

# Conception et optimisation d'algorithmes de coordination multi-robots

Exploration autonome de réseaux de galeries

Fabien MATHÉ

SeaTech - MOCA 3A  
Soutenance - Projet de fin d'études

27 février 2025

# Introduction

- Fascination pour l'inconnu
- Meilleure compréhension de notre planète
- Défis inhérents de l'exploration souterraine
  - Accessibilité
  - Communication
  - Navigation
- Intérêt scientifique
  - Découverte de nouvelles formes de vie
  - Structures géologiques préservées



Figure – Grotte de la flûte de pan (Guilin, Chine). Paul Munhoven ©

# Introduction



Figure – Lac de la Grotte de Lombrives - Ariège



Figure – Elizabeth, le robot serpent.  
Nico Zevalios and Chaohui Gong

## Objectifs :

- Développer des algorithmes de navigation pour des robots autonomes en milieu souterrain.
- Assurer une communication autonome entre des robots sans contrôle externe.

# Sommaire

1 Introduction

2 Planification de trajectoire

3 Communication

4 Simulateur et résultats

5 Conclusion et perspectives

# Tentative d'optimisation théorique

## Objectif

Trouver le plus court chemin permettant d'explorer une carte inconnue

- Contrainte :
  - Évitement des obstacles
- Simplifications :
  - Aucune contrainte liée au robot
  - Espace 2D

## Définition de l'espace étoilé

On définit :

$$FV(t) = \{ (1 - l) \mathbf{X}(t) + l \mathbf{M}(t, \alpha) \mid l \in [0, 1], \alpha \in [0, 2\pi[ \}$$

$$\mathbf{M}(t, \alpha) = \mathbf{X}(t) + R(t, \alpha) \begin{pmatrix} \cos \alpha & 0 \\ 0 & \sin \alpha \end{pmatrix} \mathbf{X}(t)$$

Avec  $R(t, \alpha)$  la distance minimale entre le robot et le plus proche point d'intersection à un mur, borné par  $R_{max}$ .

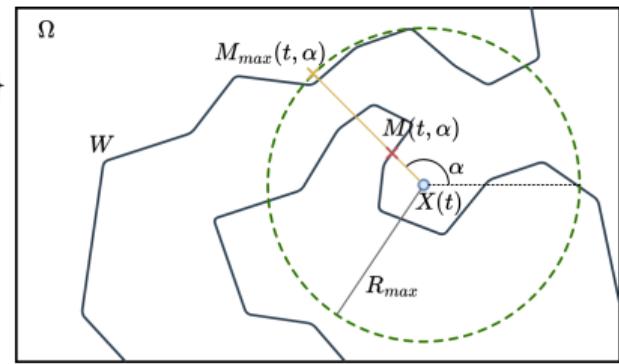


Figure – Schéma des notations

- $KM = \bigcup_{t=0}^T FV(t)$  : La partie de la carte connue par le robot
- $EM$  : La partie de la carte explorable par le robot

# Définition de la fonctionnelle et problèmes rencontrés

Fonctionnelle :

$$J(\mathbf{X}(0), T) = \int_0^T \|\dot{\mathbf{X}}(t)\| dt$$

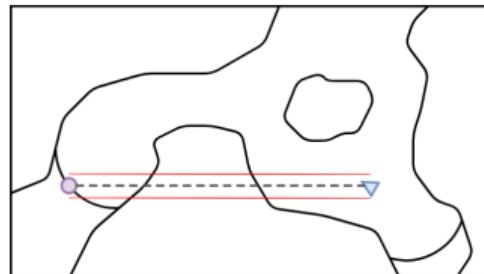
avec pour contrainte  $\mathbf{X}(0) = \mathbf{X}_{Robot}$ ,  $\mathbf{X}(T) = \mathbf{X}_{WP}$ ,  $KM(T) = EM$  et  $\dot{\mathbf{X}}$  bornée.

