

Planification et contrôle

Cours ENSTA Paris - ROB316 / 04

David FILLIAT

david.filliat@ensta-paris.fr

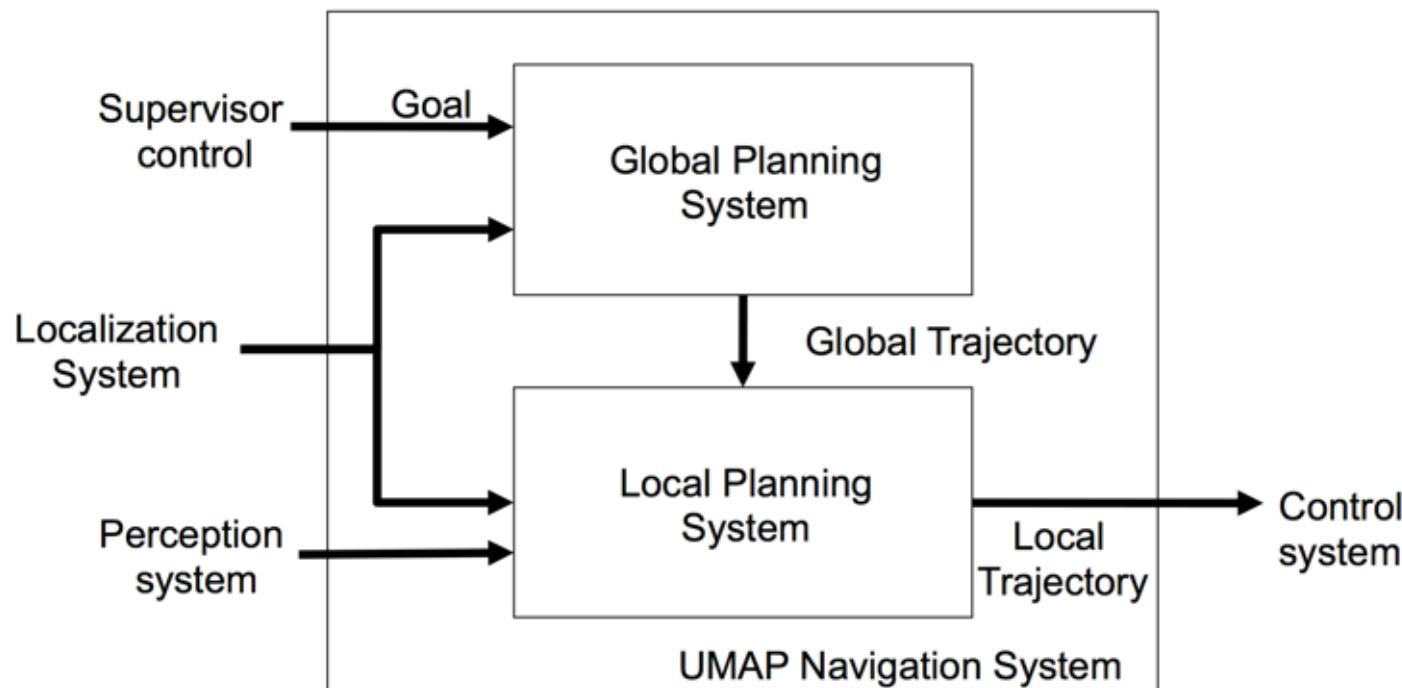
2019-2020



Planification de trajectoire

- Evitement d'obstacles
- Planification réactive
- Recherche de chemin stochastique
- Exploration

Deux niveaux de planification (cf architecture hybrides)



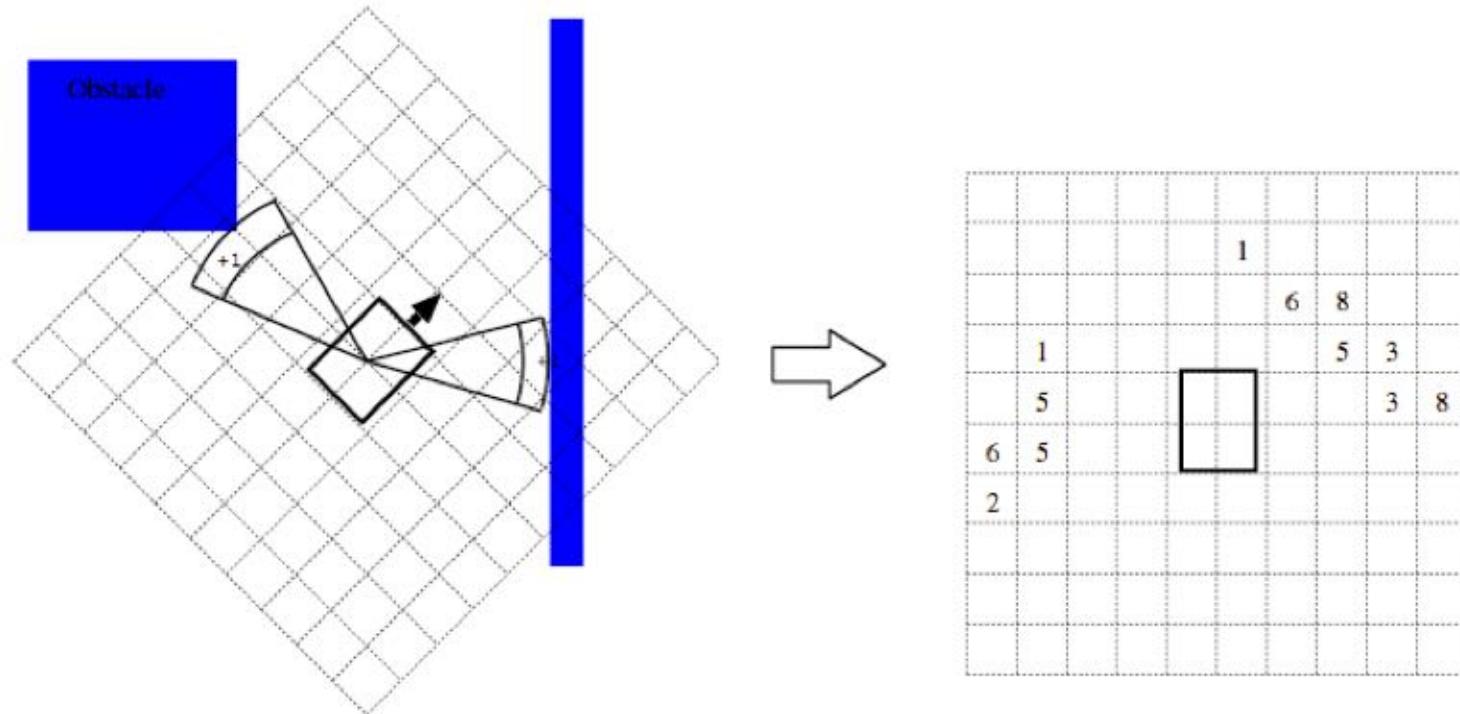
- Planification locale pour la prise en compte des obstacles dynamiques
- Très nombreux algorithmes : Vector Field Histogram, Elastic Bands, Potential fields, Dynamic Window...

Evitement d'obstacles

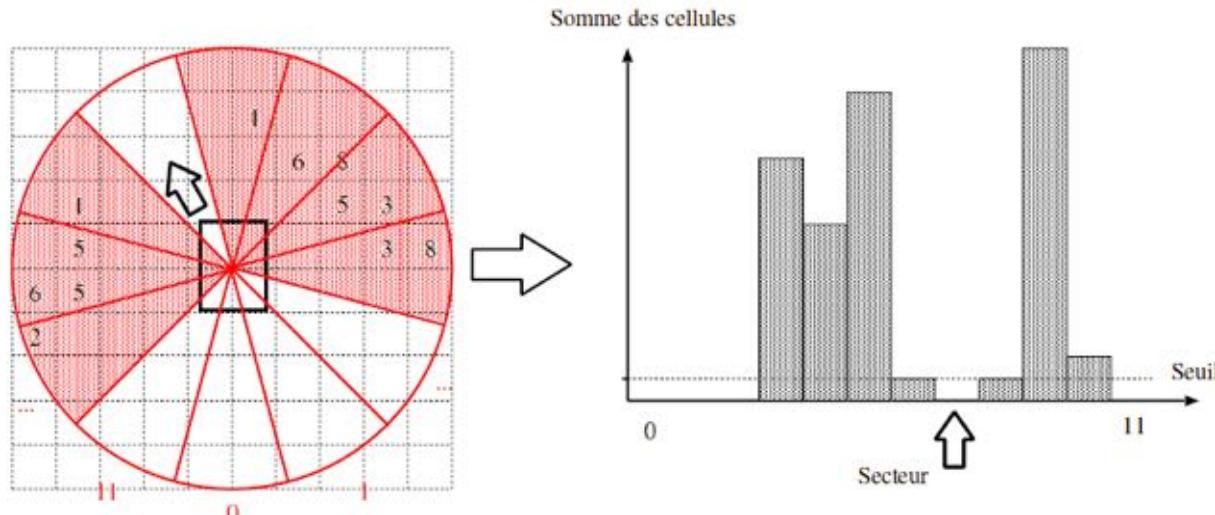


Utilise une représentation locale de l'environnement
(grille d'occupation histogrammique)

- Utilisable avec des capteurs peu précis (sonars)



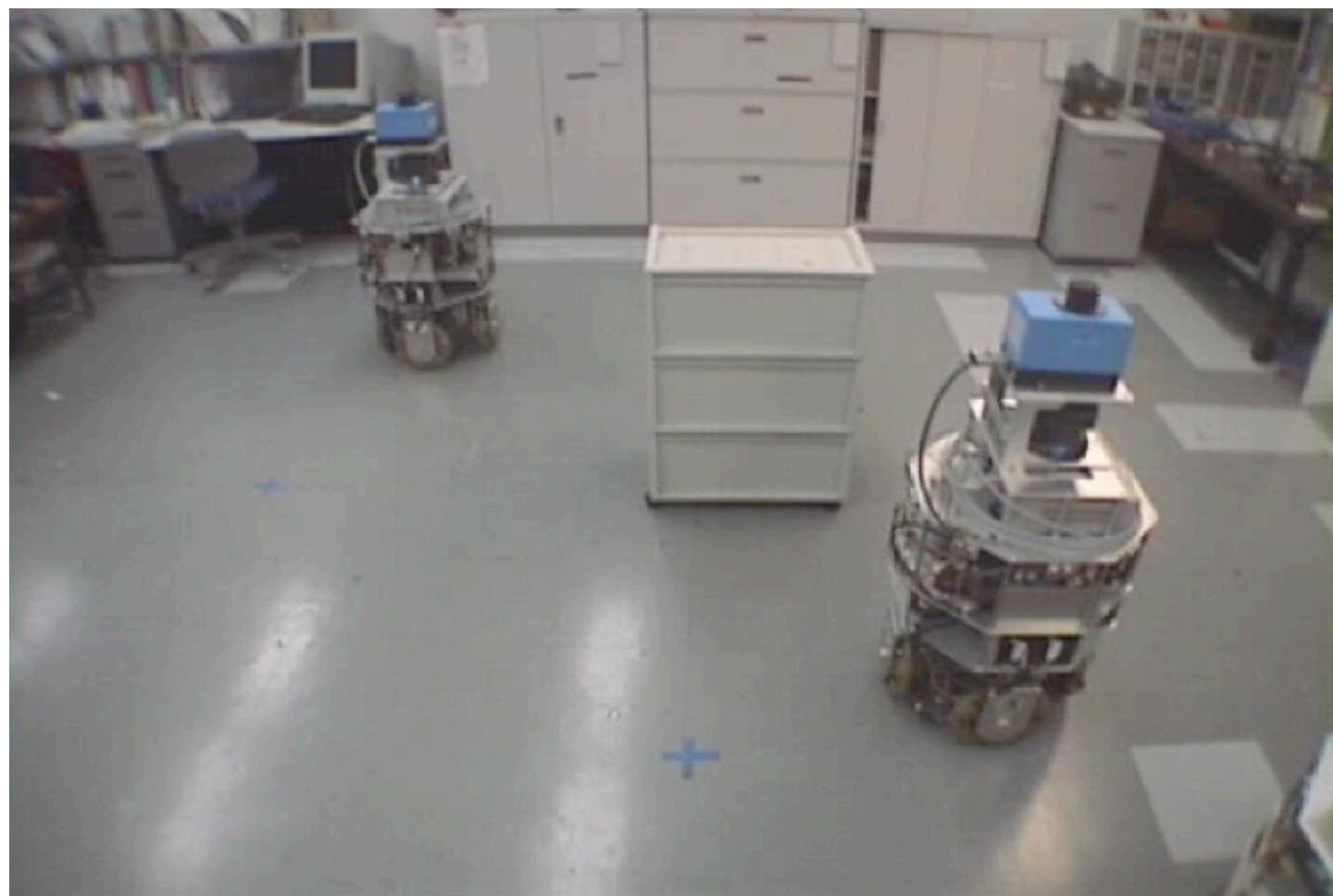
- Construction d'un histogramme d'occupation par secteur angulaire autour du robot
- Choix de la direction parmi les directions libres



