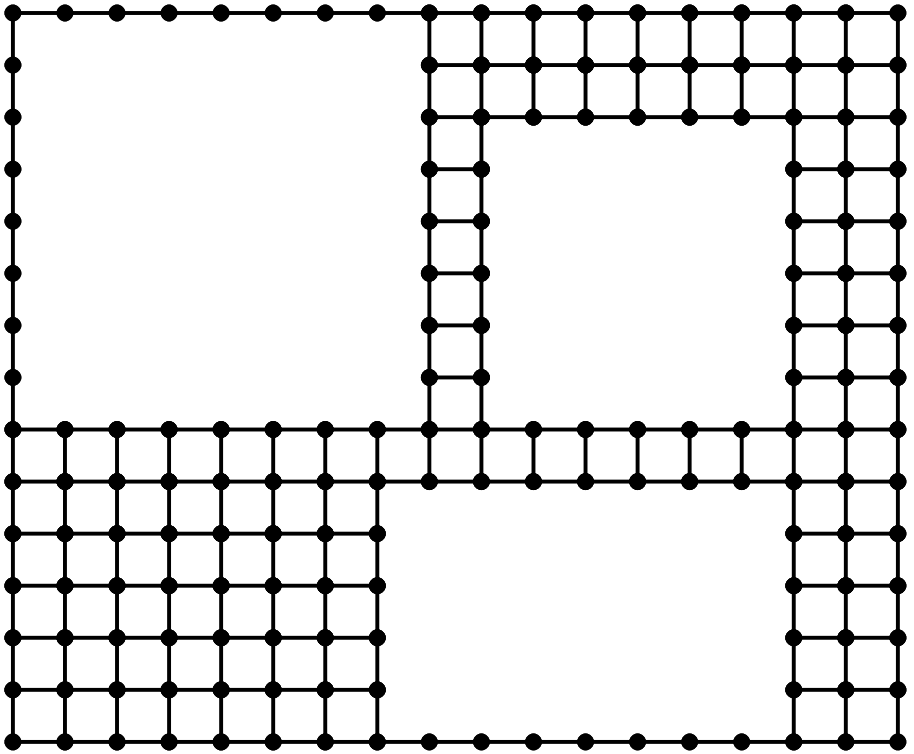
Exemple

Graphe de cellules



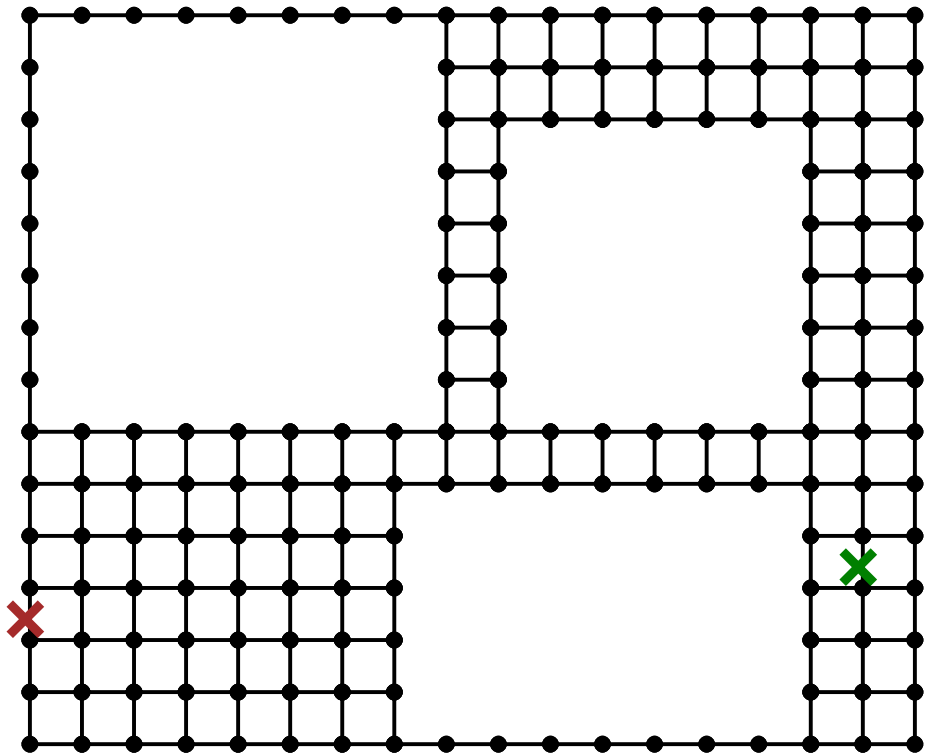
Exemple

Graphe de cellules



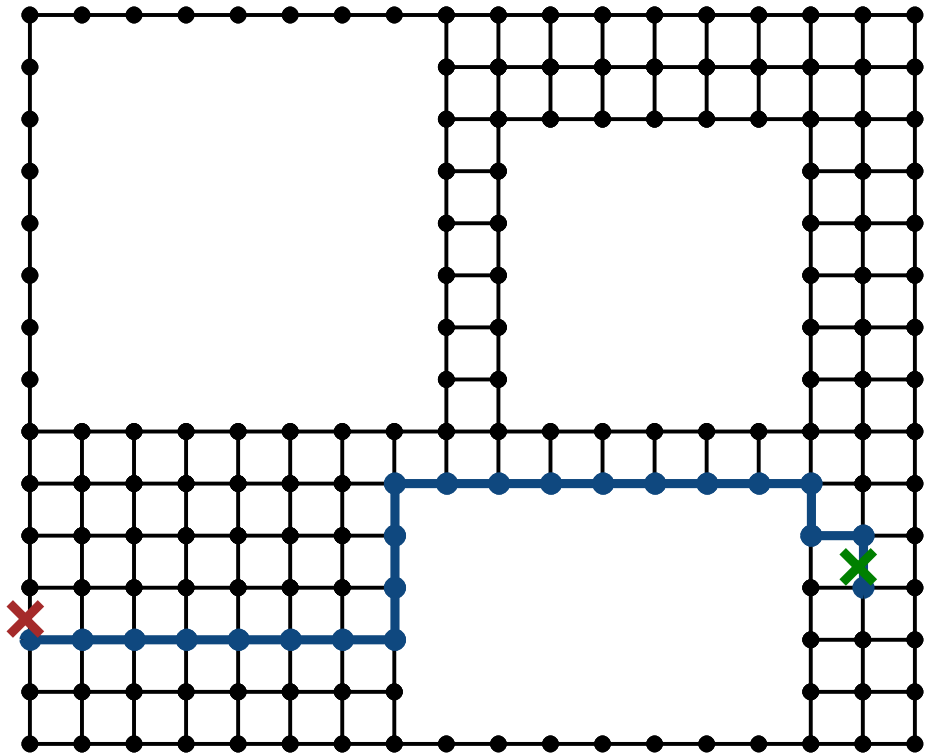
Exemple

Position actuelle et but



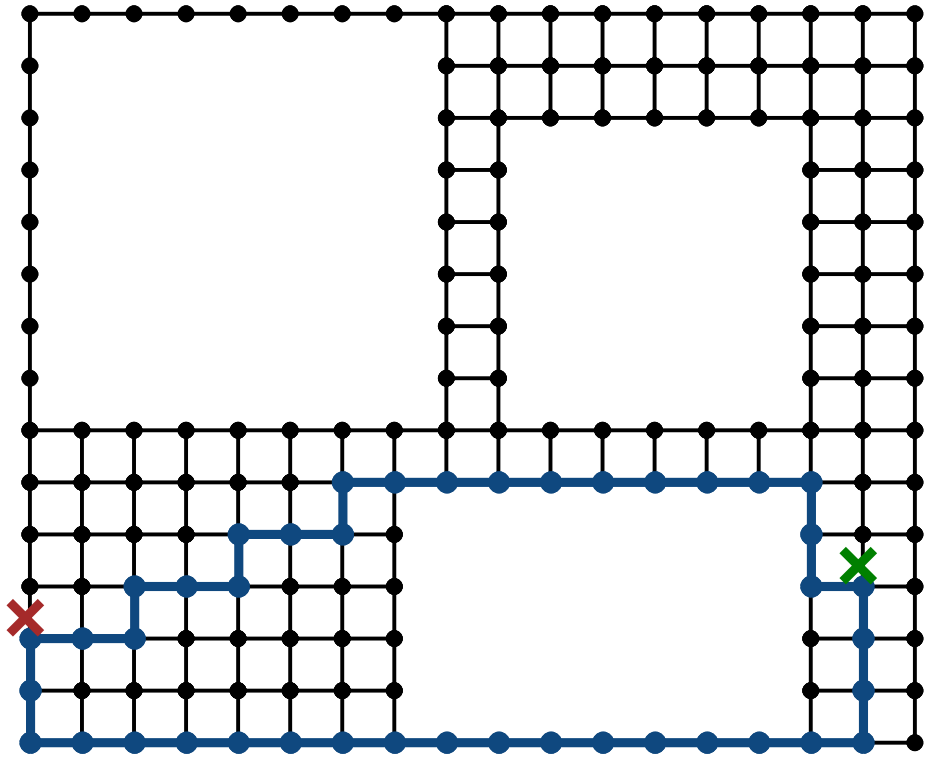
Exemple

Planification (A* ou autre)