- Comment garantir  $\mathbf{X}(T) = \mathbf{X}_{WP}$  en contraignant  $\dot{\mathbf{X}}(t)$  ?
- Comment choisir le temps  $T$ ?
  - $T$  doit être suffisamment grand pour atteindre la cible.
  - Si le robot s'arrête en  $X_{WP}$  à  $t = t_1$ , alors :

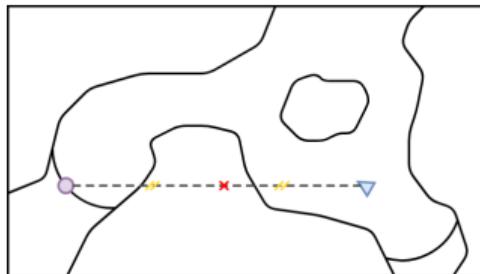
$$\int_{t_1}^T \|\dot{\mathbf{X}}(t)\| dt = 0$$

- Comment prendre en compte l'exploration de la carte ?

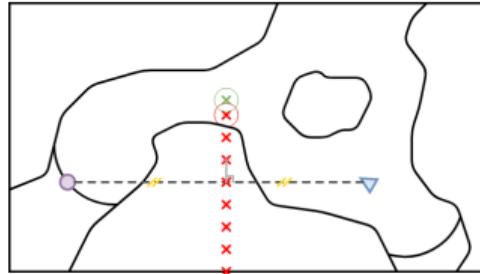
## Proposition d'une nouvelle méthode



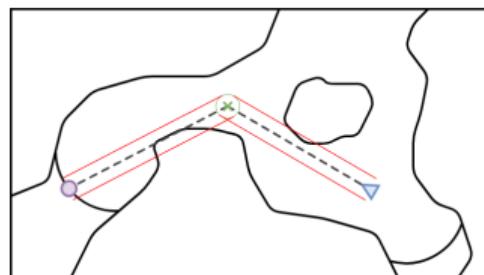
(a) Tracer une ligne



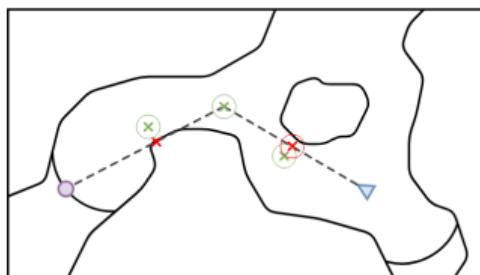
(b) Vérifier si le chemin est libre



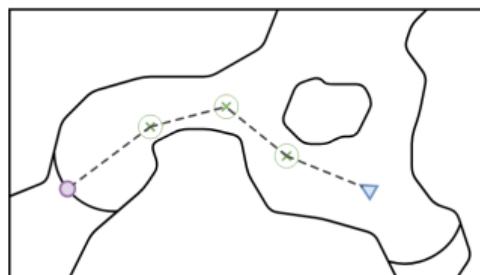
(c) Tirer des points normaux



(d) Valider le point



(e) Répéter la méthode



(f) Connecter les points

## Proposition d'une nouvelle méthode

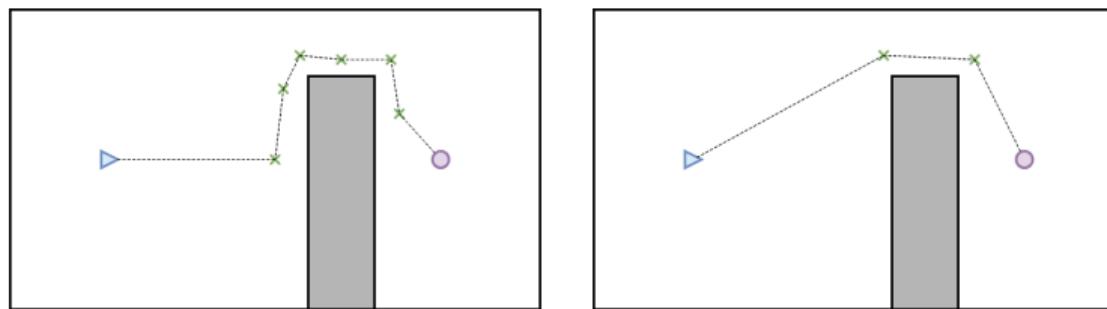


Figure – Processus de simplification du chemin trouvé

# Méthode de vérification

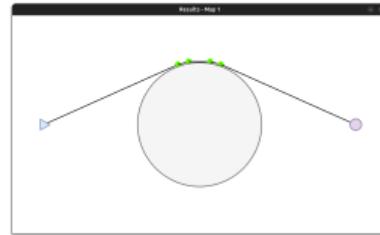
Le chemin le plus court est modélisé par la propagation d'une onde à vitesse constante.[1]