Méthode très rapide

Permet des déplacements à vitesse élevée

Améliorations :
- régler la vitesse en fonction de l'occupation de l'espace
- utiliser la dynamique du robot

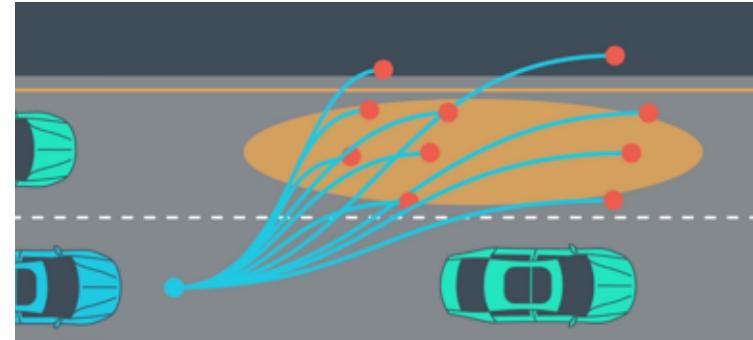


Principe

- Paramétriser des trajectoires
- Estimer des contraintes
- Recherche des paramètres

qui respectent différentes contraintes :

- Évitement d'obstacles (contrainte dure)
- Direction préférentielle (souple)
- Distance min aux obstacles (souple)
- ...



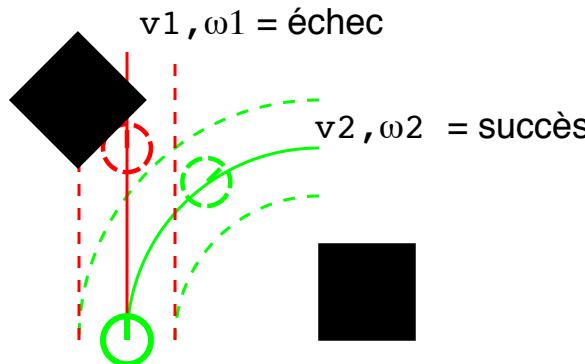
Application robot unicycle

- Tracé des contraintes sur un graphe des vitesses
- Recherche de vitesses (v, ω) qui respectent les contraintes

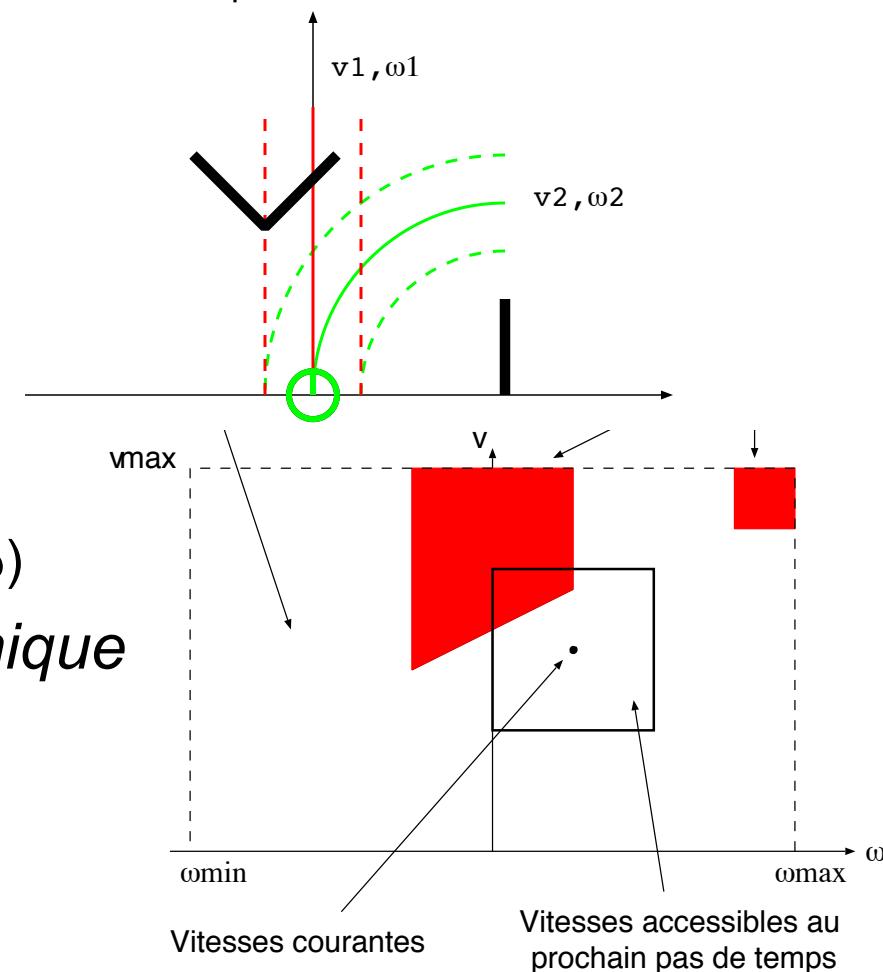
Contrainte d'évitement d'obstacles :

(estimé à un pas de temps donné dans le futur) :

Environnement réel



Perceptions du robot



Report dans le graphe (v, ω)

Tracé de la *fenêtre dynamique*

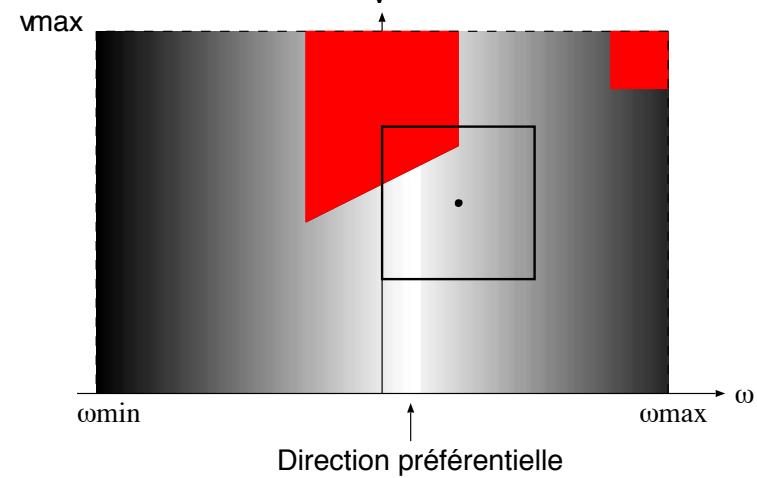
(accel max)

Choix d'un couple de vitesses :

- sans obstacle au sein de la fenêtre

Utilisation de contraintes supplémentaires (souples):

- ex : direction préférentielle (but, suivi de trajectoire globale...)

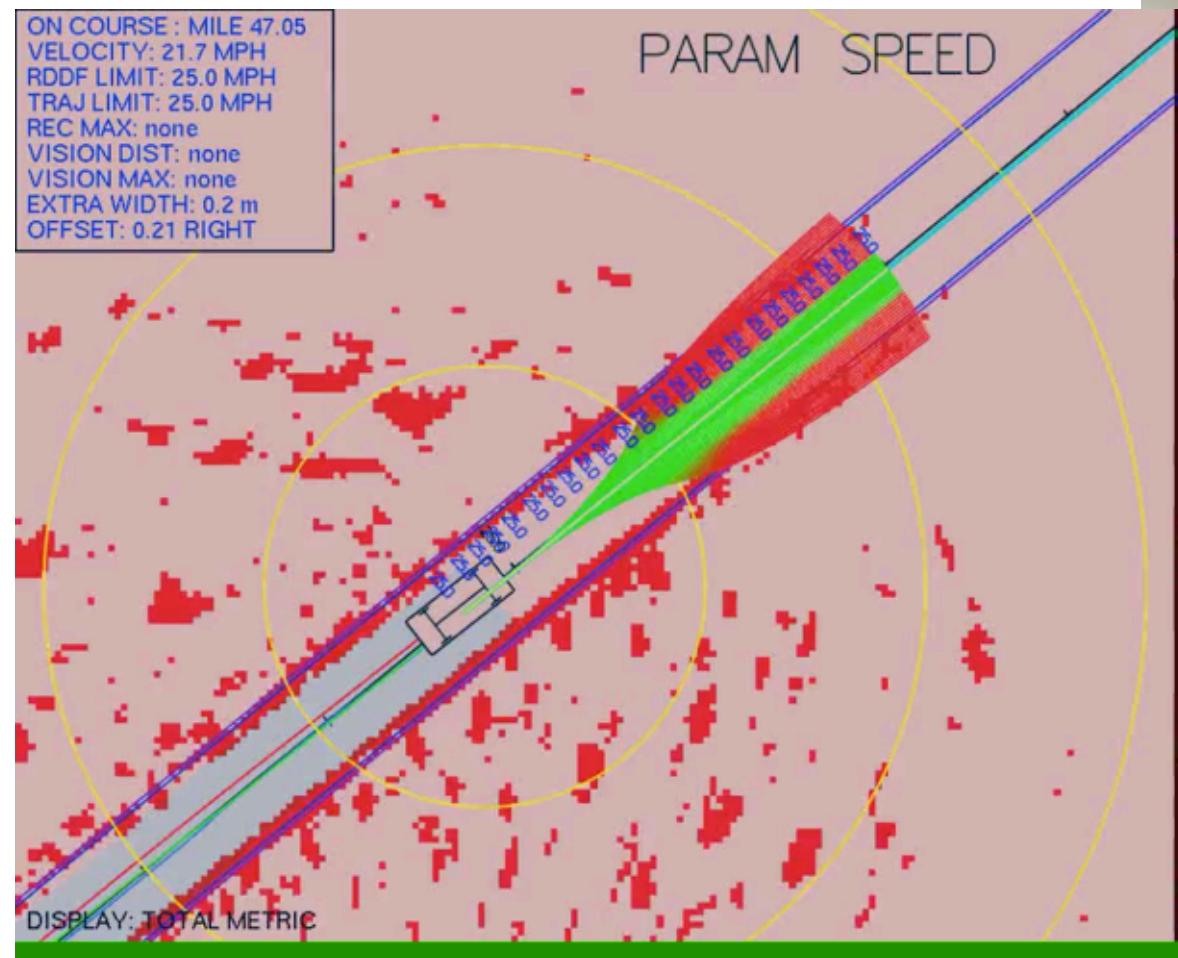


Variantes :

- Les valeurs du graphe sont estimées pour des valeurs discrétisées des vitesses ou à partir des obstacles
- Pour un robot relativement lent avec de bonnes accélérations, on peut n'utiliser qu'un point (sans fenêtre)