Exemple

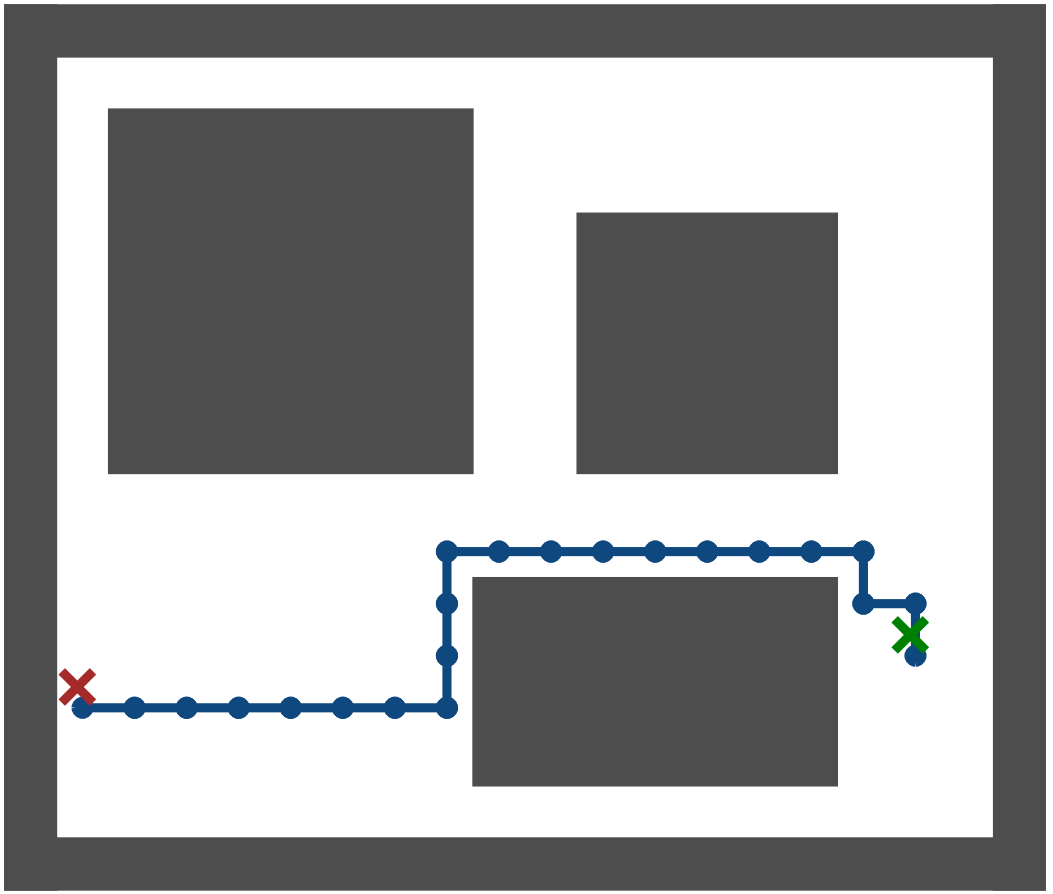
Planification (A* ou autre)



Pas d'unicité du chemin le plus court !

Exemple

Résultat



Planification par décomposition de l'espace

Diagramme de Voronoi

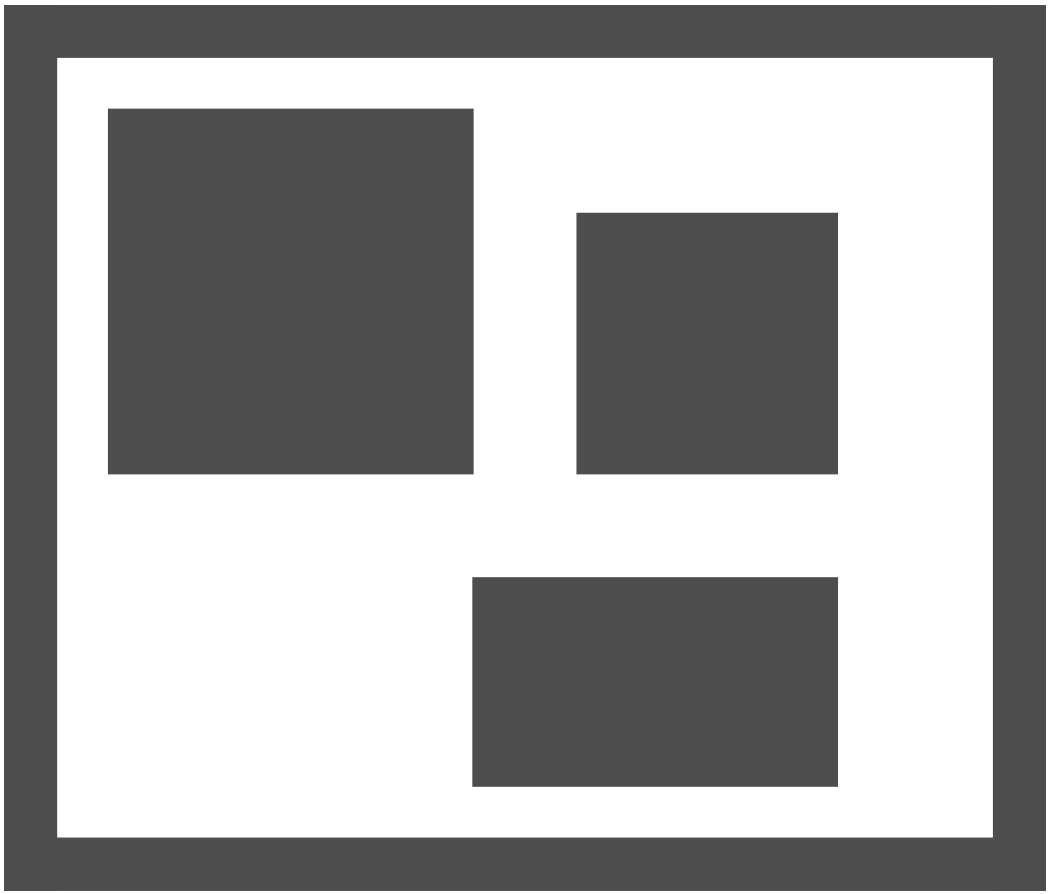
- pavage de l'espace à partir de la distance aux obstacles ;
- dual de la triangulation de Delaunay ;
- suivi des frontières des cellules.

Résultat

- chemin ;
- le plus éloigné possible des obstacles ;
- diagramme compliqué à construire en grandes dimensions ;
- pas optimal en distance.