- Utilisation d'un automate cellulaire (CA)
- La structure en réseau définit :
  - Les cellules activées
  - Les obstacles
  - Les sources secondaires d'ondes
  - Les espaces vides
- Un vecteur de distance suit la progression de l'onde.
- Sources secondaires d'ondes ajoutées pour gérer les obstacles, inspirées du principe de Huygens.

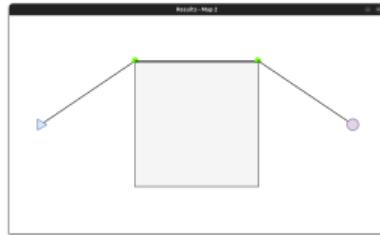


Figure – Propagation d'une onde à vitesse constante

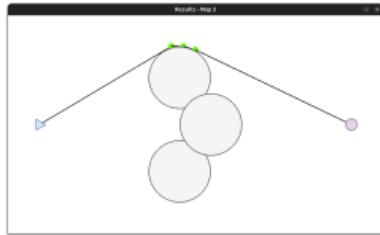
# Résultats et performance



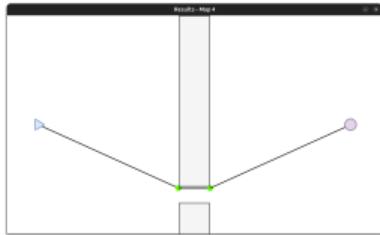
Résultats carte n°1



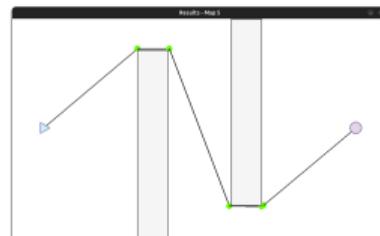
Résultat carte n°2



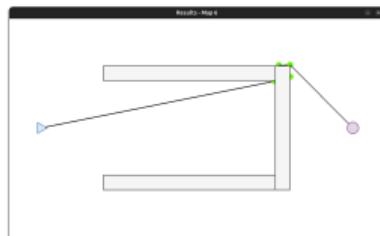
Résultat carte n°3



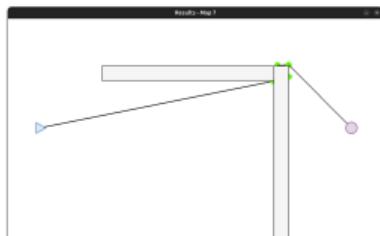
Résultat carte n°4



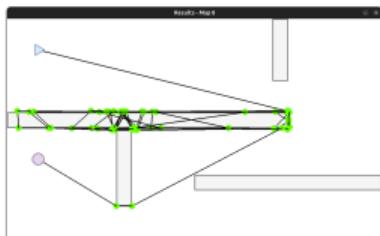
Résultat carte n°5



Résultat carte n°6



Résultat carte n°7



Résultat carte n°8

Figure – Résultats sur les cartes de test

## Résultats et performance

Cartes	Théoriquement	Méthode	Rel. diff. (%)	CA ( $\pm 2$ mu)	Rel. Error (%)
Carte 1	1081	1084	0.3	1086	0.5
Carte 2	1121	1125	0.4	1122	0.1
Carte 3	1122	1125	0.4	1122	0
Carte 4	1084	1088	0.4	1086	0.2
Carte 5	1521	1531	0.7	1522	0.1
Carte 6	1166	-	-	1168	0.2
Carte 7	1166	-	-	1168	0.2
Carte 8	1776	-	-	1780	0.2

Table – Comparaisons des résultats

# Algorithme de Dijkstra

- Détection d'impasse :
  - Le robot reste dans un certain périmètre autour de sa position pendant un certain temps.
- En cas d'impasse :
  - Redéfinition du point étape :
    - Plus proche point appartenant aux cellules traversées de l'étape.
  - Algorithme de Dijkstra :
    - Existence du chemin garanti.
    - Simplification du chemin.