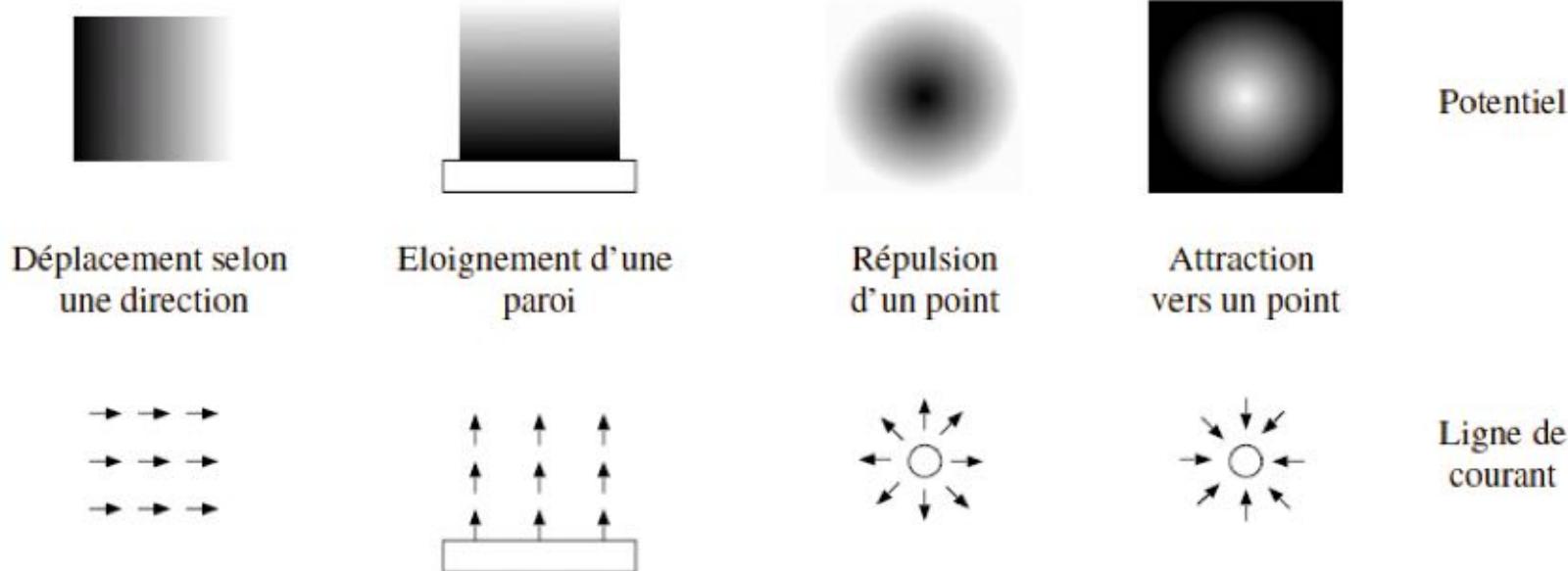
Exemple sur un véhicule autonome

- STANLEY, Stanford (2006)



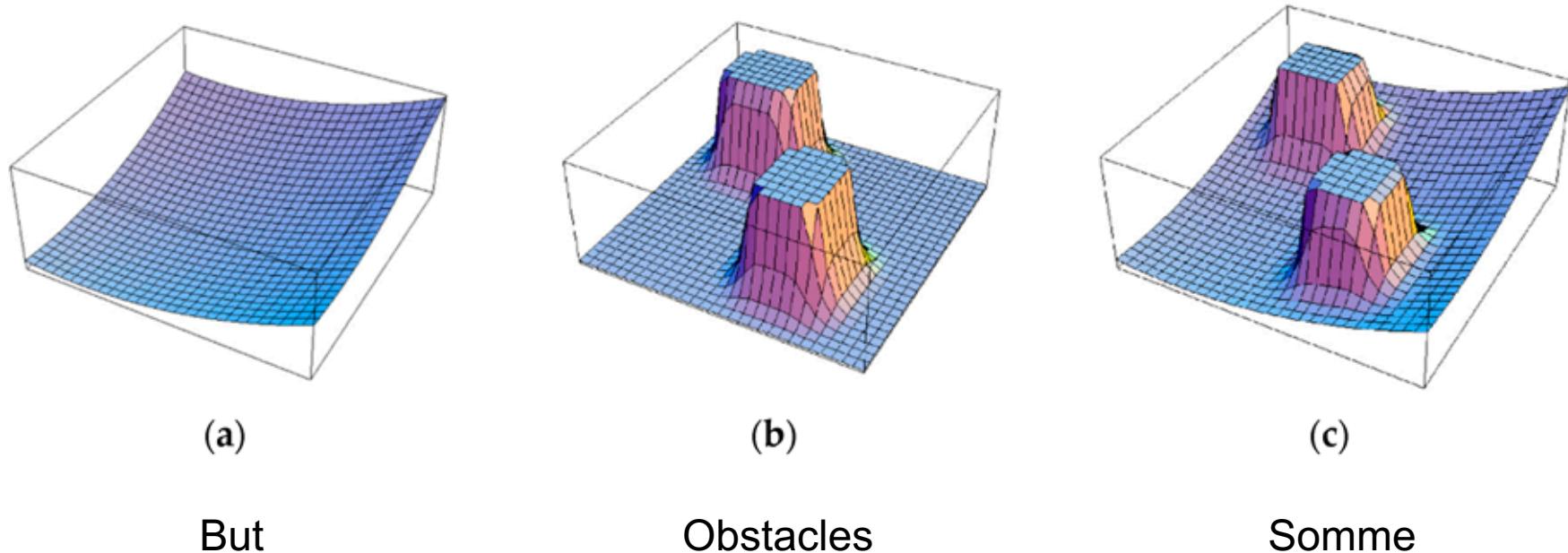
Méthode de combinaison de comportements

- Chaque objet de l'environnement est associé à un champ de forces
- Les forces peuvent varier avec la distance, peuvent agir sur une étendue limitée
- Le robot suit le champ de forces



Exemple de combinaison

- Tiré de « Water Sink Model for Robot Motion Planning » by Gi-Yoon Jeon and Jin-Woo Jung

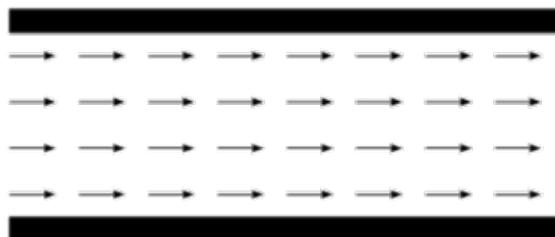


But

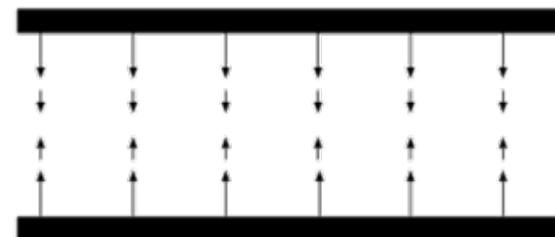
Obstacles

Somme

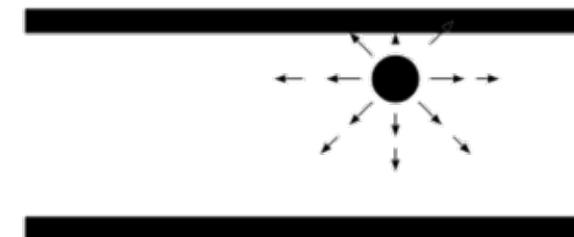
La combinaison donne la direction à prendre en chaque point



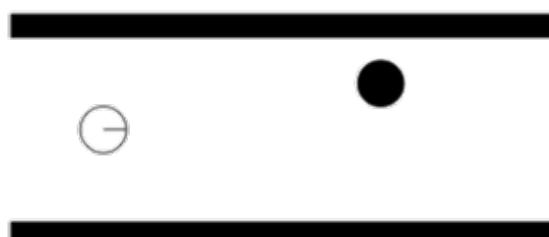
Suivi d'un couloir



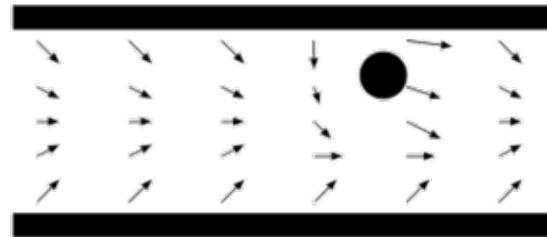
Centrage dans un couloir



Evitement d'un obstacle



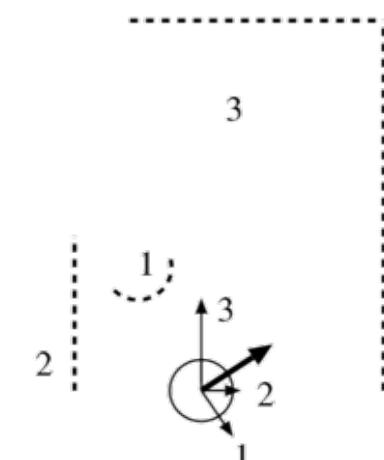
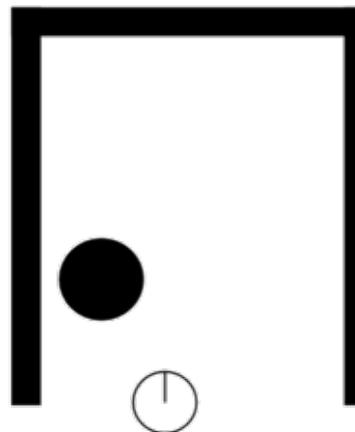
Environnement



Potentiel résultant

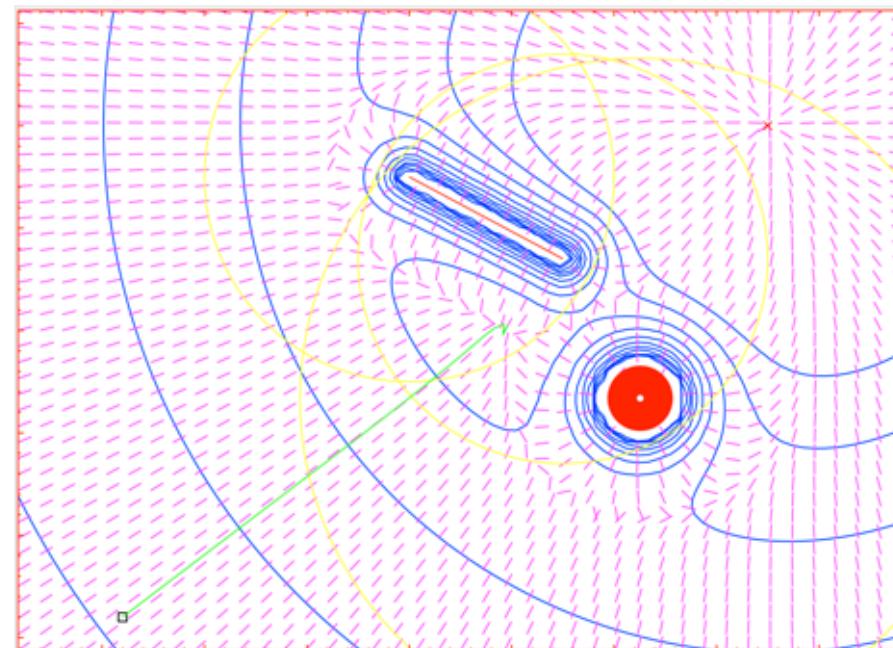
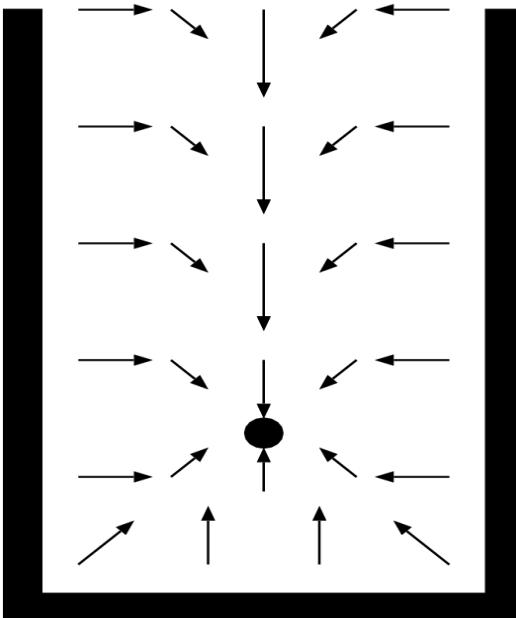
Calculs possibles dans l'espace relatif au robot