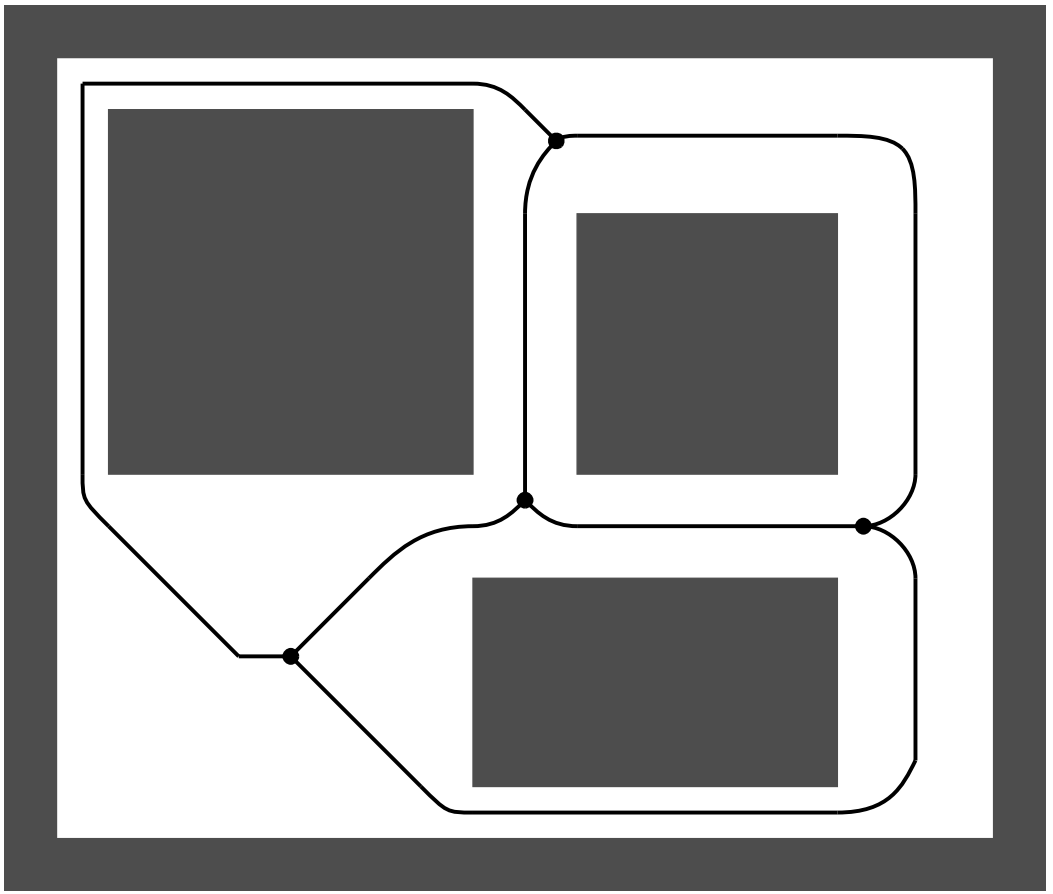
Exemple

Espace libre



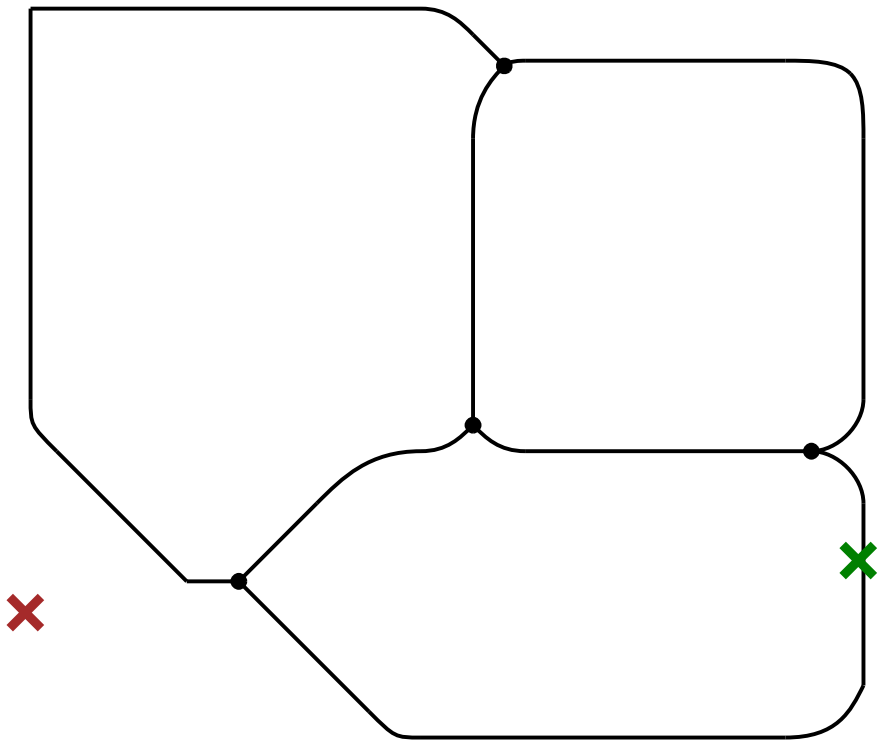
Exemple

Diagramme de Voronoi



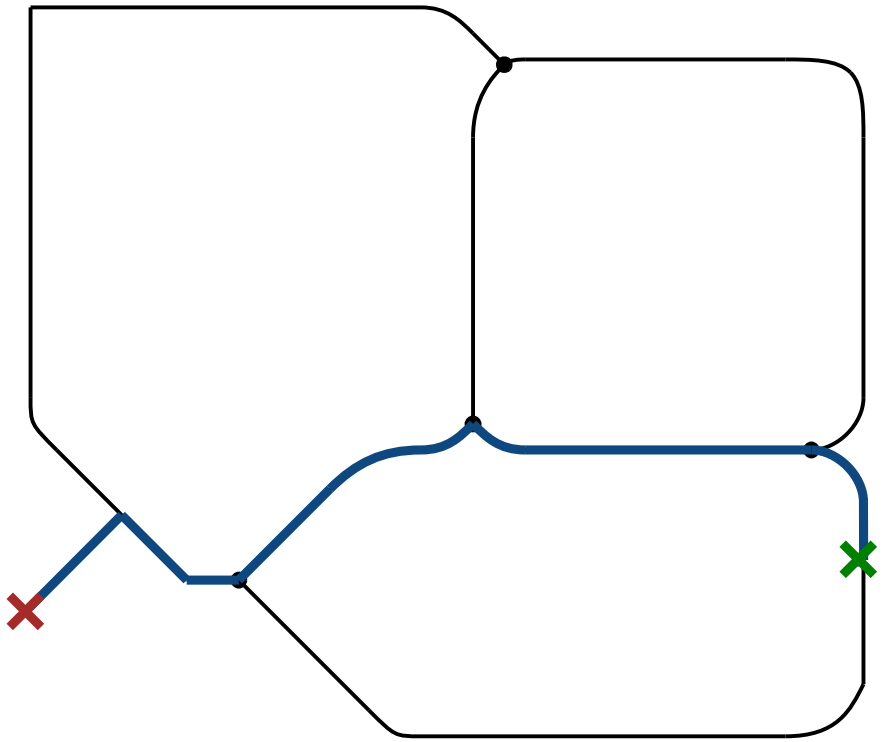
Exemple

Diagramme de Voronoi



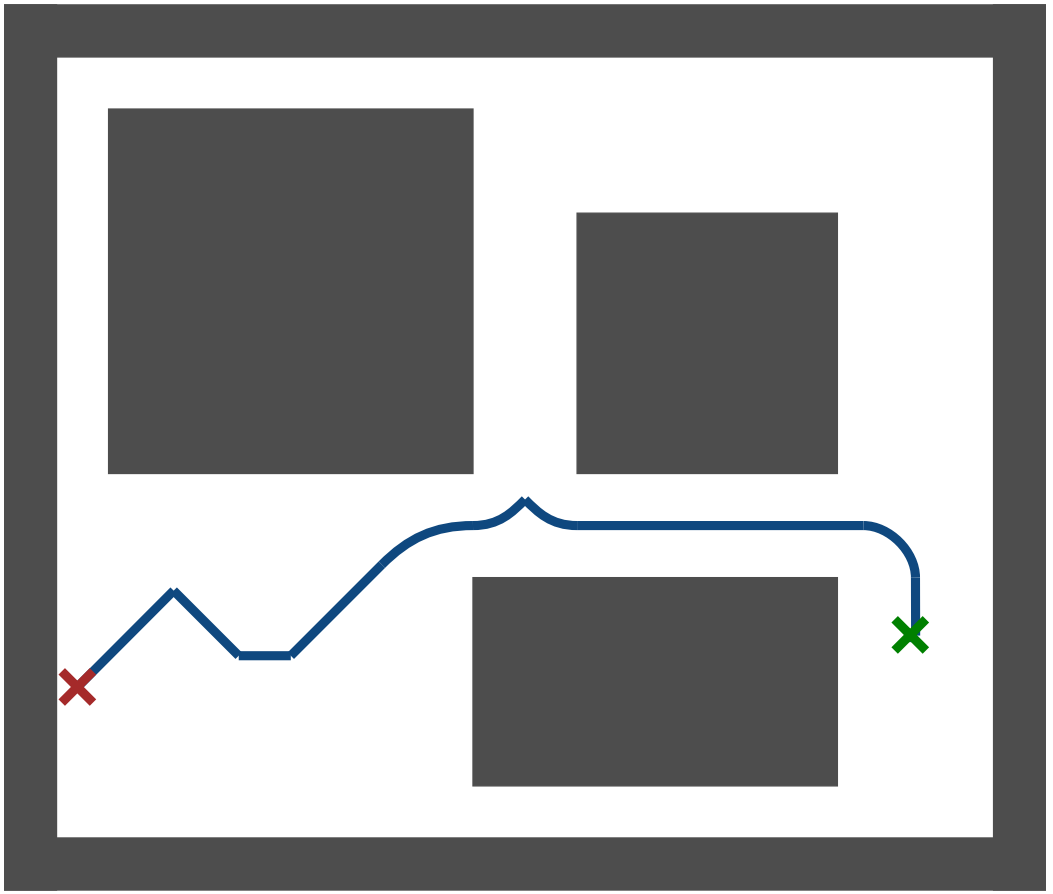
Exemple

Planification dans le graphe



Exemple

Résultat



Planification par échantillonnage

Rapidly-expanding Random Trees

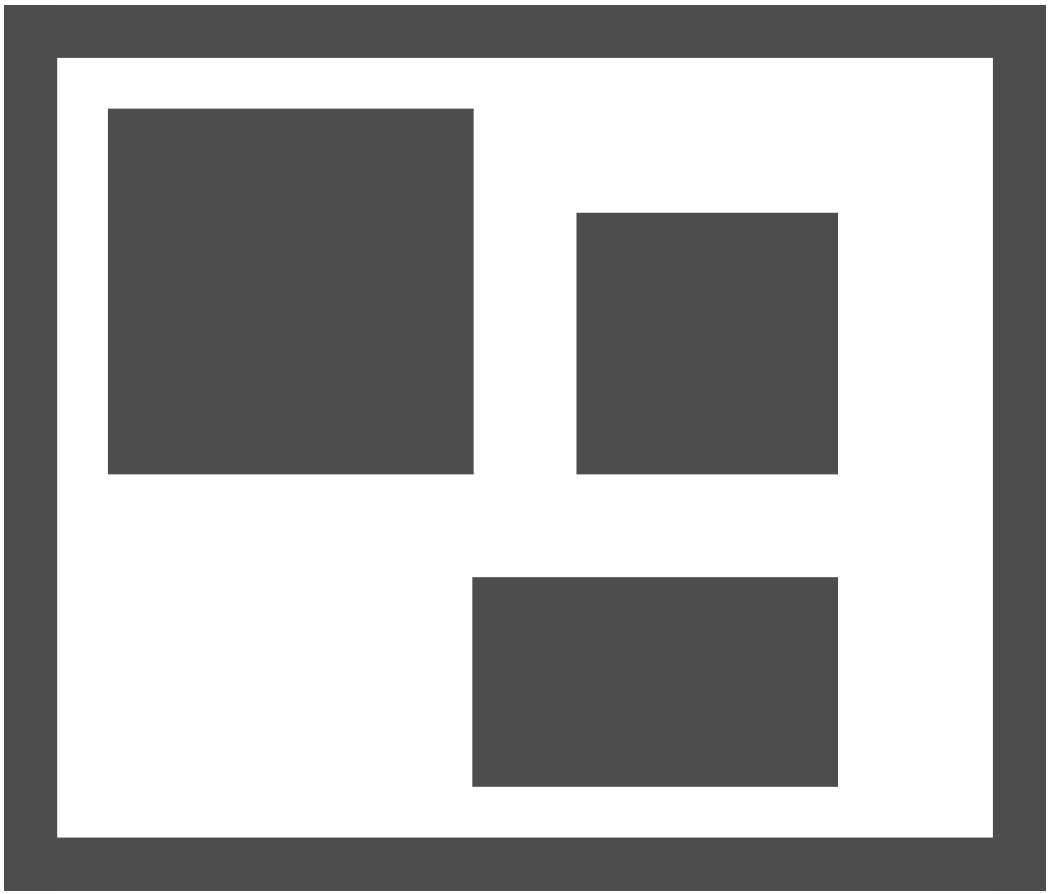
- RRT, RRT*...
- échantillonnage de l'espace ;
- création d'un arbre de connectivité ;
- au moins jusqu'à trouver le but.