# Analyses

- Portée délimitée d'application
  - Adaptée aux espaces ouverts et cavités moyennes.
  - En cavités étroites, exécutions fréquentes de Dijkstra.
  - Implémentation possible en 2D et 3D.
- Robustesse face aux pannes et à l'incertitude
  - Gestion des blocages et sortie de situations difficiles.
  - Dijkstra assure un retour en arrière en cas de besoin.
- Complétude
  - Exploration de tout le domaine possible.
  - Nécessité de tests approfondis.
  - Complétude incertaine à ce stade.

# Communication

## Objectif

Coordonner l'exploration de zones inconnues par un essaim de robots, sans entité centrale pour gérer la coordination.

- Contraintes
  - Environnement confiné
  - Liaisons souvent coupées
- Simplifications
  - Communication possible si contact visuel
  - Pas de modélisation d'atténuation
  - Pas d'erreur de communication

# Principe

- Communication robot-à-robot :
  - Communication par lumière.
- Sélection dynamique du robot maître :
  - Le robot avec le plus petit ID devient le maître.
- Communication hiérarchique forte, le robot maître :
  - donne des instructions aux robots avec un ID plus élevé.
  - partage toutes les informations acquises par tout le groupe.



Figure – Exemple de capteur de couleur,  
ref AS7341 - 10 canaux

# Principe

- Choix des directions de groupe :
  - Synchronisation de la carte connue.
  - Envoie des information du robot esclave vers le maître.
  - Calcul de la table de coût.
  - Envoie des directions par le maître en minimisant la combinaison des coût.

	R. 1	R. 2	R. 3	R. 4	R. 5
WP 1	23	87	234	56	192
WP 2	245	76	11	68	39
WP 3	90	21	73	50	164
WP 4	132	58	49	77	25
WP 5	181	13	66	39	70

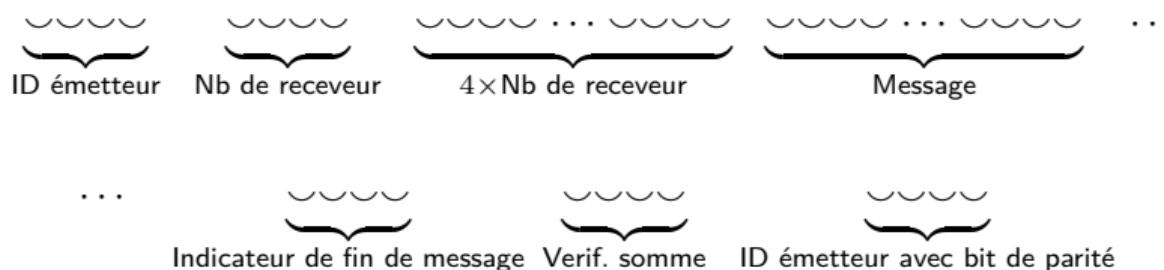
Table – Exemple de table de coût

	R. 1	R. 2	R. 3	R. 4	R. 5
WP 1	61	125	93	47	59
WP 2	88	12	53	29	174

Table – Autre exemple de table de coût

# Encodage

Structure d'un message type :



Resistance aux erreurs par méthode ARQ

# Analyses

- Portée délimitée d'application
  - Exploration robotique en essaim, scalabilité
  - Intérieur ou extérieur
  - Communication optique, propagation lumineuse
- Robustesse face aux pannes et à l'incertitude
  - Détection d'erreurs (ARQ, vérification de somme, bit de parité)
  - Réaffectation dynamique du rôle de maître
- Vitesse
  - Encodage compact, bande passante réduite
  - Cycle requête-réponse, réduire retransmissions
  - Communication parallèle, traitement simultané

# Simulateur

## Objectif

Implementer et tester les algorithmes proposés pour les améliorer avant un déploiement sur robot

- Contraintes
  - Simulateur en temps réel
  - Contrainte cinématique du robot prise en considération
- Simplifications
  - Les robots opèrent dans les airs et se déplacent sur le sol
  - Le sol est approximé à un plan 2D
  - Aucune irrégularité de surface n'est prise en compte