- pas de calcul global du champ de force
- permet l'évitement d'obstacles



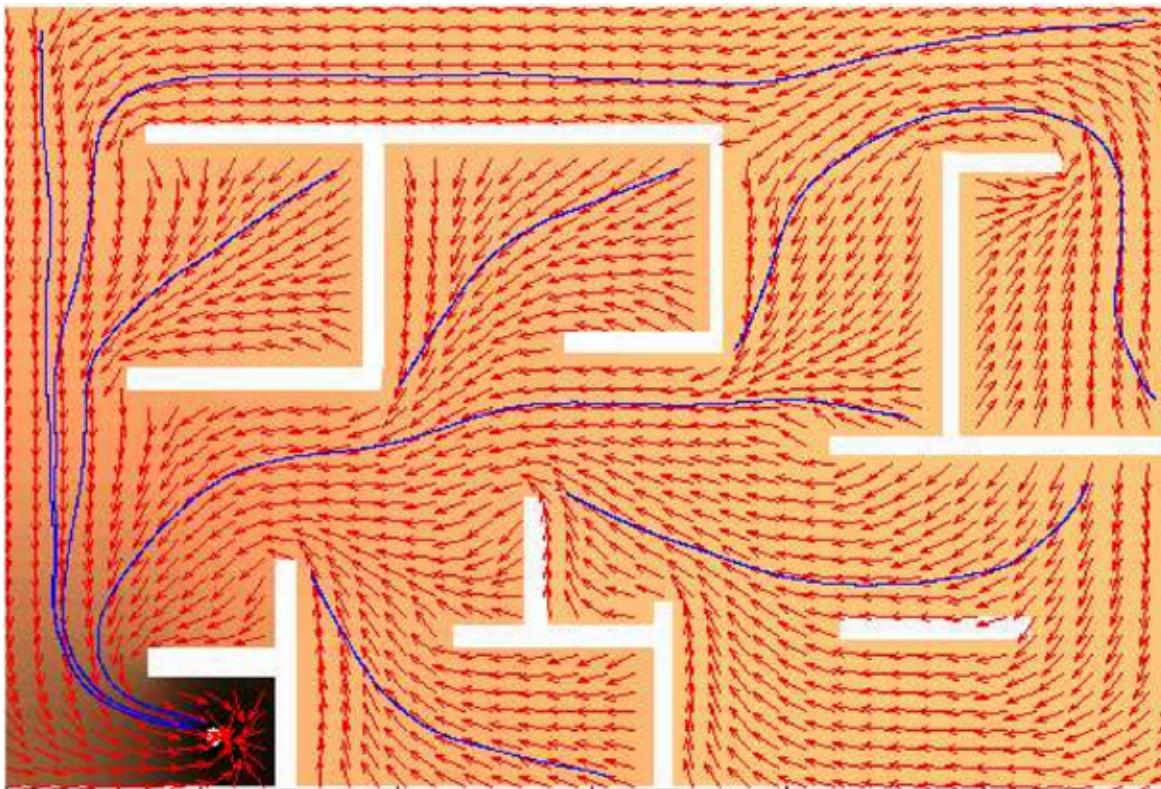
Problèmes de minimum locaux

- Peut se produire même sans 'cul de sac'
- Solutions :
 - Ajout de valeurs aléatoires (limité aux minimums 'peu profonds')
 - Comportement spécifique (suivi de mur...)
 - Mémorisation et évitement des zones déjà traversées (associé à localisation)



Problèmes de minimum locaux

- Champ de potentiel harmonique
 - Solution telle que $\Delta f = 0$
 - Garanti sans minimum local
 - Calcul global, plus coûteux



Planification réactive



Approche directe

- N'utilise qu'un niveau de planification

Hypothèses

- Carte inconnue
- Direction du but connue
- Perception locale (tactile)

Inspiration biologique

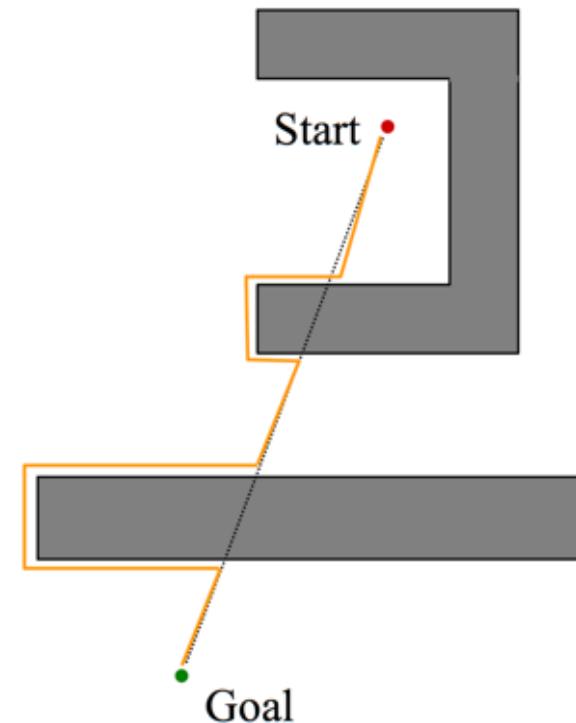
- Algorithme « Bug »
- Variantes : Bug 0, Bug 1, Bug 2, Tangent Bug

Limitations

- Ne garantit pas un chemin optimal
- N'exploite pas la connaissance de la carte
- Risque de comportements aberrants dans certains environnements

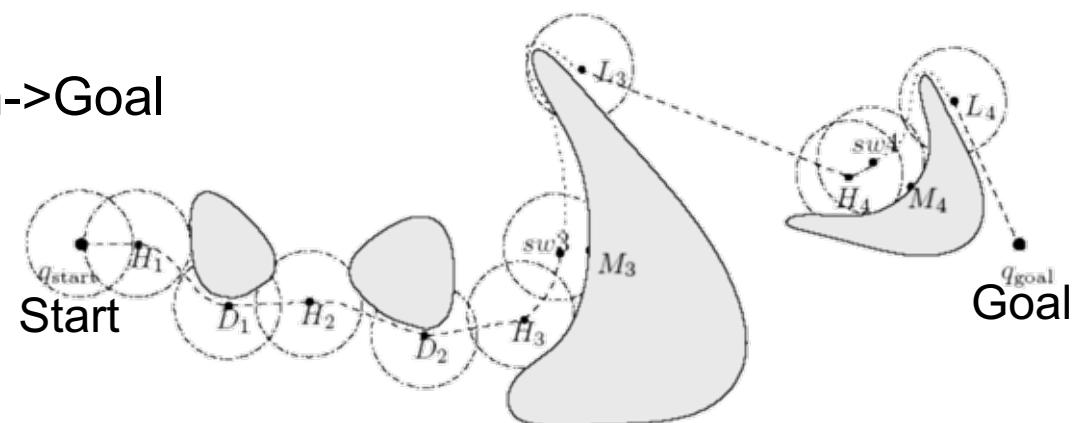
Algorithme Bug 2

- Suivre la ligne Start->Goal
- En cas d'obstacles, suivre l'obstacle jusqu'à retomber sur la ligne
- Reprendre la ligne



Algorithme Tangent Bug

- Suivre la ligne Start->Goal
- En cas d'obstacles, contourner Jusqu'à « voir » le but
- Suivre la ligne Position->Goal

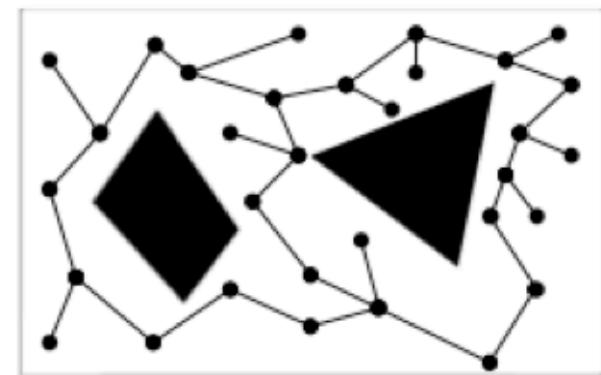


Recherche de chemin stochastique



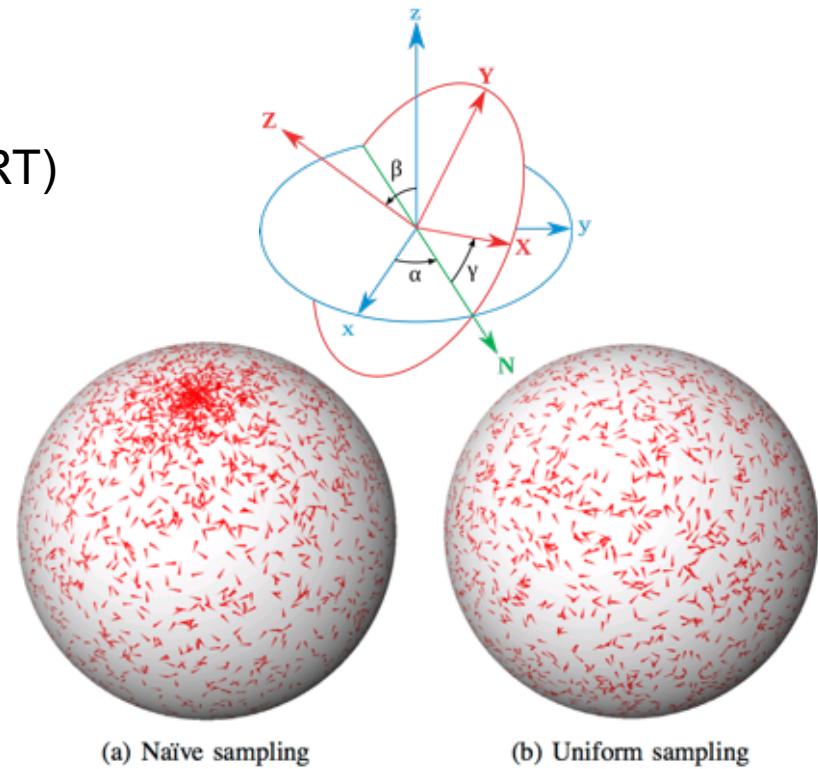
Echantillonnage aléatoire

- Permet de représenter les grands espaces de manière minimale
- Nécessite simplement une procédure de détection de collision
- Différentes méthodes dont
 - Probabilistic Road Maps (PRM)
 - Rapidly exploring Random Trees (RRT)



Echantillonnage des angles

- Echantillonnage uniforme en 3D
- Attention méthode naïve avec Euler
 - > quaternions



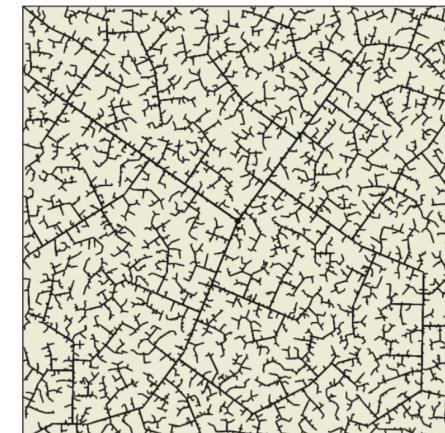
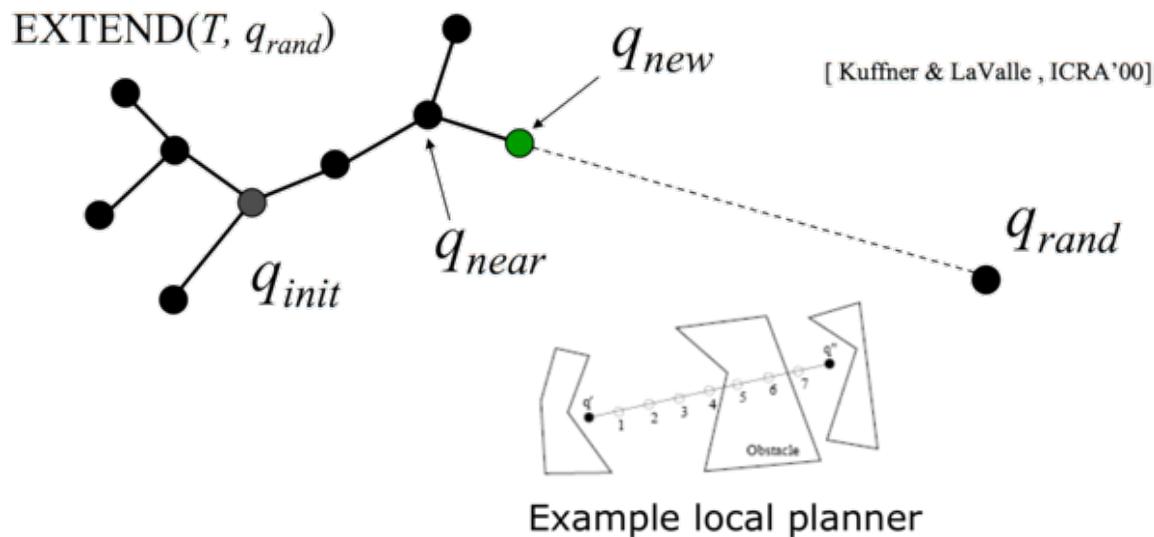
Rapidly exploring Random Trees