Résultat

- chemin ;
- optimal en distance pour un nombre infini d'échantillons ;
- rapide et *anytime* dès que le chemin est trouvé.

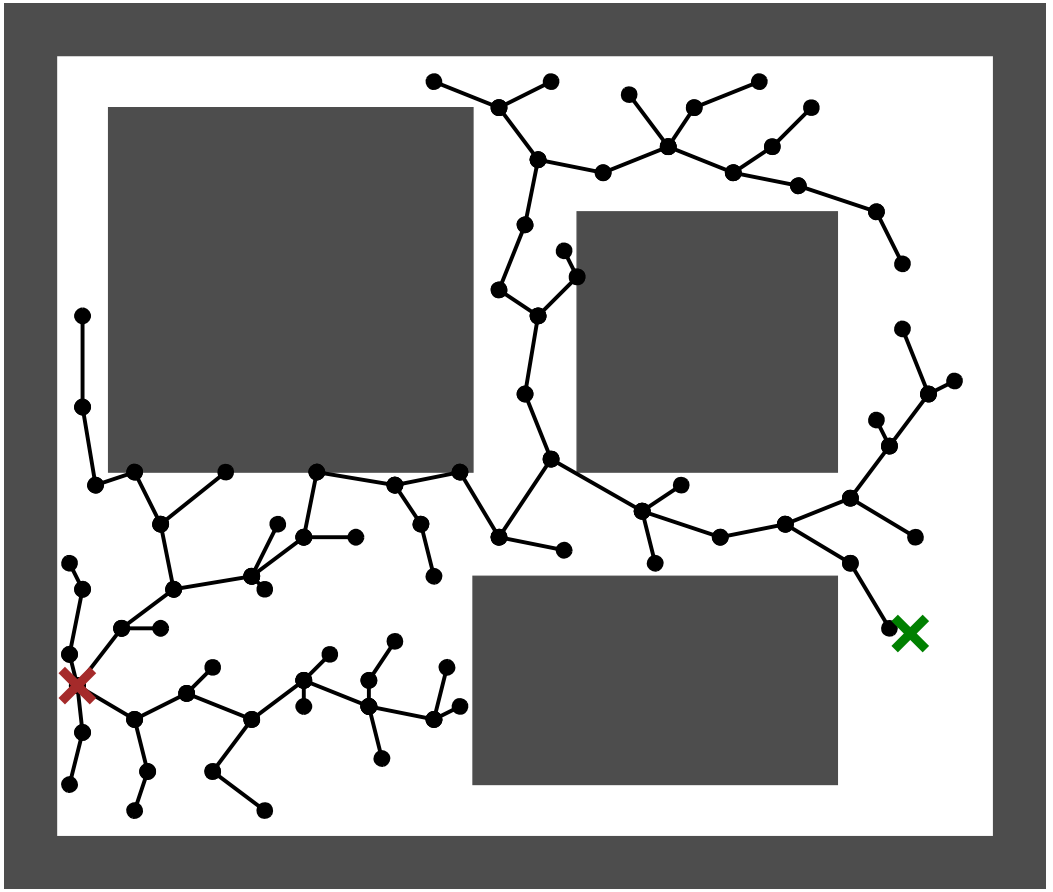
Exemple

Espace libre



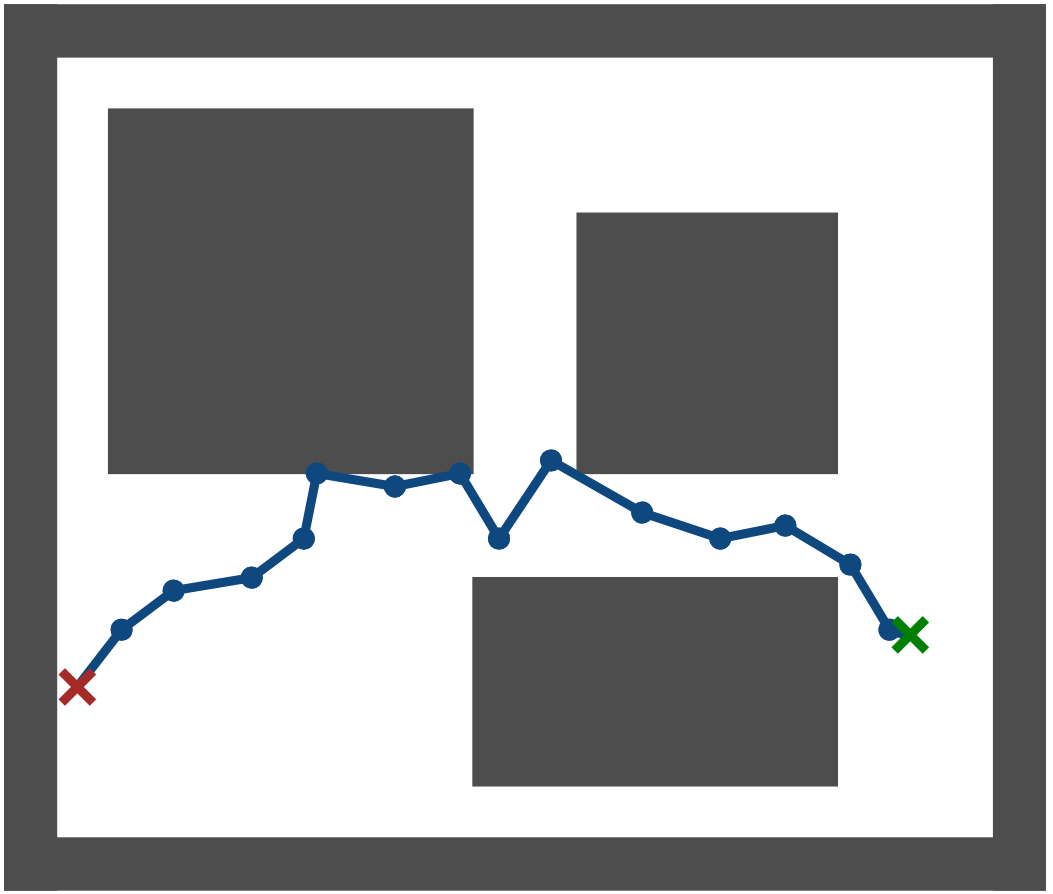
Exemple

Expansion du graphe



Exemple

Résultat



Algorithme

RRT

$V \leftarrow \{x_{\text{init}}\}; E \leftarrow \emptyset$
for $i = 1, \ldots, n$ **do**
  $x_{\text{rand}} \leftarrow \texttt{SampleFree}()$
  $x_{\text{nearest}} \leftarrow \texttt{Nearest}\left(G = (V, E), x_{\text{rand}}\right)$
  $x_{\text{new}} \leftarrow \texttt{Steer}\left(x_{\text{nearest}}, x_{\text{rand}}\right)$
  if $\texttt{CollFree}\left(x_{\text{nearest}}, x_{\text{new}}\right)$ **then**
    $V \leftarrow V \cup \{x_{\text{new}}\}$
    $E \leftarrow E \cup \{(x_{\text{nearest}}, x_{\text{new}})\}$
  end if
end for
return $G = (V, E)$

Fonctions

- $\texttt{SampleFree}()$: échantillonne un point dans l'espace libre ;
- $\texttt{Nearest}(G, x)$: point le plus proche de x dans le graphe G ;
- $\texttt{Steer}(x_1, x_2)$: point à une distance donnée de x_1 vers x_2 ;
- $\texttt{CollFree}(x_1, x_2)$: pas d'obstacle entre x_1 et x_2 ;

Algorithme

Amélioration de RRT

$V \leftarrow \{x_{\text{init}}\}; E \leftarrow \emptyset$
for $i = 1, \ldots, n$ **do**
  $x_{\text{rand}} \leftarrow \texttt{SampleFree}()$
  $x_{\text{nearest}} \leftarrow \texttt{Nearest}\left(G = (V, E), x_{\text{rand}}\right)$
  $x_{\text{new}} \leftarrow \texttt{Steer}\left(x_{\text{nearest}}, x_{\text{rand}}\right)$
  if $\texttt{CollFree}\left(x_{\text{nearest}}, x_{\text{new}}\right)$ **then**
    $X_{\text{near}} \leftarrow \texttt{Near}\left(G = (V, E), x_{\text{rand}}, \delta\right)$
    $x_{\min} \leftarrow \underset{x \in X_{\text{near}}}{\arg\min}\, \texttt{C}(x) + c(x, x_{\text{new}})$
    $V \leftarrow V \cup \{x_{\text{new}}\}$
    $E \leftarrow E \cup \{(x_{\min}, x_{\text{new}})\}$
  end if
end for
return $G = (V, E)$