# Génération de la carte

Génération des obstacles par automate cellulaire sur grille cartésienne

Règles :

Si 4 cellules sont actives dans le voisinage de Moore alors la cellule s'active sinon elle se désactive

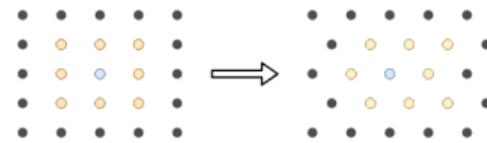


Figure – Transformation de la grille

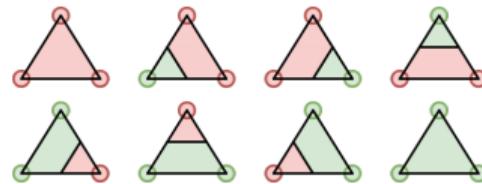


Figure – Méthode de Marching Square appliquée aux triangles

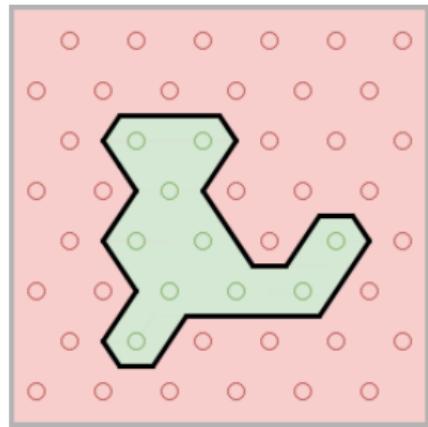
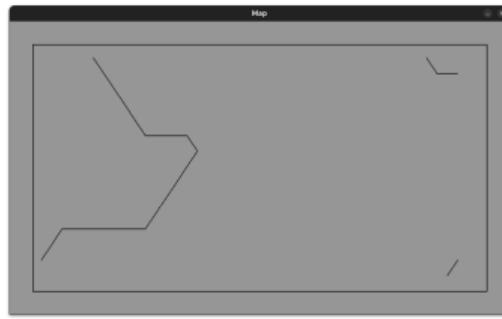
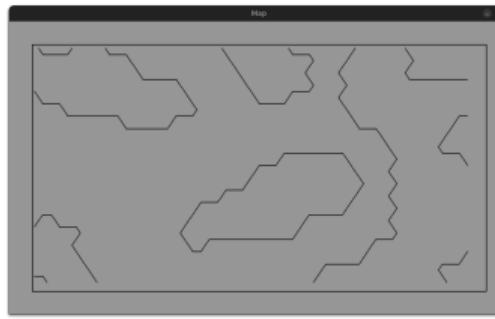


Figure – Exemple de carte générée

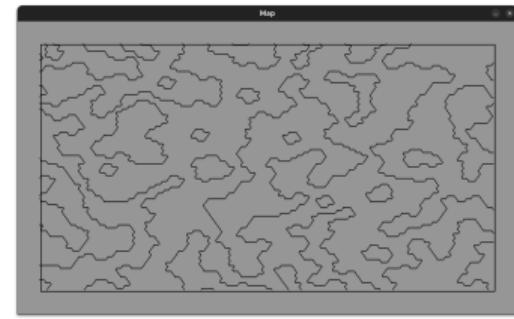
## Exemple de cartes générées



(a)  $\delta x = 100 \text{ mu}$



(b)  $\delta x = 40 \text{ mu}$



(c)  $\delta x = 10 \text{ mu}$

Figure – Exemple de cartes générées

# Implémentation des capteurs

- Centrale inertielle
- LiDAR
  - Émission
  - Réflexion
  - Réception
  - Calcul du temps de vol → distance.
- Optimisations
  - Segmentation de l'espace
  - Algorithme de Bresenham modifié
- Simulation des erreurs par loi normale