- Famille des méthodes de discréétisation en chemins aléatoires
- Mais ne crée pas un graphe complet
- Cherche seulement un chemin pour le but courant
- Algorithm **'Single Query'**
Vs **'Multiple Query'** pour PRM
- Construction incrémentale plutôt que globale



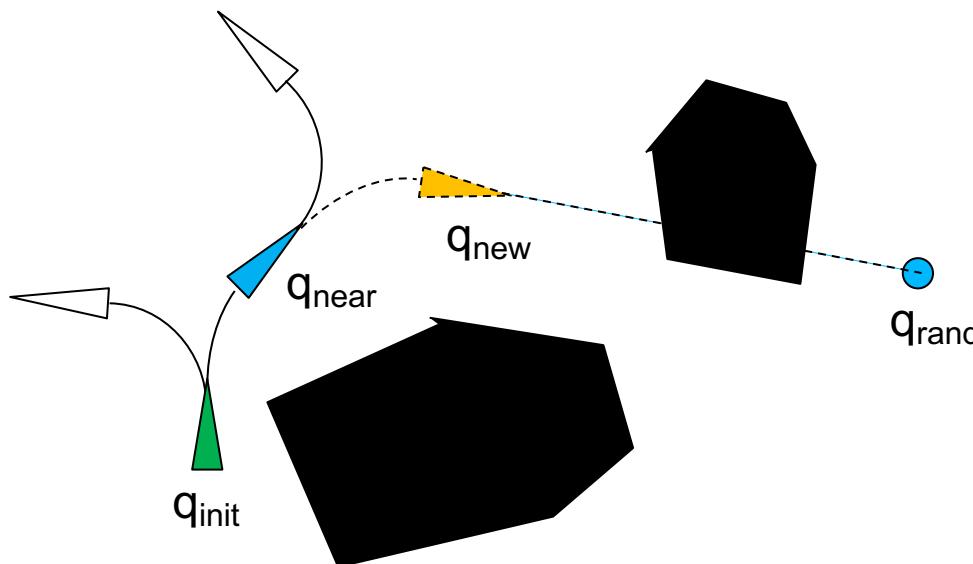
Rapidly exploring Random Trees (RRT, 99)

- Construction d'un arbre à partir de la position initiale
- Choisir un point q_{rand} aléatoirement
- Trouver le plus proche voisin dans l'arbre courant
- Etendre l'arbre d'une distance fixe depuis ce point
 - Utilise un planificateur local
 - Arrêt en cas d'obstacles
- De temps en temps, utiliser $q_{rand} = \text{but}$



Rapidly exploring Random Trees (RRT, 99)

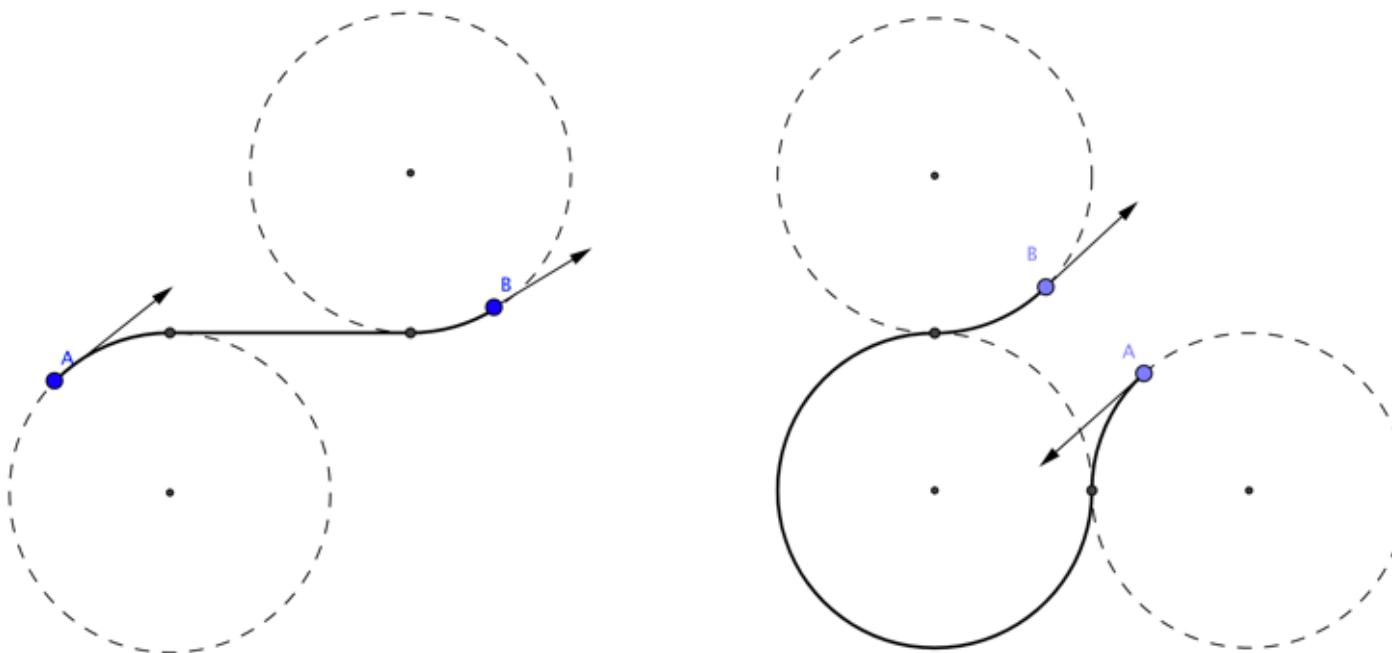
- Possibilité de planifier avec des modèles non-holonomes quelconques
- Utilisation du modèle dans le planificateur local



- Attention au calcul du ‘point le plus proche’ (fusion distance/angle)

Rapidly exploring Random Trees (RRT, 99)

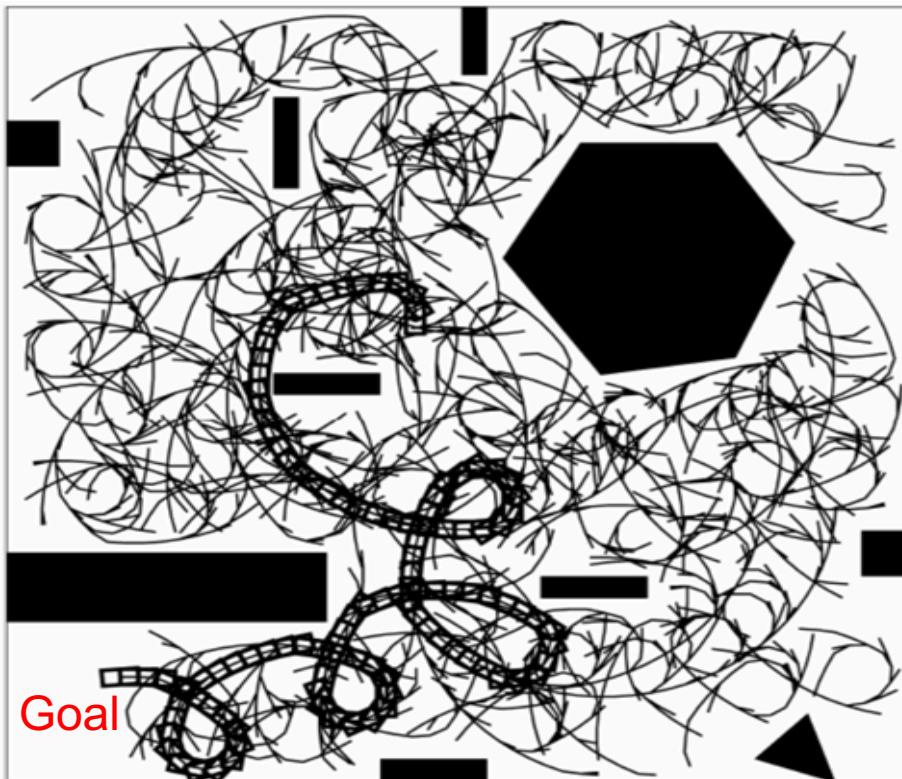
- Pour un modèle bicycle possibilité d'utiliser les courbes de Dubbins
- Donnent le plus court chemin entre 2 points orientés
- Calcul géométrique ou analytique rapide



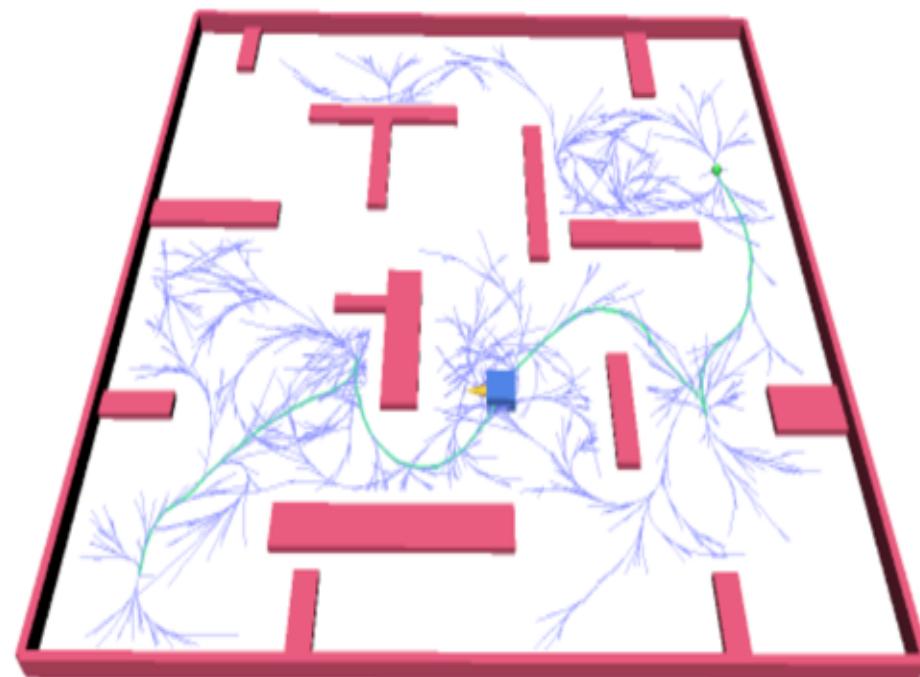
- Généralisation avec marche arrière : courbe de Reeds-Shepp

Rapidly exploring Random Trees (RRT, 99)

- Exemples non holonomes :



Voiture tournant à gauche

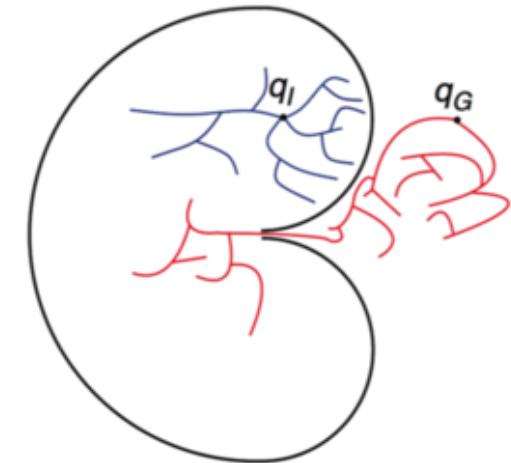
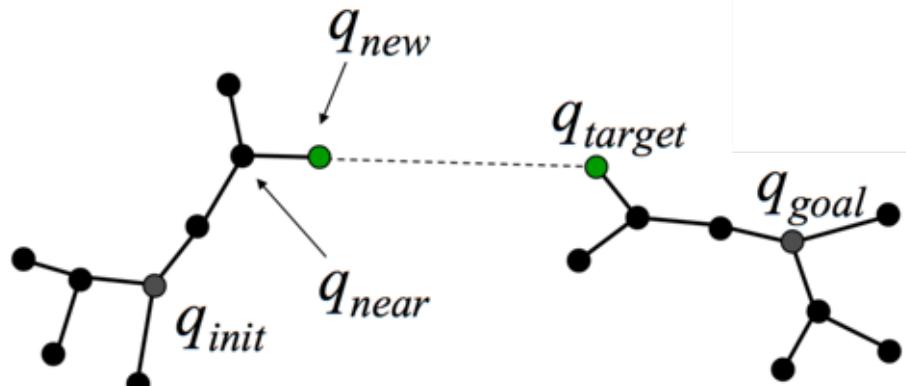