Fonctions

- $\texttt{SampleFree}()$: échantillonne un point dans l'espace libre ;
- $\texttt{Nearest}(G, x)$: point le plus proche de x dans le graphe G ;
- $\texttt{Steer}(x_1, x_2)$: point à une distance donnée de x_1 vers x_2 ;
- $\texttt{CollFree}(x_1, x_2)$: pas d'obstacle entre x_1 et x_2 ;
- $\texttt{Near}(G, x, d)$: points de G à une distance de x inférieure à d ;
- $\texttt{C}(x)$: cout entre x_{init} et x en remontant le graphe ;
- $c(x_1, x_2)$: cout entre x_1 et x_2 ;

Algorithme

RRT*

$V \leftarrow \{x_{\text{init}}\}; E \leftarrow \emptyset$
for $i = 1, \ldots, n$ **do**
  $x_{\text{rand}} \leftarrow \texttt{SampleFree}()$
  $x_{\text{nearest}} \leftarrow \texttt{Nearest}\left(G = (V, E), x_{\text{rand}}\right)$
  $x_{\text{new}} \leftarrow \texttt{Steer}\left(x_{\text{nearest}}, x_{\text{rand}}\right)$
  if $\texttt{CollFree}\left(x_{\text{nearest}}, x_{\text{new}}\right)$ **then**
    $X_{\text{near}} \leftarrow \texttt{Near}\left(G = (V, E), x_{\text{rand}}, \delta\right)$
    $x_{\min} \leftarrow \underset{x \in X_{\text{near}}}{\arg\min}\, \texttt{C}(x) + c\left(x, x_{\text{new}}\right)$
    $V \leftarrow V \cup \{x_{\text{new}}\}$
    $E \leftarrow E \cup \{(x_{\min}, x_{\text{new}})\}$
    for all $x \in X_{\text{near}}$ **do**
      if $\texttt{C}(x_{\text{new}}) + c\left(x_{\text{new}}, x\right) < \texttt{C}(x)$ **then**
        $E \leftarrow E \setminus \{(\texttt{P}(x), x)\}$
        $E \leftarrow E \cup \{(x_{\text{new}}, x)\}$
      end if
    end for
  end if
end for
return $G = (V, E)$

Fonctions

- $\texttt{SampleFree}()$: échantillonne un point dans l'espace libre ;
- $\texttt{Nearest}(G, x)$: point le plus proche de x dans le graphe G ;
- $\texttt{Steer}(x_1, x_2)$: point à une distance donnée de x_1 vers x_2 ;
- $\texttt{CollFree}(x_1, x_2)$: pas d'obstacle entre x_1 et x_2 ;
- $\texttt{Near}(G, x, d)$: points de G à une distance de x inférieure à d ;
- $\texttt{C}(x)$: cout entre x_{init} et x en remontant le graphe ;
- $c(x_1, x_2)$: cout entre x_1 et x_2 ;
- $\texttt{P}(x)$: parent de x.

Planification par champs de potentiel

Potential fields

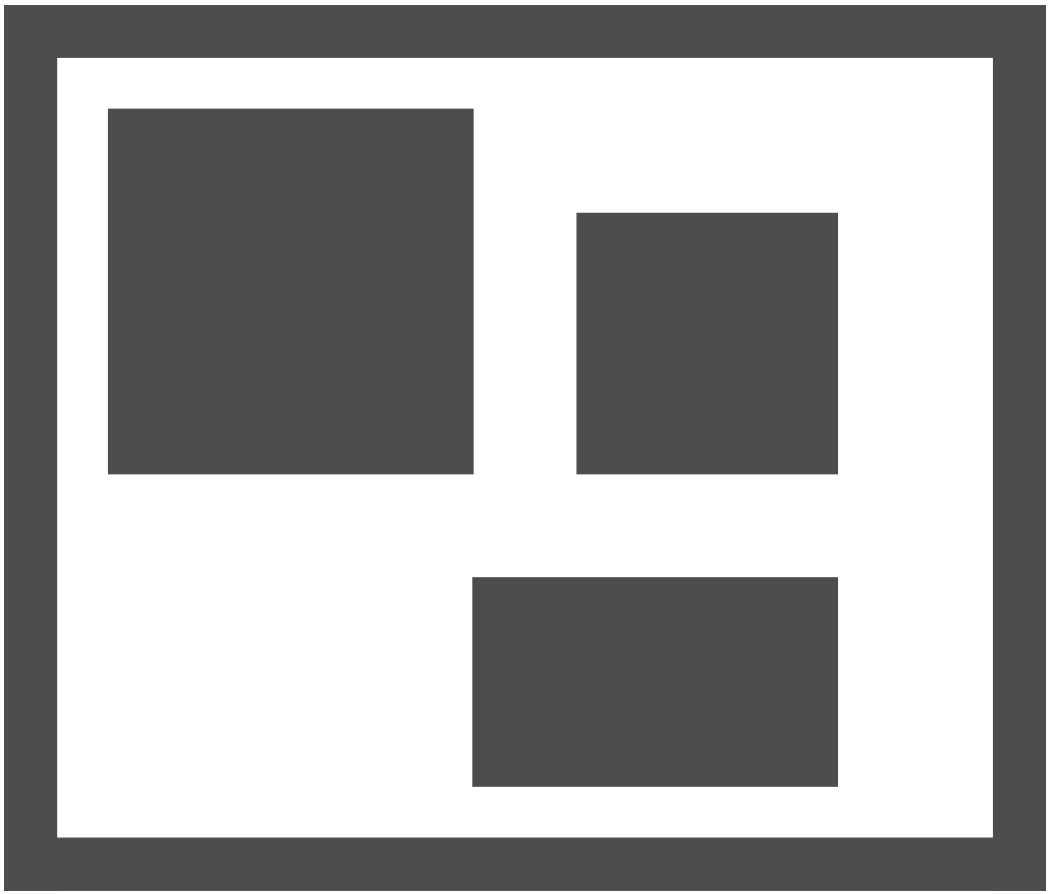
- génération d'un champ répulsif autour des obstacles ;
- génération d'un champ attractif depuis le but ;
- combinaison des deux ;
- remontée de gradient.

Résultat

- chemin ;
- éloigné des obstacles ;
- rapide à calculer ;
- minimums locaux.