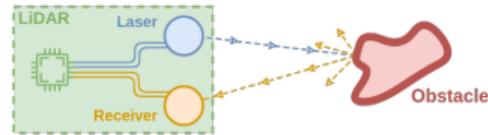


Figure – Fonctionnement d'un LiDAR

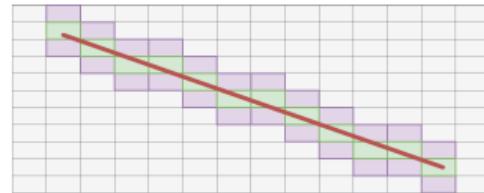


Figure – Algorithme de Bresenham modifié

# Localisation et cartographie simultanées (SLAM)

- Création d'une carte de caractéristiques avec 4 états :
  - Inconnue (Gris)
  - Obstacle (Violet)
  - Libre (Vert clair)
  - Traversé (Vert)
- Calcul et correction de la position
  - Centrale inertielle
  - Triangulation par LiDAR

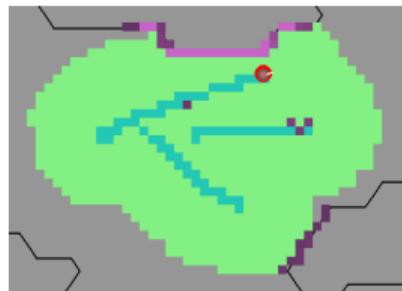


Figure – Carte des caractéristiques

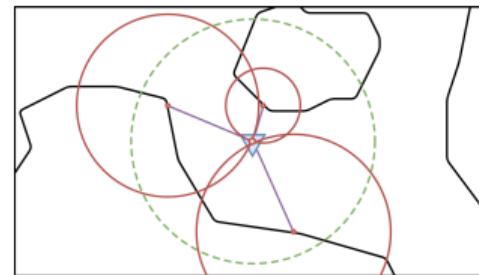


Figure – Correction de la position par le LiDAR

# Carte de caractéristiques dynamique

Environnement hautement dynamique

- Carte jumelle : carte d'occurrence
- Compte le nombre de rayons de LiDAR ayant touché les cellules de la carte de caractéristiques
  - Si collision avec un mur : Ajouter 3
  - Sinon : Soustraire 1

Assure une carte des caractéristiques sécurisante tout en supprimant de celle-ci les objets dynamiques

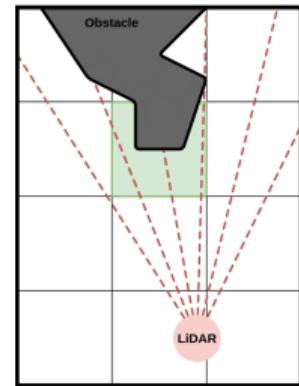


Figure – Carte dynamique des caractéristiques

Amélioration possible : Filtre de Kalman

# Résultats

Carte n°	Nb Robots	Temps exploration	Réduction
30	1	2'20"	-
30	2	1'3"	55%
40	1	2'35"	-
40	2	1'32"	41%
55	1	2'7"	-
55	2	54"	57%
100	1	3'16"	-
100	3	1'28"	55%
100	5	1'12"	63%

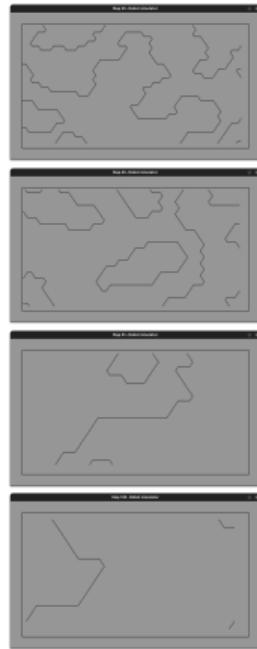


Table – Temps d'exploration pour différentes tailles de carte et nombre de robots