Hovercraft

- Mais plans proposés loin de l'optimal

Rapidly exploring Random Trees (RRT, 99)

- Variant Dual RRT (RRT connect) : construire 2 arbres depuis le départ et le but
- Tenter de connecter q_{new} à l'autre arbre à chaque extension



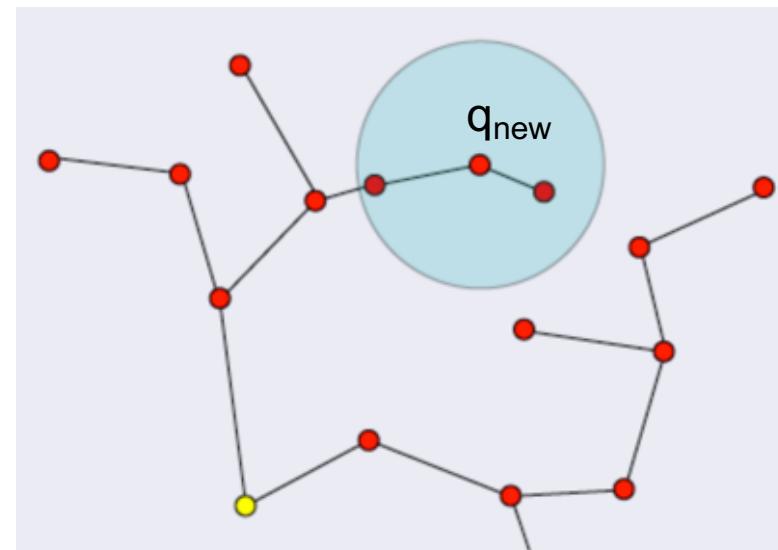
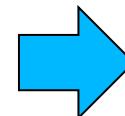
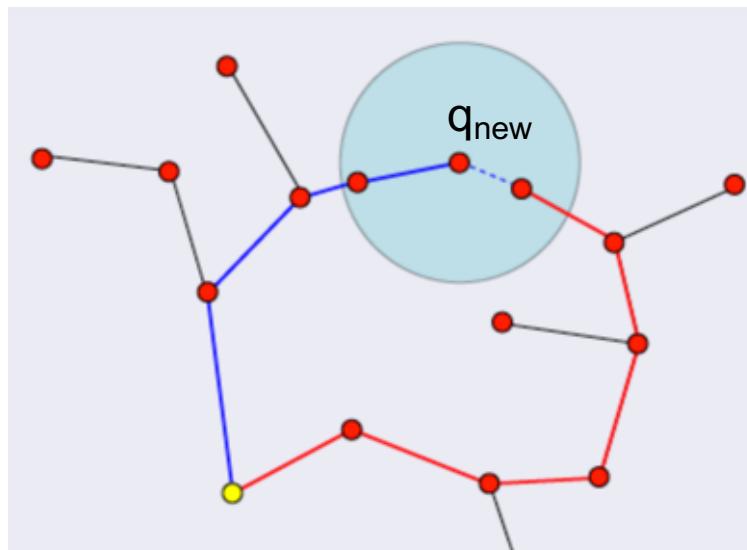
- Permet de planifier dans des environnements plus encombrés
- Nombreuses autres extensions
 - Real time
 - Anytime
 - Dynamic environments

RRT*

- RRT trouve un chemin si il existe, mais pas l'optimal
- RRT* : amélioration de la mise à jour en restructurant l'arbre
- Permet de trouver les chemins optimaux

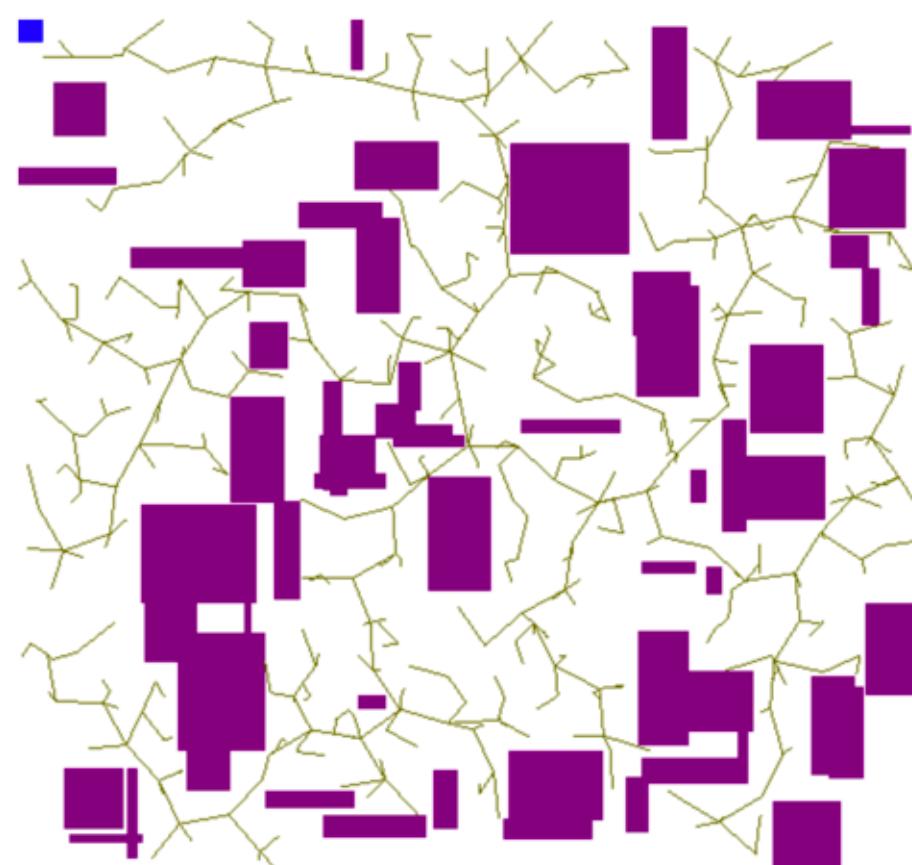
Mise à jour RRT*

- Sélectionner tous les nœuds autour de q_{new}
- Reconnecter les nœuds aux voisins permettant des chemins plus courts



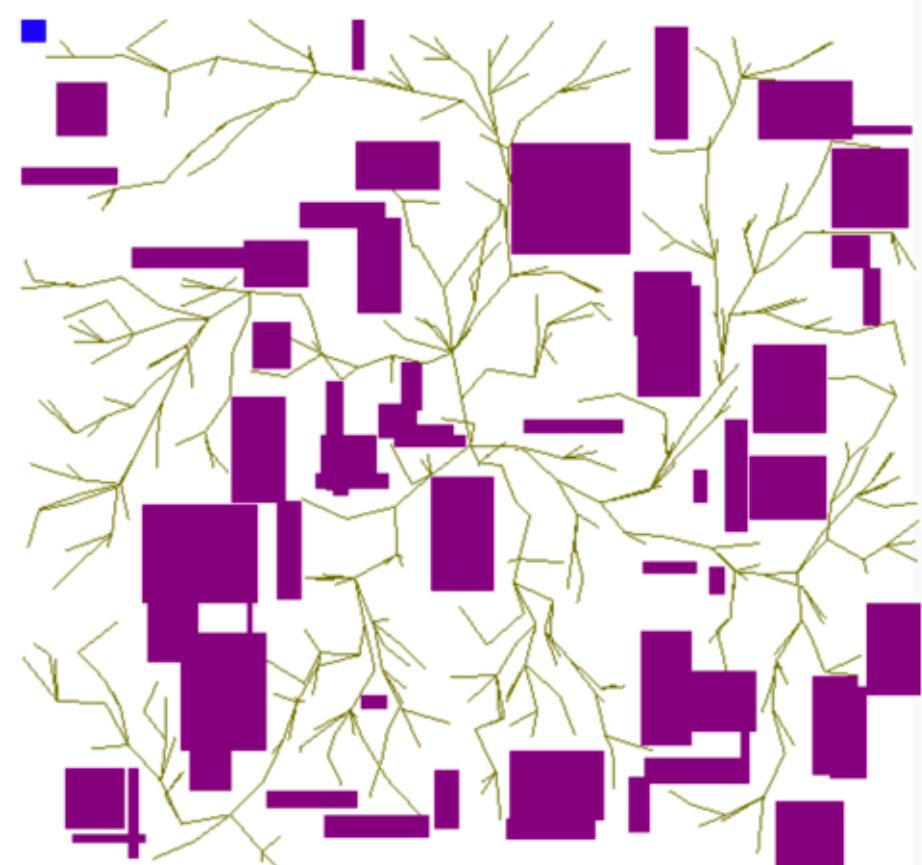
Comparaison RRT vs RRT*

- Tiré de « S. Karaman, E. Frazzoli, Sampling-based Algorithms for Optimal Motion Planning. »