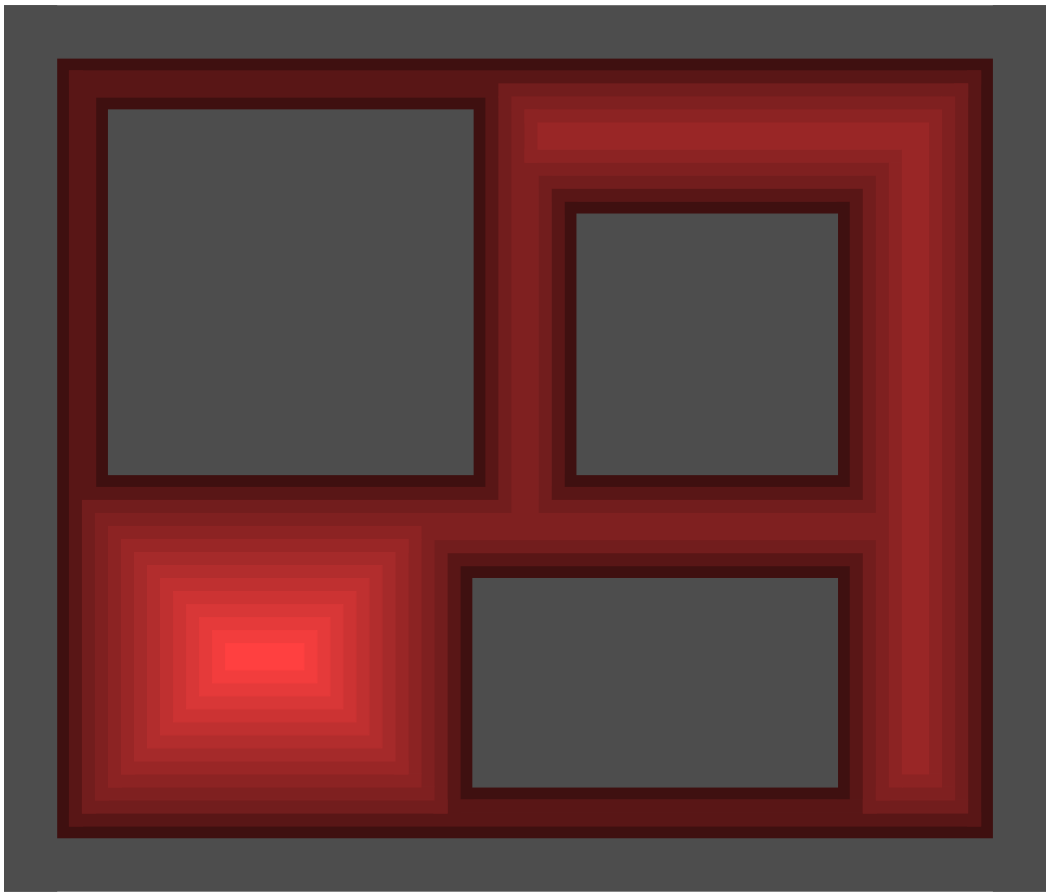
Exemple

Espace libre



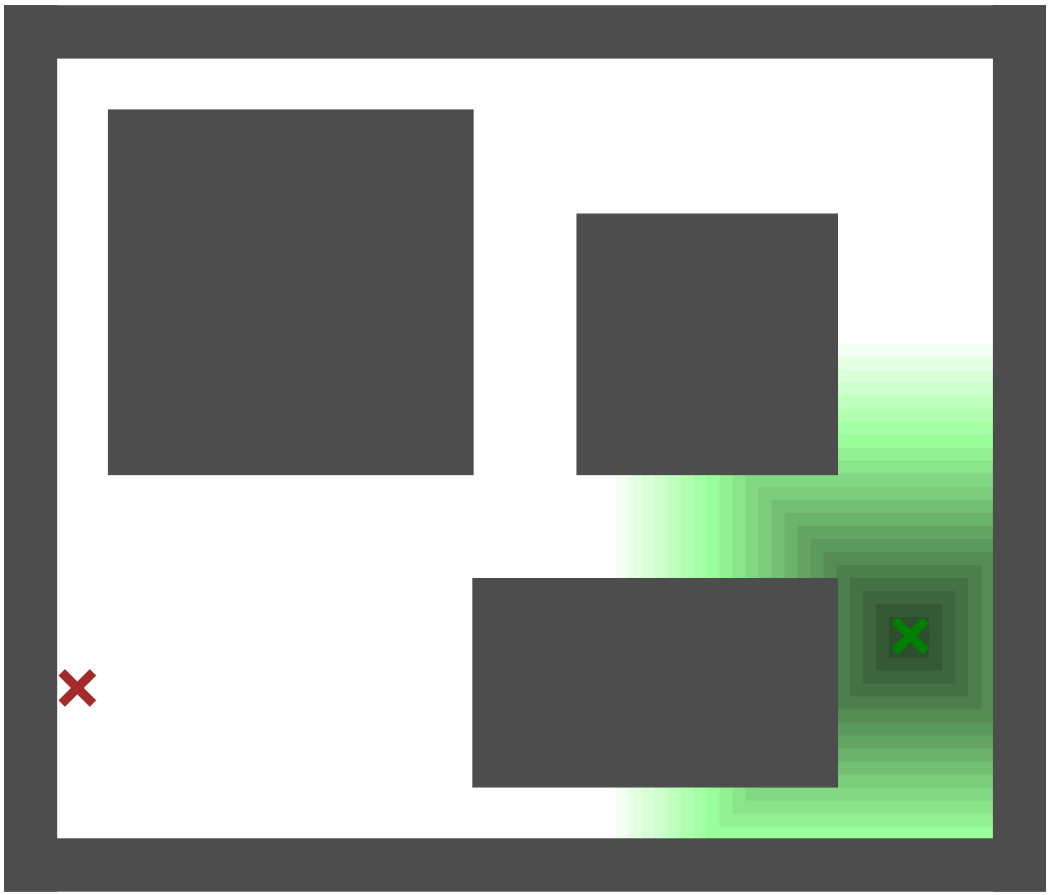
Exemple

Champ répulsif



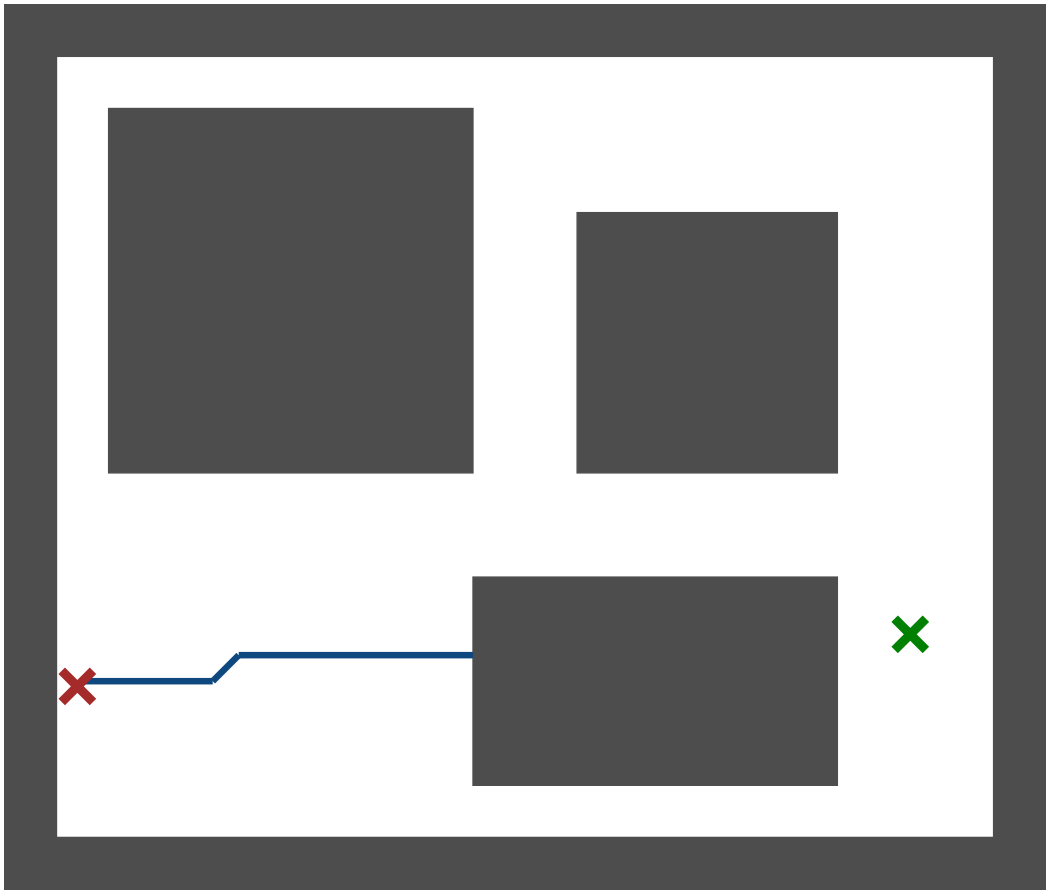
Exemple

Champ attractif



Exemple

Remontée de gradient



Minimum local !

Planification avec graphe de visibilité

Graphe de visibilité

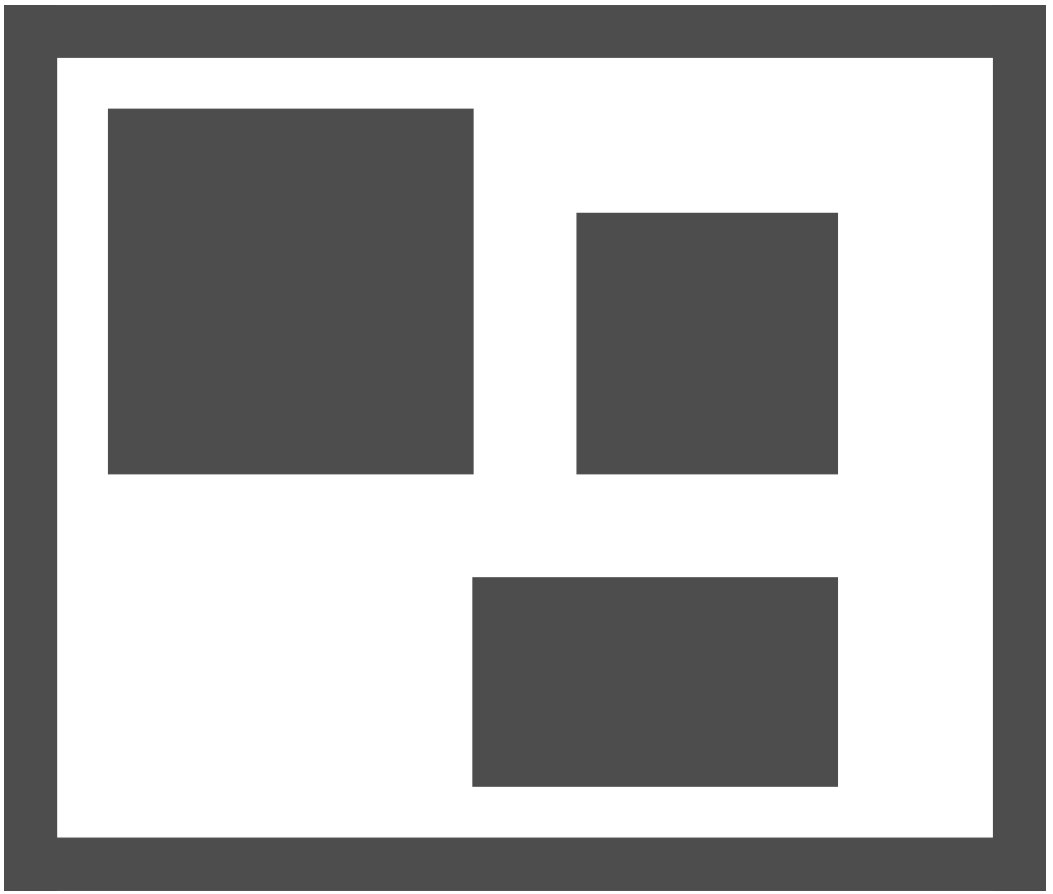
- nœuds : sommets des obstacles ;
- arêtes : ssi ligne de vue entre nœuds ;
- position courante et but comme nœuds ;
- recherche dans le graphe.