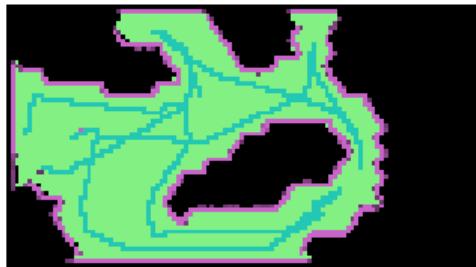
# Résultats



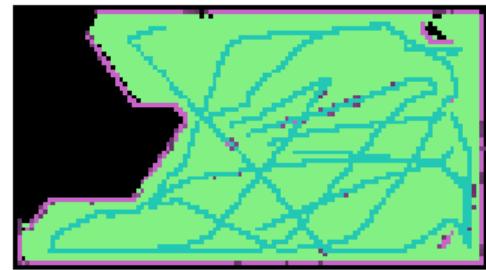
Carte n°55 - 2 robots



Carte n°40 - 1 robot



Carte n°40 - 2 robots



Carte n°100 - 5 robots

# Conclusion

- Développement d'un simulateur :
  - Environ 3500 lignes de code
  - Une dizaine de fonctions utiles pour de futures implémentations
- Validation des algorithmes d'exploration dans des environnements complexes
- Mise en place d'une communication simple mais robuste entre les robots
- Première étape vers des recherches approfondies en thèse

# Perspectives

- Amélioration de la performance du simulateur :
  - Passage en C++
  - Parallélisation
- Implémentation ROS
- Implémentation du terrain

# Perspectives

## Simplification du problème d'optimisation

- Minimiser la consommation d'énergie lors du déplacement de A à B.
- Consommation d'énergie :
 
$$E(T) = \int_0^T \|\omega(t)\| dt$$
- Où :  $\|\omega(t)\| = \sqrt{\omega_L^2(t) + \omega_R^2(t)}$
- Contraintes :  $X(0) = X_R$  et  $X(T) = X_{WP}$ .
- Problématique : Assurer  $X(T) = X_{WP}$ .

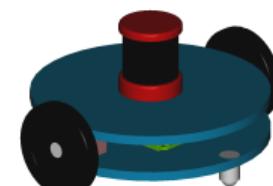
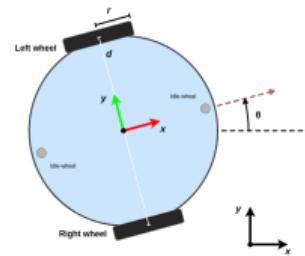
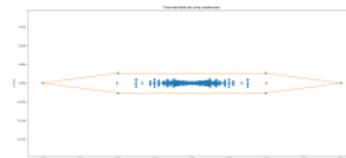
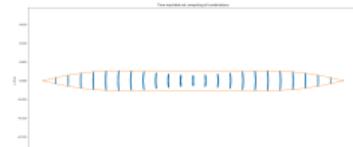


Figure – Modèle de robot utilisé



(a) Chemin  
stochastique



(b) Chemin empirique

# Références



Calvo Tapia, Carlos, Villacorta-Atienza, José, Mironov, Vasily, Gallego, Victor, and Makarov, Valeri. (2016). WAVES in ISOTROPIC TOTALISTIC CELLULAR AUTOMATA : APPLICATION to REAL-TIME ROBOT NAVIGATION. *Advances in Complex Systems*, 19, 1650012. DOI 10.1142/S0219525916500120.

## Table des types

Indicateur de type	Type	Taille des données
0000	Fin de message	4 bits
0001	Entier	8 bits
0010	Entier	32 bits
0011	Flottant	32 bits
0100	Chaîne de caractères	8 bits pour la longueur + 8 bits par caractère
0101	Liste d'entiers 8 bits	16 bits pour la longueur + 8 bits par entier
0110	Liste de flottants	16 bits pour la longueur + 32 bits par flottant
0111	ID de type de message	8 bits
1000	Info robot	168 bits
1010	Message binaire	16 bits pour la longueur + 1 bit par bit binaire

Table – Tableau des indicateurs de type

## Table des identifiants de message

Message ID	Description
00000000	Demander de répéter le dernier message
00000001	Dernier message reçu correctement
00010000	Demander l'ensemble des robots connectés
00010001	Envoyer l'ensemble des robots connectés
00100000	Demander à tous où ils veulent aller
00100001	Envoyer la position du prochain point de passage avec l'ID de la frontière
00100010	Envoyer les informations du robot
00100011	Envoyer la prochaine position à atteindre
00110000	Demande la synchronisation de la carte en direct
00110001	Envoyer la carte en direct mise à jour
01010101	Transmettre un message d'un autre robot

Table – Table des identifiants de message