RRT

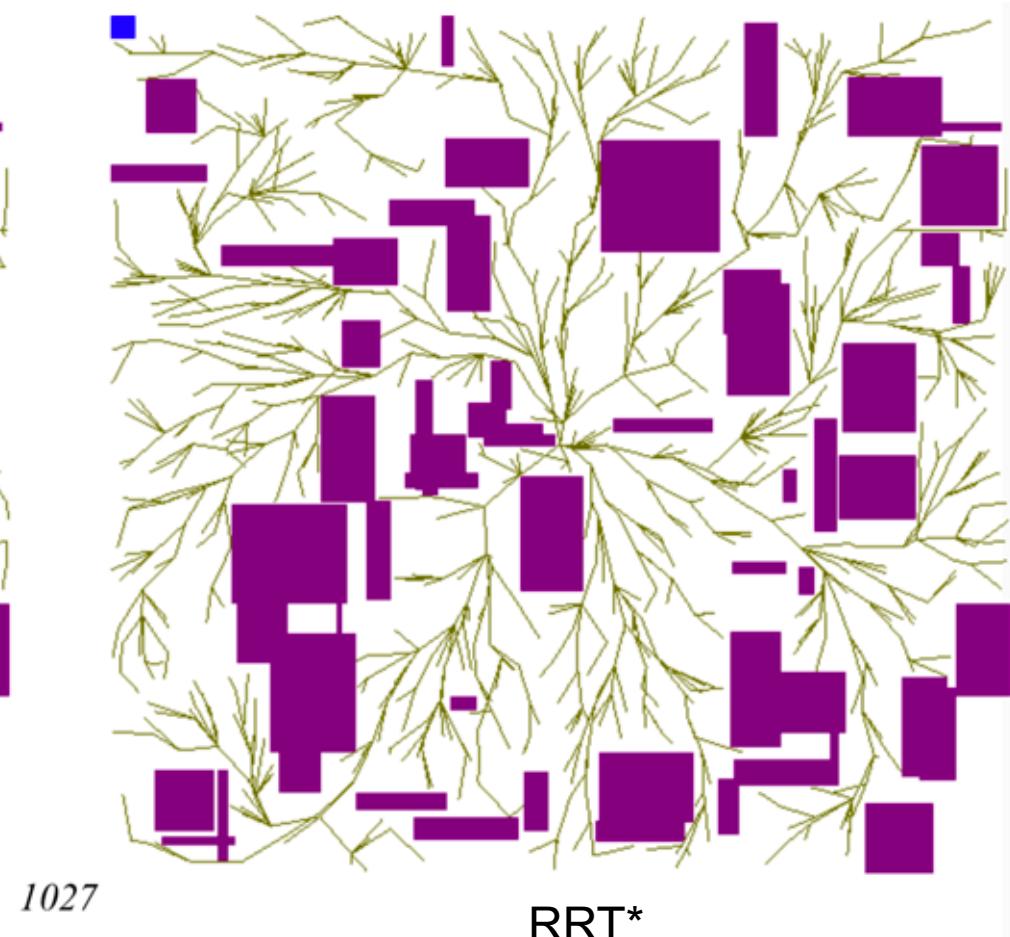
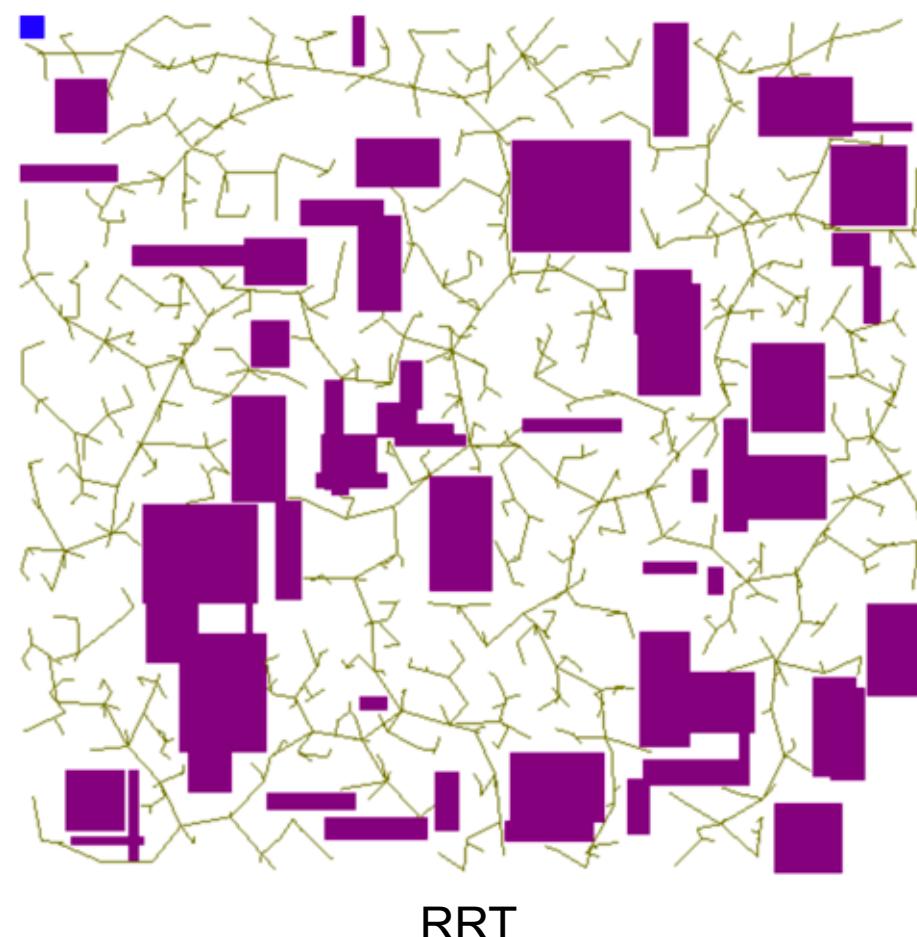
503 iterations



RRT*

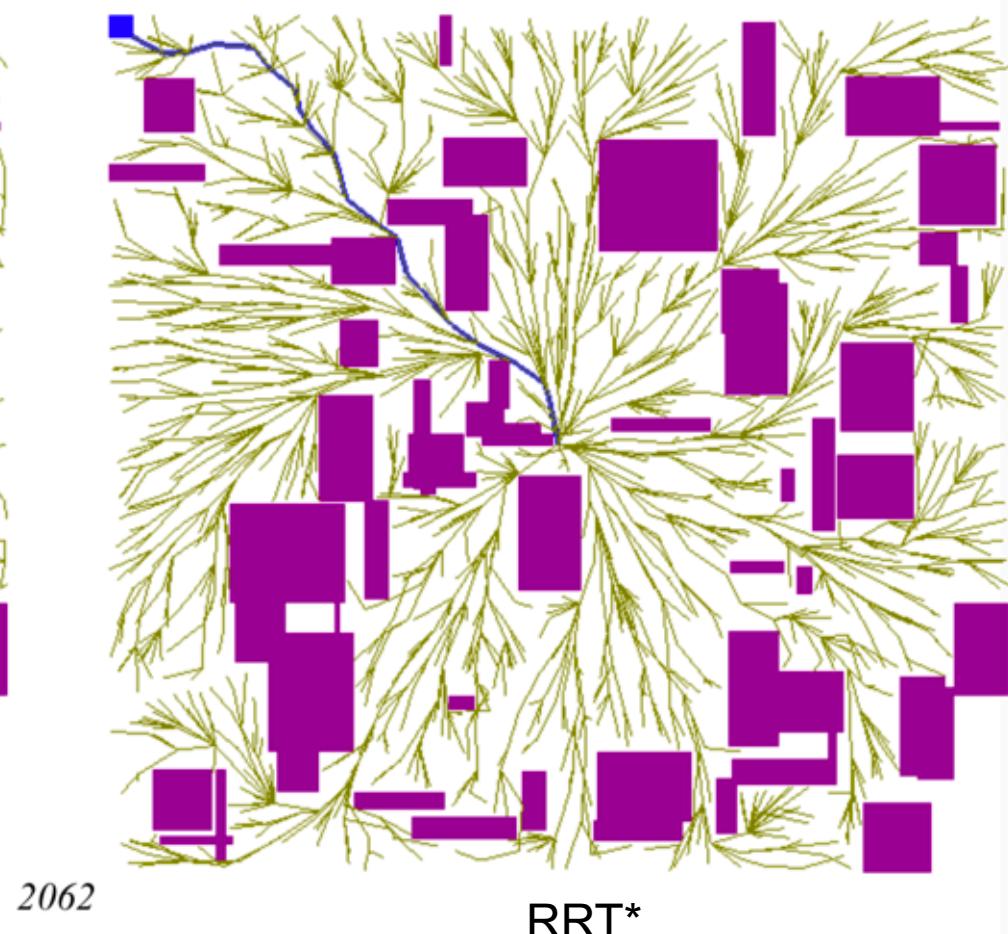
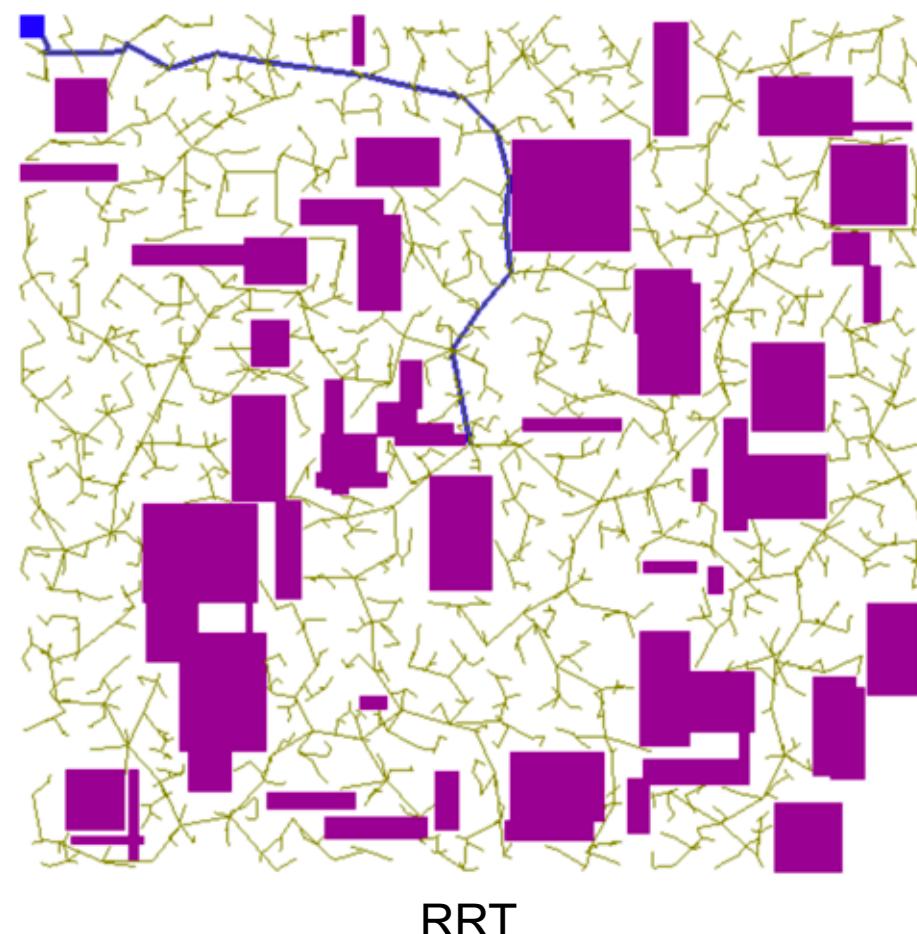
Comparaison RRT vs RRT*

- Tiré de « S. Karaman, E. Frazzoli, Sampling-based Algorithms for Optimal Motion Planning. »



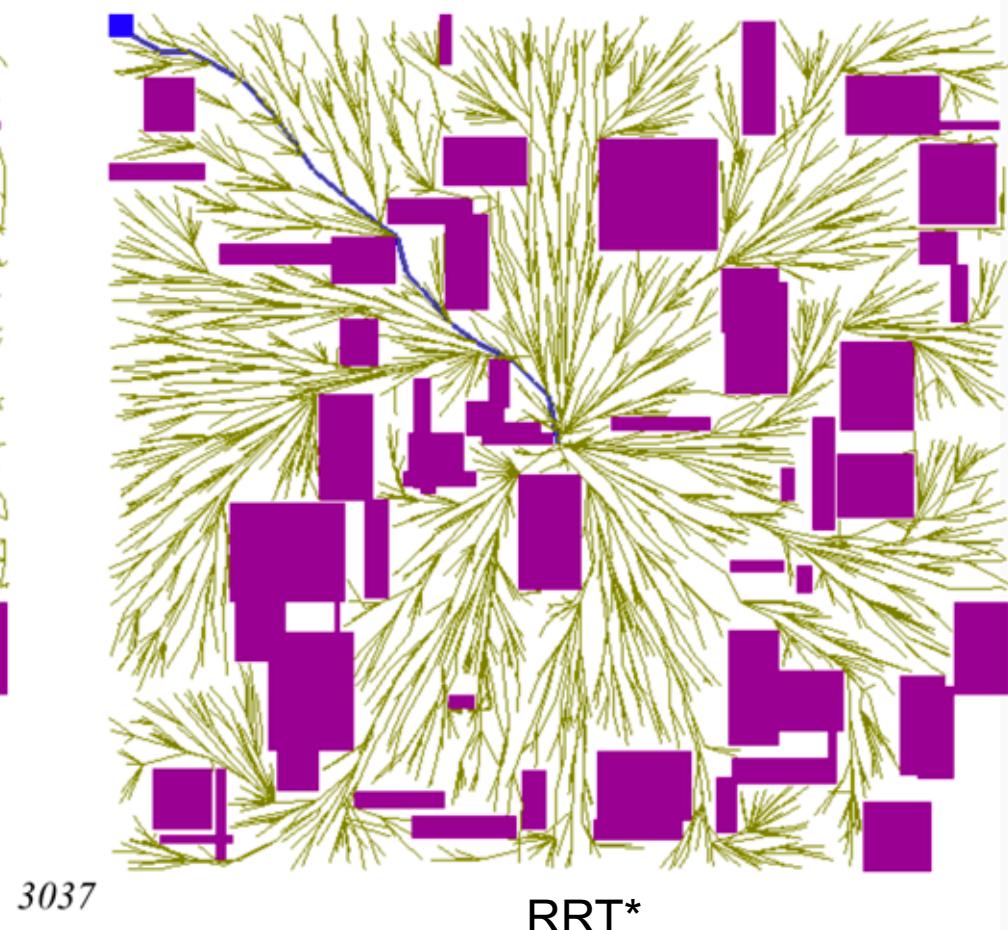
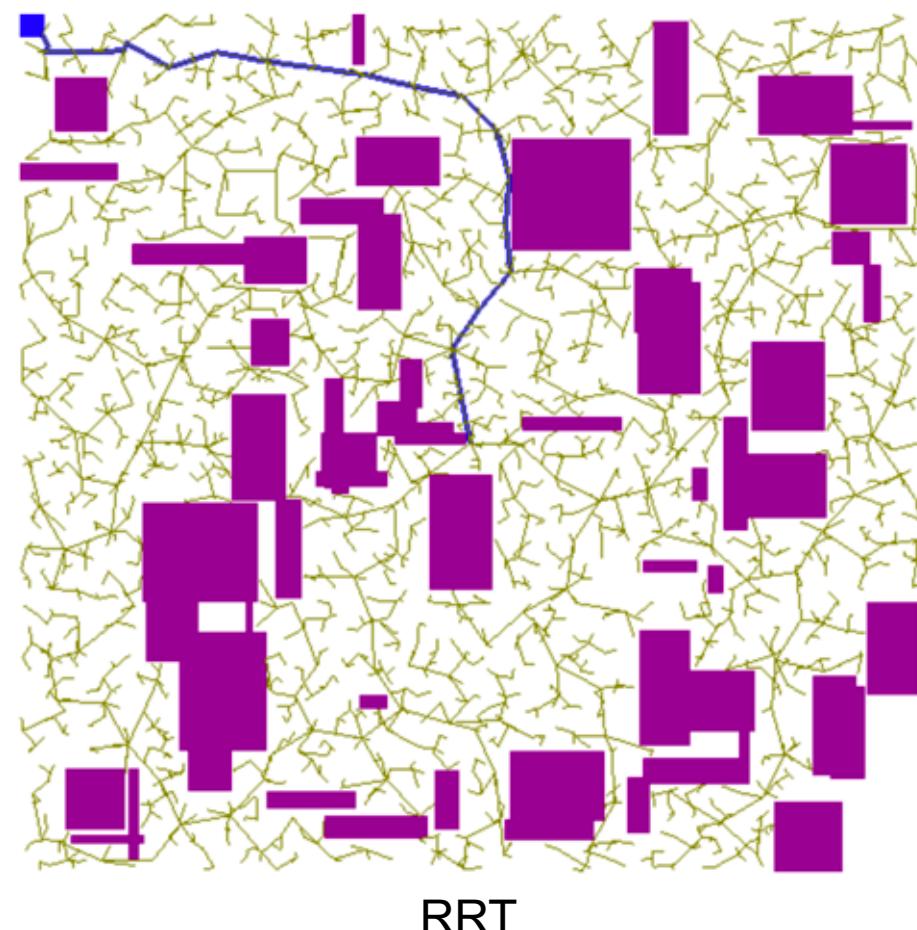
Comparaison RRT vs RRT*