Résultat

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

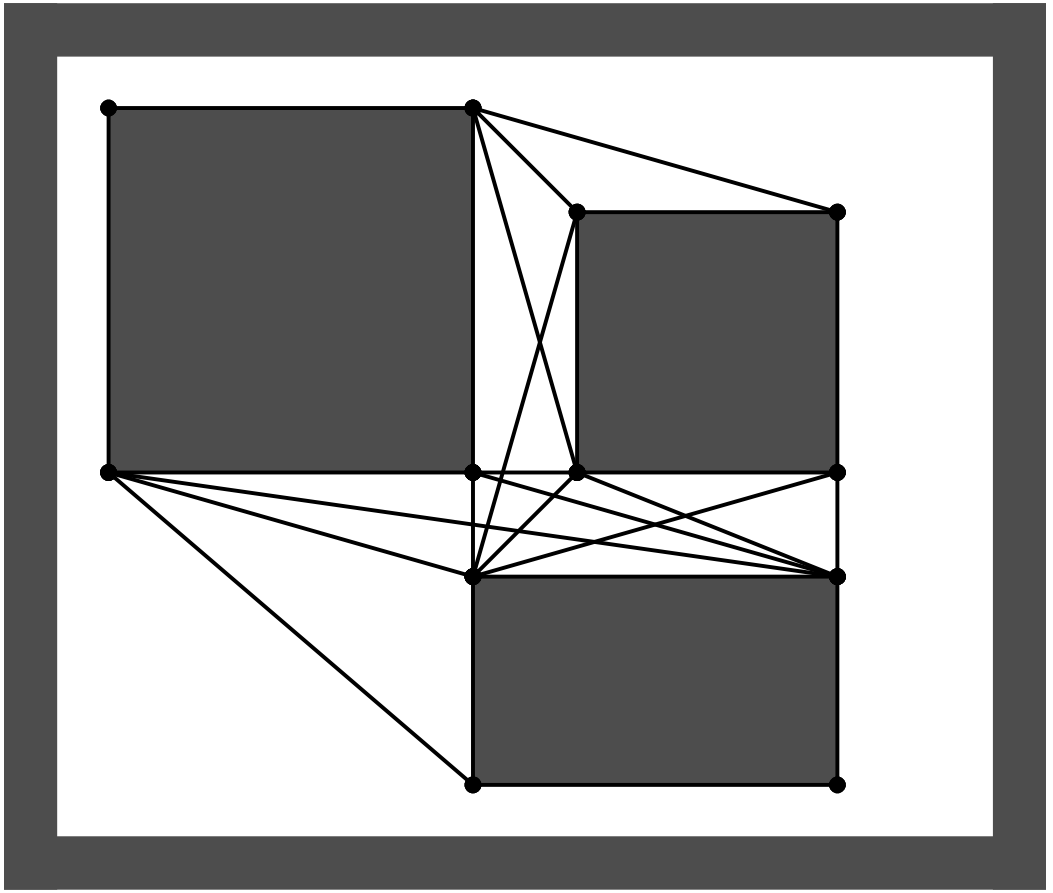
Exemple

Espace libre



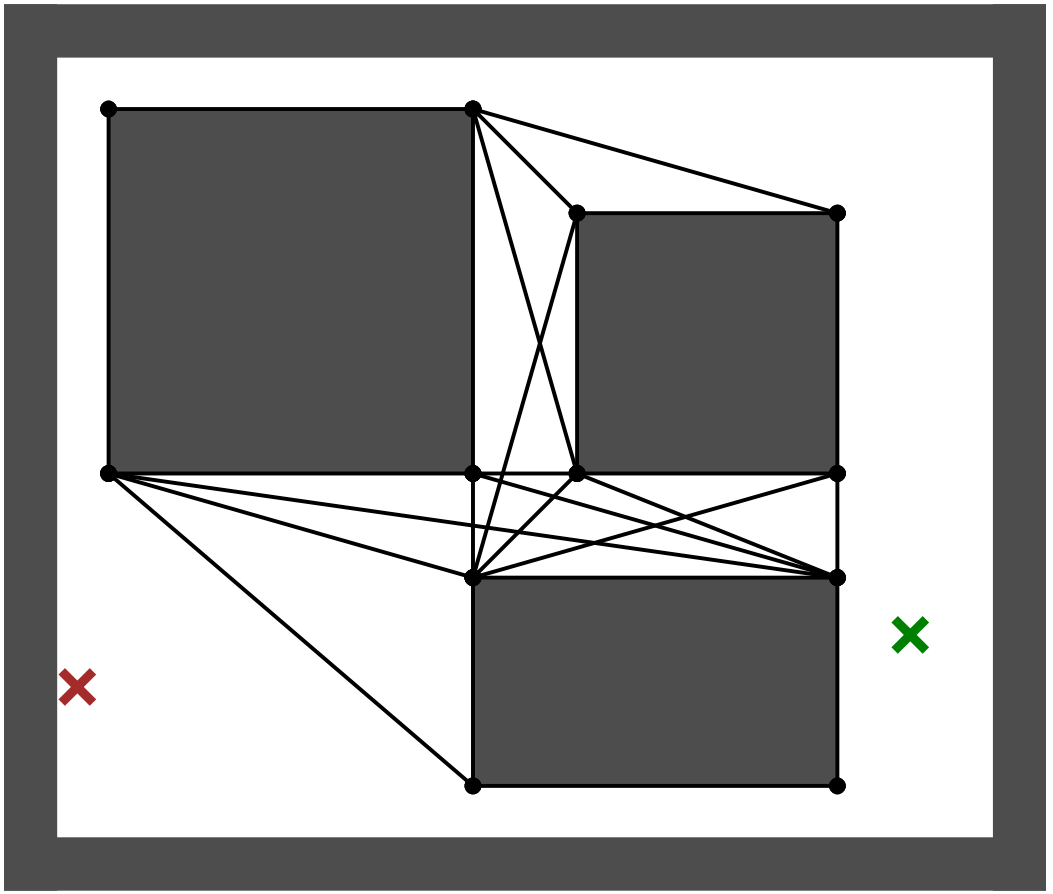
Exemple

Graphe de visibilité des obstacles



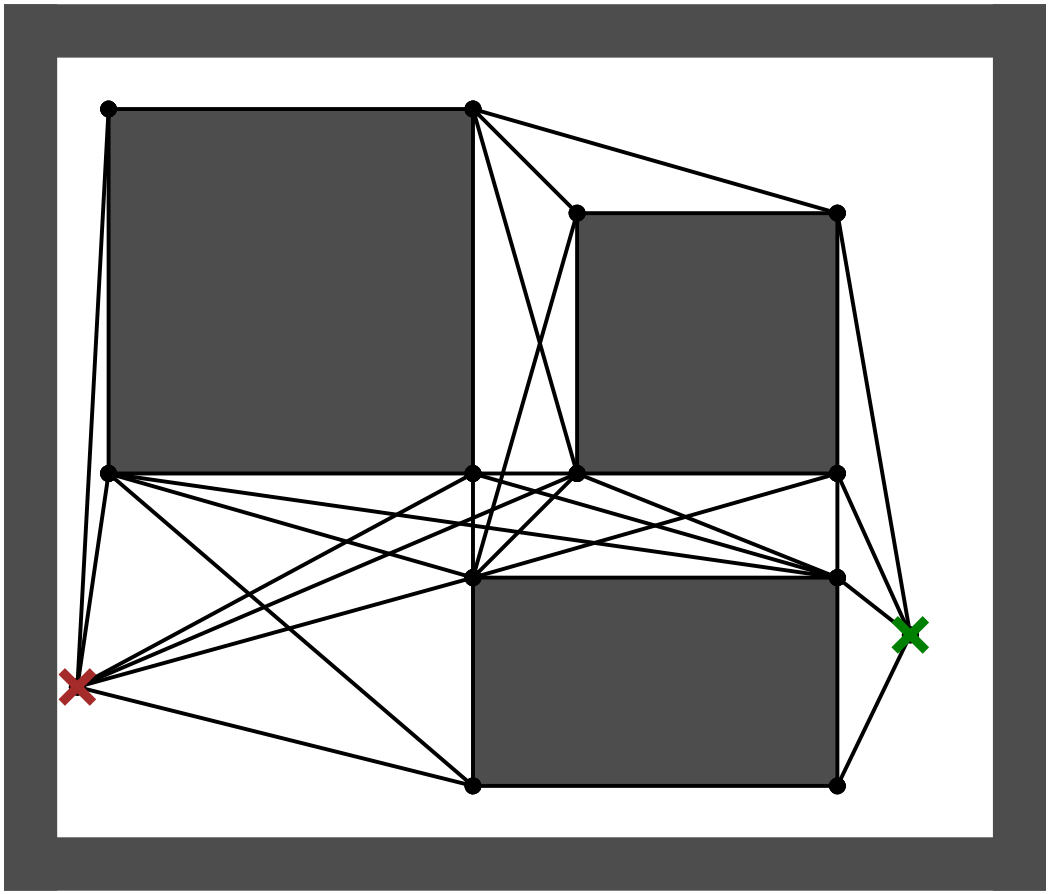
Exemple

Inclusion des extrémités



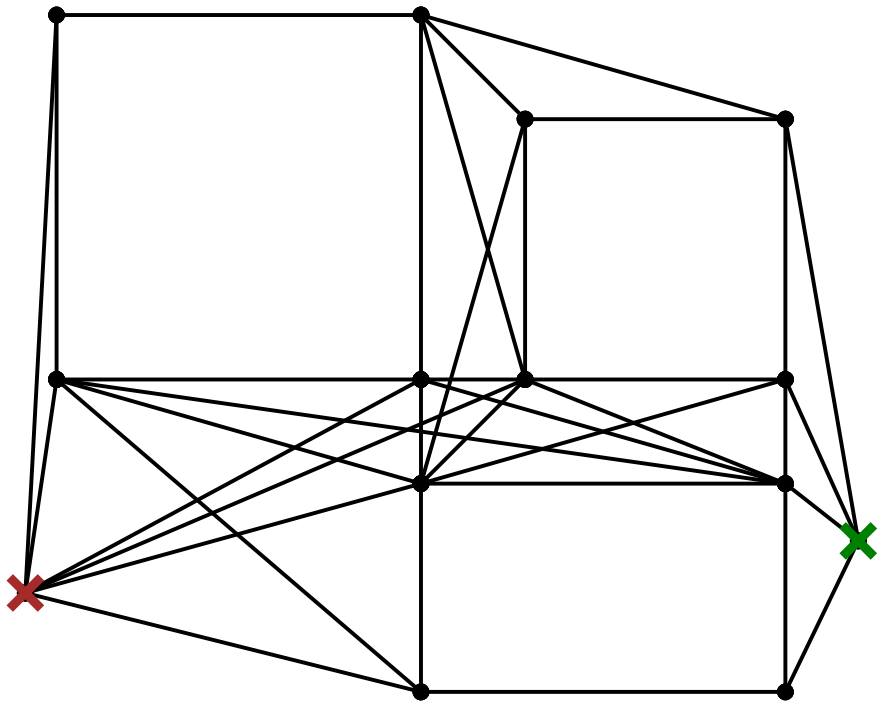
Exemple

Inclusion des extrémités



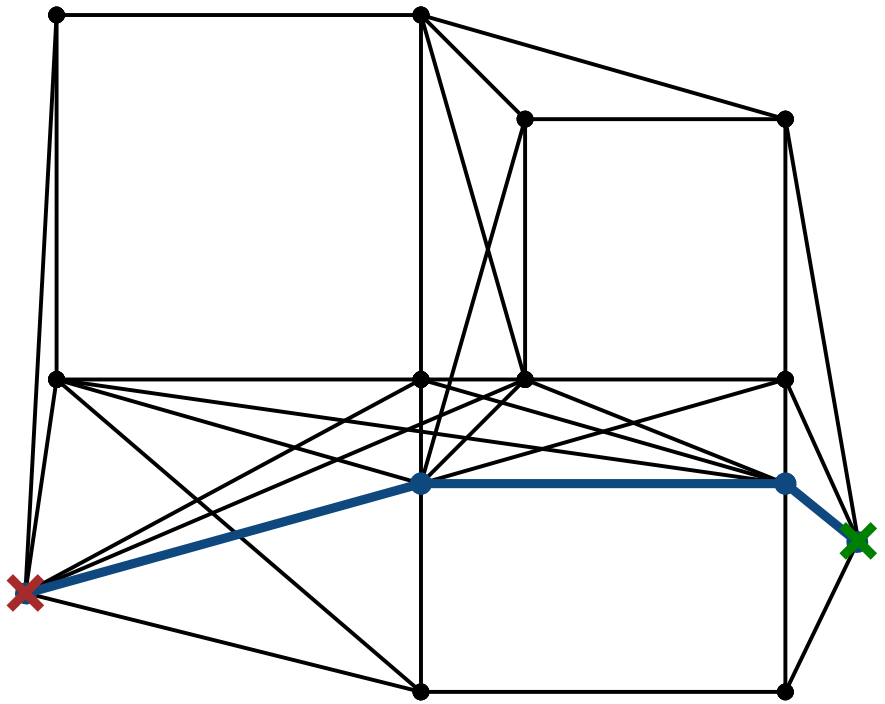
Exemple

Graphe complet



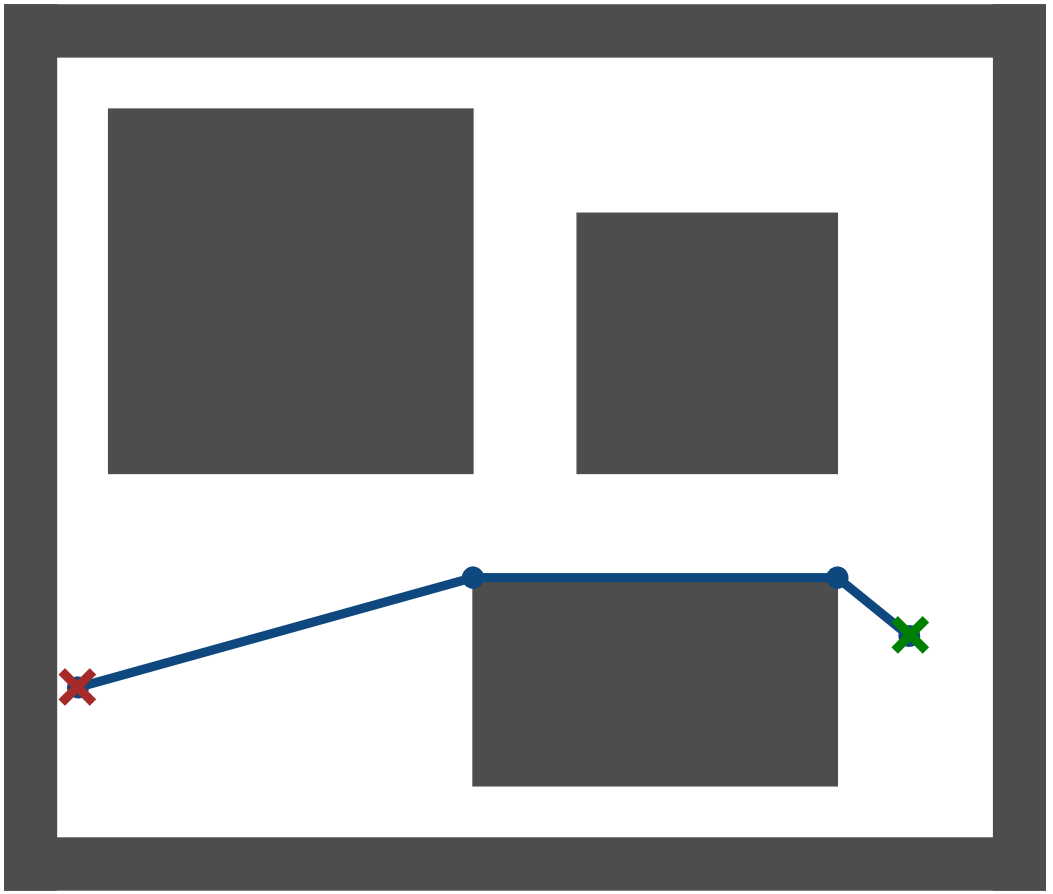
Exemple

Planification (A* ou autre)



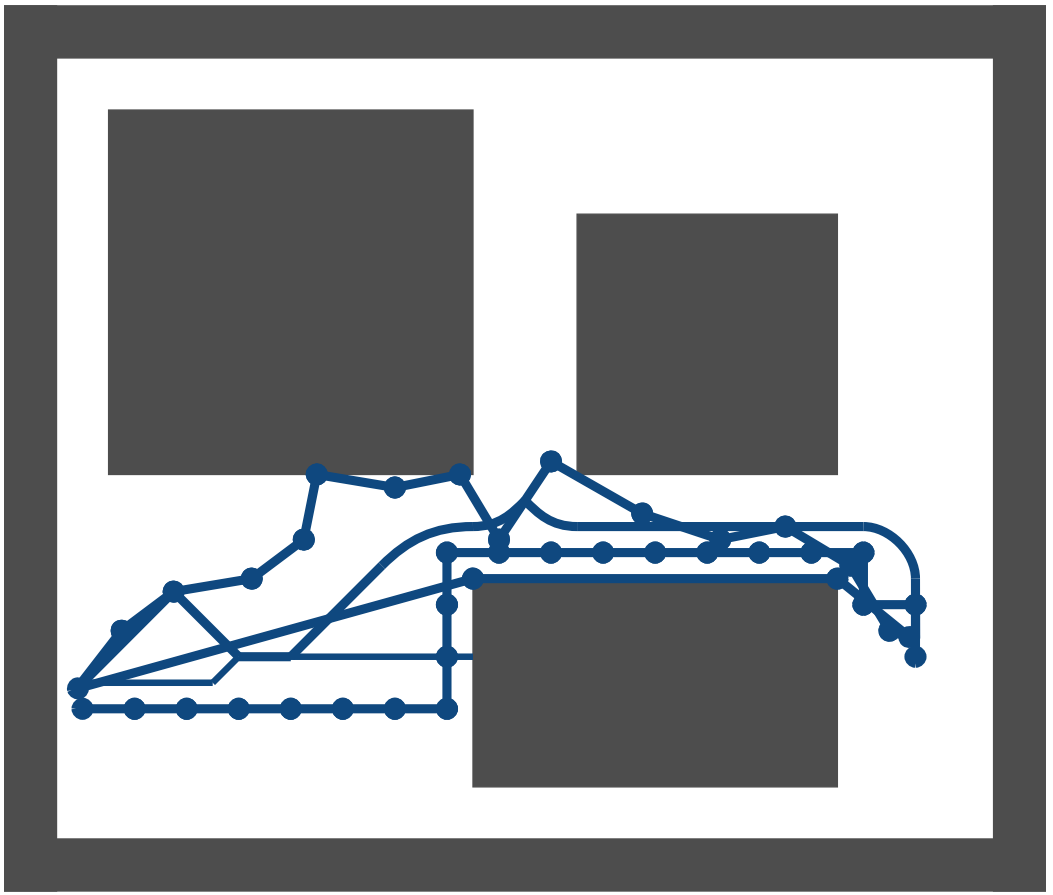
Exemple

Résultat



Comparaison

Comparaison des chemins obtenus



3

Conclusion

Conclusion

Algorithmes de planification

- plusieurs familles ;
- représentations internes différentes ;
- critères d'optimisation différents.

Limites

- carte connue ;
- obstacles statiques.

Bibliographie

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.

RRT*, PRM*, etc.

- Karaman and Frazzoli, *Sampling-based algorithms for optimal motion planning*, IJRR 2011.

Merci de votre attention.

Des questions ?