- Tiré de « S. Karaman, E. Frazzoli, Sampling-based Algorithms for Optimal Motion Planning. »



Comparaison RRT vs RRT*

- Tiré de « S. Karaman, E. Frazzoli, Sampling-based Algorithms for Optimal Motion Planning. »



Comparaison RRT vs RRT*

- Effet de la sous-optimalité en dimension 12 !



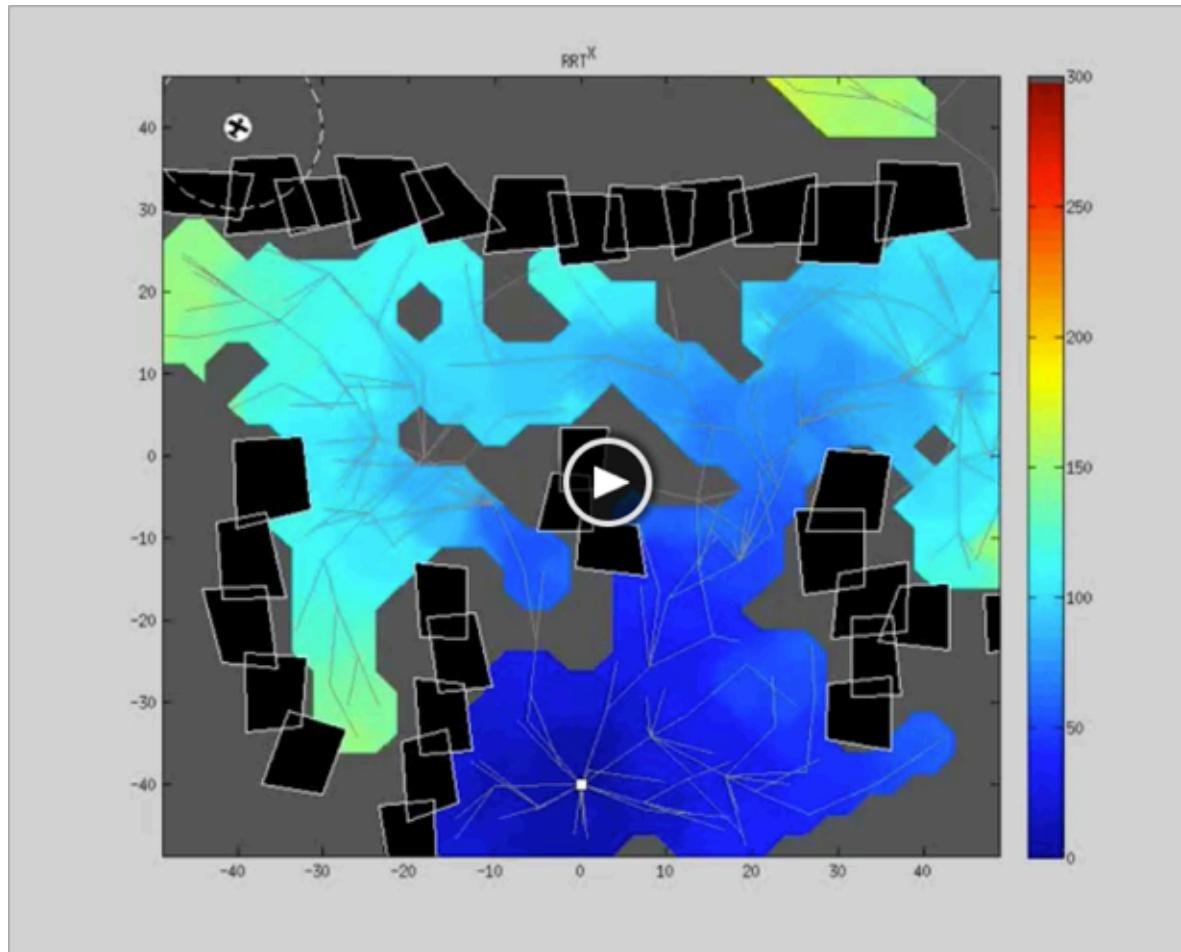
RRT



RRT*

RRT^X

- Amélioration de RRT* pour les env. dynamiques/inconnus
- Reconnexion/amélioration permanente en fct de l'env. (cf D*)

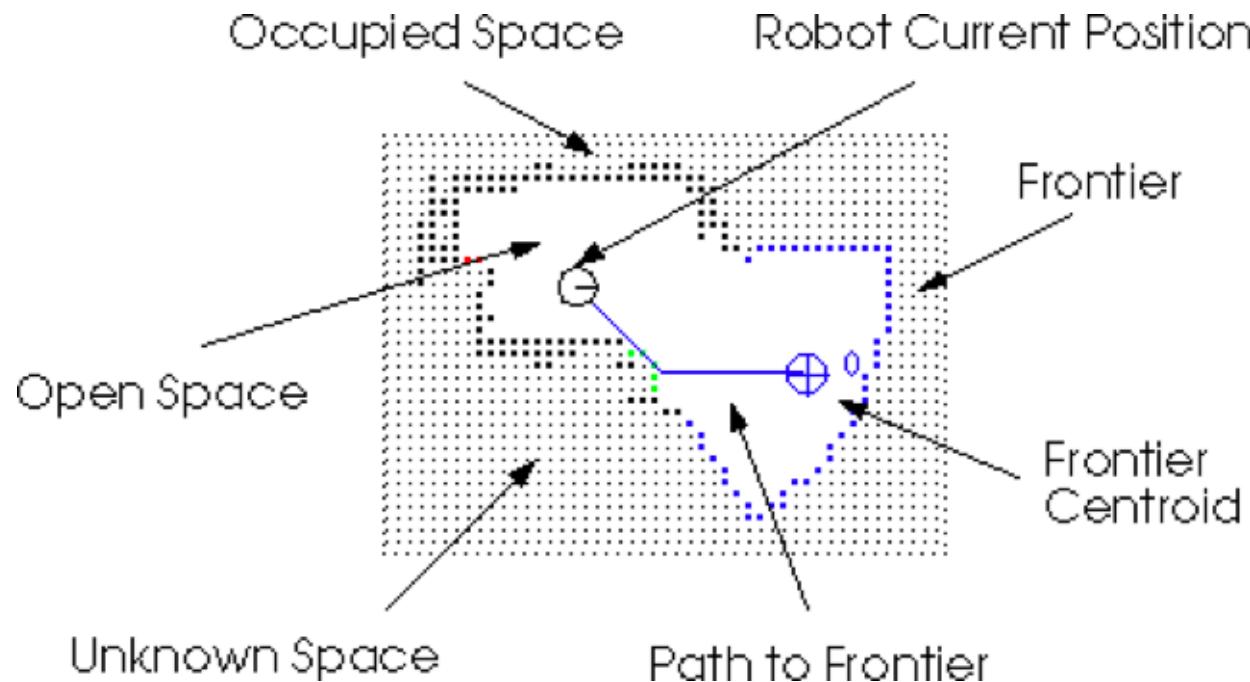


Exploration



Exploration d'un environnement inconnu

- B. Yamauchi, « A Frontier-Based Approach for Autonomous Exploration », CIRA 97
- Planification à partir de grilles d'occupations
- Frontière : cellules « vides » touchant des cellules « inconnues »



Planification pour exploration

- Planification vers centre de la frontière la plus proche avec A*
- Extensions multi-robots possibles (attribution des frontières)

Exploring with explore_lite

Jiri Horner

Cours « Robotic Motion planning »

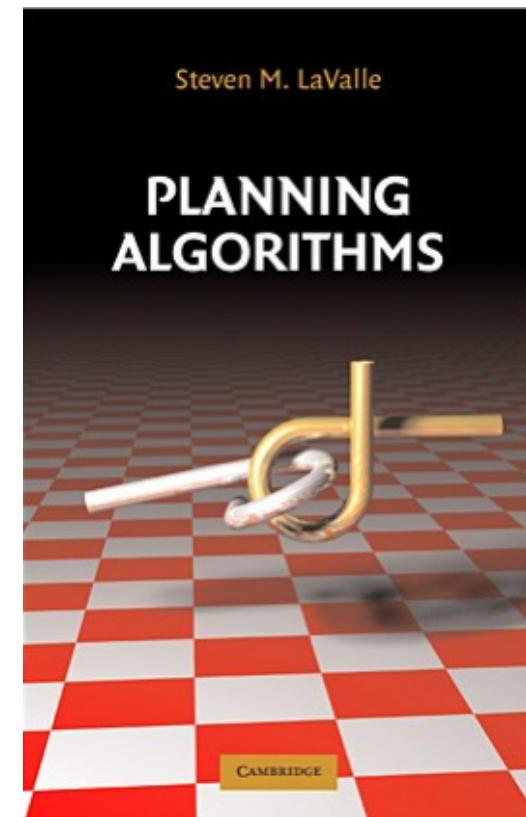
- Howie Choset : <http://www.cs.cmu.edu/~choset>

Livre « Planning Algorithms »

- Steven Lavalle : <http://planning.cs.uiuc.edu>

Slides

- Introduction to Mobile Robotics , Robot Motion Planning - Wolfram Burgard, Cyrill Stachniss, Maren Bennewitz, Kai Arras
<http://ais.informatik.uni-freiburg.de/teaching/ss11/robotics/slides/18- robot-motion-planning.pdf>
- Optimal Rapidly-exploring Random Trees, Michel Vargas



Fin du cours

